A Knowledge-Based Approach to Class Scheduling

Mara Zell
College of Saint Benedict/Saint John's University

Follow this and additional works at: https://digitalcommons.csbsju.edu/honors_theses

Part of the Computer Sciences Commons

Recommended Citation

Available by permission of the author. Reproduction or retransmission of this material in any form is prohibited without expressed written permission of the author.
A Knowledge-Based Approach to Class Scheduling

A THESIS
The Honors Program
College of St. Benedict/St. John's University

In Partial Fulfillment
of the Requirements for the Distinction "All College Honors"
and the Degree Bachelor of Arts
In the Department of Computer Science

Advisor
Dr. Dan Challou

by
Mara Susan Zell
May, 1995
PROJECT TITLE: A Knowledge-Based Approach to Class Scheduling

Approved by:

Dr. Dan Challou
Assistant Professor of Computer Science

Dr. J. Andrew Holey
Assistant Professor of Computer Science

Dennis Myers
Lab/Equipment Manager, Physics

Dr. Lynn Ziegler
Chair, Department of Computer Science

Margaret Cook
Director, Honors Thesis Program

Anthony Cunningham
Director, Honors Program
# TABLE OF CONTENTS

<table>
<thead>
<tr>
<th>Section</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>List of Figures</td>
<td>6</td>
</tr>
<tr>
<td>Abstract</td>
<td>7</td>
</tr>
<tr>
<td>Introduction</td>
<td>8</td>
</tr>
<tr>
<td>Background</td>
<td>9</td>
</tr>
<tr>
<td>System Development</td>
<td>13</td>
</tr>
<tr>
<td>System Overview</td>
<td>15</td>
</tr>
<tr>
<td>Experimental Results</td>
<td>26</td>
</tr>
<tr>
<td>Discussion of Results</td>
<td>31</td>
</tr>
<tr>
<td>Summary and Conclusion</td>
<td>32</td>
</tr>
<tr>
<td>Acknowledgments</td>
<td>33</td>
</tr>
<tr>
<td>Bibliography</td>
<td>34</td>
</tr>
<tr>
<td>Appendix A</td>
<td>36</td>
</tr>
<tr>
<td>Appendix B</td>
<td>44</td>
</tr>
<tr>
<td>Figure</td>
<td>Description</td>
</tr>
<tr>
<td>--------</td>
<td>--------------------------------------------------</td>
</tr>
<tr>
<td>1</td>
<td>Class Information Window</td>
</tr>
<tr>
<td>2</td>
<td>Scheduling Window</td>
</tr>
<tr>
<td>3</td>
<td>Add Term/Department Window</td>
</tr>
<tr>
<td>4</td>
<td>Class Maintenance Window</td>
</tr>
<tr>
<td>5</td>
<td>Faculty Member Information Window</td>
</tr>
<tr>
<td>6</td>
<td>Add Department Window</td>
</tr>
<tr>
<td>7</td>
<td>Faculty Member Maintenance Window</td>
</tr>
</tbody>
</table>
ABSTRACT

A class scheduling application was developed to assist department chairs in producing class schedules each semester. This was accomplished using a knowledge-based system. The system utilized the many constraints involved in the class scheduling process to solve the problem. This application was developed and implemented in an object oriented package called Powerbuilder. Thus, the application is windows based with point and click features. Three trial schedules were produced. These results demonstrate the ability of the application to schedule three types of classes: classes without labs, classes with one lab, and classes with two labs. The end result is that an automated scheduling program is capable of solving the general class scheduling problem at CSB/SJU.
Introduction:

Class scheduling is the process of assigning faculty members to teach the courses offered by a department. The Department Chairperson is usually responsible for scheduling the courses each semester to the satisfaction of the faculty and the needs of the students. To accomplish this task, he or she must consider many factors. For example, the Chairperson must attempt to accommodate the course requests made by the faculty members while following the guidelines of the department and the university. Also, he or she must insure that scheduled classes have the proper facilities. Overall, many constraints must be considered to produce a good schedule.

As department size and number of classes offered increase, scheduling becomes more difficult and time consuming. Currently at CSB/SJU, class schedules are manually produced. No matter how often this process is completed, each semester provides new data and new possibilities that require a non trivial amount of time and effort. Therefore, automating the class scheduling process should significantly reduce the amount of time required to produce departmental schedules.

The computational difficulty arises with the fact that scheduling is an NP-complete problem (p. 82, Garey and Johnson). In an NP-complete problem, the amount of time required to solve the problem grows exponentially with problem size. Thus for scheduling, problem difficulty grows exponentially with the number of items that must be scheduled. Moreover, an NP-complete problem is unlikely to have a general polynomial time algorithmic solution.

For class scheduling, knowledge about the rules and constraints involved and the class scheduling process itself must be obtained. The general and specific scheduling needs of each department are obtained from the interviews with people from each department. For example, class scheduling is usually supervised by the Department Chair,
thus he or she is one of the primary sources for the majority of the required problem solving knowledge. Other general sources of scheduling knowledge include journal articles and books.

**Background:**

Knowledge-based systems have been used to solve various scheduling problems with some success. Knowledge-based systems arose from research in the area of expert systems. The basis of expert system design originated in the early 1940's with Herbert Simon and Allen Newell. Along with J.C. Shaw, Simon and Newell began development on one of the first rule-based systems (a type of expert system), the General Problem Solver, in 1957. They saw the potential the expert system technology could possess:

there are now in the world machines that think, that learn and that create. Moreover, their ability to do these things is going to increase rapidly until - in a visible future - the range of problems they can handle will be coextensive with the range to which the human mind had been applied. (p. 16, Zahedi)

It was not until the late 1960's that others witnessed expert systems as the emerging practical side of Artificial Intelligence. In less than three decades, expert and knowledge-based systems entered the computerized decision mainstream and have become the most successful applications of Artificial Intelligence. Today many problems are solved using expert system technology. The areas to which expert systems can be applied include: interpretation, prediction, diagnosis, planning, scheduling, monitoring, debugging, repair, instruction, and control.

A computer system that emulates the decision-making ability of a human expert is considered an expert system. Expert systems have been most successful in the areas that require categorization and pattern recognition. These systems are designed using symbolic manipulation and are qualitative by nature. Expert systems function with knowledge that has been gained by experience. This knowledge is modeled
probabilistically, so there is no deterministic solution, knowledge is acquired and becomes better over time. Since there is no algorithmic solution, expert systems rely on inferences for achieving a reasonable solution; thus expert systems choose between options by weighing the evidence and then determine further processing. Professor Edward Feigenbaum of Stanford University defined expert systems as "an intelligent computer program that uses knowledge and inference procedures to solve problems that are difficult enough to require significant human expertise for their solution" (p. 1, Giarratano and Riley).

Knowledge-based systems are a subset of expert systems. The goal of both expert and knowledge-based systems is to function at the level of the expert. Knowledge-based systems function primarily with knowledge that is deterministic. It uses constraints to choose the appropriate elements in the solution set, and thus are more suitable for solving scheduling problems. Unlike an expert system, a knowledge-based system does have an algorithmic solution so it does not rely on inferences to achieve a solution. Instead, the knowledge necessary to solve the given problem is incorporated into the code. This code works with a given data set and produces a solution. A main advantage of a knowledge-based system is that the human "expert" no longer needs to spend the time solving problems that a knowledge-based system can solve.

Currently, the knowledge within a knowledge-based system can be either expert and/or general knowledge. The knowledge in the system is limited to a specific domain and designed to deal only with the encoded knowledge of the domain. Therefore, unlike human experts, it cannot make analogies to similar knowledge. Moreover, expertise from one domain does not necessarily carry over to another. For example, a chess expert is not necessarily an expert in medicine.
There are four components to a knowledge-based system: domain knowledge, the knowledge-base, the human component, and the system software. Domain knowledge is the area of knowledge that the system uses to make recommendations or solve problems. The knowledge-base contains all the facts or knowledge. The human component is comprised of three parts: the knowledge group, the developer group, and the user group. Individuals that create the knowledge-base are considered members of the knowledge group, and they are often called knowledge engineers. They acquire knowledge from the experts, determine what is expertise within the domain knowledge, and then formalize the structure of the knowledge. This process is often called knowledge acquisition. This task is not easily accomplished by one person, so the knowledge group is frequently broken down into the domain expert and the knowledge engineer. The role of the domain expert is to provide the expert knowledge that is to be emulated by the expert system. The knowledge engineer maps the knowledge into machine usable form, so that it can be used by the knowledge-based system. In order to create this knowledge, the expert must explain how he or she would solve a given problem to the knowledge engineer. If this process does not occur, then the knowledge cannot be encoded; otherwise, system development can proceed. The developer group programs the knowledge-based system. They work closely with both the knowledge group and the user group. The user group is composed of the individuals that will use the system. In developing the expert system one person may fulfill more than one function above. For example, in this thesis the author served as both the knowledge engineer and developer.

Limitations of knowledge-based systems do exist. As mentioned previously, one of these limitations is the fact that the system does not have an understanding of the underlying causes and effects. Thus, it can be easier to program shallow, empirical, and heuristic knowledge than deep knowledge, basic structure, function, and behavior of the objects. Another limitation is the existence of the knowledge acquisition bottleneck. This
term is used since "the knowledge acquisition bottleneck constricts the building of a
topology-based system like an ordinary bottle neck constricts fluid flow into a bottle" (p.
8, Giarratano and Riley). In other words, it is difficult and time consuming to acquire
knowledge from experts. An overly large knowledge-base also reduces the efficiency of a
knowledge-based system. It can also increase the possibility of inconsistencies within the
knowledge-base. These inconsistencies may occur due to conflicting rules or procedures.

There has been much work done in the scheduling area. For example, papers on
the topic include: methods for trains through the Channel Tunnel (Fu and Wright), vehicle
scheduling (Atkinson), cricket season scheduling in Australia (Willis and Terrill),
scheduling service personnel (Collins and Sisley), fleet scheduling (Christodoulou,
Wallace, and Kuchenhoff), flight crew scheduling (Yau), cockpit simulator scheduling
(Bell), bus driver scheduling (Paias and Paixao), and nursing scheduling (Randhawa and
Sitompul). Many of these projects used constraints to assist in developing the final
schedule. In fact, without these constraints a final schedule would be difficult and in some
cases impossible to produce.

Of all the papers regarding class scheduling, the approach proposed by Guyette
was closest to the proposed application. His program consists of seven modules, which
are executed in the following sequence:

* Module 1 CRT display
* Module 2 Collect knowledge from input files
* Module 3 Modify facts
* Module 4 Assign courses to faculty
* Module 5 Modify facts
* Module 6 Exchange classes between faculty members
* Module 7 Display and store results

(p. 154, Guyette)

The program uses actual data files containing the information about classes and faculty
members. After the program loads the data, the program converts the information into
computer usable form. After the information is converted, the program assigns classes to
the faculty members by preference. Each class is assigned a point value of one for use
within the modules. The program aims for a point total of three for each time area of
morning, afternoon, and evening and does not allow more than two classes per time slot.
Also, the program then tries to assign classes from the faculty members with extra classes
to faculty members still needing classes based on preference. A final schedule produced is
then converted back into human readable form. One disadvantage of Guyette's program is
that it does not schedule classes with labs. Another disadvantage is that it can produce an
incomplete schedule as shown in his paper. Guyette's test consisted of assigning forty
classes to eleven faculty members. In total, his program assigned thirty-nine of the forty
classes.

**System Development:**

In general, the most difficult part of any system development project is determining
what the application is really supposed to accomplish. In order to remedy this problem a
thorough knowledge acquisition process was attempted. The plan was to interview the
department chairs who are responsible for creating the class schedules each semester in
order to obtain a general model of how the scheduling process works for CSB/SJU and
more specific knowledge for various departments. The plan was motivated by two
reasons. First was to see how difficult it would be to make the program usable by any
department. Second was to see how often scheduling information overlapped between
departments.

After talking to Dr. Ziegler, Dr. Valley, Dr. Challou, and reviewing journal articles
about scheduling, the following model of the current manual process of class scheduling
was developed. As questions and problems arose, further information was obtained about
scheduling from Dr. Challou. As a start, the following concept was gathered regarding
class scheduling at CSB/SJU. To begin, faculty members request specific courses to teach each semester. Previous schedules determine what courses the faculty members have taught in the past. This gives the Department Chair an idea of what the faculty member wishes to teach and what classes faculty members have taught previously. With these ideas in mind, the Department Chair determines classes for each faculty member.

After completing the interview process, it was decided that it might be possible to make the program usable by all departments, but not to attempt to produce such a system unless time permitted. The remaining information was used to determine the rules and process the program would need to follow to create a class schedule. The following describes some of the underlying assumptions and constraints used in the program. It was found that there were overlapping steps between departments. For example both the Mathematics and Computer Science Departments assign classes with one lab. The program was designed to look at a semester rather than a full year for the required course load. Full time faculty members are assigned a semester course load of three and one half. In addition, the program assumes that a class is worth one credit of total semester course load and each lab is worth one half credit of semester course load. The information obtained from the department chairs and Guyette's paper were also beneficial in determining the scheduling process. The final set of scheduling constraints used by the program include: the classes being offered by the department, when the classes are offered, the faculty member's seniority, number of classes a faculty member can teach in a semester, the faculty member's time preference, and a faculty member cannot be in two places at one time. The implementation of these constraints will be discussed later.

Once the constraints were formulated, and the rules and ideas of scheduling began to take shape, the next step was to decide officially what tools to use for system implementation. There were two key issues that had a major influence in the decision making process. First, the data needed to be easily retrieved and stored for the program
to be effective. Second, the program needed to be user-friendly to people with a wide range of computer skills. With these ideas in mind, the program was created with PowerBuilder (by Powersoft), an object oriented window driven programming utility. In essence, the program is a point and click application that is self-explanatory to the user. The Watcom SQL relational database is used to store the necessary knowledge. Use of a database allows quick and easy updates of the knowledge used by the program and efficient implementation of the system.

System Overview:

This application is designed to schedule three different types of classes: regular classes without labs, a class with a lab, and a class with two labs. The first type of class, a class without labs, is the easiest type of class to schedule. This is because there are not as many possible time conflicts for the faculty member. A second type, a class with one lab, is a bit more difficult since it requires a faculty member to have more time slots available to teach the class and lab. The last type of class, a class with two labs, is the most difficult to schedule. A major reason for the difficulty is the classes are split into two sections and there needs to be even more time slots open for the faculty members. Consequently, three test sets were developed. The first has only classes without labs. The second has both classes without labs and classes with one lab. A third and final test set had all three types of classes. Actual data is discussed further in the experimental results.

Before the program could be tested, the data needs to be entered. This data is information regarding the classes that need to be taught and the faculty members that are available for teaching each semester. The required information for each class is as follows: term, department, course number, section number, course name, class meeting days, start time, end time, campus the class is located on, maximum credits, minimum credits, building the class is taught in, room number, class limit, flags associated with the class, if
there is a lab with the class, days the lab meets, start time of lab, end time of lab, campus
the lab is on, maximum credits for the lab, minimum credits for the lab, building the lab is
in, and room number the lab is in. Only the term, department, course number, class
meeting days, start time, end time, and similar lab information is used by the scheduling
algorithm. The remainder of the information is used for the final schedule only. A final
schedule can be printed and submitted without having to add the additional information
that is needed for the class schedule catalog.

The scheduling algorithm must also attempt to accommodate the requests of the
faculty members. This accommodation is carried out by the computer provided that the
faculty member information is inputted. This information is as follows: the department
they teach, their name, whether they are currently teaching, what their time preference is,
whether they teach full time or part time, the first four classes they would like to teach,
four of the courses they have taught before, either the priority they have in the scheduling
process or three pieces of information (whether they are tenured, years at CSB/SJU,
highest degree earned) to determine their seniority, and whether he or she is the
department chair. Faculty member information is used to create a schedule with classes
and times that they prefer.

The system uses four different user objects to manipulate the previously mentioned
data and produce a schedule. As the program starts, a window displaying all class
information opens (Figure 1). The user has four options. First, the user can select a term
and department and go straight to scheduling (Figure 2). The second option is to select a
term and department and edit existing class information or add new class information
(Figure 3). A third option is to add a new term and/or a new department so new
information can be added regarding classes and faculty members (Figure 4). The fourth
option is to view the faculty member's information (Figure 5). If the fourth option is
selected the user has four more options. First, if a term was previously selected the user
can select a department and go to scheduling. Second, the user can add a new department (Figure 6). Third, the user can select a department and edit existing information about the faculty members or add new faculty members (Figure 7). Fourth, the user can go back and view the class information. Overall, these windows enable the user to enter or to modify the information necessary for creating a class schedule.

![Class Information Window](image)

**Figure 1: Class Information Window**

<table>
<thead>
<tr>
<th>Term</th>
<th>Department</th>
<th>Class Name</th>
<th>Course Number</th>
<th>Section Number</th>
</tr>
</thead>
<tbody>
<tr>
<td>Fall 94</td>
<td>csci</td>
<td>intro to computing</td>
<td>120</td>
<td>01a</td>
</tr>
<tr>
<td>Fall 94</td>
<td>csci</td>
<td>intro/com science</td>
<td>150</td>
<td>01a</td>
</tr>
<tr>
<td>Fall 94</td>
<td>csci</td>
<td>intro/com science</td>
<td>150</td>
<td>02a</td>
</tr>
<tr>
<td>Fall 94</td>
<td>csci</td>
<td>intro/com science</td>
<td>150</td>
<td>03a</td>
</tr>
<tr>
<td>Fall 94</td>
<td>csci</td>
<td>intro/com science</td>
<td>150</td>
<td>04a</td>
</tr>
<tr>
<td>Fall 94</td>
<td>csci</td>
<td>prob/solv/prog &amp; com</td>
<td>160</td>
<td>01a</td>
</tr>
<tr>
<td>Fall 94</td>
<td>csci</td>
<td>prob/solv/prog &amp; com</td>
<td>190</td>
<td>02a</td>
</tr>
<tr>
<td>Fall 94</td>
<td>csci</td>
<td>data structures</td>
<td>200</td>
<td>01a</td>
</tr>
<tr>
<td>Fall 94</td>
<td>csci</td>
<td>business systems</td>
<td>330</td>
<td>01a</td>
</tr>
<tr>
<td>Fall 94</td>
<td>csci</td>
<td>esoterica</td>
<td>350</td>
<td>01a</td>
</tr>
<tr>
<td>Fall 94</td>
<td>phy</td>
<td>physics I</td>
<td>105</td>
<td>01a</td>
</tr>
<tr>
<td>Fall 94</td>
<td>phy</td>
<td>physics I</td>
<td>105</td>
<td>02a</td>
</tr>
<tr>
<td>Fall 94</td>
<td>phy</td>
<td>physics II</td>
<td>105</td>
<td>03a</td>
</tr>
<tr>
<td>Fall 94</td>
<td>phy</td>
<td>physics II</td>
<td>150</td>
<td>01a</td>
</tr>
</tbody>
</table>
Figure 2: Scheduling Window

Figure 3: Add Term/Department Window
Figure 4: Class Maintenance Window
Figure 5: Faculty Member Information Window

Figure 6: Add Department Window
The outline of the algorithm is as follows:

//Phase 1 - Assign as many of the classes by preference
TriggerEvent(ch_start, "phase1")

//Phase 2 - Sort the entries into to many classes, to few classes, okay number of classes
TriggerEvent(ch_start, "phase2")

//Phase 3 - Finish assigning any remaining classes (if any) according to previous experience
TriggerEvent(ch_start, "phase3")

//Phase 4 - Assign any remaining classes (if any) to any professor that has a slot open
lv_i_Ans = MessageBox("Seeking Information", "At this time, would you like to have any remaining classes \"+ & "assigned to a faculty member that is available during this class time? The other option is to wait until the \"+ & 
\"end to assign the remaining classes\", Question!, YesNo!
IF (lv_i_Ans = 1) THEN
TriggerEvent(ch_start, "phase4")
END IF

//Phase 5 - Move all unassigned classes (if any) to a separate data window
TriggerEvent(ch_start, "phase5")

//Phase 6 - Let professors with the least amount of credits choose from the last acquired courses
// of the professors with extra credits (based on preference)
TriggerEvent(ch_start, "phase6")
//Phase 7 - Let professors with the least amount of credits choose from the last acquired courses
// of the professors with extra credits (based on previously taught)
TriggerEvent(cb_start, "phase7")

//Phase 8 - Assign any remaining classes to any professor that has that time slot open - provided that this was not done
during Phase 4
TriggerEvent(cb_start, "phase8")

//Phase 9 - Move all assigned classes to the done data window ... notify if there are any leftover classes
TriggerEvent(cb_start, "phase9")

There are a couple of preprocessing steps followed by a nine phase scheduling process. The following paragraphs give an overall description of the entire process. The first step occurs as the window opens. All data regarding classes and faculty members for the selected term and department are loaded and the program checks for an existing final schedule. If it finds a final schedule, it notifies the user and explains what needs to be done if the user desires a new schedule. The remaining steps and phases occur once the start button is clicked. The program then initializes the array containing the faculty members availability, determines the number of classes needing faculty members, and determines the number of faculty members needing classes. Also, it initializes the temporary data window with the professor's name, priority, course credits desired, and course credits obtained. Course credits obtained will have a value of one-half if the faculty member is the department chair, and zero otherwise. After these initial steps are completed, the actual scheduling process begins.

In phase one, the faculty members are given as many of their preferred courses as are available. The faculty member with highest priority gets first choice and the faculty member with lowest priority gets last choice. As each faculty member gets the chance to find a course, the program goes through the preferences until a course is found that fits. This means that if the faculty member with priority three cannot get their first preference, then they will have an opportunity to get their second preference before a faculty member with priority four has any opportunities. Once all four preferences are checked the faculty member is temporarily out of luck for obtaining more classes.
During phase two, faculty members are moved around according to the number of classes they obtained. Any faculty member having less than their required course credits minus a half course credit has their data moved to the group that still needs classes. If the faculty member is within a half credit of his or her required course credit load, their data is moved to the group that does not need classes or does not need to delete classes. All other data remains with the group designed for faculty members that need to give up some of their classes.

In order for phase three to execute, there must be some classes that still need to be assigned to faculty members. If there are classes remaining to be scheduled, then the faculty members that still need classes are assigned a class that they have previously taught. The first faculty member assigned classes is the faculty member with the least number of credits. If there is more than one faculty member with the same number of credits, the faculty member with the highest priority goes first. After the preferences of all the faculty members still needing classes have been processed or all classes have been assigned, the faculty member's data is moved to the appropriate group. If the faculty member has more credits than his or her required course credit load, their data is moved to the group that needs to give up classes. If the faculty member has sufficient course credits, the faculty member's data is moved to the group whose schedule is complete. The remaining faculty member's data stays with the group requiring more classes.

Phase four requires two preprocessing steps. First it requires the user to specify if the remaining classes are to be assigned to any faculty member that needs classes and is available during the time slot. If the user does, then there needs to be classes remaining to be assigned. When both these conditions are met, the program simply goes through each faculty member and looks for an available class to teach. After a class is assigned to a faculty member, the program checks to see if the faculty member's data should be moved to the group with extra credits or the group with just enough credits. Again, the first
faculty member who searches for a class is the faculty member with the least credits. If there is more than one faculty member with the same number of credits, the faculty member with highest priority is first.

In phase five, if there are any classes that have not been assigned yet, the data on these classes is moved. If there are classes remaining, it copies the course number and section number of the class into another location. This class information is used later in phase eight.

Phase six attempts to swap classes between faculty members. The group needing to reduce their course credit load is sorted so the faculty member with the most credits is first. If there is more than one faculty member with the same number of credits, the faculty member with highest priority is first. The group of faculty members, requiring more course credits, is sorted so that the faculty member requiring course credits is first. Sorting then occurs within each similar course credit group, so the highest priority faculty member goes first. Next, courses are assigned to each faculty member needing a class by searching for their course preference in the last class assigned to the faculty member with the most number of extra course credits. If that is not the correct course or if the faculty member is not available to teach it, the faculty member looks for a class from the next faculty member in need of deleting a class. All classes are considered until a class is found. If no matches occur, then the search begins again with the next preference. After a class is reassigned in this manner, the faculty member that gained a class and the faculty member that lost a class have the credits checked against their required course load. They are then moved to the appropriate group. This process continues until either there are no faculty members in the group that still needs course credits or all preferences have been exhausted.
Phase seven does the same processing as phase six with one exception. Instead of looking for classes based on preference it looks for classes based on past experience.

Phase eight is the last attempt to assign any remaining classes to faculty members that still need course credits. If there are any classes, the course number and section number can be found in the unassigned course group created in phase five. As a class is assigned it is removed from the unassigned course group, the faculty member's credits are compared, and the faculty member moves to the appropriate group if necessary.

During phase nine, all the faculty member's classes have their information copied into the final schedule along with the faculty member's name. If there is a lab with the course, the faculty member's name is placed in the lab faculty member location. The user is notified when classes still need to be scheduled. If the user wants the final class schedule saved to the database, they click the okay button. If they do not want the changes saved, they need to click the cancel button.

The program has an enhanced feature set beyond those offered by Guyette's program as follows. The need for input/output files was eliminated by using a database and entering and modifying the input data with the application. Also, the use of a database eliminated the need to convert the input data into workable information for the system and the need to convert the final schedule back into understandable information for the user. Moreover, the scheduling program allows classes to be assigned on previous experience whereas Guyette's program does not. Unlike Guyette's scheduling program, this program does try to schedule classes with separate labs. Other obvious differences occur due to the times and days classes are offered at CSB/SJU. For example, Guyette's scheduling program works with classes offered on days of the week not the six-day cycle. His program also allows two classes per time slot instead of working with the times the class is offered.
Experimental Results:

Experiment one was to create a class schedule for the English department classes offered during Spring term of 1995. Input data for three of the twenty-three classes are shown below:

spring 95  engl  352  01a intro to lit  246  08:00:00  09:10:00  csb  d  4  hab  117  30  hml  n
spring 95  engl  383  01a capstone  135  11:20:00  12:30:00  csb  d  4  bac  130  30  hml disc w n

This class data represented the first section of the Intro to Lit course that is offered on days 246 from 8:00 to 9:10 on the CSB campus in the HAB room 117. There is a limit of thirty students in the class. It carries a hml distinction and has no labs. The only course of Shakespeare is offered on days 246 from 11:20 to 12:30 on the CSB campus in the HAB room 120. Again, there is a limit of thirty students in the class. It carries a hmu distinction, discussion and writing flags, and has no labs. The third class featured is the only section of Capstone offered on days 135 from 11:20 to 12:30 on the CSB campus in the BAC room 130. There is a class limit of thirty students. It carries a writing flag and has no labs. The other input data is information on the faculty members. This featured data includes three of the fifteen active faculty members in the English Department.

First is Nancy Hynes who teaches full time and prefers to teach in the morning. She prefers to teach courses 352 Shakespeare, 135 Intro to Lit, 383 Post Colonial Lit, and 213 Creative Writing I. Previously she taught 364, 383, 135, and 242. She has tenure, worked at CSB/SJU for 19 years, and has a Ph.D. These attributes give her a priority of five and she is not the chair of the department. In this experiment, priorities are determined by sorting on the following three fields in this order: the faculty member having tenure, the number of years teaching at CSB/SJU, and finally on the faculty member's highest degree earned. Another faculty member is Cindy Malone. She teaches full time and does not have a time preference for teaching. She would like to teach 135
Intro to Lit, 385 Capstone, 211 Writing NonFict Prose, and 242 Amer Lit in Context. Previously, Cindy taught courses 135, 382, 311, and 383. She has tenure, worked at CSB/SJU for four years, and has a Ph.D. These attributes give her a priority of fourteen and she is not the department chair. The last featured faculty member is Luke Mancuso who also teaches full time and has no time preference. He would like to teach 242 Amer Lit in Context, 135 Intro to Lit, 211 Writing NonFict Prose, and 364 Mod Poetry/Engl. He has no previous teaching experience with these classes because this is his first year. Along with being his first year, he does not have tenure, but does have a Ph.D. Thus, his priority is fifteen and he is not the department chair. For this experiment the faculty member's course preferences and previously taught courses were determined by random drawing. A complete listing of both the class information and faculty member information for the English department for Spring term 1995, can be found in Appendix A.

After the application processed the input data, a final class schedule was created. All twenty-three classes were assigned faculty members to teach the courses. No comparison could be made to the actual class schedule made by the department chair because, as mentioned earlier, course preferences and course experience was selected by a random drawing. A sample of the final schedule for the three faculty members discussed above is as follows:

| Spring 95 | Eng | 136 | 01 | Intro to Lit | 216 | 08:00:00 | 09:50:00 | Cab 4 4 | Hab 14 36 | Hyness, Nancy |
| Spring 95 | Eng | 135 | 02 | Intro to Lit | 135 | 09:40:00 | 10:50:00 | Cab 4 4 | Hab 120 30 | Mancuso, Luke |
| Spring 95 | Eng | 135 | 03 | Intro to Lit | 135 | 11:00:40 | 12:10:00 | Cab 4 4 | Oehl 107 30 | Hyness, Nancy |
| Spring 95 | Eng | 352 | 01 | Shakespeare | 246 | 11:00:00 | 12:10:00 | Cab 4 4 | Hab 120 30 | Mancuso, Luke |

The entire schedule may be found in Appendix A. Nancy Hynes obtained her first preference most likely because she has a higher priority and her second choice because of her priority and the class' lack of popularity among faculty members. Cindy Malone was also able to obtain her first preference not directly because of her priority but because she desired a course with low popularity among the faculty as well. Luke Mancuso was
unable to obtain his first choice but was given his second choice due to his low priority. In any case, everyone was assigned some of the courses they desired.

The goal of experiment two was to test the application with some classes that have one lab associated with it. As mentioned previously, the Mathematics Department will be the source for the test. In addition, more classes must be assigned to fewer faculty when compared with the English Department. Portions of the input data regarding the thirty classes offered by the Mathematics Department for Spring term 1995 are as follows:

| Spring 95 | Math 114 01A | Math exploration | 246 08:00:00 | 09:10:00 | Sju 4 | 4 Science 232 36 30 30  n |
| Spring 95 | Math 110 01A | Calculus I       | 246 08:00:00 | 09:10:00 | SjB 4 | 4 Arzella 142 30 30  n y |
| Spring 95 | Math 124 01A | Prob Stat Infer  | 245 08:00:00 | 09:10:00 | SjB 4 | 4 Arzella 127  30 30  n y |
| Spring 95 | Math 239 01A | Linear Algebra  | 246 13:00:00 | 14:10:00 | SjB 4 | 4 Hart 120 30 30  n n    |
| Spring 95 | Math 246 01A | Math Stat II    | 125 11:30:00 | 12:30:00 | Sju 4 | 4 Science 233 30 30  n n |

Of the above classes, only the 119 course has a lab associated with it. There are more classes with labs in the complete input data for all Spring term 1995 classes located in Appendix A. The results for only three faculty members are considered as follows:

| Math  | Gass, Michael | Y Either | Full Time | 124 114 322 344 120 123 346 241 Y 9 PhD 6 Y |
|-------|---------------|---------|-----------|----------|----------------------------------|
| Math  | Dumonceaux, Robert | Y Either | Full Time | 346 124 120 322 124 121 120 119 Y 30 PhD 2 n |
| Math  | Brodie, Marc  | Y Either | Full Time | 299 114 119 124 | n i 1 PhD 11 n |

Michael Gass has the added responsibility of being the department chair. Robert Dumonceaux has a very high priority because he has taught at CSB/SJU for many years. On the other hand, Marc Brodie has a low priority because this is his first year teaching at CSB/SJU. The information regarding the faculty members preferred courses and previously taught classes were again assigned by a random drawing. All data concerning the thirty classes and eleven faculty members will be used to produce a final class schedule of the classes taught by the Mathematics Department for Spring term 1995.

By using the input data, a final class schedule was produced by the application. All thirty classes and seven labs were assigned faculty members. Many faculty members obtained the classes they wanted to teach. A complete final schedule is located in Appendix A but the schedule for the three faculty members considered is as follows:
The classes obtained by the three faculty members were in their list of preferred courses. In fact, Robert Dumonceaux obtained his first preference and two sections of his second preference. Besides being the department chair, Michael Gass will be teaching his first preference, Probs and Stats, and his second preference, Math Explorations. The new faculty member, Marc Brodie, will be teaching his first preference, linear algebra, and his third preference, Calculus I. The class scheduling application assigned all courses taught by the Mathematics Department for Spring term 1995. Again, a comparison could not be made to the actual class schedule because the faculty member’s course preferences and course experiences were selected randomly.

The final experiment was to determine how the application scheduled classes having two labs. Experiment three attempted to produce a schedule for the Computer Science Department. Although there are fewer classes to schedule for this department, this case is difficult because all three types of classes, including the classes with two labs must be scheduled. As usual the standard input data for the classes offered and faculty members were entered. The following is a sample of the seven classes offered:

Although the 150 classes are listed in two sections, it is one class with two labs. The 200 class is a class with one lab. Finally, the 330 class is a class without a lab. The following is a sample of two of the four faculty members in the Computer Science Department:
One feature being used is the ability to assign a priority to a faculty member and not determine the priority by using the following three pieces of information: tenure, years at CSB/SJU, and highest degree earned. One may also note that both types of data may be entered regarding methods of determining priority. The priority values used during the scheduling process are determined from the existing data and stored in the database. The faculty members did not request the classes listed, but they were assigned for the purpose of testing. Complete data regarding classes and faculty members is located in Appendix A.

A class schedule was produced using the existing data for the Computer Science Department for Fall term 1994. All seven classes and eight labs were assigned faculty members. Four of these classes are listed below:

- **Fall 94 CSCI 150 001 intro/comp science**
  - 8-10:00:00 09:00:00 10:50:00:00 10:00:00 11:50:00:00 11:00:00 12:50:00 13:40:00 14:30:00:00
  - Chollou, Dan

- **Fall 94 CSCI 150 002 intro/comp science**
  - 8-10:00:00 09:00:00 10:50:00:00 10:00:00 11:50:00:00 11:00:00 12:50:00 13:40:00 14:30:00:00
  - Chollou, Dan

- **Fall 94 CSCI 150 003 intro/comp science**
  - 8-10:00:00 09:00:00 10:50:00:00 10:00:00 11:50:00:00 11:00:00 12:50:00 13:40:00 14:30:00:00
  - Holey, Andy

- **Fall 94 CSCI 330 01A algorithms**
  - 8-10:00:00 09:00:00 10:50:00:00 10:00:00 11:50:00:00 11:00:00 12:50:00 13:40:00 14:30:00:00
  - Chollou, Dan

Dan Chollou obtained his second course preference and third course preference; Andy Holey obtained his first course preference and third course preference. The complete final schedule list is located in Appendix A. As with the first two experiments, the class scheduling application produced a reasonable class schedule for the Computer Science Department courses for Fall term 1994. The course preferences were assigned by random drawing and course experiences were selected by the experimenter's experience with the Computer Science Department so no comparisons were made to the actual schedule. However, it can be noted that the final schedule seems reasonable to similar schedules for the Computer Science Department.
Discussion of Results:

The most important result of all three experiments is simply that a full class schedule was produced for each data set. In other words, all courses were assigned a faculty member to teach the class. The application is also able to handle scheduling three types of classes: regular class without a lab, a class with one lab, and a class with two labs. These were shown by experiment one, experiment two, and experiment three respectively. One feature is that the application tries to accommodate for the faculty member's teaching time preference. The options for time preference are morning, afternoon, or either. Input data for the courses and faculty members along with the final schedule are located in Appendix A. Potential problems can arise from trying to schedule classes with two labs. In fact, on a trial run of data for the Computer Science Department Spring term 1995, one of the 150 courses was not assigned to a faculty member. The cause of the problem stems from the fact that an assumption was coded into the application. This assumption was that the faculty member who taught the class would also teach the labs. This scenario is not the case in the Computer Science Department. However, it is difficult to have the computer keep track of labs that are not attached to a class or a section number. Some difficulties arose when the information given by different people conflicted. When this occurs the developer tried to make reasonable assumptions. If this method fails, one goes back to the department chairs for clarification. These discrepancies occurred due to a lack of understanding on the developer's part. Some discrepancies were caused because the experts do not necessarily have the ability to completely explain how the process works. Another main problem was obtaining incorrect information. For example, at this point, the developer is not positive what the correct answer is to the relative worth of a lab. For the most part it was a reasonable assumption to have the faculty member teach both the class and the labs that are a part of the class. In any case, the results of the three experiments produced complete and viable class schedules. Because Guyette's paper did
not provide all of the class data and faculty member data, his data could not be run to test
the application or to compare the final schedule produced by both applications.

Summary and Conclusion:

By the end of the project, a reasonable understanding of how class scheduling
should work was obtained. This required understanding both the general concept of
scheduling and how a knowledge-based system could be used for scheduling. With this
knowledge, a working scheduling program using PowerBuilder (by Powersoft) was
created.

The scheduling program was developed and tested using data for three different
departments: English, Mathematics, and Computer Science. These departments were
chosen to demonstrate the application's ability to handle three different types of classes:
regular class without a lab, a class with one lab, and a class with two labs. The English
Department offers only classes without labs, the Mathematics Department offers both
classes without labs and classes with one lab, and the Computer Science Department
offers all three types of classes. The application produced a complete and viable class
schedule for each department. This result indicated a possible improvement over the
application described in Guyette's system since his system scheduled only thirty-nine out of
the forty classes. In addition, a clearer understanding of the software package used in the
development and implementation of the application was ascertained. PowerBuilder (by
Powersoft) not only enhanced the developers understanding of a window based
applications but also taught the developer more about the practical use of databases.
ACKNOWLEDGMENTS

Special thanks to my advisor Dr. Dan Challou for finding a technically challenging thesis topic. His direction, guidance, and patience were greatly appreciated. Appreciation and thanks to Dr. Lynn Ziegler and Dr. Leonard Valley for providing class scheduling information. Thanks are due to the readers, Dr. Andrew Holey and Dennis Myers, for their time and effort to make this project complete. To each of you, thank you for the interest you have shown in my thesis.

Acknowledgment and thanks to the Honors Department for creating a stimulating and challenging Honors Program. And special thanks to Andersen Consulting Co., Inc., especially the Customer Service Direct group, for the opportunity to learn and work with Powerbuilder during my summer internship. Thank you to my family and friends, who each helped in their own way.
BIBLIOGRAPHY


APPENDIX A
### Class information for English Department Spring term 1995:

<table>
<thead>
<tr>
<th>term</th>
<th>dept</th>
<th>course/ses</th>
<th>title</th>
<th>days</th>
<th>start</th>
<th>end</th>
<th>loc</th>
<th>credit</th>
<th>building/rm</th>
<th>limit</th>
<th>flags</th>
<th>lab</th>
</tr>
</thead>
<tbody>
<tr>
<td>spring 95</td>
<td>engl 135</td>
<td>01a</td>
<td>intro to lit</td>
<td>246</td>
<td>08:00</td>
<td>09:10</td>
<td>csb</td>
<td>4</td>
<td>hab 117</td>
<td>30</td>
<td>hml</td>
<td>n</td>
</tr>
<tr>
<td>spring 95</td>
<td>engl 135</td>
<td>02a</td>
<td>intro to lit</td>
<td>135</td>
<td>09:40</td>
<td>10:50</td>
<td>csb</td>
<td>4</td>
<td>hab 120</td>
<td>30</td>
<td>hml</td>
<td>n</td>
</tr>
<tr>
<td>spring 95</td>
<td>engl 135</td>
<td>03a</td>
<td>intro to lit</td>
<td>135</td>
<td>11:20</td>
<td>12:30</td>
<td>csb</td>
<td>4</td>
<td>ars 107</td>
<td>30</td>
<td>hml</td>
<td>n</td>
</tr>
<tr>
<td>spring 95</td>
<td>engl 135</td>
<td>04a</td>
<td>intro to lit</td>
<td>246</td>
<td>09:40</td>
<td>10:50</td>
<td>csb</td>
<td>4</td>
<td>hab 128a</td>
<td>30</td>
<td>hml</td>
<td>n</td>
</tr>
<tr>
<td>spring 95</td>
<td>engl 135</td>
<td>05a</td>
<td>intro to lit</td>
<td>246</td>
<td>09:40</td>
<td>10:50</td>
<td>csb</td>
<td>4</td>
<td>hab 015</td>
<td>30</td>
<td>hml</td>
<td>n</td>
</tr>
<tr>
<td>spring 95</td>
<td>engl 211</td>
<td>01a</td>
<td>writing, nonfic prose</td>
<td>246</td>
<td>11:20</td>
<td>12:30</td>
<td>csb</td>
<td>4</td>
<td>hab 118</td>
<td>30</td>
<td>glo</td>
<td>w</td>
</tr>
<tr>
<td>spring 95</td>
<td>engl 211</td>
<td>02a</td>
<td>writing, nonfic prose</td>
<td>246</td>
<td>13:00</td>
<td>14:10</td>
<td>csb</td>
<td>4</td>
<td>hab 118</td>
<td>30</td>
<td>glo</td>
<td>w</td>
</tr>
<tr>
<td>spring 95</td>
<td>engl 213</td>
<td>01a</td>
<td>creative writing i</td>
<td>135</td>
<td>13:00</td>
<td>14:10</td>
<td>csb</td>
<td>4</td>
<td>bac 108</td>
<td>30</td>
<td>writ</td>
<td>n</td>
</tr>
<tr>
<td>spring 95</td>
<td>engl 241</td>
<td>01a</td>
<td>brit lit in context</td>
<td>135</td>
<td>11:20</td>
<td>12:30</td>
<td>sju</td>
<td>4</td>
<td>quad 457</td>
<td>30</td>
<td>n</td>
<td>n</td>
</tr>
<tr>
<td>spring 95</td>
<td>engl 241</td>
<td>02a</td>
<td>brit lit in context</td>
<td>246</td>
<td>11:20</td>
<td>12:30</td>
<td>sju</td>
<td>4</td>
<td>quad 353</td>
<td>30</td>
<td>witer</td>
<td>n</td>
</tr>
<tr>
<td>spring 95</td>
<td>engl 242</td>
<td>01a</td>
<td>amer lit in context</td>
<td>135</td>
<td>11:20</td>
<td>12:30</td>
<td>sju</td>
<td>4</td>
<td>quad 459</td>
<td>30</td>
<td>gender</td>
<td>n</td>
</tr>
<tr>
<td>spring 95</td>
<td>engl 242</td>
<td>02a</td>
<td>amer lit in context</td>
<td>246</td>
<td>11:20</td>
<td>12:30</td>
<td>sju</td>
<td>4</td>
<td>quad 361</td>
<td>30</td>
<td>n</td>
<td>n</td>
</tr>
<tr>
<td>spring 95</td>
<td>engl 283</td>
<td>01a</td>
<td>western lit</td>
<td>135</td>
<td>13:00</td>
<td>14:10</td>
<td>csb</td>
<td>4</td>
<td>hab 128a</td>
<td>30</td>
<td>hml</td>
<td>n</td>
</tr>
<tr>
<td>spring 95</td>
<td>engl 311</td>
<td>01a</td>
<td>advanced writing</td>
<td>135</td>
<td>14:40</td>
<td>15:50</td>
<td>csb</td>
<td>4</td>
<td>cleml 131</td>
<td>30</td>
<td>writing</td>
<td>n</td>
</tr>
<tr>
<td>spring 95</td>
<td>engl 311</td>
<td>02a</td>
<td>advanced writing</td>
<td>246</td>
<td>14:40</td>
<td>15:50</td>
<td>csb</td>
<td>4</td>
<td>hab 101</td>
<td>30</td>
<td>writing</td>
<td>n</td>
</tr>
<tr>
<td>spring 95</td>
<td>engl 347</td>
<td>01a</td>
<td>amer lit after 1865</td>
<td>135</td>
<td>14:40</td>
<td>15:50</td>
<td>sju</td>
<td>4</td>
<td>quad 353</td>
<td>30</td>
<td>n</td>
<td>n</td>
</tr>
<tr>
<td>spring 95</td>
<td>engl 351</td>
<td>01a</td>
<td>arious writer</td>
<td>246</td>
<td>09:40</td>
<td>10:50</td>
<td>sju</td>
<td>4</td>
<td>quad 341</td>
<td>30</td>
<td>hmsm</td>
<td>w</td>
</tr>
<tr>
<td>spring 95</td>
<td>engl 352</td>
<td>01a</td>
<td>shakespeare</td>
<td>246</td>
<td>11:20</td>
<td>12:30</td>
<td>csb</td>
<td>4</td>
<td>hab 120</td>
<td>30</td>
<td>hmsm</td>
<td>w</td>
</tr>
<tr>
<td>spring 95</td>
<td>engl 364</td>
<td>01a</td>
<td>modern poetry/engl</td>
<td>135</td>
<td>13:00</td>
<td>14:10</td>
<td>sju</td>
<td>4</td>
<td>quad 457</td>
<td>30</td>
<td>n</td>
<td>n</td>
</tr>
<tr>
<td>spring 95</td>
<td>engl 369</td>
<td>01a</td>
<td>crit. theory/prac</td>
<td>135</td>
<td>09:40</td>
<td>10:50</td>
<td>sju</td>
<td>4</td>
<td>quad 457</td>
<td>30</td>
<td>n</td>
<td>n</td>
</tr>
<tr>
<td>spring 95</td>
<td>engl 382</td>
<td>01a</td>
<td>minority literature</td>
<td>135</td>
<td>09:40</td>
<td>10:50</td>
<td>csb</td>
<td>4</td>
<td>hab 121</td>
<td>30</td>
<td>n</td>
<td>n</td>
</tr>
<tr>
<td>spring 95</td>
<td>engl 383</td>
<td>01a</td>
<td>post-colonial lit</td>
<td>246</td>
<td>13:00</td>
<td>14:10</td>
<td>csb</td>
<td>4</td>
<td>hab 121</td>
<td>30</td>
<td>global</td>
<td>n</td>
</tr>
<tr>
<td>spring 95</td>
<td>engl 385</td>
<td>01a</td>
<td>capstone</td>
<td>135</td>
<td>11:20</td>
<td>12:30</td>
<td>csb</td>
<td>4</td>
<td>bac 130</td>
<td>30</td>
<td>writing</td>
<td>n</td>
</tr>
</tbody>
</table>

### Faculty member information for English Department Spring term 1995:

<table>
<thead>
<tr>
<th>dept</th>
<th>faculty member</th>
<th>active time</th>
<th>work load</th>
<th>course preferences</th>
<th>previous courses</th>
<th>tenure, yrs, degree, priority, chair</th>
</tr>
</thead>
<tbody>
<tr>
<td>engl</td>
<td>dufner, angeline</td>
<td>y either</td>
<td>full time</td>
<td>385 242 352 383</td>
<td>353 313 211 383</td>
<td>y 35 ma 1 n</td>
</tr>
<tr>
<td>engl</td>
<td>carls, jr</td>
<td>y either</td>
<td>full time</td>
<td>351 311 242 242</td>
<td>347 313 385 213</td>
<td>y 18 phd 7 n</td>
</tr>
<tr>
<td>engl</td>
<td>faulkner, maria</td>
<td>y either</td>
<td>full time</td>
<td>382 242 352 347</td>
<td>341 241 211 369</td>
<td>y 12 phd 10 n</td>
</tr>
<tr>
<td>engl</td>
<td>freeman, barbara</td>
<td>y either</td>
<td>full time</td>
<td>369 311 211 383</td>
<td>211 351 311 382</td>
<td>y 19 phd 5 n</td>
</tr>
<tr>
<td>engl</td>
<td>hynes, nancy</td>
<td>y am</td>
<td>full time</td>
<td>352 315 383 213</td>
<td>364 383 135 242</td>
<td>y 4 phd 14 n</td>
</tr>
<tr>
<td>engl</td>
<td>malone, cindy</td>
<td>y either</td>
<td>full time</td>
<td>355 385 211 242</td>
<td>353 382 311 383</td>
<td>n 1 phd 15 n</td>
</tr>
<tr>
<td>engl</td>
<td>mancebo, luke</td>
<td>y either</td>
<td>full time</td>
<td>242 335 211 364</td>
<td></td>
<td>y 7 phd 12 n</td>
</tr>
<tr>
<td>engl</td>
<td>marin, elaine</td>
<td>y either</td>
<td>full time</td>
<td>383 347 242 213</td>
<td>351 335 211 311</td>
<td>y 21 phd 4 n</td>
</tr>
<tr>
<td>engl</td>
<td>mcelroy, paul</td>
<td>y either</td>
<td>full time</td>
<td>213 241 352 242</td>
<td>353 351 242 347</td>
<td>y 14 phd 9 n</td>
</tr>
<tr>
<td>engl</td>
<td>melon, virginia</td>
<td>y either</td>
<td>full time</td>
<td>383 211 352 242</td>
<td>213 311 135 242</td>
<td>y 17 phd 8 n</td>
</tr>
<tr>
<td>engl</td>
<td>mitra, madhu</td>
<td>y either</td>
<td>full time</td>
<td>383 242 385 135</td>
<td>347 369 242 364</td>
<td>y 5 phd 13 n</td>
</tr>
<tr>
<td>engl</td>
<td>opitz, michael</td>
<td>y either</td>
<td>full time</td>
<td>311 241 383 347</td>
<td>351 211 241 382</td>
<td>y 27 phd 3 n</td>
</tr>
<tr>
<td>engl</td>
<td>peters, patricia</td>
<td>y either</td>
<td>full time</td>
<td>352 241 213 369</td>
<td>242 353 385 347</td>
<td>y 22 phd 3 n</td>
</tr>
<tr>
<td>engl</td>
<td>thomson, hilary</td>
<td>y either</td>
<td>full time</td>
<td>213 242 383 135</td>
<td>385 283 311 241</td>
<td>y 17 phd 8 n</td>
</tr>
<tr>
<td>engl</td>
<td>thomson, charles</td>
<td>y either</td>
<td>full time</td>
<td>364 211 352 352</td>
<td>311 242 351 364</td>
<td>y 17 phd 8 n</td>
</tr>
<tr>
<td>term</td>
<td>dept</td>
<td>course/sec</td>
<td>title</td>
<td>days</td>
<td>start</td>
<td>end</td>
</tr>
<tr>
<td>-------</td>
<td>------</td>
<td>------------</td>
<td>----------------------------</td>
<td>------</td>
<td>-----------</td>
<td>----------</td>
</tr>
<tr>
<td>spring 95</td>
<td>engl</td>
<td>135 01a</td>
<td>intro to lit</td>
<td>246</td>
<td>08:00:00</td>
<td>09:10:00</td>
</tr>
<tr>
<td>spring 95</td>
<td>engl</td>
<td>135 02a</td>
<td>intro to lit</td>
<td>135</td>
<td>09:40:00</td>
<td>10:50:00</td>
</tr>
<tr>
<td>spring 95</td>
<td>engl</td>
<td>135 03a</td>
<td>intro to lit</td>
<td>135</td>
<td>11:20:00</td>
<td>12:30:00</td>
</tr>
<tr>
<td>spring 95</td>
<td>engl</td>
<td>135 04a</td>
<td>intro to lit</td>
<td>246</td>
<td>09:40:00</td>
<td>10:50:00</td>
</tr>
<tr>
<td>spring 95</td>
<td>engl</td>
<td>135 05a</td>
<td>intro to lit</td>
<td>246</td>
<td>09:40:00</td>
<td>10:50:00</td>
</tr>
<tr>
<td>spring 95</td>
<td>engl</td>
<td>211 01a</td>
<td>writing, nonfic prose</td>
<td>246</td>
<td>11:20:00</td>
<td>12:30:00</td>
</tr>
<tr>
<td>spring 95</td>
<td>engl</td>
<td>211 02a</td>
<td>writing, nonfic prose</td>
<td>246</td>
<td>13:00:00</td>
<td>14:10:00</td>
</tr>
<tr>
<td>spring 95</td>
<td>engl</td>
<td>213 01a</td>
<td>creative writing I</td>
<td>135</td>
<td>13:00:00</td>
<td>14:10:00</td>
</tr>
<tr>
<td>spring 95</td>
<td>engl</td>
<td>241 01a</td>
<td>brit lit in context</td>
<td>135</td>
<td>11:20:00</td>
<td>12:30:00</td>
</tr>
<tr>
<td>spring 95</td>
<td>engl</td>
<td>241 02a</td>
<td>brit lit in context</td>
<td>246</td>
<td>11:20:00</td>
<td>12:30:00</td>
</tr>
<tr>
<td>spring 95</td>
<td>engl</td>
<td>242 01a</td>
<td>amer lit in context</td>
<td>135</td>
<td>11:20:00</td>
<td>12:30:00</td>
</tr>
<tr>
<td>spring 95</td>
<td>engl</td>
<td>242 02a</td>
<td>amer lit in context</td>
<td>246</td>
<td>11:20:00</td>
<td>12:30:00</td>
</tr>
<tr>
<td>spring 95</td>
<td>engl</td>
<td>283 01a</td>
<td>western lit</td>
<td>135</td>
<td>13:00:00</td>
<td>14:10:00</td>
</tr>
<tr>
<td>spring 95</td>
<td>engl</td>
<td>311 01a</td>
<td>advanced writing</td>
<td>135</td>
<td>14:40:00</td>
<td>15:50:00</td>
</tr>
<tr>
<td>spring 95</td>
<td>engl</td>
<td>311 02a</td>
<td>advanced writing</td>
<td>246</td>
<td>14:40:00</td>
<td>15:50:00</td>
</tr>
<tr>
<td>spring 95</td>
<td>engl</td>
<td>347 01a</td>
<td>amer lit after 1865</td>
<td>135</td>
<td>14:40:00</td>
<td>15:50:00</td>
</tr>
<tr>
<td>spring 95</td>
<td>engl</td>
<td>351 01a</td>
<td>chamiere</td>
<td>246</td>
<td>09:40:00</td>
<td>10:50:00</td>
</tr>
<tr>
<td>spring 95</td>
<td>engl</td>
<td>352 01a</td>
<td>shakespeare</td>
<td>246</td>
<td>11:20:00</td>
<td>12:30:00</td>
</tr>
<tr>
<td>spring 95</td>
<td>engl</td>
<td>364 01a</td>
<td>mod poetry/english</td>
<td>135</td>
<td>13:00:00</td>
<td>14:10:00</td>
</tr>
<tr>
<td>spring 95</td>
<td>engl</td>
<td>369 01a</td>
<td>crit. theory/prac</td>
<td>135</td>
<td>09:40:00</td>
<td>10:50:00</td>
</tr>
<tr>
<td>spring 95</td>
<td>engl</td>
<td>382 01a</td>
<td>minority literature</td>
<td>135</td>
<td>09:40:00</td>
<td>10:50:00</td>
</tr>
<tr>
<td>spring 95</td>
<td>engl</td>
<td>383 01a</td>
<td>post-colonial lit</td>
<td>246</td>
<td>13:00:00</td>
<td>14:10:00</td>
</tr>
<tr>
<td>spring 95</td>
<td>engl</td>
<td>385 01a</td>
<td>capstone</td>
<td>135</td>
<td>11:20:00</td>
<td>12:30:00</td>
</tr>
</tbody>
</table>

faculty member:
- malone, cindy
- maccioso, luke
- hynes, nancy
- thimmesh, hilary
- melton, virginia
- modarby, patrick
- thornbury, charles
- modarby, patrick
- peters, patricia
- opitz, michael
- thimmesh, hilary
- mattr, madhu
- peters, patricia
- opitz, michael
- freedman, barbara
- martin, elaine
- earls, jp
- hynes, nancy
- thornbury, charles
- freedman, barbara
- faulkner, mara
- melton, virginia
- dufner, angeline

### Final class schedule for the Mathematics Department Spring term 1995:

<table>
<thead>
<tr>
<th>term</th>
<th>dept</th>
<th>course/sec</th>
<th>title</th>
<th>days</th>
<th>start</th>
<th>end</th>
<th>loc</th>
<th>credits</th>
<th>building</th>
<th>term</th>
<th>limit</th>
<th>flags</th>
<th>faculty member</th>
<th>lab information</th>
</tr>
</thead>
<tbody>
<tr>
<td>spring 95</td>
<td>math 114 01a</td>
<td>math exploration</td>
<td>246</td>
<td>08:00:00</td>
<td>09:10:00</td>
<td>sja 4 4</td>
<td>science 233</td>
<td>30</td>
<td>mt</td>
<td>gass, michael</td>
<td>hartz, dave</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>spring 95</td>
<td>math 114 02a</td>
<td>math exploration</td>
<td>246</td>
<td>09:40:00</td>
<td>10:50:00</td>
<td>sju 4 4</td>
<td>science 231</td>
<td>30</td>
<td>mt</td>
<td>brodie, marc</td>
<td>35</td>
<td>14:40:00</td>
<td>15:50:00</td>
<td>sja 0</td>
</tr>
<tr>
<td>spring 95</td>
<td>math 119 01a</td>
<td>calculus i</td>
<td>246</td>
<td>08:00:00</td>
<td>09:10:00</td>
<td>ceb 4 4</td>
<td>ardorf 142</td>
<td>30</td>
<td>mt</td>
<td>byrne, phil</td>
<td>24</td>
<td>14:40:00</td>
<td>15:50:00</td>
<td>sju 0</td>
</tr>
<tr>
<td>spring 95</td>
<td>math 119 02a</td>
<td>calculus i</td>
<td>135</td>
<td>09:40:00</td>
<td>10:50:00</td>
<td>sju 4 4</td>
<td>science 215</td>
<td>30</td>
<td>mt</td>
<td>byrne, phil</td>
<td>35</td>
<td>08:00:00</td>
<td>09:10:00</td>
<td>sju 0</td>
</tr>
<tr>
<td>spring 95</td>
<td>math 120 01a</td>
<td>calculus ii</td>
<td>246</td>
<td>08:00:00</td>
<td>09:10:00</td>
<td>ceb 4 4</td>
<td>hab 101</td>
<td>30</td>
<td>mt</td>
<td>lenz, jerry</td>
<td>24</td>
<td>14:40:00</td>
<td>15:50:00</td>
<td>ceb 0</td>
</tr>
<tr>
<td>spring 95</td>
<td>math 120 02a</td>
<td>calculus ii</td>
<td>135</td>
<td>11:20:00</td>
<td>12:30:00</td>
<td>ceb 4 4</td>
<td>hab 101</td>
<td>30</td>
<td>mt</td>
<td>lange, jack</td>
<td>24</td>
<td>08:00:00</td>
<td>09:10:00</td>
<td>sja 0</td>
</tr>
<tr>
<td>spring 95</td>
<td>math 120 03a</td>
<td>calculus ii</td>
<td>246</td>
<td>14:40:00</td>
<td>15:50:00</td>
<td>ceb 4 4</td>
<td>hab 101</td>
<td>30</td>
<td>mt</td>
<td>tangredi, mike</td>
<td>24</td>
<td>14:40:00</td>
<td>15:50:00</td>
<td>sja 0</td>
</tr>
<tr>
<td>spring 95</td>
<td>math 121 01a</td>
<td>foundation</td>
<td>246</td>
<td>08:00:00</td>
<td>09:10:00</td>
<td>ceb 4 4</td>
<td>hab 102b</td>
<td>30</td>
<td>mt</td>
<td>lenz, jerry</td>
<td>sibley, tom</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>spring 95</td>
<td>math 121 02a</td>
<td>foundation</td>
<td>246</td>
<td>09:40:00</td>
<td>10:50:00</td>
<td>ceb 4 4</td>
<td>hab 012b</td>
<td>30</td>
<td>mt</td>
<td>gass, michael</td>
<td>sibley, tom</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>spring 95</td>
<td>math 122 01a</td>
<td>finite math</td>
<td>135</td>
<td>08:00:00</td>
<td>09:10:00</td>
<td>sju 4 4</td>
<td>science 231</td>
<td>30</td>
<td>mt</td>
<td>galovich, jennifer</td>
<td>lange, jack</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>spring 95</td>
<td>math 122 02a</td>
<td>finite math</td>
<td>135</td>
<td>11:20:00</td>
<td>12:30:00</td>
<td>sju 4 4</td>
<td>science 233</td>
<td>30</td>
<td>mt</td>
<td>galovich, jennifer</td>
<td>lange, jack</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>spring 95</td>
<td>math 123 01a</td>
<td>essential calc</td>
<td>135</td>
<td>11:20:00</td>
<td>12:30:00</td>
<td>ceb 4 4</td>
<td>ardorf 105</td>
<td>30</td>
<td>mt</td>
<td>tangredi, mike</td>
<td>24</td>
<td>08:00:00</td>
<td>09:10:00</td>
<td>sja 0</td>
</tr>
<tr>
<td>spring 95</td>
<td>math 124 01a</td>
<td>prob stat infer</td>
<td>135</td>
<td>08:00:00</td>
<td>09:10:00</td>
<td>ceb 4 4</td>
<td>ardorf 105</td>
<td>30</td>
<td>mt</td>
<td>lenz, jerry</td>
<td>sibley, tom</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>spring 95</td>
<td>math 124 02a</td>
<td>prob stat infer</td>
<td>135</td>
<td>09:40:00</td>
<td>10:50:00</td>
<td>ceb 4 4</td>
<td>ardorf 105</td>
<td>30</td>
<td>mt</td>
<td>gass, michael</td>
<td>sibley, tom</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>spring 95</td>
<td>math 124 03a</td>
<td>prob stat infer</td>
<td>246</td>
<td>09:40:00</td>
<td>10:50:00</td>
<td>sju 4 4</td>
<td>science 215</td>
<td>30</td>
<td>mt</td>
<td>galati, shobha</td>
<td>sibley, tom</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>spring 95</td>
<td>math 124 04a</td>
<td>prob stat infer</td>
<td>246</td>
<td>11:20:00</td>
<td>12:30:00</td>
<td>sju 4 4</td>
<td>science 213</td>
<td>30</td>
<td>mt</td>
<td>galati, shobha</td>
<td>sibley, tom</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>spring 95</td>
<td>math 124 05a</td>
<td>prob stat infer</td>
<td>135</td>
<td>11:20:00</td>
<td>12:30:00</td>
<td>sju 4 4</td>
<td>science 213</td>
<td>30</td>
<td>mt</td>
<td>galati, shobha</td>
<td>sibley, tom</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>spring 95</td>
<td>math 124 06a</td>
<td>prob stat infer</td>
<td>246</td>
<td>11:20:00</td>
<td>12:30:00</td>
<td>sju 4 4</td>
<td>science 231</td>
<td>30</td>
<td>mt</td>
<td>galati, shobha</td>
<td>sibley, tom</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>spring 95</td>
<td>math 124 07a</td>
<td>prob stat infer</td>
<td>246</td>
<td>13:00:00</td>
<td>14:10:00</td>
<td>sju 4 4</td>
<td>science 231</td>
<td>30</td>
<td>mt</td>
<td>galati, shobha</td>
<td>sibley, tom</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>spring 95</td>
<td>math 124 08a</td>
<td>prob stat infer</td>
<td>135</td>
<td>14:40:00</td>
<td>15:50:00</td>
<td>sju 4 4</td>
<td>science 233</td>
<td>30</td>
<td>mt</td>
<td>galati, shobha</td>
<td>sibley, tom</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>spring 95</td>
<td>math 239 01a</td>
<td>linear algebra</td>
<td>246</td>
<td>13:00:00</td>
<td>14:10:00</td>
<td>ceb 4 4</td>
<td>hab 101</td>
<td>30</td>
<td>mt</td>
<td>brodie, marc</td>
<td>sibley, tom</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>spring 95</td>
<td>math 241 01a</td>
<td>found/struct</td>
<td>246</td>
<td>09:40:00</td>
<td>10:50:00</td>
<td>ceb 4 4</td>
<td>ardorf 121</td>
<td>30</td>
<td>mt</td>
<td>byrne, phil</td>
<td>galovich, jennifer</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>spring 95</td>
<td>math 241 02a</td>
<td>found/struct</td>
<td>135</td>
<td>09:40:00</td>
<td>10:50:00</td>
<td>sju 4 4</td>
<td>science 231</td>
<td>30</td>
<td>mt</td>
<td>galovich, jennifer</td>
<td>galovich, jennifer</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>spring 95</td>
<td>math 320 01a</td>
<td>combinatorics</td>
<td>246</td>
<td>09:40:00</td>
<td>10:50:00</td>
<td>sju 4 4</td>
<td>science 231</td>
<td>30</td>
<td>mt</td>
<td>galovich, jennifer</td>
<td>galovich, jennifer</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>spring 95</td>
<td>math 332 01a</td>
<td>algebraic struct ii</td>
<td>135</td>
<td>09:40:00</td>
<td>10:50:00</td>
<td>sju 4 4</td>
<td>science 233</td>
<td>30</td>
<td>mt</td>
<td>sibley, tom</td>
<td>sibley, tom</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>spring 95</td>
<td>math 337 01a</td>
<td>diff equations</td>
<td>246</td>
<td>13:00:00</td>
<td>14:10:00</td>
<td>ceb 4 4</td>
<td>bac 104</td>
<td>30</td>
<td>mt</td>
<td>tangredi, mike</td>
<td>lange, jack</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>spring 95</td>
<td>math 341 01a</td>
<td>fourier series</td>
<td>135</td>
<td>13:00:00</td>
<td>14:10:00</td>
<td>sju 4 4</td>
<td>science 215</td>
<td>30</td>
<td>mt</td>
<td>lange, jack</td>
<td>lange, jack</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>spring 95</td>
<td>math 343 01a</td>
<td>analysis i</td>
<td>246</td>
<td>11:20:00</td>
<td>12:30:00</td>
<td>ceb 4 4</td>
<td>ardorf 105</td>
<td>30</td>
<td>mt</td>
<td>hartz, dave</td>
<td>sibley, tom</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>spring 95</td>
<td>math 346 01a</td>
<td>math stat ii</td>
<td>135</td>
<td>11:20:00</td>
<td>12:30:00</td>
<td>sju 4 4</td>
<td>science 231</td>
<td>30</td>
<td>mt</td>
<td>galovich, jennifer</td>
<td>sibley, tom</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Class information for Computer Science Department Fall term 1994:

<table>
<thead>
<tr>
<th>term</th>
<th>dept</th>
<th>course/sec title</th>
<th>days</th>
<th>start</th>
<th>end</th>
<th>loc</th>
<th>credit</th>
<th>building/ rm</th>
<th>limit</th>
<th>flags</th>
<th>lab</th>
<th>lab lab</th>
<th>information</th>
</tr>
</thead>
<tbody>
<tr>
<td>fall 94</td>
<td>csci</td>
<td>120 01a intro to computing</td>
<td>24</td>
<td>08:00</td>
<td>09:10</td>
<td>sju</td>
<td>4 4</td>
<td>science 215</td>
<td>25</td>
<td>q r</td>
<td>y</td>
<td>6</td>
<td>08:00</td>
</tr>
<tr>
<td>fall 94</td>
<td>csci</td>
<td>150 01a intro/comp science</td>
<td>246</td>
<td>08:00</td>
<td>09:10</td>
<td>csh</td>
<td>4 4</td>
<td>arsdlf 104</td>
<td>15</td>
<td>q r, ns</td>
<td>y</td>
<td>1</td>
<td>08:00</td>
</tr>
<tr>
<td>fall 94</td>
<td>csci</td>
<td>150 02a intro/comp science</td>
<td>246</td>
<td>08:00</td>
<td>09:10</td>
<td>csh</td>
<td>4 4</td>
<td>arsdlf 104</td>
<td>15</td>
<td>q r, ns</td>
<td>y</td>
<td>3</td>
<td>08:00</td>
</tr>
<tr>
<td>fall 94</td>
<td>csci</td>
<td>150 03a intro/comp science</td>
<td>246</td>
<td>08:00</td>
<td>09:10</td>
<td>csh</td>
<td>4 4</td>
<td>arsdlf 104</td>
<td>15</td>
<td>q r, ns</td>
<td>y</td>
<td>6</td>
<td>08:00</td>
</tr>
<tr>
<td>fall 94</td>
<td>csci</td>
<td>150 04a intro/comp science</td>
<td>135</td>
<td>14:40</td>
<td>15:50</td>
<td>sju</td>
<td>4 4</td>
<td>science 233</td>
<td>15</td>
<td>q r, ns</td>
<td>y</td>
<td>4</td>
<td>08:00</td>
</tr>
<tr>
<td>fall 94</td>
<td>csci</td>
<td>160 01a prob/solv/Prob &amp; Cons</td>
<td>246</td>
<td>09:40</td>
<td>10:50</td>
<td>csh</td>
<td>4 4</td>
<td>arsdlf 105</td>
<td>15</td>
<td>q r, ns</td>
<td>y</td>
<td>5</td>
<td>08:00</td>
</tr>
<tr>
<td>fall 94</td>
<td>csci</td>
<td>160 02a prob/solv/Prob &amp; Cons</td>
<td>246</td>
<td>09:40</td>
<td>10:50</td>
<td>csh</td>
<td>4 4</td>
<td>arsdlf 105</td>
<td>15</td>
<td>q r, ns</td>
<td>y</td>
<td>5</td>
<td>13:00</td>
</tr>
<tr>
<td>fall 94</td>
<td>csci</td>
<td>200 01a data structures</td>
<td>135</td>
<td>09:40</td>
<td>10:50</td>
<td>sju</td>
<td>4 4</td>
<td>science 215</td>
<td>15</td>
<td>wrt</td>
<td>y</td>
<td>4</td>
<td>13:00</td>
</tr>
<tr>
<td>fall 94</td>
<td>csci</td>
<td>330 01a business systems</td>
<td>135</td>
<td>13:00</td>
<td>14:10</td>
<td>sju</td>
<td>4 4</td>
<td>science 233</td>
<td>30</td>
<td>n</td>
<td>0</td>
<td>00:00</td>
<td>00:00:00</td>
</tr>
<tr>
<td>fall 94</td>
<td>csci</td>
<td>338 01a algorithms</td>
<td>246</td>
<td>13:00</td>
<td>14:10</td>
<td>sju</td>
<td>4 4</td>
<td>science 215</td>
<td>30</td>
<td>wrt</td>
<td>n</td>
<td>0</td>
<td>00:00</td>
</tr>
</tbody>
</table>

Faculty member information for Computer Science Department Fall term 1994:

<table>
<thead>
<tr>
<th>dept</th>
<th>faculty member</th>
<th>active time</th>
<th>work load</th>
<th>course preferences</th>
<th>previous courses</th>
<th>set priority</th>
<th>tenure, yrs, degree, priority, chair</th>
</tr>
</thead>
<tbody>
<tr>
<td>csci</td>
<td>chann, dam</td>
<td>y either full time</td>
<td>160 150 317 120 150 330 317</td>
<td>3</td>
<td>n</td>
<td></td>
<td></td>
</tr>
<tr>
<td>csci</td>
<td>howey, andy</td>
<td>y either full time</td>
<td>350 210 150 160 150 200 317</td>
<td>n</td>
<td>4</td>
<td>phd</td>
<td>2</td>
</tr>
<tr>
<td>csci</td>
<td>miller, john</td>
<td>y either full time</td>
<td>210 120 150 160 120 150 200 210</td>
<td>n</td>
<td>1</td>
<td>ms</td>
<td>4</td>
</tr>
<tr>
<td>csci</td>
<td>ziegler, lynn</td>
<td>y either full time</td>
<td>160 210 350 120 200 338 150 160 1</td>
<td>1</td>
<td>y</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Final class schedule for the Computer Science Department Fall term 1994:

<table>
<thead>
<tr>
<th>term</th>
<th>dept</th>
<th>course/sec title</th>
<th>days</th>
<th>start</th>
<th>end</th>
<th>loc</th>
<th>credit</th>
<th>building/ rm</th>
<th>limit</th>
<th>flags</th>
<th>faculty member</th>
<th>lab lab</th>
<th>information</th>
</tr>
</thead>
<tbody>
<tr>
<td>fall 94</td>
<td>csci</td>
<td>120 01a intro to computing</td>
<td>24</td>
<td>08:00</td>
<td>09:10</td>
<td>sju</td>
<td>4 4</td>
<td>science 215</td>
<td>25</td>
<td>q r</td>
<td>ziegler, lynn</td>
<td>6</td>
<td>08:00</td>
</tr>
<tr>
<td>fall 94</td>
<td>csci</td>
<td>150 01a intro/comp science</td>
<td>246</td>
<td>08:00</td>
<td>09:10</td>
<td>csh</td>
<td>4 4</td>
<td>arsdlf 104</td>
<td>15</td>
<td>q r, ns</td>
<td>chann, dam</td>
<td>1</td>
<td>08:00</td>
</tr>
<tr>
<td>fall 94</td>
<td>csci</td>
<td>150 02a intro/comp science</td>
<td>246</td>
<td>08:00</td>
<td>09:10</td>
<td>csh</td>
<td>4 4</td>
<td>arsdlf 104</td>
<td>15</td>
<td>q r, ns</td>
<td>chann, dam</td>
<td>3</td>
<td>08:00</td>
</tr>
<tr>
<td>fall 94</td>
<td>csci</td>
<td>150 03a intro/comp science</td>
<td>246</td>
<td>08:00</td>
<td>09:10</td>
<td>csh</td>
<td>4 4</td>
<td>arsdlf 104</td>
<td>15</td>
<td>q r, ns</td>
<td>howey, andy</td>
<td>6</td>
<td>08:00</td>
</tr>
<tr>
<td>fall 94</td>
<td>csci</td>
<td>150 04a intro/comp science</td>
<td>246</td>
<td>08:00</td>
<td>09:10</td>
<td>csh</td>
<td>4 4</td>
<td>arsdlf 105</td>
<td>15</td>
<td>q r, ns</td>
<td>howey, andy</td>
<td>4</td>
<td>08:00</td>
</tr>
<tr>
<td>fall 94</td>
<td>csci</td>
<td>160 01a prob/solv/Prob &amp; Cons</td>
<td>246</td>
<td>09:40</td>
<td>10:50</td>
<td>csh</td>
<td>4 4</td>
<td>arsdlf 105</td>
<td>15</td>
<td>q r, ns</td>
<td>ziegler, lynn</td>
<td>5</td>
<td>08:00</td>
</tr>
<tr>
<td>fall 94</td>
<td>csci</td>
<td>160 02a prob/solv/Prob &amp; Cons</td>
<td>246</td>
<td>09:40</td>
<td>10:50</td>
<td>csh</td>
<td>4 4</td>
<td>arsdlf 105</td>
<td>15</td>
<td>q r, ns</td>
<td>ziegler, lynn</td>
<td>5</td>
<td>13:00</td>
</tr>
<tr>
<td>fall 94</td>
<td>csci</td>
<td>200 01a data structures</td>
<td>135</td>
<td>09:40</td>
<td>10:50</td>
<td>sju</td>
<td>4 4</td>
<td>science 215</td>
<td>15</td>
<td>wrt</td>
<td>miller, john</td>
<td>4</td>
<td>13:00</td>
</tr>
<tr>
<td>fall 94</td>
<td>csci</td>
<td>330 01a business systems</td>
<td>135</td>
<td>13:00</td>
<td>14:10</td>
<td>sju</td>
<td>4 4</td>
<td>science 233</td>
<td>30</td>
<td>wrt</td>
<td>howey, andy</td>
<td>4</td>
<td>13:00</td>
</tr>
<tr>
<td>fall 94</td>
<td>csci</td>
<td>338 01a algorithms</td>
<td>246</td>
<td>13:00</td>
<td>14:10</td>
<td>sju</td>
<td>4 4</td>
<td>science 215</td>
<td>30</td>
<td>wrt</td>
<td>howey, andy</td>
<td>4</td>
<td>13:00</td>
</tr>
</tbody>
</table>
**Class information for Computer Science Department Spring term 1995:**

<table>
<thead>
<tr>
<th>term</th>
<th>dept</th>
<th>course/sec</th>
<th>title</th>
<th>days</th>
<th>start</th>
<th>end</th>
<th>loc</th>
<th>credit</th>
<th>building/room</th>
<th>limit</th>
<th>flags</th>
<th>lab info</th>
</tr>
</thead>
<tbody>
<tr>
<td>spring 95</td>
<td>csci</td>
<td>120 01a</td>
<td>intro to computing</td>
<td>2</td>
<td>08:00</td>
<td>09:10</td>
<td>csb</td>
<td>4</td>
<td>arldof</td>
<td>104</td>
<td>25</td>
<td>y</td>
</tr>
<tr>
<td>spring 95</td>
<td>csci</td>
<td>150 01a</td>
<td>intro/comp science</td>
<td>246</td>
<td>09:40</td>
<td>10:50</td>
<td>sju</td>
<td>4</td>
<td>science</td>
<td>215</td>
<td>13</td>
<td>ns, y</td>
</tr>
<tr>
<td>spring 95</td>
<td>csci</td>
<td>150 02a</td>
<td>intro/comp science</td>
<td>246</td>
<td>09:40</td>
<td>10:50</td>
<td>sju</td>
<td>4</td>
<td>science</td>
<td>215</td>
<td>13</td>
<td>ns, qr</td>
</tr>
<tr>
<td>spring 95</td>
<td>csci</td>
<td>150 03a</td>
<td>intro/comp science</td>
<td>246</td>
<td>14:40</td>
<td>15:50</td>
<td>csb</td>
<td>4</td>
<td>arldof</td>
<td>104</td>
<td>13</td>
<td>ns, qr</td>
</tr>
<tr>
<td>spring 95</td>
<td>csci</td>
<td>150 04a</td>
<td>intro/comp science</td>
<td>246</td>
<td>14:40</td>
<td>15:50</td>
<td>csb</td>
<td>4</td>
<td>arldof</td>
<td>104</td>
<td>13</td>
<td>ns, qr</td>
</tr>
<tr>
<td>spring 95</td>
<td>csci</td>
<td>160 01a</td>
<td>problem-solvprog</td>
<td>246</td>
<td>14:40</td>
<td>15:50</td>
<td>sju</td>
<td>4</td>
<td>science</td>
<td>215</td>
<td>15</td>
<td>y</td>
</tr>
<tr>
<td>spring 95</td>
<td>csci</td>
<td>160 02a</td>
<td>problem-solvprog</td>
<td>246</td>
<td>14:40</td>
<td>15:50</td>
<td>sju</td>
<td>4</td>
<td>science</td>
<td>215</td>
<td>15</td>
<td>y, qr</td>
</tr>
<tr>
<td>spring 95</td>
<td>csci</td>
<td>210 01a</td>
<td>arch, lang, and apps</td>
<td>246</td>
<td>09:40</td>
<td>10:50</td>
<td>sju</td>
<td>4</td>
<td>arldof</td>
<td>104</td>
<td>15</td>
<td>y</td>
</tr>
<tr>
<td>spring 95</td>
<td>csci</td>
<td>210 02a</td>
<td>arch, lang, and apps</td>
<td>246</td>
<td>09:40</td>
<td>10:50</td>
<td>sju</td>
<td>4</td>
<td>arldof</td>
<td>104</td>
<td>15</td>
<td>y</td>
</tr>
<tr>
<td>spring 95</td>
<td>csci</td>
<td>317 01a</td>
<td>topics - graphics</td>
<td>246</td>
<td>11:20</td>
<td>12:30</td>
<td>sju</td>
<td>4</td>
<td>science</td>
<td>215</td>
<td>30</td>
<td>n</td>
</tr>
<tr>
<td>spring 95</td>
<td>csci</td>
<td>350 01a</td>
<td>operating systems</td>
<td>246</td>
<td>14:40</td>
<td>15:50</td>
<td>sju</td>
<td>4</td>
<td>science</td>
<td>233</td>
<td>30</td>
<td>disc</td>
</tr>
</tbody>
</table>

**Faculty member information for Computer Science Department Spring term 1995:**

<table>
<thead>
<tr>
<th>dept</th>
<th>faculty member</th>
<th>active time</th>
<th>work load</th>
<th>course preferences</th>
<th>previous courses</th>
<th>set priority, tenure, yrs, degree, priority, chair</th>
</tr>
</thead>
<tbody>
<tr>
<td>csci</td>
<td>chailou, dan</td>
<td>y</td>
<td>either</td>
<td>full time</td>
<td>160 150 330 120 150 330 317 3</td>
<td>3 n</td>
</tr>
<tr>
<td>csci</td>
<td>holey, andy</td>
<td>y</td>
<td>either</td>
<td>full time</td>
<td>338 200 150 160 150 160 200 317 n 4 phd 2 n</td>
<td>1 n</td>
</tr>
<tr>
<td>csci</td>
<td>miller, john</td>
<td>y</td>
<td>either</td>
<td>full time</td>
<td>200 120 150 160 120 150 160 210 n 1 ms 4 n</td>
<td>1 y</td>
</tr>
<tr>
<td>csci</td>
<td>ziegler, lynx</td>
<td>y</td>
<td>either</td>
<td>full time</td>
<td>160 200 338 120 200 338 150 160 1</td>
<td>1 y</td>
</tr>
</tbody>
</table>

**Final class schedule for the Computer Science Department Spring term 1995:**

<table>
<thead>
<tr>
<th>term</th>
<th>dept</th>
<th>course/sec</th>
<th>title</th>
<th>days</th>
<th>start</th>
<th>end</th>
<th>loc</th>
<th>credit</th>
<th>building/room</th>
<th>limit</th>
<th>flags</th>
<th>lab info</th>
</tr>
</thead>
<tbody>
<tr>
<td>spring 95</td>
<td>csci</td>
<td>120 01a</td>
<td>intro to computing</td>
<td>2</td>
<td>08:00</td>
<td>09:10</td>
<td>csb</td>
<td>4</td>
<td>arldof</td>
<td>104</td>
<td>25</td>
<td>y</td>
</tr>
<tr>
<td>spring 95</td>
<td>csci</td>
<td>150 01a</td>
<td>intro/comp science</td>
<td>246</td>
<td>09:40</td>
<td>10:50</td>
<td>sju</td>
<td>4</td>
<td>science</td>
<td>215</td>
<td>13</td>
<td>ns, y</td>
</tr>
<tr>
<td>spring 95</td>
<td>csci</td>
<td>150 02a</td>
<td>intro/comp science</td>
<td>246</td>
<td>09:40</td>
<td>10:50</td>
<td>sju</td>
<td>4</td>
<td>science</td>
<td>215</td>
<td>13</td>
<td>ns, qr</td>
</tr>
<tr>
<td>spring 95</td>
<td>csci</td>
<td>150 03a</td>
<td>intro/comp science</td>
<td>246</td>
<td>14:40</td>
<td>15:50</td>
<td>csb</td>
<td>4</td>
<td>arldof</td>
<td>104</td>
<td>13</td>
<td>ns, qr</td>
</tr>
<tr>
<td>spring 95</td>
<td>csci</td>
<td>150 04a</td>
<td>intro/comp science</td>
<td>246</td>
<td>14:40</td>
<td>15:50</td>
<td>csb</td>
<td>4</td>
<td>arldof</td>
<td>104</td>
<td>13</td>
<td>ns, qr</td>
</tr>
<tr>
<td>spring 95</td>
<td>csci</td>
<td>160 01a</td>
<td>problem-solvprog</td>
<td>246</td>
<td>14:40</td>
<td>15:50</td>
<td>sju</td>
<td>4</td>
<td>science</td>
<td>215</td>
<td>15</td>
<td>y</td>
</tr>
<tr>
<td>spring 95</td>
<td>csci</td>
<td>160 02a</td>
<td>problem-solvprog</td>
<td>246</td>
<td>14:40</td>
<td>15:50</td>
<td>sju</td>
<td>4</td>
<td>science</td>
<td>215</td>
<td>15</td>
<td>y</td>
</tr>
<tr>
<td>spring 95</td>
<td>csci</td>
<td>210 01a</td>
<td>arch, lang, and apps</td>
<td>246</td>
<td>09:40</td>
<td>10:50</td>
<td>sju</td>
<td>4</td>
<td>arldof</td>
<td>104</td>
<td>15</td>
<td>y</td>
</tr>
<tr>
<td>spring 95</td>
<td>csci</td>
<td>210 02a</td>
<td>arch, lang, and apps</td>
<td>246</td>
<td>09:40</td>
<td>10:50</td>
<td>sju</td>
<td>4</td>
<td>arldof</td>
<td>104</td>
<td>15</td>
<td>y</td>
</tr>
<tr>
<td>spring 95</td>
<td>csci</td>
<td>317 01a</td>
<td>topics - graphics</td>
<td>246</td>
<td>11:20</td>
<td>12:30</td>
<td>sju</td>
<td>4</td>
<td>science</td>
<td>215</td>
<td>30</td>
<td>disc</td>
</tr>
<tr>
<td>spring 95</td>
<td>csci</td>
<td>350 01a</td>
<td>operating systems</td>
<td>246</td>
<td>14:40</td>
<td>15:50</td>
<td>sju</td>
<td>4</td>
<td>science</td>
<td>233</td>
<td>30</td>
<td>disc</td>
</tr>
</tbody>
</table>
APPENDIX B
w_class_main_test

Showing uo_class_main with initial data
Showing uo_class_main with data
d_add_dept_term

* Includes two hidden data windows:
  1. d_department_list
  2. d_term_list

w_add_term_dept

Invoked in response to Add Term/Department Button
w_maint_main (includes uo_class_maint)

Invoked in response to Edit Button
w_maint_main (includes uo_class_maint)

Invoked in response to Edit Button and Select Button
```
<table>
<thead>
<tr>
<th>Course Number</th>
<th>Section Number</th>
<th>Name</th>
<th>Day(s)</th>
<th>Start Time</th>
<th>End Time</th>
<th>Location</th>
</tr>
</thead>
<tbody>
<tr>
<td>100</td>
<td>01x</td>
<td>Intro to computing</td>
<td>M,W</td>
<td>09:00-09:50</td>
<td>09:00-10:00</td>
<td>sp</td>
</tr>
<tr>
<td>101</td>
<td>01x</td>
<td>Intro to comp science</td>
<td>T,TH</td>
<td>14:30-15:20</td>
<td>14:30-15:50</td>
<td>sp</td>
</tr>
<tr>
<td>102</td>
<td>01x</td>
<td>Intro to comp science</td>
<td>M,W</td>
<td>12:00-12:50</td>
<td>12:00-13:00</td>
<td>sp</td>
</tr>
<tr>
<td>103</td>
<td>01x</td>
<td>Intro to comp science</td>
<td>T,TH</td>
<td>14:30-15:20</td>
<td>14:30-15:50</td>
<td>sp</td>
</tr>
<tr>
<td>104</td>
<td>01x</td>
<td>Intro to comp science</td>
<td>M,W</td>
<td>12:00-12:50</td>
<td>12:00-13:00</td>
<td>sp</td>
</tr>
<tr>
<td>105</td>
<td>01x</td>
<td>Intro to comp science</td>
<td>T,TH</td>
<td>14:30-15:20</td>
<td>14:30-15:50</td>
<td>sp</td>
</tr>
<tr>
<td>106</td>
<td>01x</td>
<td>Intro to comp science</td>
<td>M,W</td>
<td>12:00-12:50</td>
<td>12:00-13:00</td>
<td>sp</td>
</tr>
<tr>
<td>107</td>
<td>01x</td>
<td>Intro to comp science</td>
<td>T,TH</td>
<td>14:30-15:20</td>
<td>14:30-15:50</td>
<td>sp</td>
</tr>
<tr>
<td>108</td>
<td>01x</td>
<td>Intro to comp science</td>
<td>M,W</td>
<td>12:00-12:50</td>
<td>12:00-13:00</td>
<td>sp</td>
</tr>
<tr>
<td>109</td>
<td>01x</td>
<td>Intro to comp science</td>
<td>T,TH</td>
<td>14:30-15:20</td>
<td>14:30-15:50</td>
<td>sp</td>
</tr>
<tr>
<td>110</td>
<td>01x</td>
<td>Intro to comp science</td>
<td>M,W</td>
<td>12:00-12:50</td>
<td>12:00-13:00</td>
<td>sp</td>
</tr>
</tbody>
</table>

w_maint_main (Includes uo_class_maint)
Invoked in response to Reset Button
```
### Class Information

<table>
<thead>
<tr>
<th>Location</th>
<th>Max Credits</th>
<th>Min Credits</th>
<th>Building</th>
<th>Room</th>
<th>Class Limit</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>108</td>
<td>10</td>
</tr>
</tbody>
</table>

**Flag:** Is there a scheduled lab with the class?

- y

### Lab Information:

<table>
<thead>
<tr>
<th>Day(s)</th>
<th>Start Time</th>
<th>End Time</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>06:00:00</td>
<td>10:30:00</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Location</th>
<th>Max Credits</th>
<th>Min Credits</th>
<th>Building</th>
<th>Room</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>217</td>
</tr>
</tbody>
</table>

Invoked with More Button
w_class_main_test

Showing uo_prof_main with initial data
w_class_main_test

Showing uo_prof_main with data
d_add_dept

- Includes two hidden data windows:
  1. d_department_list
  2. d_term_list

w_add_dept

Invoked in response to Add Department Button
includes hidden data window:
1. d_professor

uses functions:
1. last_name_length
2. calc_priority

w_maint_main (includes uo_prof_maint)

invoked in response to Edit Button
Facility Maintenance Management System

- Includes hidden data window:
  1. d_professor

- Uses functions:
  1. last_name_length
  2. calc_priority

w_maint_main (Includes uo_prof_maint)

Invoked in response to Edit Button
and
Select Button
Invoked with More Button
w_scheduling

Invoked in response to Scheduling Button
### Course Schedule for Fall 94-95

<table>
<thead>
<tr>
<th>Course Number</th>
<th>Section</th>
<th>Course Name</th>
<th>Faculty Name</th>
<th>Class Faculty Name</th>
</tr>
</thead>
<tbody>
<tr>
<td>120</td>
<td>01a</td>
<td>Intro to Computing</td>
<td>thirty-four</td>
<td>thirty-four</td>
</tr>
<tr>
<td>150</td>
<td>01c</td>
<td>Intro to Speech Science</td>
<td>thirty-four</td>
<td>thirty-four</td>
</tr>
<tr>
<td>150</td>
<td>02c</td>
<td>Intro to Speech Science</td>
<td>thirty-four</td>
<td>thirty-four</td>
</tr>
<tr>
<td>150</td>
<td>03a</td>
<td>Intro to Speech Science</td>
<td>thirty-four</td>
<td>thirty-four</td>
</tr>
<tr>
<td>150</td>
<td>04a</td>
<td>Intro to Speech Science</td>
<td>thirty-four</td>
<td>thirty-four</td>
</tr>
<tr>
<td>150</td>
<td>05a</td>
<td>Intro to Speech Science</td>
<td>thirty-four</td>
<td>thirty-four</td>
</tr>
<tr>
<td>150</td>
<td>06a</td>
<td>Intro to Speech Science</td>
<td>thirty-four</td>
<td>thirty-four</td>
</tr>
<tr>
<td>150</td>
<td>07a</td>
<td>Intro to Speech Science</td>
<td>thirty-four</td>
<td>thirty-four</td>
</tr>
<tr>
<td>150</td>
<td>08a</td>
<td>Intro to Speech Science</td>
<td>thirty-four</td>
<td>thirty-four</td>
</tr>
<tr>
<td>150</td>
<td>09a</td>
<td>Intro to Speech Science</td>
<td>thirty-four</td>
<td>thirty-four</td>
</tr>
<tr>
<td>150</td>
<td>10a</td>
<td>Intro to Speech Science</td>
<td>thirty-four</td>
<td>thirty-four</td>
</tr>
<tr>
<td>150</td>
<td>11a</td>
<td>Intro to Speech Science</td>
<td>thirty-four</td>
<td>thirty-four</td>
</tr>
<tr>
<td>150</td>
<td>12a</td>
<td>Intro to Speech Science</td>
<td>thirty-four</td>
<td>thirty-four</td>
</tr>
<tr>
<td>150</td>
<td>13a</td>
<td>Intro to Speech Science</td>
<td>thirty-four</td>
<td>thirty-four</td>
</tr>
<tr>
<td>150</td>
<td>14a</td>
<td>Intro to Speech Science</td>
<td>thirty-four</td>
<td>thirty-four</td>
</tr>
<tr>
<td>150</td>
<td>15a</td>
<td>Intro to Speech Science</td>
<td>thirty-four</td>
<td>thirty-four</td>
</tr>
<tr>
<td>150</td>
<td>16a</td>
<td>Intro to Speech Science</td>
<td>thirty-four</td>
<td>thirty-four</td>
</tr>
<tr>
<td>150</td>
<td>17a</td>
<td>Intro to Speech Science</td>
<td>thirty-four</td>
<td>thirty-four</td>
</tr>
<tr>
<td>150</td>
<td>18a</td>
<td>Intro to Speech Science</td>
<td>thirty-four</td>
<td>thirty-four</td>
</tr>
<tr>
<td>150</td>
<td>19a</td>
<td>Intro to Speech Science</td>
<td>thirty-four</td>
<td>thirty-four</td>
</tr>
<tr>
<td>150</td>
<td>20a</td>
<td>Intro to Speech Science</td>
<td>thirty-four</td>
<td>thirty-four</td>
</tr>
<tr>
<td>150</td>
<td>21a</td>
<td>Intro to Speech Science</td>
<td>thirty-four</td>
<td>thirty-four</td>
</tr>
<tr>
<td>150</td>
<td>22a</td>
<td>Intro to Speech Science</td>
<td>thirty-four</td>
<td>thirty-four</td>
</tr>
<tr>
<td>150</td>
<td>23a</td>
<td>Intro to Speech Science</td>
<td>thirty-four</td>
<td>thirty-four</td>
</tr>
<tr>
<td>150</td>
<td>24a</td>
<td>Intro to Speech Science</td>
<td>thirty-four</td>
<td>thirty-four</td>
</tr>
<tr>
<td>150</td>
<td>25a</td>
<td>Intro to Speech Science</td>
<td>thirty-four</td>
<td>thirty-four</td>
</tr>
<tr>
<td>150</td>
<td>26a</td>
<td>Intro to Speech Science</td>
<td>thirty-four</td>
<td>thirty-four</td>
</tr>
<tr>
<td>150</td>
<td>27a</td>
<td>Intro to Speech Science</td>
<td>thirty-four</td>
<td>thirty-four</td>
</tr>
<tr>
<td>150</td>
<td>28a</td>
<td>Intro to Speech Science</td>
<td>thirty-four</td>
<td>thirty-four</td>
</tr>
<tr>
<td>150</td>
<td>29a</td>
<td>Intro to Speech Science</td>
<td>thirty-four</td>
<td>thirty-four</td>
</tr>
<tr>
<td>150</td>
<td>30a</td>
<td>Intro to Speech Science</td>
<td>thirty-four</td>
<td>thirty-four</td>
</tr>
<tr>
<td>150</td>
<td>31a</td>
<td>Intro to Speech Science</td>
<td>thirty-four</td>
<td>thirty-four</td>
</tr>
<tr>
<td>150</td>
<td>32a</td>
<td>Intro to Speech Science</td>
<td>thirty-four</td>
<td>thirty-four</td>
</tr>
<tr>
<td>150</td>
<td>33a</td>
<td>Intro to Speech Science</td>
<td>thirty-four</td>
<td>thirty-four</td>
</tr>
<tr>
<td>150</td>
<td>34a</td>
<td>Intro to Speech Science</td>
<td>thirty-four</td>
<td>thirty-four</td>
</tr>
<tr>
<td>150</td>
<td>35a</td>
<td>Intro to Speech Science</td>
<td>thirty-four</td>
<td>thirty-four</td>
</tr>
<tr>
<td>150</td>
<td>36a</td>
<td>Intro to Speech Science</td>
<td>thirty-four</td>
<td>thirty-four</td>
</tr>
<tr>
<td>150</td>
<td>37a</td>
<td>Intro to Speech Science</td>
<td>thirty-four</td>
<td>thirty-four</td>
</tr>
<tr>
<td>150</td>
<td>38a</td>
<td>Intro to Speech Science</td>
<td>thirty-four</td>
<td>thirty-four</td>
</tr>
<tr>
<td>150</td>
<td>39a</td>
<td>Intro to Speech Science</td>
<td>thirty-four</td>
<td>thirty-four</td>
</tr>
<tr>
<td>150</td>
<td>40a</td>
<td>Intro to Speech Science</td>
<td>thirty-four</td>
<td>thirty-four</td>
</tr>
<tr>
<td>150</td>
<td>41a</td>
<td>Intro to Speech Science</td>
<td>thirty-four</td>
<td>thirty-four</td>
</tr>
<tr>
<td>150</td>
<td>42a</td>
<td>Intro to Speech Science</td>
<td>thirty-four</td>
<td>thirty-four</td>
</tr>
<tr>
<td>150</td>
<td>43a</td>
<td>Intro to Speech Science</td>
<td>thirty-four</td>
<td>thirty-four</td>
</tr>
<tr>
<td>150</td>
<td>44a</td>
<td>Intro to Speech Science</td>
<td>thirty-four</td>
<td>thirty-four</td>
</tr>
<tr>
<td>150</td>
<td>45a</td>
<td>Intro to Speech Science</td>
<td>thirty-four</td>
<td>thirty-four</td>
</tr>
<tr>
<td>150</td>
<td>46a</td>
<td>Intro to Speech Science</td>
<td>thirty-four</td>
<td>thirty-four</td>
</tr>
<tr>
<td>150</td>
<td>47a</td>
<td>Intro to Speech Science</td>
<td>thirty-four</td>
<td>thirty-four</td>
</tr>
<tr>
<td>150</td>
<td>48a</td>
<td>Intro to Speech Science</td>
<td>thirty-four</td>
<td>thirty-four</td>
</tr>
<tr>
<td>150</td>
<td>49a</td>
<td>Intro to Speech Science</td>
<td>thirty-four</td>
<td>thirty-four</td>
</tr>
<tr>
<td>150</td>
<td>50a</td>
<td>Intro to Speech Science</td>
<td>thirty-four</td>
<td>thirty-four</td>
</tr>
<tr>
<td>150</td>
<td>51a</td>
<td>Intro to Speech Science</td>
<td>thirty-four</td>
<td>thirty-four</td>
</tr>
<tr>
<td>150</td>
<td>52a</td>
<td>Intro to Speech Science</td>
<td>thirty-four</td>
<td>thirty-four</td>
</tr>
<tr>
<td>150</td>
<td>53a</td>
<td>Intro to Speech Science</td>
<td>thirty-four</td>
<td>thirty-four</td>
</tr>
<tr>
<td>150</td>
<td>54a</td>
<td>Intro to Speech Science</td>
<td>thirty-four</td>
<td>thirty-four</td>
</tr>
<tr>
<td>150</td>
<td>55a</td>
<td>Intro to Speech Science</td>
<td>thirty-four</td>
<td>thirty-four</td>
</tr>
<tr>
<td>150</td>
<td>56a</td>
<td>Intro to Speech Science</td>
<td>thirty-four</td>
<td>thirty-four</td>
</tr>
<tr>
<td>150</td>
<td>57a</td>
<td>Intro to Speech Science</td>
<td>thirty-four</td>
<td>thirty-four</td>
</tr>
<tr>
<td>150</td>
<td>58a</td>
<td>Intro to Speech Science</td>
<td>thirty-four</td>
<td>thirty-four</td>
</tr>
<tr>
<td>150</td>
<td>59a</td>
<td>Intro to Speech Science</td>
<td>thirty-four</td>
<td>thirty-four</td>
</tr>
<tr>
<td>150</td>
<td>60a</td>
<td>Intro to Speech Science</td>
<td>thirty-four</td>
<td>thirty-four</td>
</tr>
<tr>
<td>150</td>
<td>61a</td>
<td>Intro to Speech Science</td>
<td>thirty-four</td>
<td>thirty-four</td>
</tr>
<tr>
<td>150</td>
<td>62a</td>
<td>Intro to Speech Science</td>
<td>thirty-four</td>
<td>thirty-four</td>
</tr>
<tr>
<td>150</td>
<td>63a</td>
<td>Intro to Speech Science</td>
<td>thirty-four</td>
<td>thirty-four</td>
</tr>
<tr>
<td>150</td>
<td>64a</td>
<td>Intro to Speech Science</td>
<td>thirty-four</td>
<td>thirty-four</td>
</tr>
<tr>
<td>150</td>
<td>65a</td>
<td>Intro to Speech Science</td>
<td>thirty-four</td>
<td>thirty-four</td>
</tr>
<tr>
<td>150</td>
<td>66a</td>
<td>Intro to Speech Science</td>
<td>thirty-four</td>
<td>thirty-four</td>
</tr>
<tr>
<td>150</td>
<td>67a</td>
<td>Intro to Speech Science</td>
<td>thirty-four</td>
<td>thirty-four</td>
</tr>
<tr>
<td>150</td>
<td>68a</td>
<td>Intro to Speech Science</td>
<td>thirty-four</td>
<td>thirty-four</td>
</tr>
<tr>
<td>150</td>
<td>69a</td>
<td>Intro to Speech Science</td>
<td>thirty-four</td>
<td>thirty-four</td>
</tr>
<tr>
<td>150</td>
<td>70a</td>
<td>Intro to Speech Science</td>
<td>thirty-four</td>
<td>thirty-four</td>
</tr>
<tr>
<td>150</td>
<td>71a</td>
<td>Intro to Speech Science</td>
<td>thirty-four</td>
<td>thirty-four</td>
</tr>
<tr>
<td>150</td>
<td>72a</td>
<td>Intro to Speech Science</td>
<td>thirty-four</td>
<td>thirty-four</td>
</tr>
<tr>
<td>150</td>
<td>73a</td>
<td>Intro to Speech Science</td>
<td>thirty-four</td>
<td>thirty-four</td>
</tr>
<tr>
<td>150</td>
<td>74a</td>
<td>Intro to Speech Science</td>
<td>thirty-four</td>
<td>thirty-four</td>
</tr>
<tr>
<td>150</td>
<td>75a</td>
<td>Intro to Speech Science</td>
<td>thirty-four</td>
<td>thirty-four</td>
</tr>
<tr>
<td>150</td>
<td>76a</td>
<td>Intro to Speech Science</td>
<td>thirty-four</td>
<td>thirty-four</td>
</tr>
<tr>
<td>150</td>
<td>77a</td>
<td>Intro to Speech Science</td>
<td>thirty-four</td>
<td>thirty-four</td>
</tr>
<tr>
<td>150</td>
<td>78a</td>
<td>Intro to Speech Science</td>
<td>thirty-four</td>
<td>thirty-four</td>
</tr>
<tr>
<td>150</td>
<td>79a</td>
<td>Intro to Speech Science</td>
<td>thirty-four</td>
<td>thirty-four</td>
</tr>
<tr>
<td>150</td>
<td>80a</td>
<td>Intro to Speech Science</td>
<td>thirty-four</td>
<td>thirty-four</td>
</tr>
<tr>
<td>150</td>
<td>81a</td>
<td>Intro to Speech Science</td>
<td>thirty-four</td>
<td>thirty-four</td>
</tr>
<tr>
<td>150</td>
<td>82a</td>
<td>Intro to Speech Science</td>
<td>thirty-four</td>
<td>thirty-four</td>
</tr>
<tr>
<td>150</td>
<td>83a</td>
<td>Intro to Speech Science</td>
<td>thirty-four</td>
<td>thirty-four</td>
</tr>
<tr>
<td>150</td>
<td>84a</td>
<td>Intro to Speech Science</td>
<td>thirty-four</td>
<td>thirty-four</td>
</tr>
<tr>
<td>150</td>
<td>85a</td>
<td>Intro to Speech Science</td>
<td>thirty-four</td>
<td>thirty-four</td>
</tr>
<tr>
<td>150</td>
<td>86a</td>
<td>Intro to Speech Science</td>
<td>thirty-four</td>
<td>thirty-four</td>
</tr>
<tr>
<td>150</td>
<td>87a</td>
<td>Intro to Speech Science</td>
<td>thirty-four</td>
<td>thirty-four</td>
</tr>
<tr>
<td>150</td>
<td>88a</td>
<td>Intro to Speech Science</td>
<td>thirty-four</td>
<td>thirty-four</td>
</tr>
<tr>
<td>150</td>
<td>89a</td>
<td>Intro to Speech Science</td>
<td>thirty-four</td>
<td>thirty-four</td>
</tr>
<tr>
<td>150</td>
<td>90a</td>
<td>Intro to Speech Science</td>
<td>thirty-four</td>
<td>thirty-four</td>
</tr>
<tr>
<td>150</td>
<td>91a</td>
<td>Intro to Speech Science</td>
<td>thirty-four</td>
<td>thirty-four</td>
</tr>
<tr>
<td>150</td>
<td>92a</td>
<td>Intro to Speech Science</td>
<td>thirty-four</td>
<td>thirty-four</td>
</tr>
<tr>
<td>150</td>
<td>93a</td>
<td>Intro to Speech Science</td>
<td>thirty-four</td>
<td>thirty-four</td>
</tr>
<tr>
<td>150</td>
<td>94a</td>
<td>Intro to Speech Science</td>
<td>thirty-four</td>
<td>thirty-four</td>
</tr>
<tr>
<td>150</td>
<td>95a</td>
<td>Intro to Speech Science</td>
<td>thirty-four</td>
<td>thirty-four</td>
</tr>
<tr>
<td>150</td>
<td>96a</td>
<td>Intro to Speech Science</td>
<td>thirty-four</td>
<td>thirty-four</td>
</tr>
<tr>
<td>150</td>
<td>97a</td>
<td>Intro to Speech Science</td>
<td>thirty-four</td>
<td>thirty-four</td>
</tr>
<tr>
<td>150</td>
<td>98a</td>
<td>Intro to Speech Science</td>
<td>thirty-four</td>
<td>thirty-four</td>
</tr>
<tr>
<td>150</td>
<td>99a</td>
<td>Intro to Speech Science</td>
<td>thirty-four</td>
<td>thirty-four</td>
</tr>
</tbody>
</table>

**w_scheduling** (with data from run)

Data created by Start Button
Script : open for schedule

sqlca.DBMS = ProfileString("schedule.ini","sqlca","dbms","")
sqlca.database = ProfileString("schedule.ini","sqlca","database","")
sqlca.userid = ProfileString("schedule.ini","sqlca","userid","")
sqlca.dbpass = ProfileString("schedule.ini","sqlca","dbpass","")
sqlca.logid = ProfileString("schedule.ini","sqlca","logid","")
sqlca.logpass = ProfileString("schedule.ini","sqlca","logpass","")
sqlca.servername = ProfileString("schedule.ini","sqlca","servername","")
sqlca.dbparm = ProfileString("schedule.ini","sqlca","dbparm","")

connect;
if sqlca.sqlcode <> 0 then
   MessageBox ("Sorry! Cannot Connect to Database "+ string(sqlca.sqlcode), sqlca.sqlerror)
   return
end if

open(w_class_main_test)
Window: w_class_main_test
Library: e:\thesis\appl\schedule.pbl
Date: 5/1/95   Time: 21:22:34

Window: w_class_main_test
X = 919   Y = 593   Width = 2844   Height = 1905
Visible = true   Enabled = true   TitleBar = true
ControlMenu = true   MinBox = true   MaxBox = true
Resizable = true
WindowType = main!   WindowState = normal!
BackColor = 12632256

Instance Variables

End of Instance Variables

Script for: open   event
//Paste the user object uo_class_main on the window to display the class information.

// Local Variables

// End Local Variables

OpenUserObject(uo_class_main, 10, 325)
gv_s_MainUoType = "class"

End of Script

StaticText: st_1
X = 270   Y = 21   Width = 298   Height = 69
TabOrder = 0   Visible = true   Text = "Information"
TextColor = 8388608   BackColor = 12632256   Alignment = left!
FillPattern = solid!
RadioButton: rb_professors
X = 1669    Y = 133    Width = 641    Height = 69
TabOrder = 0    Visible = true    Enabled = true    Text = "on Faculty Members"
Automatic = true    TextColor = 33554432    BackColor = 12632256
BorderStyle = stylelowered!

Script for: clicked event
//Pastes the user object uo_prof_main and displays faculty member information.

IF (gv_s_MainUoType = "class") THEN
    SetRedraw(w_class_main_test, FALSE)
    CloseUserObject(uo_class_main)
    OpenUserObject(uo_prof_main, 10, 325)
    gv_s_MainUoType = "prof"
    w_class_main_test.Title = "Faculty Member Information"
    SetRedraw(w_class_main_test, TRUE)
END IF

End of Script

RadioButton: rb_class
X = 330    Y = 133    Width = 458    Height = 69
TabOrder = 0    Visible = true    Enabled = true    Text = "on Classes"
Automatic = true    Checked = true    TextColor = 33554432
BackColor = 12632256
BorderStyle = stylelowered!

Script for: clicked event
//Pastes the user object uo_class_main to display the class information.

IF (gv_s_MainUoType = "prof") THEN
    SetRedraw(w_class_main_test, FALSE)
    CloseUserObject(uo_prof_main)
User Object: uo_class_main
Library: e:\thesis\appl\schedule.pbl
Date: 5/2/95    Time: 16:50:58

Instance Variables
DataWindowChild    iv_dwc_Term
DataWindowChild    iv_dwc_Department
DataWindowChild    iv_dwc_CourseSection

Boolean    iv_b_SelectionMade
Boolean    iv_b_TabDone
Window: uo_class_main
Library: e:\thesis\appl\schedule.plb
Date: 5/2/95    Time: 16:50:58

Boolean    iv_bUpDownArrowDone

End of Instance Variables

Script for: constructor event
//Retrieves all datawindow information from the database used by
//the user object.

// Local Variables

// End Local Variables

dwGetChild(dw_term, "term", iv_dwc_term)
dwGetChild(dw_department, "department", iv_dwc_department)
dwGetChild(dw_course_section, "course_num", iv_dwc_CourseSection)

SetTransObject(iv_dwc_term, SQLCA)
SetTransObject(iv_dwc_department, SQLCA)
SetTransObject(iv_dwc_CourseSection, SQLCA)
SetTransObject(dw_term, SQLCA)
SetTransObject(dw_department, SQLCA)
SetTransObject(dw_course_section, SQLCA)
SetTransObject(dw_class_xref, SQLCA)

Retrieve(dw_term)
Retrieve(dw_department)
Retrieve(dw_class_xref)

SetSort(dw_term, "term A")
SetSort(iv_dwc_term, "term A")
SetSort(dw_department, "department A")
SetSort(iv_dwc_department, "department A")

Sort(dw_term)
User Object: uo_class_main
Library: e:\thesis\appl\schedule.pbl
Date: 5/2/95   Time: 16:50:58

Sort(iv_dwc_term)
Sort(dw_department)
Sort(iv_dwc_department)

SetSort(dw_class_xref, "term A, department A, course_num A, section_num A")
Sort(dw_class_xref)

InsertRow(dw_term, 1)
InsertRow(dw_department, :)
InsertRow(dw_course_section, 1)

iv_b_SelectionMade = FALSE

SetFocus(dw_term)

End of Script

Script for: other  event
//Tries to control the tabbing that the user object seems to have
//difficulties with.

//Local Variables
CommandButton  lv_cb_which
DataWindow     lv_dw_which
GraphicObject  lv_go_WhichControl
String         lv_s_TextValue
String         lv_s_DataObject
//End Local Variables

IF (KeyDown(KeyTab!) AND KeyDown(KeyShift!)) THEN
  IF (iv_b_TabDone) THEN
    iv_b_TabDone = FALSE
    // not sure how to explain the color setting ... trust me ... it needs to be there for now.
    st_1.textcolor = RGB(0, 0, 128)
User Object: uo_class_main
Library: e:\thesis\appl\schedule.pbl
Date: 5/2/95     Time: 16:50:58

st_2.textcolor = RGB(0, 0, 128)
  IF (dw_course_section.enabled) THEN
    st_course_section_t.textcolor = RGB(0, 0, 128)
  ELSE
    st_course_section_t.textcolor = RGB(128, 128, 128)
  END IF
END IF

lv_go_WhichControl = GetFocus()

choose case TypeOf(lv_go_WhichControl)

case CommandButton!
  lv_cb_which = lv_go_WhichControl
  lv_s_TextValue = lv_cb_which.text
  IF (lv_s_TextValue = "&Scheduling ...") THEN
    IF (dw_course_section.enabled) THEN
      SetFocus(dw_course_section)
      Return
    ELSE
      SetFocus(dw_department)
      Return
    END IF
  ELSEIF (lv_s_TextValue = "&Add Term/Department ...") THEN
    IF (cb_scheduling.enabled) THEN
      SetFocus(cb_scheduling)
      Return
    ELSE
      SetFocus(dw_course_section)
      Return
    END IF
  ELSEIF (lv_s_TextValue = "&Reset") THEN
    SetFocus(cb_add)
    Return
  ELSEIF (lv_s_TextValue = "&Edit ...") THEN
User Object: uo_class_main
Library: e:\thesis\appl\schedule.pbl
Date: 5/2/95     Time: 16:50:58

IF (cb_reset.enabled) THEN
   SetFocus(cb_reset)
   Return
ELSE
   SetFocus(cb_add)
   Return
END IF
ELSEIF (lv_s_TextValue = "&OK") THEN
   IF (cb_edit.enabled) THEN
      SetFocus(cb_edit)
      Return
   ELSEIF (cb_reset.enabled) THEN
      SetFocus(cb_reset)
      Return
   ELSE
      SetFocus(cb_add)
      Return
   END IF
ELSE
   Return
END IF

case DataWindow!
   lv_dw_which = lv_go_WhichControl

   lv_s_DataObject = lv_dw_which.dataobject

   IF (lv_s_DataObject = "d_term_ddw") THEN
      SetFocus(cb_ok)
      Return
   ELSEIF (lv_s_DataObject = "d_department_ddw") THEN
      SetFocus(dw_term)
      Return
   ELSEIF (lv_s_DataObject = "d_course_section_ddw") THEN
      SetFocus(dw_department)
      Return
   END IF
ELSEIF (lv_s_Dataobject = "d_class_xref") THEN
  IF (dw_course_section.enabled) THEN
    SetFocus(dw_course_section)
    Return
  ELSE
    SetFocus(dw_department)
    Return
  END IF
ELSE
  Return
END IF

case else
  Return
END Choose

ELSEIF (KeyDown(KeyTab!!)) THEN
  IF (lv_b_TabDone) THEN
    lv_b_TabDone = FALSE
// not sure how to explain the color setting ... trust me ... it needs to be there for now.
    st_1.textcolor = RGB(0, 0, 128)
    st_2.textcolor = RGB(0, 0, 128)
    Return
  END IF

lv_go_WhichControl = GetFocus()

choose case TypeOf(lv_go_WhichControl)

  case CommandButton!
    lv_cb_which = lv_go_WhichControl
    lv_s_TextValue = lv_cb_which.text
    IF (lv_s_TextValue = "&Scheduling ...") THEN
      SetFocus(cb_add)
      Return
    ELSEIF (lv_s_TextValue = "&Add Term/Department ...") THEN
      IF (cb_reset.enabled) THEN

User Object:  uo_class_main
Library:  e:\thesis\appl\schedule.pbl
Date:  5/2/95   Time:  16:50:58

SetFocus(cb_reset)
   Return
ELSEIF (cb_edit.enabled) THEN
   SetFocus(cb_edit)
   Return
ELSE
   SetFocus(cb_ok)
   Return
END IF
ELSEIF (lv_s_TextValue = "&Reset") THEN
   IF (cb_edit.enabled) THEN
      SetFocus(cb_edit)
      Return
   ELSE
      SetFocus(cb_ok)
      Return
   END IF
ELSEIF (lv_s_TextValue = "&Edit ...") THEN
   SetFocus(cb_ok)
   Return
ELSEIF (lv_s_TextValue = "&OK") THEN
   SetFocus(dw_term)
   Return
ELSE
   Return
END IF

case DataWindow!
lv_dw_Which = lv_go_WhichControl

lv_s_DataObject = lv_dw_Which.dataobject

IF (lv_s_DataObject = "d_term_dddw") THEN
   SetFocus(dw_department)
   Return
ELSEIF (lv_s_DataObject = "d_department_dddw") THEN

IF (dw_course_section.enabled) THEN
   SetFocus(dw_course_section)
   Return
ELSEIF (cb_scheduling.enabled) THEN
   SetFocus(cb_scheduling)
   Return
ELSE
   SetFocus(cb_add)
   Return
END IF
ELSEIF (lv_s_DataObject = "d_course_section_dddw") THEN
   IF (cb_scheduling.enabled) THEN
      SetFocus(cb_scheduling)
      Return
   ELSE
      SetFocus(cb_add)
      Return
   END IF
ELSEIF (lv_s_DataObject = "d_class_xref") THEN
   IF (cb_scheduling.enabled) THEN
      SetFocus(cb_scheduling)
      Return
   ELSE
      SetFocus(cb_add)
      Return
   END IF
ELSE
   Return
END IF
END IF
Script for: getcontrolagain event
//Re-retrieves data into the data windows after changes have been made in
//uo_class_maint.

//Local Variables
String lv_s_FilterString

//End Local Variables

SetRedraw(dw_term, FALSE)
SetRedraw(dw_department, FALSE)
SetRedraw(dw_class_xref, FALSE)
SetRedraw(dw_course_section, FALSE)

Retrieve(dw_term)
Retrieve(dw_department)
Retrieve(dw_class_xref)
Retrieve(dw_course_section, gv_struct_parms.term, gv_struct_parms.department)

lv_s_FilterString = "term = " + \\
" + " + gv_struct_parms.term + \\
+ " and department = " + \\
+ " + " + gv_struct_parms.department + ""

SetFilter(iv_dwc_CourseSection, lv_s_FilterString)
Filter(iv_dwc_CourseSection)

SetSort(dw_term, "term A")
SetSort(iv_dwc_term, "term A")
SetSort(dw_department, "department A")
SetSort(iv_dwc_department, "department A")
SetSort(dw_course_section, "course_num A, section_num A")
SetSort(iv_dwc_CourseSection, "course_num A, section_num A")

Sort(dw_term)
Sort(iv_dwc_term)
Sort(dw_department)
User Object: uo_class_main
Library: e:\thesis\app\schedule.pbl
Date: 5/2/95     Time: 16:50:58

Sort(iv_dwc_department)
Sort(dw_course_section)
Sort(iv_dwc_CourseSection)

SetSort(dw_class_xref, "term A, department A, course_num A, section_num A")
Sort(dw_class_xref)

InsertRow(dw_term, 1)
InsertRow(dw_department, 1)
InsertRow(dw_course_section, 1)

ScrollToRow(dw_course_section, 1)
setItem(dw_course_section, 1, "section_num", "")

st_course_section_t.textcolor = RGB(128, 128, 128)
dw_course_section.enabled = FALSE

SelectRow(dw_class_xref, 0, FALSE)

SetRedraw(dw_term, TRUE)
SetRedraw(dw_department, TRUE)
SetRedraw(dw_course_section, TRUE)
SetRedraw(dw_class_xref, TRUE)

//Reset gv_struct_parms
gv_struct_parms.term = ""
gv_struct_parms.department = ""
gv_struct_parms.course_number = 0
gv_struct_parms.section_number = ""
gv_struct_parms.faculty_member = ""

cb_reset.enabled = FALSE
cb_scheduling.enabled = FALSE
cb_edit.enabled = FALSE

tv_b_SelectionMade = FALSE
User Object: uo_class_main
Library: e:\thesis\appl\schedule.pbl
Date: 5/2/95    Time: 16:50:58

SetFocus(dw_term)

End of Script

StaticText: st_course_section_t
X = 1697  Y = 9  Width = 714  Height = 73
TabOrder = 0  Visible = true  Text = "Course Number/Section Number"
TextColor = 8421504  BackColor = 12632256  Alignment = left!
FillPattern = solid!

DataWindow: dw_course_section
X = 1697  Y = 85  Width = 1025  Height = 81
TabOrder = 20  Visible = true  DataObject = "d_course_section_dddw"
LiveScroll = true  BorderStyle = stylebox!

Script for: other event
//Used to catch the WM_PAINT event in order to catch a change in the drop down datawindow

//Local Variables
Integer lv_i_ClickedRow
Integer lv_i_Row
Integer lv_i_CourseNumber
String lv_s_SectionNumber
String lv_s_Term
String lv_s_Department
String lv_s_FindString
//End Local Variables

IF (KeyDown(keyDownArrow!)) THEN
   TriggerEvent(dw_course_section, "controldisplay")
   lv_b_UpDownArrowDone = TRUE
ELSEIF (KeyDown(keyUpArrow!)) THEN
Window: uo_class_main
Library: e:\thesis\appl\schedule.pbl
Date: 5/2/95     Time: 16:50:58

    TriggerEvent(dw_course_section, "controldisplay")
    iv_b_UpDownArrowDone = TRUE
END IF

//IF Message.Number = 15 THEN windows has triggered WM_PAINT. I'm using this method to catch when a non-
//course_number is selected but it is a unique course/section number.

IF (Message.Number = 15) THEN
    TriggerEvent(dw_course_section, "dwclosedropdown")
END IF

End of Script

Script for: dwnkey  event
//DropDownDataWindows do strange things on tab ...

IF (KeyDown(keyTab!) AND KeyDown(keyShift!)) THEN
    SetFocus(dw_department)
    SetActionCode(dw_course_section, 1)
    // not sure how to explain this ... but right now it needs to be there for tabbing control
    st_course_section_t_textcolor = RGB(255, 0, 0)
    iv_b_TabDone    = TRUE
    Return
ELSEIF (KeyDown(keyTab!)) THEN
    SetFocus(cb_scheduling)
    SetActionCode(dw_course_section, 1)
    // not sure how to explain this ... but right now it needs to be there for tabbing control
    st_course_section_t_textcolor = RGB(0, 255, 0)
    iv_b_TabDone    = TRUE
    Return
END IF
Window: wo_class_main
Library: e:\thesis\appl\schedule.pbl
Date: 5/2/95   Time: 16:50:58

End of Script

Script for: dwclosedropdown event
//When a course/section selection is changed, triggers the controldisplay event which
//selects the row in class_xref data window.

//Local Variables
Integer lv_i_ClickedRow
//End Local Variables

//Currently, this event is triggered from the "other" event. When and if this event
//occurs the dropdown closes, this script will be executed and the TriggerEvent in
//the "other" event could be removed

lv_i_ClickedRow = GetSelectedRow(lv_dwc_CourseSection, 0)

IF (lv_i_ClickedRow > 0) THEN
  TriggerEvent(dw_course_section, "controldisplay")
END IF

End of Script

Script for: controldisplay event
//When an item is selected, it highlights the row in the class_xref data window

//Local Variables
Integer lv_i_Row
Integer lv_i_SelectedRow
Integer lv_i_CourseNumber
String lv_s_Term
String lv_s_Department
lv_i_SelectedRow = GetSelectedRow(lv_dwc_CourseSection, 0)

lv_s_Term = gv_struct_parms.term
lv_s_Department = gv_struct_parms.department
lv_i_CourseNumber = GetItemNumber(lv_dwc_CourseSection, lv_i_SelectedRow, "course_num")
lv_s_SectionNumber = Trim(GetItemString(lv_dwc_CourseSection, lv_i_SelectedRow, "section_num"))

SetItem(dw_course_section, 1, "section_num", lv_s_SectionNumber)

lv_s_FindString = "term = " + "+" + lv_s_Term + "+" &
+ "and department = " + "+" + lv_s_Department + "+" &
+ "and course_num = " + string(lv_i_CourseNumber) &
+ "and section_num = " + "+" + lv_s_SectionNumber + "+"

lv_i_Row = dwFind(dw_class_xref, lv_s_FindString, 0, RowCount(dw_class_xref))

IF (lv_i_Row > 0) THEN
  SelectRow(dw_class_xref, 0, FALSE)
  SelectRow(dw_class_xref, lv_i_Row, TRUE)
  ScrollToRow(dw_class_xref, lv_i_Row)
END IF

End of Script
DataWindow: dw_department
X = 1006  Y = 85  Width = 494  Height = 93
TabOrder = 10  Visible = true  Enabled = true  DataObject = "d_department_dddw"
LiveScroll = true  BorderStyle = stylebox!

Script for: itemchanged event
//Sets structure variables passed to other windows and user objects.
//Once a term and a department are selected, it fills the course/section
//drop down datawindow with information.

//Local Variables
Integer lv_i_Row
Integer lv_i_RowFound
String lv_s_Term
String lv_s_FilterString
String lv_s_FindString
String lv_s_Department
//End Local Variables

lv_s_Term = gv_struct_parms.term

lv_i_Row = GetSelectedRow(iv_dwc_department, 0)

IF (lv_i_Row < 1) THEN
    Return
END IF

lv_s_Department = Trim(GetItemString(iv_dwc_department, lv_i_Row, "department"))
gv_struct_parms.department = lv_s_Department

cb_reset.enabled = TRUE

//Does the find string need to account for the fact that there is a term selected.
IF (NOT (IsNull(lv_s_Term) OR (lv_s_Term = ""))) THEN
SetRedraw(dw_course_section, FALSE)
SetRedraw(st_course_section_t, FALSE)

SelectRow(dw_class_xref, 0, FALSE)

Retrieve(dw_course_section, lv_s_Term, lv_s_Department)

lv_s_FilterString = "term = " + '"' + lv_s_Term + '"' &
                     + " and department = " + '"' + lv_s_Department + '"
SetFilter(iv_dwc_CourseSection, lv_s_FilterString)
Filter(iv_dwc_CourseSection)

SetSort(dw_course_section, "course_num A, section_num A")
SetSort(iv_dwc_CourseSection, "course_num A, section_num A")
Sort(dw_course_section)
Sort(iv_dwc_CourseSection)

lv_s_FindString = "term = " + '"' + lv_s_Term + '"' &
                     + " and department = " + '"' + lv_s_Department + '"

lv_i_RowFound = dwFind(dw_class_xref, lv_s_FindString, 0, RowCount(dw_class_xref))

IF (lv_i_RowFound > 0) THEN
  SetRedraw(dw_class_xref, FALSE)
  SelectRow(dw_class_xref, lv_i_RowFound, TRUE)
  ScrollToRow(dw_class_xref, lv_i_RowFound)
  SetRedraw(dw_class_xref, TRUE)
  lv_b_SelectionMade = TRUE
  st_course_section_t.textcolor = RGB(0, 0, 128)
  dw_course_section.enabled = TRUE
ELSE
  InsertRow(dw_course_section, 0)
  st_course_section_t.textcolor = RGB(128, 128, 128)
  dw_course_section.enabled = FALSE
ENDIF
SetRedraw(dw_course_section, TRUE)
SetRedraw(st_course_section_t, TRUE)

cb_scheduling.enabled = TRUE
cb_edit.enabled = TRUE
END IF

End of Script

Script for: updateend event
//Called after a new department is added in the add_term_dept window.

//Local Variables
Integer lv_i_InsertedRow
String lv_s_Term
String lv_s_Department
String lv_s_FindString
//End Local Variables

SetRedraw(dw_class_xref, FALSE)
SetRedraw(dw_term, FALSE)
SetRedraw(dw_department, FALSE)

SelectRow(dw_class_xref, 0, FALSE)

lv_s_Department = gv_structParms.department
lv_s_FindString = "department = " + "+" + lv_s_Department + "+"

IF (dwFind(iv_dwc_department, lv_s_FindString, 1, RowCount(iv_dwc_department)) = 0) THEN
  lv_i_InsertedRow = InsertRow(iv_dwc_department, 0)
  SetItem(iv_dwc_department, lv_i_InsertedRow, "department", lv_s_Department)
Window: uo_class_main
Library: e:\thesis\appl\schedule.pbl
Date: 5/2/95   Time: 16:50:58

SetSort(dw_department, "department A")
SetSort(lv_dwc_department, "department A")

Sort(dw_department)
Sort(lv_dwc_department)
END IF

InsertRow(dw_department, 1)

SetItem(dw_department, 1, "department", lv_s_Department)
ScrollToRow(dw_department, 1)

lv_s_Term = Trim(GetItemString(dw_term, 1, "term"))

IF ((IsNull(lv_s_Term)) OR (lv_s_Term = "")) THEN
   cb_edit.enabled = FALSE
   cb_scheduling.enabled = FALSE
ELSE
   cb_scheduling.enabled = TRUE
   cb_edit.enabled = TRUE
   cb_edit.default = TRUE
   cb_scheduling.default = FALSE
   cb_add.default = FALSE
   cb_reset.default = FALSE
   cb_ok.default = FALSE
END IF

SetRedraw(dw_class_xref, TRUE)
SetRedraw(dw_term, TRUE)
SetRedraw(dw_department, TRUE)

End of Script

Script for: dwkey event
/* In some doxh data windows do strange things on tabs so the following overrides tab and sets focus where it
IF (KeyDown(keyTab!) AND KeyDown(keyShift!)) THEN
    SetFocus(dw_term)
    SetActionCode(dw_department, 1)
    lv_b_TabDone = TRUE
    // Don't know how to explain how this helps with tabbing but it does.
    st_2.textcolor = RGB(255, 0, 0)
    Return
ELSEIF (KeyDown(keyTab!)) THEN
    IF (dw_course_section.enabled) THEN
        SetFocus(dw_course_section)
    ELSE
        SetFocus(cb_scheduling)
    END IF
    SetActionCode(dw_department, 1)
    lv_b_TabDone = TRUE
    // Don't know how to explain how this helps with tabbing but it does.
    st_2.textcolor = RGB(0, 255, 0)
    Return
END IF

End of Script

CommandButton: cb_reset
X = 1500  Y = 1325  Width = 289  Height = 109
TabOrder = 0  Visible = true  Text = "&Reset"

Script for: clicked  event
//Resets all information to the way it looked when the user object opened.

//Local Variables
Integer  lv_i_Row
//End Local Variables

SetRedraw(dw_class_xref, FALSE)
SetRedraw(dw_term, FALSE)
SetRedraw(dw_department, FALSE)
SetRedraw(dw_course_section, FALSE)
SelectRow(dw_class_xref, 0, FALSE)
ScrollToRow(dw_class_xref, 1)

lv_i_Row = dwFind(dw_term, "term = " + '"' + '"', 0, RowCount(dw_term))
IF (lv_i_Row = 0) THEN
  InsertRow(dw_term, 1)
  ScrollToRow(dw_term, 1)
ELSE
  ScrollToRow(dw_term, lv_i_Row)
END IF

lv_i_Row = dwFind(dw_department, "department = " + '"' + '"', 0, RowCount(dw_department))
IF (lv_i_Row = 0) THEN
  InsertRow(dw_department, 1)
  ScrollToRow(dw_department, 1)
ELSE
  ScrollToRow(dw_department, lv_i_Row)
END IF

InsertRow(dw_course_section, 1)
ScrollToRow(dw_course_section, 1)
dw_course_section.enabled = FALSE
st_course_section.t.textcolor = RGB(128, 128, 128)

SetFocus(dw_term)

SetRedraw(dw_class_xref, TRUE)
SetRedraw(dw_term, TRUE)
SetRedraw(dw_department, TRUE)
SetRedraw(dw_course_section, TRUE)

gv_struct_parms.term = ""
gv_struct_parms.department = ""

//Command Button Control
Window: uo_class_main
Library: e:\thesis\appl\schedule.pbl
Date: 5/2/95    Time: 16:50:58

cb_reset.enabled = FALSE
cb_scheduling.enabled = FALSE
cb_edit.enabled = FALSE

End of Script

CommandButton: cb_ok
X = 2350    Y = 1325    Width = 289    Height = 109
TabOrder = 0    Visible = true    Enabled = true    Text = "&OK"

Script for: clicked  event
//Allows the user to exit the program.

//Local Variables

//End Local Variables

close (w_class_main_test)

End of Script

CommandButton: cb_edit
X = 1925    Y = 1325    Width = 289    Height = 109
TabOrder = 0    Visible = true    Text = "&Edit ..."

Script for: clicked  event
//Opens user object used for class maintenance.

//Local Variables
Integer    lv_i_SelectedRow
s_main_win_parms    s_parms
Window: uo_class_main
Library: e:\thesis\appl\schedule.pbl
Date: 5/2/95  Time: 16:50:58

//End Local Variables

lv_i_SelectedRow = GetSelectedRow(dw_class_xref, 0)

s_parms.Department = gv_struct_parms.department
s_parms.Term = gv_struct_parms.term

IF (lv_i_SelectedRow > 1) THEN
  s_parms.Course_Number = GetItemNumber(dw_class_xref, lv_i_SelectedRow, "course_num")
  s_parms.Section_Number = Trim(GetItemString(dw_class_xref, lv_i_SelectedRow, "section_num"))
END IF

OpenWithParm(w_maint_main, s_parms)

End of Script

CommandButton: cb_add
X = 750  Y = 1325  Width = 613  Height = 109
TabOrder = 0  Visible = true  Enabled = true  Text = "&Add Term/Department ..."

Script for: clicked  event
//Opens add term department window, which is used to enter in a new term and/or department.
//Could be eliminated once the drop down datawindows become editable.

//Local Variables

//End Local Variables

Open(w_add_term_dept)

End of Script
CommandButton: cb_scheduling
X = 142   Y = 1325   Width = 471   Height = 109
TabOrder = 0   Visible = true   Text = "&Scheduling ..."

Script for: clicked event
//Opens the w_scheduling window which is used to create the actual schedule
OpenWithParm(w_scheduling, gv_struct_parms)

End of Script

DataWindow: dw_class_xref
X = 42   Y = 265   Width = 2725   Height = 989
TabOrder = 0   Visible = true   Enabled = true   DataObject = "d_class_xref"
VScrollBar = true   Border = true   LiveScroll = true   BorderStyle = stylebox

Script for: clicked event
//When a row is clicked, gets information from that row and sets them into global variables
//passed to other user objects and windows. If the row was already selected, then the row
//is deselected.

//Local Variables
Integer lv_i_SelectedRow
Integer lv_i_ClickedRow
Integer lv_i_FindRow
Integer lv_i_CourseNumber
String lv_s_Term
String lv_s_Department
String lv_s_SectionNumber
String lv_s_FindTerm
String lv_s_FindDepartment
String lv_s_FindCourseSectionNum
Window: uo_class_main
Library: e:\thesis\appl\schedule.pbl
Date: 5/2/95 Time: 16:50:58

String lv_s_FilterString
//End Local Variables

lv_i_SelectedRow = GetSelectedRow(dw_class_xref, 0)
lv_i_ClickedRow = GetClickedRow(dw_class_xref)

SetRedraw(dw_class_xref, FALSE)

IF (lv_i_SelectedRow = lv_i_ClickedRow) THEN
  SelectRow(dw_class_xref, lv_i_ClickedRow, FALSE)
  gv_struct_parms.term = ""
  gv_struct_parms.department = ""
  cb_reset.enabled = FALSE
  cb_scheduling.enabled = FALSE
  cb_edit.enabled = FALSE
ELSE
  SelectRow(dw_class_xref, lv_i_SelectedRow, FALSE)
  SelectRow(dw_class_xref, lv_i_ClickedRow, TRUE)
lv_s_Department = Trim(GetItemString(dw_class_xref, lv_i_ClickedRow, "department"))
lv_s_Term = Trim(GetItemString(dw_class_xref, lv_i_ClickedRow, "term"))
lv_i_CourseNumber = GetItemNumber(dw_class_xref, lv_i_ClickedRow, "course_num")
lv_s_SectionNumber = Trim(GetItemString(dw_class_xref, lv_i_ClickedRow, "section_num"))

  gv_struct_parms.term = lv_s_Term
  gv_struct_parms.department = lv_s_Department
  gv_struct_parms.course_number = lv_i_CourseNumber
  gv_struct_parms.section_number = lv_s_SectionNumber

  lv_s_FindTerm = "term = " + '"' + lv_s_Term + '"'
lv_i_FindRow = dwFind(dw_term, lv_s_FindTerm, 0, RowCount(dw_term))
  IF (lv_i_FindRow > 0) THEN
    ScrollToRow(dw_term, lv_i_FindRow)
  END IF

  lv_s_FindDepartment = "department = " + '"' + lv_s_Department + '"'
lv_i_FindRow = dwFind(dw_department, lv_s_FindDepartment, 0, RowCount(dw_department))
IF (lv_i_FindRow > 0) THEN
  ScrollToRow(dw_department, lv_i_FindRow)
END IF

st_course_section_t.textColor = RGB(0, 0, 128)
dw_course_section.enabled = TRUE
Retrieve(dw_course_section, lv_s_Term, lv_s_Department)

lv_s_FilterString = "term = " + "'" + lv_s_Term + "'" &
  "and department = " + "'" + lv_s_Department + "'
SetFilter(lv_dwc_CourseSection, lv_s_FilterString)
Filter(lv_dwc_CourseSection)
SetSort(dw_course_section, "course_num A, section_num A")
SetSort(lv_dwc_CourseSection, "course_num A, section_num A")
Sort(dw_course_section)
Sort(lv_dwc_CourseSection)

lv_s_FindCourseSectionNum = "course_num = " + string(lv_i_CourseNumber) &
  "and section_num = " + "'" + lv_s_SectionNumber + "'
lv_i_FindRow = dwFind(dw_course_section, lv_s_FindCourseSectionNum, 0, RowCount(dw_course_section))

IF (lv_i_FindRow > 0) THEN
  ScrollToRow(dw_course_section, lv_i_FindRow)
END IF

cb_reset.enabled = TRUE
cb_scheduling.enabled = TRUE
cb_edit.enabled = TRUE

SetFocus(cb_edit)
END IF

SetRedraw(dw_class_xref, 'TRUE)
lv_b_SelectionMade = TRUE

End of Script

Script for: doubleclicked event
//Gets information for the doubleclicked row and sets them into global variables.
//Triggers event "clicked" of command button edit (which opens class maintenance datawindow)

//Local Variables
Integer lv_i_Row
//End Local Variables

lv_i_Row = GetSelectedRow(dw_class_xref, 0)

IF (lv_i_Row > 0) THEN
    gv_struct_Parms.term = Trim(GetItemString(dw_class_xref, lv_i_Row, "term"))
    gv_struct_Parms.department = Trim(GetItemString(dw_class_xref, lv_i_Row, "department"))
    gv_struct_Parms.course_number = GetItemNumber(dw_class_xref, lv_i_Row, "course_num")
    gv_struct_Parms.section_number = Trim(GetItemString(dw_class_xref, lv_i_Row, "section_num"))
    cb_edit.enabled = TRUE
    cb_scheduling.enabled = TRUE

    TriggerEvent(cb_edit, "clicked")
END IF

End of Script
DataWindow: dw_term
X = 275   Y = 85   Width = 449   Height = 101
TabOrder = 0   Visible = true   Enabled = true   DataObject = "d_term_dddw"
LiveScroll = true   BorderStyle = stylebox!

Script for: editchanged event
//This script will only work if I can get the dddd editable
//As the user types in information, datawindow scrolls to the closest item being typed.

//Local Variables
Integer   lv_i_Row
Integer   lv_i_NumRows
String    lv_s_Term
String    lv_s_Department
String    lv_s_FindString
//End Local Variables

lv_s_Term    = Trim(GetText(dw_term))
lv_s_Department   = gv_struct_Parms.department

IF (IsNull(lv_s_Department) OR (lv_s_Department = "")) THEN
  lv_s_FindString    = "term = " + "+" + lv_s_Term + "+"
  lv_i_NumRows   = RowCount(dw_class_xref)
ENDIF

lv_i_Row = dwFind(dw_class_xref, lv_s_FindString, 0, lv_i_NumRows)

IF (lv_i_Row > 0) THEN
  SelectRow(dw_class_xref, 0, FALSE)
  SelectRow(dw_class_xref, lv_i_Row, TRUE)
  ScrollToRow(dw_class_xref, lv_i_Row)
ELSEIF (lv_i_Row = 0) THEN
  //No match found, will highlight row after which term would be displayed
  lv_s_FindString    = "term > " + "+" + lv_s_Term + "+"
ENDIF
lv_i_Row = dwFind(dw_class_xref, lv_s_FindString, 0, lv_i_NumRows)

IF (lv_i_Row > 0) THEN
    SelectRow(dw_class_xref, 0, FALSE)
    SelectRow(dw_class_xref, lv_i_Row, TRUE)
    ScrollToRow(dw_class_xref, lv_i_Row)
ELSEIF (lv_i_Row = 0) THEN
    //Select last row
    SelectRow(dw_class_xref, 0, FALSE)
    SelectRow(dw_class_xref, lv_i_NumRows, TRUE)
    ScrollToRow(dw_class_xref, lv_i_NumRows)
END IF

ELSE
    lv_s_FindString = "term = " + '"' + lv_s_Term + '"' &
    + "and department = " + '"' + lv_s_Department + '"'
    lv_i_NumRows = rowCount(dw_class_xref)

lv_i_Row = dwFind(dw_class_xref, lv_s_FindString, 0, lv_i_NumRows)

IF (lv_i_Row > 0) THEN
    SelectRow(dw_class_xref, 0, FALSE)
    SelectRow(dw_class_xref, lv_i_Row, TRUE)
    ScrollToRow(dw_class_xref, lv_i_Row)
ELSEIF (lv_i_Row = 0) THEN
    //No match found, will highlight row after which term would be displayed
    lv_s_FindString = "term > " + '"' + lv_s_Term + '"' &
    + "and department = " + '"' + lv_s_Department + '"'

lv_i_Row = dwFind(dw_class_xref, lv_s_FindString, 0, lv_i_NumRows)

IF (lv_i_Row > 0) THEN
    SelectRow(dw_class_xref, 0, FALSE)
    SelectRow(dw_class_xref, lv_i_Row, TRUE)
    ScrollToRow(dw_class_xref, lv_i_Row)
ELSEIF (lv_i_Row = 0) THEN
Window: uo_class_main
Library: e:\thesis\appl\schedule.pbl
Date: 5/2/95   Time: 16:50:58

//Select last row
SelectRow(dw_class_xref, 0, FALSE)
SelectRow(dw_class_xref, lv_i_NumRows, TRUE)
ScrollToRow(dw_class_xref, lv_i_NumRows)
END IF
END IF
END IF
End of Script

Script for: itemchanged  event
//Sets structure variables passed to other windows and user objects.
//Once a term and a department are selected, it fills the course/section
//drop down datawindow with information.

//Local Variables
Integer   lv_i_Row
Integer   lv_i_RowFound
String    lv_s_Term
String    lv_s_FilterString
String    lv_s_Department
//End Local Variables
lv_s_Department = gv_struct_parms.department

lv_i_Row = GetSelectedRow(iv_dwc_term, 0)

IF (lv_i_Row < 1) THEN
   Return
END IF

lv_s_Term = Trim(GetItemString(iv_dwc_term, lv_i_Row, "term"))
gv_struct parms.term = lv_s_Term
cb_reset.enabled = TRUE

// Does the filter string need to account for the fact that there is a department selected.
IF (NOT (IsNull(lv_s_Department) OR (lv_s_Department = ""))) THEN
    SelectRow(dw_class_xref, 0, FALSE)

    lv_s_FilterString = ":term = " + "'" + lv_s_Term + "'" &
     + "and department = " + "'" + lv_s_Department + "'"

    lv_i_RowFound = dwFind(dw_class_xref, lv_s_FilterString, 0, RowCount(dw_class_xref))

IF (lv_i_RowFound > 0) THEN
    SetRedraw(dw_class_xref, FALSE)
    SelectRow(dw_class_xref, lv_i_RowFound, TRUE)
    ScrollToRow(dw_class_xref, lv_i_RowFound)
    SetRedraw(dw_class_xref, TRUE)
    lv_b_SelectionMade = TRUE

    st_course_section_t.textcolor = RGB(0, 0, 128)
    dw_course_section.enabled = TRUE

    Retrieve(dw_course_section, lv_s_Term, lv_s_Department)

    lv_s_FilterString = ":term = " + "'" + lv_s_Term + "'" &
     + "and department = " + "'" + lv_s_Department + "'"

    SetFilter(iv_dwc_CourseSection, lv_s_FilterString)
    Filter(iv_dwc_CourseSection)

    SetSort(dw_course_section, "course_num A, section_num A")
    SetSort(lv_dwc_CourseSection, "course_num A, section_num A")
    Sort(dw_course_section)
    Sort(lv_dwc_CourseSection)

END IF

cb_scheduling.enabled = TRUE
Window: uo_class_main
Library: e:\thesis\appl\schedule.pbl
Date: 5/2/95    Time: 16:50:58

    cb_edit.enabled = TRUE
END IF

End of Script

Script for: updateend event
//Called after a new term is added in the add_term_dept window.

//Local Variables
Integer  lv_i_InsertedRow
String   lv_s_Term
String   lv_s_Department
String   lv_s_FindString
//End Local Variables

SetRedraw(dw_class_xref, FALSE)
SetRedraw(dw_term, FALSE)
SetRedraw(dw_department, FALSE)

SelectRow(dw_class_xref, 0, FALSE)

lv_s_Term = gv_struct_parms.term

lv_s_FindString = "term = " + '"' + lv_s_Term + '"

IF (dwFind(iv_dwc_term, lv_s_FindString, 1, RowCount(iv_dwc_term)) = 0) THEN
    lv_i_InsertedRow = InsertRow(iv_dwc_term, 0)
    SetItem(iv_dwc_term, lv_i_InsertedRow, "term", lv_s_Term)
    SetSort(dw_term, "term A")
    Sort(iv_dwc_term)
    Sort(dw_term)
    Sort(iv_dwc_term)

ELSE
    dwInsertValue(dw_dwc_term, lv_s_Term)
    lv_i_InsertedRow = InsertRow(iv_dwc_term, 0)
    SetItem(iv_dwc_term, lv_i_InsertedRow, "term", lv_s_Term)
    SetSort(dw_term, "term A")
    Sort(iv_dwc_term)
    Sort(dw_term)
    Sort(iv_dwc_term)
ENDIF
Window: uo_class_main
Library: e:\thesis\appl\schedule.pbl
Date: 5/2/95 Time: 16:50:58

END IF

InsertRow(dw_term, 1)

SetItem(dw_term, 1, "term", lv_s_Term)
ScrollToRow(dw_term, 1)

dl_s_Department = Trim(GetItemString(dw_department, 1, "department"))

IF '((IsNull(dl_s_Department)) OR (dl_s_Department = "")) THEN
  cb_scheduling.enabled = FALSE
  cb_edit.enabled = FALSE
ELSE
  cb_scheduling.enabled = TRUE
  cb_edit.enabled = TRUE
  cb_edit.default = TRUE
  cb_scheduling.default = FALSE
  cb_add.default = FALSE
  cb_reset.default = FALSE
  cb_ok.default = FALSE
END IF

SetRedraw(dw_class_xref, TRUE)
SetRedraw(dw_term, TRUE)
SetRedraw(dw_department, TRUE)

End of Script

Script for: dwnkey event
//drop down datawindows do strange things on tabs so the following overrides tab and sets focus where it
would be.

IF (KeyDown(KeyTab!) AND KeyDown(KeyShift!)) THEN
  SetFocus(cb_ok)
Window: uo_class_main
Library: e:\thesis\appl\schedule.pbl
Date: 5/2/95    Time: 16:50:58

    SetActionCode(dw_term, 1)
    // Used to control tabbing ... not sure why it works
    st_1.textcolor = RGB(255, 0, 0)
    iv_b_TabDone = TRUE
    Return
    ELSEIF (KeyDown(KeyTab!)) THEN
        SetFocus(dw_department)
        SetActionCode(dw_term, 1)
        // Used to control tabbing ... not sure why it works
        st_1.textcolor = RGB(0, 255, 0)
        iv_b_TabDone = TRUE
        Return
    END IF

End of Script

StaticText: st_2
X = 1006    Y = 13    Width = 330    Height = 69
TabOrder = 0    Visible = true    Text = "Department"    TextColor = 8388608
BackColor = 12632256    Alignment = left!    FillPattern = solid!

StaticText: st_1
X = 275    Y = 13    Width = 243    Height = 69
TabOrder = 0    Visible = true    Text = "Term"    TextColor = 8388608
BackColor = 12632256    Alignment = left!    FillPattern = solid!
Retrieve: PBSELECT(TABLE(NAME="term_xref") COLUMN(NAME="term_xref.term"))

Arguments: None
Update Table: term_xref
Filter: None
Sort: None
Sparse: term
Column: term
  Updateable: Yes
  Key: Yes
  Format: "[general]"
  Border style: None
  Validation: None
  Validation Message: None
  Tab Sequence: 10
  Initial Value: None
  Edit Style: DropDownDataWindow
    Name: d_term_list
    Data Column: term
    Display Column: term
DataWindow: d_term_list
Library: e:\thesis\appl\schedule.pbl
Date: 5/2/95   Time: 17:13:19

Retrieve: PBSELECT(TABLE(NAME="term_xref") COLUMN(NAME="term_xref.term"))
Arguments: None
Update Table: term_xref
Filter: None
Sort: None
Sparse: None
Column: term
  Updateable: Yes
  Key: Yes
  Format: "[general]"
  Border style: None
  Validation: None
  Validation Message: None
  Tab Sequence: 10
  Initial Value: None
  Edit Style: Edit
  Edit limit: 10
DataWindow: d_department_ddw
Library: e:\thesis\appl\schedule.pbl
Date: 5/2/95 Time: 17:43:23

Retrieve: PBSELECT(TABLE(NAME="department_xref") COLUMN(NAME="department_xref.department"))
Arguments: None
Update Table: department_xref
Filter: None
Sort: None
Sparse: None
Column: department
  Updateable: Yes
  Key: Yes
  Format: "[general]"
  Border style: None
  Validation: None
  Validation Message: None
  Tab Sequence: 10
  Initial Value: None
  Edit Style: DropDownDataWindow
    Name: d_department_list
    Data Column: department
    Display Column: &department
DataWindow: d_department_list
Library: e:\thesis\appl\schedule.pbl
Date: 5/2/95    Time: 17:43:06
DataWindow: d_department_list
Library: e:\thesis\appl\schedule.pbl
Date: 5/2/95   Time: 17:43:06

Retrieve: PBSELECT(TABLE(NAME="department_xref") COLUMN(NAME="department_xref.department"))
Arguments: None
Update Table: department_xref
Filter: None
Sort: None
Sparse: None
Column: department
  Updateable: Yes
  Key: Yes
  Format: "[general]"
  Border style: None
  Validation: None
  Validation Message: None
  Tab Sequence: 10
  Initial Value: None
  Edit Style: Edit
  Edit limit: 4
Retrieve: PBSELECT(TABLE(NAME="class_information") COLUMN(NAME="class_information.course_num") COLUMN(NAME="class_information.section_num") COLUMN(NAME="class_information.term") COLUMN(NAME="class_information.department") WHERE( EXP1 ="-"class_information-."-term-" OP ="==" EXP2 =":arg_s_term" LOGIC ="and" ) WHERE( EXP1 ="-"class_information-."department-" OP ="==" EXP2 =":arg_s_department" ) ) ARG(NAME = "arg_s_term" TYPE = string) ARG(NAME = "arg_s_department" TYPE = string)
Arguments: arg_s_term arg_s_department
Update Table: class_information
Filter: None
Sort: None
Sparse: None
Column: course_num
  Updateable: No
  Key: Yes
  Format: "[general]"
  Border style: None
  Validation: None
  Validation Message: None
  Tab Sequence: 10
  Initial Value: None
Edit Style: DropDownDataWindow
  Name: d_course_section_list
  Data Column: course_num
  Display Column: course_num
Column: section_num
Update Table: class_information
Filter: None
Sort: None
Sparse: None
Column: section_num
  Updateable: No
  Key: Yes
  Format: "[general]"
  Border style: None
  Validation: None
  Validation Message: None
  Tab Sequence: 0
  Initial Value: None
  Edit Style: Edit
  Edit limit: 3
Column: course_num
  Updateable: No
  Key: Yes
  Format: "[general]"
  Border style: None
  Validation: None
  Validation Message: None
<table>
<thead>
<tr>
<th>Term</th>
<th>Department</th>
<th>Class Name</th>
<th>Course Number</th>
<th>Section Number</th>
</tr>
</thead>
<tbody>
<tr>
<td>Header</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Detail</td>
<td>Dept#</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Summary</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Footer</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Retrieve: PBSSELECT(TABLE(NAME="class_information") COLUMN(NAME="class_information.term")
COLUMN(NAME="class_information.department") COLUMN(NAME="class_information.name")
COLUMN(NAME="class_information.course_num") COLUMN(NAME="class_information.section_num"))
Arguments: None
Update Table: class_information
Filter: None
Sort: None
Sparse: None
Column: term
    Updateable: Yes
    Key: Yes
    Format: "[general]"
    Border style: None
    Validation: None
    Validation Message: None
    Tab Sequence: 0
    Initial Value: None
    Edit Style: Edit
    Edit limit: 10
Column: department
    Updateable: Yes
    Key: Yes
    Format: "[general]"
    Border style: None
    Validation: None
DataWindow: d_class_xref
Library: e:\thesis\appl\schedule.pl1
Date: 5/2/95    Time: 17:46:32

Tab Sequence: 0
Initial Value: None
Edit Style: Edit
Edit limit: 4

Column: name
  Updateable: No
  Key: No
  Format: "[general]"
 -border style: None
  Validation: None
  Validation Message: None
  Tab Sequence: 0
  Initial Value: None
  Edit Style: Edit
  Edit limit: 0

Column: course_num
  Updateable: No
  Key: Yes
  Format: "[general]"
  Border style: None
  Validation: None
  Validation Message: None
  Tab Sequence: 0
  Initial Value: None
  Edit Style: Edit
DataWindow: d_class_xref
Library: e:\thesis\appl\schedule.pbl
Date: 5/2/95    Time: 17:46:32

Column: section_num
  Updateable: No
  Key: Yes
  Format: "[general]"
  Border style: None
  Validation: None
  Validation Message: None
  Tab Sequence: 0
  Initial Value: None
  Edit Style: Edit
  Edit limit: 0
<table>
<thead>
<tr>
<th>Header</th>
<th>course_number</th>
<th>sect</th>
<th>name</th>
<th>days</th>
<th>start_time</th>
<th>end_time</th>
</tr>
</thead>
<tbody>
<tr>
<td>Detail</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Summary</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Footer</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
DataWindow: d_class_info_update
Library: e:\thesis\appl\schedule.pbl
Date: 5/2/95    Time: 17:46:49

Retrieve: Script
Arguments: None
Update Table: Not Allowed
Filter: None
Sort: None
Sparse: None
Column: days
    Format: "[general]"
    Border style: Shadow Box
    Validation: None
    Validation Message: None
    Tab Sequence: 10
    Initial Value: None
    Edit Style: Edit
    Edit limit: 0
Column: start_time
    Format: "[time]"
    Border style: Shadow Box
    Validation: None
    Validation Message: None
    Tab Sequence: 20
    Initial Value: None
    Edit Style: Edit
    Edit limit: 0
Column: end_time
DataWindow: d_class_info_update
Library: e:\thesis\appl\schedule.pbl
Date: 5/2/95   Time: 17:46:49

Border style: Shadow Box
Validation: None
Validation Message: None
Tab Sequence: 30
Initial Value: None
Edit Style: Edit
Edit limit: 0

Column: course_number
Format: "[general]"
Border style: None
Validation: None
Validation Message: None
Tab Sequence: 0
Initial Value: None
Edit Style: Edit
Edit limit: 0

Column: section_number
Format: "[general]"
Border style: None
Validation: None
Validation Message: None
Tab Sequence: 0
Initial Value: None
Edit Style: Edit
Edit limit: 0
DataWindow: d_class_info_update
Library: e:\thesis\appl\schedule.pbl
Date: 5/2/95    Time: 17:46:49

Format: "[general]"
Border style: None
Validation: None
Validation Message: None
Tab Sequence: 0
  Initial Value: None
  Edit Style: Edit
  Edit limit: 0
Visible = true   Enabled = true   TitleBar = true   Title = "Add New Term/Department"
ControlMenu = true  MinBox = true  MaxBox = true  Resizable = true
WindowType = main!  WindowState = normal!  BackColor = 12632256

Script for: open  event
//Connects the hidden datawindow with the database. Inserts a blank row into the add datawindow.

//Local Variables

//End Local Variables

SetTransObject(dw_term, SQLCA)
SetTransObject(dw_department, SQLCA)
dwShareData(uo_class_main.dw_term, dw_term)
dwShareData(uo_class_main.dw_department, dw_department)

InsertRow(dw_add, 1)
ScrollToRow(dw_add, 1)

End of Script

Graph: gr_2
X = 5   Y = 737   Width = 988   Height = 721
TabOrder = 0  Visible = true  TextColor = 0  BackColor = 12632256
TextColor = 6316128  Spacing = 100  Elevation = 20  Rotation = -20
Perspective = 2  Title = "(None)"  Border = true  BorderStyle = stylebox!
GraphType = colgraph!
Legend = atbottom!

DataWindow: dw_department
X = 910   Y = 13   Width = 494   Height = 165
TabOrder = 0  Enabled = true  DataObject = "d_department_list"
TitleBar = true  Title = "Department List"  Border = true
LiveScroll = true  BorderStyle = stylebox!
Window: w_add_term_dept
Library: e:\thesis\appl\schedule.pbl
Date: 5/1/95    Time: 21:18:01

DataWindow: dw_term
X = 33         Y = 457     Width = 494     Height = 205
TabOrder = 0   Enabled = true DataObject = "d_term_list"
TitleBar = true Title = "Term List" Border = true LiveScroll = true
BorderStyle = stylebox!

StaticText: st_1
X = 55         Y = 73      Width = 1171   Height = 93
TabOrder = 0   Visible = true Text = "Either enter a term, a department, or both:"
TextColor = 3354432 BackColor = 12632256
Alignment = left! FillPattern = solid!

CommandButton: cb_ok
X = 961        Y = 473     Width = 247     Height = 109
TabOrder = 30  Visible = true Enabled = true Text = "&OK"

Script for: clicked event
//Closes the window w_add_term_dept with saving the new term and/or new department back to the database.

//Local Variables
Boolean  lv_b_TermOkay
Boolean  lv_b_DepartmentOkay
Integer  lv_i_FoundRow
Integer  lv_i_InsertedRow
String   lv_s.Term
String   lv_s.Department
String   lv_s.FilterString
String   lv_s.OldTerm
String   lv_s.OldDepartment
//End Local Variables

AcceptText(dw_add)
lv_s_Term = Trim(GetItemString(dw_add, 1, "term"))
lv_s_Department = Trim(GetItemString(dw_add, 1, "department"))

IF (IsNull(lv_s_Term) OR (lv_s_Term = "")) THEN
  IF (IsNull(lv_s_Department) OR (lv_s_Department = "")) THEN
    TriggerEvent(cb_cancel, "clicked")
    Return
  ELSE
    lv_s_FilterString = "department = " + "'" + lv_s_Department + "'
    lv_i_FoundRow = dwFind(dw_department, lv_s_FilterString, 0, RowCount(dw_department))
    IF (lv_i_FoundRow = 0) THEN
      lv_i_InsertedRow = InsertRow(dw_department, 0)
      SetItem(dw_department, lv_i_InsertedRow, "department", lv_s_Department)
      IF (NOT IsNull(lv_s_OldDepartment)) OR (NOT (lv_s_OldDepartment = "")) THEN
        SetItem(dw_department, 1, "department", "")
      END IF
      gv_struct_parms.department = lv_s_Department
    END IF
    MessageBox("Information", "This department already exists. No additions will be made.", Information!, OR!)
    TriggerEvent(cb_cancel, "clicked")
    Return
  END IF
ENDIF
ELSEIF (IsNull(lv_s_Department) OR (lv_s_Department = "")) THEN
  lv_s_FilterString = "term = " + "'" + lv_s_Term + "'
  lv_i_FoundRow = dwFind(dw_term, lv_s_FilterString, 0, RowCount(dw_term))
  IF (lv_i_FoundRow = 0) THEN
    lv_i_InsertedRow = InsertRow(dw_term, 0)
    SetItem(dw_term, lv_i_InsertedRow, "term", lv_s_Term)
    IF (NOT IsNull(lv_s_OldTerm)) OR (NOT (lv_s_OldTerm = "")) THEN
      SetItem(dw_term, 1, "term", "")
    END IF
  END IF
ENDIF
gv_struct_parms.term = lv_s_Term
ELSE
MessageBox("Information", "This term already exists. No additions will be made", Information!, OK!)

    TriggerEvent(cb_cancel, "clicked")
    Return
END IF
ELSE
lv_s_FilterString = "term = " + "'" + lv_s_Term + "'
lv_i_FoundRow = dwFind(dw_term, lv_s_FilterString, 0, RowCount(dw_term))
IF (lv_i_FoundRow = 0) THEN
    lv_i_InsertedRow = InsertRow(dw_term, 0)
    SetItem(dw_term, lv_i_InsertedRow, "term", lv_s_Term)
    gv struct_parms.term = lv_s_Term
    lv_b_TermOkay = TRUE
END IF

lv_s_FilterString = "department = " + "'" + lv_s_Department + "'
lv_i_FoundRow = dwFind(dw_department, lv_s_FilterString, 0, RowCount(dw_department))
IF (lv_i_FoundRow = 0) THEN
    lv_i_InsertedRow = InsertRow(dw_department, 0)
    SetItem(dw_department, lv_i_InsertedRow, "department", lv_s_Department)
    gv struct_parms.department = lv_s_Department
    lv_b_DepartmentOkay = TRUE
END IF
IF (NOT lv_b_TermOkay) THEN
    IF (NOT lv_b_DepartmentOkay) THEN
        MessageBox("Information", "Both the department and term already exist.", Information!, OK!)
        TriggerEvent(cb_cancel, "clicked")
        Return
    ELSE
        MessageBox("Information", "This term already exists but the department will be added.", Information!, OK!)
        END IF
    ELSE
        THIS code block is not visible in the image.
Window: w_add_term_dept
Library: e:\thesis\appl\schedule.pbl
Date: 5/1/95    Time: 21:18:01

IF (NOT lv_b_DepartmentOkay) THEN
    MessageBox("Information", "This department already exists but the term will be added.", Information, OK!)
    END IF
END IF

// Update database with any changes
IF ((ModifiedCount(dw_term) > 0) OR (DeletedCount(dw_term) > 0)) THEN
    DeleteRow(dw_term, 1)
    IF (Update(dw_term) = 1) THEN
        COMMIT;
    ELSE
        ROLLBACK;
    END IF
END IF

IF ((ModifiedCount(dw_department) > 0) OR (DeletedCount(dw_department) > 0)) THEN
    DeleteRow(dw_department, 1)
    IF ((Update(dw_department) = 1)) THEN
        COMMIT;
    ELSE
        ROLLBACK;
    END IF
END IF

dwShareDataOff(dw_term)
dwShareDataOff(dw_department)

close(w_add_term_dept)

End of Script
Window: w_add_term_dept
Library: e:\thesis\appl\schedule.pbl
Date: 5/1/95    Time: 21:18:01

CommandButton: cb_cancel
X = 567        Y = 473        Width = 247       Height = 109
TabOrder = 20   Visible = true  Enabled = true  Text = "&Cancel"
Default = true

Script for: clicked  event
//Closes the window w_add_term_dept without saving the new term and/or new department.

dwShareDataOff(dw_term)
dwShareDataOff(dw_department)
close(w_add_term_dept)

End of Script

DataWindow: dw_add
X = 14        Y = 201        Width = 1358       Height = 197
TabOrder = 10   Visible = true  Enabled = true  DataObject = "d_add_dept_term"
BorderStyle = stylebox!
Retrieve: Script
Arguments: None
Update Table: Not Allowed
Filter: None
Sort: None
Sparse: None
Column: term
   Format: "[general]"
   Border style: Shadow Box
   Validation: None
   Validation Message: None
   Tab Sequence: 10
   Initial Value: None
   Edit Style: Edit
   Edit limit: 10

Column: department
   Format: "[general]"
   Border style: Shadow Box
   Validation: None
   Validation Message: None
   Tab Sequence: 20
   Initial Value: None
   Edit Style: Edit
   Edit limit: 4
Window: w_maint_main

Library: e:\thesis\appl\schedule.pbl
Date: 5/1/95 Time: 21:24:28

Window: w_maint_main
X = 919  Y = 593  Width = 2844  Height = 1905
Visible = true  Enabled = true  TitleBar = true  Title = "Untitled"
ControlMenu = true  MinBox = true  MaxBox = true  Resizable = true
WindowType = main!  WindowState = normal!  BackColor = 12632256

Script for: open event
//Opens either user object uo_class_maint or uo_prof_maint depending on whether w_class_main_test
//was displaying class information or faculty member information.

//Local Variables
String lv_s_Title
s_main_win_parms s_parms
//End Local Variables

s_parms = Message.PowerObjectParm

IF (gv_s_MainUoType = "class") THEN
  lv_s_Title = "Class Maintenance - " + s_parms.department + " " + s_parms.term
  This.Title = lv_s_Title
  OpenUserObjectWithParm(uo_class_maint, s_parms, 13, 13)
ELSEIF (gv_s_MainUoType = "prof") THEN
  lv_s_Title = "Faculty Member Maintenance - " + s_parms.department
  This.Title = lv_s_Title
  OpenUserObjectWithParm(uo_prof_maint, s_parms, 13, 13)
ELSE
  MessageBox("Information", "Oops!", Information!, Ok!)
END IF

End of Script
Window: uo_class_maint
Library: e:\thesis\appl\schedule.pbl
Date: 5/2/95    Time: 16:37:45

CommandButton  iv_cb_CurrentDefault
Integer        iv_i_UpdateRow
Integer        iv_i_Row

s_main_winParms  s_MainParms
s_class_infoParms  s_InfoParms

End of Instance Variables

Script for: constructor  event
//Retrieves any needed information from the database.

//Local Variables
String        lv_s_FindString
Integer       lv_i_FoundRow
//End Local Variables

SetPointer(Hourglass!)

s_MainParms = Message.PowerObjectParm

SetTransObject(dw_class_info_xref, SQLCA)

Retrieve(dw_class_info_xref, s_MainParms.term, s_MainParms.department)

SetSort(dw_class_info_xref, "course_num A, section_num A")
Sort(dw_class_info_xref)

IF (NOT s_MainParms.course_number = 0) THEN
  lv_s_FindString = "course_num = " + String(s_MainParms.course_number) &
    + " and section_num = " + "" + s_MainParms.section_number + ""
  lv_i_FoundRow = dwFind\dw_class_info_xref, lv_s_FindString, 1, RowCount(dw_class_info_xref)
SelectRow(dw_class_info_xref, 0, FALSE)
SelectRow(dw_class_info_xref, lv_i_FoundRow, TRUE)

iv_i_Row = lv_i_FoundRow

dw_class_info_add.enabled = FALSE
dw_class_info_update.enabled = FALSE

`cb_add.enabled = FALSE
cb_add.default = FALSE
cb_select.enabled = TRUE
cb_select.default = TRUE
iv_cb_CurrentDefault = cb_select
cb_delete.enabled = TRUE
cb_delete.default = FALSE
cb_reset maint.enabled = TRUE
cb_reset maint.default = FALSE
cb_update.enabled = FALSE
cb_update.default = FALSE
cb_cancel.enabled = FALSE
cb_cancel.default = FALSE
cb_more.enabled = FALSE
cb_more.default = FALSE

SetFocus(cb_select)
ELSE
    iv_cb_CurrentDefault = cb_add
    SetFocus(dw_class_info_add)
END IF

InsertRow(dw_class_info_add, 1)
InsertRow(dw_class_info_update, 1)

iv_b_ChangesMade = FALSE
s_InfoParms.first_time = TRUE
User Object: uo_class_maint
Library: e:\thesis\appl\schedule.pbl
Date: 5/2/95    Time: 16:37:45

End of Script

Script for: other  event
  //Tries to control tabbing within the user object.

  //Local Variables
  CommandButton  lv_cb_which
  DataWindow     lv_dw_which
  GraphicObject  lv_go_WhichControl
  String         lv_s_TextValue
  String         lv_s_DataObject

  //End Local Variables

  IF (KeyDown(KeyTab!) AND KeyDown(KeyShift!)) THEN
    IF lv_b_TabDone THEN
      lv_b_TabDone = FALSE
      Return
    END IF
    lv_go_WhichControl = GetFocus()
    choose case TypeOf(lv_go_WhichControl)
    case CommandButton!
        lv_cb_which = lv_go_WhichControl
        lv_s_TextValue = lv_cb_which.text
        IF (lv_s_TextValue = "&More ...") THEN
            IF (dw_class_info_add.enabled) THEN
                SetFocus(dw_class_info_add)
                SetColumn(dw_class_info_add, "end_time")
                Return
            ELSEIF (dw_class_info_update.enabled) THEN
                SetFocus(dw_class_info_update)
SetColumn(dw_class_info_update, "end_time")
Return
ELSEIF (cb_ok.enabled) THEN
  SetFocus(cb_ok)
  Return
ELSE
  Return
END IF
ELSEIF (lv_s_TextValue = "&Add") THEN
  IF (cb_more.enabled) THEN
    SetFocus(cb_more)
  Return
ELSEIF (dw_class_info_add.enabled) THEN
  SetFocus(dw_class_info_add)
  SetColumn(dw_class_info_add, "end_time")
  Return
ELSEIF (dw_class_info_update.enabled) THEN
  SetFocus(dw_class_info_update)
  SetColumn(dw_class_info_update, "end_time")
  Return
ELSEIF (cb_ok.enabled) THEN
  SetFocus(cb_ok)
  Return
ELSE
  Return
END IF
ELSEIF (lv_s_TextValue = "&Select") THEN
  IF (cb_add.enabled) THEN
    SetFocus(cb_add)
  Return
ELSEIF (cb_more.enabled) THEN
  SetFocus(cb_more)
  Return
ELSEIF (dw_class_info_add.enabled) THEN
  SetFocus(dw_class_info_add)
  SetColumn(dw_class_info_add, "end_time")
Return
ELSEIF (dw_class_info_update.enabled) THEN
    SetFocus(dw_class_info_update)
    SetColumn(dw_class_info_update, "end_time")
    Return
ELSEIF (cb_ok.enabled) THEN
    SetFocus(cb_ok)
    Return
ELSE
    Return
END IF
ELSEIF (lv_s_TextValue = "&Delete") THEN
    IF (cb_select.enabled) THEN
        SetFocus(cb_select)
        Return
    ELSEIF (cb_add.enabled) THEN
        SetFocus(cb_add)
        Return
    ELSEIF (cb_more.enabled) THEN
        SetFocus(cb_more)
        Return
    ELSEIF (dw_class_info_add.enabled) THEN
        SetFocus(dw_class_info_add)
        SetColumn(dw_class_info_add, "end_time")
        Return
    ELSEIF (dw_class_info_update.enabled) THEN
        SetFocus(dw_class_info_update)
        SetColumn(dw_class_info_update, "end_time")
        Return
    ELSEIF (cb_ok.enabled) THEN
        SetFocus(cb_ok)
        Return
    ELSE
        Return
    END IF
ELSEIF (lv_s_TextValue = "&Update") THEN
IF (cb_delete.enabled) THEN
  SetFocus(cb_delete)
  Return
ELSEIF (cb_select.enabled) THEN
  SetFocus(cb_select)
  Return
ELSEIF (cb_add.enabled) THEN
  SetFocus(cb_add)
  Return
ELSEIF (cb_more.enabled) THEN
  SetFocus(cb_more)
  Return
ELSEIF (dw_class_info_add.enabled) THEN
  SetFocus(dw_class_info_add)
  SetColumn(dw_class_info_add, "end_time")
  Return
ELSEIF (dw_class_info_update.enabled) THEN
  SetFocus(dw_class_info_update)
  SetColumn(dw_class_info_update, "end_time")
  Return
ELSEIF (cb_ok.enabled) THEN
  SetFocus(cb_ok)
  Return
ELSE
  Return
END IF
ELSIF (lv_s_TextValue = "&Reset") THEN
  IF (cb_update.enabled) THEN
    SetFocus(cb_update)
    Return
  ELSEIF (cb_delete.enabled) THEN
    SetFocus(cb_delete)
    Return
  ELSEIF (cb_select.enabled) THEN
    SetFocus(cb_select)
    Return
ELSEIF (cb_add.enabled) THEN
  SetFocus(cb_add)
  Return
ELSEIF (cb_more.enabled) THEN
  SetFocus(cb_more)
  Return
ELSEIF (dw_class_info_add.enabled) THEN
  SetFocus(dw_class_info_add)
  SetColumn(dw_class_info_add, "end_time")
  Return
ELSEIF (dw_class_info_update.enabled) THEN
  SetFocus(dw_class_info_update)
  SetColumn(dw_class_info_update, "end_time")
  Return
ELSEIF (cb_ok.enabled) THEN
  SetFocus(cb_ok)
  Return
ELSE
  Return
END IF
ELSEIF (lv_s_TextValue = "&Cancel") THEN
  IF (cb_reset_maint.enabled) THEN
    SetFocus(cb_reset_maint)
    Return
  ELSEIF (cb_update.enabled) THEN
    SetFocus(cb_update)
    Return
  ELSEIF (cb_delete.enabled) THEN
    SetFocus(cb_delete)
    Return
  ELSEIF (cb_select.enabled) THEN
    SetFocus(cb_select)
    Return
  ELSEIF (cb_add.enabled) THEN
    SetFocus(cb_add)
    Return
ELSEIF (cb_more.enabled) THEN
  SetFocus(cb_more)
  Return
ELSEIF (dw_class_info_add.enabled) THEN
  SetFocus(dw_class_info_add)
  SetColumn(dw_class_info_add, "end_time")
  Return
ELSEIF (dw_class_info_update.enabled) THEN
  SetFocus(dw_class_info_update)
  SetColumn(dw_class_info_update, "end_time")
  Return
ELSEIF (cb_ok.enabled) THEN
  SetFocus(cb_ok)
  Return
ELSE
  Return
END IF
ELSEIF (lv_g_TextValue = "&OK") THEN
  IF (cb_cancel.enabled) THEN
    SetFocus(cb_cancel)
    Return
  ELSEIF (cb_reset_maint.enabled) THEN
    SetFocus(cb_reset_maint)
    Return
  ELSEIF (cb_update.enabled) THEN
    SetFocus(cb_update)
    Return
  ELSEIF (cb_delete.enabled) THEN
    SetFocus(cb_delete)
    Return
  ELSEIF (cb_select.enabled) THEN
    SetFocus(cb_select)
    Return
  ELSEIF (cb_add.enabled) THEN
    SetFocus(cb_add)
    Return
ELSEIF (cb_more.enabled) THEN
    SetFocus(cb_more)
    Return
ELSEIF (dw_class_info_add.enabled) THEN
    SetFocus(dw_class_info_add)
    SetColumn(dw_class_info_add, "end_time")
    Return
ELSEIF (dw_class_info_update.enabled) THEN
    SetFocus(dw_class_info_update)
    SetColumn(dw_class_info_update, "end_time")
    Return
ELSE
    Return
END IF
ELSE
    Return
END IF

case DataWindow!
    lv_dw_which = lv_go_WhichControl
    lv_s_DataObject = lv_dw_which.dataobject
    IF (lv_s_DataObject = "d_class_info_add") THEN
        Return
    ELSEIF (lv_s_DataObject = "d_class_info_update") THEN
        Return
    ELSEIF (lv_s_DataObject = "d_class_info") THEN
        IF (cb_more.enabled) THEN
            SetFocus(cb_more)
            Return
        ELSEIF (dw_class_info_add.enabled) THEN
            SetFocus(dw_class_info_add)
            SetColumn(dw_class_info_add, "end_time")
            Return
        ELSEIF (dw_class_info_update.enabled) THEN
            Return
        ELSE
            Return
        END IF
    END IF
User Object: uo_class_maint
Library: e:\thesis\appl\schedule.pbl
Date: 5/2/95     Time: 16:37:45

    SetFocus(dw_class_info_update)
    SetColumn(dw_class_info_update, "end_time")
    Return
    ELSEIF (cb_ok.enabled) THEN
        SetFocus(cb_ok)
        Return
    ELSE
        Return
    END IF
    ELSE
        Return
    END IF

    case else
    Return
    END Choose

    ELSEIF (KeyDown(KeyTab!)) THEN
        IF (lv_b_TabDone) THEN
            lv_b_TabDone = FALSE
            Return
        END IF

        lv_go_WhichControl = GetFocus()

        choose caseTypeOf(lv_go_WhichControl)

        case CommandButton!
            lv_cb_which = lv_go_WhichControl
            lv_s_TextValue = lv_cb_which.text
            IF (lv_s_TextValue = "&More ...") THEN
                IF (cb_add.enabled) THEN
                    SetFocus(cb_add)
                    Return
                ELSEIF (cb_select.enabled) THEN
                    SetFocus(cb_select)
                    Return

                ...
ELSEIF (cb_delete.enabled) THEN
  SetFocus(cb_delete)
  Return
ELSEIF (cb_update.enabled) THEN
  SetFocus(cb_update)
  Return
ELSEIF (cb_reset_maint.enabled) THEN
  SetFocus(cb_reset_maint)
  Return
ELSEIF (cb_cancel.enabled) THEN
  SetFocus(cb_cancel)
  Return
ELSEIF (cb_ok.enabled) THEN
  SetFocus(cb_ok)
  Return
ELSE
  Return
END IF
ELSEIF (lv_s_TextValue = "&Add") THEN
  IF (cb_select.enabled) THEN
    SetFocus(cb_select)
    Return
  ELSEIF (cb_delete.enabled) THEN
    SetFocus(cb_delete)
    Return
  ELSEIF (cb_update.enabled) THEN
    SetFocus(cb_update)
    Return
  ELSEIF (cb_reset_maint.enabled) THEN
    SetFocus(cb_reset_maint)
    Return
  ELSEIF (cb_cancel.enabled) THEN
    SetFocus(cb_cancel)
    Return
  ELSEIF (cb_ok.enabled) THEN
    SetFocus(cb_ok)
  END IF
Return
ELSE
    Return
END IF
ELSEIF (lv_s_TextValue = "&Select") THEN
    IF (cb_delete.enabled) THEN
        SetFocus(cb_delete)
        Return
    ELSEIF (cb_update.enabled) THEN
        SetFocus(cb_update)
        Return
    ELSEIF (cb_reset_maint.enabled) THEN
        SetFocus(cb_reset_maint)
        Return
    ELSEIF (cb_cancel.enabled) THEN
        SetFocus(cb_cancel)
        Return
    ELSEIF (cb_ok.enabled) THEN
        SetFocus(cb_ok)
        Return
    ELSE
        Return
    END IF
ELSEIF (lv_s_TextValue = "&Delete") THEN
    IF (cb_update.enabled) THEN
        SetFocus(cb_update)
        Return
    ELSEIF (cb_reset_maint.enabled) THEN
        SetFocus(cb_reset_maint)
        Return
    ELSEIF (cb_cancel.enabled) THEN
        SetFocus(cb_cancel)
        Return
    ELSEIF (cb_ok.enabled) THEN
        SetFocus(cb_ok)
        Return
    ELSE
        Return
    END IF
ELSE
    Return
END IF
ELSEIF (lv_s_TextValue = "&Update") THEN
    IF (cb_reset_maint.enabled) THEN
        SetFocus(cb_reset_maint)
        Return
    ELSEIF (cb_cancel.enabled) THEN
        SetFocus(cb_cancel)
        Return
    ELSEIF (cb_ok.enabled) THEN
        SetFocus(cb_ok)
        Return
    ELSE
        Return
    END IF
ELSEIF (lv_s_TextValue = "&Reset") THEN
    IF (cb_cancel.enabled) THEN
        SetFocus(cb_cancel)
        Return
    ELSEIF (cb_ok.enabled) THEN
        SetFocus(cb_ok)
        Return
    ELSE
        Return
    END IF
ELSEIF (lv_s_TextValue = "&Cancel") THEN
    IF (cb_ok.enabled) THEN
        SetFocus(cb_ok)
        Return
    ELSE
        Return
    END IF
ELSEIF (lv_s_TextValue = "&OK") THEN
    IF (dw_class_info_add.enabled) THEN
        SetFocus(dw_class_info_add)
User Object: uo_class_maint
Library: e:\thesis\appl\schedule.pbl
Date: 5/2/95      Time: 16:37:45

SetColumn(dw_class_info_add, "course_number")
Return
ELSEIF (dw_class_info_update.enabled) THEN
SetFocus(dw_class_info_update)
SetColumn(dw_class_info_update, "days")
Return
ELSEIF (cb_more.enabled) THEN
SetFocus(cb_more)
Return
ELSEIF (cb_add.enabled) THEN
SetFocus(cb_add)
Return
ELSEIF (cb_select.enabled) THEN
SetFocus(cb_select)
Return
ELSEIF (cb_delete.enabled) THEN
SetFocus(cb_delete)
Return
ELSEIF (cb_update.enabled) THEN
SetFocus(cb_update)
Return
ELSEIF (cb_reset_maint.enabled) THEN
SetFocus(cb_reset_maint)
Return
ELSEIF (cb_cancel.enabled) THEN
SetFocus(cb_cancel)
Return
ELSE
Return
END IF
ELSE
Return
END IF

case DataWindow!
1v_dw_Which = 1v_go_WhichControl
lv_s_DataObject = lv_dw_ Which.dataobject

IF (lv_s_DataObject = "d_class_info_add") THEN
    Return
ELSEIF (lv_s_DataObject = "d_class_info_update") THEN
    Return
ELSEIF (lv_s_DataObject = "d_class_info") THEN
    IF (cb_add.enabled) THEN
        SetFocus(cb_add)
        Return
    ELSEIF (cb_select.enabled) THEN
        SetFocus(cb_select)
        Return
    ELSEIF (cb_delete.enabled) THEN
        SetFocus(cb_delete)
        Return
    ELSEIF (cb_update.enabled) THEN
        SetFocus(cb_update)
        Return
    ELSEIF (cb_reset_maint.enabled) THEN
        SetFocus(cb_reset_maint)
        Return
    ELSEIF (cb_cancel.enabled) THEN
        SetFocus(cb_cancel)
        Return
    ELSEIF (cb_ok.enabled) THEN
        SetFocus(cb_ok)
        Return
    ELSE
        Return
    END IF
ELSE
    Return
END IF
ELSE
    Return
END IF
User Object: uo_class_maint
Library: e:\thesis\appl\schedule.pbl
Date: 5/2/95    Time: 16:37:45

    case else
    Return
    END Choose
END IF

End of Script

DataWindow: dw_class_info_update
  X = 37    Y = 225    Width = 2369    Height = 129
  TabOrder = 10    Enabled = true    DataObject = "d_class_info_update"
  LiveScroll = true    BorderStyle = stylebox!

Script for: editchanged event
//After a change has been made, allows the update button to be enabled so that changes can be
//moved to the class_info_xref datawindow and be saved (if desired).

//Local Variables

//End Local Variables

//Indicate that changes were made
iv_b_ChangesMade = TRUE

//Command Button Control
cb_update.enabled = TRUE

End of Script

Script for: dwnkey event
//Tries to control tabbing.

//Local Variables
Window: uo_class_maint
Library: e:\thesis\appl\schedule.pbl
Date: 5/2/95     Time: 16:37:45

String    lv_s_CurrentColumn
//End Local Variables

lv_s_CurrentColumn = GetColumnName(dw_class_info_update)

IF (KeyDown(keyTab!) AND KeyDown(keyShift!)) THEN
    IF (lv_s_CurrentColumn = "days") THEN
        iv_b_TabDone    = TRUE
        SetFocus(cb_ok)
        Return
    ELSE
        Return
    END IF
ELSEIF (KeyDown(keyTab!) AND (lv_s_CurrentColumn = "end_time")) THEN
    iv_b_TabDone    = TRUE
    cb_update.enabled = TRUE
    cb_reset_maint.enabled = TRUE
    iv_cb_CurrentDefault.default  = FALSE
    cb_more.default    = TRUE
    iv_cb_CurrentDefault = cb_more
    SetFocus(cb_more)
    Return
END IF

End of Script

---

StaticText: st_7
X = 92        Y = 441     Width = 563     Height = 73
TabOrder = 0    Visible = true   Text = "Current Term/Department"
TextColor = 8388608  BackColor = 12632256     Alignment = left!
FillPattern = solid!
Window: uo_class_maint
Library: e:\thesis\appl\schedule.pbl
Date: 5/2/95     Time: 16:37:45

StaticText: st_6
X = 2081      Y = 113
TabOrder = 0   Visible = true
BackColor = 12632256
Width = 220     Height = 73
Text = "End Time"    TextColor = 8388608
Alignment = left!  FillPattern = solid!

StaticText: st_5
X = 1751      Y = 113
TabOrder = 0   Visible = true
BackColor = 12632256
Width = 247     Height = 73
Text = "Start Time"  TextColor = 8388608
Alignment = left!  FillPattern = solid!

StaticText: st_4
X = 1532      Y = 113
TabOrder = 0   Visible = true
BackColor = 12632256
Width = 124     Height = 73
Text = "Days"    TextColor = 8388608
Alignment = left!  FillPattern = solid!

StaticText: st_3
X = 1125      Y = 113
TabOrder = 0   Visible = true
BackColor = 12632256
Width = 142     Height = 73
Text = "Name"    TextColor = 8388608
Alignment = left!  FillPattern = solid!

StaticText: st_2
X = 531       Y = 113
TabOrder = 0   Visible = true
TextColor = 8388608 BackColor = 12632256
Width = 357     Height = 73
Text = "Section Number"  Alignment = left!
FillPattern = solid!
Window: uo_class_maint
Library: e:\thesis\appl\schedule.pbl
Date: 5/2/95   Time: 16:37:45

StaticText: st_1
  X = 129    Y = 113    Width = 343    Height = 73
  TabOrder = 0    Visible = true    Text = "Course Number"
  TextColor = 8388608 BackColor = 12632256    Alignment = left!
  FillPattern = solid!

CommandButton: cb_more
  X = 2423    Y = 157    Width = 247    Height = 109
  TabOrder = 30    Visible = true    Text = "&More ..."

Script for: clicked event
//Opens another window used to get more class (and lab) information

  //Local Variables

  //End Local Variables

  IF (dw_class_info_add.visible) THEN
    s_InfoParms.add_mode = TRUE
  ELSE
    s_InfoParms.add_mode = FALSE
    cb_update.enabled = TRUE
  END IF

  OpenWithParm(w_more_class_info, s_InfoParms)

  End of Script

  Script for: uponreturn event
  //Updates information when the user returns from the window used to obtain more class (and lab) information.
// Local Variables
// End Local Variables

s_InfoParms = Message.PowerObjectParm

IF (dw_class_info_add.enabled) THEN
  iv_cb_CurrentDefault.default = FALSE
  cb_add.default = TRUE
  iv_cb_CurrentDefault = cb_add
  SetFocus(cb_add)
ELSEIF (dw_class_info_update.enabled) THEN
  iv_cb_CurrentDefault.default = FALSE
  cb_update.default = TRUE
  iv_cb_CurrentDefault = cb_update
  SetFocus(cb_update)
END IF

iv_b_ChangesMade = TRUE

End of Script

Script for: lbuttondown event
// Used to track which button has default.

// Local Variables
// End Local Variables

iv_cb_CurrentDefault.default = FALSE
iv_cb_CurrentDefault = cb_more

End of Script
CommandButton: cb_ok
X = 2387  Y = 1601  Width = 247  Height = 109
TabOrder = 110  Visible = true  Enabled = true  Text = "&OK"

Script for: clicked event
//Returns back to the class information user object.
//But first updates the database with any changes made.

//Local Variables

//End Local Variables

//Update database with current changes if any were made.
IF ((ModifiedCount(dw_class_info_xref) > 0) OR (DeletedCount(dw_class_info_xref) > 0)) THEN
  IF (Update(dw_class_info_xref) = 1) THEN
    COMMIT;
  ELSE
    ROLLBACK;
  END IF
END IF

close(w_maint_main)

TriggerEvent(uo_class_main, "getcontrolagain")

End of Script

Script for: lbuttondown event
//Used to track which button has default.

//Local Variables
Window: uo_class_maint
Library: e:\thesis\appl\schedule.pbl
Date: 5/2/95    Time: 16:37:45

//End Local Variables

iv_cb_CurrentDefault.default = FALSE
iv_cb_CurrentDefault = cb_ok

End of Script

CommandButton: cb_cancel
X = 2058       Y = 1601       Width = 247       Height = 109
TabOrder = 100  Visible = true  Text = "&Cancel"

Script for: clicked event
//Returns back to the class information user object.
//Any changes will NOT be saved.

close(w_maint_main)

End of Script

Script for: buttondown event
//Used to track which button has default.

//Local Variables

//End Local Variables

iv_cb_CurrentDefault.default = FALSE
iv_cb_CurrentDefault = cb_cancel

End of Script
CommandButton: cb_reset_maint
X = 1495   Y = 1601   Width = 247   Height = 109
TabOrder = 90   Visible = true   Text = "$\&Reset$"

Script for: clicked event
//Removes any information in the class_info_add datawindow or class_info_update datawindow
//and unselects any selected row in dw_class_info_xref.

//Local Variables

//End Local Variables

IF (cb_add.enabled) THEN
    Reset(dw_class_info_add)
    InsertRow(dw_class_info_add, 0)
    ScrollToRow(dw_class_info_add, 1)
    SetFocus(dw_class_info_add)
    SetColumn(dw_class_info_add, "course_number")

    //Reset the structure values
    s_InfoParms.first_time = TRUE
    s_InfoParms.location = ""
    s_InfoParms.max_credits = 0
    s_InfoParms.min_credits = 0
    s_InfoParms.building = ""
    s_InfoParms.room = ""
    s_InfoParms.class_limit = 0
    s_InfoParms.flags = ""
    s_InfoParms.lab = ""
    s_InfoParms.lab_days = ""
    s_InfoParms.lab_start_time = 00:00:00
    s_InfoParms.lab_end_time = 00:00:00
    s_InfoParms.lab_location = ""
    s_InfoParms.lab_max_credits = 0
s_InfoParams.lab_min_credits = 0
s_InfoParams.lab_building = ""
    s_InfoParams.lab_room = ""

    cb_more.enabled = FALSE
    cb_reset_maint.enabled = FALSE
    iv_cb_CurrentDefault.default = FALSE
    cb_add.default = TRUE
    iv_cb_CurrentDefault = cb_add

Return
END IF

SetRedraw(dw_class_info_xref, FALSE)
SetRedraw(dw_class_info_add, FALSE)
SetRedraw(dw_class_info_update, FALSE)

SelectRow(dw_class_info_xref, 0, FALSE)

Reset(dw_class_info_update)
InsertRow(dw_class_info_update, 0)
ScrollToRow(dw_class_info_update, 1)

dw_class_info_update.visible = FALSE
dw_class_info_add.visible = TRUE

dw_class_info_add.enabled = TRUE
dw_class_info_update.enabled = FALSE
dw_class_info_xref.enabled = TRUE

//Reset the structure values
s_InfoParams.first_time = TRUE
s_InfoParams.location = ""
    s_InfoParams.max_credits = 0
    s_InfoParams.min_credits = 0
    s_InfoParams.building = ""
Window: uo_class_maint
Library: e:\thesis\appl\schedule.pbl
Date: 5/2/95    Time: 16:37:45

s_InfoParms.room = ""
s_InfoParms.class_limit = 0
s_InfoParms.flags = ""
s_InfoParms.lab = ""
s_InfoParms.lab_days = ""
s_InfoParms.lab_start_time = 00:00:00
s_InfoParms.lab_end_time = 00:00:00
s_InfoParms.lab_location = ""
s_InfoParms.lab_max_credits = 0
s_InfoParms.lab_min_credits = 0
s_InfoParms.lab_building = ""
s_InfoParms.lab_room = ""

iv_b_ChangesMade = FALSE

SetRedraw(dw_class_info_update, TRUE)
SetRedraw(dw_class_info_add, TRUE)
SetRedraw(dw_class_info_xref, TRUE)

//Command Button Control
ch_reset_maint.enabled = FALSE
ch_add.enabled = TRUE
iv_cb_CurrentDefault.default = FALSE
ch_add.default = TRUE
iv_cb_CurrentDefault = ch_add
ch_select.enabled = FALSE
ch_delete.enabled = FALSE
ch_update.enabled = FALSE
ch_more.enabled = FALSE

SetFocus(dw_class_info_add)

End of Script
//Local Variables

//End Local Variables

iv_cb_CurrentDefault.default = FALSE
iv_cb_CurrentDefault = cb_reset_maint

End of Script

CommandButton: cb_update
X = 1166     Y = 1601       Width = 247       Height = 109
TabOrder = 80     Visible = true       Text = "&Update"

Script for: clicked event
//Updates the changed information from dw_class_info_update (and more class/lab information window) in
//the class_info_xref datawindow.

//Local Variables
Integer     lv_i_CourseNumber
String      lv_s_SectionNumber
String      lv_s_Name
String      lv_s_Days
Time         lv_t_StartTime
Time         lv_t_EndTime
//End Local Variables

SetPointer(Hourglass!)

AcceptText(dw_class_info_update)

lv_i_CourseNumber = GetItemNumber(dw_class_info_update, 1, "course_number")
lv_s_SectionNumber = Trim(GetItemString(dw_class_info_update, 1, "section_number"))
lv_s_Name = Trim(GetItemString(dw_class_info_update, 1, "name"))
lv_s_Days = Trim(GetItemString(dw_class_info_update, 1, "days"))
lv_t_StartTime = GetItemTime(dw_class_info_update, 1, "start_time")
lv_t_EndTime = GetItemTime(dw_class_info_update, 1, "end_time")
SetItem(dw_class_info_xref, iv_i_UpdateRow, "course_num", lv_i_CourseNumber)
SetItem(dw_class_info_xref, iv_i_UpdateRow, "section_num", lv_s_SectionNumber)
SetItem(dw_class_info_xref, iv_i_UpdateRow, "name", lv_s_Name)
SetItem(dw_class_info_xref, iv_i_UpdateRow, "start_time", lv_t_StartTime)
SetItem(dw_class_info_xref, iv_i_UpdateRow, "end_time", lv_t_EndTime)

SetItem(dw_class_info_xref, iv_i_UpdateRow, "location", s_InfoParms.location)
SetItem(dw_class_info_xref, iv_i_UpdateRow, "max_credits", s_InfoParms.max_credits)
SetItem(dw_class_info_xref, iv_i_UpdateRow, "min_credits", s_InfoParms.min_credits)
SetItem(dw_class_info_xref, iv_i_UpdateRow, "building", s_InfoParms.building)
SetItem(dw_class_info_xref, iv_i_UpdateRow, "room", s_InfoParms.room)
SetItem(dw_class_info_xref, iv_i_UpdateRow, "class_limit", s_InfoParms.class_limit)
SetItem(dw_class_info_xref, iv_i_UpdateRow, "flags", s_InfoParms.flags)
SetItem(dw_class_info_xref, iv_i_UpdateRow, "lab", s_InfoParms.lab)
SetItem(dw_class_info_xref, iv_i_UpdateRow, "lab_days", s_InfoParms.lab_days)
SetItem(dw_class_info_xref, iv_i_UpdateRow, "lab_location", s_InfoParms.lab_location)
SetItem(dw_class_info_xref, iv_i_UpdateRow, "lab_max_credits", s_InfoParms.lab_max_credits)
SetItem(dw_class_info_xref, iv_i_UpdateRow, "lab_min_credits", s_InfoParms.lab_min_credits)
SetItem(dw_class_info_xref, iv_i_UpdateRow, "lab_start_time", s_InfoParms.lab_start_time)
SetItem(dw_class_info_xref, iv_i_UpdateRow, "lab_end_time", s_InfoParms.lab_end_time)
SetItem(dw_class_info_xref, iv_i_UpdateRow, "lab_building", s_InfoParms.lab_building)
SetItem(dw_class_info_xref, iv_i_UpdateRow, "lab_room", s_InfoParms.lab_room)

//Reset the structure values
s_InfoParms.first_time = TRUE
s_InfoParms.location = ""
size_InfoParms.max_credits = 0
s_InfoParms.min_credits = 0
s_InfoParms.building = ""
size_InfoParms.room = ""
size_InfoParms.class_limit = 0
size_InfoParms.flags = ""
size_InfoParms.lab = ""
size_InfoParms.lab_days = ""
size_InfoParms.lab_start_time = 00:00:00
size_InfoParms.lab_end_time = 00:00:00
Windcw: uc_class_maint
Library: e:\thesis\appl\schedule.pbl
Date: 5/2/95      Time: 16:37:45

s_InfoParms.lab_location = ""
s_InfoParms.lab_max_credits = 0
s_InfoParms.lab_min_credits = 0
s_InfoParms.lab_building = ""
s_InfoParms.lab_room = ""

SetRedraw(dw_class_info_xref, FALSE)
SetRedraw(dw_class_info_add, FALSE)
SetRedraw(dw_class_info_update, FALSE)

SelectRow(dw_class_info_xref, 0, FALSE)

Reset(dw_class_info_update)
InsertRow(dw_class_info_update, 0)
ScrollToRow(dw_class_info_update, 1)

dw_class_info_update.visible = FALSE
dw_class_info_add.visible = TRUE

dw_class_info_update.enabled = FALSE
dw_class_info_add.enabled = TRUE

SetFocus(dw_class_info_add)

//Command Button Control
cb_more.enabled = FALSE
cb_add.enabled = TRUE
iv_cb_CurrentDefault.default = FALSE
cb_add.default = TRUE
iv_cb_CurrentDefault = cb_add
cb_select.enabled = FALSE
cb_delete.enabled = FALSE
cb_update.enabled = FALSE
cb_reset_maint.enabled = FALSE
cb_cancel.enabled = TRUE
iv_b_ChangesMade = FALSE

SetRedraw(dw_class_info_xref, TRUE)
SetRedraw(dw_class_info_add, TRUE)
SetRedraw(dw_class_info_update, TRUE)

End of Script

...

Script for: lbuttondown event
//Used to track which button has default.

//Local Variables

//End Local Variables

iv_cb_CurrentDefault.default = FALSE
iv_cb_CurrentDefault = cf_update

End of Script

CommandButton: cb_delete
X = 837  Y = 1601  Width = 247  Height = 109
TabOrder = 70  Visible = true  Text = "&Delete"

Script for: clicked event
//Deletes the current selected row in dw_class_info_xref after verifying the user wants to delete //the information.

//Local Variables
Integer lv_i_Row
//End Local Variables
Window: uc_class_maint
Library: e:\thesis\appl\schedule.pb1
Date: 5/2/95 Time: 16:37:45

//Capture the row number to delete.
lv_i_Row = GetSelectedRow(dw_class_info_xref, 0)

//Ensure a row has been selected prior to deletion.
IF (lv_i_Row = 0) THEN
    MessageBox("Information", "A row must be selected before deleting.", Information!, OK!)
    Return
END IF

IF (MessageBox("Information", "Delete current row?", Question!, YesNo!, 2) = 1) THEN
    DeleteRow(dw_class_info_xref, lv_i_Row)
    dw_class_info_add.enabled = TRUE
    dw_class_info_update.enabled = FALSE

    //Command Button Control
    cb_add.enabled = TRUE
    iv_cb_CurrentDefault.default = FALSE
    cb_add.default = TRUE
    iv_cb_CurrentDefault = cb_add
    cb_select.enabled = FALSE
    cb_delete.enabled = FALSE
    cb_update.enabled = FALSE
    cb_reset_maint.enabled = FALSE
    cb_cancel.enabled = TRUE

    SetFocus(dw_class_info_add)
END IF

End of Script

Script for: lbbuttondown event
//Used to track which button has default.
Window: uo_class_main
Library: e:\thesis\appl\schedule.pbl
Date: 5/2/95    Time: 16:37:45

//Local Variables

//End Local Variables

iv_cb_CurrentDefault.default = FALSE
iv_cb_CurrentDefault = cb_delete

End of Script

CommandButton: cb_add
X = 119    Y = 1601    Width = 247    Height = 109
TabOrder = 50    Visible = true    Enabled = true    Text = "&Add"
Default = true

Script for: clicked event
//Takes data entered from the add datawindow, validates the information, and sets the information
//into the class_xref datawindow.

//Local Variables
Integer  lv_i_InsertedRow
Integer  lv_i_CourseRow
Integer  lv_i_CourseNumber
String   lv_s_SectionNumber
String   lv_s_Name
String   lv_s_Days
String   lv_s_FindString
Time     lv_t_StartTime
Time     lv_t_EndTime

Integer  lv_i_MaxCredits
//End Local Variables

SetPointer(Hourglass!)

AcceptText(dw_class_info_add)
SetRedraw(dw_class_info_xref, TRUE)
SetFocus(dw_class_info_add)
Return
END IF

lv_i_InsertedRow = InsertRow(dw_class_info_xref, 0)
SetItem(dw_class_info_xref, lv_i_InsertedRow, "term", s_MainParms.term)
SetItem(dw_class_info_xref, lv_i_InsertedRow, "department", s_MainParms.department)
SetItem(dw_class_info_xref, lv_i_InsertedRow, "course_num", lv_i_CourseNumber)
SetItem(dw_class_info_xref, lv_i_InsertedRow, "section_num", lv_s_SectionNumber)
SetItem(dw_class_info_xref, lv_i_InsertedRow, "name", lv_s_Name)
SetItem(dw_class_info_xref, lv_i_InsertedRow, "days", lv_s_Days)
SetItem(dw_class_info_xref, lv_i_InsertedRow, "start_time", lv_t_StartTime)
SetItem(dw_class_info_xref, lv_i_InsertedRow, "end_time", lv_t_EndTime)

lv_i_MaxCredits = s_InfoParms.max_credits
SetItem(dw_class_info_xref, lv_i_InsertedRow, "location", s_InfoParms.location)
SetItem(dw_class_info_xref, lv_i_InsertedRow, "max_credits", s_InfoParms.max_credits)
SetItem(dw_class_info_xref, lv_i_InsertedRow, "min_credits", s_InfoParms.min_credits)
SetItem(dw_class_info_xref, lv_i_InsertedRow, "building", s_InfoParms.building)
SetItem(dw_class_info_xref, lv_i_InsertedRow, "room", s_InfoParms.room)
SetItem(dw_class_info_xref, lv_i_InsertedRow, "class_limit", s_InfoParms.class_limit)
SetItem(dw_class_info_xref, lv_i_InsertedRow, "flags", s_InfoParms.flags)
SetItem(dw_class_info_xref, lv_i_InsertedRow, "lab", s_InfoParms.lab)
SetItem(dw_class_info_xref, lv_i_InsertedRow, "lab_days", s_InfoParms.lab_days)
SetItem(dw_class_info_xref, lv_i_InsertedRow, "lab_start_time", s_InfoParms.lab_start_time)
SetItem(dw_class_info_xref, lv_i_InsertedRow, "lab_end_time", s_InfoParms.lab_end_time)
SetItem(dw_class_info_xref, lv_i_InsertedRow, "lab_location", s_InfoParms.lab_location)
SetItem(dw_class_info_xref, lv_i_InsertedRow, "lab_max_credits", s_InfoParms.lab_max_credits)
SetItem(dw_class_info_xref, lv_i_InsertedRow, "lab_min_credits", s_InfoParms.lab_min_credits)
SetItem(dw_class_info_xref, lv_i_InsertedRow, "lab_building", s_InfoParms.lab_building)
SetItem(dw_class_info_xref, lv_i_InsertedRow, "lab_room", s_InfoParms.lab_room)

SelectRow(dw_class_info_xref, 0, FALSE)
SetSort(dw_class_info_xref, "course_num A, section_num A")
Sort(dw_class_info_xref)

//Don't think this will work ... but what the heck ... Did it?? What was it supposed to accomplish?
lv_i_InsertedRow = GetSelectedRow(dw_class_info_xref, 0)
ScrollToRow(dw_class_info_xref, lv_i_InsertedRow)

SetRedraw(dw_class_info_add, FALSE)

Reset(dw_class_info_add)
InsertRow(dw_class_info_add, 0)
ScrollToRow(dw_class_info_add, 1)

SetFocus(dw_class_info_add)

SetRedraw(dw_class_info_xref, TRUE)
SetRedraw(dw_class_info_add, TRUE)

//Reset the structure values
s_InfoParms.first_time = TRUE
s_InfoParms.location = ""
s_InfoParms.max_credits = 0
s_InfoParms.min_credits = 0
s_InfoParms.building = ""
s_InfoParms.room = ""
s_InfoParms.class_limit = 0
s_InfoParms.flags = ""
s_InfoParms.lab = ""
s_InfoParms.lab_days = ""
s_InfoParms.lab_start_time = 00:00:00
s_InfoParms.lab_end_time = 00:00:00
s_InfoParms.lab_location = ""
s_InfoParms.lab_max_credits = 0
s_InfoParms.lab_min_credits = 0
s_InfoParms.lab_building = ""
s_InfoParms.lab_room = ""
iv_b_ChangesMade = FALSE

//Command Button Control
cb_reset_maint.enabled = FALSE
cb_more.enabled = FALSE
cb_cancel.enabled = TRUE

End of Script

Script for: lbbuttondown event
//Used to track which button has default.

//Local Variables

//End Local Variables

iv_cb_CurrentDefault.default = FALSE
iv_cb_CurrentDefault = cb_add

End of Script

CommandButton: cb_select
X = 449       Y = 1601       Width = 307       Height = 109
TabOrder = 60       Visible = true       Text = "&Select"

Script for: clicked event
//Triggers event "doubledclicked" for dw_class_info_xref

//Local Variables
Window: uo_class_maint
Library: e:\thesis\appl\schedule.pbl
Date: 5/2/95    Time: 16:37:45

//End Local Variables

TriggerEvent(dw_class_info_xref, "DoubleClicked")

End of Script

Script for: lbuttondown event
//Used to track which button has default.

//Local Variables

//End Local Variables

iv_cb_CurrentDefault.default = FALSE
iv_cb_CurrentDefault = cb_select

End of Script

DataWindow: dw_class_info_add
X = 37    Y = 225    Width = 2327    Height = 129
TabOrder = 20    Visible = true    Enabled = true    DataObject = "d_class_info_add"
LiveScroll = true    BorderStyle = stylebox

Script for: editchanged event
//Once data has been entered into the datawindow, buttons are enabled to get more information and
//reset the information.

//Local Variables

//End Local Variables
Window: uo_class_maint
Library: e:\thesis\appl\schedule.pbl
Date: 5/2/95 Time: 16:37:45

cb_more.enabled = TRUE
cb_reset_maint.enabled = TRUE

SelectRow(dw_class_info_xref, 0, FALSE)

cb_select.enabled = FALSE
cb_delete.enabled = FALSE

End of Script

Script for: dwnkey event
//Used to control tabbing within the datawindow.

//Local Variables
String   lv_s_CurrentColumn
Boolean   lv_defaults[8]
//End Local Variables

lv_s_CurrentColumn = GetColumnName(dw_class_info_add)

IF (KeyDown(keyTab!) AND KeyDown(keyShift!)) THEN
   IF (lv_s_CurrentColumn = "course_number") THEN
      lv_b_TabDone = TRUE
      SetFocus(cb_ok)
      Return
   ELSE
      Return
   END IF
ENDIF
ELSEIF (KeyDown(keyTab!) AND (lv_s_CurrentColumn = "end_time")) THEN
   lv_b_TabDone = TRUE
   IF (cb_more.enabled) THEN
      lv_cb_CurrentDefault.default = FALSE
      cb_more.default = TRUE
      lv_cb_CurrentDefault = cb_more
   ELSE
SetFocus(cb_more)
Return
ELSEIF (cb_add.enabled) THEN
  SetFocus(cb_add)
  Return
ELSEIF (cb_select.enabled) THEN
  SetFocus(cb_select)
  Return
ELSEIF (cb_delete.enabled) THEN
  SetFocus(cb_delete)
  Return
ELSEIF (cb_update.enabled) THEN
  SetFocus(cb_update)
  Return
ELSEIF (cb_reset_maint.enabled) THEN
  SetFocus(cb_reset_maint)
  Return
ELSEIF (cb_cancel.enabled) THEN
  SetFocus(cb_cancel)
  Return
ELSEIF (cb_ok.enabled) THEN
  SetFocus(cb_ok)
  Return
ELSE
  Return
END IF
ELSEIF (KeyDown(keyEnter!)) THEN
  Return
END IF

End of Script
DataWindow: dw_class_info_xref
X = 33    Y = 473    Width = 2721    Height = 1053
TabOrder = 40    Visible = true    Enabled = true    DataObject = "d_class_info"
VScrollBar = true    Border = true    LiveScroll = true    BorderStyle = stylebox!

Script for: clicked event
//Highlights the clicked row and changes which buttons are enabled and disabled.

//Local Variables
Integer  lv_i_SelectedRow
//End Local Variables

iv_i_Row = GetClickedRow(dw_class_info_xref)
IF (iv_i_Row < 1) THEN
    Return
END IF

lv_i_SelectedRow = GetSelectedRow(dw_class_info_xref, 0)
IF (iv_i_Row = lv_i_SelectedRow) THEN
    SelectRow(dw_class_info_xref, 0, FALSE)
    cb_delete.enabled = FALSE
    cb_add.enabled = TRUE
    iv_cb_CurrentDefault.default = FALSE
    cb_add.default = TRUE
    iv_cb_CurrentDefault = cb_add
    cb_reset_maint.enabled = FALSE
    cb_update.enabled = FALSE
    cb_select.enabled = FALSE

    IF (iv_b_ChangesMade) THEN
        IF (MessageBox("Changes Made", "Do you wish to save changes?", Question!, YesNo!) = 1) THEN
            TriggerEvent(cb_update, "clicked")
        END IF
    END IF
END IF
ELSE
    iv_b_ChangesMade = FALSE
END IF
END IF

SetRedraw(dw_class_info_add, FALSE)
SetRedraw(dw_class_info_update, FALSE)
Reset(dw_class_info_update)

 InsertRow(dw_class_info_update, 0)
ScrollToRow(dw_class_info_update, 1)

dw_class_info_add.visible       = TRUE
dw_class_info_update.visible    = FALSE
dw_class_info_add.enabled       = TRUE
dw_class_info_update.enabled    = FALSE

SetRedraw(dw_class_info_add, TRUE)
SetRedraw(dw_class_info_update, TRUE)

SetFocus(dw_class_info_add)
ELSE
SelectRow(dw_class_info_xref, 0, FALSE)
SelectRow(dw_class_info_xref, iv_i_Row, TRUE)

IF (ch_add.enabled) THEN
    SetRedraw(dw_class_info_add, FALSE)
dw_class_info_add.enabled = FALSE

    Reset(dw_class_info_add)
    InsertRow(dw_class_info_add, 0)
    ScrollToRow(dw_class_info_add, 1)

    SetRedraw(dw_class_info_add, TRUE)
END IF
Window: uo_class_maint
Library: e:\thesis\appl\schedule.pbl
Date: 5/2/95 Time: 16:37:45

//Command Button Control
cb_delete.enabled = TRUE
iv_cb_CurrentDefault.default = FALSE
cb_select.default = TRUE
iv_cb_CurrentDefault = cb_select
cb_add.enabled = FALSE
cb_reset_maint.enabled = TRUE
cb_select.enabled = TRUE
END IF

End of Script

Script for: doubleclicked event
//Gets the information from the doubleclicked row, sets it into class_info_update datawindow
//and structure passed to more class/lab information window.

//Local Variables
Integer  lv_i_CourseNumber
String   lv_s_SectionNumber
String   lv_s_Name
String   lv_s_Days
Time     lv_t_StartTime
Time     lv_t_EndTime
//End Local Variables

IF (lv_i_Row < 1) THEN
  SetFocus(dw_class_info_add)
  Return
END IF

//Should be taken care of by clicked event ....
IF (lv_b_ChangesMade) THEN
  IF {MessageBox("Changes Made", "Do you wish to save changes?", Question!, YesNo!) = 1} THEN
TriggerEvent(cb_update, "clicked")
ELSE
  iv_b_ChangesMade = FALSE
END IF
END IF

SetRedraw(dw_class_info_update, FALSE)
SetRedraw(dw_class_info_add, FALSE)

dw_class_info_update.enabled = TRUE
dw_class_info_add.enabled = FALSE

dw_class_info_update.visible = TRUE
dw_class_info_add.visible = FALSE

Reset(dw_class_info_add)
InsertRow(dw_class_info_add, 0)
ScrollToRow(dw_class_info_add, 1)

Reset(dw_class_info_update)
InsertRow(dw_class_info_update, 0)
ScrollToRow(dw_class_info_update, 1)

iv_i_UpdateRow = iv_i_Row

lv_i_CourseNumber = GetItemNumber(dw_class_info_xref, iv_i_UpdateRow, "course_num")
lv_s_SectionNumber = Trim(GetItemString(dw_class_info_xref, iv_i_UpdateRow, "section_num"))
lv_s_Name = Trim(GetItemString(dw_class_info_xref, iv_i_UpdateRow, "name"))
lv_s_Days = Trim(GetItemString(dw_class_info_xref, iv_i_UpdateRow, "days"))
lv_t_StartTime = GetItemTime(dw_class_info_xref, iv_i_UpdateRow, "start_time")
lv_t_EndTime = GetItemTime(dw_class_info_xref, iv_i_UpdateRow, "end_time")

SetItem(dw_class_info_update, 1, "course_number", lv_i_CourseNumber)
SetItem(dw_class_info_update, 1, "section_number", lv_s_SectionNumber)
SetItem(dw_class_info_update, 1, "name", lv_s_Name)
SetItem(dw_class_info_update, 1, "days", lv_s_Days)
Window: uo_class_maint
Library: e:\thesis\appl\schedule.pbl
Date: 5/2/95   Time: 16:37:45

SetItem(dw_class_info_update, 1, "start_time", lv_t_StartTime)
SetItem(dw_class_info_update, 1, "end_time", lv_t_EndTime)

s_InfoParms.add_mode = FALSE
s_InfoParms.first_time = FALSE
s_InfoParms.location = Trim(GetItemString(dw_class_info_xref, iv_i_UpdateRow, "location"))
s_InfoParms.max_credits = GetItemNumber(dw_class_info_xref, iv_i_UpdateRow, "max_credits")
s_InfoParms.min_credits = GetItemNumber(dw_class_info_xref, iv_i_UpdateRow, "min_credits")
s_InfoParms.building = Trim(GetItemString(dw_class_info_xref, iv_i_UpdateRow, "building"))
s_InfoParms.room = Trim(GetItemString(dw_class_info_xref, iv_i_UpdateRow, "room"))
s_InfoParms.class_limit = GetItemNumber(dw_class_info_xref, iv_i_UpdateRow, "class_limit")
s_InfoParms.flags = Trim(GetItemString(dw_class_info_xref, iv_i_UpdateRow, "flags"))
s_InfoParms.lab = Trim(GetItemString(dw_class_info_xref, iv_i_UpdateRow, "lab"))
s_InfoParms.lab_days = Trim(GetItemString(dw_class_info_xref, iv_i_UpdateRow, "lab_days"))
s_InfoParms.lab_start_time = GetItemTime(dw_class_info_xref, iv_i_UpdateRow, "lab_start_time")
s_InfoParms.lab_end_time = GetItemTime(dw_class_info_xref, iv_i_UpdateRow, "lab_end_time")
s_InfoParms.lab_location = Trim(GetItemString(dw_class_info_xref, iv_i_UpdateRow, "lab_location"))
s_InfoParms.lab_max_credits = GetItemNumber(dw_class_info_xref, iv_i_UpdateRow, "lab_max_credits")
s_InfoParms.lab_min_credits = GetItemNumber(dw_class_info_xref, iv_i_UpdateRow, "lab_min_credits")
s_InfoParms.lab_building = Trim(GetItemString(dw_class_info_xref, iv_i_UpdateRow, "lab_building"))
s_InfoParms.lab_room = Trim(GetItemString(dw_class_info_xref, iv_i_UpdateRow, "lab_room"))

SelectRow(dw_class_info_xref, 0, FALSE)
SelectRow(dw_class_info_xref, iv_i_UpdateRow, TRUE)

//Command Button Control
cb_more.enabled = TRUE
cb_add.enabled = FALSE
cb_delete.enabled = FALSE
cb_update.enabled = FALSE
cb_reset_maint.enabled = TRUE
iv_cb_CurrentDefault.default = FALSE
cb_reset_maint.default = TRUE
iv_cb_CurrentDefault = cb_reset_maint
cb_select.enabled = FALSE
iv_b_ChangesMade = FALSE
SetFocus(dw_class_info_update)
SetColumn(dw_class_info_update, "days")
SetRedraw(dw_class_info_add, TRUE)
SetRedraw(dw_class_info_update, TRUE)
End of Script

Line: ln_1
BeginX = 33
Visible = true
LineThickness = 5
BeginY = 361
EndX = 2743
EndY = 361
LineColor = 33554432
LineStyle = continuous!

Line: ln_2
BeginX = 33
Visible = true
LineThickness = 5
BeginY = 65
EndX = 2743
EndY = 65
LineColor = 33554432
LineStyle = continuous!

Line: ln_3
BeginX = 33
Visible = true
LineThickness = 5
BeginY = 69
EndX = 33
EndY = 365
LineColor = 33554432
LineStyle = continuous!

Line: ln_4
BeginX = 2739
Visible = true
LineThickness = 5
BeginY = 69
EndX = 2739
EndY = 365
LineColor = 33554432
LineStyle = continuous!
DataWindow: d_class_info_update
Library: e:\thesis\appl\schedule.pbl
Date: 5/3/95 Time: 23:24:17

Retrieve: Script
Arguments: None
Update Table: Not Allowed
Filter: None
Sort: None
Sparse: None
Column: days
  Format: "[general]"
  Border style: Shadow Box
  Validation: None
  Validation Message: None
  Tab Sequence: 10
  Initial Value: None
  Edit Style: Edit
  Edit limit: 0
Column: start_time
  Format: "[time]"
  Border style: Shadow Box
  Validation: None
  Validation Message: None
  Tab Sequence: 20
  Initial Value: None
  Edit Style: Edit
  Edit limit: 0
Column: end_time
DataWindow: d_class_info_update
Library: e:\thesis\appl\schedule.pbl
Date: 5/3/95    Time: 23:24:17

Border style: Shadow Box
Validation: None
Validation Message: None
Tab Sequence: 30
Initial Value: None
  Edit Style: Edit
  Edit limit: 0

Column: course_number
  Format: "[general]"
Border style: None
Validation: None
Validation Message: None
Tab Sequence: 0
Initial Value: None
Edit Style: Edit
Edit limit: 0

Column: section_number
  Format: "[general]"
Border style: None
Validation: None
Validation Message: None
Tab Sequence: 0
Initial Value: None
Edit Style: Edit
Edit limit: 0
<table>
<thead>
<tr>
<th>Course Number</th>
<th>Section Number</th>
<th>Name</th>
<th>Day(s)</th>
<th>Start Time</th>
<th>End Time</th>
<th>Location</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**Summary**

**Footer**
DataWindow: d_class_info  
Library: e:\thesis\apl\schedule.plb  
Date: 5/2/95    Time: 17:55:46  

Key: No  
Format: "[time]"  
Border style: None  
Validation: None  
Validation Message: None  
Tab Sequence: 0  
Initial Value: None  
Edit Style: Edit  
Edit limit: 0  

Column: end_time  
Updateable: Yes  
Key: No  
Format: "[time]"  
Border style: None  
Validation: None  
Validation Message: None  
Tab Sequence: 0  
Initial Value: None  
Edit Style: Edit  
Edit limit: 0  

Column: location  
Updateable: Yes  
Key: No  
Format: "[general]"  
Border style: None
DataWindow: d_class_info_add
Library: e:\thesis\appl\schedule.pbl
Date: 5/2/95   Time: 17:53:56

Retrieve: Script
Arguments: None
Update Table: Not Allowed
Filter: None
Sort: None
Sparse: None

Column: course_number
  Format: "[general]"
  Border style: Shadow Box
  Validation: None
  Validation Message: None
  Tab Sequence: 10
  Initial Value: None
  Edit Style: Edit
  Edit limit: 0

Column: section_number
  Format: "[general]"
  Border style: Shadow Box
  Validation: None
  Validation Message: None
  Tab Sequence: 20
  Initial Value: None
  Edit Style: Edit
  Edit limit: 0

Column: name
DataWindow: d_class_info_add
Library: e:\thesis\appl\schedule.pbl
Date: 5/2/95   Time: 17:53:56

Border style: Shadow Box
Validation: None
Validation Message: None
Tab Sequence: 30
Initial Value: None
Edit Style: Edit
Edit limit: 0

Column: days
Format: "[general]"
Border style: Shadow Box
Validation: None
Validation Message: None
Tab Sequence: 40
Initial Value: None
Edit Style: Edit
Edit limit: 0

Column: start_time
Format: "[time]"
Border style: Shadow Box
Validation: None
Validation Message: None
Tab Sequence: 50
Initial Value: None
Edit Style: Edit
Edit limit: 0
Window: w_more_class_info
Library: e:\thesis\appl\schedule.pbl
Date: 5/1/95    Time: 21:26:05

Window: w_more_class_info
X = 1079   Y = 489   Width = 2346   Height = 1529
Visible = true   Enabled = true   TitleBar = true   Title = "Class Information"
ControlMenu = true   MinBox = true   MaxBox = true   Resizable = true
WindowType = main!   WindowState = normal!   BackColor = 12632256
Instance Variables
s_class_infoParms  s_parms

End of Instance Variables

Script for: open  event
//Adds a blank row into the two datawindows. If in update mode, sets data into the datawindows
//from the information in the structures. If in add mode, the lab datawindow is hidden.

//Local Variables
Boolean   lv_b_AddMode
Boolean   lv_b_FirstTime
String    lv_s_Lab
//End Local Variables

s_parms = Message.PowerObjectParm

Reset(dw_class_info)
InsertRow(dw_class_info, 0)
ScrollToRow(dw_class_info, 1)

Reset(dw_lab_info)
InsertRow(dw_lab_info, 0)
ScrollToRow(dw_lab_info, 1)

lv_b_AddMode  =  s_parms.add_mode
lv_b_FirstTime =  s_parms.first_time

IF ((NOT lv_b_AddMode) OR (NOT lv_b_FirstTime)) THEN

  //Set the class info dw
  SetItem(dw_class_info, 1, "location", s_parms.location)
  SetItem(dw_class_info, 1, "max_credits", s_parms.max_credits)
Window: w_more_class_info
Library: e:\thesis\appl\schedule.pbl
Date: 5/1/95   Time: 21:26:05

setItem(dw_class_info, 1, "min_credits", sparms.min_credits)
setItem(dw_class_info, 1, "building", sparms.building)
setItem(dw_class_info, 1, "room", sparms.room)
setItem(dw_class_info, 1, "class_limit", sparms.class_limit)
setItem(dw_class_info, 1, "flags", sparms.flags)
setItem(dw_class_info, 1, "lab", sparms.lab)

//Check to see if the lab info needs to be filled.
lv_s_Lab = trim(sparms.lab)

IF ((lv_s_Lab = "Y") OR (lv_s_Lab = "y")) THEN
  dw_lab_info.visible = TRUE
  dw_lab_info.enabled = TRUE

  setItem(dw_lab_info, 1, "lab_days", sparms.lab_days)
  setItem(dw_lab_info, 1, "lab_start_time", sparms.lab_start_time)
  setItem(dw_lab_info, 1, "lab_end_time", sparms.lab_end_time)
  setItem(dw_lab_info, 1, "lab_location", sparms.lab_location)
  setItem(dw_lab_info, 1, "lab_max_credits", sparms.lab_max_credits)
  setItem(dw_lab_info, 1, "lab_min_credits", sparms.lab_min_credits)
  setItem(dw_lab_info, 1, "lab_building", sparms.lab_building)
  setItem(dw_lab_info, 1, "lab_room", sparms.lab_room)
END IF
END IF

SetFocus(dw_class_info)
SetColumn(dw_class_info, "location")

End of Script

CommandButton: cb_ok
X = 1953   Y = 1261   Width = 247   Height = 109
TabOrder = 40   Visible = true   Enabled = true   Text = "&OK"
Script for: clicked event
//Closes the window w_more_class_info after storing the new or modified data into the structure
//that is returned to the command button more to be processed.

//Local Variables
Integer  lv_i_MaxCredits
Integer  lv_i_MinCredits
Integer  lv_i_ClassLimit
Integer  lv_i_LabMaxCredits
Integer  lv_i_LabMinCredits
String   lv_s_Locaton
String   lv_s_Building
String   lv_s_Room
String   lv_s_Flags
String   lv_s_Lab
String   lv_s_LabDays
String   lv_s_LabLocation
String   lv_s_LabBuilding
String   lv_s_LabRoom
Time     lv_t_LabStartTime
Time     lv_t_LabEndTime

lv_s_Location = Trim(GetItemString(dw_class_info, 1, "location"))
lv_i_MaxCredits = GetItemNumber(dw_class_info, 1, "max_credits")
lv_i_MinCredits = GetItemNumber(dw_class_info, 1, "min_credits")
lv_s_Building = Trim(GetItemString(dw_class_info, 1, "building"))
lv_s_Room = Trim(GetItemString(dw_class_info, 1, "room"))
lv_i_ClassLimit = GetItemNumber(dw_class_info, 1, "class_limit")
lv_s_Flags = Trim(GetItemString(dw_class_info, 1, "flags"))
lv_s_Lab = Trim(GetItemString(dw_class_info, 1, "lab"))

IF ((lv_s_Lab = "Y") OR (lv_s_Lab = "y")) THEN
   lv_s_LabDays = Trim(GetItemString(dw_lab_info, 1, "lab_days"))
lv_t_LabStartTime = GetItemTime(dw_lab_info, 1, "lab_start_time")
lv_t_LabEndTime = GetItemTime(dw_lab_info, 1, "lab_end_time")
lv_s_LabLocation = Trim(GetItemString(dw_lab_info, 1, "lab_location"))
lv_i_LabMaxCredits = GetItemNumber(dw_lab_info, 1, "lab_max_credits")
lv_i_LabMinCredits = GetItemNumber(dw_lab_info, 1, "lab_min_credits")
lv_s_LabBuilding = Trim(GetItemString(dw_lab_info, 1, "lab_building"))
lv_s_LabRoom = Trim(GetItemString(dw_lab_info, 1, "lab_room"))

END IF

//Validation
IF (IsNull(lv_s_Location) OR (lv_s_Location = "")) THEN
    MessageBox("Validation Error", "A location must be entered before leaving.", StopSign!, OK!)
    SetFocus(dw_class_info)
    SetColumn(dw_class_info, "location")
    Return
ELSEIF (IsNull(lv_i_MaxCredits)) THEN
    MessageBox("Validation Error", "The maximum credits must be entered before leaving.", StopSign!, OK!)
    SetFocus(dw_class_info)
    SetColumn(dw_class_info, "max_credits")
    Return
ELSEIF (IsNull(lv_i_MinCredits)) THEN
    MessageBox("Validation Error", "The minimum credits must be entered before leaving.", StopSign!, OK!)
    SetFocus(dw_class_info)
    SetColumn(dw_class_info, "min_credits")
    Return
ELSEIF (lv_i_MinCredits > lv_i_MaxCredits) THEN
    MessageBox("Validation Error", "The minimum credits cannot be greater than the maximum credits.", StopSign!, OK!)
    SetFocus(dw_class_info)
    SetColumn(dw_class_info, "min_credits")
    Return
ELSEIF (IsNull(lv_s_Building) OR (lv_s_Building = "")) THEN
    MessageBox("Validation Error", "The building must be entered before leaving.", StopSign!, OK!)
    SetFocus(dw_class_info)
    SetColumn(dw_class_info, "building")
    Return
ELSEIF (IsNull(lv_s_Room) OR (lv_s_Room = "")) THEN
    MessageBox("Validation Error", "The room must be entered before leaving.", StopSign!, OK!)
    SetFocus(dw_class_info)
    SetColumn(dw_class_info, "room")
    Return
ELSEIF (IsNull(lv_i_ClassLimit)) THEN
    MessageBox("Validation Error", "The class limit must be entered before leaving.", StopSign!, OK!)
    SetFocus(dw_class_info)
    SetColumn(dw_class_info, "class_limit")
    Return
ELSEIF (IsNull(lv_s_Lab) OR (lv_s_Lab = "")) THEN
    IF (MessageBox("Information", "Because there is no data in the lab question, there will be no scheduled lab with the course.", Information!, OkCancel!) = 1) THEN
        SetItem(dw_class_info, 1, "lab", "N")
        lv_s_Lab = "N"
    ELSE
        SetItem(dw_class_info, 1, "lab", "Y")
        lv_s_Lab = "Y"
        dw_lab_info.visible = TRUE
        SetFocus(dw_lab_info)
        SetColumn(dw_lab_info, "lab_days")
    Return
END IF
END IF

IF ((lv_s_Lab = "Y") OR (lv_s_Lab = "y")) THEN
    IF (IsNull(lv_s_LabDays) OR (lv_s_LabDays = "")) THEN
        MessageBox("Validation Error", "Lab days must be entered before leaving.", StopSign!, OK!)
        SetFocus(dw_lab_info)
        SetColumn(dw_lab_info, "lab_days")
        Return
    ELSEIF (IsNull(lv_t_LabStartTime)) THEN
        MessageBox("Validation Error", "A lab start time must be entered before leaving.", StopSign!, OK!)
        SetFocus(dw_lab_info)
        SetColumn(dw_lab_info, "lab_start_time")
        Return
ELSEIF (IsNull(lv_t_LabEndTime)) THEN
    MessageBox("Validation Error", "A lab end time must be entered before leaving.", StopSign!, OK!)
    SetFocus(dw_lab_info)
    SetColumn(dw_lab_info, "lab_end_time")
    Return
ELSEIF (lv_t_LabStartTime > lv_t_LabEndTime) THEN
    MessageBox("Validation Error", "The lab start time cannot be later than the lab end time.", StopSign!, OK!)
    SetFocus(dw_lab_info)
    SetColumn(dw_lab_info, "lab_start_time")
    Return
ELSEIF (IsNull(lv_s_LabLocation) OR (lv_s_LabLocation = "")) THEN
    MessageBox("Validation Error", "The lab location must be entered before leaving.", StopSign!, OK!)
    SetFocus(dw_lab_info)
    SetColumn(dw_lab_info, "lab_location")
    Return
ELSEIF (IsNull(lv_i_LabMaxCredits)) THEN
    MessageBox("Validation Error", "The lab maximum credits must be entered before leaving.", StopSign!, OK!)
    SetFocus(dw_lab_info)
    SetColumn(dw_lab_info, "lab_max_credits")
    Return
ELSEIF (IsNull(lv_i_LabMinCredits)) THEN
    MessageBox("Validation Error", "The lab minimum credits must be entered before leaving.", StopSign!, OK!)
    SetFocus(dw_lab_info)
    SetColumn(dw_lab_info, "lab_min_credits")
    Return
ELSEIF (lv_i_LabMinCredits > lv_i_LabMaxCredits) THEN
    MessageBox("Validation Error", "The lab minimum credits cannot be greater than the lab maximum credits.", StopSign!, OK!)
    SetFocus(dw_lab_info)
    SetColumn(dw_lab_info, "lab_max_credits")
    Return
ELSEIF (IsNull(lv_s_LabBuilding) OR (lv_s_LabBuilding = "")) THEN
    MessageBox("Validation Error", "The lab building must be entered before leaving.", StopSign!, OK!)
    SetFocus(dw_lab_info)
    SetColumn(dw_lab_info, "lab_building")
    Return
Window: w_more_class_info
Library: e:\thesis\appl\schedule.pbl
Date: 5/1/95     Time: 21:26:05

(getActivity(dw_lab_info))
SetFocus(dw_lab_info)
SetColumn(dw_lab_info, "lab_building")
If s_cim == 2 THEN
   MsgBox("Validation Error", "The lab room must be entered before leaving.", StopSign!, OK!)
   SetFocus(dw_lab_info)
   SetColumn(dw_lab_info, "lab_room")
Exit
ELSEIF IsNull(lv_s_LabRoom) OR (lv_s_LabRoom = ") THEN
   MsgBox("Validation Error", "The lab room must be entered before leaving.", StopSign!, OK!)
   SetFocus(dw_lab_info)
   SetColumn(dw_lab_info, "lab_room")
Exit
END IF

s_parms.first_time = FALSE
s_parms.location = lv_s_Location
s_parms.max_credits = lv_i_MaxCredits
s_parms.min_credits = lv_i_MinCredits
s_parms.building = lv_s_Building
s_parms.room = lv_s_Room
s_parms.class_limit = lv_i_ClassLimit
s_parms.flags = lv_s_FLAGS
s_parms.lab = lv_s_Lab
s_parms.lab_days = lv_s_LabDays
s_parms.lab_start_time = lv_t_LabStart
s_parms.lab_end_time = lv_t_LabEnd
s_parms.lab_location = lv_s_LabLocation
s_parms.lab_max_credits = lv_i_LabMaxCredits
s_parms.lab_min_credits = lv_i_LabMinCredits
s_parms.lab_building = lv_s_LabBuilding
s_parms.lab_room = lv_s_LabRoom

PostEvent(uo_class_maint.cb_more, "uponreturn")

CloseWithReturn(w_more_class_info, s_parms)

End of Script
CommandButton: cb_cancel
X = 1605    Y = 1261    Width = 247    Height = 109
TabOrder = 30  Visible = true  Enabled = true  Text = "&Cancel"
Default = true

Script for: clicked  event
//Closes window w_more_class_info without setting the information into the structure to be returned.

close(w_more_class_info)

End of Script

DataWindow: dw_lab_info
X = 106    Y = 533    Width = 1898    Height = 653
TabOrder = 20  DataObject = "d_lab_info"  Border = true
LiveScroll = true  BorderStyle = stylebox!

DataWindow: dw_class_info
X = 46    Y = 65    Width = 2163    Height = 437
TabOrder = 10  Visible = true  Enabled = true  DataObject = "d_more_class_info"
LiveScroll = true  BorderStyle = stylebox!

Script for: dwnkey  event
//Used to determine whether the lab info datawindow should be displayed.
//The datawindow is displayed only if the field lab is y.

//Local Variables
String  lv_s_CurrentColumn
String  lv_s_Lab
//End Local Variables
Window: w_more_class_info
Library: e:\thesis\appl\schedule.pbl
Date: 5/1/95   Time: 21:26:05

lv_s_CurrentColumn = GetColumnName(dw_class_info)

IF ((lv_s_CurrentColumn = "lab") AND (NOT dw_lab_info.visible)) THEN
  AcceptText(dw_class_info)
  lv_s_Lab = GetItemString(dw_class_info, 1, "lab")
  IF ((lv_s_Lab = "Y") OR (lv_s_Lab = "y")) THEN
    dw_lab_info.visible = TRUE
    dw_lab_info.enabled = TRUE
  END IF
END IF

IF ((KeyDown(KeyTab!)) AND (KeyDown(KeyShift!))) THEN
  SetColumn(dw_class_info, "flags")
END IF

IF (KeyDown(KeyTab!)) THEN
  SetFocus(dw_lab_info)
  SetColumn(dw_lab_info, "lab_days")
END IF
END IF

End of Script
<table>
<thead>
<tr>
<th>Header</th>
<th>Max Credits</th>
<th>Min Credits</th>
<th>Building</th>
<th>Room</th>
<th>Class Limit</th>
</tr>
</thead>
<tbody>
<tr>
<td>location</td>
<td>max_credits</td>
<td>min_credits</td>
<td>building</td>
<td>room</td>
<td>class_limit</td>
</tr>
<tr>
<td>Flags</td>
<td>Is there a scheduled lab with the class?</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>flags</td>
<td>lab</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Summary</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Footer</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
DataWindow: d_more_class_info
Library: e:\thesis\appl\schedule.pbl
Date: 5/2/95   Time: 17:34:26

Retrieve: Script
Arguments: None
Update Table: Not Allowed
Filter: None
Sort: None
Sparse: None

Column: flags
  Format: "[general]"
  Border style: Shadow Box
  Validation: None
  Validation Message: None
  Tab Sequence: 70
  Initial Value: None
  Edit Style: Edit
  Edit limit: 0

Column: location
  Format: "[general]"
  Border style: Shadow Box
  Validation: None
  Validation Message: None
  Tab Sequence: 10
  Initial Value: None
  Edit Style: Edit
  Edit limit: 0

Column: max_credits
Border style: Shadow Box
Validation: None
Validation Message: None
Tab Sequence: 20
Initial Value: None
Edit Style: Edit
Edit limit: 0

Column: min Credits
Format: "[general]"
Border style: Shadow Box
Validation: None
Validation Message: None
Tab Sequence: 30
Initial Value: None
Edit Style: Edit
Edit limit: 0

Column: building
Format: "[general]"
Border style: Shadow Box
Validation: None
Validation Message: None
Tab Sequence: 40
Initial Value: None
Edit Style: Edit
Edit limit: 0
DataWindow: d_more_class_info
Library: e:\thesis\appl\schedule.pbl
Date: 5/2/95  Time: 17:34:26

Format: "[general]"
Border style: Shadow Box
Validation: None
Validation Message: None
Tab Sequence: 50
Initial Value: None
Edit Style: Edit
Edit limit: 0

Column: class_limit
Format: "[general]"
Border style: Shadow Box
Validation: None
Validation Message: None
Tab Sequence: 60
Initial Value: None
Edit Style: Edit
Edit limit: 0

Column: lab
Format: "[general]"
Border style: Shadow Box
Validation: None
Validation Message: None
Tab Sequence: 80
Initial Value: None
Edit Style: Edit
DataWindow: d_lab_info
Library: e:\thesis\appl\schedule.pbl
Date: 5/2/95 Time: 17:36:07

Retrieve: Script
Arguments: None
Update Table: Not Allowed
Filter: None
Sort: None
Sparse: None

Column: lab_days
   Format: \"[general]\"
   Border style: Shadow Box
   Validation: None
   Validation Message: None
   Tab Sequence: 10
   Initial Value: None
   Edit Style: Edit
   Edit limit: 0

Column: lab_start_time
   Format: \"[time]\"
   Border style: Shadow Box
   Validation: None
   Validation Message: None
   Tab Sequence: 20
   Initial Value: None
   Edit Style: Edit
   Edit limit: 0

Column: lab_end_time
Border style: Shadow Box
Validation: None
Validation Message: None
Tab Sequence: 30
Initial Value: None
  Edit Style: Edit
  Edit limit: 0
Column: lab_location
  Format: "[general]"
  Border style: Shadow Box
  Validation: None
  Validation Message: None
  Tab Sequence: 40
  Initial Value: None
  Edit Style: Edit
  Edit limit: 0
Column: lab_max_credits
  Format: "[general]"
  Border style: Shadow Box
  Validation: None
  Validation Message: None
  Tab Sequence: 50
  Initial Value: None
  Edit Style: Edit
  Edit limit: 0
DataWindow: d_lab_info
Library: e:\thesis\appl\schedule.pbl
Date: 5/2/95  Time: 17:36:07

Format: "{general}"  
Border style: Shadow Box  
Validation: None  
Validation Message: None  
Tab Sequence: 60  
Initial Value: None  
Edit Style: Edit  
Edit limit: 0

Column: lab_building  
Format: "{general}"  
Border style: Shadow Box  
Validation: None  
Validation Message: None  
Tab Sequence: 70  
Initial Value: None  
Edit Style: Edit  
Edit limit: 0

Column: lab_room  
Format: "{general}"  
Border style: Shadow Box  
Validation: None  
Validation Message: None  
Tab Sequence: 80  
Initial Value: None  
Edit Style: Edit
User Object: uo_prof_main

Library: e:\thesis\appl\schedule.pbl
Date: 5/2/95    Time: 16:30:26

User Object: uo_prof_main
X = 0          Y = 0          Width = 2812   Height = 1473
TabOrder = 0   Visible = true  Enabled = true  BackColor = 12632256
ObjectType = customvisual!

Instance Variables
DataWindowChild  iv_dwc_Faculty
DataWindowChild  iv_dwc_Department

Boolean          iv_b_SelectionMade
Boolean          iv_b_TabDone
Boolean          iv_b_UpDownArrowDone
End of Instance Variables

Script for: constructor event

//Retrieves all data used in the datawindows in the user object.

// Local Variables

// End Local Variables

dwGetChild(dw_faculty, "name_combined", iv_dwc_faculty)
dwGetChild(dw_department, "department", iv_dwc_department)

SetTransObject(iv_dwc_faculty, SQLCA)
SetTransObject(iv_dwc_department, SQLCA)
SetTransObject(dw_faculty, SQLCA)
SetTransObject(dw_department, SQLCA)
SetTransObject(dw_prof_xref, SQLCA)

Retrieve(dw_faculty)
Retrieve(dw_department)
Retrieve(dw_prof_xref)

SetSort(dw_faculty, "name_combined A")
SetSort(iv_dwc_faculty, "name_combined A")
SetSort(dw_department, "department A")
SetSort(iv_dwc_department, "department A")

Sort(dw_faculty)
Sort(iv_dwc_faculty)
Sort(dw_department)
Sort(iv_dwc_department)

SetSort(dw_prof_xref, "department A, name A")
User Object: uo_prof_main
Library: e:\thesis\appl\schedule.pbl
Date: 5/2/95     Time: 16:30:26

Sort(dw_prof_xref)

InsertRow(dw_faculty, 1)
InsertRow(dw_department, 1)

IF NOT (IsNull(gv_struct_parms.term) OR (gv_struct_parms.term = "")) THEN
  IF NOT (IsNull(gv_struct_parms.department) AND (gv_struct_parms.department = "")) THEN
    cb_scheduling.enabled = TRUE
  END IF
END IF

iv_b_SelectionMade = FALSE

SetFocus(dw_department)

End of Script

Script for: other event
//Used to attempt to control tabbing.

//Local Variables
CommandButton   lv_cb_which
DataWindow       lv_dw_which
GraphicObject    lv_go_WhichControl
String           lv_s_TextValue
String           lv_s_DataObject
//End Local Variables

IF (KeyDown(KeyTab!) AND KeyDown(KeyShift!)) THEN
  IF (iv_b_TabDone) THEN
    iv_b_TabDone = FALSE
  // not sure how to explain the color setting ... trust me ... it needs to be there for now.
    st_1.textcolor = RGB(0, 0, 128)
    st_2.textcolor = RGB(0, 0, 128)
Return
END IF

lv_go_WhichControl = GetFocus()

choose case TypeOf(lv_go_WhichControl)

  case CommandButton!
  . lv_cb_which = lv_go_WhichControl
  . lv_s_TextValue = lv_cb_which.text
  IF (lv_s_TextValue = "&Scheduling ...") THEN
    SetFocus(dw_faculty)
    Return
  ELSEIF (lv_s_TextValue = "&Add Department ...") THEN
    IF (cb_scheduling.enabled) THEN
      SetFocus(cb_scheduling)
      Return
    ELSE
      SetFocus(dw_faculty)
      Return
    END IF
  ELSEIF (lv_s_TextValue = "&Reset") THEN
    SetFocus(Cb_add)
    Return
  ELSEIF (lv_s_TextValue = "&Edit ...") THEN
    IF (cb_reset.enabled) THEN
      SetFocus(cb_reset)
      Return
    ELSE
      SetFocus(cb_add)
      Return
    END IF
  ELSEIF (lv_s_TextValue = "&OK") THEN
    IF (cb_edit.enabled) THEN
      SetFocus(cb_edit)
      Return
  END IF
User Object: uc_prof_main
Library: e:\thesis\appl\schedule.pbl
Date: 5/2/95    Time: 16:30:26

ELSEIF (cb_reset.enabled) THEN
    SetFocus(cb_reset)
    Return
ELSE
    SetFocus(cb_add)
    Return
END IF
ELSE
    Return
END IF

case DataWindow!
    lv_dw_which = lv_go_WhichControl

    lv_s_DataObject = lv_dw_which.dataobject

    IF (lv_s_DataObject = "d_department_dddw") THEN
        SetFocus(cb_ok)
        Return
    ELSEIF (lv_s_DataObject = "d_professor_dddw") THEN
        SetFocus(dw_department)
        Return
    ELSEIF (lv_s_DataObject = "d_prof_main_xref") THEN
        SetFocus(dw_faculty)
        Return
    ELSE
        Return
    END IF

case else
    Return
END Choose
ELSEIF (KeyDown(KeyTab!!)) THEN
    IF (lv_b_TabDone) THEN
        lv_b_TabDone = FALSE
    
    // not sure how to explain the color setting ... trust me ... it needs to be there for now.
User Object: uo_prof_main
Library: e:\thesis\appl\schedule.pbl
Date: 5/2/95      Time: 16:30:26

    st_1.textcolor = RGB(0, 0, 128)
st_2.textcolor = RGB(0, 0, 128)
    Return
END IF

lv_go_WhichControl = GetFocus()

choose case TypeOf(lv_go_WhichControl)

'case CommandButton!
    lv_cb_which = lv_go_WhichControl
    lv_s_TextValue = lv_cb_which.text
    IF (lv_s_TextValue = "&Scheduling ...") THEN
        SetFocus(cb_add)
        Return
    ELSEIF (lv_s_TextValue = "&Add Department ...") THEN
        IF (cb_reset.enabled) THEN
            SetFocus(cb_reset)
            Return
        ELSEIF (cb_edit.enabled) THEN
            SetFocus(cb_edit)
            Return
        ELSE
            SetFocus(cb_ok)
            Return
        END IF
    ELSEIF (lv_s_TextValue = "&Reset") THEN
        IF (cb_edit.enabled) THEN
            SetFocus(cb_edit)
            Return
        ELSE
            SetFocus(cb_ok)
            Return
        END IF
    ELSEIF (lv_s_TextValue = "&Edit ...") THEN
        SetFocus(cb_ok)
Return
ELSEIF (lv_s_TextValue = "&OK") THEN
  SetFocus(dw_department)
  Return
ELSE
  Return
END IF

  case DataWindow!
    lv_dw_Which = lv_go_WhichControl

    lv_s_DataObject = lv_dw_Which.dataobject

    IF (lv_s_DataObject = "d_department_ddddw") THEN
      SetFocus(dw_faculty)
      Return
    ELSEIF (lv_s_DataObject = "d_professor_ddddw") THEN
      IF (cb_scheduling.enabled) THEN
        SetFocus(cb_scheduling)
        Return
      ELSE
        SetFocus(cb_add)
        Return
      END IF
    ELSEIF (lv_s_DataObject = "d_class_xref") THEN
      IF (cb_scheduling.enabled) THEN
        SetFocus(cb_scheduling)
        Return
      ELSE
        SetFocus(cb_add)
        Return
      END IF
    ELSE
      Return
    END IF
  case else
User Object: uo_prof_main
Library: e:\thesis\appl\schedule.pbl
Date: 5/2/95 Time: 16:30:26

    Return
    END Choose
    END IF

End of Script

Script for: getcontrolagain event
//Called from the close of uo_prof_maint, used to re-retrieve data after changes have been made.

//Local Variables

//End Local Variables

Retrieve(dw_faculty)
Retrieve(dw_department)
Retrieve(dw_prof_xref)

SetSort(dw_faculty, "name_combined A")
SetSort(iv_dwc_faculty, "name_combined A")
SetSort(dw_department, "department A")
SetSort(iv_dwc_department, "department A")

Sort(dw_faculty)
Sort(iv_dwc_faculty)
Sort(dw_department)
Sort(iv_dwc_department)

SetSort(dw_prof_xref, "department A, last_name A, first_name A")
Sort(dw_prof_xref)

InsertRow(dw_faculty, 1)
InsertRow(dw_department, 1)
User Object: uo_prof_main
Library: e:\thesis\appl\schedule.pbl
Date: 5/2/95   Time: 16:30:26

SelectRow(dw_prof_xref, 0, FALSE)

//Reset iv_structParms
gv_struct_parms.term = ""
gv_struct_parms.department = ""
 gv_struct_parms.course_number = 0
gv_struct_parms.section_number = ""
gv_struct_parms.faculty_member = ""

cb_reset.enabled = FALSE
cb_edit.enabled = FALSE
cb_scheduling.enabled = FALSE

iv_b_SelectionMade = FALSE

SetFocus(dw_department)

End of Script

DataWindow: dw_department
X = 403   Y = 85   Width = 494   Height = 93
TabOrder = 10   Visible = true   Enabled = true   DataObject = "d_department_dddw"
LiveScroll = true   BorderStyle = stylebox

Script for: itemchanged  event
//Selects a row in the dw_prof_xref if a faculty member is selected in the ddddw.

//Local Variables
Integer   lv_i_Row
Integer   lv_i_RowFound
String    lv_s_FacultyMember
String    lv_s_FilterString
String    lv_s_FindString
String lv_s_Department
//End Local Variables

lv_s_FacultyMember = gv_structParms.faculty_member
lv_i_Row = GetSelectedRow(iv_dwc_department, 0)

IF (lv_i_Row < 1) THEN
   Return
END IF

lv_s_Department = Trim(GetItemString(iv_dwc_department, lv_i_Row, "department"))
gv_structParms.department = lv_s_Department

cb_reset.enabled = TRUE

//Does the find string need to account for the fact that there is a term selected.  
IF (NOT (IsNull(lv_s_FacultyMember) OR (lv_s_FacultyMember = ""))) THEN
   SelectRow(dw_prof_xref, 0, FALSE)
   lv_s_FindString = "department = " + '"' + lv_s_Department + '"' &  
        + "and name = " + '"' + lv_s_FacultyMember + '"

   lv_i_RowFound = dwFind(dw_prof_xref, lv_s_FindString, 0, RowCount(dw_prof_xref))

   IF (lv_i_RowFound > 0) THEN
      SetRedraw(dw_prof_xref, FALSE)
      SelectRow(dw_prof_xref, lv_i_RowFound, TRUE)
      ScrollToRow(dw_prof_xref, lv_i_RowFound)
      SetRedraw(dw_prof_xref, TRUE)
      lv_b_SelectionMade = TRUE
   END IF

END IF

cb_edit.enabled = TRUE
End of Script

Script for: updateend  event
//Triggered when a new department is added in the w_add_dept window.

//Local Variables
Integer  lv_i_InsertedRow
String   lv_s_FacultyMember
String   lv_s_Department
String   lv_s_FindString
//End Local Variables

SetRedraw(dw_prof_xref, FALSE)
SetRedraw(dw_faculty, FALSE)
SetRedraw(dw_department, FALSE)

SelectRow(dw_prof_xref, 0, FALSE)

lv_s_Department = gv_struct_parms.department

lv_s_FindString = "department = " + `'" + lv_s_Department + `'"

IF (dwFind(iv_dwc_department, lv_s_FindString, 1, RowCount(iv_dwc_department)) = 0) THEN
lv_i_InsertedRow = InsertRow(iv_dwc_department, 0)
SetItem(iv_dwc_department, lv_i_InsertedRow, "department", lv_s_Department)

SetSort(dw_department, "department A")
SetSort(iv_dwc_department, "department A")

Sort(dw_department)
Sort(iv_dwc_department)
END IF
Window: uo_prof_main
Library: e:\thesis\appl\schedule.pbl
Date: 5/2/95    Time: 16:30:26

InsertRow(dw_department, 1)

SetItem(dw_department, 1, "department", lv_s_Department)
ScrollToRow(dw_department, 1)

lv_s_FacultyMember = Trim(GetItemString(dw_faculty, 1, "name_combined"))

IF ((IsNull(lv_s_FacultyMember)) OR (lv_s_FacultyMember = ")) THEN
  cb_edit.enabled = FALSE
ELSE
  cb_edit.enabled = TRUE
  cb_edit.default = TRUE
  cb_scheduling.default = FALSE
  cb_add.default = FALSE
  cb_reset.default = FALSE
  cb_ok.default = FALSE
END IF

SetRedraw(dw_prof_xref, TRUE)
SetRedraw(dw_faculty, TRUE)
SetRedraw(dw_department, TRUE)

End of Script

Script for: dwnkey_event
//drop down datawindows do strange things on tabs so the following overrides tab and sets focus where it
should be.

IF (KeyDown(keyTab!) AND KeyDown(keyShift!)) THEN
  SetFocus(cb_ok)
  SetActionCode(dw_department, 1)
  lv_b_TabDone = TRUE
// Don't know how to explain how this helps with tabbing but it does.
  st_2.textcolor = RGB(255, 0, 0)
Window: uo_prof_main
Library: e:\thesis\appl\schedule.pbl
Date: 5/2/95   Time: 16:30:26

Return
ELSEIF (KeyDown(keyTab)) THEN
    SetFocus(dw_faculty)
    SetActionCode(dw_department, 1)
    lv_b_TabDone = TRUE
// Don't know how to explain how this helps with tabbing but it does.
    st_2.textColor = RGB(0, 255, 0)
    Return
END IF

End of Script

CommandButton: cb_reset
X = 1500   Y = 1325   Width = 289   Height = 109
TabOrder = 0   Visible = true   Text = "&Reset"

Script for: clicked event
//Resets the data to what it looked like before rows were selected.

//@Local Variables
Integer   lv_i_Row
//@End Local Variables

SetRedraw(dw_prof_xref, FALSE)
SetRedraw(dw_faculty, FALSE)
SetRedraw(dw_department, FALSE)

SelectRow(dw_prof_xref, 0, FALSE)
ScrollToRow(dw_prof_xref, 1)

lv_i_Row = dwFind(dw_faculty, "name_combined = " + "'" + ", 0, RowCount(dw_faculty)
IF (lv_i_Row = 0) THEN
    InsertRow(dw_faculty, 1)
    ScrollToRow(dw_faculty, 1)
Window: uo_prof_main
Library: e:\thesis\appl\schedule.pbl
Date: 5/2/95   Time: 16:30:26

ELSE
   ScrollToRow(dw_facyulty, lv_i_Row)
END IF

lv_i_Row = dwFind(dw_department, "department = " + '#' + ",", 0, RowCount(dw_department))
IF (lv_i_Row = 0) THEN
   InsertRow(dw_department, 1)
   ScrollToRow(dw_department, 1)
ELSE
   ScrollToRow(dw_department, lv_i_Row)
END IF

SetFocus(dw_department)

SetRedraw(dw_prof_xref, TRUE)
SetRedraw(dw_facyulty, TRUE)
SetRedraw(dw_department, TRUE)

gv_struct_parms.term = ""
gv_struct_parms.department = ""

//Command Button Control
cb_reset.enabled = FALSE
cb_scheduling.enabled = FALSE
cb_edit.enabled = FALSE

End of Script

CommandButton: cb_ok
X = 2350   Y = 1325   Width = 289   Height = 109
TabOrder = 0   Visible = true   Enabled = true   Text = "&OK"

Script for: clicked  event
//Local Variables

//End Local Variables

close (w_class_main_test)

End of Script

CommandButton: cb_edit
X = 1925      Y = 1325      Width = 289      Height = 109
TabOrder = 0   Visible = true   Text = "&Edit ..."

Script for: clicked event
//Opens uo_prof_maint when a department is selected (a faculty member may also be selected but not required).

//Local Variables
Integer        lv_i_SelectedRow
s_main_win_parms  s_parms
//End Local Variables

s_parms.Department  =  gv_struct_parms.department
s_parms.Faculty_Member =  gv_struct_parms.faculty_member
OpenWithParm(w_maint_main, s_parms)

End of Script

CommandButton: cb_add
X = 750      Y = 1325      Width = 613      Height = 109
TabOrder = 0   Visible = true   Enabled = true   Text = "&Add Department ..."
Script for: clicked event
//Opens w_add_dept so that a new department can be added.

//Local Variables

//End Local Variables
Open(w_add_dept)
End of Script

CommandButton: cb_scheduling
X = 142          Y = 1325          Width = 471          Height = 109
TabOrder = 0        Visible = true      Text = "&Scheduling ...

Script for: clicked event
//Opens the w_scheduling window with the department and term already selected.
OpenWithParm(w_scheduling, gv_struct_parms)
End of Script

DataWindow: dw_prof_xref
X = 42          Y = 265          Width = 2725          Height = 989
TabOrder = 0        Visible = true      Enabled = true    DataObject = "d_prof_main_xref"
VScrollBar = true    Border = true      LiveScroll = true    BorderStyle = stylebox!

Script for: clicked event
//Selects the clicked row and sets the faculty member and department in the drop down datawindows.
// Local Variables

Integer lv_i_SelectedRow
Integer lv_i_ClickedRow
Integer lv_i_FindRow
String lv_s_Department
String lv_s_FacultyMember
String lv_s_FindFacultyMember
String lv_s_FindDepartment
String lv_s_FindCourseSectionNum
String lv_s_FilterString

// End Local Variables

lv_i_SelectedRow = GetSelectedRow(dw_prof_xref, 0)
lv_i_ClickedRow = GetClickedRow(dw_prof_xref)

SetRedraw(dw_prof_xref, FALSE)

IF (lv_i_SelectedRow = lv_i_ClickedRow) THEN
  SelectRow(dw_prof_xref, lv_i_ClickedRow, FALSE)
  gv_struct_parms.department = ""
  gv_struct_parms.faculty_member = ""
  cb_reset.enabled = FALSE
  cb_edit.enabled = FALSE
ELSE
  SelectRow(dw_prof_xref, lv_i_SelectedRow, FALSE)
  SelectRow(dw_prof_xref, lv_i_ClickedRow, TRUE)
  lv_s_Department = Trim(GetItemString(dw_prof_xref, lv_i_ClickedRow, "department"))
  lv_s_FacultyMember = Trim(GetItemString(dw_prof_xref, lv_i_ClickedRow, "name"))
  gv_struct_parms.department = lv_s_Department
  gv_struct_parms.faculty_member = lv_s_FacultyMember

  lv_s_FindFacultyMember = "name_combined = " + "'" + lv_s_FacultyMember + "'"
  lv_i_FindRow = dwFind(dw_faculty, lv_s_FindFacultyMember, 0, RowCount(dw_faculty))
  IF (lv_i_FindRow > 0) THEN
    ScrollToRow(dw_faculty, lv_i_FindRow)
  END_IF
ENDIF
Window:  w0_prof_main
Library:  e:\thesis\appl\schedule.pbl
Date:  5/2/95   Time:  16:30:26

END IF

lv_s_FindDepartment =  "department = "  + endquote + lv_s_Department + endquote
lv_i_FindRow = dwFind(dw_department, lv_s_FindDepartment, 0, RowCount(dw_department))
IF  (lv_i_FindRow > 0) THEN
   ScrollToRow(dw_department, lv_i_FindRow)
END IF

.cb_reset.enabled = TRUE
.cb_edit.enabled = TRUE

SetFocus(cb_edit)
END IF

SetRedraw(dw_prof_xref, TRUE)

lv_b_SelectionMade = TRUE

End of Script

Script for:  doubleclicked  event
//Highlights the clicked row and triggers the "clicked" event of command button edit.

//Local Variables
Integer  lv_i_Row
//End Local Variables

lv_i_Row = GetSelectedRow(dw_prof_xref, 0)

IF  (lv_i_Row > 0) THEN
   gv_struct_parms.department = Trim(GetItemString(dw_prof_xref, lv_i_Row,  "department"))
   gv_struct_parms.faculty_member = Trim(GetItemString(dw_prof_xref, lv_i_Row,  "name"))
   cb_edit.enabled = TRUE
Window: uo_prof_main
Library: e:\thesis\appl\schedule.pbl
Date: 5/2/95   Time: 16:30:26

    TriggerEvent(cb_edit, "clicked")
END IF

End of Script

DataWindow: dw_faculty
X = 1207   Y = 85   Width = 1047   Height = 81
TabOrder = 0   Visible = true   Enabled = true   DataObject = "d_professor_ddw"
LiveScroll = true   BorderStyle = stylebox!

Script for: itemchanged   event
//Selects a row in the dw_prof_xref if a department is also selected.

//Local Variables
Integer   lv_i_Row
Integer   lv_i_RowFound
String    lv_s_FacultyMember
String    lv_s_FilterString
String    lv_s_Department
//End Local Variables

lv_s_Department = gv_struct_parms.department
lv_i_Row = GetSelectedRow(iv_dwc_faculty, 0)

IF (lv_i_Row < 1) THEN
    Return
END IF

lv_s_FacultyMember = Trim(GetItemString(iv_dwc_faculty, lv_i_Row, "name_combined"))
gv_struct_parms.faculty_member = lv_s_FacultyMember

cb_reset.enabled = TRUE
Window: uo_prof_main
Library: e:\thesis\appl\schedule.pbl
Date: 5/2/95   Time: 16:30:26

//Does the filter string need to account for the fact that there is a department selected.
IF (NOT (IsNull(lv_s_Department) OR (lv_s_Department = "$"))) THEN
    SelectRow(dw_prof_xref, 0, FALSE)
    lv_s_FilterString = "department = "$" + '"' + lv_s_Department + '"' &
                        + '"' + "and name = "$" + '"' + lv_s_FacultyMember + '"'
    lv_i_RowFound = dwFind(dw_prof_xref, lv_s_FilterString, 0, RowCount(dw_prof_xref))

    IF (lv_i_RowFound > 0) THEN
        SetRedraw(dw_prof_xref, FALSE)
        SelectRow(dw_prof_xref, lv_i_RowFound, TRUE)
        ScrollToRow(dw_prof_xref, lv_i_RowFound)
        SetRedraw(dw_prof_xref, TRUE)
        lv_b_SelectionMade = TRUE
    END IF
    cb_edit.enabled = TRUE
END IF

End of Script

Script for: updateend event
//Triggered when a faculty member is added in the uo_prof_main user object.

//Local Variables
Integer   lv_i_InsertedRow
String    lv_s_FacultyMember
String    lv_s_Department
String    lv_s_FindString
//End Local Variables

SetRedraw(dw_prof_xref, FALSE)
SetRedraw(dw_faculty, FALSE)
SetRedraw(dw_department, FALSE)

SelectRow(dw_prof_xref, 0, FALSE)

lv_s_FacultyMember = gv_struct_parms.faculty_member

lv_s_FindString = "name_combined = " + '"' + lv_s_FacultyMember + '"'

IF (dwFind(iv_dwc_faculty, lv_s_FindString, 1, RowCount(iv_dwc_faculty)) = 0) THEN
  'lv_i_InsertedRow = InsertRow(iv_dwc_faculty, 0)
  SetItem(iv_dwc_faculty, lv_i_InsertedRow, "name_combined", lv_s_FacultyMember)

  SetSort(dw_faculty, "name_combined A")
  SetSort(iv_dwc_faculty, "name_combined A")

  Sort(dw_faculty)
  Sort(iv_dwc_faculty)
END IF

InsertRow(dw_faculty, 1)

SetItem(dw_faculty, 1, "name_combined", lv_s_FacultyMember)
ScrollToRow(dw_faculty, 1)

lv_s_Department = Trim(GetItemString(dw_department, 1, "department"))

IF ((IsNull(lv_s_Department)) OR (lv_s_Department = "")) THEN
  cb_edit.enabled = FALSE
ELSE
  cb_edit.enabled = TRUE
  cb_edit.default = TRUE
  cb_scheduling.default = FALSE
  cb_add.default = FALSE
  cb_reset.default = FALSE
  cb_ok.default = FALSE
END IF
Window: uo_prof_main
Library: e:\thesis\appl\schedule.phl
Date: 5/2/95    Time: 16:30:26

SetRedraw(dw_prof_xref, TRUE)
SetRedraw(dw_faculty, TRUE)
SetRedraw(dw_department, TRUE)

End of Script

Script for: dwnkey event
//drop down datawindows do strange things on tabs so the following overrides tab and sets focus where it
//should be.

IF (KeyDown(KeyTab!) AND KeyDown(KeyShift!)) THEN
    SetFocus(dw_department)
    SetActionCode(dw_faculty, 1)
//Used to control tabbing ... not sure why it works
    st_l.textcolor = RGB(255, 0, 0)
    iv_b_TabDone = TRU
    Return
ELSIF (KeyDown(KeyTab!)) THEN
    SetFocus(cb_scheduling)
    SetActionCode(dw_faculty, 1)
//Used to control tabbing ... not sure why it works
    st_l.textcolor = RGB(0, 255, 0)
    iv_b_TabDone = TRU
    Return
END IF

End of Script
StaticText: st_2
X = 403     Y = 13     Width = 330     Height = 69
TabOrder = 0  Visible = true  Text = "Department"  TextColor = 8388608
BackColor = 12632256  Alignment = left!  FillPattern = solid!

StaticText: st_1
X = 1207     Y = 13     Width = 353     Height = 69
TabOrder = 0  Visible = true  Text = "Faculty Member"
TextColor = 8388608  BackColor = 12632256  Alignment = left!
FillPattern = solid!
DataWindow: d_professor_dddw
Library: e:\thesis\appl\schedule.pbl
Date: 5/2/95   Time: 17:17:06

Retrieve: PBSELECT(TABLE(NAME="professor_xref") COLUMN(NAME="professor_xref.name_combined"))
Arguments: None
Update Table: professor_xref
Filter: None
Sort: None
Sparse: None
Column: name_combined
  Updateable: Yes
  Key: Yes
  Format: "[general]"
  Border style: None
  Validation: None
  Validation Message: None
  Tab Sequence: 10
  Initial Value: None
  Edit Style: DropDownListDataWindow
    Name: d_professor_list
    Data Column: name_combined
    Display Column: name_combined
DataWindow: d_professor
Library: e:\thesis\appl\schedule.pbl
Date: 5/2/95    Time: 17:17:34

Retrieve: PBSELECT(TABLE(NAME="professor_xref") COLUMN(NAME="professor_xref.name_combined"))
Arguments: None
Update Table: professor_xref
Filter: None
Sort: None
Sparse: None
Column: name_combined
   Updateable: Yes
   Key: Yes
   Format: "%[general]"
   Border style: None
   Validation: None
   Validation Message: None
   Tab Sequence: 10
   Initial Value: None
   Edit Style: Edit
   Edit limit: 35
DataWindow: d_professor_list
Library: e:\thesis\appl\schedule.pbl
Date: 5/2/95   Time: 17:15:46

Retrieve: PBSELECT(TABLE(NAME="professor_xref") COLUMN(NAME="professor_xref.name_combined"))
Arguments: None
Update Table: professor_xref
Filter: None
Sort: None
Sparse: None

Column: name_combined
   Updateable: Yes
   Key: Yes
   Format: "[general]"
   Border style: None
   Validation: None
   Validation Message: None
   Tab Sequence: 10
   Initial Value: None
   Edit Style: Edit
   Edit limit: 35
Retrieve: PBSELECT(TABLE(NAME="professor_information") COLUMN(NAME="professor_information.department")

Arguments: None
Update Table: professor_information
Filter: None
Sort: None
Sparse: None
Column: department
  Updateable: Yes
  Key: Yes
  Format: 
  Border style: None
  Validation: None
  Validation Message: None
  Tab Sequence: 0
  Initial Value: None
  Edit Style: Edit
  Edit limit: 4
Column: name
  Updateable: No
  Key: Yes
  Format: 
  Border style: None
  Validation: None
  Validation Message: None
Visible = true    Enabled = true    TitleBar = true    Title = "Add New Department"
ControlMenu = true  MinBox = true    MaxBox = true    Resizable = true
WindowType = main!  WindowState = normal!  BackColor = 12632256

Script for: open event
//Set the datawindow dw_department to retrieve data. Inserts a row into the datawindow dw_add.

//Local Variables

//End Local Variables

SetTransObject(dw_department, SQLCA)

dwShareData(uo_prof_main.dw_department, dw_department)

InsertRow(dw_add, 1)
ScrollToRow(dw_add, 1)

End of Script

Graph: gr_2
X = 5    Y = 737    Width = 988    Height = 721
TabOrder = 0    Visible = true    TextColor = 0    BackColor = 12632256
TextColor = 6316128    Spacing = 100    Elevation = 20    Rotation = -20
Perspective = 2    Title = "(None)"    Border = true    BorderStyle = stylebox!
GraphType = colgraph!
Legend = atbottom!

DataWindow: dw_department
X = 910    Y = 13    Width = 494    Height = 165
TabOrder = 0    Enabled = true    DataObject = "d_department_list"
TitleBar = true    Title = "Department List"    Border = true
LiveScroll = true    BorderStyle = stylebox!
DataWindow: dw_term
X = 33         Y = 457         Width = 494        Height = 205
TabOrder = 0   Enabled = true  DataObject = "d_term_list"
TitleBar = true Title = "Term List" Border = true  LiveScroll = true
BorderStyle = stylebox!

StaticText: st_l
X = 55         Y = 73         Width = 732         Height = 93
TabOrder = 0   Visible = true Text = "Enter a new department:"
TextColor = 33554432  BackColor = 12632256
Alignment = left! FillPattern = solid!

CommandButton: cb_ok
X = 961        Y = 473        Width = 247         Height = 109
TabOrder = 30  Visible = true Enabled = true  Text = "&OK"

Script for: clicked event
//Closes the window w_add_dept with saving the changes to the database regarding the new department.

//Local Variables
Boolean lv_b_DepartmentOkay
Integer lv_i_FoundRow
Integer lv_i_InsertedRow
Integer lv_i_NumberOfRows
Integer lv_i_NumberOfRowsDWC
String lv_s_Department
String lv_s_FilterString
String lv_s_OldDepartment
//End Local Variables

AcceptText(dw_add)
lv_s_Department = Trim(GetItemString(dw_add, 1, "department"))

IF (NOT (IsNull(lv_s_Department) OR (lv_s_Department = ""))) THEN
  lv_s_FilterString = "department = " + "'" + lv_s_Department + "'
  lv_i_FoundRow = dwFind(dw_department, lv_s_FilterString, 0, RowCount(dw_department))
  IF (lv_i_FoundRow = 0) THEN
    lv_i_InsertedRow = InsertRow(dw_department, 0)
    SetItem(dw_department, lv_i_InsertedRow, "department", lv_s_Department)
    gvstruct_parms.department = lv_s_Department
  ELSE
    MessageBox("Information", "This department already exists.", Information!, OK!)
    TriggerEvent(cb_cancel, "clicked")
    Return
  END IF
END IF

// Update database with any changes
IF ((ModifiedCount(dw_department) > 0) OR (DeletedCount(dw_department) > 0)) THEN
  DeleteRow(dw_department, 1)
  IF ((Update(dw_department) = 1)) THEN
    COMMIT;
  ELSEB
    ROLLBACK;
  END IF
END IF

dwShareDataOff(dw_department)

close(w_add_dept)

End of Script
Window: w_add_dept
Library: e:\thesis\appl\schedule.pbl
Date: 5/1/95    Time: 20:35:06

CommandButton: cb_cancel
X = 567       Y = 473       Width = 247       Height = 109
TabOrder = 20   Visible = true   Enabled = true   Text = "&Cancel"
Default = true

Script for: clicked   event
//Closes the window w_add_dept without saving the new department.

dwShareDataOff(dw_department)
close(w_add_dept)

End of Script

DataWindow: dw_add
X = 343       Y = 201       Width = 691       Height = 197
TabOrder = 10   Visible = true   Enabled = true   DataObject = "d_add_dept"
BorderStyle = stylebox!
User Object: uo_prof_maint
Libray: e:\thesis\appl\schedule.pbl
Date: 5/1/95   Time: 21:32:19

User Object: uo_prof_maint
X = 0       Y = 0
TabOrder = 0   Visible = true
BackColor = 12632256

Instance Variables
Boolean    iv_b_ChangesMade
Boolean    iv_b_TabDone

Width = 2803   Height = 1789
Enabled = true   Border = true
ObjectType = customvisual
Window: uo_prof_maint
Library: e:\thesis\appl\schedule.pbl
Date: 5/1/95    Time: 21:32:19

CommandButton  iv_cb_CurrentDefault
DataWindowChild  iv_dwc_time
Integer  iv_i_UpdateRow
Integer  iv_i_Row

s_main_win_parms  s_MainParms
s_prof_info_parms  s_InfoParms

End of Instance Variables

Script for: constructor  event
//Retrieves all data from the database needed for the user object.

//Local Variables
String  lv_s_FindString
Integer  iv_i_FoundRow
//End Local Variables

SetPointer(Hourglass!)

dwShareData(uo_prof_main.dw_faculty, dw_professor)

s_MainParms = Message.PowerObjectParm

dwGetChild(dw_time, "time_type", iv_dwc_time)

SetTransObject(dw_prof_info_xref, SQLCA)
SetTransObject(iv_dwc_time, SQLCA)
SetTransObject(dw_time, SQLCA)

Retrieve(dw_prof_info_xref, s_MainParms.department)
User Object: uo_prof_maint
Library: e:\thesis\appl\schedule.pb1
Date: 5/1/95    Time: 21:32:19

Retrieve(dw_time)

SetSort(iv_dwc_time, "time_type A")
Sort(iv_dwc_time)

SetSort(dw_time, "time_type A")
Sort(dw_time)

SetSort(dw_prof_info_xref, "name A")
Sort(dw_prof_info_xref)

IF (NOT s_MainParms.faculty_member = ") THEN
  lv_s_FindString = "name = " + "+" + s_MainParms.faculty_member + "+"
  lv_i_FoundRow = dwFind(dw_prof_info_xref, lv_s_FindString, 1, RowCount(dw_prof_info_xref))

SelectRow(dw_prof_info_xref, 0, FALSE)
SelectRow(dw_prof_info_xref, lv_i_FoundRow, TRUE)

iv_i_Row = lv_i_FoundRow

dw_prof_info_add.enabled = FALSE
dw_prof_info_update.enabled = FALSE

cb_add.enabled = FALSE
cb_add.default = FALSE
cb_select.enabled = TRUE
cb_select.default = TRUE
iv_cb_CurrentDefault = cb_select
cb_delete.enabled = TRUE
cb_delete.default = FALSE
cb_reset.enabled = TRUE
cb_reset.default = FALSE
cb_update.enabled = FALSE
cb_update.default = FALSE
cb_cancel.enabled = FALSE
cb_cancel.default = FALSE
User Object: uo_prof_maint
Library: e:\thesis\appl\schedule.pbl
Date: 5/1/95    Time: 21:32:19

    cb_more.enabled = FALSE
    cb_more.default = FALSE

    SetFocus(cb_select)
ELSE
    iv_cb_CurrentDefault = cb_add
    SetFocus(dw_prof_info_add)
END IF

InsertRow(dw_time, 1)
InsertRow(dw_prof_info_add, 1)
InsertRow(dw_prof_info_update, 1)

iv_b_ChangesMade = FALSE
s_InfoParms.first_time = TRUE

End of Script

Script for: other event
//Used to attempt to control tabbing.

//Local Variables
CommandButton     lv_cb_which
DataWindow         lv_dw_which
GraphicObject     lv_go_WhichControl
String             lv_s_TextValue
String             lv_s_DataObject
//End Local Variables

IF (KeyDown(KeyTab!) AND KeyDown(KeyShift!)) THEN
    IF (iv_b_TabDone) THEN
        iv_b_TabDone = FALSE
        Return
    END IF
END IF
lv_go_WhichControl = GetFocus()

choose case TypeOf(lv_go_WhichControl)

case CommandButton:
    lv_cb which = lv_go_WhichControl
    lv_s_TextValue = lv_cb_which.text
    IF (lv_s_TextValue = "&More ...") THEN
        IF (dw_time.enabled) THEN
            SetFocus(dw_time)
            Return
        ELSEIF (dw_prof_info_add.enabled) THEN
            SetFocus(dw_prof_info_add)
            SetColumn(dw_prof_info_add, "emp_type")
            Return
        ELSEIF (dw_prof_info_update.enabled) THEN
            SetFocus(dw_prof_info_update)
            SetColumn(dw_prof_info_update, "emp_type")
            Return
        ELSEIF (cb_ok.enabled) THEN
            SetFocus(cb_ok)
            Return
        ELSE
            Return
    END IF
    ELSEIF (lv_s_TextValue = "&Add") THEN
        IF (cb_more.enabled) THEN
            SetFocus(cb_more)
            Return
        ELSEIF (dw_time.enabled) THEN
            SetFocus(dw_time)
            Return
        ELSEIF (dw_prof_info_add.enabled) THEN
            SetFocus(dw_prof_info_add)
            SetColumn(dw_prof_info_add, "emp_type")
    END IF
ELSEIF (dw_prof_info_update enabled) THEN
    Return
    SetColumn (dw_prof_info_update, "emp_type")
    SetFocus (cb-ok)
    Return
ELSEIF (lw_s_textValue = "Select") THEN
    IF (cb-more enabled) THEN
        SetFocus (cb-more)
        Return
    ELSEIF (dw_time enabled) THEN
        SetColumn (dw_time, "emp_type")
        Return
    ELSEIF (lw_s_textValue = "Delete") THEN
        Return
ELSEIF (lw_s_textValue = "Select") THEN
    IF (cb-more enabled) THEN
        SetFocus (cb-more)
        Return
    ELSEIF (dw_time enabled) THEN
        SetColumn (dw_time, "emp_type")
        Return
    ELSEIF (lw_s_textValue = "Delete") THEN
        Return
ELSE
    Return
IF (cb_select.enabled) THEN
    SetFocus(cb_select)
    Return
ELSEIF (cb_add.enabled) THEN
    SetFocus(cb_add)
    Return
ELSEIF (cb_more.enabled) THEN
    SetFocus(cb_more)
    Return
ELSEIF (dw_time.enabled) THEN
    SetFocus(dw_time)
    Return
ELSEIF (dw_prof_info_add.enabled) THEN
    SetFocus(dw_prof_info_add)
    SetColumn(dw_prof_info_add, "emp_type")
    Return
ELSEIF (dw_prof_info_update.enabled) THEN
    SetFocus(dw_prof_info_update)
    SetColumn(dw_prof_info_update, "emp_type")
    Return
ELSEIF (cb_ok.enabled) THEN
    SetFocus(cb_ok)
    Return
ELSE
    Return
END IF
ELSEIF (lv_s_TextValue = "&Update") THEN
    IF (cb_delete.enabled) THEN
        SetFocus(cb_delete)
        Return
    ELSEIF (cb_select.enabled) THEN
        SetFocus(cb_select)
        Return
    ELSEIF (cb_add.enabled) THEN
        SetFocus(cb_add)
        Return
ELSEIF (cb_more.enabled) THEN
    SetFocus(cb_more)
    Return
ELSEIF (dw_time.enabled) THEN
    SetFocus(dw_time)
    Return
ELSEIF (dw_prof_info_add.enabled) THEN
    SetFocus(dw_prof_info_add)
    SetColumn(dw_prof_info_add, "emp_type")
    Return
ELSEIF (dw_prof_info_update.enabled) THEN
    SetFocus(dw_prof_info_update)
    SetColumn(dw_prof_info_update, "emp_type")
    Return
ELSEIF (cb_ok.enabled) THEN
    SetFocus(cb_ok)
    Return
ELSE
    Return
END IF
ELSEIF (iv_s_TextValue = "&Reset") THEN
    IF (cb_update.enabled) THEN
        SetFocus(cb_update)
        Return
    ELSEIF (cb_delete.enabled) THEN
        SetFocus(cb_delete)
        Return
    ELSEIF (cb_select.enabled) THEN
        SetFocus(cb_select)
        Return
    ELSEIF (cb_add.enabled) THEN
        SetFocus(cb_add)
        Return
    ELSEIF (cb_more.enabled) THEN
        SetFocus(cb_more)
        Return
    ELSE
ELSEIF (dw_time.enabled) THEN
    SetFocus(dw_time)
    Return
ELSEIF (dw_prof_info_add.enabled) THEN
    SetFocus(dw_prof_info_add)
    SetColumn(dw_prof_info_add, "emp_type")
    Return
ELSEIF (dw_prof_info_update.enabled) THEN
    SetFocus(dw_prof_info_update)
    SetColumn(dw_prof_info_update, "emp_type")
    Return
ELSEIF (cb_ok.enabled) THEN
    SetFocus(cb_ok)
    Return
ELSE
    Return
ENDIF
ELSEIF (lv_s_TextValue = "&Cancel") THEN
    IF (cb_reset.enabled) THEN
        SetFocus(cb_reset)
        Return
    ELSEIF (cb_update.enabled) THEN
        SetFocus(cb_update)
        Return
    ELSEIF (cb_delete.enabled) THEN
        SetFocus(cb_delete)
        Return
    ELSEIF (cb_select.enabled) THEN
        SetFocus(cb_select)
        Return
    ELSEIF (cb_add.enabled) THEN
        SetFocus(cb_add)
        Return
    ELSEIF (cb_more.enabled) THEN
        SetFocus(cb_more)
        Return
ELSEIF (dw_time.enabled) THEN
  SetFocus(dw_time)
  Return
ELSEIF (dw_prof_info_add.enabled) THEN
  SetFocus(dw_prof_info_add)
  SetColumn(dw_prof_info_add, "emp_type")
  Return
ELSEIF (dw_prof_info_update.enabled) THEN
  SetFocus(dw_prof_info_update)
  SetColumn(dw_prof_info_update, "emp_type")
  Return
ELSEIF (cb_ok.enabled) THEN
  SetFocus(cb_ok)
  Return
ELSE
  Return
END IF
ELSEIF (lv_s_TextValue = "&OK") THEN
  IF (cb_cancel.enabled) THEN
    SetFocus(cb_cancel)
    Return
  ELSEIF (cb_reset.enabled) THEN
    SetFocus(cb_reset)
    Return
  ELSEIF (cb_update.enabled) THEN
    SetFocus(cb_update)
    Return
  ELSEIF (cb_delete.enabled) THEN
    SetFocus(cb_delete)
    Return
  ELSEIF (cb_select.enabled) THEN
    SetFocus(cb_select)
    Return
  ELSEIF (cb_add.enabled) THEN
    SetFocus(cb_add)
    Return
ELSEIF (cb_more.enabled) THEN
    SetFocus(cb_more)
    Return
ELSEIF (dw_time.enabled) THEN
    SetFocus(dw_time)
    Return
ELSEIF (dw_prof_info_add.enabled) THEN
    SetFocus(dw_prof_info_add)
    SetColumn(dw_prof_info_add, "emp_type")
    Return
ELSEIF (dw_prof_info_update.enabled) THEN
    SetFocus(dw_prof_info_update)
    SetColumn(dw_prof_info_update, "emp_type")
    Return
ELSE
    Return
END IF
ELSE
    Return
END IF

case DataWindow!
  lv_dw_which = lv_go_WhichControl

  lv_s_DataObject = lv_dw_which.dataobject

  IF (lv_s_DataObject = "d_prof_info_add") THEN
    Return
  ELSEIF (lv_s_DataObject = "d_prof_info_update") THEN
    Return
  ELSEIF (lv_s_DataObject = "d_prof_info") THEN
  IF (cb_more.enabled) THEN
    SetFocus(cb_more)
    Return
  ELSEIF (dw_time.enabled) THEN
    SetFocus(dw_time)
END Choose
ELSEIF (KeyDown(KeyTab!!)) THEN
  IF (lv_b_TabDone) THEN
    lv_b_TabDone = FALSE
    Return
  END IF

lv_go_WhichControl = GetFocus()

'choose case TypeOf(lv_go_WhichControl)

case CommandButton!
  lv_cb_which = lv_go_WhichControl
  lv_s_TextValue = lv_cb_which.text
  IF (lv_s_TextValue = "More ...") THEN
    IF (cb_add.enabled) THEN
      SetFocus(cb_add)
      Return
    ELSEIF (cb_select.enabled) THEN
      SetFocus(cb_select)
      Return
    ELSEIF (cb_delete.enabled) THEN
      SetFocus(cb_delete)
      Return
    ELSEIF (cb_update.enabled) THEN
      SetFocus(cb_update)
      Return
    ELSEIF (cb_reset.enabled) THEN
      SetFocus(cb_reset)
      Return
    ELSEIF (cb_cancel.enabled) THEN
      SetFocus(cb_cancel)
      Return
    ELSEIF (cb_ok.enabled) THEN
      SetFocus(cb_ok)
      Return
  END IF
END CASE
ELSE
   Return
END IF
ELSEIF (lv_s_TextValue = "&Add") THEN
   IF (cb_select.enabled) THEN
      SetFocus(cb_select)
      Return
   ELSEIF (cb_delete.enabled) THEN
      SetFocus(cb_delete)
      Return
   ELSEIF (cb_update.enabled) THEN
      SetFocus(cb_update)
      Return
   ELSEIF (cb_reset.enabled) THEN
      SetFocus(cb_reset)
      Return
   ELSEIF (cb_cancel.enabled) THEN
      SetFocus(cb_cancel)
      Return
   ELSEIF (cb_ok.enabled) THEN
      SetFocus(cb_ok)
      Return
   ELSE
      Return
   END IF
ELSEIF (lv_s_TextValue = "&Select") THEN
   IF (cb_delete.enabled) THEN
      SetFocus(cb_delete)
      Return
   ELSEIF (cb_update.enabled) THEN
      SetFocus(cb_update)
      Return
   ELSEIF (cb_reset.enabled) THEN
      SetFocus(cb_reset)
      Return
   ELSEIF (cb_cancel.enabled) THEN
      Return
END IF
SetFocus(cb_cancel)
Return
ELSEIF (cb_ok.enabled) THEN
SetFocus(cb_ok)
Return
ELSE
Return
END IF
ELSEIF (lv_s_TextValue = "&Delete") THEN
IF (cb_update.enabled) THEN
SetFocus(cb_update)
Return
ELSEIF (cb_reset.enabled) THEN
SetFocus(cb_reset)
Return
ELSEIF (cb_cancel.enabled) THEN
SetFocus(cb_cancel)
Return
ELSEIF (cb_ok.enabled) THEN
SetFocus(cb_ok)
Return
ELSE
Return
END IF
ELSEIF (lv_s_TextValue = "&Update") THEN
IF (cb_reset.enabled) THEN
SetFocus(cb_reset)
Return
ELSEIF (cb_cancel.enabled) THEN
SetFocus(cb_cancel)
Return
ELSEIF (cb_ok.enabled) THEN
SetFocus(cb_ok)
Return
ELSE
Return
END IF
ELSEIF (lv_s_TextValue = "&Reset") THEN
  IF (cb_cancel.enabled) THEN
    SetFocus(cb_cancel)
    Return
  ELSEIF (cb_ok.enabled) THEN
    SetFocus(cb_ok)
    Return
  ELSE
    Return
ENDIF
ELSEIF (lv_s_TextValue = "&Cancel") THEN
  IF (cb_ok.enabled) THEN
    SetFocus(cb_ok)
    Return
  ELSE
    Return
ENDIF
ELSEIF (lv_s_TextValue = "&OK") THEN
  IF (dw_prof_info_add.enabled) THEN
    SetFocus(dw_prof_info_add)
    SetColumn(dw_prof_info_add, "last_name")
    Return
  ELSEIF (dw_prof_info_update.enabled) THEN
    SetFocus(dw_prof_info_update)
    SetColumn(dw_prof_info_update, "active")
    Return
  ELSEIF (dw_time.enabled) THEN
    SetFocus(dw_time)
    Return
  ELSEIF (cb_more.enabled) THEN
    SetFocus(cb_more)
    Return
  ELSEIF (cb_add.enabled) THEN
    SetFocus(cb_add)
    Return
ELSEIF (cb_select.enabled) THEN
    SetFocus(cb_select)
    Return
ELSEIF (cb_delete.enabled) THEN
    SetFocus(cb_delete)
    Return
ELSEIF (cb_update.enabled) THEN
    SetFocus(cb_update)
    Return
ELSEIF (cb_reset.enabled) THEN
    SetFocus(cb_reset)
    Return
ELSEIF (cb_cancel.enabled) THEN
    SetFocus(cb_cancel)
    Return
ELSE
    Return
END IF
ELSE
    Return
END IF

case DataWindow!
    lv_dw_Which = lv_go_WhichControl

    lv_s_DataObject = lv_dw_Which.dataobject
    IF (lv_s_DataObject = "d_prof_info_add") THEN
        Return
    ELSEIF (lv_s_DataObject = "d_prof_info_update") THEN
        Return
    ELSEIF (lv_s_DataObject = "d_prof_info") THEN
        IF (cb_add.enabled) THEN
            SetFocus(cb_add)
            Return
        ELSEIF (cb_select.enabled) THEN

Page: 17
SetFocus(cb_select)
Return
ELSEIF (cb_delete.enabled) THEN
    SetFocus(cb_delete)
    Return
ELSEIF (cb_update.enabled) THEN
    SetFocus(cb_update)
    Return
ELSEIF (cb_reset.enabled) THEN
    SetFocus(cb_reset)
    Return
ELSEIF (cb_cancel.enabled) THEN
    SetFocus(cb_cancel)
    Return
ELSEIF (cb_ok.enabled) THEN
    SetFocus(cb_ok)
    Return
ELSE
    Return
ENDIF
ELSEIF (lv_s_DataObject = "d_time_ddw") THEN
    IF (cb_more.enabled) THEN
        SetFocus(cb_more)
        Return
    ELSEIF (cb_add.enabled) THEN
        SetFocus(cb_add)
        Return
    ELSEIF (cb_select.enabled) THEN
        SetFocus(cb_select)
        Return
    ELSEIF (cb_delete.enabled) THEN
        SetFocus(cb_delete)
        Return
    ELSEIF (cb_update.enabled) THEN
        SetFocus(cb_update)
        Return
    END IF
User Object: uo_prot_maint
Library: e:\thesis\appl\schedule.pbl
Date: 5/1/95    Time: 21:32:19

ELSEIF (cb_reset.enabled) THEN
    SetFocus(cb_reset)
    Return

ELSEIF (cb_cancel.enabled) THEN
    SetFocus(cb_cancel)
    Return

ELSEIF (cb_ok.enabled) THEN
    SetFocus(cb_ok)
    Return

ELSE
    Return
END IF

ELSE
    Return
END IF

END Choose
END IF

End of Script

DataWindow: dw_professor
X = 1793    Y = 1357    Width = 494    Height = 361
TabOrder = 60    DataObject = "d_professor"    Border = true
LiveScroll = true    BorderStyle = stylebox!

DataWindow: dw_time
X = 1962    Y = 233    Width = 366    Height = 73
TabOrder = 20    Visible = true    Enabled = true    DataObject = "d_time_dddw"
LiveScroll = true    BorderStyle = stylebox!
//Local Variables

//End Local Variables

IF (dw_prof_info_update.enabled) THEN
  cb_update.enabled = TRUE
END IF

End of Script

StaticText: st_7
X = 92  Y = 441  Width = 439  Height = 73
TabOrder = 0  Visible = true  Text = "Current Department"
TextColor = 8388608  BackColor = 12632256  Alignment = left!
FillPattern = solid!

DataWindow: dw_prof_info_update
X = 37  Y = 237  Width = 1829  Height = 89
TabOrder = 10  Enabled = true  DataObject = "d_prof_info_update"
LiveScroll = true  BorderStyle = stylebox!

Script for: editchanged event
//Once data has been modified, the update command button is enabled.

//Local Variables

//End Local Variables

//Indicate that changes were made
iv_b_ChangesMade = TRUE

//Command Button Control
cb_update.enabled = TRUE

End of Script
Window: uo_prof_maint
Library: e:\thesis\apl\schedule.pbl
Date: 5/1/95    Time: 21:32:19

Script for: dwkey event
  //Used to attempt to control tabbing.

  //Local Variables
  String lv_s_CurrentColumn
  //End Local Variables

  lv_s_CurrentColumn = GetColumnname(dw_prof_info_updates)

  IF (KeyDown(keyTab!) AND KeyDown(keyShift!) AND (lv_s_CurrentColumn = "active")) THEN
    lv_b_TabDone = TRUE
    SetFocus(cb_ok)
    Return
  ELSIF (KeyDown(keyTab!) AND (lv_s_CurrentColumn = "emp_type")) THEN
    lv_b_TabDone = TRUE
    cb_update.enabled = TRUE
    cb_reset.enabled = TRUE
    SetFocus(dw_time)
    Return
  END IF

End of Script

StaticText: st_6
  X = 1939    Y = 125    Width = 366    Height = 73
  TabOrder = 0    Visible = true    Text = "Time Preference"
  TextColor = 8388608    BackColor = 12632256    Alignment = left!
  FillPattern = solid!
Window: uo_prof_maint
Library: e:\thesis\appl\schedule.pbl
Date: 5/1/95     Time: 21:32:19

StaticText: st_4
X = 1569        Y = 77        Width = 293        Height = 121
TabOrder = 0    Visible = true Text = "Employment Classification"
TextColor = 8388608 BackColor = 12632256 Alignment = left!
FillPattern = solid!

StaticText: st_3
X = 1354        Y = 125       Width = 147        Height = 73
TabOrder = 0    Visible = true Text = "Active"         TextColor = 8388608
BackColor = 12632256 Alignment = left!   FillPattern = solid!

StaticText: st_2
X = 842         Y = 125       Width = 252        Height = 73
TabOrder = 0    Visible = true Text = "First Name"     TextColor = 8388608
BackColor = 12632256 Alignment = left!   FillPattern = solid!

StaticText: st_1
X = 110         Y = 125       Width = 275        Height = 73
TabOrder = 0    Visible = true Text = "Last Name"      TextColor = 8388608
BackColor = 12632256 Alignment = left!   FillPattern = solid!

CommandButton: cb_more
X = 2423        Y = 157       Width = 247        Height = 109
TabOrder = 40   Visible = true Text = "&More ..."

Script for: clicked event
//Opens the w_more_prof_info window used to get the additional required information about
//the faculty member needed for the scheduling portion.

//Local Variables
Window: wo_prof_main
Library: e:\thesis\appl\schedule.pbl
Date: 5/1/95    Time: 21:32:19

//End Local Variables

IF (dw_prof_info_add.visible) THEN
    s_InfoParams.add_mode = TRUE
ELSE
    s_InfoParams.add_mode = FALSE
    cb_update.enabled = TRUE
END IF

OpenWithParm(w_more_prof_info, s_InfoParams)

End of Script

Script for: uponreturn event
//Enables the appropriate buttons and keeps track of the default button.

//Local Variables

//End Local Variables

s_InfoParams = Message.PowerObjectParm

IF (dw_prof_info_add.enabled) THEN
    iv_cb_CurrentDefault.default = FALSE
    cb_add.default = TRUE
    iv_cb_CurrentDefault = cb_add
    SetFocus(cb_add)
ELSEIF (dw_prof_info_update.enabled) THEN
    iv_cb_CurrentDefault.default = FALSE
    cb_update.default = TRUE
    iv_cb_CurrentDefault = cb_update
    SetFocus(cb_update)
END IF
iv_b_ChangesMade = TRUE

End of Script

Script for: lbbuttondown event
//Used to keep track of the default button.

//Local Variables

//End Local Variables

iv_cb_CurrentDefault.default = FALSE
iv_cb_CurrentDefault = cb_more

End of Script

CommandButton: cb_ok
X = 2387   Y = 1601   Width = 247   Height = 109
TabOrder = 130   Visible = true   Enabled = true   Text = "&OK"

Script for: clicked event
//Saves any new and changed data back to the database before closing the window.

//Local Variables

//End Local Variables

//Update database with current changes if any were made.
IF ( (ModifiedCount(dw_prof_info_xref) > 0) OR (DeletedCount(dw_prof_info_xref) > 0)) THEN
  IF (Update(dw_prof_info_xref) = 1) THEN
    COMMIT;
ELSE
  ROLLBACK;
END IF
END IF

IF ((ModifiedCount(dw_professor) > 0) OR (DeletedCount(dw_professor) > 0)) THEN
  DeleteRow(dw_professor, 1)
  IF ((Update(dw_professor) = 1)) THEN
    COMMIT;
    ELSE
    ROLLBACK;
  END IF
END IF

dwShareDataOff(dw_professor)
close(w maint main)

TriggerEvent(uo_prof main, "getcontrolagain")

End of Script

Script for: lbutton down event  
//Used to keep track of the default button.

//Local Variables

//End Local Variables

iv cb_CurrentDefault.default = FALSE
iv cb_CurrentDefault = cb ok

End of Script
CommandButton: cb_cancel
X = 2058      Y = 1601      Width = 247      Height = 109
TabOrder = 120  Visible = true  Text = "&Cancel"

Script for: clicked event
//Exits the window without saving any changes back to the database.

dwShareDataOff(dw_professor)
close(w_maint_main)

End of Script

Script for: lbuttondown event
//Used to keep track of the default button.

//Local Variables

//End Local Variables

iv_cb_CurrentDefault.default = FALSE
iv_cb_CurrentDefault = cb_cancel

End of Script

CommandButton: cb_reset
X = 1495      Y = 1601      Width = 247      Height = 109
TabOrder = 110  Visible = true  Text = "&Reset"

Script for: clicked event
Window: uo_prof_maint
Library: e:\thesis\appl\schedule.pbl
Date: 5/1/95   Time: 21:32:19

//was entered and before changed (yet not) updated data is modified.

//Local Variables
Integer  lv_i_Row
//End Local Variables

IF (cb_add.enabled) THEN
  Reset(dw_prof_info_add)
  InsertRow(dw_prof_info_add, 0)
  ScrollToRow(dw_prof_info_add, 1)
  SetFocus(dw_prof_info_add)
  SetColumn(dw_prof_info_add, "last_name")
  lv_i_Row = dwFind(dw_time, "time_type = " + "+" + "+", 0, RowCount(dw_time))
IF (lv_i_Row = 0) THEN
  InsertRow(dw_time, 1)
  ScrollToRow(dw_time, 1)
ELSE
  ScrollToRow(dw_time, lv_i_Row)
END IF

//Reset the structure values
s_InfoParms.first_time = TRUE
s_InfoParms.class_preference_1 = 0
s_InfoParms.class_preference_2 = 0
s_InfoParms.class_preference_3 = 0
s_InfoParms.class_preference_4 = 0
s_InfoParms.previous_class_1 = 0
s_InfoParms.previous_class_2 = 0
s_InfoParms.previous_class_3 = 0
s_InfoParms.previous_class_4 = 0
s_InfoParms.chair = ";"
s_InfoParms.priority = 0
s_InfoParms.degree_earned = ""
s_InfoParms.yrs_csbsju = 0
s_InfoParms.tenured = ";"
Window: uo_prof_maint
Library: e:\thesis\appl\schedule.pbl
Date: 5/1/95 Time: 21:32:19

    cb_more.enabled = FALSE
    cb_reset.enabled = FALSE
    lv_cb_CurrentDefault.default = FALSE
    cb_add.default = TRUE
    lv_cb_CurrentDefault = cb_add

    Return
END IF

SetRedraw(dw_prof_info_xref, FALSE)
SetRedraw(dw_prof_info_add, FALSE)
SetRedraw(dw_prof_info_update, FALSE)

SelectRow(dw_prof_info_xref, 0, FALSE)

Reset(dw_prof_info_update)
InsertRow(dw_prof_info_update, 0)
ScrollToRow(dw_prof_info_update, 1)

lv_i_Row = dwFind(dw_time, "time_type = " + '"' + '"', 0, RowCount(dw_time))
IF (lv_i_Row = 0) THEN
    InsertRow(dw_time, 1)
    ScrollToRow(dw_time, 1)
ELSE
    ScrollToRow(dw_time, lv_i_Row)
END IF

    dw_prof_info_update.visible = FALSE
    dw_prof_info_add.visible = TRUE

    dw_prof_info_add.enabled = TRUE
    dw_prof_info_update.enabled = FALSE
    dw_prof_info_xref.enabled = TRUE

    //Reset the structure values
Window: uc_prof_saint
Library: e:\thesis\appl\schedule.pbl
Date: 5/1/95     Time: 21:32:19

s_InfoParms.first_time = TRUE
s_InfoParms.class_preference_1 = 0
s_InfoParms.class_preference_2 = 0
s_InfoParms.class_preference_3 = 0
s_InfoParms.class_preference_4 = 0
s_InfoParms.previous_class_1 = 0
s_InfoParms.previous_class_2 = 0
s_InfoParms.previous_class_3 = 0
s_InfoParms.previous_class_4 = 0
s_InfoParms.chair = ""
s_InfoParms.priority = 0
s_InfoParms.degree_earned = ""
s_InfoParms.yrs_csbsju = 0
s_InfoParms.tenured = ""

iv_b_ChangesMade = FALSE

SetRedraw(dw_prof_info_update, TRUE)
SetRedraw(dw_prof_info_add, TRUE)
SetRedraw(dw_prof_info_xref, TRUE)

//Command Button Control
cb_reset.enabled = FALSE
cb_add.enabled = TRUE
iv_cb_CurrentDefault.default = FALSE
cb_add.default = TRUE
iv_cb_CurrentDefault = cb_add
cb_select.enabled = FALSE
cb_delete.enabled = FALSE
cb_update.enabled = FALSE
cb_more.enabled = FALSE

SetFocus(dw_prof_info_add)

End of Script
Window: uo_prof_maint
Library: e:\thesis\appl\schedule.pbl
Date: 5/1/95    Time: 21:32:19

Script for: lbButtondown event
//Used to keep track of the default button.

//Local Variables

//End Local Variables

lv_cb_CurrentDefault.default = FALSE
lv_cb_CurrentDefault = cb_reset

End of Script

CommandButton: cb_update
X = 1166      Y = 1601      Width = 247      Height = 109
TabOrder = 100     Visible = true     Text = "&Update"

Script for: clicked event
//Gets all data from the prof_info_update datawindow and updates the prof_info_xref datawindow.  
//Returns the user object back to add mode.

//Local Variables
Integer   lv_i_Row
String    lv_s_LastName
String    lv_s_FirstName
String    lv_s_Name
String    lv_s_Active
String    lv_s_EmpType
String    lv_s_TimePreferences
//End Local Variables

SetPointer(Hourglass!)
AcceptText(dw_prof_info_update)

lv_i_Row = GetSelectedRow(lv_dwc_time, 0)

lv_s_LastName = Trim(GetItemString(dw_prof_info_update, 1, "last_name"))
lv_s_FirstName = Trim(GetItemString(dw_prof_info_update, 1, "first_name"))
lv_s_Active = Trim(GetItemString(dw_prof_info_update, 1, "active"))
lv_s_EmpType = Trim(GetItemString(dw_prof_info_update, 1, "emp_type"))
lv_s_TimePreference = Trim(GetItemString(lv_dwc_time, lv_i_Row, "time_type"))

lv_s_Name = lv_s_LastName + "," + lv_s_FirstName

SetItem(dw_prof_info_xref, lv_i_UpdateRow, "name", lv_s_Name)
SetItem(dw_prof_info_xref, lv_i_UpdateRow, "active", lv_s_Active)
SetItem(dw_prof_info_xref, lv_i_UpdateRow, "emp_type", lv_s_EmpType)
SetItem(dw_prof_info_xref, lv_i_UpdateRow, "time_preference", lv_s_TimePreference)

SetItem(dw_prof_info_xref, lv_i_UpdateRow, "class_preference_1", s_InfoParms.class_preference_1)
SetItem(dw_prof_info_xref, lv_i_UpdateRow, "class_preference_2", s_InfoParms.class_preference_2)
SetItem(dw_prof_info_xref, lv_i_UpdateRow, "class_preference_3", s_InfoParms.class_preference_3)
SetItem(dw_prof_info_xref, lv_i_UpdateRow, "class_preference_4", s_InfoParms.class_preference_4)
SetItem(dw_prof_info_xref, lv_i_UpdateRow, "previous_class_1", s_InfoParms.previous_class_1)
SetItem(dw_prof_info_xref, lv_i_UpdateRow, "previous_class_2", s_InfoParms.previous_class_2)
SetItem(dw_prof_info_xref, lv_i_UpdateRow, "previous_class_3", s_InfoParms.previous_class_3)
SetItem(dw_prof_info_xref, lv_i_UpdateRow, "previous_class_4", s_InfoParms.previous_class_4)
SetItem(dw_prof_info_xref, lv_i_UpdateRow, "priority", s_InfoParms.priority)
SetItem(dw_prof_info_xref, lv_i_UpdateRow, "tenured", s_InfoParms.tenured)
SetItem(dw_prof_info_xref, lv_i_UpdateRow, "yrs_at_csbsu", s_InfoParms.yrs_csbsu)
SetItem(dw_prof_info_xref, lv_i_UpdateRow, "high_degree_earned", s_InfoParms.high_degree_earned)
SetItem(dw_prof_info_xref, lv_i_UpdateRow, "chair", s_InfoParms.chair)

dw_prof_info_xref = calculate_priority(dw_prof_info_xref)
SetSort(dw_prof_info_xref, "name A")
Sort(dw_prof_info_xref)

//Reset the structure values
Window: uo_prof_maint
Library: e:\thesis\appl\schedule.pl1
Date: 5/1/95    Time: 21:32:19

s_InfoParms.first_time = TRUE
s_InfoParms.class_preference_1 = 0
s_InfoParms.class_preference_2 = 0
s_InfoParms.class_preference_3 = 0
s_InfoParms.class_preference_4 = 0
s_InfoParms.previous_class_1 = 0
s_InfoParms.previous_class_2 = 0
s_InfoParms.previous_class_3 = 0
s_InfoParms.previous_class_4 = 0
s_InfoParms.chair ="
.s_InfoParms.priority = 0
s_InfoParms.degree_earned ="
.s_InfoParms.yrs_csbsju = 0
.s_InfoParms.tenured ="

//Reset the drop down datawindow
lv_i_Row = dwFind(dw_time, "time_type = " + ' ' + ' ' + ' ', 0, RowCount(dw_time))
IF (lv_i_Row = 0) THEN
 InsertRow(dw_time, 1)
 ScrollToRow(dw_time, 1)
ELSE
 ScrollToRow(dw_time, lv_i_Row)
 END IF

SetRedraw(dw_prof_info_xref, FALSE)
SetRedraw(dw_prof_info_add, FALSE)
SetRedraw(dw_prof_info_update, FALSE)

SelectRow(dw_prof_info_xref, 0, FALSE)

Reset(dw_prof_info_update)
InsertRow(dw_prof_info_update, 0)
ScrollToRow(dw_prof_info_update, 1)

dw_prof_info_update.visible = FALSE
dw_prof_info_add.visible = TRUE
Window: uo_prof_maint
Library: e:\thesis\appl\schedule.pbl
Date: 5/1/95   Time: 21:32:19

dw_prof_info_update.enabled = FALSE
dw_prof_info_add.enabled = TRUE

SetFocus(dw_prof_info_add)

//Command Button Control
cb_more.enabled = FALSE
cb_add.enabled = TRUE
iv_cb_CurrentDefault.default = FALSE
cb_add.default = TRUE
iv_cb_CurrentDefault = cb_add
cb_select.enabled = FALSE
cb_delete.enabled = FALSE
cb_update.enabled = FALSE
cb_reset.enabled = FALSE
cb_cancel.enabled = TRUE

iv_b_ChangesMade = FALSE

SetRedraw(dw_prof_info_xref, TRUE)
SetRedraw(dw_prof_info_add, TRUE)
SetRedraw(dw_prof_info_update, TRUE)

End of Script

Script for: lbuttondown event
//Used to keep track of the default button.

//Local Variables

//End Local Variables

iv_cb_CurrentDefault.default = FALSE
iv_cb_CurrentDefault = cb_update

End of Script

CommandButton: cb_delete
X = 837   Y = 1601   Width = 247   Height = 109
TabOrder = 90   Visible = true   Text = "&Delete"

Script for: clicked event
//Deletes the selected row in the prof_info_xref dataview after the user verifies the deletion.

//Local Variables
Integer  lv_i_Row
String   lv_s_Name
String   lv_s_FindString
//End Local Variables

//Capture the row number to delete.
lv_i_Row = GetSelectedRow(dw_prof_info_xref, 0)

//Ensure a row has been selected prior to deletion.
IF (lv_i_Row = 0) THEN
   MessageBox("Information", "A row must be selected before deleting.", Information!, OK!)
   Return
END IF

IF (MessageBox("Information", "Delete current row?", Question!, YesNo!, 2) = 1) THEN
   lv_s_Name = Trim:GetItemString(dw_prof_info_xref, lv_i_Row, "name")
   DeleteRow(dw_prof_info_xref, lv_i_Row)
   lv_s_FindString = "name_combined = " + `'` + lv_s_Name + `'`
   lv_i_Row = dwFind(dw_professor, lv_s_FindString, 0, RowCount(dw_professor))
DeleteRow(dw_professor, lv_i_Row)

dw_prof_info_add.enabled = TRUE
dw_prof_info_update.enabled = FALSE

//Command Button Control
cb_add.enabled = TRUE
iv_cb_CurrentDefault.default = FALSE
cb_add.default = TRUE
iv_cb_CurrentDefault = cb_add
cb_select.enabled = FALSE
cb_delete.enabled = FALSE
cb_update.enabled = FALSE
cb_reset.enabled = FALSE
cb_cancel.enabled = TRUE

SetFocus(dw_prof_info_add)
END IF

End of Script

Script for: lbbuttondown event
//Used to keep track of the default button.

//Local Variables

//End Local Variables

iv_cb_CurrentDefault.default = FALSE
iv_cb_CurrentDefault = cb_delete

End of Script
Window: uc_prof_maint
Library: e:\thesis\appl\schedule.pbl
Date: 5/1/95    Time: 21:32:19

CommandButton: cb_add
X = 119        Y = 1601       Width = 247      Height = 109
TabOrder = 70  Visible = true  Enabled = true    Text = "&Add"
Default = true

Script for: clicked  event
//Gets and information from the prof_info_add datawindow and set the information into
//the prof_info_xref window along with the faculty member structure information.

//Local Variables
Integer    lv_i_InsertedRow  
Integer    lv_i_Row
Integer    lv_i_FoundRow
String     lv_s_LastName
String     lv_s_FirstName
String     lv_s_Name
String     lv_s_Active
String     lv_s_EmpType
String     lv_s_TimePreference
String     lv_s_FindString
String     lv_s_FilterString
//End Local Variables

SetPointer(Hourglass!)

AcceptText(dw_prof_info_add)

lv_i_Row = GetSelectedRow(iv_dwc_time, 0)

//Retrieve information entered.
lv_s_LastName = Trim(.GetItemString(dw_prof_info_add, 1, "last_name"))
lv_s_FirstName = Trim(.GetItemString(dw_prof_info_add, 1, "first_name"))
lv_s_Active = Trim(.GetItemString(dw_prof_info_add, 1, "active"))
lv_s_EmpType = Trim(.GetItemString(dw_prof_info_add, 1, "emp_type"))
lv_s_TimePreference = Trim(GetItemString(iv_dwc_time, lv_i_Row, "time_type"))

//Validation
IF (IsNull(lv_s_LastName) OR (lv_s_LastName = "")) THEN
MessageBox("Validation Error", "A last name must be entered prior to adding the faculty member.", StopSign!, OK!)
   SetFocus(dw_prof_info_add)
   SetColumn(dw_prof_info_add, "last_name")
   Return
ELSIF (IsNull(lv_s_FirstName) OR (lv_s_FirstName = "")) THEN
MessageBox("Validation Error", "A first name must be entered prior to adding the faculty member.", StopSign!, OK!)
   SetFocus(dw_prof_info_add)
   SetColumn(dw_prof_info_add, "first_name")
   Return
ELSIF (IsNull(lv_s_Active) OR (lv_s_Active = "")) THEN
MessageBox("Validation Error", "The Active field must be entered prior to adding the faculty member.", StopSign!, OK!)
   SetFocus(dw_prof_info_add)
   SetColumn(dw_prof_info_add, "active")
   Return
ELSIF (IsNull(lv_s_EmpType) OR (lv_s_EmpType = "")) THEN
MessageBox("Validation Error", "The employment classification must be entered prior to adding the faculty member.", StopSign!, OK!)
   SetFocus(dw_prof_info_add)
   SetColumn(dw_prof_info_add, "emp_type")
   Return
ELSIF (IsNull(lv_s_TimePreference) OR (lv_s_TimePreference = "")) THEN
MessageBox("Validation Error", "A time preference must be selected prior to adding the faculty member.", StopSign!, OK!)
   SetFocus(dw_time)
   SetColumn(dw_time, "time_preference")
   Return
END IF

lv_s_Name = lv_s_LastName + ", " + lv_s_FirstName
// Add information into xref window.
SetRedraw(dw_prof_info_xref, FALSE)

lv_s_FindString = "name = " + '"' + lv_s_Name + '"

IF (dwFind(dw_prof_info_xref, lv_s_FindString, 0, RowCount(dw_prof_info_xref)) > 0) THEN
MessageBox("Validation Error", "This faculty member already exist. If you wish to make changes, select record and update.", &
Information!, OK!)
SetRedraw(dw_prof_info_xref, TRUE)
SetFocus(dw_prof_info_add)
Return
END IF

lv_i_InsertedRow = InsertRow(dw_prof_info_xref, 0)
SetItem(dw_prof_info_xref, lv_i_InsertedRow, "department", s_MainParms.department)
SetItem(dw_prof_info_xref, lv_i_InsertedRow, "name", lv_s_Name)
SetItem(dw_prof_info_xref, lv_i_InsertedRow, "active", lv_s_Active)
SetItem(dw_prof_info_xref, lv_i_InsertedRow, "emp_type", lv_s_EmpType)
SetItem(dw_prof_info_xref, lv_i_InsertedRow, "time_preference", lv_s_TimePreference)
SetItem(dw_prof_info_xref, lv_i_InsertedRow, "class_preference_1", s_InfoParms.class_preference_1)
SetItem(dw_prof_info_xref, lv_i_InsertedRow, "class_preference_2", s_InfoParms.class_preference_2)
SetItem(dw_prof_info_xref, lv_i_InsertedRow, "class_preference_3", s_InfoParms.class_preference_3)
SetItem(dw_prof_info_xref, lv_i_InsertedRow, "class_preference_4", s_InfoParms.class_preference_4)
SetItem(dw_prof_info_xref, lv_i_InsertedRow, "previous_class_1", s_InfoParms.previous_class_1)
SetItem(dw_prof_info_xref, lv_i_InsertedRow, "previous_class_2", s_InfoParms.previous_class_2)
SetItem(dw_prof_info_xref, lv_i_InsertedRow, "previous_class_3", s_InfoParms.previous_class_3)
SetItem(dw_prof_info_xref, lv_i_InsertedRow, "previous_class_4", s_InfoParms.previous_class_4)
SetItem(dw_prof_info_xref, lv_i_InsertedRow, "priority", s_InfoParms.priority)
SetItem(dw_prof_info_xref, lv_i_InsertedRow, "tenured", s_InfoParms.tenured)
SetItem(dw_prof_info_xref, lv_i_InsertedRow, "yrs_at_csbsju", s_InfoParms.yrs_csbsju)
SetItem(dw_prof_info_xref, lv_i_InsertedRow, "high_degree_earned", s_InfoParms.degree_earned)
SetItem(dw_prof_info_xref, lv_i_InsertedRow, "chair", s_InfoParms.chair)

dw_prof_info_xref = calculate_priority(dw_prof_info_xref)
// add name to invisible datawindow -- shares data with faculty member dddw in uo_prof_main
lv_s_FilterString = "name_combined = " + '"' + lv_s_Name + '"

lv_i_FoundRow = dwFind(dw_professor, lv_s_FilterString, 0, RowCount(dw_professor))

IF (lv_i_FoundRow = 0) THEN
    lv_i_InsertedRow = InsertRow(dw_professor, 0)
    SetItem(dw_professor, lv_i_InsertedRow, "name_combined", lv_s_Name)
    SetSort(dw_professor, "name_combined A")
    Sort(dw_professor)
END IF

SelectRow(dw_prof_info_xref, 0, FALSE)
SetSort(dw_prof_info_xref, "name A")
Sort(dw_prof_info_xref)

// Don't think this will work ... but what the heck ... Did it?? What was it supposed to accomplish?
lv_i_InsertedRow = GetSelectedRow(dw_prof_info_xref, 0)
ScrollToRow(dw_prof_info_xref, lv_i_InsertedRow)

SetRedraw(dw_prof_info_add, FALSE)

Reset(dw_prof_info_add)
InsertRow(dw_prof_info_add, 0)
ScrollToRow(dw_prof_info_add, 1)

lv_i_Row = dwFind(dw_time, "time_type = " + '"' + '"", 0, RowCount(dw_time))
IF (lv_i_Row = 0) THEN
    InsertRow(dw_time, 1)
    ScrollToRow(dw_time, 1)
ELSE
    ScrollToRow(dw_time, lv_i_Row)
END IF

SetFocus(dw_prof_info_add)

SetRedraw(dw_prof_info_xref, TRUE)
SetRedraw(dw_prof_info_add, TRUE)

//Reset the structure values
s_InfoParms.first_time = TRUE
s_InfoParms.class_preference_1 = 0
s_InfoParms.class_preference_2 = 0
s_InfoParms.class_preference_3 = 0
s_InfoParms.class_preference_4 = 0
s_InfoParms.previous_class_1 = 0
s_InfoParms.previous_class_2 = 0
s_InfoParms.previous_class_3 = 0
s_InfoParms.previous_class_4 = 0
s_InfoParms.chair = ""
s_InfoParms.priority = 0
s_InfoParms.degreeEarned = ""
s_InfoParms.yrs CsbsJu = 0
s_InfoParms.tenured = ""

iv_b_ChangesMade = FALSE

//Command Button Control
cb_reset.enabled = FALSE
cb_more.enabled = FALSE
cb_cancel.enabled = TRUE

End of Script

Script for: lbmouseup event
//Used to keep track of the default button.

//Local Variables
Window: wo_prof_maint
Library: e:\thesis\appl\schedule.pbl
Date: 5/1/95    Time: 21:32:19

//End Local Variables

iv_cb_CurrentDefault.default = FALSE
iv_cb_CurrentDefault = cb_add

End of Script

CommandButton: cb_select
X = 449    Y = 1601    Width = 307    Height = 109
TabOrder = 80    Visible = true    Text = "Select"

Script for: clicked event
//Triggers the "doubleclicked" event of the prof_info_xref datawindow.

//Local Variables

//End Local Variables

TriggerEvent(dw_prof_info_xref, "DoubleClicked")

End of Script

Script for: lbuttondown event
//Used to keep track of the default button.

//Local Variables

//End Local Variables

iv_cb_CurrentDefault.default = FALSE
iv_cb_CurrentDefault = cb_select
End of Script

DataWindow: dw_prof_info_add
X = 37    Y = 237    Width = 1829    Height = 109
TabOrder = 30    Visible = true    Enabled = true    DataObject = "d_prof_info_add"
LiveScroll = true    BorderStyle = stylebox!

Script for: editchanged event
//After any data is entered into the datawindow the more and reset command buttons are enabled.

//Local Variables

//End Local Variables

cb_more.enabled = TRUE
cb_reset.enabled = TRUE

SelectRow(dw_prof_info_xref, 0, FALSE)

cb_select.enabled = FALSE
cb_delete.enabled = FALSE

End of Script

Script for: dwnkey event
//Attempts to control tabbing.

//Local Variables
String   lv_s_CurrentColumn
Boolean   lv_defaults[8]
//End Local Variables
lv_s_CurrentColumn = GetColumnNames(dw_prof_info_add)

IF (KeyDown(keyTab!) AND KeyDown(keyShift!) AND (lv_s_CurrentColumn = "course_number") THEN
  lv_b_TabDone = TRUE
  SetFocus(cb_ok)
  Return
ELSEIF (KeyDown(keyTab!) AND (lv_s_CurrentColumn = "emp_type")) THEN
  lv_b_TabDone = TRUE
  IF (dw_time.enabled) THEN
    SetFocus(dw_time)
    Return
  ELSEIF (cb_more.enabled) THEN
    lv_cb_CurrentDefault.default = FALSE
    cb_more.default = TRUE
    lv_cb_CurrentDefault = cb_more
    SetFocus(cb_more)
    Return
  ELSEIF (cb_add.enabled) THEN
    SetFocus(cb_add)
    Return
  ELSEIF (cb_select.enabled) THEN
    SetFocus(cb_select)
    Return
  ELSEIF (cb_delete.enabled) THEN
    SetFocus(cb_delete)
    Return
  ELSEIF (cb_update.enabled) THEN
    SetFocus(cb_update)
    Return
  ELSEIF (cb_reset.enabled) THEN
    SetFocus(cb_reset)
    Return
  ELSEIF (cb_cancel.enabled) THEN
    SetFocus(cb_cancel)
    Return


ELSEIF (cb_ok.enabled) THEN
    SetFocus(cb_ok)
    Return
ELSE
    Return
END IF
END IF

End of Script

DataWindow: dw_prof_info_xref
X = 23  Y = 473  Width = 2721  Height = 1053
TabOrder = 50  Visible = true  Enabled = true  DataObject = "d_prof_info"
VScrollBar = true  Border = true  LiveScroll = true  BorderStyle = stylebox!

Script for: clicked  event
//Selects the clicked row and enables/disables the appropriate command buttons.

//Local Variables
Integer  lv_i_SelectedRow
//End Local Variables

lv_i_Row = GetClickedRow(dw_prof_info_xref)

IF (lv_i_Row < 1) THEN
    Return
END IF

lv_i_SelectedRow = GetSelectedRow(dw_prof_info_xref, 0)
IF (lv_i_Row = lv_i_SelectedRow) THEN
    SelectRow(dw_prof_info_xref, 0, FALSE)
    cb_delete.enabled = FALSE
    cb_add.enabled = TRUE
Window: uo_prof_main
Library: e:\thesis\appl\schedule.pbl
Date: 5/1/95    Time: 21:32:19

iv_cb_CurrentDefault.default  = FALSE
cb_add.default = TRUE
iv_cb_CurrentDefault  = cb_add
cb_reset.enabled = FALSE
cb_update.enabled = FALSE
cb_select.enabled = FALSE

IF (iv_b_ChangesMade) THEN
  IF (MessageBox("Changes Made", "Do you wish to save changes?", Question!, YesNo!) = 1) THEN
    TriggerEvent(cb_update, "clicked")
  ELSE
    iv_b_ChangesMade = FALSE
  END IF
END IF

SetRedraw(dw_prof_info_add, FALSE)
SetRedraw(dw_prof_info_update, FALSE)

Reset(dw_prof_info_update)

InsertRow(dw_prof_info_update, 0)
ScrollToRow(dw_prof_info_update, 1)

dw_prof_info_add.visible = TRUE
dw_prof_info_update.visible = FALSE
dw_prof_info_add.enabled = TRUE
dw_prof_info_update.enabled = FALSE

SetRedraw(dw_prof_info_add, TRUE)
SetRedraw(dw_prof_info_update, TRUE)

SetFocus(dw_prof_info_add)
ELSE
  SelectRow(dw_prof_info_xref, 0, FALSE)
  SelectRow(dw_prof_info_xref, iv_i_ROW, TRUE)
IF (cb_add.enabled) THEN
    SetRedraw(dw_prof_info_add, FALSE)
    dw_prof_info_add.enabled = FALSE

    Reset(dw_prof_info_add)
    InsertRow(dw_prof_info_add, 0)
    ScrollToRow(dw_prof_info_add, 1)

    SetRedraw(dw_prof_info_add, TRUE)
END IF

// Command Button Control
cb_delete.enabled = TRUE
iv_cb_CurrentDefault.default = FALSE
cb_select.default = TRUE
iv_cb_CurrentDefault = cb_select
cb_add.enabled = FALSE
cb_reset.enabled = TRUE
cb_select.enabled = TRUE
END IF

End of Script

Script for: doubleclicked event
// Verifies that no changes have been made if data exists in update mode. Highlights the
// clicked row and gets all information regarding the faculty member and fills the prof_info_update
// datawindow and professor structure and goes into update mode.

// Local Variables
Char lv_c_Name[35]
Char lv_c_LastName[20]
Char lv_c_FirstName[15]
Integer lv_i_ArrayIndex
Integer lv_i_Loop
Window: uw_prof_maint
Library: e:\thesis\appl\schedule.pbl
Date: 5/1/95   Time: 21:32:19

Integer   lv_i_Length
Integer   lv_i_NameLength
Integer   lv_i_FindRow
String    lv_s_Name
String    lv_s_FirstName
String    lv_s_LastName
String    lv_s_Active
String    lv_s_EmpType
String    lv_s_TimePreference
String    lv_s_FindTime

//End Local Variables

IF (lv_i_Row < 1) THEN
  SetFocus(dw_prof_info_add)
  Return
END IF

//Should be taken care of by clicked event ....
IF (lv_b_ChangesMade) THEN
  IF (MessageBox("Changes Made", "Do you wish to save changes?", Question!, YesNo!) = 1) THEN
    TriggerEvent(cb_update, "clicked")
  ELSE
    lv_b_ChangesMade = FALSE
  END IF
END IF

End IF

SetRedraw(dw_prof_info_update, FALSE)
SetRedraw(dw_prof_info_add, FALSE)

dw_prof_info_update.enabled   = TRUE
dw_prof_info_add.enabled      = FALSE

dw_prof_info_update.visible   = TRUE
dw_prof_info_add.visible      = FALSE

Reset (dw prof info add)
Window: uo_prof_maint
Library: e:\thesis\appl\schedule.pb1
Date: 5/1/95 Time: 21:32:19

InsertRow(dw_prof_info_add, 0)
ScrollToRow(dw_prof_info_add, 1)

Reset(dw_prof_info_update)
InsertRow(dw_prof_info_update, 0)
ScrollToRow(dw_prof_info_update, 1)

iv_i_UpdateRow = iv_i_Row
lv_s_Name = Trim(GetItemString(dw_prof_info_xref, iv_i_UpdateRow, "name"))
lv_s_Active = Trim(GetItemString(dw_prof_info_xref, iv_i_UpdateRow, "active"))
lv_s_EmpType = Trim(GetItemString(dw_prof_info_xref, iv_i_UpdateRow, "emp_type"))
lv_s_TimePreference = Trim(GetItemString(dw_prof_info_xref, iv_i_UpdateRow, "time_preference"))

lv_c_Name = lv_s_Name
lv_i_NameLength = Len(lv_c_Name)
lv_i_Length = last_name_length(lv_s_Name)

lv_i_ArrayIndex = 1
FOR lv_i_Loop = lv_i_Length + 3 TO lv_i_NameLength
    lv_c_FirstName[lv_i_ArrayIndex] = lv_c_Name[lv_i_Loop]
    lv_i_ArrayIndex = lv_i_ArrayIndex + 1
NEXT
lv_c_FirstName[lv_i_ArrayIndex] = Char(0)

lv_i_ArrayIndex = 1
FOR lv_i_Loop = 1 TO lv_i_Length
    lv_c_LastName[lv_i_ArrayIndex] = lv_c_Name[lv_i_Loop]
    lv_i_ArrayIndex = lv_i_ArrayIndex + 1
NEXT
lv_c_LastName[lv_i_ArrayIndex] = Char(0)

lv_s_FirstName = lv_c_FirstName
lv_s_LastName = lv_c_LastName

SetItem(dw_prof_info_update, 1, "last_name", lv_s_LastName)
Window: uo_prof_maint
Library: e:\thesis\appl\schedule.pbl
Date: 5/1/95 Time: 21:32:19

SetItem(dw_prof_info_update, 1, "first_name", lv_s_FirstName)
SetItem(dw_prof_info_update, 1, "active", lv_s_Active)
SetItem(dw_prof_info_update, 1, "emp_type", lv_s_EmpType)

lv_s_FindTime = "time_type = " + '"' + lv_s_TimePreference + '""
LV_i_FindRow = dwFind(dw_time, lv_s_FindTime, 0, RowCount(dw_time))

IF (lv_i_FindRow > 0) THEN
  ScrollToRow(dw_time, lv_i_FindRow)
END IF

s_InfoParms.add_mode = FALSE
s_InfoParms.first_time = FALSE
s_InfoParms.class_preference_1 = GetItemNumber(dw_prof_info_xref, iv_i_UpdateRow, "class_preference_1")
s_InfoParms.class_preference_2 = GetItemNumber(dw_prof_info_xref, iv_i_UpdateRow, "class_preference_2")
s_InfoParms.class_preference_3 = GetItemNumber(dw_prof_info_xref, iv_i_UpdateRow, "class_preference_3")
s_InfoParms.class_preference_4 = GetItemNumber(dw_prof_info_xref, iv_i_UpdateRow, "class_preference_4")

s_InfoParms.previous_class_1 = GetItemNumber(dw_prof_info_xref, iv_i_UpdateRow, "previous_class_1")

s_InfoParms.previous_class_2 = GetItemNumber(dw_prof_info_xref, iv_i_UpdateRow, "previous_class_2")

s_InfoParms.previous_class_3 = GetItemNumber(dw_prof_info_xref, iv_i_UpdateRow, "previous_class_3")

s_InfoParms.previous_class_4 = GetItemNumber(dw_prof_info_xref, iv_i_UpdateRow, "previous_class_4")

s_InfoParms.chair = Trim(GetItemString(dw_prof_info_xref, iv_i_UpdateRow, "chair"))

s_InfoParms.priority = GetItemNumber(dw_prof_info_xref, iv_i_UpdateRow, "priority")

s_InfoParms.degree_earned = Trim(GetItemString(dw_prof_info_xref, iv_i_UpdateRow, "high_degree_earned"))

s_InfoParms.yrs_csbsju = GetItemNumber(dw_prof_info_xref, iv_i_UpdateRow, "yrs_at_csbsju")

s_InfoParms.tenured = Trim(GetItemString(dw_prof_info_xref, iv_i_UpdateRow, "tenured"))
Window: uo_prof_maint
Library: e:\thesis\appl\schedule.pbl
Date: 5/1/95  Time: 21:32:19

SelectRow(dw_prof_info_xref, 0, FALSE)
SelectRow(dw_prof_info_xref, iv_i_UpdateRow, TRUE)

//Command Button Control
cb_more.enabled = TRUE
cb_add.enabled = FALSE
cb_delete.enabled = FALSE
cb_update.enabled = FALSE
cb_reset.enabled = TRUE
iv_cb_CurrentDefault.default = FALSE
cb_reset.default = TRUE
iv_cb_CurrentDefault = cb_reset
cb_select.enabled = FALSE

iv_b_ChangesMade = FALSE

SetFocus(dw_prof_info_update)
SetColumn(dw_prof_info_update, "active")

SetRedraw(dw_prof_info_add, TRUE)
SetRedraw(dw_prof_info_update, TRUE)

End of Script

Line: ln_1
BeginX = 33  BeginY = 361  EndX = 2743  EndY = 361
Visible = true  LineColor = 33554432  LineStyle = continuous!
LineThickness = 5

Line: ln_2
BeginX = 33  BeginY = 65  EndX = 2743  EndY = 65
Visible = true  LineColor = 33554432  LineStyle = continuous!
LineThickness = 5
Line: ln_3
BeginX = 33  BeginY = 69  EndX = 33  EndY = 365  LineColor = 33554432  LineStyle = continuous!
Visible = true  LineThickness = 5

Line: ln_4
BeginX = 2739  BeginY = 69  EndX = 2739  EndY = 365  LineColor = 33554432  LineStyle = continuous!
Visible = true  LineThickness = 5
DataWindow: d_prof_info_add
Library: e:\thesis\appl\schedule.pbl
Date: 5/2/95    Time: 17:25:10

Retrieve: Script
Arguments: None
Update Table: Not Allowed
Filter: None
Sort: None
Sparse: None

Column: last_name
  Format: "[general]"
  Border style: Shadow Box
  Validation: None
  Validation Message: None
  Tab Sequence: 10
  Initial Value: None
  Edit Style: Edit
  Edit limit: 0

Column: first_name
  Format: "[general]"
  Border style: Shadow Box
  Validation: None
  Validation Message: None
  Tab Sequence: 20
  Initial Value: None
  Edit Style: Edit
  Edit limit: 0

Column: active
DataWindow: d_prot_into_add
Library: e:\thesis\appl\schedule.pbl
Date: 5/2/95   Time: 17:25:10

Border style: Shadow Box
Validation: None
Validation Message: None
Tab Sequence: 30
Initial Value: None
    Edit Style: Edit
    Edit limit: 0

Column: emp_type
    Format: "[general]"
    Border style: Shadow Box
    Validation: None
    Validation Message: None
    Tab Sequence: 40
    Initial Value: None
    Edit Style: Edit
    Edit limit: 0
DataWindow: d_time_ddw
Library: e:\thesis\appl\schedule.pbl
Date: 5/2/95   Time: 17:12:01

Retrieve: PBSELECT(TABLE(NAME="time_xref") COLUMN(NAME="time_xref.time_type"))
Arguments: None
Update Table: time_xref
Filter: None
Sort: None
Sparse: None
Column: time_type
  Updateable: Yes
  Key: Yes
  Format: "[general]"
  Border style: None
  Validation: None
  Validation Message: None
  Tab Sequence: 10
  Initial Value: None
  Edit Style: DropDownDataWindow
    Name: d_time_list
    Data Column: time_type
    Display Column: time_type
DataWindow: d_time_list
Library: e:\thesis\appl\schedule.pbl
Date: 5/2/95   Time: 17:10:24

Retrieve: PBSELECT(TABLE(NAME=\"time_xref\") COLUMN(NAME=\"time_xref.time_type\"))
Arguments: None
Update Table: time_xref
Filter: None
Sort: None
Sparse: None
Column: time_type
  Updateable: Yes
  Key: Yes
  Format: \"[general]\"
  Border style: None
  Validation: None
  Validation Message: None
  Tab Sequence: 10
  Initial Value: None
  Edit Style: Edit
  Edit limit: 7
Retrieve: PSELECT

Arguments: arg_s_department
Update Table: professor_information
Filter: None
Sort: None
Sparse: None

Column: class_preference_1
  Updateable: Yes
  Key: No
  Format: "[general]"
  Border style: None
  Validation: None
  Validation Message: None
  Tab Sequence: 0
  Initial Value: None
  Edit Style: Edit
  Edit limit: 0

Column: class_preference_2
  Updateable: Yes
  Key: No
  Format: "[general]"
  Border style: None
  Validation: None
  Validation Message: None
  Tab Sequence: 0
DataWindow: d_prof_into
Library: e:\thesis\appl\schedule.pbl
Date: 5/2/95    Time: 17:26:39

Edit Style: Edit
Edit limit: 0

Column: class_preference_3
  Updateable: Yes
  Key: No
  Format: "[general]"
  Border style: None
  Validation: None
  Validation Message: None
  Tab Sequence: 0
  Initial Value: None
  Edit Style: Edit
  Edit limit: 0

Column: class_preference_4
  Updateable: Yes
  Key: No
  Format: "[general]"
  Border style: None
  Validation: None
  Validation Message: None
  Tab Sequence: 0
  Initial Value: None
  Edit Style: Edit
  Edit limit: 0

Column: previous_class_1
DataWindow: d_prot_into
Library: e:\thesis\appl\schedule.pbl
Date: 5/2/95    Time: 17:26:39

Key: No
Format: "[general]"
Border style: None
Validation: None
Validation Message: None
Tab Sequence: 0
Initial Value: None
Edit Style: Edit
Edit limit: 0

Column: previous_class_2
Updateable: Yes
Key: No
Format: "[general]"
Border style: None
Validation: None
Validation Message: None
Tab Sequence: 0
Initial Value: None
Edit Style: Edit
Edit limit: 0

Column: previous_class_3
Updateable: Yes
Key: No
Format: "[general]"
Border style: None
DataWindow: d_prot_info
Library: e:\thesis\appl\schedule.pbl
Date: 5/2/95  Time: 17:26:39

Edit limit: 0
Column: yrs_at_csbsju
  Updateable: Yes
  Key: No
  Format: "[general]"
  Border style: None
  Validation: None
  Validation Message: None
  Tab Sequence: 0
  Initial Value: None
  Edit Style: Edit
  Edit limit: 0

Column: high_degree_earned
  Updateable: Yes
  Key: No
  Format: "[general]"
  Border style: None
  Validation: None
  Validation Message: None
  Tab Sequence: 0
  Initial Value: None
  Edit Style: Edit
  Edit limit: 0

Column: priority
  Updateable: Yes
Column: emp_type
  Updateable: Yes
  Key: No
  Format: "[general]"
  Border style: None
  Validation: None
  Validation Message: None
  Tab Sequence: 0
  Initial Value: None
  Edit Style: Edit
  Edit limit: 0

Column: time_preference
  Updateable: Yes
  Key: No
  Format: "[general]"
  Border style: None
  Validation: None
  Validation Message: None
  Tab Sequence: 0
  Initial Value: None
  Edit Style: Edit
  Edit limit: 7
<table>
<thead>
<tr>
<th>Name Combined</th>
</tr>
</thead>
<tbody>
<tr>
<td>Header</td>
</tr>
<tr>
<td>name_combined</td>
</tr>
<tr>
<td>Detail</td>
</tr>
<tr>
<td>Summary</td>
</tr>
<tr>
<td>Footer</td>
</tr>
</tbody>
</table>
DataWindow: d_professor
Library: e:\thesis\appl\schedule.pbl
Date: 5/3/95   Time: 23:26:31

Retrieve: PBASELECT(TABLE(NAME="professor_xref") COLUMN(NAME="professor_xref.name_combined"))
Arguments: None
Update Table: professor_xref
Filter: None
Sort: None
Sparse: None
Column: name_combined
  Updateable: Yes
  Key: Yes
  Format: "[general]"
  Border style: None
  Validation: None
  Validation Message: None
  Tab Sequence: 10
  Initial Value: None
  Edit Style: Edit
  Edit limit: 35
Function : last_name_length

//Given a name (last name, first name) will determine the length
//of the last name so the name can be broken down into first and
//last names.

//Function variables
Char  fv_c_NameArray[35]
Integer  fv_i_NameLength
Integer  fv_i_LastNameLength
Integer  fv_i_LoopIndex
//End Function variables

fv_i_NameLength  =  Len(arg_name)
fv_c_NameArray  =  arg_name
fv_i_LastNameLength  =  0

IF  (fv_i_NameLength > 0)  THEN
    FOR  fv_i_LoopIndex = 1  TO  fv_i_NameLength
        IF  (fv_c_NameArray[fv_i_LoopIndex]  =  ",")  THEN
            EXIT
        ELSE
            fv_i_LastNameLength  =  fv_i_LastNameLength  +  1
        END IF
    NEXT
END IF

Return  fv_i_LastNameLength
Function: calculate_priority

// This function is passed a datawindow containing the professor information.  
// It will recalculate the calculated priority field for the department.

// Local Variables
Integer lv_i_FirstPriority
Integer lv_i_Loop
Integer lv_i_NumberOfProfessors
Integer lv_i_Priority
Integer lv_i_CurrentPriority
Integer lv_i_CurrentProf
Integer lv_i_CurrentOtherInfo
Integer lv_i_CurrentPriorityInfo
String lv_s_FilterString
// End Local Variables

lv_s_FilterString = "active = 'y'"

SetFilter(fv_dw_professors, lv_s_FilterString)
Filter(fv_dw_professors)

SetSort(fv_dw_professors, "priority A, tenured D, yrs_at_csbsju D, high_degree_earned D")
Sort(fv_dw_professors)

lv_i_NumberOfProfessors =RowCount(fv_dw_professors)

IF (lv_i_NumberOfProfessors <= 0) THEN
  Return(fv_dw_professors)
END IF

lv_i_CurrentPriority = 1
lv_i_FirstPriority = GetItemNumber(fv_dw_professors, 1, "priority")

// See if override priority can be used.  If the first line in datawindow <> 0 then override priorities are
IF (lv_i_FirstPriority <> 0) THEN
  FOR lv_i_Loop = 1 to lv_i_NumberOfProfessors
    lv_i_Priority = GetItemNumber(fv_dw_professors, lv_i_Loop, "priority")
    SetItem(fv_dw_professors, lv_i_Loop, "calculated_priority", lv_i_Priority)
  NEXT
lv_i_CurrentProf = 2
DO UNTIL ((lv_i_Priority <> 0) OR (lv_i_CurrentProf = lv_i_NumberOfProfessors + 1))
  lv_i_Priority = GetItemNumber(fv_dw_professors, lv_i_CurrentProf, "priority")
  lv_i_CurrentProf = lv_i_CurrentProf + 1
LOOP
// All entries use other information (not override priority)
IF (lv_i_CurrentProf = lv_i_NumberOfProfessors + 1) THEN
  lv_i_CurrentPriority = 1
  FOR lv_i_Loop = 1 to lv_i_NumberOfProfessors
    SetItem(fv_dw_professors, lv_i_Loop, "calculated_priority", lv_i_CurrentPriority)
    lv_i_CurrentPriority = lv_i_CurrentPriority + 1
  NEXT
ELSE
  lv_i_CurrentPriorityInfo = lv_i_CurrentProf - 1
  lv_i_CurrentOtherInfo = 1
  lv_i_CurrentPriority = 1
  DO UNTIL ((lv_i_CurrentPriorityInfo = lv_i_NumberOfProfessors + 1) AND (lv_i_CurrentOtherInfo = lv_i_CurrentPriority))
    lv_i_Loop = lv_i_CurrentPriority
    DO UNTIL ((lv_i_Loop = lv_i_Priority) OR (lv_i_CurrentOtherInfo = lv_i_CurrentProf - 1))
      SetItem(fv_dw_professors, lv_i_CurrentOtherInfo, "calculated_priority", lv_i_CurrentPriority)
      lv_i_CurrentPriority = lv_i_CurrentPriority + 1
      lv_i_CurrentOtherInfo = lv_i_CurrentOtherInfo + 1
    NEXT
  lv_i_CurrentOtherInfo = lv_i_CurrentOtherInfo + 1
  lv_i_Loop = lv_i_Loop + 1
  LOOP
DO UNTIL (lv_i_Priority <> lv_i_CurrentPriority)
  SetItem(fv_dw_professors, lv_i_CurrentPriorityInfo, "calculated_priority", lv_i_CurrentPriority)
  lv_i_CurrentPriority = lv_i_CurrentPriority + 1
  lv_i_CurrentPriorityInfo = lv_i_CurrentPriorityInfo + 1
IF (lv_i_CurrentPriorityInfo <> lv_i_NumberOfProfessors + 1) THEN
  lv_i_Priority = GetItemNumber(fv_dw_professors, lv_i_CurrentPriorityInfo, "priority")
END IF
END IF
END LOOP
IF (lv_i_Loop = lv_i_CurrentProf) THEN
  FOR lv_i_Loop = lv_i_CurrentPriority TO lv_i_NumberOfProfessors
    SetItem(fv_dw_professors, lv_i_CurrentPriorityInfo, "calculated_priority", lv_i_CurrentPriority)
    lv_i_CurrentPriority = lv_i_CurrentPriority + 1
    lv_i_CurrentPriorityInfo = lv_i_CurrentPriorityInfo + 1
  NEXT
// Done with calculating priorities
LOOP
  END IF
  END IF

Return fv_dw_professors
DataWindow: d_prof_info_update
Library: e:\thesis\appl\schedule.pbl
Date: 5/2/95   Time: 17:20:07

Retrieve: Script
Arguments: None
Update Table: Not Allowed
Filter: None
Sort: None
Sparse: None

<table>
<thead>
<tr>
<th>Column</th>
<th>Format</th>
<th>Border style</th>
<th>Validation</th>
<th>Validation Message</th>
<th>Tab Sequence</th>
<th>Initial Value</th>
<th>Edit Style</th>
<th>Edit limit</th>
</tr>
</thead>
<tbody>
<tr>
<td>last_name</td>
<td>&quot;[general]&quot;</td>
<td>None</td>
<td>None</td>
<td>None</td>
<td>0</td>
<td>None</td>
<td>Edit</td>
<td>0</td>
</tr>
<tr>
<td>first_name</td>
<td>&quot;[general]&quot;</td>
<td>None</td>
<td>None</td>
<td>None</td>
<td>0</td>
<td>None</td>
<td>Edit</td>
<td>0</td>
</tr>
<tr>
<td>active</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
DataWindow: d_prof_info_update
Library: e:\thesis\appl\schedule.pbl
Date: 5/2/95   Time: 17:20:07

Border style: Shadow Box
Validation: None
Validation Message: None
Tab Sequence: 10
Initial Value: None
Edit Style: Edit
Edit limit: 0

Column: emp_type
Format: "{general}"
Border style: Shadow Box
Validation: None
Validation Message: None
Tab Sequence: 40
Initial Value: None
Edit Style: Edit
Edit limit: 0
Window: w_more_prof_info  
Library: e:\thesis\appl\schedule.pbl 
Date: 5/1/95    Time: 21:29:23 

Instance Variables 
s_prof_info_parms    s_parms 

End of Instance Variables 

Script for: open   event 
//Adds a blank row into the datawindow. If in update mode, sets data into the datawindow
//from the information in the structure. If in add mode, the datawindow is empty. 

//Local Variables 
Boolean    lv_b_AddMode 
Boolean    lv_b_FirstTime 
//End Local Variables 

s_parms = Message.PowerObjectParm 

Reset(dw_prof_info)  
InsertRow(dw_prof_info, 0)  
ScrollToRow(dw_prof_info, 1)  

lv_b_AddMode = s_parms.add_mode 
lv_b_FirstTime = s_parms.first_time 

IF ((NOT lv_b_AddMode) OR (NOT lv_b_FirstTime)) THEN 

  //Set the prof info dw 
  SetItem(dw_prof_info, 1, "class_preference_1", s_parms.class_preference_1) 
  SetItem(dw_prof_info, 1, "class_preference_2", s_parms.class_preference_2) 
  SetItem(dw_prof_info, 1, "class_preference_3", s_parms.class_preference_3) 
  SetItem(dw_prof_info, 1, "class_preference_4", s_parms.class_preference_4) 
  SetItem(dw_prof_info, 1, "previous_class_1", s_parms.previous_class_1) 
  SetItem(dw_prof_info, 1, "previous_class_2", s_parms.previous_class_2) 
  SetItem(dw_prof_info, 1, "previous_class_3", s_parms.previous_class_3)
Window: w_more_prof_info
Library: e:\thesis\appl\schedule.pbl
Date: 5/1/95    Time: 21:29:23

SetItem(dw_prof_info, 1, "previous_class_4", sParms.previous_class_4)
SetItem(dw_prof_info, 1, "chair", sParms.chair)
SetItem(dw_prof_info, 1, "priority", sParms.priority)
SetItem(dw_prof_info, 1, "degree_earned", sParms.degree_earned)
SetItem(dw_prof_info, 1, "years_csbsju", sParms.yrs_csbsju)
SetItem(dw_prof_info, 1, "tenured", sParms.tenured)
END IF

SetFocus(dw_prof_info)
SetColumn(dw_prof_info, "class_preference_1")

End of Script

CommandButton: cb_ok
X = 1820    Y = 1337    Width = 247    Height = 109
TabOrder = 30    Visible = true    Enabled = true    Text = "&OK"

Script for: clicked event
//Closes the window w_more_prof_info after storing the new or modified data into the structure
//that is returned to the command button more to be processed.

//Local Variables
Integer  lv_i_ClassPref1
Integer  lv_i_ClassPref2
Integer  lv_i_ClassPref3
Integer  lv_i_ClassPref4
Integer  lv_i_PrevClass1
Integer  lv_i_PrevClass2
Integer  lv_i_PrevClass3
Integer  lv_i_PrevClass4
Integer  lv_i_Priority
Integer  lv_i_Years
String  lv_s_Chair
String lv_s_Degree
String lv_s_Tenured

//End Local Variables

lv_i_ClassPref1 = GetItemNumber(dw_prof_info, 1, "class_preference_1")
lv_i_ClassPref2 = GetItemNumber(dw_prof_info, 1, "class_preference_2")
lv_i_ClassPref3 = GetItemNumber(dw_prof_info, 1, "class_preference_3")
lv_i_ClassPref4 = GetItemNumber(dw_prof_info, 1, "class_preference_4")
lv_i_PrevClass1 = GetItemNumber(dw_prof_info, 1, "previous_class_1")
lv_i_PrevClass2 = GetItemNumber(dw_prof_info, 1, "previous_class_2")
lv_i_PrevClass3 = GetItemNumber(dw_prof_info, 1, "previous_class_3")
lv_i_PrevClass4 = GetItemNumber(dw_prof_info, 1, "previous_class_4")
lv_s_Chair = Trim(GetItemString(dw_prof_info, 1, "chair"))
lv_i_Priority = GetItemNumber(dw_prof_info, 1, "priority")
lv_s_Degree = Trim(GetItemString(dw_prof_info, 1, "degree_earned"))
lv_i_Years = GetItemNumber(dw_prof_info, 1, "years_csbsj")
lv_s_Tenured = Trim(GetItemString(dw_prof_info, 1, "tenured"))

//Validation
IF (IsNull(lv_i_ClassPref1)) THEN
    MessageBox("Validation Error", "All class preferences must be entered before leaving.", StopSign!, OK!)
    SetFocus(dw_prof_info)
    SetColumn(dw_prof_info, "class_preference_1")
    Return
ELSEIF (lv_i_ClassPref1 < 0) THEN
    MessageBox("Validation Error", "A course number is greater than zero.", StopSign!, OK!)
    SetFocus(dw_prof_info)
    SetColumn(dw_prof_info, "class_preference_1")
    Return
ELSEIF (IsNull(lv_i_ClassPref2)) THEN
    MessageBox("Validation Error", "All class preferences must be entered before leaving.", StopSign!, OK!)
    SetFocus(dw_prof_info)
    SetColumn(dw_prof_info, "class_preference_2")
    Return
 Return
ELSEIF (lv_i_PrevClass3 < 0) THEN
    MessageBox("Validation Error", "A course number is greater than zero.", StopSign!, OK!)
    SetFocus(dw_prof_info)
    SetColumn(dw_prof_info, "previous_class_3")
    Return
ELSEIF (lv_i_PrevClass4 < 0) THEN
    MessageBox("Validation Error", "A course number is greater than zero.", StopSign!, OK!)
    SetFocus(dw_prof_info)
    SetColumn(dw_prof_info, "previous_class_4")
    Return
ELSIF (IsNull(lv_s_Chair) OR (lv_s_Chair = "")) THEN
    MessageBox("Validation Error", "The department chair field must be entered before leaving.", StopSign!, OK!)
    SetFocus(dw_prof_info)
    SetColumn(dw_prof_info, "chair")
    Return
ELSIF (IsNull(lv_i_Priority)) THEN
    IF ((IsNull(lv_s_Degree) OR (lv_s_Degree = "")) AND (IsNull(lv_i_Years)) AND (IsNull(lv_s_Tenured) OR (lv_s_Tenured = ""))) THEN
        MessageBox("Validation Error", "Either the override priority or other information must be entered before leaving.", StopSign!, OK!)
        SetFocus(dw_prof_info)
        SetColumn(dw_prof_info, "priority")
        Return
    ELSEIF (IsNull(lv_s_Degree) OR (lv_s_Degree = "")) THEN
        MessageBox("Validation Error", "The highest degree earned must be entered before leaving.", StopSign!, OK!)
        SetFocus(dw_prof_info)
        SetColumn(dw_prof_info, "degree_earned")
        Return
    ELSEIF (IsNull(lv_i_Years)) THEN
        MessageBox("Validation Error", "The number of years at CSB/SJU must be entered before leaving.", StopSign!, OK!)
        SetFocus(dw_prof_info)
        SetColumn(dw_prof_info, "years_csbsju")
Return
ELSEIF (IsNull(lv_s_Tenured) OR (lv_s_Tenured = "")) THEN
    MessageBox("Validation Error", "The tenured field must be entered before leaving.", StopSign!, OK!)
    SetFocus(dw_prof_info)
    SetColumn(dw_prof_info, "tenured")
    Return
END IF
ELSEIF (lv_i_Priority < 0) THEN
    MessageBox("Validation Error", "The priority should be greater than zero.", StopSign!, OK!)
    SetFocus(dw_prof_info)
    SetColumn(dw_prof_info, "priority")
    Return
END IF

s_parms.first_time = FALSE
s_parms.class_preference_1 = lv_i_ClassPref1
s_parms.class_preference_2 = lv_i_ClassPref2
s_parms.class_preference_3 = lv_i_ClassPref3
s_parms.class_preference_4 = lv_i_ClassPref4
s_parms.previous_class_1 = lv_i_PrevClass1
s_parms.previous_class_2 = lv_i_PrevClass2
s_parms.previous_class_3 = lv_i_PrevClass3
s_parms.previous_class_4 = lv_i_PrevClass4
s_parms.chair = lv_s_Chair
s_parms.priority = lv_i_Priority
s_parms.degree_earned = lv_s_Degree
s_parms.yrs_cmbsju = lv_i_Years
s_parms.tenured = lv_s_Tenured

PostEvent(uo_prof_maint.cb_more, "uponreturn")

CloseWithReturn(w_more_prof_info, s_parms)

End of Script
Window: w_more_prof_info
Library: e:\thesis\appl\schedule.pbl
Date: 5/1/95 Time: 21:29:23

CommandButton: cb_cancel
X = 1473 Y = 1337 Width = 247 Height = 109
TabOrder = 20 Visible = true Enabled = true Text = "&Cancel"
Default = true

Script for: clicked event
//Closes window w_more_prof_info without setting the information into the structure to be returned.

close(w_more_prof_info)

End of Script

DataWindow: dw_prof_info
X = 65 Y = 65 Width = 2003 Height = 1241
TabOrder = 10 Visible = true Enabled = true DataObject = "d_more_prof_info"
LiveScroll = true BorderStyle = stylebox!
<table>
<thead>
<tr>
<th>Class Preferences:</th>
<th>Previous Classes Taught:</th>
</tr>
</thead>
<tbody>
<tr>
<td>class_pre</td>
<td>previous_</td>
</tr>
<tr>
<td>class_pre</td>
<td>previous_</td>
</tr>
<tr>
<td>class_pre</td>
<td>previous_</td>
</tr>
<tr>
<td>class_pre</td>
<td>previous_</td>
</tr>
<tr>
<td>Department Chair:</td>
<td>chair</td>
</tr>
<tr>
<td>Override Priority:</td>
<td>priority</td>
</tr>
<tr>
<td>Highest Degree Earned:</td>
<td>degree_ea</td>
</tr>
<tr>
<td>Number of Years at CSU/JU:</td>
<td>years_cs</td>
</tr>
<tr>
<td>Tenured:</td>
<td>tenu</td>
</tr>
<tr>
<td>Column: class_preference_1</td>
<td>Column: class_preference_2</td>
</tr>
<tr>
<td>---------------------------------------------------</td>
<td>---------------------------------------------------</td>
</tr>
<tr>
<td>Format: &quot;[general]&quot;</td>
<td>Format: &quot;[general]&quot;</td>
</tr>
<tr>
<td>Border style: Shadow Box</td>
<td>Border style: Shadow Box</td>
</tr>
<tr>
<td>Validation: None</td>
<td>Validation: None</td>
</tr>
<tr>
<td>Validation Message: None</td>
<td>Validation Message: None</td>
</tr>
<tr>
<td>Tab Sequence: 10</td>
<td>Tab Sequence: 20</td>
</tr>
<tr>
<td>Initial Value: None</td>
<td>Initial Value: None</td>
</tr>
<tr>
<td>Edit Style: Edit</td>
<td>Edit Style: Edit</td>
</tr>
<tr>
<td>Edit limit: 0</td>
<td>Edit limit: 0</td>
</tr>
</tbody>
</table>
Border style: Shadow Box
Validation: None
Validation Message: None
Tab Sequence: 30
Initial Value: None
Edit Style: Edit
Edit limit: 0

Column: class_preference_4
Format: "[general]"
Border style: Shadow Box
Validation: None
Validation Message: None
Tab Sequence: 40
Initial Value: None
Edit Style: Edit
Edit limit: 0

Column: chair
Format: "[general]"
Border style: Shadow Box
Validation: None
Validation Message: None
Tab Sequence: 90
Initial Value: None
Edit Style: Edit
Edit limit: 0
DataWindow: d_more_prof_info
Library: e:\thesis\appl\schedule.pbl
Date: 5/2/95   Time: 17:30:24

Format: "[general]"
Border style: Shadow Box
Validation: None
Validation Message: None
Tab Sequence: 120
Initial Value: None
Edit Style: Edit
Edit limit: 0

Column: tenured
Format: "[general]"
Border style: Shadow Box
Validation: None
Validation Message: None
Tab Sequence: 130
Initial Value: None
Edit Style: Edit
Edit limit: 0

Column: degree_earned
Format: "[general]"
Border style: Shadow Box
Validation: None
Validation Message: None
Tab Sequence: 110
Initial Value: None
Edit Style: Edit
Column: priority
  Format: "[general]"
  Border style: Shadow Box
  Validation: None
  Validation Message: None
  Tab Sequence: 100
  Initial Value: None
  Edit Style: Edit
  Edit limit: 0

Column: previous_class_4
  Format: "[general]"
  Border style: Shadow Box
  Validation: None
  Validation Message: None
  Tab Sequence: 80
  Initial Value: None
  Edit Style: Edit
  Edit limit: 0

Column: previous_class_3
  Format: "[general]"
  Border style: Shadow Box
  Validation: None
  Validation Message: None
  Tab Sequence: 70
  Initial Value: None
Column: previous_class_2
  Format: "[general]"
  Border style: Shadow Box
  Validation: None
  Validation Message: None
  Tab Sequence: 60
  Initial Value: None
  Edit Style: Edit
  Edit limit: 0

Column: previous_class_1
  Format: "[general]"
  Border style: Shadow Box
  Validation: None
  Validation Message: None
  Tab Sequence: 50
  Initial Value: None
  Edit Style: Edit
  Edit limit: 0
Window: w_scheduling
Library: e:\thesis\appi\schedule.pbl
Date: 5/1/95    Time: 20:37:42

Window: w_scheduling
X = 910        Y = 577        Width = 2844      Height = 1905
Visible = true  Enabled = true  TitleBar = true  Title = "Class Scheduling"
ControlMenu = true  MinBox = true  MaxBox = true  Resizable = true
WindowType = main!  WindowState = normal!  BackColor = 12632256

Instance Variables
Integer    iv_ia_ProfClass[20, 6]
Integer    iv_day1 = 1
Integer    iv_day2 = 2
Integer    iv_day3 = 4
Integer    iv_day4 = 8
Integer    iv_day5 = 16
Integer    iv_day6 = 32
Integer    iv_NumProfs

s_main_winParms    s_parms

End of Instance Variables

Script for: open event
//Retrieves data from the database regarding class information, faculty member information,
//and class schedule information.

//Local Variables
Integer    lv_i_Loop
Integer    lv_i_RowCounts
String     lv_s_Term
String     lv_s_Department
String     lv_s_FilterString
//End Local Variables

s_parms = Message.PowerObjectParm
SetTransObject(dw_class_info, SQLCA)
SetTransObject(dw_prof_info, SQLCA)
SetTransObject(dw_final_classes, SQLCA)

Retrieve(dw_class_info, lv_s.Term, lv_s.Department)
Retrieve(dw_prof_info, lv_s.Department)
Retrieve(dw_final_classes, lv_s.Term, lv_s.Department)

SetSort(dw_class_info, "course_num A, section_num A")
Sort(dw_class_info)

lv_s.FilterString = "active = 'y'"
SetFilter(dw_prof_info, lv_s.FilterString)
Filter(dw_prof_info)

SetSort(dw_prof_info, "calculated_priority")
Sort(dw_prof_info)

lv_NumProfs = RowCount(dw_prof_info)

lv_i_RowCounts = RowCount(dw_prof_extra)
IF (lv_i_RowCounts <> 0) THEN
    FOR lv_i_Loop = 1 to lv_i_RowCounts
        DeleteRow(dw_prof_extra, 1)
    NEXT
END IF

lv_i_RowCounts = RowCount(dw_prof_okay)
IF (lv_i_RowCounts <> 0) THEN
    FOR lv_i_Loop = 1 to lv_i_RowCounts
        FOR lv_j_Loop = 1 to lv_j_RowCounts
            FOR lv_k_Loop = 1 to lv_k_RowCounts
Window: w_scheduling  
Library: e:\thesis\appl\schedule.pbl  
Date: 5/1/95    Time: 20:37:42  

DELETE(dw_prof_okay, 1)
NEXT
END IF

lv_i_RowCounts = RowCount(dw_prof_few)
IF (lv_i_RowCounts <> 0) THEN
   FOR lv_i_Loop = 1 to lv_i_RowCounts
      DELETE(dw_prof_few, 1)
      NEXT
   END IF

PostEvent(w_scheduling, "afteropen")

End of Script

Script for: afteropen  event
//If any data is in the class schedule datawindow the user is notified of what to do if they desire a new schedule.

//Local Variables
Integer   lv_i_FinalCount  
//End Variables

lv_i_FinalCount = RowCount(dw_final_classes)

IF (lv_i_FinalCount <> 0) THEN
   cb_clear.enabled = TRUE
   cb_start.enabled = FALSE
   MessageBox("Information", "A final schedule already exists for this term and department. ", Information!, OK!)
END IF
End of Script

DataWindow: dw_class_info
X = 0
Y = 1181
Width = 2817
Height = 433
TabOrder = 10
DataObject = "d_class_info_schedule"
TitleBar = true
Title = "Class Information"
HScrollBar = true
VScrollBar = true
Border = true
LiveScroll = true
BorderStyle = stylebox!

DataWindow: dw_extra_class
X = 1454
Y = 1361
Width = 494
Height = 361
TabOrder = 20
DataObject = "d_extra_classes"
TitleBar = true
Title = "Extra Classes"
Border = true
LiveScroll = true
BorderStyle = stylebox!

DataWindow: dw_prof_few
X = 5
Y = 421
Width = 2807
Height = 365
TabOrder = 60
DataObject = "d_assign_prof_long"
TitleBar = true
Title = "Prof Short"
HScrollBar = true
VScrollBar = true
Border = true
LiveScroll = true
BorderStyle = stylebox!

DataWindow: dw_prof_okay
X = 0
Y = 793
Width = 2807
Height = 385
TabOrder = 50
DataObject = "d_assign_prof_long"
TitleBar = true
Title = "Prof done"
HScrollBar = true
VScrollBar = true
Border = true
LiveScroll = true
BorderStyle = stylebox!

CommandButton: cb_start
X = 188
Y = 1625
Width = 247
Height = 109
Script for: clicked event
// Initializes the faculty member array, gets ready for producing a class schedule,
// and controls the scheduling algorithm.

// Local Variables
Character lv_ca_RmpType[11]
Integer lv_i_Answer
Integer lv_i_TotalNumProf
Integer lv_i_TotalNumClass
Integer lv_i_CurrentProf
Integer lv_i_AddedProf
Integer lv_i_Priority
Integer lv_i_Loop1
Integer lv_i_Loop2
String lv_s_Name
String lv_s_Chair
// End Local Variables

SetPointer(Hourglass!)

SetRedraw(dw_final_classes, FALSE)

// Initialize the array for storing the class periods a professor has assigned
FOR lv_i_Loop1 = 1 TO 20
    FOR lv_i_Loop2 = 1 TO 6
        lv_i_a_ProfClass[lv_i_Loop1, lv_i_Loop2] = 0
    NEXT
NEXT

// Sort the professors by priority
SetSort(dw_prof_info, "calculated_priority A")
Sort(dw_prof_info)

// Establish the number of classes and professors
lv_i_TotalNumProf    = RowCount(dw_prof_info)
lv_i_TotalNumClass   = RowCount(dw_class_info)

//Set-up the main prof class datawindow (dw_prof_extra)
FOR lv_i_CurrentProf = 1 TO lv_i_TotalNumProf

  lv_s_Name    = Trim(GetItemString(dw_prof_info, lv_i_CurrentProf, "name"))
  lv_i_Priority = GetItemNumber(dw_prof_info,lv_i_CurrentProf, "calculated_priority")
  lv_s_Chair   = Trim(GetItemString(dw_prof_info,lv_i_CurrentProf, "chair"))
  lv_ca_EmpType = Trim(GetItemString(dw_prof_info,lv_i_CurrentProf, "emp_type"))

  lv_i_AddedProf = InsertRow(dw_prof_extra, 0)

  SetItem(dw_prof_extra, lv_i_AddedProf, "name", lv_s_Name)
  SetItem(dw_prof_extra, lv_i_AddedProf, "priority", lv_i_Priority)
  SetItem(dw_prof_extra, lv_i_AddedProf, "current_preference", 0)
  SetItem(dw_prof_extra, lv_i_AddedProf, "current_previous", 0)

  IF (lv_s_Chair = "y") THEN
    SetItem(dw_prof_extra, lv_i_AddedProf, "credits", 0.5)
  ELSE
    SetItem(dw_prof_extra, lv_i_AddedProf, "credits", 0)
  END IF

  IF (lv_ca_EmpType[1] = "p") THEN
    SetItem(dw_prof_extra, lv_i_AddedProf, "needed_credits", 2)
  ELSE
    SetItem(dw_prof_extra, lv_i_AddedProf, "needed_credits", 3.5)
  END IF

NEXT

//Phase 1 -- Assign as many of the classes by preference
TriggerEvent(cb_start, "phase1")

//Phase 2 -- Sort the entries into to many classes, too few classes, okay number of classes
TriggerEvent(cb_start, "phase2")
//Phase 3 -- finish assigning any remaining classes (if any) according to previous experience
TriggerEvent(cb_start, "phase3")

//Phase 4 -- Assign any remaining classes (if any) to any professor that has that slot open
lv_i_Answer=MessageBox("Seeking Information", "At this time, would you like to have any remaining classes " + & "assigned to a faculty member available during this classes time? The other option is to wait until the end " + & "to assign the remaining classes.", Question!, YesNo!)
IF (lv_i_Answer = 1) THEN
  TriggerEvent(cb_start, "phase4")
END IF

//Phase 5 -- Move all unassigned classes (if any) to a separate datawindow
TriggerEvent(cb_start, "phase5")

//Phase 6 -- Let professors with the least amount of credits chose from the last acquired courses // of the professors with extra credits (based on preference)
TriggerEvent(cb_start, "phase6")

//Phase 7 -- Let professors with the least amount of credits chose from the last acquired courses // of the professors with extra credits (based on previously taught)
TriggerEvent(cb_start, "phase7")

//Phase 8 -- Assign any remaining classes to any professor that has that time slot open
TriggerEvent(cb_start, "phase8")

//Phase 9 -- Move all assigned classes to the done datawindow ... notify if there are any leftover classes
TriggerEvent(cb_start, "phase9")

SetRedraw(dw_final_classes, TRUE)

cb_start.enabled = FALSE
cb_clear.enabled = TRUE
cb_cancel.enabled = TRUE
// Local Variables

Boolean lv_b_Done
Boolean lv_b_FoundClass
Boolean lv_b_Continue = TRUE
Boolean lv_b_Next
Boolean lv_b_Above
Boolean lv_b_NotEmpty = TRUE
Decimal lv_d_TotalCredits
Decimal lv_d_NeedCredits
Integer lv_i_TotalNumProf
Integer lv_i_TotalNumClass
Integer lv_i_CurrentProf
Integer lv_i_CurrentClass
Integer lv_i_CurrentCourseNum
Integer lv_i_CurrentPreference
Integer lv_i_PrefLoop
Integer lv_i_OtherSection
Integer lv_i_Priority
Integer lv_i_Loop1
Integer lv_i_CourseNum2
Integer lv_i_PrefCourseNum
String lv_s_TempSection
String lv_s_TempSectionNum
String lv_s_SectionNum
String lv_s_SectionNum2
String lv_s_Days1
String lv_s_Days2
String lv_s_Lab
String lv_s_LabDays1
Window: w_scheduling
Library: e:\thesis\appl\schedule.pbl
Date: 5/1/95 Time: 20:37:42

String lv_s_LabDays2
String lv_s_TimePref
Time lv_t_StartTime1
Time lv_t_StartTime2
Time lv_t_EndTime1
Time lv_t_EndTime2
Time lv_t_LabStartTime1
Time lv_t_LabStartTime2
Time lv_t_LabEndTime1
Time lv_t_LabEndTime2
//End Local Variables

lv_i_TotalNumProf = RowCount(dw_prof_info)
lv_i_TotalNumClass = RowCount(dw_class_info)

FOR lv_i_PrefLoop = 1 TO 4
  FOR lv_i_CurrentProf = 1 to lv_i_TotalNumProf
    lv_b_FoundClass = FALSE
    DO WHILE (NOT lv_b_FoundClass)
      lv_i_CurrentPreference = GetItemNumber(dw_prof_extra, lv_i_CurrentProf, "current_preference")
      lv_i_CurrentPreference = lv_i_CurrentPreference + 1
      SetItem(dw_prof_extra, lv_i_CurrentProf, "current_preference", lv_i_CurrentPreference)
    IF (lv_i_CurrentPreference > 4) THEN
      //No more preferences -- assign no more classes until phase 3
      //Should exit just past the first DO WHILE loop
      EXIT
    ELSEIF (lv_i_CurrentPreference = 1) THEN
      lv_i_PrefCourseNum = GetItemNumber(dw_prof_info, lv_i_CurrentProf, "class_preference_1")
    ELSEIF (lv_i_CurrentPreference = 2) THEN
      lv_i_PrefCourseNum = GetItemNumber(dw_prof_info, lv_i_CurrentProf, "class_preference_2")
    ELSEIF (lv_i_CurrentPreference = 3) THEN
      lv_i_PrefCourseNum = GetItemNumber(dw_prof_info, lv_i_CurrentProf, "class_preference_3")
    ELSEIF (lv_i_CurrentPreference = 4) THEN
      lv_i_PrefCourseNum = GetItemNumber(dw_prof_info, lv_i_CurrentProf, "class_preference_4")
    END IF
lv_i_CurrentClass = 1
lv_b_Done = FALSE
DO WHILE (NOT lv_b_Done)
  lv_b_Continue = TRUE
  lv_b_Next = FALSE
  lv_b_Above = FALSE
  lv_i_CurrentCourseNum = GetItemNumber(dw_class_info, lv_i_CurrentClass, "course_num")
  IF (lv_i_CurrentCourseNum = lv_i_PrefCourseNum) THEN
    lv_s_TimePref = Trim(GetItemString(dw_prof_info, lv_i_CurrentProf, "time_preference"))
    lv_t_StartTime1 = GetItemTime(dw_class_info, lv_i_CurrentClass, "start_time")
  END IF
  IF (lv_s_TimePref = "am") THEN
    IF (NOT (lv_t_StartTime1 <= Time(11,20,0))) THEN
      lv_b_Continue = FALSE
    END IF
  ELSE IF (lv_s_TimePref = "pm") THEN
    IF (NOT (lv_t_StartTime1 >= Time(11,20,0))) THEN
      lv_b_Continue = FALSE
    END IF
  END IF
END IF

IF (lv_b_Continue) THEN
  lv_s_Lab = Trim(GetItemString(dw_class_info, lv_i_CurrentClass, "lab"))
  IF (lv_s_Lab = "y") THEN
    lv_s_SectionNum = Trim(GetItemString(dw_class_info, lv_i_CurrentClass, "section_num"))
    lv_s_Days1 = Trim(GetItemString(dw_class_info, lv_i_CurrentClass, "days"))
    lv_t_EndTime1 = GetItemTime(dw_class_info, lv_i_CurrentClass, "end_time")
    IF (lv_i_CurrentClass + 1 <= lv_i_TotalNumClass) THEN
      lv_i_CourseNum2 = GetItemNumber(dw_class_info, lv_i_CurrentClass + 1, "course_num")
      lv_s_Days2 = Trim(GetItemString(dw_class_info, lv_i_CurrentClass + 1, "days"))
      lv_t_StartTime2 = GetItemTime(dw_class_info, lv_i_CurrentClass + 1, "start_time")
    END IF
  END IF
lv_t_EndTime2 = GetItemTime(dw_class_info, lv_i_CurrentClass + 1, "end_time")

IF ((lv_i_CourseNum2 = lv_i_CurrentCourseNum) AND (lv_s_Days1 = lv_s_Days2) AND (lv_t_StartTime1 = lv_t_StartTime2) AND (lv_t_EndTime1 = lv_t_EndTime2)) THEN
    lv_b_Next = TRUE
    lv_b_Above = FALSE
    lv_i_OtherSection = lv_i_CurrentClass + 1
END IF
ELSEIF (lv_i_CurrentClass - 1 >= 1) THEN
    lv_i_CourseNum2=GetItemNumber(dw_class_info, lv_i_CurrentClass - 1, "course_num")
    lv_s_Days2 = Trim(GetItemString(dw_class_info, lv_i_CurrentClass - 1, "days"))
    lv_t_StartTime2=GetItemTime(dw_class_info, lv_i_CurrentClass - 1, "start_time")
    lv_t_EndTime2 = GetItemTime(dw_class_info, lv_i_CurrentClass - 1, "end_time")

IF ((lv_i_CourseNum2 = lv_i_CurrentCourseNum) AND (lv_s_Days1 = lv_s_Days2) AND (lv_t_StartTime1 = lv_t_StartTime2) AND (lv_t_EndTime1 = lv_t_EndTime2)) THEN
    lv_b_Next = FALSE
    lv_b_Above = TRUE
    lv_i_OtherSection = lv_i_CurrentClass - 1
END IF
END IF

lv_s_LabDays1 = Trim(GetItemString(dw_class_info, lv_i_CurrentClass, "lab_days"))
lv_t_LabStartTime1=GetItemTime(dw_class_info, lv_i_CurrentClass, "lab_start_time")

lv_t_LabEndTime1 = GetItemTime(dw_class_info, lv_i_CurrentClass, "lab_end_time")

IF (lv_b_Next OR lv_b_Above) THEN
    lv_s_LabDays2=Trim(GetItemString(dw_class_info, lv_i_OtherSection, "lab_days"))
    lv_t_LabStartTime2=GetItemTime(dw_class_info, lv_i_OtherSection, "lab_start_time")
    lv_t_LabEndTime2 = GetItemTime(dw_class_info, lv_i_OtherSection, "lab_end_time")
END IF
ELSIF

lv_s_SectionNum = Trim(GetItemString(dw_class_info, lv_i_CurrentClass, "section_num"))

lv_s_Days1 = Trim(GetItemString(dw_class_info, lv_i_CurrentClass, "days"))

lv_t_EndTime1 = GetItemTime(dw_class_info, lv_i_CurrentClass, "end_time")

lv_s_LabDays1 = ""

lv_t_LabStartTime1 = Time(0, 0, 0)

lv_t_LabEndTime1 = Time(0, 0, 0)

lv_s_LabDays2 = ""

lv_t_LabStartTime2 = Time(0, 0, 0)

lv_t_LabEndTime2 = Time(0, 0, 0)

END IF

lv_i_Priority = GetItemNumber(dw_prof_extra, lv_i_CurrentProf, "priority")

lv_b_FoundClass = add_class_lab(lv_ia_ProfClass, lv_i_Priority, lv_s_Days1, lv_t_StartTime1, lv_t_EndTime1, &

lv_s_LabDays1, lv_t_LabStartTime1, lv_t_LabEndTime1, lv_s_LabDays2, lv_t_LabStartTime2, &

lv_t_LabEndTime2)

IF (lv_b_FoundClass) THEN

lv_i_Course = 0

lv_b_NotEmpty = TRUE

DO WHILE (lv_b_NotEmpty)

lv_i_Loop1 = lv_i_Loop1 + 1

lv_s_TempSection = Trim(GetItemString(dw_prof_extra, lv_i_CurrentProf, "section" + String(lv_i_Loop1)))

IF (ISNull(lv_s_TempSection) OR (lv_s_TempSection = "")) THEN

lv_b_NotEmpty = FALSE

END IF

LOOP

SetItem(dw_prof_extra, lv_i_CurrentProf, "course" + String(lv_i_Loop1), lv_i_CurrentCourseNum)

SetItem(dw_prof_extra, lv_i_CurrentProf, "section" + String(lv_i_Loop1), lv_s_SectionNum)
IF (lv_b_Next OR lv_b_Above) THEN
lv_s_SectionNum2 = Trim(GetItemString(dw_class_info, lv_i_OtherSection, "section_num"))
setItem(dw_prof_extra, lv_i_CurrentProf, "course_" + String(lv_i_Loop1 + 1), lv_i_CourseNum2)
setItem(dw_prof_extra, lv_i_CurrentProf, "section_" + String(lv_i_Loop1 + 1), lv_s_SectionNum2)
END IF
lv_d_TotalCredits = GetItemDecimal(dw_prof_extra, lv_i_CurrentProf, "credits")
IF (lv_s_Lab = "y") THEN
  IF (lv_b_Next OR lv_b_Above) THEN
    lv_d_TotalCredits = lv_d_TotalCredits + 2
    DeleteRow(dw_class_info, lv_i_OtherSection)
  ELSE
    lv_d_TotalCredits = lv_d_TotalCredits + 1.5
  END IF
ELSE
  lv_d_TotalCredits = lv_d_TotalCredits + 1
END IF
setItem(dw_prof_extra, lv_i_CurrentProf, "credits", lv_d_TotalCredits)
DeleteRow(dw_class_info, lv_i_CurrentClass)
lv_b_Done = TRUE
lv_i_TotalNumClass = RowCount(dw_class_info)
END IF
ELSE
  lv_b_Continue = TRUE
END IF
END IF
lv_i_CurrentClass = lv_i_CurrentClass + 1
IF (lv_i_CurrentClass > lv_i_TotalNumClass) THEN
  lv_b_Done = TRUE
END IF
END LOOP
LOOP
Window: w_scheduling
Library: e:\thesis\app1\schedule.pb1
Date: 5/1/95 Time: 20:37:42

IF (lv_i_TotalNumClass = 0) THEN
    EXIT
END IF
NEXT
IF (lv_i_TotalNumClass = 0) THEN
    EXIT
END IF
NEXT

End of Script

Script for: phase2 event
//Sort the faculty members with too many classes, faculty members with too few classes,
//and faculty members with an okay number of classes into the appropriate data windows.

//Local Variables
Decimal  lv_d_TotalCredits
Decimal  lv_d_NeedCredits
Integer   lv_i_TotalNumProf
Integer   lv_i_CurrentProf
Integer   lv_i_CurrentPreference
Integer   lv_i_CurrentRow
Integer   lv_i_CurrentPrevious
Integer   lv_i_AddedRow
Integer   lv_i_OtherSection
Integer   lv_i_Loop1
Integer   lv_i_Priority
Integer   lv_i_TempCourse
Integer   lv_i_CourseNum
Integer   lv_i_CourseNum2
Integer   lv_i_PrefCourseNum
Integer   lv_i_PastCourseNum
String    lv_s_TempSection
Window: w_scheduling
Library: e:\thesis\appl\schedule.pb1
Date: 5/1/95     Time: 20:37:42

String lv_s_Name  
//End Local Variables

lv_i_TotalNumProf = RowCount(dw_prof_extra)
lv_i_CurrentRow = 1

FOR lv_i_CurrentProf = 1 TO lv_i_TotalNumProf 
  lv_d_TotalCredits = GetItemDecimal(dw_prof_extra, lv_i_CurrentRow, "credits")
  lv_d_NeedCredits = GetItemDecimal(dw_prof_extra, lv_i_CurrentRow, "needed_credits")
  IF (lv_d_TotalCredits < lv_d_NeedCredits - 0.5) THEN 
    lv_s_Name = Trim(GetItemString(dw_prof_extra, lv_i_CurrentRow, "name"))
    lv_i_Priority = GetItemNumber(dw_prof_extra, lv_i_CurrentRow, "priority")
    lv_i_CurrentPreference = GetItemNumber(dw_prof_extra, lv_i_CurrentRow, "current_preference")
    lv_i_CurrentPrevious = GetItemNumber(dw_prof_extra, lv_i_CurrentRow, "current_previous")
    lv_i_AddedRow = InsertRow(dw_prof_extra, 0)
    SetItem(dw_prof_extra, lv_i_AddedRow, "name", lv_s_Name)
    SetItem(dw_prof_extra, lv_i_AddedRow, "priority", lv_i_Priority)
    SetItem(dw_prof_extra, lv_i_AddedRow, "needed_credits", lv_d_NeedCredits)
    SetItem(dw_prof_extra, lv_i_AddedRow, "current_preference", lv_i_CurrentPreference)
    SetItem(dw_prof_extra, lv_i_AddedRow, "current_previous", lv_i_CurrentPrevious)
    SetItem(dw_prof_extra, lv_i_AddedRow, "credits", lv_d_TotalCredits)
    FOR lv_i_Loop1 = 1 TO 8
      lv_i_TempCourse = GetItemNumber(dw_prof_extra, lv_i_CurrentRow, "course_" + String(lv_i_Loop1))
      lv_s_TempSection = Trim(GetItemString(dw_prof_extra, lv_i_CurrentRow, "section_" + String(lv_i_Loop1)))
      SetItem(dw_prof_extra, lv_i_AddedRow, "course_" + String(lv_i_Loop1), lv_i_TempCourse)
      SetItem(dw_prof_extra, lv_i_AddedRow, "section_" + String(lv_i_Loop1), lv_s_TempSection)
    NEXT
    DeleteRow(dw_prof_extra, lv_i_CurrentRow)
  ELSEIF ((lv_d_TotalCredits >= lv_d_NeedCredits - 0.5) AND (lv_d_TotalCredits <= lv_d_NeedCredits)) THEN
Window: w_scheduling
Library: e:\thesis\appl\schedule.pbl
Date: 5/1/95      Time: 20:37:42

lv_s_Name = Trim(GetItemString(dw_prof_extra, lv_i_CurrentRow, "name"))

lv_i_AddedRow = InsertRow(dw_prof_okay, 0)

SetItem(dw_prof_okay, lv_i_AddedRow, "name", lv_s_Name)
SetItem(dw_prof_okay, lv_i_AddedRow, "credits", lv_d_TotalCredits)

FOR lv_i_Loop1 = 1 TO 8
lv_s_TempCourse=GetItemNumber(dw_prof_extra, lv_i_CurrentRow, "course_" + String(lv_i_Loop1 ))
lv_s_TempSection=Trim(GetItemString(dw_prof_extra, lv_i_CurrentRow, "section_" + String(lv_i_Loop1 )))
    SetItem(dw_prof_okay, lv_i_AddedRow, "course_" + String(lv_i_Loop1), lv_s_TempCourse)
    SetItem(dw_prof_okay, lv_i_AddedRow, "section_" + String(lv_i_Loop1), lv_s_TempSection)
NEXT

DeleteRow(dw_prof_extra, lv_i_CurrentRow)
ELSE
    lv_i_CurrentRow = lv_i_CurrentRow + 1
END IF
NEXT

End of Script

Script for: phase3 event
//Assign any remaining classes (if any) to faculty members needing classes according to their previous experience.
//Sort these entities into the faculty members with too many classes, faculty members with too few classes, //and faculty members with an okay number of classes into the appropriate datawindows.

//Local Variables
Boolean lv_b_DoneClass
Boolean lv_b_Done
Boolean lv_b_FoundClass
Window: w_scheduling
Library: e:\thesis\appl\schedule.pbl
Date: 5/1/95 Time: 20:37:42

Time    lv_t_EndTime1
Time    lv_t_EndTime2
Time    lv_t_LabStartTime1
Time    lv_t_LabStartTime2
Time    lv_t_LabEndTime1
Time    lv_t_LabEndTime2
//End Local Variables

lv_i_TotalNumClass = RowCount(dw_class_info)
IF (lv_i_TotalNumClass > 0) THEN
  SetSort(dw_prof_few, "credits A, priority A")
  Sort(dw_prof_few)

  lv_b.DoneClass = FALSE
  lv_i_TotalFewProf = RowCount(dw_prof_few)

FOR lv_i_PastLoop = 1 TO 4
  FOR lv_i_CurrentProf = 1 TO lv_i_TotalFewProf
    lv_b.FoundClass = FALSE
    DO WHILE (NOT lv_b.FoundClass)
      lv_i_CurrentPrevious = GetItemNumber(dw_prof_few, lv_i_CurrentProf, "current_previous")
      lv_i_CurrentPrevious = lv_i_CurrentPrevious + 1
      SetItem(dw_prof_few, lv_i_CurrentProf, "current_previous", lv_i_CurrentPrevious)
      IF (lv_i_CurrentPrevious > 4) THEN
        // No more preferences -- assign no more classes until phase 3
        EXIT
      ELSEIF (lv_i_CurrentPrevious = 1) THEN
        lv_i_PastCourseNum = GetItemNumber(dw_prof_info, lv_i_CurrentProf, "previous_class_1")
      ELSEIF (lv_i_CurrentPrevious = 2) THEN
        lv_i_PastCourseNum = GetItemNumber(dw_prof_info, lv_i_CurrentProf, "previous_class_2")
      ELSEIF (lv_i_CurrentPrevious = 3) THEN
        lv_i_PastCourseNum = GetItemNumber(dw_prof_info, lv_i_CurrentProf, "previous_class_3")
      ELSEIF (lv_i_CurrentPrevious = 4) THEN
        lv_i_PastCourseNum = GetItemNumber(dw_prof_info, lv_i_CurrentProf, "previous_class_4")
    END IF

END IF
lv_i_CurrentClass = 1
lv_b_Done = FALSE
DO WHILE (NOT lv_b_Done)
    lv_b_Continue = TRUE
    lv_b_Next = FALSE
    lv_b_Above = FALSE
    IF (lv_i_CurrentClass > lv_i_TotalNumClass) THEN
        EXIT
    END IF

    lv_i_CurrentCourseNum = GetItemNumber(dw_class_info, lv_i_CurrentClass, "course_num")
    IF (lv_i_CurrentCourseNum = lv_i_PastCourseNum) THEN
        lv_s_TimePref = Trim(GetItemString(dw_prof_info, lv_i_CurrentProf, "time_preference"))
        lv_t_StartTime1 = GetItemTime(dw_class_info, lv_i_CurrentClass, "start_time")
        IF (lv_s_TimePref = "am") THEN
            IF (NOT (lv_t_StartTime1 <= Time(11, 20, 0))) THEN
                lv_b_Continue = FALSE
            END IF
        ELSEIF (lv_s_TimePref = "pm") THEN
            IF (NOT (lv_t_StartTime1 >= Time(11, 20, 0))) THEN
                lv_b_Continue = FALSE
            END IF
        END IF
    END IF
    IF (lv_b_Continue) THEN
        lv_s_Lab = Trim(GetItemString(dw_class_info, lv_i_CurrentClass, "lab"))
        IF (lv_s_Lab = "y") THEN
            lv_s_SectionNum = Trim(GetItemString(dw_class_info, lv_i_CurrentClass, "section_num"))
        END IF
        lv_s_Days1 = Trim(GetItemString(dw_class_info, lv_i_CurrentClass, "days"))
    END IF
    lv_i_CurrentClass = lv_i_CurrentClass + 1
    lv_b_Continue = FALSE
    IF (lv_i_CurrentCourseNum = lv_i_PastCourseNum) THEN
        lv_s_TimePref = Trim(GetItemString(dw_prof_info, lv_i_CurrentProf, "time_preference"))
        lv_t_StartTime1 = GetItemTime(dw_class_info, lv_i_CurrentClass, "start_time")
    END IF
END DO
lv_t_EndTime1 = GetItemTime(dw_class_info, lv_i_CurrentClass, "end_time")

IF (lv_i_CurrentClass + 1 <= lv_i_TotalNumClass) THEN
lv_i_CourseNum2 = GetItemNumber(dw_class_info, lv_i_CurrentClass + 1, "course_num")
lv_s_Days2 = Trim(GetItemString(dw_class_info, lv_i_CurrentClass + 1, "days "))
lv_t_StartTime2 = GetItemTime(dw_class_info, lv_i_CurrentClass + 1, "start_time")
lv_t_EndTime2 = GetItemTime(dw_class_info, lv_i_CurrentClass + 1, "end_time")

IF ((lv_i_CourseNum2 = lv_i_CurrentCourseNum) AND (lv_s_Days1 = lv_s_Days2) AND (lv_t_StartTime1 = lv_t_StartTime2) AND (lv_t_EndTime1 = lv_t_EndTime2)) THEN
lv_b_Next = TRUE
lv_b_Above = FALSE
lv_i_OtherSection = lv_i_CurrentClass + 1
END IF
ELSEIF (lv_i_CurrentClass - 1 >= 0) THEN
lv_i_CourseNum2 = GetItemNumber(dw_class_info, lv_i_CurrentClass - 1, "course_num")
lv_s_Days2 = Trim(GetItemString(dw_class_info, lv_i_CurrentClass - 1, "days "))
lv_t_StartTime2 = GetItemTime(dw_class_info, lv_i_CurrentClass - 1, "start_time")
lv_t_EndTime2 = GetItemTime(dw_class_info, lv_i_CurrentClass - 1, "end_time")

IF ((lv_i_CourseNum2 = lv_i_CurrentCourseNum) AND (lv_s_Days1 = lv_s_Days2) AND (lv_t_StartTime1 = lv_t_StartTime2) AND (lv_t_EndTime1 = lv_t_EndTime2)) THEN
lv_b_Next = FALSE
lv_b_Above = TRUE
lv_i_OtherSection = lv_i_CurrentClass - 1
END IF
ENDIF
lv_s_LabDays1=Trim(GetItemString(dw_class_info, lv_i_CurrentClass, "lab_days "))
lv_t_LabStartTime1=GetItemTime(dw_class_info, lv_i_CurrentClass, "lab_start_time")
lv_t_LabEndTime1=GetItemTime(dw_class_info, lv_i_CurrentClass, "lab_end_time ")

IF (lv_b_Next OR lv_b_Above) THEN
lv_s_LabDays2=Trim(GetItemString(dw_class_info, lv_i_OtherSection, "lab_days"))
lv_t_LabStartTime2=GetItemTime(dw_class_info, lv_i_OtherSection, "lab_start_time")
lv_t_LabEndTime2=GetItemTime(dw_class_info, lv_i_OtherSection, "lab_end_time")
END IF
ELSE
lv_s_SectionNum=Trim(GetItemString(dw_class_info, lv_i_CurrentClass, "section_num"))

lv_s_Days1 = Trim(GetItemString(dw_class_info, lv_i_CurrentClass, "days"))
lv_t_EndTime1 = GetItemTime(dw_class_info, lv_i_CurrentClass, "end_time")
lv_s_LabDays1 = ""
lv_t_LabStartTime1 = Time(0,0,0)
lv_t_LabEndTime1 = Time(0,0,0)
lv_s_LabDays2 = ""
lv_t_LabStartTime2 = Time(0,0,0)
lv_t_LabEndTime2 = Time(0,0,0)
END IF

lv_i_Priority = GetItemNumber(dw_prof_few, lv_i_CurrentProf, "priority")

lv_b_FoundClass = add_class_lab(lv_ia_ProfClass, lv_i_Priority, lv_s_Days1, lv_t_StartTime1, lv_t_EndTime1, &
lv_s_LabDays1, lv_t_LabStartTime1, lv_t_LabEndTime1, lv_s_LabDays2, lv_t_LabStartTime2, &
lv_t_LabEndTime2)
IF (lv_b_FoundClass) THEN
    lv_i_Loop1 = 0
    lv_b_NotEmpty = TRUE
    DO WHILE (lv_b_NotEmpty)
        lv_i_Loop1 = lv_i_Loop1 + 1
        lv_s_TempSection = Trim(GetItemString(dw_prof_few, lv_i_CurrentProf, "section" + String(lv_i_Loop1)))
        IF (ISNull(lv_s_TempSection) OR (lv_s_TempSection = "")) THEN
            lv_b_NotEmpty = FALSE
        END IF
    END DO
    SETITEM(dw_prof_few, lv_i_CurrentProf, "course" + String(lv_i_Loop1), lv_i_CurrentCourseNum)
    SETITEM(dw_prof_few, lv_i_CurrentProf, "section" + String(lv_i_Loop1), lv_s_SectionNum)
    IF (lv_b_Next OR lv_b_Above) THEN
        lv_s_SectionNum2 = Trim(GetItemString(dw_class_info, lv_i_OtherSection, "section_num"))
        SETITEM(dw_prof_few, lv_i_CurrentProf, "course" + String(lv_i_Loop1 + 1), lv_i_CurrentCourseNum2)
        SETITEM(dw_prof_few, lv_i_CurrentProf, "section" + String(lv_i_Loop1 + 1), lv_s_SectionNum2)
    END IF
    lv_d_TotalCredits = GetItemDecimal(dw_prof_few, lv_i_CurrentProf, "credits")
    IF (lv_s_Lab = "y") THEN
        IF (lv_b_Next OR lv_b_Above) THEN
            lv_d_TotalCredits = lv_d_TotalCredits + 2
        ELSE
            DeleteRow(dw_class_info, lv_i_OtherSection)
        END IF
    ELSE
        lv_d_TotalCredits = lv_d_TotalCredits + 1.5
    END IF
ELSE
    lv_d_TotalCredits = lv_d_TotalCredits + 1
END IF
SetItem(dw_prof_few, lv_i_CurrentProf, "credits", lv_d_TotalCredits)

DeleteRow(dw_class_info, lv_i_CurrentClass)
  lv_b_Done = TRUE
  lv_i_TotalNumClass =RowCount(dw_class_info)
END IF
ELSE
  lv_b_Continue = TRUE
END IF

IF (lv_i_TotalNumClass = 0) THEN
  lv_b_DoneClass = TRUE
  EXIT
END IF

lv_i_CurrentClass = lv_i_CurrentClass - 1
IF (lv_i_CurrentClass > lv_i_TotalNumClass) THEN
  lv_b_Done = TRUE
END IF
LOOP
  LOOP
    NEXT
NEXT

//Move the professors to the appropriate datawindow (done, extra, or stay at few)
lv_i_CurrentRow = 1
FOR lv_i_CurrentProf = 1 TO lv_i_TotalFewProf
  lv_d_TotalCredits = GetItemDecimal(dw_prof_few, lv_i_CurrentRow, "credits")
lv_d_NeedCredits = GetItemDecimal(dw_prof_few, lv_i_CurrentRow, "needed_credits")
  IF (lv_d_TotalCredits > lv_d_NeedCredits) THEN
    lv_s_Name = Trim(GetItemString(dw_prof_few, lv_i_CurrentRow, "name"))
lv_i_Priority = GetItemNumber(dw_prof_few, lv_i_CurrentRow, "priority")
lv_i_CurrentPreference = GetItemNumber(dw_prof_few, lv_i_CurrentRow, "current_preference")
lv_i_CurrentPrevious = GetItemNumber(dw_prof_few, lv_i_CurrentRow, "current_previous")
  lv_i_AddedRow = InsertRow(dw_prof_extra, 0)
SetItem(dw_prof_extra, lv_i_AddedRow, "name", lv_s_Name)
SetItem(dw_prof_extra, lv_i_AddedRow, "priority", lv_i_Priority)
SetItem(dw_prof_extra, lv_i_AddedRow, "needed_credits", lv_d_NeededCredits)
SetItem(dw_prof_extra, lv_i_AddedRow, "current_preference", lv_i_CurrentPreference)
SetItem(dw_prof_extra, lv_i_AddedRow, "current_previous", lv_i_CurrentPrevious)
SetItem(dw_prof_extra, lv_i_AddedRow, "credits", lv_d_TotalCredits)

FOR lv_i_Loop1 = 1 TO 8
lv_i_TempCourse=GetItemNumber(dw_prof_few, lv_i_CurrentRow, "course_" + String(lv_i_Loop1))
lv_s_TempSection=Trim(GetItemString(dw_prof_few, lv_i_CurrentRow, "section_" + String(lv_i_Loop1)))
SetItem(dw_prof_extra, lv_i_AddedRow, "course_" + String(lv_i_Loop1), lv_i_TempCourse)
SetItem(dw_prof_extra, lv_i_AddedRow, "section_" + String(lv_i_Loop1), lv_s_TempSection)
NEXT

DeleteRow(dw_prof_few, lv_i_CurrentRow)
ELSEIF ((lv_d_TotalCredits >= lv_d_NeededCredits - 0.5) AND (lv_d_TotalCredits <= lv_d_NeededCredits)) THEN

lv_s_Name = Trim(GetItemString(dw_prof_few, lv_i_CurrentRow, "name"))

lv_i_AddedRow = InsertRow(dw_prof_okay, 0)

SetItem(dw_prof_okay, lv_i_AddedRow, "name", lv_s_Name)
SetItem(dw_prof_okay, lv_i_AddedRow, "credits", lv_d_TotalCredits)

FOR lv_i_Loop1 = 1 TO 8
lv_i_TempCourse=GetItemNumber(dw_prof_few, lv_i_CurrentRow, "course_" + String(lv_i_Loop1))
lv_s_TempSection=Trim(GetItemString(dw_prof_few, lv_i_CurrentRow, "section_" + String(lv_i_Loop1)))
SetItem(dw_prof_okay, lv_i_AddedRow, "course_" + String(lv_i_Loop1), lv_i_TempCourse)
SetItem(dw_prof_okay, lv_i_AddedRow, "section_" + String(lv_i_Loop1), lv_s_TempSection)
NEXT
Window: w_scheduling
Library: e:\thesis\appl\schedule.pbl
Date: 5/1/95    Time: 20:37:42

    DeleteRow(dw_prof_few, lv_i_CurrentRow)
    ELSE
      lv_i_CurrentRow = lv_i_CurrentRow + 1
    END IF
  NEXT
END IF

End of Script

Script for: phase4  event
//Assign any remaining classes (if any) to any faculty member that has the time slot open and needs a cla
s.
//After each class is assigned, those faculty members are sorted into the faculty members with to many cl
asses,
//faculty members with to few classes, and faculty members with an okay number of classes
//into the appropriate data windows.

//Local Variables
Boolean  lv_b_Continue = TRUE
Boolean  lv_b_Next
Boolean  lv_b_Above
Boolean  lv_b_FoundClass
Boolean  lv_b_NotEmpty = TRUE
Decimal  lv_d_TotalCredits
Decimal  lv_d_NeedCredits
Integer  lv_i_Loop1
Integer  lv_i_TotalFewProf
Integer  lv_i_TotalNumClass
Integer  lv_i_CurrentProf
Integer  lv_i_CurrentProf2
Integer  lv_i_CurrentClass
Integer  lv_i_CurrentPreference
Integer  lv_i_CurrentPrevious
Window: w_scheduling
Library: e:\thesis\appl\schedule.pbl
Date: 5/1/95    Time: 20:37:42

Integer    lv_i_CurrentCourseNum
Integer    lv_i_CurrentRow
Integer    lv_i_AddedRow
Integer    lv_i_OtherSection
Integer    lv_i_Priority
Integer    lv_i_CourseNum2
String     lv_s_TempCourse
String     lv_s_TempSection
String     lv_s_SectionNum
String     lv_s_SectionNum2
String     lv_s_Days1
String     lv_s_Days2
String     lv_s_Lab
String     lv_s_LabDays1
String     lv_s_LabDays2
String     lv_s_Name
String     lv_s_TimePref
Time       lv_t_StartTime1
Time       lv_t_StartTime2
Time       lv_t_EndTime1
Time       lv_t_EndTime2
Time       lv_t_LabStartTime1
Time       lv_t_LabStartTime2
Time       lv_t_LabEndTime1
Time       lv_t_LabEndTime2

//End Local Variables

lv_i_TotalNumClass = RowCount(dw_class_info)
IF (lv_i_TotalNumClass > 0) THEN
  SetSort(dw_prof few, "credits A, priority A")
  Sort (dw_prof few)

lv_i_TotalFewProf = RowCount(dw_prof few)

FOR lv_i_CurrentClass = 1 TO lv_i_TotalNumClass
  FOR lv_i_CurrentProf = 1 TO lv_i_TotalFewProf
lv_i_OtherSection = lv_i_CurrentClass + 1
END IF

ELSEIF (lv_i_CurrentClass - 1 >= 1) THEN
lv_i_CourseNum2 = GetItemNumber(dw_class_info, lv_i_CurrentClass - 1, "course_num")
lv_s_Days2 = Trim.GetItemString(dw_class_info, lv_i_CurrentClass - 1, "days")
lv_t_StartTime2 = GetItemTime(dw_class_info, lv_i_CurrentClass - 1, "start_time")
lv_t_EndTime2 = GetItemTime(dw_class_info, lv_i_CurrentClass - 1, "end_time")

IF ((lv_i_CourseNum2 = lv_i_CurrentCourseNum) AND (lv_s_Days1 = lv_s_Days2) AND (lv_t_StartTime1 = lv_t_StartTime2) AND (lv_t_EndTime1 = lv_t_EndTime2)) THEN
lv_b_Next = FALSE
lv_b_Above = TRUE
lv_i_OtherSection = lv_i_CurrentClass - 1
END IF
END IF

lv_s_LabDays1 = Trim.GetItemString(dw_class_info, lv_i_CurrentClass, "lab_days")
lv_t_LabStartTime1 = GetItemTime(dw_class_info, lv_i_CurrentClass, "lab_start_time")
lv_t_LabEndTime1 = GetItemTime(dw_class_info, lv_i_CurrentClass, "lab_end_time")

IF (lv_b_Next OR lv_b_Above) THEN
lv_s_LabDays2 = Trim.GetItemString(dw_class_info, lv_i_OtherSection, "lab_days")
lv_t_LabStartTime2 = GetItemTime(dw_class_info, lv_i_OtherSection, "lab_start_time")
lv_t_LabEndTime2 = GetItemTime(dw_class_info, lv_i_OtherSection, "lab_end_time")
END IF
ELSE
lv_s_SectionNum = Trim.GetItemString(dw_class_info, lv_i_CurrentClass, "section_num")
lv_s_Days1 = Trim.GetItemString(dw_class_info, lv_i_CurrentClass, "days")
lv_t_EndTime1 = GetItemTime(dw_class_info, lv_i_CurrentClass, "end_time")
lv_s_LabDays1 = ""
lv_t_LabStartTime1 = Time(0, 0, 0)
lv_t_LabEndTime1 = Time(0, 0, 0)
lv_s_LabDays2 = ""
lv_t_LabStartTime2 = Time(0, 0, 0)
lv_t_LabEndTime2 = Time(0, 0, 0)
BEGIN IF

lv_i_Priority = GetItemNumber(dw_prof_few, lv_i_CurrentProf, "priority")

lv_b_FoundClass = add_class_lab(lv_ia_ProfClass, lv_i_Priority, lv_s_Days1, lv_t_StartTime1, 1
lv_t_EndTime1, &
    lv_s_LabDays1, lv_t_LabStartTime1, lv_t_LabEndTime1, lv_s_LabDays2, lv_t_LabStartTime2, &
    lv_t_LabEndTime2)

IF (lv_b_FoundClass) THEN
    lv_i_Loop1 = 0
    lv_b_NotEmpty = TRUE
    DO WHILE (lv_b_NotEmpty)
        lv_i_Loop1 = lv_i_Loop1 + 1
        lv_s_TempSection=Trim(GetItemString(dw_prof_few, lv_i_CurrentProf, "section_" + String(lv_i_Loop1)))
        IF (ISNull(lv_s_TempSection) OR (lv_s_TempSection = "")) THEN
            lv_b_NotEmpty = FALSE
        END IF
    LOOP

SetItem(dw_prof_few, lv_i_CurrentProf, "course_" + String(lv_i_Loop1), lv_i_CurrentCourseNum)

SetItem(dw_prof_few, lv_i_CurrentProf, "section_" + String(lv_i_Loop1), lv_s_SectionNum)

IF (lv_b_Next OR lv_b_Above) THEN
    lv_s_SectionNum2=Trim(GetItemString(dw_class_info, lv_i_OtherSection, "section_num "))
SetItem(dw_prof_few, lv_i_CurrentProf, "course_" + String(lv_i_Loop1 + 1), lv_i_CourseNum)
SetItem(dw_prof_few, lv_i_CurrentProf, "section_" + String(lv_i_Loop1 + 1), lv_s_Section Num2)

END IF

lv_d_TotalCredits = GetItemDecimal(dw_prof_few, lv_i_CurrentProf, "credits")
IF (lv_s_Lab = "y") THEN
    IF (lv_b_Next OR lv_b_Above) THEN
lv_d_TotalCredits = lv_d_TotalCredits + 2
DeleteRow(dw_class_info, lv_i_OtherSection)
ELSE
lv_d_TotalCredits = lv_d_TotalCredits + 1.5
END IF
ELSE
lv_d_TotalCredits = lv_d_TotalCredits + 1
END IF
setItem(dw_prof_few, lv_i_CurrentProf, "credits", lv_d_TotalCredits)
DeleteRow(dw_class_info, lv_i_CurrentClass)
lv_i_TotalNumClass = RowCount(dw_class_info)
END IF
ELSE
lv_b_Continue = TRUE
END IF

//Move the professors to the appropriate datawindow (done, extra, or stay at few)
lv_i_CurrentRow = 1
FOR lv_i_CurrentProf2 = 1 TO lv_i_TotalFewProf
lv_d_TotalCredits = getItemDecimal(dw_prof_few, lv_i_CurrentRow, "credits")
lv_d_NeedCredits = getItemDecimal(dw_prof_few, lv_i_CurrentRow, "needed_credits")
IF (lv_d_TotalCredits > lv_d_NeedCredits) THEN
lv_s_Name = Trim(getItemString(dw_prof_few, lv_i_CurrentRow, "name"))
lv_i_Priority = getItemNumber(dw_prof_few, lv_i_CurrentRow, "priority")
lv_i_CurrentPreference = getItemNumber(dw_prof_few, lv_i_CurrentRow, "current_preference")
lv_i_CurrentPrevious = getItemNumber(dw_prof_few, lv_i_CurrentRow, "current_previous")

lv_i_AddedRow = insertRow(dw_prof_extra, 0)
setItem(dw_prof_extra, lv_i_AddedRow, "name", lv_s_Name)
setItem(dw_prof_extra, lv_i_AddedRow, "priority", lv_i_Priority)
setItem(dw_prof_extra, lv_i_AddedRow, "needed_credits", lv_d_NeedCredits)
Window: w_scheduling
Library: e:\thesis\app\schedule.pbl
Date: 5/1/95  Time: 20:37:42

SetItem(dw_prof_extra, lv_i_AddedRow, "current_preference", lv_i_CurrentPreference)
SetItem(dw_prof_extra, lv_i_AddedRow, "current_previous", lv_i_CurrentPrevious)
SetItem(dw_prof_extra, lv_i_AddedRow, "credits", lv_d_TotalCredits)

FOR lv_i_Loop1 = 1 TO 8
lv_i_TempCourse=getItemNumber(dw_prof_few, lv_i_CurrentRow, "course_" + String(lv_i_Loop1))
lv_s_TempSection=Trim(GetItemString(dw_prof_few, lv_i_CurrentRow, "section_" + String(lv_i_Loop1)))
SetItem(dw_prof_extra, lv_i_AddedRow, "course_" + String(lv_i_Loop1), lv_i_TempCourse)
SetItem(dw_prof_extra, lv_i_AddedRow, "section_" + String(lv_i_Loop1), lv_s_TempSection)
NEXT

DeleteRow(dw_prof_few, lv_i_CurrentRow)
ELSEIF ((lv_d_TotalCredits >= lv_d_NeedCredits - 0.5) AND (lv_d_TotalCredits <= lv_d_NeedCredits)) THEN
lv_s_Name = Trim(GetItemString(dw_prof_few, lv_i_CurrentRow, "name"))
LV_i_AddedRow = InsertRow(dw_prof_okay, 0)
SetItem(dw_prof_okay, lv_i_AddedRow, "name", lv_s_Name)
SetItem(dw_prof_okay, lv_i_AddedRow, "credits", lv_d_TotalCredits)

FOR lv_i_Loop1 = 1 TO 8
lv_i_TempCourse=getItemNumber(dw_prof_few, lv_i_CurrentRow, "course_" + String(lv_i_Loop1))
lv_s_TempSection=Trim(GetItemString(dw_prof_few, lv_i_CurrentRow, "section_" + String(lv_i_Loop1)))
SetItem(dw_prof_okay, lv_i_AddedRow, "course_" + String(lv_i_Loop1), lv_i_TempCourse)
SetItem(dw_prof_okay, lv_i_AddedRow, "section_" + String(lv_i_Loop1), lv_s_TempSection)
NEXT

DeleteRow(dw_prof_few, lv_i_CurrentRow)
ELSE
lv_i_CurrentRow = lv_i_CurrentRow + 1
Window: w_scheduling
Library: e:\thesis\appl\schedule.pbl
Date: 5/1/95    Time: 20:37:42

END IF
NEXT
lv_i_TotalFewProf = RowCount(dw_prof_few)
IF ((DeletedCount(dw_class_info) > 0) AND (lv_i_CurrentClass > lv_i_TotalNumClass)) THEN
  EXIT
END IF
NEXT
NEXT
END IF

End of Script

Script for: phase5  event
//Moves all unassigned classes (if any) to a separate datawindow

//Local Variables
Integer  lv_i_TotalNumClass
Integer  lv_i_Loop
Integer  lv_i_CourseNum
String   lv_s_SectionNum
//End Local Variables

lv_i_TotalNumClass = RowCount(dw_class_info)

IF (lv_i_TotalNumClass > 0) THEN
  FOR lv_i_Loop = 1 TO lv_i_TotalNumClass
    lv_i_CourseNum = GetItemNumber(dw_class_info, lv_i_Loop, "course_num")
    lv_s_SectionNum = Trim(GetItemString(dw_class_info, lv_i_Loop, "section_num"))
    InsertRow(dw_extra_class, 0)
    SetItem(dw_extra_class, lv_i_Loop, "course_num", lv_i_CourseNum)
    SetItem(dw_extra_class, lv_i_Loop, "section_num", lv_s_SectionNum)
  NEXT
Window: w_scheduling
Library: e:\thesis\appl\schedule.pbl
Date: 5/1/95    Time: 20:37:42

END IF

End of Script

Script for: phases  event
//Let faculty members with the least amount of credits choose from the last acquired courses
//of the professors with extra credits (based on preference). After each transfer is made, those
//faculty members are sorted into the faculty members with to many classes, faculty members with
//to few classes, and faculty members with an okay number of classes into the appropriate datawindows.

//Local Variables
Boolean lv_b_DeleteClass
Boolean lv_b.Done
Boolean lv_b.Continue
Boolean lv_b_Found
Boolean lv_b.Next
Boolean lv_b_Above
Boolean lv_b.Exit
Boolean lv_b_FoundClass
Boolean lv_b_NotEmpty
Decimal lv_d_NeedCredits
Decimal lv_d_TotalCredits
Decimal lv_dExtraTotalCredits
Integer lv_i_TotalNumClass
Integer lv_i_TotalFewProf
Integer lv_i_TotalExtraProf
Integer lv_i_AddedRow
Integer lv_i_ClassRow
Integer lv_i_CurrentRow
Integer lv_i_CurrentFewProf
Integer lv_i_CurrentExtraProf
Integer lv_i_CurrentProf
Integer lv_i_CurrentPreference
Integer lv_i_CurrentPrevious
Window: w_scheduling
Library: e:\thesis\appl\schedule.pbl
Date: 5/1/95    Time: 20:37:42

Integer lv_i_Class
Integer lv_i_CurrentClass
Integer lv_i_CurrentCourseNum
Integer lv_i_CourseNum2
Integer lv_i_TempCourse
Integer lv_i_PrefCourseNum
Integer lv_i_OtherSection
Integer lv_i_Priority
Integer lv_i_MaxClass = 0
Integer lv_i_Loop
Integer lv_i_Loop1
Integer lv_i_Loop2
Integer lv_i_Loop3
String lv_s_Name
String lv_s_TimePref
String lv_s_SectionNum
String lv_s_SectionNum2
String lv_s_TempSection
String lv_s_Lab
String lv_s_Days1
String lv_s_Days2
String lv_s_LabDays1
String lv_s_LabDays2
Time lv_t_StartTime1
Time lv_t_StartTime2
Time lv_t_EndTime1
Time lv_t_EndTime2
Time lv_t_LabStartTime1
Time lv_t_LabStartTime2
Time lv_t_LabEndTime1
Time lv_t_LabEndTime2
//End Local Variables

Retrieve(dw_class_info, s_parms.term, s_parms.department)

SetSort(dw_class_info, "course_num A, section_num A")
Sort(dw_class_info)

lv_i_TotalFewProf = RowCount(dw_prof_few)
lv_i_TotalExtraProf = RowCount(dw_prof_extra)

FOR lv_i_CurrentfewProf = 1 TO lv_i_TotalFewProf
    SetItem(dw_prof_few, lv_i_CurrentfewProf, "current_preference", 0)
    SetItem(dw_prof_few, lv_i_CurrentfewProf, "current_previous", 0)
NEXT

FOR lv_i_Loop3 = 1 TO lv_i_TotalExtraProf
    FOR lv_i_Loop = 1 TO lv_i_TotalExtraProf
        lv_i_Loop1 = 0
        lv_b_NotEmpty = TRUE
        DO WHILE (lv_b_NotEmpty)
            lv_i_Loop1 = lv_i_Loop1 + 1
            lv_s_TempSection = Trim(GetItemString(dw_prof_extra, lv_i_Loop, "section_" + String(lv_i_Loop 1)));
        IF ((ISNull(lv_s_TempSection) OR (lv_s_TempSection = "")) AND (lv_i_Loop1 - 1 > lv_i_MaxClass)) T
        THEN
            lv_i_MaxClass = lv_i_Loop1 - 1
            lv_b_NotEmpty = FALSE
        EXIT
        END IF
        IF (lv_i_Loop1 >= 8) THEN
            lv_b_NotEmpty = FALSE
        EXIT
        END IF
        LOOP
    NEXT
END IF

SetSort(dw_prof_extra, "credits D, priority A")
Sort(dw_prof_extra)

SetSort(dw_prof_few, "credits A, priority A")
Sort(dw_prof_few)
Window: w_scheduling  
Library: e:\thesis\appl\schedule.pbl  
Date: 5/1/95      Time: 20:37:42  

Integer  lv_i_CurrentCourseNum  
Integer  lv_i_CurrentRow  
Integer  lv_i_AddedRow  
Integer  lv_i_OtherSection  
Integer  lv_i_Priority  
Integer  lv_i_TempCourse  
Integer  lv_i_CourseNum2  
String   lv_s_TempSection  
String   lv_s_SectionNum  
String   lv_s_SectionNum2  
String   lv_s_Days1  
String   lv_s_Days2  
String   lv_s_Lab  
String   lv_s_LabDays1  
String   lv_s_LabDays2  
String   lv_s_Name  
String   lv_s_TimePref  
Time    lv_t_StartTime1  
Time    lv_t_StartTime2  
Time    lv_t_EndTime1  
Time    lv_t_EndTime2  
Time    lv_t_LabStartTime1  
Time    lv_t_LabStartTime2  
Time    lv_t_LabEndTime1  
Time    lv_t_LabEndTime2  

//End Local Variables  

lv_i_TotalNumClass  =  RowCount(dw_class_info)  
IF (lv_i_TotalNumClass > 0) THEN  
    SetSort(dw_prof_few, "credits A, priority A")  
    Sort(dw_prof_few)  

lv_i_TotalFewProf  =  RowCount(dw_prof_few)  

FOR lv_i_CurrentClass = 1 TO lv_i_TotalNumClass  
    FOR lv_i_CurrentProf = 1 TO lv_i_TotalFewProf
lv_b_Continue = TRUE
lv_b_Next = FALSE
lv_b_Above = FALSE
lv_i_CurrentCourseNum = GetItemNumber(dw_class_info, lv_i_CurrentClass, "course_num")

lv_s_TimePref = Trim(GetItemString(dw_prof_info, lv_i_CurrentProf, "time_preference"))
lv_t_StartTime1 = GetItemTime(dw_class_info, lv_i_CurrentClass, "start_time")
IF (lv_s_TimePref = "am") THEN
    IF (NOT (lv_t_StartTime1 <= Time(11,20,0))) THEN
        lv_b_Continue = FALSE
    END IF
ELSEIF (lv_s_TimePref = "pm") THEN
    IF (NOT (lv_t_StartTime1 >= Time(11,20,0))) THEN
        lv_b_Continue = FALSE
    END IF
END IF

IF (lv_b_Continue) THEN
    lv_s_Lab = Trim(GetItemString(dw_class_info, lv_i_CurrentClass, "lab"))
    IF (lv_s_Lab = "y") THEN
        lv_s_SectionNum = Trim(GetItemString(dw_class_info, lv_i_CurrentClass, "section_num"))
    END IF
    lv_s_Days1 = Trim(GetItemString(dw_class_info, lv_i_CurrentClass, "days"))
lv_t_EndTime1 = GetItemTime(dw_class_info, lv_i_CurrentClass, "end_time")
    IF (lv_i_CurrentClass + 1 <= lv_i_TotalNumClass) THEN
        lv_i_CourseNum2 = GetItemNumber(dw_class_info, lv_i_CurrentClass + 1, "course_num")
lv_s_Days2 = Trim(GetItemString(dw_class_info, lv_i_CurrentClass + 1, "days"))
lv_t_StartTime2 = GetItemTime(dw_class_info, lv_i_CurrentClass + 1, "start_time")
lv_t_EndTime2 = GetItemTime(dw_class_info, lv_i_CurrentClass + 1, "end_time")
    END IF
    IF ((lv_i_CourseNum2 = lv_i_CurrentCourseNum) AND (lv_s_Days1 = lv_s_Days2) AND (lv_t_StartTime1 = lv_t_StartTime2) AND (lv_t_EndTime1 = lv_t_EndTime2)) THEN
        lv_b_Next = TRUE
        lv_b_Above = FALSE
lv_i_OtherSection = lv_i_CurrentClass + 1
END IF
ELSEIF (lv_i_CurrentClass - 1 >= 1) THEN
  lv_i_CourseNum2 = GetItemNumber(dw_class_info, lv_i_CurrentClass - 1, "course_num")
  lv_s_Days2 = Trim(GetItemString(dw_class_info, lv_i_CurrentClass - 1, "days"))
  lv_t_StartTime2 = GetItemTime(dw_class_info, lv_i_CurrentClass - 1, "start_time")
  lv_t_EndTime2 = GetItemTime(dw_class_info, lv_i_CurrentClass - 1, "end_time")
ENDIF
IF ((lv_i_CourseNum2 = lv_i_CurrentCourseNum) AND (lv_s_Days1 = lv_s_Days2) AND
    (lv_t_StartTimel = lv_t_StartTime2) AND (lv_t_EndTimel = lv_t_EndTime2)) THEN
  lv_b_Next = FALSE
  lv_b_Above = TRUE
  lv_i_OtherSection = lv_i_CurrentClass - 1
ENDIF
END IF
lv_s_LabDays1 = Trim(GetItemString(dw_class_info, lv_i_CurrentClass, "lab_days"))
lv_t_LabStartTime1 = GetItemTime(dw_class_info, lv_i_CurrentClass, "lab_start_time")
lv_t_LabEndTime1 = GetItemTime(dw_class_info, lv_i_CurrentClass, "lab_end_time")
IF (lv_b_Next OR lv_b_Above) THEN
  lv_s_LabDays2 = Trim(GetItemString(dw_class_info, lv_i_OtherSection, "lab_days"))
  lv_t_LabStartTime2 = GetItemTime(dw_class_info, lv_i_OtherSection, "lab_start_time")
  lv_t_LabEndTime2 = GetItemTime(dw_class_info, lv_i_OtherSection, "lab_end_time")
ENDIF
ELSE
  lv_s_SectionNum = Trim(GetItemString(dw_class_info, lv_i_CurrentClass, "section_num"))
  lv_s_Days1 = Trim(GetItemString(dw_class_info, lv_i_CurrentClass, "days"))
  lv_t_EndTime1 = GetItemTime(dw_class_info, lv_i_CurrentClass, "end_time")
  lv_s_LabDays1 = ""
  lv_t_LabStartTime1 = Time(0, 0, 0)
  lv_t_LabEndTime1 = Time(0, 0, 0)
  lv_s_LabDays2 = ""
  lv_t_LabStartTime2 = Time(0, 0, 0)
  lv_t_LabEndTime2 = Time(0, 0, 0)
Window: w_scheduling
Library: e:\thesis\appl\schedule.pbl
Date: 5/1/95    Time: 20:37:42

END IF

lv_i_Priority = GetItemNumber(dw_prof_few, lv_i_CurrentProf, "priority")

lv_b_FoundClass = add_class_lab(lv_ia_ProfClass, lv_i_Priority, lv_s_Days1, lv_t_StartTime1, lv_t_EndTime1, &

    lv_s_LabDays1, lv_t_LabStartTime1, lv_t_LabEndTime1, lv_s_LabDays2, lv_t_LabStartTime2, &

    lv_t_LabEndTime2)

    IF (lv_b_FoundClass) THEN
    lv_i_Loop1 = 0
    lv_b_NotEmpty = TRUE
    DO WHILE (lv_b_NotEmpty)
    lv_i_Loop1 = lv_i_Loop1 + 1
    lv_s_TempSection=Trim(GetItemString(dw_prof_few, lv_i_CurrentProf, "section_" + String(lv_i_Loop1)))

    IF (ISNull(lv_s_TempSection) OR (lv_s_TempSection = "")) THEN
    lv_b_NotEmpty = FALSE
    END IF
    LOOP

SetItem(dw_prof_few, lv_i_CurrentProf, "course_" + String(lv_i_Loop1), lv_i_CurrentCourseNum)

SetItem(dw_prof_few, lv_i_CurrentProf, "section_" + String(lv_i_Loop1), lv_s_SectionNum)

    IF (lv_b_Next OR lv_b_Above) THEN
    lv_s_SectionNum2=Trim(GetItemString(dw_class_info, lv_i_OtherSection, "section_num "))
    SetItem(dw_prof_few, lv_i_CurrentProf, "course_" + String(lv_i_Loop1 + 1), lv_i_CourseNum)
    SetItem(dw_prof_few, lv_i_CurrentProf, "section_" + String(lv_i_Loop1 + 1), lv_s_SectionNum2)
    END IF

lv_d_TotalCredits = GetItemDecimal(dw_prof_few, lv_i_CurrentProf, "credits")
IF (lv_s_Lab = "y") THEN
    IF (lv_b_Next OR lv_b_Above) THEN
lv_d_TotalCredits = lv_d_TotalCredits + 2
DeleteRow(dw_class_info, lv_i_OtherSection)
ELSE
lv_d_TotalCredits = lv_d_TotalCredits + 1.5
END IF
ELSE
lv_d_TotalCredits = lv_d_TotalCredits + 1
END IF
SetItem(dw_prof_few, lv_i_CurrentProf, "credits", lv_d_TotalCredits)
DeleteRow(dw_class_info, lv_i_CurrentClass)
lv_i_TotalNumClass = RowCount(dw_class_info)
END IF
ELSE
lv_b_Continue = TRUE
END IF

//Move the professors to the appropriate datawindow (done, extra, or stay at few)
lv_i_CurrentRow = 1
FOR lv_i_CurrentProf2 = 1 TO lv_i_TotalFewProf
lv_d_TotalCredits = GetItemDecimal(dw_prof_few, lv_i_CurrentRow, "credits")
lv_d_NeedCredits = GetItemDecimal(dw_prof_few, lv_i_CurrentRow, "needed_credits")
IF (lv_d_TotalCredits > lv_d_NeedCredits) THEN
lv_s_Name = Trim(GetItemString(dw_prof_few, lv_i_CurrentRow, "name"))
lv_i_Priority = GetItemNumber(dw_prof_few, lv_i_CurrentRow, "priority")
lv_i_CurrentPreference=GetItemNumber(dw_prof_few, lv_i_CurrentRow, "current_preference")
lv_i_CurrentPrevious=GetItemNumber(dw_prof_few, lv_i_CurrentRow, "current_previous")
lv_i_AddedRow = InsertRow(dw_prof_extra, 0)
SetItem(dw_prof_extra, lv_i_AddedRow, "name", lv_s_Name)
SetItem(dw_prof_extra, lv_i_AddedRow, "priority", lv_i_Priority)
SetItem(dw_prof_extra, lv_i_AddedRow, "needed_credits", lv_d_NeedCredits)
Window: w_scheduling
Library: e:\thesis\appl\schedule.pbl
Date: 5/1/95      Time: 20:37:42

SetItem(dw_prof_extra, lv_i_AddedRow, "current_preference", lv_i_CurrentPreference)
SetItem(dw_prof_extra, lv_i_AddedRow, "current_previous", lv_i_CurrentPrevious)
SetItem(dw_prof_extra, lv_i_AddedRow, "credits", lv_d_TotalCredits)

FOR lv_i_Loop1 = 1 TO 8
lv_i_TempCourse=GetItemNumber(dw_prof_few, lv_i_CurrentRow, "course_" + String(lv_i_Loop1))
lv_s_TempSection=Trim(GetItemString(dw_prof_few, lv_i_CurrentRow, "section_" + String(lv_i_Loop1)))
SetItem(dw_prof_extra, lv_i_AddedRow, "course_" + String(lv_i_Loop1), lv_i_TempCourse)
SetItem(dw_prof_extra, lv_i_AddedRow, "section_" + String(lv_i_Loop1), lv_s_TempSection)
NEXT

DeleteRow(dw_prof_few, lv_i_CurrentRow)
ELSEIF ((lv_d_TotalCredits >= lv_d_NeedCredits - 0.5) AND (lv_d_TotalCredits <= lv_d_NeedCredits)) THEN
lv_s_Name = Trim(GetItemString(dw_prof_few, lv_i_CurrentRow, "name"))
lv_i_AddedRow = InsertRow(dw_prof_okay, 0)

SetItem(dw_prof_okay, lv_i_AddedRow, "name", lv_s_Name)
SetItem(dw_prof_okay, lv_i_AddedRow, "credits", lv_d_TotalCredits)

FOR lv_i_Loop1 = 1 TO 8
lv_i_TempCourse=GetItemNumber(dw_prof_few, lv_i_CurrentRow, "course_" + String(lv_i_Loop1))
lv_s_TempSection=Trim(GetItemString(dw_prof_few, lv_i_CurrentRow, "section_" + String(lv_i_Loop1)))
SetItem(dw_prof_okay, lv_i_AddedRow, "course_" + String(lv_i_Loop1), lv_i_TempCourse)
SetItem(dw_prof_okay, lv_i_AddedRow, "section_" + String(lv_i_Loop1), lv_s_TempSection)
NEXT

DeleteRow(dw_prof_few, lv_i_CurrentRow)
ELSE
lv_i_CurrentRow = lv_i_CurrentRow + 1
Window: w_scheduling
Library: e:\thesis\appl\schedule.pbl
Date: 5/1/95   Time: 20:37:42

END IF
NEXT
lv_i_TotalFewProf = RowCount(dw_prof_few)
IF((DeletedCount(dw_class_info) > 0) AND (lv_i_CurrentClass > lv_i_TotalNumClass)) THEN
  EXIT
END IF
NEXT
NEXT
END IF

End of Script

Script for: phase5  event
//Moves all unassigned classes (if any) to a separate datawindow

//Local Variables
Integer   lv_i_TotalNumClass
Integer   lv_i_Loop
Integer   lv_i_CourseNum
String    lv_s_SectionNum
//End Local Variables

lv_i_TotalNumClass = RowCount(dw_class_info)

IF (lv_i_TotalNumClass > 0) THEN
  FOR lv_i_Loop = 1 TO lv_i_TotalNumClass
    lv_i_CourseNum = GetItemNumber(dw_class_info, lv_i_Loop, "course_num")
    lv_s_SectionNum = Trim(GetItemString(dw_class_info, lv_i_Loop, "section_num"))
    InsertRow(dw_extra_class, 0)
    SetItem(dw_extra_class, lv_i_Loop, "course_num", lv_i_CourseNum)
    SetItem(dw_extra_class, lv_i_Loop, "section_num", lv_s_SectionNum)
  NEXT
Script for: phase6 event
//Let faculty members with the least amount of credits choose from the last acquired courses
//of the professors with extra credits (based on preference). After each transfer is made, those
//faculty members are sorted into the faculty members with too many classes, faculty members with
//too few classes, and faculty members with an okay number of classes into the appropriate data windows.

//Local Variables
Boolean lv_b_DeleteClass
Boolean lv_b_Done
Boolean lv_b_Continue
Boolean lv_b_Found
Boolean lv_b_Next
Boolean lv_b_Above
Boolean lv_b_Exit
Boolean lv_b_FoundClass
Boolean lv_b_NotEmpty
Decimal lv_d_NeedCredits
Decimal lv_d_TotalCredits
Decimal lv_d_ExtraTotalCredits
Integer lv_i_TotalNumClass
Integer lv_i_TotalFewProf
Integer lv_i_TotalExtraProf
Integer lv_i_AddedRow
Integer lv_i_ClassRow
Integer lv_i_CurrentRow
Integer lv_i_CurrentFewProf
Integer lv_i_CurrentExtraProf
Integer lv_i_CurrentProf
Integer lv_i_CurrentPreference
Integer lv_i_CurrentPrevious
Window: w_scheduling
Library: e:\thesis\appl\schedule.pbl
Date: 5/1/95    Time: 20:37:42

Integer  lv_i_Class
Integer  lv_i_CurrentClass
Integer  lv_i_CurrentCourseNum
Integer  lv_i_CourseNum2
Integer  lv_i_TempCourse
Integer  lv_i_PrefCourseNum
Integer  lv_i_OtherSection
Integer  lv_i_Priority
Integer  lv_i_MaxClass = 0
Integer  lv_i_Loop
Integer  lv_i_Loop1
Integer  lv_i_Loop2
Integer  lv_i_Loop3
String   lv_s_Name
String   lv_s_TimePref
String   lv_s_SectionNum
String   lv_s_SectionNum2
String   lv_s_TempSection
String   lv_s_Lab
String   lv_s_Days1
String   lv_s_Days2
String   lv_s_LabDays1
String   lv_s_LabDays2
Time     lv_t_StartTime1
Time     lv_t_StartTime1
Time     lv_t_EndTime1
Time     lv_t_EndTime2
Time     lv_t_LabStartTime1
Time     lv_t_LabStartTime2
Time     lv_t_LabEndTime1
Time     lv_t_LabEndTime2

//End Local Variables

Retrieve(dw_class_info, s_parms.term, s_parms.department)

SetSort(dw_class_info, "course_num A, section_num A")
Sort (dw_class_info)

1v_i_TotalFewProf = RowCount (dw_prof_few)
1v_i_TotalExtraProf = RowCount (dw_prof_extra)

FOR 1v_i_CurrentFewProf = 1 TO 1v_i_TotalFewProf
    SetItem (dw_prof_few, 1v_i_CurrentFewProf, "current_preference", 0)
    SetItem (dw_prof_few, 1v_i_CurrentFewProf, "current_previous", 0)
NEXT

FOR 1v_i_Loop3 = 1 TO 1v_i_TotalExtraProf
    FOR 1v_i_Loop = 1 TO 1v_i_TotalExtraProf
        1v_i_Loop1 = 0
        lv_b_NotEmpty = TRUE
        DO WHILE (lv_b_NotEmpty)
            1v_i_Loop1 = 1v_i_Loop1 + 1
            lv_s_TempSection = Trim (GetItemString (dw_prof_extra, 1v_i_Loop, "section_" + String (1v_i_Loop1)))
        IF ((IsNull (lv_s_TempSection) OR (lv_s_TempSection = "")) AND (1v_i_Loop1 - 1 > 1v_i_MaxClass)) THEN
            1v_i_MaxClass = 1v_i_Loop1 - 1
            lv_b_NotEmpty = FALSE
            EXIT
        END IF
        IF (1v_i_Loop1 >= 8) THEN
            lv_b_NotEmpty = FALSE
            EXIT
        END IF
    LOOP
NEXT

SetSort (dw_prof_extra, "credits D, priority A")
Sort (dw_prof_extra)

SetSort (dw_prof_few, "credits A, priority A")
Sort (dw_prof_few)
FOR lv_i_CurrentFewProf = lv_i_TotalFewProf TO 1
    lv_b_Exit = FALSE
    lv_b_FoundClass = FALSE
    DO WHILE (NOT lv_b_FoundClass)
        lv_i_CurrentPreference = GetItemNumber(dw_prof_few, lv_i_CurrentFewProf, "current_preference")
        lv_i_CurrentPreference = lv_i_CurrentPreference + 1
        SetItem(dw_prof_few, lv_i_CurrentFewProf, "current_preference", lv_i_CurrentPreference)
        IF (lv_i_CurrentPreference > 4) THEN
            // No more preferences -- assign no more classes until phase 3
            EXIT
        ELSEIF (lv_i_CurrentPreference = 1) THEN
            lv_i_PrefCourseNum = GetItemNumber(dw_prof_info, lv_i_CurrentFewProf, "class_preference_1")
        ELSEIF (lv_i_CurrentPreference = 2) THEN
            lv_i_PrefCourseNum = GetItemNumber(dw_prof_info, lv_i_CurrentFewProf, "class_preference_2")
        ELSEIF (lv_i_CurrentPreference = 3) THEN
            lv_i_PrefCourseNum = GetItemNumber(dw_prof_info, lv_i_CurrentFewProf, "class_preference_3")
        ELSEIF (lv_i_CurrentPreference = 4) THEN
            lv_i_PrefCourseNum = GetItemNumber(dw_prof_info, lv_i_CurrentFewProf, "class_preference_4")
        END IF

        lv_i_CurrentClass = lv_i_MaxClass
        lv_i_CurrentExtraProf = 1
        lv_b_Done = FALSE
        DO WHILE (NOT lv_b_Done)
            lv_b_Continue = TRUE
            lv_b_Next = FALSE
            lv_b_Above = FALSE
            lv_i_CurrentCourseNum = GetItemNumber(dw_prof_extra, lv_i_CurrentExtraProf, "course" + String(lv_i_CurrentClass))
lv_s_SectionNum = Trim(GetItemString(dw_prof_extra, lv_i_CurrentExtraProf, "section_" + String(lv_i_CurrentClass)))
IF (lv_i_CurrentCourseNum = lv_i_PrefCourseNum) THEN
lv_s_TimePref = Trim(GetItemString(dw_prof_info, lv_i_CurrentPrefProf, "time_preference"))
ENDIF

lv_b_Found = FALSE
lv_i_ClassRow = 1
DO WHILE (NOT lv_b_Found)
    lv_i_TempCourse = GetItemNumber(dw_class_info, lv_i_ClassRow, "course_num")
    lv_s_TempSection = Trim(GetItemString(dw_class_info, lv_i_ClassRow, "section_num"))
ENDIF

IF ((lv_i_TempCourse = lv_i_CurrentCourseNum) AND (lv_s_TempSection = lv_s_SectionNum)) THEN
    lv_b_Found = TRUE
    EXIT
ENDIF

lv_i_ClassRow = lv_i_ClassRow + 1
IF (lv_i_Class > lv_i_TotalNumClass) THEN
    EXIT
ENDIF
LOOP

IF (lv_b_Found) THEN
    lv_t_StartTime1 = GetItemTime(dw_class_info, lv_i_ClassRow, "start_time")
ENDIF

IF (lv_s_TimePref = "am") THEN
    IF (NOT (lv_t_StartTime1 <= Time(11,20,0))) THEN
        lv_b_Continue = FALSE
    END IF
ELSEIF (lv_s_TimePref = "pm") THEN
    IF (NOT (lv_t_StartTime1 >= Time(11,20,0))) THEN
        lv_b_Continue = FALSE
    END IF
ENDIF
IF (lv_b_Continue) THEN
  lv_s_Lab = Trim(GetItemString(dw_class_info, lv_i_ClassRow, "lab"))
  IF (lv_s_Lab = "y") THEN
    lv_s_Days1 = Trim(GetItemString(dw_class_info, lv_i_ClassRow, "days"))
    lv_t_EndTime1 = GetItemTime(dw_class_info, lv_i_ClassRow, "end_time")
  END IF
  IF (lv_i_CurrentClass + 1 <= lv_i_TotalNumClass) THEN
    lv_i_CourseNum2=GetItemNumber(dw_class_info, lv_i_ClassRow + 1, "course_num")
    lv_s_Days2 = Trim(GetItemString(dw_class_info, lv_i_ClassRow + 1, "days"))
    lv_t_StartTime2=GetItemTime(dw_class_info, lv_i_ClassRow + 1, "start_time")
    lv_t_EndTime2=GetItemTime(dw_class_info, lv_i_ClassRow + 1, "end_time")
  END IF
  IF ((lv_i_CourseNum2 = lv_i_CurrentCourseNum) AND (lv_s_Days1 = lv_s_Days2) AND
      (lv_t_StartTime1 = lv_t_StartTime2) AND (lv_t_EndTime1 = lv_t_EndTime2)) THEN
    lv_b_Next = TRUE
    lv_b_Above = FALSE
    lv_i_OtherSection = lv_i_ClassRow + 1
  END IF
ELSEIF (lv_i_ClassRow - 1 >= 1) THEN
  lv_i_CourseNum2=GetItemNumber(dw_class_info, lv_i_ClassRow - 1, "course_num")
  lv_s_Days2 = Trim(GetItemString(dw_class_info, lv_i_ClassRow - 1, "days"))
  lv_t_StartTime2=GetItemTime(dw_class_info, lv_i_ClassRow - 1, "start_time")
  lv_t_EndTime2=GetItemTime(dw_class_info, lv_i_ClassRow - 1, "end_time")
  IF ((lv_i_CourseNum2 = lv_i_CurrentCourseNum) AND (lv_s_Days1 = lv_s_Days2) AND
      (lv_t_StartTime1 = lv_t_StartTime2) AND (lv_t_EndTime1 = lv_t_EndTime2)) THEN
    lv_b_Next = FALSE
    lv_b_Above = TRUE
lv_i_OtherSection = lv_i_ClassRow - 1
END IF
END IF

lv_s_LabDays1 = Trim(GetItemString(dw_class_info, lv_i_ClassRow, "lab_days"))
lv_t_LabStartTime1 = GetItemTime(dw_class_info, lv_i_ClassRow, "lab_start_time")
lv_t_LabEndTime1 = GetItemTime(dw_class_info, lv_i_ClassRow, "lab_end_time")

IF (lv_b_Next OR lv_b_Above) THEN
lv_s_LabDays2 = Trim(GetItemString(dw_class_info, lv_i_OtherSection, "lab_days"))
lv_t_LabStartTime2 = GetItemTime(dw_class_info, lv_i_OtherSection, "lab_start_time")
lv_t_LabEndTime2 = GetItemTime(dw_class_info, lv_i_OtherSection, "lab_end_time")
END IF
ELSE
lv_s_Days1 = Trim(GetItemString(dw_class_info, lv_i_ClassRow, "days"))
lv_t_EndTime1 = GetItemTime(dw_class_info, lv_i_ClassRow, "end_time")
lv_s_LabDays1 = ""
lv_t_LabStartTime1 = Time(0,0,0)
lv_t_LabEndTime1 = Time(0,0,0)
lv_s_LabDays2 = ""
lv_t_LabStartTime2 = Time(0,0,0)
lv_t_LabEndTime2 = Time(0,0,0)
END IF

lv_i_Priority = GetItemNumber(dw_prof_few, lv_i_CurrentFewProf, "priority")

lv_b_RoundClass = add_class_lab(lv_ia_ProfClass, lv_i_Priority, lv_s_Days1, lv_t_StartTime1, lv_t_EndTime1, &
lv_s_LabDays1, lv_t_LabStartTime1, lv_t_LabEndTime1, lv_s_LabDays2, lv_t_LabStartTime2, &
    lv_t_LabEndTime2)
IF (lv_b_FoundClass) THEN
lv_b_DeleteClass = delete_class_lab(lv_ia_ProfClass, lv_i_Priority, lv_s_Days1,lv_t_StartTime1, lv_t_EndTime1, &
lv_s_LabDays1, lv_t_LabStartTime1, lv_t_LabEndTime1, lv_s_LabDays2, lv_t_LabStartt ime2, &
lv_t_LabEndTime2)
END IF

IF (lv_b_FoundClass AND lv_b_DeleteClass) THEN
lv_i_Loop2 = 0
lv_b_NotEmpty = TRUE
DO WHILE (lv_b_NotEmpty)
   lv_i_Loop2 = lv_i_Loop2 + 1
   lv_s_TempSection = Trim(GetItemString(dw_prof_few, lv_i_CurrentFewProf, "section_") + String(lv_i_Loop2))
   IF (ISNull(lv_s_TempSection) OR (lv_s_TempSection = "")) THEN
      lv_b_NotEmpty = FALSE
      END IF
   END IF
   LOO

SetItem(dw_prof_few, lv_i_CurrentFewProf, "course_") + String(lv_i_Loop2), lv_i_CurrentCourseNum)
SetItem(dw_prof_few, lv_i_CurrentFewProf, "section_") + String(lv_i_Loop2), lv_s_SectionNum)

SetItem(dw_prof_extra, lv_i_CurrentExtraProf, "course_") + String(lv_i_CurrentClass ), 0)
SetItem(dw_prof_extra, lv_i_CurrentExtraProf, "section_") + String(lv_i_CurrentClass s), "")

IF (lv_b_Next OR lv_b_Above) THEN
lv_s_SectionNum2 = Trim(GetItemString(dw_class_Info, lv_i_OtherSection, "section_num"))
SetItem(dw_prof_few, lv_i_CurrentFewProf, "course_") + String(lv_i_Loop2 + 1), lv_i_CourseNum2)
SetItem(dw_prof_few, lv_i_CurrentFewProf, "section_") + String(lv_i_Loop2 + 1),
IF (lv_b_Next) THEN
    SetItem(dw_prof_extra, lv_i_CurrentExtraProf, "course" + String(lv_i_CurrentClass + 1), 0)
    SetItem(dw_prof_extra, lv_i_CurrentExtraProf, "section" + String(lv_i_CurrentntClass + 1), "")
ELSE
    SetItem(dw_prof_extra, lv_i_CurrentExtraProf, "course" + String(lv_i_CurrentClass - 1), 0)
    SetItem(dw_prof_extra, lv_i_CurrentExtraProf, "section" + String(lv_i_CurrentntClass - 1), "")
END IF
END IF
lv_d_TotalCredits = GetItemDecimal(dw_prof_few, lv_i_CurrentFewProf, "credits")
lv_d_ExtraTotalCredits = GetItemDecimal(dw_prof_extra, lv_i_CurrentExtraProf, "credits")

IF (lv_s_Lab = "y") THEN
    IF (lv_b_Next OR lv_b_Above) THEN
        lv_d_TotalCredits = lv_d_TotalCredits + 2
        lv_d_ExtraTotalCredits = lv_d_ExtraTotalCredits - 2
    ELSE
        lv_d_TotalCredits = lv_d_TotalCredits + 1.5
        lv_d_ExtraTotalCredits = lv_d_ExtraTotalCredits - 1.5
    END IF
ELSE
    lv_d_TotalCredits = lv_d_TotalCredits + 1
    lv_d_ExtraTotalCredits = lv_d_ExtraTotalCredits - 1
END IF
SetItem(dw_prof_few, lv_i_CurrentFewProf, "credits", lv_d_TotalCredits)
SetItem(dw_prof_extra, lv_i_CurrentExtraProf, "credits", lv_d_ExtraTotalCredits)
lv_b_Done = TRUE
lv_i_TotalNumClass = RowCount(dw_class_info)
END IF
ELSE
    lv_b_Continue = TRUE
END IF
END IF
END IF
lv_i_CurrentExtraProf = lv_i_CurrentExtraProf + 1
IF (lv_i_CurrentExtraProf > lv_i_TotalExtraProf) THEN
  lv_i_CurrentExtraProf = 1
  lv_i_CurrentClass = lv_i_CurrentClass - 1
  IF (lv_i_CurrentClass < 1) THEN
    lv_b_Done = TRUE
    lv_b_Exit = TRUE
  END IF
END IF
LOOP
LOOP
IF (lv_b_FoundClass OR lv_b_Exit) THEN
  EXIT
END IF
NEXT

// move "done" professors ... recalculate rows in extra and few
lv_i_CurrentRow = 1
FOR lv_i_CurrentProf = 1 TO lv_i_TotalExtraProf
  lv_d_TotalCredits = GetItemDecimal(dw_prof_extra, lv_i_CurrentRow, "credits")
  lv_d_NeedCredits = GetItemDecimal(dw_prof_extra, lv_i_CurrentRow, "needed_credits")
  IF (lv_d_TotalCredits < lv_d_NeedCredits - 0.5) THEN
    lv_s_Name = Trim(ItemString(dw_prof_extra, lv_i_CurrentRow, "name"))
    lv_i_Priority = GetItemNumber(dw_prof_extra, lv_i_CurrentRow, "priority")
    lv_i_CurrentPreference = GetItemNumber(dw_prof_extra, lv_i_CurrentRow, "current_preference")
    lv_i_CurrentPrevious = GetItemNumber(dw_prof_extra, lv_i_CurrentRow, "current_previous")
    lv_i_AffectedRow = InsertRow(dw_prof_few, 0)
    SetItem(dw_prof_few, lv_i_AffectedRow, "name", lv_s_Name)
    SetItem(dw_prof_few, lv_i_AffectedRow, "priority", lv_i_Priority)
    SetItem(dw_prof_few, lv_i_AffectedRow, "needed_credits", lv_d_NeedCredits)
Window: w_scheduling
Library: e:\thesis\appl\schedule.pbl
Date: 5/1/95 Time: 20:37:42

SetItem(dw_prof_few, lv_i_AddedRow, "current_preference", lv_i_CurrentPreference)
SetItem(dw_prof_few, lv_i_AddedRow, "current_previous", lv_i_CurrentPrevious)
SetItem(dw_prof_few, lv_i_AddedRow, "credits", lv_d_TotalCredits)

FOR lv_i_Loop1 = 1 TO 8
lv_i_TempCourse=GetItemNumber(dw_prof_extra, lv_i_CurrentRow, "course_" + String(lv_i_Loop1))
lv_s_TempSection=Trim(GetItemString(dw_prof_extra, lv_i_CurrentRow, "section_" + String(lv_i_Loop1)))
SetItem(dw_prof_few, lv_i_AddedRow, "course_" + String(lv_i_Loop1), lv_i_TempCourse)
SetItem(dw_prof_few, lv_i_AddedRow, "section_" + String(lv_i_Loop1), lv_s_TempSection)
NEXT

DeleteRow(dw_prof_extra, lv_i_CurrentRow)
ELSEIF ((lv_d_TotalCredits >= lv_d_NeedCredits - 0.5) AND (lv_d_TotalCredits <= lv_d_NeedCredits)) THEN
lv_s_Name = Trim(GetItemString(dw_prof_extra, lv_i_CurrentRow, "name"))
lv_i_AddedRow = InsertRow(dw_prof_okay, 0)
SetItem(dw_prof_okay, lv_i_AddedRow, "name", lv_s_Name)
SetItem(dw_prof_okay, lv_i_AddedRow, "credits", lv_d_TotalCredits)

FOR lv_i_Loop1 = 1 TO 8
lv_i_TempCourse=GetItemNumber(dw_prof_extra, lv_i_CurrentRow, "course_" + String(lv_i_Loop1))
lv_s_TempSection=Trim(GetItemString(dw_prof_extra, lv_i_CurrentRow, "section_" + String(lv_i_Loop1)))
SetItem(dw_prof_okay, lv_i_AddedRow, "course_" + String(lv_i_Loop1), lv_i_TempCourse)
SetItem(dw_prof_okay, lv_i_AddedRow, "section_" + String(lv_i_Loop1), lv_s_TempSection)
NEXT

DeleteRow(dw_prof_extra, lv_i_CurrentRow)
ELSE
lv_i_CurrentRow = lv_i_CurrentRow + 1
END IF
NEXT

lv_i_CurrentRow = 1
FOR lv_i_CurrentProf = 1 TO lv_i_TotalFewProf
  lv_d_TotalCredits = GetItemDecimal(dw_prof_few, lv_i_CurrentRow, "credits")
  lv_d_NeedCredits = GetItemDecimal(dw_prof_few, lv_i_CurrentRow, "needed_credits")

  IF (lv_d_TotalCredits > lv_d_NeedCredits) THEN
    lv_s_Name = Trim(GetItemString(dw_prof_few, lv_i_CurrentRow, "name"))
    lv_i_Priority = GetItemNumber(dw_prof_few, lv_i_CurrentRow, "priority")
    lv_i_CurrentPreference = GetItemNumber(dw_prof_few, lv_i_CurrentRow, "current_preference")
    lv_i_CurrentPrevious = GetItemNumber(dw_prof_few, lv_i_CurrentRow, "current_previous")
    lv_i_AddedRow = InsertRow(dw_prof_extra, 0)
    SetItem(dw_prof_extra, lv_i_AddedRow, "name", lv_s_Name)
    SetItem(dw_prof_extra, lv_i_AddedRow, "priority", lv_i_Priority)
    SetItem(dw_prof_extra, lv_i_AddedRow, "needed_credits", lv_d_NeedCredits)
    SetItem(dw_prof_extra, lv_i_AddedRow, "current_preference", lv_i_CurrentPreference)
    SetItem(dw_prof_extra, lv_i_AddedRow, "current_previous", lv_i_CurrentPrevious)
    SetItem(dw_prof_extra, lv_i_AddedRow, "credits", lv_d_TotalCredits)

    FOR lv_i_Loop1 = 1 TO 8
      lv_i_TempCourse = GetItemNumber(dw_prof_few, lv_i_CurrentRow, "course_" + String(lv_i_Loop1))
      lv_s_TempSection = Trim(GetItemString(dw_prof_few, lv_i_CurrentRow, "section_" + String(lv_i_Loop1)))
      SetItem(dw_prof_extra, lv_i_AddedRow, "course_" + String(lv_i_Loop1), lv_i_TempCourse)
      SetItem(dw_prof_extra, lv_i_AddedRow, "section_" + String(lv_i_Loop1), lv_s_TempSection)
    NEXT

    DeleteRow(dw_prof_few, lv_i_CurrentRow)
  ELSEIF ((lv_d_TotalCredits >= lv_d_NeedCredits - 0.5) AND (lv_d_TotalCredits <= lv_d_NeedCredits)) THEN
    lv_s_Name = Trim(GetItemString(dw_prof_few, lv_i_CurrentRow, "name"))
lv_i_AddedRow = InsertRow(dw_prof_okay, 0)
SetItem(dw_prof_okay, lv_i_AddedRow, "name", lv_s_Name)
SetItem(dw_prof_okay, lv_i_AddedRow, "credits", lv_d_TotalCredits)

FOR lv_i_Loop1 = 1 TO 8
lv_i_TempCourse=GetItemNumber(dw_prof_few, lv_i_CurrentRow, "course_" + String(lv_i_Loop1))
lv_s_TempSection=Trim(GetItemString(dw_prof_few, lv_i_CurrentRow, "section_" + String(lv_i_Loop1)))
    SetItem(dw_prof_okay, lv_i_AddedRow, "course_" + String(lv_i_Loop1), lv_i_TempCourse)
    SetItem(dw_prof_okay, lv_i_AddedRow, "section_" + String(lv_i_Loop1), lv_s_TempSection)
NEXT
DeleteRow(dw_prof_few, lv_i_CurrentRow)
ELSE
    lv_i_CurrentRow = lv_i_CurrentRow + 1
END IF
NEXT
lv_i_TotalFewProf = RowCount(dw_prof_few)
lv_i_TotalExtraProf = RowCount(dw_prof_extra)
NEXT

End of Script

---

Script for: phase7  event
//Let faculty members with the least amount of credits choose from the last acquired courses
//of the professors with extra credits (based on previous experience). After each transfer is made, thos

//faculty members are sorted into the faculty members with too many classes, faculty members with
//too few classes, and faculty members with an okay number of classes into the appropriate datawindows.

//Local Variables
Boolean lv_b_DeleteClass
Boolean lv_b_Done
Boolean lv_b_Continue
Boolean lv_b_Found
Boolean lv_b_Next
Boolean lv_b_Above
Boolean lv_b_Exit
Boolean lv_b_FoundClass
Boolean lv_b_NotEmpty
Decimal lv_d_NeedCredits
Decimal lv_d_TotalCredits
Decimal lv_d_ExtraTotalCredits
Integer lv_i_TotalNumClass
Integer lv_i_TotalFewProf
Integer lv_i_TotalExtraProf
Integer lv_i_AddedRow
Integer lv_i_ClassRow
Integer lv_i_CurrentRow
Integer lv_i_CurrentFewProf
Integer lv_i_CurrentExtraProf
Integer lv_i_CurrentProf
Integer lv_i_CurrentPreference
Integer lv_i_CurrentPrevious
Integer lv_i_Class
Integer lv_i_CurrentClass
Integer lv_i_CurrentCourseNum
Integer lv_i_CourseNum2
Integer lv_i_TempCourse
Integer lv_i_PrevCourseNum
Integer lv_i_OtherSection
Integer lv_i_Priority
Integer lv_i_MaxClass = 0
Integer lv_i_Loop
Integer lv_i_Loop1
Integer lv_i_Loop2
Integer lv_i_Loop3
Window: w_scheduling
Library: c:\thesis\apl\schedule.pbl
Date: 5/1/95   Time: 20:37:42

String   lv_s_Name
String   lv_s_TimePref
String   lv_s_SectionNum
String   lv_s_SectionNum2
String   lv_s_TempSection
String   lv_s_Lab
String   lv_s_Days1
String   lv_s_Days2
String   lv_s_LabDays1
String   lv_s_LabDays2
Time     lv_t_StartTime1
Time     lv_t_StartTime2
Time     lv_t_EndTime1
Time     lv_t_EndTime2
Time     lv_t_LabStartTime1
Time     lv_t_LabStartTime2
Time     lv_t_LabEndTime1
Time     lv_t_LabEndTime2

//End Local Variables

lv_i_TotalFewProf = RowCount(dw_prof_few)
lv_i_TotalExtraProf = RowCount(dw_prof_extra)

FOR lv_i_Loop3 = 1 TO lv_i_TotalExtraProf
   FOR lv_i_Loop = 1 TO lv_i_TotalExtraProf
      lv_i_Loop1 = 0
      lv_b_NotEmpty = TRUE
      DO WHILE (lv_b_NotEmpty)
         lv_i_Loop1 = lv_i_Loop1 + 1
         lv_s_TempSection=Trim(GetItemString(dw_prof_extra, lv_i_Loop, "section_" + String(lv_i_Loop
1))
      IF ((ISNull(lv_s_TempSection) OR (lv_s_TempSection = ")) AND (lv_i_Loop1 - 1 > lv_i_MaxClass)) T
      HEN
         lv_i_MaxClass = lv_i_Loop1 - 1
         lv_b_NotEmpty = FALSE
      EXIT
END IF

IF (lv_i_Loop1 >= 8) THEN
  lv_b_NotEmpty = FALSE
  EXIT
END IF

LOOP

NEXT

SetSort(dw_prof_extra, "credits D, priority A")
Sort(dw_prof_extra)

SetSort(dw_prof_few, "credits A, priority A")
Sort(dw_prof_few)

FOR lv_i_CurrentFewProf = 1 TO lv_i_TotalFewProf
  lv_b_Exit = FALSE
  lv_b_FoundClass = FALSE
  DO WHILE (NOT lv_b_FoundClass)
    lv_i_CurrentPrevious = GetItemNumber(dw_prof_few, lv_i_CurrentFewProf, "current_previous")
    lv_i_CurrentPrevious = lv_i_CurrentPrevious + 1
    SetItem(dw_prof_few, lv_i_CurrentFewProf, "current_previous", lv_i_CurrentPrevious)
    IF (lv_i_CurrentPrevious > 4) THEN
      EXIT
    ELSEIF (lv_i_CurrentPrevious = 1) THEN
      lv_i_PrevCourseNum = GetItemNumber(dw_prof_info, lv_i_CurrentFewProf, "previous_class_1")
    ELSEIF (lv_i_CurrentPrevious = 2) THEN
      lv_i_PrevCourseNum = GetItemNumber(dw_prof_info, lv_i_CurrentFewProf, "previous_class_2")
    ELSEIF (lv_i_CurrentPrevious = 3) THEN
      lv_i_PrevCourseNum = GetItemNumber(dw_prof_info, lv_i_CurrentFewProf, "previous_class_3")
    ELSEIF (lv_i_CurrentPrevious = 4) THEN
      lv_i_PrevCourseNum = GetItemNumber(dw_prof_info, lv_i_CurrentFewProf, "previous_class_4")
    END IF

  lv_i_CurrentClass = lv_i_MaxClass
  lv_i_CurrentExtraProf = 1
lv_b_Done = FALSE
DO WHILE (NOT lv_b_Done)
  lv_b_Continue = TRUE
  lv_b_Next = FALSE
  lv_b_Above = FALSE
lv_i_CurrentCourseNum = GetItemNumber(dw_prof_extra, lv_i_CurrentExtraProf, "course_" + String(lv_i_CurrentClass))
lv_s_SectionNum = Trim(GetItemString(dw_prof_extra, lv_i_CurrentExtraProf, "section_" + String(lv_i_CurrentClass)))
  IF (lv_i_CurrentCourseNum = lv_i_PrevCourseNum) THEN
lv_s_TimePref = Trim(GetItemString(dw_prof_info, lv_i_CurrentFewProf, "time_preference"))
ENDIF
lv_b_Found = FALSE
lv_i_ClassRow = 1
DO WHILE (NOT lv_b_Found)
  lv_i_TempCourse = GetItemNumber(dw_class_info, lv_i_ClassRow, "course_num")
  lv_s_TempSection = Trim(GetItemString(dw_class_info, lv_i_ClassRow, "section_num"))
ENDIF
IF ((lv_i_TempCourse = lv_i_CurrentCourseNum) AND (lv_s_TempSection = lv_s_SectionNum)) THEN
  lv_b_Found = TRUE
  EXIT
ENDIF
lv_i_ClassRow = lv_i_ClassRow + 1
IF (lv_i_Class > lv_i_TotalNumClass) THEN
  EXIT
ENDIF
LOOP
IF (lv_b_Found) THEN
  lv_t_StartTime1 = GetItemTime(dw_class_info, lv_i_ClassRow, "start_time")
  IF (lv_s_TimePref = "am") THEN
    IF (NOT (lv_t_StartTime1 <= Time(11, 20, 0))) THEN
      ...
lv_b_Continue = FALSE
END IF
ELSEIF (lv_s_TimePref = "pm") THEN
  IF (NOT (lv_t_StartTime1 >= Time(11,20,0))) THEN
    lv_b_Continue = FALSE
  END IF
END IF

IF (lv_b_Continue) THEN
  lv_s_Lab = Trim(GetItemString(dw_class_info, lv_i_ClassRow, "lab"))

  IF (lv_s_Lab = "y") THEN
    lv_s_Days1 = Trim(GetItemString(dw_class_info, lv_i_ClassRow, "days"))
    lv_t_EndTime1 = GetItemTime(dw_class_info, lv_i_ClassRow, "end_time")
  END IF

  IF (lv_i_CurrentClass + 1 <= lv_i_TotalNumClass) THEN
    lv_i_CourseNum2 = GetItemNumber(dw_class_info, lv_i_ClassRow + 1, "course_num")
    lv_s_Days2 = Trim(GetItemString(dw_class_info, lv_i_ClassRow + 1, "days"))
    lv_t_StartTime2 = GetItemTime(dw_class_info, lv_i_ClassRow + 1, "start_time")
    lv_t_EndTime2 = GetItemTime(dw_class_info, lv_i_ClassRow + 1, "end_time")
  END IF

  IF ((lv_i_CourseNum2 = lv_i_CurrentCourseNum) AND (lv_s_Days1 = lv_s_Days2) AND
      (lv_t_StartTime1 = lv_t_StartTime2) AND (lv_t_EndTime1 = lv_t_EndTime2)) THEN
    lv_b_Next = TRUE
    lv_b_Above = FALSE
    lv_i_OtherSection = lv_i_ClassRow + 1
  END IF
ELSEIF (lv_i_ClassRow - 1 >= 1) THEN
  lv_i_CourseNum2 = GetItemNumber(dw_class_info, lv_i_ClassRow - 1, "course_num")
  lv_s_Days2 = Trim(GetItemString(dw_class_info, lv_i_ClassRow - 1, "days"))
  lv_t_StartTime2 = GetItemTime(dw_class_info, lv_i_ClassRow - 1, "Start_time")

  IF (lv_t_EndTime2 < Time(9,0,0)) THEN
    lv_t_EndTime2 = Time(9,0,0)
  END IF
  IF (lv_t_StartTime2 > Time(9,0,0)) THEN
    lv_t_StartTime2 = Time(9,0,0)
  END IF

  IF (lv_t_StartTime2 <= lv_t_EndTime2) THEN
    lv_b_Above = FALSE
    lv_b_Next = TRUE
  ELSEIF (lv_t_StartTime1 < lv_t_StartTime2) THEN
    lv_b_Above = TRUE
    lv_b_Next = FALSE
  ELSEIF (lv_t_EndTime1 > lv_t_StartTime2) THEN
    lv_b_Above = TRUE
    lv_b_Next = FALSE
  ELSEIF (lv_t_EndTime1 <= lv_t_StartTime2) THEN
    lv_b_Above = FALSE
    lv_b_Next = TRUE
  END IF
END IF
lv_t_EndTime2=GetItemTime(dw_class_info, lv_i_ClassRow - 1, "end_time")

IF ((lv_i_CourseNum2 = lv_i_CurrentCourseNum) AND (lv_s_Days1 = lv_s_Days2) AND (lv_t_StartTime1 = lv_t_StartTime2) AND (lv_t_EndTime1 = lv_t_EndTime2)) THEN
  lv_b_Next   = FALSE
  lv_b_Above  = TRUE
  lv_i_OtherSection = lv_i_ClassRow - 1
END IF
END IF

lv_s_LabDays1 = Trim(GetItemString(dw_class_info, lv_i_ClassRow, "lab_days"))
lv_t_LabStartTime1=GetItemTime(dw_class_info, lv_i_ClassRow, "lab_start_time")

lv_t_LabEndTime1  = GetItemTime(dw_class_info, lv_i_ClassRow, "lab_end_time")

IF (lv_b_Next OR lv_b_Above) THEN
  lv_s_LabDays2=Trim(GetItemString(dw_class_info, lv_i_OtherSection, "lab_days"))
  lv_t_LabStartTime2=GetItemTime(dw_class_info, lv_i_OtherSection, "lab_start_time")
  lv_t_LabEndTime2=GetItemTime(dw_class_info, lv_i_OtherSection, "lab_end_time")
END IF
ELSE
  lv_s_Days1 = Trim(GetItemString(dw_class_info, lv_i_ClassRow, "days"))
  lv_t_EndTime1 = GetItemTime(dw_class_info, lv_i_ClassRow, "end_time")
  lv_s_LabDays1 = ""
  lv_t_LabStartTime1 = Time(0,0,0)
  lv_t_LabEndTime1 = Time(0,0,0)
  lv_s_LabDays2 = ""
  lv_t_LabStartTime2 = Time(0,0,0)
  lv_t_LabEndTime2 = Time(0,0,0)
END IF

lv_i_Priority = GetItemNumber(dw_prof_few, lv_i_CurrentFewProf, "priority")
lv_b_FoundClass = add_class_lab(lv_ia_ProfClass, lv_i_Priority, lv_s_Days1, lv_t_StartTime1, lv_t_EndTime1, &
  lv_s_LabDays1, lv_t_LabStartTime1, lv_t_LabEndTime1, lv_s_LabDays2, lv_t_LabStartTime2, &
  lv_t_LabEndTime2)

IF (lv_b_FoundClass) THEN
  lv_b_DeleteClass = delete_class_lab(lv_ia_ProfClass, lv_i_Priority, lv_s_Days1, lv
  _t_StartTime1, lv_t_EndTime1, &
  lv_s_LabDays1, lv_t_LabStartTime1, lv_t_LabEndTime1, lv_s_LabDays2, lv_t_LabStartT
  ime2, &
  lv_t_LabEndTime2)
  END IF

IF (lv_b_FoundClass AND lv_b_DeleteClass) THEN
  lv_i_Loop2 = 0
  lv_b_NotEmpty = TRUE
  DO WHILE (lv_b_NotEmpty)
    lv_i_Loop2 = lv_i_Loop2 + 1
    lv_s_TempSection=Trim(GetItemString(dw_prof_few, lv_i_CurrentFewProf, "se
    ction_" + String(lv_i_Loop2))
    IF (ISNull(lv_s_TempSection) OR (lv_s_TempSection = ") THEN
      lv_b_NotEmpty = FALSE
    END IF
  LOOP

SetItem(dw_prof_few, lv_i_CurrentFewProf, "course_" + String(lv_i_Loop2), lv_i_Cur
  rentCourseNum)
SetItem(dw_prof_few, lv_i_CurrentFewProf, "section_" + String(lv_i_Loop2), lv_s_Se
  ctionNum)
SetItem(dw_prof_extra, lv_i_CurrentExtraProf, "course_" + String(lv_i_CurrentClass
  s), 0)
SetItem(dw_prof_extra, lv_i_CurrentExtraProf, "section_" + String(lv_i_CurrentClas
  s), ")
IF (lv_b_Next OR lv_b_Above) THEN
  lv_s_SectionNum2=Trim(GetItemString(dw_class_info, lv_i_OtherSection, "section_num"))
  SetItem(dw_prof_few, lv_i_CurrentFewProf, "course_" + String(lv_i_Loop2 + 1), lv_v_CourseNum2)
  SetItem(dw_prof_few, lv_i_CurrentFewProf, "section_" + String(lv_i_Loop2 + 1), lv_s_SectionNum2)
ENDIF

IF (lv_b_Next) THEN
  SetItem(dw_prof_extra, lv_i_CurrentExtraProf, "course_" + String(lv_i_CurrentClass + 1), 0)
  SetItem(dw_prof_extra, lv_i_CurrentExtraProf, "section_" + String(lv_i_CurrentClass + 1), "")
ELSE
  SetItem(dw_prof_extra, lv_i_CurrentExtraProf, "course_" + String(lv_i_CurrentClass - 1), 0)
  SetItem(dw_prof_extra, lv_i_CurrentExtraProf, "section_" + String(lv_i_CurrentClass - 1), ")")
ENDIF

lv_d_TotalCredits = GetItemDecimal(dw_prof_few, lv_i_CurrentFewProf, "credits")
lv_d_ExtraTotalCredits = GetItemDecimal(dw_prof_extra, lv_i_CurrentExtraProf, "credits")

IF (lv_s_Lab = "y") THEN
  IF (lv_b_Next OR lv_b_Above) THEN
    lv_d_TotalCredits = lv_d_TotalCredits + 2
    lv_d_ExtraTotalCredits = lv_d_ExtraTotalCredits - 2
  ELSE
    lv_d_TotalCredits = lv_d_TotalCredits + 1.5
    lv_d_ExtraTotalCredits = lv_d_ExtraTotalCredits - 1.5
  END IF
ELSE
  lv_d_TotalCredits = lv_d_TotalCredits + 1
  lv_d_ExtraTotalCredits = lv_d_ExtraTotalCredits - 1
ENDIF

SetItem(dw_prof_few, lv_i_CurrentFewProf, "credits", lv_d_TotalCredits)
SetItem(dw_prof_extra, lv_i_CurrentExtraProf, "credits", lv_d_ExtraTotalCredits)

    lv_b_Done = TRUE
    lv_i_TotalNumClass = RowCount(dw_class_info)
    END IF
ELSE
    lv_b_Continue = TRUE
    END IF
END IF
END IF
lv_i_CurrentExtraProf = lv_i_CurrentExtraProf + 1
IF (lv_i_CurrentExtraProf > lv_i_TotalExtraProf) THEN
    lv_i_CurrentExtraProf = 1
    lv_i_CurrentClass = lv_i_CurrentClass - 1
    IF (lv_i_CurrentClass < 1) THEN
        lv_b_Done = TRUE
        lv_b_Exit = TRUE
    END IF
END IF
END IF
LOOP
    EXIT
END IF
NEXT

//move "done" professors ... recalculate rows in extra and few
lv_i_CurrentRow = 1
FOR lv_i_CurrentProf = 1 TO lv_i_TotalExtraProf
    lv_d_TotalCredits = GetItemDecimal(dw_prof_extra, lv_i_CurrentRow, "credits")
    lv_d_NeedCredits = GetItemDecimal(dw_prof_extra, lv_i_CurrentRow, "needed_credits")
    IF (lv_d_TotalCredits < lv_d_NeedCredits - 0.5) THEN
        lv_s_Name = Trim(GetItemString(dw_prof_extra, lv_i_CurrentRow, "name"))
        lv_i_Priority = GetItemNumber(dw_prof_extra, lv_i_CurrentRow, "priority")
        lv_i_CurrentPreference = GetItemNumber(dw_prof_extra, lv_i_CurrentRow, "current_preference")
lv_i_CurrentPrevious = GetItemNumber(dw_prof_extra, lv_i_CurrentRow, "current_previous")

lv_i_AddedRow = InsertRow(dw_prof_few, 0)

SetItem(dw_prof_few, lv_i_AddedRow, "name", lv_s_Name)
SetItem(dw_prof_few, lv_i_AddedRow, "priority", lv_i_Priority)
SetItem(dw_prof_few, lv_i_AddedRow, "needed_credits", lv_d_NeedCredits)
SetItem(dw_prof_few, lv_i_AddedRow, "current_preference", lv_i_CurrentPreference)
SetItem(dw_prof_few, lv_i_AddedRow, "current_previous", lv_i_CurrentPrevious)
SetItem(dw_prof_few, lv_i_AddedRow, "credits", lv_d_TotalCredits)

FOR lv_i_Loop1 = 1 TO 8
lv_i_TempCourse=GetItemNumber(dw_prof_extra, lv_i_CurrentRow, "course_" + String(lv_i_Loop1))
lv_s_TempSection=Trim(GetItemString(dw_prof_extra, lv_i_CurrentRow, "section_" + String(lv_i_Loop1)))
    SetItem(dw_prof_few, lv_i_AddedRow, "course_" + String(lv_i_Loop1), lv_i_TempCourse)
    SetItem(dw_prof_few, lv_i_AddedRow, "section_" + String(lv_i_Loop1), lv_s_TempSection)
NEXT

DeleteRow(dw_prof_extra, lv_i_CurrentRow)
ELSEIF ((lv_d_TotalCredits >= lv_d_NeedCredits - 0.5) AND (lv_d_TotalCredits <= lv_d_NeedCredits)) THEN

lv_s_Name = Trim(GetItemString(dw_prof_extra, lv_i_CurrentRow, "name"))

lv_i_AddedRow = InsertRow(dw_prof_okay, 0)

SetItem(dw_prof_okay, lv_i_AddedRow, "name", lv_s_Name)
SetItem(dw_prof_okay, lv_i_AddedRow, "credits", lv_d_TotalCredits)

FOR lv_i_Loop1 = 1 TO 8
lv_i_TempCourse=GetItemNumber(dw_prof_extra, lv_i_CurrentRow, "course_" + String(lv_i_Loop1))
lv_s_TempSection=Trim(GetItemString(dw_prof_extra, lv_i_CurrentRow, "section_" + String(lv_i_Loop1)))
    SetItem(dw_prof_okay, lv_i_AddedRow, "course_" + String(lv_i_Loop1), lv_i_TempCourse)
SetItem(dw_prof_okay, lv_i_AddedRow, "section_" + String(lv_i_Loop1), lv_s_TempSection)
NEXT

DeleteRow(dw_prof_extra, lv_i_CurrentRow)
ELSE
lv_i_CurrentRow = lv_i_CurrentRow + 1
ENDIF
NEXT

lv_i_CurrentRow = 1
FOR lv_i_CurrentProf = 1 TO lv_i_TotalFewProf
lv_d_TotalCredits = GetItemDecimal(dw_profフエム, lv_i_CurrentRow, "credits")
lv_d_NeedCredits = GetItemDecimal(dw_profフエム, lv_i_CurrentRow, "needed_credits")

IF (lv_d_TotalCredits > lv_d_NeedCredits) THEN
lv_s_Name = Trim(GetItemString(dw_profフエム, lv_i_CurrentRow, "name"))
lv_i_Priority = GetItemNumber(dw_profフエム, lv_i_CurrentRow, "priority")
lv_i_CurrentPreference = GetItemNumber(dw_profフエム, lv_i_CurrentRow, "current_preference")
lv_i_CurrentPrevious = GetItemNumber(dw_profフエム, lv_i_CurrentRow, "current_previous")

lv_i_AddedRow = InsertRow(dw_prof_extra, 0);

SetItem(dw_prof_extra, lv_i_AddedRow, "name", lv_s_Name)
SetItem(dw_prof_extra, lv_i_AddedRow, "priority", lv_i_Priority)
SetItem(dw_prof_extra, lv_i_AddedRow, "needed_credits", lv_d_NeedCredits)
SetItem(dw_prof_extra, lv_i_AddedRow, "current_preference", lv_i_CurrentPreference)
SetItem(dw_prof_extra, lv_i_AddedRow, "current_previous", lv_i_CurrentPrevious)
SetItem(dw_prof_extra, lv_i_AddedRow, "credits", lv_d_TotalCredits)

FOR lv_i_Loop1 = 1 TO 8
lv_i_TempCourse=GetItemNumber(dw_profフエム, lv_i_CurrentRow, "course_" + String(lv_i_Loop 1))
lv_s_TempSection=Trim(GetItemString(dw_profフエム, lv_i_CurrentRow, "section_" + String(lv_i_Loop1)))
SetItem(dw_prof_extra, lv_i_AddedRow, "course_" + String(lv_i_Loop1), lv_i_TempCourse)
SetItem(dw_prof_extra, lv_i_AddedRow, "section_" + String(lv_i_Loop1), lv_s_TempSection)
NEXT

DeleteRow(dw_prof_few, lv_i_CurrentRow)
ELSEIF ((lv_d_TotalCredits >= lv_d_NeedCredits - 0.5) AND (lv_d_TotalCredits <= lv_d_NeedCredits)) THEN

lv_s_Name = Trim(GetItemString(dw_prof_few, lv_i_CurrentRow, "name"))

lv_i_AddedRow = InsertRow(dw_prof_okay, 0)

setItem(dw_prof_okay, lv_i_AddedRow, "name", lv_s_Name)
setItem(dw_prof_okay, lv_i_AddedRow, "credits", lv_d_TotalCredits)

FOR lv_i_Loop1 = 1 TO 8
lv_i_TempCourse = GetItemNumber(dw_prof_few, lv_i_CurrentRow, "course_" + String(lv_i_Loop1))
lv_s_TempSection = Trim(GetItemString(dw_prof_few, lv_i_CurrentRow, "section_" + String(lv_i_Loop1)))

setItem(dw_prof_okay, lv_i_AddedRow, "course_" + String(lv_i_Loop1), lv_i_TempCourse)
setItem(dw_prof_okay, lv_i_AddedRow, "section_" + String(lv_i_Loop1), lv_s_TempSection)
NEXT

DeleteRow(dw_prof_few, lv_i_CurrentRow)
ELSE

lv_i_CurrentRow = lv_i_CurrentRow + 1
END IF

NEXT

lv_i_TotalFewProf = RowCount(dw_prof_few)
lv_i_TotalExtraProf = RowCount(dw_prof_extra)

NEXT

End of Script

Script for: phase8 event
// After each class is assigned, those faculty members are sorted into the faculty members with too many classes, faculty members with too few classes, and faculty members with an okay number of classes into the appropriate data windows.

// Local Variables
Boolean lv_b_Continue
Boolean lv_b_Next
Boolean lv_b_Above
Boolean lv_b_Found
Boolean lv_b_FoundClass
Boolean lv_b_NotEmpty
Decimal lv_d_TotalCredits
Decimal lv_d_NeedCredits
Integer lv_i_ClassRow
Integer lv_i_TotalNumClass
Integer lv_i_TotalFewProf
Integer lv_i_CurrentClass
Integer lv_i_CurrentProf
Integer lv_i_CurrentProf2
Integer lv_i_CurrentPreference
Integer lv_i_CurrentPrevious
Integer lv_i_CurrentCourseNum
Integer lv_i_CourseNum2
Integer lv_i_OtherSection
Integer lv_i_AddedRow
Integer lv_i_TempCourse
Integer lv_i_Priority
Integer lv_i_CurrentRow
Integer lv_i_Loop1
String lv_s_Name
String lv_s_TempSection
String lv_s_SectionNum
String lv_s_SectionNum2
String lv_s_TimePref
String lv_s_Days1
String lv_s_Days2
String lv_s_Lab
String lv_s_LabDays1
String lv_s_LabDays2
lv_i_TotalNumClass = RowCount(dw_extra_class)
IF (lv_i_TotalNumClass > 0) THEN
  SetSort(dw_prof_few, "credits A, priority A")
  Sort(dw_prof_few)

lv_i_TotalFewProf = RowCount(dw_prof_few)

FOR lv_i_CurrentClass = 1 TO lv_i_TotalNumClass
  FOR lv_i_CurrentProf = 1 TO lv_i_TotalFewProf
    lv_b_Continue = TRUE
    lv_b_Next = FALSE
    lv_b_Above = FALSE
    lv_i_CurrentCourseNum = GetItemNumber(dw_extra_class, lv_i_CurrentClass, "course_num")
    lv_s_SectionNum = Trim(GetItemString(dw_extra_class, lv_i_CurrentClass, "section_num"))
    lv_s_TimePref = Trim(GetItemString(dw_prof_info, lv_i_CurrentProf, "time_preference"))

    lv_b_Found = FALSE
    lv_i_ClassRow = 1
    DO WHILE (NOT lv_b_Found)
      lv_i_TempCourse = GetItemNumber(dw_class_info, lv_i_ClassRow, "course_num")
      lv_s_TempSection = Trim(GetItemString(dw_class_info, lv_i_ClassRow, "section_num"))
      IF ((lv_i_TempCourse = lv_i_CurrentCourseNum) AND (lv_s_TempSection = lv_s_SectionNum)) THEN
        lv_b_Found = TRUE
        EXIT
    ENDWHILE
IF (lv_b_Found) THEN
  lv_t_StartTime1 = GetItemTime(dw_class_info, lv_i_ClassRow, "start_time")
  IF (lv_s_TimePref = "am") THEN
    IF (NOT (lv_t_StartTime1 <= Time(11,20,0))) THEN
      lv_b_Continue = FALSE
    END IF
  END IF
ELSEIF (lv_s_TimePref = "pm") THEN
  IF (NOT (lv_t_StartTime1 >= Time(11,20,0))) THEN
    lv_b_Continue = FALSE
  END IF
END IF

IF (lv_b_Continue) THEN
  lv_s_Lab = Trim(GetItemString(dw_class_info, lv_i_ClassRow, "lab"))
  IF (lv_s_Lab = "y") THEN
    lv_s_Days1 = Trim(GetItemString(dw_class_info, lv_i_ClassRow, "days"))
    lv_t_EndTime1 = GetItemTime(dw_class_info, lv_i_ClassRow, "end_time")
    IF (lv_i_ClassRow + 1 <= lv_i_TotalNumClass) THEN
      lv_i_CourseNum2 = GetItemNumber(dw_class_info, lv_i_ClassRow + 1, "course_num")
      lv_s_Days2 = Trim(GetItemString(dw_class_info, lv_i_ClassRow + 1, "days"))
      lv_t_StartTime2 = GetItemTime(dw_class_info, lv_i_ClassRow + 1, "start_time")
      lv_t_EndTime2 = GetItemTime(dw_class_info, lv_i_ClassRow + 1, "end_time")
    END IF
  END IF
END IF

IF ((lv_i_CourseNum2 = lv_i_CurrentCourseNum) AND (lv_s_Days1 = lv_s_Days2) AND (lv_t_StartTime1 = lv_t_StartTime2) AND (lv_t_EndTime1 = lv_t_EndTime2)) THEN
  lv_b_Next = TRUE
lv_b_Above    = FALSE
lv_i_OtherSection = lv_i_ClassRow + 1
END IF
ELSEIF (lv_i_ClassRow - 1 >= 1) THEN
lv_i_CourseNum2  = GetItemNumber(dw_class_info, lv_i_ClassRow - 1, "course_num")
lv_s_Days2       = Trim.GetItemString(dw_class_info, lv_i_ClassRow - 1, "days")
lv_t_StartTime2  = GetItemTime(dw_class_info, lv_i_ClassRow - 1, "start_time")
lv_t_EndTime2    = GetItemTime(dw_class_info, lv_i_ClassRow - 1, "end_time")

IF ((lv_i_CourseNum2 = lv_i_CurrentCourseNum) AND (lv_s_Days1 = lv_s_Days2) AND (lv_t_StartTime1 = lv_t_StartTime2) AND (lv_t_EndTime1 = lv_t_EndTime2)) THEN
lv_b_Next       = FALSE
lv_b_Above      = TRUE
lv_i_OtherSection = lv_i_ClassRow - 1
END IF
END IF

lv_s_LabDays1   = Trim.GetItemString(dw_class_info, lv_i_ClassRow, "lab_days")
lv_t_LabStartTime1 = GetItemTime(dw_class_info, lv_i_ClassRow, "lab_start_time")
lv_t_LabEndTime1 = GetItemTime(dw_class_info, lv_i_ClassRow, "lab_end_time")

IF (lv_b_Next OR lv_b_Above) THEN
lv_s_LabDays2=Trim.GetItemString(dw_class_info, lv_i_OtherSection, "lab_days")
lv_t_LabStartTime2=GetItemTime(dw_class_info, lv_i_OtherSection, "lab_start_time")
lv_t_LabEndTime2  = GetItemTime(dw_class_info, lv_i_OtherSection, "lab_end_time")
END IF
ELSE
lv_s_Days1      = Trim.GetItemString(dw_class_info, lv_i_CurrentClass, "days")
lv_t_EndTime1   = GetItemTime(dw_class_info, lv_i_CurrentClass, "end_time")
lv_s_LabDays1   = ""
lv_t_LabStartTime1 = Time(0,0,0)
lv_t_LabEndTime1 = Time(0,0,0)
lv_s_LabDays2   = ""
lv_t_LabStartTime2 = Time(0,0,0)
Window: w_scheduling
Library: e:\thesis\appl\schedule.pb1
Date: 5/1/95      Time: 20:37:42

lv_t_LabEndTime2 = Time(0,0,0)
END IF

lv_i_Priority = GetItemNumber(dw_prof_few, lv_i_CurrentProf, "priority")

lv_b_FoundClass = add_class_lab(lv_ia_ProfClass, lv_i_Priority, lv_s_Days1, lv_t_StartTime1
, lv_t_EndTime1, &
  lv_s_LabDays1, lv_t_LabStartTime1, lv_t_LabEndTime1, lv_s_LabDays2, lv_t_LabStartTime2,
  &
  lv_t_LabEndTime2)

IF (lv_b_FoundClass) THEN
  lv_i_Loop1 = 0
  lv_b_NotEmpty = TRUE
  DO WHILE (lv_b_NotEmpty)
    lv_i_Loop1 = lv_i_Loop1 + 1
    lv_s_TempSection=Trim(GetItemString(dw_prof_few, lv_i_CurrentProf, "section_" +
      String(lv_i_Loop1)))
    IF (ISNull(lv_s_TempSection) OR (lv_s_TempSection = "")) THEN
      lv_b_NotEmpty = FALSE
    END IF
  LOOP

SetItem(dw_prof_few, lv_i_CurrentProf, "course_" + String(lv_i_Loop1), lv_i_CurrentCourse
  eNum)
SetItem(dw_prof_few, lv_i_CurrentProf, "section_" + String(lv_i_Loop1), lv_s_SectionNum)

IF (lv_b_Next OR lv_b_Above) THEN
  lv_s_SectionNum2=Trim(GetItemString(dw_class_info, lv_i_OtherSection, "section_"
    num"))
SetItem(dw_prof_few, lv_i_CurrentProf, "course_" + String(lv_i_Loop1 + 1), lv_i_Course
  eNum2)
SetItem(dw_prof_few, lv_i_CurrentProf, "section_" + String(lv_i_Loop1 + 1), lv_s_Sect
  ionNum2)
END IF
lv_d_TotalCredits = GetItemDecimal(dw_prof_few, lv_i_CurrentProf, "credits")
IF (lv_s_Lab = "y") THEN
  IF (lv_b_Next OR lv_b_Above) THEN
    lv_d_TotalCredits = lv_d_TotalCredits + 2
    DeleteRow(dw_extra_class, lv_i_OtherSection)
  ELSE
    lv_d_TotalCredits = lv_d_TotalCredits + 1.5
  END IF
ELSE
  lv_d_TotalCredits = lv_d_TotalCredits + 1
END IF
SetItem(dw_prof_few, lv_i_CurrentProf, "credits", lv_d_TotalCredits)
DeleteRow(dw_extra_class, lv_i_ClassRow)
lv_i_TotalNumClass = RowCount(dw_extra_class)
END IF
ELSE
  lv_b_Continue = TRUE
END IF

//Move the professors to the appropriate datawindow (done, extra, or stay at few)
lv_i_CurrentRow = 1
FOR lv_i_CurrentProf2 = 1 TO lv_i_TotalProf
  lv_d_TotalCredits = GetItemDecimal(dw_prof_few, lv_i_CurrentRow, "credits")
  lv_d_NeedCredits = GetItemDecimal(dw_prof_few, lv_i_CurrentRow, "needed_credits")
  IF (lv_d_TotalCredits > lv_d_NeedCredits) THEN
    lv_s_Name = Trim(GetItemString(dw_prof_few, lv_i_CurrentRow, "name"))
    lv_i_Priority = GetItemNumber(dw_prof_few, lv_i_CurrentRow, "priority")
    lv_i_CurrentPreferences = GetItemNumber(dw_prof_few, lv_i_CurrentRow, "current_preference")
    lv_i_CurrentPrevious = GetItemNumber(dw_prof_few, lv_i_CurrentRow, "current_previous")
  END IF
END FOR
lv_i_AddedRow = InsertRow(dw_prof_extra, 0)
SetItem(dw_prof_extra, lv_i_AddedRow, "name", lv_s_Name)
SetItem(dw_prof_extra, lv_i_AddedRow, "priority", lv_i_Priority)
SetItem(dw_prof_extra, lv_i_AddedRow, "needed_credits", lv_d_NeedCRedits)
SetItem(dw_prof_extra, lv_i_AddedRow, "current_preference", lv_i_CurrentPreference)
SetItem(dw_prof_extra, lv_i_AddedRow, "current_previous", lv_i_CurrentPrevious)
SetItem(dw_prof_extra, lv_i_AddedRow, "credits", lv_d_TotalCredits)

FOR lv_i_Loop1 = 1 TO 8
lv_i_TempCourse = GetItemNumber(dw_prof_few, lv_i_CurrentRow, "course_" + String(lv_i_Loop1))
lv_s_TempSection = Trim(GetItemString(dw_prof_few, lv_i_CurrentRow, "section_" + String(lv_i_Loop1)))
SetItem(dw_prof_extra, lv_i_AddedRow, "course_" + String(lv_i_Loop1), lv_i_TempCourse)
SetItem(dw_prof_extra, lv_i_AddedRow, "section_" + String(lv_i_Loop1), lv_s_TempSection)

NEXT

DeleteRow(dw_prof_few, lv_i_CurrentRow)
ELSEIF ((lv_d_TotalCredits >= lv_d_NeedCRedits - 0.5) AND (lv_d_TotalCredits <= lv_d_NeedCRedits)) THEN
 lv_s_Name = Trim(GetItemString(dw_prof_few, lv_i_CurrentRow, "name"))
 lv_i_AddedRow = InsertRow(dw_prof_okay, 0)
SetItem(dw_prof_okay, lv_i_AddedRow, "name", lv_s_Name)
SetItem(dw_prof_okay, lv_i_AddedRow, "credits", lv_d_TotalCredits)

FOR lv_i_Loop1 = 1 TO 8
lv_i_TempCourse = GetItemNumber(dw_prof_few, lv_i_CurrentRow, "course_" + String(lv_i_Loop1))
lv_s_TempSection = Trim(GetItemString(dw_prof_few, lv_i_CurrentRow, "section_" + String(lv_i_Loop1)))
SetItem(dw_prof_okay, lv_i_AddedRow, "course_" + String(lv_i_Loop1), lv_i_TempCourse)

Window: _w_scheduling_
Library: e:\thesis\appl\schedule.pbl
Date: 5/1/95  Time: 20:37:42

SetItem(dw_prof_okay, lv_i_AddedRow, "section_" + String(lv_i_Loop1), lv_s_TempSection)
        NEXT
        DeleteRow(dw_prof_few, lv_i_CurrentRow)
ELSE
        lv_i_CurrentRow = lv_i_CurrentRow + 1
        END IF
NEXT
lv_i_TotalFewProf = RowCount(dw_prof_few)
IF ((DeletedCount(dw_extra_class) > 0) AND (lv_i_CurrentClass > lv_i_TotalNumClass)) THEN
    EXIT
    RND IF
    END IF
NEXT
NEXT
END IF

End of Script

Script for: phase9  event
//Move all assigned classes to the done datawindow with the faculty member assigned to the class.  
//If any classes remain in the extra class datawindow, notify the user of the remaining classes.

//Local Variables
Boolean  lv_b_Continue
Boolean  lv_b_Found
DataWindow lv_dw_Current
Integer  lv_i_ExtraClasses
Integer  lv_i_Loop
Integer  lv_i_ClassRow
Integer  lv_i_TotalRows
Integer  lv_i_CurrentRow


Integer lv_i_InsertedRow
Integer lv_i_CurrentColumn
Integer lv_i_MaxColumn
Integer lv_i_TotalNumClass
Integer lv_i_TempCourse
Integer lv_i_CourseNum
Integer lv_i_MaxCredits
Integer lv_i_MinCredits
Integer lv_i_ClassLimit
Integer lv_i_LabMaxCredits
Integer lv_i_LabMinCredits
String lv_s_ProfessorName
String lv_s_TempSection
String lv_s_MessageString
String lv_s_Term
String lv_s_Department
String lv_s_SectionNum
String lv_s_CourseName
String lv_s_Days
String lv_s_Location
String lv_s_Building
String lv_s_Room
String lv_s_Flags
String lv_s_Lab
String lv_s_LabDays
String lv_s_LabLocation
String lv_s_LabBuilding
String lv_s_LabRoom
Time lv_t_StartTime
Time lv_t_EndTime
Time lv_t_LabStartTime
Time lv_t_LabEndTime

//End Local Variables

Retrieve(dw_class_info, s_parms.term, s_parms.department)
SetSort(dw_class_info, "course_num A, section_num A")
Sort(dw_class_info)
1v_i_TotalNumClass = RowCount(dw_class_info)

FOR 1v_i_Loop = 1 TO 3
   IF (1v_i_Loop = 1) THEN
      1v_dw_Current = dw_prof_extra
      1v_i_MaxColumn = 8
   ELSEIF (1v_i_Loop = 2) THEN
      1v_dw_Current = dw_prof_okay
      1v_i_MaxColumn = 4
   ELSEIF (1v_i_Loop = 3) THEN
      1v_dw_Current = dw_prof_few
      1v_i_MaxColumn = 8
   END IF

   1v_i_TotalRows = RowCount(1v_dw_Current)

FOR 1v_i_CurrentRow = 1 TO 1v_i_TotalRows
   1v_s_ProfessorName = Trim(GetItemString(1v_dw_Current, 1v_i_CurrentRow, "name"))
   FOR 1v_i_CurrentColumn = 1 TO 1v_i_MaxColumn
      1v_b_Continue = FALSE
      1v_s_SectionNum = Trim(GetItemString(1v_dw_Current, 1v_i_CurrentRow, "section_" + String(1v_i_CurrentColumn)))
      IF (NOT (IsNull(1v_s_SectionNum) OR (1v_s_SectionNum = ""))) THEN
         1v_b_Continue = TRUE
      END IF
   END FOR

   IF (1v_b_Continue) THEN
      1v_i_CourseNum = GetItemNumber(1v_dw_Current, 1v_i_CurrentRow, "course_" + String(1v_i_CurrentColumn))
   END IF

   1v_b_Found = FALSE
   1v_i_ClassRow = 0
   DO WHILE (NOT 1v_b_Found)
      1v_i_ClassRow = 1v_i_ClassRow + 1
      1v_i_TempCourse = GetItemNumber(dw_class_info, 1v_i_ClassRow, "course_num")
   END DO

   ...
lv_s_TempSection = Trim.GetItemString(dw_class_info, lv_i_ClassRow, "section_num")

IF ((lv_i_TempCourse = lv_i_CourseNum) AND (lv_s_TempSection = lv_s_SectionNum)) THEN
lv_b_Found = TRUE
EXIT
END IF

IF (lv_i_ClassRow = lv_i_TotalNumClass) THEN
EXIT
END IF
LOOP

IF (lv_b_Found) THEN
lv_s_Term = Trim.GetItemString(dw_class_info, lv_i_ClassRow, "term")
lv_s_Department = Trim.GetItemString(dw_class_info, lv_i_ClassRow, "department")
lv_s_CourseName = Trim.GetItemString(dw_class_info, lv_i_ClassRow, "name")
lv_s_Days = Trim.GetItemString(dw_class_info, lv_i_ClassRow, "days")
lv_t_StartTime = GetItemTime(dw_class_info, lv_i_ClassRow, "start_time")
lv_t_EndTime = GetItemTime(dw_class_info, lv_i_ClassRow, "end_time")
lv_s_Location = Trim.GetItemString(dw_class_info, lv_i_ClassRow, "location")
lv_i_MaxCredits = GetItemNumber(dw_class_info, lv_i_ClassRow, "max_credits")
lv_i_MinCredits = GetItemNumber(dw_class_info, lv_i_ClassRow, "min_credits")
lv_s_Building = Trim.GetItemString(dw_class_info, lv_i_ClassRow, "building")
lv_s_Room = Trim.GetItemString(dw_class_info, lv_i_ClassRow, "room")
lv_i_ClassLimit = GetItemNumber(dw_class_info, lv_i_ClassRow, "class_limit")
lv_s_Flags = Trim.GetItemString(dw_class_info, lv_i_ClassRow, "flags")
lv_s_Lab = Trim.GetItemString(dw_class_info, lv_i_ClassRow, "lab")

IF (lv_s_Lab = "y") THEN
lv_s_LabDays = Trim.GetItemString(dw_class_info, lv_i_ClassRow, "lab_days")
lv_t_LabStartTime = GetItemTime(dw_class_info, lv_i_ClassRow, "lab_start_time")
lv_t_LabEndTime = GetItemTime(dw_class_info, lv_i_ClassRow, "lab_end_time")
lv_s_LabLocation = Trim.GetItemString(dw_class_info, lv_i_ClassRow, "lab_location")
lv_i_LabMaxCredits = GetItemNumber(dw_class_info, lv_i_ClassRow, "lab_max_credits")
lv_i_LabMinCredits = GetItemNumber(dw_class_info, lv_i_ClassRow, "lab_min_credits")
lv_s_LabBuilding = Trim(GetItemString(dw_class_info, lv_i_ClassRow, "lab_building "))

lv_s_LabRoom = Trim(GetItemString(dw_class_info, lv_i_ClassRow, "lab_room"))

END IF

lv_i_InsertedRow = InsertRow(dw_final_classes, 0)

SetItem(dw_final_classes, lv_i_InsertedRow, "term", lv_s_Term)
SetItem(dw_final_classes, lv_i_InsertedRow, "department", lv_s_Department)
SetItem(dw_final_classes, lv_i_InsertedRow, "course_num", lv_i_CourseNum)
SetItem(dw_final_classes, lv_i_InsertedRow, "section_num", lv_s_SectionNum)
SetItem(dw_final_classes, lv_i_InsertedRow, "name", lv_s_Name)
SetItem(dw_final_classes, lv_i_InsertedRow, "days", lv_s_Days)
SetItem(dw_final_classes, lv_i_InsertedRow, "start_time", lv_t_StartTime)
SetItem(dw_final_classes, lv_i_InsertedRow, "end_time", lv_t_EndTime)
SetItem(dw_final_classes, lv_i_InsertedRow, "location", lv_s_Location)
SetItem(dw_final_classes, lv_i_InsertedRow, "max_credits", lv_i_MaxCredits)
SetItem(dw_final_classes, lv_i_InsertedRow, "min_credits", lv_i_MinCredits)
SetItem(dw_final_classes, lv_i_InsertedRow, "building", lv_s_Building)
SetItem(dw_final_classes, lv_i_InsertedRow, "room", lv_s_Room)
SetItem(dw_final_classes, lv_i_InsertedRow, "class_limit", lv_i_ClassLimit)
SetItem(dw_final_classes, lv_i_InsertedRow, "flags", lv_s_Flags)
SetItem(dw_final_classes, lv_i_InsertedRow, "professor", lv_s_ProfessorName)

IF (lv_s_Lab = "y") THEN
SetItem(dw_final_classes, lv_i_InsertedRow, "lab_days", lv_s_LabDays)
SetItem(dw_final_classes, lv_i_InsertedRow, "lab_start_time", lv_t_LabStartTime)
SetItem(dw_final_classes, lv_i_InsertedRow, "lab_end_time", lv_t_LabEndTime)
SetItem(dw_final_classes, lv_i_InsertedRow, "lab_location", lv_s_LabLocation)
SetItem(dw_final_classes, lv_i_InsertedRow, "lab_max_credits", lv_i_LabMaxCredits)
SetItem(dw_final_classes, lv_i_InsertedRow, "lab_min_credits", lv_i_LabMinCredits)
SetItem(dw_final_classes, lv_i_InsertedRow, "lab_building", lv_s_LabBuilding)
SetItem(dw_final_classes, lv_i_InsertedRow, "lab_room", lv_s_LabRoom)
END IF

END IF
Window: w_scheduling
Library: e:\thesis\appl\schedule.pbl
Date: 5/1/95   Time: 20:37:42

END IF
NEXT
NEXT

lv_i_ExtraClasses = RowCount(dw_extra_class)
IF (lv_i_ExtraClasses > 0) THEN
    lv_s_MessageString = "Unassigned classes include:-n"
    FOR lv_i_Extra = 1 TO lv_i_ExtraClasses
        lv_i_TempCourse = GetItemNumber(dw_extra_class, lv_i_Extra, "course_num")
        lv_i_TempSection = Trim(GetItemString(dw_extra_class, lv_i_Extra, "section_num"))
        lv_s_MessageString = lv_s_MessageString + "    " + String(lv_i_TempCourse) + "    " + lv_i_TempSection + "-n"
    NEXT
    MessageBox("Information", lv_s_MessageString, Information!, OK!)
END IF

SetSort(dw_final_classes, "course_num A, section_num A")
Sort(dw_final_classes)

End of Script

DataWindow: dw_prof_extra
X = 0       Y = 13       Width = 2821    Height = 405
TabOrder = 40 DataObject = "d_assign_prof_long"   TitleBar = true
Title = "Prof w/Extra (main)"   HScrollBar = true   VScrollBar = true
Border = true   LiveScroll = true   BorderStyle = stylebox!

CommandButton: cb_clear
X = 572      Y = 1625      Width = 554    Height = 109
TabOrder = 90 Visible = true    Text = "Clear Existing Schedule"
//Local Variables
Integer lv_i_RowCount
Integer lv_i_Loop
//End Local Variables

SetPointer(Hourglass!)

SetRedraw(dw_final_classes, FALSE)

//Clear out final schedule datawindow
lv_i_RowCount = RowCount(dw_final_classes)
FOR lv_i_Loop = 1 to lv_i_RowCount
  DeleteRow(dw_final_classes, 1)
NEXT

//Clear out extra professor info datawindow
lv_i_RowCount = RowCount(dw_prof_extra)
FOR lv_i_Loop = 1 to lv_i_RowCount
  DeleteRow(dw_prof_extra, 1)
NEXT

//Clear out few professor info datawindow
lv_i_RowCount = RowCount(dw_prof_few)
FOR lv_i_Loop = 1 to lv_i_RowCount
  DeleteRow(dw_prof_few, 1)
NEXT

//Clear out done professor info datawindow
lv_i_RowCount = RowCount(dw_prof_okay)
FOR lv_i_Loop = 1 to lv_i_RowCount
  DeleteRow(dw_prof_okay, 1)
NEXT

//Clear out extra classes datawindow
lv_i_RowCount = RowCount(dw_extra_class)
FOR lv_i_Loop = 1 to lv_i_RowCount
  DeleteRow(dw_extra_class,1)
NEXT

//Re-Retrieve data for class information
Retrieve(dw_class_info, s_parms.term, s_parms.department)
SetSort(dw_class_info, "course_num A, section_num A")
Sort(dw_class_info)

//Re-Retrieve data for professor information
Retrieve(dw_prof_info, s_parms.department)
SetSort(dw_prof_info, "calculated_priority A")
Sort(dw_prof_info)

cb_start.enabled = TRUE
cb_cancel.enabled = TRUE
cb_clear.enabled = FALSE

SetRedraw(dw_final_classes, TRUE)

End of Script

DataWindow: dw_prof_info
X = 577 Y = 33 Width = 494 Height = 361
TabBorder = 30 DataObject = "d_prof_info_schedule"
Title = "Professor Information" Border = true
BorderStyle = stylebox!
TitleBar = true LiveScroll = true
DataWindow: dw_final_classes
X = 28  Y = 361  Width = 2766  Height = 1161
TabOrder = 70  Visible = true  Enabled = true  DataObject = "d_final_schedule"
VScrollBar = true  Border = true  LiveScroll = true  BorderStyle = stylebox!

CommandButton: cb_cancel
X = 2090  Y = 1625  Width = 247  Height = 109
TabOrder = 80  Visible = true  Text = "Cancel"

Script for: clicked event
//Close the window w_scheduling and does not save the changes made to the class schedule.

//Local Variables
//End Local Variables

close(w_scheduling)

End of Script

CommandButton: cb_ok
X = 2446  Y = 1625  Width = 247  Height = 109
TabOrder = 110  Visible = true  Enabled = true  Text = "$OK"

Script for: clicked event
//Close the window w_scheduling and saves the new class schedule.

//Local Variables
//End Local Variables
Window: w_scheduling
Library: e:\thesis\appl\schedule.pbl
Date: 5/1/95    Time: 20:37:42

// Update database with new final class schedule, if one was made.
IF ((ModifiedCount(dw_final_classes) > 0) OR (DeletedCount(dw_final_classes) > 0)) THEN
  IF (Update(dw_final_classes) = 1) THEN
    COMMIT;
  ELSE
    ROLLBACK;
  END IF
END IF

// Close window
close(w_scheduling)

End of Script
DataWindow: d_final_schedule
Library: e:\thesis\appl\schedule.pbl
Date: 5/2/95   Time: 17:37:51

Retrieve: FBSELECT(TABLE(NAME="class_schedule")  COLUMN(NAME="class_schedule.term")  COLUMN(NAME="class_sc
Arguments: arg_s_term arg_s_department
Update Table: class_schedule
Filter: None
Sort: None
Sparse: None
Column: days
   Updateable: Yes
   Key: No
   Format: "[general]"
   Border style: None
   Validation: None
   Validation Message: None
   Tab Sequence: 0
   Initial Value: None
   Edit Style: Edit
   Edit limit: 6
Column: start_time
   Updateable: Yes
   Key: No
   Format: "[time]"
   Border style: None
   Validation: None
   Validation Message: None
   Tab Sequence: 0
DataWindow: d_final_schedule
Library: e:\thesis\appl\schedule.pbl
Date: 5/2/95   Time: 17:37:51

Edit Style: Edit
Edit limit: 0

Column: end_time
  Updateable: Yes
  Key: No
  Format: "[time]"
  Border style: None
  Validation: None
  Validation Message: None
  Tab Sequence: 0
  Initial Value: None
  Edit Style: Edit
  Edit limit: 0

Column: location
  Updateable: Yes
  Key: No
  Format: "[general]"
  Border style: None
  Validation: None
  Validation Message: None
  Tab Sequence: 0
  Initial Value: None
  Edit Style: Edit
  Edit limit: 3

Column: max_credits
DataWindow: d_final_schedule
Library: e:\thesis\appl\schedule.pbl
Date: 5/2/95  Time: 17:37:51

Key: No
Format: "[general]"
Border style: None
Validation: None
Validation Message: None
Tab Sequence: 0
Initial Value: None
Edit Style: Edit
Edit limit: 0

Column: min_credits
  Updateable: Yes
  Key: No
  Format: "[general]"
  Border style: None
  Validation: None
  Validation Message: None
  Tab Sequence: 0
  Initial Value: None
  Edit Style: Edit
  Edit limit: 0

Column: building
  Updateable: Yes
  Key: No
  Format: "[general]"
  Border style: None
DataWindow: d_final_schedule
Library: e:\thesis\appl\schedule.pbl
Date: 5/2/95  Time: 17:37:51

Validation Message: None
Tab Sequence: 0
Initial Value: None
Edit Style: Edit
Edit limit: 10

Column: room
Updateable: Yes
Key: No
Format: "[general]"
Border style: None
Validation: None
Validation Message: None
Tab Sequence: 0
Initial Value: None
Edit Style: Edit
Edit limit: 4

Column: class_limit
Updateable: Yes
Key: No
Format: "[general]"
Border style: None
Validation: None
Validation Message: None
Tab Sequence: 0
Initial Value: None
DataWindow: _d_final_schedule_
Library: e:\thesis\appl\schedule.pbl
Date: 5/2/95  Time: 17:37:51

   Edit limit: 0
Column: flags
   Updateable: Yes
   Key: No
   Format: "[general]"
   Border style: None
   Validation: None
   Validation Message: None
   Tab Sequence: 0
   Initial Value: None
   Edit Style: Edit
   Edit limit: 10
Column: lab_days
   Updateable: Yes
   Key: No
   Format: "[general]"
   Border style: None
   Validation: None
   Validation Message: None
   Tab Sequence: 0
   Initial Value: None
   Edit Style: Edit
   Edit limit: 6
Column: lab_start_time
   Updateable: Yes
DataWindow: d_final_schedule
Library: e:\thesis\appl\schedule.pbl
Date: 5/2/95   Time: 17:37:51

Format: "[time]"
Border style: None
Validation: None
Validation Message: None
Tab Sequence: 0
Initial Value: None
Edit Style: Edit
Edit limit: 0

Column: lab_end_time
Updateable: Yes
Key: No
Format: "[time]"
Border style: None
Validation: None
Validation Message: None
Tab Sequence: 0
Initial Value: None
Edit Style: Edit
Edit limit: 0

Column: lab_location
Updateable: Yes
Key: No
Format: "[general]"
Border style: None
Validation: None
Column: lab_max_credits
  Updateable: Yes
  Key: No
  Format: "[general]"
  Border style: None
  Validation: None
  Validation Message: None
  Tab Sequence: 0
  Initial Value: None
  Edit Style: Edit
  Edit limit: 0

Column: lab_min_credits
  Updateable: Yes
  Key: No
  Format: "[general]"
  Border style: None
  Validation: None
  Validation Message: None
  Tab Sequence: 0
  Initial Value: None
  Edit Style: Edit
Column: lab_building
  Updateable: Yes
  Key: No
  Format: "[general]"
  Border style: None
  Validation: None
  Validation Message: None
  Tab Sequence: 0
  Initial Value: None
  Edit Style: Edit
  Edit limit: 10

Column: lab_room
  Updateable: Yes
  Key: No
  Format: "[general]"
  Border style: None
  Validation: None
  Validation Message: None
  Tab Sequence: 0
  Initial Value: None
  Edit Style: Edit
  Edit limit: 4

Column: course_num
  Updateable: Yes
  Key: Yes
DataWindow: d_final_schedule
Library: e:\thesis\appl\schedule.pbl
Date: 5/2/95    Time: 17:37:51

Border style: None
Validation: None
Validation Message: None
Tab Sequence: 0
Initial Value: None
Edit Style: Edit
Edit limit: 0

Column: section_num
Updateable: Yes
Key: Yes
Format: "[general]"
Border style: None
Validation: None
Validation Message: None
Tab Sequence: 0
Initial Value: None
Edit Style: Edit
Edit limit: 3

Column: name
Updateable: Yes
Key: No
Format: "[general]"
Border style: None
Validation: None
Validation Message: None
DataWindow: d_final_schedule
Library: e:\thesis\appl\schedule.pbl
Date: 5/2/95   Time: 17:37:51

    Initial Value: None
    Edit Style: Edit
    Edit limit: 20
Column: lab_professor
    Updateable: Yes
    Key: No
    Format: "[general]"
    Border style: None
    Validation: None
    Validation Message: None
    Tab Sequence: 0
    Initial Value: None
    Edit Style: Edit
    Edit limit: 35
Column: professor
    Updateable: Yes
    Key: No
    Format: "[general]"
    Border style: None
    Validation: None
    Validation Message: None
    Tab Sequence: 0
    Initial Value: None
    Edit Style: Edit
    Edit limit: 35
DataWindow: d_class_info_schedule
Library: e:\thesis\appl\schedule.pbl
Date: 5/2/95    Time: 17:45:41

Retrieve: FBSELECT(TABLE(NAME="class_information") COLUMN(NAME="class_information.term") COLUMN(NAME="c1

Arguments: arg_s_term arg_s_department
Update Table: class_information
Filter: None
Sort: None
Sparse: None

Column: term
  Updateable: Yes
  Key: Yes
  Format: "[general]"
  Border style: None
  Validation: None
  Validation Message: None
  Tab Sequence: 0
  Initial Value: None
  Edit Style: Edit
  Edit limit: 10

Column: department
  Updateable: Yes
  Key: Yes
  Format: "[general]"
  Border style: None
  Validation: None
  Validation Message: None
  Tab Sequence: 0
DataWindow: d_class_info_schedule
Library: e:\thesis\appl\schedule.pbl
Date: 5/2/95   Time: 17:49:41

Edit Style: Edit
Edit limit: 4

Column: course_num
  Updateable: Yes
  Key: Yes
  Format: "[general]"
  Border style: None
  Validation: None
  Validation Message: None
  Tab Sequence: 0
  Initial Value: None
  Edit Style: Edit
  Edit limit: 0

Column: section_num
  Updateable: Yes
  Key: Yes
  Format: "[general]"
  Border style: None
  Validation: None
  Validation Message: None
  Tab Sequence: 0
  Initial Value: None
  Edit Style: Edit
  Edit limit: 0

Column: name
Edit limit: 3

Column: max_credits
  Updateable: Yes
  Key: No
  Format: "[general]"
  Border style: None
  Validation: None
  Validation Message: None
  Tab Sequence: 0
  Initial Value: None
  Edit Style: Edit
  Edit limit: 0

Column: min_credits
  Updateable: Yes
  Key: No
  Format: "[general]"
  Border style: None
  Validation: None
  Validation Message: None
  Tab Sequence: 0
  Initial Value: None
  Edit Style: Edit
  Edit limit: 0

Column: building
  Updateable: Yes
DataWindow: d_class_info_schedule
Library: e:\thesis\appl\schedule.pbl
Date: 5/2/95    Time: 17:49:41

Format: "[general]"
Border style: None
Validation: None
Validation Message: None
Tab Sequence: 0
Initial Value: None
Edit Style: Edit
Edit limit: 10

Column: room
Updateable: Yes
Key: No
Format: "[general]"
Border style: None
Validation: None
Validation Message: None
Tab Sequence: 0
Initial Value: None
Edit Style: Edit
Edit limit: 4

Column: class_limit
Updateable: Yes
Key: No
Format: "[general]"
Border style: None
Validation: None
DataWindow: d_class_info_schedule
Library: e:\thesis\appl\schedule.pbl
Date: 5/2/95   Time: 17:49:41

Tab Sequence: 0
Initial Value: None
Edit Style: Edit
Edit limit: 0

Column: flags
  Updateable: Yes
  Key: No
  Format: "[general]"
  Border style: None
  Validation: None
  Validation Message: None
  Tab Sequence: 0
  Initial Value: None
  Edit Style: Edit
  Edit limit: 10

Column: lab
  Updateable: Yes
  Key: No
  Format: "[general]"
  Border style: None
  Validation: None
  Validation Message: None
  Tab Sequence: 0
  Initial Value: None
  Edit Style: Edit
Column: lab_days
  Updateable: Yes
  Key: No
  Format: "[general]"
  Border style: None
  Validation: None
  Validation message: None
  Tab Sequence: 0
  Initial Value: None
  Edit Style: Edit
  Edit limit: 6

Column: lab_start_time
  Updateable: Yes
  Key: No
  Format: "[time]"
  Border style: None
  Validation: None
  Validation message: None
  Tab Sequence: 0
  Initial Value: None
  Edit Style: Edit
  Edit limit: 0

Column: lab_end_time
  Updateable: Yes
  Key: No
DataWindow: d_class_into_schedule
Library: e:\thesis\appl\schedule.pbl
Date: 5/2/95   Time: 17:49:41

Border style: None
Validation: None
Validation Message: None
Tab Sequence: 0
Initial Value: None
  Edit Style: Edit
    Edit limit: 0
Column: lab_location
  Updateable: Yes
  Key: No
  Format: "[general]"
  Border style: None
  Validation: None
  Validation Message: None
  Tab Sequence: 0
  Initial Value: None
  Edit Style: Edit
    Edit limit: 3
Column: lab_max_credits
  Updateable: Yes
  Key: No
  Format: "[general]"
  Border style: None
  Validation: None
  Validation Message: None
DataWindow: d_class_info_schedule
Library: e:\thesis\appl\schedule.pbl
Date: 5/2/95   Time: 17:49:41

Updateable: Yes
Key: No
Format: "{general}"
Border style: None
Validation: None
Validation Message: None
Tab Sequence: 0
Initial Value: None
Edit Style: Edit
Edit limit: 4
DataWindow: d_prof_info_schedule  
Library: e:\thesis\appl\schedule.pbl  
Date: 5/2/95  Time: 17:21:31

<table>
<thead>
<tr>
<th>department</th>
<th>Name</th>
<th>activity</th>
<th>time Preference</th>
<th>Emp Type</th>
<th>class Preference</th>
<th>class Preference</th>
<th>class Preference</th>
<th>previous Class</th>
<th>previous Class</th>
</tr>
</thead>
<tbody>
<tr>
<td>Header</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Detail</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Summary</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Footer</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Class</td>
<td>previous_class</td>
<td>calculated_phil</td>
<td>calculated_pli</td>
<td>...</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>-------</td>
<td>----------------</td>
<td>----------------</td>
<td>----------------</td>
<td>-----</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Retrieve: PBSELECT(TABLE(NAME="professor_information") COLUMN(NAME="professor_information.department")
COLUMN(NAME="professor_information.name") COLUMN(NAME="professor_information.active")
COLUMN(NAME="professor_information.time_preference") COLUMN(NAME="professor_information.emp_type")
COLUMN(NAME="professor_information.class_preference_1")
COLUMN(NAME="professor_information.class_preference_2")
COLUMN(NAME="professor_information.class_preference_3")
COLUMN(NAME="professor_information.class_preference_4")
COLUMN(NAME="professor_information.previous_class_1")
COLUMN(NAME="professor_information.previous_class_2")
COLUMN(NAME="professor_information.previous_class_3")
COLUMN(NAME="professor_information.previous_class_4")
COLUMN(NAME="professor_information.calculated_priority")
COLUMN(NAME="professor_information.chair") WHERE( EXP1 =""professor_information.-"department-" OP
="" EXP2 =":arg_s_department" ) ARG(NAME ="arg_s_department" TYPE = string)

Arguments: arg_s_department

Update Table: professor_information

Filter: None
Sort: None
Sparse: None

Column: department
  Updateable: Yes
  Key: Yes
  Format: "[general]"
  Border style: None
  Validation: None
DataWindow: d_prot_into_schedule
Library: e:\thesis\appl\schedule.pbl
Date: 5/2/95    Time: 17:21:31

Tab Sequence: 0
Initial Value: None
Edit Style: Edit
Edit limit: 4
Column: active
  Updateable: Yes
  Key: No
  Format: "[general]"
  Border style: None
  Validation: None
  Validation Message: None
Tab Sequence: 0
Initial Value: None
Edit Style: Edit
Edit limit: 1

Column: time_preference
  Updateable: Yes
  Key: No
  Format: "[general]"
  Border style: None
  Validation: None
  Validation Message: None
Tab Sequence: 0
Initial Value: None
Edit Style: Edit
DataWindow: d_prof_into_schedule
Library: e:\thesis\appl\schedule.pbl
Date: 5/2/95  Time: 17:21:31

Column: class_preference_1
  Updateable: Yes
  Key: No
  Format: "{general}"
  Border style: None
  Validation: None
  Validation Message: None
  Tab Sequence: 0
  Initial Value: None
  Edit Style: Edit
  Edit limit: 0

Column: class_preference_2
  Updateable: Yes
  Key: No
  Format: "{general}"
  Border style: None
  Validation: None
  Validation Message: None
  Tab Sequence: 0
  Initial Value: None
  Edit Style: Edit
  Edit limit: 0

Column: class_preference_3
  Updateable: Yes
  Key: No
DataWindow: d_prof_info_schedule
Library: e:\thesis\appl\schedule.pbl
Date: 5/2/95   Time: 17:21:31

Border style: None
Validation: None
Validation Message: None
Tab Sequence: 0
Initial Value: None
Edit Style: Edit
Edit limit: 0

Column: class_preference_4
Updateable: Yes
Key: No
Format: \\"[general]"\
Border style: None
Validation: None
Validation Message: None
Tab Sequence: 0
Initial Value: None
Edit Style: Edit
Edit limit: 0

Column: previous_class_1
Updateable: Yes
Key: No
Format: \\"[general]"\
Border style: None
Validation: None
Validation Message: None
DataWindow: d_prot_info_schedule
Library: e:\thesis\appl\schedule.pbl
Date: 5/2/95   Time: 17:21:31

Initial Value: None
Edit Style: Edit
Edit limit: 0
Column: previous_class_2
   Updateable: Yes
   Key: No
   Format: "[general]"
   Border style: None
   Validation: None
   Validation Message: None
   Tab Sequence: 0
   Initial Value: None
   Edit Style: Edit
   Edit limit: 0
Column: previous_class_3
   Updateable: Yes
   Key: No
   Format: "[general]"
   Border style: None
   Validation: None
   Validation Message: None
   Tab Sequence: 0
   Initial Value: None
   Edit Style: Edit
   Edit limit: 0
DataWindow: d_prof_info_schedule
Library: e:\thesis\appl\schedule.pbl
Date: 5/2/95  Time: 17:21:31

Updateable: Yes
Key: No
Format: "[general]"
Border style: None
Validation: None
Validation Message: None
Tab Sequence: 0
Initial Value: None
Edit Style: Edit
Edit limit: 0

Column: calculated_priority
Updateable: Yes
Key: No
Format: "[general]"
Border style: None
Validation: None
Validation Message: None
Tab Sequence: 0
Initial Value: None
Edit Style: Edit
Edit limit: 0

Column: chair
Updateable: Yes
Key: No
Format: "[general]"
DataWindow: d_prot_into_schedule
Library: e:\thesis\appl\schedule.plb
Date: 5/2/95   Time: 17:21:31

Validation: None
Validation Message: None
Tab Sequence: 0
Initial Value: None
Edit Style: Edit
  Edit limit: 1

Column: name
  Updateable: Yes
  Key: Yes
  Format: "[general]"
  Border style: None
  Validation: None
  Validation Message: None
  Tab Sequence: 0
  Initial Value: None
  Edit Style: Edit
  Edit limit: 35

Column: emp_type
  Updateable: Yes
  Key: No
  Format: "[general]"
  Border style: None
  Validation: None
  Validation Message: None
  Tab Sequence: 0
DataWindow: d_extra_classes
Library: e:\thesis\appl\schedule.pbl
Date: 5/2/95 Time: 17:41:56

Retrieve: Script
Arguments: None
Update Table: Not Allowed
Filter: None
Sort: None
Sparse: None

Column: course_num
 Format: "$[general]"
 Border style: None
 Validation: None
 Validation Message: None
 Tab Sequence: 10
 Initial Value: None
 Edit Style: Edit
 Edit limit: 0

Column: section_num
 Format: "$[general]"
 Border style: None
 Validation: None
 Validation Message: None
 Tab Sequence: 20
 Initial Value: None
 Edit Style: Edit
 Edit limit: 0
<table>
<thead>
<tr>
<th>Name</th>
<th>Priority</th>
<th>Needed Credit</th>
<th>current Preference</th>
<th>current Prerequisite</th>
<th>radi</th>
<th>course</th>
<th>action</th>
<th>course</th>
<th>action</th>
<th>course</th>
<th>action</th>
<th>course</th>
<th>action</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**Summary**

**Footer**
DataWindow: d_assign_prof_long
Library: e:\thesis\appl\schedule.pbl
Date: 5/2/95   Time: 17:57:47

Format: "[general]"
Border style: None
Validation: None
Validation Message: None
Tab Sequence: 40
   Initial Value: None
   Edit Style: Edit
   Edit limit: 0
Column: course_1
   Format: "[general]"
   Border style: None
   Validation: None
   Validation Message: None
   Tab Sequence: 50
   Initial Value: None
   Edit Style: Edit
   Edit limit: 0
Column: section_1
   Format: "[general]"
   Border style: None
   Validation: None
   Validation Message: None
   Tab Sequence: 60
   Initial Value: None
   Edit Style: Edit
Column: course_2
  Format: "[general]"
  Border style: None
  Validation: None
  Validation Message: None
  Tab Sequence: 70
  Initial Value: None
  Edit Style: Edit
  Edit limit: 0

Column: section_2
  Format: "[general]"
  Border style: None
  Validation: None
  Validation Message: None
  Tab Sequence: 80
  Initial Value: None
  Edit Style: Edit
  Edit limit: 0

Column: course_3
  Format: "[general]"
  Border style: None
  Validation: None
  Validation Message: None
  Tab Sequence: 90
  Initial Value: None
DataWindow: d_assign_prof_long
Library: e:\thesis\appl\schedule.pbl
Date: 5/2/95    Time: 17:57:47

Edit Style: Edit
Edit limit: 0

Column: course_5
Format: "[general]"
Border style: None
Validation: None
Validation Message: None
Tab Sequence: 130
Initial Value: None
Edit Style: Edit
Edit limit: 0

Column: section_5
Format: "[general]"
Border style: None
Validation: None
Validation Message: None
Tab Sequence: 140
Initial Value: None
Edit Style: Edit
Edit limit: 0

Column: course_6
Format: "[general]"
Border style: None
Validation: None
Validation Message: None
DataWindow: d_assign_prot_long
Library: e:\thesis\appl\schedule.pbl
Date: 5/2/95   Time: 17:57:47

Tab Sequence: 180
Initial Value: None
Edit Style: Edit
Edit limit: 0

Column: course_8
  Format: "[general]"
  Border style: None
  Validation: None
  Validation Message: None
  Tab Sequence: 190
  Initial Value: None
  Edit Style: Edit
  Edit limit: 0

Column: section_8
  Format: "[general]"
  Border style: None
  Validation: None
  Validation Message: None
  Tab Sequence: 200
  Initial Value: None
  Edit Style: Edit
  Edit limit: 0
Function : add_class_lab

// Determines if the faculty member is available for a given class and lab(s).
// If the faculty member is available during those times the faculty member is
// assigned the course and the associated lab(s)

// Local Variables
Boolean lv_b_AddClass = TRUE
Boolean lv_b_AddLab1 = TRUE
Boolean lv_b_AddLab2 = TRUE
Character lv_ca_TempDay[7]
Integer lv_i_Day1 = 1
Integer lv_i_Day2 = 2
Integer lv_i_Day3 = 4
Integer lv_i_Day4 = 8
Integer lv_i_Day5 = 16
Integer lv_i_Day6 = 32
Integer lv_i_ClassPeriod
Integer lv_i_NumClassPeriods
Integer lv_i_LabPeriod1
Integer lv_i_NumLabPeriods1
Integer lv_i_LabPeriod2
Integer lv_i_NumLabPeriods2
Integer lv_i_Loop
Integer lv_i_Day
Integer lv_i_BitsSet[6]
Integer lv_i_ClassBits[6,6]
Integer lv_i_Temp
Integer lv_i_Count
Integer lv_i_CurrentValue
// End Local Variables

lv_ca_TempDay = fv_s_ClassDays

IF (fv_t_classsstart = Time(8,0,0)) THEN
lv_i_ClassPeriod = 1
ELSEIF (fv_t_classsstart = Time(9,40,0)) THEN
lv_i_ClassPeriod = 2
ELSEIF (fv_t_classsstart = Time(11,20,0)) THEN
lv_i_ClassPeriod = 3

ELSEIF (fv_t_classstart = Time(14,40,0)) THEN
  lv_i_ClassPeriod = 5
END IF

lv_i_NumClassPeriods = num_periods(fv_t_classstart, fv_t_classend)

FOR lv_i_Count = 1 to lv_i_NumClassPeriods
  lv_i_BitsSet[1] = 0
  lv_i_BitsSet[2] = 0
  lv_i_BitsSet[3] = 0
  lv_i_BitsSet[4] = 0
  lv_i_BitsSet[5] = 0
  lv_i_BitsSet[6] = 0

  lv_i_TEMP = fv_ia_pc[fv_i_prof, lv_i_ClassPeriod + lv_i_Count - 1]

  IF (lv_i_TEMP - lv_i_Day6 >= 0) THEN
    lv_i_BitsSet[6] = lv_i_Day6
    lv_i_TEMP = lv_i_TEMP - lv_i_Day6
  END IF

  IF (lv_i_TEMP - lv_i_Day5 >= 0) THEN
    lv_i_BitsSet[5] = lv_i_Day5
    lv_i_TEMP = lv_i_TEMP - lv_i_Day5
  END IF

  IF (lv_i_TEMP - lv_i_Day4 >= 0) THEN
    lv_i_BitsSet[4] = lv_i_Day4
    lv_i_TEMP = lv_i_TEMP - lv_i_Day4
  END IF

  IF (lv_i_TEMP - lv_i_Day3 >= 0) THEN
    lv_i_BitsSet[3] = lv_i_Day3
    lv_i_TEMP = lv_i_TEMP - lv_i_Day3
  END IF

  IF (lv_i_TEMP - lv_i_Day2 >= 0) THEN
    lv_i_BitsSet[2] = lv_i_Day2
    lv_i_TEMP = lv_i_TEMP - lv_i_Day2
  END IF

  IF (lv_i_TEMP - lv_i_Day1 >= 0) THEN
    lv_i_BitsSet[1] = lv_i_Day1
    lv_i_TEMP = lv_i_TEMP - lv_i_Day1
  END IF
DO WHILE (lv_ca_TempDay[lv_i_Loop] <> char(0))
    lv_i_Day = Integer(lv_ca_TempDay[lv_i_Loop])

    lv_i_ClassBits[lv_i_Count, lv_i_Day] = 2^(lv_i_Day - 1)

    lv_i_Loop = lv_i_Loop + 1
LOOP

FOR lv_i_Loop = 1 TO 6
    IF (lv_i_ClassBits[lv_i_Count, lv_i_Loop] > 0) THEN
        IF (lv_i_BitsSet[lv_i_Loop] > 0) THEN
            lv_b_AddClass = FALSE
        END IF
    END IF
NEXT

IF (lv_b_AddClass AND (fv_t_labstart1 <> Time(0, 0, 0))) THEN

    lv_ca_TempDay = fv_s_labdays1

    IF (fv_t_labstart1 = Time(8, 0, 0)) THEN
        lv_i_LabPeriod1 = 1
    ELSEIF (fv_t_labstart1 = Time(9, 0, 0)) THEN
        lv_i_LabPeriod1 = 2
    ELSEIF (fv_t_labstart1 = Time(10, 0, 0)) THEN
        lv_i_LabPeriod1 = 3
    ELSEIF (fv_t_labstart1 = Time(11, 0, 0)) THEN
        lv_i_LabPeriod1 = 4
    ELSEIF (fv_t_labstart1 = Time(12, 0, 0)) THEN
        lv_i_LabPeriod1 = 5
    END IF

    lv_i_NumLabPeriods1 = num_periods(fv_t_labstart1, fv_t_labend1)

FOR lv_i_Count = 1 TO lv_i_NumLabPeriods1
    lv_i_BitsSet[1] = 0
    lv_i_BitsSet[2] = 0
    lv_i_BitsSet[3] = 0
    lv_i_BitsSet[4] = 0
    lv_i_BitsSet[5] = 0
NEXT
lv_i_Temp = fv_ia_pc[fv_i_prof, lv_i_LabPeriod1 + lv_i_Count - 1]

IF (lv_i_Temp - lv_i_Day6 >= 0) THEN
  lv_i_BitsSet[6] = lv_i_Day6
  lv_i_Temp = lv_i_Temp - lv_i_Day6
END IF

IF (lv_i_Temp - lv_i_Day5 >= 0) THEN
  lv_i_BitsSet[5] = lv_i_Day5
  lv_i_Temp = lv_i_Temp - lv_i_Day5
END IF

IF (lv_i_Temp - lv_i_Day4 >= 0) THEN
  lv_i_BitsSet[4] = lv_i_Day4
  lv_i_Temp = lv_i_Temp - lv_i_Day4
END IF

IF (lv_i_Temp - lv_i_Day3 >= 0) THEN
  lv_i_BitsSet[3] = lv_i_Day3
  lv_i_Temp = lv_i_Temp - lv_i_Day3
END IF

IF (lv_i_Temp - lv_i_Day2 >= 0) THEN
  lv_i_BitsSet[2] = lv_i_Day2
  lv_i_Temp = lv_i_Temp - lv_i_Day2
END IF

IF (lv_i_Temp - lv_i_Day1 >= 0) THEN
  lv_i_BitsSet[1] = lv_i_Day1
  lv_i_Temp = lv_i_Temp - lv_i_Day1
END IF

lv_i_Loop = 1
DO WHILE (lv_ca_TempDay[lv_i_Loop] <> char(0))
  lv_i_Day = Integer(lv_ca_TempDay[lv_i_Loop])
  lv_i_CLASSBits[lv_i_Count + lv_i_NumClassPeriods, lv_i_Day] = 2^(lv_i_Day - 1)
  lv_i_Loop = lv_i_Loop + 1
LOOP

FOR lv_i_Loop = 1 to 6
  IF (lv_i_CLASSBits[lv_i_Count + lv_i_NumClassPeriods, lv_i_Loop] > 0) THEN
    IF (lv_i_BitsSet[lv_i_Loop] > 0) THEN
      ...
IF (lv_b_AddClass AND lv_b_AddLab1 AND (fv_t_labstart2 <> Time(0, 0, 0))) THEN

lv_ca_TempDay = fv_s_labdays2

IF (fv_t_labstart2 = Time(8, 0, 0)) THEN
  lv_i_LabPeriod2 = 1
ELSEIF (fv_t_labstart2 = Time(9, 40, 0)) THEN
  lv_i_LabPeriod2 = 2
ELSEIF (fv_t_labstart2 = Time(11, 20, 0)) THEN
  lv_i_LabPeriod2 = 3
ELSEIF (fv_t_labstart2 = Time(13, 0, 0)) THEN
  lv_i_LabPeriod2 = 4
ELSEIF (fv_t_labstart2 = Time(14, 40, 0)) THEN
  lv_i_LabPeriod2 = 5
END IF

lv_i_NumLabPeriods2 = num_periods(fv_t_labstart2, fv_t_labend2)

FOR lv_i_Count = 1 to lv_i_NumLabPeriods2
  lv_i_BitsSet[1] = 0
  lv_i_BitsSet[2] = 0
  lv_i_BitsSet[3] = 0
  lv_i_BitsSet[4] = 0
  lv_i_BitsSet[5] = 0
  lv_i_BitsSet[6] = 0

  lv_i_Temp = fv ia_pc[fv_i_prof, lv_i_LabPeriod2 + lv_i_Count - 1]

  IF (lv_i_Temp = lv_i_Day6 = 0) THEN
    lv_i_BitsSet[6] = lv_i_Day6
    lv_i_Temp = lv_i_Temp - lv_i_Day6
  END IF

  IF (lv_i_Temp = lv_i_Day5 = 0) THEN
    lv_i_BitsSet[5] = lv_i_Day5
    lv_i_Temp = lv_i_Temp - lv_i_Day5
  END IF

END FOR
lv_i_BitsSet[4] = lv_i_Day4
lv_i_Temp = lv_i_Temp - lv_i_Day4
END IF
IF (lv_i_Temp - lv_i_Day3 >= 0) THEN
lv_i_BitsSet[3] = lv_i_Day3
lv_i_Temp = lv_i_Temp - lv_i_Day3
END IF
IF (lv_i_Temp - lv_i_Day2 >= 0) THEN
lv_i_BitsSet[2] = lv_i_Day2
lv_i_Temp = lv_i_Temp - lv_i_Day2
END IF
IF (lv_i_Temp - lv_i_Day1 >= 0) THEN
lv_i_BitsSet[1] = lv_i_Day1
lv_i_Temp = lv_i_Temp - lv_i_Day1
END IF

lv_i_Loop = 1
DO WHILE (lv_ca_TempDay[lv_i_Loop] <> char(0))
lv_i_Day = Integer(lv_ca_TempDay[lv_i_Loop])

lv_i_ClassBits[lv_i_Count + lv_i_NumClassPeriods + lv_i_NumLabPeriods, lv_i_Day] = 2^(lv_i_Day

lv_i_Loop = lv_i_Loop + 1
LOOP

FOR lv_i_Loop = 1 to 6
 IF (lv_i_BitsSet[lv_i_Count + lv_i_NumClassPeriods + lv_i_NumLabPeriods, lv_i_Loop] > 0) THEN
   IF (lv_i_BitsSet[lv_i_Loop] > 0) THEN
     lv_b_AddLab2 = FALSE
   END IF
 END IF
NEXT

NEXT
END IF

IF (lv_b_AddClass AND lv_b_AddLab1 AND lv_b_AddLab2) THEN
FOR lv_i_Count = 1 TO lv_i_NumClassPeriods
lv_i_CurrentValue = lv_i_CurrentValue + lv_i_ClassBits[lv_i_Count, lv_i_Loop]
END FOR
ELSE
FOR lv_i_Count = 1 TO lv_i_NumClassPeriods
lv_i_CurrentValue = lv_i_CurrentValue + lv_i_ClassBits[lv_i_Count, lv_i_Loop]
END FOR
ELSE
FOR lv_i_Count = 1 TO lv_i_NumClassPeriods
lv_i_CurrentValue = lv_i_CurrentValue + lv_i_ClassBits[lv_i_Count, lv_i_Loop]
END FOR
ELSE
FOR lv_i_Count = 1 TO lv_i_NumClassPeriods
lv_i_CurrentValue = lv_i_CurrentValue + lv_i_ClassBits[lv_i_Count, lv_i_Loop]
END FOR
ELSE
FOR lv_i_Count = 1 TO lv_i_NumClassPeriods
lv_i_CurrentValue = lv_i_CurrentValue + lv_i_ClassBits[lv_i_Count, lv_i_Loop]
END FOR
END IF
NEXT

IF (fv_t_labstart1 <> Time(0, 0, 0)) THEN
  FOR lv_i_Count = 1 TO lv_i_NumLabPeriods1
    lv_i_CurrentValue = fv ia pc[lv_i_prof, lv_i_LabPeriod1 + lv_i_Count - 1]
    FOR lv_i_Loop = 1 to 6
      lv_i_CurrentValue = lv_i_CurrentValue + lv_i_ClassBits[lv_i_Count + lv_i_NumClassPeriods, lv_i_Loop]
    NEXT
    fv ia pc[lv_i_prof, lv_i_LabPeriod1 + lv_i_Count - 1] = lv_i_CurrentValue
  NEXT
END IF

IF (fv_t_labstart2 <> Time(0, 0, 0)) THEN
  FOR lv_i_Count = 1 TO lv_i_NumLabPeriods2
    lv_i_CurrentValue = fv ia pc[lv_i_prof, lv_i_LabPeriod2 + lv_i_Count - 1]
    FOR lv_i_Loop = 1 to 6
      lv_i_CurrentValue = lv_i_CurrentValue + lv_i_ClassBits[lv_i_Count + lv_i_NumClassPeriods + lv_i_Loop]
    NEXT
    fv ia pc[lv_i_prof, lv_i_LabPeriod2 + lv_i_Count - 1] = lv_i_CurrentValue
  NEXT
END IF
END IF

Return (lv_b_AddClass AND lv_b_AddLab1 AND lv_b_AddLab2)
Function: delete_class_lab

// If a new faculty member is assigned a class that was originally assigned to //a faculty member, then the class and labs must be deleted from the original //faculty member.

// Local Variables
Character   lv_ca_TempDay[7]
Integer     lv_i_ClassPeriod
Integer     lv_i_NumClassPeriods
Integer     lv_i_LabPeriod1
Integer     lv_i_NumLabPeriods1
Integer     lv_i_LabPeriod2
Integer     lv_i_NumLabPeriods2
Integer     lv_i_Loop
Integer     lv_i_Day
Integer     lv_i_ClassBits[6]
Integer     lv_i_Count
Integer     lv_i_CurrentValue
// End Local Variables

lv_ca_TempDay = fv_s_ClassDays

IF (fv_t_classstart = Time(8,0,0)) THEN
  lv_i_ClassPeriod = 1
ELSEIF (fv_t_classstart = Time(9,40,0)) THEN
  lv_i_ClassPeriod = 2
ELSEIF (fv_t_classstart = Time(11,20,0)) THEN
  lv_i_ClassPeriod = 3
ELSEIF (fv_t_classstart = Time(13,0,0)) THEN
  lv_i_ClassPeriod = 4
ELSEIF (fv_t_classstart = Time(14,40,0)) THEN
  lv_i_ClassPeriod = 5
END IF

lv_i_NumClassPeriods = num_periods(fv_t_classstart, fv_t_classsend)

FOR lv_i_Count = 1 to lv_i_NumClassPeriods
  lv_i_Loop = 1
  DO WHILE (lv_ca_TempDay[lv_i_Loop] <> char(0))

lv_i_ClassBits[lv_i_Day] = 2**(lv_i_Day - 1)

lv_i_Loop = lv_i_Loop + 1
LOOP

lv_i_CurrentValue = fvi_pc[fv_i_prof, lv_i_ClassPeriod + lv_i_Count - 1]
FOR lv_i_Loop = 1 to 6
    lv_i_CurrentValue = lv_i_CurrentValue - lv_i_ClassBits[lv_i_Loop]
NEXT
fvi_pc[fv_i_prof, lv_i_ClassPeriod + lv_i_Count - 1] = lv_i_CurrentValue
NEXT

IF (fv_t_labstart1 <> Time(0, 0, 0)) THEN

lv_ca_TempDay = fvi_s_labdays1

IF (fv_t_labstart1 = Time(8,0,0)) THEN
    lv_i_LabPeriod1 = 1
ELSEIF (fv_t_labstart1 = Time(9,40,0)) THEN
    lv_i_LabPeriod1 = 2
ELSEIF (fv_t_labstart1 = Time(11,20,0)) THEN
    lv_i_LabPeriod1 = 3
ELSEIF (fv_t_labstart1 = Time(13,0,0)) THEN
    lv_i_LabPeriod1 = 4
ELSEIF (fv_t_labstart1 = Time(14,40,0)) THEN
    lv_i_LabPeriod1 = 5
END IF

lv_i_NumLabPeriods1 = num_periods(fv_t_labstart1, fv_t_labend1)

FOR lv_i_Count = 1 to lv_i_NumLabPeriods1
    lv_i_Loop = 1
    DO WHILE (lv_ca_TempDay[lv_i_Loop] <> char(0))
        lv_i_Day = Integer(lv_ca_TempDay[lv_i_Loop])
        lv_i_ClassBits[lv_i_Day] = 2**(lv_i_Day - 1)
    lv_i_Loop = lv_i_Loop + 1
    LOOP
    lv_i_CurrentValue = fvi_pc[fv_i_prof, lv_i_LabPeriod1 + lv_i_Count - 1]

NEXT
  lv_i_CurrentValue = fv_i_pc[fv_i_prof, lv_i_LabPeriod1 + lv_i_Count - 1] = lv_i_CurrentValue
NEXT
END IF

IF (fv_t_labstart2 <> Time(0, 0, 0)) THEN
  lv_ca_TempDay = fv_s_labdays2

  lv_i_NumLabPeriods2 = num_periods(fv_t_labstart2, fv_t_labend2)

  FOR lv_i_Count = 1 to lv_i_NumLabPeriods2
    lv_i_Loop = 1
    DO WHILE (lv_ca_TempDay[lv_i_Loop] <> char(0))
      lv_i_Day = Integer(lv_ca_TempDay[lv_i_Loop])
      lv_i_ClassBits[lv_i_Day] = 2^(lv_i_Day - 1)
    END DO
    lv_i_Loop = lv_i_Loop + 1
  LOOP
  lv_i_CurrentValue = fv_i_pc[fv_i_prof, lv_i_LabPeriod2 + lv_i_Count - 1]
  FOR lv_i_Loop = 1 to 6
    lv_i_CurrentValue = lv_i_CurrentValue - lv_i_ClassBits[lv_i_Loop]
  NEXT
  fv_i_pc[fv_i_prof, lv_i_LabPeriod2 + lv_i_Count - 1] = lv_i_CurrentValue
NEXT
END IF

Return (TRUE)
Function: num_periods

// Determines the number of class periods that a class or lab covers.

// Local Variables
Integer lv_i_NumPeriods
// End Local Variables

IF (start_time = Time(8,0,0)) THEN
  IF (end_time = Time(9,10,0)) THEN
    lv_i_NumPeriods = 1
  ELSEIF (end_time = Time(10,50,0)) THEN
    lv_i_NumPeriods = 2
  ELSEIF (end_time = Time(12,30,0)) THEN
    lv_i_NumPeriods = 3
  ELSEIF (end_time = Time(14,10,0)) THEN
    lv_i_NumPeriods = 4
  ELSEIF (end_time = Time(15,50,0)) THEN
    lv_i_NumPeriods = 5
  END IF
ELSEIF (start_time = Time(9,40,0)) THEN
  IF (end_time = Time(10,50,0)) THEN
    lv_i_NumPeriods = 1
  ELSEIF (end_time = Time(12,30,0)) THEN
    lv_i_NumPeriods = 2
  ELSEIF (end_time = Time(14,10,0)) THEN
    lv_i_NumPeriods = 3
  ELSEIF (end_time = Time(15,50,0)) THEN
    lv_i_NumPeriods = 4
  END IF
ELSEIF (start_time = Time(11,20,0)) THEN
  IF (end_time = Time(12,30,0)) THEN
    lv_i_NumPeriods = 1
  ELSEIF (end_time = Time(14,10,0)) THEN
    lv_i_NumPeriods = 2
  ELSEIF (end_time = Time(15,50,0)) THEN
    lv_i_NumPeriods = 3
  END IF
ELSEIF (start_time = Time(13,0,0)) THEN
  IF (end_time = Time(14,10,0)) THEN


lv_i_NumPeriods = 2
END IF
ELSEIF (start_time = Time(14,40,0)) THEN
  IF (end_time = Time(15,50,0)) THEN
    lv_i_NumPeriods = 1
  END IF
ELSE
  lv_i_NumPeriods = 0
END IF

Return(lv_i_NumPeriods)