A Student scheduling system for a microcomputer

Michael Mahaney
A Student Scheduling System for a Microcomputer

by

Michael C. Mahaney Jr.

A thesis, submitted to
The Faculty of the School of Computer Science and Technology,
in partial fulfillment of the requirements for the degree of
Master of Science in Computer Science.

Approved by:  
__________________________  
Henry A. Etlinger

__________________________  
Henry A. Etlinger (Advisor)

__________________________  
Chris Comte

__________________________  
Chris Comte

__________________________  
Andrew Kitchen

__________________________  
Andrew Kitchen

May 1, 1985
Title of Thesis: A Student Scheduling System for a Microcomputer

I Michael Mahaney hereby grant permission to the Wallace Memorial Library, of RIT, to reproduce my thesis in whole or in part. Any reproduction will not be for commercial use or profit.

Date: May 20, 1985
ACKNOWLEDGMENTS

I would like to express gratitude to the members of my thesis committee, Chris Comte and Andrew Kitchen for their suggestions and criticisms. I am especially indebted to Hank Etlinger, chairperson of my committee, for help in reviewing and revising this paper.

Most of all I would like to thank my family, who provided support and understanding during the past four years.
1. PRELIMINARY INFORMATION

1.2. ABSTRACT

A STUDENT SCHEDULING SYSTEM FOR A MICROCOMPUTER

This student scheduling system was written for use on a microcomputer to give the local high school more control over the scheduling process. Scheduling systems are used by high schools to schedule their students, balance classes, and print student schedules and class lists. A scheduling system must be flexible to provide for the generation of many different types of schedules. Most scheduling programs currently used are shared by high schools through the use of BOCES (Board of Cooperative Educational Services). Schools send their schedules to the BOCES regional computer centers to be run on mainframe computers.

This scheduling system was written using Apple Pascal for use on an Apple IIE microcomputer. The system was developed to do student scheduling for a high school of less than 1200 students. The system will section students and provide the school with student schedules, class lists as well as many scheduling tools which are useful in the development of the schedule.
1.3. KEY WORDS AND PHRASES

Balancing (Class size) - Making the number of students in each section (or class) of a course the same, or as nearly so, as possible.

Block Scheduling - Placement of students with common subject selections into a group. The group then could be assigned to a team of teachers for large segments of instructional time.

Code Number - A three-digit number identifying each course offered in a school.

Computer Scheduling - A general term describing the use of a computer to assist in student scheduling. The range of assistance extends from the preparation of simple reports or printouts to the complete processing of both the master schedule and student schedules.

Conflict - A condition in which two or more courses selected by a student are offered at the same time.

Conflict Matrix - A list of all courses that would conflict with a given course. The number of students who would have this conflict is also listed.

Conflict Rate - The percentage of students who have one or more conflicts.

Conventional Schedule - A traditional scheduling method based on a weekly cycle of class periods of equal length, usually meeting the same time each day.

Course Tallies - Enrollment figures for each course offered in the curriculum.

Doubleton - A course which has sufficient enrollment to require two sections.

Flexible Modular Schedule - A type of schedule which divides the school day into 15 to 30 abbreviated periods (10 to 20 minutes each) in order to provide for a greater variety of instructional experiences.

Manual Scheduling - The completion of the scheduling process by hand, without computerized data processing assistance.
Master Schedule - A comprehensive listing of the class offerings that assigns every section of every course to specific teachers in designated rooms at various times of the day.

Mosaic Scheduling - A loading technique used to schedule student courses individually rather than in block fashion.

Scheduling - The process by which the curriculum, the students, and the teachers are brought together in an organized manner.

Section - A class of students of a predetermined number range that study a particular course together in the same period.

Simulation Run - A computerized testing of the efficiency of the master schedule in which problems or possible solutions are identified.

Singleton - A course which has only one section.

Student Requests - A list of courses that a student would like to schedule.

Turnaround Time - The elapsed time between the school submitting information to the computer center and the delivery of the resulting computer printout to the school.

Tripleton - A course that has sufficient enrollment to require three distinct sections.

1.4. COMPUTING REVIEW SUBJECT CODE

Not applicable.
1.5. TABLE OF CONTENTS

1. Preliminary Information
   1.1. Title and Acceptance Page ........................................1
   1.2. Abstract ........................................................................2
   1.3. Key Words and Phrases ......................................................3
   1.4. Computing Review Subject Code .......................................4
   1.5. Table of Contents ..............................................................5

2. Proposal ..............................................................................8

3. Introduction and Background
   3.1. Problem Statement ............................................................8
   3.2. Previous Work
      3.2.1. Types of Schedules ....................................................12
      3.2.2. Conflicts in Scheduling ..............................................13
      3.2.3. Manual vs. Computer Scheduling .................................15
      3.2.4. Research Findings .....................................................18
   3.3. Theoretical and Conceptual Development
      3.3.1. Phase 1 - Schedule Development .................................22
      3.3.2. Phase 2 - Section Students .......................................23
      3.3.3. Phase 3 - Final Output .............................................25
      3.3.4. Scheduling Calendar ..................................................26

4. Functional Specification
   4.1. Functions Performed .........................................................28
   4.2. System Data Flow Chart ..................................................30
   4.3. Limitations and Restrictions .............................................36
   4.4. User Inputs .......................................................................36
   4.5. User Outputs .....................................................................37
   4.6. System Files .....................................................................39
5. System Specification
   5.1. System Organizational Chart .................................. 40
   5.2. Equipment Configuration ...................................... 48
   5.3. Implementation Tools ........................................ 48
   5.4. Overview of Programming System ............................... 48

6. Verification and Validation
   6.1. Test Plan ..................................................... 56
   6.2. Test Procedures ............................................. 56
   6.3. Test Results ................................................ 57

7. Conclusions
   7.1. Problems Encountered and Solved ............................. 59
   7.2. Discrepancies and Shortcomings of System .................. 61
   7.3. Lessons Learned .............................................. 61
       7.3.1. Alternative Approaches for Improved System ...... 61
       7.3.2. Suggestions for Future Extensions .................... 62
   7.4. Future Directions in Scheduling ............................ 63

8. Bibliography .................................................................. 65

9. Appendices
   9.1. List of Student Records ....................................... 67
       9.1.1. General Information ....................................... 67
       9.1.2. Student Course Request List ............................ 68
   9.2. Conflict Matrix ................................................ 69
   9.3. Course Tallies List .......................................... 70
   9.4. Course List ..................................................... 71
   9.5. Results - Trial and Final Run ............................... 72
   9.6. Student Schedules ............................................ 73
   9.7. Course Class Lists ........................................... 74
10. Program Listings

10.1. Entstuddata .................................................. 75
10.2. Unit One .................................................... 76
10.3. Cmatrix ....................................................... 77
10.4. Tallylist ...................................................... 78
10.5. Entcoursesdata .............................................. 79
10.6. Scheduler .................................................... 80
10.7. Unit Two ..................................................... 81
10.8. Printschedules .............................................. 82
10.9. Printclasslists ............................................. 83
2. PROPOSAL

A copy of the thesis proposal is on file.

3. INTRODUCTION AND BACKGROUND

3.1 PROBLEM STATEMENT

Every high school faces the problem each year of scheduling their students and teachers. The most widely used scheduling format is one in which the computer generates data from which the master schedule is built. The master schedule is a listing that assigns every section of every course to a specific teacher at a designated time of the day. The data that the computer generates includes a course tallies list and the conflict matrix. The course tallies list is a list of enrollment figures for each course in the curriculum. The conflict matrix is a list of all courses that would conflict with a given course. The number of students who would have each conflict are also listed.

After the schedule is built the computer tests the efficiency of the master schedule through a simulation run in which the students are loaded into the schedule. The computer then prints partial schedules, conflicts and course tallies so that the scheduler can make adjustments in the schedule. A partial schedule is a student schedule for all courses which where successfully scheduled along with a list of unscheduled courses. Finally student and teacher schedules, class lists and other helpful reports are printed.
The computer usually is a mainframe located at BOCES (Board of Cooperative Educational Services) and all the schools in a given region would use the same scheduling services. This form of centralization causes some problems for local schools.

Problems encountered by schools include:

1.) BOCES allows a school only three runs of a schedule with a two to three week turnaround time for each run. As a result the scheduling process starts in May and the final run is usually in early August.

2.) After the first run, if the scheduler finds a problem with a course placement he/she corrects it. The scheduler must then wait for two to three weeks to find out if his/her solution works. The scheduler only has two chances or runs to resolve all the problems in a schedule.

3.) Any student who enters after the final run (first week in August) has to be hand scheduled and all class lists must be updated.

4.) Any student who changes a course during the first week in August or the first few weeks of school has an incorrect schedule, resulting in class lists which are often inaccurate.

SOLUTION

This thesis will solve the problems listed above by developing a student scheduling system for a microcomputer. Since most schools have a microcomputer, the control of the scheduling process could be held in the local school.
Difficulties previously encountered that could be solved by this system are:

1.) Eliminating the two to three week turnaround time between runs of the schedule.
2.) The scheduler would not be limited to only three runs which would greatly increase his/her efficiency and accuracy.
3.) The first run could be made before the end of school in June so guidance counselors would have time to contact students regarding conflicts in the students' schedules.
4.) The final run of schedules could be made during the week before school begins to ensure more accurate student schedules and class lists.

THE STUDENT SCHEDULING SYSTEM

This student scheduling system will allow the school to enter all required data using a microcomputer. The computer will then be able to generate the information necessary to build the schedule. Finally the schedule and final reports will be printed.

1.) Data entered by the user.
   A.) Student information
      1.) Name, grade, address.
      2.) Courses requested.
   B.) Schedule information
      1.) Courses offered.
      2.) Time and room information.
C. Teacher information
   1.) Courses and sections offered.
   2.) Room or rooms used.

2.) Data generated by the system
A.) Information used to help build the master schedule.
   1.) A conflict matrix which is used to determine how many students are taking a given combination of courses.
   2.) A list of the number of students registered for each course by grade levels.
B.) After scheduling the students, the system will provide information used to help resolve conflicts.
   1.) A list of students unable to schedule all requested courses.
   2.) A partial schedule for each student whose schedule contained a conflict, listing any course in which a student was unable to enroll.
   3.) A listing of all classes and the number of students enrolled in each. This list would be used to consider balance problems.
C.) Changes can then be made to both student course selections and course offerings. The schedule can then be rerun until the results are acceptable.
D.) After arriving at an acceptable schedule the system will print the following information:
   1.) Individual student schedules for every student.
   2.) Class lists.
   3.) Master list of classes with enrollments.
3.2 PREVIOUS WORK

3.2.1 TYPES OF SCHEDULES

The schedule is an essential component of the school program. Scheduling combines all of a school’s essential resources - faculty and staff, curriculum, space and facilities, and students - into an integrated and efficient learning environment. A successful master schedule holds curricular objectives, student course requests, and faculty strengths and preferences in appropriate balance. Some of the different types of schedules used by schools include block schedules, mosaic schedules, a combination of block and mosaic schedules, and modular or flexible modular schedules.

BLOCK SCHEDULES

Students can be scheduled in groups or individually. Block scheduling is often used when students are taking a similar set of courses. When a block schedule is used students are scheduled in groups. Block scheduling is often preferred in junior high schools in conjunction with team teaching. All students might be taking English, math, social studies, science and reading courses. Students would first be divided into groups and then the students in each group would follow the same schedule.

MOSAIC SCHEDULING

At times the block structure method is impractical, such as when large groups of students do not take similar subjects. In
high schools where required courses are offered along with a wide range of electives, a mosaic method is used. Mosaic scheduling methods are concerned with individual sections of courses. In this method the scheduler places one course section at a time into the master schedule, continually building upon the previous sections until the total mosaic is completed.

**FLEXIBLE MODULAR SCHEDULING**

A flexible modular schedule divides the school day into fifteen to thirty abbreviated periods which range from ten to twenty minutes each. These schedules attempt to provide a greater variety of instructional experience by the use of varying time blocks. Modular scheduling permits many alternatives that were impossible under traditional scheduling. To provide for all the individual differences required in the flexible modular schedule, computer generation of the schedule is required.

3.2.2 CONFLICTS IN SCHEDULING

No matter which type of schedule is used, conflicts can arise. Conflicts occur as a result of problems scheduling rooms, programs, and courses. The most common type of conflict arises when two of a student's preferred courses are scheduled in the same period. If a single section course (singleton) is placed in a particular period along with another singleton, the student will be unable to schedule one of them.

Conflicts do not arise so much from individual student preferences as from the difficulty of constructing a schedule that will accommodate the seemingly infinite combination of
different courses. The scheduler's goal for each scheduling year is the building of an efficient schedule that is as conflict-free as possible. This would result in only a small number of students needing to change their course requests.

In an attempt to limit potential conflicts the scheduler uses a conflict matrix (an array of all courses in the curriculum paired against each other). The matrix would usually include singletons, doubletons and some tripletons. The scheduler would read the matrix to anticipate potential conflicts. Although different implications arise when comparing two-section courses (doubletons) as opposed to singletons, the matrix maintains a high degree of usefulness.

According to Richard Dempsey and Henry Traverso¹ a number of elements affect the scheduler's ability to construct a schedule with a minimum of conflicts. Some of these are:

1) High density within the schedule (e.g. six required courses in a six-period schedule).
2) An excessive number of singletons.
3) Non-graded courses (e.g. those open to students in several grade levels).
4) A large number of restrictive factors (e.g. part-time faculty).
5) Constraints in the teacher contract (e.g. length of teachers' day).
6) Many teachers assigned to more than one department.
7) Many double-period subjects (e.g. science labs).


-14-
As more of these factors are present, the more difficult it will be to build a conflict-reduced schedule.

Scheduling techniques have changed frequently and dramatically since the early 1960's, in keeping with changing national priorities. The increase in school enrollments and proliferation of curricular offerings in the 1960's spurred the development of computerized scheduling techniques. The number of courses offered, and the individualization of instruction resulted in a surge in the variety and flexibility of scheduling models available to schools. By the early 1970's, a growing awareness of accountability prompted educators to revitalize conventional scheduling.

3.2.3 MANUAL VS COMPUTER SCHEDULING

Many procedures employed by manual and computerized scheduling are the same. Steps requiring analysis and evaluation must be performed by the scheduler regardless of the scheduling method used. They include:

1) Determining student needs.
2) Reviewing the curriculum.
3) Formulating the program of studies.
4) Preparing registration materials.
5) Setting the calendar for registration.
6) Interpreting the tallies.
7) Identifying staffing needs.
8) Utilizing a conflict matrix.
9) Building the master schedule.
The computer is useful in preparation of the conflict matrix and providing course tallies. A conflict matrix can be prepared by hand but it is a very time consuming process. Computers are generally used in the "load" phase where the computer does not generate the master schedule but does assign students to classes. The computer also provides valuable reports and partial schedules to assist in refining the master schedule.

Junior high schools which use block scheduling techniques can manually produce their schedules while most high schools using mosaic schedules use computers to assist in the scheduling of students. There are two types of scheduling programs: student sectioning programs, and integrated class scheduling and student sectioning programs.

**STUDENT SECTIONING PROGRAMS**

Programs for assigning students to course sections in a pre-determined master schedule make up the majority of operational school scheduling computer programs.

One of the first operational models of this type was developed at Purdue University in 1956. I.B.M. followed this development with a program called CLASS (Class Load and Student Scheduling) based on the work at Purdue. Several other student sectioning programs have been written which are all basically quite similar. A large number of private and public organizations provide student sectioning services to schools.

Most of these models take as inputs the master class schedule, the student course requests, and class size parameters. Using various types of heuristics, students are sectioned, student schedules are printed out, and names of students for whom
the technique cannot find a no-conflict schedule are printed out. The scheduler makes some adjustments and the process is repeated.

INTEGRATED CLASS SCHEDULING AND STUDENT SECTIONING PROGRAMS

A number of research groups have written computer programs which integrate the two jobs of building the master class schedule and sectioning the students. Two models are GASP (Generalized Academic Simulation Program), developed at M.I.T., and SSSS (Stanford School Scheduling System), developed at Stanford University.

Input to these programs includes rooms and instructors available, desired time pattern for the school day and each course, and the student course requests. Using this data, a master schedule is developed and the students are sectioned. This procedure is most commonly performed by the scheduler who utilizes the conflict matrix to develop the master schedule. The main difference between these integrated class scheduling programs and programs that only "section" students is that the master schedule is computer generated.

High schools began to use these programs in the 1960's with the introduction of flexible modular scheduling. The varying time blocks and individualization provided for in a flexible modular schedule made it almost impossible to generate these schedules by hand. As a result schools started to use the services of computer companies to computer generate their schedule.
3.2.4 RESEARCH FINDINGS

COMPUTERIZED SCHEDULING

Lynne M. Durward in "Computerized Scheduling In Vancouver Schools" examined computerized scheduling in sixteen Vancouver secondary schools. Fifteen of these schools used the Honeywell Scheduling Program (a sectioning program) and one used Columbia Computing Services Limited (an integrated class and sectioning program). Durward listed the main advantages of computer scheduling as saving of secretarial time, more complete and accurate student lists and better balancing of class size. 2 Schools using the Honeywell Scheduling Program had a median conflict rate after the final scheduling run of 4.7%. The school using a flexible modular system of scheduling (the Columbia Service) had a 62% conflict rate. Durward found the 62% rate to be somewhat misleading because most of these were resolved by teachers agreeing to hold classes during their free modules to accommodate students. 3

The report's main criticisms of the Honeywell System were that the turnaround time for simulation runs was too long and that the run dates were too early and inflexible. While she found no significant relationship between the number of simulation runs and the conflict rate it appeared that schools with extra simulation runs were operating smoothly earlier.

3 Ibid., p. 10.
FLEXIBLE MODULAR SCHEDULING

Clayton Braddock in his 1967 article "Changing Times are Changing Schools" examined flexible modular scheduling. Braddock listed some of the advantages of modular scheduling as:

1) More efficient use of time.
2) More opportunities for independent study and research.
3) Closer contact with teachers and other students as well as smaller groups.
4) Highly personalized class schedules for each student.
5) Greater opportunities for teachers to teach in depth.
6) More decisions by students about their own school work.

At Trezevant High School in Memphis, Tennessee, Braddock found that as much as 40% of a student's time was unscheduled. Some provisions had to be made for students who were unable to handle the unscheduled time. Braddock found that many people felt that a modular schedule did not always accomplish its objectives. Dr. Frank Brown, a Superintendent of Schools in Florida, called modular scheduling a "gold plated fad." Brown felt modular scheduling took an excessive amount of administrative time and involved higher costs.

In 1965, Robert Oakford and Dwight Allen began a three year development program to determine the desirability of modular scheduling. In their report Oakford and Allen investigated the impact of the Stanford School Scheduling System (SSSS) on

5 Ibid., p. 1.
scheduling eighteen secondary schools. They found that in many project schools the technology for flexibility is present, but teachers and administrators were not geared to implement the full potential.

Oakford and Allen found that even in the schools where staffs were deeply committed to the new educational objectives, 99 percent of the possible alternatives permitted by modular scheduling had yet to be tried. The authors stated that "only people can make a modular schedule, a flexible schedule." 6

SECTIONING VS. INTEGRATED PROGRAMS

Project Pass (Project in Automated School Scheduling) was sponsored by the Western New York School Study Council during the summer of 1965. The goal was to provide in-service education for school personnel contemplating the use of automated approaches to school scheduling. The two techniques utilized were CLASS and GASP. Two pilot schools were selected and these schools were scheduled with both processes.

Analysis of the data showed CLASS and GASP could not be compared in terms of their ultimate objectives. CLASS is a "sectioning" technique, where as GASP develops a master schedule.

Dr. Leonard Chaffee and Dr. Robert Heller in their report on Project Pass 7 also concluded it would be financially impractical for schools with conventional programs to adopt GASP. GASP is for schools contemplating innovative instructional

programs which increase the number of variables to be considered in the development of the master schedule. Since it was originally designed for university scheduling it assumed more flexibility than was found in pilot schools. Many of the alternatives it presented would not have been so awkward in a higher education setting. As a result the GASP scheduler still had to build a significant portion of the schedule manually and then lock it into GASP.

SCHEDULING PROBLEMS

Robert Harding, in *The Problem of Generating Class Schedules for Schools*, examined the practical and theoretical approaches to generating class schedules. He found that no method was entirely satisfactory in terms of both the quality of output and the cost of the process. Harding found existing techniques suffered from one or a combination of the following weaknesses:

1) The process takes too long, either in total time and/or in computer time.
2) It costs too much.
3) It requires the attention of a high-level administrator for too long a time period.
4) It requires too much computer core capacity when compared with what is commonly available.
5) The schedules generated are of poor quality in terms of the following criteria:
   a) The number of students unable to schedule their requested courses.
   b) The number of teachers reassigned to areas in which they are less well qualified, to accomplish the scheduling process.
   c) Utilization of facilities.
   d) Schedule balance.
6) The analytical model is logically sound, but when applied to a real problem, the combinatorial properties so swell the problem that the method becomes computationally infeasible and hence not useful.
7) The approach does not accurately reflect the true conditions in the schools and therefore cannot generate a useful schedule of high quality.

---

3.3 THEORETICAL AND CONCEPTUAL DEVELOPMENT

The scheduling system proposed in this thesis will be able to schedule the students at Sodus Central School. The parameters will be left general so that this system could be used to schedule other schools. This project is broken down into three phases which correspond to normal schedule development.

3.3.1 PHASE 1

DESCRIPTION

Guidance counselors meet with each student to determine the student's course requests. The scheduler works with the department chairpersons and the principal to determine the course offerings. Two files are developed: one which contains course offerings and the other the student requests.

In this phase a program must be written which uses the student request file and course offering file as inputs. The output of this phase is a conflict matrix and the course tallies list. The normal procedure is that the conflict matrix and the course tallies list are printed and they are then used by the scheduler to determine the final course offering and to develop the master schedule.

This project will give the scheduler some added flexibilities. Course tallies are used to make decisions which might involve dropping a course from the offering. It would be possible to run the conflict matrix at a later time after the guidance department updates student requests. This would result
in a conflict matrix which is more accurate.

CURRENT PROBLEMS

1) Course tallies and conflict matrix are run at the same time. The course tally sheet is then used to make decisions about dropping courses with insufficient enrollment. Some students have to pick up new courses to replace courses dropped and as a result the conflict matrix is not accurate.

2) The conflict matrix is run once in May and by the second simulation run in July the data is too old to be of use in working out problems.

3) The scheduler must individually tally the number of students conflicting with a given course.

SOLUTIONS

1) The course tallies list can be regenerated after the course decisions have been made and the conflict matrix can then be run to provide more accurate information.

2) A conflict matrix could be regenerated whenever changes have been made to student course requests.

3) A conflict matrix could individually generate and tally all conflicts with a given course.

3.3.2 PHASE 2

DESCRIPTION

In phase 2 the Student Course File and the Course Schedule File become inputs for the sectioning program. The sectioning program loads students into the courses while attempting to maintain a balance in the class size of different sections of the
same course. If students are unable to schedule they are added to the conflict list and a partial schedule is printed. At the end of this simulation run, the school results are printed.

Guidance counselors use the partial schedules to resolve direct conflicts by having the students make a choice between the courses which conflicted. The scheduler uses the school results to examine problems with class balance and make other decisions about changing the times of course offerings. After changes have been made to the Student Request File and the Course Schedule File there will be another simulation run.

This process is continued until the scheduler is satisfied with the results and is ready for a final run. This thesis will give the scheduler some added flexibilities. Under present conditions BOCES allows only two simulation runs before the final run. Since the scheduler will have control of this process he/she can take more runs in attempting to provide a better schedule.

CURRENT PROBLEMS

1) BOCES only permits two trial simulation runs in the general package. A school must pay for extra runs.

2) Turnaround time is about two weeks for each run.

3) As a result of having only two trial runs, the first run is usually held until after June exams in order to include failures.

4) After the first trial run the scheduler might have to move some courses to different periods. He is unable to determine the possible effect the moves will have on each other. It is possible for a change to cause more problems
than were present originally.

5) It is not possible to resolve all balance problems.

SOLUTIONS

1) By having the capability of running the schedule on the school's equipment the scheduler can take as many trial runs as necessary to obtain a satisfactory schedule.

2) Turnaround time will be about one day instead of two weeks.

3) The first trial run could be held before school ended so guidance counselors would have time to personally contact each student with a direct conflict before they leave for summer vacation.

4) The scheduler does not have to solve all problems during a single trial run. A better schedule can result by isolating the results of a single change and having extra trial runs. This is especially true when trying to resolve balance problems.

5) If a trial run does not accomplish its objective the scheduler can back up and start from the last good run.

3.3.3 PHASE 3

DESCRIPTION

In phase 3 the same inputs are used as in phase 2 but now all student schedules and teacher class lists are printed with the school results.

CURRENT PROBLEMS

1) With only two trial runs in phase 2 the scheduler often makes some additional changes before the final run. If these changes cause any conflicts then the conflicts must be
hand scheduled and all class lists updated.

2) The final run is during the first two weeks in August, so any student entering after this must be hand scheduled and class lists updated.

3) Second semester class lists are inaccurate when January arrives.

4) Study halls must be manually divided after the final run.

5) Double gym classes for students who failed must be hand scheduled after the final run which results in many class lists being inaccurate.

SOLUTIONS

1) The final run can be the same as the last trial run so there are no untested results in the final run.

2) This thesis allows for the development of extensions after the final run which would permit additions and deletions while updating the class lists.

3.3.4. SCHEDULING CALENDAR

Scheduling is a year-long project for most schools. A typical calendar for a secondary school is:

November – January

Review the curriculum and make decisions on course offerings, type of schedule, and length of school day for the following year.

February – April

Guidance counselors meet with students to prepare the students' course requests.
May

Course offerings and student requests are used as input to the computer program to produce the conflict matrix and course tallies. Registration figures are adjusted for cancelled and merged courses and staffing needs are formulated.

June (first three weeks)

Construction of master schedule by the scheduler.

June (last week)

Change student requests to reflect results of the June exams.

July

Two trial or simulation runs of the schedule. During these runs student requests are matched with course offerings. The results are reviewed to determine whether the master schedule and/or student course selections must be modified. Students are contacted regarding conflicts.

August (first two weeks)

The final run of the schedule which produces all student schedules and class lists.

August (last two weeks)

Register new students and update class lists. Hand schedule students unassigned time (study halls) and make faculty study hall coverage assignments. Some classes must be hand balanced and lists updated.

September

Guidance counselors continue to enroll new students and add and delete courses from student’s schedules.
4. FUNCTIONAL SPECIFICATION

4.1. FUNCTIONS PERFORMED

The student scheduling system requires the user to input student and course information which is stored in system files. This data is used to develop the course tallies list and the conflict matrix. After allowing for updating and deleting of information, the system sections students and prints their schedules. The functions performed include:

1. Develop student information files.
   A. Student course request file.
   B. Student general information file.

2. Update student information file.

3. Develop course file.

4. Update course file.

5. Develop course tallies list.

   The course tallies list summarizes the student request information. The list will include the course number, course name, total course enrollment and the course enrollment by grade level.


   The conflict matrix gives the total number of students registered for any given pair of courses.
7. Section students.

A. Develop student schedule.
   
1. Get a student request record.
2. Get course records for the student's requests.
3. Prioritize the course requests.
   
Courses are first prioritized according to the number of sections of a given course that are being offered. Each section is then prioritized according to the number of seats remaining.

4. Schedule the student's courses.

B. Print student schedule for any student with a conflict. If a student is unable to schedule all of his courses his student ID is added to the conflict list file.

C. Print school results.

   After each run a list of course numbers and their enrollments is printed along with the scheduling results by grade level. The number of students assigned to study hall is also printed.

D. Store student schedule.

   The student's schedule is stored in the schedule file (during the final run).

8. Print results.

A. Print schedules of all students after the final run.

B. Print class lists from student schedule records after the final run.
4.2. SYSTEM DATA FLOW CHART

CONTEXT DIAGRAM

LEVEL 0

STUDENT SCHEDULING SYSTEM

STUDENT INFORMATION

Course Information

Course Tally

Course List

Conflict Matrix

Student Schedules

Class Lists

Conflicts & Partial Schedules

School Report

USER

PRINTER
EXPLOSION 5

5.1 Course Tally List

USER

5.2 Conflict Matrix

PRINTER

Matrix Request

Tally Request

CRS * CRS NAME

CRS ** CRS NAME, Enrollment

Sent

SEND
EXPLOSION 7.4

7.4.4. Remove Conflict CSs from TABLE. ADD TO UNSCHEDULED LIST

7.4.1. Schedule CSs AT HEAD OF TABLE

7.4.2. Check TABLE for another section. ADD Scheduled CSs TO PROBLIST

7.4.3. Check to see if scheduled CS is singleton

7.4.5. Replace all sections of CS at head of TABLE

7.4.6. Replace all sections of course at head of PROBLIST

- 34 -
4.3. LIMITATIONS AND RESTRICTIONS

The system will not generate the master schedule. The master schedule information must be an input.

4.4. USER INPUTS

Student and course records developed from user inputs are stored in system files. The student information is entered once but stored in two separate records, the student record and the general student information record. The system scheduler develops schedule and class list records which are also stored in system files. The inputs to the system include:

1. Student information
   A. Student record (entered by user)
      - Student name
      - Student ID
      - Grade
      - Course requests
   B. General student information record (entered by user)
      - Student name
      - Student ID
      - Address
      - Phone number
      - Homeroom number
      - Birthdate

2. Course record (entered by user)
   - Course number
   - Course name
   - Section number
   - Maxsize
   - Period number
   - PD number (scheduling period)
   - Days
   - Semester
   - Room number
   - Teacher number
3. Schedule record (generated by scheduler (DFD EXPLOSION 7))
   - Student ID
   - Student name
   - Grade
   - Scheduled courses
4. Course requests Array [1 to 9] (from user)
   - Course numbers

4.5. USER OUTPUTS

1. Course tallies list - This list is printed using information from the student request file and the course file (DFD EXPLOSION 5). The list contains:
   - Course number
   - Course name
   - Total enrollment
   - Enrollment by grade level

2. Conflict Matrix - This matrix is a printed listing of the enrollment for courses which could conflict with a requested course (DFD EXPLOSION 5). The matrix contains:
   - Course for which matrix is requested.
   - List of all courses which will conflict with that course.
   - Enrollment for each given pair of course numbers.

3. Student Schedule - The schedule is stored in the schedule file for printing at a later time. It is developed using information from the course file and the student request file (DFD EXPLOSION 7). The stored schedule contains:
   - Student name
   - Student ID
   - Student grade
   - Schedule course records
When printing the schedule (DFD EXPLOSION 8) the following information is added from the general student information file:

- Address
- Phone number
- Homeroom number
- Birthdate

4. School results report - This report is generated after each run of the schedule using information stored in the course file (DFD EXPLOSION 7). The report contains:

- Course number
- Course name
- Section number
- Maxsize
- Period number
- PD number
- Days
- Semester
- Room number
- Teacher number
- Enrollment

5. Partial Schedules - The schedules of students who were unable to schedule all of their courses are printed along with the course numbers of their unscheduled courses (DFD EXPLOSION 7).
6. **Class lists** - These lists are only printed after the final run. They are printed from information stored in the schedule file (DFD EXPLOSION B). The lists contain:
- Course name
- Course number
- Section
- Period number
- Days
- Teacher number
- Student list

4.6. **SYSTEM FILES**

The student request and general information files, and the course file are inputed by the user. The system generates the schedule file during a run of the schedule. The temporary class record file is a temporary file that is generated in order to print class lists. The following files are developed and maintained by the student scheduling system:

1. **Student request file**
   - Student request records
2. **Student information file**
   - General student information records
3. **Course file**
   - Course records
4. **Schedule file**
   - Schedule records
5. **Temporary class record file**
   - Schedule records
5. SYSTEM SPECIFICATION

5.1. SYSTEM ORGANIZATION CHART
EXPLOSION 1

1

A  B  C

Use Existing File to Change Student Info

I

E  G  D

Use Existing File to Change Stud Course Requests

H

Replace Stud Course Req Record

Get a Student Course Req Record

A  B  C  D  E  F  G  H

Get a Gen Info Record

Add Gen Info Record to List

Save List Gen Info Records to File

Save List Stud Course Req to File

Retrieve Stud Course Req List from File

Print List Gen Info Records

Print List Stud Course Req Records

Query List by Student Name
EXPLOSION 2

2

Enter Course Numbers First Time

Load Course File Into List

Enter Other Course Info

Add Courses

Delete Courses

Change Course Data

Query List By Course Number

Send Course List To Printer

Save Course List To File
EXPLOSION 3

Get Range of Grades
Get Course List
Get Matrix Request
Get Student Record
Update Matrix
Print Matrix
EXPLOSION 4

- Get Range of Grades
- Get a Course List
- Get a Student Record
- Update Course Tallies List
- Print Course Tallies List
EXPLOSION 6 AND 7

6

Get Range of Grades
Get List of Courses
Get Student Record
Update Class List
Print Class List

7

Get Range of Grades
Get Course List
Get Student Schedule Record
Print Schedule
5.2. EQUIPMENT CONFIGURATION

This scheduling system requires an Apple IIE with four disk drives and 64K main memory with an extended eighty column card.

5.3. IMPLEMENTATION TOOLS

This student scheduling system is written in Apple Pascal 1.2.

5.4. OVERVIEW OF THE PROGRAMMING SYSTEM

The student scheduling system presented in this thesis consists of seven programs written in Apple Pascal. The following sections present an overview of each program.

5.4.1. Enstuddata (see program listing 10.1.)

This program allows the user to create and update all student information. Entstuddata uses Unit One (see program listing 10.2.) which contains subroutines used by options one and two. The user is allowed to select one of four options:

   Option 1

   This option is used the first time the user wants to enter student information for each grade. The user enters general information for a grade and the information is stored in a student request record (used for student course requests) and a general information record (general student information). At this time the course requests are unknown so they are not entered. The
student request records and general information records are placed in separate linked lists in alphabetical order. The user is then given the opportunity to save the two lists to files on a disk. One file will contain student request records (ST:STREC(grade)) and the other will contain general information records (ST:GINFO(grade)) for the same grade.

Option 2

This option allows the user to select from nine options in order to use an existing file to enter and change student information. The grade is used as the key to locate the correct files. The records are then placed into separate linked lists. As information is changed for any student, both the student request record and the general information record are updated. It is also possible to add or delete students. Any student who is added is placed in both lists in alphabetical order. Options are provided to print either the student course request list (see appendix 9.1.2.) or the general information list (see Appendix 9.1.1). After updating student information both lists can be saved to files on disk.

Option 3

The user can use option 3 to select from six options in order to use an existing student request file to enter and change course records. This option is used after the student request records have been created by options one or two. The records for a selected grade are read from the student request file for that grade and placed in a linked list in alphabetical order. The user then is allowed to add up to 10 course requests (course numbers) per student. The course requests are sorted and stored in an
array in the course request record. The user may query the course request list by student name or send the list to the printer.

Option 4

Exit the program.

5.4.2 Entcourdata (see program listing 10.5.)

This program allows the user to select from ten options to enter and update all course information. The first option allows the user to enter course information (course name and number) for the first time. The course number and name are read into a course record which is then inserted into a linked list ordered by course number. This option allows the user to develop the list of courses to be offered before other specific course information is known. After the Coursefile (CR: COURSEFILE) has been developed, it may be retrieved to add, delete and change courses or course information. The course file contains course records which are placed in a linked list in order of course number. The user may query the list by course number or send the list to the printer (see Appendix 9.4.). The updated course list may be saved to disk.

5.4.3. Cmatrix (see program listing 10.3.)

This program produces the conflict matrix (see Appendix 9.2.). The user enters the range of grades that the matrix will cover. The matrix is a list of all courses which would conflict with the course requested by the user. The list would also
display the number of students who would have this conflict.

The program uses two parallel arrays to tally the number of students in conflict with the course entered by the user. One array which contains course names is initialized to all blanks while the other array which contains tallies (integers) is initialized all to zero. Course records are read from the course file (CR:COURSEFILE). The course name is placed into the course name array using the course number as the index.

The user enters the number of the course for which the matrix is to be constructed. The student request records, with file names which match the range of grade numbers entered by the user, are read. If the student's course request array contains the same course number as requested for the matrix then the course number is used as an index to add one to the tally in the tally array for each course that the student is requesting. When this is completed for all students in the requested grade range the matrix for the requested course is complete.

The conflict matrix is then printed by indexing through both arrays and printing only when the course name array is not blank and the tally array is greater than zero.

5.4.4 Tallylist (see program listing 10.4)

Tallylist prints the course tallies list (see Appendix 9.3). This list shows the enrollment for each course in the curriculum by grade level.

A two-dimensional array is initialized to zero. This array is used as an array of counters which keeps track of course
tallies. One dimension is indexed by the course number and the other by grade. The user enters the range of grades. The grade is then used to get the correct student course request files (ST:STREC(grade)). The student's record is read from the file and every course the student requested is used as an index along with the student's grade, so one can be added to the tally. The sum of the rows provides the total enrollment for a course.

A course name array which is used to print the course names is filled by reading course records from the coursefile (CR:COURSEFILE). The course tallies list includes the course name, number, total enrollment and enrollment by grade level for each course offered.

5.4.5. Scheduler (see program listing 10.6.)

This program schedules students and prints the results of each scheduling run. Scheduler uses Unit Two (see program listing 10.7.) which contains sub-routines used by the Scheduler. Unit Two loads all course records from the coursefile into course enrollment records and sets the class enrollment to zero. The class enrollment records are loaded into a linked list in order of course, section and PD (scheduling period) number.

The user must also enter the type of scheduling run (trial or final) and the range of grades to be scheduled. During a final scheduling run the student schedule records are saved to a file (SCH:SCH(grade)).

A student record is read from the student request file and the course records of all sections of the course which a student
requested are found in the linked list of course enrollment records. These records are inserted into a linked list of course records in order of course, section and PD number for the student being scheduled. This linked list is then used to create a table which is a two-dimensional array of course number, course section, class balance (enrollment subtracted from max seats), and priority. The priority is the number of sections of any course. The table is then sorted first by priority then by the class balance.

The table which contains a prioritized list of all sections of a student's requested courses is then used to schedule the student. The first course in the table is placed in the schedule (a two-dimensional array) indexed by PD numbers (1..13) and days of the week (M,T,W,R,F). The schedule array contains course numbers which are initialized to zero. To schedule a course the schedule array is checked for zero and if a zero is found then the course is scheduled in place of the zero. If a conflict exists and if another section of the same course exists in the table then the attempt is made to schedule that section. The course conflicting with the original section is then added to the problem list (an array of course numbers which resulted in conflicts). If another section of the same course does not exist then the conflict course is removed from the schedule and all sections of this conflict course are added to the head of the table. The original course that the program was trying to schedule is scheduled in place of the conflict course. The conflict course which is now at the head of the table is now scheduled and if this results in another conflict then the course

-53-
is removed from the table and declared a conflict. If only one conflict exists in a schedule then the student is considered partially scheduled but if a second conflict occurs the student is considered unscheduled.

Courses are taken from the head of the table and placed in the schedule until the table is empty. If a course is scheduled then one number is added to the enrollment for that section of the course.

When the student records of an entire grade have been scheduled then the scheduling results of the grade are printed and if it is the final run the schedule records are saved to a file. When all requested grades have been scheduled the final school results (see Appendix 9.5.) are printed.

5.4.6. Printschedules (see program listing 10.8.)

This program prints out the student schedules (see Appendix 9.6.) after the final run of the schedule has been completed. The course records are read from the course file and inserted into a linked list in order of course number, course section and PD number.

The user enters the range of grades for which schedules are to be printed. The student schedule records are read from the schedule file (SCH:SCH(grade)) and information is used from the course records list to print the students' schedules.
5.4.7. Printclasslists (see program listing 10.9.)

This program prints course class lists (see Appendix 9.6) using records from the schedule file. The course records are read from the course file and inserted into a linked list in order of course number, course section, and PD number.

The user enters the range of grades for which the class lists are to be generated. All courses for which a student is scheduled are copied from student schedule records in the schedule files. These records are transferred into temporary class records containing the student name, course, section, student ID and grade and are then stored in a temporary class record file.

A course class list is then developed for each course and section listed in the linked list of courses. The temporary class record file is read and every record which contains the given course and section is inserted into a linked list of temporary class records in alphabetical order. This list is then printed as the class list.
6. VERIFICATION AND VALIDATION

6.1. TEST PLAN

In this thesis a programming system to schedule students using a microcomputer was developed. The scheduling at Sodus High School is currently done using a mainframe computer at BOCES. To test this thesis programming system a schedule was developed using the same data that was used for scheduling Sodus High School with the BOCES computer.

6.2 TEST PROCEDURES

The schedule for the 1984-85 school year at Sodus High School was developed during the summer of 1984. This schedule was run on the BOCES computer for grades nine through twelve. In testing the student scheduling program written for this thesis the same data was used.

The test of this program was done in the following sequence:

1. Development of a list of student records (see Appendix 9.1).
2. Development of a conflict matrix (see Appendix 9.2).
3. Development of a course tallies list (see Appendix 9.3).
4. Development of a course list (see Appendix 9.4).
5. Two trial runs of the schedule (see Appendix 9.5).
6. A final run of the schedule (see Appendix 9.5).
7. Printing student schedules (see Appendix 9.6).
8. Printing course lists (see Appendix 9.7).
6.3 TEST RESULTS

The results of the trial and final runs of this project were very similar to the results of the BOCES run of the schedule. The main advantage of having a microcomputer scheduling program was to provide more local control over the scheduling process and this project does provide that control. Some advantages that local control provided were:

1. Turnaround time was reduced from about two weeks to one day. The thesis program takes approximately 3.5 hours to run.

2. It is possible to do more than two trial runs or to go back to a previous run if a run is unsatisfactory.

3. A conflict matrix can be generated before any run, not just the first run. This is very helpful in making changes in later runs of the schedule. Also, conflict matrices are much more useful because they are printed for each individual course.

4. The course tallies list will be more useful because it can be generated anytime. Using the BOCES system the course tallies list is generated after all students have signed up for their courses. Before the list is received from BOCES the tallies have already been counted by hand. In order to make decisions regarding staff assignment for the next year lists are made by secretaries of all students enrolled in a given course. This step could be eliminated by an extension to this project. The result would be a savings in time spent by secretaries in compiling and counting student course requests.
5. Since more than two trial runs are possible the first run of the schedule can occur before school is over in June. This will permit guidance counselors to contact students regarding schedule changes before they leave for the summer.

6. If a run was completed before the end of the school year it would be possible for the scheduler to consult with the department chairpersons and teachers regarding possible changes in their schedules before they leave for the summer vacation. This would allow for more staff input to the scheduling process.

7. By developing some of the extensions suggested in section 7.3.2., this student scheduling system would become even more useful to a school.

**COMPARISON OF RESULTS BY RUN**

<table>
<thead>
<tr>
<th>RUN</th>
<th>TOTAL STUDENTS</th>
<th>BOCES</th>
<th>THESIS PROJECT</th>
</tr>
</thead>
<tbody>
<tr>
<td>RUN 1</td>
<td>TOTAL STUDENTS</td>
<td>499</td>
<td>496</td>
</tr>
<tr>
<td></td>
<td>STUDENTS FULLY SCHEDULED</td>
<td>352</td>
<td>356</td>
</tr>
<tr>
<td></td>
<td>STUDENTS PARTIALLY SCHEDULED</td>
<td>131</td>
<td>123</td>
</tr>
<tr>
<td></td>
<td>UNSCHEDULED STUDENTS</td>
<td>16</td>
<td>17</td>
</tr>
<tr>
<td>RUN 2</td>
<td>TOTAL STUDENTS</td>
<td>498</td>
<td>496</td>
</tr>
<tr>
<td></td>
<td>STUDENTS FULLY SCHEDULED</td>
<td>462</td>
<td>464</td>
</tr>
<tr>
<td></td>
<td>STUDENTS PARTIALLY SCHEDULED</td>
<td>35</td>
<td>30</td>
</tr>
<tr>
<td></td>
<td>UNSCHEDULED STUDENTS</td>
<td>1</td>
<td>2</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>----------------------</td>
<td>-------</td>
<td>-----</td>
<td></td>
</tr>
<tr>
<td>FINAL RUN TOTAL STUDENTS</td>
<td>498</td>
<td>496</td>
<td></td>
</tr>
<tr>
<td>STUDENTS FULLY SCHEDULED</td>
<td>489</td>
<td>486</td>
<td></td>
</tr>
<tr>
<td>STUDENTS PARTIALLY SCHEDULED</td>
<td>9</td>
<td>10</td>
<td></td>
</tr>
<tr>
<td>UNSCHEDULED STUDENTS</td>
<td>0</td>
<td>0</td>
<td></td>
</tr>
</tbody>
</table>

7. CONCLUSIONS

7.1. PROBLEMS ENCOUNTERED AND SOLVED

In the development of this program, Apple Pascal was used with an Apple IIE microcomputer. Apple Pascal 1.1 has several bugs, one of which was that IORESULT did not always return a correct value. IORESULT is supposed to return a value from one to sixteen if an error occurs in an input/output operation and a zero if no error occurs. One of the bugs in Apple Pascal 1.1 was that a fourteen should be returned for an error in reading integer or real data. The IORESULT function on Apple Pascal 1.1 did not function correctly and a zero was returned.

Apple Pascal 1.2 corrected 43 bugs in Apple Pascal 1.1 but it did not correct the IORESULT bug. To get around the problem, it was discovered that when an integer was read and an alpha character was entered it left IORESULT as zero but it also left the integer variable a zero. A loop was then used to read until the variable was greater than zero.

While Apple Pascal 1.2 did not solve the IORESULT problem, it did provide some solutions to other problems that were encountered. This newer version of Apple Pascal required an extended 80-column card which provided another 64K of memory. In Apple Pascal 1.1 it was only possible to edit files which were less than 38 blocks (19K bytes). If the program exceeded 38
blocks then some subroutines could be compiled as units and stored in the system library. Apple Pascal 1.2 allowed the editing of files up to 58 blocks (29K bytes). This allowed for greater flexibility in program development.

One of the major problems which was encountered in writing this program for an Apple, was in careful use of main memory. It was very important to develop algorithms which did not result in stack overflow problems. One example of this occurred in the development of the conflict matrix. The conflict matrix printed by the BOCES program was a large matrix (100 by 100) of course tallies. When printed this matrix was eleven sheets of paper which were then taped together to form the giant conflict matrix. In trying to develop this matrix the Apple ran out of main memory so ways had to be found to break the problem down so the Apple could handle it. This matrix is used to find how many students are enrolled in any given pair of courses. In scheduling a course, a list is prepared of the courses and number of students that would conflict with the given course. The solution was to compare one course against all the others and print all the courses that would conflict with the given course and how many students would have the conflict. This turned out to be a better solution to the problem than the BOCES matrix.

Another area where insufficient main memory caused changes in algorithm development was in the Scheduler program. This program grew very large and used many arrays so the use of main memory became an important part of algorithm design. The initial scheduling algorithm used in the Scheduler did not find all possible scheduling combinations, which resulted in

-60-
scheduling results about 12% lower than the BOCES results. This problem was solved by using a problem list, an array which keep track of other scheduling possibilities. This enabled the Scheduler to find the other scheduling combinations if all courses where unable to be scheduled.

The original design planned to add each student's course selections to the appropriate class list after scheduling the student. The design was changed to write student schedule records to a file only after the final run and then use this file to print both class lists and student schedules. Besides saving some run time the improved design provides for some future extensions which would make the system even more useful.

7.2 DISCREPANCIES AND SHORTCOMINGS OF THE SYSTEM

This microcomputer version of the scheduling program takes considerable time to run. It requires approximately 3.5 hours for a trial run and about four hours to print the class lists after the final run. If another microcomputer such as the IBM XT, with a larger main memory was used, it might be possible to improve some of the scheduling algorithms and reduce run time.

7.3 LESSONS LEARNED

7.3.1 ALTERNATIVE APPROACHES FOR IMPROVED SYSTEM

This scheduling program could be modified to work more efficiently and run faster on a microcomputer with a larger main memory.
7.3.2. SUGGESTIONS OF FUTURE EXTENSIONS

Many extensions could be added which would greatly simplify the scheduling process. Some of these are:

1. An extension could be written to schedule all study halls that a student would have in his/her unscheduled time.

2. A report card and attendance extension could make this a complete system to replace the services now offered by BOCES and used by many districts. Local districts contract with BOCES for scheduling, attendance and report card services. The three services come as a package so the local district would have to replace all three services.

3. An extension to update all schedules and class lists as students add and drop courses would be of great value. Since the students’ schedules are stored in the schedule file, an extension could be written to enable the user to change all courses in an individual student’s schedule and rewrite the schedule to the schedule file. Once this schedule was available then the Printclasslists and Printschedule programs could be used to generate new class lists and schedules. This would allow a school to print completely accurate class lists and schedules for second semester (January). These second semester class lists and schedules are currently printed at schedule run-time in August and as a result they are very inaccurate by January.

4. Another extension could be written to allow the guidance counselors to generate a schedule for a single student. The student could be added to the appropriate class lists and the schedule could then be sent to a printer
so that the student could leave with a copy of the schedule. This would be especially useful during the first days of school to enroll new students.

5. It would be possible to write an extension to allow the school offices to have some on-line query capabilities of student schedules. The five offices at Sodus High School currently use handwritten copies of student schedules which are often inaccurate.

7.3.3 FUTURE DIRECTIONS IN SCHEDULING

Computers started to play a role in scheduling of high school students in the 1960's. There was a great deal of research into scheduling theory at that time and most schools began to do their scheduling using computers. Some companies provided scheduling services to schools but this was often very expensive. As a result schools began to share the use of computer services through BOCES. These services included payroll, scheduling, attendance and report cards.

In the 1960's and 1970's the climate of schools began to change and the curriculum began to expand to include more electives and free time for students. Many scheduling patterns such as flexible modular scheduling were developed using computers, which allowed schools to develop schedules to accommodate the many changes in schools. This lead to the development of open schools and other innovative approaches to education.

After the big push for change started to wear off, schools in the past few years have pulled back to more
traditional patterns and schedules. The literature shows a reduction in approaches to school scheduling in the late 1970's and early 1980's but in the past few years schools are starting to use microcomputers both in the classroom and in doing administrative tasks. This has lead to an increased interest in scheduling and in the next few years many programs will be written to do scheduling for schools on microcomputers.
8. BIBLIOGRAPHY

In this article the author examines some of the advantages and disadvantages of modular or flexible class scheduling.

This project (Project PASS) utilized the CLASS and GASP scheduling techniques to provide in-service education for school personnel.

The authors provide technical information and a common sense approach to the scheduling process.

This study examined computerized scheduling in Vancouver Secondary Schools and analyzed the results.

This dissertation reported research in scheduling theory and examined the use of analytical models to solve the scheduling problem.

A step-by-step process is described for registering students in Junior and Senior High Schools by computer.

An examination of the impact of the Stanford Scheduling System on 18 Secondary Schools.
Simair, Dennis J., "Computer Uses in School Administration: A pilot Project," British Journal of Educational Technology, N02 Vol13 (May, 1982), pp114-128. This report describes the implementation of a project which used microcomputers in administration.

9. APPENDICES

The following pages contain samples of the indicated reports.

9.1. LIST OF STUDENT RECORDS

9.1.1. GENERAL INFORMATION
<table>
<thead>
<tr>
<th>STUDENT NAME</th>
<th>ST.NO.</th>
<th>HR</th>
<th>BIRTH</th>
<th>PHONE</th>
<th>SEX</th>
</tr>
</thead>
<tbody>
<tr>
<td>ABARE ANN</td>
<td>21610</td>
<td>113</td>
<td>9/1/70</td>
<td>483-9999</td>
<td>F</td>
</tr>
<tr>
<td>123 MILL ST.</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>SODUS 14551</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>ALLEN CRYSTAL</td>
<td>21465</td>
<td>113</td>
<td>9/1/70</td>
<td>483-9999</td>
<td>F</td>
</tr>
<tr>
<td>123 MILL ST.</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>SODUS 14551</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>ALLEN MATTHEW</td>
<td>21465</td>
<td>113</td>
<td>9/1/70</td>
<td>483-9999</td>
<td>M</td>
</tr>
<tr>
<td>123 MILL ST.</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>SODUS 14551</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>BARCLAY TINA</td>
<td>22532</td>
<td>113</td>
<td>9/1/70</td>
<td>483-9999</td>
<td>F</td>
</tr>
<tr>
<td>123 MILL ST.</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>SODUS 14551</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>BAUMBARTNER MARK</td>
<td>22556</td>
<td>113</td>
<td>9/1/70</td>
<td>483-9999</td>
<td>M</td>
</tr>
<tr>
<td>123 MILL ST.</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>SODUS 14551</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>BAUMBARTNER MICHAEL</td>
<td>22557</td>
<td>113</td>
<td>9/1/70</td>
<td>483-9999</td>
<td>M</td>
</tr>
<tr>
<td>123 MILL ST.</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>SODUS 14551</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>BAURER JAMES</td>
<td>21471</td>
<td>113</td>
<td>9/1/70</td>
<td>483-9999</td>
<td>M</td>
</tr>
<tr>
<td>123 MILL ST.</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>SODUS 14551</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>BECKER DONNA</td>
<td>22763</td>
<td>113</td>
<td>9/1/70</td>
<td>483-9999</td>
<td>F</td>
</tr>
<tr>
<td>123 MILL ST.</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>SODUS 14551</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>BRANCH HEIDI</td>
<td>22807</td>
<td>113</td>
<td>9/1/70</td>
<td>483-9999</td>
<td>F</td>
</tr>
<tr>
<td>123 MILL ST.</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>SODUS 14551</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>BRET CHAITY</td>
<td>22557</td>
<td>113</td>
<td>9/1/70</td>
<td>483-9999</td>
<td>F</td>
</tr>
<tr>
<td>123 MILL ST.</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>SODUS 14551</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>BROWN KRISTIN</td>
<td>21476</td>
<td>113</td>
<td>9/1/70</td>
<td>483-9999</td>
<td>F</td>
</tr>
<tr>
<td>123 MILL ST.</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>SODUS 14551</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>BROWN IROY</td>
<td>20563</td>
<td>113</td>
<td>9/1/70</td>
<td>483-9999</td>
<td>M</td>
</tr>
<tr>
<td>123 MILL ST.</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>SODUS 14551</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>BULLOCK JOHN</td>
<td>21477</td>
<td>113</td>
<td>9/1/70</td>
<td>483-9999</td>
<td>M</td>
</tr>
<tr>
<td>123 MILL ST.</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>SODUS 14551</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>BURNAP WENDRA</td>
<td>22587</td>
<td>113</td>
<td>9/1/70</td>
<td>483-9999</td>
<td>F</td>
</tr>
<tr>
<td>123 MILL ST.</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>SODUS 14551</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
9.1.2. STUDENT COURSE REQUEST LIST
<table>
<thead>
<tr>
<th>STUDENT NAME</th>
<th>GR</th>
<th>ST.NO.</th>
</tr>
</thead>
<tbody>
<tr>
<td>ABAARE ANN</td>
<td>9</td>
<td>21610</td>
</tr>
<tr>
<td></td>
<td>701</td>
<td>505 462 308 205 105 6</td>
</tr>
<tr>
<td>ALLEN CRYSTAL</td>
<td>9</td>
<td>21465</td>
</tr>
<tr>
<td></td>
<td>701</td>
<td>410 309 205 106 73 52 7</td>
</tr>
<tr>
<td>ALLEN MATTHEW</td>
<td>9</td>
<td>21465</td>
</tr>
<tr>
<td></td>
<td>701</td>
<td>410 307 204 105 6</td>
</tr>
<tr>
<td>BARCLAY TINA</td>
<td>9</td>
<td>22532</td>
</tr>
<tr>
<td></td>
<td>701</td>
<td>555 554 410 308 204 105 6</td>
</tr>
<tr>
<td>BAUMGARTNER MARK</td>
<td>9</td>
<td>22556</td>
</tr>
<tr>
<td></td>
<td>701</td>
<td>462 410 307 204 105 6</td>
</tr>
<tr>
<td>BAUMGARTNER MICHAEL</td>
<td>9</td>
<td>22537</td>
</tr>
<tr>
<td></td>
<td>701</td>
<td>410 307 255 203 105 6</td>
</tr>
<tr>
<td>BAURER JAMES</td>
<td>9</td>
<td>21471</td>
</tr>
<tr>
<td></td>
<td>701</td>
<td>603 404 308 204 105 6</td>
</tr>
<tr>
<td>BECKER DONNA</td>
<td>9</td>
<td>22783</td>
</tr>
<tr>
<td></td>
<td>701</td>
<td>655 654 410 308 204 105 6</td>
</tr>
<tr>
<td>BRANCH HEIDI</td>
<td>9</td>
<td>22807</td>
</tr>
<tr>
<td></td>
<td>701</td>
<td>464 404 308 205 105 6</td>
</tr>
<tr>
<td>BRETT CHARITY</td>
<td>9</td>
<td>22557</td>
</tr>
<tr>
<td></td>
<td>701</td>
<td>551 550 309 206 106 7</td>
</tr>
<tr>
<td>BROWN KRISTIN</td>
<td>9</td>
<td>21476</td>
</tr>
<tr>
<td></td>
<td>701</td>
<td>462 404 308 203 105 6</td>
</tr>
<tr>
<td>BROWN TROY</td>
<td>9</td>
<td>20563</td>
</tr>
<tr>
<td></td>
<td>701</td>
<td>505 503 309 206 106 73 52 7</td>
</tr>
<tr>
<td>BULLOCK JOHN</td>
<td>9</td>
<td>21477</td>
</tr>
<tr>
<td></td>
<td>701</td>
<td>505 503 309 206 106 73 52 7</td>
</tr>
<tr>
<td>DURNAP KENDRA</td>
<td>9</td>
<td>22587</td>
</tr>
<tr>
<td></td>
<td>701</td>
<td>654 464 404 307 204 105 6</td>
</tr>
<tr>
<td>CASTIGLIONE JODY</td>
<td>9</td>
<td>22677</td>
</tr>
<tr>
<td></td>
<td>701</td>
<td>655 654 462 404 307 204 105 6</td>
</tr>
<tr>
<td>CASTLE THOMAS</td>
<td>9</td>
<td>21481</td>
</tr>
<tr>
<td></td>
<td>701</td>
<td>404 308 204 105 6</td>
</tr>
<tr>
<td>CLEVELAND GREGORY</td>
<td>9</td>
<td>22580</td>
</tr>
<tr>
<td></td>
<td>701</td>
<td>462 410 307 204 105 6</td>
</tr>
<tr>
<td>COBB MICHAEL</td>
<td>9</td>
<td>22633</td>
</tr>
<tr>
<td></td>
<td>701</td>
<td>655 404 307 204 105 6</td>
</tr>
<tr>
<td>CONROW DONNA</td>
<td>9</td>
<td>22578</td>
</tr>
<tr>
<td></td>
<td>701</td>
<td>551 550 307 306 106 73 52 7</td>
</tr>
</tbody>
</table>
9.2. CONFLICT MATRIX
<table>
<thead>
<tr>
<th>CRS NAME</th>
<th>CODE</th>
<th>TALLY</th>
</tr>
</thead>
<tbody>
<tr>
<td>ENG 12</td>
<td>1</td>
<td>8</td>
</tr>
<tr>
<td>ENG 11-2</td>
<td>2</td>
<td>1</td>
</tr>
<tr>
<td>ENG 11-3</td>
<td>3</td>
<td>1</td>
</tr>
<tr>
<td>AP ENGLISH</td>
<td>9</td>
<td>3</td>
</tr>
<tr>
<td>SS IND STD</td>
<td>100</td>
<td>11</td>
</tr>
<tr>
<td>AM STD 3</td>
<td>102</td>
<td>1</td>
</tr>
<tr>
<td>LAW</td>
<td>107</td>
<td>4</td>
</tr>
<tr>
<td>MATH 12-1</td>
<td>200</td>
<td>2</td>
</tr>
<tr>
<td>MATH 12-2</td>
<td>201</td>
<td>3</td>
</tr>
<tr>
<td>MATH CRS 3</td>
<td>202</td>
<td>2</td>
</tr>
<tr>
<td>COMP PRG 4</td>
<td>250</td>
<td>1</td>
</tr>
<tr>
<td>COMP-AWARE</td>
<td>255</td>
<td>5</td>
</tr>
<tr>
<td>ADV BIO</td>
<td>300</td>
<td>1</td>
</tr>
<tr>
<td>PHYSICS 2</td>
<td>301</td>
<td>4</td>
</tr>
<tr>
<td>BIOLOGY 3</td>
<td>306</td>
<td>2</td>
</tr>
<tr>
<td>SPAN 1</td>
<td>410</td>
<td>1</td>
</tr>
<tr>
<td>OFF PROCED</td>
<td>451</td>
<td>1</td>
</tr>
<tr>
<td>ACCT</td>
<td>453</td>
<td>1</td>
</tr>
<tr>
<td>BUS LAW</td>
<td>455</td>
<td>1</td>
</tr>
<tr>
<td>PER TYPING</td>
<td>462</td>
<td>1</td>
</tr>
<tr>
<td>MECH DWG</td>
<td>507</td>
<td>2</td>
</tr>
<tr>
<td>PHOTOG</td>
<td>604</td>
<td>3</td>
</tr>
<tr>
<td>YEARBOOK</td>
<td>605</td>
<td>1</td>
</tr>
<tr>
<td>CON BAND</td>
<td>654</td>
<td>1</td>
</tr>
<tr>
<td>CON CHOIR</td>
<td>655</td>
<td>1</td>
</tr>
<tr>
<td>PE 11-12</td>
<td>700</td>
<td>8</td>
</tr>
<tr>
<td>PE 9-10</td>
<td>701</td>
<td>2</td>
</tr>
<tr>
<td>PE BOCES</td>
<td>702</td>
<td>1</td>
</tr>
<tr>
<td>DR ED TH</td>
<td>750</td>
<td>6</td>
</tr>
<tr>
<td>DR ED TH</td>
<td>751</td>
<td>1</td>
</tr>
<tr>
<td>PSYCH</td>
<td>790</td>
<td>4</td>
</tr>
</tbody>
</table>
# CONFLICT MATRIX FOR COURSE 901

<table>
<thead>
<tr>
<th>CRS NAME</th>
<th>CODE</th>
<th>TALLY</th>
</tr>
</thead>
<tbody>
<tr>
<td>ENG 12</td>
<td>1</td>
<td>40</td>
</tr>
<tr>
<td>ENG 11-2</td>
<td>2</td>
<td>3</td>
</tr>
<tr>
<td>ENG 11-3</td>
<td>3</td>
<td>12</td>
</tr>
<tr>
<td>WRIT LAB</td>
<td>52</td>
<td>3</td>
</tr>
<tr>
<td>WRIT 11-12</td>
<td>53</td>
<td>2</td>
</tr>
<tr>
<td>READING 10</td>
<td>72</td>
<td>2</td>
</tr>
<tr>
<td>READ 11-12</td>
<td>74</td>
<td>1</td>
</tr>
<tr>
<td>AM STD 2</td>
<td>101</td>
<td>3</td>
</tr>
<tr>
<td>AM STD 3</td>
<td>102</td>
<td>1</td>
</tr>
<tr>
<td>E CUL ST 3</td>
<td>104</td>
<td>1</td>
</tr>
<tr>
<td>LAW</td>
<td>107</td>
<td>2</td>
</tr>
<tr>
<td>AMER STD 3</td>
<td>110</td>
<td>14</td>
</tr>
<tr>
<td>MATH CRS 3</td>
<td>202</td>
<td>2</td>
</tr>
<tr>
<td>MATH CRS 1</td>
<td>204</td>
<td>1</td>
</tr>
<tr>
<td>COMP-AWARE</td>
<td>255</td>
<td>4</td>
</tr>
<tr>
<td>CHEM 2</td>
<td>303</td>
<td>1</td>
</tr>
<tr>
<td>SPAN 1</td>
<td>410</td>
<td>1</td>
</tr>
<tr>
<td>PER TYPING</td>
<td>462</td>
<td>2</td>
</tr>
<tr>
<td>WORD PROC</td>
<td>463</td>
<td>2</td>
</tr>
<tr>
<td>APP CERAM</td>
<td>500</td>
<td>2</td>
</tr>
<tr>
<td>BAS CERAM</td>
<td>501</td>
<td>3</td>
</tr>
<tr>
<td>APP WOOD</td>
<td>504</td>
<td>1</td>
</tr>
<tr>
<td>ARCH DWG</td>
<td>506</td>
<td>1</td>
</tr>
<tr>
<td>MECH DWG 2</td>
<td>510</td>
<td>2</td>
</tr>
<tr>
<td>BAS FOODS</td>
<td>553</td>
<td>3</td>
</tr>
<tr>
<td>CH CARE 1</td>
<td>555</td>
<td>2</td>
</tr>
<tr>
<td>STUDIO ART</td>
<td>603</td>
<td>1</td>
</tr>
<tr>
<td>PHOTOG</td>
<td>604</td>
<td>1</td>
</tr>
<tr>
<td>CON CHOIR</td>
<td>655</td>
<td>1</td>
</tr>
<tr>
<td>PE 11-12</td>
<td>700</td>
<td>27</td>
</tr>
<tr>
<td>PE 9-10</td>
<td>701</td>
<td>5</td>
</tr>
<tr>
<td>PE BOCES</td>
<td>702</td>
<td>5</td>
</tr>
<tr>
<td>DR ED TH</td>
<td>750</td>
<td>6</td>
</tr>
<tr>
<td>DR ED TH</td>
<td>751</td>
<td>20</td>
</tr>
<tr>
<td>HEALTH 11</td>
<td>770</td>
<td>12</td>
</tr>
<tr>
<td>PSYCH</td>
<td>790</td>
<td>6</td>
</tr>
<tr>
<td>BOCES PM</td>
<td>901</td>
<td>40</td>
</tr>
</tbody>
</table>
9.3. COURSE TALLIES LIST
<table>
<thead>
<tr>
<th>COURSE CODE</th>
<th>TOTAL</th>
<th>9</th>
<th>10</th>
<th>11</th>
<th>12</th>
</tr>
</thead>
<tbody>
<tr>
<td>ENG 12</td>
<td>91</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>91</td>
</tr>
<tr>
<td>ENG 11-2</td>
<td>78</td>
<td>0</td>
<td>2</td>
<td>68</td>
<td>0</td>
</tr>
<tr>
<td>ENG 11-3</td>
<td>57</td>
<td>0</td>
<td>3</td>
<td>54</td>
<td>17</td>
</tr>
<tr>
<td>ENG 10-2</td>
<td>87</td>
<td>2</td>
<td>83</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>ENG 10-3</td>
<td>57</td>
<td>13</td>
<td>40</td>
<td>4</td>
<td>0</td>
</tr>
<tr>
<td>ENG 9-2</td>
<td>100</td>
<td>98</td>
<td>2</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>ENG 9-3</td>
<td>46</td>
<td>46</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>AP ENGLISH</td>
<td>12</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>12</td>
</tr>
<tr>
<td>WRITING 12</td>
<td>50</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>WRITING 11</td>
<td>51</td>
<td>7</td>
<td>0</td>
<td>5</td>
<td>0</td>
</tr>
<tr>
<td>WRIT LAB</td>
<td>52</td>
<td>60</td>
<td>34</td>
<td>0</td>
<td>3</td>
</tr>
<tr>
<td>WRIT 11-12</td>
<td>53</td>
<td>11</td>
<td>0</td>
<td>1</td>
<td>8</td>
</tr>
<tr>
<td>READING 11</td>
<td>71</td>
<td>9</td>
<td>0</td>
<td>5</td>
<td>0</td>
</tr>
<tr>
<td>READING 10</td>
<td>72</td>
<td>25</td>
<td>21</td>
<td>0</td>
<td>2</td>
</tr>
<tr>
<td>READING 9</td>
<td>73</td>
<td>44</td>
<td>2</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>READ 11-12</td>
<td>74</td>
<td>10</td>
<td>1</td>
<td>8</td>
<td>1</td>
</tr>
<tr>
<td>SS IND STD</td>
<td>100</td>
<td>11</td>
<td>0</td>
<td>0</td>
<td>11</td>
</tr>
<tr>
<td>AM STD 2</td>
<td>101</td>
<td>75</td>
<td>0</td>
<td>69</td>
<td>6</td>
</tr>
<tr>
<td>AM STD 3</td>
<td>102</td>
<td>42</td>
<td>1</td>
<td>32</td>
<td>9</td>
</tr>
<tr>
<td>E CUL ST 2</td>
<td>103</td>
<td>85</td>
<td>2</td>
<td>81</td>
<td>3</td>
</tr>
<tr>
<td>E CUL ST 3</td>
<td>104</td>
<td>56</td>
<td>9</td>
<td>44</td>
<td>3</td>
</tr>
<tr>
<td>AFR-ASIA 2</td>
<td>105</td>
<td>100</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>AFR-ASIA 3</td>
<td>106</td>
<td>52</td>
<td>3</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>LAW 2</td>
<td>107</td>
<td>15</td>
<td>0</td>
<td>2</td>
<td>13</td>
</tr>
<tr>
<td>AMER STD 5</td>
<td>110</td>
<td>18</td>
<td>0</td>
<td>2</td>
<td>16</td>
</tr>
<tr>
<td>MATH 12-1</td>
<td>200</td>
<td>15</td>
<td>0</td>
<td>0</td>
<td>15</td>
</tr>
<tr>
<td>MATH 12-2</td>
<td>201</td>
<td>25</td>
<td>0</td>
<td>8</td>
<td>17</td>
</tr>
<tr>
<td>MATH CRS 3</td>
<td>202</td>
<td>52</td>
<td>8</td>
<td>36</td>
<td>8</td>
</tr>
<tr>
<td>MATH CRS 2</td>
<td>203</td>
<td>56</td>
<td>46</td>
<td>5</td>
<td>0</td>
</tr>
<tr>
<td>MATH CRS 1</td>
<td>204</td>
<td>103</td>
<td>72</td>
<td>27</td>
<td>3</td>
</tr>
<tr>
<td>MATH 9-2</td>
<td>205</td>
<td>47</td>
<td>10</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>GEN MATH</td>
<td>206</td>
<td>59</td>
<td>35</td>
<td>4</td>
<td>0</td>
</tr>
<tr>
<td>C PREP M</td>
<td>207</td>
<td>5</td>
<td>2</td>
<td>2</td>
<td>1</td>
</tr>
<tr>
<td>COMP PRG 4</td>
<td>250</td>
<td>6</td>
<td>0</td>
<td>1</td>
<td>5</td>
</tr>
<tr>
<td>COMP PRG 6</td>
<td>251</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>COMP PRG 2</td>
<td>252</td>
<td>12</td>
<td>2</td>
<td>6</td>
<td>4</td>
</tr>
<tr>
<td>COMP PRG 1</td>
<td>253</td>
<td>15</td>
<td>1</td>
<td>3</td>
<td>1</td>
</tr>
<tr>
<td>COMP-AWARE</td>
<td>255</td>
<td>48</td>
<td>1</td>
<td>14</td>
<td>22</td>
</tr>
<tr>
<td>ADV BIO</td>
<td>300</td>
<td>21</td>
<td>0</td>
<td>15</td>
<td>6</td>
</tr>
<tr>
<td>PHYSICS 2</td>
<td>301</td>
<td>20</td>
<td>0</td>
<td>0</td>
<td>20</td>
</tr>
<tr>
<td>CHEM 2</td>
<td>303</td>
<td>54</td>
<td>0</td>
<td>23</td>
<td>28</td>
</tr>
<tr>
<td>BIOLOGY 2</td>
<td>305</td>
<td>57</td>
<td>0</td>
<td>16</td>
<td>0</td>
</tr>
<tr>
<td>BIOLOGY 3</td>
<td>306</td>
<td>12</td>
<td>1</td>
<td>8</td>
<td>0</td>
</tr>
<tr>
<td>ER SCI R</td>
<td>307</td>
<td>39</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>SCI 9-2</td>
<td>308</td>
<td>64</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>SCI 9-3</td>
<td>309</td>
<td>61</td>
<td>5</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>FRENCH 3</td>
<td>402</td>
<td>12</td>
<td>0</td>
<td>10</td>
<td>0</td>
</tr>
<tr>
<td>FRENCH 2</td>
<td>403</td>
<td>24</td>
<td>0</td>
<td>22</td>
<td>1</td>
</tr>
<tr>
<td>FRENCH 1</td>
<td>404</td>
<td>36</td>
<td>31</td>
<td>5</td>
<td>1</td>
</tr>
<tr>
<td>SPAN 3</td>
<td>407</td>
<td>16</td>
<td>0</td>
<td>7</td>
<td>2</td>
</tr>
<tr>
<td>SPAN 2</td>
<td>408</td>
<td>18</td>
<td>1</td>
<td>16</td>
<td>1</td>
</tr>
<tr>
<td>SPAN 1</td>
<td>410</td>
<td>35</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>OFF PROCED</td>
<td>451</td>
<td>16</td>
<td>0</td>
<td>0</td>
<td>16</td>
</tr>
<tr>
<td>Course</td>
<td>Section</td>
<td>Hours</td>
<td>Credits</td>
<td></td>
<td></td>
</tr>
<tr>
<td>------------------</td>
<td>---------</td>
<td>-------</td>
<td>---------</td>
<td></td>
<td></td>
</tr>
<tr>
<td>BUS LAW</td>
<td>455</td>
<td>8</td>
<td>3</td>
<td></td>
<td></td>
</tr>
<tr>
<td>SHRT/TRANS</td>
<td>457</td>
<td>7</td>
<td>3</td>
<td></td>
<td></td>
</tr>
<tr>
<td>SHRT TH</td>
<td>458</td>
<td>14</td>
<td>2</td>
<td></td>
<td></td>
</tr>
<tr>
<td>BUS DYN</td>
<td>459</td>
<td>19</td>
<td>5</td>
<td></td>
<td></td>
</tr>
<tr>
<td>ADV KEYBD</td>
<td>460</td>
<td>13</td>
<td>3</td>
<td></td>
<td></td>
</tr>
<tr>
<td>PER TYPING</td>
<td>462</td>
<td>29</td>
<td>2</td>
<td></td>
<td></td>
</tr>
<tr>
<td>WORD PROC</td>
<td>463</td>
<td>15</td>
<td>6</td>
<td></td>
<td></td>
</tr>
<tr>
<td>KEY-COM</td>
<td>464</td>
<td>40</td>
<td>1</td>
<td></td>
<td></td>
</tr>
<tr>
<td>APP CERAM</td>
<td>500</td>
<td>25</td>
<td>5</td>
<td></td>
<td></td>
</tr>
<tr>
<td>BAS CERAM</td>
<td>501</td>
<td>74</td>
<td>7</td>
<td></td>
<td></td>
</tr>
<tr>
<td>APP METAL</td>
<td>502</td>
<td>17</td>
<td>2</td>
<td></td>
<td></td>
</tr>
<tr>
<td>BAG METAL</td>
<td>503</td>
<td>38</td>
<td>7</td>
<td></td>
<td></td>
</tr>
<tr>
<td>APP WOOD</td>
<td>504</td>
<td>20</td>
<td>1</td>
<td></td>
<td></td>
</tr>
<tr>
<td>BAS WOOD</td>
<td>505</td>
<td>45</td>
<td>7</td>
<td></td>
<td></td>
</tr>
<tr>
<td>ARCH DWG</td>
<td>506</td>
<td>4</td>
<td>2</td>
<td></td>
<td></td>
</tr>
<tr>
<td>MECH DWG</td>
<td>507</td>
<td>49</td>
<td>5</td>
<td></td>
<td></td>
</tr>
<tr>
<td>MECH DWG 2</td>
<td>510</td>
<td>5</td>
<td>2</td>
<td></td>
<td></td>
</tr>
<tr>
<td>ADV CLOTH</td>
<td>550</td>
<td>11</td>
<td>0</td>
<td></td>
<td></td>
</tr>
<tr>
<td>BAS SEWING</td>
<td>551</td>
<td>15</td>
<td>1</td>
<td></td>
<td></td>
</tr>
<tr>
<td>FAM MEAL</td>
<td>552</td>
<td>9</td>
<td>0</td>
<td></td>
<td></td>
</tr>
<tr>
<td>BAS FOODS</td>
<td>553</td>
<td>17</td>
<td>4</td>
<td></td>
<td></td>
</tr>
<tr>
<td>CH CARE 2</td>
<td>554</td>
<td>14</td>
<td>0</td>
<td></td>
<td></td>
</tr>
<tr>
<td>CH CARE 1</td>
<td>555</td>
<td>16</td>
<td>4</td>
<td></td>
<td></td>
</tr>
<tr>
<td>HOME EC 1</td>
<td>556</td>
<td>3</td>
<td>1</td>
<td></td>
<td></td>
</tr>
<tr>
<td>ART IND ST</td>
<td>600</td>
<td>2</td>
<td>1</td>
<td></td>
<td></td>
</tr>
<tr>
<td>DWP/PTNG</td>
<td>601</td>
<td>10</td>
<td>5</td>
<td></td>
<td></td>
</tr>
<tr>
<td>STUDIO ART</td>
<td>603</td>
<td>30</td>
<td>6</td>
<td></td>
<td></td>
</tr>
<tr>
<td>PHOTO</td>
<td>604</td>
<td>57</td>
<td>21</td>
<td></td>
<td></td>
</tr>
<tr>
<td>YEARBOOK</td>
<td>605</td>
<td>10</td>
<td>8</td>
<td></td>
<td></td>
</tr>
<tr>
<td>CON BAND</td>
<td>654</td>
<td>57</td>
<td>21</td>
<td></td>
<td></td>
</tr>
<tr>
<td>CON CHOIR</td>
<td>655</td>
<td>15</td>
<td>5</td>
<td></td>
<td></td>
</tr>
<tr>
<td>HB CHOIR</td>
<td>656</td>
<td>14</td>
<td>2</td>
<td></td>
<td></td>
</tr>
<tr>
<td>PE 11-12</td>
<td>700</td>
<td>156</td>
<td>77</td>
<td></td>
<td></td>
</tr>
<tr>
<td>PE 9-10</td>
<td>701</td>
<td>294</td>
<td>15</td>
<td></td>
<td></td>
</tr>
<tr>
<td>FE BOCES</td>
<td>702</td>
<td>40</td>
<td>15</td>
<td></td>
<td></td>
</tr>
<tr>
<td>DR ED TH</td>
<td>750</td>
<td>73</td>
<td>21</td>
<td></td>
<td></td>
</tr>
<tr>
<td>DR ED TH</td>
<td>751</td>
<td>31</td>
<td>9</td>
<td></td>
<td></td>
</tr>
<tr>
<td>HEALTH 11</td>
<td>770</td>
<td>101</td>
<td>18</td>
<td></td>
<td></td>
</tr>
<tr>
<td>HEALTH 11</td>
<td>772</td>
<td>38</td>
<td>23</td>
<td></td>
<td></td>
</tr>
<tr>
<td>PSYCH</td>
<td>790</td>
<td>25</td>
<td>4</td>
<td></td>
<td></td>
</tr>
<tr>
<td>BOCES AM</td>
<td>980</td>
<td>52</td>
<td>17</td>
<td></td>
<td></td>
</tr>
<tr>
<td>BOCES PM</td>
<td>981</td>
<td>40</td>
<td>10</td>
<td></td>
<td></td>
</tr>
<tr>
<td>LUNCH 7B</td>
<td>999</td>
<td>0</td>
<td>0</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
9.4. COURSE LIST
## COURSE LIST - SODUS

<table>
<thead>
<tr>
<th>CODE</th>
<th>SEC</th>
<th>SEATS</th>
<th>SEM</th>
<th>FD</th>
<th>DAYS</th>
<th>MET</th>
<th>DESCRIPTION</th>
<th>ROOM</th>
<th>TCH</th>
<th>PER</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>1</td>
<td>200</td>
<td>3</td>
<td>11</td>
<td>MTWF</td>
<td></td>
<td>ENG 12</td>
<td>N</td>
<td>1</td>
<td>8</td>
</tr>
<tr>
<td>2</td>
<td>1</td>
<td>35</td>
<td>3</td>
<td>5</td>
<td>MTWF</td>
<td></td>
<td>ENG 11-2</td>
<td>125</td>
<td>5</td>
<td>5</td>
</tr>
<tr>
<td>2</td>
<td>2</td>
<td>35</td>
<td>3</td>
<td>7</td>
<td>MTWF</td>
<td></td>
<td>ENG 11-2</td>
<td>125</td>
<td>1</td>
<td>6</td>
</tr>
<tr>
<td>2</td>
<td>3</td>
<td>35</td>
<td>3</td>
<td>9</td>
<td>MTWF</td>
<td></td>
<td>ENG 11-2</td>
<td>125</td>
<td>8</td>
<td>7</td>
</tr>
<tr>
<td>2</td>
<td>4</td>
<td>35</td>
<td>3</td>
<td>13</td>
<td>MTWF</td>
<td></td>
<td>ENG 11-2</td>
<td>125</td>
<td>2</td>
<td>10</td>
</tr>
<tr>
<td>3</td>
<td>1</td>
<td>35</td>
<td>3</td>
<td>4</td>
<td>MTWF</td>
<td></td>
<td>ENG 11-3</td>
<td>125</td>
<td>8</td>
<td>4</td>
</tr>
<tr>
<td>3</td>
<td>2</td>
<td>35</td>
<td>3</td>
<td>3</td>
<td>MTWF</td>
<td></td>
<td>ENG 11-3</td>
<td>125</td>
<td>8</td>
<td>3</td>
</tr>
<tr>
<td>4</td>
<td>1</td>
<td>35</td>
<td>3</td>
<td>12</td>
<td>MTWF</td>
<td></td>
<td>ENG 11-3</td>
<td>125</td>
<td>6</td>
<td>9</td>
</tr>
<tr>
<td>4</td>
<td>2</td>
<td>35</td>
<td>3</td>
<td>13</td>
<td>MTWF</td>
<td></td>
<td>ENG 10-2</td>
<td>129</td>
<td>4</td>
<td>6</td>
</tr>
<tr>
<td>4</td>
<td>3</td>
<td>35</td>
<td>3</td>
<td>9</td>
<td>MTWF</td>
<td></td>
<td>ENG 10-2</td>
<td>129</td>
<td>9</td>
<td>7</td>
</tr>
<tr>
<td>4</td>
<td>4</td>
<td>35</td>
<td>3</td>
<td>13</td>
<td>MTWF</td>
<td></td>
<td>ENG 10-2</td>
<td>129</td>
<td>9</td>
<td>10</td>
</tr>
<tr>
<td>5</td>
<td>1</td>
<td>35</td>
<td>3</td>
<td>5</td>
<td>MTWF</td>
<td></td>
<td>ENG 10-3</td>
<td>220</td>
<td>3</td>
<td>5</td>
</tr>
<tr>
<td>5</td>
<td>2</td>
<td>35</td>
<td>3</td>
<td>3</td>
<td>MTWF</td>
<td></td>
<td>ENG 10-3</td>
<td>129</td>
<td>9</td>
<td>3</td>
</tr>
<tr>
<td>5</td>
<td>3</td>
<td>35</td>
<td>3</td>
<td>12</td>
<td>MTWF</td>
<td></td>
<td>ENG 10-3</td>
<td>129</td>
<td>9</td>
<td>9</td>
</tr>
<tr>
<td>6</td>
<td>1</td>
<td>35</td>
<td>3</td>
<td>3</td>
<td>MTWF</td>
<td></td>
<td>ENG 9-2</td>
<td>220</td>
<td>10</td>
<td>3</td>
</tr>
<tr>
<td>6</td>
<td>2</td>
<td>35</td>
<td>3</td>
<td>7</td>
<td>MTWF</td>
<td></td>
<td>ENG 9-2</td>
<td>220</td>
<td>10</td>
<td>6</td>
</tr>
<tr>
<td>6</td>
<td>3</td>
<td>35</td>
<td>3</td>
<td>8</td>
<td>MTWF</td>
<td></td>
<td>ENG 9-2</td>
<td>220</td>
<td>10</td>
<td>6</td>
</tr>
<tr>
<td>6</td>
<td>4</td>
<td>35</td>
<td>3</td>
<td>13</td>
<td>MTWF</td>
<td></td>
<td>ENG 9-2</td>
<td>219</td>
<td>5</td>
<td>10</td>
</tr>
<tr>
<td>6</td>
<td>5</td>
<td>35</td>
<td>3</td>
<td>12</td>
<td>MTWF</td>
<td></td>
<td>ENG 9-2</td>
<td>220</td>
<td>5</td>
<td>9</td>
</tr>
<tr>
<td>6</td>
<td>5</td>
<td>35</td>
<td>3</td>
<td>9</td>
<td>MTWF</td>
<td></td>
<td>ENG 9-2</td>
<td>220</td>
<td>3</td>
<td>7</td>
</tr>
<tr>
<td>7</td>
<td>1</td>
<td>35</td>
<td>3</td>
<td>5</td>
<td>MTWF</td>
<td></td>
<td>ENG 9-3</td>
<td>129</td>
<td>4</td>
<td>5</td>
</tr>
<tr>
<td>7</td>
<td>2</td>
<td>35</td>
<td>3</td>
<td>4</td>
<td>MTWF</td>
<td></td>
<td>ENG 9-3</td>
<td>220</td>
<td>10</td>
<td>4</td>
</tr>
<tr>
<td>7</td>
<td>3</td>
<td>35</td>
<td>3</td>
<td>13</td>
<td>MTWF</td>
<td></td>
<td>ENG 9-3</td>
<td>123</td>
<td>4</td>
<td>10</td>
</tr>
<tr>
<td>9</td>
<td>1</td>
<td>20</td>
<td>3</td>
<td>11</td>
<td>MTWF</td>
<td></td>
<td>AP ENGLISH</td>
<td>129</td>
<td>9</td>
<td>8</td>
</tr>
</tbody>
</table>
9.5. RESULTS - TRIAL AND FINAL RUN
RUN - FINAL

RESULTS FOR GRADE 9

<table>
<thead>
<tr>
<th>FULL SCHEDULES</th>
<th>PARTIAL SCHEDULES</th>
<th>IRR CONFLICTS</th>
<th>TOTAL</th>
</tr>
</thead>
<tbody>
<tr>
<td>157</td>
<td>0</td>
<td>0</td>
<td>157</td>
</tr>
</tbody>
</table>

STUDY HALL COUNT

1ST SEM SCHEDULE

<p>| | | | | | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>2</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>3</td>
<td>6</td>
<td>35</td>
<td>8</td>
<td>29</td>
<td>35</td>
</tr>
<tr>
<td>4</td>
<td>16</td>
<td>16</td>
<td>16</td>
<td>16</td>
<td>16</td>
</tr>
<tr>
<td>5</td>
<td>29</td>
<td>60</td>
<td>29</td>
<td>60</td>
<td>29</td>
</tr>
<tr>
<td>6</td>
<td>48</td>
<td>79</td>
<td>48</td>
<td>79</td>
<td>48</td>
</tr>
<tr>
<td>7</td>
<td>46</td>
<td>30</td>
<td>46</td>
<td>30</td>
<td>47</td>
</tr>
<tr>
<td>8</td>
<td>40</td>
<td>24</td>
<td>40</td>
<td>24</td>
<td>41</td>
</tr>
<tr>
<td>9</td>
<td>31</td>
<td>30</td>
<td>31</td>
<td>30</td>
<td>34</td>
</tr>
<tr>
<td>10</td>
<td>10</td>
<td>9</td>
<td>10</td>
<td>9</td>
<td>13</td>
</tr>
<tr>
<td>11</td>
<td>1</td>
<td>30</td>
<td>1</td>
<td>30</td>
<td>30</td>
</tr>
<tr>
<td>12</td>
<td>47</td>
<td>8</td>
<td>47</td>
<td>14</td>
<td>8</td>
</tr>
<tr>
<td>13</td>
<td>17</td>
<td>22</td>
<td>17</td>
<td>17</td>
<td>22</td>
</tr>
</tbody>
</table>

2ND SEM SCHEDULE

<p>| | | | | | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>2</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>3</td>
<td>4</td>
<td>35</td>
<td>4</td>
<td>27</td>
<td>10</td>
</tr>
<tr>
<td>4</td>
<td>18</td>
<td>18</td>
<td>16</td>
<td>18</td>
<td>12</td>
</tr>
<tr>
<td>5</td>
<td>23</td>
<td>54</td>
<td>23</td>
<td>54</td>
<td>55</td>
</tr>
<tr>
<td>6</td>
<td>42</td>
<td>73</td>
<td>42</td>
<td>73</td>
<td>74</td>
</tr>
<tr>
<td>7</td>
<td>57</td>
<td>41</td>
<td>57</td>
<td>41</td>
<td>41</td>
</tr>
<tr>
<td>8</td>
<td>51</td>
<td>35</td>
<td>51</td>
<td>35</td>
<td>35</td>
</tr>
<tr>
<td>9</td>
<td>31</td>
<td>30</td>
<td>31</td>
<td>30</td>
<td>30</td>
</tr>
<tr>
<td>10</td>
<td>10</td>
<td>9</td>
<td>10</td>
<td>9</td>
<td>9</td>
</tr>
<tr>
<td>11</td>
<td>1</td>
<td>30</td>
<td>1</td>
<td>30</td>
<td>1</td>
</tr>
<tr>
<td>12</td>
<td>48</td>
<td>8</td>
<td>48</td>
<td>15</td>
<td>50</td>
</tr>
<tr>
<td>13</td>
<td>16</td>
<td>21</td>
<td>16</td>
<td>16</td>
<td>21</td>
</tr>
</tbody>
</table>

NORTIER STEPHANIE

PARTIAL SCHEDULE - COURSE 103 IS NOT SCHEDULED

1ST SEM SCHEDULE

<p>| | | | | | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>2</td>
<td>654</td>
<td>0</td>
<td>654</td>
<td>0</td>
<td>654</td>
</tr>
<tr>
<td>3</td>
<td>453</td>
<td>453</td>
<td>453</td>
<td>453</td>
<td>453</td>
</tr>
<tr>
<td>4</td>
<td>4</td>
<td>4</td>
<td>4</td>
<td>4</td>
<td>4</td>
</tr>
<tr>
<td>5</td>
<td>999</td>
<td>999</td>
<td>999</td>
<td>999</td>
<td>999</td>
</tr>
<tr>
<td>6</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>7</td>
<td>403</td>
<td>403</td>
<td>403</td>
<td>403</td>
<td>403</td>
</tr>
<tr>
<td>8</td>
<td>403</td>
<td>403</td>
<td>403</td>
<td>403</td>
<td>403</td>
</tr>
<tr>
<td>9</td>
<td>203</td>
<td>203</td>
<td>203</td>
<td>203</td>
<td>203</td>
</tr>
<tr>
<td>10</td>
<td>203</td>
<td>203</td>
<td>203</td>
<td>203</td>
<td>203</td>
</tr>
<tr>
<td>11</td>
<td>305</td>
<td>305</td>
<td>305</td>
<td>305</td>
<td>305</td>
</tr>
<tr>
<td>12</td>
<td>700</td>
<td>305</td>
<td>700</td>
<td>305</td>
<td>0</td>
</tr>
<tr>
<td>13</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
</tbody>
</table>
### 2ND SEM SCHEDULE
```
<table>
<thead>
<tr>
<th></th>
<th>0</th>
<th>0</th>
<th>0</th>
<th>0</th>
<th>0</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>2</td>
<td>654</td>
<td></td>
<td>654</td>
<td></td>
<td>654</td>
</tr>
<tr>
<td>3</td>
<td>453</td>
<td>453</td>
<td>453</td>
<td>453</td>
<td>453</td>
</tr>
<tr>
<td>4</td>
<td>4</td>
<td></td>
<td>4</td>
<td></td>
<td>4</td>
</tr>
<tr>
<td>5</td>
<td>999</td>
<td>999</td>
<td>999</td>
<td>999</td>
<td>999</td>
</tr>
<tr>
<td>6</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>7</td>
<td>403</td>
<td>403</td>
<td>403</td>
<td>403</td>
<td>403</td>
</tr>
<tr>
<td>8</td>
<td>403</td>
<td>403</td>
<td>403</td>
<td>403</td>
<td>403</td>
</tr>
<tr>
<td>9</td>
<td>203</td>
<td>203</td>
<td>203</td>
<td>203</td>
<td>203</td>
</tr>
<tr>
<td>10</td>
<td>305</td>
<td>305</td>
<td>305</td>
<td>305</td>
<td>305</td>
</tr>
<tr>
<td>11</td>
<td>700</td>
<td>700</td>
<td>700</td>
<td>700</td>
<td>700</td>
</tr>
<tr>
<td>12</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>13</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
```

### RHINE DAVID

**PARTIAL SCHEDULE - COURSE 403 IS NOT SCHEDULED**

### 1ST SEM SCHEDULE
```
<table>
<thead>
<tr>
<th></th>
<th>0</th>
<th>0</th>
<th>0</th>
<th>0</th>
<th>0</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>2</td>
<td>654</td>
<td></td>
<td>654</td>
<td></td>
<td>654</td>
</tr>
<tr>
<td>3</td>
<td>103</td>
<td>103</td>
<td>103</td>
<td>103</td>
<td>103</td>
</tr>
<tr>
<td>4</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>5</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>6</td>
<td>999</td>
<td>999</td>
<td>999</td>
<td>999</td>
<td>999</td>
</tr>
<tr>
<td>7</td>
<td>4</td>
<td></td>
<td>4</td>
<td></td>
<td>4</td>
</tr>
<tr>
<td>8</td>
<td>4</td>
<td></td>
<td>4</td>
<td></td>
<td>4</td>
</tr>
<tr>
<td>9</td>
<td>203</td>
<td>203</td>
<td>203</td>
<td>203</td>
<td>203</td>
</tr>
<tr>
<td>10</td>
<td>203</td>
<td>203</td>
<td>203</td>
<td>203</td>
<td>203</td>
</tr>
<tr>
<td>11</td>
<td>305</td>
<td>305</td>
<td>305</td>
<td>305</td>
<td>305</td>
</tr>
<tr>
<td>12</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>13</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
```

### 2ND SEM SCHEDULE
```
<table>
<thead>
<tr>
<th></th>
<th>0</th>
<th>0</th>
<th>0</th>
<th>0</th>
<th>0</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>2</td>
<td>654</td>
<td></td>
<td>654</td>
<td></td>
<td>654</td>
</tr>
<tr>
<td>3</td>
<td>103</td>
<td>103</td>
<td>103</td>
<td>103</td>
<td>103</td>
</tr>
<tr>
<td>4</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>5</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>6</td>
<td>999</td>
<td>999</td>
<td>999</td>
<td>999</td>
<td>999</td>
</tr>
<tr>
<td>7</td>
<td>4</td>
<td></td>
<td>4</td>
<td></td>
<td>4</td>
</tr>
<tr>
<td>8</td>
<td>4</td>
<td></td>
<td>4</td>
<td></td>
<td>4</td>
</tr>
<tr>
<td>9</td>
<td>203</td>
<td>203</td>
<td>203</td>
<td>203</td>
<td>203</td>
</tr>
<tr>
<td>10</td>
<td>203</td>
<td>203</td>
<td>203</td>
<td>203</td>
<td>203</td>
</tr>
<tr>
<td>11</td>
<td>305</td>
<td>305</td>
<td>305</td>
<td>305</td>
<td>305</td>
</tr>
<tr>
<td>12</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>13</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
```
PARTIAL SCHEDULE - COURSE 303 IS NOT SCHEDULED

1ST SEM SCHEDULE

<p>| | | | | | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>2</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>3</td>
<td>103</td>
<td>103</td>
<td>103</td>
<td>103</td>
<td>103</td>
</tr>
<tr>
<td>4</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>5</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>6</td>
<td>999</td>
<td>999</td>
<td>999</td>
<td>999</td>
<td>999</td>
</tr>
<tr>
<td>7</td>
<td>403</td>
<td>403</td>
<td>403</td>
<td>403</td>
<td>403</td>
</tr>
<tr>
<td>8</td>
<td>403</td>
<td>403</td>
<td>403</td>
<td>403</td>
<td>403</td>
</tr>
<tr>
<td>9</td>
<td>204</td>
<td>204</td>
<td>204</td>
<td>204</td>
<td>204</td>
</tr>
<tr>
<td>10</td>
<td>204</td>
<td>204</td>
<td>204</td>
<td>204</td>
<td>204</td>
</tr>
<tr>
<td>11</td>
<td>701</td>
<td>0</td>
<td>701</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>12</td>
<td>603</td>
<td>603</td>
<td>603</td>
<td>603</td>
<td>603</td>
</tr>
<tr>
<td>13</td>
<td>4</td>
<td>4</td>
<td>4</td>
<td>4</td>
<td>4</td>
</tr>
</tbody>
</table>

2ND SEM SCHEDULE

<p>| | | | | | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>2</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>3</td>
<td>103</td>
<td>103</td>
<td>103</td>
<td>103</td>
<td>103</td>
</tr>
<tr>
<td>4</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>5</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>6</td>
<td>999</td>
<td>999</td>
<td>999</td>
<td>999</td>
<td>999</td>
</tr>
<tr>
<td>7</td>
<td>403</td>
<td>403</td>
<td>403</td>
<td>403</td>
<td>403</td>
</tr>
<tr>
<td>8</td>
<td>403</td>
<td>403</td>
<td>403</td>
<td>403</td>
<td>403</td>
</tr>
<tr>
<td>9</td>
<td>204</td>
<td>204</td>
<td>204</td>
<td>204</td>
<td>204</td>
</tr>
<tr>
<td>10</td>
<td>204</td>
<td>204</td>
<td>204</td>
<td>204</td>
<td>204</td>
</tr>
<tr>
<td>11</td>
<td>701</td>
<td>0</td>
<td>701</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>12</td>
<td>603</td>
<td>603</td>
<td>603</td>
<td>603</td>
<td>603</td>
</tr>
<tr>
<td>13</td>
<td>4</td>
<td>4</td>
<td>4</td>
<td>4</td>
<td>4</td>
</tr>
</tbody>
</table>

RUN - FINAL

RESULTS FOR GRADE 10

<p>| | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>FULL SCHEDULES</td>
<td>124</td>
</tr>
<tr>
<td>PARTIAL SCHEDULES</td>
<td>3</td>
</tr>
<tr>
<td>IRR CONFLICTS</td>
<td>0</td>
</tr>
<tr>
<td>TOTAL</td>
<td>127</td>
</tr>
</tbody>
</table>

STUDY HALL COUNT

1ST SEM SCHEDULE

<p>| | | | | | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>2</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>3</td>
<td>4</td>
<td>16</td>
<td>4</td>
<td>16</td>
<td>16</td>
</tr>
<tr>
<td>4</td>
<td>23</td>
<td>15</td>
<td>23</td>
<td>15</td>
<td>24</td>
</tr>
<tr>
<td>5</td>
<td>25</td>
<td>21</td>
<td>25</td>
<td>21</td>
<td>21</td>
</tr>
<tr>
<td>6</td>
<td>25</td>
<td>21</td>
<td>25</td>
<td>21</td>
<td>21</td>
</tr>
<tr>
<td>7</td>
<td>23</td>
<td>23</td>
<td>23</td>
<td>23</td>
<td>28</td>
</tr>
<tr>
<td>8</td>
<td>27</td>
<td>27</td>
<td>27</td>
<td>27</td>
<td>32</td>
</tr>
<tr>
<td>9</td>
<td>30</td>
<td>29</td>
<td>30</td>
<td>29</td>
<td>36</td>
</tr>
<tr>
<td>10</td>
<td>29</td>
<td>28</td>
<td>29</td>
<td>28</td>
<td>35</td>
</tr>
<tr>
<td>11</td>
<td>8</td>
<td>34</td>
<td>8</td>
<td>34</td>
<td>34</td>
</tr>
<tr>
<td>12</td>
<td>32</td>
<td>16</td>
<td>32</td>
<td>19</td>
<td>45</td>
</tr>
<tr>
<td>13</td>
<td>33</td>
<td>44</td>
<td>33</td>
<td>33</td>
<td>44</td>
</tr>
</tbody>
</table>
2ND SEM SCHEDULE
1 0 0 0 0 0
2 0 0 0 0 0
3 4 17 4 17 4
4 22 14 22 14 16
5 25 20 25 20 22
6 25 20 25 20 22
7 26 24 26 24 24
8 30 28 30 28 28
9 33 32 33 32 32
10 32 31 32 31 31
11 8 34 8 34 8
12 31 17 31 20 50
13 31 42 31 31 42

BURNAP KIRSTEN

PARTIAL SCHEDULE - COURSE 101 IS NOT SCHEDULED

1ST SEM SCHEDULE
1 0 0 0 0 0
2 654 0 654 0 654
3 253 253 253 253 253
4 303 700 303 700 0
5 303 303 303 303 303
6 303 303 303 303 303
7 2 2 2 2 2
8 2 2 2 2 2
9 999 999 999 999 999
10 0 0 0 0 0
11 0 0 0 0 0
12 0 770 0 770 770
13 202 202 202 202 202

2ND SEM SCHEDULE
1 0 0 0 0 0
2 654 0 654 0 654
3 253 253 253 253 253
4 303 700 303 700 700
5 303 303 303 303 303
6 303 303 303 303 303
7 2 2 2 2 2
8 2 2 2 2 2
9 999 999 999 999 999
10 0 0 0 0 0
11 0 0 0 0 0
12 0 0 0 0 0
13 202 202 202 202 202
<table>
<thead>
<tr>
<th></th>
<th>1ST SEM SCHEDULE</th>
<th>2ND SEM SCHEDULE</th>
<th>DEPAUL KENNETH</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td>PARTIAL SCHEDULE - COURSE 999 IS NOT SCHEDULED</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>1ST SEM SCHEDULE</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>2</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>3</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>4</td>
<td>0</td>
<td>700</td>
<td>0</td>
</tr>
<tr>
<td>5</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>6</td>
<td>999</td>
<td>999</td>
<td>999</td>
</tr>
<tr>
<td>7</td>
<td>202</td>
<td>202</td>
<td>202</td>
</tr>
<tr>
<td>8</td>
<td>202</td>
<td>202</td>
<td>202</td>
</tr>
<tr>
<td>9</td>
<td>2</td>
<td>2</td>
<td>2</td>
</tr>
<tr>
<td>10</td>
<td>2</td>
<td>2</td>
<td>2</td>
</tr>
<tr>
<td>11</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>12</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>13</td>
<td>101</td>
<td>101</td>
<td>101</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th></th>
<th>PARTIAL SCHEDULE COURSE 303 IS NOT SCHEDULED</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>1ST SEM SCHEDULE</td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>2</td>
<td>0</td>
</tr>
<tr>
<td>3</td>
<td>101</td>
</tr>
<tr>
<td>4</td>
<td>0</td>
</tr>
<tr>
<td>5</td>
<td>0</td>
</tr>
<tr>
<td>6</td>
<td>0</td>
</tr>
<tr>
<td>7</td>
<td>202</td>
</tr>
<tr>
<td>8</td>
<td>202</td>
</tr>
<tr>
<td>9</td>
<td>2</td>
</tr>
<tr>
<td>10</td>
<td>2</td>
</tr>
<tr>
<td>11</td>
<td>0</td>
</tr>
<tr>
<td>12</td>
<td>0</td>
</tr>
<tr>
<td>13</td>
<td>305</td>
</tr>
</tbody>
</table>
### 2ND SEM SCHEDULE

<p>| | | | | | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>2</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>3</td>
<td>101</td>
<td>101</td>
<td>101</td>
<td>101</td>
<td>101</td>
</tr>
<tr>
<td>4</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>5</td>
<td>0</td>
<td>700</td>
<td>750</td>
<td>700</td>
<td>700</td>
</tr>
<tr>
<td>6</td>
<td>0</td>
<td>700</td>
<td>750</td>
<td>700</td>
<td>700</td>
</tr>
<tr>
<td>7</td>
<td>202</td>
<td>202</td>
<td>202</td>
<td>202</td>
<td>202</td>
</tr>
<tr>
<td>8</td>
<td>202</td>
<td>202</td>
<td>202</td>
<td>202</td>
<td>202</td>
</tr>
<tr>
<td>9</td>
<td>2</td>
<td>2</td>
<td>2</td>
<td>2</td>
<td>2</td>
</tr>
<tr>
<td>10</td>
<td>2</td>
<td>2</td>
<td>2</td>
<td>2</td>
<td>2</td>
</tr>
<tr>
<td>11</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>12</td>
<td>305</td>
<td>0</td>
<td>305</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>13</td>
<td>305</td>
<td>305</td>
<td>305</td>
<td>305</td>
<td>305</td>
</tr>
</tbody>
</table>

### RUN - FINAL RESULTS FOR GRADE 11

- **FULL SCHEDULES**: 106
- **PARTIAL SCHEDULES**: 3
- **IRR CONFLICTS**: 0
- **TOTAL**: 109

### STUDY HALL COUNT

#### 1ST SEM SCHEDULE

<p>| | | | | | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>2</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>3</td>
<td>15</td>
<td>6</td>
<td>15</td>
<td>6</td>
<td>6</td>
</tr>
<tr>
<td>4</td>
<td>38</td>
<td>27</td>
<td>38</td>
<td>27</td>
<td>47</td>
</tr>
<tr>
<td>5</td>
<td>16</td>
<td>4</td>
<td>16</td>
<td>4</td>
<td>16</td>
</tr>
<tr>
<td>6</td>
<td>29</td>
<td>17</td>
<td>29</td>
<td>17</td>
<td>29</td>
</tr>
<tr>
<td>7</td>
<td>15</td>
<td>20</td>
<td>15</td>
<td>20</td>
<td>15</td>
</tr>
<tr>
<td>8</td>
<td>0</td>
<td>5</td>
<td>0</td>
<td>5</td>
<td>0</td>
</tr>
<tr>
<td>9</td>
<td>26</td>
<td>22</td>
<td>26</td>
<td>22</td>
<td>26</td>
</tr>
<tr>
<td>10</td>
<td>27</td>
<td>23</td>
<td>27</td>
<td>23</td>
<td>27</td>
</tr>
<tr>
<td>11</td>
<td>50</td>
<td>53</td>
<td>50</td>
<td>53</td>
<td>53</td>
</tr>
<tr>
<td>12</td>
<td>17</td>
<td>15</td>
<td>17</td>
<td>15</td>
<td>23</td>
</tr>
<tr>
<td>13</td>
<td>8</td>
<td>8</td>
<td>8</td>
<td>8</td>
<td>8</td>
</tr>
</tbody>
</table>

#### 2ND SEM SCHEDULE

<p>| | | | | | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>2</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>3</td>
<td>15</td>
<td>18</td>
<td>15</td>
<td>18</td>
<td>15</td>
</tr>
<tr>
<td>4</td>
<td>58</td>
<td>27</td>
<td>58</td>
<td>27</td>
<td>44</td>
</tr>
<tr>
<td>5</td>
<td>34</td>
<td>4</td>
<td>29</td>
<td>4</td>
<td>4</td>
</tr>
<tr>
<td>6</td>
<td>47</td>
<td>17</td>
<td>42</td>
<td>17</td>
<td>17</td>
</tr>
<tr>
<td>7</td>
<td>21</td>
<td>20</td>
<td>21</td>
<td>20</td>
<td>20</td>
</tr>
<tr>
<td>8</td>
<td>6</td>
<td>5</td>
<td>6</td>
<td>5</td>
<td>5</td>
</tr>
<tr>
<td>9</td>
<td>26</td>
<td>22</td>
<td>26</td>
<td>22</td>
<td>22</td>
</tr>
<tr>
<td>10</td>
<td>27</td>
<td>23</td>
<td>27</td>
<td>23</td>
<td>23</td>
</tr>
<tr>
<td>11</td>
<td>49</td>
<td>52</td>
<td>49</td>
<td>52</td>
<td>49</td>
</tr>
<tr>
<td>12</td>
<td>16</td>
<td>28</td>
<td>16</td>
<td>28</td>
<td>25</td>
</tr>
<tr>
<td>13</td>
<td>8</td>
<td>8</td>
<td>8</td>
<td>8</td>
<td>8</td>
</tr>
</tbody>
</table>
### HUBER RICHARD

**PARTIAL SCHEDULE - COURSE 900 IS NOT SCHEDULED**

#### 1ST SEM SCHEDULE

<p>| | | | | | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>110</td>
<td>110</td>
<td>110</td>
<td>110</td>
<td>110</td>
</tr>
<tr>
<td>2</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>3</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>4</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>5</td>
<td>999</td>
<td>999</td>
<td>999</td>
<td>999</td>
<td>999</td>
</tr>
<tr>
<td>6</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>7</td>
<td>0</td>
<td>/01</td>
<td>0</td>
<td>701</td>
<td>0</td>
</tr>
<tr>
<td>8</td>
<td>0</td>
<td>/01</td>
<td>0</td>
<td>701</td>
<td>0</td>
</tr>
<tr>
<td>9</td>
<td>504</td>
<td>504</td>
<td>504</td>
<td>504</td>
<td>504</td>
</tr>
<tr>
<td>10</td>
<td>504</td>
<td>504</td>
<td>504</td>
<td>504</td>
<td>504</td>
</tr>
<tr>
<td>11</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>12</td>
<td>3</td>
<td>3</td>
<td>3</td>
<td>3</td>
<td>3</td>
</tr>
<tr>
<td>13</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
</tbody>
</table>

#### 2ND SEM SCHEDULE

<p>| | | | | | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>110</td>
<td>110</td>
<td>110</td>
<td>110</td>
<td>110</td>
</tr>
<tr>
<td>2</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>3</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>4</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>5</td>
<td>999</td>
<td>999</td>
<td>999</td>
<td>999</td>
<td>999</td>
</tr>
<tr>
<td>6</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>7</td>
<td>0</td>
<td>701</td>
<td>0</td>
<td>701</td>
<td>701</td>
</tr>
<tr>
<td>8</td>
<td>0</td>
<td>/01</td>
<td>0</td>
<td>701</td>
<td>701</td>
</tr>
<tr>
<td>9</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>10</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>11</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>12</td>
<td>3</td>
<td>3</td>
<td>3</td>
<td>3</td>
<td>3</td>
</tr>
<tr>
<td>13</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
</tbody>
</table>

### MOON KIM

**PARTIAL SCHEDULE - COURSE 999 IS NOT SCHEDULED**

#### 1ST SEM SCHEDULE

<p>| | | | | | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>2</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>3</td>
<td>457</td>
<td>457</td>
<td>457</td>
<td>457</td>
<td>457</td>
</tr>
<tr>
<td>4</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>5</td>
<td>0</td>
<td>/00</td>
<td>0</td>
<td>700</td>
<td>0</td>
</tr>
<tr>
<td>6</td>
<td>0</td>
<td>700</td>
<td>0</td>
<td>700</td>
<td>0</td>
</tr>
<tr>
<td>7</td>
<td>463</td>
<td>463</td>
<td>463</td>
<td>463</td>
<td>463</td>
</tr>
<tr>
<td>8</td>
<td>463</td>
<td>463</td>
<td>463</td>
<td>463</td>
<td>463</td>
</tr>
<tr>
<td>9</td>
<td>2</td>
<td>2</td>
<td>2</td>
<td>2</td>
<td>2</td>
</tr>
<tr>
<td>10</td>
<td>2</td>
<td>2</td>
<td>2</td>
<td>2</td>
<td>2</td>
</tr>
<tr>
<td>11</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>12</td>
<td>303</td>
<td>0</td>
<td>303</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>13</td>
<td>303</td>
<td>303</td>
<td>303</td>
<td>303</td>
<td>303</td>
</tr>
<tr>
<td>2ND SEM SCHEDULE</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>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>2</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>3</td>
<td>457</td>
<td>457</td>
<td>457</td>
<td>457</td>
<td>457</td>
</tr>
<tr>
<td>4</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>5</td>
<td>0</td>
<td>700</td>
<td>750</td>
<td>700</td>
<td>700</td>
</tr>
<tr>
<td>6</td>
<td>0</td>
<td>700</td>
<td>750</td>
<td>700</td>
<td>700</td>
</tr>
<tr>
<td>7</td>
<td>463</td>
<td>463</td>
<td>463</td>
<td>463</td>
<td>463</td>
</tr>
<tr>
<td>8</td>
<td>463</td>
<td>463</td>
<td>463</td>
<td>463</td>
<td>463</td>
</tr>
<tr>
<td>9</td>
<td>2</td>
<td>2</td>
<td>2</td>
<td>2</td>
<td>2</td>
</tr>
<tr>
<td>10</td>
<td>2</td>
<td>2</td>
<td>2</td>
<td>2</td>
<td>2</td>
</tr>
<tr>
<td>11</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>12</td>
<td>303</td>
<td>0</td>
<td>303</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>13</td>
<td>303</td>
<td>303</td>
<td>303</td>
<td>303</td>
<td>303</td>
</tr>
</tbody>
</table>

0 NEIL ALEXANDAR

PARTIAL SCHEDULE - COURSE 999 IS NOT SCHEDULED

<table>
<thead>
<tr>
<th>1ST SEM SCHEDULE</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
</tr>
<tr>
<td>2</td>
</tr>
<tr>
<td>3</td>
</tr>
<tr>
<td>4</td>
</tr>
<tr>
<td>5</td>
</tr>
<tr>
<td>6</td>
</tr>
<tr>
<td>7</td>
</tr>
<tr>
<td>8</td>
</tr>
<tr>
<td>9</td>
</tr>
<tr>
<td>10</td>
</tr>
<tr>
<td>11</td>
</tr>
<tr>
<td>12</td>
</tr>
<tr>
<td>13</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>2ND SEM SCHEDULE</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
</tr>
<tr>
<td>2</td>
</tr>
<tr>
<td>3</td>
</tr>
<tr>
<td>4</td>
</tr>
<tr>
<td>5</td>
</tr>
<tr>
<td>6</td>
</tr>
<tr>
<td>7</td>
</tr>
<tr>
<td>8</td>
</tr>
<tr>
<td>9</td>
</tr>
<tr>
<td>10</td>
</tr>
<tr>
<td>11</td>
</tr>
<tr>
<td>12</td>
</tr>
<tr>
<td>13</td>
</tr>
</tbody>
</table>
SMITH PATRICIA

PARTIAL SCHEDULE - COURSE 999 IS NOT SCHEDULED

1ST SEM SCHEDULE

<table>
<thead>
<tr>
<th></th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
<th>6</th>
<th>7</th>
<th>8</th>
<th>9</th>
<th>10</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>2</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>3</td>
<td>0</td>
<td>770</td>
<td>0</td>
<td>770</td>
<td>770</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>4</td>
<td>102</td>
<td>102</td>
<td>102</td>
<td>102</td>
<td>102</td>
<td>102</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>5</td>
<td>0</td>
<td>700</td>
<td>0</td>
<td>700</td>
<td>0</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>6</td>
<td>0</td>
<td>700</td>
<td>0</td>
<td>700</td>
<td>0</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>7</td>
<td>463</td>
<td>463</td>
<td>463</td>
<td>463</td>
<td>463</td>
<td>463</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>8</td>
<td>463</td>
<td>463</td>
<td>463</td>
<td>463</td>
<td>463</td>
<td>463</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>9</td>
<td>458</td>
<td>458</td>
<td>458</td>
<td>458</td>
<td>458</td>
<td>458</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>10</td>
<td>458</td>
<td>458</td>
<td>458</td>
<td>458</td>
<td>458</td>
<td>458</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>11</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>12</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>13</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

2ND SEM SCHEDULE

<table>
<thead>
<tr>
<th></th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
<th>6</th>
<th>7</th>
<th>8</th>
<th>9</th>
<th>10</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>2</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>3</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>4</td>
<td>102</td>
<td>102</td>
<td>102</td>
<td>102</td>
<td>102</td>
<td>102</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>5</td>
<td>0</td>
<td>700</td>
<td>750</td>
<td>700</td>
<td>700</td>
<td>700</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>6</td>
<td>0</td>
<td>700</td>
<td>750</td>
<td>700</td>
<td>700</td>
<td>700</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>7</td>
<td>463</td>
<td>463</td>
<td>463</td>
<td>463</td>
<td>463</td>
<td>463</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>8</td>
<td>463</td>
<td>463</td>
<td>463</td>
<td>463</td>
<td>463</td>
<td>463</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>9</td>
<td>458</td>
<td>458</td>
<td>458</td>
<td>458</td>
<td>458</td>
<td>458</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>10</td>
<td>458</td>
<td>458</td>
<td>458</td>
<td>458</td>
<td>458</td>
<td>458</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>11</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>12</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>13</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
RUN - FINAL
RESULTS FOR GRADE 12
FULL SCHEDULES - 99
PARTIAL SCHEDULES - 4
IRR CONFLICTS - 0
TOTAL - 103

STUDY HALL COUNT

1ST SEM SCHEDULE
1  0  0  0  0  0  0
2  0  0  0  0  0  0
3  20 16 20 16 16 16
4  47 24 47 24 31 31
5  20  3 20  3 20 20
6  25  8 25  8 25 25
7  29 31 29 31 29 29
8  12 14 12 14 14 14
9  41 31 41 31 41 41
10 59 49 59 49 59 59
11  0  0  0  0  0  0
12 22 54 22 54 37 37
13 53 55 53 53 53 53

2ND SEM SCHEDULE
1  0  0  0  0  0  0
2  0  0  0  0  0  0
3  20 18 20 18 20 20
4  48 25 48 25 31 31
5  20  2 20  2 2 2
6  25  7 25  7 7 7
7  47 38 47 38 38 38
8  30 21 30 21 21 21
9  41 31 41 31 31 31
10 59 49 59 49 49 49
11  0  0  0  0  0  0
12 26 61 26 61 29 29
13 38 40 38 38 40 40
## COURSES IN NUMERICAL ORDER ARE:

<table>
<thead>
<tr>
<th>CODE</th>
<th>SEC</th>
<th>SEATS</th>
<th>SEM</th>
<th>FD</th>
<th>DAYS</th>
<th>MET</th>
<th>DESCRIPTION</th>
<th>ROOM</th>
<th>TCH</th>
<th>FER</th>
<th>ENROL</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>1</td>
<td>200</td>
<td>3</td>
<td>11</td>
<td>MTWRF</td>
<td></td>
<td>ENG 12</td>
<td>N</td>
<td>1</td>
<td>8</td>
<td>91</td>
</tr>
<tr>
<td>2</td>
<td>1</td>
<td>200</td>
<td>3</td>
<td>5</td>
<td>MTWRF</td>
<td></td>
<td>ENG 11-2</td>
<td>125</td>
<td>5</td>
<td>5</td>
<td>16</td>
</tr>
<tr>
<td>2</td>
<td>1</td>
<td>200</td>
<td>3</td>
<td>6</td>
<td>MTWRF</td>
<td></td>
<td>ENG 11-2</td>
<td>125</td>
<td>5</td>
<td>5</td>
<td>16</td>
</tr>
<tr>
<td>3</td>
<td>1</td>
<td>200</td>
<td>3</td>
<td>7</td>
<td>MTWRF</td>
<td></td>
<td>ENG 11-2</td>
<td>125</td>
<td>1</td>
<td>6</td>
<td>21</td>
</tr>
<tr>
<td>3</td>
<td>1</td>
<td>200</td>
<td>3</td>
<td>8</td>
<td>MTWRF</td>
<td></td>
<td>ENG 11-2</td>
<td>125</td>
<td>1</td>
<td>6</td>
<td>21</td>
</tr>
<tr>
<td>3</td>
<td>1</td>
<td>200</td>
<td>3</td>
<td>9</td>
<td>MTWRF</td>
<td></td>
<td>ENG 11-2</td>
<td>125</td>
<td>8</td>
<td>7</td>
<td>24</td>
</tr>
<tr>
<td>4</td>
<td>1</td>
<td>200</td>
<td>3</td>
<td>10</td>
<td>MTWRF</td>
<td></td>
<td>ENG 11-2</td>
<td>125</td>
<td>8</td>
<td>8</td>
<td>24</td>
</tr>
<tr>
<td>4</td>
<td>1</td>
<td>200</td>
<td>3</td>
<td>13</td>
<td>MTWRF</td>
<td></td>
<td>ENG 11-2</td>
<td>125</td>
<td>2</td>
<td>10</td>
<td>16</td>
</tr>
<tr>
<td>5</td>
<td>1</td>
<td>200</td>
<td>3</td>
<td>4</td>
<td>MTWRF</td>
<td></td>
<td>ENG 11-3</td>
<td>125</td>
<td>8</td>
<td>3</td>
<td>26</td>
</tr>
<tr>
<td>5</td>
<td>1</td>
<td>200</td>
<td>3</td>
<td>12</td>
<td>MTWRF</td>
<td></td>
<td>ENG 11-3</td>
<td>125</td>
<td>6</td>
<td>9</td>
<td>21</td>
</tr>
<tr>
<td>6</td>
<td>1</td>
<td>200</td>
<td>3</td>
<td>8</td>
<td>MTWRF</td>
<td></td>
<td>ENG 10-2</td>
<td>129</td>
<td>4</td>
<td>6</td>
<td>16</td>
</tr>
<tr>
<td>6</td>
<td>1</td>
<td>200</td>
<td>3</td>
<td>7</td>
<td>MTWRF</td>
<td></td>
<td>ENG 10-2</td>
<td>129</td>
<td>4</td>
<td>6</td>
<td>16</td>
</tr>
<tr>
<td>6</td>
<td>1</td>
<td>200</td>
<td>3</td>
<td>10</td>
<td>MTWRF</td>
<td></td>
<td>ENG 10-2</td>
<td>129</td>
<td>5</td>
<td>9</td>
<td>21</td>
</tr>
<tr>
<td>6</td>
<td>1</td>
<td>200</td>
<td>3</td>
<td>12</td>
<td>MTWRF</td>
<td></td>
<td>ENG 10-2</td>
<td>129</td>
<td>9</td>
<td>4</td>
<td>22</td>
</tr>
<tr>
<td>7</td>
<td>1</td>
<td>200</td>
<td>3</td>
<td>3</td>
<td>MTWRF</td>
<td></td>
<td>ENG 11-3</td>
<td>129</td>
<td>9</td>
<td>3</td>
<td>20</td>
</tr>
<tr>
<td>7</td>
<td>1</td>
<td>200</td>
<td>3</td>
<td>13</td>
<td>MTWRF</td>
<td></td>
<td>ENG 11-3</td>
<td>129</td>
<td>9</td>
<td>9</td>
<td>21</td>
</tr>
<tr>
<td>7</td>
<td>1</td>
<td>200</td>
<td>3</td>
<td>8</td>
<td>MTWRF</td>
<td></td>
<td>ENG 9-2</td>
<td>220</td>
<td>10</td>
<td>3</td>
<td>20</td>
</tr>
<tr>
<td>7</td>
<td>1</td>
<td>200</td>
<td>3</td>
<td>7</td>
<td>MTWRF</td>
<td></td>
<td>ENG 9-2</td>
<td>220</td>
<td>10</td>
<td>6</td>
<td>11</td>
</tr>
<tr>
<td>7</td>
<td>1</td>
<td>200</td>
<td>3</td>
<td>9</td>
<td>MTWRF</td>
<td></td>
<td>ENG 9-2</td>
<td>220</td>
<td>10</td>
<td>6</td>
<td>11</td>
</tr>
<tr>
<td>7</td>
<td>1</td>
<td>200</td>
<td>3</td>
<td>10</td>
<td>MTWRF</td>
<td></td>
<td>ENG 9-2</td>
<td>220</td>
<td>3</td>
<td>7</td>
<td>5</td>
</tr>
<tr>
<td>7</td>
<td>1</td>
<td>200</td>
<td>3</td>
<td>13</td>
<td>MTWRF</td>
<td></td>
<td>ENG 9-2</td>
<td>220</td>
<td>3</td>
<td>7</td>
<td>5</td>
</tr>
<tr>
<td>7</td>
<td>1</td>
<td>200</td>
<td>3</td>
<td>5</td>
<td>MTWRF</td>
<td></td>
<td>ENG 9-3</td>
<td>129</td>
<td>4</td>
<td>5</td>
<td>14</td>
</tr>
<tr>
<td>7</td>
<td>1</td>
<td>200</td>
<td>3</td>
<td>6</td>
<td>MTWRF</td>
<td></td>
<td>ENG 9-3</td>
<td>129</td>
<td>4</td>
<td>5</td>
<td>14</td>
</tr>
<tr>
<td>7</td>
<td>2</td>
<td>200</td>
<td>3</td>
<td>4</td>
<td>MTWRF</td>
<td></td>
<td>ENG 9-3</td>
<td>220</td>
<td>10</td>
<td>4</td>
<td>15</td>
</tr>
<tr>
<td>7</td>
<td>3</td>
<td>200</td>
<td>3</td>
<td>13</td>
<td>MTWRF</td>
<td></td>
<td>ENG 9-3</td>
<td>123</td>
<td>4</td>
<td>10</td>
<td>17</td>
</tr>
<tr>
<td>9</td>
<td>1</td>
<td>200</td>
<td>3</td>
<td>11</td>
<td>MTWRF</td>
<td></td>
<td>AP ENGLISH</td>
<td>129</td>
<td>9</td>
<td>8</td>
<td>12</td>
</tr>
<tr>
<td>50</td>
<td>1</td>
<td>10</td>
<td>3</td>
<td>4</td>
<td>M W</td>
<td></td>
<td>WRITING 12</td>
<td>228</td>
<td>1</td>
<td>4</td>
<td>1</td>
</tr>
<tr>
<td>51</td>
<td>1</td>
<td>10</td>
<td>3</td>
<td>4</td>
<td>M W</td>
<td></td>
<td>WRITING 11</td>
<td>228</td>
<td>1</td>
<td>4</td>
<td>1</td>
</tr>
<tr>
<td>51</td>
<td>2</td>
<td>10</td>
<td>3</td>
<td>3</td>
<td>M W</td>
<td></td>
<td>WRITING 11</td>
<td>228</td>
<td>1</td>
<td>3</td>
<td>0</td>
</tr>
<tr>
<td>52</td>
<td>2</td>
<td>9</td>
<td>3</td>
<td>3</td>
<td>T F</td>
<td></td>
<td>WRIT LAB</td>
<td>228</td>
<td>3</td>
<td>3</td>
<td>9</td>
</tr>
<tr>
<td>52</td>
<td>3</td>
<td>9</td>
<td>3</td>
<td>3</td>
<td>T R</td>
<td></td>
<td>WRIT LAB</td>
<td>228</td>
<td>6</td>
<td>4</td>
<td>8</td>
</tr>
<tr>
<td>52</td>
<td>3</td>
<td>9</td>
<td>3</td>
<td>5</td>
<td>T R</td>
<td></td>
<td>WRIT LAB</td>
<td>228</td>
<td>6</td>
<td>5</td>
<td>3</td>
</tr>
<tr>
<td>52</td>
<td>4</td>
<td>9</td>
<td>3</td>
<td>6</td>
<td>T R</td>
<td></td>
<td>WRIT LAB</td>
<td>228</td>
<td>6</td>
<td>5</td>
<td>3</td>
</tr>
<tr>
<td>52</td>
<td>4</td>
<td>9</td>
<td>3</td>
<td>5</td>
<td>M W</td>
<td></td>
<td>WRIT LAB</td>
<td>228</td>
<td>6</td>
<td>5</td>
<td>7</td>
</tr>
<tr>
<td>52</td>
<td>5</td>
<td>9</td>
<td>3</td>
<td>6</td>
<td>M W</td>
<td></td>
<td>WRIT LAB</td>
<td>228</td>
<td>6</td>
<td>6</td>
<td>6</td>
</tr>
<tr>
<td>52</td>
<td>5</td>
<td>9</td>
<td>3</td>
<td>8</td>
<td>M W</td>
<td></td>
<td>WRIT LAB</td>
<td>228</td>
<td>6</td>
<td>6</td>
<td>6</td>
</tr>
<tr>
<td>52</td>
<td>6</td>
<td>9</td>
<td>3</td>
<td>12</td>
<td>M W</td>
<td></td>
<td>WRIT LAB</td>
<td>222</td>
<td>1</td>
<td>9</td>
<td>9</td>
</tr>
<tr>
<td>52</td>
<td>7</td>
<td>9</td>
<td>3</td>
<td>9</td>
<td>M W</td>
<td></td>
<td>WRIT LAB</td>
<td>228</td>
<td>6</td>
<td>7</td>
<td>9</td>
</tr>
<tr>
<td>52</td>
<td>7</td>
<td>9</td>
<td>3</td>
<td>10</td>
<td>M W</td>
<td></td>
<td>WRIT LAB</td>
<td>228</td>
<td>6</td>
<td>7</td>
<td>9</td>
</tr>
<tr>
<td>52</td>
<td>8</td>
<td>9</td>
<td>3</td>
<td>12</td>
<td>T F</td>
<td></td>
<td>WRIT LAB</td>
<td>228</td>
<td>3</td>
<td>9</td>
<td>9</td>
</tr>
<tr>
<td>52</td>
<td>9</td>
<td>9</td>
<td>3</td>
<td>13</td>
<td>T F</td>
<td></td>
<td>WRIT LAB</td>
<td>228</td>
<td>6</td>
<td>10</td>
<td>9</td>
</tr>
<tr>
<td>53</td>
<td>1</td>
<td>10</td>
<td>3</td>
<td>1</td>
<td>T R</td>
<td></td>
<td>WRIT 11-12</td>
<td>228</td>
<td>1</td>
<td>1</td>
<td>10</td>
</tr>
<tr>
<td>53</td>
<td>1</td>
<td>10</td>
<td>3</td>
<td>4</td>
<td>T R</td>
<td></td>
<td>READING 11</td>
<td>209</td>
<td>4</td>
<td>4</td>
<td>1</td>
</tr>
<tr>
<td>53</td>
<td>1</td>
<td>10</td>
<td>3</td>
<td>5</td>
<td>T R</td>
<td></td>
<td>READING 10</td>
<td>207</td>
<td>1</td>
<td>5</td>
<td>11</td>
</tr>
<tr>
<td>53</td>
<td>1</td>
<td>10</td>
<td>3</td>
<td>6</td>
<td>T R</td>
<td></td>
<td>READING 10</td>
<td>207</td>
<td>1</td>
<td>5</td>
<td>11</td>
</tr>
<tr>
<td>53</td>
<td>2</td>
<td>10</td>
<td>3</td>
<td>13</td>
<td>M WR</td>
<td></td>
<td>READING 10</td>
<td>209</td>
<td>15</td>
<td>10</td>
<td>14</td>
</tr>
<tr>
<td>53</td>
<td>3</td>
<td>10</td>
<td>3</td>
<td>3</td>
<td>M WR</td>
<td></td>
<td>READING 9</td>
<td>207</td>
<td>17</td>
<td>3</td>
<td>15</td>
</tr>
<tr>
<td>53</td>
<td>3</td>
<td>10</td>
<td>3</td>
<td>4</td>
<td>M W F</td>
<td></td>
<td>READING 9</td>
<td>207</td>
<td>17</td>
<td>4</td>
<td>12</td>
</tr>
<tr>
<td>53</td>
<td>3</td>
<td>10</td>
<td>3</td>
<td>15</td>
<td>M WR</td>
<td></td>
<td>READING 9</td>
<td>207</td>
<td>17</td>
<td>10</td>
<td>13</td>
</tr>
<tr>
<td>64</td>
<td>1</td>
<td>15</td>
<td>3</td>
<td>1</td>
<td>M W F</td>
<td></td>
<td>READ 11-12</td>
<td>228</td>
<td>15</td>
<td>1</td>
<td>9</td>
</tr>
<tr>
<td>CRN</td>
<td>Code</td>
<td>Seat</td>
<td>Title</td>
<td>Credits</td>
<td>Days</td>
<td>Time</td>
<td>Location</td>
<td>Instructor</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>
</tr>
<tr>
<td>100</td>
<td>1</td>
<td>30</td>
<td>12</td>
<td>MTWRF</td>
<td>SS IND STD</td>
<td>222</td>
<td>24</td>
<td>9 11</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>101</td>
<td>1</td>
<td>35</td>
<td>3</td>
<td>MTWRF</td>
<td>AM STD 2</td>
<td>213</td>
<td>20</td>
<td>3 5</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>101</td>
<td>2</td>
<td>35</td>
<td>7</td>
<td>MTWRF</td>
<td>AM STD 2</td>
<td>219</td>
<td>24</td>
<td>6 10</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>101</td>
<td>3</td>
<td>35</td>
<td>8</td>
<td>MTWRF</td>
<td>AM STD 2</td>
<td>219</td>
<td>24</td>
<td>6 10</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>101</td>
<td>4</td>
<td>35</td>
<td>12</td>
<td>MTWRF</td>
<td>AM STD 2</td>
<td>213</td>
<td>20</td>
<td>9 18</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>101</td>
<td>5</td>
<td>35</td>
<td>13</td>
<td>MTWRF</td>
<td>AM STD 2</td>
<td>215</td>
<td>24</td>
<td>10 12</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>102</td>
<td>1</td>
<td>35</td>
<td>4</td>
<td>MTWRF</td>
<td>AM STD 3</td>
<td>213</td>
<td>20</td>
<td>4 20</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>102</td>
<td>2</td>
<td>35</td>
<td>5</td>
<td>MTWRF</td>
<td>AM STD 3</td>
<td>213</td>
<td>23</td>
<td>5 20</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>103</td>
<td>1</td>
<td>35</td>
<td>3</td>
<td>MTWRF</td>
<td>E CUL STD</td>
<td>219</td>
<td>26</td>
<td>3 20</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>103</td>
<td>2</td>
<td>35</td>
<td>5</td>
<td>MTWRF</td>
<td>E CUL ST 2</td>
<td>219</td>
<td>26</td>
<td>5 17</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>103</td>
<td>3</td>
<td>35</td>
<td>6</td>
<td>MTWRF</td>
<td>E CUL ST 2</td>
<td>219</td>
<td>26</td>
<td>5 17</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>103</td>
<td>4</td>
<td>35</td>
<td>9</td>
<td>MTWRF</td>
<td>E CUL ST 2</td>
<td>219</td>
<td>26</td>
<td>7 20</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>103</td>
<td>5</td>
<td>35</td>
<td>10</td>
<td>MTWRF</td>
<td>E CUL ST 2</td>
<td>219</td>
<td>26</td>
<td>7 20</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>104</td>
<td>1</td>
<td>35</td>
<td>7</td>
<td>MTWRF</td>
<td>E CUL ST 2</td>
<td>222</td>
<td>24</td>
<td>8 26</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>104</td>
<td>2</td>
<td>35</td>
<td>11</td>
<td>MTWRF</td>
<td>E CUL ST 2</td>
<td>219</td>
<td>26</td>
<td>8 18</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>104</td>
<td>3</td>
<td>35</td>
<td>11</td>
<td>MTWRF</td>
<td>E CUL ST 2</td>
<td>219</td>
<td>26</td>
<td>8 18</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>105</td>
<td>1</td>
<td>35</td>
<td>3</td>
<td>MTWRF</td>
<td>AFR-ASIA 2</td>
<td>215</td>
<td>23</td>
<td>6 20</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>105</td>
<td>2</td>
<td>35</td>
<td>3</td>
<td>MTWRF</td>
<td>AFR-ASIA 2</td>
<td>215</td>
<td>23</td>
<td>6 20</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>105</td>
<td>3</td>
<td>35</td>
<td>3</td>
<td>MTWRF</td>
<td>AFR-ASIA 2</td>
<td>215</td>
<td>22</td>
<td>7 21</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>105</td>
<td>4</td>
<td>35</td>
<td>9</td>
<td>MTWRF</td>
<td>AFR-ASIA 2</td>
<td>215</td>
<td>22</td>
<td>7 21</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>105</td>
<td>5</td>
<td>35</td>
<td>12</td>
<td>MTWRF</td>
<td>AFR-ASIA 2</td>
<td>215</td>
<td>23</td>
<td>9 21</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>106</td>
<td>1</td>
<td>35</td>
<td>3</td>
<td>MTWRF</td>
<td>AFR-ASIA 3</td>
<td>215</td>
<td>22</td>
<td>3 19</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>106</td>
<td>2</td>
<td>35</td>
<td>3</td>
<td>MTWRF</td>
<td>AFR-ASIA 3</td>
<td>215</td>
<td>22</td>
<td>3 19</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>106</td>
<td>3</td>
<td>35</td>
<td>6</td>
<td>MTWRF</td>
<td>AFR-ASIA 3</td>
<td>215</td>
<td>22</td>
<td>5 13</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>106</td>
<td>4</td>
<td>35</td>
<td>11</td>
<td>MTWRF</td>
<td>AFR-ASIA 3</td>
<td>215</td>
<td>22</td>
<td>5 13</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>107</td>
<td>1</td>
<td>35</td>
<td>13</td>
<td>MTWRF</td>
<td>LAW</td>
<td>213</td>
<td>20</td>
<td>10 12</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>110</td>
<td>1</td>
<td>35</td>
<td>1</td>
<td>MTWRF</td>
<td>AMER STD 3</td>
<td>213</td>
<td>23</td>
<td>1 19</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>200</td>
<td>1</td>
<td>35</td>
<td>5</td>
<td>MTWRF</td>
<td>MATH 12-1</td>
<td>221</td>
<td>30</td>
<td>5 15</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>200</td>
<td>2</td>
<td>35</td>
<td>6</td>
<td>MTWRF</td>
<td>MATH 12-1</td>
<td>221</td>
<td>30</td>
<td>5 15</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>201</td>
<td>1</td>
<td>35</td>
<td>3</td>
<td>MTWRF</td>
<td>MATH 12-2</td>
<td>222</td>
<td>30</td>
<td>3 12</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>201</td>
<td>2</td>
<td>35</td>
<td>13</td>
<td>MTWRF</td>
<td>MATH 12-2</td>
<td>221</td>
<td>30</td>
<td>10 12</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>202</td>
<td>1</td>
<td>35</td>
<td>13</td>
<td>MTWRF</td>
<td>MATH CRS 3</td>
<td>224</td>
<td>32</td>
<td>10 23</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>202</td>
<td>2</td>
<td>35</td>
<td>7</td>
<td>MTWRF</td>
<td>MATH CRS 3</td>
<td>224</td>
<td>32</td>
<td>6 22</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>202</td>
<td>3</td>
<td>35</td>
<td>8</td>
<td>MTWRF</td>
<td>MATH CRS 3</td>
<td>224</td>
<td>32</td>
<td>6 22</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>203</td>
<td>1</td>
<td>35</td>
<td>7</td>
<td>MTWRF</td>
<td>MATH CRS 2</td>
<td>221</td>
<td>33</td>
<td>3 18</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>203</td>
<td>2</td>
<td>35</td>
<td>7</td>
<td>MTWRF</td>
<td>MATH CRS 2</td>
<td>221</td>
<td>36</td>
<td>6 18</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>203</td>
<td>3</td>
<td>35</td>
<td>9</td>
<td>MTWRF</td>
<td>MATH CRS 2</td>
<td>221</td>
<td>33</td>
<td>7 19</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>203</td>
<td>4</td>
<td>35</td>
<td>7</td>
<td>MTWRF</td>
<td>MATH CRS 2</td>
<td>221</td>
<td>33</td>
<td>7 19</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>204</td>
<td>1</td>
<td>35</td>
<td>3</td>
<td>MTWRF</td>
<td>MATH CRS 1</td>
<td>224</td>
<td>32</td>
<td>3 23</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>204</td>
<td>2</td>
<td>35</td>
<td>3</td>
<td>MTWRF</td>
<td>MATH CRS 1</td>
<td>221</td>
<td>31</td>
<td>4 18</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>204</td>
<td>3</td>
<td>35</td>
<td>9</td>
<td>MTWRF</td>
<td>MATH CRS 1</td>
<td>224</td>
<td>32</td>
<td>4 18</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>204</td>
<td>4</td>
<td>35</td>
<td>10</td>
<td>MTWRF</td>
<td>MATH CRS 1</td>
<td>224</td>
<td>32</td>
<td>7 19</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>204</td>
<td>5</td>
<td>35</td>
<td>12</td>
<td>MTWRF</td>
<td>MATH CRS 1</td>
<td>221</td>
<td>33</td>
<td>9 21</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>205</td>
<td>1</td>
<td>35</td>
<td>9</td>
<td>MTWRF</td>
<td>MATH 9-3</td>
<td>217</td>
<td>37</td>
<td>7 17</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>205</td>
<td>2</td>
<td>35</td>
<td>10</td>
<td>MTWRF</td>
<td>MATH 9-3</td>
<td>217</td>
<td>37</td>
<td>7 17</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>205</td>
<td>3</td>
<td>35</td>
<td>11</td>
<td>MTWRF</td>
<td>MATH 9-3</td>
<td>224</td>
<td>37</td>
<td>8 16</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>205</td>
<td>4</td>
<td>35</td>
<td>12</td>
<td>MTWRF</td>
<td>MATH 9-3</td>
<td>217</td>
<td>37</td>
<td>9 13</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>206</td>
<td>1</td>
<td>30</td>
<td>7</td>
<td>MTWRF</td>
<td>GEN MATH</td>
<td>217</td>
<td>37</td>
<td>6 18</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>206</td>
<td>2</td>
<td>30</td>
<td>8</td>
<td>MTWRF</td>
<td>GEN MATH</td>
<td>217</td>
<td>37</td>
<td>6 18</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>206</td>
<td>3</td>
<td>30</td>
<td>13</td>
<td>MTWRF</td>
<td>GEN MATH</td>
<td>222</td>
<td>35</td>
<td>10 19</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>207</td>
<td>1</td>
<td>20</td>
<td>12</td>
<td>MTWRF</td>
<td>C PREP M</td>
<td>116</td>
<td>35</td>
<td>9 5</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>250</td>
<td>1</td>
<td>20</td>
<td>3</td>
<td>MTWRF</td>
<td>COMP PRG 4</td>
<td>119</td>
<td>33</td>
<td>4 6</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>251</td>
<td>1</td>
<td>20</td>
<td>3</td>
<td>MTWRF</td>
<td>COMP PRG 4</td>
<td>119</td>
<td>33</td>
<td>4 6</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Week</td>
<td>Monday</td>
<td>Tuesday</td>
<td>Wednesday</td>
<td>Thursday</td>
<td>Friday</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>
</tr>
<tr>
<td>MTWRF</td>
<td>COMP PHY 2</td>
<td>119</td>
<td>40</td>
<td>9</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>MTWRF</td>
<td>COMP PHY 1</td>
<td>119</td>
<td>40</td>
<td>3</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>MTWRF</td>
<td>COMP-AWARE</td>
<td>112</td>
<td>30</td>
<td>6</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>MTWRF</td>
<td>COMP-AWARE</td>
<td>113</td>
<td>30</td>
<td>6</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>MTWRF</td>
<td>COMP-AWARE</td>
<td>113</td>
<td>30</td>
<td>7</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>MTWRF</td>
<td>COMP-AWARE</td>
<td>113</td>
<td>40</td>
<td>10</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>MTWRF</td>
<td>COMP-AWARE</td>
<td>113</td>
<td>40</td>
<td>10</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>MTWRF</td>
<td>ADV BIO</td>
<td>101</td>
<td>46</td>
<td>3</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>MTWRF</td>
<td>ADV BIO</td>
<td>101</td>
<td>46</td>
<td>4</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>MTWRF</td>
<td>PHYSICS 2</td>
<td>108</td>
<td>45</td>
<td>6</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>MTWRF</td>
<td>PHYSICS 2</td>
<td>108</td>
<td>45</td>
<td>6</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>MTWRF</td>
<td>PHYSICS 2</td>
<td>108</td>
<td>45</td>
<td>7</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>MTWRF</td>
<td>CHEM 2</td>
<td>107</td>
<td>50</td>
<td>8</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>MTWRF</td>
<td>CHEM 2</td>
<td>107</td>
<td>50</td>
<td>9</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>MTWRF</td>
<td>CHEM 2</td>
<td>107</td>
<td>50</td>
<td>4</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>MTWRF</td>
<td>CHEM 2</td>
<td>107</td>
<td>50</td>
<td>5</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>MTWRF</td>
<td>CHEM 2</td>
<td>107</td>
<td>50</td>
<td>9</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>MTWRF</td>
<td>CHEM 2</td>
<td>107</td>
<td>50</td>
<td>10</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>MTWRF</td>
<td>BIOLOGY 2</td>
<td>101</td>
<td>46</td>
<td>5</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>MTWRF</td>
<td>BIOLOGY 2</td>
<td>101</td>
<td>46</td>
<td>5</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>MTWRF</td>
<td>BIOLOGY 2</td>
<td>101</td>
<td>46</td>
<td>6</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>MTWRF</td>
<td>BIOLOGY 2</td>
<td>101</td>
<td>46</td>
<td>8</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>MTWRF</td>
<td>BIOLOGY 2</td>
<td>101</td>
<td>46</td>
<td>9</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>MTWRF</td>
<td>BIOLOGY 2</td>
<td>101</td>
<td>46</td>
<td>9</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>MTWRF</td>
<td>BIOLOGY 2</td>
<td>101</td>
<td>46</td>
<td>9</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>MTWRF</td>
<td>BIOLOGY 2</td>
<td>101</td>
<td>46</td>
<td>10</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>MTWRF</td>
<td>BIOLOGY 3</td>
<td>107</td>
<td>50</td>
<td>3</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>MTWRF</td>
<td>ER SCI R</td>
<td>103</td>
<td>49</td>
<td>4</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>MTWRF</td>
<td>ER SCI R</td>
<td>103</td>
<td>49</td>
<td>5</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>MTWRF</td>
<td>ER SCI R</td>
<td>108</td>
<td>45</td>
<td>7</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>MTWRF</td>
<td>ER SCI R</td>
<td>108</td>
<td>45</td>
<td>8</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>MTWRF</td>
<td>SCI 9-2</td>
<td>103</td>
<td>49</td>
<td>8</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>MTWRF</td>
<td>SCI 9-2</td>
<td>103</td>
<td>49</td>
<td>8</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>MTWRF</td>
<td>SCI 9-2</td>
<td>103</td>
<td>49</td>
<td>10</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>MTWRF</td>
<td>SCI 9-3</td>
<td>108</td>
<td>45</td>
<td>4</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>MTWRF</td>
<td>SCI 9-3</td>
<td>103</td>
<td>84</td>
<td>6</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>MTWRF</td>
<td>SCI 9-3</td>
<td>103</td>
<td>49</td>
<td>7</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>MTWRF</td>
<td>SCI 9-3</td>
<td>103</td>
<td>49</td>
<td>7</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>MTWRF</td>
<td>FRENCH 3</td>
<td>227</td>
<td>55</td>
<td>5</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>MTWRF</td>
<td>FRENCH 3</td>
<td>227</td>
<td>55</td>
<td>5</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>MTWRF</td>
<td>FRENCH 2</td>
<td>227</td>
<td>55</td>
<td>6</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>MTWRF</td>
<td>FRENCH 2</td>
<td>227</td>
<td>55</td>
<td>6</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>MTWRF</td>
<td>FRENCH 1</td>
<td>227</td>
<td>55</td>
<td>10</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>MTWRF</td>
<td>FRENCH 1</td>
<td>227</td>
<td>55</td>
<td>8</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>MTWRF</td>
<td>SPAN 3</td>
<td>217</td>
<td>56</td>
<td>10</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>MTWRF</td>
<td>SPAN 2</td>
<td>217</td>
<td>56</td>
<td>5</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>MTWRF</td>
<td>SPAN 2</td>
<td>217</td>
<td>56</td>
<td>5</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>MTWRF</td>
<td>SPAN 1</td>
<td>227</td>
<td>56</td>
<td>4</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>MTWRF</td>
<td>SPAN 1</td>
<td>227</td>
<td>56</td>
<td>7</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>MTWRF</td>
<td>SPAN 1</td>
<td>227</td>
<td>56</td>
<td>7</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>MTWRF</td>
<td>OFF PROCED</td>
<td>121</td>
<td>61</td>
<td>5</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>MTWRF</td>
<td>OFF PROCED</td>
<td>121</td>
<td>61</td>
<td>5</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>MTWRF</td>
<td>ACCT</td>
<td>123</td>
<td>62</td>
<td>3</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>MTWRF</td>
<td>BUS LAW</td>
<td>123</td>
<td>62</td>
<td>6</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>MTWRF</td>
<td>BUS LAW</td>
<td>123</td>
<td>62</td>
<td>6</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>MTWRF</td>
<td>SHIRT/TRANS</td>
<td>121</td>
<td>60</td>
<td>3</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>MTWRF</td>
<td>SHIRT TH</td>
<td>121</td>
<td>60</td>
<td>7</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>MTWRF</td>
<td>SHIRT TH</td>
<td>121</td>
<td>60</td>
<td>7</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>MTWRF</td>
<td>BUS DYN</td>
<td>123</td>
<td>62</td>
<td>9</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>MTWRF</td>
<td>ADV KEYBD</td>
<td>121</td>
<td>61</td>
<td>4</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Grade</td>
<td>Period</td>
<td>CTE Course</td>
<td>Description</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>
</tr>
<tr>
<td>9</td>
<td>T R</td>
<td>CERAM</td>
<td>ARCH CERAM</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>10</td>
<td>4 MTW</td>
<td>MECH DWG</td>
<td>PRINT DWG</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>11</td>
<td>3 MTW</td>
<td>APP METAL</td>
<td>APP CERAM</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>12</td>
<td>2 MTW</td>
<td>APP METAL</td>
<td>APP WOOD</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>13</td>
<td>2 MTW</td>
<td>APP METAL</td>
<td>APP CERAM</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>14</td>
<td>2 MTW</td>
<td>APP METAL</td>
<td>APP CERAM</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>15</td>
<td>1 MTW</td>
<td>APP METAL</td>
<td>APP CERAM</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**Key Courses:**
- **ART:** STUDIO ART, ARCH, CH CARE, HOME EC, ART IND ST, DWG/PTNG
- **BAND:** CH CARE, H 70, ART IND ST, DWG/PTNG
- **CH CARE:** CH CARE, H 70, ART IND ST, DWG/PTNG
- **CERAM:** CERAM, ARCH CERAM, MECH CERAM, APP CERAM
- **CHOP:** CH CARE, H 70, ART IND ST, DWG/PTNG
- **DWP/PTNG:** CH CARE, H 70, ART IND ST, DWG/PTNG
- **DWP:** CH CARE, H 70, ART IND ST, DWG/PTNG
- **MATH:** MECH DWG, PRINT DWG
- **PHOTO:** MECH DWG, PRINT DWG
<table>
<thead>
<tr>
<th>Day</th>
<th>700</th>
<th>1</th>
<th>80</th>
<th>2</th>
<th>4</th>
<th>T</th>
<th>R</th>
<th>PE 11-12</th>
<th>GYM</th>
<th>80</th>
<th>4</th>
<th>61</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>700</td>
<td>1</td>
<td>80</td>
<td>2</td>
<td>5</td>
<td>T</td>
<td>R</td>
<td>PE 11-12</td>
<td>GYM</td>
<td>80</td>
<td>5</td>
<td>45</td>
</tr>
<tr>
<td></td>
<td>700</td>
<td>2</td>
<td>80</td>
<td>1</td>
<td>5</td>
<td>T</td>
<td>R</td>
<td>PE 11-12</td>
<td>GYM</td>
<td>80</td>
<td>5</td>
<td>45</td>
</tr>
<tr>
<td></td>
<td>700</td>
<td>2</td>
<td>80</td>
<td>2</td>
<td>5</td>
<td>T</td>
<td>R</td>
<td>PE 11-12</td>
<td>GYM</td>
<td>80</td>
<td>5</td>
<td>45</td>
</tr>
<tr>
<td></td>
<td>700</td>
<td>3</td>
<td>80</td>
<td>1</td>
<td>12</td>
<td>M</td>
<td>W</td>
<td>PE 11-12</td>
<td>GYM</td>
<td>80</td>
<td>9</td>
<td>70</td>
</tr>
<tr>
<td></td>
<td>700</td>
<td>3</td>
<td>80</td>
<td>2</td>
<td>12</td>
<td>M</td>
<td>W</td>
<td>F</td>
<td>PE 11-12</td>
<td>GYM</td>
<td>80</td>
<td>9</td>
</tr>
<tr>
<td></td>
<td>701</td>
<td>1</td>
<td>80</td>
<td>1</td>
<td>3</td>
<td>M</td>
<td>W</td>
<td>PE 9-10</td>
<td>GYM</td>
<td>80</td>
<td>3</td>
<td>47</td>
</tr>
<tr>
<td></td>
<td>701</td>
<td>1</td>
<td>80</td>
<td>1</td>
<td>5</td>
<td>M</td>
<td>W</td>
<td>PE 9-10</td>
<td>GYM</td>
<td>80</td>
<td>3</td>
<td>47</td>
</tr>
<tr>
<td></td>
<td>701</td>
<td>1</td>
<td>80</td>
<td>2</td>
<td>5</td>
<td>M</td>
<td>W</td>
<td>PE 9-10</td>
<td>GYM</td>
<td>80</td>
<td>3</td>
<td>47</td>
</tr>
<tr>
<td></td>
<td>701</td>
<td>2</td>
<td>80</td>
<td>2</td>
<td>6</td>
<td>T</td>
<td>R</td>
<td>PE 9-10</td>
<td>GYM</td>
<td>80</td>
<td>5</td>
<td>34</td>
</tr>
<tr>
<td></td>
<td>701</td>
<td>2</td>
<td>80</td>
<td>1</td>
<td>6</td>
<td>M</td>
<td>W</td>
<td>PE 9-10</td>
<td>GYM</td>
<td>80</td>
<td>5</td>
<td>34</td>
</tr>
<tr>
<td></td>
<td>701</td>
<td>3</td>
<td>80</td>
<td>1</td>
<td>7</td>
<td>T</td>
<td>R</td>
<td>PE 9-10</td>
<td>GYM</td>
<td>80</td>
<td>6</td>
<td>34</td>
</tr>
<tr>
<td></td>
<td>701</td>
<td>3</td>
<td>80</td>
<td>1</td>
<td>8</td>
<td>T</td>
<td>R</td>
<td>PE 9-10</td>
<td>GYM</td>
<td>80</td>
<td>6</td>
<td>34</td>
</tr>
<tr>
<td></td>
<td>701</td>
<td>3</td>
<td>80</td>
<td>2</td>
<td>8</td>
<td>T</td>
<td>R</td>
<td>PE 9-10</td>
<td>GYM</td>
<td>80</td>
<td>6</td>
<td>34</td>
</tr>
<tr>
<td></td>
<td>701</td>
<td>4</td>
<td>80</td>
<td>1</td>
<td>9</td>
<td>T</td>
<td>R</td>
<td>PE 9-10</td>
<td>GYM</td>
<td>80</td>
<td>7</td>
<td>26</td>
</tr>
<tr>
<td></td>
<td>701</td>
<td>4</td>
<td>80</td>
<td>1</td>
<td>9</td>
<td>T</td>
<td>R</td>
<td>PE 9-10</td>
<td>GYM</td>
<td>80</td>
<td>7</td>
<td>26</td>
</tr>
<tr>
<td></td>
<td>701</td>
<td>4</td>
<td>80</td>
<td>2</td>
<td>9</td>
<td>T</td>
<td>R</td>
<td>PE 9-10</td>
<td>GYM</td>
<td>80</td>
<td>7</td>
<td>26</td>
</tr>
<tr>
<td></td>
<td>701</td>
<td>5</td>
<td>80</td>
<td>1</td>
<td>10</td>
<td>T</td>
<td>R</td>
<td>PE 9-10</td>
<td>GYM</td>
<td>80</td>
<td>8</td>
<td>58</td>
</tr>
<tr>
<td></td>
<td>701</td>
<td>5</td>
<td>80</td>
<td>1</td>
<td>10</td>
<td>M</td>
<td>W</td>
<td>PE 9-10</td>
<td>GYM</td>
<td>80</td>
<td>8</td>
<td>58</td>
</tr>
<tr>
<td></td>
<td>701</td>
<td>6</td>
<td>90</td>
<td>1</td>
<td>12</td>
<td>T</td>
<td>R</td>
<td>PE 9-10</td>
<td>GYM</td>
<td>80</td>
<td>9</td>
<td>66</td>
</tr>
<tr>
<td></td>
<td>701</td>
<td>6</td>
<td>90</td>
<td>2</td>
<td>12</td>
<td>T</td>
<td>R</td>
<td>PE 9-10</td>
<td>GYM</td>
<td>80</td>
<td>9</td>
<td>66</td>
</tr>
<tr>
<td></td>
<td>702</td>
<td>1</td>
<td>60</td>
<td>1</td>
<td>1</td>
<td>T</td>
<td>R</td>
<td>PE BOCES</td>
<td>GYM</td>
<td>80</td>
<td>1</td>
<td>22</td>
</tr>
<tr>
<td></td>
<td>702</td>
<td>1</td>
<td>80</td>
<td>2</td>
<td>1</td>
<td>T</td>
<td>R</td>
<td>PE BOCES</td>
<td>GYM</td>
<td>80</td>
<td>1</td>
<td>22</td>
</tr>
<tr>
<td></td>
<td>702</td>
<td>2</td>
<td>80</td>
<td>1</td>
<td>2</td>
<td>M</td>
<td>W</td>
<td>PE BOCES</td>
<td>GYM</td>
<td>80</td>
<td>2</td>
<td>21</td>
</tr>
<tr>
<td></td>
<td>702</td>
<td>2</td>
<td>80</td>
<td>2</td>
<td>2</td>
<td>M</td>
<td>W</td>
<td>PE BOCES</td>
<td>GYM</td>
<td>80</td>
<td>2</td>
<td>21</td>
</tr>
<tr>
<td></td>
<td>750</td>
<td>1</td>
<td>35</td>
<td>1</td>
<td>4</td>
<td>F</td>
<td>DR ED TH</td>
<td>107</td>
<td>91</td>
<td>4</td>
<td>24</td>
<td></td>
</tr>
<tr>
<td></td>
<td>750</td>
<td>2</td>
<td>35</td>
<td>2</td>
<td>5</td>
<td>W</td>
<td>DR ED TH</td>
<td>108</td>
<td>91</td>
<td>5</td>
<td>17</td>
<td></td>
</tr>
<tr>
<td></td>
<td>750</td>
<td>2</td>
<td>35</td>
<td>2</td>
<td>6</td>
<td>W</td>
<td>DR ED TH</td>
<td>108</td>
<td>91</td>
<td>5</td>
<td>17</td>
<td></td>
</tr>
<tr>
<td></td>
<td>750</td>
<td>3</td>
<td>35</td>
<td>1</td>
<td>12</td>
<td>F</td>
<td>DR ED TH</td>
<td>107</td>
<td>91</td>
<td>9</td>
<td>23</td>
<td></td>
</tr>
<tr>
<td></td>
<td>751</td>
<td>1</td>
<td>35</td>
<td>2</td>
<td>1</td>
<td>W</td>
<td>DR ED TH</td>
<td>108</td>
<td>91</td>
<td>1</td>
<td>35</td>
<td></td>
</tr>
<tr>
<td></td>
<td>770</td>
<td>1</td>
<td>35</td>
<td>1</td>
<td>3</td>
<td>T</td>
<td>RF</td>
<td>HEALTH 11</td>
<td>108</td>
<td>87</td>
<td>3</td>
<td>15</td>
</tr>
<tr>
<td></td>
<td>770</td>
<td>1</td>
<td>35</td>
<td>1</td>
<td>4</td>
<td>M</td>
<td>W</td>
<td>HEALTH 11</td>
<td>206</td>
<td>87</td>
<td>4</td>
<td>20</td>
</tr>
<tr>
<td></td>
<td>770</td>
<td>1</td>
<td>35</td>
<td>1</td>
<td>5</td>
<td>M</td>
<td>W</td>
<td>HEALTH 11</td>
<td>108</td>
<td>87</td>
<td>5</td>
<td>20</td>
</tr>
<tr>
<td></td>
<td>770</td>
<td>1</td>
<td>35</td>
<td>1</td>
<td>6</td>
<td>M</td>
<td>W</td>
<td>HEALTH 11</td>
<td>108</td>
<td>87</td>
<td>5</td>
<td>20</td>
</tr>
<tr>
<td></td>
<td>770</td>
<td>1</td>
<td>35</td>
<td>1</td>
<td>7</td>
<td>M</td>
<td>W</td>
<td>HEALTH 11</td>
<td>107</td>
<td>87</td>
<td>6</td>
<td>19</td>
</tr>
<tr>
<td></td>
<td>770</td>
<td>1</td>
<td>35</td>
<td>1</td>
<td>8</td>
<td>M</td>
<td>W</td>
<td>HEALTH 11</td>
<td>108</td>
<td>87</td>
<td>6</td>
<td>19</td>
</tr>
<tr>
<td></td>
<td>772</td>
<td>1</td>
<td>35</td>
<td>1</td>
<td>12</td>
<td>T</td>
<td>RF</td>
<td>HEALTH 11</td>
<td>108</td>
<td>87</td>
<td>2</td>
<td>21</td>
</tr>
<tr>
<td></td>
<td>772</td>
<td>2</td>
<td>35</td>
<td>1</td>
<td>12</td>
<td>F</td>
<td>HEALTH 11</td>
<td>108</td>
<td>87</td>
<td>1</td>
<td>22</td>
<td></td>
</tr>
<tr>
<td></td>
<td>790</td>
<td>1</td>
<td>35</td>
<td>2</td>
<td>13</td>
<td>MTWRF</td>
<td>PSYCH</td>
<td>213</td>
<td>20</td>
<td>10</td>
<td>22</td>
<td></td>
</tr>
<tr>
<td></td>
<td>999</td>
<td>1</td>
<td>300</td>
<td>3</td>
<td>5</td>
<td>MTWRF</td>
<td>BOCES AM</td>
<td>BOC</td>
<td>99</td>
<td>8</td>
<td>56</td>
<td></td>
</tr>
<tr>
<td></td>
<td>999</td>
<td>2</td>
<td>300</td>
<td>3</td>
<td>6</td>
<td>MTWRF</td>
<td>BOCES AM</td>
<td>BOC</td>
<td>99</td>
<td>8</td>
<td>56</td>
<td></td>
</tr>
<tr>
<td></td>
<td>999</td>
<td>3</td>
<td>300</td>
<td>3</td>
<td>6</td>
<td>MTWRF</td>
<td>BOCES AM</td>
<td>BOC</td>
<td>99</td>
<td>8</td>
<td>56</td>
<td></td>
</tr>
<tr>
<td></td>
<td>999</td>
<td>4</td>
<td>300</td>
<td>3</td>
<td>8</td>
<td>MTWRF</td>
<td>BOCES AM</td>
<td>BOC</td>
<td>99</td>
<td>8</td>
<td>56</td>
<td></td>
</tr>
<tr>
<td></td>
<td>999</td>
<td>5</td>
<td>300</td>
<td>3</td>
<td>9</td>
<td>MTWRF</td>
<td>BOCES AM</td>
<td>BOC</td>
<td>99</td>
<td>9</td>
<td>38</td>
<td></td>
</tr>
<tr>
<td></td>
<td>999</td>
<td>6</td>
<td>300</td>
<td>3</td>
<td>10</td>
<td>MTWRF</td>
<td>BOCES AM</td>
<td>BOC</td>
<td>99</td>
<td>10</td>
<td>38</td>
<td></td>
</tr>
</tbody>
</table>
9.6. STUDENT SCHEDULES
### STUDENT NAME: ABARE ANN

**ID:** 21610  
**GRADE:** 9

<table>
<thead>
<tr>
<th>PERIOD</th>
<th>DAYS</th>
<th>COURSE</th>
<th>ROOM</th>
<th>SEM</th>
<th>TEACHER</th>
<th>CODE</th>
</tr>
</thead>
<tbody>
<tr>
<td>3</td>
<td>M W F</td>
<td>PE 9-10</td>
<td>GYM 2</td>
<td>80</td>
<td>701</td>
<td></td>
</tr>
<tr>
<td>3</td>
<td>M W</td>
<td>PE 9-10</td>
<td>GYM 1</td>
<td>80</td>
<td>701</td>
<td></td>
</tr>
<tr>
<td>5</td>
<td>MTWRF</td>
<td>PEF TYPING</td>
<td>119</td>
<td>60</td>
<td>462</td>
<td></td>
</tr>
<tr>
<td>5</td>
<td>MTWRF</td>
<td>BUS WOOD</td>
<td>155</td>
<td>67</td>
<td>505</td>
<td></td>
</tr>
<tr>
<td>6</td>
<td>MTWRF</td>
<td>ENG 9-2</td>
<td>220</td>
<td>10</td>
<td>6</td>
<td></td>
</tr>
<tr>
<td>7</td>
<td>MTWRF</td>
<td>LUNCH 7B</td>
<td>CAF 1</td>
<td>99</td>
<td>999</td>
<td></td>
</tr>
<tr>
<td>8</td>
<td>MTWRF</td>
<td>AFR-ASIA 2</td>
<td>213</td>
<td>23</td>
<td>105</td>
<td></td>
</tr>
<tr>
<td>9</td>
<td>MTWRF</td>
<td>MATH 9-3</td>
<td>217</td>
<td>37</td>
<td>205</td>
<td></td>
</tr>
<tr>
<td>10</td>
<td>MTWRF</td>
<td>SCI 9-2</td>
<td>103</td>
<td>49</td>
<td>308</td>
<td></td>
</tr>
</tbody>
</table>

### STUDENT NAME: ALLEN CRYSTAL

**ID:** 21465  
**GRADE:** 9

<table>
<thead>
<tr>
<th>PERIOD</th>
<th>DAYS</th>
<th>COURSE</th>
<th>ROOM</th>
<th>SEM</th>
<th>TEACHER</th>
<th>CODE</th>
</tr>
</thead>
<tbody>
<tr>
<td>3</td>
<td>M W R</td>
<td>READING 9</td>
<td>207</td>
<td>17</td>
<td>73</td>
<td></td>
</tr>
<tr>
<td>3</td>
<td>T F</td>
<td>WRIT LAB</td>
<td>228</td>
<td>3</td>
<td>52</td>
<td></td>
</tr>
<tr>
<td>4</td>
<td>MTWRF</td>
<td>SPAN 1</td>
<td>227</td>
<td>56</td>
<td>410</td>
<td></td>
</tr>
<tr>
<td>5</td>
<td>MTWRF</td>
<td>LUNCH 5B</td>
<td>CAF 3</td>
<td>99</td>
<td>999</td>
<td></td>
</tr>
<tr>
<td>6</td>
<td>MTWRF</td>
<td>SCI 9-3</td>
<td>103</td>
<td>84</td>
<td>309</td>
<td></td>
</tr>
<tr>
<td>7</td>
<td>MTWRF</td>
<td>MATH 9-3</td>
<td>217</td>
<td>37</td>
<td>205</td>
<td></td>
</tr>
<tr>
<td>8</td>
<td>MTWRF</td>
<td>AFR-ASIA 3</td>
<td>215</td>
<td>22</td>
<td>106</td>
<td></td>
</tr>
<tr>
<td>9</td>
<td>T R</td>
<td>PE 9-10</td>
<td>GYM 2</td>
<td>80</td>
<td>701</td>
<td></td>
</tr>
<tr>
<td>9</td>
<td>T RF</td>
<td>PE 9-10</td>
<td>GYM 1</td>
<td>80</td>
<td>701</td>
<td></td>
</tr>
<tr>
<td>10</td>
<td>MTWRF</td>
<td>ENG 9-3</td>
<td>123</td>
<td>4</td>
<td>7</td>
<td></td>
</tr>
</tbody>
</table>

### STUDENT NAME: ALLEN MATTHEW

**ID:** 21465  
**GRADE:** 9

<table>
<thead>
<tr>
<th>PERIOD</th>
<th>DAYS</th>
<th>COURSE</th>
<th>ROOM</th>
<th>SEM</th>
<th>TEACHER</th>
<th>CODE</th>
</tr>
</thead>
<tbody>
<tr>
<td>3</td>
<td>MTWRF</td>
<td>MATH CRS 1</td>
<td>224</td>
<td>32</td>
<td>204</td>
<td></td>
</tr>
<tr>
<td>4</td>
<td>MTWRF</td>
<td>SPAN 1</td>
<td>227</td>
<td>56</td>
<td>410</td>
<td></td>
</tr>
<tr>
<td>6</td>
<td>T RF</td>
<td>PE 9-10</td>
<td>GYM 2</td>
<td>80</td>
<td>701</td>
<td></td>
</tr>
<tr>
<td>6</td>
<td>T R</td>
<td>PE 9-10</td>
<td>GYM 1</td>
<td>80</td>
<td>701</td>
<td></td>
</tr>
<tr>
<td>7</td>
<td>MTWRF</td>
<td>LUNCH 7A</td>
<td>CAF 3</td>
<td>99</td>
<td>999</td>
<td></td>
</tr>
<tr>
<td>7</td>
<td>MTWRF</td>
<td>ER SCI R</td>
<td>108</td>
<td>45</td>
<td>307</td>
<td></td>
</tr>
<tr>
<td>8</td>
<td>MTWRF</td>
<td>ER SCI R</td>
<td>108</td>
<td>45</td>
<td>307</td>
<td></td>
</tr>
<tr>
<td>9</td>
<td>MTWRF</td>
<td>AFR-ASIA 2</td>
<td>215</td>
<td>23</td>
<td>105</td>
<td></td>
</tr>
<tr>
<td>10</td>
<td>MTWRF</td>
<td>ENG 9-2</td>
<td>220</td>
<td>3</td>
<td>6</td>
<td></td>
</tr>
</tbody>
</table>
9.7. COURSE CLASS LISTS
**CLASS LIST**

**COURSE:** SS IND STD  
**NUMBER:** 100 SEC 1  
**PERIOD:** 9  
**DAYS:** MTWRF  
**SEM:** 3  
**ROOM:** 222  
**TEACHER:** 24

<table>
<thead>
<tr>
<th>ID#</th>
<th>GRADE</th>
<th>STUDENT NAME</th>
</tr>
</thead>
<tbody>
<tr>
<td>19771</td>
<td>12</td>
<td>COLASURDO CHRISTOPH</td>
</tr>
<tr>
<td>19502</td>
<td>12</td>
<td>GALEK DAWN</td>
</tr>
<tr>
<td>18778</td>
<td>12</td>
<td>HAYNES JERRY</td>
</tr>
<tr>
<td>19566</td>
<td>12</td>
<td>MILLON DENISE</td>
</tr>
<tr>
<td>19854</td>
<td>12</td>
<td>PATCHETT THOMAS</td>
</tr>
<tr>
<td>19485</td>
<td>12</td>
<td>PETERSEN PAUL</td>
</tr>
<tr>
<td>19574</td>
<td>12</td>
<td>PIAZZA JOHN</td>
</tr>
<tr>
<td>19805</td>
<td>12</td>
<td>SHULTZ MARY</td>
</tr>
<tr>
<td>19713</td>
<td>12</td>
<td>SMITH SCOTT</td>
</tr>
<tr>
<td>19820</td>
<td>12</td>
<td>VALENCE CHARLES</td>
</tr>
<tr>
<td>19635</td>
<td>12</td>
<td>WOOD J MICHAEL</td>
</tr>
</tbody>
</table>
10. PROGRAM LISTINGS

10.1. ENTSTUDDATA
**Program:** ENTSTUDDATA;

(* *************************************************** *)
(* THIS PROGRAM ALLOWS THE USER TO CREATE, AND UPDATE ALL STUDENT *)
(* INFORMATION. *)
(* *)
(* FILES USED *)
(* - ST:CRSREC(GRADE) *)
(* *)
(* UNITS USED *)
(* - UNIT ONE (IN SYSTEM.LIBRARY) *)
(* THIS UNIT CONTAINS SOME SUBROUTINES USED BY OPTIONS ONE *)
(* AND TWO OF THE PROGRAM. *)
(* *)
(* OPTIONS *)
(* *)
(* 1. OPEN A NEW FILE FOR THE FIRST TIME. *)
(* *)
(* 2. USE AN EXISTING FILE TO ENTER AND CHANGE STUDENT INFORMATION *)
(* 1. RETRIEVE A STUDENT LIST FROM A FILE. *)
(* 2. SAVE A STUDENT LIST TO A FILE. *)
(* 3. ADD A STUDENT. *)
(* 4. DELETE A STUDENT. *)
(* 5. CHANGE STUDENT DATA. *)
(* 6. QUERY LIST BY STUDENT NAME. *)
(* 7. SEND LIST OF STUDENT COURSE REQUEST RECORDS TO PRINTER. *)
(* 8. SEND LIST OF GENERAL INFORMATION RECORDS TO PRINTER. *)
(* 9. RETURN TO MAIN PROGRAM. *)
(* *)
(* 3. USE AN EXISTING FILE TO ENTER AND CHANGE COURSE RECORDS. *)
(* 1. RETRIEVE A STUDENT LIST FROM A FILE. *)
(* 2. SAVE A STUDENT LIST TO A FILE. *)
(* 3. CHANGE OR ADD STUDENT COURSE REQUESTS. *)
(* 4. QUERY LIST BY STUDENT NAME. *)
(* 5. SEND LIST OF STUDENT COURSE REQUESTS TO PRINTER. *)
(* 6. RETURN TO MAIN MENU. *)
(* *)
(* 4. EXIT PROGRAM. *)
(* *)
(* *************************************************** *)

USES ONE;

PROCEDURE OPTION1:
(* ALLOWS USER TO OPEN A FILE FOR THE FIRST TIME AND ENTER STUDENT *)
(* INFORMATION *)

VAR FLAG : INTEGER; (* REWRITE FILE IF 1 *)
CH : CHAR; (* OR N ANSWER *)

BEGIN
PAGE(OUTPUT);
HEAD1 := NIL;
HEAD2 := NIL;

(* *************************************************** *)
REPEAT
  NEW(STNODE);
  NEW(GENNODE);
  SETUP(GR);
  GOTOXY(0,1);
  WRITELN('WHEN YOU WISH TO EXIT ENTER STOP');
  GETDATA(GR,STNODE,GENNODE);
  UNTIL STNODE^.STUDNAM = 'STOP';
  PAGE(OUTPUT);
  GOTOXY(0,5);
  WRITELN('DO YOU WISH TO SAVE THIS TO A FILE?');
  GOTOXY(20,6);
  READLN(CH);
  IF CH = 'Y' THEN
    BEGIN
      FLAG := 1;
      OPENFILE(GR,FLAG);
      FILEIT(HEAD1,HEAD2);
    END;
  PAGE(OUTPUT);
  GOTOXY(0,5);
  WRITELN('DO YOU WISH TO ENTER NEW INFORMATION FOR A DIFFERENT GRADE');
  GOTOXY(20,6);
  READLN(CH);
  IF CH = 'Y' THEN OPTION1;
END;

PROCEDURE MENU0P2(VAR NUMBER : INTEGER):
  (* MENU FOR OPTION #2 *)
BEGIN
  PAGE(OUTPUT);
  GOTOXY(30,1);
  WRITELN('STUDENT LIST OPTIONS');
  GOTOXY(12,3);
  WRITELN('1. RETRIEVE A STUDENT LIST FROM A FILE.');
  GOTOXY(12,5);
  WRITELN('2. SAVE A STUDENT LIST TO A FILE.');
  GOTOXY(12,7);
  WRITELN('3. ADD A STUDENT. ');
  GOTOXY(12,9);
  WRITELN('4. DELETE A STUDENT. ');
  GOTOXY(12,11);
  WRITELN('5. CHANGE STUDENT DATA');
  GOTOXY(12,13);
  WRITELN('6. QUERY LIST BY STUDENT NAME. ');
  GOTOXY(12,15);
  WRITELN('7. SEND LIST OF STUDENT COURSE REQUEST RECORDS TO PRINTER. ');
  GOTOXY(12,17);
  WRITELN('8. SEND LIST OF GENERAL INFORMATION RECORDS TO PRINTER. ');
  GOTOXY(12,19);
  WRITELN('9. RETURN TO MAIN MENU. ');
  GOTOXY(15,21);
  WRITELN('ENTER A NUMBER FROM 1 TO 9');
  GOTOXY(15,22);
  READLN(NUMBER);
  WHILE (NUMBER < 1 ) OR (NUMBER > 9 ) DO
    BEGIN
      PAGE(OUTPUT);
      GOTOXY(30,1);
      WRITELN('STUDENT LIST OPTIONS');
      GOTOXY(12,3);
      WRITELN('1. RETRIEVE A STUDENT LIST FROM A FILE.');
      GOTOXY(12,5);
      WRITELN('2. SAVE A STUDENT LIST TO A FILE.');
      GOTOXY(12,7);
      WRITELN('3. ADD A STUDENT. ');
      GOTOXY(12,9);
      WRITELN('4. DELETE A STUDENT. ');
      GOTOXY(12,11);
      WRITELN('5. CHANGE STUDENT DATA');
      GOTOXY(12,13);
      WRITELN('6. QUERY LIST BY STUDENT NAME. ');
      GOTOXY(12,15);
      WRITELN('7. SEND LIST OF STUDENT COURSE REQUEST RECORDS TO PRINTER. ');
      GOTOXY(12,17);
      WRITELN('8. SEND LIST OF GENERAL INFORMATION RECORDS TO PRINTER. ');
      GOTOXY(12,19);
      WRITELN('9. RETURN TO MAIN MENU. ');
      GOTOXY(15,21);
      WRITELN('ENTER A NUMBER FROM 1 TO 9');
      GOTOXY(15,22);
      READLN(NUMBER);
    END;
GOTOXY(10,23);
WRITELN('YOU MUST SELECT A NUMBER FROM 1 TO 9');
GOTOXY(32,22);
READLN(NUMBER);
END
END;

PROCEDURE FETCHIT;
(* GETS STUDENT COURSE RECORDS AND GENERAL INFORMATION RECORDS AND
STORES THEM IN LINKED LISTS *)
VAR FLAG : INTEGER;
BEGIN
FLAG := 0;
HEAD1 := NIL;
HEAD2 := NIL;
PAGE(OUTPUT);
FINDGRADE(GR);
OPENFILE(GR,FLAG);
WHILE NOT EOF(STREC) DO
BEGIN
NEW(STNODE);
STNODE^ := STREC^;
IF HEAD1 = NIL
THEN HEAD1 := STNODE
ELSE INSERTST(HEAD1,STNODE);
GET(STREC)
END;
CLOSE(STREC);
WHILE NOT EOF(GENREC) DO
BEGIN
NEW(GENNODE);
GENNODE^ := GENREC^;
IF HEAD2 = NIL
THEN HEAD2 := GENNODE
ELSE INSERTGEN(HEAD2,GENNODE);
GET(GENREC)
END;
CLOSE(GENREC);
END;
PROCEDURE ADDST;
(* ADD A STUDENT RECORD TO THE LIST *)
BEGIN
NEW(STNODE);
NEW(GENNODE);
PAGE(OUTPUT);
FINDGRADE(GR);
SETUP(GR);
GETDATA(GR,STNODE,GENNODE)
END;

PROCEDURE FINDSTNODE(VAR LOOKAHEAD,CHASER,ITEM :POINT1; VAR TEST : BOOLEAN);
(* LOCATES A STUDENT COURSE REQUEST RECORD *)
BEGIN
WHILE (LOOKAHEAD <> NIL) AND (ITEM^ .STUDNAM = LOOKAHEAD^.STUDNAM) DO
BEGIN
CHASER := LOOKAHEAD;
LOOKAHEAD := LOOKAHEAD^.LINK1
END;

IF (LOOKAHEAD <> NIL) AND (ITEM^.STUDNAM = LOOKAHEAD^.STUDNAM)
THEN ITEM^ := LOOKAHEAD^:

IF (LOOKAHEAD = NIL) OR (ITEM^.STUDNAM < LOOKAHEAD^.STUDNAM)
THEN
BEGIN
  GOTOXY(10,12);
  WRITELN(ITEM^.STUDNAM, ' IS NOT IN THE STUDENT REQUEST LIST');
  TEST := TRUE;
  WAIT;
END;
END;

PROCEDURE FINDGENNODE(VAR LOOKAHEAD, CHASER, ITEM : POINT2;
VAR TEST : BOOLEAN);
(* LOCATES A GEN INFO RECORD IN THE LIST *)
BEGIN
  WHILE (LOOKAHEAD <> NIL) AND (ITEM^.STUDNAM > LOOKAHEAD^.STUDNAM)
DO BEGIN
  CHASER := LOOKAHEAD;
  LOOKAHEAD := LOOKAHEAD^.LNI-2
END;

  IF (LOOKAHEAD <> NIL) AND (ITEM^.STUDNAM = LOOKAHEAD^.STUDNAM)
THEN ITEM^ := LOOKAHEAD^:

  IF (LOOKAHEAD = NIL) OR (ITEM^.STUDNAM < LOOKAHEAD^.STUDNAM)
THEN
BEGIN
  GOTOXY(10,12);
  WRITELN(ITEM^.STUDNAM, ' IS NOT IN THE GENERAL INFORMATION LIST');
  TEST := TRUE;
  WAIT;
END;
END;

PROCEDURE WRINFO(FRONT1 : POINT1; FRONT2 : POINT2);
(* WRITES STUDENT INFORMATION TO A SETUP SCREEN *)
BEGIN
  PAGE(OUTPUT);
  SETUP(GR);
  GOTOXY(17,6);
  WRITELN(FRONT1^.STUDNAM);
  GOTOXY(17,7);
  WRITELN(FRONT1^.STUDID);
  GOTOXY(17,8);
  WRITELN(FRONT1^.GRADE);
  GOTOXY(17,9);
  WRITELN(FRONT1^.STREET);
  GOTOXY(17,10);
  WRITELN(FRONT1^.TOWN);
  GOTOXY(17,11);
  WRITELN(FRONT1^.ZIP);
  GOTOXY(17,12);
  WRITELN(FRONT1^.BIRTH);
  GOTOXY(17,13);
  WRITELN(FRONT1^.SEX);
  GOTOXY(17,14);
  WRITELN(FRONT1^.HR);
  GOTOXY(17,15);
  WRITELN(FRONT1^.PHONE);
END:
PROCEDURE DELETTEST;
(* DELETES A STUDENT REQUEST RECORD AND STUDENT INFO RECORD FROM THE LIST *)

VAR
  FLAG1, FLAG2, LOOK1, LOOK2, CHAS1, CHAS2 : BOOLEAN;
  (CRS REQ NODE FOUND *)
  (GEN INFO NODE FOUND *)
  (CRS REQ - LOOKAHEAD *)
  (GEN INFO - LOOKAHEAD *)
  (CHASER *)
  (CHASER *)
  (Y OR N - ANSWER *)
BEGIN
  NEW(STNODE);
  NEW(GENNODE);
  PAGE(OUTPUT);
  FLAG1 := FALSE;
  FLAG2 := FALSE;
  LOOK1 := HEAD1;
  LOOK2 := HEAD2;
  CHAS1 := HEAD1;
  CHAS2 := HEAD2;
  GOTOXY(0,1);
  WRITELN('WHAT STUDENT WOULD YOU LIKE TO DELETE?');
  WRITELN('ENTER THE LAST NAME FIRST');
  WRITELN('STUDENT NAME: ');
  GOTOXY(15,3);
  READLN(STNODE.STUDNAM);
  GENNODE.STUDNAM := STNODE.STUDNAM;
  FINDSTNODE(LOOK1,CHAS1,STNODE,FLAG1);
  FINDGENNODE(LOOK2,CHAS2,GENNODE,FLAG2);
  IF (FLAG1 = FALSE) AND (FLAG2 = FALSE) THEN
    BEGIN
      (* STUDENT IS IN LIST *)
      WRINFO(STNODE,GENNODE);
      GOTOXY(10,23);
      WRITE('DO YOU WANT TO MAKE THE CHANGE IN THIS RECORD? ');
      READLN(ANS);
      IF ANS = 'Y' THEN
        BEGIN
          (* DELETE STUDENT *)
          IF HEAD1.Studnam = STNODE.Studnam THEN HEAD1 := LOOK1.LINK1
          ELSE
            BEGIN
              CHAS1.LINK1 := LOOK1.LINK1;
              LOOK1.LINK1 := NIL;
            END;
          IF HEAD2.Studnam = GENNODE.Studnam THEN HEAD2 := LOOK2.LINK2
          ELSE
            BEGIN
              CHAS2.LINK2 := LOOK2.LINK2;
              LOOK2.LINK2 := NIL;
            END;
        PAGE(OUTPUT);
        GOTOXY(10,5);
        WRITELN('STUDENT REMOVED');
        WRITE( STNODE.Studnam);
        WRITELN('STUDENT HAS BEEN DELETED');
        STNODE := NIL;
        GENNODE := NIL;
        WAIT;
      END;
    END;
  END;
END;
PROCEDURE CHANGEDATA;
(* ALLOWS USER TO MAKE CHANGES IS STUDENT DATA *)

VAR
  FLAG1, FLAG2: BOOLEAN;
  SSTNODE, LOOK1, CHAS1: POINT1;
  GENNODE, LOOK2, CHAS2: POINT2;
  ANS: CHAR;

BEGIN
  NEW(STNODE);
  NEW(GENNODE);
  PAGE(OUTPUT);
  FLAG1 := FALSE;
  FLAG2 := FALSE;
  LOOK1 := HEAD1;
  LOOK2 := HEAD2;
  CHAS1 := HEAD1;
  CHAS2 := HEAD2;
  GOTOXY(0,1);
  WRITELN('WHAT STUDENT INFORMATION WOULD YOU LIKE TO CHANGE?');
  WRITELN('ENTER THE LAST NAME FIRST');
  GOTOXY(15,3);
  WRITELN('STUDENT NAME :');
  GOTOXY(0,1);
  READLN(STNODE.STUDNAM);
  GENNODE.STUDNAM := STNODE.STUDNAM;
  FINDSTNODE(LOOK1, CHAS1, STNODE, FLAG1);
  FINGENNODE(LOOK2, CHAS2, GENNODE, FLAG2);
  IF (FLAG1 = FALSE) AND (FLAG2 = FALSE) THEN
    BEGIN
      WRINFO(STNODE, GENNODE);
      GOTOXY(45,2);
      WRITELN('TO MAKE CHANGES');
      GOTOXY(45,3);
      WRITELN('REENTER ALL INFORMATION');
      NEW(STNODE);
      NEW(GENNODE);
      GOTOXY(40,0);
      READLN(STNODE.STUDNAM);
      GOTOXY(40,4);
      READLN(STNODE.STUDID);
      GOTOXY(40,5);
      READLN(STNODE.GRADE);
      GOTOXY(40,9);
      READLN(GENNODE.STREET);
      GOTOXY(40,10);
      READLN(GENNODE.TOWN);
      GOTOXY(40,11);
      READLN(GENNODE.ZIP);
      GOTOXY(40,12);
      READLN(GENNODE.BIRTH);
      GOTOXY(40,13);
      READLN(GENNODE.SEX);
      GOTOXY(40,14);
      READLN(GENNODE.HR);
      GOTOXY(40,15);
      READLN(GENNODEPHONE);
      GOTOXY(10,23);
      WRITE('DO YOU WANT TO MAKE THE CHANGE IN THIS RECORD? (Y OR N) :');
      READLN(ANS);
      IF ANS = 'Y' THEN
        ...
THEN
BEGIN
GGENNODE^.STUDNAM := SSTNODE^.STUDNAM;
GGENNODE^.STUDID := SSTNODE^.STUDID;
SSTNODE^.LINK1 := NIL;
GGENNODE^.LINK2 := NIL;
IF HEAD1^.STUDNAM = STNODE^.STUDNAM
THEN HEAD1 := LOOK1^.LINK1
ELSE
BEGIN
CHAS1^.LINK1 := LOOK1^.LINK1;
LOOK1^.LINK1 := NIL;
END;
INSERTST(HEAD1, SSTNODE);
IF HEAD2^.STUDNAM = GENNODE^.STUDNAM
THEN HEAD2 := LOOK2^.LINK2
ELSE
BEGIN
CHAS2^.LINK2 := LOOK2^.LINK2;
LOOK2^.LINK2 := NIL;
END;
INSERTGEN(HEAD2, GENNODE);
WRINFO(SSTNODE, GGENNODE);
GOTOXY(10, 2);
WRITELN('THE CHANGE HAS BEEN MADE');
WAIT;
END;
END;

PROCEDURE QUERY;
(* DISPLAYS INFORMATION FOR ANY STUDENT REQUESTED *)

VAR
FLAG1, FLAG2 : BOOLEAN; (* CRS REC NODE FOUND +
FLAG1 := FALSE;
L0OK1 := HEAD1;
CHAS1 := HEAD1;
CHAS2 := HEAD2;
GOTOXY(0, 1);
WRITELN('WHAT STUDENT WOULD YOU LIKE TO FIND?');
WRITELN('ENTER THE LAST NAME FIRST');
WRITELN('STUDENT NAME: ');
GOTOXY(15, 7);
READLN(STNODE^.STUDNAM);
GENNODE^.STUDNAM := STNODE^.STUDNAM;
FINDSTNODE(LOOK1, CHAS1, STNODE, FLAG1);
FINDGENNODENode(LOOK2, CHAS2, GENNODE, FLAG2);
IF (FLAG1 = FALSE) AND (FLAG2 = FALSE)
THEN
BEGIN
  WRINFO(STNODE,GENNODE):
  WAIT:
END;

PROCEDURE SAVEIT;
  (* SAVES STUDENT REQUEST LIST AND GEN INFO LIST TO FILES *)
VAR FLAG : INTEGER; (* REWRITE FILE IF 1 *)
BEGIN
  FLAG := 1;
  OPENFILE(GR,FLAG);
  FILEIT(HEAD1,HEAD2);
  PAGE(OUTPUT);
  GOTOXY(10,5);
  WRITELN('FILE SAVED ON DISK');
  WAIT;
END;

PROCEDURE OPTION2;
  (* SELECTIONS FOR OPTION #2 *)
VAR OPT : INTEGER; (* OPTION NUMBER *)
BEGIN
  REPEAT
    PAGE(OUTPUT);
    MENUOP2(OPT);
    CASE OPT OF
      1 : FETCHIT;
      2 : SAVEIT;
      3 : ADDST;
      4 : DELETEST;
      5 : CHANGEDATA;
      6 : QUERY;
      7 : PRINT1LIST(HEAD1);
      8 : PRINT2LIST(HEAD2);
      9 : PAGE(OUTPUT);
    END;
  UNTIL OPT = 9;
END;

PROCEDURE FETCHCRS;
  (* LOADS STUDENT REQUEST RECORDS INTO LINKED LIST *)
BEGIN
  HEAD1 := NIL;
  PAGE(OUTPUT);
  FINDGRADE(GR):
  RESET(STREC,CONCAT('ST:STREC',GR));
  WHILE NOT EOF(STREC) DO
    BEGIN
      NEW(STNODE);
      STNODE := STREC;
      IF HEAD1 = NIL
        THEN HEAD1 := STNODE
      ELSE INSERTST(HEAD1,STNODE);
      GET(STREC)
    END;
  CLOSE(STREC);
END;
PROCEDURE MENUOPs(VAR NUMBER : INTEGER);
(* DISPLAYS MENU FOR OPTION #3 *)

BEGIN
PAGE(OUTPUT);
GOTOXY(30,1);
WRITELN('STUDENT LIST OPTIONS');
GOTOXY(12,3);
WRITELN('1. RETRIEVE A STUDENT LIST FROM A FILE.');
GOTOXY(12,5);
WRITELN('2. SAVE A STUDENT LIST TO A FILE.');
GOTOXY(12,7);
WRITELN('3. CHANGE OR ADD STUDENT COURSE REQUESTS.');
GOTOXY(12,9);
WRITELN('4. QUERY LIST BY STUDENT NAME.');
GOTOXY(12,11);
WRITELN('5. SEND LIST OF STUDENT COURSE REQUEST RECORDS TO PRINTER');
GOTOXY(12,13);
WRITELN('6. RETURN TO MAIN MENU.');
GOTOXY(15,21);
WRITELN('ENTER A NUMBER FROM 1 TO 6');
GOTOXY(15,22);
READLN(NUMBER);
WHILE (NUMBER < 1) OR (NUMBER > 6) DO
BEGIN
GOTOXY(10,23);
WRITELN('YOU MUST SELECT A NUMBER FROM 1 TO 6');
GOTOXY(52,22);
READLN(NUMBER);
END;
END;

PROCEDURE FILECRS(FRONT : POINT);
(* STORES LIST OF STUDENT REQUEST RECORDS TO A FILE *)

BEGIN
IF FRONT = NIL
THEN
BEGIN
GOTOXY(10,12);
WRITELN('STUDENT REQUEST LIST IS EMPTY');
END
ELSE
BEGIN
WHILE FRONT <> NIL DO
BEGIN
STREC^ := FRONT;
STREC^.LINK1 := NIL;
PUT(STREC);
FRONT := FRONT^.LINK1
END;
END;
CLOSE(STREC,LOC);
END;

PROCEDURE SAVECRS;
(* OPENS FILE TO SAVE COURSE LIST *)

BEGIN
REWRITE:STREC,CONCAT('ST:STREC',GR));
FILECRS(HEAD1)
PROGRAM OUTPUT;
GOTOXY(10,5);
WRITELN('FILE SAVED ON DISK');
WAIT;
END;

PROCEDURE SETCRS;
(* SETS UP SCREEN FOR ENTRY OF STUDENTS COURSES *)
BEGIN
PAGE(OUTPUT);
GOTOXY(15,3);
WRITELN('STUDENT COURSE INFORMATION');
GOTOXY(0,6);
WRITELN('STUDENT NAME :');
GOTOXY(0,7);
WRITELN('STUDENT ID :');
GOTOXY(0,8);
WRITELN('GRADE');
GOTOXY(0,9);
WRITELN('COURSE 1 :');
GOTOXY(0,10);
WRITELN('COURSE 2 :');
GOTOXY(0,11);
WRITELN('COURSE 3 :');
GOTOXY(0,12);
WRITELN('COURSE 4 :');
GOTOXY(0,13);
WRITELN('COURSE 5 :');
GOTOXY(0,14);
WRITELN('COURSE 6 :');
GOTOXY(0,15);
WRITELN('COURSE 7 :');
GOTOXY(0,16);
WRITELN('COURSE 8 :');
GOTOXY(0,17);
WRITELN('COURSE 9 :');
GOTOXY(0,18);
WRITELN('COURSE 10 :');
END;

PROCEDURE WRCRS(FRONT : POINT1);
(* WRITES STUDENTS COURSES TO A SET UP SCREEN *)
BEGIN
PAGE(OUTPUT);
SETCRS;
GOTOXY(17,6);
WRITELN(FRONT^STUDNAM);
GOTOXY(17,7);
WRITELN(FRONT^STUDID);
GOTOXY(17,8);
WRITELN(FRONT^GRADE);
GOTOXY(17,9);
WRITELN(FRONT^CRSREQ1);
GOTOXY(17,10);
WRITELN(FRONT^CRSREQ2);
GOTOXY(17,11);
WRITELN(FRONT^CRSREQ3);
GOTOXY(17,12);
WRITELN(FRONT^CRSREQ4);
GOTOXY(17,13);
WRITELN(FRONT^CRSREQ5);
GOTOXY(17,14);
WRITELN(FRONT^CRSREQ6);
GOTOXY(17,15);
PROCEDURE LOCATECRS(VAR FL : BOOLEAN; VAR L1, C1 : POINT);
 (* WRITES A STUDENT COURSE RECORD TO THE SCREEN *)

BEGIN
NEW(STNODE);
FL := FALSE;
L1 := HEAD1;
C1 := HEAD1;
GOTOXY(0,2);
WRITELN('ENTER THE LAST NAME FIRST');
WRITELN('STUDENT NAME : ');
GOTOXY(15,5);
READLN(STNODE, STUDNAM);
FINDSTNODE(L1, C1, STNODE, FL);
IF FL = FALSE THEN WRITELN(STNODE);
END;

PROCEDURE QUERYCRS;
 (* CALLS LOCATORS TO FIND A STUDENTS COURSE REQUEST RECORD *)

VAR FLAG1 : BOOLEAN;
LOOK1, CHAS1 : POINT;

BEGIN
PAGE(OUTPUT);
GOTOXY(0,1);
WRITELN('WHAT STUDENT WOULD YOU LIKE TO FIND? ');
LOCATECRS(FLAG1, LOOK1, CHAS1);
WAIT;
END;

PROCEDURE SORTARRAY(VAR LIST : COURSELIST);
 (* SORTS THE COURSE ARRAY IN DECREASING ORDER *)

VAR TEMP : COURSENUMBER;
I, J : INTEGER;

BEGIN
FOR I := 1 TO MAX-1 DO
FOR J := (I + 1) TO MAX DO
IF LIST[I] < LIST[J] THEN
BEGIN
TEMP := LIST[I];
LIST[I] := LIST[J];
LIST[J] := TEMP
END;
END;

PROCEDURE NEWCRS(VAR NODE : POINT);
 (* ALLOWS USER TO INPUT COURSES AND CALLS THE SORT ROUTINE *)
BEGIN
GOTOXY(40,9);
READLN(NODE1^.CRSREQ[1]);
GOTOXY(40,10);
READLN(NODE1^.CRSREQ[2]);
GOTOXY(40,11);
READLN(NODE1^.CRSREQ[3]);
GOTOXY(40,12);
READLN(NODE1^.CRSREQ[4]);
GOTOXY(40,13);
READLN(NODE1^.CRSREQ[5]);
GOTOXY(40,14);
READLN(NODE1^.CRSREQ[6]);
GOTOXY(40,15);
READLN(NODE1^.CRSREQ[7]);
GOTOXY(40,16);
READLN(NODE1^.CRSREQ[8]);
GOTOXY(40,17);
READLN(NODE1^.CRSREQ[9]);
GOTOXY(40,18);
READLN(NODE1^.CRSREQ[10]);
SORTARRAY(NODE1^.CRSREQ);
END;

PROCEDURE CHANGEDRS;
(* ALLOWS USER TO CHANGE OR ADD COURSE TO THE STUDENT REQUEST RECORD *)
VAR
  FLAG1 : BOOLEAN;
  SSTNODE, LOOK1, CHAS1 : POINTER;
  ANS : CHAR;
BEGIN
PAGE (OUTPUT);
GOTOXY(0,1):
WRITELN('CHANGE OR ADD COURSES FOR WHAT STUDENTS');
LOCATECRS(FLAG1,LOOK1,CHAS1);
IF FLAG1 = FALSE
THEN
BEGIN
  NEW(SSTNODE):
  GOTOXY(45,2):
  WRITELN('TO MAKE CHANGES');
  GOTOXY(45,3):
  WRITELN('REENTER ALL COURSE NUMBERS');
  NEWCRS(SSTNODE);
  GOTOXY(10,22):
  WRITE('DO YOU WANT TO MAKE THE CHANGE IN THIS RECORD? Y OR N');
  READLN(ANS);
  IF ANS = 'Y'
  THEN
  BEGIN
    SSTNODE^.LINK1 := NIL;
    IF HE@G1^.STUDNAM = STNODE^.STUDNAM
    THEN HEAD1 := LOOK1^.LINK1
    ELSE
    BEGIN
      CHAS1^.LINK1 := LOOK1^.LINK1;
      LOOK1^.LINK1 := NIL;
    END;
    SSTNODE^.STUDNAM := STNODE^.STUDNAM;
    SSTNODE^.STUDID := STNODE^.STUDID;
    SSTNODE^.GRADE := STNODE^.GRADE;
    IF HEAD1 = NIL
    THEN HEAD1 := SSTNODE
    ELSE INSERTIT(HEAD1,SSTNODE);
  END;
END;
WRCS(SSTNODE);
GOTOXY(10,2);
WRITELN('THE CHANGE HAS BEEN MADE.');
WAIT;
END
END;

PROCEDURE OPTION3;
(* SELECTION FOR OPTION #3 *)
VAR OP3 : INTEGER;
(* OPTION NUMBER *)
BEGIN
REPEAT
PAGE(OUTPUT);
MENUOP3(OP3);
CASE OP3 OF
  1 : FETCHCRS;
  2 : SAVECRS;
  3 : CHANGECRS;
  4 : QUERYCRS;
  5 : PRINTILIST(HEAD1);
  6 : PAGE(OUTPUT);
END;
UNTIL OP3 = 6
END;

PROCEDURE SELECTOPTION(VAR OPTION : INTEGER);
(* MENU FOR MAIN PROGRAM *)
BEGIN
PAGE(OUTPUT);
GOTOXY(33,1):
WRITELN('OPTIONS');
GOTOXY(12,5):
WRITELN('1. OPEN NEW FILE FOR THE FIRST TIME.');
GOTOXY(12,7):
WRITELN('2. USE EXISTING FILE TO ENTER AND CHANGE STUDENT INFORMATION.');
GOTOXY(12,9):
WRITELN('3. USE EXISTING FILE TO ENTER AND CHANGE COURSE REQUESTS.');
GOTOXY(15,21):
WRITELN('4. EXIT PROGRAM.');
GOTOXY(15,22):
WRITELN('ENTER - NUMBER FROM 1 TO 4');
GOTOXY(15,22):
WRITELN('WHICH OPTION : ');
GOTOXY(15,22):
READLN(OPTION);
WHILE (OPTION < 1) OR (OPTION > 4) DO
BEGIN
GOTOXY(10,25):
WRITELN('YOU MUST SELECT A NUMBER FROM 1 TO 4');
GOTOXY(22,25):
READLN(OPTION);
END
END.
BEGIN

PAGE(OUTPUT);
HEAD1 := NIL;
HEAD2 := NIL;
PAGE(OUTPUT);

REPEAT
SELECTOPTION(NUMBER);
CASE NUMBER OF
  1  : OPTION1;
  2  : OPTION2;
  3  : OPTION3;
  4  : PAGE(OUTPUT);
END;
UNTIL NUMBER = 4;

END.  

(* ENSTUDDATA *)
10.2. UNIT ONE
/*8S++*/              (* ALLOW SWAPPING *)

UNIT ONE: INTRINSIC CODE 25 DATA 26:

(*******************************************************************************)
(*
(* THIS UNIT CONTAINS SUBROUTINES USED BY OPTIONS ONE AND TWO *)
(* OF PROGRAM 'ENTERSTUDDATA' *)
(*
*******************************************************************************)

INTERFACE

CONST
MAX = 10;        (* MAX NUMBER OF COURSES *)

TYPE
POINT1  = STUDENT;  (* TO STUDENT RECORD *)
POINT2  = GENINFO;  (* TO GENERAL INFORMATION RECORD *)
YEAR    = 7..12;
COURSENUMBER = 0..999;
COURREQLIST = ARRAY[1..MAX] OF COURSENUMBER;
STUDENT  = RECORD
  STUDNAM  : STRING[20];  (* STUDENT NAME *)
  STUDID   : INTEGER;     (* STUDENT ID NUMBER *)
  GRADE    : YEAR;        (* GRADE IN SCHOOL *)
  CRSGREQ  : COURREQLIST;(* ARRAY OF STUDENT COURSES *)
  LINK1    : POINT1;      (* POINTER TO NEXT RECORD *)
END;
GENINFO  = RECORD
  STUDNAM  : STRING[20];  (* STUDENT NAME *)
  STUDID   : INTEGER;     (* STUDENT ID NUMBER *)
  STREET   : STRING[15];  (* STREET NAME *)
  TOWN     : STRING[10];  (* TOWN NAME *)
  ZIP      : INTEGER;     (* ZIP CODE *)
  PHONE    : STRING[8];   (* PHONE NUMBER *)
  HR       : STRING[8];   (* HOME ROOM NUMBER *)
  BIRTH    : STRING[8];   (* BIRTHDATE *)
  SEX      : CHAR;        (* SEX *)
  LINK2    : POINT2;      (* POINTER TO NEXT RECORD *)
END;

VAR
STREET   : STRING[15];  (* STUDENT COURSE FILE *)
TOWN     : STRING[10];  (* GENERAL INFORMATION FILE *)
GR,
TITLE1,  : STRING;      (* MENU SELECTION *)
TITLE2,   (* MAP OF COURSE LIST *)
PHONE,   (* MAP OF COURSE LIST *)
BIRTH,   (* MAP OF COURSE LIST *)
HR,      (* MAP OF COURSE LIST *)
SEX,     (* MAP OF COURSE LIST *)
ZIP,     (* MAP OF COURSE LIST *)
NUMBER   (* MAP OF COURSE LIST *)
HR..NO1,
STNODE  : POINT1;  (* STUDENT COURSE RECORD *)
HEAD2,  : POINT2;  (* HEAD OF GEN INFO LIST *)
GENNODE : POINT2;  (* GEN INFO RECORD *)
STREC   : FILE OF STUDENT;  (* FILE OF STUDENT RECORDS *)
GENREC  : FILE OF GENINFO;  (* FILE OF GEN INFO RECORDS *)
LENGTH  : 0..MAX;  (* COUNTER *)
INDEX   : TEXT;  (* ARRAY INDEX *)
OUTDEVICE: INTL;

PROCEDURE WAIT;
PROCEDURE SETUP(Grad : STRING);
PROCEDURE INSERTST(VAR FRONT, ITEM :POINT1);
PROCEDURE INSERTGEN(VAR FRONT, ITEM :POINT2);
PROCEDURE GETDATA(Grad :STRING; VAR NODE1 :POINT1; VAR NODE2 :POINT2);
PROCEDURE OPENFILE(Grad : STRING; VAR FLAG : INTEGER);
PROCEDURE FILEIT(FRONT1 : POINT1; FRONT2 : POINT2);
PROCEDURE PRINT1LIST(FRONT1 : POINT1);
PROCEDURE PRINT2LIST(FRONT2 : POINT2);
PROCEDURE FINDGRADE(VAR GRAD : STRING);

IMPLEMENTATION

VAR
   CH : CHAR;  (* ANY CHARACTER *)

PROCEDURE WAIT;
   (* LETS THE USER DECIDE WHEN TO GO TO NEXT SCREEN *)
BEGIN
   GOTOXY(10,23);
   WRITELN('PRESS ANY KEY TO CONTINUE');
   READ(CH)
   END;

PROCEDURE SETUP;
   (* DISPLAY SCREEN FOR INPUT OF STUDENT DATA *)
BEGIN
   PAGE(OUTPUT);
   GOTOXY(13,3);
   WRITE('STUDENT INFORMATION GRADE ');
   WRITELN(Grad);
   GOTOXY(0,6);
   WRITELN('STUDENT NAME ');
   GOTOXY(0,7);
   WRITELN('STUDENT ID ');
   GOTOXY(0,8);
   WRITELN('GRADE ');
   GOTOXY(0,9);
   WRITELN('STREET ');
   GOTOXY(0,10);
   WRITELN('TOWN ');
   GOTOXY(0,11);
   WRITELN('ZIP ');
   GOTOXY(0,12);
   WRITELN('BIRTHDATE ');
   GOTOXY(0,13);
   WRITELN('SEX (M,F) ');
   GOTOXY(0,14);
   WRITELN('HOMEROOM ');
   GOTOXY(0,15);
   WRITELN('PHONE ');
END:
PROCEDURE INSERTST; (* INSERTS A STUDENT COURSE RECORD INTO LINKED LIST *)

VAR LOOKAHEAD, CHASER : POINT1; (* POINTER TO FIND PLACE *) (* POINTER FOR INSERTION *)

BEGIN
LOOKAHEAD := FRONT;
CHASER := FRONT;
WHILE (LOOKAHEAD <> NIL) AND (ITEM^.STUDNAM > LOOKAHEAD^.STUDNAM) DO
BEGIN
CHASER := LOOKAHEAD;
LOOKAHEAD := LOOKAHEAD^.LINK1
END;

IF LOOKAHEAD <> NIL
THEN
BEGIN
ITEM^.STUDNAM := LOOKAHEAD^.STUDNAM;
ITEM^.LINK1 := LOOKAHEAD;
CHASER^.LINK1 := ITEM;
END
ELSE
BEGIN
ITEM^.LINK1 := FRONT;
FRONT := ITEM;
END
ELSE
CHASER^.LINK1 := ITEM;
END;

PROCEDURE INSERTGEN; (* INSERTS GEN INFO RECORD INTO LINKED LIST *)

VAR LOOKAHEAD, CHASER : POINT2; (* GEN INFO - LOOKAHEAD *) (* - CHASER -

BEGIN
LOOKAHEAD := FRONT;
CHASER := FRONT;
WHILE (LOOKAHEAD <> NIL) AND (ITEM^.STUDNAM > LOOKAHEAD^.STUDNAM) DO
BEGIN
CHASER := LOOKAHEAD;
LOOKAHEAD := LOOKAHEAD^.LINK2
END;

IF LOOKAHEAD <> NIL
THEN
BEGIN
ITEM^.STUDNAM := LOOKAHEAD^.STUDNAM;
ITEM^.LINK2 := LOOKAHEAD;
CHASER^.LINK2 := ITEM;
END
ELSE
BEGIN
ITEM^.LINK2 := FRONT;
FRONT := ITEM;
END
ELSE
CHASER^.LINK2 := ITEM;
PROCEDURE GETDATA;
  (* READS INFORMATION INPUTED ON SETUP SCREEN *)
BEGIN
  WITH NODE1^ DO
    BEGIN
      (* GET NAME *)
      GOTOXY(15,6);
      READLN(STUDNAM);
      END;
      IF (NODE1^.STUDNAM <> 'STOP') THEN
        (* IF 'STOP' THEN EXIT *)
      BEGIN
        WITH NODE1^ DO
          BEGIN
            GOTOXY(15,7);
            READLN(STUDID);
            GOTOXY(15,8);
            READLN(GRADE);
            END;
          NODE2^.STUDNAM := NODE1^.STUDNAM;
          NODE2^.STUDID := NODE1^.STUDID;
        WITH NODE2^ DO
          BEGIN
            GOTOXY(15,9);
            READLN(STREET);
            GOTOXY(15,10);
            READLN(TOWN);
            GOTOXY(15,11);
            READLN(ZIP);
            GOTOXY(15,12);
            READLN(BIRTH);
            GOTOXY(15,13);
            READLN(SEX);
            GOTOXY(15,14);
            READLN(HR);
            GOTOXY(15,15);
            READLN(PHONE);
            END;
          NODE1^.L1NK1 := NIL;
          NODE2^.L1NK2 := NIL;
          FOR LENGTH := 1 TO MAX DO
            NODE1^.CRSEQLEN = 0;
            IF HEAD1 = NIL
              THEN HEAD1 := NODE1
              ELSE INSERTST(HEAD1,NODE1);
              IF HEAD2 = NIL
              THEN HEAD2 := NODE2
              ELSE INSERTGEN(HEAD2,NODE2);
          END;
  END;

PROCEDURE OPENFILE;
  (* OPEN FILES FOR STUDENT COURSE RECORDS AND GEN INFO RECORDS *)
BEGIN
  TITLE1 := CONCAT('ST:STREC',GRAD);
  TITLE2 := CONCAT('ST:STINFO',GRAD);
END;
IF FLAG = 1
THEN
BEGIN
  REWRITE(STREC,TITLE1);
  REWRITE(GENREC,TITLE2);
END
ELSE
BEGIN
  RESET(STREC,TITLE1);
  RESET(GENREC,TITLE2);
END;

PROCEDURE FILEIT;
(* SAVE LINKED LISTS TO FILE *)
BEGIN
  IF FRONT1 = NIL (* STUDENT COURSE REQUEST LIST *)
  THEN
    BEGIN
      GOTOXY(10,12);
      WRITELN('STUDENT REQUEST LIST IS EMPTY');
      END
    ELSE
      BEGIN
        WHILE FRONT1 <> NIL DO
          BEGIN
            STREC := FRONT1;
            STREC.LINK1 := NIL;
            PUT(STREC);
            FRONT1 := FRONT1.LINK1;
          END;
        CLOSE(STREC,LOCK);
      END;
    IF FRONT2 = NIL (* GEN INFO LIST *)
    THEN
      BEGIN
        GOTOXY(10,14);
        WRITELN('GENERAL INFORMATION LIST IS EMPTY');
      END
    ELSE
      BEGIN
        WHILE FRONT2 <> NIL DO
          BEGIN
            GENREC := FRONT2;
            GENREC.LINK2 := NIL;
            PUT(GENREC);
            FRONT2 := FRONT2.LINK2;
          END;
        CLOSE(GENREC,LOCK);
      END;
END;

PROCEDURE PRINTLIST;
(* PRINTS LIST OF STUDENT COURSE REQUESTS *)
BEGIN
  REWRITE(OUTDEVICE,'PRINTER: ');
  PAGE(OUTDEVICE);
  IF FRONT = NIL
    THEN

BEGIN
  WRITELN(OUTDEVICE,'STUDENT LIST IS EMPTY');
END
ELSE
BEGIN
  WRITELN(OUTDEVICE,'STUDENT COURSE REQUEST LIST');
  WRITELN(OUTDEVICE);
  WRITELN(OUTDEVICE,'STUDENT NAME  GR ST.NO.');
  WRITELN(OUTDEVICE);
  WHILE FRONT <> NIL DO
  BEGIN
    WITH FRONT' DO
    BEGIN
      WRITE(OUTDEVICE,STUDNAM,'');
      FOR NUMBER := 1 TO (20 - LENGTH(STUDNAM)) DO
        WRITE(OUTDEVICE,'');
      WRITE(OUTDEVICE,GRADE : 2,'');
      WRITELN(OUTDEVICE,STUDID);
      WRITE(OUTDEVICE,'');
      FOR INDEX := 1 TO MAX DO
      BEGIN
        IF CRSREQCINDEX3 > 0 THEN
          WRITE(OUTDEVICE,CRSREQCINDEX3,');
      END;
      WRITELN(OUTDEVICE);
      WRITELN(OUTDEVICE);
      FRONT := FRONT-.LINK1;
    END
  END
END
PROCEDURE PRINT2LIST;
(* PRINTS LIST OF GENERAL STUDENT INFORMATION *)
BEGIN
  REWRITE(OUTDEVICE,PRINTER:);
  PAGE(OUTDEVICE);
  IF FRONT = NIL THEN
  BEGIN
    WRITELN(OUTDEVICE,'GENERAL INFORMATION STUDENT LIST IS EMPTY');
  END
ELSE
BEGIN
  WRITELN(OUTDEVICE,'GENERAL INFORMATION STUDENT RECORDS');
  WRITELN(OUTDEVICE);
  WRITE(OUTDEVICE,STUDENT NAME ST.NO. HR BIRTH ,);
  WRITELN(OUTDEVICE,PHONE SEX');;
  WRITELN(OUTDEVICE);
  WHILE FRONT <> NIL DO
  BEGIN
    WITH FRONT' DO
    BEGIN
      WRITE(OUTDEVICE,STUDNAM,'');
      FOR NUMBER := 1 TO (20 - LENGTH(STUDNAM)) DO
        WRITE(OUTDEVICE,'');
      WRITE(OUTDEVICE,STUDID:6,HR:5,BIRTH:10,PHONE:10,SEX:4);
      WRITELN(OUTDEVICE,STREET);
      WRITELN(OUTDEVICE,'TOWN,'ZIP');
    END;
    WRITELN(OUTDEVICE);
    FRONT := FRONT-.LINK1;
  END
END
CLOSE(OUTDEVICE);
END;

PROCEDURE FINDGRADE:
(* FINDS OUT STUDENT GRADE FROM USER *)
BEGIN
PAGE(OUTPUT);
GOTOXY(0,2);
WRITELN('FOR WHAT GRADE DO YOU WISH TO ENTER STUDENTS?');
WRITELN('ENTER (7,8,9,10,11 OR 12):');
GOTOXY(30,3);
READLN(GRAD);
END;

BEGIN
(* UNIT ONE *)
END. (* UNIT ONE *)
10.3. CMATRIX
PROGRAM CMATRIX;

(* PRODUCES CONFLICT MATRIX *)
(* THE USER ENTERS *)
(* - THE RANGE OF GRADES IN THE SCHOOL *)
(* - THE NUMBER OF THE COURSE THAT HE/SHE WANTS A LIST FOR *)
(* FILES USED *)
(* - CR:COURSEFILE *)
(* - ST:STREC(GRADE) *)
(* OUTPUT *)
(* - A LIST OF EVERY COURSE WHICH WOULD CONFLICT WITH THE *)
(* INPUTED COURSE AND A TALLY OF HOW MANY STUDENTS HAVE *)
(* THAT GIVEN COMBINATION OF COURSES. *)

CONST
MAXN = 10;
MAXC = 999;

TYPE
POINT1 = 'STUDENT;
POINT2 = 'COURSE;
YEAR = 7..12;
COURSENUMBER = 0..999;
COURSESECTION = 1..99;
SEMESTER = 1..3;
PERIODNUMBER = 1..16;
TEACHERNUMBER = 1..99;
COURREQLIST = ARRAY[1..MAXSTUDENT] OF COURSENUMBER;
STUDENT = RECORD
STUDNAM : STRING[20];
STUDID : INTEGER;
GRADE : YEAR;
CRSREQ : COURREQLIST;
LIN:1 : POINT1;
END;

COURSE = RECORD
CNUM : COURSENUMBER;
CRNAM : STRING[10];
CRSEC : COURSESECTION;
MAX : INTEGER;
PERIOD : PERIODNUMBER;
PD : PERIODNUMBER;
DAY : STRING[5];
SEM : SEMESTER;
ROOM : STRING[3];
TEACHER : TEACHERNUMBER;
LIN:2 : POINT2;
END;
NAMEA  = ARRAY[1..MAXC] OF STRING[10];
TALLY  = ARRAY[1..MAXC] OF INTEGER;

VAR
  LOW, HIGH : YEAR; (* LOWEST GRADE IN SCHOOL *)
  CRSNA : NAMEA; (* HIGHEST GRADE IN SCHOOL *)
  NUMBER : COURSENUMBER; (* COURSE NAME ARRAY *)
  COUNT : TALLY; (* COURSE COUNT *)

PROCEDURE WAIT;
  (* ALLOWS USER TO DECIDE WHEN TO CONTINUE *)
  VAR CH : CHAR; (* ANY KEY *)
  BEGIN
    GOTOXY(10,23);
    WRITELN('P PRES ANY KEY TO CONTINUE');
    READ(CH)
  END;

FUNCTION ASKCHOICE(X,Y: INTEGER; Q: STRING; MIN, MAX: INTEGER) : INTEGER;
  (* ASK USER FOR A NUMBER FROM MIN TO MAX, AND REPEAT UNTIL OBTAINED *)
  VAR ANSWER : INTEGER; (* NUMBER INPUTED *)
  BEGIN
    REPEAT
      GOTOXY(X,Y);
      WRITE(Q, (',MIN,..,MAX,'), ');
      READLN(ANSWER);
      UNTIL (ANSWER MIN) AND (ANSWER MAX);
      ANSWER := ANSWER
  END;

PROCEDURE INITIALIZE( VAR N : NAMEA; VAR T : TALLY);
  (* INITIALIZES NAMEARRAY AND TALLYARRAY *)
  VAR ROW : COURSENUMBER; (* LIST INDEX *)
  BEGIN
    FOR ROW := 1 TO MAXC DO (* INITIALIZES THE NAME ARRAY *)
      N[ROW] := 0;
    FOR ROW := 1 TO MAXC DO (* INITIALIZES THE COUNT ARRAY *)
      T[ROW] := 0;
  END;

PROCEDURE GETCRSNAMES(VAR TABLE1 : NAMEA);
  (* LOADS COURSE NAMES INTO COURSE ARRAY *)
  VAR CRREC : FILE OF COURSE; (* COURSE RECORDS *)
  BEGIN
    RESET(CRREC, CR:COURSEFILE);
    WHILE NOT EOF (CRREC) DO BEGIN
      TABLE1[CRREC^.CRNUM] := CRREC^.CRNAM;
      GET (CRREC);
    END;
    CLOSE (CRREC);
PROCEDURE FILLMAT(L,H : YEAR; VAR C : TALLY; VAR N : NAME; CNUM : COURSENUMBER):
(* FINDS ALL COURSES WHICH MIGHT BE CONFLICTS *)

VAR
GETOUT : BOOLEAN;  (* EXIT LOOP *)
NULLA : COURSEQLIST;  (* NULL ARRAY *)
PRESENT : YEAR;  (* CURRENT VALUE OF YEAR *)
STREC : FILE OF STUDENT;  (* STUDENT RECORD *)
FILEGRADE, FILENAME : STRING;  (* STRING VALUE OF YEAR *)
I, J : INTEGER;  (* LOOP COUNTERS *)

BEGIN
FOR J := 1 TO MAXN DO  (* SET NULL ARRAY TO Ø *)
    NULLA[J] := 0;

GETOUT := FALSE;
PRESENT := L;
REPEAT
    STR(PRESENT, FILEGRADE);
    FILENAME := CONCAT('ST:STREC', FILEGRADE);
    RESET(STREC, FILENAME);
    WHILE NOT EOF (STREC) DO
        BEGIN
            IF STREC.CRSREQ <> NULLA
                THEN
                    BEGIN
                        FOR J := 1 TO MAXN DO
                            BEGIN
                                IF STREC.CRSREQ[J] = CNUM
                                    THEN
                                        BEGIN
                                            J := MAXN;
                                            FOR I := 1 TO MAXN DO
                                                BEGIN
                                                    WITH STREC DO
                                                        BEGIN
                                                            IF CRSREQ[I] = 0
                                                                THEN C[CRSREQ[I]] := CRSREQ[I] + 1;
                                                        END;
                                                END;
                                            END;
                                        ELSE
                                            BEGIN
                                                IF STREC.CRSREQ[J] = CNUM
                                                    THEN J := MAXN;
                                            END;
                                END;
                    END;
            GET(STREC);
        END;
    CLOSE(STREC);
    IF PRESENT = H
        THEN GETOUT := TRUE
    ELSE PRESENT := PRESENT + 1;
UNTIL GETOUT = TRUE
END;

PROCEDURE PRINTMAT(C : TALLY; N : NAME; NUM : COURSENUMBER):
(* PRINTS THE CONFLICT MATRIX *)

VAR ROW : COURSENUMBER;
OUTDEVICE : TEXT;  (* LIST INDEX *)

BEGIN
END;
BEGIN
REWRITE(OUTDEVICE,'PRINTER:');
PAGE(OUTDEVICE);
WRITE(OUTDEVICE,'CONFLICT MATRIX FOR COURSE','NUM);
WRITE(OUTDEVICE):WRITELN(OUTDEVICE);
WRITE(OUTDEVICE,'CRS NAME CODE TALLY');
WRITE(OUTDEVICE);
FOR ROW := 1 TO MAXC DO
BEGIN
IF [ROW] <> 0 THEN
BEGIN
WRITE(OUTDEVICE,'NCROW3GEST OUTDEVICE,'':11 - LENGTH(N[ROW]));
WRITE(OUTDEVICE,ROW:3,'');
WRITELN(OUTDEVICE,CCROW3:3);
END
END;
CLOSE(OUTDEVICE);
END;
BEGIN
(* CONFLICT MATRIX *)
PAGE(OUTPUT);
LOW := ASKCHOICE(C,5,'WHAT IS THE LOWEST GRADE IN THE SCHOOL',7,12);
HIGH := ASKCHOICE(2,10,'WHAT IS THE HIGHEST GRADE IN THE SCHOOL',7,12);
WAIT;
PAGE(OUTPUT);
NUMBER := ASKCHOICE(0,10,'FOR WHAT COURSE WOULD YOU LIKE A MATRIX',0,9);
WHILE NUMBER <> 0 DO
BEGIN
INITIALIZE(CRNSNA,COUNT);
GETCRSNAME(CRNSNA);
FILLMAT(LOW,HIGH,COUNT,CRNSNA,NUMBER);
PRINTMAT(COUNT,CRNSNA,NUMBER);
PAGE(OUTPUT);
WRITELN('ENTER 0 TO EXIT');
NUMBER := ASKCHOICE(0,10,'IF YOU WOULD LIKE ANOTHER COURSE MATRIX ENTER THE NUMBER : ',0,999);
END;
END.

(* CONFLICT MATRIX *)
10.4. TALLYLIST
PROGRAM TALLYLIST;

C**************************

(* PRINTS THE COURSE TALLY LIST *)
(* THE USER ENTERS *)
(* - THE RANGE OF GRADES TO BE TALLIED *)
(* FILES USED *)
(* - CR:COURSEFILE *)
(* - ST:STREC(GRADE) *)
(* OUTPUT - A LIST OF *)
(* - COURSE NAME *)
(* - COURSE NUMBER *)
(* - TOTAL COURSE ENROLLMENT *)
(* - ENROLLMENT BY GRADE *)

(*CONST* MAXN = 10; (* MAX NUMBER OF COURSES *)
MAXC = 999; (* MAX COURSE NUMBER *)

TYPE
POINT1 = 'STUDENT; (* TO COURSE RECORD *)
POINT2 = 'COURSE; (* TO COURSE RECORD *)
YEAR = 7..12;
COURSENUMBER = 0..999;
COURSESECTION = 1..99;
SEMESTER = 1..3;
PERIODNUMBER = 1..16;
TEACHERNUMBER = 1..99;
COURSELIST = ARRAY[1..MAXC] OF COURSENUMBER;

STUDENT =
RECORD
STUDNAM : STRING[20]; (* STUDENT NAME *)
STUDID : INTEGER; (* STUDENT ID NUMBER *)
GRADE : YEAR; (* GRADE IN SCHOOL *)
CRSREQ : COURSELIST; (* ARRAY OF STUDENT COURSES *)
LINK1 : POINT1; (* POINTER TO NEXT RECORD *)
END;

COURSE =
RECORD
CRNUM : COURSENUMBER; (* COURSE NUMBER *)
CRNAM : STRING[10]; (* COURSE NAME *)
CRSEC : COURSESECTION; (* COURSE SECTION *)
MAX : INTEGER; (* MAX SIZE OF CLASS *)
PERIOD : PERIODNUMBER; (* PERIOD OF DAY *)
PD : PERIODNUMBER; (* PERIOD FOR SCHEDULING *)
DAY : STRING[3]; (* DAY OF WEEK *)
SEM : SEMESTER; (* SEMESTER *)
ROOM : STRING[3]; (* ROOM NUMBER *)
TEACHER : TEACHERNUMBER; (* TEACHER NUMBER *)
LINK2 : POINT2; (* POINTER TO NEXT RECORD *)
END;

TWOID = ARRAY[1..MAXC,7..12] OF INTEGER;
NAMEA = ARRAY[1..MAXC] OF STRING[10];

VAR
LOW, HIGH : YEAR; (* LOWEST GRADE *)
MAT : TWOD; (* HIGHEST GRADE *)
CRSNA : NAMEA; (* LIST OF CRS NAMES *)

FUNCTION ASKCHOICE(X,Y : INTEGER; Q : STRING; MIN,MAX : INTEGER) : INTEGER;
(* ASK USER FOR A NUMBER FROM MIN TO MAX, AND REPEAT UNTIL OBTAINED *)
VAR ANSWER : INTEGER; (* GRADE *)
BEGIN
REPEAT
  GOTOXY(X,Y);
  WRITE(Q,' (',MIN,'..',MAX,') : ');
  READLN(ANSWER);
UNTIL (ANSWER = MIN) AND (ANSWER <= MAX);
ASKCHOICE := ANSWER
END;

PROCEDURE INITIALIZE(VAR L,H : YEAR; VAR TABLE2 : TWOD; VAR TABLE1 : NAMEA);
(* Initializes NAMEARRAY and COURSE TALLY MATRIX *)
VAR ROW : COURSENUMBER; (* TABLE INDEX - CRS NUMBER *)
    COL : YEAR; (* - GRADE *)
BEGIN
PAGE(OUTPUT);
L := ASKCHOICE(0,5, WHAT IS THE LOWEST GRADE IN THE SCHOOL?,7,12);
H := ASKCHOICE(0,10, WHAT IS THE HIGHEST GRADE IN THE SCHOOL?,7,12);
FOR ROW := 1 TO MAXC DO
  FOR COL := L TO H DO
    TABLE2[ROW,COL] := 0;
FOR ROW := 1 TO MAXC DO
  TABLE1[ROW] := '';
END;

PROCEDURE GETCRSNAME(VAR TABLE1 : NAMEA);
(* Loads COURSE NAMES INTO COURSE ARRAY *)
VAR CRREC : FILE OF COURSE; (* COURSE RECORDS *)
BEGIN
  RESET(CRREC,'CR:COURSEFILE');
  WHILE NOT EOF (CRREC) DO
    BEGIN
      TABLE1[CRREC.CRNUM] := CRREC.CRNAM;
      GET (CRREC);
    END;
  CLOSE(CRREC);
END;

PROCEDURE GETSTUDREQ(L,H : YEAR; VAR TABLE2 : TWOD; VAR TABLE1 : NAMEA);
(* TALLIES STUDENT REQUESTS BY COURSE NUMBER AND GRADE *)
BEGIN
FOR ROW := 1 TO MAXN DO
  NULLA[ROW] := 0;
GETOUT := FALSE;
PRESENT := L;
REPEAT
  STR(PRESENT, FILEGRADE);
  FILENAME := CONCAT('ST:STREC', FILEGRADE);
  RESET(STREC, FILENAME);
  WHILE NOT EOF (STREC) DO
    BEGIN
      IF STREC^.CRSREQ <> NULLA THEN
        BEGIN
          FOR ROW := 1 TO MAXN DO
            BEGIN
              IF STREC^.CRSREQ[ROW] = 0 THEN ROW := MAXN
              ELSE BEGIN
                TABLE2[STREC^.CRSREQ[ROW], PRESENT] :=
                TABLE2[STREC^.CRSREQ[ROW], PRESENT] + 1
                END;
            END;
          END;
        END;
    END;
  GET(STREC);
END;
CLOSE(STREC);
IF PRESENT = H THEN GETOUT := TRUE
ELSE PRESENT := PRESENT + 1;
UNTIL GETOUT = TRUE
END:

PROCEDURE PRINTMAT(L, H : YEAR; VAR TABLE2 : TWD; VAR TABLE1 : NAMEA); 
(* PRINTS THE TALLIES LIST *)

VAR ROW : COURSEN NUMBER;
      COL : YEAR;
      ENROL : INTEGER;
      OUTDEVICE : TEXT;
BEGIN
  REWRITE(OUTDEVICE, PRINTER: );
  PAGE(OUTDEVICE);
  WRITELN(OUTDEVICE, SODUS: '37');
  WRITELN(OUTDEVICE, 'STUDENT TALLY OF COURSES OFFERED : %0);
  WRITELN(OUTDEVICE, 'STUDENT TALLY OF COURSES OFFERED : %0);
  WRITE(OUTDEVICE, ' ');
  FOR COL := L TO H DO
    WRITE(OUTDEVICE, ' ');
  WRITELN(OUTDEVICE);
  WRITE(OUTDEVICE, 'COURSE CODE TOTAL ');
  FOR COL := L TO H DO
    WRITE(OUTDEVICE, COL: 3, ' ');
  WRITELN(OUTDEVICE):
FOR ROW := 1 TO MAXC DO
BEGIN
    IF TABLE1[ROW] <> ' ' THEN BEGIN
        ENROL := 0;
        WRITE(OUTDEVICE,TABLE1[ROW], ' ');
        WRITE(OUTDEVICE,':11 - LENGTH(TABLE1[ROW]));
        WRITE(OUTDEVICE,ROW:3, ' ');
        FOR COL := L TO H DO
            ENROL := ENROL + TABLE2[ROW, COL];
        WRITE(OUTDEVICE,ENROL:3, ' ');
        FOR COL := L TO H DO
            WRITE(OUTDEVICE,TABLE2[ROW, COL]:3, ' ');
        WRITELN(OUTDEVICE);
    END
END;
BEGIN
INITIALIZE(LOW, HIGH, MAT, CRSNA);
GETCRSNAMES(CRSNA);
GETSTUDREQ(LOW, HIGH, MAT, CRSNA);
PRINTMAT(LOW, HIGH, MAT, CRSNA);
END.
10.5. ENTCOURSEDATA
PROGRAM ENTCOURSEDATA;

(* THIS PROGRAM STORES COURSE RECORDS IN CR:COURSEFILE *)
(* THE USER HAS 10 OPTIONS *)
(* 1. ENTER COURSE NUMBERS AND NAMES FOR FIRST TIME. *)
(* 2. LOAD THE COURSEFILE FROM DISK. *)
(* 3. ENTER OTHER COURSE INFORMATION. *)
(* 4. ADD COURSES. *)
(* 5. DELETE COURSES. *)
(* 6. CHANGE COURSE DATA. *)
(* 7. QUERY FILE BY COURSE NUMBER. *)
(* 8. SEND COURSE LIST TO PRINTER. *)
(* 9. SAVE COURSE LIST TO DISK. *)
(* 10. EXIT PROGRAM. *)

END:

VAR

CRNUM : COURSENUMBER;
CRNAM : STRING[10];
CRSEC : COURSESECTION;
MAX : INTEGER;
PERIOD : PERIODNUMBER;
PD : PERIODNUMBER;
DAY : STRING[5];
SEM : SEMESTER;
ROOM : STRING[3];
TEACHER : TEACHERNUMBER;
LINK : POINTER;

CRREC : FILE OF COURSE;

END:

TYPE

POINTER = *COURSE;
COURSENUMBER = 0..999;
COURSESECTION = 1..99;
SEMESTER = 1..3;
PERIODNUMBER = 1..16;
TEACHERNUMBER = 1..99;
COURSE = RECORD

CRNUM : COURSENUMBER;
CRNAM : STRING[10];
CRSEC : COURSESECTION;
MAX : INTEGER;
PERIOD : PERIODNUMBER;
PD : PERIODNUMBER;
DAY : STRING[5];
SEM : SEMESTER;
ROOM : STRING[3];
TEACHER : TEACHERNUMBER;
LINK : POINTER;

(* FILE BUFFER *)

(* FILE BUFFER *)
PROCEDURE WAIT;  
(* ALLOWS USER TO DECIDE WHEN TO CONTINUE *)
VAR CH : CHAR;  
(* INPUT CHAR *)
BEGIN 
GOTOXY(10,23); 
WRITELN('[ PRESS ANY KEY TO CONTINUE]'); 
READ(CH) 
END;

PROCEDURE GETNUM(X,Y : INTEGER; VAR NUM : INTEGER; L,H:INTEGER); 
(* ERROR CK - IF A CHAR IS ENTERED INSTEAD OF AN INTEGER 
USER WILL BE ASKED TO RE-ENTER NUMBER *)
BEGIN 
(*#I-*)  
READLN(NUM);  
(*#I+*)  
WHILE (NUM < L) OR (NUM > H) DO 
BEGIN 
   GOTOXY(10,22); 
   WRITELN('ERROR IN INPUT - REENTER A NUMBER FROM ',L,' TO ',H); 
   GOTOXY(X,Y); 
   (*#I-*)  
   READLN(NUM);  
   (*#I+*)  
   GOTOXY(10,22); 
   WRITELN(''); 
END; 
END;

PROCEDURE ENTERCOURSESETUP; 
(* SETS UP SCREEN FOR ENTERING A COURSE RECORD *)
BEGIN 
GOTOXY(15,3); 
WRITELN(' A COURSE RECORD'); 
GOTOXY(0,6); 
WRITELN('COURSE NUMBER : '); 
GOTOXY(0,7); 
WRITELN('COURSE NAME : '); 
GOTOXY(0,8); 
WRITELN('COURSE SECTION : '); 
GOTOXY(0,9); 
WRITELN('MAXSIZE : '); 
GOTOXY(0,10); 
WRITELN('PERIOD NUMBER : '); 
GOTOXY(0,11); 
WRITELN('PD NUMBER : '); 
GOTOXY(0,12); 
WRITELN('DAYS : '); 
GOTOXY(0,13); 
WRITELN('SEMESTER : '); 
GOTOXY(0,14); 
WRITELN('ROOM NUMBER : '); 
GOTOXY(0,15); 
WRITELN('TEACHER NUMBER : ');
PROCEDURE INSERTNODE (VAR FRONT, ITEM : POINTER);
(* TO PUT NEW NODE IN LIST IN ORDER OF COURSE, SECTION AND PD NUMBER *)
VAR LOOKAHEAD, (* LOOKAHEAD POINTER TO FIND PLACE *)
    CHASER : POINTER; (* CHASER POINTER FOR INSERTION *)
BEGIN (* INSERTNODE *)
    LOOKAHEAD := FRONT;
    CHASER := FRONT;
    WHILE (LOOKAHEAD <> NIL) AND (ITEM^.CRNUM > LOOKAHEAD^.CRNUM) DO
        CHASER := LOOKAHEAD;
        LOOKAHEAD := LOOKAHEAD^.LINK;
    END; (* WHILE *)
    WHILE (LOOKAHEAD <> NIL) AND (ITEM^.CRNUM = LOOKAHEAD^.CRNUM) AND
        (ITEM^.CRSEC > LOOKAHEAD^.CRSEC) DO
        CHASER := LOOKAHEAD;
        LOOKAHEAD := LOOKAHEAD^.LINK;
    END; (* WHILE *)
    WHILE (LOOKAHEAD <> NIL) AND (ITEM^.CRNUM = LOOKAHEAD^.CRNUM) AND
        (ITEM^.CRSEC = LOOKAHEAD^.CRSEC) AND (ITEM^.PD > LOOKAHEAD^.PD) DO
        CHASER := LOOKAHEAD;
        LOOKAHEAD := LOOKAHEAD^.LINK;
    END; (* WHILE *)
    IF (LOOKAHEAD <> NIL) THEN
        IF LOOKAHEAD <> FRONT THEN
            BEGIN
                ITEM^.LINK := LOOKAHEAD;
                CHASER^.LINK := ITEM;
            END
        ELSE
            BEGIN
                ITEM^.LINK := FRONT;
                FRONT := ITEM;
            END
        ELSE
            CHASER^.LINK := ITEM;
    END;

PROCEDURE PRINTLIST (FRONT : POINTER);
(* PRINTS COURSE LIST GIVEN THE POINTER TO START OF LIST *)
VAR OUTDEVICE : TEXT; (* PRINTER *)
    PREV : COURSENUMBER; (* PREV COURSE NUMBER *)
    PREVS : COURSESECTION; (* PREV SECTION NUMBER *)
BEGIN
    REWRITE (OUTDEVICE, 'PRINTER:' );
    PAGE (OUTDEVICE);
IF FRONT = NIL
THEN (* LIST IS EMPTY *)
BEGIN
  WRITELN(OUTDEVICE,'LIST IS EMPTY.');
END
ELSE (* THERE ARE COURSES *)
BEGIN
  WRITELN(OUTDEVICE,'COURSE LIST - SODUS':42);
  WRITELN(OUTDEVICE):
  WRITE(OUTDEVICE,'CODE SEC SEATS SEM PD DAYS MET ');
  WRITELN(OUTDEVICE,DESCRIPTION ROOM TCH PER');
  PREV := 0;
  PREVS := 1;
  WHILE FRONT <> NIL DO
    BEGIN
      WITH FRONT DO
      BEGIN
        IF PREV < CRNUM THEN WRITELN(OUTDEVICE);
        IF (PREV = CRNUM) AND (PREVS < CRSEC)
          THEN WRITELN(OUTDEVICE);
        WRITE(OUTDEVICE,CRNUM:3,' ',CRSEC:2,' ',MAX:3,
              ' ',CRNAM:10,
              WRITELN(OUTDEVICE,ROOM:3,' ',TEACHER:2,' ',PERIOD:2);
      END;
      PREV := FRONT^.CRNUM;
      PREVS := FRONT^.CRSEC;
      FRONT := FRONT^.LINK;
    END;
  END;
  WRITELN(OUTDEVICE);WRITELN(OUTDEVICE);
END;

PROCEDURE FILEIT (FRONT : POINTER):
(* SENDS COURSE RECORD TO FILE NAMED COURSEFILE *)
BEGIN
  REWRITE(CRREC,'CR:COURSEFILE');
  IF FRONT = NIL
    THEN (* LIST IS EMPTY *)
    BEGIN
      GOTOXY(10,12);
      WRITELN('LIST IS EMPTY.');
    END
  ELSE (* THERE ARE COURSES *)
  BEGIN
    WHILE FRONT <> NIL DO
      BEGIN
        CRREC^ := FRONT^;
        CRREC^.LINK := NIL;
        PUT(CRREC);
        FRONT := FRONT^.LINK;
      END;
    END;
  END;
END;

PROCEDURE SETUP:
(* FORMATS SCREEN FOR ENTRY OF COURSE NUMBER AND NAME *)
BEGIN
  GOTOXY(0,10):
  WRITELN('ENTER COURSE NUMBER : ');
  GOTOXY(0,12):
PROCEDURE OPTION1:
(* Allows user to enter course number and name for first time
   Dummy values added for other fields *)
BEGIN
PAGE(OUTPUT);
WRITELN('ENTER COURSE NUMBER');
WRITELN('ON NEXT LINE ENTER COURSE NAME');
WRITELN('WHEN YOU ARE FINISHED ENTER 999 AND STOP');
WRITELN:
SETUP;
NEW(NODE);
WITH NODE` DO
BEGIN
GETNUM(22,10,CRNUM,1,999):
IF CRNUM <> 999
THEN
BEGIN
GOTOXY(20,15):
READLN(CRNAM):
CRSEC := 1;
MAX := 35;
PERIOD := 1;
PD := 1;
DAY := 'MTWRF';
SEM := 3:
ROOM := 'N';
TEACHER := 99;
LINK := NIL;
END:
END:
WHILE NODE^.CRNUM <> 999 DO
BEGIN
IF HEAD = NIL
THEN HEAD := NODE
ELSE INSERTNODE (HEAD, NODE):
NEW(NODE):
SETUP:
WITH NODE` DO
BEGIN
GETNUM(22,10,CRNUM,1,999):
GOTOXY(20,15):
READLN(CRNAM):
CRSEC := 1:
MAX := 35:
PERIOD := 1:
PD := 1:
DAY := 'MTWRF';
SEM := 3:
ROOM := 'N';
TEACHER := 99:
LINK := NIL:
END:
END:
END;
PROCEDURE FETCHITCVAR (VAR FIRST : POINTER):
(* Retrieves course list from file and enters courses in a
   linked list *)
VAR CRS : POINTER;

BEGIN
RESET(CRREC,'CR:COURSEFILE');

WHILE NOT EOF(CRREC) DO
BEGIN

NEW(CRS);
CRS := CRREC;
IF FIRST = NIL
THEN FIRST := CRS
ELSE INSERTNODE(FIRST,CRS);
GET(CRREC);
END;
CLOSE(CRREC);
END;

PROCEDURE MENU(VAR NUMBER : INTEGER);
(* SENDS MAIN MENU TO SCREEN AND WAITS FOR USER CHOICE *)

BEGIN
PAGE(OUTPUT);
GOTOXY(30,3);
WRITELN('COURSE LIST OPTIONS');
GOTOXY(12,5);
WRITELN('1. ENTER COURSE NUMBERS AND NAMES FOR FIRST TIME. ');
GOTOXY(12,6);
WRITELN('2. LOAD COURSE FILE FROM DISK.');
GOTOXY(12,7);
WRITELN('3. ENTER OTHER COURSE INFORMATION. ');
GOTOXY(12,8);
WRITELN('4. ADD COURSES.');
GOTOXY(12,9);
WRITELN('5. DELETE COURSES. ');
GOTOXY(12,10);
WRITELN('6. CHANGE COURSE DATA. ');
GOTOXY(12,11);
WRITELN('7. QUERY FILE BY COURSE NUMBER. ');
GOTOXY(12,12);
WRITELN('8. SEND COURSE LIST TO PRINTER. ');
GOTOXY(12,13);
WRITELN('9. SAVE COURSE LIST TO DISK. ');
GOTOXY(12,14);
WRITELN('10. EXIT PROGRAM. ');
GOTOXY(15,20);
WRITELN('ENTER A NUMBER FROM 1 TO 10');
GOTOXY(15,21);
WRITELN('WHICH OPTION: ');
GOTOXY(32,21);
GETNUM(32,21,NUMBER,1,10)
END;

PROCEDURE ENTERINFO(FRONT : POINTER);
(* ALLOWS USER TO INPUT INFORMATION MISSING FROM A COURSE RECORD *)

BEGIN
WHILE FRONT <> NIL DO
BEGIN
PAGE(OUTPUT);
ENTERCOURSESETUP;
GOTOXY(5,22);
WRITE('FOR COURSE NUMBER :');
WRITE(FRONT^.CRNUM);
WRITELN(' FILL IN THE MISSING INFORMATION');
GOTOXY(20,6);
WRITE(FRONT^.CRNUM);
GOTOXY(20,7);
WRITELN(FRONT^.CRNAM);
GOTOXY(20,8);
GETNUM(20,8,FRONT^.CRSEC,1,99);
GOTOXY(20,9);
GETNUM(20,9,FRONT^.MAX,1,999);
GOTOXY(20,10);
GETNUM(20,10,FRONT^.PERIOD,1,16);
GOTOXY(20,11);
GETNUM(20,11,FRONT^.PD,1,16);
GOTOXY(20,12);
READLN(FRONT^.DAY);
GOTOXY(20,13);
GETNUM(20,13,FRONT^.SEM,1,3);
GOTOXY(20,14);
READLN(FRONT^.ROOM);
GOTOXY(20,15);
GETNUM(20,15,FRONT^.TEACHER,1,99);
FRONT := FRONT^.LINK
END;
PROCEDURE ADDC;
(* ALLOWS USER TO INPUT A NEW COURSE TO LIST *)
BEGIN
PAGE(OUTPUT);
NEW(NODE);
ENTERCOURSESETUP;
GOTOXY(20,6);
GETNUM(20,6,NODE^.CRNUM,1,999);
GOTOXY(20,7);
READLN(NODE^.CRNAM);
GOTOXY(20,8);
GETNUM(20,8,NODE^.CRSEC,1,99);
GOTOXY(20,9);
GETNUM(20,9,NODE^.MAX,1,999);
GOTOXY(20,10);
GETNUM(20,10,NODE^.PERIOD,1,16);
GOTOXY(20,11);
GETNUM(20,11,NODE^.PD,1,16);
GOTOXY(20,12);
READLN(NODE^.DAY);
GOTOXY(20,13);
GETNUM(20,13,NODE^.SEM,1,3);
GOTOXY(20,14);
READLN(NODE^.ROOM);
GOTOXY(20,15);
GETNUM(20,15,NODE^.TEACHER,1,99);
NODE^.LINK := NIL;
INSERTNODE(HEAD,NODE);
END;
PROCEDURE FINDNODE (VAR LOOKAHEAD,CHASER, ITEM : POINTER; VAR TEST,DUF : BOOLEAN):
(* TO LOCATE A NODE USING THE COURSE NUMBER *)
IF ANOTHER NODE WITH SAME CR NUMBER EXISTS DUP IS SET TRUE*)

VAR TEMPA : POINTER; (* TEMP LOOKAHEAD POINTER *)

BEGIN

WHILE (LOOKAHEAD <> NIL) AND (ITEM^.CRNUM > LOOKAHEAD^.CRNUM) DO
BEGIN
CHASER := LOOKAHEAD;
LOOKAHEAD := LOOKAHEAD^.LINK;
END;

IF (LOOKAHEAD <> NIL) AND (ITEM^.CRNUM = LOOKAHEAD^.CRNUM) THEN
BEGIN
ITEM^ := LOOKAHEAD^;
TEMPA := LOOKAHEAD^.LINK;
IF (TEMPA <> NIL) AND (ITEM^.CRNUM = TEMPA^.CRNUM) THEN DUP := TRUE
END;

IF (LOOKAHEAD = NIL) OR (ITEM^.CRNUM < LOOKAHEAD^.CRNUM) THEN
BEGIN
GOTOXY(10,12);
WRITE('COURSE ');
WRITEC(ITEM^.CRNUM);
WRITELN(' IS NOT IN THE LIST');
TEST := TRUE;
WAIT;
END;

PROCEDURE WRINFO(FRONT : POINTER):
(* WRITE A COURSE RECORD TO SCREEN *)
BEGIN
PAGE(OUTPUT):
ENTERCOURSESETUP:
GOTOXY(20,6):
WRITELN(FRONT^.CRNUM):
GOTOXY(20,7):
WRITELN(FRONT^.CRNAM):
GOTOXY(20,8):
WRITELN(FRONT^.CRSEC):
GOTOXY(20,9):
WRITELN(FRONT^.MAX):
GOTOXY(20,10):
WRITELN(FRONT^.PERIOD):
GOTOXY(20,11):
WRITELN(FRONT^.PD):
GOTOXY(20,12):
WRITELN(FRONT^.DAY):
GOTOXY(20,13):
WRITELN(FRONT^.SEM):
GOTOXY(20,14):
WRITELN(FRONT^.ROOM):
GOTOXY(20,15):
WRITELN(FRONT^.TEACHER):
END;

PROCEDURE CKDUP(ST:STRING;VAR L,C,N:POINTER;VAR F,D:BOOLEAN):
(* ALLOWS USER TO CHECK THE COURSE WITH SAME CR NUMBER *)
VAR ANS : CHAR;  (* Y OR N - ANSWER *)

BEGIN
GOTOXY(0,17):
WRITE('THERE IS ANOTHER COURSE WITH NUMBER ',N^,CRNUM):
WRITE('WOULD YOU LIKE TO ',ST,' THE OTHER SECTION? '):
WRITE((ENTER Y OR N): '):
READLN(ANS):
IF ANS = 'Y'
  THEN
    BEGIN
      FINDNODE(L,C,N,F,D):
      PAGE(OUTPUT):
      WRINFO(N)
    END
  ELSE F := TRUE;
END;

PROCEDURE QUERY:
  (* USES COURSE NUMBER WHICH IS INPUTED TO LOCATE A COURSE *)

VAR DUP,
  FLAG :BOOLEAN;
  LOOK,
  CHAS :POINTER;

BEGIN
  NEW(NODE);
  PAGE(OUTPUT);
  FLAG := FALSE;
  DUP := FALSE;
  LOOK := HEAD;
  CHAS := HEAD;
  GOTOXY(0,2):
  WRITELN('WHAT COURSE WOULD YOU LIKE TO FIND?');
  WRITELN('ENTER THE COURSE NUMBER');
  GOTOXY(16,4):
  GETNUM(16,4,NODE^,CRNUM,1,999):
  FINDNODE(LOOK,CHAS,NODE,FLAG,DUP);
  IF FLAG = FALSE THEN
    BEGIN
      WRINFO(NODE):
      WHILE DUP DO
        BEGIN
          CHAS := LOOK:
          LOOK := LOOK^,LINK:
          DUP := FALSE;
          CKDUP('FIND',LOOK,CHAS,NODE,FLAG,DUP):
        END:
      END;
      WAIT;
    END
END;

PROCEDURE REMOVE(VAR LOOK,CHAS,NODE : POINTER; VAR R:BOOLEAN);

VAR ANS : CHAR;  (* Y OR N - ANSWER *)

BEGIN
GOTOXY(10,23):
WRITE('DO YOU WANT TO DELETE THIS RECORD? ' (Y OR N) '):
READLN(ANS):
IF ANS = 'Y'
  THEN
BEGIN
IF (HEAD^.CRNUM = NODE^.CRNUM) AND (HEAD^.CRSEC = HEAD^.CRSEC) AND
(HEAD^.PD = NODE^.PD)
THEN
BEGIN
HEAD := LOOK^.LINK;
END
ELSE
BEGIN
CHAS^.LINK := LOOK^.LINK;
LOOK^.LINK := NIL;
LOOK := CHAS^.LINK;
END;
PAGE(OUTPUT);
GOTOXY(0,5);
WRITE('COURSE ');
WRITE(NODE^.CRNUM);
WRITELN(' IS DELETED');
R := TRUE;
END;

PROCEDURE DELETEC:
((* REMOVES A COURSE FROM THE LIST *)

VAR DUF, REM, FLAG :BOOLEAN;
LOOK, CHAS :POINTER;
ANS :CHAR;

BEGIN
NEW(NODE);
PAGE(OUTPUT):
FLAG := FALSE;
DUP := FALSE;
REM := FALSE;
LOOK := HEAD;
CHAS := HEAD;
GOTOXY(0,2);
WRITELN('WHAT COURSE WOULD YOU LIKE TO DELETE?');
WRITELN('ENTER THE COURSE NUMBER');
WRITELN('COURSE NUMBER: ');;
GOTOXY(16,4);
GETNUM(16,4,NODE^.CRNUM,1,999);
FINDNODE(LOOK,CHAS,NODE,FLAG,DUP);
IF FLAG = FALSE
THEN
BEGIN
WRINFO(NODE);
REMOVE(LOOK,CHAS,NODE,REM);
WHILE DUP DO
BEGIN
IF REM = FALSE
THEN
BEGIN
CHAS := LOOK;
LOOK := LOOK^.LINK;
END;
REM := 'FALSE';
END;
DUP := FALSE;
CKDUP('FIND',LOOK,CHAS,NODE,FLAG,DUP);
IF NOT FLAG
  THEN REMOVE(LOOK,CHAS,NODE,REM);
END;
END;

PROCEDURE CHANGEC;
(* ALLOWS FOR CHANGE OF ANY FIELD IN COURSE RECORD *)
VAR DUP,
    FLAG :BOOLEAN;
    NNODE,
    LOOK,
    CHAS :POINTER;
    ANS :CHAR;
BEGIN
GOTOXY(0,17):
WRITE('IS THIS THE RECORD YOU WISH TO CHANGE? (ENTER Y OR N) : ');
READLN(ANS);
IF ANS = 'N'
  THEN
BEGGIN
  WHILE D DO
  BEGIN
GOTOXY(0,19):
WRITE('THERE IS ANOTHER RECORD WITH THE SAME CR NUMBER' );
WRITE('WOULD YOU RATHER CHANGE THAT RECORD? (ENTER Y OR N) : ');
READLN(ANS);
IF ANS = 'Y'
  THEN
BEGIN
  C := L;
  L := L .LINK;
  D := FALSE;
  FINDNODE(L,C,N,F,D);
  PAGE(OUTPUT);
  WRINFO(NODE)
END
ELSE D := FALSE;
END
END
END;

BEGIN
NEW(NODE);
PAGE(OUTPUT);
FLAG := FALSE;
DUP := FALSE;
LOOK := HEAD;
CHAS := HEAD;
GOTOXY(0,2):
WRITELN('WHAT COURSE WOULD YOU LIKE TO CHANGE? ');
WRITELN('ENTER THE COURSE NUMBER ');
WRITELN('COURSE NUMBER : ');
GOTOXY(16,4);
GETNUM('16.4,NODE.CNUM',I);
FINDNODE(LOOK, CHAS, NODE, FLAG, DUP);
IF FLAG = FALSE
THEN
BEGIN
WRINFO(NODE);
DUPNODE(NODE, LOOK, CHAS, FLAG, DUP);
GOTOXY(45, 2);
WRITELN('MAKE CHANGES TO THIS RECORD');
GOTOXY(45, 3);
WRITE('ENTER Y OR N: ');
READLN(ANS);
IF ANS = 'Y' THEN
BEGIN
GOTOXY(45, 4);
WRITELN('ENTER ALL INFORMATION');
NEW(NNODE);
GETNUM(40, 6, NNODE^.CRNUM, 1, 999);
GOTOXY(40, 7);
READLN(NNODE^.CRNUM);
GOTOXY(40, 8);
GETNUM(40, 8, NNODE^.CRSEC, 1, 99);
GOTOXY(40, 9);
GETNUM(40, 9, NNODE^.MAX, 1, 999);
GOTOXY(40, 10);
GETNUM(40, 10, NNODE^.PERIOD, 1, 16);
GOTOXY(40, 11);
GETNUM(40, 11, NNODE^.PD, 1, 16);
GOTOXY(40, 12);
READLN(NNODE^.DAY);
GOTOXY(40, 13)
GETNUM(40, 13, NNODE^.SEM, 1, 3);
GOTOXY(40, 14);
READLN(NNODE^.ROOM);
GOTOXY(40, 15);
GETNUM(40, 15, NNODE^.TEACHER, 1, 99);
GOTOXY(10, 23);
WRITE('DO YOU WANT TO MAKE THE CHANGE IN THIS RECORD? Y OR N: ');
READLN(ANS);
IF ANS = 'Y' THEN
BEGIN
NNODE^.LINK := NIL;
IF HEAD^.CRNUM = NODE^.CRNUM THEN
BEGIN
HEAD := LOOK^.LINK;
END
ELSE
BEGIN
CHAS^.LINK := LOOK^.LINK;
LOOK^.LINK := NIL;
END;
INSERTNODE(HEAD, NNODE);
WRINFO(NNODE);
GOTOXY(10, 2);
WRITELN('THE CHANGE HAS BEEN MADE');
WAIT;
END;
END;
END;}
PROCEDURE SELECTOPTION;
   (* ALLOWS USER TO SELECT AN OPTION *)

VAR OPTION : INTEGER;      (* OPTION NUMBER *)

BEGIN

   REPEAT
      PAGE(OUTPUT);
      MENU(OPTION);

      CASE OPTION OF
         1 : OPTION1;
         2 : FETCHIT(HEAD);
         3 : ENTERINFO(HEAD);
         4 : ADDC;
         5 : DELETEx;
         6 : CHANGEx;
         7 : QUERY;
         8 : PRINTLIST(HEAD);
         9 : FILEIT(HEAD);
        10 : PAGE(OUTPUT);
      END;
   UNTIL OPTION = 10;

   END;

BEGIN
   HEAD := NIL:
   PAGE(OUTPUT);
   SELECTOPTION;

   END.  (* ENTERCOURSEDATA *)
10.6. SCHEDULER
PROGRAM SCHEDULER:

(* THIS PROGRAM SCHEDULES STUDENTS AND PRINTS THE RESULTS OF THE *
(* SCHEDULING RUN. *)
(* THE USER ENTERS *
(* - RUN TYPE (TRIAL OR FINAL) *)
(* - GRADE LEVELS *)
(* FILES USED *
(* - ST:CRREC(GRADE) *)
(* - CR:COURSEFILE *)
(* - SCH:SCH(GRADE) *)
(* UNIT USED *
(* - UNIT TWO (IN SYSTEM.LIBRARY) *)
(* THIS UNIT CONTAINS PROCEDURES USED IN SCHEDULING STUDENTS *)
(* OUTPUT *
(* - SCHEDULES PRINTED FOR CONFLICT SCHEDULES. *)
(* - NUMBER OF STUDENTS ASSIGNED TO STUDY HALL ARE PRINTED BY GRADE LEVEL. *)
(* - SCHEDULING RESULTS ARE REPORTED BY GRADE LEVEL. *)
(* - CLASS TALLY RESULTS ARE PRINTED BY CRS AND SECTION *)
(* - ALL STUDENTS ARE SCHEDULED IN EACH RUN BUT THE SCHEDULES ARE ONLY SAVED ON THE FINAL RUN TO FILE (SCH:SCH(GRADE)) *)

USES TWO:

VAR
CRNSCH1, CRNSCH2, (* 1ST COURSE NOT SCHEDULED*)
CRNUNUM : COURSENUMBER; (* 2ND COURSE NOT SCHEDULED*)
RUNITYPE, (* CRS NUMBER *)
CRNAME : STRING[10]; (* TRIAL OR FINAL *)
CRSEC : COURSESECTION; (* CRS SECTION *)
TEST : INTEGER; (* COUNTS RECORDS FOR FINAL RUN *)
PD, PERIOD : PERIODNUMBER; (* SCHEDULING PERIOD *)
DAY : STRING[5]; (* PERIOD IN DAY *)
TEACHER : TEACHERNUMBER; (* DAYS SCH IN WEEK *)
SEM : SEMESTER; (* TEACHER NUMBER *)
ROOM : STRING[3]; (* SEM NUMBER *)
ANS : CHAR; (* ROOM NUMBER *)
NODE : PTR; (* YES OR NO (Y OR N) *)
SH1, (* POINTER TO COURSE REC *)
SH2, (* STUDY HALL SCH - 1ST SEM *)
SB1, (* STUDY HALL SCH - 2ND SEM *)
S1, (* TEMP SCHEDULE - 1ST SEM *)
SB2, (* TEMP SCHEDULE - 2ND SEM *)
S2 (* SCHEDULE - 1ST SEM *)
KUN : CHAR; (* TRIAL OR FINAL RUN *)
PROGRAM STUDSCHEDULE;

CONST
ALT1 = 700; (* FIRST ALTERNATE *)
ALT2 = 701; (* SECOND ALTERNATE *)

VAR
STNODE : POINT1; (* STUDENT COURSE RECORD *)
STREC : FILE OF STUDENT; (* FILE OF STUDENT RECORDS *)
LEN,
FULL, (* NUMBER OF FULL SCHEDULES *)
PART, (* NUMBER OF PARTIAL SCHED *)
IRR : INTEGER; (* NUMBER OF IRR CONFLICTS *)
ALTUSED, (* MARKS IF ALT IS USED *)
NOTOPEN, (* FILE IS NOT YET OPEN *)
ALTOF, (* TURNS ALT OFF *)
GETOUT : BOOLEAN; (* EXIT LOOP *)
PRESENT : YEAR; (* CURRENT VALUE OF YEAR *)
FILEGRADE, (* STRING VALUE OF YEAR *)
S, (* NAME OF SCHEDULE FILE *)
FNAME, (* NAME OF STUDENT FILE *)
FILENAME : STRING; (* NAME OF STUDENT FILE *)
LOW, (* LOWEST GRADE IN SCHOOL *)
HIGH; (* HIGHEST GRADE IN SCHOOL *)
HEAP1, (* TOP OF HEAP MARKER *)
HEAP, (* TOP OF HEAP STUD SCH *)
ENDROW : 1..MAXROWS; (* LAST ROW USED IN TABLE *)
T : TABLE; (* SCHEDULING PRIORITY TABLE *)
TAL : TALLY; (* LIST OF COURSES TO BE TALLIED *)
SNODE, (* STUDENT SCHEDULE RECORD *)
SHEAD, (* HEAD OF STUD SCH LIST *)

PROCEDURE MAKESCHEDULE (VAR TAB : TABLE; VAR S1,S2 :SCH; VAR TAL : TALLY;
NAME : STRING); (* GIVEN A PRIORITIZED LOOK-UP TABLE THIS PROCEDURE SCHEDULES A STUDENTS COURSES *)

TYPE
WEEK = (M,T,W,R,F); (* DAYS OF WEEK *)

VAR
NUMBERCONF, (* COUNTS CONFLICTS *)
PROBCRS, (* COURSE CAUSING CONFLICT *)
BESTBALNUM, (* CRS WITH BEST BAL FOR RESCHEDULE *)
BESTBALSEC, (* SEC WITH BEST BAL FOR RESCHEDULE *)
REMOVE : INTEGER; (* COURSE TO BE REMOVED IF CONFLICT *)
FINALCHANCE, (* TFY COURSES ON PROBLEM LIST *)
FOUND:
OK:
MARKER:
PROB:
CONFLICT: BOOLEAN;
J : 1..TOTC;
PROBLST : COURREQLIST;

(* COURSE LOCATED *)
(* SCHEDULED OK *)
(* PREVIOUS CONFLICT *)
(* KEEPS TRACK OF BESTBAL *)
(* CONFLICT EXISTS *)
(* ROWS - COURSE TALLIES LIST *)
(* COURSES WITH SCHEDULING PROB *)

PROCEDURE REMOVETL(VAR TAL: TALLY; CRS : INTEGER);
(* REMOVES A COURSE FROM LIST OF COURSES TO BE TALLIED *)

VAR I : 1..TOTC;
KEEP : SC;
FLAG : BOOLEAN;

BEGIN
KEEP[1] := 0;
KEEP[2] := 0;
FLAG := TRUE;
FOR J := 1 TO TOTC DO
  IF FLAG
    THEN IF TAL[J,1] = CRS
        THEN BEGIN
            TAL[J] := KEEP;
            FLAG := FALSE
        END;
    FLAG := TRUE;
FOR J := 1 TO TOTC - 1 DO
  BEGIN
    IF FLAG
      THEN IF TAL[J,1] = 0
        THEN FOR I := J TO TOTC - 1 DO
            BEGIN
                TAL[I] := TAL[I + 1];
                FLAG := FALSE
            END
  END
END:

PROCEDURE TALLYLIST(VAR TAL : TALLY; CRS, SEC : INTEGER);
(* KEEPS LIST OF SCHEDULED CRS FOR COURSE TALLIES LIST *)

VAR KEEP : SC;
FLAG : BOOLEAN;

BEGIN
KEEP[1] := CRS;
FLAG := TRUE;
FOR J := 1 TO TOTC DO
  IF TAL[J,1] = CRS
    THEN BEGIN
        TAL[J] := KEEP;
        FLAG := FALSE
    END
FOR J := 1 TO TOTC DO
  IF FLAG
    THEN IF TAL[J,1] = 0
        THEN BEGIN
            TAL[J] := KEEP;
        END

ELSE IF CRNSCH1 > 0 THEN
BEGIN
    PART := PART + 1;
    TALLYC(TAL);
    SHCOUNT(SH1,SH2,S1,S2);
    IF (RUN = 'F') OR (RUN = 'T') THEN
        BEGIN
            REWRITE(OUTDEVICE,'PRINTER:');
            WRITELN(OUTDEVICE);
            WRITELN(OUTDEVICE, NAME);
            WRITE(OUTDEVICE,'PARTIAL SCHEDULE - COURSE ',CRNSCH1);
            WRITELN(OUTDEVICE,' IS NOT SCHEDULED');
            CLOSE(OUTDEVICE);
            STD3CHCS1,32)
        END;
END;
ELSE IF (CRNSCH1 = 0) AND (CRNSCH2 = 0) THEN
BEGIN
    FULL := FULL + 1;
    TALLYC(TAL);
    SHCOUNT(SH1,SH2,S1,S2);
END END;

PROCEDURE QUERYCL(CRS,SEC : INTEGER; VAR NODE,CH,CT : PTR; VAR FOUND,NEXT : BOOLEAN); (* F inds course in circular list
    - if found then found is true and course node retrieved
    - if next CRS is same then Next set True *)
VAR
    CHMARK : PTR;
    (* Marks head of list *)
BEGIN
    FOUND := FALSE;
    CHMARK := CH;
    REPEAT
        IF (CRS = CH'.CRNUM) AND (SEC = CH'.CRSEC) THEN
            BEGIN
                NODE := CH;
                FOUND := TRUE
            END;
            CH := CH'.LIMK;
            UNTIL FOUND OR ( CH = CHMARK);
    IF FOUND THEN IF (CRS = CH'.CRNUM) AND (SEC = CH'.CRSEC) THEN NEXT := TRUE ELSE NEXT := FALSE;
END;

PROCEDURE SCHEDULECRS(CRS,SEC:INTEGER; VAR OK,CONFLICT,MARKER : BOOLEAN;
    VAR PROBCRS : INTEGER; VAR SB1,SB2 : SCH); (* Places a CRS and SEC into schedule array- sets OK,CONFLICT and
    marker flags to indicate result of operation *)
VAR
    NEXT : BOOLEAN;
    (* if another P:1 of this section exists-
- FLAG := FALSE
END;

PROCEDURE TALLYC(VAR TAL : TALLY):
(* TAKES COURSES IN LIST AND SENDS THEM TO FINDC TO BE COUNTED *)
BEGIN
  FOR J := 1 TO TOTC DO
    IF TAL[J,1] <> 0 THEN FINDC(HEAD,TAL[J,1],TAL[J,2]);
END;

PROCEDURE INITIAL(VAR TAL: TALLY):
(* INITIALIZE LIST TO ALL 0 *)
VAR KEEP : SC;
BEGIN
  KEEP[1] := 0;
  KEEP[2] := 0;
  FOR J := 1 TO TOTC DO TAL[J] := KEEP;
END;

PROCEDURE PRINTTAL(TAL : TALLY):
(* PRINTS LIST OF COURSES THAT A STUDENT SCHEDULED *)
VAR I : 1..TOTC;
BEGIN
  REWRITE(OUTDEVICE,'PRINTER:');
  WRITELN(OUTDEVICE);
  FOR I := 1 TO TOTC DO
    IF TAL[I,1] = 0 THEN WRITELN(OUTDEVICE,TAL[I,1],',',TAL[I,2]);
  CLOSE(OUTDEVICE);
END;

PROCEDURE PRINTSTSCH(VAR S1,32 : SCH):
(* PRINTS CONFLICT MESSAGES WITH STUDENT SCHEDULES *)
BEGIN
  IF (CRNSCH1 > 0) AND (CRNSCH2 > 0) THEN
    BEGIN
      IRR := IRR + 1:
      IF (RUN = 'F') OR (RUN = 'T') THEN
        BEGIN
          REWRITE(OUTDEVICE,'PRINTER:');
          WRITELN(OUTDEVICE);
          WRITELN(OUTDEVICE,'COURSES SCHEDULED');
          WRITELN(OUTDEVICE);
          FOR I := 1 TO TOTC DO
            IF TAL[I,1] = 0 THEN WRITELN(OUTDEVICE,TAL[I,1],',',TAL[I,2]);
          CLOSE(OUTDEVICE);
        END;
      END;
    END;
PROCEDURE FILLMAT (VAR SB: SCH):
    (* PLACES COURSE IN SCHEDULE MATRIX BY SEM, PD AND DAY *)
BEGIN
D := 1;
FOR J := M TO F DO
    BEGIN
        IF NOT CONFLICT THEN
            BEGIN
                IF NODE^:.DAY[D] = DAYS[D]
                    THEN IF SB(NODE^:.PD,D) = 0
                        THEN
                            BEGIN
                                SB(NODE^:.PD,D) := CRS;
                                OK := TRUE;
                                CONFLICT := FALSE;
                            END
                        ELSE
                            BEGIN
                                OK := FALSE;
                                CONFLICT := TRUE;
                                PROBCRS := SB(NODE^:.PD,D);
                            END;
                END
                D := D + 1;
            END
    END
END;

BEGIN
DAYS := 'MTWRF';
CONFLICT := FALSE;
NEXT := TRUE;
HEAD := CHEAD;
WHILE NEXT AND NOT CONFLICT DO
    BEGIN
        NEW(NODE);
        QUERYCL(CRS,SEC,NODE,HEAD,CTAIL,FOUND,NEXT);
        IF FOUND
            THEN
                BEGIN
                    IF (NODE^.SEM = 1) OR (NODE^.SEM > 3)
                        THEN WRITELN('ERROR IN SEMESTER ')
                    ELSE
                        BEGIN
                            CASE NODE^.SEM OF
                                1: FILLMAT(SB1);
                                2: FILLMAT(SB2);
                                3: BEGIN
                                    FILLMAT(SB1);
                                    FILLMAT(SB2)
                                END
                            END
                        ELSE
                            BEGIN
                                REWRITE(OUTDEVICE,'PRINTER:');
                                WRITELN(OUTDEVICE,'COURSE ' ,CRS, ' SECTION ' ,SEC, ' IS NOT IN LIST'
                                CLOSE(OUTDEVICE)
PROCEDURE ADDPROBLIST(VAR LIST : COURREQLIST; PROBCRS : INTEGER):
(* Adds a course to problem list so other sections can be
scheduled if it becomes necessary *)

VAR INDEX : 1..MAXN; (* List index *)
FLAG : BOOLEAN; (* Course is in list *)

BEGIN
  IF LIST = NULLA THEN
    LIST[1] := PROBCRS
  ELSE BEGIN
    FLAG := TRUE;
    FOR INDEX := 1 TO MAXN DO IF LIST[INDEX] = PROBCRS
      THEN FLAG := FALSE;
    IF FLAG THEN BEGIN
      INDEX := 2;
      WHILE FLAG DO BEGIN
        IF LIST[INDEX] = 0 THEN BEGIN
          LIST[INDEX] := PROBCRS;
          FLAG := FALSE
        END;
        IF INDEX < 10 THEN INDEX := INDEX + 1
        ELSE FLAG := FALSE
      END
    END
  END
END;

PROCEDURE TRYALT(VAR ATAB : TABLE);
(* tries to schedule alternate courses *)

PROCEDURE ADDALT(VAR NODE : POINT1; FALTCRS,RALTCRS : INTEGER):
(* creates new clist using the alternate course *)

VAR CH, (* ptr to head of clist *)
CT, (* ptr to tail of clist *)
LOOKAHEAD, (* CRS list - look ahead *)
CHASER : PTR; (* - chaser *)

BEGIN
  FOR INDEX := 1 TO MAXN DO
    IF NODE^.CRSREQ[index] = FALTCRS
      THEN NODE^.CRSREQ[index] := RALTCRS;
  CH := CHEAD;
  CT := CTAI L;
  LOOKAHEAD := HEAD;
  CHASER := HEAD;
  WHILE (LOOKAHEAD <> NIL) AND (RALTCRS > LOOKAHEAD^.CRNUM) DO
    BEGIN
      CHASER := LOOKAHEAD;
      LOOKAHEAD := LOOKAHEAD^.LINK
    END
WHILE (LOOKAHEAD <> NIL) AND (RALTCRS = LOOKAHEAD^.CRNUM) DO
BEGIN
  IF LOOKAHEAD^.ENROL < LOOKAHEAD^.MAX
  THEN BEGIN
    NEW(CORNODE);
    CORNODE^ := LOOKAHEAD^;
    CORNODE^.LINK := CH;
    CT^.LINK := CORNODE;
    CH := CORNODE
  END;
  CHASER := LOOKAHEAD;
  LOOKAHEAD := LOOKAHEAD^.LINK
END END;
BEGIN
  IF ATAB[1,1] = ALT1
  THEN BEGIN
    NEXTCRS(ATAB);
    REPLACE(ALT2,HEAD,CTAIL,ATAB);
    IF NOT ALTUSED
    THEN BEGIN
      ADDALT(STNODE,ALT1,ALT2);
      ALTUSED := TRUE
    END END;
  ELSE IF ATAB[1,1] = ALT2
  THEN BEGIN
    NEXTCRS(ATAB);
    REPLACE(ALT1,HEAD,CTAIL,ATAB);
    IF NOT ALTUSED
    THEN BEGIN
      ADDALT(STNODE,ALT2,ALT1);
      ALTUSED := TRUE
    END END;
END END;

PROCEDURE TRYPROBLIST(VAR TAB: TABLE; VAR PROBLIST: COURREQLIST; VAR TAL: TALLY; VAR SI, S2: SCH);
(* ADDS ALL SECTIONS OF COURSES ON PROBLEM LIST TO THE TABLE AND TRIES TO SCHEDULE THEM *)

VAR
  STAB, (* TABLE FOR OTHEROPT *)
  BTAB, (* TABLE FOR BACKUP *)
  CTAB, (* COPY OF TAB FOR REPLACEMENT *)
  NTAB : TABLE;
  BS1, (* SCHED FOR BACKUP *)
  BS2, (* BACKUP COPY OF NS1*)
  SB1, (* BACKUP COPY OF NS2*)
  SB2, (* SCHED FOR OTHEROPT *)
  SS1, (* SCHED FOR OTHEROPT *)
  SS2, (* SCHED FOR OTHEROPT *)
  NS1, (* COPY OF S1 *)
  NS2, (* COPY OF S2 *)
  LIST : COURREQLIST;
  STL, (* COPY OF PROBLIST *)
  NTAL, (* TAL FOR OTHEROPT *)
  NPROBCRS, (* COPY OF TALLY *)
  PROBCRS, (* COURSE CAUSING CONFLICT *)
PROCEDURE SAVEENVIROMENT;
(* SAVES THE VALUES IN CASE A SCHEDULE TRY DOESN'T WORK *)
BEGIN
  CTAB := TAB;
  NEXTCRS(CTAB);
  NTAB := CTAB;
  NS1 := S1;
  NS2 := S2;
  LIST := PROBLIST;
  NTAL := TAL;
  RESTORE := TAB[1,1];
  NOCONFLICT := TRUE
END;

PROCEDURE RESTOREENVIRONMENT;
(* SCHEDULE WAS SUCCESSFUL DO RESTORE ENVIROMENT BEFORE RETURNING *)
BEGIN
  TAB[1,1] := Ø;
  S1 := NS1;
  S2 := NS2;
  TAL := NTAL
END;

PROCEDURE CRSSCHEDULED;
(* COURSE WAS SCHEDULED SO UPDATE SCHEDULES *)
BEGIN
  FOR RO := 1 TO NUMPD DO
    FOR CO := 1 TO NUMD DO
      IF NS1[RO,CO] = NTAB[1,1] THEN
        BEGIN
          SB1[RO,CO] := Ø;
          REM := TRUE
        END;
    FOR RO := 1 TO NUMPD DO
      FOR CO := 1 TO NUMD DO
        IF NS2[RO,CO] = NTAB[1,1] THEN
          BEGIN
            SB2[RO,CO] := Ø;
            REM := TRUE
          END;
  IF REM THEN
    REMOVETL(NTAL,NTAB[1,1]);
    TALLYLIST(NTAL,NTAB[1,1],NTAB[1,2]);
  IF (NTAB[1,1] = NTAB[2,1]) AND (NOT BACKUP) THEN
    BEGIN
      BACKUP := TRUE;
      NEXTSEC(NTAB);
    END;
BEGIN

BTAB := NTAB;
BS1 := NS1;
BS2 := NS2;
END;
NS1 := SB1;
NS2 := SB2;
NEXTCRS(NTAB);
IF NOT OTHEROPT
THEN
BEGIN
STAB := NTAB;
SSI := NS1;
SS2 := NS2;
STAL := NTAL;
OTHEROPT := TRUE
END
END;

PROCEDURE CHECKT(CRSNUM : COURSENUMBER; VAR TAB : TABLE):
(* IF CRS IS IN TABLE THEN IT IS REMOVED *)
VAR J,
C :
(* LOOP COUNTER *)
1..MAXROWS;
(* LOOP COUNTER *)
BEGIN
  FOR J := 1 TO MAXROWS DO
    IF TAB[J,1] = CRSNUM THEN TAB[J,1] := 0;
  FOR J := 1 TO MAXROWS DO
    IF TAB[J,2] > J THEN IF TAB[J,1] = 0
      THEN FOR C := J TO MAXROWS - 1 DO
        TAB[C] := TAB[C+1];
END;

BEGIN
  (* TRYPROBLEM *)
  SAVEENVIROMENT;
  REPLACE(RESTORE,CHEAD,CTAIL,NTAB);
  INDEX := 1;
  SINDEX := 0;
  OTHEROPT := FALSE;
  BACKUF := FALSE;
  WHILE LISTINDEX <= 0 DO
  BEGIN
    ALTOFF := TRUE;
    CHECKT(LISTINDEX,NTAB);
    REPLACE(LISTINDEX,CHEAD,CTAIL,NTAB);
    WHILE (NTAB[1,1] <> 0) AND (NOCONFLICT) DO
    BEGIN
      SB1 := NS1;
      SB2 := NS2;
      SCHEDULECRS(NTAB[1,1],NTAB[1,2],OK,CONFLICT,MARKER,PROBCRS,SB1,SB2).
      REM := FALSE;
      IF OK
      THEN CRSSCHEDULED;
      IF CONFLICT
      THEN
      BEGIN
        IF SECT(PROBCRS,CHEAD,CTAIL) > 1
        THEN ADDPROBLIST(LIST,PROBCRS);
        IF NTAB[2,1] = NTAB[1,1]
        THEN
        BEGIN
          SB1 := NS1;
          SB2 := NS2;
          NEXTSEC(NTAB)
        END;
      END;
    END;
  END;
END;
ELSE IF ((NTAB[1,1] = ALT1) OR (NTAB[1,1] = ALT2)) AND ALTOFF THEN
   BEGIN
      TRYALT(NTAB);
      ALTOFF := FALSE
      END
ELSE BEGIN
   IF BACKUP THEN
      BEGIN
         BACKUP := FALSE;
         NTAB := BTAB;
         NS1 := BS1;
         NS2 := BS2
         END
   END
END IF NOCONFLICT THEN
   BEGIN
      RESTOREENVIRONMENT;
      LIST[INDEX + 1] := []
   END
ELSE
   NOCONFLICT := TRUE;
   IF OTHEROPT THEN
      BEGIN
         NS1 := SS1;
         NS2 := SS2;
         NTAL := STAL;
         NTAB := STAB;
         OTHEROPT := FALSE;
      END
   ELSE
      BEGIN
         NTAB := CTAB;
         RESTORE := TAB[1,1];
         REPLACE := TAB[1,1];
         RESTORE := TAB[1,1];
         REPLACE := TAB[1,1];
      END;
   END
   IF INDEX < 10 THEN
      INDEX := INDEX + 1
   ELSE LIST[INDEX] := []
   END
END (* TRYPROBLIST *)

(**********************************************************************

BEGIN

OK := TRUE;
MARKER := FALSE;
ALTUSED := FALSE;
initch(s1);

(**********************************************************************

BEGIN (* MAKESCHEDULE *)


SB1 := S1;
INITCH(S2);
SB2 := S2;
NUMBERCONF := 0;
INITTAL(TAL);
PROBLIST := NULLA;
FINALCHANCE := FALSE;
WHILE TAB[1,1] :: 0 DO (* WHILE MORE COURSES TO SCHEDULE *)
BEGIN
  IF OK THEN
  BEGIN
    PROB := FALSE;
    BESTBALNUM := TAB[1,1];
    BESTFALSEC := TAB[1,2]
  END;
  SCHEDULECRS(TAB[1,1],TAB[1,2],OK,CONFLICT,MARKER,PROBCRS,SB1,SE2);
  IF OK THEN
  BEGIN
    S1 := SB1;
    S2 := SB2;
    TALLYLIST(TAL,TAB[1,1],TAB[1,2]);
    NEXTCRS(TAB):
    MARKER := FALSE;
    PROBLIST := NULLA;
    FINALCHANCE := FALSE;
  END;
  IF CONFLICT THEN
  BEGIN
    SB1 := S1;
    SB2 := S2;
    IF SECT(PROBCRS,CHEAD,CTAIL) > 1 THEN
    BEGIN
      ADDPROBLIST(PROBLIST,PROBCRS):;
      FINALCHANCE := TRUE
    END;
    IF TAB[2,1] = TAB[1,1] THEN (* TRY OTHER SECTIONS *)
    BEGIN
      IF NOT PROB THEN
      BEGIN
        REMOVE := PROBCRS;
        PROB := TRUE;
      END;
      NEXTSEC(TAB)
    END
    ELSE IF MARKER THEN (* NO OTHER SECTION *)
    BEGIN
      MARKER := FALSE;
      NUMBERCONF := NUMBERCONF + 1;
      IF NUMBERCONF = 1 THEN CRNSCH1 := TAB[1,1]:
      IF NUMBERCONF = 2 THEN
      BEGIN
        CRNSCH2 := TAB[1,1]:
        TAB[1,1] := 0:
      END
      ELSE NEXTCRS(TAB):
    END
    ELSE
    BEGIN
    END
  END;
IF FINALCHANCE
THEN TRYPROBLIST(TAB,PROBLIST,TAL,S1,S2);
IF TAB[1,1] <> Ø
THEN
BEGIN
  IF (TAB[1,1] = ALT1) OR (TAB[1,1] = ALT2) AND ALTOF
  THEN
    BEGIN
      TRYALT(TAB);
      ALTOF := FALSE
    END
  ELSE
    BEGIN
      IF NOT Prob
      THEN
        BEGIN
          REMOVE := PROBCRS;
          Prob := TRUE
        END;
      REMOVE(TAL,REMOVE);
      FOR RO := 1 TO NUMPD DO
      FOR CO := 1 TO NUMD DO
        IF SB1[RO,CO] = REMOVE
        THEN SB1[RO,CO] := Ø;
      FOR RO := 1 TO NUMPD DO
      FOR CO := 1 TO NUMD DO
        IF SB2[RO,CO] = REMOVE
        THEN SB2[RO,CO] := Ø;
      NEXTCRS(TAB);
    SCHEDULECRS(BESTBALNUM,BESTBALSEC,OK,CONFLICT,MARKER,PROB);
    END
  END
  ELSE
    BEGIN
      MARKER := TRUE;
      S1 := SB1;
      S2 := SB2;
      TALLYLIST(TAL,BESTBALNUM,BESTBALSEC);
      IF SECT(REMOVE,CHEAD,CTAIL) = 2
      THEN
        BEGIN
          NUMBERCONF := NUMBERCONF + 1;
          IF NUMBERCONF = 1
          THEN CRNSCH1 := BESTBALNUM
          ELSE
            BEGIN
              CRNSCH2 := BESTBALNUM;
              TAB[1,1] := Ø
            END
        END
      ELSE
        BEGIN
          MARKER := TRUE;
          S1 := SB1;
          S2 := SB2;
          TALLYLIST(TAL,BESTBALNUM,BESTBALSEC);
          IF SECT(REMOVE,CHEAD,CTAIL) = 2
          THEN
            BEGIN
              NUMBERCONF := NUMBERCONF + 1;
              IF NUMBERCONF = 1
              THEN CRNSCH1 := REMOVE
              ELSE IF NUMBERCONF = 2
              THEN
                BEGIN
                  CRNSCH2 := REMOVE;
                  TAB[1,1] := Ø
                END
            END
        END
      ELSE
        REPLACE(REMOVE,CHEAD,CTAIL,TAB);
(* MAKESCHEDULE *)

PROCEDURE SCHEDASTUD;
(* SCHEDULES ONE STUDENT *)

BEGIN
CRNSCH1 := Ø;
CRNSCH2 := Ø;
ALTOF := TRUE;
NEW(SCHNODE);
MARK(HEAP);
NEW(STNODE);
STNODE^ := STREC;
MAKECLIST(STNODE);
IF STNODE^.CRSREQ <> NULLA
THEN
BEGIN
   MAKETABLE(T,COUNT,CHEAD,CTAIL);
   MAKESCHEDULE(T,S1,S2,TAL,STNODE^.STUDNAM);
   FILEST(CHEAD,SCNODE,STNODE,TAL)
END;
TEST := TEST + 1;
RELEASE(HEAP);
IF (TEST = 100) AND NOTOPEN AND (RUN = 'F')
THEN
BEGIN
   STR(PRESENT,S);
   FNAME := CONCAT('SCH:SCH',S);
   OPENFILE(FNAME);
   IF SHEAD <> NIL
      THEN-SAVEST(SHEAD);
   NOTOPEN := FALSE;
   RELEASE(HEAP);
   SHEAD := NIL;
   TEST := 1;
END;
GET(STREC);
END;

BEGIN
(* STUDSCHEDULE *)

FOR INDEX := 1 TO MAXN DO
   NULLA(INDEX) := Ø;
GOTOXY(10,10);
MESS('TRIAL OR FINAL RUN? (ENTER T OR F):'');
READ(RUN);
IF RUN = 'T'
THEN
   BEGIN
      MESS('WHAT RUN IS THIS? ');
      READLN(RUNTYPE)
   END
ELSE RUNTYPE := 'FINAL';
PAGE(OUTPUT):
LOW := ASKCHOICE(0,5,'WHAT IS THE LOWEST GRADE YOU WISH TO SCHEDULE',7,1)
HIGH := ASKCHOICE(0,10,'WHAT IS THE HIGHEST GRADE YOU WISH TO SCHEDULE',7,1)
GETOUT := FALSE
PRESENT := LOW
REPEAT
TEST := 1;
FULL := 0;
PART := 0;
NOTOPEN := TRUE;
MARK := (HEAP1)
SHEAD := NIL;
INITSCH(SH1);
INITSCH(SH2);
STR := CONCAT('ST:STREC',FILEGRADE);
FILENAME := CONCAT('ST:STREC',FILEGRADE);
RESET(STREC,FILENAME);
WHILE NOT EOF (STREC) DO BEGIN 
SCHEDASTUD
END;
CLOSE(STREC);
REWRITE(OUTDEVICE,'PRINTER:');
WRITELN(OUTDEVICE);
WRITELN(OUTDEVICE, RUN = RUNTYPE);
WRITELN(OUTDEVICE, RESULTS FOR GRADE ',PRESENT);
WRITELN(OUTDEVICE, FULL SCHEDULES = FULL);
WRITELN(OUTDEVICE, PARTIAL SCHEDULES = ',PART);
WRITELN(OUTDEVICE, IRR CONFLICTS = ',IRR);
WRITELN(OUTDEVICE, TOTAL = ',FULL + PART + IRR);
WRITELN(OUTDEVICE); 
WRITELN(OUTDEVICE, 'STUDY HALL COUNT ');
CLOSE(OUTDEVICE);
STDSCH(SH1,SH2);
IF NOTOPEN AND (RUN = 'F') THEN BEGIN 
STF := CONCAT('SCH:SCH',S);
OPENFILE(STF)
IF SHEAD <> NIL THEN SAVESCH(SHEAD);
END ELSE IF (SHEAD <> NIL) AND (RUN = 'F') THEN SAVESCH(SHEAD);
IF RUN = 'F' THEN CLOSE(SCHRECD,LOCK);
IF PRESENT = HIGH THEN GETOUT := TRUE 
ELSE PRESENT := PRESENT + 1;
RELEASE(HEAP1)
UNTIL GETOUT = TRUE 
END: (* STUDSCHEDULE *)

BEGIN (* SCHEDULER *)
STUDSCHEDULE;
PRINTLIST(HEAD);
END. (* SCHEDULER *)
10.7. UNIT TWO
(***S++*)

(* ALLOW SWAPPING *)

UNIT TWO; INTRINSIC CODE 22 DATA 24;

**********************************************************************
(* THIS UNIT CONTAINS SUBROUTINES USED BY 'SCHED'. WHEN CALLED *)
(* THIS UNIT LOADS ALL COURSES INTO A LIST AND SETS CLASS *)
(* ENROLLMENT TO 0. *)

**********************************************************************

INTERFACE

CONST
MAXN   = 10;
MAXCOLS = 4;
MAXROWS = 75;
NUMPD  = 13;
NUMD   = 5;
TOTC   = 11;

TYPE
POINTER = 'COURSE;
PTR    = 'CRENROL;
COURSENUMBER = 0..999;
COURSESECTION = 1..99;
SEMESTER = 1..3;
PERIODNUMBER = 1..16;
TEACHERNUMBER = 1..99;
COURSE =
RECORD
CRNUM  : COURSENUMBER;
CRNAM  : STRING[10];
CRSEC  : COURSESECTION;
MAX    : INTEGER;
PERIOD : PERIODNUMBER;
PD     : PERIODNUMBER;
DAY    : STRING[5];
SEM    : SEMESTER;
ROOM   : STRING[3];
TEACHER : TEACHERNUMBER;
LINK1  : POINTER;
END;

CRENROL =
RECORD
CRNUM  : COURSENUMBER;
CRNAM  : STRING[10];
CRSEC  : COURSESECTION;
MAX    : INTEGER;
PERIOD : PERIODNUMBER;
PD     : PERIODNUMBER;
DAY    : STRING[5];
SEM : SEMESTER; (* SEMESTER (1..2..3) *)
ROOM : STRING[2]; (* ROOM NUMBER *)
TEACHER : TEACHERNUMBER; (* TEACHER NUMBER *)
ENROL : INTEGER; (* COURSE ENROLMENT *)
LINK : PTR; (* POINTER TO NEXT RECORD *)
END;
POINT1 = STUDENT; (* TO STUDENT RECORD *)
YEAR = 7..12;
COURREQLIST = ARRAY[1..MAXN] OF COURSENUMBER;
STUDENT RECORD =
STUDNAM : STRING[20]; (* STUDENT NAME *)
STUDID : INTEGER; (* STUDENT ID NUMBER *)
GRADE : YEAR; (* GRADE IN SCHOOL *)
CRSREQ : COURREQLIST; (* ARRAY OF STUDENT COURSES *)
LINK1 : PTR; (* POINTER TO NEXT RECORD *)
END;
ONER = ARRAY[1..NUMD] OF ONER;
SCH = ARRAY[1..MAXD] OF SCHREC;
ONECOLUMN = ARRAY[1..MAXCOLS] OF INTEGER;
ONEROW = ARRAY[1..MAXROWS] OF ONER;
SC = ARRAY[1..2] OF INTEGER; (* CRS NUMBER AND SEC *)
TALLY = ARRAY[1..TOTC] OF SC; (* LIST OF CRS SCHED *)
SCHPTR = ~SCHREC; (* PTR TO SCH RECORDS *)
SCHREC =
RECORD
STUDNAM : STRING[20]; (* STUDENT NAME *)
STUDID : INTEGER; (* STUDENT ID *)
GRADE : YEAR; (* STUDENT GRADE *)
CRSSCH : TALLY; (* STUD SCHEDULE *)
L : SCHPTR; (* POINTER *)
END;
VAR
HEAD : PTR; (* POINTER TO START OF LIST *)
CAREC : FILE OF COURSE; (* FILE BUFFER *)
SCHREC : FILE OF SCHREC; (* FILE OF STUDENT SCHEDULES *)
OUTDEVICE : TEXT; (* PRINTER *)
RO : 1..NUMD; (* ROWS - PD *)
CO : 1..NUMC; (* COLUMNS - DAYS *)
CORNODE : PTR; (* COURSE RECORD *)
INDEX : 0..MAXN; (* ARRAY INDEX *)
NULLA : COURREQLIST; (* ALL COURSES ARE 0 *)
CHEAD, CTAIL : PTR; (* TAIL OF CIRCULAR LIST *)
COUNT : INTEGER; (* NUMBER OF COURSES IN C.L. *)
ROW : 1..MAXROWS; (* INDEX FOR TABLE *)
HOLD : ONEROW; (* TEMP STORAGE *)
COL : 1..MAXCOLS; (* A COLUMN *)

PROCEDURE OPENFILE(FILENAME : STRING);
PROCEDURE SAVESCH(FRONT : SCHPTR);
PROCEDURE INSERTSCH(VAR FRONT, ITEM : SCHPTR);
PROCEDURE FILEST(VAR SHEAD, SCHNODE : SCHPTR; NODE : POINT1; TALLY : TALLY);
PROCEDURE IOPRINT(ERRTYPE : INTEGER);
PROCEDURE NODISK(DISKNAME : STRING; DISKERR : INTEGER);
PROCEDURE INSERTNODE (VAR FRONT, ITEM : PTR);
PROCEDURE FETCHIT(VAR FIRST : PTR);
PROCEDURE PRINTLIST (FRONT : PTR);
PROCEDURE INITSCH(VAR S : SCH);
PROCEDURE PRINTSCH(S:SCH);
PROCEDURE MESS(STR:STRING);
PROCEDURE STDSCH(VAR S1:S2 : SCH);
PROCEDURE FETCHCRCNQDE:POINT1; VAR CH, CT, HD : PTR);
PROCEDURE MAKECLIST(VAR NODE : POINT1);
FUNCTION ASKCHOICE(X, Y : INTEGER; Q : STRING; MIN, MAX : INTEGER): INTEGER;
PROCEDURE PRINTMAT(MAT : TABLE; ST : STRING);
PROCEDURE MAKEHTAB(VAR M : TABLE; ROWCOUNT : INTEGER; VAR CH, CT : PTR);
FUNCTION SEC(T REMOVE : INTEGER; CH, CT : PTR): INTEGER;
PROCEDURE REPLACET(REMOVE : INTEGER; CH, CT : PTR; VAR TAB : TABLE);
PROCEDURE NEXTC(TAB : TABLE);
IF LOOKAHEAD <> FRONT
    THEN
        BEGIN
            ITEM^.L := LOOKAHEAD;
            CHASER^.L := ITEM;
        END
        ELSE
            BEGIN
                ITEM^.L := FRONT;
                FRONT := ITEM;
            END
        ELSE
            CHASER^.L := ITEM;

END;

PROCEDURE FILEST;
(* SENDS STUDENT SCHEDULE RECORD TO FILE *)
BEGIN
    SCHNODE^.STUDNAM := NODE^.STUDNAM;
    SCHNODE^.STUDID := NODE^.STUDID;
    SCHNODE^.GRADE := NODE^.GRADE;
    SCHNODE^.CRSSCH := TAL;
    SCHNODE^.L := NIL;
    IF SHEAD = NIL
        THEN SHEAD := SCHNODE
        ELSE INSERTSCH(SHEAD,SCHNODE)
END:

PROCEDURE IOPRINT;
(* PRINT ERROR MESSAGE *)
BEGIN
    GOTOXY(0,2);
    WRITE(CHR(7) ) ;
    CASE ERRTYPE OF
        1 : WRITELN('DISK READ ERROR.');
        2 : WRITELN('BAD UNIT NUMBER.');
        3 : WRITELN('ILLEGAL OPERATION. ');
        4 : WRITELN('UNDEFINED HARDWARE ERROR. ');
        5 : WRITELN('UNIT NO LONGER ON LINE.');
        6 : WRITELN('FILE NOT IN DIRECTORY. ');
        7 : WRITELN('ILLEGAL FILE NAME. ');
        8 : WRITELN('INSUFFICIENT DISK SPACE. ');
        9 : WRITELN('NO SUCH FILE ON VOLUME. ');
       10 : WRITELN('NO SUCH FILE ON VOLUME. ');
       11 : WRITELN('DUPLICATE FILE. ');
       12 : WRITELN('ATTEMPT TO OPEN AN OPEN FILE. ');
       13 : WRITELN('FILE NOT OPEN. ');
       14 : WRITELN('ERROR IN READING INTEGER OR REAL DATA. ');
       15 : WRITELN('RING BUFFER OVERFLOW. ');
    END:
END:

PROCEDURE NODISK;
(* CHECKS FOR ERROR 9 *)
VAR CH : CHAR;
BEGIN
    IF (DISKERR <> 0) AND (DISKERR <> 9)
        THEN IOPRINT(DISKERR):
IF (DISKERR = 9)
THEN
BEGIN
GOTOXY(0,22):
WRITELN('NO ',DISKNAME,' DISKETTE!');
WRITELN('PLEASE INSERT DISKETTE AND HIT ANY KEY');
READ(KEYBOARD,CH);
END;
END;

PROCEDURE INSERTNODE;
(* TO PUT NEW NODE IN LIST IN ORDER OF COURSE, SECTION AND PD NUMBER *)
VAR
LOOKAHEAD, CHASER, (* LOOKAHEAD POINTER TO FIND PLACE *)
PTR; (* CHASER POINTER FOR INSERTION *)
BEGIN (* INSERTNODE *)
LOOKAHEAD := FRONT;
CHASER := FRONT;
WHILE (LOOKAHEAD <> NIL) AND (ITEM^.CRNUM > LOOKAHEAD^.CRNUM) DO
BEGIN
CHASER := LOOKAHEAD;
LOOKAHEAD := LOOKAHEAD^.LINK
END; (* WHILE *)
WHILE (LOOKAHEAD <> NIL) AND (ITEM^.CRNUM = LOOKAHEAD^.CRNUM) AND
(ITEM^.CRSEC > LOOKAHEAD^.CRSEC) DO
BEGIN
CHASER := LOOKAHEAD;
LOOKAHEAD := LOOKAHEAD^.LINK
END; (* WHILE *)
WHILE (LOOKAHEAD <> NIL) AND (ITEM^.CRNUM = LOOKAHEAD^.CRNUM) AND
(ITEM^.CRSEC = LOOKAHEAD^.CRSEC) AND (ITEM^.PD > LOOKAHEAD^.PD) DO
BEGIN
CHASER := LOOKAHEAD;
LOOKAHEAD := LOOKAHEAD^.LINK
END; (* WHILE *)
IF (LOOKAHEAD <> NIL)
THEN
IF LOOKAHEAD <> FRONT
THEN
BEGIN
ITEM^.LINK := LOOKAHEAD;
CHASER^.LINK := ITEM;
END; (* INSERT IN MIDDLE OF LIST *)
ELSE
BEGIN
ITEM^.LINK := FRONT;
FRONT := ITEM;
END; (* INSERT AT FRONT *)
ELSE
CHASER^.LINK := ITEM;
END;
PROCEDURE FETCHIT;
(* RETRIEVES COURSE LIST FROM FILE AND ENTERS COURSES IN A LINKED LIST *)
VAR CRS : PTR; (* CRS NODE WITH ENROLLMENT *)

BEGIN
RESET (CRREC,'CR: COURSEFILE');

WHILE NOT EOF (CRREC) DO
BEGIN

NEW(CRS):
CRS^.CRNUM := CRREC^.CRNUM;
CRS^.CRNAM := CRREC^.CRNAM;
CRS^.CRSEC := CRREC^.CRSEC;
CRS^.MAX := CRREC^.MAX;
CRS^.PERIOD := CRREC^.PERIOD;
CRS^.PD := CRREC^.PD;
CRS^.DAY := CRREC^.DAY;
CRS^.SEM := CRREC^.SEM;
CRS^.ROOM := CRREC^.ROOM;
CRS^.TEACHER := CRREC^.TEACHER:
CRS^.ENROL := 0;
CRS^.LINK := NIL;
IF FIRST = NIL
THEN FIRST := CRS
ELSE INSERT (FIRST, CRS):
GET(CRREC):
END;
CLOSE(CRREC):
END:

PROCEDURE PRINTLIST;
(* PRINTS COURSE LIST GIVEN THE POINTER TO START OF LIST *)

BEGIN
REWRITE (OUTDEVICE,'PRINTER:');
PAGE(OUTDEVICE);
IF FRONT = NIL
THEN
(* LIST IS EMPTY *)
BEGIN
WRITELN (OUTDEVICE,'LIST IS EMPTY.');
END
ELSE
(* THERE ARE COURSES *)
BEGIN
WRITELN (OUTDEVICE,'COURSES IN NUMERICAL ORDER ARE : '):
WRITELN (OUTDEVICE,'CODE SEC SEATS SEM PD DAYS MET '):
WRITELN (OUTDEVICE,'DESCRIPTION ROOM TCH PER ENROL '):
WHILE FRONT <> NIL DO
BEGIN
WITH FRONT^ DO
BEGIN
WRITE (OUTDEVICE,CRNUM:3,' ,CRSEC:2,' ,MAX:3.):
WRITE (OUTDEVICE,SEM:1,' ,PD:2,' ,DAY:5,' ,CRNAM:10,' ,SEM:
WRITE (OUTDEVICE,ROOM:3,' ,TEACHER:2,' ,PERIOD:2,ENROL:7)
END;
FRONT := FRONT^.LINK:
END;
WRITELN (OUTDEVICE);WRITELN (OUTDEVICE);
END;
CLOSE (OUTDEVICE);
END:
PROCEDURE INITSCH;
    (* INITIALIZES A SCHEDULE TO ALL 0 *)
BEGIN
    FOR RO := 1 TO NUMPD DO
        FOR CO := 1 TO NUMD DO
            S[RO,CO] := 0;
END;

PROCEDURE PRINTSCH;
    (* PRINTS THE STUDENT SCHEDULE *)
BEGIN
    REWRITE(OUTDEVICE,'PRINTER:');
    FOR RO := 1 TO NUMPD DO
        BEGIN
            WRITE(OUTDEVICE,RO);
            FOR CO := 1 TO NUMD DO
                WRITE(OUTDEVICE,S[RO,CO]:6);
            Writeln(OUTDEVICE);
        END;
    Writeln(OUTDEVICE);
    CLOSE(OUTDEVICE);
END;

PROCEDURE MESS;
    (* PRINTS A HEADER FOR SCHEDULE *)
BEGIN
    REWRITE(OUTDEVICE,'PRINTER:');
    Writeln(OUTDEVICE);
    Writeln(OUTDEVICE,STR);
    CLOSE(OUTDEVICE);
END;

PROCEDURE STDSCH;
    (* PRINTS STUDENTS SCHEDULE FOR 1ST AND 2ND SEM *)
BEGIN
    MESS('1ST SEM SCHEDULE');
    PRINTSCH(S1);
    MESS('2ND SEM SCHEDULE');
    PRINTSCH(S2);
END;

PROCEDURE FETCHCR;
    (* GET ALL POSSIBLE SECTIONS FOR A STUDENTS COURSE SELECTION AND PLACE THEM IN A CIRCULUR LIST *)
VAR
    LOOKAHEAD, (* CRS NODE - LOOKAHEAD PTR *)
    CHASER : PTR; (* - CHASER *)
    COURSEFOUND : BOOLEAN; (* CRS IS FOUND *)
    NAME : STRING[20]; (* STUDENT - NAME *)
    LIST : COURSEQLIST; (* - CRS LIST *)
BEGIN
  NAME := NODE^STUDNAM;
  LIST := NODE^CRSREQ;
  FOR INDEX := 1 TO MAXN DO
    BEGIN
      IF LIST[INDEX] <> Ø THEN
        BEGIN
          LOOKAHEAD := HD;
          CHASER := HD;
          COURSEFOUND := FALSE;
          WHILE (LOOKAHEAD <> NIL) AND (LIST[INDEX] = LOOKAHEAD^CRNUM) DO
            BEGIN
              CHASER := LOOKAHEAD;
              LOOKAHEAD := LOOKAHEAD^LINK;
            END;
          END;
          WHILE (LOOKAHEAD <> NIL) AND (LIST[INDEX] = LOOKAHEAD^CRNUM) DO
            BEGIN
              IF LOOKAHEAD^ENROL < LOOKAHEAD^MAX THEN
                BEGIN
                  NEW(CORNODE);
                  CORNODE := LOOKAHEAD;
                  COURSEFOUND := TRUE;
                  COUNT := COUNT + 1;
                  IF (CH = NIL) AND (CT = NIL) THEN
                    BEGIN
                      CORNODE^LINK := CORNODE;
                      CH := CORNODE;
                      CT := CORNODE
                    END
                  ELSE
                    BEGIN
                      CORNODE^LINK := CH;
                      CT^LINK := CORNODE;
                      CH := CORNODE
                    END;
                END;
              CHASER := LOOKAHEAD;
              LOOKAHEAD := LOOKAHEAD^LINK;
            END;
          END;
          IF ((LOOKAHEAD = NIL) OR (LIST[INDEX] < LOOKAHEAD^CRNUM)) AND (COURSEFOUND = FALSE) THEN
            BEGIN
              REWRITE(OUTDEVICE,'PRINTER:');
              WRITE(OUTDEVICE,NAME,' IS SCHEDULED FOR COURSE ',LIST[INDEX]);
              WRITELN(OUTDEVICE,' WHICH IS NOT COURSE LIST.');
            END;
          END;
        END;
      END;
      WHILE (LOOKAHEAD^CRNUM < 999) DO
        BEGIN
          CHASER := LOOKAHEAD;
          LOOKAHEAD := LOOKAHEAD^LINK;
        END;
      END;
    END;
END;
CORNODE  .LINK := CORNODE:
CH := CORNODE:
CT := CORNODE
END
ELSE
BEGIN
CORNODE  .LINK := CH:
CT  .LINK := CORNODE:
CH := CORNODE
END:
CHASER := LOOKAHEAD;
LOOKAHEAD := LOOKAHEAD  .LINK:
END:

PROCEDURE MAKECLIST:
(* CREATES CLIST FOR A STUDENT COURSE SELECTIONS *)
BEGIN
IF NODE  CRSREQ <> NULLA
THEN
BEGIN
COUNT := 0:
CHEAD := NIL:
CTAIL := NIL:
FETCHCR(NODE, CHEAD, CTAIL.HEAD)
END
ELSE
BEGIN
REWRITE(OUTDEVICE, 'PRINTER: ');
WRITELN(OUTDEVICE, NODE  STUDNAM, ' IS SCHEDULED FOR NO COURSES');
CLOSE(OUTDEVICE)
END;
END:

FUNCTION ASKCHOICE:
(* ASK USER FOR A NUMBER FROM MIN TO MAX, AND REPEAT UNTIL OBTAINED *)
VAR ANSWER : INTEGER:
BEGIN
REPEAT
GOTOXY(X,Y):
WRITE(X,'( 'MIN, ' ,MAX, ' ) : '):
READLN(ANSWER);
UNTIL (ANSWER >= MIN) AND (ANSWER <= MAX):
ASKCHOICE := ANSWER
END:

PROCEDURE PRINTMAT:
(* PRINTS LOOKUP TABLE *)
BEGIN
REWRITE(OUTDEVICE, 'PRINTER: ');
WRITELN(OUTDEVICE):
WRITELN(OUTDEVICE):
WRITELN(OUTDEVICE, 'LOOKUP TABLE - ',ST):
WRITE(OUTDEVICE); WRITE(OUTDEVICE,'CRNUM SEC BAL PRIOR');

FOR ROW := 1 TO MAXROWS DO BEGIN
  IF MAT[ROW,1] > 0 THEN BEGIN
    FOR COL := 1 TO MAXCOLS DO
      WRITE(OUTDEVICE,MAT[ROW,COL]+4,' ');
    WRITELN(OUTDEVICE);
  END;
  WRITELN(OUTDEVICE);
END;
CLOSE(OUTDEVICE);
END:

************************************************************************

PROCEDURE MAKETABLE;
(* CREATES LOOKUP TABLE FOR CLIST *)
VAR
  BAL : INTEGER; (* SEATS REMAINING IN A CLASS*)
PROCEDURE FILLTABLE(VAR LASTROW : INTEGER);
(* FILLS LOOKUP TABLE WITH VALUES AND SETS PRIOR TO 0 *)
VAR
  PREVNUM : COURSENUMBER; (* PREVIOUS COURSE *)
  PREVSEC : COURSESECTION; (* PREVIOUS SECTION *)
  CHMARK : PTR; (* MARK HEAD OF LIST *)
PROCEDURE INITTABLE;
BEGIN
  FOR ROW := 1 TO MAXROWS DO
    FOR COL := 1 TO MAXCOLS DO
      MCROW,COL3 := CH'.CRNUM;
  END:
BEGIN
  INITTABLE;
  ROW := 1;
  CHMARK := CH;
  REPEAT
    IF (PREVNUM = CH'.CRNUM) AND (PREVSEC = CH'.CRSEC)
      THEN
        CH := CH'.LINK
      ELSE
        BEGIN
          COL := 1;
          MCROW,COL1 := CH'.CRNUM;
          MCROW,COL+1 := CH'.CRSEC;
          BAL := CH'.MAX - CH'.ENROL;
          MCROW,COL+2 := BAL;
          MCROW,COL+3 := 0;
          PREVNUM := CH'.CRNUM;
          PREVSEC := CH'.CRSEC;
          LASTROW := ROW;
          ROW := ROW + 1;
          CH := CH'.LINK;
        END;
    UNTIL CH = CHMARK;
END:

PROCEDURE FINDPRIOR(LASTROW : INTEGER);

(* FILLTABLE *)
(* FILLTABLE *)
VAR PRIOR : INTEGER;
  TEMP, INDEX : 1..MAXROWS:
BEGIN
  FOR ROW := 1 TO LASTROW DO
    BEGIN
      PRIOR := 1;
      IF MCROW,1 < MCROW+1,1 THEN
        MCROW,4 := 1
      ELSE
        BEGIN
          TEMP := ROW:
          WHILE M[TEMP,1] = M[TEMP+1,1] DO
            BEGIN
              IF M[TEMP,2] <> M[TEMP+1,2] THEN PRIOR := PRIOR + 1;
              TEMP := TEMP + 1;
            END;
          FOR INDEX := ROW TO TEMP DO
            M[INDEX,4] := PRIOR;
          ROW := TEMP
        END;
    END;
END;

PROCEDURE SORTBYPRIOR(LIMIT : INTEGER):
  (* SORT TABLE BY PRIORITY (COLUMN 4) *)
CONST PRIMARY = 4:
VAR INORDER : BOOLEAN:
BEGIN
  REPEAT
    INORDER := TRUE:
    IF LIMIT \\ 1 THEN LIMIT := LIMIT - 1:
    FOR ROW := 1 TO LIMIT DO
      BEGIN
        IF M[ROW,PRIMARY] > M[ROW + 1,PRIMARY] THEN
          BEGIN
            INORDER := FALSE:
            HOLD := M[ROW];
            M[ROW] := M[ROW+1];
            M[ROW + 1] := HOLD
          END;
        END;
    UNTIL INORDER;
  END;
END:

PROCEDURE SORTBYBAL(LIMIT : INTEGER):
  (* SORTS TABLE BY CLASS BALANCE *)
CONST PRIMARY = 3:
VAR INORDER : BOOLEAN:
BEGIN
  REPEAT
    INORDER := TRUE:
    LIMIT := LIMIT - 1:
    FOR ROW := 1 TO LIMIT DO
BEGIN

IF (M[ROW,1] = M[ROW+1,1]) AND (M[ROW,PRIMARY] < M[ROW+1,PRIMARY]) THEN

BEGIN

    INORDER := FALSE;
    HOLD := M[ROW];
    MROW := M[ROW+1];
    M[ROW+1] := HOLD

END;

UNTIL INORDER;
END;

BEGIN

FILLTABLE(ROWCOUNT);
FINDPRIOR(ROWCOUNT);
SORTBYPRIOR(ROWCOUNT);
SORTBYBAL(ROWCOUNT);
END;

FUNCTION SECT:
(* FINDS THE NUMBER OF SECTIONS OF REMOVE *)

VAR PREVSEC,

SEC : INTEGER;  (* NUMBER OF SECTIONS *)
CHMARK : PTR;   (* MARKS HEAD OF LIST *)
BEGIN

CHMARK := CH;
SEC := 0;
REPEAT

    IF REMOVE = CH^.CRNUM
    THEN

        BEGIN

            SEC := SEC + 1;
            PREVSEC := CH^.CRSEC

        END;

        CH := CH^.LINK;

    UNTIL (CH = CHMARK) OR (CH^.CRNUM = REMOVE);

    REPEAT

        IF (REMOVE = CH^.CRNUM) AND (PREVSEC <> CH^.CRSEC)
        THEN

            BEGIN

                SEC := SEC + 1;
                PREVSEC := CH^.CRSEC

            END;

            CH := CH^.LINK;

        UNTIL (CH = CHMARK) OR (CH^.CRNUM <> REMOVE);

    END

SECT := SEC
END;

PROCEDURE REPLACE:
(* PLACES THE OTHER SECTIONS OF REMOVED COURSES INTO THE TABLE *)

VAR COUNT,

PREVSEC : INTEGER;  (* COUNTS NUMBER OF INSERTS *)
CHMARK : PTR;       (* MARKS HEAD OF LIST *)


PROCEDURE SORT(VAR TAB: TABLE; LIMIT: INTEGER); (* SORTS THE SECTIONS INSERTED INTO TABLE BY BALANCE *)

CONST PRIMARY = 3; (* BALANCE IS KEY *)

VAR INORDER : BOOLEAN; (* TRUE - IF TABLE IS IN ORDER *)

BEGIN
  REPEAT
    INORDER := TRUE;
    IF LIMIT > 1
    THEN
      BEGIN
        LIMIT := LIMIT - 1;
        FOR ROW := 1 TO LIMIT DO
          BEGIN
            IF (TAB[ROW,1] = TAB[ROW+1,1]) AND (TAB[ROW,PRIMARY] < TAB[ROW+1,PRIMARY])
            THEN
              BEGIN
                INORDER := FALSE;
                HOLD := TAB[ROW];
                TAB[ROW] := TAB[ROW+1];
                TAB[ROW+1] := HOLD;
              END;
          END;
      END;
    END;
  UNTIL INORDER;
END;

PROCEDURE MOVEDOWN(VAR TAB : TABLE); (* MOVE EVERYTHING IN THE TABLE DOWN A ROW TO INSERT AT HEAD OF TABLE *)

VAR
  SAVE1, SAVE2 : ONEROW; (* SAVE FIRST ROW FOR SWAP *)
  C : INTEGER; (* TABLE INDEX *)

BEGIN
  SAVE1 := TAB[1];
  SAVE2 := TAB[2];
  C := 1;
  WHILE TAB[C,1] < 0 DO
    BEGIN
      C := C + 1;
      TAB[C] := SAVE1;
      SAVE1 := SAVE2;
      SAVE2 := TAB[C+1]
    END;
END;

PROCEDURE ASSIGN; (* ASSIGNS A COURSE TO TABLE *)

BEGIN
  PREVSEC := CH^..CRSEC;
  COUNT := COUNT + 1;
  MOVEDOWN(TAB);
  TAB[1,1] := CH^..CRNUM;
  TAB[1,2] := CH^..CRSEC;
  TAB[1,3] := CH^..MAX - CH^..ENROL;
  TAB[1,4] := Ø
END;

BEGIN (* REPLACE *)
COUNT := 0;
CHMARK := CH;
IF REMOVE = CH^.CRNUM
THEN ASSIGN
ELSE
BEGIN
REPEAT
  CH := CH^.LINK;
  IF REMOVE = CH^.CRNUM
  THEN ASSIGN;
  UNTIL (CH = CHMARK) OR (REMOVE = CH^.CRNUM);
END;
REPEAT
  IF (REMOVE = CH^.CRNUM) AND (CH^.CRSEC <> PREVSEC)
  THEN ASSIGN;
  CH := CH^.LINK;
  UNTIL (REMOVE <> CH^.CRNUM) OR (CH = CHMARK);
SORT(TAB,COUNT);
END;
(* REPLACE *)

PROCEDURE FINDC;
(* FINDS THE COURSE IN LIST AND ADDS ONE TO COURSE ENROLLMENT *)
VAR LOOKAHEAD,
  CHASER   : PTR;
BEGIN
  LOOKAHEAD := FRONT;
  CHASER := FRONT;
  WHILE (LOOKAHEAD <> NIL) AND (CRS > LOOKAHEAD^.CRNUM) DO
    BEGIN
      CHASER := LOOKAHEAD;
      LOOKAHEAD := LOOKAHEAD^.LINK
    END;

  WHILE (LOOKAHEAD <> NIL) AND (CRS = LOOKAHEAD^.CRNUM) AND
       (SEC <> LOOKAHEAD^.CRSEC) DO
    BEGIN
      CHASER := LOOKAHEAD;
      LOOKAHEAD := LOOKAHEAD^.LINK
    END;

  WHILE (LOOKAHEAD <> NIL) AND (CRS = LOOKAHEAD^.CRNUM) AND
       (SEC = LOOKAHEAD^.CRSEC) DO
    BEGIN
      LOOKAHEAD^.ENROL := LOOKAHEAD^.ENROL + 1;
      CHASER := LOOKAHEAD;
      LOOKAHEAD := LOOKAHEAD^.LINK
    END;
END;

PROCEDURE SHCOUNT;
(* ADDS STUDENT TO STUDY HALL COUNT FOR UNSCHEDULED PERIODS *)
BEGIN
  FOR RO := 3 TO NUMPD DO
    FOR CO := 1 TO NUMD DO
      BEGIN
        IF CHASER.CRS.CRS > 0
        THEN
          CHASER.CRS.CRS.CRS := CHASER.CRS.CRS.CRS + 1;
        END;
      END;
  END;
\[
\begin{align*}
\text{IF } S1[RO,CO] = 0 \\
\text{THEN } SH1[RO,CO] := SH1[RO,CO] + 1;
\end{align*}
\]

\[
\begin{align*}
\text{IF } S2[RO,CO] = \emptyset \\
\text{THEN } SH2[RO,CO] := SH2[RO,CO] + 1;
\end{align*}
\]

\[
\text{END;}
\]

(***********************************************************************)

\begin{verbatim}
PROCEDURE NEXTCRS;
(* REMOVES ALL SECTIONS OF A COURSE FROM TABLE AND MOVES UP THE
  NEXT COURSE TO ROW 1 *)
VAR
  ROWC, (* COUNTS ROWS *)
  PREV : INTEGER; (* PREVIOUSLY SCHEDULED COURSE*)
BEGIN
  PREV := TAB[1,1];
  REPEAT
    ROWC := 1;
    WHILE TAB[ROWC,1] > 0 DO
      BEGIN
        TAB[ROWC] := TAB[ROWC + 1];
        ROWC := ROWC + 1
      END;
    UNTIL PREV <> TAB[1,1]
END;
(***********************************************************************)

PROCEDURE NEXTSEC;
(* MOVES NEXT SECTION OF A COURSE TO ROW 1 OF TABLE *)
VAR
  PREVSEC, (* PREVIOUS SECTION *)
  ROWC : INTEGER; (* COUNTS SECTIONS *)
BEGIN
  PREVSEC := TAB[1,2];
  REPEAT
    ROWC := 1;
    WHILE TAB[ROWC,1] > 0 DO
      BEGIN
        TAB[ROWC] := TAB[ROWC + 1];
        ROWC := ROWC + 1
      END;
    UNTIL PREVSEC <> TAB[1,2]
END;
(***********************************************************************)

BEGIN
  HEAD := NIL;
  FETCHIT(HEAD);
END.
(* UNIT TWO *)
\end{verbatim}

(***********************************************************************)
10.8. PRINTSCHEDULES
PROGRAM PRINTSCHEDULES;

(* THIS PROGRAM PRINTS OUT STUDENT SCHEDULES AFTER THE FINAL RUN *)
(* OF THE SCHEDULE HAS BEEN COMPLETED. *)
(* FILES USED *)
(* - SCH : SCH(GRADES) *)
(* OUTPUT *)
(* - STUDENT SCHEDULES BY GRADE *)
(* - A SCHEDULE IS A LIST OF *)
(* - STUDENT NAME *)
(* - STUDENT ID *)
(* - STUDENT GRADE *)
(* - COURSES SCHEDULED (A LIST OF *)
(* - COURSE NAME *)
(* - COURSE NUMBER *)
(* - PERIOD *)
(* - DAYS SCHEDULED *)
(* - ROOM *)
(* - SEMESTER *)
(* - TEACHER NUMBER *)
(*)

CONST
TOTC = 11; (* TOTAL COURSES *)
TOT = 16; (* MAX PERIODS *)

TYPE
SC = ARRAY[1..2] OF INTEGER;
TALLY = ARRAY[1..TOTC] OF SC;
SCHPTR = *SCHREC;
YEAR = 7..12;
SCHREC =
RECORD
STUDNAM : STRING[20]; (* STUDENT NAME *)
STUDID : INTEGER; (* STUDENT ID *)
GRADE : YEAR; (* STUDENT GRADE *)
CROSSCH : TALLY; (* STUD SCHEDULE *)
L : SCHPTR; (* POINTER *)
END:
POINTER = *COURSE;
COURSENUMBER = 0..999;
COURSESECTION = 1..99;
MAXSIZE = 1..999;
PERIODNUMBER = 1..16;
SEMESTER = 1..3;
TEACHERNUMBER = 1..99;
COURSE =
RECORD
CRNUM : COURSENUMBER; (* COURSE NUMBER (0..999) *)
CRNAM : STRING[10]; (* COURSE NAME (10 CHAR) *)
CRSEC : COURSESECTION; (* COURSE SECTION (1..99) *)
MAX : INTEGER; (* MAX SIZE OF CLASS *)
PERIOD : PERIODNUMBER;        (* PERIOD OF DAY (1..16 *)
PD : PERIODNUMBER;            (* SCHEDULING PERIOD *)
DAY : STRING[5];             (* DAY OF WEEK (MTWRF) *)
SEM : SEMESTER;               (* SEMESTER *)
ROOM : STRING[3];            (* ROOM NUMBER *)
TEACHER : TEACHERNUMBER;      (* TEACHER NUMBER *)
LINK : POINTER;              (* POINTER TO NEXT RECORD *)
END;
PRINTREC =
PRINT RECORD
PD : PERIODNUMBER;            (* SCHEDULING PERIOD *)
PERIOD : PERIODNUMBER;        (* PERIOD OF DAY (1..16 *)
DAY : STRING[5];             (* DAY OF WEEK (MTWRF) *)
COURSE : STRING[10];         (* COURSE NAME (10 CHAR) *)
ROOM : STRING[3];            (* ROOM NUMBER *)
SEM : SEMESTER;               (* SEMESTER *)
TEACHER : TEACHERNUMBER;      (* TEACHER NUMBER *)
CODE : COURSENUMBER;          (* COURSE NUMBER (0..999) *)
END;
PRAR = ARRAY[1..TOT] OF PRINTREC;
INDEXLIST = ARRAY[1..TOT] OF INTEGER;

VAR
SCHRECD : FILE OF SCHREC;      (* FILE OF STUDENT SCHEDULES *)
DIFFER, DIFSEM, FOUND;         (* ANOTHER SECTION W DIF PER *)
GETOUT, PRESENT, FILEGRADE, S,
NAME, FILENAME: STRING;        (* NAME OF SCHEDULE FILE *)
LOW, HIGH: YEAR;               (* LOWEST GRADE IN SCHOOL *)
HEAP1, TAL, SHNODE, SHEAD:
HEAP: "INTEGER;              (* TOP OF HEAP MARKER *)
TALLY: * LIST OF COURSES TO BE TALLIED *)
STUDENT SCHEDULE RECORD
SCHNODE, SHEAD: SCHPTR;
CRNUM : COURSENUMBER;
CPNAM : STRING[10];
CRSEC : COURSESECTION;
MAX : MAXSIZE;
SAVEFER, PERIOD : PERIODNUMBER;
SAVESEM, SEM : SEMESTER;
DAY : STRING[5];
TEACHER : TEACHERNUMBER;
ROOM : STRING[3];
NODE, HEAD : POINTER;          (* POINTER TO START OF LIST *)
ANS : CHAR;                    (* YES OR NO (Y OR N) *)
CRREC : FILE OF COURSE;        (* FILE BUFFER *)
OUTDEVICE : TEXT;             (* PRINTER *)
LIST : PRAR;                   (* LIST OF SCHEDULED CRS *)
INDEXLIST : INDEXLIST;
INDEX, C : INTEGER;           (* COUNT OF COURSES *)

SEGMENT PROCEDURE PRINTSTSCH;
(* PRINTS OUT STUDENT SCHEDULES FROM SCH FILE *)
PROCEDURE FINDDSNODE (HD: POINTER; VAR ITEM: POINTER; VAR TEST, DP, DS: BOOLEAN);

(* TO LOCATE A NODE WHICH HAS THE SAME CRS NUM AND SEC BUT A DIF SEM *)

VAR LOOKAHEAD, CHASER: POINTER;

BEGIN
TEST := FALSE;
DP := FALSE;
DS := FALSE;
LOOKAHEAD := HD;
CHASER := HD;
WHILE (LOOKAHEAD <> NIL) AND (ITEM^.CRNUM > LOOKAHEAD^.CRNUM) DO
BEGIN
CHASER := LOOKAHEAD;
LOOKAHEAD := LOOKAHEAD^.LINK;
END;
WHILE (LOOKAHEAD <> NIL) AND (ITEM^.CRNUM = LOOKAHEAD^.CRNUM) AND
(ITEM^.CRSEC <> LOOKAHEAD^.CRSEC) DO
BEGIN
CHASER := LOOKAHEAD;
LOOKAHEAD := LOOKAHEAD^.LINK;
END;
WHILE (LOOKAHEAD <> NIL) AND (ITEM^.CRSEC = LOOKAHEAD^.CRSEC) AND
(ITEM^.SEM <> LOOKAHEAD^.SEM) DO
BEGIN
CHASER := LOOKAHEAD;
LOOKAHEAD := LOOKAHEAD^.LINK;
END;

IF (LOOKAHEAD <> NIL) AND (ITEM^.CRNUM = LOOKAHEAD^.CRNUM) AND
(ITEM^.CRSEC = LOOKAHEAD^.CRSEC) AND (ITEM^.SEM <> LOOKAHEAD^.SEM)
THEN
BEGIN
TEST := TRUE;
ITEM^ := LOOKAHEAD^;
END;
END;

PROCEDURE FINDDDPNODE (HD: POINTER; VAR ITEM: POINTER; VAR TEST, DP, DS: BOOLEAN);

(* TO LOCATE A NODE WHICH HAS THE SAME CRS NUM AND SEC BUT A DIF PER *)

VAR LOOKAHEAD, CHASER: POINTER;

BEGIN
TEST := FALSE;
DP := FALSE;
DS := FALSE;
LOOKAHEAD := HD;
CHASER := HD;
WHILE (LOOKAHEAD <> NIL) AND (ITEM^.CRNUM > LOOKAHEAD^.CRNUM) DO
BEGIN
CHASER := LOOKAHEAD;
LOOKAHEAD := LOOKAHEAD^.LINK;
END;
WHILE (LOOKAHEAD <> NIL) AND (ITEM^.CRNUM = LOOKAHEAD^.CRNUM) AND
(ITEM^.CRSEC <> LOOKAHEAD^.CRSEC) DO
BEGIN
CHASER := LOOKAHEAD;
LOOKAHEAD := LOOKAHEAD^.LINK;
END;
WHILE (LOOKAHEAD <> NIL) AND (ITEM^.CRNUM = LOOKAHEAD^.CRNUM) AND (ITEM^.CRSEC = LOOKAHEAD^.CRSEC) AND (ITEM^.PERIOD >= LOOKAHEAD^.PERIOD) DO
BEGIN
  CHASER := LOOKAHEAD;
  LOOKAHEAD := LOOKAHEAD^.LINK;
END;

IF (LOOKAHEAD <> NIL) AND (ITEM^.CRNUM = LOOKAHEAD^.CRNUM) AND (ITEM^.CRSEC = LOOKAHEAD^.CRSEC) AND (ITEM^.PERIOD < LOOKAHEAD^.PERIOD) THEN
BEGIN
  TEST := TRUE;
  ITEM^- := LOOKAHEAD^-;
  CHASER := LOOKAHEAD;
  LOOKAHEAD := LOOKAHEAD^.LINK;
  WHILE (LOOKAHEAD <> NIL) AND (ITEM^.CRNUM = LOOKAHEAD^.CRNUM) AND (ITEM^.CRSEC = LOOKAHEAD^.CRSEC) AND NOT DP DO
BEGIN
    IF ITEM^.PERIOD < LOOKAHEAD .PERIOD THEN
BEGIN
      WRITELN('FOUND IT');
      DP := TRUE;
    END;
    CHASER := LOOKAHEAD;
    LOOKAHEAD := LOOKAHEAD^.LINK;
END;
  END;
END;
PROCEDURE FINDNODE (HD: POINTER; VAR ITEM : POINTER; VAR TEST,DP,DS :BOOLEAN); (* TO LOCATE A NODE USING THE COURSE NUMBER *)

VAR LOOKAHEAD, CHASER : POINTER; (* LOOKAHEAD POINTER TO FIND PLACE *) (* CHASER POINTER FOR INSERTION *)

BEGIN
  TEST := FALSE;
  DP := FALSE;
  DS := FALSE;
  LOOKAHEAD := HD;
  CHASER := HD;
  WHILE (LOOKAHEAD <> NIL) AND (ITEM^.CRNUM = LOOKAHEAD^.CRNUM) DO
BEGIN
  CHASER := LOOKAHEAD;
  LOOKAHEAD := LOOKAHEAD^.LINK;
END;
  WHILE (LOOKAHEAD <> NIL) AND (ITEM^.CRNUM = LOOKAHEAD^.CRNUM) AND (ITEM^.CRSEC <> LOOKAHEAD^.CRSEC) DO
BEGIN
  CHASER := LOOKAHEAD;
  LOOKAHEAD := LOOKAHEAD^.LINK;
END;

  IF (LOOKAHEAD <> NIL) AND (ITEM^.CRNUM = LOOKAHEAD^.CRNUM) AND (ITEM^.CRSEC = LOOKAHEAD^.CRSEC) THEN
BEGIN
    TEST := TRUE;
    ITEM^- := LOOKAHEAD^-;
  END;
END;
WHILE (LOOKAHEAD <> NIL) AND (ITEM^.CRNUM = LOOKAHEAD^.CRNUM) AND (ITEM^.CRSEC = LOOKAHEAD^.CRSEC) AND (NOT DP AND NOT DS) DO
BEGIN
  CHASER := LOOKAHEAD;
  LOOKAHEAD := LOOKAHEAD^.LINK;
  IF ITEM^.PERIOD <> LOOKAHEAD^.PERIOD
    THEN DP := TRUE;
  IF ITEM^.SEM <> LOOKAHEAD^.SEM
    THEN DS := TRUE;
END;
END;

END;
PROCEDURE OPENFILE(FILENAME : STRING):
(* OPENS A FILE AND CHECKS FOR ERRORS *)

VAR IOERR : INTEGER; (* ERROR NUMBER *)
BEGIN
REPEAT (**I--**)
    RESET(SCHRECD,FILENAME);
    (**I++**)
    IOERR := IORESULT;
    NODISK(FILENAME,IOERR);
UNTIL IOERR = 0;
END;

PROCEDURE WRSCH;
(* PRINTS A STUDENTS SCHEDULE *)

VAR I, J : INTEGER; (* LOOP COUNTER *)

PROCEDURE INITINDEX(VAR INDEX : INDEXLIST);
(* Initiates INDEX LIST TO 99 *)
BEGIN
FOR J := 1 TO TOT DO
    INDEX[J] := 99;
END;

PROCEDURE ADDTOINDEX(VAR INDEX : INDEXLIST; NEWPER : PERIODNUMBER);
(* Adds PD of SCHEDULED CRS TO INDEX LIST *)

VAR FLAG : BOOLEAN; (* Course entered into list *)
SAV1, SAV2, J1, I : INTEGER; (* Temp values *)
(* Loop counters *)
BEGIN
FLAG := TRUE;
FOR J1 := 1 TO TOT DO
BEGIN
    IF FLAG
    THEN
    BEGIN
        IF NEWPER = INDEX[J1]
        THEN FLAG := FALSE;
        IF NEWPER < INDEX[J1]
        THEN
        BEGIN
            1 := J1;
            SAV1 := NEWPER;
            SAV2 := INDEX[I];
            WHILE SAV1 < 99 DO
            BEGIN
                INDEX[I] := SAV1;
                SAV1 := SAV2;
                SAV2 := INDEX[I + 1];
                I := I + 1;
            END;
            FLAG := FALSE;
        END;
    END;
END;
END;

PROCEDURE ADDTOPLIST(NODE : POINTER; VAR REC : PRTTREC);
BEGIN
WITH REC DO
BEGIN
PD := NODE^.PD;
PERIOD := NODE^.PERIOD;
DAY := NODE^.DAY;
COURSE := NODE^.CRNAM;
ROOM := NODE^.ROOM;
SEM := NODE^.SEM;
TEACHER := NODE^.TEACHER;
CODE := NODE^.CRNUM;
END;
END;

PROCEDURE PRINTCOUR(INDLIST : INDEXLIST; LIST : PRAR; COUNT : INTEGER);
(* PRINTS SCHEDULED COURSES FROM LIST *)
BEGIN
FOR J := 1 TO COUNT - 1 DO
BEGIN
IF INDLIST[J] : 99
THEN
BEGIN
FOR I := 1 TO COUNT - 1 DO
BEGIN
IF LIST[I].PD = INDLIST[J]
THEN
BEGIN
WITH LIST[I] DO
BEGIN
WRITE(OUTDEVICE,PERIOD:4,DAY:9,COURSE:12,ROOM:7);
WRITELN(OUTDEVICE,SEM:5,TEACHER:8,CODE:8);
END
END
END
END
END
END;

BEGIN
(* WR SCH *)
REWRITE(OUTDEVICE,'PRINTER:');
WRITELN(OUTDEVICE);
WRITELN(OUTDEVICE);
WRITELN(OUTDEVICE);
WRITELN(OUTDEVICE,
STUDENT NAME: \text{\texttt{.SCHRECD\.STUDNAM:20}});
WRITELN(OUTDEVICE,
ID : \text{\texttt{.SCHRECD\.STUDID:5}});
WRITELN(OUTDEVICE,
GRADE: \text{\texttt{.SCHRECD\.GRADE:2}});
WRITELN(OUTDEVICE);
WRITELN(OUTDEVICE,
PERIOD DAYS COURSE ROOM SEM TEACHER CODE');
WRITELN(OUTDEVICE);
INDEX := 1;
INITINDEX(INDLIST);
FOR J := 1 TO TOTC DO
BEGIN
IF SCHRECD\.CRSHC[J,1] = 0
THEN
BEGIN
DIFFER := FALSE;
DIFSEM := FALSE;
NEW(NODE);
NODE\.CRNUM := SCHRECD\.CRSSCH[J,1];
NODE\.CRSEC := SCHRECD\.CRSSCH[J,2];
FINNODE(HEAD,NODE,FOUND,DIFFER,DIFSEM);
END
WHILE FOUND DO
BEGIN
SAVEPER := NODE^.PERIOD;
SAVESEM := NODE^.SEM;
ADDTOINDEX(INDLIST,NODE^.FD);
ADDTOPRLIST(NODE,LIST[INDEX]);
INDEX := INDEX + 1;
FOUND := FALSE;
IF DIFPER OR DIFSEM
THEN
BEGIN
NEW(NODE);
NODE^.CRNUM := SCHRECD^.CRSSCH[J,1];
NODE^.CRSEC := SCHRECD^.CRSSCH[J,2];
END;
IF DIFPER
THEN
BEGIN
DIFPER := FALSE;
NODE^.PERIOD := SAVEPER;
FINDDPNODE(HEAD,NODE,FOUND,DIFPER,DIFSEM)
END
ELSE
BEGIN
IF DIFSEM
THEN
BEGIN
DIFSEM := FALSE;
NODE^.SEM := SAVESEM;
FINDDSnode(HEAD,NODE,FOUND,DIFPER,DIFSEM)
END
END;
END;
PRINTCOUR(INDLIST,LIST,INDEX);
CLOSE(OUTDEVICE);
END;

BEGIN
GETOUT := FALSE;
PRESENT := LOW;
REPEAT
STR(PRESENT,FILEGRADE);
FILENAME := CONCAT(‘SCH:SCH’,FILEGRADE);
OPENFILE(Filename);
WHILE NOT EOF (SCHRECD) DO
BEGIN
MARK(HEAP);
WASCH;
RELEASE(HEAP);
SET(SCHRECD);
END;
CLOSE(SCHRECD);
IF PRESENT = HIGH
THEN GETOUT := TRUE
ELSE PRESENT := PRESENT + 1;
UNTIL GETOUT = TRUE;
END;

PROCEDURE INSERTNODE (VAR FRONT, ITEM : POINFE):
(* TO PUT NEW NODE IN LIST IN ORDER OF COURSE NUMBER, CRS SECTION, AND PER *)

VAR LOOKAHEAD, CHASER: POINTER;  (* LOOKAHEAD POINTER TO FIND PLACE *)  (* CHASER POINTER FOR INSERTION *)

BEGIN
LOOKAHEAD := FRONT;
CHASER := FRONT;

WHILE (LOOKAHEAD <> NIL) AND (ITEM^.CRNUM > LOOKAHEAD^.CRNUM) DO
  BEGIN
    CHASER := LOOKAHEAD;
    LOOKAHEAD := LOOKAHEAD^.LINK;
  END;

WHILE (LOOKAHEAD <> NIL) AND (ITEM^.CRNUM = LOOKAHEAD^.CRNUM) AND
      (ITEM^.CRSEC > LOOKAHEAD^.CRSEC) DO
  BEGIN
    CHASER := LOOKAHEAD;
    LOOKAHEAD := LOOKAHEAD^.LINK;
  END;

WHILE (LOOKAHEAD <> NIL) AND (ITEM^.CRNUM = LOOKAHEAD^.CRNUM) AND
      (ITEM^.CRSEC = LOOKAHEAD^.CRSEC) AND (ITEM^.PD > LOOKAHEAD^.PD) DO
  BEGIN
    CHASER := LOOKAHEAD;
    LOOKAHEAD := LOOKAHEAD^.LINK;
  END;

IF (LOOKAHEAD <> NIL)
  THEN
    IF LOOKAHEAD <> FRONT
      THEN
        BEGIN
          ITEM^.LINK := LOOKAHEAD;
          CHASER^.LINK := ITEM;
        END  (* INSERT IN MIDDLE OF LIST *)
      ELSE
        BEGIN
          ITEM^.LN := FRONT;
          FRONT := ITEM;
        END  (* INSERT AT FRONT *)
    ELSE
      CHASER^.LN := ITEM;

    END;

PROCEDURE FETCHIT(VAR FIRST: POINTER);
  (* RETRIEVES COURSE LIST FROM FILE AND ENTERS COURSES IN A LINKED LIST *)

VAR CRS: POINTER;  (* COURSE RECORD *)

BEGIN
RESET(CRREC^.CR; COURSEFILE);

WHILE NOT EOF(CRREC) DO
  BEGIN
    NEW(CRS);
    CRS^. := CRREC;
    IF FIRST = NIL
      THEN FIRST := CRS
    ELSE INSERTNODE(FIRST, CRS);
    GET(CRREC);
  END;
CLOSE(CRREC);
END;
FUNCTION ASKCHOICE(X,Y : INTEGER; Q : STRING; MIN,MAX : INTEGER) : INTEGER;
    (* ASK USER FOR A NUMBER FROM MIN TO MAX, AND REPEAT UNTIL OBTAINED *)
VAR
    ANSWER : INTEGER;  (* A NUMBER *)
BEGIN
    REPEAT
        GOTOXY(X,Y);
        WRITE(Q,' ( ','MIN','..','MAX',') : ');
        READLN(ANSWER);
        UNTIL (ANSWER >= MIN) AND (ANSWER <= MAX);
    ASKCHOICE := ANSWER
END;

BEGIN
    (* PRINTSCHEDULES *)
    HEAD := NIL;
    FETCHIT(HEAD);
    PAGE(OUTPUT);
    LOW := ASKCHOICE(0,5,'WHAT IS THE LOWEST GRADE YOU WISH CLASS LISTS FOR',7,12);
    HIGH := ASKCHOICE(0,10,'WHAT IS THE HIGHEST GRADE YOU WISH CLASS LISTS FOR',7,12);
    PRINTSTSCCH;
END.  (* PRINTSCHEDULES *)
10.9. PRINTCLASSLISTS
PROGRAM PRINTCLASSLISTS;

(*-----------------------------------------------*)
(*                                                  *)
(* THIS PROGRAM PRINTS CLASS LISTS USING RECORDS FROM THE *)
(* SCHEDULE FILES.                                   *)
(*                                                  *)
(* FILES USED                                        *)
(* - CR : COURSEFILE                                *)
(* - SCH : SCH(GRADE)                              *)
(*                                                  *)
(* OUTPUT                                           *)
(* - A CLASS LIST FOR EVERY COURSE SCHEDULED.       *)
(* - A CLASS LIST CONTAINS                          *)
(*   - COURSE NAME                                 *)
(*   - COURSE NUMBER                               *)
(*   - COURSE SECTION                              *)
(*   - DAYS THE CLASS IS SCHEDULED                 *)
(*   - ROOM                                        *)
(*   - TEACHER NUMBER                              *)
(*   - LIST OF STUDENTS IN CLASS                   *)
(*     - STUDENT ID                                *)
(*     - STUDENT GRADE                             *)
(*     - STUDENT NAME                              *)
(*                                                  *)
(*---------------------------------------------------*)

CONST
TOTC    = 11;        (* TOTAL COURSES *)

TYPE
SC      = ARRAY[1..2] OF INTEGER;
TALLY   = ARRAY[1..TOTC] OF SC;
SCHPTR  = ^SCHREC;
CLPTR   = ^CLREC;
YEAR    = 7..12;
POINTER = ^COURSE;
COURSENUMBER = 0..999;
COURSESECTION = 1..99;
MAXSIZE  = 1..999;
PERIODNUMBER = 1..16;
SEMESTER = 1..3;
TEACHERNUMBER = 1..99;
CLREC    =
    RECORD
    CRS    : COURSENUMBER;        (* COURSE NUMBER *)
    SEC    : COURSESECTION;       (* COURSE SECT *)
    STUDNAM : STRING[20];        (* STUDENT NAME *)
    STUDID  : INTEGER;            (* STUDENT ID *)
    GRADE  : YEAR;                (* STUDENT GRADE*)
    LCL    : CLPTR;                (* POINTER *)
    END;

SCHREC  =
    RECORD
    STUDNAM1 : STRING[20];        (* STUDENT NAME *)
    STUDID  : INTEGER;            (* STUDENT ID *)
    END;
GRADE YEAR;
CRSSCH TALLY;
L SCHPTR;
END;

COURSE RECORD
CRNUM : COURSENUMBER;
CRNAM : STRING[10];
CRSEC : COURSESECTION;
MAX : INTEGER;
PERIOD : PERIODNUMBER;
PD : PERIODNUMBER;
DAY : STRING[5];
SEM : SEMESTER;
ROOM : STRING[3];
TEACHER : TEACHERNUMBER;
LINK : POINTER;
END;

VAR SCHREC TEMP FOUND, GETOUT PRESEN FILEGR S, FNAME, FILENAI LOW, HIGH HEAP SCHNOD SHEAD CRNUM CRNAM CRSEC MAX PERIOD SEM DAY TEACHER ROOM NODE, HEAD CRREC OUTDEV C TAL STUDC J CLNODE CLHEAD

**********

PROCEDURE (* Pf

BEGIN
REWFF
PAGE
IF FRONT = NIL
THEN
BEGIN
    WRITELN (OUTDEVICE,'LIST IS EMPTY. ');
END
(* EMPTY *)
ELSE
(* THERE ARE COURSES *)
BEGIN
    WRITELN(OUTDEVICE,'CLASS LIST COURSE : ','NODE^.CRNAME);
    WRITELN(OUTDEVICE,' NUMBER : ','NODE^.CRNUM: 3, ', NODE^.CRSEC: 2);
    WRITELN(OUTDEVICE,' PERIOD : ','NODE^.PERIOD: 2);
    WRITELN(OUTDEVICE,' DAYS : ','NODE^.DAY: 5);
    WRITELN(OUTDEVICE,' SEM : ','NODE^.SEM: 2);
    WRITELN(OUTDEVICE,' ROOM : ','NODE^.ROOM: 3);
    WRITELN(OUTDEVICE,' TEACHER : ','NODE^.TEACHER: 2);
    WRITELN(OUTDEVICE);
    WHILE FRONT <> NIL DO
BEGIN
    WRITELN(OUTDEVICE,FRONT^.STUDID: 5,FRONT^.GRADE: 6, ',FRONT^.STUDNAM);
    FRONT := FRONT^.LCL;
END;
END;
(* ELSE LIST NOT EMPTY *)
CLOSE(OUTDEVICE);
(* TURN OFF PRINTER *)
END;
(* PRINTLIST *)

*********************************************************************

PROCEDURE INSERTSCH(VAR FRONT, ITEM : CLPTR);
(* INSERTS A STUDENT CLASS LIST RECORD INTO LINKED LIST *)
VAR LOOKAHEAD, CHASER : CLPTR;
(* POINTER TO FIND PLACE *)
(* POINTER FOR INSERTION *)
BEGIN
    LOOKAHEAD := FRONT;
    CHASER := FRONT;
    WHILE (LOOKAHEAD <> NIL) AND (ITEM^.STUDNAM > LOOKAHEAD^.STUDNAM) DO
BEGIN
    CHASER := LOOKAHEAD;
    LOOKAHEAD := LOOKAHEAD^.LCL
END;
IF LOOKAHEAD <> NIL
THEN
    IF LOOKAHEAD <> FRONT
THEN
BEGIN
    ITEM^.LCL := LOOKAHEAD;
    CHASER^.LCL := ITEM;
END
ELSE
BEGIN
    ITEM^.LCL := FRONT;
    FRONT := ITEM;
END
ELSE
    CHASER^.LCL := ITEM;
END:

END:

END:

END:
PROCEDURE IOPRINT(ERRTYPE: INTEGER);
(* PRINT ERROR MESSAGE *)

BEGIN
GOTOXY(0,22);
WRITE(CHR(7));
CASE ERRTYPE OF
  1 : WRITELN('DISK READ ERROR.');
  2 : WRITELN('BAD UNIT NUMBER.');
  3 : WRITELN('ILLEGAL OPERATION.');
  4 : WRITELN('UNDEFINED HARDWARE ERROR.');
  5 : WRITELN('UNIT NO LONGER ON LINE.');
  6 : WRITELN('FILE NOT IN DIRECTORY.');
  7 : WRITELN('ILLEGAL FILE NAME.');
  8 : WRITELN('INSUFFICIENT DISK SPACE.');
  9 : WRITELN('NO SUCH VOLUME ON LINE.');
 10 : WRITELN('NO SUCH FILE ON VOLUME.');
 11 : WRITELN('DUPLICATE FILE.');
 12 : WRITELN('ATTEMPT TO OPEN AN OPEN FILE.');
 13 : WRITELN('FILE NOT OPEN.');
 14 : WRITELN('ERROR IN READING INTEGER OR REAL DATA.');
 15 : WRITELN('RING BUFFER OVERFLOW.');
END;
END;

PROCEDURE NODISK(DISKNAME:STRING; DISKERR : INTEGER);
(* CHECKS FOR ERROR 9 *)
VAR CH : CHAR;

BEGIN
IF (DISKERR <> 0) AND ((DISKERR < 9) OR (DISKERR > 10))
  THEN IOPRINT(DISKERR);
IF (DISKERR = 9)
  THEN
    BEGIN
      GOTOXY(0,22);
      WRITELN('NO ',DISKNAME, ' DISKETTE!');
      WRITELN('PLEASE INSERT DISKETTE AND HIT ANY KEY');
      READ(KEYBOARD,CH);
    END;
    IF (DISKERR = 10)
      THEN REWRITE(SCHRECD,DISKNAME):
END;

PROCEDURE OPENFILE(FILENAME : STRING);
(* OPENS A FILE AND CHECKS FOR ERRORS *)
VAR IOERR : INTEGER;

BEGIN
REPEAT
(*$I-*)
  RESET(SCHRECD,FILENAME);
(*$I++*)
  IOERR := IORESULT;
  NODISK(FILENAME,IOERR);
UNTIL IOERR = 0;
END;
PROCEDURE OPENCFILE(FILENAME : STRING);
(* OPENS A COURSE FILE AND CHECKS FOR ERRORS *)

VAR IOERR : INTEGER;

BEGIN
REPEAT
(*!-*)
RESET(CRREC,FILENAME);
(*!+*)
IOERR := IORESULT;
NODISK(FILENAME,IOERR);
UNTIL IOERR = 0;
END;

******************************************************************************

PROCEDURE INSERTNODE (VAR FRONT, ITEM : POINTER);
(* TO PUT NEW NODE IN LIST IN ORDER OF COURSE NUMBER, CRS SECTION, AND PER *)

VAR LOOKAHEAD,
CHASER : POINTER;
GETOUT : BOOLEAN; (* DON'T INSERT THIS NODE*)

BEGIN (* INSERTNODE *)
LOOKAHEAD := FRONT; (* LOOKAHEAD POINTER TO FIND PLACE *)
CHASER := FRONT; (* CHASER POINTER FOR INSERTION *)
GETOUT := FALSE;
WHILE (LOOKAHEAD <> NIL) AND (ITEM^.CRNUM > LOOKAHEAD^.CRNUM) DO
BEGIN (* TRAVERSE THE LIST TO THE INSERT POINT *)
CHASER := LOOKAHEAD (* MOVE CHASER DOWN LIST *)
LOOKAHEAD := LOOKAHEAD^.LINK (* MOVE LOOKAHEAD DOWN *)
END; (* WHILE *)
WHILE (LOOKAHEAD <> NIL) AND (ITEM^.CRNUM = LOOKAHEAD^.CRNUM) AND
(ITEM^.CSSEC > LOOKAHEAD^.CSSEC) DO
BEGIN (* TRAVERSE THE LIST TO THE INSERT POINT *)
CHASER := LOOKAHEAD (* MOVE CHASER DOWN LIST *)
LOOKAHEAD := LOOKAHEAD^.LINK (* MOVE LOOKAHEAD DOWN *)
END; (* WHILE *)
IF (LOOKAHEAD <> NIL) AND (ITEM^.CRNUM = LOOKAHEAD^.CRNUM) AND
(ITEM^.CSSEC = LOOKAHEAD^.CSSEC)
THEN GETOUT := TRUE;
IF NOT GETOUT
THEN
BEGIN
IF (LOOKAHEAD <> NIL)
THEN (* NOT AT END OF LIST *)
IF LOOKAHEAD <> FRONT
THEN (* IN MIDDLE OF LIST *)
BEGIN
ITEM^.LINK := LOOKAHEAD;
CHASER^.LINK := ITEM;
END
ELSE
BEGIN
ITEM^.LINK := FRONT;
FRONT := ITEM;
END
ELSE (* MUST BE LARGEST, SO INSERT AT END *)

CHASER^.LINK := ITEM;
PROCEDURE FETCHIT(VAR FIRST : POINTER);
(* RETRIEVES COURSE LIST FROM FILE AND ENTERS COURSES IN A
LINKED LIST *)

VAR CRS : POINTER;

BEGIN
RESET(CRREC,'CR:COURSEFILE');

WHILE NOT EOF(CRREC) DO
BEGIN

NEW(CRS);
CRS^ := CRREC^;
IF FIRST = NIL
THEN FIRST := CRS
ELSE INSERTNODE(FIRST,CRS);
GET(CRREC);
END;
CLOSE(CRREC);
END;

FUNCTION ASKCHOICE(X,Y : INTEGER; Q : STRING; MIN,MAX :INTEGER) : INTEGER;
(* ASK USER FOR A NUMBER FROM MIN TO MAX, AND REPEAT UNTIL OBTAINED *)

VAR ANSWER : INTEGER;

BEGIN
REPEAT
GOTOXY(X,Y);
WRITE(Q,' (','MIN','..',MAX,') : ');
READLN(ANSWER);
UNTIL (ANSWER >= MIN) AND (ANSWER <= MAX);
ASKCHOICE := ANSWER
END;

PROCEDURE FILLTEMP;
(* FILLS TEMP FILE WITH STUDENT CLASS LIST RECORDS *)

BEGIN
PAGE(OUTPUT);
LOW := ASKCHOICE(0,5,'WHAT IS THE LOWEST GRADE YOU WISH CLASS LISTS FOR',7,12);
HIGH := ASKCHOICE(0,10,'WHAT IS THE HIGHEST GRADE YOU WISH CLASS LISTS FOR',7,12);
GETOUT := FALSE;
PRESENT := LOW;
REWRITETEMP(TEMP,'TEMP:T');
REPEAT
STR(PRESENT,FILEGRADE);
FILENAME := CONCAT('SCH:SCH',FILEGRADE);
OPENFILE(FILENAME);
C := 1;
WHILE NOT EOF (SCHREC) DO
BEGIN
FOR J := 1 TO TOTC DO
BEGIN
IF SCHREC^..CRSSCH[J,1] > 0
THEN
BEGIN
STUDC.CRS := SCHREC^..CRSSCH[J,1];
STUDC.SCH := SCHREC^..CRSSCH[J,2];
END;
END;
END;
STUDC.STUDNAM := SCHRECD^ .STUDNAM;
STUDC.STUDID := SCHRECD^ .STUDID;
STUDC.GRADE := SCHRECD^ .GRADE;
STUDC.LCL := NIL;
TEMP^ := STUDC;
PUT(TMP);
END;
END;
GET(SCHRECD);
END;
CLOSE(SCHRECD);
IF PRESENT = HIGH
THEN GETOUT := TRUE
ELSE PRESENT := PRESENT + 1;
UNTIL GETOUT = TRUE;
CLOSE(TMP,LOCK);
END;

BEGIN
(* PRINT CLASS LISTS *)

FILL(TMP);
HEAD := NIL;
FETCHIT(HEAD);
NODE := HEAD;
WHILE NODE <> NIL DO
BEGIN
RESET(TMP, 'TEMP: T ');
MARK(HEAP);
CLHEAD := NIL;
REPEAT
IF (NODE^ .CRNUM = TEMP^ .CRS) AND (NODE^ .CRSEC = TEMP^ .SEC)
THEN
BEGIN
NEW(CLNODE);
CLNODE^ := TEMP^ ;
IF CLHEAD = NIL
THEN CLHEAD := CLNODE
ELSE INSERTSCH(CLHEAD, CLNODE);
END;
GET(TMP);
UNTIL EOF(TMP);
CLOSE(TMP);
PRINTLIST(CLHEAD, NODE);
RELEASE(HEAP);
NODE := NODE^ .LINK
END;
END. (* PRINT CLASS LISTS *)