# A SYSTEMS ENGINEERING APPROACH TOWARDS SCHEDULING AND OPERATIONS AT THE PENINSULA CENTER

by

Steven Macri

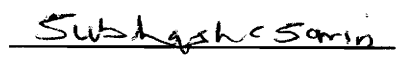
Report submitted to the Graduate Faculty of the Virginia Polytechnic Institute and State University in partial fulfillment of the requirements for the degree of

MASTER OF SCIENCE

in

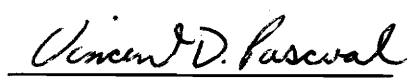
Systems Engineering

APPROVED:



S. C. Sarin, Chairman

B. S. Blanchard



V. D. Pascual

December, 1992

Blacksburg, Virginia

# A SYSTEMS ENGINEERING APPROACH TOWARDS SCHEDULING AND OPERATIONS AT THE PENINSULA CENTER

by Steven Macri

Committee Chairman: S. C. Sarin

Industrial and Systems Engineering

(ABSTRACT)

This report identifies the activities of the Peninsula Center (P.C.) located in Hampton, Virginia and defines the dynamic system that exists at this facility. It explains the functions and responsibilities of the personnel, and ultimately details the site's activities throughout the semester. This is done in an effort to illustrate the opportunity for improvements in efficiency and enhancement of system capabilities.

In particular, considerable effort is concentrated on defining a system of scheduling that will efficiently utilize the site's facilities to meet the demands of the televised, broadcast and live classes offered. Currently, this service is being performed manually. This results in several weeks of trial-and error course-classroom assignments leading to adjustments and re-work at the beginning of each semester. This is a tedious job which is a

burden on the technical staff during the critical portion of the semester.

Given the definition of the site requirements, parameters and limitations, a scheduling system is defined and a software package is developed to support the Center's needs. This software package takes into account the resources (personnel, equipment, available space, etc.) at the site. Considerations are made for the uncertainty of class enrollment, differing session priorities and staff convenience. The result is a software package that is run at the site which allows the planners to achieve a feasible and efficient classroom configuration throughout the semester.

A system life-cycle approach is assumed throughout the project. Initial establishment of system requirements is achieved through a series of sessions with the P.C. technical personnel. This contact is continued throughout the lifetime of the product development to ensure that the product will meet the site's needs. The software system is evaluated through formal testing/re-testing, analysis, data collection and is finally installed at the site for personnel use.

TABLE OF CONTENTS

## LIST OF TABLES

Table Page

LIST OF FIGURES

# Chapter 1

# INTRODUCTION

As the world moves closer to a global community the exchange of ideas on a professional level frequently demands technological assistance. By virtue of the fact that technology is accelerating at such a rapid rate, it is essential that the professional work force continuously improve its knowledge base of today's complicated systems. These two points have caused a need that is not satisfied by traditional educational institutions. It is becoming more apparent that full time professionals require a method of education that suits their needs and conforms to their schedule. The State of Virginia has employed such a program. Through a coordinated effort among several of the state's schools and technology centers, Old Dominion University (ODU) has provided a highly successful educational distribution network that makes available undergraduate, graduate and doctorate level course work to students throughout the state.

## Identification of Need

Figure 1 shows the locations of the various educational sites throughout the state involved in the educational

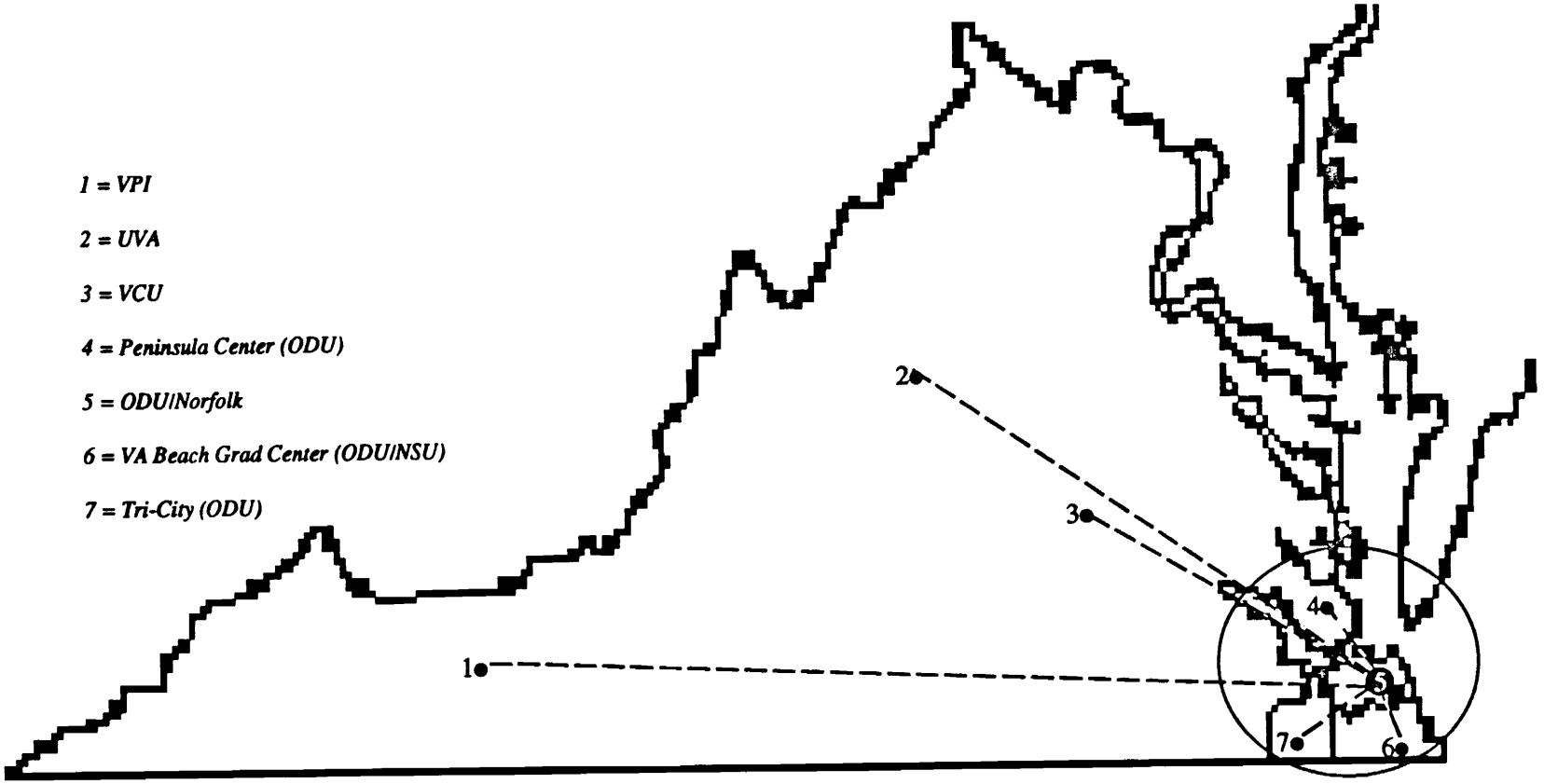



**Figure 1.** Virginia Educational Distribution Network - Serving Hampton Roads

distribution network that serves the Hampton Roads region of Virginia. This shows participating universities such as Virginia Polytechnic Institute and State University (VPI), University of Virginia (UVA) and Virginia Commonwealth University (VCU). These universities serve as a network to provide students throughout the state educational opportunities not previously offered to them because no college or university was conveniently located in their area. The Hampton Roads (or Tidewater) area benefits a great deal from this network. As circled in Figure 1, it consists of the Hampton Roads Peninsula, as well as the cities of Norfolk, Virginia Beach, Suffolk, Chesapeake and Portsmouth. The region has within it many large employers, some of which are located on the Peninsula and shown in Figure 2, whose employees often seek further education. Demographic data collected for the Peninsula Virginia Planning District No. 1 and ODU in 1984 showed that over 6,000 engineers worked on the Hampton Roads Peninsula alone. Of those surveyed, 38% of respondents said they planned on pursuing a graduate program. Another influx of engineers has been estimated (some figures are as high as 2,000 to 4,000) as the Continuous Electron Beam Accelerator Facility (CEBAF) project progresses. There was clearly a need for a technology and education center in this area. In response to this need, the participating universities shown in Figure 1 (ODU, VPI, UVA and VCU) established this network which

Newport News Shipbuilding

NASA

Canon Corporation

Bendix Corporation

CEBAF

State of Virginia

Langley Air Force Base

Gannet Fleming

Siemens

Smith, Demer and Norman

PRC Kentron

ITT

Figure 2. Hampton Roads Peninsula Regional Employers

consisted of learning centers such as the Peninsula Center (also used throughout this report as the P.C. or the Center) located in Hampton, Virginia. Although the Peninsula Center is only one of the sites within the educational network, a highlight of its activities will serve to illustrate the capabilities at the other sites and the network as a whole.

The Peninsula Center was established in August 1986 and has experienced growth ever since. Televised, live and broadcast classes as well as tele/video conferences are provided for the professional and business community on the Peninsula. Figure 3 shows an organizational chart of the Hampton Roads Academic System. This shows the capabilities available at each of the sites previously shown in Figure 1. The success of the P.C. is evident by the community interest and corresponding expansion over the past six years. Its recent expansion has nearly doubled the area and added thirteen classrooms. With this increase in size came the availability of nearly twice the course work that had previously been offered at the Center. Figure 4 shows the twenty four degrees now being offered through ODU, VPI and UVA. Also provided is Table 1 which shows the enrollment statistics for each semester since the Center's conception. As can be seen, student enrollment in the Fall 1992 semester is nearly ten times the amount present in the Fall 1986

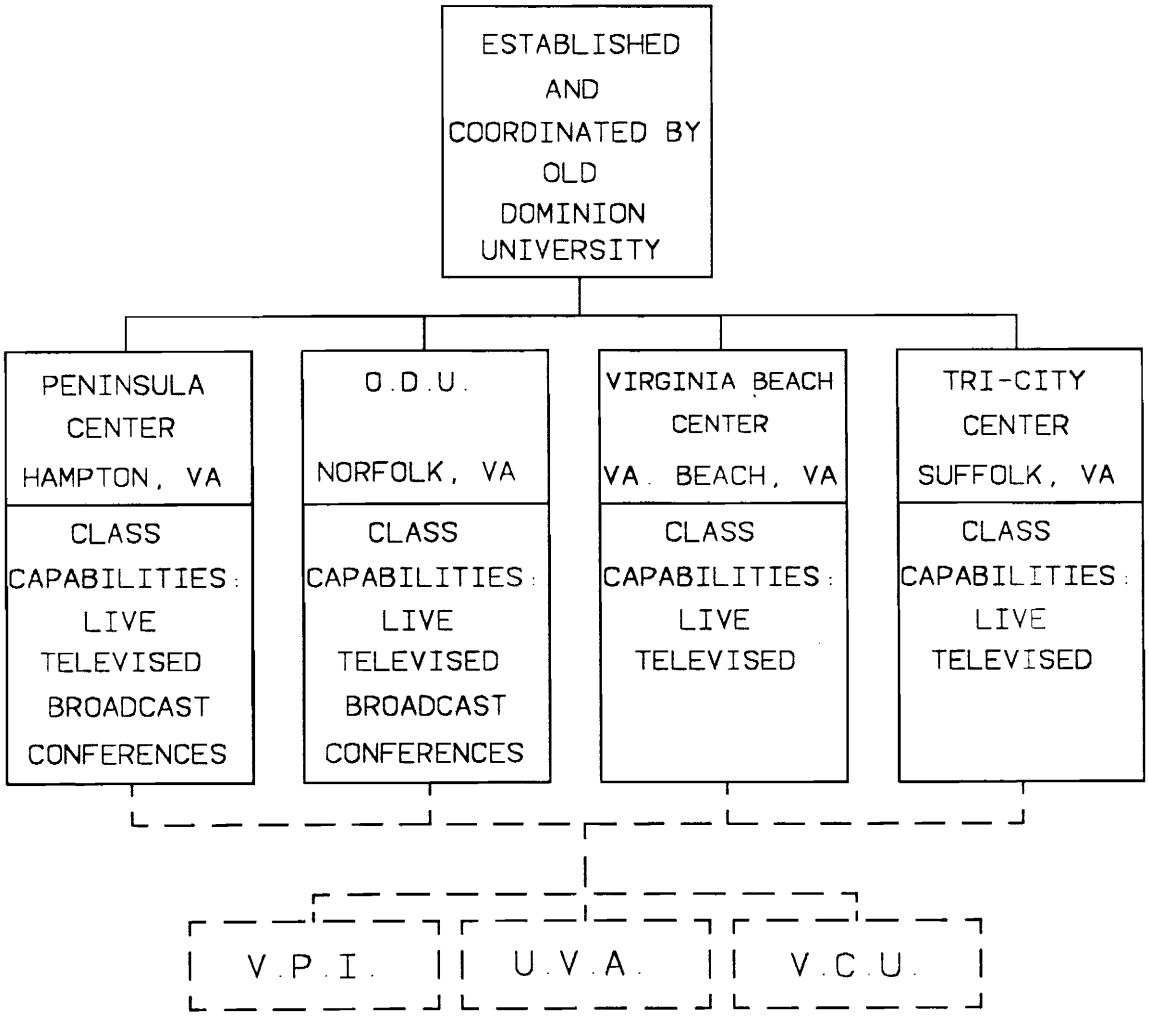



FIGURE 3. HAMPTON ROADS ACADEMIC SYSTEM

## OLD DOMINION UNIVERSITY'S
## *PENINSULA CENTER*
## DEGREE PROGRAM

**ODU:**

- Business Administration
- Civil Engineering
- Computer Science/Engineering
- Early Childhood Education
- Educational Leadership
- Electrical Engineering
- Engineering Management (Ph.D.)
- International Studies
- Nursing
- Mechanical Engineering and Mechanics
- Public Administration

**UVA:**

- Chemical Engineering
- Civil Engineering (Structural)
- Electrical Engineering
- Material Science
- Nuclear Engineering
- Systems Engineering

**VPI:**

- Aerospace and Ocean Engineering
- Civil Engineering (Environmental)
- Electrical Engineering
- Industrial and Systems Engineering (Master of Engineering Administration)
- Mechanical Engineering
- Systems Engineering

FIGURE 4. DEGREES AT THE CENTER

TABLE I. PENINSULA CENTER ENROLLMENT STATISTICS

| SEMESTER | NO. OF CLASSES | NO. OF STUDENTS |
|---|---|---|
| FALL 1986 | 20 | 75 |
| SPRING 1987 | 23 | 85 |
| SUMMER 1987 | 6 | 15 |
| FALL 1987 | 26 | 100 |
| SPRING 1988 | 30 | 108 |
| SUMMER 1988 | 5 | 31 |
| FALL 1988 | 40 | 124 |
| SPRING 1989 | 42 | 136 |
| SUMMER 1989 | 7 | 40 |
| FALL 1989 | 40 | 133 |
| SPRING 1990 | 45 | 157 |
| SUMMER 1990 | 10 | 65 |
| FALL 1990 | 40 | 160 |
| SPRING 1991 | 48 | 228 |
| SUMMER 1991 | 12 | 74 |
| FALL 1991 | 50 | 287 |
| SPRING 1992 | 51 | 316 |
| SUMMER 1992 | 20 | 127 |
| FALL 1992 | 82 | 622 |

semester. Figure 5 gives a graphical representation of the past growth. Also provided are projected enrollments for the future. These are based on a Least Squares Fit of the past enrollments.

With these increases in classroom capability and student activity come complications beyond those that were present in 1986. Tasks which were relatively easy to accomplish now require more time and resources from a staff size that is not planned to grow due to budget limitations. The most apparent example is the scheduling of classes. At one time, this was a task that was easily accomplished by one person manually. A semester enrollment of only seventy five students made it very easy to accommodate all classes in the Center's rooms. As the enrollment and classes increased over the semesters, scheduling became much more of a burden. In order to accommodate enrollment changes during the first weeks in the semester, it required the constant attention of one staff member to update the course-classroom assignments. Given the semester start is so critical, this situation was highly undesirable. Figure 6 shows this dynamic process in flow diagram form.

The expansion of the Center in the Fall 1992 semester and projected increases in student enrollment in the future

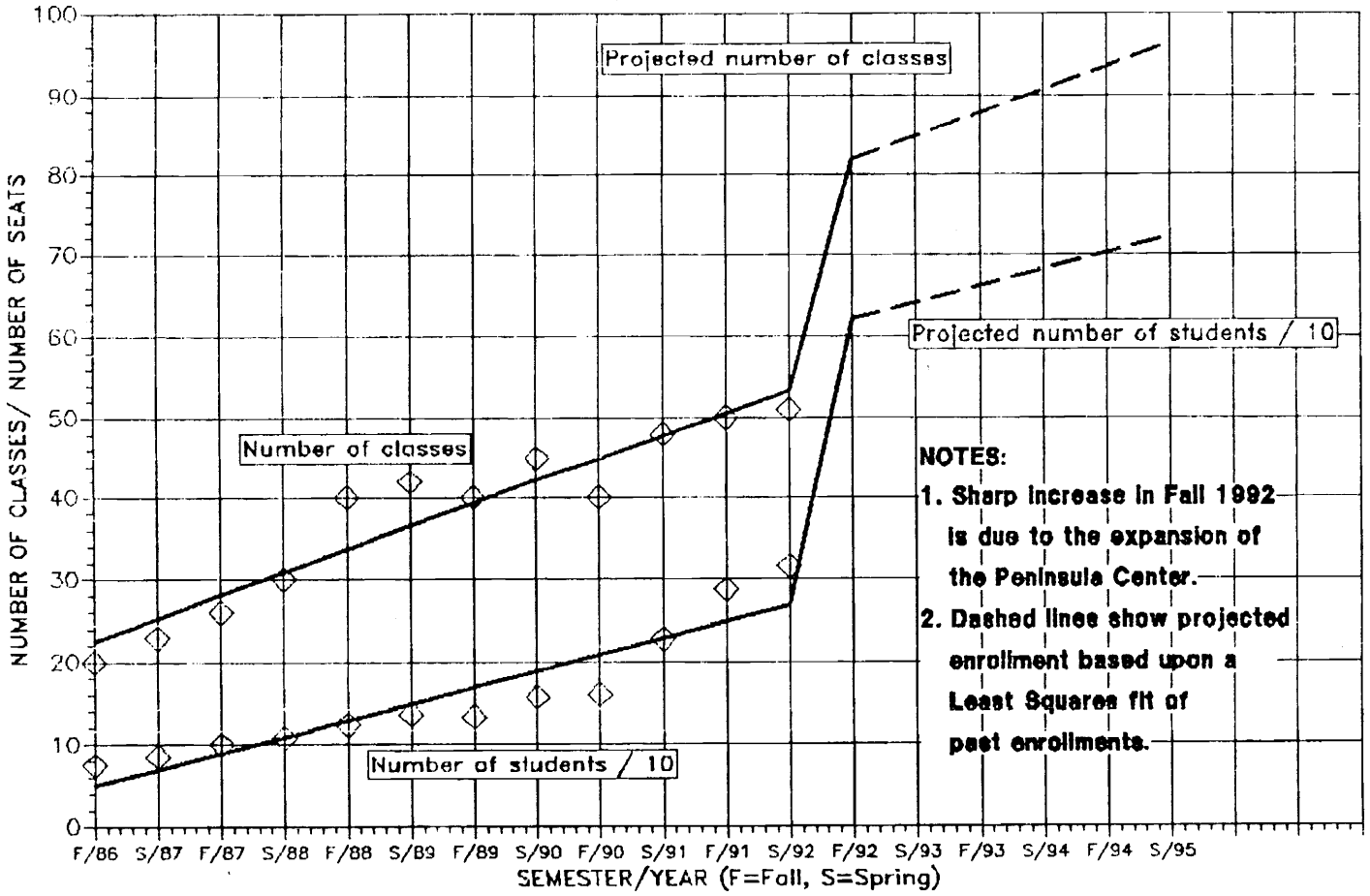



FIGURE 5. PENINSULA CENTER ENROLLMENT-PAST AND PROJECTED

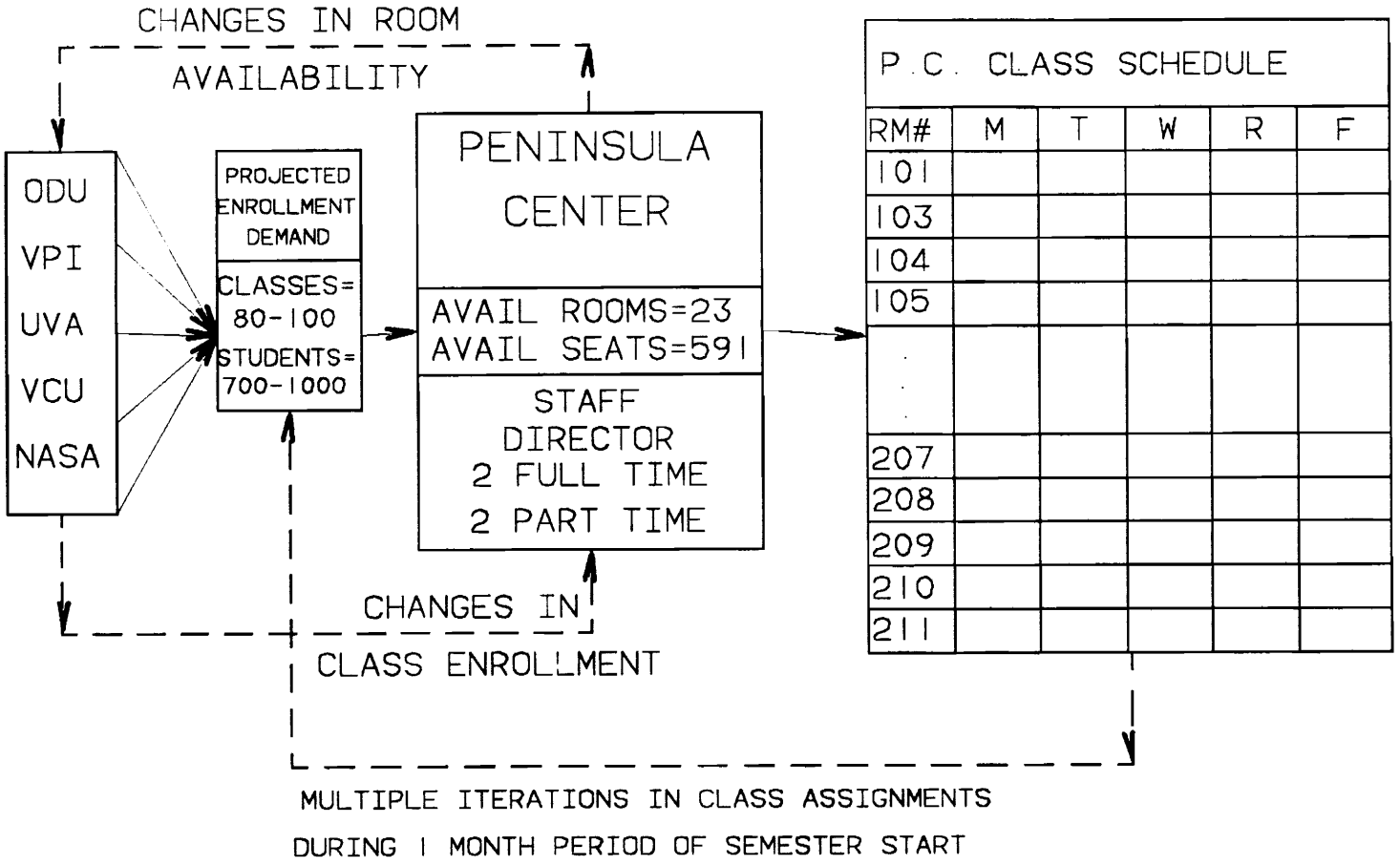



FIGURE 6. PENINSULA CENTER SCHEDULING DIAGRAM

would only compound existing scheduling problems. During discussions in the Fall of 1991 with the staff, the decision was made to investigate the feasibility of an automated system of scheduling. This was necessary to meet the anticipated needs of the future. This would greatly improve the efficiency of the Center. Not only would such a system allow the staff to react quickly to day-to-day changes in enrollments but it would eliminate the need for additional staff personnel. With the manual method of scheduling, it was estimated that one staff member would be required to monitor and update the course and classroom assignments during the one month period centered around the semester start. This was based on past enrollments and projected future growth of demand. Also, the automated method could provide a way of most efficiently utilizing the available resources by incorporating a systematic approach to allocating classrooms based on class size. These aspects, as well as the system development, are discussed in more detail in Chapters 3 and 4, **LIFE-CYCLE OF THE P.C. SCHEDULER** and **THE COURSE-CLASSROOM ASSIGNMENT PROCESS.**

## Peninsula Center

In order to meet the needs of the Peninsula's engineering community identified above, ODU established a

high-technology center in 1986 at Hampton, Virginia. The facility provides graduate-level training and research-related services that enhances the region's economic development. It was initially established as the Peninsula Graduate Engineering Center and has since been renamed as the Peninsula Center. As a result of the strong interest and funding from the state, the Peninsula Center now offers courses in undergraduate non-engineering disciplines as well. The principal mission of ODU's P.C. is to extend the resources of ODU, VPI, UVA and VCU to the Peninsula by serving adult, part-time students, providing a full range of educational services to working adults at times convenient to them.

Through the instructional television program the P.C. broadcasts and receives live, fully interactive telecourses. Students can see and hear the broadcast courses on TV monitors, located in each classroom, and may interact with the instructor by means of live two-way audio linkage [1]. All televised classes are recorded for student viewing at a later time if needed.

Due to the increased usage of the P.C. since its establishment, it has expanded beyond its original size of

8000 sq. ft. in 1986. The changes include the following [2]:

(1) Growth to 10,500 sq. ft. in 1989

(2) Broadcasting capabilities in 1990

(3) Two-way video conferences and national broadcast transmission via fiber link in 1991

(4) Expansion to 20,700 sq. ft. in 1992.

With this expansion comes more challenges.

## Virginia Cooperative Graduate Engineering Program

Graduate engineering education in Hampton Roads has been significantly expanded through a cooperative agreement among the Commonwealth's three largest engineering schools : Old Dominion University (ODU), Virginia Polytechnic Institute and State University (VPI) and University of Virginia (UVA). The Schools of Engineering at these three universities have developed the program in response to the diverse continuing education needs of engineers working in industry and government. Graduate engineering courses, leading to a Master of Science or Master of Engineering degree, are offered through an interactive satellite telecommunications system providing live televised instruction from broadcast studios operated by ODU, VPI, and

UVA [3]. "Receive" classrooms are located on the Old Dominion University campus, in Hampton at the university's Peninsula Center (P.C.) and other sites in Virginia Beach, Norfolk and Suffolk.

## Non-Engineering Opportunities

The Hampton Roads Peninsula has a high concentration of technical and professional personnel. The need to offer undergraduate and graduate level courses, as discussed above, was recognized. With this, several locations on the Peninsula were established to provide programs for working adults. These programs are offered primarily in the late afternoon or evening. Programs such as Nursing, Education and Public Administration are available at the locations either live or by interactive video/audio [2]. Instructors who participate are employed at facilities such as ODU, NASA, and CEBAF.

## Chapter 2

## SYSTEM OVERVIEW OF THE P.C. - CURRENT OPERATIONS

### Physical Configuration

The Center is located in Hampton, Virginia in the center of the Hampton Roads Peninsula. Its location makes it a convenient learning center for the area's engineering and professional personnel. With the most recent expansion in September, 1992 the area of the facility is now 20,700 square feet. The area is informally divided into two areas. The "Existing Space" consists of 10 classrooms that have been in existence since 1989 while the "Expanded Space" consists of 13 newly added classrooms. These amounts are based on the use of the Computer Lab, one Conference Room and the Ampitheatre for class instruction. Figures 7 and 8 show the plan view of each space while Table 2 lists each room and its capability. As can be seen, there are four types of rooms at the Center. These are described below;

(1) "Live" Classroom - These rooms are capable of live instruction only. A blackboard and instructor table are required equipment.

(2) "Television" Classroom - These rooms are capable of television instruction only. Television monitors, student microphones and wiring/cable to support this service are required equipment.

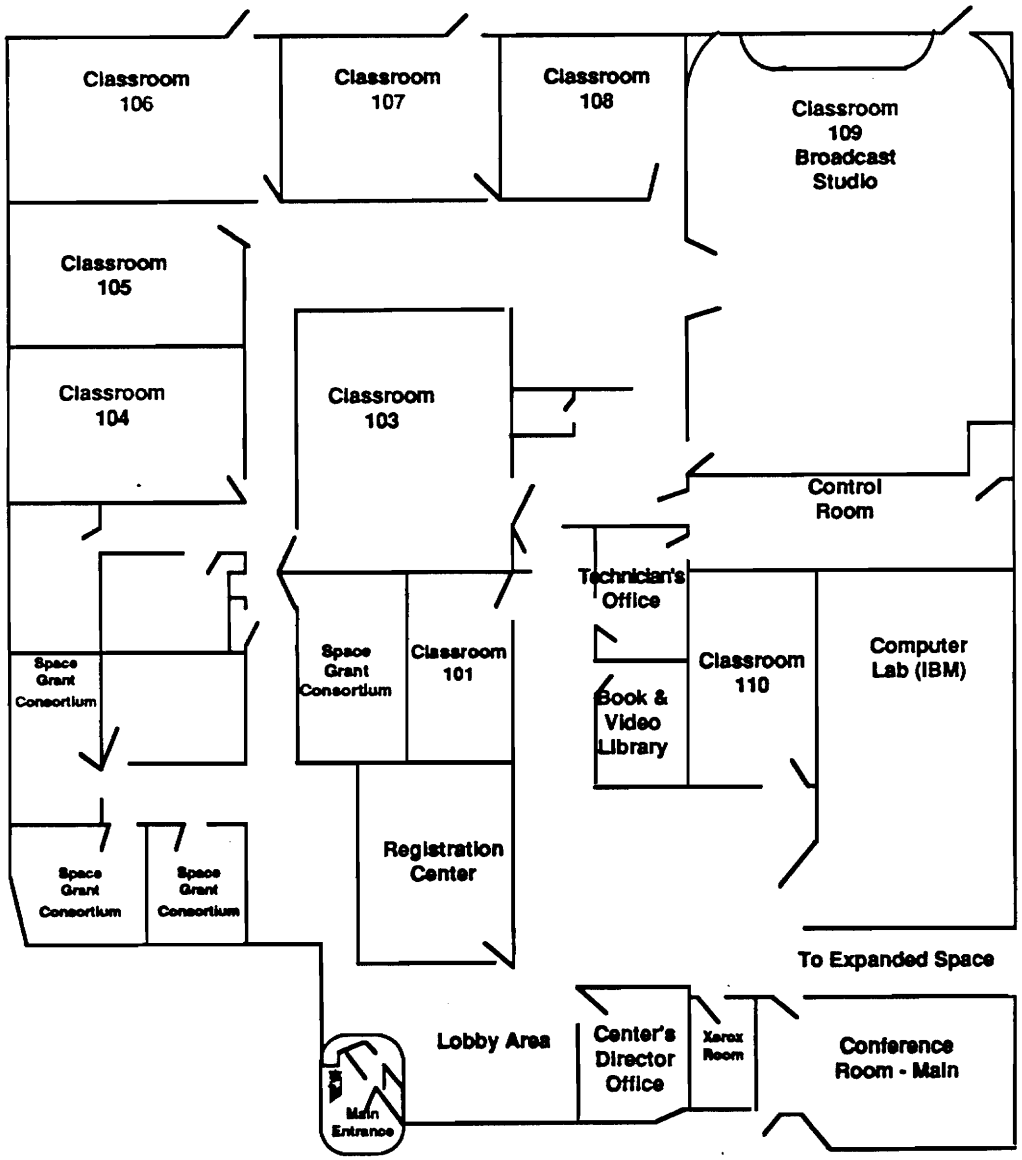



FIGURE 7. EXISTING SPACE

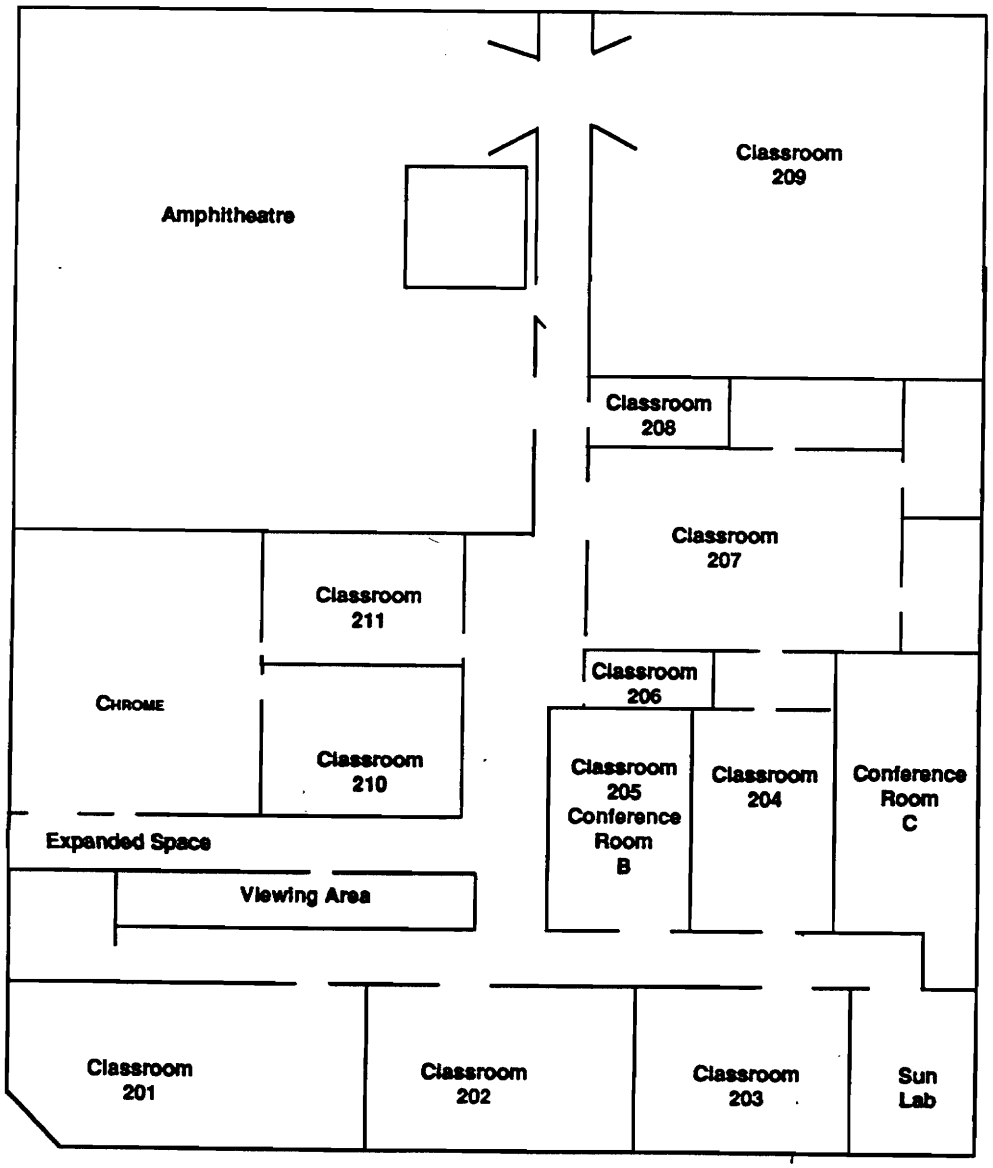



FIGURE 8. EXPANDED SPACE

TABLE 2. PENINSULA CENTER CLASSROOMS

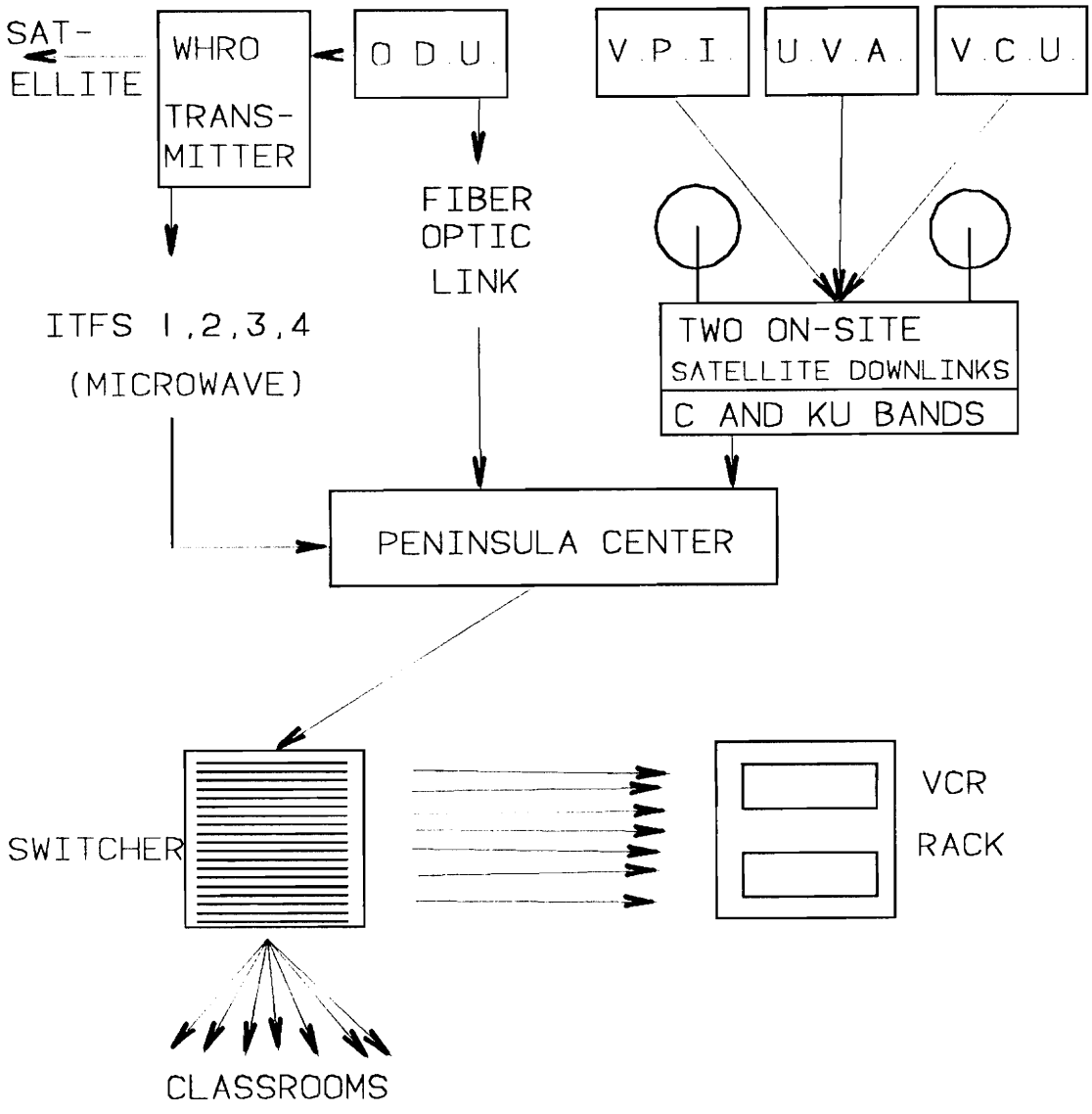
| ROOM I.D. | ROOM TYPE | NO. OF SEATS |
|---|---|---|
| 101 | TELEVISION | 8 |
| 103 | BOTH | 40 |
| 104 | TELEVISION | 16 |
| 105 | TELEVISION | 16 |
| 106 | BOTH | 30 |
| 107 | BOTH | 18 |
| 108 | BOTH | 18 |
| 109 | BROADCAST | 80 |
| 110 | TELEVISION | 10 |
| 201 | LIVE | 20 |
| 202 | LIVE | 20 |
| 203 | LIVE | 26 |
| 204 | LIVE | 14 |
| 205 | LIVE | 20 |
| 206 | LIVE | 6 |
| 207 | LIVE | 24 |
| 208 | LIVE | 8 |
| 209 | LIVE | 40 |
| 210 | LIVE | 16 |
| 211 | LIVE | 6 |
| 212 | BROADCAST | 120 |
| COMP | LIVE | 15 |
| CONF | LIVE | 20 |

(3) "Both" Classroom - These rooms are capable of live and television instruction. They are equipped with the required hardware to support both types of instruction.

(4) "Broadcast" Classroom - These rooms are capable of broadcasting lessons to other sites. A studio type environment is necessary to support this service. This includes an instructor table with monitor, student microphones, cameras, wiring/cable to support these and access to a control room that is occupied by an Audio Visual Technician. The technician changes camera angle, camera direction and microphone volume at the discretion of the instructor.

As the funding becomes available, the intention is to supply the hardware and cable necessary for television instruction to all the rooms at the Center. This will allow maximum flexibility for the staff since all rooms would be available for either television or live instruction.

The equipment needed to provide the televised class service to the Center is quite extensive. This includes the satellite links, fiber optic links, antennae, switchers, etc. that get the signal to the classrooms. Figure 9 shows the equipment and connections present at the Center. From this arrangement, it is possible to gain a better understanding of the components essential in the televised




EACH CLASSROOM IS EQUIPPED WITH

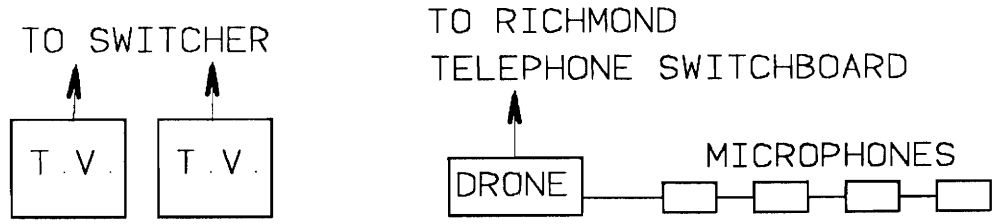



FIGURE 9. EQUIPMENT ARRANGEMENT

instruction system. In this system, there are constraints that limit the amount of televised classes received by the Center at any one time. This configuration can be divided into three parts:

(1) Origination process - The receiving of signals from other sources is the purpose of this process. As can be seen, the two satellite dishes at the Center can receive signals from VPI, UVA and VCU. Currently, these schools operate on C band. Since they may operate on two bands (C and Ku) at a later date, two separate classes can be received from each of the three schools for a total of six. The Center can receive four separate classes from ODU via microwave ITFS stations 1 through 4. The classes are transmitted from ODU through the local television station (WHRO) and received at the P.C. using a microwave tower. Also, a direct fiber optic link from ODU allows the Center to receive an additional televised class. The cable for the link is run between the two sites through the Hampton Roads Bridge Tunnel. There are a total of eleven sources that are available with this configuration.

(2) Receiving Process - The main component of this process is the switcher. The function of the switcher is to receive the signals from the originating sites and then direct them to the appropriate classes. There are various

sizes of switchers which determine the amount of input signals and output signals that can be processed at any one time. Since all classes that are televised must also be recorded on VHS tapes, the switcher acts as the distributor that sends the signal to the bank of video recorders. For every class signal that is received by the switcher, one line must be dedicated to the video recorder and another to the televised classroom that is displaying it. Therefore, the switcher must be capable of outputting twice as many lines as it is inputting in order to utilize its potential. The current switcher being used at the Center is rated as (10*24)*2. This means that it is capable of receiving 10 input lines and transmitting 24 output lines. All televised classes must be recorded for future viewing use of the students. At the present time, there are only seven video recorders present in the system. Therefore, this is the maximum number of classes that can be recorded at one time.

(3) Distribution Process - This consists of the hardware components necessary to view the televised classes in each classroom. For each room that is capable of televised instruction, there are at least two televisions for video support. Audio support consists of one Drone unit and several student microphones (usually one for every two seats in the room) that are connected via phone link dialed

into the Richmond, Virginia switchboard. Currently, the Center has ten rooms equipped for televised instruction. As stated earlier, there are plans to make all rooms capable of televised instruction.

With this physical configuration, it is easy to determine the maximum number of televised classes that can be received simultaneously. Figure 10 shows the equipment in block diagram form to illustrate the governing constraints in this system. As can be seen, the number of video recorders limits this amount to seven. Until the recent expansion, it was not considered feasible to increase this capability because the class schedule for live and televised instruction made it difficult to accommodate more classes. In other words, the number of rooms was the limiting criterion in the televised system. With the expansion to double the amount of rooms, it is important to note that the purchase of three more video recorders would increase the televised capacity to ten at any one time. The increased capacity will make more difficult an already complicated scheduling problem for the staff but will improve the Center's resource utilization.

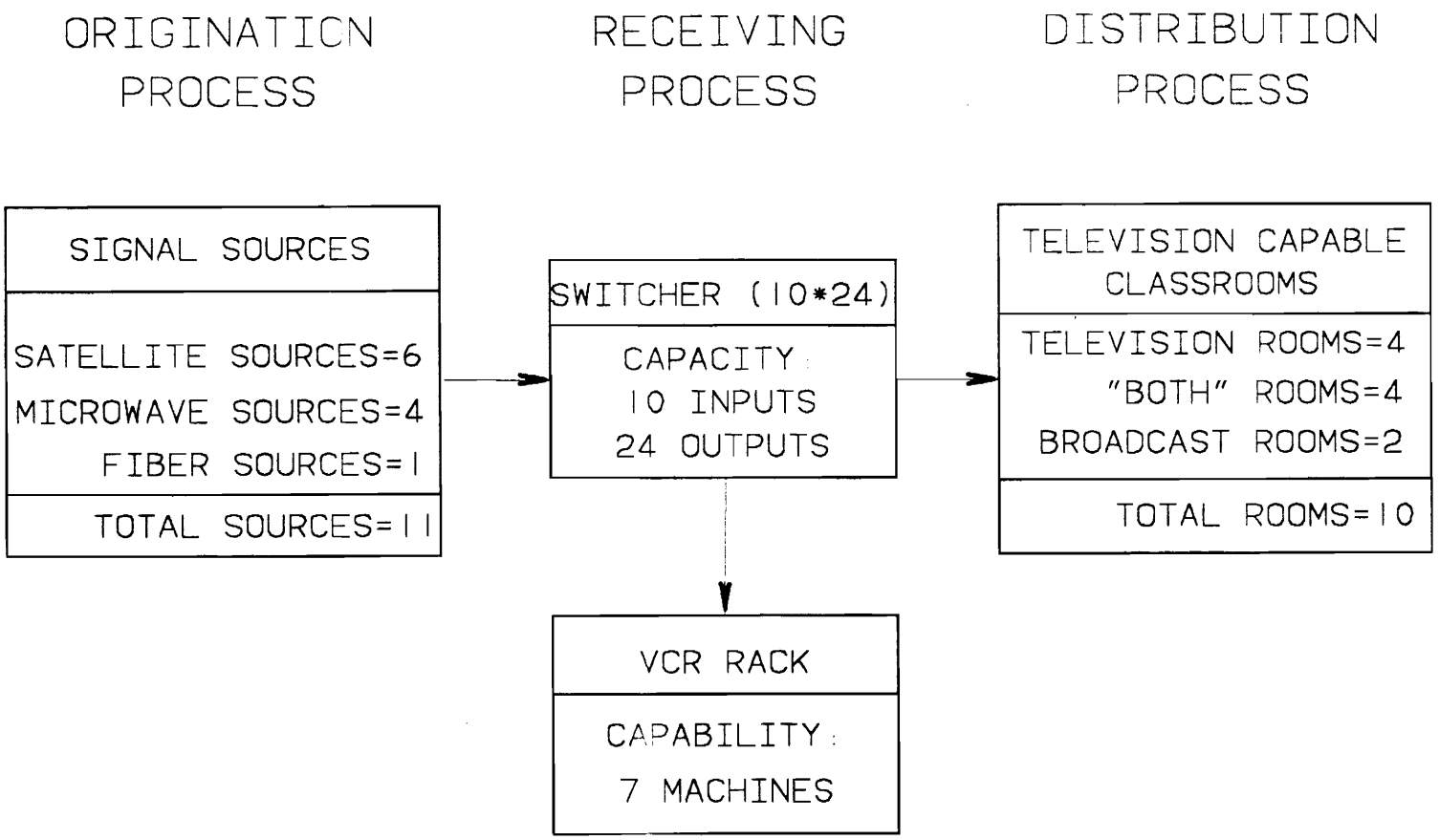



FIGURE 10. TELEVISED CLASS EQUIPMENT CONSTRAINTS

## Staff and Services

Although the Center is an extension of ODU, it functions as a separate learning center from the host university. It is responsible for every task from registering students at the semester start to administering and proctoring examinations. It supports the needs of the students as well as firms and organizations for live training seminars, teleconferences, and broadcast productions. Figure 11 shows the staff organization. It should be noted that during the first month of the semester start, when activities are the most hectic, it is difficult to invest time in determining the class schedule. The determination of a working class schedule is very labor intensive since the enrollment during this time changes quite frequently.

## Semester Activities

For a given semester, the classes offered at the Center have been determined by the O.D.U. scheduling staff months prior to the semester start. The determination of classes offered is based upon considerations such as room availability, class demand, and hardware support. As described above, classes can be taught one of three ways - (1) Live, (2)

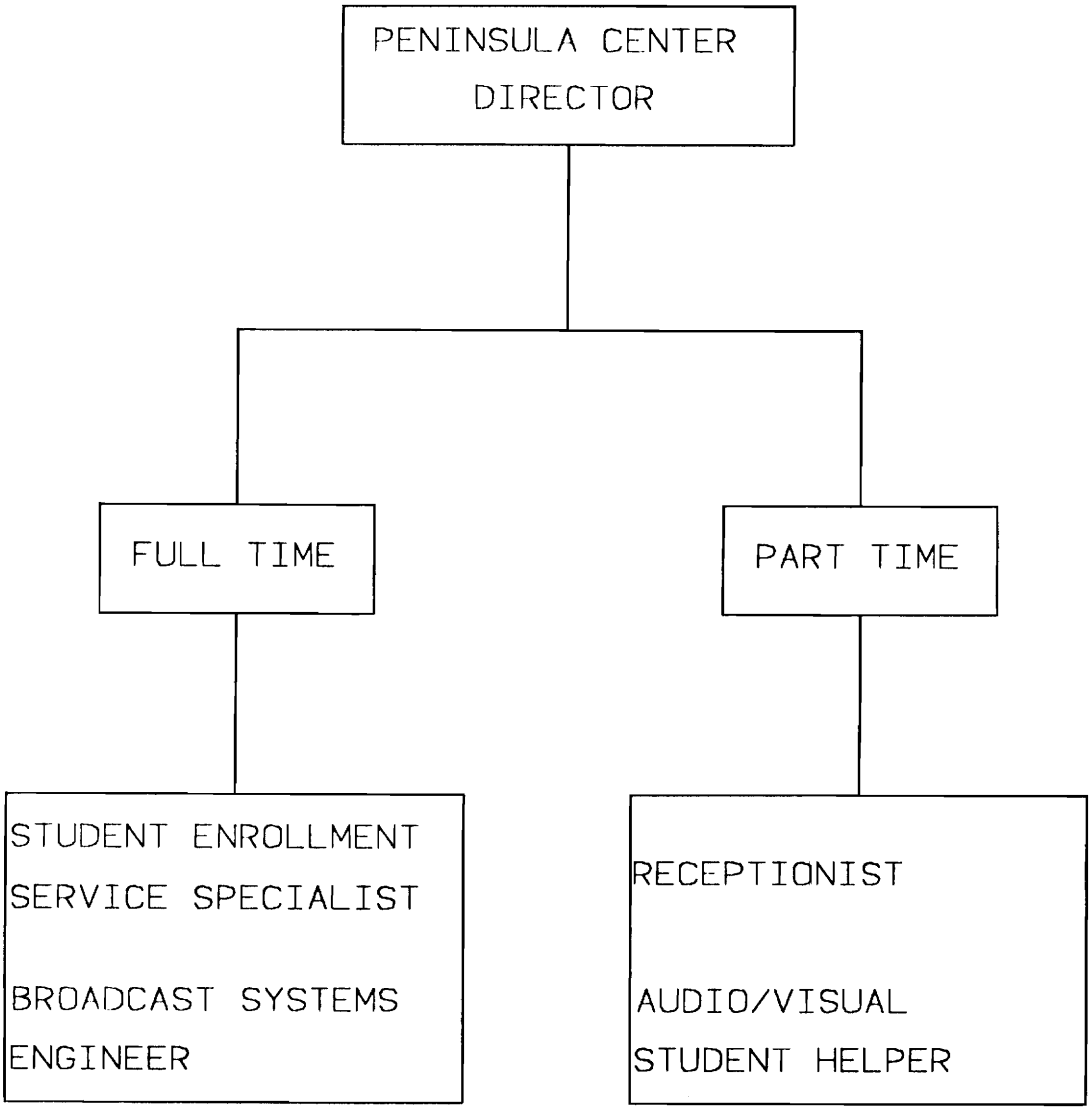



FIGURE II. STAFF ORGANIZATION

Televised and (3) Broadcast. Occasionally, classes are added to the agenda at the last minute due to unexpected demand or instructor availability. Conversely, some are stricken from the list due to lack of interest. These factors add to the complication of class scheduling.

The activities at the Center are shown in Figure 12. This depicts a typical semester and should illustrate the peak periods of activity for the staff. In 1986, when student enrollment for a given semester was only seventy five, the staff size described above could easily perform its duties and still make the course-classroom assignments. As can be seen from Table 1, the enrollment has increased to nearly ten times the original amount. This has made it more difficult for the staff to perform its functions and find time to create the class schedule. An automated class scheduler would release at least one person from this task to perform other functions.

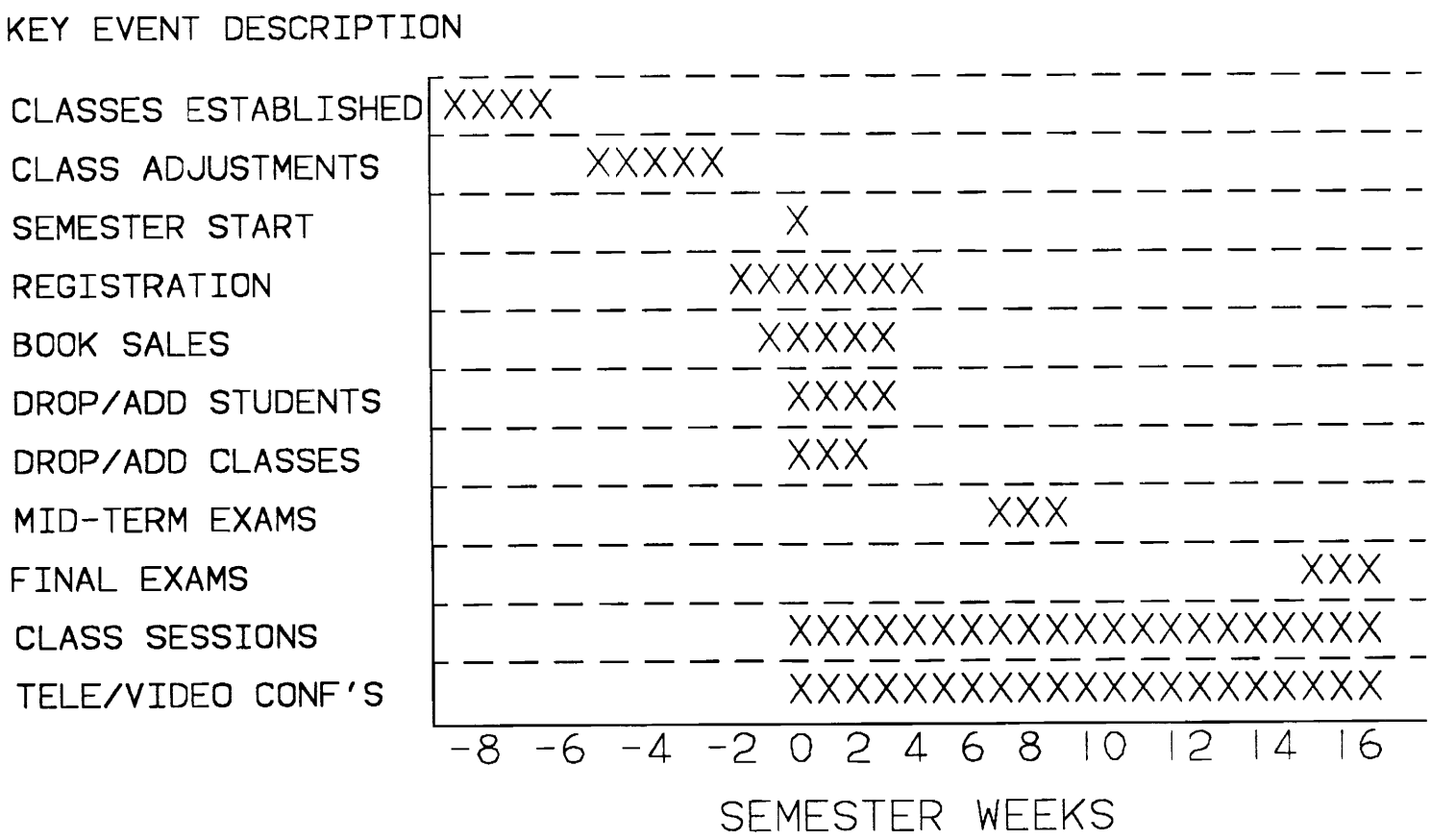



FIGURE 12. SEMESTER ACTIVITIES

## Chapter 3

### LIFE-CYCLE OF THE P.C. SCHEDULER

The following discussion provides an overview of the life-cycle approach that was employed during the Peninsula Center Scheduler design process. This approach considers the planning, design, implementation and maintenance phases of the project. The discussion highlights the systems engineering functions utilized in addressing the scheduling problem.

In the Fall of 1991, the decision to expand current operations at the Center was made by the state of Virginia and ODU. The Center had been surpassing all expectations of success and it was time to broaden the curricula. This brought many challenges to the Center's staff. The university's intention was to have the expanded space available by the Fall 1992 semester.

During several meetings with the Center's staff, the Director was reviewing current operations to determine how to adapt in order to meet the expansion challenges. One problem became quite evident - that of class scheduling. The expected class load and student enrollment would create a scheduling situation very difficult to solve manually.

For the following reasons, the need for an automated class scheduler was realized:

(1) The number of classes and curricula were being doubled.

(2) The class sizes were expected to increase. This is evidenced by the projections shown in Figure 5.

(3) With the increase in student enrollment, the other duties for the Center's staff would become more extensive.

(4) There were no plans to increase the staff size.

Having an automated class scheduler would free up one staff member that would otherwise be occupied with scheduling duties. This would considerably reduce the workload during the busiest portion of the semester. With this identification of the need, a process to understand the scheduling system at the Center was begun. Figure 13 is an overview of the functions in the process that were implemented in determining this system. Addressing each of these functions would help to ensure that all aspects of the scheduling process were being considered [4]. Throughout the 1992 year, frequent meetings with the Center's staff were held to ensure constant communication during the project's progression. The functions of the life-cycle activities are detailed below.

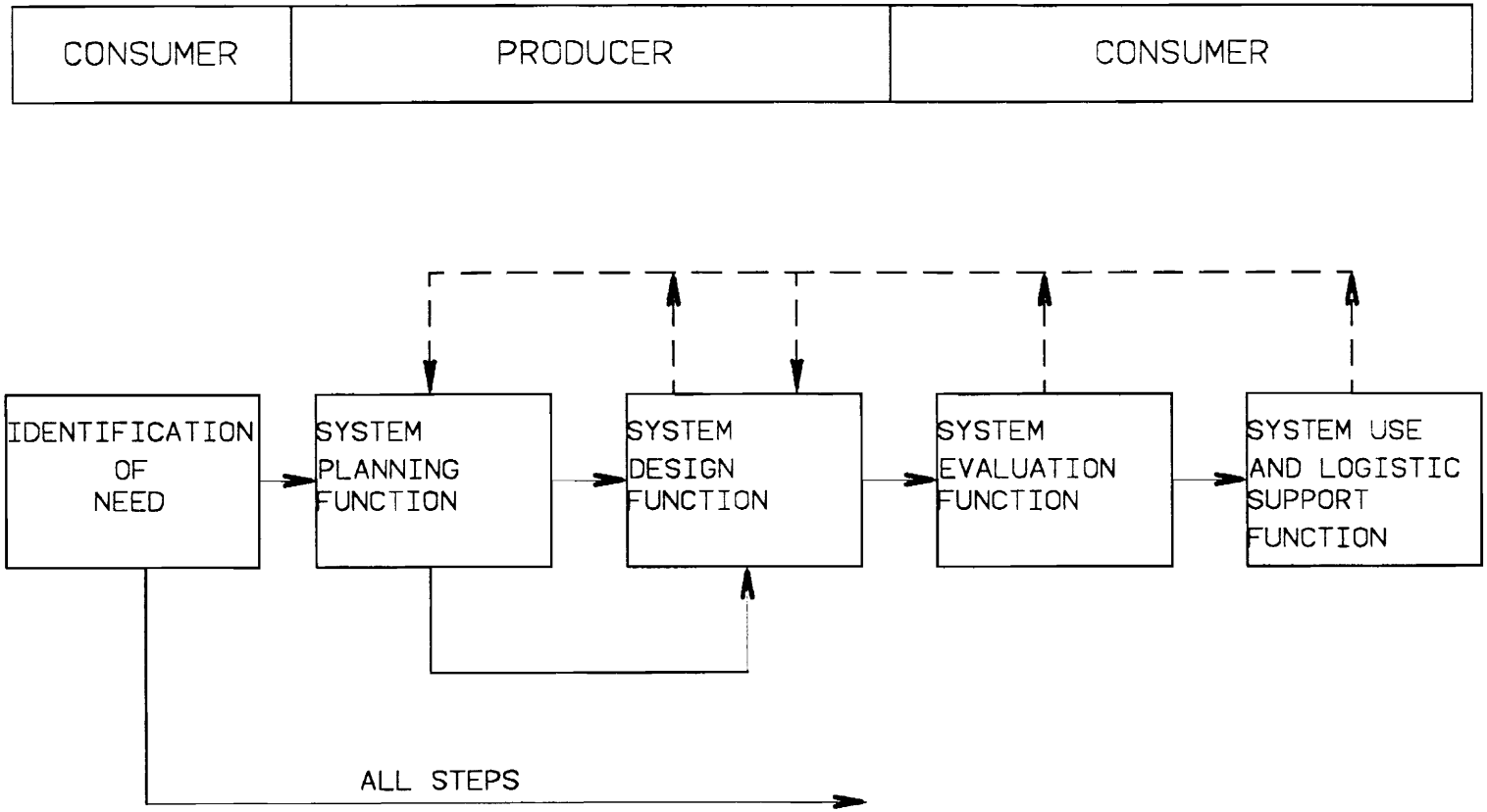



FIGURE 13. SYSTEM LIFE-CYCLE FUNCTIONS

Statement of Need

This was based upon past performance and the growth expected to occur in the Fall semester of 1992. Even more importantly, the projected increase in student enrollment was a large determinant. As discussed in more detail in Chapter 1, the automated scheduler would not only meet the needs based on past enrollments but it would meet the anticipated needs based on enrollments projected in Figure 5. It was determined that an increase to the staff personnel was not an option so the need for an automated class scheduler was recognized.

System Planning Function

The system parameters and its performance requirements were established during the planning stage of the project. The decision was made to implement a software application to perform the scheduling operation. In order to determine which would be the most appropriate for this project, it was necessary to address many questions such as the following:

- What type of software would accomplish the desired goal in the most efficient manner?
- What resources are available?

- Which would be the easiest to implement?
- How flexible does the software need to be in order to perform adequately with changes in the system?
- Which personnel would be expected to use the software?

The first step of this phase was to establish the goals of the software application. The main purpose was to find a suitable classroom match for each class being offered without resulting in conflicts. It should perform the same tasks that are currently being done manually. These considerations and constraints are explained in detail below in the System Design Function section. There were two approaches that could have been followed for determining classroom assignments:

(1) Fit each class roll in the largest classroom that is available. This would provide the students the maximum space available.

(2) Fit each class roll in the smallest classroom that is available. This would allow for maximum utilization of seating capacity.

The decision was made to use the second approach because the staff was interested in determining the most efficient room allocation. With this type of approach, the

staff could use the results to determine not only their present scheduling parameters but it would serve as a tool in determining the expansion possibilities for the future semesters. One critical purpose was to enable the director to make scheduling decisions months ahead of the semester start. "What if" questions could be asked to plan for the maximum class load capable of being supported at the Center. This was very appealing because it provided a way of maximizing the use of the Center's resources. In support of this, the following output was desired:

(1) A schedule of each day's classes.

(2) A schedule of each room's classes.

(3) A summary showing totals and statistics for the Center.

A planning schedule was developed to identify key events in the program development project. As shown in Figure 14, the implementation of the scheduler was planned for the Fall 1992 semester. It was necessary to allow sufficient time for testing, evaluation and modifications of the program as well as providing instruction to the staff to familiarize them with the program.

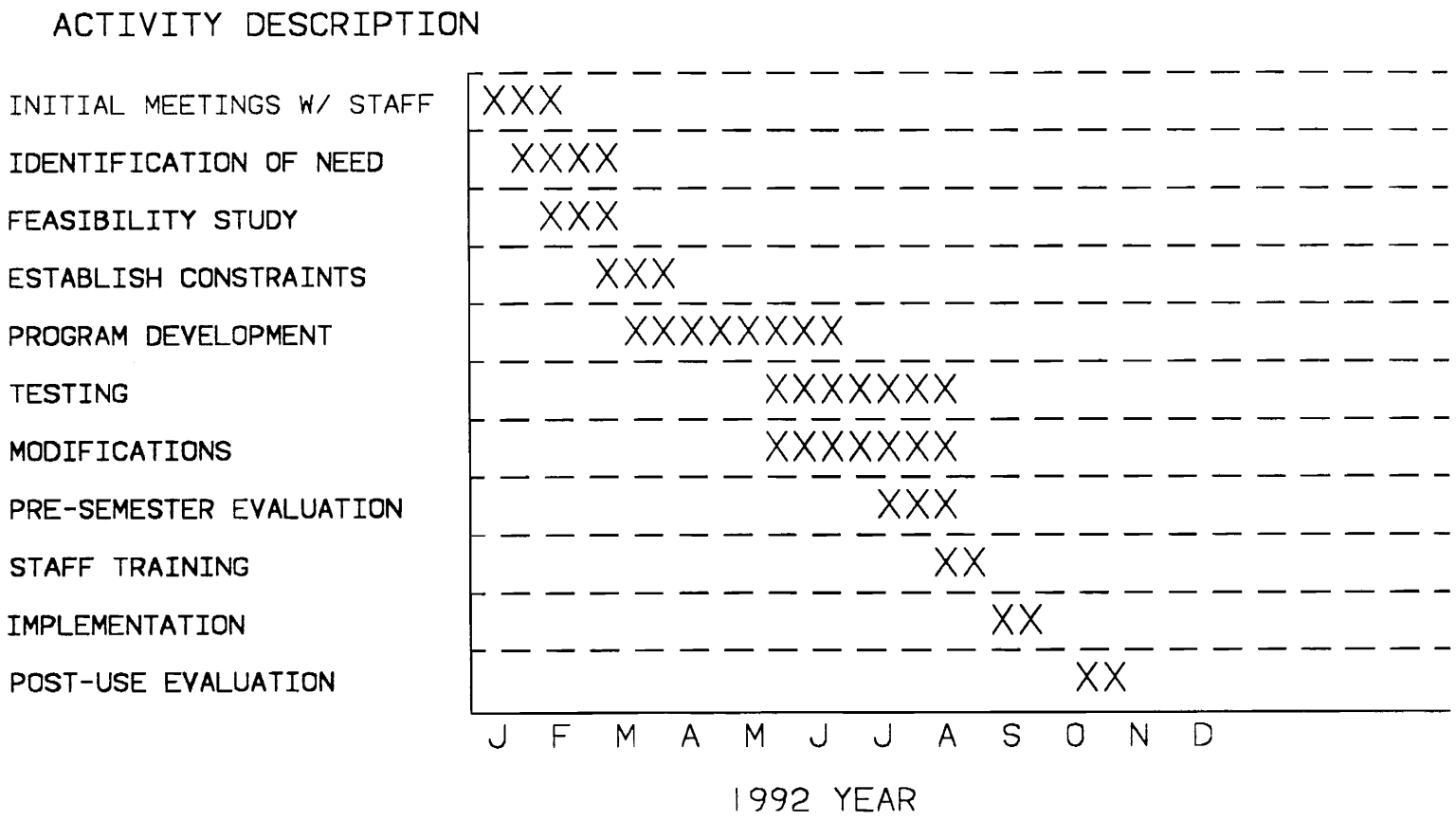



FIGURE 14. SCHEDULE OF PROJECT ACTIVITIES

There were several system performance requirements that had to be established. A feasibility study was performed to determine the most suitable alternative for this application. The Analytic Hierarchy Process (AHP) of pairwise comparisons and attributes [5] was used in the analysis. The general methodology of the AHP is to break down the problem and to make pairwise comparisons of all elements(attributes, alternatives, etc.). The decision maker(s) (in this case, the Center's staff members) evaluated each entry by quantifying their preferences. The set of attributes considered was a the result of several discussions amongst the staff members. In their opinion, these were the most important aspects to consider. A scale of 1 to 10 was used for the attribute weights and the scoring of the alternatives. These values were then placed in a matrix to determine the following:

(1) The relative importance of the attributes.

(2) The relative weight of each alternative with respect to each attribute.

(3) The overall priority weight of each alternative.

In each case during the evaluation, the score for each attribute (or alternative) was divided by the total for all attributes (or alternatives) to arrive at the resultant

priority weight. Figures 15 and 16 show this study in block diagram form and the rating of alternatives. Table 3 provides the summary of alternative scores. Cost, as well as qualitative attributes including flexibility, maintainability, and ease of use were considered in the selection of the scheduling software.

It was decided to develop the Center's scheduling software using the programming language FORTRAN 77. This decision was made because it provided maximum flexibility since the program could be written to accomplish the same scheduling tasks manually performed by the staff. FORTRAN 77 was the choice because the Center personnel were most familiar with this language and this would allow the program to be updated and maintained on site during the course of its use. Also, it was already available at the site so no investment in software was necessary.

There were several other aspects of the system planning function that were considered:

(1) The scheduler would need to be accessed by all staff personnel.

(2) A quick turn-around of results would be necessary in order to provide the staff with scheduling

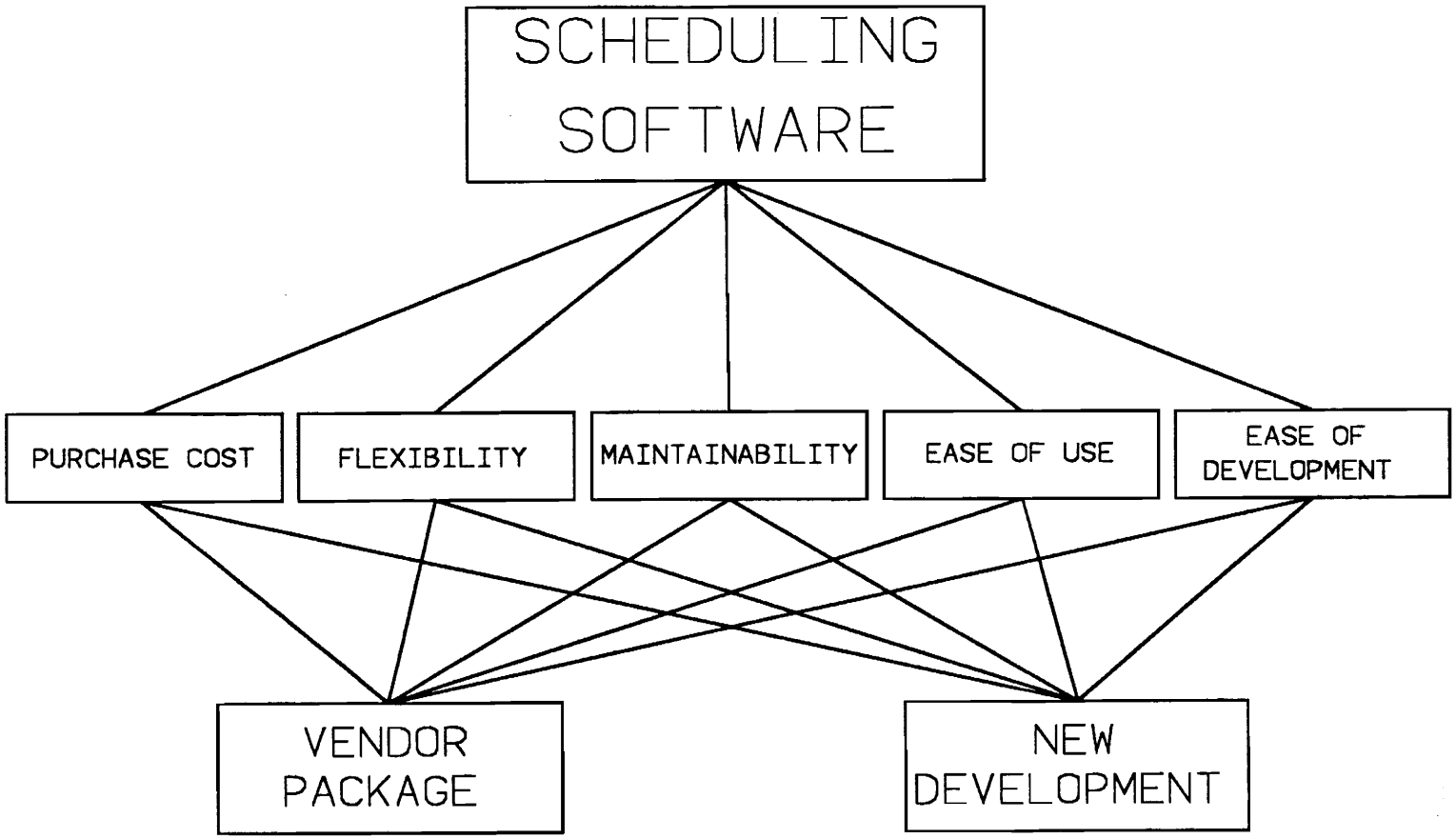



FIGURE 15. FEASIBILITY STUDY

WEIGHTING OF ATTRIBUTES

| Level with Respect to: | Purchase Cost | Flex-ibility | Maintain-ability | Ease of Use | Ease of Dev. | Priority Weight |
|---|---|---|---|---|---|---|
| Purch. Cost | 1 | 1/2 | 1/2 | 1/3 | 2 | 0.13 |
| Flexibility | 2 | 1 | 1 | 1/3 | 2 | 0.22 |
| Maintain. | 2 | 1 | 1 | 1/2 | 3 | 0.22 |
| Ease of Use | 3 | 3 | 2 | 1 | 3 | 0.35 |
| Ease of Dev. | 1/2 | 1/3 | 1/3 | 1/3 | 1 | 0.08 |
| Total | 8.5 | 5.83 | 4.83 | 4.50 | 12 | 1.00 |

SCORING OF ALTERNATIVES
Vendor Package = V.P.
New Development = N.D.

Purchase Cost:

| | V.P. | N.D. | Priority Weight |
|---|---|---|---|
| V.P. | 1 | 0.1 | 0.09 |
| N.D. | 10 | 1 | 0.91 |
| | 11 | 1.1 | 1.00 |

Flexibility:

| | V.P. | N.D. | Priority Weight |
|---|---|---|---|
| V.P. | 1 | 0.3 | 0.23 |
| N.D. | 3.3 | 1 | 0.77 |
| | 4.3 | 1.3 | 1.00 |

Maintainability:

| | V.P. | N.D. | Priority Weight |
|---|---|---|---|
| V.P. | 1 | 0.3 | 0.23 |
| N.D. | 3.3 | 1 | 0.77 |
| | 4.3 | 1.3 | 1.00 |

Ease of Use:

| | V.P. | N.D. | Priority Weight |
|---|---|---|---|
| V.P. | 1 | 0.4 | 0.29 |
| N.D. | 2.5 | 1 | 0.71 |
| | 3.5 | 1.4 | 1.00 |

Ease of Development:

| | V.P. | N.D. | Priority Weight |
|---|---|---|---|
| V.P. | 1 | 10 | 0.91 |
| N.D. | 0.1 | 1 | 0.09 |
| | 1.1 | 11 | 1.00 |

FIGURE 16. FEASIBILITY STUDY - ANALYTIC HIERARCHY PROCESS

TABLE 3. SUMMARY OF ALTERNATIVE SCORES

| DESCRIPTION | PURCHASE COST | FLEXIBILITY | MAINTAIN-ABILITY | EASE OF USE | EASE OF DEVELOPMENT | PRIORITY WEIGHT |
|---|---|---|---|---|---|---|
| ATTRIBUTE WEIGHT | 0.13 | 0.22 | 0.22 | 0.35 | 0.08 | |
| VENDOR PACKAGE | 0.09 | 0.23 | 0.23 | 0.29 | 0.91 | 0.29 |
| NEW DEVELOPMENT | 0.91 | 0.77 | 0.77 | 0.71 | 0.09 | 0.71 |
| SINCE THE WEIGHTED EVALUATION OF THE NEW DEVELOPMENT IS 0.71 AND THE VENDOR PACKAGE IS 0.29, THE NEW DEVELOPMENT ALTERNATIVE SHOULD BE CHOSEN. | | | | | | 1.00 |

information based on the latest enrollment figures and room availabilities.

(3) The scheduler interface would have to be user-friendly. In order for the scheduler to be successful, it was essential that it be easy to operate so that all staff personnel would feel comfortable using it. For these reasons, it was decided to install the package in the Center's computer laboratory on a 386 WIN workstation. The user interface is a menu system that provides an easy method for the user to access all information and perform the tasks (i.e. editing input files, running the program, printing the output reports, etc.).

The menu system as well as program organization are described in Chapter 4, **THE COURSE-CLASSROOM ASSIGNMENT PROCESS.**

System Design Function

With the system performance requirements defined and the planning schedule established, the system planning activities had been addressed. The next step was to understand the constraints and parameters of the existing system. It was necessary to establish these criteria for

the design of the software system. For this, it was necessary to meet with the two staff members who were most familiar with the existing scheduling policy at the Center. While done manually, consideration was being given to aspects of the scheduling problem only understood by the staff. During several discussions with the Broadcast Systems Engineer and the Student Enrollment Services Specialist several scheduling constraints were identified. It was agreed that the automated scheduler should address these same issues. These are listed in Table 4 and are explained in detail below:

(1) Room and Class types - As explained earlier, there are three types of class instruction offered at the Center: live, televised and broadcast. Correspondingly, there are three room types available with the addition of a fourth: rooms that can accommodate both televised and live instruction. They are equipped with a blackboard & instructor's table (to support live instruction) and televisions & microphone units (to support televised instruction). These rooms are referred to as "both" type rooms. The scheduler must determine the class type then find a room type that is compatible and large enough to fit the class enrollment. Current scheduling policy at the Center dictates that the live and televised classrooms will be

TABLE 4. SCHEDULING CONSTRAINTS

| CONSTRAINT | DESCRIPTION |
|---|---|
| 1. ROOM AND CLASS TYPE | ROOM ASSIGNMENT MUST BE COMPATIBLE WITH CLASS TYPE. |
| 2. NON-LOCAL SCHOOL ASSIGNMENTS | ALL CLASSES TAUGHT FROM A NON-LOCAL SCHOOL ON A GIVEN DAY MUST BE LOCATED IN THE SAME ROOM. |
| 3. STUDENT CONVENIENCE | ALL SESSIONS OF MULTI-DAY CLASSES MUST BE TAUGHT IN THE SAME ROOM. |
| 4. INCREASE IN ENROLLMENT | ROOM ASSIGNMENTS MUST ACCOUNT FOR INCREASES IN STUDENT ENROLLMENT IN ORDER TO AVOID CLASS SIZE EXCEEDING ROOM SIZE. |

utilized first, followed by the "both" rooms when only these are available. This is done to allow maximum flexibility during the semester in the event of equipment failure. For instance, if at least one "both" room is available at all times, a class that is experiencing equipment failure in its normal room can be relocated to the empty room and still receive the lesson.

(2) Non-local school classes - Classes being taught from VPI, UVA and VCU are considered non-local. Classes being taught from ODU are considered local. With the current method of scheduling, there is a fundamental difference in the selection process of rooms for these two class types. For the local classes, it is a straightforward process: determine the smallest room that will accommodate the class roll. Scheduling of the non-local classes, on the other hand, is approached in a different way. Because of the connection necessary to show these non-local schools' classes, the hardware link is different for each of them. As a result, each time a different school's class is taught in a room, the Broadcast Systems Engineer(BSE) must modify the connection at the room site. This is counterproductive for the staff because

there is excessive time involved in this task. It makes the job of supervising the televised and broadcast classes very difficult because it requires much time from the BSE who is usually the only person responsible for this job. This is especially true during the peak hours of the day (between 4:00 P.M. and 8:00 P.M.).

For these reasons, the BSE's efficiency is maximized when all same-school classes for these non-local schools are taught in the same room on any given day. This minimizes the amount of staff time dedicated towards "switching" connections among classrooms. In other words, all VPI classes should be scheduled in one room for a given day. UVA and VCU classes would be scheduled in the same manner.

In order to perform this type of "matching" the scheduler needs to do the following:

(a) Examine all non-local classes taught each day

(b) Search for the largest enrollment in this list

(c) Find a suitable room for this largest class

(d) Place all other same-school classes in this room.

This should be done for each of the non-local schools. Currently, there are three but as the Center expands, this may also be increased. As will be shown in Chapter 4, THE COURSE-CLASSROOM ASSIGNMENT PROCESS, this algorithm for searching, identifying and matching these classes is applied in the subprograms "MATCHUP" and "FINDROOM" in the P.C. Scheduler program.

(3) Student Convenience - Many classes are taught on multiple days. For instance, many three credit classes are offered two days a week for 1 1/4 hours per session. These multiple day classes currently are being scheduled in the same room for each day as a convenience to the students attending the class. A scheduler that only searches for the smallest room to fit each class would not necessarily locate two sessions of the same class in the same room. It is important that the scheduler make provisions to meet this requirement. As will be shown in Chapter 4, THE COURSE-CLASSROOM ASSIGNMENT PROCESS, the algorithm that ensures this type of class to room consistency is incorporated in the subprogram "FINDROOM".

(4) Increase in enrollment - Typically, class enrollments at the Center begin roughly two weeks prior

to the semester start and continuously change until two weeks after the semester start. Because of this, it is difficult to establish a "working" schedule of classes based on an initial enrollment status. When the scheduling is done manually, there are many iterations performed during that initial four week period because of increases in class enrollment. These changes in enrollment cause classes to "outgrow" the currently allocated rooms. When this occurs, the class must be relocated to a larger room. This is labor intensive because, in most cases, this affects other classes and causes potential conflicts which then have to be resolved by rescheduling.

When the Center first opened with twenty classes in 1986, this was a manageable problem. Now that the total class load has exceeded eighty, it is one of the biggest challenges that face the staff at the Center. The P.C. scheduler reduces the problems associated with enrollment changes during the semester start by addressing the increase in class size. The program allows the user (Center staff) to specify an expected increase in enrollment above the current status. This allows for class size expansion and results in a first-cut solution that withstands enrollment changes during the semester start. This is explained in

detail in Chapter 4, THE COURSE-CLASSROOM ASSIGNMENT PROCESS.

In order to facilitate the design process of the FORTRAN 77 software package itself, the organization of the program is formatted in a "modular" fashion. This is done by the use of subprograms called subroutines. This keeps the individual functions of the program separate so that they can be easily identified during the testing and evaluation phase. Also, this type of organization allows for easier modifications in the future because the programmer will be better able to determine the separate functions and operations.

System Evaluation Function

The system performance requirements have been identified and the current scheduling policy at the Center has been described. With the equipment and system constraints understood, it is possible to transform the current manual operation of scheduling into software. All parameters present in the physical system must be considered in the automated scheduler if it is to be successfully used by the Center's staff. As shown in Figure 14, the program

development phase of the scheduling project spanned from March to June of 1992.

During the development, it was necessary to conduct [illegible] with the staff in order to receive input to ensure that the software being created was conforming to their needs. This task was facilitated by the fact that the software was largely developed on-site. The WIN 386 workstation computer laboratory was used in this effort.

The System Evaluation Function [4] in this system's life-cycle process consists of the following activities:

(1) Identification of test and evaluation requirements- During the development of each of the program functions, the operations being performed were tested for the staff after which an evaluation was given to determine areas of improvement. It was decided to use the previous semester's class load and room resources during much of the testing phase. This helped to identify any weaknesses or strengths of the automated scheduler because it was compared to the manual operation that the staff was accustomed to working with during the past six years. Table 5 shows the functions

TABLE 5. PROGRAM EVALUATION CRITERIA

| PROGRAM FUNCTION | EVALUATION CRITERIA |
|---|---|
| 1. DATA INPUT | EASE OF DATA ENTRY<br>EASE OF ACCESS TO FILES |
| 2. PROGRAM RESULTS | EFFICIENCY OF ASSIGNMENTS<br>EASE OF USE<br>SATISFACTION OF PARAMETERS<br>TURN-AROUND TIME |
| 3. REPORT OUTPUT | REPORT FORMAT<br>REQUIRED INFORMATION PROVIDED<br>CLARITY OF REPORTS |

of the program as well as the criteria used by the staff in their evaluation.

(2) Determination of logistic support resources for test completion - Since the computer laboratory at the Center was the location of much of the program design, it was logical to use this as the testing area, as well. Although most of the staff members are experienced computer users, running the FORTRAN program became quite cumbersome without some type of user interface. One of the resources available was the Three Dimensional Menu system that is resident on the computers. With this serving as the user's window, it made the job of accessing files, running the program and reviewing output reports much easier. This was invaluable during the testing phase because the staff was able to do the testing with very little effort concentrated on file organization, command names, etc. This was found to be quite "user friendly" during the testing phase and was eventually implemented in the production model of the scheduler. This will be explained in more detail in Chapter 4, THE COURSE-CLASSROOM ASSIGNMENT PROCESS.

(3) Initiation of system modifications - As described above, the program design and development stage of the scheduler spanned four months. A broad framework for the program functions was established based on the initial problem description and as the testing process progressed, this framework was modified and improved to suit the staff's recommendations. These improvements were based upon room utilization, staff efficiency and student convenience.

System Use and Logistic Support Function

With the design and testing of the automated scheduler complete, the final phase of the system process was the incorporation of the program at the Center. The target program completion date of mid August was met. At this time, a formal demonstration was given to all staff members to instruct them in the use of the program. Features such as menu options, file management and output reports were explained. At that point, the semester class schedule was not final but it was used as input to the program.

During the first several weeks of the semester, the plan was to have the staff use the program and provide comments and suggestions based on their experience. A

follow-up meeting one month after the semester start was arranged to discuss their criticisms. Chapter 5, **RECOMMENDATIONS FOR PROCESS IMPROVEMENTS**, details the improvement suggestions that resulted during this semester's utilization of the program. The intention was to provide a forum for suggestions to the process so that improvements could be made which would facilitate scheduling of subsequent semesters. This feedback phase would exist for each semester during the lifetime of the Center [4]. This would be done to ensure that the program would undergo a constant process of change to coincide with the Center's progress over time.

This process of change with time was considered during the planning and design of the program. As the program is utilized from one semester to the next, it is almost certain that modifications will be necessary to ensure that it retains its usefulness. The future updates will be performed by the systems engineer in conjunction with the staff personnel. It was written in FORTRAN 77 because the staff felt most comfortable with this language. This, as well as the modular design of the program, will facilitate future modifications. The program is intended to be used for at least the next five year period at which time an evaluation will be done to consider other alternatives that

are currently not available. Figure 17 provides an outline of process improvement recommendations that resulted from the first of the follow-up meetings with the Center's staff.

## O *PRESENT OPERATIONS*

1. CLASS PRIORITIZING
2. IMPROVED OUTPUT
3. PROJECTIONS IN ENROLLMENT CHANGES

## O *FUTURE EXPANSION*

1. AUTOMATION OF CLASS INPUT
2. DETERMINATION OF MAXIMUM CLASS LOAD
3. IMPLEMENTATION AT OTHER SITES

FIGURE 17. RECOMMENDATIONS FOR PROCESS IMPROVEMENTS

## Chapter 4

## THE COURSE-CLASSROOM ASSIGNMENT PROCESS

In the preceding chapters, we have discussed the activities of the Peninsula Center as a process and have identified the need for an automated scheduler. Also, we have described the steps taken to understand the scheduling problem in order to design the software package that will most efficiently utilize the Center's resources while reducing the staff's time spent performing the job. The following discussion will explain the process of classroom assignments as seen by the staff and will outline the aspects of the process as performed by the program.

### Use of the Program

As described earlier, during the typical semester, the scheduler is used most frequently during the first three to four weeks. It begins once the class load has been established. The following information is known at this time:

(1) Number of rooms (and type) available.

(2) Number of seats available in each room.

(3) Number of classes (and type) being taught.

(4) Number of students enrolled in each class.

Inputting the above into the appropriate files is the first necessary step. These are explained in greater detail in the next sections. Of these listed, items three and four are the most dynamic and sometimes are not firmly established until two weeks after the semester start. [illegible] will not commit to classes until they've attended one or two sessions. Since no record of past semester's enrollments for individual classes was retained, it is difficult to analyze previous class rolls for those courses that had been repeatedly offered. As a result of this, and instructor availability, classes often either are cancelled or added. This causes the enrollment figures to vary during the period. When done manually, the Center's scheduling consisted of many iterations of attempts at "fitting" the current class load in the available rooms. As classes increased in size, this caused conflicts which resulted in relocation of classes to other rooms and ultimately caused additional conflicts. The expansion of the Center meant further difficulty in this already unmanageable task.

With the P.C. scheduler, the staff member needing to determine the most efficient course to classroom assignment would access the input file and update it for the current class information. Once accomplished, the program can be

run and output reports can be printed to show the resulting assignments. In the Fall 1992 semester, in order to remain current with the enrollment figures, the program was utilized an average of five times daily during the first two weeks. This is an indication of the dynamic nature of the class enrollment process.

Explanation of Scheduling Algorithm

Much work has been done in the field of institutional scheduling. Approaches such as Linear Programming, Heuristics and Simulation are just a few that have been used to describe the problem at hand. Not only are these methods attempted for course scheduling but also they are used for examination timetabling. Below are examples of some constraints that have to be met in the bigger scheduling problems [6, 7]:

(1) Limit on the number of students and/or examinations during any one period.

(2) Nonconsecutive examination constraints (i.e., no examinations in succession for any student).

(3) Preassignments (i.e., certain courses/examinations are preassigned to specific periods).

(4) Room capacity constraints.

(5) Exclusions and time preferences (i.e., certain examinations are excluded from particular periods).

While these methods are certainly valuable in their applications, the algorithms employed in the P.C. scheduler use a different approach. Due to the fact that the class sessions and times for all courses taught at the Center are predetermined by the participating schools and because of the constraints present in this system, a sorting algorithm is used. These constraints, as shown earlier in Table 4, define the limitations present in the system. This method is explained below and presented in the pseudocode of Figure 18.

Classes are assigned to the smallest room that will fit the student enrollment for each class. The two subprograms which contain the algorithms that perform the search and match the best room for the class are "MATCHUP" and "FINDROOM". The pseudocode in Figure 18 will attempt to explain this process.

First of all, the classroom and class schedule data are located in input files "ROOMS.DAT" and "CLASSES.DAT", respectively. Each line in the files consist of information that describes the attributes of each entry. In the case of

```
FIRST CALL OF "MATCHUP" finds rooms for all non-local classes.
**Non-local classes are taught from V.P.I., U.V.A. and V.C.U.**
  LOOP through classes from largest to smallest.
    IF class is non-local AND not assigned to a room THEN
      CALL "FINDROOM"
        LOOP through rooms from smallest to largest
          IF room is large enough to accomodate students AND
            room is a compatible type AND it does not
            conflict (both day and time) with existing classes
            in the room, then assign the class to this room.
          END IF;
        END LOOP;
      END CALL;
      LOOP through all classes smaller than the class that was
      just assigned a room.
        IF this class (class 2) is from the same school as that
          which was assigned a room (class 1) AND it has not been
          assigned a room THEN
            LOOP through each day of P.G.E.C. class week.
              IF class 2 is taught on the same day as class 1,
                assign class 2 to the same room as class 1. This
                is done to reduce hardware hookups for televised
                classes. In other words, all V.P.I. classes will
                be taught in the same room. This eliminates the
                need for the technicians to switch channels among
                rooms. This is only a consideration for non-local
                classes and is therefore not necessary for O.D.U.
                classes.
              END IF;
            END LOOP;
        END IF;
      END LOOP;
    END IF; (Since the remainder of "MATCHUP" is only invoked
             during the second CALL, this is effectively an end
             to the IF statement during the first loop)
  END LOOP;
END FIRST CALL OF "MATCHUP";

SECOND CALL OF "MATCHUP" finds rooms for all local classes.
**Local classes are taught from O.D.U.**
  LOOP through classes from smallest to largest.
    IF class is local AND not assigned to a room THEN
      CALL "FINDROOM"
        LOOP through rooms from smallest to largest
          IF room is large enough to accomodate students AND
            room is a compatible type AND it does not
            conflict (both day and time) with existing classes
            in the room, then assign the class to this room.
          END IF;
        END LOOP;
      END CALL;
    END IF;
  END SECOND CALL OF "MATCHUP";
```

FIGURE 18. PSEUDOCODE OF SORTING ALGORITHM

file "ROOMS.DAT", there are 23 line entries describing each of the rooms at the Center. In the file "CLASSES.DAT", there are 82 entries describing each of the classes offered at the Center.

These files are input by the user as the schedules are being updated prior to the start of the semester. Therefore, they are not sorted in any way. In order to use the searching algorithm in "MATCHUP" and "FINDROOM", these need to be sorted. Subprogram "SEATSORT" sorts the room entries by seating capacity (ascending order). Subprogram "STUDSORT" sorts the class schedule entries by student enrollment in each class (ascending order).

This process most closely emulates the manual process of classroom scheduling. With this algorithm, the classes are assigned to the smallest room available thereby maximizing seat utilization. Currently, the results are based on this criterion. This was desired by the Center director in order to determine maximum capability. It is recognized that this method of assignments does not guarantee the optimum solution but it does provide a feasible allocation of classes to rooms in a systematic way that is much more efficient than the previously existing manual method.

It should be noted that the program is designed to allow the user to control the occupancy ratio of seats to students during the program run. As will be seen later this chapter, the user can specify an additional percentage of student enrollment for each existing class roll. This was primarily established to compensate for enrollment increases during the semester but it can be used to assign rooms based on a desired occupancy ratio. In other words, if the desired occupancy ratio for each existing class is 80 %, the user would specify an additional 25 % enrollment to satisfy this requirement. As will be discussed in Chapter 5, **RECOMMENDATIONS FOR PROCESS IMPROVEMENTS**, with this approach the staff of the Center as well as the host university, ODU, can make decisions concerning the increase of class offerings in the future.

As noted above, the "search and assign" algorithms are located in the subprograms "MATCHUP" and "FINDROOM". Figure 19 shows a flowchart of the FORTRAN 77 P.C. Scheduler Program. There are a total of twenty one subprogram modules in the program organization. As discussed earlier, the modular format was chosen for ease of identification and maintainability of the program. Figure 20 lists each subprogram and gives a description of its purpose. This

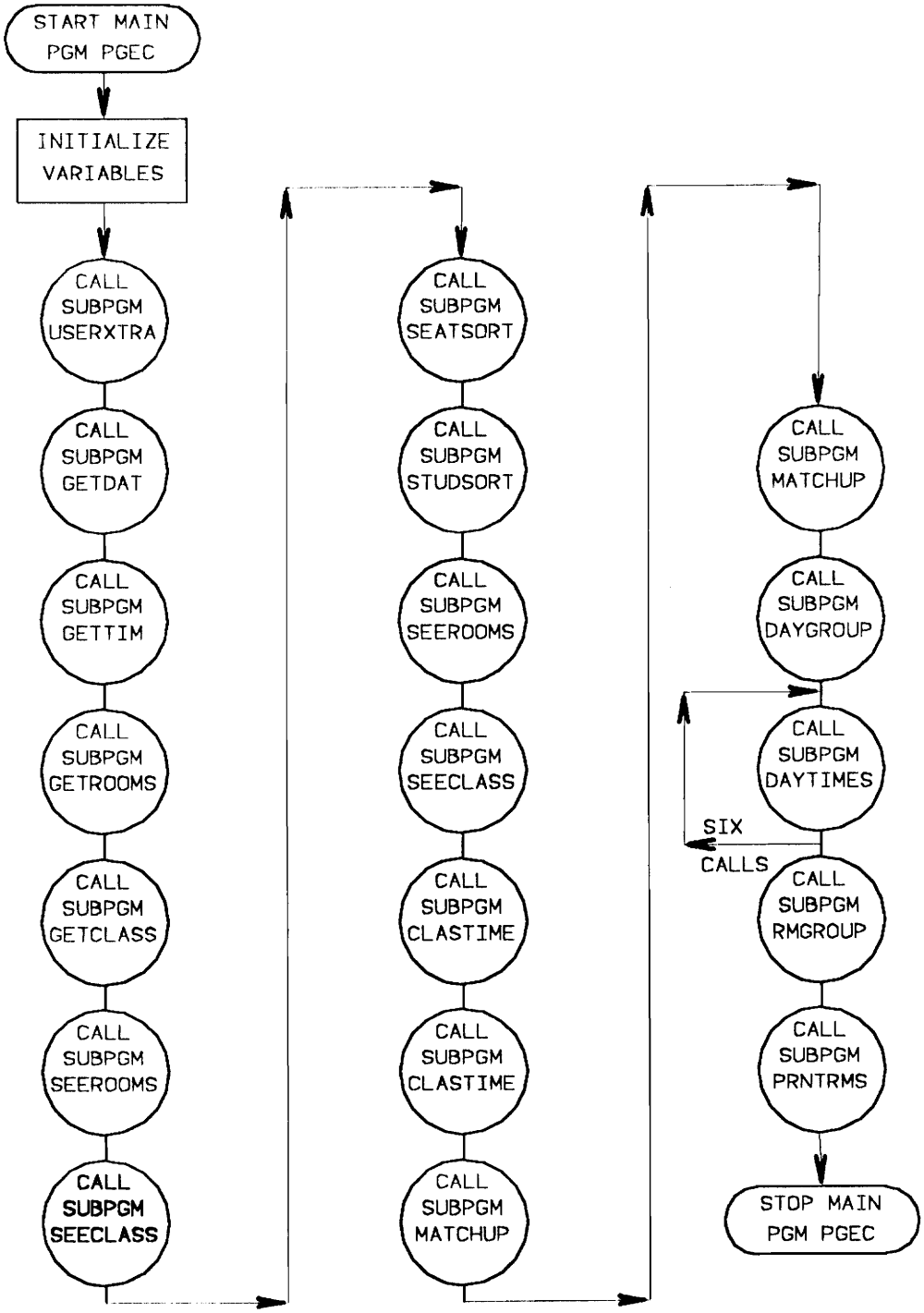



FIGURE 19. MAIN PROGRAM FLOWCHART

```
C
C  ------------------------------------------------------------------
C  ------------------------------------------------------------------
C
C  DESCRIPTION OF SUBPROGRAMS -
C
C  userxtra--PROMPTS USER TO INPUT METHOD OF ENROLLMENT INCREASE(IF DESIRED).
C  getrooms--READS IN ROOM INFO;WRITES ROOM INFO TO SUMMARY OUTPUT FILE.
C  getclass--READS IN CLASS INFO;WRITES CLASS INFO TO SUMMARY OUTPUT FILE.
C  seatsort--SORTS ROOM INFORMATION IN ASCENDING SEAT ORDER.
C  studsort--SORTS CLASS INFORMATION IN ASCENDING STUDENT ORDER.
C  swichi--SWITCHES INTEGER ARRAY ELEMENTS AS PART OF THE SORT.
C  swichc--SWITCHES CHARACTER ARRAY ELEMENTS AS PART OF THE SORT.
C  swichl--SWITCHES LOGICAL ARRAY ELEMENTS AS PART OF THE SORT.
C  clastime--CONVERTS CHARACTER EXPRESSION CLASS TIMES TO REAL NUMBERS.
C  seerooms--WRITES ROOM INFO TO DATA CHECK OUTPUT FILE.
C  seeclass--WRITES CLASS INFO TO DATA CHECK OUTPUT FILE.
C  matchup--FINDS LARGEST CLASS WITHOUT A ROOM AND THEN ASSIGNS
C            SAME-SCHOOL SAME-NIGHT CLASSES TO THE SAME ROOM.
C  findroom--DETERMINES THE SMALLEST ROOM THAT CAN ACCOMODATE A CLASS WITHOUT
C            CAUSING A CONFLICT WITH ALREADY ALLOCATED CLASSES.
C  daygroup--GROUPS CLASSES BY DAY.
C  daytimes--SORTS CLASSES FOR EACH DAY BY BEGINNING AND END TIMES.
C  rmgroup--GROUPS CLASSES BY ROOM THEN BY DAY.
C  rmtimes--SORTS CLASSES BY BEGINNING TIME WITHIN DAYS, WITHIN ROOM.
C  prntrms--WRITES CLASS SCHEDULE FOR EACH ROOM.
C  locbig1--FUNCTION THAT DETERMINES THE LARGEST INTEGER IN A 1-D ARRAY.
C  locbig2--FUNCTION THAT DETERMINES THE LARGEST INTEGER IN A 2-D ARRAY.
C  locbig3--FUNCTION THAT DETERMINES THE LARGEST INTEGER IN A 3-D ARRAY.
C
C  ------------------------------------------------------------------
C  ------------------------------------------------------------------
C
```

FIGURE 20. P.C. SCHEDULER SUBPROGRAM DESCRIPTION

same list is provided in the comment lines at the beginning of the program code. The complete set of flowcharts and FORTRAN 77 program source code are provided in Appendices A and B, respectively.

## File Organization

The P.C. Scheduler is designed to run on an IBM compatible Personal Computer. Currently used in the Center's computer laboratory are WIN 386 workstations. The program is installed on several stations to allow the staff access even during peak student usage times. In order to better understand the organization of the files used in the program, Figure 21 is provided to show the arrangement. Included in each block are the file purpose, file name and the resident DOS directory that stores it. Flow of data into the program as well as output from the program are shown for clarity.

## Input File Description

There is input from three sources required to run the scheduler. The user needs to ensure that the two data files reflect the current system status before running the program. These two files are:

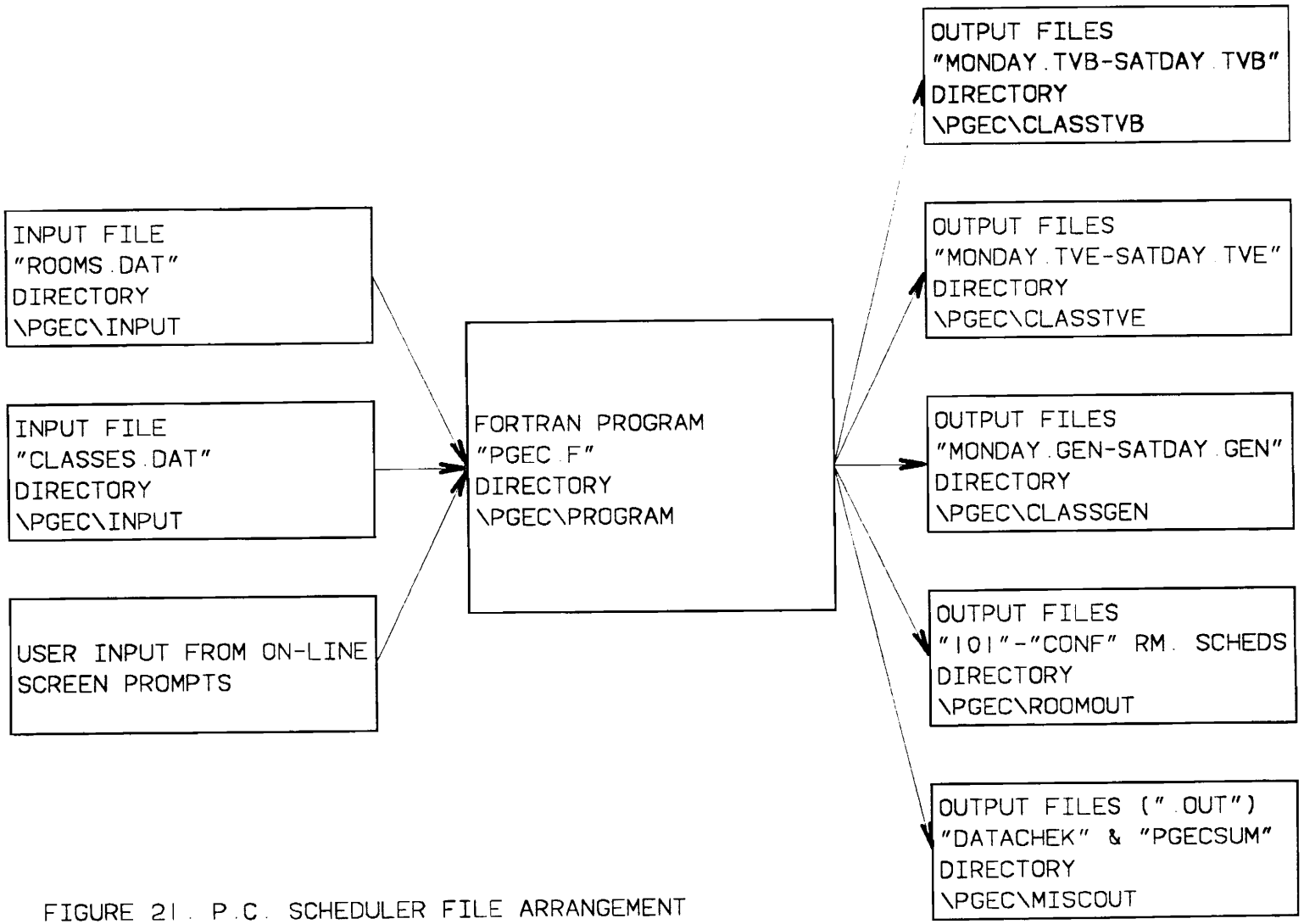



FIGURE 21. P.C. SCHEDULER FILE ARRANGEMENT

(1) ROOMS.DAT - Figure 22 shows an example of the classroom data file. The file shown reflects room data as of Fall 1992. As shown in this figure, the room identification, room type and room capacity (number of seats) are needed. Once established at the semester start, this file will not need to be frequently updated. There are instances when the user may need to modify the classroom information. For instance, at the beginning of the Fall 1992 semester, due to the expansion, some rooms were not complete. In this case, the room entry would remain in the file but a zero would be entered for seating capacity. This was the case for Room 212 since its broadcasting equipment was not ready for use.

(2) CLASSES.DAT - Since ODU is the host university for the Center's activities, they coordinate the list of classes offered. Figure 23 shows a typical class load listing for the semester. Once the Center is given this information, it is used for scheduling purposes. This class information is entered in the class schedule data file. Figure 24 shows an example of the data file. The file shown reflects the status of classes offered in the Fall of 1992. As shown in this figure, the following information is entered:

- class identification (by subject and course)

```
'101' 'TV'   8          * DATA FILE OF CLASSROOM INFORMATION:  *
'103' 'BO'  40          * FIRST ITEM IS ROOM NUMBER.           *
'104' 'TV'  16          * SECOND ITEM IS ROOM TYPE WHERE       *
'105' 'TV'  16          *  'TV' = CLASSES TAUGHT ON TV         *
'106' 'BO'  30          *  'LI' = CLASSES TAUGHT LIVE          *
'107' 'BO'  18          *  'BO' = CLASSES TAUGHT EITHER WAY    *
'108' 'BO'  18          *  'BR' = CLASSES BROADCAST            *
'109' 'BR'  80          * THIRD ITEM IS SEATING CAPACITY       *
'110' 'TV'  10          ****************************************
'201' 'LI'  20
'202' 'LI'  20
'203' 'LI'  26
'204' 'LI'  14
'205' 'LI'  20
'206' 'LI'   6
'207' 'LI'  24
'208' 'LI'   8
'209' 'LI'  40
'210' 'LI'  16
'211' 'LI'   6
'212' 'BR'   0 146
'COMP' 'LI'  0  15
'CONF' 'LI'  0  20
```

FIGURE 22. CLASSROOM DATA FILE "ROOMS.DAT"

# Fall '92 Classes

Revised 6/24/92

| | | | | |
|---|---|---|---|---|
| **ODU:** | | | | |
| ACCT | 600 | Foundations of Accounting | 5:15-7:00 | MW |
| CE | 695 | Introduction Computer Hy | 5:00-6:15 | TR |
| CE | 651 | Water & Wastewater Treatment | 7:00-9:30 | M |
| CHP | 776 | International Health | 4:00-6:30 | R |
| CS | 450 | Data Base Concepts | 7:10-9:50 | T |
| CS | 650 | Computer Aided Design | 3:00-4:15 | TR |
| CRJS | 625 | Administration of Criminal Justice | 9:00-3:00 | S |
| DNTH | 400/500 | Teach Strat for Health Prof. | TBA | TBA |
| DSCI | 600 | Foundations of Statistics | 12:00-12:50 | MWF |
| DSCI | 604 | Intro to Quant Analysis | 5:30-8:30 | TR |
| ECE | 774/874 | Semiconductor Characterization | 1:00-2:15 | MW |
| ECE | 746/846 | Sequential Machines | 7:00-8:15 | TR |
| ECE | 601 | Signal and System Theory | 7:00-8:15 | MW |
| ECE | 461/561 | Auto Controls | 7:10-8:35 | MW |
| ECE | 201 | Circuit Theory | 5:30-7:00 | TR |
| ECE | 676 | Quantum Electronics & Lasers | 4:00-5:15 | MW |
| ECE | 652 | Random Processes | 5:30-6:45 | MW |
| ECE | 796/896 | Multivariable Control | 4:00-5:15 | MW |
| EET | 360 | Electrical Power and Machinery | 7:00-9:45 | T |
| ECI | 495/595 | Topics: Middle School Education | TBA | TBA |
| ECI | 735/835 | Teaching Urban Ed | 4:30-7:00 | W |
| ECI | 795/895 | Topics: Global Curriculum Dev | 4:20-7:00 | R |
| ELS | 600 | Principal Orientation & Leadership Semi. | 4:20-7:05 | T |
| ELS | 783/883 | Issues in Urban Ed Leadership | 4:20-7:00 | R |
| ECON | 613 | Microecon/for Managers | 5:30-8:30 | MW |
| ECON | 630 | Bus Conditions Analysis Forecasting | 5:30-8:30 | MW |
| ENMA | 607 | Advanced Engineering Economics | 6:30-9:15 | R |
| ENMA | 709/809 | Computer Models and Simulation | 6:30-9:45 | T |
| ENMA | 706/806 | Engineering Law | 6:30-9:45 | W |
| ENMA | 695 | Engineering Statistics | 5:00-6:15 | MW |
| ENMA | 713/813 | Logistic Engineering | 7:30-10.15 | R |
| ENMA | 717/817 | Parametric Cost Engineering | 7:00-9:45 | W |
| ENMA | 605 | Problems in Engineering Mgmt | 7:30-10:15 | M |
| ENMA | 741/841 | Production Engineering | 6:30-9.45 | R |
| ENMA | 604 | Project Management | 6:30-9:15 | T |
| HIST | 634 | History of Military Intel. | 7:00-9.50 | T |
| MATH | 691 | Engineering Analysis I | 5:30-6:45 | TR |
| MEM | 625 | Aeromechanics | 2:00-3:15 | TR |
| MEM | 608 | Computational Fluid Dynamics | 2:00-3:15 | MW |
| MEM | 340 | Computational Methods in Mech. Eng | 5:30-6.45 | MW |
| MEM | 639 | Engineering Optimization | 11:30-12:45 | MW |
| MEM | 622 | Mech Behavior of Materials | 11:30-12:45 | TBA |
| MEM | 607 | Mech of Continuous Media I | 7:30-8:45 | TR |
| MEM | 635 | Finite Element Analysis | 5:30-6:45 | TR |
| MEM | 332 | Mechanical Engineering Design I | 5:45-7:00 | TR |
| MEM | 724/824 | Nonlinear Oscillations | 5:30-6:45 | MW |
| MEM | 414/514 | Introduction to Gas Dynamics | 7:15-8:30 | MW |
| MEM | 713/813 | Theory of Transfer Phenomena | 3:30-4.45 | TR |
| MEM | 695 | Topics: MEM Seminar | 2:00-3.30 | F |
| MET | 320 | Design of Machine Elements | 7:00-9:45 | R |
| MGMT | 650 | Organizational Behavior | 5:30-8:30 | TR |
| MGMT | 725/825 | Business Government & Society | 5:30-8:30 | TR |
| NURS | 304 | Concepts of Professional Practice | 4:00-6:45 | M |
| NURS | 680 | Nursing Leadership in Health Care Org. | 4:15-7:15 | T |
| NURS | 363 | Nursing Science | 7:00-9:45 | M |
| NURS | 610 | Perspectives of Nursing Theory | 4:15-7:15 | R |
| NURS | 306 | Theoretical Foundations | 7:00-9:00 | W |
| | | P.E. Review - Electrical | 7:10-9:50 | T |
| | | P.E. Review - Civil | 7:10-9:15 | R |
| | | P.E. Review - Mechanical | 7:10-9.15 | W |
| PADM | 603 | The Environment of Public Adm | 7:10-9:50 | T |
| PADM | 651 | Intro to Public Adm | 6:00-9:00 | T |
| PADM | 725/825 | Business Government and Society | 5:30-8:30 | TR |
| PADM | 733/833 | Legal Foundation Pub. Adm | 5:30-8:30 | TR |
| PSYC | 303U | Industrial Organization PS | 7:10-9:50 | T |
| SOC | 440U | Sociology of Health & Illness | 4:00-6:45 | W |
| URBN | 606 | Urban Research Methods I | 5:30-8:30 | M-W |
| | | BOCA Basic Building Code | 6:30-9:15 | R |
| | | 1-2 Family Dwelling Code | 6:30-9:15 | W |
| | | National Electrical Code I | 6:30-9:15 | T |
| NASA | 101 | NASE Teleconference | 2:30-4:00 | W |
| **UVA:** | | | | |
| ChE | 665 | Techniques for Chemical Eng. Analysis & Design | 6:30-7:45 | TR |
| CE | 623 | Vibrations | 8:00-9:15 | MW |
| EE | 576 | Digital Signal Processing | 5:00-6:15 | MW |
| MAE | 667 | Introduction to Manufacturing Systems | 6:30-7:45 | MW |
| MS | 605 | Structure and Properties of Materials I | 3:30-4:45 | TR |
| MS | 741 | Christal Defect Theory | 5:00-6:15 | TR |
| NE | 614 | Reactor Engineering | 3:30-4:45 | MW |
| SYS | 603 | Mathematical Programming | 8:00-9:15 | TR |
| **VCU:** | | | | |
| STA | 541 | Applied Statistics for Eng & Scientists | 5:00-6:15 | MW |
| **VPI:** | | | | |
| CE | 5125 | Environmental Engineering Design I | 6:30-9.15 | M |
| EE | 5515 | Computer Architecture I | 3:30-4:45 | TR |
| EE | 5704 | Linear System Theory | 6:30-7:45 | TR |
| ENGR | 5004 | The Systems Engineering Process | 5:00-6:15 | MW |
| ESM | 5734 | Introduction to the finite Element Method | 3:30-4:45 | MW |
| ISE | 5434 | Economic Evaluation of Industrial Proj | 6:30-9:15 | W |
| ISE | 5204 | Manufacturing Systems Engineering | 8:00-9:15 | TR |
| ME | 4704 | Tribology | 5:00-6:15 | TR |

| | |
|---|---|
| **ODU:** | |
| Classes Start | August 31 |
| Last Day to Change or Drop w/Refund | September 8 |
| Last Day to Withdraw from Classes | October 27 |
| Holidays | November 25-28 |
| Classes End | December 11 |
| Exam Week | December 14-19 |
| **UVA:** | |
| Classes Start | September 3 |
| Last Day to Change or Drop w/ Refund | 2nd Day of Class |
| Last Day to Withdraw from Classes | November 2 |
| Holidays | Septemper 7 |
| | October 19-20 |
| | November 26 |
| Classes End | December 11 |
| Exam Week | December 14-21 |
| **VCU:** | |
| Classes Start | August 31 |
| Last day to Change or Drop w/Refund | 2nd Day of Class |
| Last Day to Withdraw from Classes | October 23 |
| Holidays | September 7 |
| | November 25-27 |
| Classes End | December 11 |
| Exam Week | December 14-19 |
| **VPI:** | |
| Classes Start | August 25 |
| Last Day to Change or Drop w/ Refund | 2nd Day of Class |
| Last Day to Withdraw from Classes | October 5 |
| Holidays | November 21-29 |
| Classes End | December 9 |
| Exam Week | December 11-17 |

| Tuition- Off Campus - (3 Credit Hours) | ODU | UVA | VCU | VPI |
|---|---|---|---|---|
| In-State | $396.00 | $507.00 | TBA | $567.00 |
| Out-State | $525.00 | $1002.00 | TBA | $960.00 |

FIGURE 23. SEMESTER CLASS LISTING

```
'ECE 774/874'     'MW'    '1:00'  '2:15'  'ODU' 'TV'   1 'ITFS2' '5016'
'ECE 746/846'     'TR'    '7:00'  '8:15'  'PGE' 'BR'   2 'ITFS1' '5020'
'ECE 601'         'MW'    '7:00'  '8:15'  'ODU' 'TV'   6 'TBA  ' '5024'
'ECE 201'         'TR'    '5:30'  '7:00'  'ODU' 'LI'  22
'ECE 461/561'     'MW'    '7:10'  '8:35'  'ODU' 'LI'  25
'ECE 676'         'MW'    '4:00'  '5:15'  'ODU' 'TV'   1 'ITFS1' '5016'
'ECE 652'         'MW'    '5:30'  '6:45'  'ODU' 'TV'   3 'ITFS1' '5016'
'ECE 796/896'     'MW'    '4:00'  '5:15'  'ODU' 'TV'   1 'ITFS2' '5024'
'EET 360'         'T'     '7:00'  '9:45'  'ODU' 'TV'   1 'ITFS-' '5011'
'ECI 495/595'     'W'     '5:30'  '8:30'  'ODU' 'LI'   4
'ECI 735/835'     'W'     '4:30'  '7:00'  'ODU' 'LI'   5
'ECI 795/895'     'T'     '4:00'  '6:00'  'ODU' 'TV'   9 'ITFS3' 'NONE'
'ELS 600'         'T'     '4:20'  '7:05'  'ODU' 'LI'  26
'ELS 783/883'     'R'     '4:20'  '7:00'  'ODU' 'LI'  26
'ECON 613'        'MW'    '5:30'  '8:30'  'ODU' 'LI'   2
'ECON 630'        'MW'    '5:30'  '8:30'  'ODU' 'LI'  33
'ENMA 607'        'R'     '6:30'  '9:15'  'ODU' 'TV'  27 'ITFS1' '5204'
'ENMA 713/813'    'R'     '7:30'  '10:15' 'ODU' 'LI'   5
'ENMA 717/817'    'W'     '7:00'  '9:45'  'ODU' 'TV'   5 'ITFS1' '5016'
'ENMA 605'        'M'     '7:00'  '10:15' 'ODU' 'LI'  24
'ENMA 741/841'    'R'     '6:30'  '9:45'  'ODU' 'LI'   5
'ENMA 604'        'T'     '6:30'  '9:15'  'ODU' 'TV'  15 'ITFS1' '5204'
'HIST 634'        'T'     '7:00'  '9:50'  'ODU' 'LI'  11
'MATH 691'        'TR'    '5:30'  '6:45'  'ODU' 'LI'  21
'MEM 625'         'TR'    '2:00'  '3:15'  'ODU' 'TV'   2 'ITFS1' '5011'
'MEM 608'         'MW'    '2:00'  '3:15'  'ODU' 'TV'   5 'ITFS2' '5011'
'MEM 340'         'MW'    '5:30'  '6:45'  'ODU' 'LI'  24
'MEM 639'         'MW'    '11:30' '12:45' 'ODU' 'TV'   2 'ITFS2' '5011'
'MEM 622'         'TR'    '11:30' '12:45' 'ODU' 'TV'   6 'ITFS1' '5011'
'MEM 607'         'TR'    '7:30'  '8:45'  'ODU' 'TV'  11 'ITFS4' '5704'
'MEM 332'         'TR'    '5:45'  '7:00'  'ODU' 'LI'   8
'MEM 724/824'     'MW'    '5:30'  '6:45'  'ODU' 'TV'   1 'ITFS2' '5024'
'MEM 414/514'     'MW'    '7:15'  '8:30'  'PGE' 'BR'  15 'TBA  ' '5705'
'MEM 713/813'     'TR'    '3:30'  '4:45'  'ODU' 'TV'   4 'ITFS1' '5016'
'MEM 695'         'F'    '2:00'  '3:30'   'ODU' 'TV'  10 'ITFS2' '5011'
'MET 320'         'R'    '7:00'  '9:45'   'ODU' 'TV'   8 'ITFS-' '5011'
'MGMT 650'        'TR'    '5:30'  '8:30'  'ODU' 'LI'   8
'NURS 304'        'M'     '4:00'  '6:45'  'ODU' 'TV'   1 'ITFS4' '5207'
'NURS 680'        'T'     '4:15'  '7:15'  'ODU' 'TV'   4 'ITFS4' '5704'
'NURS 363'        'M'     '7:00'  '9:45'  'ODU' 'TV'   2 'ITFS4' '5207'
'NURS 610'        'R'     '4:15'  '7:15'  'ODU' 'TV'   3 'ITFS4' '5704'
'NURS 306'        'W'     '7:00'  '9:00'  'ODU' 'TV'   1 'ITFS4' '5207'
'P.E. REV. ELEC'  'T'     '7:10'  '9:50'  'ODU' 'LI'   5
'P.E. REV. CIV'   'R'     '7:10'  '9:15'  'ODU' 'LI'   5
'P.E. REV. MECH'  'W'     '7:10'  '9:15'  'ODU' 'LI'   5
'PADM 603'        'T'     '7:10'  '9:50'  'ODU' 'LI'   2
'PADM 651'        'T'     '6:00'  '9:00'  'ODU' 'LI'   3
'PADM 725/825'    'MW'    '5:30'  '8:30'  'ODU' 'LI'  23
'PADM 733/833'    'TR'    '5:30'  '8:30'  'ODU' 'LI'  30
'SOC 440U'        'W'     '4:00'  '6:45'  'ODU' 'TV'   7 'ITFS4' '5207'
'URBN 606'        'MW'    '5:30'  '8:30'  'ODU' 'LI'   2
'BBC'             'R'     '6:30'  '9:15'  'ODU' 'LI'   5
'FDC'             'W'     '6:30'  '9:15'  'ODU' 'LI'   5
'NEC1'            'T'     '6:30'  '9:15'  'ODU' 'LI'   5
```

FIGURE 24. CLASS LOAD DATA FILE "CLASSES.DAT"

- days on which class is offered
- starting and ending times
- originating school
- type of class
- current enrollment
- station and phone numbers for visual and audio connections (necessary for televised classes only)

As noted earlier, this data file is much more subject to change. Changes in class offerings and student enrollments require that the data file be updated. This ensures that the most current status is reflected in the program run, and ultimately, its output.

The third source of input for the program is on-line user responses. The user is prompted for information that allows him/her to account for increases in student enrollment which may occur during the beginning weeks of the semester. If desired, the user is able to specify an addition to the existing enrollment as listed in the data file CLASSES.DAT. Two choices are available:

(1) Amount - The user specifies an amount of students that will be added to each existing class enrollment.

(2) Percentage - The user specifies a percentage that is added to each existing class enrollment.

Figure 25 is a screen print of the messages that the user is offered.

Output File Description

One very important aspect of the scheduling function at the Center is the need for the statistics that result from the course-classroom assignment process. Reports of day schedules, room utilization, and summary statistics are necessary to keep all personnel informed. When the Center first opened in 1986, this was a manageable task because of the relatively small number of rooms, classes and students. With the anticipated number of classes being offered at the Center, it will be nearly impossible to perform this job manually.

This aspect was considered during the planning and design phases of the P.C. Scheduler. As can be seen in Figure 21, there are five separate DOS directories that contain output files. The directories and files are structured so that same-purpose output is located together. The P.C. Scheduler can be grouped into three categories based on the needs they fulfill:

(1) Miscellaneous output - There are two files stored in the directory MISCOUT. The file DATACHEK.OUT serves

```
IN ORDER TO ACCOUNT FOR INCREASES IN ENROLLMENT,
DO YOU WANT TO INCREASE THE EXISTING NUMBER OF
STUDENTS IN EACH CLASS? ( Y or N) ==>
SPECIFY THE METHOD OF STUDENT INCREASE (P)or(A):

(P) = PERCENTAGE METHOD -
USER ENTERS A PERCENT VALUE THAT IS ADDED TO EACH
EXISTING CLASS ENROLLMENT. FOR EXAMPLE, IF
THE USER ENTERS A PERCENT VALUE OF 50 AND A
CLASS HAS AN EXISTING ENROLLMENT OF 6, THE
SCHEDULER WILL FIND A ROOM THAT HOLDS
6 + (50% of 6) = 9 STUDENTS.

(A) = AMOUNT METHOD -
USER ENTERS A STUDENT AMOUNT THAT IS ADDED TO
EACH EXISTING CLASS ENROLLMENT. FOR EXAMPLE, IF
THE USER ENTERS AN AMOUNT VALUE OF 3 AND A
CLASS HAS AN EXISTING ENROLLMENT OF 6, THE
SCHEDULER WILL FIND A ROOM THAT HOLDS
6 + 3 = 9 STUDENTS.

P or A ==>
ENTER AMOUNT VALUE (INTEGER) ==>
ENTER PERCENT VALUE (INTEGER) ==>
```

FIGURE 25. ON-LINE USER PROMPTS

as an echo of the room and class data as read in by the computer. Generally, it is prudent to ensure that this file agrees with the current status of room availability and class load. The file PGECSUM.OUT provides the summary statistics of the Center. Both room and class totals are listed by type of service (live, televised, broadcast). This information is used by the Center's director to provide updates to ODU as well as the other sites in the Academic System. Figure 26 shows an example of this file which reflects a run made during the Fall 1992 semester.

(2) Room schedule output - Located in the directory ROOMOUT, are output files showing the weekly class schedule for rooms at the Center. There is a separate file for each room. The naming convention for these files is simply the room identification. In other words, if Room 101 is identified as '101' in the room data file, then the output file that contains this room's schedule information is 101. The classes are sorted by weekday and by time within each day. This information is used to show the Center's current room utilization. This is helpful because it indicates rooms which are constantly being occupied and those which are seldom used. When a substitute room is needed because of equipment malfunction, this report

PGEC SEMESTER SUMMARY - CLASSES AND ROOMS

AS OF 10/28/1992
18:24 HRS

CLASSROOM INFORMATION:

| ROOM TYPE | NUMBER OF ROOMS | SEATING CAPACITY |
|---|---|---|
| TV | 4 | 50 |
| LIVE | 13 | 200 |
| BOTH* | 4 | 106 |
| BROAD. | 2 | 80 |
| TOTAL | 23 | 436 |

* Some rooms accomodate both TV and LIVE classes

CLASS SCHEDULE INFORMATION:

| CLASS TYPE | NUMBER OF CLASSES | NUMBER OF STUDENTS |
|---|---|---|
| TV | 48 | 286 |
| LIVE | 30 | 354 |
| BROAD. | 4 | 32 |
| TOTAL | 82 | 672 |

FIGURE 26. PENINSULA CENTER SUMMARY STATISTICS

reveals which rooms can be used without conflict. Even more importantly, these reports can be used as a planning tool because they provide ODU and the Center the information needed to determine whether increases in class load are feasible. The room schedule information is also used by the students. When posted at each door, it provides the information necessary to inform the student of room activity. This is helpful because many times the students will use these rooms as study halls. Figure 27 shows an example of the room schedule output file.

(3) Class schedule output - Located in the three directories CLASSTVB, CLASSTVE and CLASSGEN are the output files showing weekday class schedules. The three directories are described below:

- CLASSTVB includes all T.V. and broadcast classes sorted by Beginning class times.
- CLASSTVE includes all T.V. and broadcast classes sorted by class End times.
- CLASSGEN includes all T.V., broadcast and live classes sorted by class beginning times.

The schedules in directories CLASSTVB and CLASSTVE are used strictly by the Broadcast Systems Engineer(BSE). These schedules enable the BSE to be prepared for the classes that he/she is responsible for. Tasks such as

ROOM 101 WEEKLY CLASS SCHEDULE

AS OF 10/28/1992
18:24 HRS

| BEG TIME | END TIME | CLASSID | SCHL | CLASS TYPE | NO. OF STUDENTS |
|---|---|---|---|---|---|
| MONDAY CLASSES | | | | | |
| 11:30 | 12:45 | MEM 639 | ODU | TV | 2 |
| 1:00 | 2:15 | ECE 774/874 | ODU | TV | 1 |
| 3:30 | 4:45 | NE 614 | UVA | TV | 8 |
| 5:00 | 6:15 | EE 576 | UVA | TV | 8 |
| 6:30 | 7:45 | MAE 667 | UVA | TV | 8 |
| 8:00 | 9:15 | CE 623 | UVA | TV | 8 |
| TUESDAY CLASSES | | | | | |
| 11:30 | 12:45 | MEM 622 | ODU | TV | 6 |
| 2:00 | 3:15 | MEM 625 | ODU | TV | 2 |
| 3:30 | 4:45 | MS 605 | UVA | TV | 8 |
| 5:00 | 6:15 | MS 741 | UVA | TV | 8 |
| 6:30 | 7:45 | CHE 665 | UVA | TV | 8 |
| 8:00 | 9:15 | SYS 603 | UVA | TV | 8 |
| WEDNESDAY CLASSES | | | | | |
| 11:30 | 12:45 | MEM 639 | ODU | TV | 2 |
| 1:00 | 2:15 | ECE 774/874 | ODU | TV | 1 |
| 3:30 | 4:45 | NE 614 | UVA | TV | 8 |
| 5:00 | 6:15 | EE 576 | UVA | TV | 8 |
| 6:30 | 7:45 | MAE 667 | UVA | TV | 8 |
| 8:00 | 9:15 | CE 623 | UVA | TV | 8 |
| THURSDAY CLASSES | | | | | |
| 11:30 | 12:45 | MEM 622 | ODU | TV | 6 |
| 2:00 | 3:15 | MEM 625 | ODU | TV | 2 |
| 3:30 | 4:45 | MS 605 | UVA | TV | 8 |
| 5:00 | 6:15 | MS 741 | UVA | TV | 8 |
| 6:30 | 7:45 | CHE 665 | UVA | TV | 8 |
| 8:00 | 9:15 | SYS 603 | UVA | TV | 8 |
| FRIDAY CLASSES | | | | | |
| SATURDAY CLASSES | | | | | |

ROOM 101 SUMMARY:
ROOM TYPE = TV
SEATING CAPACITY = 8
TOTAL CLASS SESSIONS HELD DURING THE WEEK = 24
ROOM SELECTIONS ARE BASED UPON INCREASE IN CLASS ENROLLMENT OF:
0 STUDENTS IN EACH CLASS.

FIGURE 27. ROOM SCHEDULE REPORT

phone link hook-up, video connections, video tape recording, equipment checks and camera/microphone (if broadcast) preparations are all required for each class. Since one person performs these jobs, it is imperative that the BSE be ready. With these schedules sorted by beginning and end times, it is easy to identify when action is needed to fulfill the tasks. The schedules in the directory CLASSGEN are considered GENeral listings because they are used by all personnel at the Center. When posted at the Center's reception area, the students are informed of their class locations and times. During the first weeks of the semester, this is quite useful. These schedules are utilized by the staff in an effort to remain informed of class activity. This is helpful especially when performing duties such as providing examinations and collecting homeworks. Figures 28, 29 and 30 show examples of CLASSTVB, CLASSTVE and CLASSGEN weekday schedule output files, respectively.

MENU system user interface

During this chapter we have discussed some of the details of the P.C. Scheduler software. The algorithms employed, the file and directory organization, the input and

MONDAY CLASS SCHEDULE - TV AND BROADCAST (START TIMES)

AS OF 10/28/1992
18:24 HRS

| BEG TIME | CLASSID | SCHL | ROOM NO. | STA. NO. | PHONE NO. | NO. OF STUDENTS | NO. OF SEATS |
|---|---|---|---|---|---|---|---|
| 11:30 | MEM 639 | ODU | 101 | ITFS2 | 5011 | 2 | 8 |
| 12:00 | DSCI 600 | ODU | 105 | ITFS4 | 5019 | 12 | 16 |
| 1:00 | ECE 774/874 | ODU | 101 | ITFS2 | 5016 | 1 | 8 |
| 2:00 | MEM 608 | ODU | 110 | ITFS2 | 5011 | 5 | 10 |
| 3:30 | ESM 5734 | VPI | 110 | SAT1 | 5006 | 8 | 10 |
| 3:30 | NE 614 | UVA | 101 | SAT5 | 5001 | 8 | 8 |
| 4:00 | NURS 304 | ODU | 108 | ITFS4 | 5207 | 1 | 18 |
| 4:00 | ECE 796/896 | ODU | 104 | ITFS2 | 5024 | 1 | 16 |
| 4:00 | ECE 676 | ODU | 105 | ITFS1 | 5016 | 1 | 16 |
| 5:00 | ENGR 5004 | VPI | 110 | SAT1 | 5006 | 2 | 10 |
| 5:00 | STA541/ENMA695 | VCU | 106 | SAT21 | 5011 | 22 | 30 |
| 5:00 | ACCT 600 | PGE | 109 | ITFS3 | 5706 | 12 | 80 |
| 5:00 | EE 576 | UVA | 101 | SAT5 | 5001 | 8 | 8 |
| 5:30 | ECE 652 | ODU | 104 | ITFS1 | 5016 | 3 | 16 |
| 5:30 | MEM 724/824 | ODU | 105 | ITFS2 | 5024 | 1 | 16 |
| 6:30 | MAE 667 | UVA | 101 | SAT5 | 5001 | 8 | 8 |
| 6:30 | CE 5125 | VPI | 110 | SAT1 | 5006 | 1 | 10 |
| 7:00 | CE 651 | ODU | 104 | ITFS1 | 5016 | 8 | 16 |
| 7:00 | ECE 601 | ODU | 108 | TBA | 5024 | 6 | 18 |
| 7:00 | NURS 363 | ODU | 105 | ITFS4 | 5207 | 2 | 16 |
| 7:15 | MEM 414/514 | PGE | 109 | TBA | 5705 | 15 | 80 |
| 8:00 | CE 623 | UVA | 101 | SAT5 | 5001 | 8 | 8 |

T.V. CLASSES TAUGHT MONDAY = 20
BROADCAST CLASSES TAUGHT MONDAY = 2
ROOM SELECTIONS ARE BASED UPON INCREASE IN CLASS ENROLLMENT OF:
0 STUDENTS IN EACH CLASS.

FIGURE 28. WEEKDAY SCHEDULE REPORT-T.V. CLASSES SORTED BY BEG. TIMES

MONDAY CLASS SCHEDULE - TV AND BROADCAST (END TIMES)

AS OF 10/28/1992
18:24 HRS

| END TIME | CLASS ID | SCHL | ROOM NO. | STA. NO. | PHONE NO. | NO. OF STUDENTS | NO. OF SEATS |
|---|---|---|---|---|---|---|---|
| 12:45 | MEM 639 | ODU | 101 | ITFS2 | 5011 | 2 | 8 |
| 12:50 | DSCI 600 | ODU | 105 | ITFS4 | 5019 | 12 | 16 |
| 2:15 | ECE 774/874 | ODU | 101 | ITFS2 | 5016 | 1 | 8 |
| 3:15 | MEM 608 | ODU | 110 | ITFS2 | 5011 | 5 | 10 |
| 4:45 | NE 614 | UVA | 101 | SAT5 | 5001 | 8 | 8 |
| 4:45 | ESM 5734 | VPI | 110 | SAT1 | 5006 | 8 | 10 |
| 5:15 | ECE 796/896 | ODU | 104 | ITFS2 | 5024 | 1 | 16 |
| 5:15 | ECE 676 | ODU | 105 | ITFS1 | 5016 | 1 | 16 |
| 6:15 | ENGR 5004 | VPI | 110 | SAT1 | 5006 | 2 | 10 |
| 6:15 | STA541/ENMA695 | VCU | 106 | SAT21 | 5011 | 22 | 30 |
| 6:15 | EE 576 | UVA | 101 | SAT5 | 5001 | 8 | 8 |
| 6:45 | MEM 724/824 | ODU | 105 | ITFS2 | 5024 | 1 | 16 |
| 6:45 | ECE 652 | ODU | 104 | ITFS1 | 5016 | 3 | 16 |
| 6:45 | NURS 304 | ODU | 108 | ITFS4 | 5207 | 1 | 18 |
| 7:00 | ACCT 600 | PGE | 109 | ITFS3 | 5706 | 12 | 80 |
| 7:45 | MAE 667 | UVA | 101 | SAT5 | 5001 | 8 | 8 |
| 8:15 | ECE 601 | ODU | 108 | TBA | 5024 | 6 | 18 |
| 8:30 | MEM 414/514 | PGE | 109 | TBA | 5705 | 15 | 80 |
| 9:15 | CE 623 | UVA | 101 | SAT5 | 5001 | 8 | 8 |
| 9:15 | CE 5125 | VPI | 110 | SAT1 | 5006 | 1 | 10 |
| 9:30 | CE 651 | ODU | 104 | ITFS1 | 5016 | 8 | 16 |
| 9:45 | NURS 363 | ODU | 105 | ITFS4 | 5207 | 2 | 16 |

T.V. CLASSES TAUGHT MONDAY = 20
BROADCAST CLASSES TAUGHT MONDAY = 2
ROOM SELECTIONS ARE BASED UPON INCREASE IN CLASS ENROLLMENT OF:
0 STUDENTS IN EACH CLASS.

FIGURE 29. WEEKDAY SCHEDULE REPORT-T.V. CLASSES SORTED BY END TIMES

MONDAY CLASS SCHEDULE - TV, LIVE AND BROADCAST

AS OF 10/28/1992
18:24 HRS

| BEG TIME | END TIME | CLASSID | SCHL | CLASS TYPE | NO. OF STUDENTS | ROOM NO. | NO. OF SEATS |
|---|---|---|---|---|---|---|---|
| 11:30 | 12:45 | MEM 639 | ODU | TV | 2 | 101 | 8 |
| 12:00 | 12:50 | DSCI 600 | ODU | TV | 12 | 105 | 16 |
| 1:00 | 2:15 | ECE 774/874 | ODU | TV | 1 | 101 | 8 |
| 2:00 | 3:15 | MEM 608 | ODU | TV | 5 | 110 | 10 |
| 3:30 | 4:45 | ESM 5734 | VPI | TV | 8 | 110 | 10 |
| 3:30 | 4:45 | NE 614 | UVA | TV | 8 | 101 | 8 |
| 4:00 | 6:45 | NURS 304 | ODU | TV | 1 | 108 | 18 |
| 4:00 | 5:15 | ECE 796/896 | ODU | TV | 1 | 104 | 16 |
| 4:00 | 5:15 | ECE 676 | ODU | TV | 1 | 105 | 16 |
| 5:00 | 6:15 | ENGR 5004 | VPI | TV | 2 | 110 | 10 |
| 5:00 | 6:15 | STA541/ENMA695 | VCU | TV | 22 | 106 | 30 |
| 5:00 | 7:00 | ACCT 600 | PGE | BR | 12 | 109 | 80 |
| 5:00 | 6:15 | EE 576 | UVA | TV | 8 | 101 | 8 |
| 5:30 | 8:30 | PADM 725/825 | ODU | LI | 23 | 207 | 24 |
| 5:30 | 6:45 | ECE 652 | ODU | TV | 3 | 104 | 16 |
| 5:30 | 8:30 | ECON 613 | ODU | LI | 2 | 206 | 6 |
| 5:30 | 8:30 | URBN 606 | ODU | LI | 2 | 211 | 6 |
| 5:30 | 6:45 | MEM 340 | ODU | LI | 24 | 203 | 26 |
| 5:30 | 8:30 | ECON 630 | ODU | LI | 33 | 103 | 40 |
| 5:30 | 6:45 | MEM 724/824 | ODU | TV | 1 | 105 | 16 |
| 6:30 | 7:45 | MAE 667 | UVA | TV | 8 | 101 | 8 |
| 6:30 | 9:15 | CE 5125 | VPI | TV | 1 | 110 | 10 |
| 7:00 | 10:15 | ENMA 605 | ODU | LI | 24 | 203 | 26 |
| 7:00 | 9:30 | CE 651 | ODU | TV | 8 | 104 | 16 |
| 7:00 | 8:15 | ECE 601 | ODU | TV | 6 | 108 | 18 |
| 7:00 | 9:45 | NURS 363 | ODU | TV | 2 | 105 | 16 |
| 7:10 | 8:35 | ECE 461/561 | ODU | LI | 25 | 209 | 40 |
| 7:15 | 8:30 | MEM 414/514 | PGE | BR | 15 | 109 | 80 |
| 8:00 | 9:15 | CE 623 | UVA | TV | 8 | 101 | 8 |

T.V. CLASSES TAUGHT MONDAY = 20
LIVE CLASSES TAUGHT MONDAY = 7
BROADCAST CLASSES TAUGHT MONDAY = 2
ROOM SELECTIONS ARE BASED UPON INCREASE IN CLASS ENROLLMENT OF:
0 STUDENTS IN EACH CLASS.

FIGURE 30. WEEKDAY SCHEDULE REPORT-GENERAL CLASSES SORTED BY BEG. TIMES

output file descriptions are all essential to the understanding of the program. Acknowledging that, it should also be noted that any software package is only successful if it is used by the consumer. For a consumer group that has a wide range of computer experience, an interface that facilitates the accessing of files, executing of the program and the printing of reports will, no doubt, improve the product and process. To that end, it was decided that a "user friendly" interface would be invoked as the driver of the P.C. Scheduler. A menu interface such as the Three Dimensional Menu System would accomplish the goal.

A series of menus in this system allow the user to perform the above tasks with little effort. No memorization of DOS commands is necessary since all commands are "built in" to the menu system. Figure 31 shows the organization of the panels and options in the menu system. The following list describes the functions that are available to the user at each menu panel:

- Panel PCMAIN - Main Peninsula Center menu. Options include all available software available to the user.
- Panel PCSCHED - Peninsula Center Scheduler menu. Options include editing input files, running P.C. Scheduler program and printing output reports.

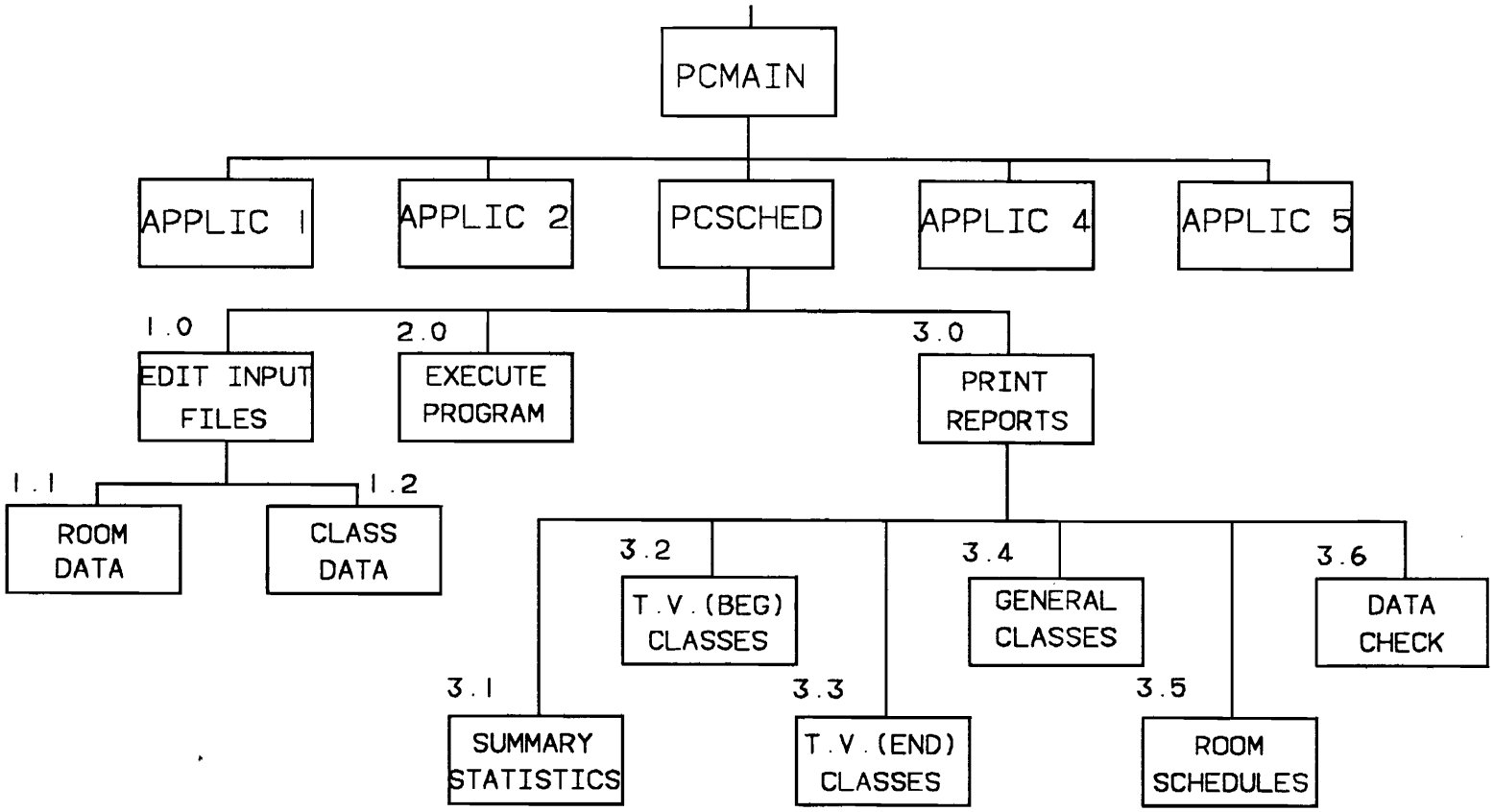



FIGURE 31. P.C. SCHEDULER MENU PANELS

- Panel 1.0 - Edit Input File menu. Options include editing room and class data files.
- Panel 3.0 - Print Output Report menu. Options include printing summary, weekday schedule, room schedule and data check output files.
- Panel 3.2 - Print T.V. Class(Beg. Time Sort) Weekday Schedule menu. Options include printing each of the day's schedules or printing the entire week's schedule.
- Panel 3.3 - Print T.V. Class(End Time Sort) Weekday Schedule menu. Options include printing each of the day's schedules or printing the entire week's schedule.
- Panel 3.4 - Print All Class(Beg. Time Sort) Weekday Schedule menu. Options include printing each of the day's schedules or printing the entire week's schedule.
- Panel 3.5 - Print Room Schedule menu. Options include printing schedules for rooms 101-110 and schedules for rooms 201-212.

## Chapter 5

## RECOMMENDATIONS FOR PROCESS IMPROVEMENTS

The P.C. Scheduler has proven to be a valuable asset to the Center's organization. As can be seen from Appendix C, it has greatly improved the efficiency of the scheduling process and has aided the Center's director and staff in performing their jobs. The Scheduler, as it is currently functioning, will most likely be used to support all class scheduling in future semesters. It is important to recognize, however, that all processes, undoubtedly, experience change with time. As these processes change, so must the components that make up the system. In order to serve the Center's system, the P.C. Scheduler system should keep the pace with these changes. As discussed earlier, it has been established that the staff will meet at least once each semester to discuss any "lessons learned" or improvement suggestions to the existing program. It is hoped that this will result in a better product for the Center. The Scheduler system was designed and written in a "modular" format to facilitate future modifications. Ultimately, this should result in a process of ongoing improvement.

With this in mind, the following sections briefly discuss some enhancements to the existing scheduling system that may ultimately improve the overall performance of the Center as an educational system. These are based upon the experience gained from the use of the P.C. Scheduler in the Fall 1992 semester and from the expectations for the Center in the future. These program enhancements are divided into two categories - present operations and future expansion.

Present Operations

This section addresses enhancements that can be implemented in the PC. Scheduler which would improve the present operations at the Center. They are based mostly on the experience gained from the implementation of the program during the Fall 1992 semester.

(1) Class Prioritizing - Presently, the Scheduler determines the smallest room that will accommodate each class. There are cases when an instructor requests a particular room because of preferences such as seat arrangement (i.e., conference style vs. conventional setting) or room location. A provision in the program to account for this type of priority would be helpful.

(2) Improving Output - The output for class and room schedules resulting from the Scheduler program is presently given in report format. This is certainly adequate for the staff and student body to determine their schedules, but there is interest in getting a more visual representation of the course-classroom assignments. A bar chart graph as shown in Figure 32 would facilitate the identification of peak activity and slack activity periods.

(3) Projecting Enrollment Changes - An inherent part of the scheduling system is the change in enrollment status during the course of the semester start. This caused much of the difficulty when scheduling at the Center was done manually. With the P.C. Scheduler this iterative process of room assignments is no longer necessary. As explained earlier, there are provisions to allow for enrollment growth during the course of the semester. The user chooses the amount of students to be added to existing enrollment. This is straightforward and effective but this process could be improved as time progresses. The staff has recently begun to keep records of enrollment changes for each class offered at the Center. These records track the change in enrollment during the first month of each semester.

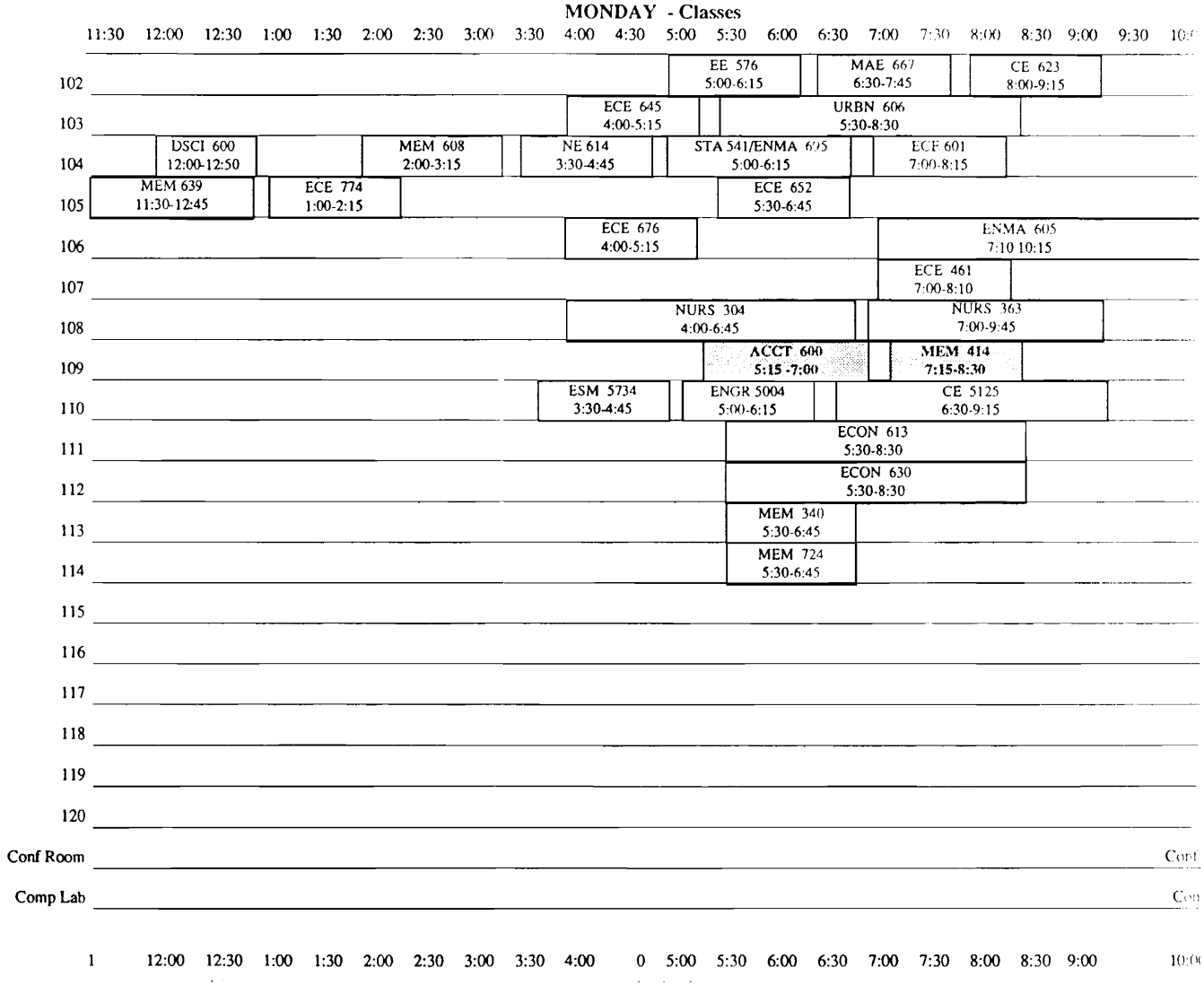



FIGURE 32. BAR CHART FORMAT FOR SCHEDULES

Since many classes are taught from year to year, it is expected that the enrollment activity in these classes will show some behavior that can be used in projections of future semesters. This may require several years of data, but the results can then be used to develop probability distributions of enrollment changes. The goal is to determine, as accurately as possible, the eventual class size of each course at a point when the initial scheduling is done at the Center. This point is approximately two months prior to the deadline of drop/add decisions.

Future Expansion

This section addresses enhancements to the P.C. Scheduler that would improve the Center's operations and that of the Academic System but are dependent on future expansion.

(1) Automating Class Input - As it exists now, the class information for the Scheduler must be manually inputted. With new computer resources, including a Sun Network area, there are plans to link the WIN 386 Computer Laboratory to the ODU computer system. Since

ODU has in their computer system, the information regarding all classes taught at the Center, this connection would allow the staff to "import" the computer file containing this information. The raw data could then be converted to a format that is compatible with the P.C. Scheduler program. This would reduce the time spent by the staff in manually inputting the initial class information data.

(2) Determining Maximum Class Load - The P.C. Scheduler can be used not only to provide the proper course-classroom assignments for those class loads already established, but it could serve as a tool to make the determination of maximum classes capable of being offered at the Center. This decision is made months prior to the semester start. When done manually, this is a time consuming process that results in many iterations. With the Scheduler, candidate classes with estimated enrollments can be used to ask "what if" questions of assignments. This would improve the overall efficiency of the Center as well as the Academic System by providing maximum utilization of the rooms at the Center.

(3) Implementing P.C. Scheduler at other sites - The Center is just one of several sites in the Hampton Roads area that provide off-campus education. Although

their arrangements vary from that of the Center, their purpose is similar. Modifications can be made to the Scheduler to account for specific constraints and parameters so that it can provide the same service that is available at the Center.

# Chapter 6

## CONCLUSION

In this report, the activities of the Peninsula Center educational site are defined and described as part of the higher-level educational distribution network. The purpose of the report is to identify the need and provide the development process of the Peninsula Center Scheduler program.

The report provides a perspective of the process as seen from the systems engineer's viewpoint during the planning, design, implementation and maintenance of the scheduling system. As a result, the interaction between the engineer and the Center's staff is described throughout the report. An overview of the Center is provided as well as an explanation of the Scheduler design life-cycle to illustrate the phases of the project from the identification of the need to program use and maintenance. A description of each program aspect is included as well as details in the Appendices.

The program allows the Center's staff to determine, based on the classroom size and type, and the time of day and length of each class, where each of the classes should

be held. The program developed allows the staff to save a considerable amount of time, allocating the classroom space with the precision and efficiency previously difficult to achieve using the manual method.

Also, recommendations for future improvement of capabilities is presented to show expansion possibilities. Based on the first semester's implementation of the P.C. Scheduler, there are several recommendations for future modifications that will result in improvements in the performance of the system. They are discussed in detail in Chapter 5 but are subdivided and summarized below by (1) Present Operations and (2) Future Expansion.

The Present Operations recommendations include those that are relatively easy to implement because they can be achieved with little disruption of current operations at the Center. The plan is to implement these by the Fall 1993 semester start. These are:

(1) Class prioritizing - A provision that will allow an instructor to specify a preferred room.

(2) Improvement in output - The capability to provide graphical scheduling output in the form of a bar chart.

(3) Enrollment change projection - Data collection of repetitious classes that will lead to more accurate projections in final class enrollments for each semester.

The Future Expansion recommendations include those that require a change to the current operations of the Peninsula Center as well as the Hampton Roads Academic System. The plan is to implement these as soon as practicable or when expansion efforts allow it. These are:

(1) Automation of class input - The ability to import class load and student enrollment information directly from ODU through a computer network.

(2) Maximum class load determination - Preliminary scheduling done by the Center to determine if increases in class load can be offered.

(3) Implementation at other sites - Installation of the P.C. Scheduler at other sites in Hampton Roads.

The implementation of these modifications will be facilitated by the fact that the continued use and

maintenance of the P.C. Scheduler was considered during its planning and design.

The Peninsula Center has proven to be a vital part of the Hampton Roads education system and is, therefore, expected to continue in its expansion efforts. The Peninsula Center Scheduler provides a system that allows the staff to support expanding student demands more efficiently and accurately than the previous method of manual scheduling. Like any other system, the key to continued success is the acceptance to change. With this attitude, the Center's staff and the systems engineer can work together to maintain a system that results in maximum efficiency for the Center and the Academic System as a whole.

## REFERENCES


[1] 'Partners with the Peninsula in Economic Development' (Advertisement Package, 1992).

[2] 'Old Dominion University on the Peninsula' (Information Brochure, 1992).

[3] 'Graduate Engineering Education' (Peninsula Center Information Booklet, 1991).

[4] Blanchard, Benjamin S., and Wolter J. Fabrycky. Systems Engineering and Analysis, 2nd Edition, Englewood Cliffs, New Jersey: Prentice Hall, Inc., 1990.

[5] Canada, John R., and William G. Sullivan. Economic and Multiattribute Evaluation of Advanced Manufacturing Systems, Englewood Cliffs, New Jersey: Prentice Hall, Inc., 1989.

[6] Smith, R.L. (1971). 'Accommodating Student Demand for Courses by Varying the Class Room Mix', Operations Research, Vol. 19, pp. 862-874.

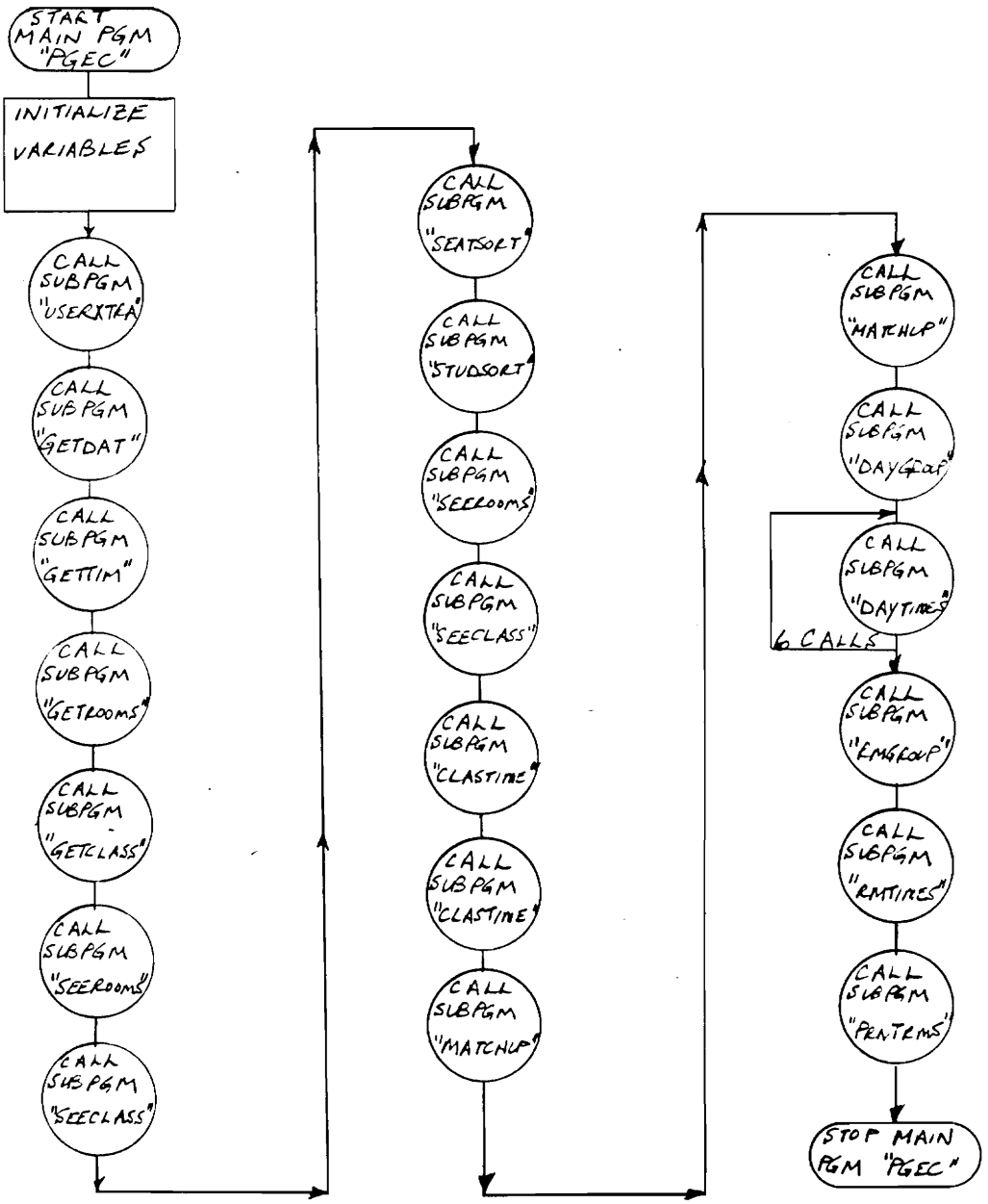
[7] Carter, M. W. (1984). 'A Survey of Practical Applications of Examination Timetabling Algorithms', Operations Research, Vol. 34, pp. 193-202.

# APPENDIX A

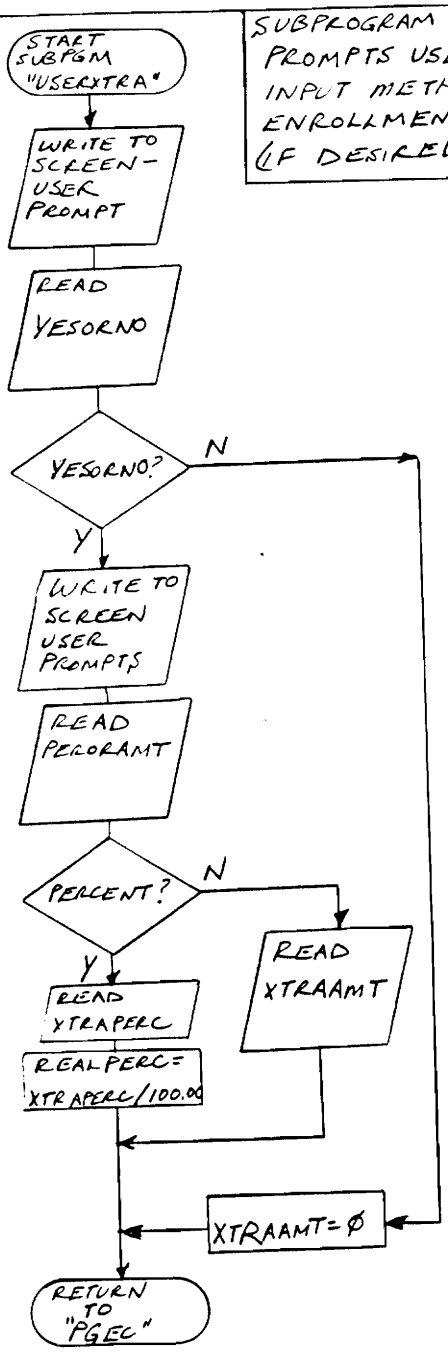
## P.C. SCHEDULER FLOWCHARTS

# FLOWCHART - MAIN PROGRAM "PGEC"

## FORTRAN 77 LANGUAGE



# SUBPROGRAMS "USERXTRA", "GETDAT" & "GETTIM"

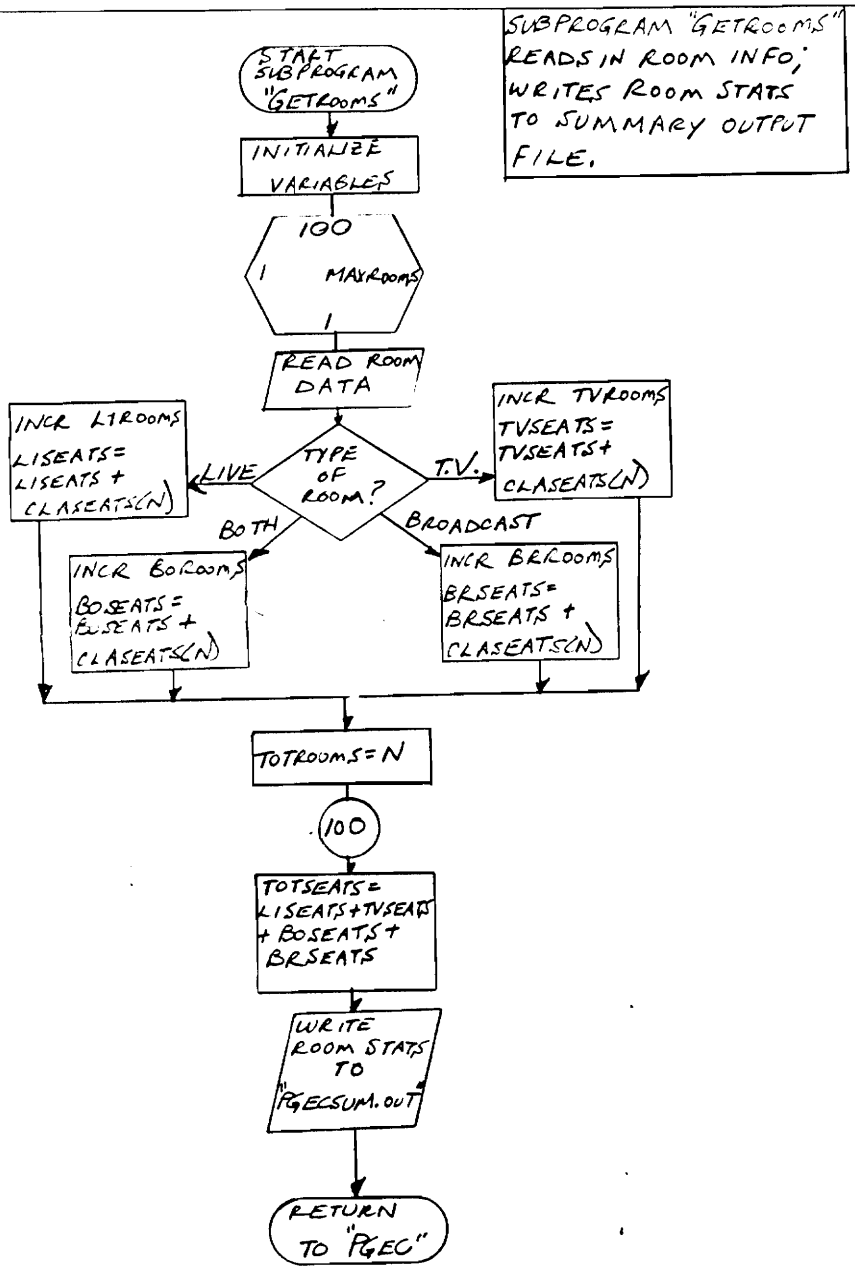



SUBPROGRAM "USERXTRA" PROMPTS USER TO INPUT METHOD OF ENROLLMENT INCREASE (IF DESIRED).

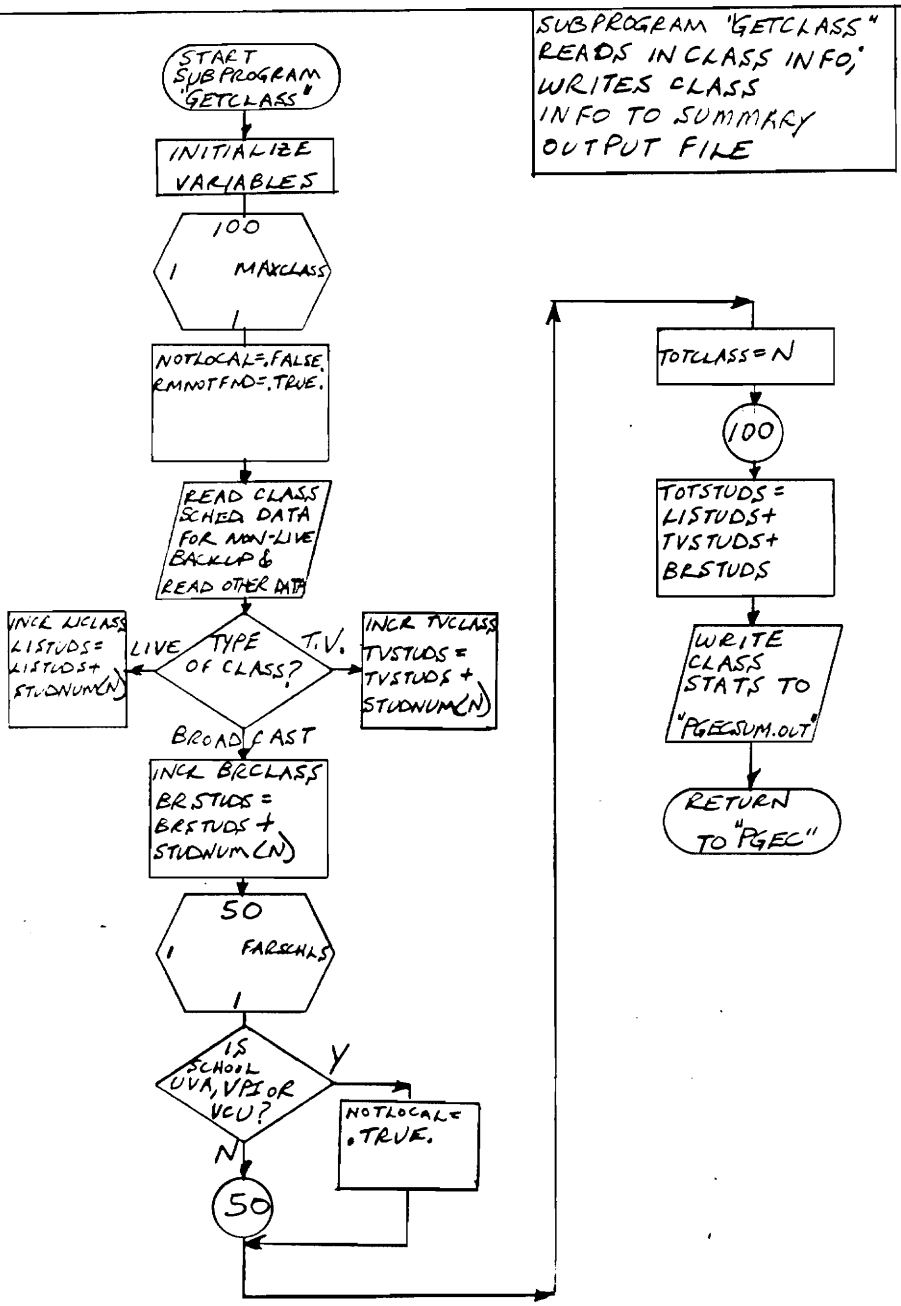
SUBPROGRAMS "GETDAT" AND "GETTIM" ARE BUILT IN FORTRAN SUBROUTINES THAT RETURN THE COMPUTER SYSTEM DATE AND TIME, RESPECTIVELY.

THE SYSTEM DATE AND TIME IS WRITTEN TO ALL OUTPUT FILES SO THAT SUBSEQUENT RUNS OF THE SCHEDULING PROGRAM CAN BE IDENTIFIED IN ORDER TO DETERMINE THE MOST UP-TO-DATE SCHEDULING INFORMATION.

# SUBPROGRAM "GETROOMS"

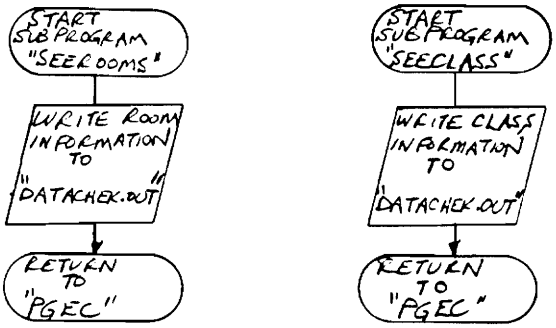


# SUBPROGRAM "GETCLASS"



## SUBPROGRAMS "SEEROOMS" AND "SEECLASS"

SUBPROGRAMS "SEEROOMS" AND "SEECLASS" WRITE ROOM AND CLASS SCHEDULE INFORMATION TO FILE "DATACHEK.OUT." USER SHOULD BROWSE THIS FILE TO ENSURE THAT THE DATA WAS READ IN AS INTENDED.

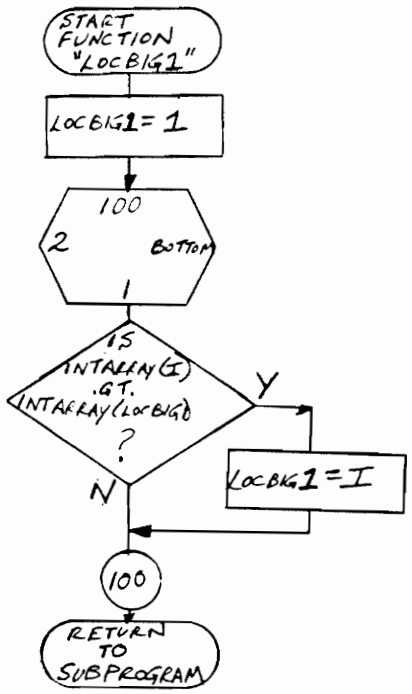



BOTH SUBPROGRAMS ARE CALLED TWICE IN THE MAIN PROGRAM. THIS IS DONE SO THAT THE USER CAN SEE THAT THE PROGRAM IS CONVERTING THE DATA VALUES PROPERLY.

# SUBPROGRAM "SEATSORT"

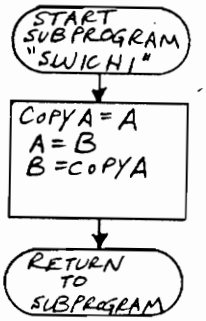
SUBPROGRAM "SEATSORT"
SORT ROOM INFORMATION
IN ASCENDING
SEAT ORDER.

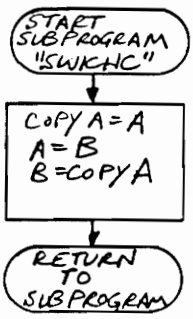
## FUNCTION "LOCBIG1" AND SUBPROGRAMS "SWICHI", "SWICHC" & "SWICHL"

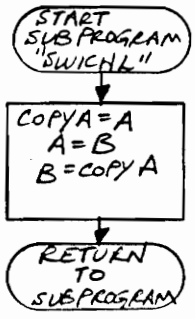



FUNCTION "LOCBIG1"* LOCATES THE LARGEST INTEGER IN AN ARRAY. SUBPROGRAM "SWICHI" SWITCHES INTEGER ARRAY ELEMENTS AS PART OF THE ASCENDING SORT. SUBPROGRAMS "SWICHC" AND "SWICHL" PERFORM SIMILAR SWITCHES WITH CHARACTER AND LOGICAL ARRAY ELEMENTS, RESPECTIVELY.

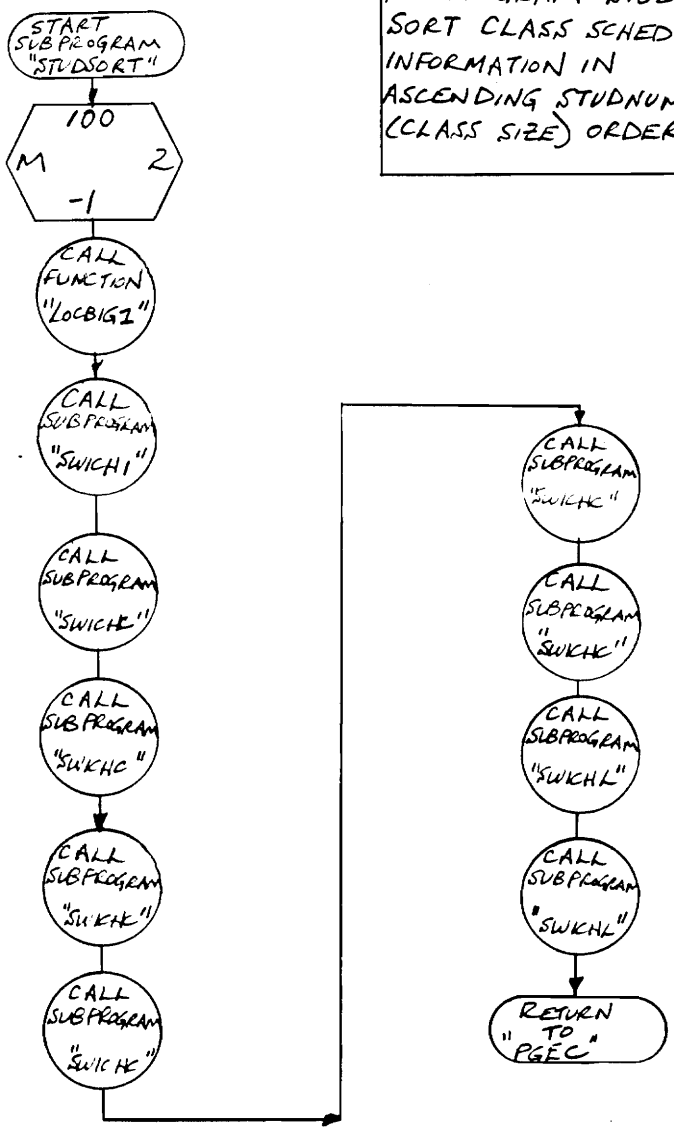
* FUNCTIONS "LOCBIG2" AND "LOCBIG3" PERFORM SIMILAR CALCULATIONS ON 2-D AND 3-D ARRAYS, RESPECTIVELY.



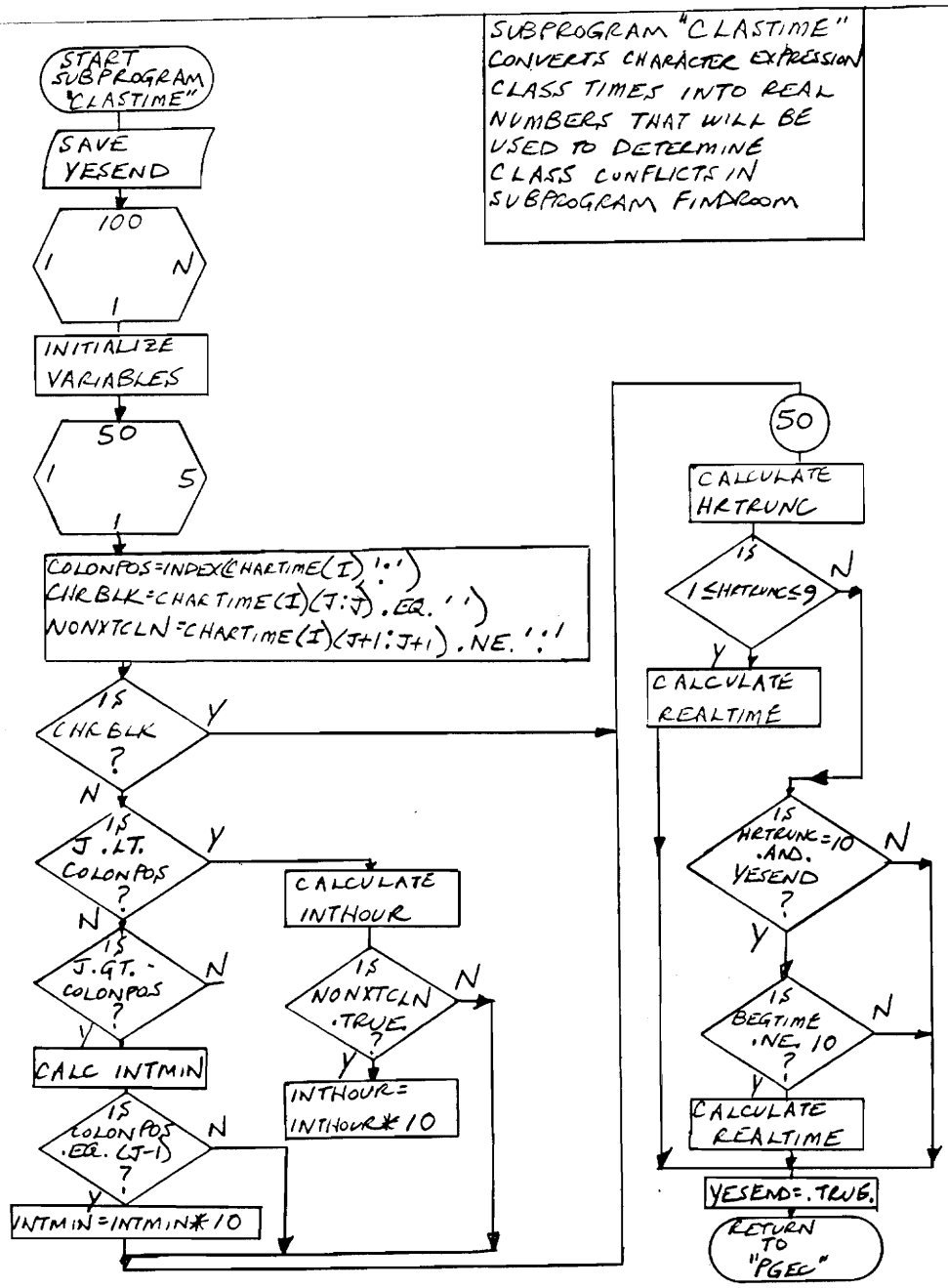



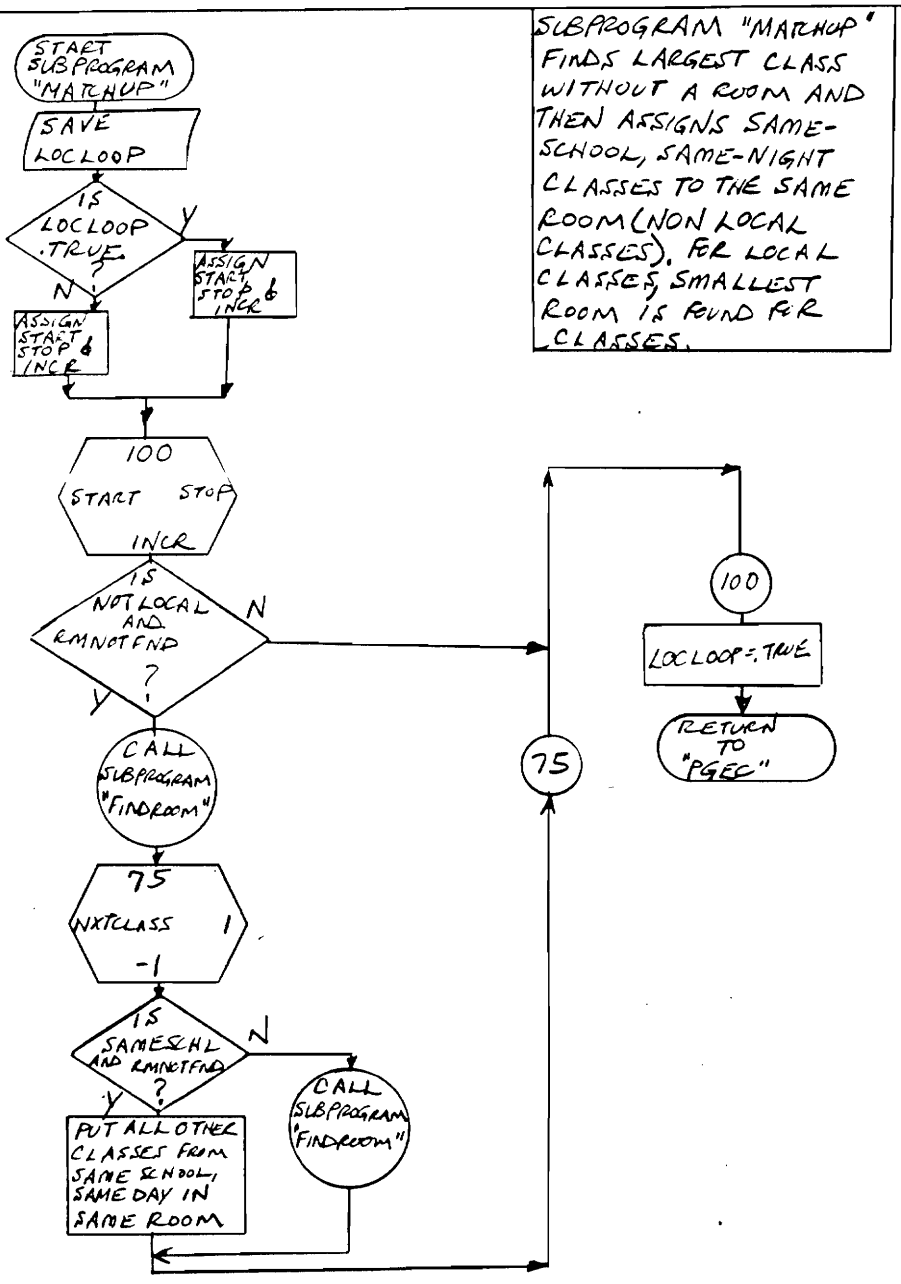
# SUBPROGRAM "STUDSORT"



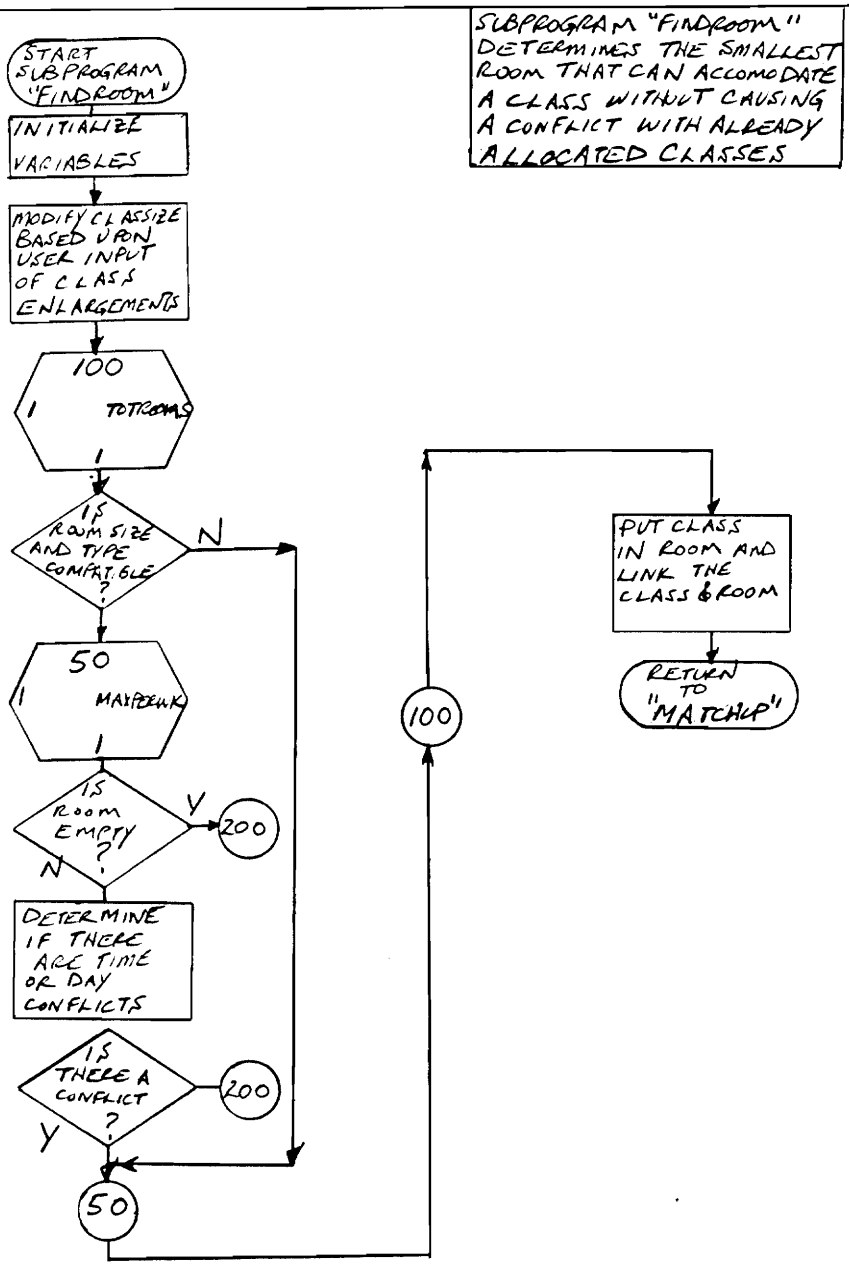
# SUBPROGRAM "CLASTIME"



# SUBPROGRAM "MATCHUP"

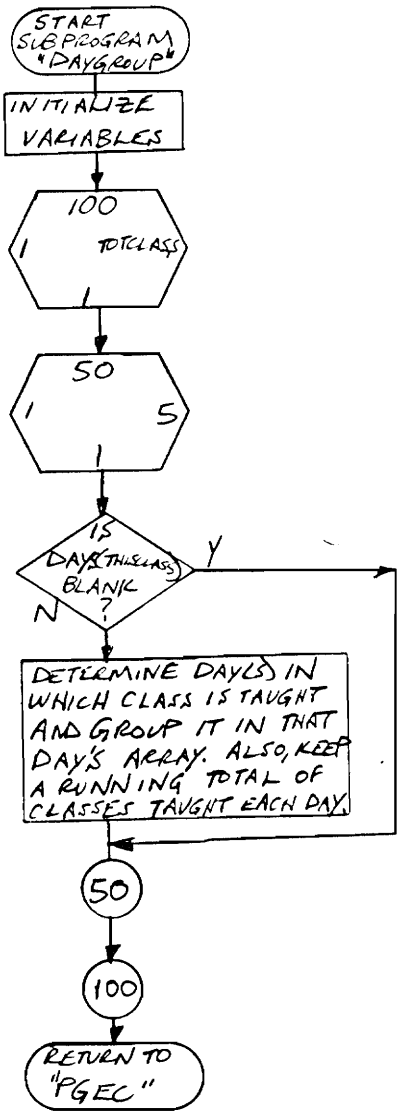


# SUBPROGRAM "FINDROOM"



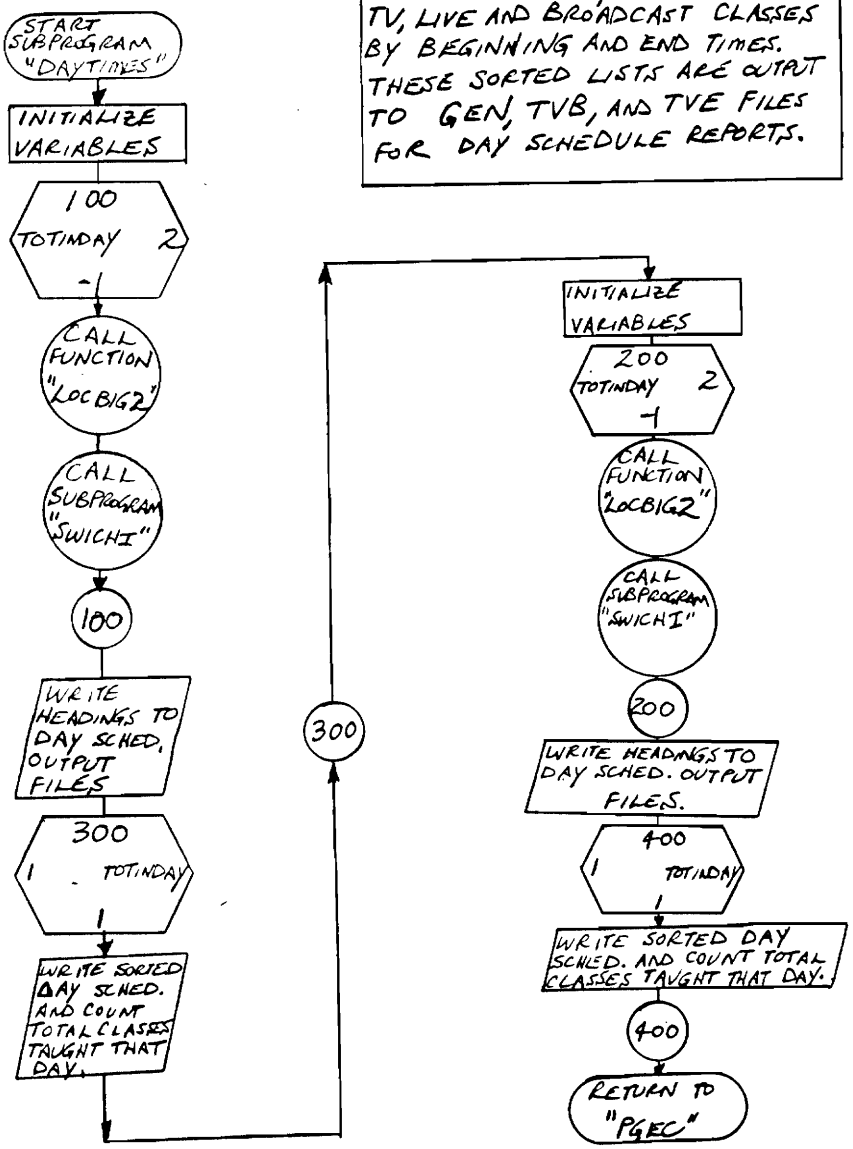
# SUBPROGRAM "DAYGROUP"

SUBPROGRAM "DAYGROUP" GROUPS CLASSES BY THE DAY(S) IN WHICH THEY'RE TAUGHT. THIS WILL BE USED WHEN THE DAYS' SCHEDULES ARE PRINTED.



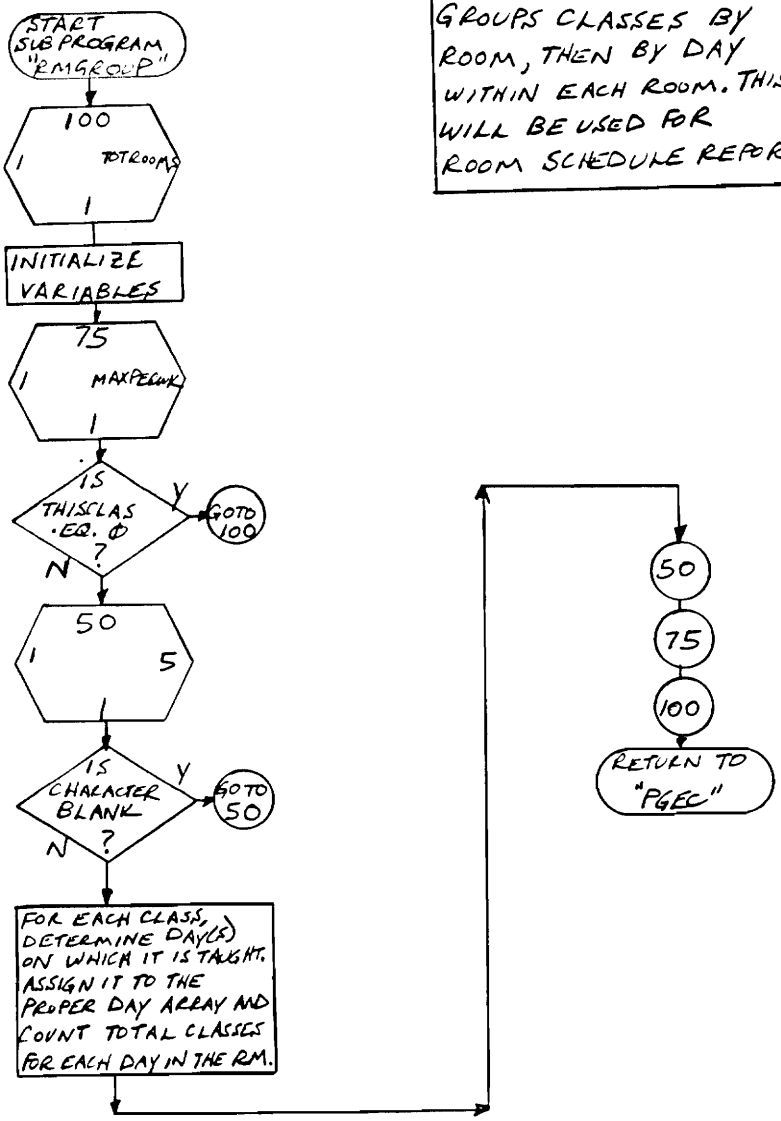
# SUBPROGRAM "DAYTIMES"

SUBPROGRAM "DAYTIMES" SORTS TV, LIVE AND BROADCAST CLASSES BY BEGINNING AND END TIMES. THESE SORTED LISTS ARE OUTPUT TO GEN, TVB, AND TVE FILES FOR DAY SCHEDULE REPORTS.

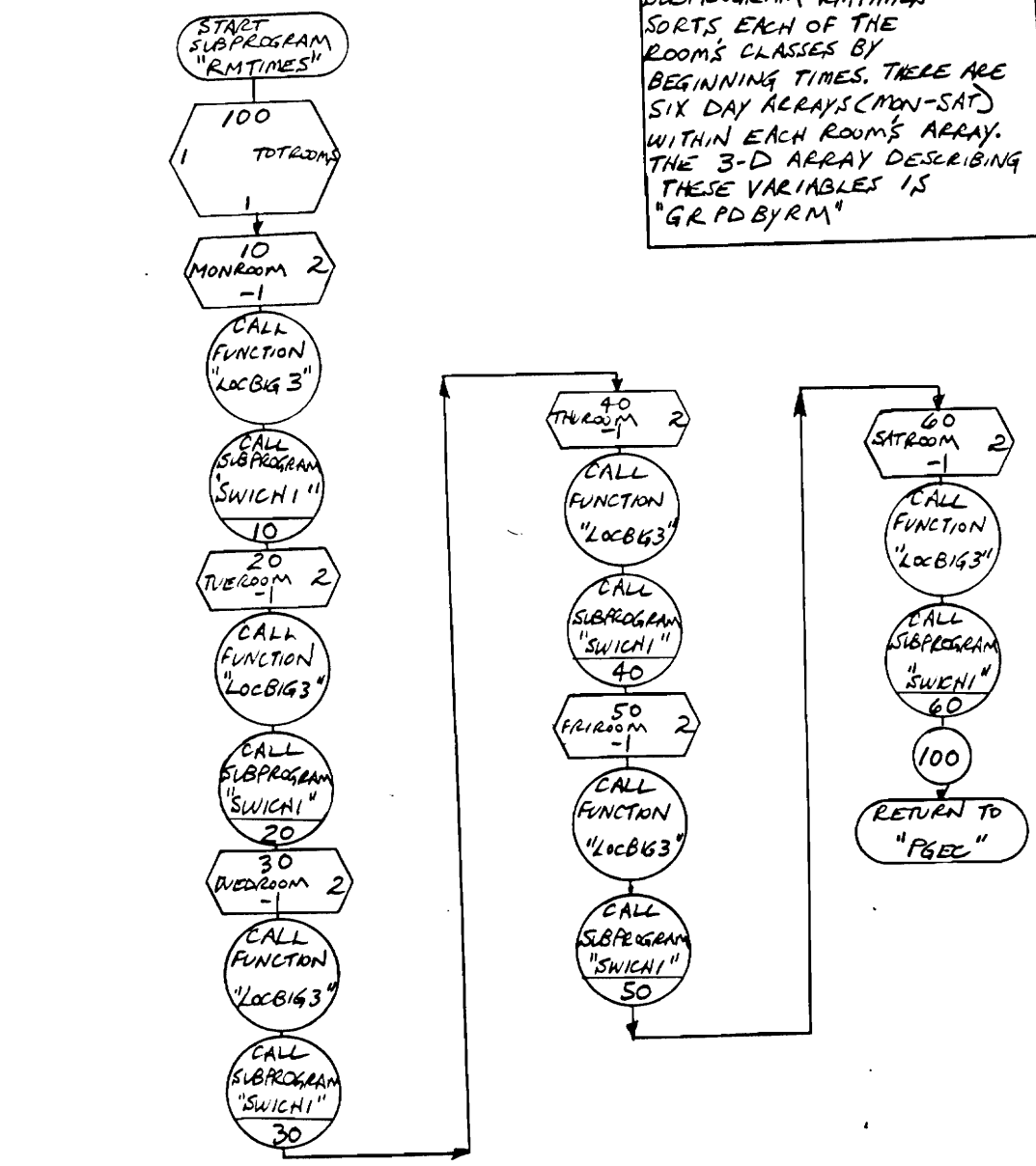


# SUBPROGRAM "RMGROUP"

SUBPROGRAM "RMGROUP" GROUPS CLASSES BY ROOM, THEN BY DAY WITHIN EACH ROOM. THIS WILL BE USED FOR ROOM SCHEDULE REPORTS

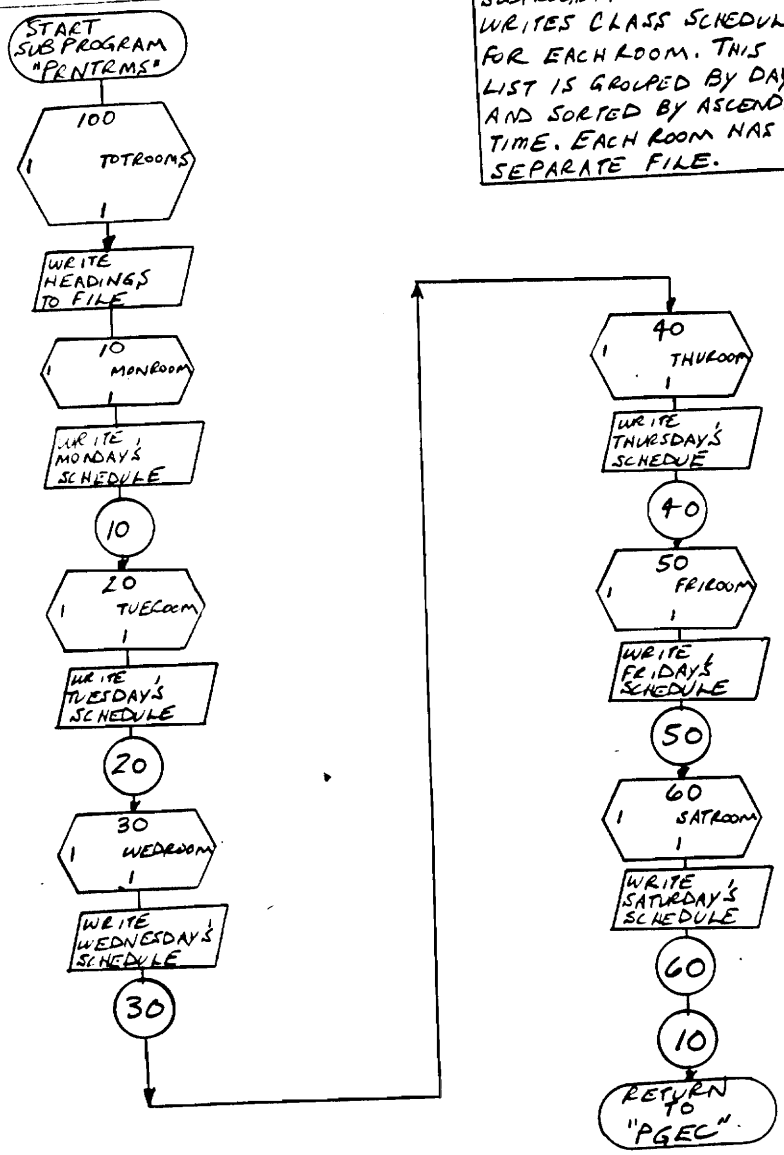


# SUBPROGRAM "RMTIMES"



# SUBPROGRAM "PRNTRMS"

SUBPROGRAM 'PRNTRMS" WRITES CLASS SCHEDULE FOR EACH ROOM. THIS LIST IS GROUPED BY DAY AND SORTED BY ASCENDING TIME. EACH ROOM HAS A SEPARATE FILE.



# APPENDIX B

## P.C.SCHEDULER SOURCE CODE

## (FORTRAN 77)

```
C  ******************SYSTEMS ENGINEERING FINAL PROJECT*******************
C  *                                                                     *
C  *                    V.P.I. AND STATE UNIVERSITY                      *
C  *                           ENGR 5409                                 *
C  *                  STEVEN MACRI    S.S.# 121-58-3238                  *
C  *                              1992                                   *
C  *                                                                     *
C  ******************SYSTEMS ENGINEERING FINAL PROJECT*******************
C
C  PURPOSE -
C  PROGRAM DETERMINES THE MOST EFFICIENT SCHEDULING ASSIGNMENT FOR CLASSES
C  TAUGHT AT THE PENINSULA GRADUATE ENGINEERING CENTER (P.G.E.C.).
C
C  ---------------------------------------------------------------------
C  ---------------------------------------------------------------------
C
C  CONSIDERATIONS AND LIMITATIONS -
C    1.) MAXIMUM ROOMS AVAILABLE = 50
C    2.) MAXIMUM COURSES TAUGHT IN A SEMESTER = 200
C    3.) MAXIMUM COURSES TAUGHT IN ANY ONE ROOM PER WEEK = 60
C    4.) CLASSES CAN BE TAUGHT DURING SIX DAY WEEK (MONDAY - SATURDAY)
C    5.) CLASS START TIME NO EARLIER THAN 10:00 A.M.
C    6.) CLASS START TIME EARLIER THAN 10:00 P.M.
C
C  ---------------------------------------------------------------------
C  ---------------------------------------------------------------------
C
C  DATA ENTRY -   .
C  TWO DATA FILES ARE NEEDED TO RUN THE PROGRAM.
C     1.) rooms.dat - A LISTING OF EACH ROOM AT P.G.E.C.
C         COLUMNS 1-4 ===> ROOM NUMBER OR IDENTIFICATION
C         COLUMNS 6-7 ===> ROOM TYPE
C                          WHERE: BR = BROADCAST CAPABLE (STUDIO)
C                                 TV = TELEVISION IN ROOM
C                                 LI = LIVE CAPABLE (BLACKBOARD IN ROOM)
C                                 BO = BOTH TV AND LIVE CAPABLE
C         COLUMNS 9-11 ===> NUMBER OF SEATS IN ROOM (INTEGER)
C
C     2.) classes.dat - A LISTING OF CLASSES BEING OFFERED AT P.G.E.C.
C         COLUMNS 1-15  ===> CLASS NUMBER OR IDENTIFICATION
C         COLUMNS 17-21 ===> DAYS WHEN THE CLASS IS TAUGHT (EXAMPLE = MWF)
C            MONDAY=M, TUESDAY=T, WED'DAY=W, THURSDAY=R, FRIDAY=F, SAT'DAY=S
C         COLUMNS 23-27 ===> CLASS START TIME (EXAMPLE = 12:00)
C         COLUMNS 29-33 ===> CLASS END TIME   (EXAMPLE =  1:30)
C       *****NOTE: DO NOT USE MILITARY TIME FORMAT FOR CLASS TIMES*****
C         COLUMNS 35-37 ===> HOST SCHOOL
C                            WHERE: ODU = OLD DOMINION UNIVERSITY
C                                   VPI = VIRGINIA POLYTECHNIC INSTITUTE & S.U.
C                                   UVA = UNIVERSITY OF VIRGINIA
C                                   VCU = VIRGINIA COMMONWEALTH UNIVERSITY
C         COLUMNS 39-40 ===> CLASS TYPE
C                            WHERE: BR = BROADCAST FROM P.G.E.C.
C                                   TV = TELEVISED AT P.G.E.C.
C                                   LI = TAUGHT LIVE AT P.G.E.C.
C         COLUMNS 42-44 ===> NUMBER OF STUDENTS IN THE CLASS (INTEGER)
```

```
C
C  ------------------------------------------------------------------
C  ------------------------------------------------------------------
C
C  DESCRIPTION OF VARIABLES -
C
C  maxrooms--THE MAXIMUM NUMBER OF ROOMS AVAILABLE FOR CLASSES.
C  maxclass--THE MAXIMUM NUMBER OF COURSES TAUGHT IN A SEMESTER.
C  farschls--NUMBER OF "FAR SCHOOLS" (OR NON-LOCAL) THAT OFFER COURSES.
C  maxperwk--THE MAXIMUM NUMBER OF COURSES TAUGHT IN ANY ONE ROOM PER WEEK.
C  totrooms--TOTAL ROOMS (AS INPUT BY USER) AVAILABLE FOR CLASSES.
C  totlive--TOTAL LIVE CLASSES BEING TAUGHT DURING SEMESTER.
C  tottv--TOTAL T.V. CLASSES BEING TAUGHT DURING SEMESTER.
C  xtraamt--EXTRA STUDENTS USER ADDS TO EACH CLASS TO ACCOUNT FOR ENROLLMENT INC
REASES.
C  maxperrm--MAXIMUM CLASSES TAUGHT IN A ROOM ON A GIVEN DAY.
C  montot-sattot--TOTAL CLASSES TAUGHT AT THE CENTER FOR EACH DAY.
C  year,month,date--SYSTEM VARIABLES THAT DEFINE THE DATE OF THE RUN.
C  hour,minute,second,hundreds--SYSTEM VARIABLES THAT DEFINE THE TIME OF THE RUN
.
C  monday( )-satday( )--ARRAYS THAT LIST ALL CLASSES TAUGHT ON EACH OF THE DAY.
C  monroom( )-satroom( )--ARRAYS OF CLASSES HELD IN EACH ROOM FOR EACH DAY.
C  grpdbyrm( )--ARRAY OF CLASSES GROUPED BY DAY AND ROOM.
C  realperc--EXTRA PERCENT USER ADDS TO EACH CLASS SIZE TO ACCOUNT FOR ENROLLMEN
T INCR.
C  phone( )--ARRAY OF PHONE NUMBER CONNECTIONS FOR TELEVISED CLASSES.
C  station( )--ARRAY OF STATION NUMBER CONNECTIONS FOR TELEVISED CLASSES.
C  percent--LOGICAL DEFINING WHETHER USER WISHES PERCENT OR AMOUNT METHOD OF ADD
L STUDS.
C  n, m, i, j-- COUNTERS FOR LOOPS.
C  totclass--TOTAL CLASSES (AS INPUT BY USER) BEING OFFERED DURING THE SEMESTER.
C  begtime( )--ARRAY OF BEGINNING TIMES(REAL NUMBERS) FOR CLASSES.
C  endtime( )--ARRAY OF ENDING TIMES(REAL NUMBERS) FOR CLASSES.
C  claseats( )--ARRAY OF SEATING CAPACITY FOR EACH ROOM.
C  studnum( )--ARRAY OF STUDENT NUMBER FOR EACH CLASS.
C  clasinrm( )--ARRAY TO KEEP TRACK OF CLASSES SCHEDULED FOR EACH ROOM.
C  rm2class( )--ARRAY TO LINK THE ROOMS THAT HOLD EACH CLASS.
C  roomtype( )--ARRAY OF COURSE TYPE THAT ROOM CAN ACCOMODATE.
C  clastype( )--ARRAY OF COURSE TYPE FOR THE CLASS BEING TAUGHT.
C  school( )--ARRAY TO IDENTIFY HOST SCHOOL OF CLASS BEING TAUGHT.
C  farschl( )--ARRAY OF "FAR SCHOOL" (OR NON-LOCAL) NAMES THAT OFFER COURSES.
C  roomno( )--ARRAY OF IDENTIFIERS FOR AVAILABLE ROOMS(FOUR CHARACTERS ALLOWED).
C  days( )--ARRAY TO IDENTIFY DAYS ON WHICH CLASS IS BEING TAUGHT.
C  begin( )--ARRAY OF BEGINNING TIMES(CHARACTER EXPRESSION) FOR CLASSES.
C  end( )--ARRAY OF ENDING TIMES(CHARACTER EXPRESSION) FOR CLASSES.
C  classid( )--ARRAY TO IDENTIFY CLASSES BEING OFFERED.
C  rmnotfnd( )--ARRAY TO IDENTIFY WHEN A ROOM HAS BEEN FOUND FOR A CLASS.
C  notlocal( )--ARRAY TO IDENTIFY ALL NON-LOCAL SCHOOLS OFFERING CLASSES.
C
C  ------------------------------------------------------------------
C  ------------------------------------------------------------------
C
C  DESCRIPTION OF SUBPROGRAMS -
C
C  getrooms--READS IN ROOM INFO;WRITES ROOM INFO TO SUMMARY OUTPUT FILE.
C  getclass--READS IN CLASS INFO;WRITES CLASS INFO TO SUMMARY OUTPUT FILE.
C  userxtra--PROMPTS USER TO INPUT METHOD OF ENROLLMENT INCREASE(IF DESIRED).
C  seatsort--SORTS ROOM INFORMATION IN ASCENDING SEAT ORDER.
C  studsort--SORTS CLASS INFORMATION IN ASCENDING STUDENT ORDER.
```

```
C  swichi--SWITCHES INTEGER ARRAY ELEMENTS AS PART OF THE SORT.
C  swichc--SWITCHES CHARACTER ARRAY ELEMENTS AS PART OF THE SORT.
C  swichl--SWITCHES LOGICAL ARRAY ELEMENTS AS PART OF THE SORT.
C  clastime--CONVERTS CHARACTER EXPRESSION CLASS TIMES TO REAL NUMBERS.
C  seerooms--WRITES ROOM INFO TO DATA CHECK OUTPUT FILE.
C  seeclass--WRITES CLASS INFO TO DATA CHECK OUTPUT FILE.
C  matchup--FINDS LARGEST CLASS WITHOUT A ROOM AND THEN ASSIGNS
C           SAME-SCHOOL SAME-NIGHT CLASSES TO THE SAME ROOM.
C  findroom--DETERMINES THE SMALLEST ROOM THAT CAN ACCOMODATE A CLASS WITHOUT
C           CAUSING A CONFLICT WITH ALREADY ALLOCATED CLASSES.
C  daygroup--GROUPS CLASSES BY DAY.
C  daytimes--SORTS CLASSES FOR EACH DAY BY BEGINNING AND END TIMES.
C  rmgroup--GROUPS CLASSES BY ROOM THEN BY DAY.
C  rmtimes--SORTS CLASSES BY BEGINNING TIME WITHIN DAYS, WITHIN ROOM.
C  prntrms--WRITES CLASS SCHEDULE FOR EACH ROOM.
C  locbig1--FUNCTION THAT DETERMINES THE LARGEST INTEGER IN A 1-D ARRAY.
C  locbig2--FUNCTION THAT DETERMINES THE LARGEST INTEGER IN A 2-D ARRAY.
C  locbig3--FUNCTION THAT DETERMINES THE LARGEST INTEGER IN A 3-D ARRAY.
C
C  ---------------------------------------------------------------------
C  ---------------------------------------------------------------------
C
C  *****BEGIN MAIN PROGRAM "PGEC"**************************************
      program PGEC
      integer maxrooms, maxclass, farschls, totdays, maxperwk
      integer totrooms, n, totclass, m, i, j, xtraamt
      integer totlive, tottv, totstuds, maxperrm
      parameter (maxrooms = 50, maxclass = 200, maxperrm = 20,
     +           farschls = 3, totdays = 6, maxperwk = 60)
      integer claseats(maxrooms),studnum(maxclass),rm2class(maxclass)
      integer clasinrm(maxrooms,maxperwk)
      integer montot, tuetot, wedtot, thutot, fritot, sattot
      integer year, month, date, hour, minute, second, hundreds
      integer monday(250), tueday(250), wedday(250)
      integer thuday(250), friday(250), satday(250)
      integer monroom(50), tueroom(50), wedroom(50)
      integer thuroom(50), friroom(50), satroom(50)
      integer grpdbyrm(maxrooms, totdays, maxperrm)
      real begtime(maxclass), endtime(maxclass), realperc
      character*(1) eachday(totdays)
      character*(2) roomtype(maxrooms), clastype(maxclass)
      character*(3) school(maxclass), farschl(farschls)
      character*(4) roomno(maxrooms), phone(maxclass)
      character*(5) days(maxclass), begin(maxclass), end(maxclass)
      character*(5) station(maxclass)
      character*(15) classid(maxclass)
      logical rmnotfnd(maxclass), notlocal(maxclass), percent
C ***** GROUP VARIABLES IN COMMON AREAS - EASIER FOR PASSING TO SUBROUTINES
      common /comrm1/ roomno, roomtype
      common /comrm2/ claseats
      common /comrm3/ monroom,tueroom,wedroom,thuroom,friroom,satroom
      common /comclas1/ classid, days, begin, end, school, clastype
      common /comclas2/ studnum, rmnotfnd, notlocal, rm2class
      common /comclas3/ begtime, endtime
      common /comclas4/ station, phone
```

```
      common /comday1/ montot, tuetot, wedtot, thutot, fritot, sattot
      common /comday2/ monday, tueday, wedday, thuday, friday, satday
      common /comdate/ year, month, date, hour, minute
      implicit none
      data (farschl(i), i = 1, farschls)/'VPI', 'VCU', 'UVA'/
      data (eachday(i), i = 1, totdays)/'M', 'T', 'W', 'R', 'F', 'S'/
C *****
C ***** INITIALIZE ARRAY THAT KEEPS TRACK OF CLASSES SCHEDULED IN EACH ROOM.
      do 10 i = 1, maxrooms
         do 5 j = 1, maxperwk
            clasinrm(i, j) = 0
    5       continue
   10    continue
C ***** INITIALIZE PHONE AND STATION ARRAYS TO ENSURE THEY ARE ALPHA BLANKS.
      do 20 i = 1, maxclass
         station(i) = ' '
         phone(i) = ' '
   20    continue
C ***** ALLOW USER TO INCREASE CURRENT ENROLLMENT TO ACCOUNT FOR INCREASES.
      call userxtra(percent, xtraamt, realperc)
C ***** GET CURRENT SYTEM DATE - THIS WILL BE PRINTED ON EACH OUTPUT REPORT.
      call getdat(year, month, date)
C ***** GET CURRENT SYSTEM TIME - THIS WILL BE PRINTED ON EACH OUTPUT REPORT.
      call gettim(hour, minute, second, hundreds)
C ***** BEGIN READING IN CLASSROOM DATA*****
      open (unit=30,file='C:\PGEC\MISCOUT\'//'pgecsum.out',
     +      form='formatted',status='old')
      call getrooms (maxrooms, totrooms)
C ***** END READING IN CLASSROOM DATA*****
C ***** BEGIN READING IN CLASS SCHEDULE DATA*****
      call getclass (maxclass, totclass, farschl, farschls)
      close (unit = 30)
C ***** END READING IN CLASS SCHEDULE DATA*****
      open (unit=31,file='C:\PGEC\MISCOUT\'//'datachek.out',
     +      form='formatted',status='old')
C ***** WRITE INPUT DATA FILE INFORMATION TO "DATACHEK" OUTPUT FILE.
      call seerooms (totrooms)
      call seeclass (totclass)
C ***** SORT THE ROOM INFORMATION IN ORDER OF ASCENDING "claseats".
      call seatsort(totrooms)
C ***** SORT THE CLASS INFORMATION IN ORDER OF ASCENDING "studnum".
      call studsort (totclass)
C *****  WRITE SORTED OUTPUT TO "DATACHEK" OUTPUT FILE.
      call seerooms (totrooms)
      call seeclass (totclass)
      close (unit = 31)
C ***** CONVERT THE BEGINNING CLASS TIMES FROM CHARACTER TO REAL NUMBERS.
      call clastime (begin, begtime, totclass, begtime)
C ***** CONVERT THE ENDING CLASS TIMES FROM CHARACTER TO REAL NUMBERS.
      call clastime (end, endtime, totclass, begtime)
C ***** FIND ROOMS FOR THE "FAR SCHOOL" CLASSES BEING OFFERED.
      call matchup (totclass, totrooms, clasinrm, maxperwk,
     +              eachday, totdays, percent, xtraamt, realperc)
C ***** FIND ROOMS FOR THE LOCAL CLASSES BEING OFFERED.
      call matchup (totclass, totrooms, clasinrm, maxperwk,
```

```
     +                eachday, totdays, percent, xtraamt, realperc)
C ***** GROUP CLASSES BY DAY OF THE WEEK.
      call daygroup (days, totclass)
C ***** SORT EACH DAY'S CLASSES BY BEGINNING/ENDING TIMES AND THEN
C ***** WRITE THESE SORTED LISTS TO SEPARATE OUTPUT FILES.....TV AND GENERAL.
      call daytimes (monday, montot, begtime, endtime, 'MONDAY',
     +                'monday.tvb', 'monday.tve', 'monday.gen',
     +                percent, xtraamt, realperc)
      call daytimes (tueday, tuetot, begtime, endtime, 'TUESDAY',
     +                'tueday.tvb', 'tueday.tve', 'tueday.gen',
     +                percent, xtraamt, realperc)
      call daytimes (wedday, wedtot, begtime, endtime, 'WEDNESDAY',
     +                'wedday.tvb', 'wedday.tve', 'wedday.gen',
     +                percent, xtraamt, realperc)
      call daytimes (thuday, thutot, begtime, endtime, 'THURSDAY',
     +                'thuday.tvb', 'thuday.tve', 'thuday.gen',
     +                percent, xtraamt, realperc)
      call daytimes (friday, fritot, begtime, endtime, 'FRIDAY',
     +                'friday.tvb', 'friday.tve', 'friday.gen',
     +                percent, xtraamt, realperc)
      call daytimes (satday, sattot, begtime, endtime, 'SATURDAY',
     +                'satday.tvb', 'satday.tve', 'satday.gen',
     +                percent, xtraamt, realperc)
C ***** GROUP CLASSES, FIRST BY ROOM, THEN BY DAY IN WHICH THEY ARE TAUGHT.
      call rmgroup (totrooms, totclass, days, grpdbyrm,
     +               clasinrm, maxperwk,
     +               roomno, classid)
C ***** SORT EACH ROOM LIST OF CLASSES BY BEGINNING TIMES.
      call rmtimes (totrooms, begtime, grpdbyrm)
C ***** WRITE EACH ROOM LIST OF CLASSES TO A SEPARATE FILE FOR EACH ROOM.
      call prntrms (totrooms, grpdbyrm, studnum,
     +               percent, xtraamt, realperc)
 1000 format (a)
      end
C **********END MAIN PROGRAM "PGEC"**************************************
C *****
C     *****BEGIN SUBROUTINE "USERXTRA"**********************************
C *****
C ***** ALLOWS USER TO INCREASE CLASS SIZE TO ACCOUNT FOR ENROLLMENT INCREASES.
      subroutine userxtra(percent, xtraamt, realperc)
      integer xtraamt, xtraperc
      real realperc
      character*(1) yesorno, peroramt
      logical percent
      percent = .false.
      write (*,*) 'IN ORDER TO ACCOUNT FOR INCREASES IN ENROLLMENT,'
      write (*,*) 'DO YOU WANT TO INCREASE THE EXISTING NUMBER OF'
      write (*,*) 'STUDENTS IN EACH CLASS? ( Y or N) ==>'
      read (*,1000) yesorno
      if (yesorno .eq. 'Y' .or. yesorno .eq. 'y') then
         write (*,*) 'SPECIFY THE METHOD OF STUDENT INCREASE (P)or(A):'
         write (*,*)
         write (*,*) '(P) = PERCENTAGE METHOD - '
         write (*,*) 'USER ENTERS A PERCENT VALUE THAT IS ADDED TO EACH'
         write (*,*) 'EXISTING CLASS ENROLLMENT. FOR EXAMPLE, IF'
```

```
         write (*,*) 'THE USER ENTERS A PERCENT VALUE OF 50 AND A '
         write (*,*) 'CLASS HAS AN EXISTING ENROLLMENT OF 6, THE '
         write (*,*) 'SCHEDULER WILL FIND A ROOM THAT HOLDS'
         write (*,*) '6 + (50% of 6) = 9 STUDENTS.'
         write (*,*)
         write (*,*) '(A) = AMOUNT METHOD - '
         write (*,*) 'USER ENTERS A STUDENT AMOUNT THAT IS ADDED TO'
         write (*,*) 'EACH EXISTING CLASS ENROLLMENT. FOR EXAMPLE, IF'
         write (*,*) 'THE USER ENTERS AN AMOUNT VALUE OF 3 AND A'
         write (*,*) 'CLASS HAS AN EXISTING ENROLLMENT OF 6, THE'
         write (*,*) 'SCHEDULER WILL FIND A ROOM THAT HOLDS'
         write (*,*) '6 + 3 = 9 STUDENTS.'
         write (*,*)
         write (*,*) 'P or A ==>'
         read (*,1000) peroramt
         if (peroramt .eq. 'P' .or. peroramt .eq. 'p') then
            percent  = .true.
            write (*,*) 'ENTER PERCENT VALUE (INTEGER) ==>'
            read (*,1001) xtraperc
            realperc = real(xtraperc)/100.00
         else
            write (*,*) 'ENTER AMOUNT VALUE (INTEGER) ==>'
            read (*,1001) xtraamt
         endif
      else
C ***** IF USER DOES NOT WANT INCREASE, THEN EXTRA AMOUNT OF STUDENTS = 0.
         xtraamt = 0
      endif
 1000 format (a)
 1001 format (i3)
      end
C     *****END SUBROUTINE "USERXTRA"**********************************
C *****
C     *****BEGIN SUBROUTINE "GETROOMS"*********************************
C *****
C ***** PRINTS SEMESTER SUMMARY FOR TOTALS (CLASS AND ROOM) AT PGEC
C ***** LI=LIVE, TV= TELEVISION, BO=BOTH, BR=BROADCAST
      subroutine getrooms(maxrooms, totrooms)
      integer lirooms, liseats
      integer tvrooms, tvseats
      integer borooms, boseats
      integer brrooms, brseats
      integer claseats(50), maxrooms, totrooms, n, totseats
      integer year, month, date, hour, minute
      character*(2) roomtype(50)
      character*(4) roomno(50)
      common /comrm1/ roomno, roomtype
      common /comrm2/ claseats
      common /comdate/ year, month, date, hour, minute
      implicit none
      open (unit=20, file='C:\PGEC\INPUT\'//'rooms.dat',
     +      form='formatted',status='old')
      lirooms = 0
      liseats = 0
      tvrooms = 0
```

```
      tvseats = 0
      borooms = 0
      boseats = 0
      brrooms = 0
      brseats = 0
C ***** READ IN ROOM DATA AND COUNT SEATS AVAILABLE IN CLASSROOMS. SEATS
C ***** ARE CATAGORIZED BY ROOM CAPABILITY.
      do 100 n = 1, maxrooms
         read (20, *, end = 110) roomno(n), roomtype(n), claseats(n)
         if (roomtype(n) .eq. 'LI') then
            lirooms = lirooms + 1
            liseats = liseats + claseats(n)
         else if (roomtype(n) .eq. 'TV') then
            tvrooms = tvrooms + 1
            tvseats = tvseats + claseats(n)
         else if (roomtype(n) .eq. 'BO') then
            borooms = borooms + 1
            boseats = boseats + claseats(n)
         else
            brrooms = brrooms + 1
            brseats = brseats + claseats(n)
         endif
         totrooms = n
  100    continue
C ***** CALC TOTAL SEATS AT PGEC.
  110 totseats = liseats + tvseats + boseats + brseats
      close (unit = 20)
C ***** WRITE ROOM SUMMARY STATISTICS TO OUTPUT FILE "PGECSUM.OUT"
      write (30, 1000) 'PGEC SEMESTER SUMMARY - CLASSES AND ROOMS'
      write (30, *)
      write (30, 1001) 'AS OF ', month, '/', date, '/', year
      write (30, 1001) '      ', hour, ':', minute, ' HRS'
      write (30, *)
      write (30, *)
      write (30, *) 'CLASSROOM INFORMATION:'
      write (30, *)
      write (30, 1100) 'ROOM', 'NUMBER', 'SEATING '
      write (30, 1100) 'TYPE', 'OF ROOMS', 'CAPACITY'
      write (30, 1200) 'TV', tvrooms, tvseats
      write (30, 1200) 'LIVE', lirooms, liseats
      write (30, 1200) 'BOTH*', borooms, boseats
      write (30, 1200) 'BROAD.', brrooms, brseats
      write (30, *)    '_____________________________________'
      write (30, 1200) 'TOTAL', totrooms, totseats
      write (30, *)
      write (30, *) '* Some rooms accomodate both TV and LIVE classes'
 1000 format (t20, a)
 1001 format (t60, 2(a, i2), a, i4)
 1100 format (t5, a, t15, a, t25, a)
 1200 format (t5, a, t15, i3, t25, i3)
      end
C     *****END SUBROUTINE "GETROOMS"**********************************
C *****
C     *****BEGIN SUBROUTINE "GETCLASS"********************************
C *****
```

```
C ***** PRINTS SEMESTER SUMMARY FOR TOTALS (CLASS AND ROOM) AT PGEC
      subroutine getclass(maxclass,totclass,farschl,farschls)
      integer liclass, listuds
      integer tvclass, tvstuds
      integer brclass, brstuds
      integer maxclass, totclass, n, totstuds, farschls, j
      integer studnum(200), rm2class(200)
      character*(2) clastype(200)
      character*(3) school(200), farschl(farschls)
      character*(4) phone(200)
      character*(5) days(200), begin(200), end(200), station(200)
      character*(15) classid(200)
      logical notlocal(200), rmnotfnd(200)
      common /comclas1/ classid, days, begin, end, school, clastype
      common /comclas2/ studnum, rmnotfnd, notlocal, rm2class
      common /comclas4/ station, phone
      implicit none
      open (unit=21, file='C:\PGEC\INPUT\'//'classes.dat',
     +      form='formatted',status='old')
      liclass = 0
      listuds = 0
      tvclass = 0
      tvstuds = 0
      brclass = 0
      brstuds = 0
C ***** READ IN CLASS SCHEDULE DATA
      do 100 n = 1, maxclass
         notlocal(n) = .false.
         rmnotfnd(n) = .true.
         read (21, *, end = 110) classid(n), days(n), begin(n),
     +        end(n), school(n), clastype(n), studnum(n)
C ***** IF THE CLASS TYPE IS NOT LIVE, THEN BACKUP AND READ IN STATION/PHONE DAT
A ALSO.
      if (clastype(n) .ne. 'LI') then
            backspace (unit = 21)
            read (21, *, end = 110) classid(n), days(n), begin(n),
     +           end(n), school(n), clastype(n), studnum(n),
     +           station(n), phone(n)
         endif
C ***** GET A COUNT OF CLASSES/STUDENTS  CATEGORIZED BY CLASS TYPE.
         if (clastype(n) .eq. 'LI') then
            liclass = liclass + 1
            listuds = listuds + studnum(n)
         else if (clastype(n) .eq. 'TV') then
            tvclass = tvclass + 1
            tvstuds = tvstuds + studnum(n)
         else
            brclass = brclass + 1
            brstuds = brstuds + studnum(n)
         endif
C ***** DETERMINE IF THIS CLASS'S SCHOOL IS NOT LOCAL.
         do 50 j = 1, farschls
            if (school(n) .eq. farschl(j)) then
               notlocal(n) = .true.
               go to 118
            endif
```

```
   50       continue
C ***** GET A TOTAL COUNT OF CLASSES AND STUDENTS.
  118    totclass = n
  100    continue
  110 totstuds = listuds + tvstuds + brstuds
      close (unit = 21)
C ***** WRITE CLASS SCHEDULE INFORMATION TO OUTPUT FILE PGECSUM.OUT.
      write (30, 1400) 'CLASS SCHEDULE INFORMATION:'
      write (30, *)
      write (30, 1100) 'CLASS', 'NUMBER', 'NUMBER OF '
      write (30, 1100) 'TYPE', 'OF CLASSES', 'STUDENTS'
      write (30, 1200) 'TV', tvclass, tvstuds
      write (30, 1200) 'LIVE', liclass, listuds
      write (30, 1200) 'BROAD.', brclass, brstuds
      write (30, *)    '_____'
      write (30, 1200) 'TOTAL', totclass, totstuds
 1000 format (t20, a)
 1100 format (t5, a, t15, a, t28, a)
 1200 format (t5, a, t15, i3, t28, i3)
 1400 format (///a)
      end
C     *****END SUBROUTINE "GETCLASS"**********************************
C
C ************BEGIN SUBROUTINE SEEROOMS**********************************
      subroutine seerooms (total)
      integer  total, n
      integer seats(50)
      character*(4) no(50)
      character*(2) type(50)
      common /comrm1/ no, type
      common /comrm2/ seats
C ***** WRITE ROOM INFO TO OUTPUT FILE "DATACHECK.OUT"
      write (31,*) 'Total rooms = ', total
      write (31, 2001) (no(n), type(n), seats(n), n = 1, total)
 2001 format(3x, a, 1x, a, 1x, i3)
      end
C ************END SUBROUTINE SEEROOMS**********************************
C
C ************BEGIN SUBROUTINE SEECLASS**********************************
      subroutine seeclass (total)
      integer total, num(200), rm2class(200), m
      real begtime(200), endtime(200)
      character*(15) id(200)
      character*(5) days(200), begin(200), end(200), station(200)
      character*(4) phone(200)
      character*(3) school(200)
      character*(2) type(200)
      logical rmnotfnd(200), notlocal(200)
      common /comclas1/ id, days, begin, end, school, type
      common /comclas2/ num, rmnotfnd, notlocal, rm2class
      common /comclas3/ begtime, endtime
      common /comclas4/ station, phone
C ***** WRITE CLASS SCHEDULE INFO TO OUTPUT FILE "DATACHEK.OUT"
      write (31,*) 'Total classes = ', total
      write (31, 2002) (id(m), days(m), begin(m), begtime(m), end(m),
```

```
     +          endtime(m), school(m), type(m), num(m),
     +          rmnotfnd(m), notlocal(m),
     +          station(m), phone(m), m = 1, total)
 2002 format (3x, a, 1x, a, 1x, a, 1x, f6.2, 1x,
     +        a,1x,f6.2,1x,a,1x,a,1x,i3,1x,l1,1x,l1,1x,a,1x,a)
      end
C ************END SUBROUTINE SEECLASS**************************************
C *****
C     *****BEGIN SUBROUTINE "SEATSORT"**************************************
C
C ***** ARRANGE THE VALUES IN roomno, roomtype INTO
C ***** INCREASING ORDER ACCORDING TO claseats.
C ***** NOTE:ARRAY SIZES SPECIFIED AS NUMBERS (NOT VARIABLES) DUE TO COMMONS.
      subroutine seatsort (n)
      integer n, claseats(50)
      character*(4) roomno(50)
      character*(2) roomtype(50)
      integer bottom, bigest
      integer locbig1
      common /comrm1/ roomno, roomtype
      common /comrm2/ claseats
      implicit none
      do 100 bottom = n,2,-1
C ***** FIND BIGGEST INTEGER IN claseats BETWEEN 1 AND bottom.
C ***** DECREMENT 'bottom' TO REFLECT NEW BOTTOM OF UNSORTED PORTION
C ***** OF ARRAYS.
        bigest = locbig1(claseats, bottom)
C ***** INTERCHANGE claseats(bottom) WITH claseats(bigest)
        call swichi(claseats(bottom), claseats(bigest))
C ***** TO AVOID MESSING UP THE CORRESPONDENCE BETWEEN VALUES IN claseats
C ***** AND THE OTHER ARRAYS, MAKE AN IDENTICAL INTERCHANGE TO EACH OF THOSE.
        call swichc(roomno(bottom), roomno(bigest))
        call swichc(roomtype(bottom), roomtype(bigest))
  100   continue
      end
C     *****END SUBROUTINE "SEATSORT"**************************************
C
C     *****BEGIN SUBROUTINE "STUDSORT"**************************************
C
C ***** ARRANGE THE VALUES IN classid, days, begin, end, school, clastype INTO
C ***** INCREASING ORDER ACCORDING TO studnum.
      subroutine studsort (m)
      integer m, studnum(200), rm2class(200)
      integer bottom, bigest
      integer locbig1
      character*(2) clastype(200)
      character*(3) school(200)
      character*(4) phone(200)
      character*(5) days(200), begin(200), end(200), station(200)
      character*(15) classid(200)
      logical rmnotfnd(200), notlocal(200)
      common /comclas1/ classid, days, begin, end, school, clastype
      common /comclas2/ studnum, rmnotfnd, notlocal, rm2class
      common /comclas4/ station, phone
      implicit none
```

```
      do 100 bottom = m, 2, -1
C ***** FIND BIGGEST INTEGER IN studnum BETWEEN 1 AND bottom.
C ***** DECREMENT 'bottom' TO REFLECT NEW BOTTOM OF UNSORTED PORTION
C ***** OF ARRAYS.
        bigest = locbig1(studnum, bottom)
C ***** INTERCHANGE studnum(bottom) WITH studnum(bigest)
        call swichi(studnum(bottom), studnum(bigest))
C ***** TO AVOID MESSING UP THE CORRESPONDENCE BETWEEN VALUES IN studnum
C ***** AND THE OTHER ARRAYS, MAKE AN IDENTICAL INTERCHANGE TO EACH OF THOSE.
        call swichc(classid(bottom), classid(bigest))
        call swichc(days(bottom), days(bigest))
        call swichc(begin(bottom), begin(bigest))
        call swichc(end(bottom), end(bigest))
        call swichc(school(bottom), school(bigest))
        call swichc(clastype(bottom), clastype(bigest))
        call swichc(station(bottom), station(bigest))
        call swichc(phone(bottom), phone(bigest))
        call swichl(rmnotfnd(bottom), rmnotfnd(bigest))
        call swichl(notlocal(bottom), notlocal(bigest))
  100   continue
      end
C     *****END SUBROUTINE "STUDSORT"**************************************
C
C     *****BEGIN SUBROUTINE "SWICHI"**************************************
C
C ***** INTEGER NUMBER SWITCH
      subroutine swichi (a,b)
      integer a,b
      integer copya
      copya = a
      a = b
      b = copya
      end
C     *****END SUBROUTINE "SWICHI"****************************************
C
C     *****BEGIN SUBROUTINE "SWICHC"**************************************
C
C ***** CHARACTER STRING SWITCH
      subroutine swichc (a,b)
      character*(*) a,b
      character*(15) copya
      copya = a
      a = b
      b = copya
      end
C     *****END SUBROUTINE "SWICHC"****************************************
C
C     *****BEGIN SUBROUTINE "SWICHL"**************************************
C
C ***** LOGICAL EXPRESSION SWITCH
      subroutine swichl (a,b)
      logical a,b
      logical copya
      copya = a
      a = b
```

```
      b = copya
      end
C     *****END SUBROUTINE "SWICHL"*************************************
C
C     *****BEGIN SUBROUTINE "CLASTIME"*********************************
      subroutine clastime (chartime, realtime, n, begtime)
      integer n, i, j, inthour, intmin, colonpos
      real realtime(n), begtime(n), hrtrunc
      character*(5) chartime(n)
      logical chrblk, nonxtcln, yesend
C ***** yesend INITIALLY SET AT .FALSE. THEN WILL BE TRUE UPON EXIT OF SUB.
C ***** THIS IS DONE TO DIFFERENTIATE BETWEEN begtime AND endtime CONVERSIONS.
      save yesend
      implicit none
      data yesend/.false./
C ***** LOOP FOR CONVERTING TIME OF EACH CLASS.
      do 100 i = 1, n
C ***** INITIALIZE THE TIME.
         inthour = 0
         intmin = 0
C ***** LOOP FOR EACH CHARACTER IN chartime.
         do 50 j = 1, 5
C ***** DEFINE LOGICALS THAT HELP DETERMINE HOURS AND MINUTES COMPONENTS.
            colonpos = index(chartime(i), ':')
            chrblk = chartime(i)(j:j) .eq. ' '
            nonxtcln = chartime(i)(j+1:j+1) .ne. ':'
C ***** IF CHARACTER IS BLANK, GO TO NEXT CHARACTER.
            if (chrblk) go to 50
C ***** IF CHARACTER IS TO THE LEFT OF COLON, RESULTING NO. IS PART OF HOUR.
            if (j .lt. colonpos) then
C ***** INTRINSIC FUNCTION "ICHAR" IS USED TO CONVERT CHARACTER TO AN INTEGER.
               inthour=inthour+ichar(chartime(i)(j:j))-ichar('0')
               if (nonxtcln) inthour = inthour * 10
C ***** IF CHARACTER IS TO THE RIGHT OF COLON, RESULTING NO.IS PART OF MINUTE.
            else if (j .gt. colonpos) then
               intmin=intmin+ichar(chartime(i)(j:j))-ichar('0')
               if (colonpos .eq. (j-1))
     +            intmin = intmin * 10
            endif
   50       continue
C ***** END OF LOOP FOR EACH CHARACTER IN chartime.
C ***** CONVERT FROM INTEGERS TO REAL TIME.
         realtime(i) = real(inthour) + real(intmin)/60.00
         hrtrunc = aint(realtime(i))
C ***** CONVERT TO MILITARY TIME BY LOOKING AT HOUR COMPONENT. MILITARY TIME
C ***** IS EASIER WHEN DETERMINING SCHEDULING CONFLICTS.
         if (hrtrunc .ge. 1.0 .and. hrtrunc .le. 9.0) then
            realtime(i) = realtime(i) + 12.0
C ***** IF TIME IS "10:XX" AND IT IS AN ENDING CLASS TIME, ADD 12 FOR MILITARY.
         else if (hrtrunc .eq. 10.0 .and. yesend) then
            if (aint(begtime(i)) .ne. 10.0) realtime(i)=realtime(i)+12.0
         endif
  100    continue
C ***** END OF LOOP FOR CONVERTING TIME FOR EACH CLASS.
      yesend = .true.
```

```
      end
C     *****END SUBROUTINE "CLASTIME"**********************************
C
C ************BEGIN SUBROUTINE MATCHUP**********************************
      subroutine matchup (totclass, totrooms, clasinrm, maxperwk,
     +                    eachday, totdays, percent, xtraamt, realperc)
      integer start, stop, incr, xtraamt
      integer i, totclass, studnum(200), claseats(50), totrooms
      integer totdays,classin,bgstclas,nxtclass,others,rm2class(200)
      integer thisroom, thisclas, maxperwk, clasinrm(totrooms,maxperwk)
      real begtime(200), endtime(200), realperc
      character*(1) today, eachday(totdays)
      character*(2) clastype(200), roomtype(50)
      character*(3) school(200)
      character*(4) roomno(50)
      character*(5) days(200), begin(200), end(200)
      character*(15) classid(200)
      logical sameday, sameschl, local, locloop, percent
      logical notlocal(200), rmnotfnd(200)
      common /comrm1/ roomno, roomtype
      common /comrm2/ claseats
      common /comclas1/ classid, days, begin, end, school, clastype
      common /comclas2/ studnum, rmnotfnd, notlocal, rm2class
      common /comclas3/ begtime, endtime
      implicit none
      save locloop
      data locloop/.false./
C ***** DIFFERENTIATES BETWEEN FAR AND LOCAL SCHOOL LOOOPS. CLASSES ARE
C ***** ADDRESSED IN A DIFFERENT ORDER FOR EACH TYPE.
      if (.not. locloop) then
         start = totclass
         stop = 1
         incr = -1
      else
         start = 1
         stop = totclass
         incr = 1
      endif
C ***** LOOP TO FIND LARGEST CLASSES WITHOUT ROOMS.
      do 100 thisclas = start, stop, incr
         local = .not. notlocal(thisclas)
         if (notlocal(thisclas) .and. rmnotfnd(thisclas)) then
C ***** THIS IS CURRENTLY THE BIGGEST NON-LOCAL CLASS W/O A ROOM.
            bgstclas = thisclas
C ***** FIND THE SMALLEST ROOM TO ACCOMODATE THE CLASS W/O A CONFLICT.
            call findroom(bgstclas, totrooms, clasinrm, maxperwk,
     +                    classin, thisroom, eachday, totdays,
     +                    percent, xtraamt, realperc)
            nxtclass = bgstclas - 1
C ***** START AN ADDITIONAL LOOP TO SEARCH FOR ALL SAME-SCHOOL, SAME-DAY,
C ***** CLASSES W/O A ROOM. LOOK AT ALL CLASSES SMALLER THAN bgstclas ABOVE.
            do 75 others = nxtclass, 1, -1
              sameschl = school(bgstclas) .eq. school(others)
              if (sameschl .and. rmnotfnd(others)) then
C ***** LOOP TO DETERMINE IF REMAINING CLASSES ARE ON SAME DAY AS bgstclas.
```

```
                  do 50 i = 1, totdays
                    today = eachday(i)
                    sameday = index(days(bgstclas),today) .ne. 0 .and.
     +                       index(days(others), today) .ne. 0
C ***** IF SAME DAY, THEN ADD IT TO THE LIST OF CLASSES TAUGHT IN THE ROOM THAT
C ***** WAS CHOSEN FOR bgstclas. THIS ENSURES THAT ALL CLASSES FOR A NON-LOCAL
C ***** SCHOOL TAUGHT THE SAME DAY ARE LOCATED IN THE SAME ROOM.
                    if (sameday) then
                      classin = classin + 1
                      rmnotfnd(others) = .false.
                      clasinrm(thisroom,classin) = others
                      rm2class(others) = thisroom
                      go to 75
                    endif
   50               continue
C ***** END OF SAME DAY SEARCH LOOP.
                  endif
   75             continue
C ***** END OF SAME SCHOOL SEARCH LOOP.
          else if (local .and. rmnotfnd(thisclas) .and. locloop) then
            bgstclas = thisclas
C ***** FIND THE SMALLEST ROOM TO ACCOMODATE THE CLASS W/O A CONFLICT.
            call findroom(bgstclas, totrooms, clasinrm, maxperwk,
     +                    classin, thisroom, eachday, totdays,
     +                    percent, xtraamt, realperc)
          endif
  100     continue
C ***** END OF LARGEST CLASS SEARCH LOOP.
        locloop = .true.
  200 end
C ************END SUBROUTINE MATCHUP**************************************
C *****
C *****
C ************BEGIN SUBROUTINE FINDROOM**********************************
C *****
      subroutine findroom(bgstclas, totrooms, clasinrm, maxperwk,
     +                    classin, thisroom, eachday, totdays,
     +                    percent, xtraamt, realperc)
      integer totrooms, studnum(200), bgstclas, claseats(50)
      integer thisroom, maxperwk, clasinrm(totrooms,maxperwk)
      integer classin, totdays, i, rm2class(200), counter
      integer addin, xtraamt, classize
      real begtime(200), endtime(200), realperc, realsize
      character*(1) today, eachday(totdays)
      character*(2) clastype(200), roomtype(50)
      character*(3) school(200)
      character*(4) roomno(50)
      character*(5) days(200), begin(200), end(200)
      character*(15) classid(200)
      logical sametype, bothtyli, later, earlier, sameday, percent
      logical rmnotfnd(200), notlocal(200), diffschl, loop2
      common /comrm1/ roomno, roomtype
      common /comrm2/ claseats
      common /comclas1/ classid, days, begin, end, school, clastype
      common /comclas2/ studnum, rmnotfnd, notlocal, rm2class
```

```
      common /comclas3/ begtime, endtime
      implicit none
      counter = 0
      add1in  = 1
      bothtvli = .false.
      loop2 = .false.
C ***** ADD THE APPROPRIATE NUMBER OF STUDENTS TO EACH EXISTING CLASS SIZE.
C ***** THIS IS BASED UPON THE USER'S INPUT FROM ON-LINE SCREEN PROMPTS.
      if (studnum(bgstclas) .eq. 0) then
         if (xtraamt .eq. 0) then
            classize = add1in
         else
            classize = xtraamt
         endif
      else if (percent) then
         realsize = real(studnum(bgstclas))*(1.00 + realperc)
         classize = int(realsize) + 1
      else
         classize = studnum(bgstclas) + xtraamt
      endif
C ***** LOOP TO SEARCH FOR THE SMALLEST ROOM TO ACCOMODATE THE CLASS.
    5 do 100 thisroom = 1, totrooms
         if (claseats(thisroom) .ge. classize) then
C ***** ENSURE THAT THE ROOM TYPE IS COMPATIBLE WITH THE CLASS TYPE.
            sametype = clastype(bgstclas) .eq. roomtype(thisroom)
            if (loop2) then
               bothtvli = clastype(bgstclas) .ne. 'BR' .and.
     +                    roomtype(thisroom) .eq. 'BO'
            endif
            if (sametype .or. bothtvli) then
C ***** GIVEN THAT ROOM SIZE AND ROOM TYPE IS GOOD, DETERMINE IF THERE IS A
C ***** CONFLICT WITH PRESENT CLASS AND THOSE ALREADY CHOSEN FOR THIS ROOM.
               do 50 classin = 1, maxperwk
C ***** IF NO CONFLICTS SO FAR, THEN PUT THE CLASS IN THIS ROOM.
                  if (clasinrm(thisroom, classin) .eq. 0) go to 200
                  do 25 i = 1, totdays
                   today = eachday(i)
                   sameday = index(days(bgstclas),today) .ne. 0 .and.
     +                   index(days(clasinrm(thisroom,classin)),today) .ne. 0
C ***** IF THERE IS A DAY CONFLICT, CHECK IF THERE IS A TIME CONFLICT.
                   if (sameday) then
                     if (notlocal(bgstclas)) then
                        diffschl=school(bgstclas) .ne.
     +                           school(clasinrm(thisroom,classin))
                        if (diffschl) go to 100
                     endif
C ***** THESE LOGICALS DEFINE THE EXTENT OF A CONFLICT; IF NONE EXISTS
C ***** THEN CHECK NEXT CLASS ALREADY CHOSEN FOR THE ROOM.
                     later = begtime(bgstclas) .gt.
     +                       endtime(clasinrm(thisroom,classin))
                     earlier = endtime(bgstclas) .lt.
     +                         begtime(clasinrm(thisroom,classin))
C ***** ONLY IF BOTH THE TIME AND DAY CONFLICT SHOULD THE ROOM BE REJECTED.
                     if (.not.(later .or. earlier)) go to 100
                   endif
```

```
   25              continue
   50             continue
C ***** END OF LOOP THAT CHECKS EACH OF ALREADY CHOSEN CLASSES FOR THE ROOM.
                  write(*,*) 'Warning! Max # of courses/room is exceeded'
                  stop
               endif
            endif
  100       continue
C ***** END OF ROOM SEARCH LOOP.
         if (rmnotfnd(bgstclas)) then
            counter = counter + 1
            loop2 = .true.
C ***** TELL USER THAT THERE IS NO ROOM THAT CAN ACCOMODATE THIS CLASS.
            if (counter .gt. 1) then
              write (*,*)'WARNING: Room not found for ',classid(bgstclas)
              go to 200
            endif
            go to 5
         endif
C ***** SINCE ROOM FITS ALL CONSTRAINTS, PLACE CLASS IN THE ROOM.
  200 rmnotfnd(bgstclas) = .false.
         classinrm(thisroom,classin) = bgstclas
C ***** rm2class LINKS THE CLASS WITH THE ROOM - USED FOR LATER SORTS
         rm2class(bgstclas) = thisroom
         end
C ************END SUBROUTINE FINDROOM**************************************
C *****
C **********BEGIN SUBROUTINE DAYGROUP*********************************
C *****
         subroutine daygroup(days, totclass)
         integer j, thisclas, totclass
         integer montot, tuetot, wedtot, thutot, fritot, sattot
         integer monday(250), tueday(250), wedday(250)
         integer thuday(250), friday(250), satday(250)
         character*(5) days(200)
         common /comday1/ montot, tuetot, wedtot, thutot, fritot, sattot
         common /comday2/ monday, tueday, wedday, thuday, friday, satday
         implicit none
         montot = 0
         tuetot = 0
         wedtot = 0
         thutot = 0
         fritot = 0
         sattot = 0
C ***** FOR EACH CLASS, PLACE IT IN THE PROPER DAY ARRAY AND KEEP A RUNNING TOTA
L.
C ***** CURRENTLY ACCOUNTS FOR A SIX DAY WEEK (MONDAY THRU SATURDAY)
         do 100 thisclas = 1, totclass
            do 50 j = 1, 5
               if (days(thisclas)(j:j) .eq. ' ') go to 50
               if (days(thisclas)(j:j) .eq. 'M') then
                  montot = montot + 1
                  monday(montot) = thisclas
               else if (days(thisclas)(j:j) .eq. 'T') then
                  tuetot = tuetot + 1
                  tueday(tuetot) = thisclas
```

```
            else if (days(thisclas)(j:j) .eq. 'W') then
               wedtot = wedtot + 1
               wedday(wedtot) = thisclas
            else if (days(thisclas)(j:j) .eq. 'R') then
               thutot = thutot + 1
               thuday(thutot) = thisclas
            else if (days(thisclas)(j:j) .eq. 'F') then
               fritot = fritot + 1
               friday(fritot) = thisclas
            else if (days(thisclas)(j:j) .eq. 'S') then
               sattot = sattot + 1
               satday(sattot) = thisclas
            endif
   50    continue
  100 continue
      end
C ************END SUBROUTINE DAYGROUP*****************************
C *****
C     *****BEGIN SUBROUTINE "DAYTIMES"*************************************
C
C ***** ARRANGE THE VALUES IN weekday arrays INTO
C ***** INCREASING ORDER ACCORDING TO time.
C ***** NOTE:ARRAY SIZES SPECIFIED AS NUMBERS (NOT VARIABLES) DUE TO COMMONS.
      subroutine daytimes (day,totinday,btime,etime,prntday,tvbfile,
     +              tvefile, genfile, percent, xtraamt, realperc)
      integer totinday, day(totinday), j, tv, live, broad, lasttv
      integer xtraamt
      integer locbig2
      integer bottom,bigest,studnum(200),rm2class(200),claseats(50)
      integer year, month, date, hour, minute
      real btime(200), etime(200), realperc
      character*(2) clastype(200), roomtype(50)
      character*(3) school(200)
      character*(4) roomno(50), phone(200)
      character*(5) begin(200), days(200), end(200), station(200)
      character*(10) tvbfile, tvefile, genfile
      character*(15) classid(200)
      character*(*) prntday
      logical rmnotfnd(200), notlocal(200), percent
      common /comclas1/ classid, days, begin, end, school, clastype
      common /comclas2/ studnum, rmnotfnd, notlocal, rm2class
      common /comclas4/ station, phone
      common /comrm1/ roomno, roomtype
      common /comrm2/ claseats
      common /comdate/ year, month, date, hour, minute
      implicit none
      tv = 0
      live = 0
      broad = 0
C ***** OPEN TWO OUTPUT FILES FOR EACH DAY OF THE WEEK:
C ***** TVB--TV CLASSES SORTED BY BEGINNING TIMES
C ***** GEN--GENERAL OUTPUT WHICH GIVES ALL CLASSES (TV, LIVE AND BROADCAST)
      open (unit=32,file='C:\PGEC\CLASSTVB\'//tvbfile,
     +      form='formatted',status='old')
      open (unit=33,file='C:\PGEC\CLASSGEN\'//genfile,
```

```
     +      form='formatted',status='old')
       do 100 bottom = totinday , 2, -1
C ***** FIND BIGGEST REAL IN btime BETWEEN 1 AND bottom.
C ***** DECREMENT 'bottom' TO REFLECT NEW BOTTOM OF UNSORTED PORTION
C ***** OF ARRAYS.
        bigest = locbig2(btime, day, bottom, totinday)
C ***** INTERCHANGE day(bottom) WITH day(bigest)
        call swichi(day(bottom), day(bigest))
  100   continue
        write (33,*)
        write (33,*) prntday,' CLASS SCHEDULE - TV, LIVE AND BROADCAST'
        write (33,*)'_____________________________________________________'
        write (33,*)
        write (33, 1001) 'AS OF ', month, '/', date, '/', year
        write (33, 1001) '      ', hour, ':', minute, ' HRS'
        write (33, *)
        write (33,850)'BEG','END','CLASSID','SCHL','CLASS',
     +                'NO. OF','ROOM','NO. OF'
        write (33,850)'TIME','TIME',' ',' ','TYPE',
     +                'STUDENTS','NO.','SEATS'
        write (33,*)
        write (32,*)
        write (32,*) prntday,' CLASS SCHEDULE - TV AND BROADCAST',
     +               ' (START TIMES)'
        write (32,*)'_____________________________________________',
     +              '_____________'
        write (32,*)
        write (32, 1001) 'AS OF ', month, '/', date, '/', year
        write (32, 1001) '      ', hour, ':', minute, ' HRS'
        write (32, *)
        write (32,900) 'BEG', 'CLASSID' , 'SCHL', 'ROOM', 'STA.',
     +                 'PHONE', 'NO. OF' , 'NO. OF'
        write (32,900) 'TIME', ' ', ' ', 'NO.', 'NO.',
     +                 'NO.', 'STUDENTS', 'SEATS'
        write (32,*)
C ***** WRITE DAY'S CLASS SCHEDULE. KEEP A RUNNING TOTAL ON DIFF'T TYPE CLASSES.
        do 300 j = 1, totinday
           if (btime(day(j)) .ne. btime(day(j-1))) then
              write (33,*)'-----------------------------------------',
     +                    '-----------------------------------'
           endif
           if (clastype(day(j)) .eq. 'LI') then
              live = live + 1
           else
              if (btime(day(j)) .ne. btime(day(lasttv))) then
                 write (32,*)'-----------------------------------------',
     +                       '-----------------------------------'
              endif
              lasttv = j
              write (32,1050) begin(day(j)), classid(day(j)),
     +              school(day(j)), roomno(rm2class(day(j))),
     +              station(day(j)), phone(day(j)), studnum(day(j)),
     +                        claseats(rm2class(day(j)))
              if (clastype(day(j)) .eq. 'BR') broad = broad + 1
              if (clastype(day(j)) .eq. 'TV') tv = tv + 1
```

```
            endif
            write (33,1000) begin(day(j)), end(day(j)), classid(day(j)),
     +                      school(day(j)), clastype(day(j)),
     +                      studnum(day(j)), roomno(rm2class(day(j))),
     +                      classeats(rm2class(day(j)))
  200       continue
            write (32,*) '----------------------------------------',
     +                   '----------------------------------'
            write (33,*) '----------------------------------------',
     +                   '----------------------------------'
            write (32,1500)'T.V. CLASSES TAUGHT ', prntday, ' = ', tv
            write (32,1500)'BROADCAST CLASSES TAUGHT ',prntday,' = ',broad
            write (32,*) 'ROOM SELECTIONS ARE BASED UPON INCREASE IN ',
     +                   'CLASS ENROLLMENT OF:'
            if (percent) then
               write (32,1550) int(realperc * 100.00), ' PERCENT OF ',
     +                         'CURRENT STUDENTS IN EACH CLASS.'
            else
               write (32,1550) xtraamt, ' STUDENTS IN EACH CLASS.'
            endif
            write (33,1500)'T.V. CLASSES TAUGHT ', prntday, ' = ', tv
            write (33,1500)'LIVE CLASSES TAUGHT ', prntday, ' = ', live
            write (33,1500)'BROADCAST CLASSES TAUGHT ',prntday,' = ',broad
            write (33,*) 'ROOM SELECTIONS ARE BASED UPON INCREASE IN ',
     +                   'CLASS ENROLLMENT OF:'
            if (percent) then
               write (33,1550) int(realperc * 100.00), ' PERCENT OF ',
     +                         'CURRENT STUDENTS IN EACH CLASS.'
            else
               write (33,1550) xtraamt, ' STUDENTS IN EACH CLASS.'
            endif
            close (unit = 32)
            close (unit = 33)
C ************SEPARATION BETWEEN BEGINNING / END TIME ROUTINES********
            tv = 0
            live = 0
            broad = 0
C ***** OPEN DAY'S CLASS SCHEDULE OUTPUT FILE:
C ***** TVE--TV CLASSES SORTED BY ENDING TIMES.
            open (unit=32,file='C:\PGEC\CLASSTVE\'//tvefile,
     +           form='formatted',status='old')
            do 200 bottom = totinday , 2, -1
C ***** FIND BIGGEST REAL IN etime BETWEEN 1 AND bottom.
C ***** DECREMENT 'bottom' TO REFLECT NEW BOTTOM OF UNSORTED PORTION
C ***** OF ARRAYS.
              bigest = locbig2(etime, day, bottom, totinday)
C ***** INTERCHANGE day(bottom) WITH day(bigest)
              call swichi(day(bottom), day(bigest))
  200       continue
            write (32,*)
            write (32,*) prntday,' CLASS SCHEDULE - TV AND BROADCAST',
     +                   ' (END TIMES)'
            write (32,*)'_____________________________________________',
     +                  '_____'
            write (32,*)
```

```
      write (32, 1001) 'AS OF ', month, '/', date, '/', year
      write (32, 1001) '      ', hour, ':', minute, ' HRS'
      write (32, *)
      write (32,900) 'END', 'CLASSID' , 'SCHL', 'ROOM', 'STA.',
     +               'PHONE', 'NO. OF' , 'NO. OF'
      write (32,900) 'TIME', ' ', ' ', 'NO.', 'NO.',
     +              'NO.', 'STUDENTS', 'SEATS'
      write (32,*)
      do 400 j = 1, totinday
         if (clastype(day(j)) .ne. 'LI') then
            if (etime(day(j)) .ne. etime(day(lasttv))) then
               write (32,*)'---------------------------------------',
     +                     '---------------------------------'
            endif
            lasttv = j
            write (32,1050) end(day(j)), classid(day(j)),
     +         school(day(j)), roomno(rm2class(day(j))),
     +         station(day(j)), phone(day(j)), studnum(day(j)),
     +                  classeats(rm2class(day(j)))
            if (clastype(day(j)) .eq. 'BR') broad = broad + 1
            if (clastype(day(j)) .eq. 'TV') tv = tv + 1
         endif
  400    continue
         write (32,*) '---------------------------------------',
     +                '---------------------------------'
      write (32,1500)'T.V. CLASSES TAUGHT ', prntday, ' = ', tv
      write (32,1500)'BROADCAST CLASSES TAUGHT ',prntday,' = ',broad
      write (32,*) 'ROOM SELECTIONS ARE BASED UPON INCREASE IN ',
     +             'CLASS ENROLLMENT OF:'
      if (percent) then
         write (32,1550) int(realperc * 100.00), ' PERCENT OF ',
     +                   'CURRENT STUDENTS IN EACH CLASS.'
      else
         write (32,1550) xtraamt, ' STUDENTS IN EACH CLASS.'
      endif
      close (unit = 32)
  850 format (t2,a,t10,a,t18,a,t38,a,t44,a,t53,a,t66,a,t75,a)
  900 format (t2,a,t9,a,t34,a,t39,a,t46,a,t54,a,t61,a,t72,a)
 1000 format (t2,a,t10,a,t18,a,t38,a,t44,a,t53,i3,t66,a,t75,i3)
 1001 format (t60, 2(a, i2), a, i4)
 1050 format (t2,a,t9,a,t34,a,t39,a,t46,a,t54,a,t61,i3,t72,i3)
 1500 format (t2, 3a, i3)
 1550 format (t2, i3, 2a)
      end
C     *****END SUBROUTINE "DAYTIMES"**************************************
C *****
C **********BEGIN SUBROUTINE RMGROUP*******************************
C *****
      subroutine rmgroup(totrooms, totclass, days, grpdbyrm,
     +                   clasinrm, maxperwk,
     +                   roomno, classid)
       integer j, totclass, maxperwk, totrooms, classin
       integer thisroom, thisclas
       integer monroom(50), tueroom(50), wedroom(50)
       integer thuroom(50), friroom(50), satroom(50)
```

```
      integer clasinrm(totrooms, maxperwk)
      integer grpdbyrm(totrooms, 6, 20)
      character*(4) roomno(50)
      character*(5) days(200)
      character*(15) classid(200)
      common /comrm3/ monroom,tueroom,wedroom,thuroom,friroom,satroom
      implicit none
C ***** FOR EACH CLASS, GROUP THEM BY ROOM, THEN BY DAY.
      do 100 thisroom = 1, totrooms
         monroom(thisroom) = 0
         tueroom(thisroom) = 0
         wedroom(thisroom) = 0
         thuroom(thisroom) = 0
         friroom(thisroom) = 0
         satroom(thisroom) = 0
C ***** SEARCHES THROUGH EACH ROOM TO EXAMINE ALL CLASSES BEING TAUGHT, THEN
C ***** DETERMINES THE DAY IT IS TAUGHT AND PUTS IT IN THE PROPER GROUP.
C ***** grpdbyrm IS A 3-D ARRAY THAT KEEPS TRACK OF CLASSES BY ROOM AND DAY.
         do 75 classin = 1, maxperwk
            thisclas = clasinrm(thisroom, classin)
            if (thisclas .eq. 0) go to 100
            do 50 j = 1, 5
               if (days(thisclas)(j:j) .eq. ' ') go to 50
               if (days(thisclas)(j:j) .eq. 'M') then
                  monroom(thisroom) = monroom(thisroom) + 1
                  grpdbyrm(thisroom,1,monroom(thisroom))=thisclas
               else if (days(thisclas)(j:j) .eq. 'T') then
                  tueroom(thisroom) = tueroom(thisroom) + 1
                  grpdbyrm(thisroom,2,tueroom(thisroom))=thisclas
               else if (days(thisclas)(j:j) .eq. 'W') then
                  wedroom(thisroom) = wedroom(thisroom) + 1
                  grpdbyrm(thisroom,3,wedroom(thisroom))=thisclas
               else if (days(thisclas)(j:j) .eq. 'R') then
                  thuroom(thisroom) = thuroom(thisroom) + 1
                  grpdbyrm(thisroom,4,thuroom(thisroom))=thisclas
               else if (days(thisclas)(j:j) .eq. 'F') then
                  friroom(thisroom) = friroom(thisroom) + 1
                  grpdbyrm(thisroom,5,friroom(thisroom))=thisclas
               else if (days(thisclas)(j:j) .eq. 'S') then
                  satroom(thisroom) = satroom(thisroom) + 1
                  grpdbyrm(thisroom,6,satroom(thisroom))=thisclas
               endif
   50       continue
   75    continue
  100 continue
      end
C ************END SUBROUTINE RMGROUP*******************************
C *****
C *****
C     *****BEGIN SUBROUTINE "RMTIMES"*************************************
C
C ***** ARRANGE THE VALUES IN grpdbyrm arrays BY
C ***** INCREASING ORDER ACCORDING TO time.
C ***** NOTE:ARRAY SIZES SPECIFIED AS NUMBERS (NOT VARIABLES) DUE TO COMMONS.
      subroutine rmtimes (totrooms, btime, grpdbyrm)
```

```
      integer totrooms, bottom, bigest, thisroom
      integer monroom(50), tueroom(50), wedroom(50)
      integer thuroom(50), friroom(50), satroom(50)
      integer grpdbyrm(totrooms,6,20)
      real btime(200)
      integer locbig3
      common /comrm3/ monroom,tueroom,wedroom,thuroom,friroom,satroom
      implicit none
C *****  FOR EACH ROOM, SORT THE CLASSES BEING TAUGHT EACH DAY BY ASCENDING ORDER

      do 100 thisroom = 1, totrooms
         do 10 bottom = monroom(thisroom), 2, -1
C *****  FIND BIGGEST REAL IN btime BETWEEN 1 AND bottom.
C *****  DECREMENT 'bottom' TO REFLECT NEW BOTTOM OF UNSORTED PORTION
C *****  OF ARRAYS.
         bigest = locbig3(btime,grpdbyrm,bottom,
     +                    thisroom,1,totrooms)
C *****  INTERCHANGE grpdbyrm(bottom) WITH grpdbyrm(bigest)
         call swichi(grpdbyrm(thisroom,1,bottom),
     +               grpdbyrm(thisroom,1,bigest))
   10    continue
         do 20 bottom = tueroom(thisroom), 2, -1
C *****  FIND BIGGEST REAL IN btime BETWEEN 1 AND bottom.
C *****  DECREMENT 'bottom' TO REFLECT NEW BOTTOM OF UNSORTED PORTION
C *****  OF ARRAYS.
         bigest = locbig3(btime,grpdbyrm,bottom,
     +                    thisroom,2,totrooms)
C *****  INTERCHANGE grpdbyrm(bottom) WITH grpdbyrm(bigest)
         call swichi(grpdbyrm(thisroom,2,bottom),
     +               grpdbyrm(thisroom,2,bigest))
   20    continue
         do 30 bottom = wedroom(thisroom), 2, -1
C *****  FIND BIGGEST REAL IN btime BETWEEN 1 AND bottom.
C *****  DECREMENT 'bottom' TO REFLECT NEW BOTTOM OF UNSORTED PORTION
C *****  OF ARRAYS.
         bigest = locbig3(btime,grpdbyrm,bottom,
     +                    thisroom,3,totrooms)
C *****  INTERCHANGE grpdbyrm(bottom) WITH grpdbyrm(bigest)
         call swichi(grpdbyrm(thisroom,3,bottom),
     +               grpdbyrm(thisroom,3,bigest))
   30    continue
         do 40 bottom = thuroom(thisroom), 2, -1
C *****  FIND BIGGEST REAL IN btime BETWEEN 1 AND bottom.
C *****  DECREMENT 'bottom' TO REFLECT NEW BOTTOM OF UNSORTED PORTION
C *****  OF ARRAYS.
         bigest = locbig3(btime,grpdbyrm,bottom,
     +                    thisroom,4,totrooms)
C *****  INTERCHANGE grpdbyrm(bottom) WITH grpdbyrm(bigest)
         call swichi(grpdbyrm(thisroom,4,bottom),
     +               grpdbyrm(thisroom,4,bigest))
   40    continue
         do 50 bottom = friroom(thisroom), 2, -1
C *****  FIND BIGGEST REAL IN btime BETWEEN 1 AND bottom.
C *****  DECREMENT 'bottom' TO REFLECT NEW BOTTOM OF UNSORTED PORTION
C *****  OF ARRAYS.
         bigest = locbig3(btime,grpdbyrm,bottom,
```

```
     +                    thisroom,5,totrooms)
C ***** INTERCHANGE grpdbyrm(bottom) WITH grpdbyrm(bigest)
        call swichi(grpdbyrm(thisroom,5,bottom),
     +              grpdbyrm(thisroom,5,bigest))
   50   continue
        do 60 bottom = satroom(thisroom), 2, -1
C ***** FIND BIGGEST REAL IN btime BETWEEN 1 AND bottom.
C ***** DECREMENT 'bottom' TO REFLECT NEW BOTTOM OF UNSORTED PORTION
C ***** OF ARRAYS.
        bigest = locbig3(btime,grpdbyrm,bottom,
     +                   thisroom,6,totrooms)
C ***** INTERCHANGE grpdbyrm(bottom) WITH grpdbyrm(bigest)
        call swichi(grpdbyrm(thisroom,6,bottom),
     +              grpdbyrm(thisroom,6,bigest))
   60   continue
  100   continue
      end
C     *****END SUBROUTINE "RMTIMES"**************************************
C *****
C **********BEGIN SUBROUTINE PRNTRMS*******************************
C *****
      subroutine prntrms(totrooms, grpdbyrm, studnum,
     +                   percent, xtraamt, realperc)
      integer totrooms, i, j, totinrm
      integer thisroom, thisclas, xtraamt
      integer monroom(50), tueroom(50), wedroom(50)
      integer thuroom(50), friroom(50), satroom(50)
      integer studnum(200), claseats(50)
      integer grpdbyrm(totrooms, 6, 20)
      integer year, month, date, hour, minute
      real realperc
      character*(2) clastype(200), roomtype(50)
      character*(3) school(200)
      character*(4) roomno(50), x
      character*(5) begin(200), end(200), days(200)
      character*(15) classid(200)
      logical percent
      common /comclas1/ classid, days, begin, end, school, clastype
      common /comrm1/ roomno, roomtype
      common /comrm2/ claseats
      common /comrm3/ monroom,tueroom,wedroom,thuroom,friroom,satroom
      common /comdate/ year, month, date, hour, minute
      implicit none
C ***** FOR EACH ROOM, WRITE THE LIST OF CLASSES TAUGHT IN THAT ROOM.
C ***** THE OUTPUT IS SENT TO A SEPARATE FILE FOR EACH ROOM.
      do 100 thisroom = 1, totrooms
         x = roomno(thisroom)
         open (unit=30,file='C:\PGEC\ROOMOUT\'//x,
     +        form='formatted',status = 'old')
         write (30,*)
         write (30,1000)'ROOM ',x,
     +                  ' WEEKLY CLASS SCHEDULE'
         write (30,1000)'_____________________________________'
         write (30,*)
         write (30, 1001) 'AS OF ', month, '/', date, '/', year
```

```
      write (30, 1001) '      ', hour, ':', minute, ' HRS'
      write (30, *)
      write (30,1002)'BEG','END','CLASSID','SCHL','CLASS','NO. OF'
      write (30,1002)'TIME','TIME',' ',' ','TYPE','STUDENTS'
      write (30,*) '-----------------------------------------',
     +             '-----------------'
      write (30,*) 'MONDAY CLASSES'
C ***** WRITE SORTED LIST FOR ALL CLASSES TAUGHT IN EACH ROOM.
      do 10 thisclas = 1, monroom(thisroom)
         i = grpdbyrm(thisroom,1,thisclas)
         write (30,1003) begin(i), end(i), classid(i),
     +                   school(i), clastype(i), studnum(i)
   10    continue
      write (30,*) '-----------------------------------------',
     +             '-----------------'
      write (30,*) 'TUESDAY CLASSES'
      do 20 thisclas = 1, tueroom(thisroom)
         i = grpdbyrm(thisroom,2,thisclas)
         write (30,1003) begin(i), end(i), classid(i),
     +                   school(i), clastype(i), studnum(i)
   20    continue
      write (30,*) '-----------------------------------------',
     +             '-----------------'
      write (30,*) 'WEDNESDAY CLASSES'
      do 30 thisclas = 1, wedroom(thisroom)
         i = grpdbyrm(thisroom,3,thisclas)
         write (30,1003) begin(i), end(i), classid(i),
     +                   school(i), clastype(i), studnum(i)
   30    continue
      write (30,*) '-----------------------------------------',
     +             '-----------------'
      write (30,*) 'THURSDAY CLASSES'
      do 40 thisclas = 1, thuroom(thisroom)
         i = grpdbyrm(thisroom,4,thisclas)
         write (30,1003) begin(i), end(i), classid(i),
     +                   school(i), clastype(i), studnum(i)
   40    continue
      write (30,*) '-----------------------------------------',
     +             '-----------------'
      write (30,*) 'FRIDAY CLASSES'
      do 50 thisclas = 1, friroom(thisroom)
         i = grpdbyrm(thisroom,5,thisclas)
         write (30,1003) begin(i), end(i), classid(i),
     +                   school(i), clastype(i), studnum(i)
   50    continue
      write (30,*) '-----------------------------------------',
     +             '-----------------'
      write (30,*) 'SATURDAY CLASSES'
      do 60 thisclas = 1, satroom(thisroom)
         i = grpdbyrm(thisroom,6,thisclas)
         write (30,1003) begin(i), end(i), classid(i),
     +                   school(i), clastype(i), studnum(i)
   60    continue
      write (30,*) '-----------------------------------------',
     +             '-----------------'
```

```
          j = thisroom
          totinrm = monroom(j)+tueroom(j)+wedroom(j)+
     +              thuroom(j)+friroom(j)+satroom(j)
          write (30,1004) 'ROOM ', x, 'SUMMARY:'
          write (30,1004) 'ROOM TYPE = ', roomtype(thisroom)
          write (30,1005) 'SEATING CAPACITY =', classeats(thisroom)
          write (30,1005) 'TOTAL CLASS SESSIONS HELD DURING THE WEEK =',
     +                    totinrm
          write (30,*) 'ROOM SELECTIONS ARE BASED UPON INCREASE IN ',
     +                 'CLASS ENROLLMENT OF:'
          if (percent) then
             write (30,1550) int(realperc * 100.00), ' PERCENT OF ',
     +                       'CURRENT STUDENTS IN EACH CLASS.'
          else
             write (30,1550) xtraamt, ' STUDENTS IN EACH CLASS.'
          endif
             close (unit = 30)
   100    continue
  1000 format (t20, 3a)
  1001 format (t00, 2(a, i2), a, i4)
  1002 format (t2, a, t10, a, t18, a, t38, a, t44, a, t53, a)
  1003 format (t2, a, t10, a, t18, a, t38, a, t44, a, t53, i3)
  1004 format (t2, 3a)
  1005 format (t2,a, i4)
  1550 format (t2, i3, 2a)
       end
C ************END SUBROUTINE PRNTRMS*******************************
C *****
C     *****BEGIN FUNCTION "LOCBIG1"************************************
C
C ***** LOCATE BIGGEST INT. IN intarray BET. intarray(1) AND THE END OF ARRAY.
       integer function locbig1(intarray, bottom)
       integer bottom
       integer intarray(bottom)
       integer i
       locbig1 = 1
       do 100  i = 2,bottom
         if (intarray(i) .gt. intarray(locbig1)) then
           locbig1 = i
         endif
   100   continue
       end
C      *****END FUNCTION "LOCBIG1"************************************
C
C *****
C     *****BEGIN FUNCTION "LOCBIG2"************************************
C
C ***** LOCATE BIGGEST REAL IN relarray BET. relarray(1) AND THE END OF ARRAY.
       integer function locbig2(time, day, bottom, totinday)
       integer totinday
       integer bottom, day(totinday)
       real time(bottom)
       integer i
       locbig2 = 1
       do 100  i = 2,bottom
```

```
        if (time(day(i)) .gt. time(day(locbig2))) then
          locbig2 = i
        endif
  100   continue
      end
C     *****END FUNCTION "LOCBIG2"*************************************
C *****
C     *****BEGIN FUNCTION "LOCBIG3"*************************************
C
C ***** LOCATE BIGGEST REAL IN relarray BET. relarray(1) AND THE END OF ARRAY.
      integer function locbig3(time, grpdbyrm, bottom,
     +                         thisroom, thisday, totrooms)
      integer thisroom, thisday, bottom, totrooms
      integer grpdbyrm(totrooms,6,20)
      real time(bottom)
      integer i
      locbig3 = 1
      do 100  i = 2,bottom
        if (time(grpdbyrm(thisroom,thisday,i)) .gt.
     +      time(grpdbyrm(thisroom,thisday,locbig3))) then
          locbig3 = i
        endif
  100   continue
      end
C     *****END FUNCTION "LOCBIG3"*************************************
C *****
C  **************END OF SYSTEMS ENGINEERING FINAL PROJECT*****************
```

# APPENDIX C

## DIRECTOR'S LETTER OF APPRECIATION

**OLD DOMINION UNIVERSITY**

Peninsula Graduate Engineering Center
2713 Magruder Blvd. Suite D
Hampton, Virginia 23666
804-865-1777
FAX 804-594-7367

**TO WHOM IT MAY CONCERN:**

The purpose of this letter is to advise the reader of the usefulness to the Old Dominion University Peninsula Center of the computer program which was developed by Mr. Steve Macri to assist in the allocation and selection of the Center's classrooms. The program allows the Center staff to determine, based on the classroom size and location, and the time of day and length of each class, where each of our approximately 90 classes should be held.

Prior to the implementation of this program in the Fall, 1992 Semester, the allocation of classroom space was a very tedious process, which was done manually, and with considerable imprecision. At that time, there were only 10 classrooms, and our head count was approximately 520 students, including Virginia Tech and University of Virginia students.

The size of the Center, in physical area, number of classes, classrooms, and student head count, approximately doubled in the Fall of 1992. The system developed by Mr. Macri allowed the staff to save a considerable amount of time, allocating the classroom space with precision and accuracy.

We have been called upon to provide information to the Old Dominion University Board of Visitors and to accreditation teams visiting the University, using the program developed by Mr. Macri. In addition, other off-campus centers - the Old Dominion University/Norfolk State University Center in Virginia Beach, and the Tri-Cities Center in Portsmouth, have asked to use this valuable planning tool as they prepare for the Spring, 1993 Semester.

We are extremely grateful to Mr. Macri for making this program available to us . It has already proven to be of tremendous utility to the staff of the Peninsula Center and to the University, and I am certain that it will continue to be in the future.

Director