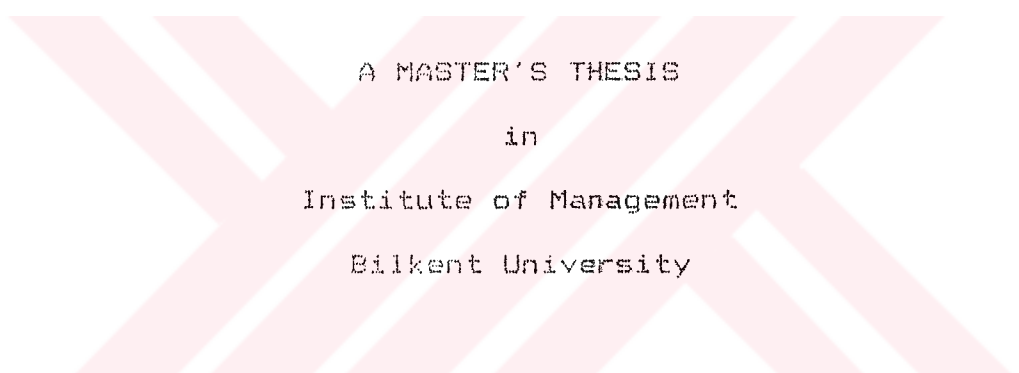


9333

A TIMETABLE SCHEDULING
COMPUTER PROGRAM



A MASTER'S THESIS
in
Institute of Management
Bilkent University

By
Mehmet Toptas
June 1990

I certify that I have read this thesis and that in my opinion it is fully adequate, in scope and quality, as a thesis for the degree of Master of Business Administration.

Erdal Erel

Asst. Prof. Dr. Erdal Erel

Supervisor

I certify that I have read this thesis and that in my opinion it is fully adequate, in scope and quality, as a thesis for the degree of Master of Business Administration.

Subidey Togan

Prof. Dr. Subidey Togan

I certify that I have read this thesis and that in my opinion it is fully adequate, in scope and quality, as a thesis for the degree of Master of Business Administration.

Kursat Aydogan

Assoc. Prof. Dr. Kursat Aydogan

Approved by the Graduate School of Business Administration

Prof. Dr. Subidey Togan

Subidey Togan

ABSTRACT

A TIMETABLE SCHEDULING COMPUTER PROGRAM

TOPTAS Mehmet

MBA in Institute of Management

Supervisor: Asst. Prof. Dr. Erdal Erel

June 1990

The use of computers makes easy the preparation of timetable schedule and eliminates a lot of manual work.

In this thesis, a timetable schedule generation program written in Fortran IV language and implemented in a Burroughs 9000 system is adapted to the Data General system at Bilkent University computing center. A case study on timetable schedule of Department of Management of Bilkent University is performed.

ÖZET

DERS ÇİZELGELERİ HAZIRLAYAN BİLGİSAYAR PROGRAMI

TOPTAS Mehmet

Yüksek Lisans Tezi, İşletme Enstitüsü

Tez Yöneticisi Yd. Doç. Dr. Erdal Erel

Haziran 1990

Bilgisayarların kullanımı ders çizelgelerinin hazırlamasını kolaylaştırmış ve bir çok el ile yapılan işi ortadan kaldırmıştır.

Bu tez çalışmasında FORTRAN IV bilgisayar dilinde yazılan ve Burroughs 9000 sisteminde çalıştırılan ders çizelgesi hazırlayan bir bilgisayar programı Bilkent Üniversitesi Bilgisayar Merkezinde bulunan Data General sistemine uyarlanmıştır. Bu program yardımıyla Bilkent Üniversitesi İşletme Bölümünün 1990-1991 öğretim yılı sonbahar dönemi ders çizelgeleri hazırlanmıştır.

ACKNOWLEDGEMENTS

I would like to express my thanks to my supervisor Asst. Prof. Dr. Erdal Erel for his kind supervision and helpfull comments.

I also appreciate Prof. Dr. Subidey Togan and Assoc. Prof Dr. Kursat Aydogan for their helps and suggestions.



TABLE of CONTENTS

ABSTRACT	i
ACKNOWLEDGEMENTS	ii
TABLE of CONTENTS	iii
1. INTRODUCTION	1
1.1. Introduction	1
1.2. Outline of the Study	2
2. LITERATURE SURVEY	4
2.1. Structural Classification	4
2.2. Classification According to Application Areas	5
2.3. Classification According to Conflict Handling	5
2.4. Classification According to the Solution Approaches	6
2.4.1. Early Work	6
2.4.2. Heuristics	7
2.4.3. Graph Theory	8
2.4.4. Mathematical Programming Models	8
3. PROPOSED SYSTEM	10
3.1. Conflicts of the Problem	11
3.2. Constraints of the Problem	12
3.2.1. Structural Constraints	12
3.2.2. Preference Constraints	13
3.3. Schedule Generation	14
3.4. Solution Procedure	16
3.4.1. Input Preparation	17
3.4.2. Decision Making	20
3.4.2.1. Construction of a Set of Timetables Feasible for Regular Students	20

3.4.2.2. Elimination of Infeasible Timetables Considering the Instructors	26
3.4.3. Evaluation of Timetables Considering the Irregular Students	27
3.4.4. Reports Generation	28
4. CASE STUDY	30
5. CONCLUSION	33
LIST of REFERENCES	38
APPENDICES	
A. CHANGES MADE in ADAPTING the PROGRAM to DATA GENERAL COMPUTER SYSTEM from BURROUGHS 9000 COMPUTER SYSTEM	
B. CASE STUDY OUTPUTS	
C. THE SOURCE PROGRAM	
D. THE INPUT FILE	

CHAPTER 1

INTRODUCTION

1.1. Introduction

The timetable scheduling problem has been investigated and approached from several different points of view. The common constraints of all approaches are as follows :

- (1) Structural requirement: No instructor or class can be assigned to more than one place in the same period.
- (2) Teaching requirement: Each instructor can meet each class a number of times predetermined for that instructor and that class.

But the approach and properties of the timetable depends heavily on the type of school and on the administrative characteristics.

The timetable schedule is the summary of the planned curriculum over a planning horizon. A course schedule answers questions such as which lecture will be given, which class, which lecturer, when and how long.

Manual course scheduling is possible for small educational systems. By intuition and experience, the scheduler can construct timetables manually. But larger the educational system gets, much more difficult this task will be. Furthermore, obtaining a good schedule by manual scheduling can result in solutions far

away from optimal or desired solutions. These drawbacks can be overcome by replacing the manual one with a computerized course scheduling system. Such a system can give more flexibility, make it easy to incorporate new requirements, save time and effort. Furthermore, the computer facilities can be used for various report generation like printing the timetables of classes, instructors and classrooms.

1.2. Outline of the Study

In the first chapter an introduction to the course scheduling concept is given. Importance and definition of course scheduling, drawbacks of manual scheduling, and an optimum course schedule concept are introduced.

Second chapter summarizes the historical background of the subject and gives the classification of timetabling algorithms in the literature.

In the third chapter, the heuristics proposed for constructing the timetables of an academic department by computer is discussed in detail. The proposed heuristic has three main stages. In the first stage alternative course schedules feasible for regular students are generated. In the second stage, among the schedules generated the ones that violate additional instructor and regular student constraints are eliminated. In the last stage, the remaining schedules are evaluated considering the desires and

needs of irregular students in order to obtain the final schedules for that semester.

A case study for 1990-1991 fall semester of Department of Management of Bilkent University is conducted and the results are given.

In the last chapter, the conclusions of the study are given and suggestions for an integrated information system and for further research are presented.



CHAPTER 2

LITERATURE SURVEY

The approaches in literature for solving timetabling problems can be classified according to their structure, their application areas, effort of handling conflicts and their solution approaches.

2.1. Structural Classification

Assignment Algorithms: These algorithms allocate classes in such a way that none of the constraints are violated. The main property of these is that the violation of a constraint is not accepted at any stage. However, when a constraint is violated, the algorithm may either backtrack on previous assignments or modifies a constraint, or refuse to assign the particular class. This type of algorithms were developed by Csima and Gotlieb (1964), Barraclough (1965), Lions (1967), Brittan and Farley (1971), Knauer (1974), Neufeld and Tartar (1974).

Improvement Algorithms: This type of algorithms attempt to resolve the conflicts of a schedule in which all meetings have been inserted but constraint violations such as unavailable classroom for some meetings exist. The main property of improvement algorithms is that they try to reduce the infeasibilities of a previously scheduled timetable by interchanging entries within the timetable. Smith (1975) and

and Aust (1976) developed this type of algorithms.

2.2. Classification According to Application Areas

Timetabling algorithms for educational systems are either for secondary schools or for college and universities. Most of the previous studies have dealt with secondary school timetabling. On the other hand studies dealing with college or university timetabling problems are very few.

Researchers that have dealt with school timetabling are Appleby (1960), Gottlieb (1965), Barroglough (1965), Csima (1965), Lions (1966,1967), Lawrie (1969), Smith (1975). Researchers that have dealt with collage and university timetabling are Almond (1965), Yule (1968), Brittan and Farley (1971), Akkoyunlu (1973) and Tripathy (1984).

2.3. Classification According to Conflict Handling

The amount of conflicts faced during timetabling is directly proportional to the amount of load on the resources. Algorithms that Do Not Avoid Conflicts: Yule (1968) described a system for university timetabling which takes no special steps to avoid conflicts. When a conflict occurs during scheduling, the process is stopped and restarted for scheduling the conflict causing item firstly.

Algorithms that Avoid Conflicts: The algorithms that avoid conflicts are developed by Gotlieb (1963,1964), Csima (1964) and Lions (1967). Conflicts are avoided where possible by careful checks before each requirement is assigned. When conflicts cannot be avoided the initial constraints are removed progressively until the difficulty is removed. Barroglough (1965) tried to remove conflicts when they occur. Entries causing the conflicts are displaced in the existing (partial) timetable.

Johnston and Wolfenden (1968) describe an algorithm such that requirements are fitted so as to minimize the risk of conflicts occurring and when conflicts occur, they are tried to be resolved. Another algorithm described by Britten and Farley (1971) is a powerful one to resolve conflicts.

2.4. Classification According to the Solution Approaches

2.4.1. Early Work

The early work on course scheduling is introduced by Gotlieb (1963). He proposes to construct a three-dimensional array, each point of which represents the meeting of a particular class with a particular instructor at a particular class with a particular instructor at a particular hour of the day.

In 1965 Csima and in 1966 Lions improved Gotlieb's method.

2.4.2. Heuristics

These are the simplest approaches since they are the computerization of manual timetabling methods. Barrouglough (1965) and Brittan and Farley (1971) introduced the interchange method.

The heuristic developed by Almond (1965, 1969) and Yule (1969) overcome the disadvantages of previous research work but they still have a number of disadvantages.

Almond (1965) proposed a simple heuristic method for university timetabling. All information is stored in two arrays: course requirement matrix and teacher availability matrix.

Yule (1969) proposed the file of requirement lines concept.

Class requirement and the timetable matrices are replaced by file requirement lines. The program tries to allocate these requirement lines to periods. One of the disadvantages of this heuristic method is that it does not detect cases where a solution does not exist until a loop of reordering of lines has occurred. A second disadvantage is, due to the constraints given by various instructors, that some lectures cannot be fitted anywhere in the timetable.

2.4.3. Graph Theory

Graphs and networks have proven to be useful in the formulation and solution of timetabling problems. Welsh and Powell (1967) presented their works. Neufeld and Tartar (1974) introduced a method in graph theory concept. But this theory does not provide an efficient algorithm which can be applied to an arbitrary timetable problem in order to determine the existence of a solution. In 1985 Werra reviewed some basic models on graphs on a theoretical basis.

2.4.4. Mathematical Programming Models

Since the beginning of 1970's researchers used mathematical programming models for solving timetabling problems. Integer linear programming formulations are made and they are attempted to be solved by various methods. Lawrie (1964) developed an integer linear programming model for school timetabling.

A partial solution is obtained and then completed by an enumerative procedure. The integer linear programming model developed by Akkoyunlu (1973) uses modified simplex algorithm for the solution method. Only a global optimum solution is obtained and classroom restriction is not taken into account. Smith (1975) introduced the integer linear formulation of Gottlieb's and Csima's methods. Konya (1978) proposed to use simplex method. Optimum solution is searched step by step by solving relatively small transportation problems. Tripatny (1984) introduced a solution for the timetabling problem which is

formulated as a large integer linear programming problem by Lagrangian relaxation.



CHAPTER 3

PROPOSED SYSTEM

In this section, the general characteristics of the proposed computerized course scheduling system are discussed.

The definitions of some concepts used in course scheduling terminology are as follows :

Period : Period is the smallest unit of time in the time table. Periods are numbered consecutively through the week.

Pre-assignment : When a certain instructor has to meet a certain class at a certain hour, then this situation is called pre-assignment.

Block period assignment : If total lecture hours of a course are scheduled to consecutive periods in the timetable, this is called block period assignment.

Pattern type : Total lecture hours of a course can be scheduled consecutively or can be distributed to the days of the week, which is called pattern type.

Instructors desire to have a concentrated schedule to the certain days of the week with adequate idle times in between their lectures in a day. Regular students desire to have their courses distributed evenly over the week with reasonable number of idle times between their lectures in a day. The desire of irregular students is to have a conflict free schedule that is distributed evenly over the week with reasonable number of idle times in

between their lectures in a day.

The desired case would be to satisfy all these objectives and make all participants comfortable with their schedules but since these objectives conflict with each other, in practice it is impossible to satisfy all of them fully. The objective of the course scheduling problem is to maximize the number of participants; students and instructors who are comfortable with their schedules.

3.1. Conflicts of the Problem

A person cannot be in more than one place at the same time. A conflict exist between two simultaneously scheduled courses, if one or more students must take both courses or both courses are given by the same instructor. Conflicts of the course scheduling problem are classified as instructor conflict, same level or different level conflict.

Instructor conflict is the allocation of an instructor in more than one place at the same period.

Same level conflict is the scheduling of the courses of same year students simultaneously.

Different level conflict is the scheduling of conflicting courses among levels at the same period. Conflicting courses among levels are the ones taken by irregular students.

Satisfaction of instructor and same level conflicts is essential during the solution of course scheduling problem but satisfaction of different level conflicts can be relaxed partially.

3.2. Constraints of the Problem

Constraints of the proposed system are either structural or dependent on preferences. Structural constraints are due to scarce resources and should be satisfied completely. On the other hand, preference constraints may be relaxed partially.

3.2.1. Structural Constraints

(1) Classroom constraint

Total number of lectures assigned to a period for all year classes cannot be greater than the available number of classrooms in that period.

(2) Administrative constraint

Since instructors are also involved in academic work, seminars and meetings, special periods and/or classrooms should be reserved for these kind of occasions. The assignment of these occasions are performed initially. So when the scheduling starts, some of the classrooms in some periods and the corresponding periods of the instructors are not available for another

assignment.

(3) Structural requirements of instructors and students

If an instructor, teaching part-time or not, is involved in another work, he/she may not be available in all periods. In that case the lecture hours given by these instructors should be pre-assigned. The pre-assignment of these kind of lectures will limit the number of available periods on hand.

An instructor should not meet the same undergraduate class third times in any half day, but he/she might meet a class in both the morning and the afternoon of one day.

The length of consecutive lectures of an instructor in a day must be kept within a reasonable limit. Thus no instructor should be asked to lecture for more than four hours per day.

Since the attendance of part-time students of graduate class is limited to few days of a week, certain graduate courses with related subject matter should be scheduled closely together.

3.2.2. Preference Constraints

Member of staff and students do not prefer lectures early in the morning and late in the afternoon.

For even distribution of courses, total lecture hours of each

course should be scheduled on two days of the week with one spare day in between.

All lectures should be given in the morning in preference to the afternoon.

For lunch break, everyday at least one lecture hour should be free.

Students do not prefer free hours in between lectures.

Since the instructors may devote one or more days to research and outside activities, they do not like their lecture hours distributed to all days of the week.

Instructors prefer specific periods of the week for lecturing.

3.3. Schedule Generation

The course scheduling problem is a version of n -job, m -machine job shop scheduling problem. If an analogy is made for a university department, jobs are the courses offered in that semester and/or year, and machines are the classrooms. Lecture hours are processing times of the jobs.

All courses to be given are known in advance. The classrooms are identical and number of classrooms available are smaller than

the total number of courses. Lecture hours for each course are known in advance. Planning horizon is a week.

Heuristic procedures for course scheduling combine intuitive appeal of manual methods with the fast speed of computers. Real timetabling problems cannot be handled either by mathematical programming models or by graph theoretic approach without making some assumptions. But constructing a schedule step by step by heuristics is usually able to handle all kinds of requirements that a real timetabling problem may have but heuristics have the drawback of giving suboptimal solutions.

The heuristic method proposed in this study will satisfy conflicts (instructor and same level) and generate schedules with the least violation of the constraints. The method has three main stages.

- (1) Constructing a set of alternative timetables feasible for regular students
- (2) Eliminating timetables which are infeasible for the instructors
- (3) Evaluating the remaining timetables according to irregular students.

In the first stage feasible schedules for regular students are generated considering the desires and needs of regular students and instructors. Instructors and same level conflicts are satisfied completely and constraints like classroom constraint,

administrative constraint, structural requirements of instructors, and preferences of instructors and students are tried to be satisfied. Whenever a constraint is violated that schedule is penalized.

Among the set of alternative feasible schedules generated in the first stage there may be some schedules which are infeasible when the additional constraints of the instructors and regular students are taken into account. In the second stage such infeasible schedules for the instructors and regular students are eliminated. This elimination process is carried out by manual inspection.

The remaining set of alternative feasible schedules at the end of second stage are suitable for regular students and instructors. In the third stage if a feasible schedule that satisfies the desires and needs of a irregular students can be found, it is the optimum schedule. But if a schedule does not exist the decision maker has to make an evaluation considering the conflicts of each irregular student.

3.4. Solution Procedure

The main stages of the solution procedure are input preperation, decision making stage , and generation of the reports.

3.4.1. Input Preparation

The main arrays used for data storage are: Class Requirements Matrix, Instructor Availability Matrix, and Conflict Matrix.

Each entry of Class Requirement Matrix represents total number of lecture hours each instructor is to meet each course in a week.

Since each instructor is assigned to a course, the dimension of this matrix is $n \times 1$, n being the number of courses. There is a course named 'idle' which is used whenever it is necessary to make no assignments in certain periods.

The entries of Instructor Availability Matrix indicate the availability of each instructor in each period. This matrix is $n \times t$; n being the course number and t being the period. The availability of each instructor is given by a set of preferences like "A, P, N, O and X". The priorities of these preferences are in decreasing order, "A" being the preference with the maximum priority.

The description of these preferences are:

A : that course should absolutely be scheduled to that period

P : that period is preferred

O : it makes no difference if the course of that instructor is scheduled to that period or not

N : that period is not preferred

X : that course should not be scheduled to that period.

Each instructor is asked to make his/her own choice for his/her lecture hours. They are asked to fill a blank timetable using the above symbols. It is not essential to use all the symbols. If an instructor is not available in period t the entry should be "X". On the other hand if she/he should absolutely lecture in certain hours due to a time limitation, pre-assignments are necessary. It means that entry should be "A". Remaining preferences do not impose any obligation on scheduling process. For example, members of staff do not like to lecture early in the morning. A priority "N" should be inserted in the instructor availability matrix for those instructors at that periods. On the other hand, if an instructor prefers afternoons for lecturing, "P" should be inserted to afternoons. Hence it can be concluded that instructor availability matrix reflects the preference constraints of each instructor.

The entries of Conflict Matrix are either zero or one. This matrix is an $n \times n$ matrix and shows if a course conflicts with another one. The entry is zero if course i conflicts with course j and one otherwise. Courses conflict with each other either when they are taken by the same year students or when they are given by the same instructor. So conflict matrix can absorb same level and instructor conflicts.

There is an additional array called Classroom Array which is

used to store number of classrooms available in each period.

The data file prepared can be seen in the appendix. The first row contains the number of courses for each year. As an example, 13, 16, 12, 10, and 4 show that, in the first year there are 13 courses, in the second year there are 16 courses including sections, and so on.

The second row contains the codes of the courses. There should be '0' at the end of this row, which denotes course IDLE.

The matrix succeeding above rows is the instructor availability matrix. The detailed information of this matrix was given above.

The next matrix is the conflict matrix, whose entries are either one or zero, depending on the case that courses conflict with each other or not.

Each row of the succeeding matrix show number of hour of each course in a week, instructor numbers and pattern of the each course, respectively.

The construction of the data file for other years are the same.

3.4.2. Decision Making

3.4.2.1. Construction of a Set of Timetables Feasible for Regular Students

The instructor availability matrix of dimension $n \times t$ is filled with preferences of the instructors. Starting with the first period, the rows of instructor availability matrix are searched in order to choose the course with the maximum priority. This is the first decision rule. This row search of the instructor availability matrix is performed by SEARCH subroutine.

A course with priority "A" is searched in the first period. If there is no such a course other priorities "P", "O", and "N" are tried consecutively. When none of these priorities are present in the first period it means that none of the instructors teaching these courses are available in this period (priority for all courses is "X"). In that case this period should be left idle so a course named IDLE is assigned.

If the maximum priority in the first period is "A" then it means that the corresponding course will absolutely be scheduled to that period. If the priority is "P" that period is preferred by that instructor. If "O", then the instructor is indifferent to have the course scheduled to that period or not. And if the priority is "N" that period is not preferred by the instructor.

If there is only one course in the first period with the

maximum priority, that course is scheduled to that period. But in case of ties another decision rule is used. This second decision rule is selecting one of the candidate courses randomly. The random selection is performed in subroutine RANDOM. The rest of the candidate courses are stored in an array to be retrieved from this array when necessary.

Either single or parallel course assignment case (for non-conflicting courses) is possible. This decision is given in CONFLICT subroutine.

When the SEARCH procedure is completed each alternative course which is not chosen is compared with the chosen course. This comparison is made in CONFLICT subroutine using conflict matrix. If the chosen course does not have the shared resources like instructors with the other candidate courses, the entry of the chosen courses, the entry of the chosen course with the candidate course in the conflict matrix is "one". This means that these two courses do not conflict with each other, so can be scheduled parallel in the same period. This is the parallel course assignment case. If the chosen course conflicts with all other alternative courses that are not chosen, then there is a single course assignment case. That is either the students or instructors are shared. Conflicts of regular students (same level conflict) and instructors (instructors conflict) are satisfied in this stage of the study.

PATTERN subroutine determines the distribution of total

lecture hours of each selected course through the lecture days of the week. Rather than assigning total lecture hours of a course to consecutive periods in a day, general approach is distributing them to periods of different lecture days. If this distribution is not done then the assignment type is named as block period assignment. Unless the opposite is stated, the lecture hours of undergraduate courses are distributed over the lecture days of the week. On the other hand, block period assignment for graduate courses is very common. Making block period assignment or not is determined by the instructor and this is fed into the system as an initial data.

It is desired to determine a pattern type for the chosen course that has a total lecture hours of three hours per week, and its instructor does not prefer block period assignment. In that case the pattern type is either 2+1 or 1+2.

Another constraint that may be faced is the period preferences of the instructors. It is supposed that there is no classroom constraint in first and second periods. Then first lecture hour of this course is assigned to the first period. But to lecture this course in the second period may not be preferred by its instructor or impossible. Then the chosen course is deleted. On the other hand, if the chosen course is to be scheduled to the second period with priority "N", then the course is assigned but a cost is incurred. This cost is equal to the number of periods with priority "N", that the chosen course is assigned. For this case, if the priority in the second period of

this course is "N", then the cost incurred is "one".

After the determination of the pattern type for the candidate course, the availability of classrooms is checked. This is done in CLASSROOM assignment subroutine. In case of available number of classrooms, these classrooms are preserved for the lecture hours of the candidate course(s). But if all classrooms are occupied then no assignment is made and those lecture hours are left idle.

Assignment of candidate course(s) to the empty timetable is performed in ASSIGNMENT subroutine. Two main events are performed in this subroutine. Either an assignment is made or an assignment is deleted. The courses with priority "A" are assigned to the desired periods without making any checks. Number of lecture hours assigned is equal to the pattern type of the course(s) which was (were) determined beforehand. A general approach is to leave at least one idle lecture hour for lunch break everyday. So if the lecture hours of the candidate course override the lunch hour, the assignment of this course is deleted. Moreover, during assigning these lecture hours if the end of the day is reached before the assignment is completed, again deletion is performed.

Updating procedure for the instructors is performed in INSTRUCTOR conflicts avoiding subroutine. If an instructor is teaching courses to more than one level of students then some precautions should be taken. After assigning the lecture hour(s) of such an instructor in one level, the corresponding periods in other levels of instructor availability matrix are updated. As an

example suppose that an instructor is lecturing to second and third year students. When the course is scheduled to some periods in the second year, the corresponding periods in the third year's instructor availability matrix should be found. Then previous priorities should be changed to "X", so that instructor will not be forced to lecture his/her second and third year courses in the same periods.

The updating of instructor availability matrix related to courses is done in TIMETABLE UPDATING subroutine. This updating is necessary for further assignments. Previously, a pattern type was determined for the selected course. This pattern type was the distribution of total lecture hours of a course over the days of the week. If the total lecture hours of each course is interfered by at least one free day the schedule obtained in the end will be an uniformly distributed one. In the light of this approach, when the assignment of a course according to the selected pattern type is performed, the class requirement matrix (giving the total number of lecture hours each course has) is checked and two specific precautions are taken.

After the assignment, total lecture hours is decreased by pattern type. If there is no lecture hours left, the priorities in the remaining hours of this assigned course are changed to "X" up to the end of the week meaning that the scheduling of this course is completed. Suppose that some lecture hours have been assigned, then total lecture hours have been decreased by pattern type and again some lecture hours are left. These remaining hours should

not be assigned before the day after therefore the available periods for assigning that course should be reduced. What is done is updating all the priorities up to the end of that day plus up to the end of the following day. All the priorities in those periods are changed to "X".

The assignment procedure described in detail in this section is applied in the same manner to all periods until the last period is reached or no courses are left for assignment. Then the whole procedure is repeated for another year until the weekly course schedules of all year are obtained. If there remains some courses unassigned when the end of the week is reached, then it means that subject to the given constraints a feasible solution does not exist. In such cases the constraints should be relaxed. Relaxing the preference constraints of the instructors is a suggested precaution in such cases. These preferences may be too demanding for the problem on hand, so they need to be smoothed.

Generating all possible schedules is computationally infeasible. Hence number of schedules to be generated for each year of regular students is left to the decision maker. Each course schedule set generated has a certain cost. This cost is the sum of all costs incurred to each year's schedule when the constraints are violated. The total cost of every schedule set will be one of the criteria for choosing the final schedule set in the end. If the cost incurring procedure is summarized, one will notice three types of cost incurring. A cost is incurred when:

- (1) a course with priority "A" is violated or
- (2) a course with priority "P" is violated or
- (3) a course with priority "N" is violated.

When a course with priority "A" is violated, it means that there is no available classrooms in those periods. Therefore that course cannot be assigned to those periods. This situation is either tolerated or classroom capacity should be increased. There are two types of cost incurring occasions to a timetable when the priority "P" is violated. Either there are many alternative courses with priority "P" and some of them are fathomed with the second decision rule or there is no available classroom for the rest of periods of a course which is a candidate for assignment process. Finally a cost is incurred when a course with priority "N" is violated. This violation occurs when the course chosen with the first decision rule among alternative courses have the priority "N". Other violation is due to the second decision rule. For example, using the second decision rule a pattern type of two periods is determined for a course. First period is preferred but if the following period is not preferred, the course has to be assigned to a period which is not desired.

3.4.2.2. Elimination of Infeasible Timetables Considering the Instructors

Timetables obtained in the previous section are feasible for regular students. Among these timetables obtained there may be some which are infeasible when the additional constraints of the

instructors are taken into account. For example, daily load of each instructor should be limited. Thus the timetables in which an instructor is asked to lecture for more than four hours per day are eliminated. Furthermore, instructors do not like their lecture hours distributed to all lecture days of the week due to other obligations. The timetables in which the lecture hours of an instructor are distributed to more than three lecture days of the week are eliminated. The elimination of these kind of timetables that are infeasible for the instructors are proposed to be done by manual inspection.

3.4.3. Evaluation of Timetables Considering the Irregular Students

All students do not follow the regular curriculum of a year. Some of them have to take courses from different years' curricula. If this is not the case, The scheduling process of each year would be easier because only the instructors and classrooms would be shared among levels. But at this stage in addition to instructor and classroom sharing, courses are shared among different years because irregular students take courses from these different years' curricula. As the number of irregular students of a department increases, handling the course scheduling system becomes more complex.

Each irregular student takes a set of courses from different years' curriculum. The problem at this stage of the study is to select the best timetable set among the available timetable sets

while considering the conflict irregular students. In the light of this, the main philosophy of the proposed approach is to compare the set of courses of an irregular student with each set of available timetables and determine the number of conflicting periods, then repeat the procedure for all irregular students.

A timetable set with minimum cost and no conflicts for all irregular students except one irregular at first sight may seem to be good candidate for selection. But if most of the courses of this irregular student conflict with each other, the decision maker should eliminate this alternative. On the other hand, the selection of a timetable set in which all the irregular students have some conflicting hours can be a better decision, or the selection of a timetable set with maximum total cost but relatively least amount of conflicts for all the irregulars can be a good decision.

3.4.4. Reports Generation

For each of the graduate and undergraduate class, the weekly timetable are obtained. These weekly timetables show which course will be given in which period to which group of students during a week.

CHAPTER 4

CASE STUDY

The adapted computerized course scheduling system proposed is carried out for generating the course schedules of 1990-1991 fall semester of Department of Management of Bilkent University. The required data to construct the timetable are the followings;

1. The number of instructors
2. The number of classes
3. The number of courses
4. The number of regular and irregular students
5. The number of lecture hours each day
6. The number of lecture days each week

Of these, the number of regular students and irregular students are not used since it is very difficult to find number of these students.

There are five different level of classes; first, second, third and fourth being undergraduate and the fifth one being the graduate class. All first year courses are nondepartmental courses.

There are four departmental courses and four nondepartmental courses in second year; of these five of them have two or three sections. For the third year courses, there are five

departmental and three nondepartmental courses. Four of these departmental courses have two sections.

For the fourth year, there are seven departmental and three nondepartmental courses; all are one section.

In the fifth year there are only five departmental courses.

The number of lecture periods in each day is nine, starting from 8.40 am until 5.30 pm. Each period is fifty minutes for lectures. For lunch break everyday 12.30 to 1.40 pm is generally left idle. Finally, each week has five lecture days.

All of the courses of the first year students are nondepartmental and students take these courses in large groups with other departments. In second, third and fourth year curricula, there are four or three nondepartmental courses. In this study, first year courses were taken as given since all instructors of these courses are almost part-time instructors. Part-time instructors are not available in all days and hours of the week. Therefore these courses have to be pre-assigned.

Second year courses are also pre-assigned since they are also taken by other departments. In the scheduling of these second year courses, it is tried that they do not conflict with first year courses.

The preferences of each instructor are input to the

instructor availability matrix of the computerized course scheduling system. These preferences are preference constraints of the instructors which are tried to be satisfied but are tolerated when necessary.

For academic meetings a pre-determined half day of every instructor can be kept idle in order to allow them to meet at that half day. This requirement is also input as a pre-assignment. Furthermore, special requirements of full-time instructors can easily be fed into the system as pre-assignment. For example, if an instructor does not want to have one free day in between his lectures but prefers them to be scheduled to two consecutive days of the week then this request can be handled in the pre-assignment routine.

After the pre-assignment process and inputting the preferences of the instructors are completed, conflict matrix is inputted. If one course has a conflict with another case, a zero is given. If it has not got a conflict, then a one is given.

Each instructor is given a different integer number. This number is '0' if he/she does not teach any other course to a different year. But if this not the case, a positive integer number is given to this instructor in all different levels.

When all data is entered, the program is run for the first stage of the assignment process. The scheduling of courses for all levels is performed in an increasing order. First the courses of

first year students, then second, third, fourth and fifth year students are scheduled. During the assignment process, maximum number of courses assigned concurrently for each class is two.

Since the program is written for a small department in METU, one should make some adjustments during the execution of the program. As an example, first year courses may have eight section for one course. Thus, one should assign only two section of this course and expand it into eight section after getting output. Then he/she should reduce the available number of classrooms in the second execution.

The program generates different schedules for different seed number. Therefore, in order to obtain a preferred timetable one may execute the program more than one with different seed numbers.

The generated outputs can be seen in Appendix B.

CHAPTER 5

CONCLUSION

In this study, the details of constructing a computerized coursescheduling system for an university department is discussed.

The data input to the computerized course scheduling system, processing methods of the data and finally results obtained from the system are presented in detail in previous chapters.

Timetabling problems can be very different between one school and another one; even in the same educational system. Therefore developing a universal timetabling problem which could be used everywhere is not reasonable.

This study proved that using the facilities of a computer during a course scheduling rather than manual scheduling saved time and effort. Furthermore, as the system is a large one, to control desires, needs, conflicts and constraints of each participant and in the end to obtain a good result manually is not possible. The proposed computerized course scheduling system introduced flexibility, made the job of the scheduler very much easier than the old manual system, and the results obtained were better than the results obtained by manual scheduling.

5.2. A Proposed Integrated Information System and Further Research

Designing a computerized course scheduling sub-system for an educational system and implementation of this sub-system independently is not adequate because an educational system has many other subsystems and all these sub-systems work more effectively together in the system than if they were operating independently. This complete system is called an integrated information system for an educational system. The specific objectives of a computerized integrated information system are to provide information for decision making on planning, organizing, and controlling major activities of the system, and initiating action. Main steps in designing such a computerized integrated information are to design and computerize each sub-system and construct the interactions among them. In this section, the necessary files for constructing the complete information system are given. Designing other sub-systems, constructing the interactions among them, finally designing the complete computerized integrated information system are left to a further research.

The proposed files for the efficient operation of computerized integrated information system are:

- (1) students records file
- (2) faculty members file
- (3) examinations file
- (4) budget and purchasing information file

- (5) department periodicals file
- (6) seminars file
- (7) research activities file
- (8) planning decisions file
- (9) department general administrative works file
- (10) other departments information file

Processing all data stored in those files using the facilities of a computer is more easier than processing them manually. Because there are large volume of data elements involved in the system, and required data processing operations are complex. Furthermore, there is a processing time constraint; amount of time permitted between when the data are available to be recorded and when the information is required is limited. These are sufficient reasons for proposing a computerized data processing method.

LIST of REFERENCES

1. Akkoyunlu, E.A (1973). "A Linear Algorithm for Computing the Optimum University Timetabling", The Computer Journal, vol.16, pp.347-350.
2. Almond, M. (1965). "An Algorithm for Constructing University Timetables", The Computer Journal, vol.8, pp.331-340.
3. Almond, M. (1969). "A University Faculty Timetable", The Computer Journal, vol.12, pp.215-217.
4. Aust, R.J. (1976). "An Improvement Algorithm for School Timetabling", The Computer Journal, Vol.19, no.4, pp.339-343.
5. Barraclough, E.(1965). "The Application of a Digital Computer to the Construction of Timetables", The Computer Journal, vol.8, pp.136-146.
6. Csima, J. and Gottlieb, C.C (1964). "Tests on a Computer Method for Constructing School Timetables", Communications of the Association for Computing Machinery, vol.7,no.3,pp.160-163.
7. Gottlieb, C.C (1963). "The Construction of Class-teacher Timetables", Proceeding of IFIP congress 1962, North-Holland pub. Co.,Amsterdam, pp.73-77.
8. Knauer, B.A.(1974). "Solution of a Timetable Problem", Computers and Operation Research, vol.1,pp.363-365.
9. Konya, I., Somogyi, P and Szabados, T. (1978). "A Method of Timetabling Construction by a Computer", Periodica Polytechnica, vol.22, pp.171-181.
10. Lawrie, N.L. (1969). "An Integer Linear Programming Model of a School Timetabling Problem", The Computer Journal, vol.12, pp.307-316.

11. Lions, J. (1966). "A Counter Example for Gottlieb's Method for the Construction of School Timetables", Communication of the Association for Computing Machinery, vol.9, no.9, pp.697-698.
12. Lions, J. (1967). "The Ontario School Scheduling Problem", The Computer Journal, vol.10, pp.14-21
13. Neufeld, G.A. and Tartar, J. (1974). "Graph Colouring Conditions for the Existence of solutions to the Timetable Problem", Communications of the Association for Computing Machinery vol.17, no.8, pp.450-453.
14. Ozerman, Pinar (1985). " Computerized Timetable Schedule Generation", M.S. Thesis, METU.
15. Schmidt, G., and Strohlein, T. (1980). "Timetable Construction an Annotated Bibliography", The Computer Journal, vol.23, no.4, pp.307-316.
16. Smith, G. (1975). "On Maintenance of the Opportunity List for Class-Teacher Timetable Problems", Communications of the Association for Computing Machinery, vol.18, no.4, pp.203-208.
17. Tripathy, A. (1984). "School Timetabling. A Case in Large Binary Integer Linear Programming", Management Science, vol.30, no.2, pp.1473-1489.
18. Welsh, D.J.A. and Powell, M.E. (1967). "an Upper Bound for the Chromatic Number of a Graph and its Application to Timetabling Problems", The Computer Journal, vol.10, pp.85-86.
19. Yule, A.p. (1968). "Extensions to the Heuristic Algorithm for University Timetables", The Computer Journal, vol.10, pp.360-364.

CHANGES MADE in ADAPTING the PROGRAM
to DATA GENERAL from BURROUGHS 9000

Running the program which is written in Fortran IV for Burroughs 9000 system in Data General system is not possible because of two main reasons.

- (1) There is no Fortran IV compiler in Data General system. Data General computer system possesses Fortran 77 compiler.
- (2) There are some differences in statements, statement declarations and statement executions.

The followings are the changes made in adapting the program from Burroughs 9000 system to Data General system.

1. Variable declarations in the main program moved so that they precede file declarations.

2. In Burroughs system, it is allowed to use ';' in between two statement, but not in DG. They are modified in lines so as to prevent the error.

3. File declarations;

* FILE 1 (KIND=REMOTE)

This statement shows output file which writes its output at console (ie, screen). This is changed to:

OPEN (6,FILE='OUTPUT') or OPEN (6,FILE='@LIST')

depending on the purpose whether to get output on screen or as hardcopy, respectively.

* FILE 2 (KIND=REMOTE)

This is the input file which reads from screen. New file declaration:

OPEN (5,FILE='@INPUT')

* FILE 3 (KIND=DISK, PROTECTION=SAVE, NEWFILE)

This is used for opening a new file which is to be stored on disk. The equivalent one in DG is:

OPEN (3,FILE='SCRATCH', STATUS='OLD')

But this statement is written as command since the function of this file is achieved through opening and closing file within the program.

* FILE 4 (KIND=DISK, FILETYPE=7)

In DG,

OPEN (4, FILE='SCRATCH', STATUS='FRESH', RECF='DYNAMIC')

Since in DG system, a dynamic file should possess unformatted data, this file is opened and closed within the program.

* FILE 5 (KIND=DISK, TITLE='DATAM', FILETYPE=7)

This is the input file, being the most important one among files. It contains instructor availability matrix, classroom availability matrix, pattern type, number of course hours and conflict matrix. The equivalent DG system file is:

```
OPEN (1, FILE='DATAM', STATUS='OLD')
```

```
* FILE 6 (KIND=DISK, TITLE='OUT", PROTECTION=SAVE)
```

This is the file which contains pre-assignments and nondepartmental courses. The equivalent one in DG is:

```
OPEN (2, FILE='OUT', STATUS='NEW')
```

This is also made an command file for the reasons mentioned above.

```
* FILE 7 (KIND=DISK, TITLE='IRREGULAR', FILETYPE=7)
```

Irregular student name and courses are kept in this file in order to obtain the least conflicting output. The equivalent one in DG is :

```
OPEN (7, FILE='IRREGULAR', STATUS='OLD', PAD='YES')
```

4. In Burroughs system, the unformatted data is read by the following statement:

```
READ /,
```

Whereas in DG it should be :

```
READ (1,*),
```

Thus all statements containing '/' for reading unformatted data were changed to '*'. Also,

```
PRINT /
```

statements were changed to,

```
PRINT *
```

5. The hexadecimal decleration

```
DATA FF/Z00000000000000C/
```

is used for outputting the program commands at screen. In WRITE

statements, FORMAT contains C1 for this purpose. In DG these C1 's were changed to A1.

6. CHANGE (3, TITLE=LAST)

statement was changed to

OPEN (4, FILE='LAST', STATUS='OLD')



T. C.
Yükseköğretim Kurulu
Dokümantasyon Merkezi

APPENDIX B



COURSE SCHEDULE FOR THE FIRST YEAR

	MON	TUE	WED	THU	FRI
08:40-09:30	0	0	0	0	0
	0	0	0	0	0
09:40-10:30	0	0	0	0	0
	0	0	0	0	0
10:40-11:30	0	0	0	0	0
	0	0	0	0	0
11:40-12:30	0	0	0	0	0
	0	0	0	0	0
12:40-13:30	0	0	0	0	0
	0	0	0	0	0
13:40-14:30	0	0	0	0	0
	0	0	0	0	0
14:40-15:30	0	0	0	0	0
	0	0	0	0	0
15:40-16:30	0	0	0	0	0
	0	0	0	0	0
16:40-17:30	0	0	0	0	0
	0	0	0	0	0

COURSE SCHEDULE FOR THE 2ND YEAR

	MON	TUE	WED	THU	FRI
08:40-09:30	0	0	0	0	0
	0	0	0	0	0
09:40-10:30	0	0	0	0	0
	0	0	0	0	0
10:40-11:30	0	0	0	0	0
	0	0	0	0	0
11:40-12:30	0	0	0	0	0
	0	0	0	0	0
12:40-13:30	0	0	0	0	0
	0	0	0	0	0
13:40-14:30	0	0	0	0	0
	0	0	0	0	0
14:40-15:30	0	0	0	0	0
	0	0	0	0	0
15:40-16:30	0	0	0	0	0
	0	0	0	0	0
16:40-17:30	0	0	0	0	0
	0	0	0	0	0

COURSE SCHEDULE FOR THE 3RD YEAR

	MON	TUE	WED	THU	FRI
08:40-09:30	31300	0	0	3134132	0
	0	0	0	0	0
09:40-10:30	31300	3132132	3134131	3133132	0
	0	0	0	3132131	0
10:40-11:30	3136132	3133131	3136132	3132132	0
	3136131	0	3136131	0	0
11:40-12:30	3136132	3133131	31300	3132132	0
	3136131	0	0	0	0
12:40-13:30	0	0	0	0	0
	0	0	0	0	0
13:40-14:30	3134131	3133132	95300	3133131	3134132
	0	0	0	0	3132131
14:40-15:30	3134131	3133132	94300	0	3134132
	0	0	0	0	3132131
15:40-16:30	0	97300	94300	0	0
	0	0	0	0	0
16:40-17:30	0	0	0	0	0
	0	0	0	0	0

COURSE SCHEDULE FOR THE 4TH YEAR

	MON	TUE	WED	THU	FRI
08:40-09:30	31400 0	31431 0	31433 0	31461 0	31401 0
09:40-10:30	31400 0	31431 0	31433 0	0 0	31403 0
10:40-11:30	31433 0	31461 0	31411 0	31431 0	31403 0
11:40-12:30	31403 0	31461 0	31403 0	31431 0	0 0
12:40-13:30	0 0	0 0	0 0	0 0	0 0
13:40-14:30	31411 0	97400 0	31401 0	0 0	94400 0
14:40-15:30	31411 0	95400 0	31401 0	0 0	94400 0
15:40-16:30	0 0	0 0	31400 0	0 0	0 0
16:40-17:30	0 0	0 0	0 0	0 0	0 0

COURSE SCHEDULE FOR THE 5TH YEAR

	MON	TUE	WED	THU	FRI
08:40-09:30	0	0	0	0	0
	0	0	0	0	0
09:40-10:30	31531	31541	31531	31541	0
	0	0	0	0	0
10:40-11:30	31531	31541	31521	31561	0
	0	0	0	0	0
11:40-12:30	31521	0	31521	0	0
	0	0	0	0	0
12:40-13:30	0	0	0	0	0
	0	0	0	0	0
13:40-14:30	0	0	0	0	31561
	0	0	0	0	0
14:40-15:30	0	0	0	0	31561
	0	0	0	0	0
15:40-16:30	0	0	0	0	0
	0	0	0	0	0
16:40-17:30	0	0	0	0	0
	0	0	0	0	0

APPENDIX C



```

COMMON TT(20,48,5),ROOM(45),ASSA(45,5),PASSA(45,5),COURSE(20,5),
*IINDEX(20,45,5),BAD(20,6,5),CONF(20,20,5),NC(20),NN(5),N,T,
*PERSUM(20,10,5),NCORSE(150),OPCODE(150,10),CONFCT(150),
*NAME(150,24),NUMBER(5),NOC(5)
character*1 ANSWER
CHARACTER*11 HOUR(9)
CHARACTER*3 GUN(5)
CHARACTER*2 MC1,MC2,MC3,MC4,MC
CHARACTER*5 LAST(1) /'LAST.'/
double precision FF
INTEGER T,PT,TTT,CLASS,DD,D,FLAG1,FLAG2,FLAG3,FLAG4,FLAG5,FLAG6,
*FLAG7,FLAG8,KPP,KPN,BAD,ROOM,ASSA,PASSA,CONF,COURSE,
*PERSUM,OPCOD,NOPCOD,OPCODE,DAY,TIME,PERIOD(10),HRS,SEED,TT,
*TCOST(5),TTCOST,DUMMYCEN,HARF
MC1='ST'
MC2='ND'
MC3='RD'
MC4='TH'
DATA (GUN(1),I=1,5) / 'MON','TUE','WED','THU','FRI' /
*(HOUR(I),I=1,9) / '08:40-09:30','09:40-10:30','10:40-11:30',
*'11:40-12:30','12:40-13:30','13:40-14:30','14:40-15:30',
*'15:40-16:30','16:40-17:30' /
DATA FF/Z'0000000000000C'/
FLAG1=0
FLAG2=0
FLAG3=0
FLAG4=0
FLAG5=0
FLAG6=0
FLAG7=0
FLAG8=0
NO=0
OPEN (6,FILE='@LIST')
OPEN (5,FILE='@INPUT')
/* OPEN (3,FILE='SCRATCH',STATUS='OLD')
/* OPEN (4,FILE='SCRATCH',STATUS='FRESH',RECFM='DYNAMIC')
OPEN (1,FILE='DATAM',STATUS='OLD')
/* OPEN (2,FILE='OUT',STATUS='NEW')
OPEN (7,FILE='IRREGULAR',STATUS='OLD',PAD='YES')
/* READING THE INPUTS FROM DATA FILE
READ(1,*)(NN(J),J=1,5)

```

```

                DO 100 J=1,5
                NNN=NN(J)
READ(1,*)(COURSE(N,J),N=1,NNN+1)
IF(NNN.EQ.0) GO TO 746
                DO 3 N=1,NN(J)
3                BAD(N,1,J)=COURSE(N,J)
746 DO 5 N=1,NN(J)+1
                READ(1,99) (TT(N,T,J),T=1,45)
99 FORMAT(45A1)
5 CONTINUE
                IF(NN(J).EQ.0) GO TO 100
                READ(1,*)((CONF(N,NK,J),N=1,NN(J)),NK=1,NN(J))
DO 61 T=46,48
61 READ(1,*)(TT(N,T,J),N=1,NN(J)+1)
100 CONTINUE
READ(1,*)(ROOM(T),T=1,45)
C* CHOOSING THE OPTION
59 WRITE(6,21) FF
21 FORMAT(A1,10X,'1-ENTER DATA FOR GENERATING TIMETABLE SET')
WRITE(6,22)
22 FORMAT(11X,'2-FIND THE CONFLICTS OF IRREGULAR STUDENTS')
WRITE(6,23)
23 FORMAT(11X,'3-EXIT')
WRITE(6,234)
234 FORMAT(10X,'CHOOSE YOUR OPTION : (1/2/3)')
READ(5,*) NUM
GO TO (468,503,559),NUM
C*
503 WRITE(6,599) FF
599 FORMAT(10X,A1,'PLEASE WAIT; CONFLICTS OF IRR. STS. ARE SEARCHED')
CALL IRREG
NIM=1
GO TO 69
C*
468 IF(NIP.NE.1) GO TO 24
NIP=0
WRITE(6,593)
593 FORMAT(10X,'YOU HAVE ENTERED OPTION 1 BEFORE')
WRITE(6,597)
597 FORMAT(/10X,'NOW GO ON TO FIND THE CONFLICTS OF IRRREG. STUDENTS')
WRITE(6,767)

```

```

767  FORMAT(10X,'SEND NULL INPUT TO CONTINUE')
      READ(5,50) NULL
      GO TO 59

C*
24   WRITE(6,25) FF
25   FORMAT(10X,A1,'SEED   (   OR = TO 13 DIGITS   )   =?')
      READ(5,*) SEED
      PRINT*, 'SEED=', SEED
307  WRITE(6,305) FF
305  FORMAT(10X,A1,'DO YOU WANT TO USE LAST DATA OF OPTION 1 ?
      * (Y/N)')
      READ(5,50) ANSWER
      GO TO 1234
      IF(ANSWER.EQ.'N') GO TO 306
      IF(ANSWER.EQ.'N') GO TO 306

C*
C*      READING THE LAST DATA ABOUT NON-DEPARTMENTAL COURSES
C*      AND PRE-ASSIGNMENTS FROM FILE
C*

      print*, 'ok up to here'
1234 OPEN (4, FILE='LAST', STATUS='old')
      DO 310 J=1,5
      IF(J.EQ.1) MC=MC1
      IF(J.EQ.2) MC=MC2
      IF(J.EQ.3) MC=MC3
      IF(J.EQ.4) MC=MC4
      IF(J.EQ.5) MC=MC4
      READ(4,*) NUMBER(J)
      IF(NUMBER(J).EQ.0) GO TO 355
      DO 313 IS=1,NUMBER(J)
      READ(4,*) NOPCOD
      READ(4,*) HRS
      IP=2
      DO 312 I=1,HRS
      READ(4,*) DAY, TIME
      PERIOD(I)=9*(DAY-1)+TIME
      IF(ASSA(PERIOD(I),J).NE.0) GO TO 315
      ASSA(PERIOD(I),J)=NOPCOD
      PERSUM(NN(J)+IS,1,J)=NOPCOD
      PERSUM(NN(J)+IS,IP,J)=PERIOD(I)
      PRINT*, 'PERSUM=', PERSUM(NN(J)+IS,IP,J)

```

```

      IP=IP+1
      GO TO 314
315  PASSA(PERIOD(I),J)=NOPCOD
      PERSUM(NN(J)+IS,1,J)=NOPCOD
      PERSUM(NN(J)+IS,IP,J)=PERIOD(I)
      PRINT*, 'PERSUM=', PERSUM(NN(J)+IS,IP,J)
      IP=IP+1
314  READ(4,50) ANSWER
      IF (ANSWER.EQ.'Y') ROOM(PERIOD(I))=ROOM(PERIOD(I))-1
      DO 311 N=1,NN(J)+1
      TT(N,PERIOD(I),J)=1HX
311  CONTINUE
312  CONTINUE
313  CONTINUE
355  READ(4,50) ANSWER
      IF(ANSWER.EQ.'N') GO TO 341
      READ(4,*) NOC(J)
      DO 318 IL=1,NOC(J)
      READ(4,*) OPCOD
      READ(4,*) HRS
      IP=2
      DO 317 I=1,HRS
      READ(4,*) DAY,TIME
      PERIOD(I)=9*(DAY-1)+TIME
      ASSA(PERIOD(I),J)=OPCOD
      NK=NN(J)+NUMBER(J)
      PERSUM(NK+IL,1,J)=OPCOD
      PERSUM(NK+IL,IP,J)=PERIOD(I)
C*  PRINT*, 'PERSUM=', PERSUM(NK+IL,IP,J)
      IP=IP+1
      DO 319 N=1,NN(J)+1
      TT(N,PERIOD(I),J)=1HX
319  CONTINUE
      ROOM(PERIOD(I))=ROOM(PERIOD(I))-1
317  CONTINUE
318  CONTINUE
341  READ(4,50) ANSWER
      IF(ANSWER.EQ.'N') GO TO 310
      READ(4,*) MDAY,MHR,MDUR
      MPER=9*(MDAY-1)+MHR
      DO 320 IH=MPER,MPER+MDUR-1

```

```

      DO 320 N=1,NN(J)
      TT(N,IH,J)=1HX
320  CONTINUE
310  CONTINUE
      GO TO 9

C*      INPUTTING THE DATA INTERACTIVELY FOR TT SET GENERATION
306  OPEN (3,FILE='out',status='new')
      DO 10 J=1,5
      IF(J.EQ.1) MC=MC1
      IF(J.EQ.2) MC=MC2
      IF(J.EQ.3) MC=MC3
      IF(J.EQ.4) MC=MC4
      IF(J.EQ.5) MC=MC4
58  WRITE(6,28) FF,J,MC
28  FORMAT(10X,A1,'NO. OF NON-DEPT. COURSES OF',2X,I1,1X,A2,2X,
* 'YEAR= ?')
      WRITE(6,54)
54  FORMAT(10X,'ENTER ZERO IF THERE IS NO NON-DEPT. COURSE')
      READ(5,*) NUMBER(J)
      WRITE(6,81) FF,NUMBER(J)
81  FORMAT(10X,A1,'NUMBER OF NON-DEPT. COURSES=',2X,I2)
      WRITE(6,52)
52  FORMAT(10X,'CORRECT ? (Y/N)')
      READ(5,50) ANSWER
      IF(ANSWER.EQ.'N') GO TO 58
      IF(ANSWER.NE.'Y') GO TO 58
      WRITE(3,*) NUMBER(J)
      IF(NUMBER(J).EQ.0) GO TO 55
      DO 13 IS=1,NUMBER(J)
67  WRITE(6,29) FF
29  FORMAT(10X,A1,'OPTIC CODE OF NON-DEPT. COURSE= ?')
      READ(5,*) NOPCOD
      WRITE(6,46) FF,NOPCOD
46  FORMAT(10X,A1,'OPTIC CODE OF NON-DEPT. COURSE=',2X,I7)
      WRITE(6,57)
57  FORMAT(10X,'CORRECT ? (Y/N)')
      READ(5,50) ANSWER
      IF(ANSWER.EQ.'N') GO TO 67
      IF(ANSWER.NE.'Y') GO TO 67
      WRITE(3,*) NOPCOD
79  WRITE(6,30) FF

```

```

30  FORMAT(10X,A1,'LECTURE HOURS= ?      (NO. OF HRS/WEEK)')
    READ(5,*) HRS
    IF(HRS.EQ.0) GO TO 79
    WRITE(6,68) FF,HRS
68  FORMAT(10X,A1,'LECTURE HOURS/WEEK=',2X,I2)
    WRITE(6,121)
121  FORMAT(10X,'CORRECT ?  (Y/N)')
    READ(5,50) ANSWER
50  FORMAT(A1)
    IF(ANSWER.EQ.'N') GO TO 79
    IF(ANSWER.NE.'Y') GO TO 79
    WRITE(3,*) HRS
    IP=2
    DO 12 I=1,HRS
84  WRITE(6,31) FF
31  FORMAT(14X,A1,'LECTURE DAY= ?,      LECTURE HOUR= ?')
    WRITE(6,116)
116  FORMAT(/10X,'ENTER 1,2,3,4 OR 5 FOR LECTURE DAY')
    WRITE(6,111)
111  FORMAT(16X,'1,2,...,8 OR 9 FOR LECTURE HOUR')
    READ(5,*) DAY,TIME
    WRITE(6,118) FF,DAY,TIME
118  FORMAT(10X,A1,'LECTURE DAY=',2X,I2,6X,'LECTURE HOUR=',2X,I2)
    WRITE(6,82)
82  FORMAT(10X,'CORRECT ?  (Y/N)')
    READ(5,50) ANSWER
    IF(ANSWER.EQ.'N') GO TO 84
    IF(ANSWER.NE.'Y') GO TO 84
    WRITE(3,*) DAY,TIME
    PERIOD(I)=9*(DAY-1)+TIME
    IF(ASSA(PERIOD(I),J).NE.0) GO TO 15
    ASSA(PERIOD(I),J)=NOPCOD
    PERSUM(NN(J)+IS,1,J)=NOPCOD
    PERSUM(NN(J)+IS,IP,J)=PERIOD(I)
    PRINT*, 'PERSUM=',PERSUM(NN(J)+IS,IP,J)
    IP=IP+1
    GO TO 14
15  PASSA(PERIOD(I),J)=NOPCOD
    PERSUM(NN(J)+IS,1,J)=NOPCOD
    PERSUM(NN(J)+IS,IP,J)=PERIOD(I)
C*  PRINT*, 'PERSUM=',PERSUM(NN(J)+IS,IP,J)

```

```

      IP=IP+1
14  WRITE(6,32) FF
32  FORMAT(10X,A1,'COURSE OCCUPIES A DEPARTMENTAL CLASSROOM ? (Y/N)')
      READ(5,50) ANSWER
      IF(ANSWER.EQ.'N') GO TO 16
      IF(ANSWER.NE.'Y') GO TO 14
      WRITE(3,50) ANSWER
      ROOM(PERIOD(I))=ROOM(PERIOD(I))-1
      GO TO 454
16  WRITE(3,50) ANSWER
454 DO 11 N=1,NN(J)+1
      TT(N,PERIOD(I),J)=1HX
11  CONTINUE
12  CONTINUE
13  CONTINUE
      WRITE(6,34) FF,J,MC
34  FORMAT(10X,A1,'ASSIGNMENT OF NON-DEPT. COURSES OF',2X,I1,1X,A2,
*2X,'YEAR IS COMPLETED')
55  WRITE(6,35)
35  FORMAT(10X,'ANY PRE-ASSIGNMENT ? (Y/N)')
      READ(5,50) ANSWER
      IF(ANSWER.EQ.'N') GO TO 414
      IF(ANSWER.NE.'Y') GO TO 55
      WRITE(3,50) ANSWER
86  WRITE(6,36) FF,J,MC
36  FORMAT(10X,A1,'NO OF COURSES PRE-ASSIGNED FOR THE',2X,I1,A2,2X,
*'YEAR= ?')
      READ(5,*) NDC(J)
      IF(NDC(J).EQ.0) GO TO 86
      WRITE(6,56) FF,NDC(J)
56  FORMAT(10X,A1,'NO OF COURSES TO BE PRE-ASSIGNED=',2X,I2)
      WRITE(6,119)
119 FORMAT(10X,'CORRECT ? (Y/N)')
      READ(5,50) ANSWER
      IF(ANSWER.EQ.'N') GO TO 86
      IF(ANSWER.NE.'Y') GO TO 86
      WRITE(3,*) NDC(J)
      DO 18 IL=1,NDC(J)
89  WRITE(6,38) FF
38  FORMAT(10X,A1,'OPTIC CODE FOR PRE-ASSIGNMENT= ?')
      READ(5,*) OFCOD

```

```

WRITE(6,87) FF,OPCOD
87  FORMAT(10X,A1,'OPTIC CODE FOR PRE-ASSIGNMENT=',2X,I7)
    WRITE(6,88)
88  FORMAT(10X,'CORRECT ? (Y/N)')
    READ(5,50) ANSWER
    IF(ANSWER.EQ.'N') GO TO 89
    IF(ANSWER.NE.'Y') GO TO 89
    WRITE(3,*) OPCOD
90  WRITE(6,39) FF
39  FORMAT(10X,A1,'LECTURE HOURS= ?      (NO OF HRS/WEEK)')
    READ(5,*) HRS
    IF(HRS.EQ.0) GO TO 90
    WRITE(6,91) FF,HRS
91  FORMAT(10X,A1,'LECTURE HOURS/WEEK=',2X,I2)
    WRITE(6,92)
92  FORMAT(10X,'CORRECT ? (Y/N)')
    READ(5,50) ANSWER
    IF(ANSWER.EQ.'N') GO TO 90
    IF(ANSWER.NE.'Y') GO TO 90
    WRITE(3,*) HRS
    IP=2
    DO 17 I=1,HRS
98  WRITE(6,40) FF
40  FORMAT(14X,A1,'LECTURE DAY= ?,      LECTURE HOUR= ?')
    READ(5,*) DAY,TIME
    WRITE(6,95) FF,DAY,TIME
95  FORMAT(10X,A1,'LECTURE DAY=',2X,I2,6X,'LECTURE HOUR=',2X,I2)
    WRITE(6,97)
97  FORMAT(10X,'CORRECT ? (Y/N)')
    READ(5,50) ANSWER
    IF(ANSWER.EQ.'N') GO TO 98
    IF(ANSWER.NE.'Y') GO TO 98
    WRITE(3,*) DAY,TIME
    PERIOD(I)=9*(DAY-1)+TIME
    ASSA(PERIOD(I),J)=OPCOD
    NK=NN(J)+NUMBER(J)
    PERSUM(NK+IL,1,J)=OPCOD
    PERSUM(NK+IL,IP,J)=PERIOD(I)
    PRINT*, 'PERSUM=', PERSUM(NK+IL,IP,J)
    IP=IP+1
    DO 19 N=1,NN(J)+1

```

```

      TT(N,PERIOD(I),J)=1HX
19  CONTINUE
      ROOM(PERIOD(I))=ROOM(PERIOD(I))-1
17  CONTINUE
18  CONTINUE
      GO TO 41
414  WRITE(3,50) ANSWER
41  WRITE(6,42) FF
42  FORMAT(10X,A1,'PRE-ASSIGNMENT FOR ACADEMIC MEETING ? (Y/N)')
      READ(5,50) ANSWER
      IF(ANSWER.EQ.'N') GO TO 405
      IF(ANSWER.NE.'Y') GO TO 41
      GO TO 107
405  WRITE(3,50) ANSWER
      GO TO 10
107  WRITE(3,50) ANSWER
      WRITE(6,44)
44  FORMAT(8X,'MEET DAY= ?,      MEET HOUR= ?,      MEET DURATION= ?')
      WRITE(6,101)
101  FORMAT(/10X,'ENTER 1,2,3,4 OR 5 FOR MEETING DAY')
      WRITE(6,102)
102  FORMAT(16X,'1,2,...,8 OR 9 FOR MEETING HOUR')
      WRITE(6,104)
104  FORMAT(16X,'1,2,...,8 OR 9 FOR MEETING DURATION')
      READ(5,*) MDAY,MHR,MDUR
      WRITE(6,105) FF,MDAY,MHR,MDUR
105  FORMAT(A1,10X,'MEETING DAY',5X,'=',2X,I2,/11X,'MEETING HOUR',4X,
      * '=',2X,I2,/11X,'MEETING DURATION=',2X,I2)
      WRITE(6,106)
106  FORMAT(10X,'CORRECT ? (Y/N)')
      READ(5,50) ANSWER
      IF(ANSWER.EQ.'N') GO TO 107
      IF(ANSWER.NE.'Y') GO TO 107
      WRITE(3,*) MDAY,MHR,MDUR
      MPER=9*(MDAY-1)+MHR
      DO 20 IH=MPER,MPER+MDUR-1
      DO 20 N=1,NN(J)
      TT(N,IH,J)=1HX
20  CONTINUE
10  CONTINUE

```

```

C*
  9  WRITE(6,120)
120  FORMAT(10X,'PLEASE WAIT; TIMETABLE SET IS BEING GENERATED')
C*
      DO 700 J=1,5
C*  CALL MAIN SUBROUTINE
      CALL MAIN(J,SEED)
700  CONTINUE
C*
C*      PRINTING THE GENERATED TIMETABLE SET
C*
      DO 800 J=1,5
      IF(J.EQ.1) MC=MC1
      IF(J.EQ.2) MC=MC2
      IF(J.EQ.3) MC=MC3
      IF(J.EQ.4) MC=MC4
      IF(J.EQ.5) MC=MC4
      WRITE(6,33) J,MC
33  FORMAT(1X,'TIMETABLE ARRAY OF',1X,I2,A2,1X,'YEAR')
      WRITE(6,43) ((TT(N,T,J),T=1,48),N=1,NN(J)+1)
43  FORMAT(/1X,45A1,3A3)
      WRITE(6,53) J,MC
53  FORMAT(///1X,'COURSE SCHEDULE FOR THE',1X,I2,A2,1X,'YEAR')
      WRITE(6,63) (GUN(I),I=1,5)
63  FORMAT(/23X,A3,8X,A3,8X,A3,8X,A3,8X,A3)
      DO 150 IT=1,9
      WRITE(6,73) HOUR(IT),(ASSA(T,J),T=IT,45,9)
73  FORMAT(/1X,A11,6X,5(I7,4X))
      WRITE(6,74) (PASSA(T,J),T=IT,45,9)
74  FORMAT(18X,5(I7,4X))
150  CONTINUE
      WRITE(6,83)
83  FORMAT(///1X,'COST ASSIGNED FOR EACH COURSE')
      WRITE(6,203)
203  FORMAT(/1X,10X,'TC(A)',3X,'TC(P)',3X,'TC(N)',3X,'ROW SUM')
      WRITE(6,93) ((BAD(N,L,J),L=1,5),N=1,NN(J))
93  FORMAT(//1X,I7,5X,I2,6X,I2,6X,I2,6X,I2)
      TDCOST(J)=0
      DO 153 N=1,NN(J)
153  TDCOST(J)=TDCOST(J)+BAD(N,5,J)
      WRITE(6,103)

```

```

103          FORMAT(///1X,'INDEX ARRAY')
        WRITE(6,113)((IINDEX(R,T,J),R=1,20),T=1,45)
113  FORMAT(/1X,A2,2X,19I2)
        WRITE(6,123)
123  FORMAT(/1X,'ADDRESSES OF THE ASSIGNED COURSES IN TIMETABLE')
        NT=NN(J)+NUMBER(J)+NOC(J)
        WRITE(6,133)((PERSUM(N,IP,J),IP=1,10),N=1,NT)
133  FORMAT(/1X,10I7)
800          CONTINUE
        TTCOST=TCOST(1)+TCOST(2)+TCOST(3)+TCOST(4)+TCOST(5)
        WRITE(6,163) TTCOST
163  FORMAT(/1X,'TOTAL COST ASSIGNED TO THIS TIMETABLE SET=',2X,I4)
        WRITE(6,173)
173  FORMAT(/1X,'NUMBER OF CLASSROOMS LEFT IN EACH PERIOD')
        WRITE(6,193) (GUN(I),I=1,5)
193  FORMAT(/18X,5(A3,2X))
        DO 183 IT=1,9
            WRITE(6,143) HOUR(IT),(ROOM(T),T=IT,45,9)
143          FORMAT(/1X,A11,6X,5(I3,2X))
183          CONTINUE
*
*
*
69  IF(NIM.EQ.0) GO TO 108
        WRITE(6,213)
213  FORMAT(/10X,'CONFLICT SEARCHING FOR IRREGULARS IS COMPLETED')
        GO TO 559
*
*
108  WRITE(6,109)
109  FORMAT(/10X,'GENERATION OF THE TIMETABLE SET IS COMPLETED')
        WRITE(6,756)
756  FORMAT(10X,'SEND NULL INPUT TO CONTINUE')
        READ(5,50) NULL
        NIP=1
        GO TO 59
559  STOP
        END
*
*
*
```

```

C* *****      MAIN LOGIC CONSTRUCTION SUBROUTINE
C*
      SUBROUTINE MAIN(J,SEED)
      COMMON TT(20,48,5),ROOM(45),ASSA(45,5),PASSA(45,5),COURSE(20,5),
      *IINDEX(20,45,5),BAD(20,6,5),CONF(20,20,5),NC(20),NN(5),N,T,
      *PERSUM(20,10,5),NCORSE(150),OPCODE(150,10),CONFCT(150),
      *NAME(150,24),NUMBER(5),NOC(5)
      INTEGER T,PT,TTT,CLASS,DD,D,FLAG1,FLAG2,FLAG3,FLAG4,FLAG5,FLAG6,
      *FLAG7,KPP,KPN,BAD,ROOM,ASSA,PASSA,COURSE,CONF,PERSUM,
      *OPCOD,NOPCOD,OPCODE,PERIOD,SEED,TT,HARF

C*
C*
C*
C* CHOOSING THE ROW IN THE TIMETABLE WITH MAX PRIORITY
C*
      IF(NN(J).EQ.0) GO TO 11
      T=1
      N=0
      D=1
      INC=1
      NNN=NN(J)
1    HARF='A'
C*    PRINT*, '
C*    PRINT*, '
C*    PRINT*, '
C*    PRINT*, 'T=',T,'D=',D
      CALL SEARCH(NNN,J,HARF,FLAG4,NO)
C*    PRINT*, 'GET[ 1'
      IF(FLAG4.NE.1) GO TO 9
      FLAG4=0
      HARF='P'
      CALL SEARCH(NNN,J,HARF,FLAG4,NO)
C*    PRINT*, 'GET[ 2'
      IF(FLAG4.NE.1) GO TO 50
      FLAG4=0
      HARF='O'
      CALL SEARCH(NNN,J,HARF,FLAG4,NO)
C*    PRINT*, 'GET[ 3'
      IF(FLAG4.NE.1) GO TO 50
      FLAG4=0
      HARF='N'

```

```

      CALL SEARCH(NNN,J,HARF,FLAG4,NO)
C*   PRINT*, 'GE@T[ 4'
      IF(FLAG4.NE.1) GO TO 50
      T=T+1
      GO TO 2

C*
C*
C*
C*   PRIORITY TYPE IS 'A'
C*
      9   IF(IINDEX(2,T,J).LE.1) GO TO 10
C*           CALL CONFLICT DETERMINATION SUB. FOR PRIORITY 'A'
C*
      CALL CONFLI(J,MM)
C*   PRINT*, 'GE@T[ 5: CONFLICT FOR A IS SEARCHED'
      GO TO 20

      10  N=IINDEX(3,T,J)
C*   PRINT*, 'N=',N
C*
C*
C*           CALL PATTERN DETERMINATION SUB.
C*
      20  CALL PATERN(J,PT,INC,NNN,MM,II,FLAG5,FLAG6,SEED)
C*   PRINT*, 'GE@T[ 6: PATTERN FOR A'
C*   PRINT*, 'PT=',PT
      IF(FLAG5.EQ.1) GO TO 70

C*
C*
C*           CALL CLASSROOM ASSIGNMENT SUB.
C*
      CALL ROOMS(J,PT,INC,II,FLAG1,FLAG3,FLAG6,FLAG7,CLASS)
C*   PRINT*, 'GE@T[ 7: ROOM SEARCH FOR A'
C*
C*
C*           CALL ASSIGNMENT SUB.
C*
      15  CALL ASSIGN(J,PT,INC,II,FLAG2,FLAG3,FLAG6,FLAG7)
C*   PRINT*, 'GE@T[ 8: ASSIGN'

```

```

C*
C*          ASSIGNMENT VERSUS DELETING
C*
IF(COURSE(N,J).EQ.0) GO TO 40
IF(FLAG2.EQ.1) GO TO 30

C*
C*          ASSIGNMENT IS DONE
C*
IF(FLAG6.EQ.1) GO TO 31

C*
C*          ASSINMENT IS DONE FOR SINGLE COURSE
C*
NON=1

C*
C*          SHARED INSTRUCTOR OR NOT
C*
IF(TT(N,47,J).EQ.0) GO TO 40

C*
C*
C*          CALL INSTRUCTOR CONFLICT SUB.
C*
45 CALL INSTOR(J,PT,FLAG6)
C*   PRINT*, 'GE@T[ 9: INSTOR'
C*
C*
C*
C*          CALL TIMETABLE UPDATING SUB.
C*
40 CALL TTUP(J,PT,INC,II,D,FLAG6)
C*   PRINT*, 'GE@T[ 10: TTUP'
C*   PRINT*, 'N=',N, 'NC(II+1)=',NC(II+1)
C*   N=0
C*   NC(II+1)=0
C*   IF(COURSE(N,J).EQ.0) GO TO 35

C*
C*          ASSIGNING COSTS TO THE COURSES
C*   IF(IINDEX(2,T-1,J).LT.1) GO TO 35
C*   IF(HARF.EQ.'P') GO TO 100
C*   IF(HARF.EQ.'N') GO TO 110
C*   GO TO 35

```

100 KPP=IINDEX(2,T-1,J)

GO TO 120

110 KPN=NON

120 IF(NON.EQ.1) GO TO 101

BAD(N,4,J)=BAD(N,4,J)+KPN

BAD(NC(II+1),4,J)=BAD(N,4,J)

GO TO 35

101 BAD(N,3,J)=BAD(N,3,J)+KPP

35 NON=0

FLAG6=0

GO TO 2

C*

C* ASSIGNMENT IS DONE FOR PARALLEL COURSES

C*

31 NON=2

C*

C* UPDATE THE INDEX ARRAY

C*

NM=IINDEX(2,T-1,J)

DO 84 KI=3,NM+3

IF(NC(II+1).EQ.IINDEX(KI,T-1,J)) GO TO 85

84 CONTINUE

85 INI=KI

DO 86 LI=INI,NM+3

86 IINDEX(LI,T-1,J)=IINDEX(LI+1,T-1,J)

C*

C* SHARED INSTRUCTOR OR NOT.

C*

IF((TT(N,47,J).EQ.0).AND.TT(NC(II+1),47,J).EQ.0) GO TO 40

GO TO 45

C*

C* DELETING IS DONE

C*

30 IF(FLAG6.EQ.1) GO TO 36

C*

C* DELETING IS DONE FOR SINGLE COURSE

C*

IF(IINDEX(2,T,J).NE.0) GO TO 32

GO TO 56

32 FLAG2=0

GO TO 2

```

C*
C*          DELETING IS DONE FOR PARALLEL COURSES
C*
36  IINDEX(2,T,J)=IINDEX(2,T,J)-1
    IF(IINDEX(2,T,J).EQ.0) GO TO 56
C*
C*          UPDATE THE INDEX ARRAY
C*
    NM=IINDEX(2,T,J)
    DO 94 IK=3,NM+3
      IF(NC(II+1).EQ.IINDEX(IK,T,J)) GO TO 95
94  CONTINUE
95  IIN=IK
    DO 96 IL=IIN,NM+3
96  IINDEX(IL,T,J)=IINDEX(IL+1,T,J)
    FLAG2=0
    FLAG6=0
    GO TO 2
C*
C*          NO ALTERNATIVE IS LEFT : SO ASSIGN IDLE
C*
56  N=NN'+1
    FLAG6=0
    FLAG2=0
    GO TO 15
C*
C*
C*
C*          CHECKING IF THERE ARE ALTERNATIVE COURSES (ANY TIE ?)
C*
50  IF(IINDEX(2,T,J).LE.1) GO TO 60
    IF(HARF.NE.'X') GO TO 130
    N=NNN+1
    GO TO 15
C*
C*
C*
C*          SELECT ONE OF THE ROWS RANDOMLY AND UPDATE IINDEX ARRAY
C*          DECISION RULE 2
C*
130  NM=IINDEX(2,T,J)

```

```

      IINDEX(2,T,J)=IINDEX(2,T,J)-1
      CALL RANDM(SEED,NM,IN)
      N=IINDEX(IN+2,T,J)
C*   PRINT*, 'N=', N
      DO 4 I=IN+2,NM+2
4     IINDEX(I,T,J)=IINDEX(I+1,T,J)
      GO TO 140

C*
C*
C*
C*           THERE IS NO TIE
C*
60    N=IINDEX(3,T,J)
C*   PRINT*, 'N=', N
140   IF(COURSE(N,J).EQ.0) GO TO 15

C*
C*
C*
C*           CALL CONFLICT DETERMINATION SUB.
C*
      CALL CONFLI(J,MM)
C*   PRINT*, 'GE@T[ 11: CONFLI'
C*
C*
C*
C*           CALL PATTERN DETERMINATION SUB.
C*
55    CALL PATERN(J,PT,INC,NNN,MM,II,FLAG5,FLAG6,SEED)
C*   PRINT*, 'GE@T[ 12: PATTERN'
C*   PRINT*, 'PT=', PT
      IF(FLAG5.EQ.1) GO TO 70

C*
C*
C*
C*           CALL CLASSROOM ASSIGNMENT SUB.
C*
      CALL ROOMS(J,PT,INC,II,FLAG1,FLAG3,FLAG6,FLAG7,CLASS)
C*   PRINT*, 'GE@T[ 13: ROOMS'
      IF(BAD(N,2,J).GT.0) GO TO 90
      IF(FLAG1.NE.1) GO TO 15
      FLAG1=0

```

```

90  N=NNN+1
C*  PRINT*, 'N=', N
    GO TO 15
70  FLAG5=0
    GO TO 1
2   IF(T.LE.45) GO TO 1
C*
C*
C*
C*          CALCULATE BAD POINTS FOR ALL COURSES
C*
    DO 12 N=1, NN(J)
12  BAD(N, 5, J)=BAD(N, 2, J)+BAD(N, 3, J)+BAD(N, 4, J)
C*
C*
C*
11  RETURN
    END
C*
C*
C*
C* *****  SUBROUTINE TO FIND THE COURSE(S) WITH MAX PRIORITY
C*
    SUBROUTINE SEARCH(NNN, J, HARF, FLAG4, NO)
    INTEGER T, FLAG4, TT, HARF
    COMMON TT(20, 48, 5), RDOM(45), ASSA(45, 5), PASSA(45, 5), COURSE(20, 5),
    *IINDEX(20, 45, 5), BAD(20, 6, 5), CONF(20, 20, 5), NC(20), NN(5), N, T,
    *PERSUM(20, 10, 5), NCORSE(150), OPCODE(150, 10), CONFCT(150),
    *NAME(150, 24), NUMBER(5), NOC(5)
    NO=0
C*          COLUMN SEARCH OF TT ARRAY
C*
    DO 5 K=1, NNN+1
    IF(TT(K, T, J).NE.HARF) GO TO 5
    NO=NO+1
    IINDEX(2+NO, T, J)=K
5   CONTINUE
    IF(NO.GT.0) GO TO 6
    FLAG4=1
    RETURN

```

```

C*          DETERMINATION OF PRIORITY TYPE AND ROW NUMBER
C*
6   IINDEX(1,T,J)=HARF
    IINDEX(2,T,J)=NO
    RETURN
    END

C*
C*
C*
C* *****   SUBROUTINE FOR FINDING THE CONFLICTING COURSES
C*
    SUBROUTINE CONFLI(J,MM)
    COMMON TT(20,48,5),ROOM(45),ASSA(45,5),PASSA(45,5),COURSE(20,5),
    *IINDEX(20,45,5),BAD(20,6,5),CONF(20,20,5),NC(20),NN(5),N,T,
    *PERSUM(20,10,5),NCORSE(150),OPCODE(150,10),CONFCT(150),
    *NAME(150,24),NUMBER(5),NOC(5)
    INTEGER T,R,CONF,TT,HARF
    DO 4 I=1,20
4   NC(I)=0
    NC(I)=N
    I=1
    NM=IINDEX(2,T,J)
    DO 5 R=3,NM+1
    IF(N.EQ.IINDEX(R,T,J)) GO TO 5
    NK=IINDEX(R,T,J)
    IF(CONF(N,NK,J).EQ.0) GO TO 5

C*          VECTOR OF NON-CONFLICTING COURSES WITH COURSE N
C*
    I=I+1
    NC(I)=NK
5   CONTINUE
    MM=I-1
    RETURN
    END

```

C*

C*

C*

C* ***** SUBROUTINE TO DETERMINE THE PATTERN TYPE

```

SUBROUTINE PATERN(J,PT,INC,NNN,MM,II,FLAG5,FLAG6,SEED)
INTEGER CLASS,PT,T,TTT,FLAG5,FLAG6,ROOM,BAD,SEED,TT,HARF

```

```

COMMON TT(20,48,5),ROOM(45),ASSA(45,5),PASSA(45,5),COURSE(20,5),
*IINDEX(20,45,5),BAD(20,6,5),CONF(20,20,5),NC(20),NN(5),N,T,
*PERSUM(20,10,5),NCORSE(150),OPCODE(150,10),CONFCT(150),
*NAME(150,24),NUMBER(5),NOC(5)

```

```

IF(MM.EQ.0) GO TO 5

```

```

C*          PATTERN CHECKS FOR PARALLEL COURSE CASE

```

```

FLAG6=1

```

```

DO 20 II=1,MM

```

```

IF((TT(N,46,J).EQ.1).AND.(TT(NC(II+1),46,J).EQ.1)) GO TO 1

```

```

IF((TT(N,46,J).EQ.2).AND.(TT(NC(II+1),46,J).EQ.2)) GO TO 2

```

```

IF((TT(N,46,J).EQ.3).AND.(TT(NC(II+1),46,J).EQ.3)) GO TO 3

```

```

IF(TT(N,46,J).EQ.TT(NC(II+1),46,J)) GO TO 4

```

```

20 CONTINUE

```

```

GO TO 5

```

```

3 IF(TT(N,48,J).NE.TT(NC(II+1),48,J)) GO TO 5

```

```

IF(TT(N,48,J).NE.3) GO TO 9

```

```

PT=3

```

```

GO TO 61

```

```

4 IF(TT(N,48,J).NE.TT(NC(II+1),48,J)) GO TO 5

```

```

IF(TT(N,48,J).EQ.3) GO TO 6

```

```

GO TO 2

```

```

C*          PATTERN CHECKS FOR SINGLE COURSE CASE

```

```

C*

```

```

5 FLAG6=0

```

```

II=0

```

```

IF(TT(N,46,J).EQ.1) GO TO 1

```

```

IF(TT(N,46,J).EQ.2) GO TO 2

```

```

IF(TT(N,46,J).EQ.3) GO TO 7

```

```

IF(TT(N,48,J).EQ.3) GO TO 6

```

```

GO TO 2

```

```

1 PT=1

```

```

RETURN

```

```

2 PT=2

```

```

GO TO 8

```

```

7 IF(TT(N,48,J).NE.3) GO TO 9

```

```

6 PT=3

```

```

GO TO 8

```

```

C*          DECIDING ON PT ; RANDOM SELECTION

```

```

C*

```

```

9 FLAG8=1

```

```

CALL RANDM(SEED,NM,PT)

```

```

      IF(PT.EQ.1) GO TO 1
      GO TO 2
8      IF(FLAG6.EQ.1) GO TO 61
C* CHECKING PT PERIODS IF ANY 'N' OR 'X' ASSIGNMENT FOR SINGLE COURSE
C*      AND 'A' ASSIGNMENT FOR OTHER COURSES
C*
      M=T+1
      MN=T+PT-1
      DO 50 TTT=M,MN
      IF(TT(N,TTT,J).EQ.N) BAD(N,4,J)=BAD(N,4,J)+1
      IF(TT(N,TTT,J).EQ.X) GO TO 52
51      DO 55 JK=1,NNN+1
      IF(JK.EQ.N) GO TO 55
      IF(TT(JK,TTT,J).EQ.A) GO TO 52
55      CONTINUE
50      CONTINUE
      RETURN
C*      UPDATING THE PRIORITY OF COURSE N IN PERIOD T
C*
52      TT(N,T,J)=1HX
      FLAG5=1
      RETURN
C* CHECKING PT PERIODS OF ANY 'N' OR 'X' ASSIGNMENT FOR TWO COURSES AND
C*      'A' ASSIGNMENT FOR OTHER COURSES
C*
61      M=T+1
      MN=T+PT-1
      DO 70 TTT=M,MN
      IF(TT(N,TTT,J).EQ.1HN) BAD(N,4,J)=BAD(N,4,J)+1
      IF(TT(NC(II+1),TTT,J).EQ.1HN)BAD(NC(II+1),4,J)=BAD(NC(II+1),4,J)+

      IF(TT(N,TTT,J).NE.1HX) GO TO 71
      TT(N,T,J)=1HX
      FLAG5=1
71      IF(TT(NC(II+1),TTT,J).NE.1HX) GO TO 72
      TT(NC(II+1),T,J)=1HX
      FLAG5=1
      RETURN
72      DO 75 JK=1,NNN+1
      IF(JK.EQ.N) GO TO 75
      IF(JK.EQ.NC(II+1)) GO TO 75

```

IF(TT(JK,TTT,J).NE.1HA) GO TO 75

TT(N,T,J)=1HX

76 IF(JK.EQ.NC(II+1)) GO TO 75

TT(NC(II+1),T,J)=1HX

FLAG5=1

75 CONTINUE

70 CONTINUE

RETURN

END

C*

C*

C*

C* ***** SUBROUTINE IN ORDER TO ASSIGN CLASSROOM(S)

C*

SUBROUTINE ROOMS(J,PT,INC,II,FLAG1,FLAG3,FLAG6,FLAG7,CLASS)

COMMON TT(20,48,5),ROOM(45),ASSA(45,5),PASSA(45,5),COURSE(20,5),

*IINDEX(20,45,5),BAD(20,6,5),CONF(20,20,5),NC(20),NN(5),N,T,

*PERSUM(20,10,5),NCORSE(150),OPCODE(150,10),CONFCT(150),

*NAME(150,24),NUMBER(5),NOC(5)

INTEGER T,PT,CLASS,FLAG1,FLAG3,FLAG6,FLAG7,BAD,ROOM,TT,HARF

IF(FLAG6.EQ.1) GO TO 11

IF(FLAG3.NE.1) GO TO 4

C* UPDATING CLASSROOM NO. AFTER ASSIGNMENT DELETING

C*

FLAG3=0

DO 3 TTT=T,T+PT-1

3 ROOM(TTT)=ROOM(TTT)+1

RETURN

C* CHECKING THE CLASSROOMS : OCCUPIED OR NOT

C*

4 CLASS=0

DO 5 TTT=T,T+PT-1

IF(ROOM(TTT).EQ.0) GO TO 5

CLASS=CLASS+1

5 CONTINUE

C* ASSIGNMENT TO CLASSROOMS ACCORDING TO PT

C*

IF(CLASS.NE.PT) GO TO 7

DO 6 TTT=T,T+PT-1

ROOM(TTT)=ROOM(TTT)-1

6 CONTINUE

RETURN

```
C*          ALL CLASSROOMS ARE OCCUPIED DURING ALL OR SOME PERIODS
C*
7  IF(TT(N,T,J).EQ.1HA) GO TO 8
   IF(TT(N,T,J).EQ.1HP) GO TO 9
   FLAG1=1
   RETURN
8  BAD(N,2,J)=BAD(N,2,J)+CLASS
   GO TO 10
9  BAD(N,3,J)=BAD(N,3,J)+CLASS
10 FLAG1=1
   RETURN
11 IF(FLAG7.EQ.1) GO TO 19
C* CHECKING THE CLASSROOMS : OCCUPIED OR NOT FOR PARALLEL COURSE CASE
C*
   CLASS=0
   DO 12 TTT=T,T+PT-1
   IF(ROOM(TTT).LT.2) GO TO 14
   CLASS=CLASS+1
12 CONTINUE
C*          ASSIGNMENT TO CLASSROOMS ACCORDING TO PT
C*
   IF(CLASS.NE.PT) GO TO 14
   DO 13 TTT=T,T+PT-1
   ROOM(TTT)=ROOM(TTT)-2
13 CONTINUE
   RETURN
C*          NUMBER OF CLASSROOMS IS LESS THAN 2
C*
14 FLAG6=0
   IF(ROOM(TTT).EQ.1) GO TO 15
C*          ALL CLASSROOMS ARE OCCUPIED DURING ALL OR SOME PERIODS
C*
   IF((TT(N,T,J).EQ.1HA).AND.(TT(NC(II+1),T,J).EQ.1HA)) GO TO 16
   IF((TT(N,T,J).EQ.1HP).AND.(TT(NC(II+1),T,J).EQ.1HP)) GO TO 17
   FLAG1=1
   RETURN
C* PARALLEL ASSIGNMENT CANNOT BE MADE BECAUSE OF ROOM UNAVAILABILITY
C*
```

```

15  ROOM(TTT)=ROOM(TTT)-1
    RETURN
16  BAD(N,2,J)=BAD(N,2,J)+CLASS

    BAD(NC(II+1),2,J)=BAD(NC(II+1),2,J)+CLASS
    GO TO 18
17  BAD(N,3,J)=BAD(N,3,J)+CLASS
    BAD(NC(II+1),3,J)=BAD(NC(II+1),3,J)+CLASS
18  FLAG1=1
    RETURN

C*          UPDATING CLASSROOM NO. AFTER ASSIGNMENT DELETING
C*
19  FLAG7=0
    DO 20 TTT=T,T+PT-1
20  ROOM(TTT)=ROOM(TTT)+2
    RETURN
    END

C*
C*
C*
C* *****SUBROUTINE FOR ASSIGNMENT OF COURSE(S)
C*
    SUBROUTINE ASSIGN(J,PT,INC,II,FLAG2,FLAG3,FLAG6,FLAG7)
    COMMON TT(20,48,5),ROOM(45),ASSA(45,5),PASSA(45,5),COURSE(20,5),
    *IINDEX(20,45,5),BAD(20,6,5),CONF(20,20,5),NC(20),NN(5),N,T,
    *PERSUM(20,10,5),NCORSE(150),OPCODE(150,10),CONFCT(150),
    *NAME(150,24),NUMBER(5),NOC(5)
    INTEGER T,PT,FLAG1,FLAG2,FLAG3,FLAG6,FLAG7,TTT,DD,ASSA,PASSA,
    *COURSE,ROOM,PERSUM,TT,HARF
    IF(FLAG6.EQ.1) GO TO 12
    IF(COURSE(N,J).EQ.0) GO TO 5
    PERSUM(N,1,J)=COURSE(N,J)
    IF(TT(N,T,J).EQ.1HA) GO TO 6
    DO 7 III=1,PT
    ASSA(T,J)=COURSE(N,J)
    IF=2
1  IF(PERSUM(N,IP,J).NE.0) GO TO 3
    PERSUM(N,IP,J)=T
C*  PRINT*, 'PERSUM',PERSUM(N,IP,J)
    GO TO 2

```

```

3    IP=IP+1
    GO TO 1
2    T=T+INC
    IF(III.EQ.PT) GO TO 21
C*          LUNCH TIME OR NOT

C*

    DO 8 TTT=1,5
    IF(T.EQ.5+9*(TTT-1)) GO TO 9
8    CONTINUE
C*          END OF DAY OR NOT
C*

    DO 10 DD=1,4
    IF(T.EQ.(9*DD)+1) GO TO 9
10   CONTINUE
    IF(TT(N,T,J).EQ.1HX) GO TO 9
7    CONTINUE
21   RETURN
C*          IDLE TIME ASSIGNMENT
C*

5    PT=1
    ASSA(T,J)=0
    T=T+INC
    RETURN
C*          ASSIGNMENT OF COURSE WITH PRIORITY 'A'
C*

6    DO 4 III=1,PT
    ASSA(T,J)=COURSE(N,J)
    IP=2
43   IF(PERSUM(N,IP,J).NE.0) GO TO 33
    PERSUM(N,IP,J)=T
C*    PRINT*, 'PERSUM=', PERSUM(N,IP,J)
    GO TO 23
33   IP=IP+1
    GO TO 43
23   T=T+INC
4    CONTINUE
    RETURN
C*          ASSIGNMENT DELETING
C*

```

```

9      T=T-III*INC
      DO 11 TTT=T,T+III-1
      TT(N,TTT,J)=1HX
      IP=2
38     IF(PERSUM(N,IP,J).EQ.TTT) GO TO 28
      IP=IP+1
      IF(IP.GT.10) GO TO 11

      GO TO 38

28     PERSUM(N,IP,J)=0
C*     PRINT*, 'PERSUM=', PERSUM(N,IP,J)
11     CONTINUE
      FLAG3=1

C*
C*
C*
C*           CALL CLASSROOM ASSIGNMENT SUBROUTINE
C*
      CALL ROOMS(J,PT,INC,II,FLAG1,FLAG3,FLAG6,FLAG7,CLASS)
      IF(IINDEX(2,T,J).EQ.0) GO TO 5
      FLAG2=1
      RETURN

C*           ASSIGNMENT FOR PARALLEL COURSE CASE
C*

12     PERSUM(N,1,J)=COURSE(N,J)
      PERSUM(NC(II+1),1,J)=COURSE(NC(II+1),J)
      IF((TT(N,T,J).EQ.1HA).AND.(TT(NC(II+1),T,J).EQ.1HA)) GO TO 13
      DO 14 III=1,PT
      ASSA(T,J)=COURSE(N,J)
      PASSA(T,J)=COURSE(NC(II+1),J)
      IP=2
44     IF(PERSUM(N,IP,J).NE.0) GO TO 34
      PERSUM(N,IP,J)=T
C*     PRINT*, 'PERSUM=', PERSUM(N,IP,J)
      GO TO 24
34     IP=IP+1
      GO TO 44
24     IP=2
45     IF(PERSUM(NC(II+1),IP,J).NE.0) GO TO 35
      PERSUM(NC(II+1),IP,J)=T

```

```

C8  PRINT*, 'PERSUM=', PERSUM(NC(II+1), IP, J)
    GO TO 25
35  IP=IP+1
    GO TO 45
25  T=T+INC
    IF(III.EQ.PT) GO TO 22
C*          LUNCH TIME OR NOT
C*
    DO 15 TTT=1,5

    IF(T.EQ.5+9*(TTT-1)) GO TO 16
15  CONTINUE
C*          END OF DAY OR NOT
C*
    DO 17 DD=1,4
    IF(T.EQ.(9*DD)+1) GO TO 16
17  CONTINUE
    IF((TT(N,T,J).EQ.1HX).AND.(TT(NC(II+1),T,J).EQ.1HX)) GO TO 16
14  CONTINUE
22  RETURN
C*          ASSIGNMENT OF COURSES WITH PRIORITY 'A'
C*
13  DO 18 III=1,PT
    ASSA(T,J)=COURSE(N,J)
    PASSA(T,J)=COURSE(NC(II+1),J)
    IF=2
46  IF(PERSUM(N,IP,J).NE.0) GO TO 36
    PERSUM(N,IP,J)=T
C*  PRINT*, 'PERSUM=', PERSUM(N,IP,J)
    GO TO 26
36  IP=IP+1
    GO TO 46
26  IP=2
47  IF(PERSUM(NC(II+1),IP,J).NE.0) GO TO 37
    PERSUM(NC(II+1),IP,J)=T
    PRINT*, 'PERSUM=', PERSUM(NC(II+1),IP,J)
    GO TO 27
37  IP=IP+1
    GO TO 47
27  T=T+INC

```

```

18    CONTINUE
      RETURN
C*          ASSIGNMENT DELETING
C*
16    T=T-III*INC
      DO 19 TTT=T,T+III-1
        TT(N,TTT,J)=1HX
        TT(NC(II+1),TTT,J)=1HX
        PASSA(TTT,J)=0
        IP=2
40    IF(PERSUM(N,IP,J).EQ.TTT) GO TO 39

      IP=IP+1
      IF(IP.GT.10) GO TO 42
      GO TO 40
39    PERSUM(N,IP,J)=0
C*    PRINT*, 'PERSUM=', PERSUM(N,IP,J)
42    IP=2
41    IF(PERSUM(NC(II+1),IP,J).EQ.TTT) GO TO 48
      IP=IP+1
      IF(IP.GT.10) GO TO 19
      GO TO 41
48    PERSUM(NC(II+1),IP,J)=0
C*    PRINT*, 'PERSUM=', PERSUM(NC(II+1),IP,J)
19    CONTINUE
      FLAG7=1

C*
C*
C*
C*          CALL CLASSROOM ASSIGNMENT SUBROUTINE
C*
      CALL ROOMS(J,PT,INC,II,FLAG1,FLAG3,FLAG6,FLAG7,CLASS)
      IF(IINDEX(2,T,J).EQ.0) GO TO 5
      IF(IINDEX(2,T,J).LT.2) GO TO 20
      FLAG2=1
      RETURN
20    FLAG2=1
      FLAG6=0
      RETURN
      END

```

```

C*
C*
C*
C* ***** SUBROUTINE FOR AVOIDING INSTRUCTOR CONFLICTS
C*

SUBROUTINE INSTOR(J,PT,FLAG6)
COMMON TT(20,48,5),ROOM(45),ASSA(45,5),PASSA(45,5),COURSE(20,5),
*IINDEX(20,45,5),BAD(20,6,5),CONF(20,20,5),NC(20),NN(5),N,T,
*PERSUM(20,10,5),NCORSE(150),OPCODE(150,10),CONFCT(150),
*NAME(150,24),NUMBER(5),NOC(5)
INTEGER T,PT,TTT,FLAG6,COURSE,TT,HARF
IF(FLAG6.EQ.1) GO TO 7
IF(J.EQ.5) GO TO 3

DO 4 JJ=J+1,5
JK=NN(J)
DO 5 K=1,JK
IF(TT(N,47,J).NE.TT(K,47,JJ)) GO TO 5
DO 6 TTT=T-PT,T-1
6 TT(K,TTT,JJ)=1HX
5 CONTINUE
4 CONTINUE
3 RETURN

C* PARALLEL COURSE CASE
C*

7 IF(J.EQ.5) GO TO 13
DO 8 JJ=J+1,5
JK=NN(J)
DO 9 K=1,JK
IF(TT(N,47,J).EQ.0) GO TO 11
IF(TT(N,47,J).NE.TT(K,47,JJ)) GO TO 11
DO 10 TTT=T-PT,T-1
10 TT(K,TTT,JJ)=X
IF(TT(NC(II+1),47,J).EQ.0) GO TO 13
11 IF(TT(NC(II+1),47,J).NE.TT(K,47,JJ)) GO TO 9
DO 12 TTT=T-PT,T-1
12 TT(K,TTT,JJ)=1HX
9 CONTINUE
8 CONTINUE
13 RETURN

```

END

C*
C*
C*
C*
C*

***** SUBROUTINE FOR UPDATING THE TIME-TABLE

SUBROUTINE TTUP(J,PT,INC,II,D,FLAG6)

COMMON TT(20,48,5),ROOM(45),ASSA(45,5),PASSA(45,5),COURSE(20,5),
*IINDEX(20,45,5),BAD(20,6,5),CONF(20,20,5),NC(20),NN(5),N,T,
*PERSUM(20,10,5),NCORSE(150),OPCODE(150,10),CONFCT(150),
*NAME(150,24),NUMBER(5),NOC(5)

INTEGER T,D,DD,TTT,PT,FLAG6,COURSE,DUMMYCEN1,TT,HARF

IF(COURSE(N,J).EQ.0) GO TO 3

IF(FLAG6.EQ.1) GO TO 11

TT(N,46,J)=TT(N,46,J)-PT

IF(TT(N,46,J).EQ.0) GO TO 9

20

M=9*(D+1)

DO 5 TTT=T,M

5

TT(N,TTT,J)=1HX

6

DO 7 DD=1,4

IF(T-1.EQ.9*DD) GO TO 8

7

CONTINUE

RETURN

8

IF(ND.EQ.1) GO TO 4

C*

8 D=D+1

D=D+1

RETURN

4

ND=0

RETURN

9

DO 10 TTT=T,45

10

TT(N,TTT,J)=1HX

3

IF(T-1.EQ.9) D=D+1

IF(T-1.EQ.18) D=D+1

IF(T-1.EQ.27) D=D+1

IF(T-1.EQ.36) D=D+1

RETURN

*

PARALLEL COURSE CASE

*

11

TT(N,46,J)=TT(N,46,J)-PT

```

TT(NC(II+1),46,J) =TT(NC(II+1),46,J)-PT
IF((TT(N,46,J).EQ.0).AND.(TT(NC(II+1),46,J).EQ.0)) GO TO 12
IF(TT(N,46,J).EQ.0) GO TO 18
IF(TT(NC(II+1),46,J).EQ.0) GO TO 19
M=9*(D+1)

```

```

DO 21 TTT=T,M
21 TT(NC(II+1),TTT,J)=1HX
DO 25 DD=1,4
IF(T-1.EQ.9*DD) GO TO 26
25 CONTINUE
GO TO 20
26 D=D+1
ND=1
GO TO 20
19 DO 27 TTT=T,45
27 TT(NC(II+1),TTT,J)=1HX
GO TO 20

```

```

18 M=9*(D+1)
DO 22 TTT=T,M
22 TT(NC(II+1),TTT,J)=1HX
DO 23 DD=1,4
IF(T-1.EQ.9*DD) GO TO 24
23 CONTINUE
GO TO 9
24 D=D+1
GO TO 9
12 DO 17 TTT=T,45
TT(N,TTT,J)=1HX
17 TT(NC(II+1),TTT,J)=1HX
IF(T-1.EQ.9) D=D+1
IF(T-1.EQ.18) D=D+1
IF(T-1.EQ.27) D=D+1
IF(T-1.EQ.36) D=D+1
RETURN
END

```

```

C*
C*
C*
C*

```

```

***** SUBROUTINE FOR SELECTING A CANDIDATE RANDOMLY

```

DECISION RULE 2

C*
C*

```

SUBROUTINE RANDM(SEED,NM,IN)
  INTEGER SEED,FLAG8
  REAL A
  IF(NM.LE.2) SEED=SEED/2
  R=RANDOM(SEED)
  IF(FLAG8.NE.1) GO TO 30
  FLAG8=0
  A=1+R
  GO TO 40
30  A=(1+((NM-1)*R))
40  TEMP=A*10
C   ITEMP=TEMP
   ITEMP=TEMP/10
  DIF=A-ITEMP
  IF(DIF.GE.0.5) GO TO 10
  IN=ITEMP
  GO TO 20

```

```

10  IN=ITEMP+1
20  RETURN
END

```

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

***** SUBROUTINE FOR IRREGULAR STUDENTS

```

SUBROUTINE IRREG
  COMMON TT(20,48,5),ROOM(45),ASSA(45,5),PASSA(45,5),COURSE(20,5),
  *IINDEX(20,45,5),BAD(20,6,5),CONF(20,20,5),NC(20),NN(5),N,T,
  *PERSUM(20,10,5),NCORSE(150),OPCODE(150,10),CONFCT(150),
  *NAME(150,24),NUMBER(5),NOC(5)
  INTEGER PERSUM,OPCODE,CONFCT,DAY,TIME,DUMMYCEN1
  DOUBLE PRECISION GUNE

```

C*
C*
C*
C*

READING THE NAMES OF IRREGULAR STUDENTS AND COURSES TAKEN
BY THEM FROM DATA FILE NAMED 'IRREGULAR'

```

READ(7,*) IRRNO
DO 700 I=1,IRRNO

```

```

      READ(7,701) (NAME(I, KK), KK=1, 24)
701  FORMAT(24A)
      READ(7,*) NCORSE(I)
      READ(7,*) (OPCODE(I, K), K=1, NCORSE(I))
700  CONTINUE

C*
C*
C*          SEARCHING THE CONFLICTS OF IRREGULARS
C*
      DO 300 I=1, IRRND
      WRITE(6, 301)
301  FORMAT(///10X, 'NAME AND SURNAME')
      WRITE(6, 302)
302  FORMAT(10X, '-----')
      WRITE(6, 303) (NAME(I, KK), KK=1, 4)
303  FORMAT(//10X, 24A)

C*
C*          CHOOSING THE FIRST AND NEXT COURSE FOR COMPARISON
C*
      DO 200 K1=1, NCORSE(I)-1

      DO 202 K2=K1+1, NCORSE(I)

C*
C*          FINDING THE OPTIC CODES OF FIRST AND NEXT COURSE
C*
      DO 400 J=1, 4
      NT=NN(J)+NUMBER(J)+NOC(J)
      DO 402 N=1, NT
      IF(PERSUM(N, 1, J).NE.OPCODE(I, K1)) GO TO 401
      N1=N
      J1=J
      GO TO 402
401  IF(PERSUM(N, 1, J).NE.OPCODE(I, K2)) GO TO 402
      N2=N
      J2=J
402  CONTINUE
400  CONTINUE

C*
C*          COMPARISON IN ORDER TO FIND CONFLICTING HOURS
      PRINT*, 'here'

```

C*

```
DO 600 IP1=2,10
  IF(PERSUM(N1,IP1,J1).EQ.0) GO TO 600
DO 601 IP2=2,10
  IF(PERSUM(N2,IP2,J2).EQ.0) GO TO 601
  IF(PERSUM(N1,IP1,J1).NE.PERSUM(N2,IP2,J2)) GO TO 601
  IF(PERSUM(N1,1,J1).EQ.PERSUM(N2,1,J2)) GO TO 601
  CONFCT(I)=CONFCT(I)+1
  IF(PERSUM(N1,IP1,J1).GT.9) GO TO 304
  GUNE='MONDAY'
  TIME=PERSUM(N1,IP1,J1)
  GO TO 308
304  IF(PERSUM(N1,IP1,J1).GT.18) GO TO 305
  DAY=2
  GUNE='TUESDAY'
  TIME=PERSUM(N1,IP1,J1)-(9*(DAY-1))
  GO TO 308
305  IF(PERSUM(N1,IP1,J1).GT.27) GO TO 306
  DAY=3
  GUNE='WEDNESDAY'
  TIME=PERSUM(N1,IP1,J1)-(9*(DAY-1))
  GO TO 308

306  IF(PERSUM(N1,IP1,J1).GT.36) GO TO 307
  DAY=4
  GUNE='THURSDAY'
  TIME=PERSUM(N1,IP1,J1)-(9*(DAY-1))
  GO TO 308
307  DAY=5
  GUNE='FRIDAY'
  TIME=PERSUM(N1,IP1,J1)-(9*(DAY-1))
308  WRITE(6,309) PERSUM(N1,1,J1),PERSUM(N2,1,J2),TIME,GUNE
309  FORMAT(1X,///,I7,2X,'CONFLICTS WITH',2X,I7,2X,'AT PERIOD',
  *2X,I2,2X,'OF',2X,A12)
601  CONTINUE
600  CONTINUE
  N2=0
202  CONTINUE
  N1=0
200  CONTINUE
```

```
WRITE(6,310) CONFCT(I)
310  FORMAT(///10X,'TOTAL CONFLICTING HOURS=',3X,I3)
300      CONTINUE
      DO 500 I=1,IRRNO
500  NTOTAL=NTOTAL+CONFCT(I)
C*  WRITE(6,510) NTOTAL
C*510  FORMAT(/10X,'TOTAL CONFLICTING HOURS FOR ALL IRREGULARS=',3X,I3)
      RETURN
      END
```

T. C.
Yükseköğretim Kurulu
Dokümantasyon Merkezi