A COMPUTER BASED SUPPORT SYSTEM FOR TIMETABLING PROBLEMS

by

HO YIN (PATRICK) WONG

A Thesis

Submitted to the Faculty of Graduate Studies

in Partial Fulfillment of the Requirements

for the Degree of

MASTER OF SCIENCE

Department of Actuarial and Management Sciences

University of Manitoba

Winnipeg, Manitoba

ISBN 0-612-13571-3

Canada

Name _____

Dissertation Abstrac.. ıternational is arranged by broad, general subject categories. Please select the one subject which most nearly describes the content of your dissertation. Enter the corresponding four-digit code in the spaces provided.

Operations Research
SUBJECT TERM

0796
SUBJECT CODE

U·M·I

Subject Categories

THE HUMANITIES AND SOCIAL SCIENCES

COMMUNICATIONS AND THE ARTS
Architecture 0729
Art History 0377
Cinema 0900
Dance 0378
Fine Arts 0357
Information Science 0723
Journalism 0391
Library Science 0399
Mass Communications 0708
Music 0413
Speech Communication 0459
Theater 0465

EDUCATION
General 0515
Administration 0514
Adult and Continuing 0516
Agricultural 0517
Art 0273
Bilingual and Multicultural 0282
Business 0688
Community College 0275
Curriculum and Instruction 0727
Early Childhood 0518
Elementary 0524
Finance 0277
Guidance and Counseling 0519
Health 0680
Higher 0745
History of 0520
Home Economics 0278
Industrial 0521
Language and Literature 0279
Mathematics 0280
Music 0522
Philosophy of 0998
Physical 0523
Psychology 0525
Reading 0535
Religious 0527
Sciences 0714
Secondary 0533
Social Sciences 0534
Sociology of 0340
Special 0529
Teacher Training 0530
Technology 0710
Tests and Measurements 0288
Vocational 0747

LANGUAGE, LITERATURE AND LINGUISTICS
Language
- General 0679
- Ancient 0289
- Linguistics 0290
- Modern 0291

Literature
- General 0401
- Classical 0294
- Comparative 0295
- Medieval 0297
- Modern 0298
- African 0316
- American 0591
- Asian 0305
- Canadian (English) 0352
- Canadian (French) 0355
- English 0593
- Germanic 0311
- Latin American 0312
- Middle Eastern 0315
- Romance 0313
- Slavic and East European 0314

PHILOSOPHY, RELIGION AND THEOLOGY
Philosophy 0422
Religion
- General 0318
- Biblical Studies 0321
- Clergy 0319
- History of 0320
- Philosophy of 0322

Theology 0469

SOCIAL SCIENCES
American Studies 0323
Anthropology
- Archaeology 0324
- Cultural 0326
- Physical 0327

Business Administration
- General 0310
- Accounting 0272
- Banking 0770
- Management 0454
- Marketing 0338

Canadian Studies 0385
Economics
- General 0501
- Agricultural 0503
- Commerce-Business 0505
- Finance 0508
- History 0509
- Labor 0510
- Theory 0511

Folklore 0358
Geography 0366
Gerontology 0351
History
- General 0578
- Ancient 0579
- Medieval 0581
- Modern 0582
- Black 0328
- African 0331
- Asia, Australia and Oceania 0332
- Canadian 0334
- European 0335
- Latin American 0336
- Middle Eastern 0333
- United States 0337

History of Science 0585
Law 0398
Political Science
- General 0615
- International Law and Relations 0616
- Public Administration 0617

Recreation 0814
Social Work 0452
Sociology
- General 0626
- Criminology and Penology ... 0627
- Demography 0938
- Ethnic and Racial Studies 0631
- Individual and Family Studies 0628
- Industrial and Labor Relations 0629
- Public and Social Welfare 0630
- Social Structure and Development 0700
- Theory and Methods 0344

Transportation 0709
Urban and Regional Planning 0999
Women's Studies 0453

THE SCIENCES AND ENGINEERING

BIOLOGICAL SCIENCES
Agriculture
- General 0473
- Agronomy 0285
- Animal Culture and Nutrition 0475
- Animal Pathology 0476
- Food Science and Technology 0359
- Forestry and Wildlife 0478
- Plant Culture 0479
- Plant Pathology 0480
- Plant Physiology 0817
- Range Management 0777
- Wood Technology 0746

Biology
- General 0306
- Anatomy 0287
- Biostatistics 0308
- Botany 0309
- Cell 0379
- Ecology 0329
- Entomology 0353
- Genetics 0369
- Limnology 0793
- Microbiology 0410
- Molecular 0307
- Neuroscience 0317
- Oceanography 0416
- Physiology 0433
- Radiation 0821
- Veterinary Science 0778
- Zoology 0472

Biophysics
- General 0786
- Medical 0760

EARTH SCIENCES
Biogeochemistry 0425
Geochemistry 0996
Geodesy 0370
Geology 0372
Geophysics 0373
Hydrology 0388
Mineralogy 0411
Paleobotany 0345
Paleoecology 0426
Paleontology 0418
Paleozoology 0985
Palynology 0427
Physical Geography 0368
Physical Oceanography 0415

HEALTH AND ENVIRONMENTAL SCIENCES
Environmental Sciences 0768
Health Sciences
- General 0566
- Audiology 0300
- Chemotherapy 0992
- Dentistry 0567
- Education 0350
- Hospital Management 0769
- Human Development 0758
- Immunology 0982
- Medicine and Surgery 0564
- Mental Health 0347
- Nursing 0569
- Nutrition 0570
- Obstetrics and Gynecology .. 0380
- Occupational Health and Therapy 0354
- Ophthalmology 0381
- Pathology 0571
- Pharmacology 0419
- Pharmacy 0572
- Physical Therapy 0382
- Public Health 0573
- Radiology 0574
- Recreation 0575
- Speech Pathology 0460
- Toxicology 0383

Home Economics 0386

PHYSICAL SCIENCES
Pure Sciences
Chemistry
- General 0485
- Agricultural 0749
- Analytical 0486
- Biochemistry 0487
- Inorganic 0488
- Nuclear 0738
- Organic 0490
- Pharmaceutical 0491
- Physical 0494
- Polymer 0495
- Radiation 0754

Mathematics 0405
Physics
- General 0605
- Acoustics 0986
- Astronomy and Astrophysics 0606
- Atmospheric Science 0608
- Atomic 0748
- Electronics and Electricity 0607
- Elementary Particles and High Energy 0798
- Fluid and Plasma 0759
- Molecular 0609
- Nuclear 0610
- Optics 0752
- Radiation 0756
- Solid State 0611

Statistics 0463

Applied Sciences
Applied Mechanics 0346
Computer Science 0984
Engineering
- General 0537
- Aerospace 0538
- Agricultural 0539
- Automotive 0540
- Biomedical 0541
- Chemical 0542
- Civil 0543
- Electronics and Electrical 0544
- Heat and Thermodynamics ... 0348
- Hydraulic 0545
- Industrial 0546
- Marine 0547
- Materials Science 0794
- Mechanical 0548
- Metallurgy 0743
- Mining 0551
- Nuclear 0552
- Packaging 0549
- Petroleum 0765
- Sanitary and Municipal 0554
- System Science 0790

Geotechnology 0428
Operations Research 0796
Plastics Technology 0795
Textile Technology 0994

PSYCHOLOGY
General 0621
Behavioral 0384
Clinical 0622
Developmental 0620
Experimental 0623
Industrial 0624
Personality 0625
Physiological 0989
Psychobiology 0349
Psychometrics 0632
Social 0451

A COMPUTER BASED SUPPORT SYSTEM

FOR TIMETABLING PROBLEMS

BY

HO YIN (PATRICK) WONG

A Thesis submitted to the Faculty of Graduate Studies of the University of Manitoba in partial fulfillment of the requirements of the degree of

MASTER OF SCIENCE

ABSTRACT

An educational timetabling design involves scheduling a number of tuples, each consisting of a course, a location, an instructor and a class of students, to a fixed number of timeslots. There are many procedures available in the literature based on mathematical programming and graph theory. This work develops a prototype computer-based support system for constructing timetables in Asian school system. The developed software can be implemented in a desktop computer. The program has been written with features to handle complicated situations, such as settings, and multiple periods. A heuristic algorithm is proposed to allocate the course-instructor pairs into the sets of rooms and timeslots, to avoid the unnecessary clustering for individual teaching workload. The system described in this work has been applied to real school data and is indicated to be an efficient and flexible tool for timetable construction. Limitations of the system and recommendations for further improvements are discussed.

ACKNOWLEDGMENTS

Throughout this course of this research study, I have been fortunate to have the support, and advice from many people. I am indebted to my immediate supervisor, Dr. Bhatt, S. K., for his faith in me and for all the tireless guidance given during my studies. A special thanks to Dr. Rosenbloom, E. S., for his valuable suggestions in developing and writing of this thesis. Appreciation is also expressed to Dr. Alfa, A. S., for gallantly serving in the thesis committee.

My gratitude must be extended to my best friend, Mr. Li, Wai Man. for his precious time-sharing and professional advice in the software development for this model. His efforts have been vital in the completion of this project.

To my family, I am truly grateful for their years of continuous support and confidence, without which, I could not have achieved my accomplishments.

Finally, to my wife, I particularly thank you for her strength and inspiration she has given me. I will always be indebted to her patience and understanding, which has been invaluable to the completion of my work.

TABLE OF CONTENTS

TABLE OF CONTENTS

TABLE OF CONTENTS

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

Chapter 1

INTRODUCTION

Timetabling in an educational system is the development of a schedule bringing students, instructors, curriculum, materials, and space into a systematic arrangement. It requires the resources in conformity with their availability in order to generate an optimal learning environment. Effective timetabling is sometimes considered by many school timetablers to be the supreme trial of their management skills. This study tries to investigate some of the timetabling problems from a timetabler's point of view and to construct techniques that will reduce the burden to the educational planner in timetable scheming. A database management system (DBMS) is used and the resulting algorithm can be implemented in a desktop computer. The sources of data are obtained from the schools in Hong Kong. However, the methodology developed in this work can be applied to other institutions.

1.1 Some Background in School Timetabling

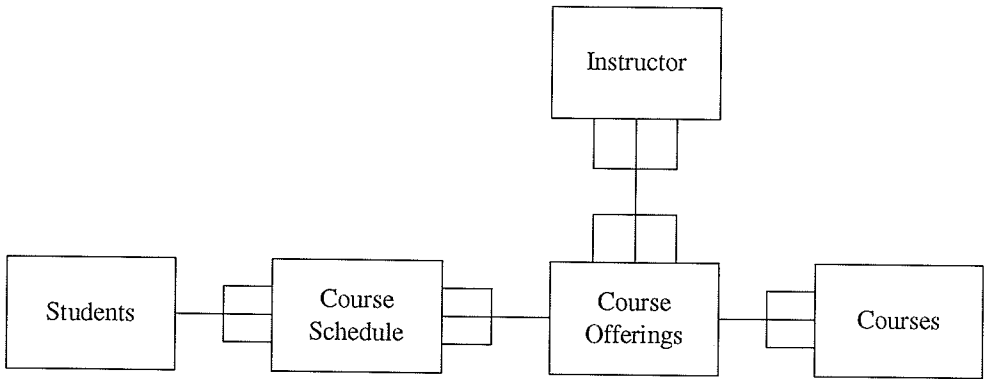
During the last three decades, a great deal of research has addressed in the school timetabling problems. One of the reasons is the continual changing pattern in education that results in the introduction of new courses and options with diverse working patterns and facility requirements. Students are permitted to have a much

greater variety of choices in their selected areas. The result is that the timetables are become more complex.

A timetable typically describes a Course-Instructor-Time-Location allocation. It is therefore characterized by the level of educational regulations and the complexities generated by the flexibility or rigidity in course, time and location options. A typical educational timetabling problem can be described by a flow chart or a graphic diagram of the database that consists of lines that may be single, forked or dashed-dot forked, to show the relationship among various components. A single line indicates a one-to-one relationship; forked line indicates the 'many' sides of a one-to-many or many-to-many relationship; dashed-dot forked line indicates the possibility of weak 'many' relationships. The followings are two such models to represent database schema of timetabling problems.

Model A : A schema for team teaching in North America.

Different components in model A are connected in a single phase through the single or forked lines, to exhibit the Course-Instructor-Time-Location relationship.

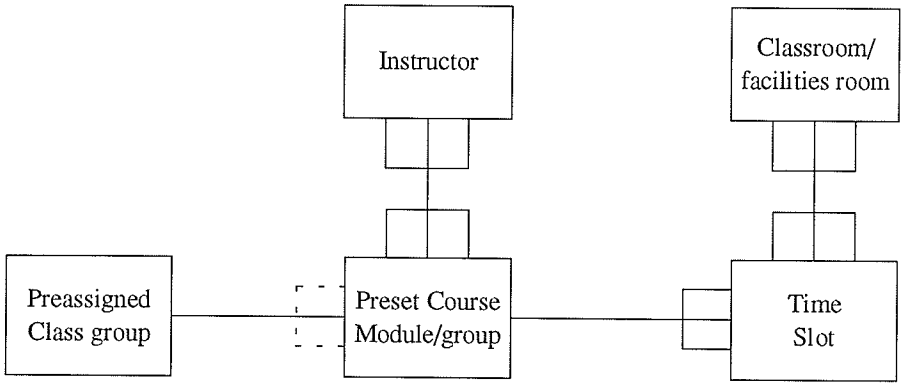


Its objective is to provide a non-conflicted environment so as to allow individuals to select the courses in a flexible manner. The schema is due to Townsend

[1991] that falls into model A category. This type of system is mostly employed in educational institutions such as Universities, Colleges, adult education training centers and the high schools in north America.

Model B : A schema for teaching in Preset Group / Class

Model B applies to the situations where the course-curriculum is determined first, which is followed by the assignment of teaching staff to specific course, considering the availabilities of teaching locations and time slots. Eventually, the pre-defined classes or group of students will be allocated to the combination. The following graphic diagram describes the sequences of components.



Phase 1 -- Class (Course) - Instructor, by group of senior administrative staffs or Principal

Phase 2 -- Class-Instructor-Time-Room, by group of staffs or instructors

It is a typical Grouped Class(course)-Instructor-Time-Location model that will require a different methodology than the previous one due to multiple phases. This system common in Asia leads to a closed environment during school time for

both students and instructors. The structure is less flexible. Each student group customarily serves a one-to-one relationship to the teaching group especially in the higher forms (grades). Most of the student timetables are compact that indicate there is no gap during the school time except in the recesses and lunch time. Free periods may be offered in the upper grades. The whole system is rigid, courses are preset, and choices are limited.

Brookes [1980] and De Werra [1985] have generalized the timetabling process consisting of three distinct stages. The initial stage is called the Thinking Stage. It involves descriptive decision making on the formulation of objective and the determination of the required curricular mandates for both instructors and groups of the students. The second stage is called the Planning stage. It involves the actual allocation of the necessary teaching resources including staff, equipment and space. Additionally, it verifies whether the constructed timetable can accomplish all the preassigned requirements in term of its feasibility and compatibility. The final stage is called the Constructing stage. It generates an imperative procedure to produce the final version of the timetables. Traditional timetabling problems as described in Operations Research and computing literature generally refer to the second and final stage of the above models.

1.2 The Timetabling Problem

Any educational timetabling problem involves scheduling a number of collections, each involving a class of students, an instructor, a set of lecturing material, a location and an interval of time. The basic process to create a valid

timetable is to designate these resources in such a way that no instructor, group of students and location are used more than once at any given interval. It would be relatively easy if there were absence of any constraint. Many distinct algorithms have been developed which suggest practical solutions for some special cases while others have never left the experimental stage.

Numerous computer software packages, such as 'SCHOLA', 'INTEGA', 'MOSAIK', 'AU.ST.ER' [Junginger 1986] have been designed to meet the different needs and restrictions to the timetabling problems. Others, such as 'SSSS', 'GASP', 'flex-mod', 'arena', etc., have claimed the capability of performing the scheduling operation [Dempsey 1983], along with the payroll and budgeting. However, none of the packages can be successfully put into the Asian school system because they are unable to satisfy their specific requirements. The above algorithms, according to Model A structure, are exclusively designed to generate a studying environment to students in which they can choose from various course options. In addition, some of them can only be run in the main frame computers. Most of the timetablers are not computer professionals; thus construction of timetables by computer may become too complex and is usually considered as an infeasible proposition. As a consequence, fewer Asian schools have provided any computer capability to their timetabling procedures.

1.3 The Objective

The motivation of this study arises from the idea that computer based procedures will reduce the unnecessary man power being allocated to the manually

handled timetabling process in Asian school system. On the basis of the schema in Model B, the idea of Johnson's database management system [Johnson 1993] is further extended and will be implemented to the existing timetabling environments. A Heuristic procedure is developed to obtain a timetable with the prescribed prior information. For example, a course is already assigned to an instructor due to limited options. The computer based support system will be used to allocate the Class(Course)-Instructor pairs into the set of rooms and time slots. The program is written in a common commercial language -- dBase III plus and dBase IV that may be found in most of the school computers. In addition, we will consider the following:

(1) To provide small perturbation during timetabling, a row and column rotation method is introduced to avoid the situation where the same course and instructor appear in the same location at the same time on a given day. This procedure will prevent the unnecessary clustering of individual teaching load.

(2) Perform the bookkeeping task by printing the individual class, instructor and room schedules, to alleviate the clerical burden.

1.4 Outline of This Thesis

The organization of this work is as follows. In chapter two, we review the literature and describe the major issues, the modelings and their applications to educational timetable planning. Chapter three provides a brief description to Johnson's database approach; its limitation will be examined through a case study. Chapter four presents a general discussion of the formulation of our school timetabling problems. A new database system incorporating a heuristic algorithm is

introduced. Detailed characteristics of the computer program like data structures, specifications and flow charts are presented in Chapter five. Finally, Chapter six presents implementation and computational results, conclusions, topics for further works, and recommendations.

Chapter 2

REVIEW OF PAST RESEARCH

The timetabling problem has been well studied in depth by researchers and several relevant articles have appeared recently. An annotated bibliography that covers the articles written before 1979 is found in Schmidt and Strohlein [1980], a survey of timetabling that concentrate on analysis based on Graph Theory is found in De Werra [1985] and a general report of computer timetabling development in Germany is in Junginger [1986]. Various models and techniques have been developed that range from manual bookkeeping systems to those based on the computer interactive system and artificial intelligence. A few achieved success in constructing school or university course timetables under various criteria, while others reported discussions on merely theoretical solutions. This chapter purports to give a detailed review of these timetabling designs. Their algorithms used to construct timetables can be classified into three main approaches.

(1) Mathematical Programming

(2) Graph Theory

(3) Computer interactive and heuristics design

Combinations of above methods have also been reported in the literature. Methods relating to scheduling examinations are customarily identified as separate techniques to the usual timetabling problems and will not be explored in this study.

2.1 Mathematical Programming

2.1.1 Two-step 0-1 Integer Linear Programming Approach

A simple approach can be described as a two-step process by combining the basic Class (Course)-Instructor-period and Room scheduling models without including all the constraints that may be presented in real case [Kang and White 1992]. In the first step, class (course)-instructor pairs are scheduled into a fixed number of periods, and in the second step classrooms are assigned to courses at any kth period. Examination timetabling problem is a typical example of the first step.

Let there be a collection of m classes (courses), i=1, 2, ..., m; n instructors, j=1, 2, ..., n; T cycles, each with p periods, k=1, 2, ..., p; q rooms, v=1, 2, ..., q; and r_{ij} be the required number times the ith course is taught by jth instructor. At any Tth curriculum cycle, $X_{ijk} = 1$ if jth instructor is assigned to ith class during kth period and $X_{ijk} = 0$ otherwise. A curriculum cycle is a general term referring the period of time before the timetable repeats itself.

2.1.1a Class (Course)-Instructor-Period Scheduling Model

$$\textit{Min } Z = \sum_{i=1}^{m}\sum_{j=1}^{n}\sum_{k=1}^{p} C_{ijk} * X_{ijk}$$

or

$$\textit{Max } Z = \sum_{i=1}^{m}\sum_{j=1}^{n}\sum_{k=1}^{p} P_{ijk} * X_{ijk}$$

Subject to

(1) $$\sum_{k=1}^{p} X_{ijk} \le r_{ij}, \quad \forall i, j$$

(2) $$\sum_{i=1}^{m} X_{ijk} \le 1, \quad \forall j, k$$

(3) $$\sum_{j=1}^{n} X_{ijk} \le 1, \quad \forall i, k$$

(4) $$X_{ijk} = [0,1]$$

2.1.1b Room Scheduling Model

$$\textbf{\textit{Min}}\ \textbf{\textit{Z}} = \sum_{i=1}^{m}\sum_{v=1}^{q} C_{iv} * X_{iv}$$

or

$$\textbf{\textit{Max}}\ \textbf{\textit{Z}} = \sum_{i=1}^{m}\sum_{v=1}^{q} P_{iv} * X_{iv}$$

Subject to

(5) $$\sum_{v=1}^{m} X_{iv} = 1, \quad \forall i,$$

(6) $$\sum_{i \in T_k}^{q} X_{iv} \le 1, \quad \forall i, k$$

(7) $$X_{iv} = [0,1]$$

The coefficient C_{ijk} and C_{iv}, P_{ijk} and P_{iv} refer to the associated cost or utility preference that in kth period we schedule ith class (course) to jth instructor and then ith class (course) to vth room. Constraints (2), and (3) enforce no more than one instructor to each class (course) that is being assigned at kth period, (5) ensures every class (course) is assigned to some room while (6) ensures that any room having no more than one class (course) in the same period. The number of required period

given by jth instructor to ith class (course) is governed by (1). The remaining constraints (4) and (7) signify the non-negativity property of the variables.

However, in some educational institutions, the timetabling problem may be defined in a different way. In these models, students who are taking exactly the same course, are defined as a group. De Werra [1985] and others formulate such a problem as the following mathematical programming model. Let there be collections of g groups, s=1, 2, ..., g, all students in each group G_s taking the same courses; m courses of one period each, i=1, 2, ..., m, course M_i assignable to l_i instructors in a curriculum cycle of p period. The maximum number of instructors that can be scheduled at period k is denoted as U_k. Then we define $X_{ik}=1$ if an instructor of course M_i is scheduled at any given period k and $X_{ik}=0$ otherwise.

2.1.1c Course-Period (Group) Scheduling Model

$$\textbf{\textit{Min}}\ Z=\sum_{i=1}^{m}\sum_{k=1}^{p} C_{ik} * X_{ik}$$

or

$$\textbf{\textit{Max}}\ Z=\sum_{i=1}^{m}\sum_{k=1}^{p} P_{ik} * X_{ik}$$

Subject to

(8) $$\sum_{i=1}^{m} X_{ik} \le U_k, \quad \forall k$$

(9) $$\sum_{k=1}^{p} X_{ik} = l_i, \quad \forall i$$

(10) $$\sum_{i \in G_s}^{m} X_{ik} \le 1, \quad \forall s,k$$

(11) $$X_{ik} = [0,1]$$

The cost C_{ik} or P_{ik} refers to the cost or utility preference when course i is assigned to kth period. Constraints (8) and (9) ensure there are only l_i instructors for ith course in a cycle and the maximum number of course assigned in each period is U_k. No more than one course is assigned to a group during a timeslot is enforced by (10).

2.1.2 Lagrangian Relaxation Approach

In an effort to reduce the considerably large number of variables and constraints generated by general integer linear programming formulation, Lagrangian relaxation method has provided one of the best existing algorithms for the timetabling problem and its extensions have been used by many researchers for practical application of course or room scheduling, e.g., Tripathy [1980], Carter [1986]. It has enabled the solution of problems of practical size by incorporating parts of the constraints into the objective function. We consider the Course-Period model as an example, the relaxation is obtained by dualizing constraints (10) with non-negative λ_i.

$$L(U_i, X_{ik}) = \textbf{Min} \sum_{i=1}^{m}\sum_{k=1}^{p} C_{ik} * X_{ik} + \sum_{i=G_s}\lambda_i \sum_{k=1}(X_{ik} - 1)$$

$$= \textbf{Min} \sum_{i=1}^{m}\sum_{k=1}^{p} (C_{ik} + \lambda_i) * X_{ik} - \sum_{i \in G_s}\lambda_i$$

subject to (8), (9), (11). Each element of the vector λ is called a Lagrangian multiplier and represents a penalty for violating the corresponding constraint. The values λ_i are obtained with a subgradient optimization method embedded into a Branch and Bound procedure [Fisher 1981]. The relaxation algorithm produces a

better bound at each node, and further proves the relationship, $Z(x) \leq L(\lambda,x) \leq Z(x^*)$ where $Z(x)$ is the optimal solution to the non-integer problem (2.1.1c), $L(\lambda,x)$ is the optimal solution of the Lagrangian and $Z(x^*)$ is the required optimal integer solution to the original problem.

2.1.3 Goal Programming Approach

Linear programming problems are limited to a single objective. However on occasion when preferences and priorities are inherent in the timetabling problem, Goal programming may be appropriate technique upon which to base a scheduling algorithm [Harwood and Lawless, 1975]. Since the problem is fundamentally an assignment-type problem, the associated goal program with 0-1 variables will be an obvious formulation.

In Course-Period (Group) model, there are m courses, each course i assignable to l_i instructors; g groups of students, and a total of p periods in a cycle. The total number of instructors which can be scheduled at period k, can range from a minimum of L_k to a Maximum of U_k. For a particular group, the courses scheduled will not exceed a particular number in a given period range. d^-, d^+ represent deviational variables indicating underutilization and overutilization. The objective function in this case is:

$$\textit{Min} \sum_{i=1} \sum_{k=1} W_{1h}(d_{1i}^- + d_{1i}^+) + W_{2h}(d_{2k}^-) + W_{3h}(d_{3k}^+) + W_{4h}(d_{4i}^+)$$

Subject to

(12)
$$\sum_{k=1}^{p} X_{ik} + d_{2i}^- - d_{1i}^+ = l_i, \qquad \forall i,$$

(13) $$\sum_{i=1}^{m} X_{ik} + d_{2k}^{-} - d_{2k}^{+} = L_k, \quad \forall k,$$

(14) $$d_{2k}^{+} + d_{3k}^{-} - d_{3k}^{+} = U_k - L_k, \quad \forall i,$$

(15) $$\sum_{k \in Kp}^{p} X_{ik} + d_{4i}^{-} - d_{4i}^{+} = U_{iw}, \quad \forall i,$$

(16) $$\sum_{j=1}^{n} X_{ik} \leq 1, \quad \forall k,$$

(17) $$X = [0,1]$$
$$d_{1i}^{-}, d_{1i}^{+}, d_{2k}^{-}, d_{2k}^{+}, d_{3k}^{-}, d_{3k}^{+}, d_{4i}^{-}, d_{4i}^{+} \geq 0$$

According to the goal programming literature, each goal is assigned a weight, W_h, corresponding with respective goals' preference. Constraint (12) enforces the number of ith targeted course in a cycle, thus d_{1i}^{-} and d_{1i}^{+} are minimized. In (13) and (14), d_{2k}^{-}, and d_{3k}^{+} are minimized so as to provide a range for possible entry in each period. d_{4i}^{+} is minimized in (15) to avoid student-compactness appeared in a desired interval K_p. A system constraint (16) assures not more than one course is assigned to a group at a time.

2.1.4 Quadratic Assignment Approach

The timetabling problem, occasionally employs a different procedure referred as the Quadratic assignment problem (QAP) [Ferland and Roy, 1985]. QAP is not only nonlinear but also not unimodal. This results in a computationally difficult problem to solve than its linear counterpart. Nevertheless, the design is obligated to minimize the cost by eluding different resources to share the same facility simultaneously. An example will be the Examination (Course)-Time formulation [Lotfi and Cerveny, 1991]. Let there be a group of m examinations (courses),

i=1, 2, ..., m, i≠ i'; p days k=1, 2, ..., p, each may assign a maximum of U_k number of examinations (courses); $D_{ii'}$ =1 if ith and i' th examinations (courses) are scheduled in the same day; X_{ik} =1 when ith examination (course) is assigned to kth day, and X_{ik} = 0 otherwise. Then the formulation is given as:

$$\textbf{\textit{Min}}\ Z = \frac{1}{2}\sum_{i=1}^{m}\sum_{k=1}^{p} C_{ii'} * X_{ik} * X_{i'k}$$

subject to

(18) $$\sum_{k=1}^{p} X_{ik} = 1, \qquad \forall i,$$

(19) $$\sum_{i=1}^{m} X_{ik} \le U_k, \qquad \forall k,$$

(20) $$X_{ik} + X_{i'k} \le 2 - D_{ii'}, \quad \forall i, i', \forall k, i \ne i'$$

(21) $$X_{ii'} = [0,1]$$

The coefficient $C_{ii'}$ in the objective function refers to realization of a penalty cost when examinations (courses) i, i' are scheduled in the same day. Constraint (18) ensures that each examination (course) is scheduled to only one day. The maximum entry per day is enforced by (19), (20) which prevent the conflict examination (course) pairs scheduled in same day.

2.2 Graph Theoretic Approach

Many scheduling problems are concerned with the arrangement of events; each requiring the same duration of time, subject to the conditions that certain events may or may not take place concurrently so that total elapsed time is minimized. Some

of these timetabling problems may involve a conflict of resources. It is well known that such problems can be expressed in terms of graphs. Structurally, the timetabling can be conceptualized as a bipartite multigraph in graph theory which consisting of two sets of vertices, one set for the courses (or groups of students) and one set for the instructors [Wood, 1969], [Salazar and Oakford, 1974].

2.2.1 Edge Coloring

Consider a school has n instructors: T_1, T_2, ...,T_n and m classes (courses): C_1, C_2, ..., C_m. which requires the instructor T_i to teach class C_j for r_{ij} th period. The problem can be formulated as a graph theoretic model in terms of a multigraph G=(C,T,E). Vertices T_i and C_j are joined by parallel edges r_{ij}. If each period corresponds to a color, the problem consists in finding an assignment of one among p colors to each edge of G such that no two adjacent edges have the same color. It is an important technique in terms of Edge-coloring in bipartite multigraph.

A basic formulation of the Class(Course)-Instructor model can be described in the form proposed by De Werra [1985]. His idea relies on concepts in network flows and graphs. The objective function is ignored as the timetabling problem requires a feasible solution to the existing constraints. Let there be m classes (course), i=1, 2, ..., m; n instructors, j=1, 2, ..., n; T curriculum cycles, each with p periods, k=1, 2, ..., p. There exists a requirement matrix R=(r_{ij}), where r_{ij} is the required number of times by which ith class (course) is taught by jth instructor

(22) $$\sum_{k=1}^{p} X_{ijk} = r_{ij}, \qquad \forall i, j$$

(23) $$\sum_{i=1}^{m} X_{ijk} \le 1, \qquad \forall j,k$$

(24) $$\sum_{j=1}^{n} X_{ijk} \le 1, \qquad \forall i,k$$

(25) $$\sum_{i=1}^{m} r_{ij} = p, \qquad \forall j$$

(26) $$\sum_{j=1}^{n} r_{ij} = p, \qquad \forall i$$

(27) $$X_{ijk} = [0,1]$$

The constraints (22), (23), (24), and (27) is similar to (1) to (4). The constraints (25) and (26) enforce a timetable can always be constructed in a p period cycle, providing the largest numbers of classes; and the total number of instructors are smaller than p. Also, X_{ijk}=1 if some edge (c_i, t_j) obtains color k and X_{ijk}=0 otherwise.

In De Werra's paper, classes involving consecutive periods or requiring special rooms are characterized by preassignment. Thus X_{ijk} =1 if instructor t_j is assigned to course c_i at kth period for an instructor that is not a preassignment. The timetable constraints are:

(28) $$\sum_{i=1}^{m} X_{ijk} \le c_{ik}, \qquad \forall j,k$$

(29) $$\sum_{j=1}^{m} X_{ijk} \le b_{jk}, \qquad \forall i,k$$

(30) $$\sum_{k=1}^{p} X_{ijk} \le r_{ij}^{'}, \qquad \forall j,k$$

(31) $$r_{ij}^{'} = r_{ij} - \sum_{k=1}^{p} r_{ijk}$$

b_{ik} =1 if c_i is available and not preassigned at period k, or 0 otherwise.

c_{jk} =1 if t_j is available and not preassigned at period k, or 0 otherwise

r_{ijk} =1 if there is a preassignment meeting of class c_i and t_j at period k, or 0 otherwise.

r'_{ij} = the required number for ith class (course) meet jth instructor that is not a preassignment.

For some special cases it can be decided whether such a timetable exists or not. It has in fact been shown that the problem of finding whether a solution exists, is NP-complete [Even, Itai and Shamir, 1976]. However, the central idea is the foundation to other combinatorial formulation of the timetabling problems.

White and Wong [1988] further combine the room capacity and student enrollment factor into the formulation Let there also be q rooms, v=1, 2, ..., q and two vectors, E, an m-vector whose ith element is the enrollment in class i and C, a q vector whose ith element gives the capacity of the room. The problem is to decide whether there is a meeting function

$$S = \{X_{ijkv}\}:\{1,...,m\}x\{1,...,n\}x\{1,...,p\}x\{1,...,q\} \rightarrow \{0,1\}$$

where $X_{ijkv} = 1$ if and only if jth instructor meets ith classes in qth room during kth period, such that

(32) $$\sum_{k=1}^{p}\sum_{v=1}^{q} X_{ijkv} = r_{ij}, \forall i, j$$

(33) $$\sum_{j=1}^{n} X_{ijkv} \le 1, \quad \forall i, k, v$$

(34) $$\sum_{i=1}^{m} X_{ijkv} \le 1, \quad \forall j, k, v$$

(35) $$\sum_{v=1}^{q} X_{ijkv} \le 1, \quad \forall i, j, k$$

(36) $$\sum_{j=1}^{n}\sum_{k=1}^{p} E_i * X_{ijkv} \le \sum_{j=1}^{n}\sum_{k=1}^{p} C_i * X_{ijkv}, \ \forall i, v$$

The constraints (32) (33) and (34) bear the same meaning as in (22), (23) and (24). The constraint (35) enforces that each instructor-period-class occupies at most one room. Constraint (36) specifies that all rooms should be large enough to contain the classes scheduled into them.

2.2.2 Vertex (Node) Coloring

Graph theory has been applied in assigning a set of events, e.g., courses or examinations to a fixed number of periods, the target is to minimize the total number of student conflicts; or to make assignments to minimize the number of periods under the restriction that no conflicts occur. The vertices (nodes) of the graph are used to represent the events, and a pair of vertices with conflicting resource are joined by an undirected edge. The timetabling problem can be formulated as a graph theoretic model in terms of Vertex (Node) Coloring. A solution of this form is called Conflict-free assignment. This kind of formulation is extensively adopted in Examination-Period problem that is primarily an analog to the Course-Period problem, e.g. Barham and Westwood [1978], Dowsland [1990]. Periods in Course timetabling are often overlapped and have varying lengths. If a practical course timetabling problem requires a conflict-free schedule in uniform time periods and with probably shorter total time committed, an examination timetabling algorithm can be used [Carter, 1986].

A minimum vertex coloring algorithm is a method to allocate colors to the vertices such that no two adjacent vertices can have the same color, two edges being

adjacent if they stem from the same vertices in the graph. The determination of the minimum number of time-periods required for the schedule is therefore the same as finding the minimum number of colors required for the graph. This unique integer, χ (G), is known as the node-chromatic number of the graph, G, or simply the chromatic number. It is subsequently known that it is NP-hard [Karp 1972] to solve this type of coloring problem within a ratio r >2. Furthermore, no polynomial time algorithm is known which solves the coloring problem within any fixed ratio r. For large random graphs, Grimmett and McDiarmid [1975] have shown that most graph coloring heuristic can use at most 2χ colors that is sufficient to find conflict-free schedules. If the number of periods p is much larger than χ, the problem of assigning p conflict-free periods becomes relatively easy. Many researchers and mathematicians have proposed various algorithms in finding the χ, but the number is not guaranteed to be a minimum. A general survey of graph coloring algorithms can be found in Carter [1986], Arani and Lotfi [1989].

2.3 Computer Interactive and Heuristic Methods

Timetabling has frequently been described as " The art of the possible " [Salt, 1978], which implies that finding a feasible solution rather than possessing an optimal solution may become its main goal. As in most realistic situation, there is not any overriding objective that must be optimized to the exclusion of all else. Rather there are numerous flexible goals that are usually seen as desirable ends rather than critically important objectives. A computer solution permits flexibility and suggests a better indication of how resources are utilized but it may not provide the best optimal

results. Accounts of some of this research are provided in Almond [1965], Lions [1967], Aust [1976], Papoulias [1979], Selim [1983], Loo, Goh and Ong [1985], Glassey and Mizrach [1986].

The heuristic approaches can be described as assignment algorithms, of which most of their selection rules are deterministic. The algorithm first starts with a blank timetable and makes class-instructor assignments to satisfy complex conditions. It will examine all the hard and soft constraints to achieve a solution, and then reassign classes accordingly to resolve the difficulties in order to obtain better allocation. Sometimes, a reserved time slot that causes difficulties in an allocation process will be relaxed to obtain a possible solution, and to modify other soft constraints in the input data file if necessary. The terms hard constraints or critical conflicts are referred to some activities with common resources that are generally not acceptable under any circumstances, e.g., the requiring of participants to be in two places simultaneously. The soft constraints or non-critical conflicts are those clashes which one would like to avoid; depending on the exact conditions of time and locations, they are of local origin and may vary greatly. Timetables that satisfy all the hard constraints are called feasible timetables.

In early sixties, moderate achievements have been reported on such programs. One of the major problems is to decide when an assignment is being entered which makes the completion of the timetable almost impossible. In De Gans [1981] paper, three deterministic heuristic strategies are suggested.

1) One-pass procedure - methods that assign requirements in a permanent way, which means that no assignment will be withdrawn. If a requirement cannot be assigned it will be skipped

2) N-Step Backtracking procedure -- assignment procedure continues until an infeasible element turns out, as many assignments as necessary are committed in reverse order of scheduling to find an assignment for the conflicting one. Thereafter the scheduling process continues, extending to an enlarged set of non-assigned requirements.

3) Reassignment procedure -- methods in which as soon as the next requirement turns out to be infeasible, one or more assigned requirements are reassigned to create a feasible assignment for the conflicting one, so that the number of assigned requirements will not be increased.

There is no conclusive test to measure which strategy will generate the best result. Using One-pass procedure will not have a complete schedule since an infeasibility may arise. On the other hand, if the final number of infeasibilities is in a tolerance range, the procedure will be very effective and time saving. Both N-Step Backtracking and Reassignment procedures are very expensive in computer time and data storage to obtain a feasible schedule. The latter mostly demand additional reassignment procedures which require another heuristic to cover less explicit constraint.

The interactive programs essentially perform the hand computations of constructing a timetable [Ferland and Roy, 1982], [White and Wong, 1988], [Chahal and De Werra, 1989]. It can be used interactively by the decision maker who considers the constraints and objectives not directly built into the model. The systems are usually composed of two phases. Users will initially prepare a database in the batch-oriented phase and then modify the database that is created earlier, in the

interactive phase. In real-world situations, an interactive algorithm consists of a mathematical or heuristic design in the final phase. This helps to prevent a complete rerun of the algorithm with updated information that would generate a new timetable entirely different from the previous result due to the last minute changes.

2.4 Timetabling in Practice

Throughout the period of growth in the educational system, many researchers have contributed their efforts to the timetabling problems. Most of the published literature is concerned with applying the timetabling algorithms to the real world problems, for example Busam [1967], Brittan and Farley [1971], Winters [1971], Loo, Goh and Ong [1985], Sabin and Winter [1986].

2.4.1 Earlier Approaches

The computer has been found as an obligatory element in nearly all automated timetabling systems and its performance is intensely affected by the devices at hand. Timetabling problems using computer programs have been discussed in the early sixties; however their algorithms' formulations were obstructed by the limitations of data storage space and the cost of computer time. Most of the programs developed are intimately related to the simulating manual methods in school problems, Appleby, Blake and Newman [1961] and layout design as in Lewis [1961]. The term layout is used to recapitulate the specification in a certain curriculum for a group of students showing the subjects to be taught at a given cycle (time period).

The earliest mathematical framework introduced by Gotlieb [1962] at the 1962 Munich IFIP Congress reviews dissimilar availability conditions. His perception has influenced most of the research in analytical aspect of the timetabling problem. The terminology in his problem formulation is based upon set and graph theory to initiate the availability array. A complete availability array can be described as a set of two dimensional tables, with factors of different level across each row and column. Separate availability arrays are maintained to speed up checks on the readiness of various instructors and courses. Room factors are not incorporated in the main array. This algorithm demands significant computing time and data storage space. Complex situation is ignored and a simple situation with nine instructors, nine classes, nine hours and the requirements that each instructor meet each class once is exhibited.

A theoretical extension in Csima and Gotlieb [1964] further develops the necessary and sufficient condition for the earlier model [Gotlieb, 1962] including the non-conflicting mode. Heuristic approach with look-ahead feature is added into Gotlieb algorithm. The problem must first break down into daily problems to reduce computation time. Conflicts are avoided where conceivable by careful checks before the data is inserted, to ensure that they will not occur. Lions [1967] modifies the above algorithm into a two-phased approach by incorporating the Hungarian method to manage lunch break, room assignments and other special requirement. When a conflict cannot be avoided, the initial constraints are removed progressively until the difficulty is eliminated.

Barraclough [1965] outlines another method by employing two-dimensional arrays for instructor, class room availability, and for the main timetable itself. Problems concerned with consecutive periods in school, e.g., Double periods are

heeded. When Conflicts do occur, they will be removed in the existing partial timetable. The displaced entries are stored for the subsequent assignment either in the normal way or by interchanging other entries to permit the insertion of those displaced. This algorithm does not readily yield information about which requirements are likely to schedule with difficulties. It also suggests an earlier schematic prototype in course selection by set partitioning with a bound when r chosen subjects are selected from a total of n subjects, with $r < n$ and $n \leq 13$.

Idea of three-dimensional Boolean array is first implemented in Almond [1965] to a university timetable problem. Her assignment heuristic algorithm uses three two-dimensional arrays (see Figure 3) to form the three rectangular faces of the timetable structure, and each timetable is stored as a two-dimensional array showing the status of every requirement at each period.

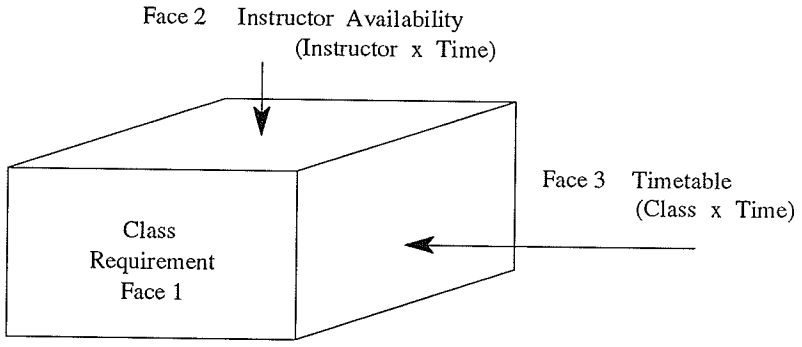


Figure 3, A Three Dimensional Boolean Array, Almond [1965]

A class requirement matrix whose elements give the number of periods an instructor meets a class. The instructor requirement matrix contains Boolean element to the availability of an instructor in a particular time interval. Timetables are set out

initially for each department in which courses for each class are designed, and then they are amalgamated to form the master timetable for the faculty. Availability and time preference of the instructors are not examined and the program may run into an infinite unclosed loop. In Almond [1969], a modification of her earlier model is illustrated by shrinking the data storage facility, evaluating priority of the non-conflict constraints and enhancing the final layout. The timetable structure can accommodate minor changes from year to year that may include staff changes. A similar problem is studied by Selim [1983] in which an algorithm similar to Almond's heuristic is proposed. His algorithm creates course and instructor timetables in a small store computer with saving storage space as its prime target. Nevertheless the problems of conflicting course and unassigned class are still unresolved.

The two-phased heuristic approach in Papoulias [1979] is designed to tackle complicated timetabling situation like settings and multiple periods in the assignment-to-days problems. It provides a total period balance for medium size schools in the range of $(N+\varepsilon 1)$, where $0 \leq \varepsilon 1 \leq 10$, and N is the number of periods over the week divided by the number of days. It shows partial success and problems arose when the daily preassignment situations become dominant.

2.4.2 The Operations Research Approach

We now discuss the problems associated with implementation of timetabling methods developed through various disciplines of operations research. This section presents a brief review of their applications which are divided into three groups, namely (a) Mathematical Programming and Pure Coloring Approach, (b) Network

Flow Analysis, (c) Computer Simulation Approach. Some of their findings have been extended, with varying degrees of success to the timetabling problems.

2.4.2a Mathematical Programming and Pure Coloring Approach

Modeling the problem as an integer programming problem has not been particularly successful in Lawrie [1969], Akkoyunlu [1973]. An integer programming formulation first exploited in Lawrie's paper is used for solving problems expressed with layouts design by Lewis [1961]. It admits classes of one period's duration but allows setting through the use of year group layouts. The problem does not address the case for those classes that occur at fixed time and require a block of consecutive periods. An ad hoc rounding procedure is adopted to obtain the approximate integer solution. Aust [1976] further extends Lawrie's idea in a three-stage conceptual model with emphasis on the decision factors of breaks and feasibility. When breaks turn up, an improvement algorithm is used to reduce infeasibilities by interchanging entries within timetables. The design presumes every subject and instructor availability in each time segment to be constant.

The intimidating numbers of variables and constraints in the analytical approach often generate practical difficulties during their problems' formulations. In an effort to reduce the complexity, researchers have suggested several different decomposition possibilities which may allow the resources to be perused in separate phases or stages. The phases of faculty member or instructor assignment and the course scheduling problems are well documented in different areas. Harwood and Lawless [1975] focus upon the development of mathematical constraints for various

faculty members' requests in a small department. They enhance the mixed-integer goal programming technique by addressing the priority of teaching load, the time component to their assignment of teaching schedules. The objective attempts to sacrifice the minimum cost according to the suggested priority levels given by the decision makers. All class times are assumed fixed in advance of the assignment.

Shih and Sullivan [1977] advocate a two-stage binary optimization model where multiperiod course scheduling is applied. The procedure in the first stage is to assign faculty member to course, followed by assigning timeslot to classes in the second stage. Weight is allocated to faculty member to reflect the degree of familiarity with a given course, educational background, research interest and experience. Their objective is to maximize faculty members' utilities for their time, subject to the constraints that limit the number of courses and sections offered during a particular time. It focuses on the capacity of capturing some dynamic aspects of teaching and research activities; nonetheless preferential dependence between schedules in adjacent terms is being ignored. They assume no class times should be fixed in advance. When the faculty member assigns utilities to the courses in a schedule, it turns out to be a problem since there is no prior knowledge on what the schedules in adjacent terms would be. In McClure and Wells [1984], they make use of a single period model which allow faculty members to assign utility values to a list of preset feasible schedules. The objective is to maximize the faculty utilities to the present schedules instead of the courses. However, both attempts are conducted in a relatively small scale with partial success.

Barham and Westwood [1978] establish their pure coloring algorithm for timetabling course under an elective system. The problem involves a small group of

students in a business school; each student is required to select five to six courses. Options are first scheduled into groups before the start of the selection process. Their strategy is to build a conflict-free schedule based on first r out of n choices and the remaining options requiring adjustment. The proposed algorithm is a "largest degree first; fill from top" method. Each individual chooses the first four courses from each group, then adjusts his or her fifth and sixth options where necessary.

Subsequently, Tripathy [1980] develops some results based on the Barham and Westwood data. His framework uses a binary integer programming to maximize the desirability of different instructors. Problem is solved by Lagrangian relaxation algorithm with a subgradient optimization approach, namely out-of-kilter algorithm. The result presents a different interpretation to the original idea. Tripathy [1984] makes a second attempt to solve a graduate course scheduling problem using a similar technique. His central idea involves finding a conflict-free course schedule for a one-year graduate program. Subgradient method is used first, then a branch and bound procedure is utilized to generate the solution. It shows partial success and requires further research. In both cases, the numbers of constraints and variables become recalcitrantly large when they are expressed mathematically. In an effort to obtain feasible solutions, Tripathy tries to identify special groups of students who are in the same stream or specialization. The difficulties also force him to define grouping operations for subjects and rooms with similar attributes.

Any timetabling system will guarantee that there are enough rooms of appropriate size to hold all the classes at all time periods. Recently, several classroom assignment algorithms have appeared seeking out an acceptable room for each class at a specified time interval. Gosselin and Truchon [1986] have synthesized the

optimization model to minimize the penalty cost in a daily classroom allocation problem. Rooms in a large building at Universite' Laval are grouped into different categories. It resembles a transportation problem with a bipartite graph structure. Rooms are assigned by an automated procedure to the requests on a first come first served basis, and a backtrack system is used to monitor the infeasible entry. Geometric penalty cost has been set to those that either departs from the requested rooms, or assigned to other fictitious rooms.

Glassey and Mizrach [1986] propose a 0-1 classic linear optimization model in their class to room problem. In the model, the objective is designed to minimize the total cost incurred from empty seats, traveled distance, and underutilized facilities in a university. This cost function demonstrates a more realistic judgment than that emphasizes only on the penalty values of unoccupied seats. An approximate solution is procured by Greedy algorithm with look-ahead features. The heuristic is founded on the idea of "Solve the hardest remaining problem next". It will allot the highest ratio of unscheduled classes to vacant rooms; the mechanism will continue with classes in order of decreasing size or interchange the assignment of least cost from the currently vacant rooms. Within each time slot, they try exchanges of up to three classes concurrently. Cost is estimated in form of $C(r-c)/(c\sqrt{n})$, where r is the room size; c is the class size with $r \geq c$, and n is the number of vacant rooms containing at least as many seats as the class enrollment. C, is the cost parameter where $0 \leq C \leq 100$, depends on the cost of extra seats. However, the time domain is not considered as a decision variable and the algorithms fail to deal with the large number of nonstandard classes.

2.4.2b Network Flow Analysis

It is prevalent to model a timetabling problem as a network flow structure. The advantage of this concept is its unimodular form. This signifies that any intermediate or optimal solution will be naturally integer-valued, whilst the lower and upper bounds in the constraints are integer-valued. Dyer and Mulvey [1976] first devise their multiperiod model in this underlying structure, and solve it successfully by an information system. The study addresses to the difficulties of assigning faculty members to specific classes, course sections and other activities in a graduate school. These model constraints ensure that the needed numbers of sections of each course are taught and it has an upper bound on the number of course sections in each faculty member's teaching load.

Bloomfield and McSharry [1979] recapitulate a two-stage heuristic scheduling mechanism as an extension of Dyer and Mulvey's idea. In the heart of the system is a network framework with emphasis on faculty preferences rather than on student needs. It is obviously an exclusive design to their own faculty. In their first stage, each class section will have an approximate position reserved in a large category in the cross-product space of days, times and room types. The corresponding section will be refined to the specific time, rooms within each category in the next stage. Mulvey [1982] places special emphasis on the inherent network structure to the basic course scheduling problem. A linear programming formulation is used to maximize the utilization of classroom space when assigning room into fixed or varied time slots. The model does not stress fully on the issue of classroom and time slot availability. His approach is not feasible in a large practical problem. Complexity in the realistic

model is reduced through a man-machine interactive solution, however, its final solution requires manual adjustment and does not achieve satisfactory implementation.

Another computer interactive system based on network flows is proposed by Chahal and De Werra [1989]. A decision support system is designed to develop feasible timetables in an adult training school with three classes. It allows different instructors to give lessons on the same topic to the same class. The algorithm provides a fairly good solution with some manual adjustments in the final phase. In Dinkel, Mote and Venkataramanan [1989] pure network model, it highlights on the body of faculty, course, classrooms and time of day. Penalty structures are set from faculty and subject node to room and time in the objective constraints. The target is to minimize the relative weights that are assigned to the objective function components to reflect the preferences of the decision maker. Difficulty arises when an instructor is allowed to select courses with preference on class and time. This results in overstatement of enrollment in attempting to capture a larger or more preferred room.

Carter [1989] successfully solved a weekly classroom assignment problem based on Lagrangian Relaxation technique. The original problem is first decomposed into fourteen subproblems, each represents a block of a day to keep the number of variables and constraints in a reasonable amount. The model is similar to that of Mulvey [1982] only except the subproblem has a network structure. At each iteration, the network will be penalized to account for infeasibilities. An ameliorated cost function founded in Glassey and Mizrach [1986] with additional factors, such as

room's preference, non-standard timeslots and room fragmentation, are also taken into account.

2.4.2c Computer Simulation Approach

Simulation is a technique that has long been an important tool of the decision makers in operations research studies. It can be referred also as a controlled statistical sampling technique for estimating the performance of complex stochastic systems when analytical models may not suffice. Recently, there has been considerable interest in the use Monte Carlo scheme called simulated annealing as an optimization technique in timetabling problem. Its groundwork is based on physics of annealing where at high temperature a collection of hot vibrating atoms is free to move around with random displacements until cooling down. The assignment process becoming stable if the probability of the atom reaches a predefined selected value known as the temperature parameter, which would decrease the probability as the process progresses. After several successful permutations the temperature is decreased by a cooling rate.

In timetabling problem, the phenomenon of atom displacement is simulated as an assignment of course and instructor to each cell of a timetable. An initial schedule is created by scheduling the timetable elements in a randomly chosen period. Both the initial costs C_o and temperature T_o are computed. The cost is used to reflect the quality of the timetable, while the temperature is used to control the likelihood of an increase in cost. This approach employs a simple swapping mechanism on the feasible solution set with predefined neighborhood structure. A feasible solution is exchanged

for improvement by swapping from the current position to a randomly selected neighbor. New movement will be made if a better solution is obtained, otherwise the movement will be only accepted according to the predetermined probability distribution. Generally, the Boltzmann distribution is used. A negative Δc indicates that a lower cost is found; if Δc is positive, a change for the worse of magnitude Δc will be accepted only with probability $e^{-\Delta c/t}$. The rate at which t is reduced is critical to the success of the method and is known as the cooling schedule. This mechanism will be terminated either when there is no further improvement or a defined T value is reached.

Eglese and Rand [1987] employ the improvement method using annealing algorithm to formulate an assignment model in which the objective is one of minimizing disappointment. Dowsland [1990] implemented the underlying technique in a university by minimizing the single objective of student disappointment in the optional course selection at the final year. The students are permitted to select the courses at the beginning with no restriction, then the options are rescheduled into six independent groups with less inter-group conflict after the final iterations. Educational flexibility may be reduced since the students are selecting their options from each of the defined groups to minimize the clashes. Abramson [1991] applies the sequential algorithm to a timetable scheduling of course and instructor in a high school with promising result. The structure is further extended into a modified prototype with parallel algorithm which may speed up the procedure but its performance obligates further research work.

Nevertheless, there is a common phenomenon for most of the simulated annealing designs of which their algorithms fully rely on a random set of

permutations. It may not guarantee that the true minimum cost value is found, or that two different annealing runs will yield similar solution. In addition, the annealing algorithms generally provide a shorter algorithm but they show exponential time complexity as the numbers of factors increase.

2.4.3 Models in the Nineties

Toward the end of eighties, there is probably no field advancing more rapidly than the computer industry. Both computer systems and specialized software packages have been developed enormously. Some hardwares, e.g., Workstations, have been designed for intensive data processing needs and retain the potential to take over the roles traditionally held by costly mainframes in business and academic computing. This kind of evolution has furnished the environment for the researchers to diversify their studies in timetabling problems. They range from the simple database management system to artificial intelligence. Dhar and Ranganathan [1990] explore their study in expert system that is one of the Artificial Intelligence research areas. This is a set of computer programs designed to solve problems in a narrow domain by using domain-specific knowledge at a level of performance that is comparable to that of experts. Unlike mathematical programming, there is no global objective function in expert system to acquire any optimal solution. The problem solving mechanism will rather seek a solution to tie up with a series of local decisions that are most preferred at each point. In studying the problem of assigning faculty members to courses, they exhibit an experimental comparison between symbolic and Integer programming for their formulations, feasibilities, and performances.

Kang and White [1992] utilize another kind of knowledge-based software package called Prolog, in their computerized timetabling area. It is a non-numeric programming language revolved around logical reasoning. The algorithm consists of its own grammar rules and the resolution of constraints framed in first order logic clauses, such as Modularity (adding or moving clause), Portability, Feasibility (no procedural specifications). Twelve hard and six soft timetabling constraints are translated syntactically into Horn Clause forms which are simple Prolog rules that contain no negation or disjunction. The timeslot is then standardized into four types; each assumed to have a fixed length and course format. Feasible solutions are achieved from the particular set of applications.

A Markovian neural network approach in Kovacic [1993] is used to tackle a high school timetabling problem on a desktop computer. The neural network that correlates to the simulation of nervous cells in human brain, is comprised of n neurons; each represents one element of state vector and is connected with other neurons by way of bidirectional links. In a school timetabling problem, the neuron represents a subject to be scheduled. Every neuron has r by p states which correspond to possible entries in the timetable. Its objective is to minimize the cost of the solution which equivalent to the number of unsatisfied constraints in the timetable problem. The operation of the Markovian neural network searches stochastically the state space defined by assignments of subjects; it works asynchronously with only one neuron can change its state at a time. Subjects are scheduled in the timetable randomly in the initial state. The underlying system performs the swapping mechanism which resembles to that of simulated annealing to exchange for improvement. After a steady state is detected, the process will be

completed. However the result cannot guarantee that the true optimal solution is found since it's stochastic in nature.

2.5 Summary

This chapter has recapitulated a general survey to both university and school timetabling systems including their mathematical models and applications. In most cases, the local criteria discussed earlier inhibit the practical implementation of programs in a real world situation.

The first group of algorithms uses mathematical programming to minimize a suitably defined objective function. These algorithms generally create solutions that fulfill a family of goals but usually cause other problems specific to the institution involved. The problem formulation is combinatorial and therefore the number of constraints and variables appeared in most optimization models happen to be astronomical large in realistic scheduling problems.

The second group uses graph theory methods to estimate an exact solution to the various coloring problems formulated. Models developed on graph theory have often incurred the inherent computational difficulties since the timetabling design are classified as Nondeterministic polynomial time complete problem even in their very restricted versions. Work to date has mostly concentrated on finding heuristic approximations to give reasonable approximation to the final solution within a reasonable amount of computing time. In addition, both mathematical and coloring methods necessitate strong mathematical background and highly sophisticated computer device to support their implementations.

As a consequence, the complexity encourages the alternatives to elaborate heuristic techniques, interactive systems and other computer designs, such as simulation, Artificial Intelligence, etc., in the automated timetabling systems. In essence, most of computer solutions may not provide the optimal results, but permit flexibility and eliminate many of the laborious procedures associated with manual systems. The computational benefits are unquestionably attractive; nonetheless several precautions should be taken into account. Except for universities, most schools in other educational institutions, in spite of their financial limitations, still have access to low-end microcomputers of limited capacity that would almost certainly be inadequate to run those programs discussed earlier. The development and maintenance of various programs generally involve computer professionals; most institutions would reckon it simpler and more economical to construct a timetable manually with the help of an experienced timetabler.

Chapter 3

DATABASE APPROACH

A database is an organized repository of data whose structure consists of a series of columns denoted as fields and rows denoted as records. It provides a simple way of extracting information efficiently to meet the specific criteria based on some chosen reference points. The computer tools designed to offer this task of expediently and constructively managing the masses of data are called database management systems (DBMS), which are broadly utilized in the business and academic fields. Sophisticated database management systems have been available on mainframes and microcomputers for over two decades. However, their adaptations to the timetabling problems are limited. Few published papers have appeared in this area. In this chapter, we outline a brief discussion of Johnson's database approach to the timetabling problem. Its limitation will be examined through a case study.

3.1 Johnson's Database Approach to Course Timetabling

Johnson [1993] develops a comprehensive DBMS to manage timetables in a university. The system is able to handle any conflict when a trivial change occurs. Furthermore, it can perform the bookkeeping task to produce printouts and reports of the final schedules. The design is founded on the criteria that the Thinking and Planning stages have formerly been completed; hence the relevant data have already

been reserved in various database files. This process will radically distinguish all the feasible allocations that turn up in a certain time interval.

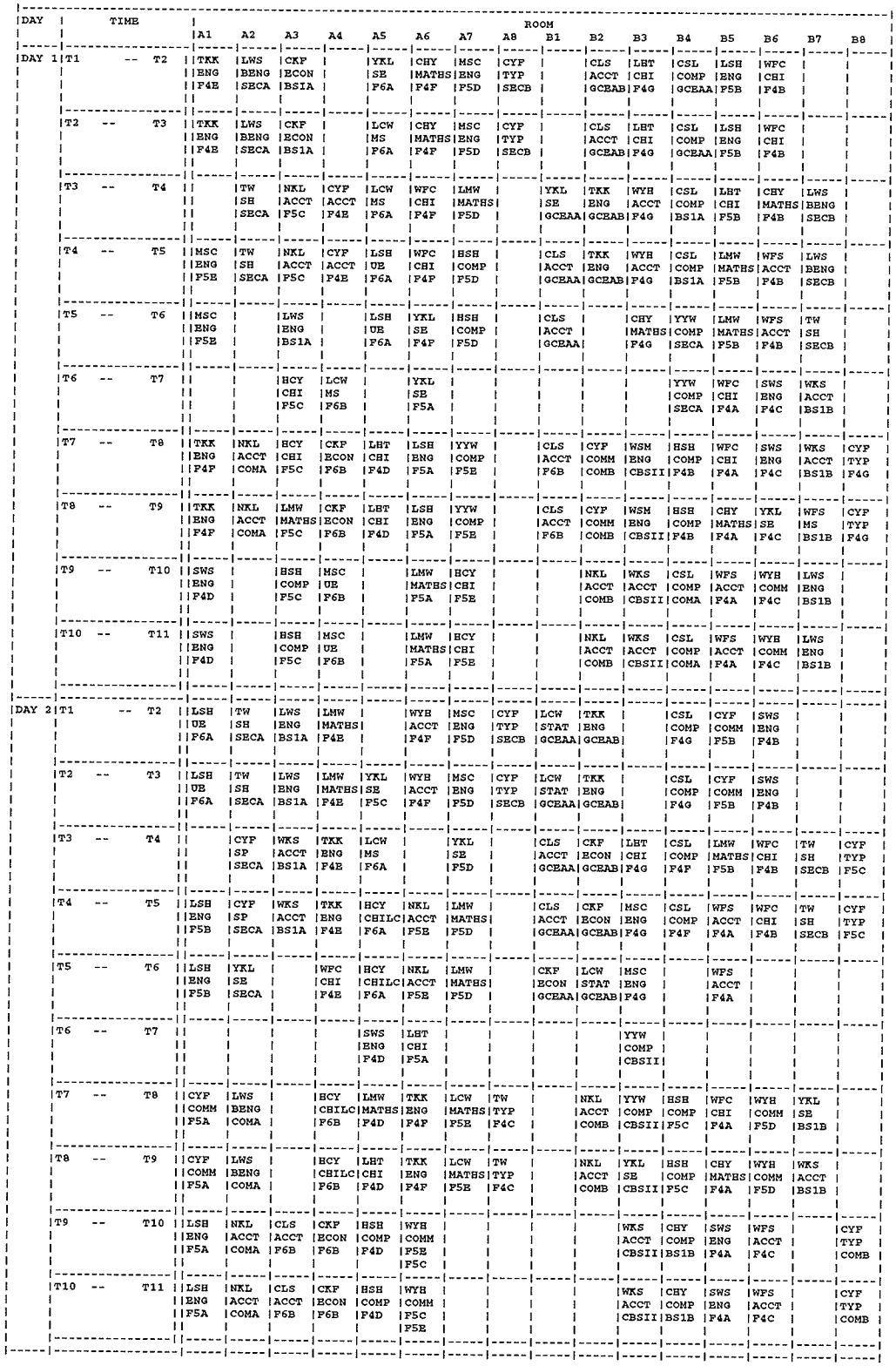
In most DBMS designs, there is a common procedure termed as normalization, that breaks down the input information set into smaller file structures with correlated key fields. The purpose is to minimize the redundant data entry and to maximize the data integrity. According to Johnson's design, the source data set is normalized into a main file called Meetings File and four additional information files, namely Structure File, Degree Names File, Course Names File, and Lecturer Names File. The meetings file contains all the information necessary to describe all the courses offered during a given time interval. It will change virtually from term to term. Each meeting consists of a record containing field for:

- Course Code
- Day of the week
- Time of day (or period number)
- Class Type (e.g. Lecture/Tutorial/Practical)
- Lecturer(s) Initials
- Room (or location) where the meeting takes place.
- Student Notes
- Lecturer Notes

Full names and other detailed descriptions for clarity of presentation are spread and reserved in the four subsidiary files, in which the data are preferred to be more stable and would only have minimal changes from term to term. During data querying, retrieving and printing, the files can be temporarily linked through the

correlated key field in a way that cross-reference information can be obtained without factually merging the data. Main outputs from the system are predominantly student and staff schedules including other ancillary report forms, such as teaching summary, teaching allocation forms, and room booking forms.

With the implementation of this database approach, Johnson's idea is subsequently adopted in a case study [Wong 1993] where the similar conditions and the mentioned assumptions are held. A database is obtained from an adult education institution in Hong Kong where it has 23 classes using 16 rooms in a 10-period daily schedule, and 5 days per scheduling week. In practice, the timetable planning is done manually in the case under study. However, the system developed here is applied to filter any undiscovered human error that will incur infeasibility to the scheduling process. The design is indispensably modified to suit the local environment by adding two extra features into the model. First, two additional Class fields are added into the meetings file. This modification may allow the Interclass merging where students from different classes to be grouped together for common lectures at the same time, or the Intraclass splitting where students in the same class are subdivided into subgroups for different lecturer and tutorials (for example, see the resulting master timetable in Table 1, in which each cell contains the instructor initials, subject codes, and two class codes). Second, a small program related to the above features is introduced into the data entry procedure. It provides duplicate records check (see Figure 4) to indicate whether there is any violation of critical conflicts. As an example, two different classes with different instructors scheduled to the same room simultaneously. The routines are written in a dBase language and tested in the existing microcomputer system with limited supporting hardware.



DAY	TIME	ROOM A1	A2	A3	A4	A5	A6	A7	A8	B1	B2	B3	B4	B5	B6	B7	B8
DAY 1	T1 -- T2	TKK	LWS	CKP		YKL	CHY	MSC	CYP		CLS	LBT	CSL	LSH	WFC		
		ENG	BENG	ECON		SE	MATHS	ENG	TYP		ACCT	CHI	COMP	ENG	CHI		
		F4E	SECA	BS1A		F6A	F4F	F5D	SECB		GCEAB	F4G	GCEAA	F5B	F4B		
	T2 -- T3	TKK	LWS	CKP		LCW	CHY	MSC	CYP		CLS	LBT	CSL	LSH	WFC		
		ENG	BENG	ECON		MS	MATHS	ENG	TYP		ACCT	CHI	COMP	ENG	CHI		
		F4E	SECA	BS1A		F6A	F4F	F5D	SECB		GCEAB	F4G	GCEAA	F5B	F4B		
	T3 -- T4		TW	NKL	CYP	LCW	WFC	LMW			YXL	TKK	WYH	CSL	LBT	CHY	LWS	
			SH	ACCT	ACCT	MS	CHI	MATHS			SE	ENG	ACCT	COMP	CHI	MATHS	BENG	
			SECA	F5C	F4E	F6A	F4F	F5D			GCEAA	GCEAB	F4G	BS1A	F5B	F4B	SECB	
	T4 -- T5	MSC	TW	NKL	CYP	LSH	WFC	BSH			CLS	TKK	WYH	CSL	LMW	WFS	LWS	
		ENG	SH	ACCT	ACCT	UE	CHI	COMP			ACCT	ENG	ACCT	COMP	MATHS	ACCT	BENG	
		F5E	SECA	F5C	F4E	F6A	F4F	F5D			GCEAA	GCEAB	F4G	BS1A	F5B	F4B	SECB	
	T5 -- T6	MSC		LWS		LSH	YKL	HSH			CLS		CHY	YYW	LMW	WFS	TW	
		ENG		ENG		UE	SE	COMP			ACCT		MATHS	COMP	MATHS	ACCT	SH	
		F5E		BS1A		F6A	F4F	F5D			GCEAA		F4G	SECA	F5B	F4B	SECB	
	T6 -- T7			HCY	LCW		YXL							YYW	WFC	SWS	WKS	
				CHI	MS		SE							COMP	CHI	ENG	ACCT	
				F5C	F6B		F5A							SECA	F4A	F4C	BS1B	
	T7 -- T8	TKK	NKL	HCY	CKP	LBT	LSH	YYW			CLS	CYP	WSM	HSH	WFC	SWS	WKS	CYP
		ENG	ACCT	CHI	ECON	CHI	ENG	COMP			ACCT	COMM	ENG	COMP	CHI	ENG	ACCT	TYP
		F4F	COMA	F5C	F6B	F4D	F5A	F5E			F6B	COMB	CBSII	F4B	F4A	F4C	BS1B	F4G
	T8 -- T9	TKK	NKL	LMW	CKP	LBT	LSH	YYW			CLS	CYP	WSM	HSH	CHY	YXL	WFS	CYP
		ENG	ACCT	MATHS	ECON	CHI	ENG	COMP			ACCT	COMM	ENG	COMP	MATHS	SE	MS	TYP
		F4F	COMA	F5C	F6B	F4D	F5A	F5E			F6B	COMB	CBSII	F4B	F4A	F4C	BS1B	F4G
	T9 -- T10	SWS		HSH	MSC		LMW	HCY				NKL	WKS	CSL	WFS	WYH	LWS	
		ENG		COMP	UE		MATHS	CHI				ACCT	ACCT	COMP	ACCT	COMM	ENG	
		F4D		F5C	F6B		F5A	F5E				COMB	CBSII	COMA	F4A	F4C	BS1B	
	T10 -- T11	SWS		HSH	MSC		LMW	HCY				NKL	WKS	CSL	WFS	WYH	LWS	
		ENG		COMP	UE		MATHS	CHI				ACCT	ACCT	COMP	ACCT	COMM	ENG	
		F4D		F5C	F6B		F5A	F5E				COMB	CBSII	COMA	F4A	F4C	BS1B	
DAY 2	T1 -- T2	LSH	TW	LWS	LMW		WYH	MSC	CYF	LCW	TKK		CSL	CYP	SWS		
		UE	SH	ENG	MATHS		ACCT	ENG	TYP	STAT	ENG		COMP	COMM	ENG		
		F6A	SECA	BS1A	F4E		F4F	F5D	SECB	GCEAA	GCEAB		F4G	F5B	F4B		
	T2 -- T3	LSH	TW	LWS	LMW	YXL	WYH	MSC	CYP	LCW	TKK		CSL	CYP	SWS		
		UE	SH	ENG	MATHS	SE	ACCT	ENG	TYP	STAT	ENG		COMP	COMM	ENG		
		F6A	SECA	BS1A	F4E	F5C	F4F	F5D	SECB	GCEAA	GCEAB		F4G	F5B	F4B		
	T3 -- T4		CYP	WKS	TKK	LCW		YKL		CLS	CKP	LBT	CSL	LMW	WPC	TW	CYP
			SP	ACCT	ENG	MS		SE		ACCT	ECON	CHI	COMP	MATHS	CHI	SH	TYP
			SECA	BS1A	F4E	F6A		F5D		GCEAA	GCEAB	F4G	F4F	F5B	F4B	SECB	F5C
	T4 -- T5	LSH	CYP	WKS	TKK	HCY	NKL	LMW		CLS	CKP	MSC	CSL	WFS	WPC	TW	CYP
		ENG	SP	ACCT	ENG	CHILC	ACCT	MATHS		ACCT	ECON	ENG	COMP	ACCT	CHI	SH	TYP
		F5B	SECA	BS1A	F4E	F6A	F5E	F5D		GCEAA	GCEAB	F4G	F4F	F4A	F4B	SECB	F5C
	T5 -- T6	LSH	YXL		WFC	HCY	NKL	LMW		CKP	LCW	MSC		WFS			
		ENG	SE		CHI	CHILC	ACCT	MATHS		ECON	STAT	ENG		ACCT			
		F5B	SECA		F4E	F6A	F5E	F5D		GCEAA	GCEAB	F4G		F4A			
	T6 -- T7					SWS	LBT					YYW					
						ENG	CHI					COMP					
						F4D	F5A					CBSII					
	T7 -- T8	CYP	LWS		HCY	LMW	TKK	LCW	TW		NKL	YYW	HSH	WFC	WYH	YXL	
		COMM	BENG		CHILC	MATHS	ENG	MATHS	TYP		ACCT	COMP	COMP	CHI	COMM	SE	
		F5A	COMA		F6B	F4D	F4F	F5E	F4C		COMB	CBSII	F5C	F4A	F5D	BS1B	
	T8 -- T9	CYP	LWS		HCY	LBT	TKK	LCW	TW		NKL	YXL	HSH	CHY	WYH	WKS	
		COMM	BENG		CHILC	CHI	ENG	MATHS	TYP		ACCT	SE	COMP	MATHS	COMM	ACCT	
		F5A	COMA		F6B	F4D	F4F	F5E	F4C		COMB	CBSII	F5C	F4A	F5D	BS1B	
	T9 -- T10	LSH	NKL	CLS	CKP	HSH	WYH					WKS	CHY	SWS	WFS		CYF
		ENG	ACCT	ACCT	ECON	COMP	COMM					ACCT	COMP	ENG	ACCT		TYP
		F5A	COMA	F6B	F6B	F4D	F5E					CBSII	BS1B	F4A	F4C		COMB
	T10 -- T11	LSH	NKL	CLS	CKP	HSH	WYH					WKS	CHY	SWS	WFS		CYF
		ENG	ACCT	ACCT	ECON	COMP	COMM					ACCT	COMP	ENG	ACCT		TYP
		F5A	COMA	F6B	F6B	F4D	F5C					CBSII	BS1B	F4A	F4C		COMB
							F5E										

Table 1, A Partial Master Timetable

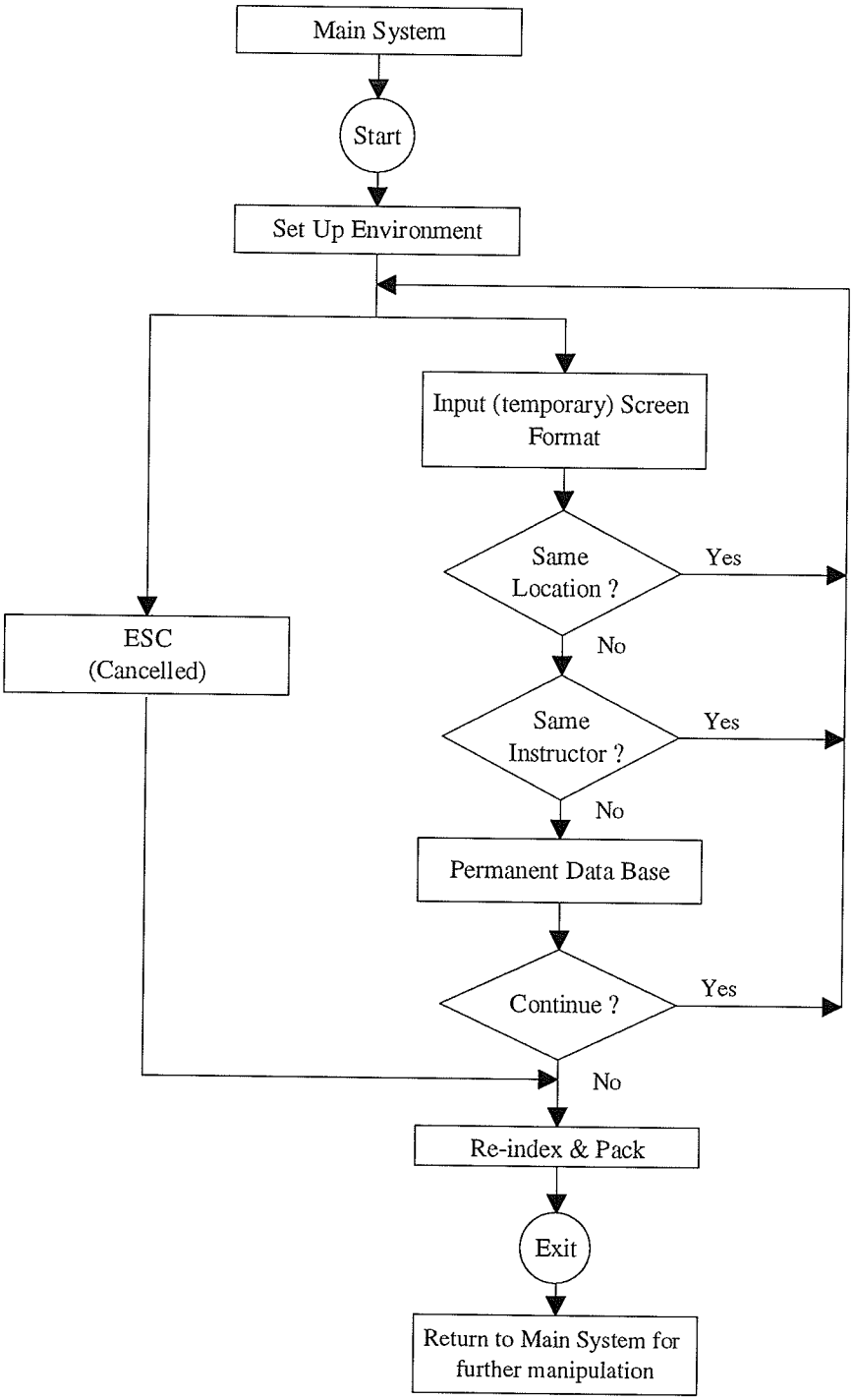


Figure 4, Data Input with Conflict Restriction Check

3.1.1 The Limitations

In reality, there are few institutions whose timetabling formulation will follow Johnson's framework without any alteration. An institution may remain static in its academic structure, but when there arises some administrative factors, they often generate a number of related conflicts into the problem. One of the common phenomenon is to allow class and subject rotations among teaching staffs. The scheduling complexity will be relatively increased if there are fewer existing free periods in the instructor schedules. The resulting timetable would likely bear no resemblance to the preceding timetable. Johnson's model is built on the constructing stage of a timetabling process which requires a sophisticated data support and printing device. Owing to its absence of any assignment mechanism, his design may not represent the mimic model of a computerized timetabling system. Thus a timetable planner may not substantially achieve any benefit from the system, as Johnson's design requires all the assignment be provided in the input. It essentially becomes a printing device. On the other hand, the ease of the system relieves the clerical burden associated with the production of timetables and other report forms. The method proposed in the next section does provide an assignment mechanism in the processor and hence can handle most variations of a given problem at hand.

Chapter 4

A CONCEPTUAL DESIGN

This chapter proposes a computer based support system for constructing school timetables. In the first section, we discuss the general requirements for the design of a support system. In addition, conditions such as multiple periods (sessions), settings and preassignment of classes will be examined before they are implanted into the design. The next section follows by introducing the general picture of the timetabling system, for implementation in school. The system captures the computational benefits from a database management system whose nature is essentially empirical as described in Johnson [1993]. It focuses on the formation of a feasible rather than the optimal way to manage the timetabling problem. In addition, a new assignment heuristic incorporating the dynamics of rotating session (period) scheme is introduced into the prototype. It will allocate the class(subject)-instructor pairs into the teaching locations in various time slots.

4.1 Problem Description

The typical school timetabling problem in this study can be outlined as follows: Let there be T curricular school cycles in an academic year, each cycle breaks into exact d days, and s be the total number of sessions (periods) within a day, all periods are standardized in length. The school is considered to have a collection of L locations, each representing a classroom or a specially equipped area;

N classes, each belongs to one of the year groups; M Instructors, with $M \geq N$; C Subjects (Courses), each has different year levels. Normally, all students within the same class are taking the exact curriculum of subjects with required frequencies in a cycle. Each subject provided to a particular class is conducted by the same instructor for the current year at the specific location. In any given curricular cycle, the total frequencies for all subjects offered to each class should not exceed the limit of $d*s$.

Let us define a class(subject)-instructor combination be a collective term specifying which classes, subjects, instructors have to be combined for a session at a particular location. The task of a timetabling system is to fill up the location and time frame with the required combinations under certain restrictions. In its simplest form, a feasible school timetable will be obtained when all the required combinations have been successfully assigned into the design, provided that there is no violation of the following critical conflicts:

(1) No instructor or class may be in two locations at a time

(2) Two sets of different combinations may not be assigned to the same location simultaneously.

(3) All classes are scheduled

Concerning the circumstance in our school problem, all the class(subject)-instructor combinations are assumed to be completed in the Thinking stage during the timetable planning. Realistically, it is the customary procedure to prepare and revise all the combinations manually at the top management level, where a list of subjective factors, e.g., class or subject rotation, teaching skill and performance, working pressure, class result in the public examination, etc., can be additionally

included into the consideration. The relevant data is reserved and will be validated during the assignment procedure.

4.1.1 Restrictions and Other Assumptions

In practice, school timetabling problems will involve a number of additional constraints. Unfortunately, there is not any clear cut distinction to practically identify those from the university timetabling problems. Lawrie and Veitch [1975] outline a brief description to both academic systems which may suggest some basics of the problem formulation.

Most universities usually adopt a master timetabling system. The scheme provides a hierarchical structure that decentralizes the timetabling problems to the levels that are largely independent of one another. Service classes provided by one faculty to others, are usually referred as the preassignments. They will be timetabled first so that the residual can be dealt with independently. Undergraduate instructors are generally organized at the level of the faculty, while tutorials and postgraduate teaching are established at departmental level. These structural changes are inconspicuous from year to year because of their continuing features. The previous year's timetable is carried forward and frequently regarded as the starting point of a new one.

The school timetabling system inherits a similar timetabling structure to that of a university, but trims down into a small scale with a single level. Students in most schools are divided into groups governed by their years attended and curricular mandates. Along with the critical constraints mentioned in previous section, there

exist several practical school timetabling characteristics that may structurally diversify from those in university for any given curricular cycle.

(4) Unlike universities, the size of a classroom is mostly of minor interest in school, of which most of them are standardized and fully utilized.

(5) Students and instructors in school spend more time in classes than their counterparts in universities. In most of the Asian schools, at least half of the instructor working time should be used to provide his or her teaching in class; the rest of the time is used as the off-class duty within the school area. The requirements for the distribution of free hours (periods) in schools over a curricular cycle for either students or instructors are more restrictive.

(6) Student timetables are commonly compact (or as compact as possible) with consecutive sessions, such as double or triple sessions not straddled by any break. In many situations, all classes are the multiples of the basic session.

(7) Some of the classes will be held in specially equipped rooms. Typically there are laboratory subjects, such as Typing, Computer Usage, Domestics Science, and Metal Work, as well as Physical Education and some others. These subsets of class(subject)-instructor combinations must be scheduled into certain subsets of rooms.

Apart from the general features mentioned above, there is another set of restrictions that a timetabler will frequently come across during the problem formation. Some of them may require preassigned scheduling, while others may refer as non-critical problems. They are flexible, nevertheless parts of their attributes factually inhibit the practical implementation of mathematical models into the realistic school system.

(8) Regarding the Conventional Modular schedule structure, subjects in the same classes are frequently located in cluster (for example, see Table 2), as if all mathematics classes may be scheduled early in the morning and science classes be scheduled late at the end of the school day. Experience in these matters has shown that the most significant source instructor unhappiness with teaching schedules arises from the way, classes are allocated to the sessions.

(9) Any combination should not be assigned to the same day where the same subject has already been scheduled to the same class at consecutive sessions. However, it is an acceptable procedure to allow the same combination of single block to repeat itself once in the morning, and once in the afternoon without merging them into a consecutive class block. This phenomenon mostly appears in the language class.

(10) Teaching workload for staff should be distributed evenly over the curricular cycle.

(11) Schools have curricula of a sort that make similar demands for the collateral availability of many resources. Option schemes may require teams of instructors and combinations of other resources. The general examples will be the intraclass splitting, or the interclass merging. In the higher forms (levels), some students will be subdivided into subgroups for tutorials or combined for common lectures.

(12) Some instructors and some rooms may not be available for certain sessions because of administrative or other duties

The above mentioned criteria will be elaborated in this study. Constraints (1), (2) and (3), related to critical conflicts together with some structural restrictions (6),

(7), (8), (9), and (10), are built directly into the algorithm. They cannot be modified during the execution of the program. Restrictions (11) and (12) may require special pre-arrangement or adjustment that is sometimes known as the preassignment. The interpretation of these constraints into sets of preassignment will be done by the person who is setting up the timetable. Dummy class, instructor or location variable may be introduced into the database where necessary, to solve the interclass merging or intraclass splitting problem. The information will be located to a separate preassignement data file and operated at a higher priority. Moreover, restrictions (4) and (5) can be classified as the structural environment in any given school system, for which they can be logically removed from the procedure without changing the algorithm of the proposed program.

4.2 A Proposed Computer Timetabling System

This proposed timetabling system is composed of three parts, namely the input, output and an assignment processor. In this section, we will provide the general description involving the above basic features. Detailed data structure, specifications of the program modules and the flow charts will be presented in Chapter five.

4.2.1 The Input

The main inputs required in the program are lists of Class(subject)-Instructor combinations and resources which are taken from three sources. The first source is a set of previously prepared database files which contain relatively stable information,

such as instructor name, class name, specially equipped room and subject description. Related files will be merged temporarily; information can be pulled out during presentation. The second source is the a set of information that requests preassignment. They have the higher priority to be scheduled into the timetable. However, their availability should be carefully examined before entering into the permanent database, to ensure no conflict has occurred. The third source is a database containing the lists of combinations which demands normal assignment. It includes all the necessary information such as class codes, subject codes, instructor initials, room code, class types (e.g., triple, double or single session), and the needed frequencies for a specific combination in a curricular cycle. A priority level code is also required which permits some specific combinations to be scheduled first. Within the same priority level, they will be arranged in descending order according to the number of consecutive sessions they acquired. The file is indexed according to priority level and subject codes. All the necessary data entries can be prepared externally from any dBase or its compatible system before they are put into practice.

4.2.2 The Output

The output is the information which gives three different kinds of finished timetables for classes, instructors, and locations. Those results can be directly accessed to the default printer or be saved to an output file in ASCII characters for further manipulation prior to the final printing. Individual timetable can be directly selected from the menu. Informations are independently stored in two data files, of which the first one is a temporary file consisting of all the assigned data for the

timetables. Data are indexed accordingly. When the program is exited, it permits further updates to the current information when required without having to redo the entire timetable from scratch. The second file records all the unassigned combinations.

4.2.3 The Assignment Processor

The algorithm that is finally developed, consists of two components. One part will primarily operate the detailed timetabling process based on the heuristic rule. It provides the ability to assign the preset class(subject)-instructor combinations reserved from the Thinking stage into the timetable where prior knowledge of the location and time slots are not required. The resulting information will be stored in a database for further operation. While the other part enables interactive editing procedure that helps the timetabler to modify the database that is created earlier from the automatic procedure.

4.2.3.1 Interactive Computer Component

There may be some special requests which cannot directly be built into the model. The main characteristic of this component is designed to work interactively with the timetabler, which takes into account the subjective judgment of timetabler in manipulating the data. It provides the ability to interchange the addresses of two combinations, or to move a combination from one assigned address to an empty one. This is a real time on-line system that serves only to help the timetabler to make

special adjustment, to re-allocate the unassigned combination, after the assigning procedure has taken place.

4.2.3.2 Automated Computer Component

The heuristic procedure designed in the automated computer component operates on a 'One-pass' system. During the run, once a class(subject)-instructor combination has been assigned a position in the schedule, it never considers that combination again. The system will directly access the information from the database in a manner that depends upon its preset priority level. According to a predefined sequence, it will attempt to locate each combination to the first available class block in the timetable. Unallocated combination will be recorded for further adjustment. The assignment sequence that we use to schedule each consecutive combination into a timetable will be discussed in the next section.

4.2.3.2a Row and Column Rotation

Despite a wide range of timetables existing in the education system, the basic conventional modular design remains popular in Asian timetabling system. In the majority of school timetabling problems, many timetablers make use of the design because of its simplicity and indisputable character. A conventional modular timetable is a rectangular D_i by S_j design, which is used to represent a curricular cycle of D_i days, i=1, 2, ..., d; each day of exact S_j sessions (periods), j=1, 2, ..., s; each having the same duration. There are totals of d*s elements in any given timetable. The sessions Sj are often regrouped to form B_b, b=1, 2, ..., k, blocks on

each day. The $B_1 \times B_2 \ldots \times B_k$ structure indicates that there is break or recess after every $B_1, B_2, \ldots, B_{k-1}$ block.

Different subjects may be sequentially scheduled across the timetable (for example, see Table 2, Dempsey [1983]) which however allow same subject to repeat its pattern on the same session every day. The format will provide certain manipulative benefit to the timetabler.

Day (D_i)		1	2	3	4	5
Session	1	(1) ALG 1	(1) ALG 1	(1) ALG 1	(1) ALG 1	(1) ALG 1
(S_j)	2	(2) ALG 1	(2) ALG 1	(2) ALG 1	(2) ALG 1	(2) ALG 1
	3	(3) ENG 1	(3) ENG 1	(3) ENG 1	(3) ENG 1	(3) ENG 1
	:	: :	: :	: :	: :	: :
	:	: :	: :	: :	: :	: :
	8	(8) SCI	(8) S LAB	(8) SCI	(8) SCI	(8) S LAB
	9	(9) SCI	(9) S LAB	(9) SCI	(9) SCI	(9) S LAB

Number inside ( ) denotes the sequence to enter a subject in any given day

Table 2, A Partial Conventional Modular Schedule; 5-Day Cycle,

In the earlier discussion, this kind of timetable is also one of the significant source that incurs instructor unhappiness. As the instructor prefers not to have his or her class appeared to the same session in each day. The following rotation algorithm will rotate diagonally, each class to next day and next session in the curricular cycle.

Let Table(Day, Session) = $T(D_i, S_j)$ be the address or position of ith day and jth session on any given table. Note that i, denotes the column and j, the row.

Suppose all the classes have been numbered to form a sequence of 1, 2, ..., d*s. For the nth subject in this sequence, denotes Order(n) = O(n), the position of class n to be assigned in the table, where n = 1, 2, ..., d*s. For this nth element, the D_i and S_j addresses are defined as $D_i \equiv n \pmod{d}$, with $D_i = 0, 1, ..., d-1$, for all s sessions and is followed by $S_j \equiv n \pmod{s}$, with $S_j = 0, 1, ..., s-1$ for all d days. The algorithm will change $D_i \rightarrow D_i + 1$ and $S_j \rightarrow S_j + 1$ to take care of the zero value. If d and s are relative prime and represent the total days and sessions in a timetable, each nth element will then have a unique D_i and S_j address correspondingly. The next n+1th element will be rotated to the new position of $D_i + 1$ and $S_j + 1$, provided that $D_i + 1 < d$, and $S_j + 1 < s$. This method will prevent any subsequent element to be allocated to the position of the same session in the next day or the next session in the same day. Figure 5 indicates the basic rotating structure to all class, instructor and location timetables.

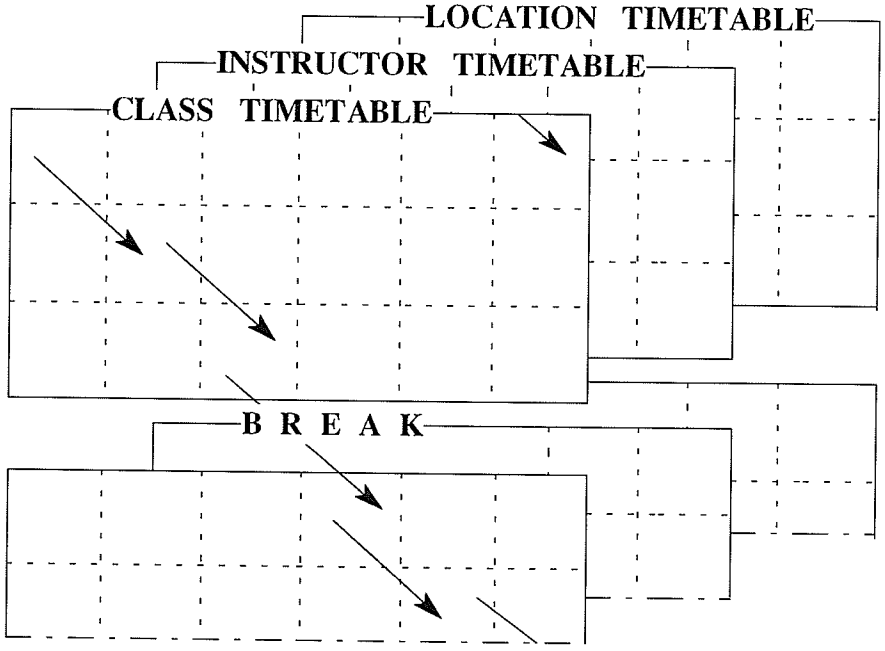


Figure 5, The Row and Column Rotation

The algorithm developed on the basis of the above problem formulation can be described by the following program. It will be mapped into the corresponding address of a given timetable accordingly.

Initialize D_i, S_j =0,

For n = 1 to d*s (the maximum entry of each timetable)

Case 1, d ≠ s, d< s, the least common multiple, LCM(D_i, S_j) = d*s (unique solution)

(1) $D_i = D_i + 1$ (Shift the day entry by 1 row or column)

(2) If $D_i > d$, set $D_i = 1$ (Reset the day entry to day 1)

(3) $S_j = S_j + 1$ (Shift the session entry by 1 column or row)

(4) If $S_j > s$, set $S_j = 1$ (Reset the session entry to session 1)

Any Order(n) can be mapped to Table(Day, Session), i.e. $O(n) \rightarrow T(D_i, S_j)$ in a unique way.

Case 2, d ≤ s, the least common multiple, LCM (D_i, S_j) = w_lcm < d*s (addresses repeat at the multiple of w_lcm, since d and s are not relatively prime.)

If n ≤ w_lcm, repeat step 1 to 4, otherwise

If n > w_lcm and n (mod w_lcm) ≡ 1

(5) $D_i = D_i + 1$ (Shift the day entry by additional 1 row or column)

Repeat step 1 to 4 otherwise.

If d > s, interchange their position and perform the above operation.

The algorithm can be extended to any problem with the similar structure. A typical example which indicates the sequence (order) to fill up a 5 day cycle of 8 sessions each, with a 3x2x3 block structure, is illustrated in the Table 3 .

Day (D_i)		1	2	3	4	5
Session	1	1	17	33	9	25
(S_j)	2	26	2	18	34	10
	3	11	27	3	19	35
		Break				
Session	4	36	12	28	4	20
	5	21	37	13	29	5
		Break				
Session	6	6	22	38	14	30
	7	31	7	23	39	15
	8	16	32	8	24	40

Table 3, A Rotating Modular Schedule; 5-Day Cycle, 3 x 2 x 3 block structure.

[Note 1: Number inside each cell indicates the sequence to enter a subject in any given day, e.g., the 22nd entry of a particular subject is placed in 22 (mod 5) ≡ 2, and 22 (mod 8) ≡ 6, i.e., the 2nd day and 6th session, thus Order(22) = T(2,6) for both entries in the timetables of instructor and class.

Note 2 : Entries 14, 39 and 24 make a consecutive triple class block]

4.2.3.2b Consecutive Class Assignment

The sequence to arrange the consecutive classes into the individual timetable are analogous to the algorithm generated in the previous section. A consecutive class

block in this study can be described as the class with s sessions without splitting by any break, where T and T_L be the Triple Class, D and D_L be the Double Class and S and S_L be the Single Class. The subscript L denotes the class will be conducted in a specially equipped location. They will be arranged into the timetable in a manner that depends upon this preference. Within each type of class block, the value of the first class in each block defines the priority sequence when it is being introduced into the timetable.

During the setting, all the possible consecutive class blocks will be automatically produced according to the predefined break provided by the timetabler. They will be recorded separately into three databases for automated assignment procedure. Those data are relatively stable and will not be changed whilst the setting remaining the same. However, the defaulted patterns, if necessary, can be altered or deleted by timetablers externally when they come across any special administrative request. This may account for some sessions that cannot be assigned to any class, like school assembly or staff meeting at a fixed session in every curricular cycle.

We consider T_n, D_n, S_n and T_M, D_M, S_M be the expected numbers of triple, double and single sessions assigned to a class, and maximum class blocks generated from a table respectively. A feasible timetable may exist when the number of assigned blocks, $T_n \leq T_M$, $D_n \leq D_M$, $S_n \leq S_M$ and $T_n + D_n + S_n \leq d*s$, i.e., the total of the expected sessions should not exceed the maximum value as mentioned in section 4.1. A specific example based on section 4.2.3.2a indicates all the possible combinations for each type of consecutive class block and the entry sequences are shown in Table 4. Results in session 4.2.3.2a, and 4.2.3.2.b will be built and stored in the relevant database during the initial setting, or through the setup/change setting procedure from the main menu.

Priority/Sequence	entry	Address 1	Address 2	Address 3	Day	Start slot
Triple Session (T)	1	1	26	11	1	1
(1st Priority)	2	6	31	16	1	6
	3	9	34	19	4	1
	4	14	39	24	4	6
Total of 10	:	:	:	:	:	:
available blocks	10	38	23	8	3	6
Double Session (D)	1	1	26	-	1	1
(2nd Priority)	2	2	27	-	2	2
	3	4	21	-	4	4
	4	6	31	-	1	6
	:	:	:	-	:	:
Total of 25	24	38	23	-	3	6
available slots	25	39	24	-	4	7
Single Session (S)	1	1	-	-	1	1
	2	2	-	-	2	2
Total of 40	:	:	-	-	:	:
available blocks	40	40	-	-	5	8

Table 4, All Possible Class Blocks in Their Attempt Sequences in a Rotating Modular Schedule; 5-Day Cycle, 3x2x3 block structures, (see also Table 3 in 4.2.3.2a).

Chapter 5

DATA STRUCTURES AND THE PROGRAM

In this Chapter, we focus on ways to present full descriptions of the timetabling system mentioned in previous section. The first section will describe the data structures. In the next section, detailed specifications of the programs and their flow charts will be followed. The main system is designed in a modular architecture containing individual modules. Each modular program is written in dBase which is sometimes mentioned as an interpreted programming language with a command line mode. The resulting source-code file with the extension (PRG) consists of a set of program instruction commands and keywords. It can be created and edited with a simple text editor or word processor. Before the program can be executed in a computer, it is an inevitable procedure to convert the program in machine-language code. In order to improve its portability and efficiency, the entire program is compiled through a Clipper compiler. The compiled program can be distinguished by the file extension (EXE) and can be implemented on any IBM or compatible computer.

5.1 Data Structures

We are given a set of class (subject)-instructor combinations, a set of location and a set of time slots. These data files are prepared in dBase system and separately

stored in different database tables that can be recognized by the file extension (DBF). In database terminology, the columns are called fields and the rows are called records. The first type of databases is automatically generated from the setting procedure to provide the default assignment pattern for the heuristic procedure. This information will not change unless the institution changes its timetable structure. Their data structures will be described by the following database files.

(1) All the predefined time slots generated from Row and Column Rotation algorithm mentioned in section 4.2.3.2a are located in (Daysespf.dbf), which include the sequence of each slot and its corresponding day and session. The Slot field is being used as the index key.

- Slot -- The actual slot number with its ranges from 1 to d*s;
- Day -- Corresponding curricular day number;
- Session -- Corresponding session in each curricular day.

(2) Another set of default database files is those generated from the Consecutive Class Arrangement in section 4.2.3.2b. The first one (Tdspf.dbf) contains fields of all the possible class type and its analogous slot number in each class block. The fields include:

- Type -- Class type code, whether it is a T for Triple, a D for Double or a S for Single class block;
- Slot1, Slot2, Slot3 -- The corresponding address (slot) of each class block in the timetable.

The second file (Clslotpf.dbf) consists of the information that described all the possible first slot numbers among each type in each class. The indexed key followed by their fields which are namely, Class code, Class type code and slot code.

For the next type of files, they contain data that will be transferred to the assignment processor for immediate manipulation.

(3) The main input database file (Clsubjpf.dbf) reserves all the necessary information from the Thinking stage to describe all the combinations that require normal assignments during a curricular cycle. Each record contains fields as below.

- Priority -- The priority level to assign the particular class and subject block in the timetable, it ranges from 1 to 99; with 1 being the highest priority and 99 being the lowest. Normally, the consecutive class blocks will be given a higher priority value;
- Class -- Individual class code;
- Subject -- Subject code;
- Tcher -- Instructor or teacher code;
- Room -- Teaching location code, usually refers to the assignment of special equipped location. If there is no entry, the algorithm will default the assignment to its own classroom, otherwise it will try the chosen location;
- Classtype -- Class type code, T being Triple class block, D, the Double class block and S, the Single class block;
- Repeat -- The repeat counter, a digit denoting the required number of frequency for each particular combination to be assigned to the timetable;
- Finished -- A blank field, the algorithm will return a value showing how many of the particular combinations have been successfully assigned in the timetable. The difference between the fields of Repeat and Finished indicates whether there is any unallocated combination.

The file is using the Priority and Subject as the index key field.

(4) The alternative location database file (Altrmpf.dbf) locates all the rooms and their alternative choices. Once the previously assigned location is not available, the algorithm will attempt to seek for the availability in its alternative. This situation appears very often in scheduling the laboratory, typing room, work shop or P.E. playing field. The Room field is the index key.

- Room -- Classroom or specially equipped location code;
- Altrm -- The alternative location code.

(5) There are two database files (Assignpf.dbf, Preasspf.dbf) possessing the identical field structures. Assignpf.dbf is a transient output database produced by the processor to accumulate all the assigned combinations. The file will be made empty before the automated procedure is started. The resulting information can be adjusted or rearranged in the manual assignment module. Preasspf.dbf is an input file containing all the data acquired from the preassignment process. Upon request, all the materials will be transferred to the Assignpf.dbf before the automated assignment procedure starts. The data are ordinarily preset and conflict free. Their corresponding fields are Class, Slot, Tcher, Room, Subject, Day, Session and Type. The last field is denoted as Type1, which designates as the ranking sequence within a class block, for example, whether it is the first, second or third values of a triple session.

The last sets of databases are those subsidiary files holding all the structural information of the school, of which most of the data are relatively stable. Each file is composed of the relevant codes and their corresponding descriptions. Information will be temporary merged together through the correlated code fields while they are being pulled out for presentation.

(6) Roompf.dbf, Classpf.dbf, Subjpf.dbf and Tcherpf.dbf are all descriptive database files where their structures are very much alike. Each file includes the record of the abbreviation used in the output format and the detailed description for each code. The Roompf.dbf file contains the codes for classrooms and other specially equipped locations; the Classpf.dbf file contains the information of the class codes; the Subjpf.dbf contains the subject codes, while Tcherpf.dbf contains all the codes for the instructors.

5.2 The Program

The main design is divided into a master control program and five individual operational program modules, to improve the flexibility and extendibility. Each of these may be directly accessed by different users, like timetable planners and office workers who may take part in one or more phases in the timetabling construction. In addition, new modules can be added into the system without much alteration to the whole structure. Each module is written and tested separately, allowing easier and more efficient maintenance of the design.

5.2.1 The Menu

The master program (TTMAIN.PRG) performs the function as an interface for linking all the separate program modules of the timetabling system. It will bring forward an on-screen menu to a user. Its ease of usage allows the user to choose the operating module directly from the screen. The modules are namely, Auto

Assignment, Inquiry / Manual Adjustment, Output to Printer or File, Set Up / Change Setting and Update File, Figure 6 shows the structure of the master control program.

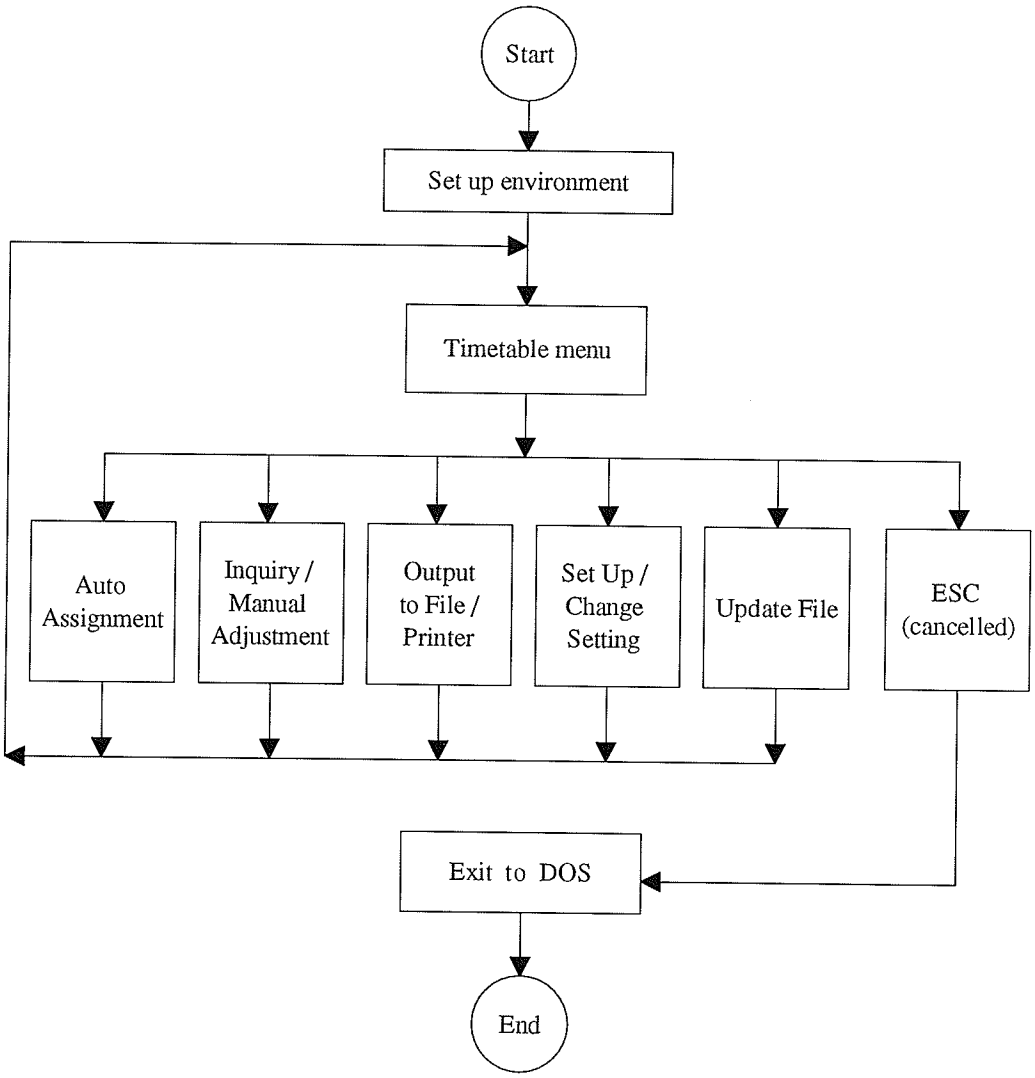


Figure 6, Structure of the Proposed Timetabling System

5.2.2 Auto Assignment Module

The first module (TTASSIGN.PRG) characterizes the core of the automated assignment mechanism. Before the process commences, it will duplicate all the preassignment records directly into the output database Assignpf.dbf upon request; data will be reindexed to ensure new pointers are created to the records in the file. The system will begin to locate the record from the input data file Clsubjpf.dbf that requires normal assignment according to the priority level and the class type. Default assignment patterns reserved in Daysespf.dbf, Tdspf.dbf and Clslotpf.dbf will be preloaded into system as the address indicator.

The algorithm canvasses the output database, to scrutinize the time availability for the same addresses in each specific class, location, and instructor beginning from the highest priority. The selected combination will therewith be assigned to the first available position where it does not conflict with any other time slots already assigned. In addition, the combination will not be assigned to the day scheduling the same subject to the same class with consecutive sessions. It is an option to allow split block of the same subject appear daily in a timetable. Furthermore, any combination will not be assigned to an instructor when one's maximum daily teaching hours are reached. The algorithm will repeat the assignment procedure for each particular combination to a timetable according to its preset counter. For every successful entry, the counter will be reduced; otherwise if every attempt is failed, the unallocated combination will be indicated during inquiry. All the combinations in the list are handled in this straight forward manner. Figure 7a and 7b indicate the flow chart for the Auto Assignment Module.

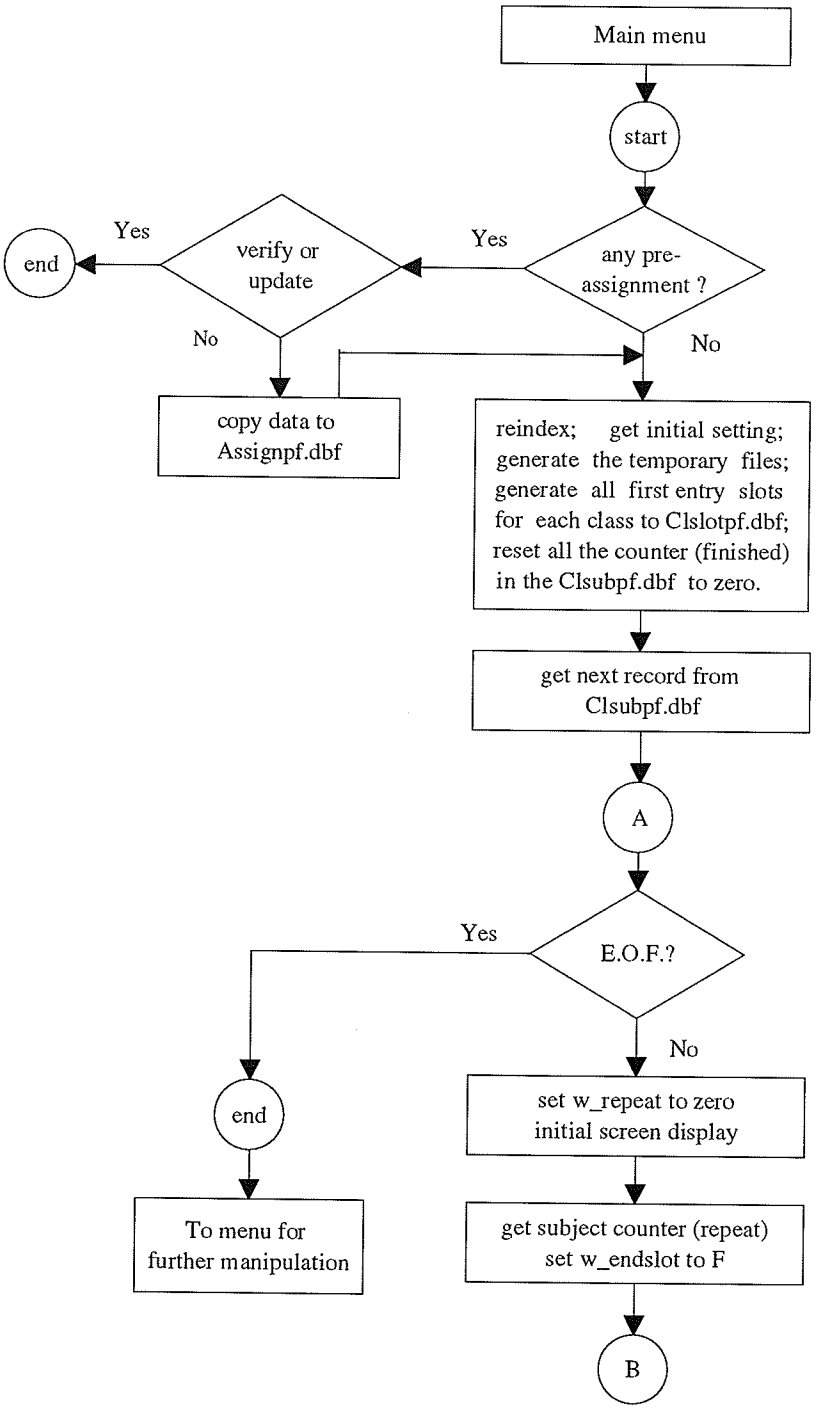


Figure 7a, Flow Chart for Auto Assignment Module

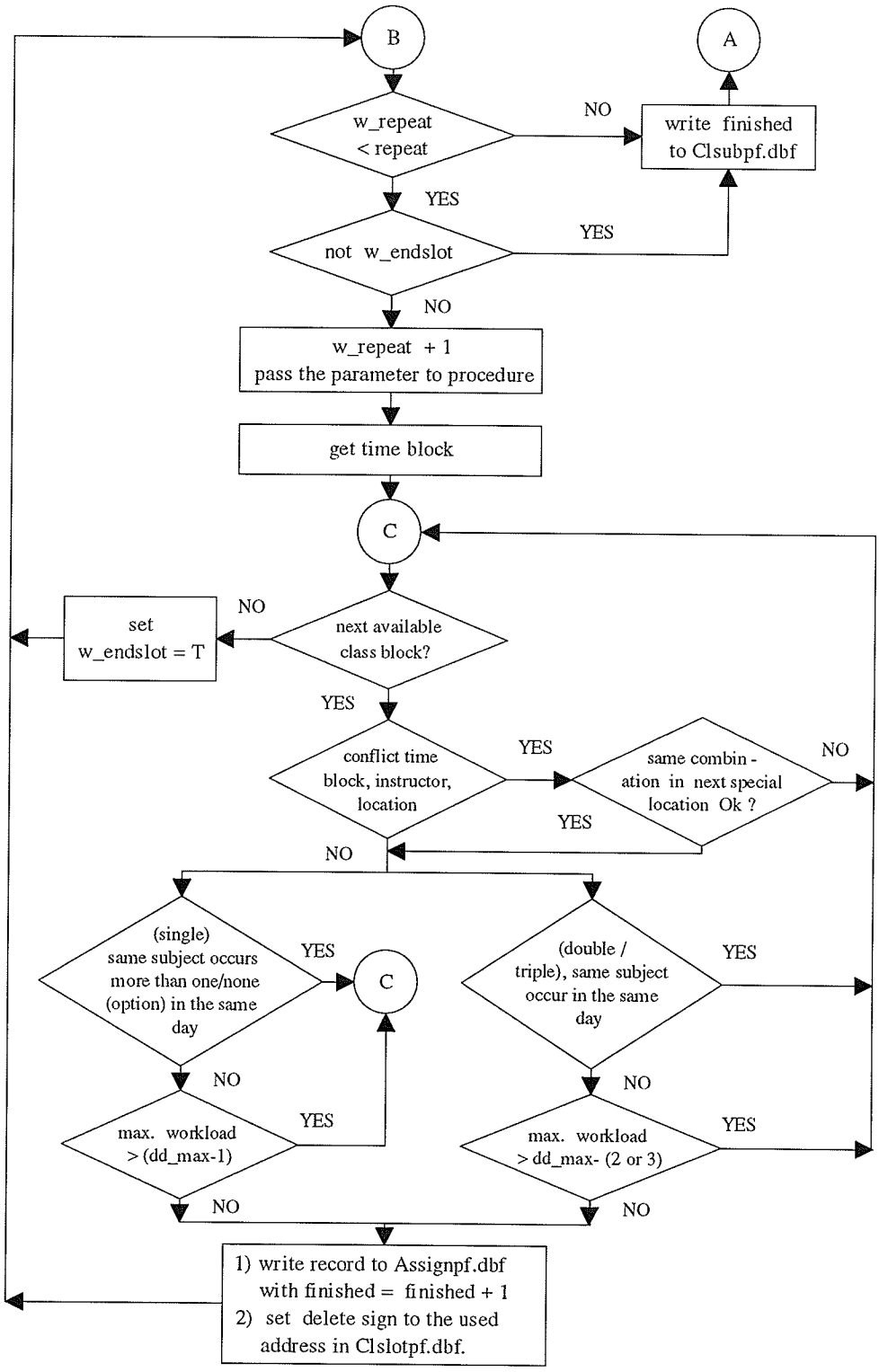


Figure 7b, Flow Chart for Auto Assignment Module

5.2.3 Inquiry / Manual Adjustment Module

For the Inquiry and Manual Adjustment Module (TTINQ.PRG), it is divided into two related parts. The first part will serve as an on-screen display that allows the user to examine class, instructor or room timetable individually. Its flow chart is shown in Figure 8a. According to the user requisition, data will be temporarily extracted out in the sequence of session by day. For any non-consecutive sequence, it denotes that there is a free period or break in between. Appropriate spaces, blank lines or recess line will be filled in accordingly. In the class timetable mode, unallocated combination with its corresponding frequencies obtained from Clsubjpf.dbf will also be indicated.

The second part provides an on-screen manual adjustment facility that allows the user to work interactively with the individual timetable. However, all the modification must be made in the class timetable screen mode. Working files will be created to replace the actual database files. Alterations are made insides those temporary files. The program allows the user to interchange two class blocks simultaneously, or to relocate a combination to the new location. Consequently, available class blocks can be obtained which allow the user to reallocate the unassigned combination into the timetable. Whenever a successful entry has been made, the counter for that particular unassigned combination will be reduced. The assignments will appear on the screen, the moment they are made. Procedures will be used to validate their availability during the rearrangement. The temporary files will be deleted, saved or replaced the actual database files before the user return to the main menu. Flow charts 8b and 8c describe the Manual Adjustment module.

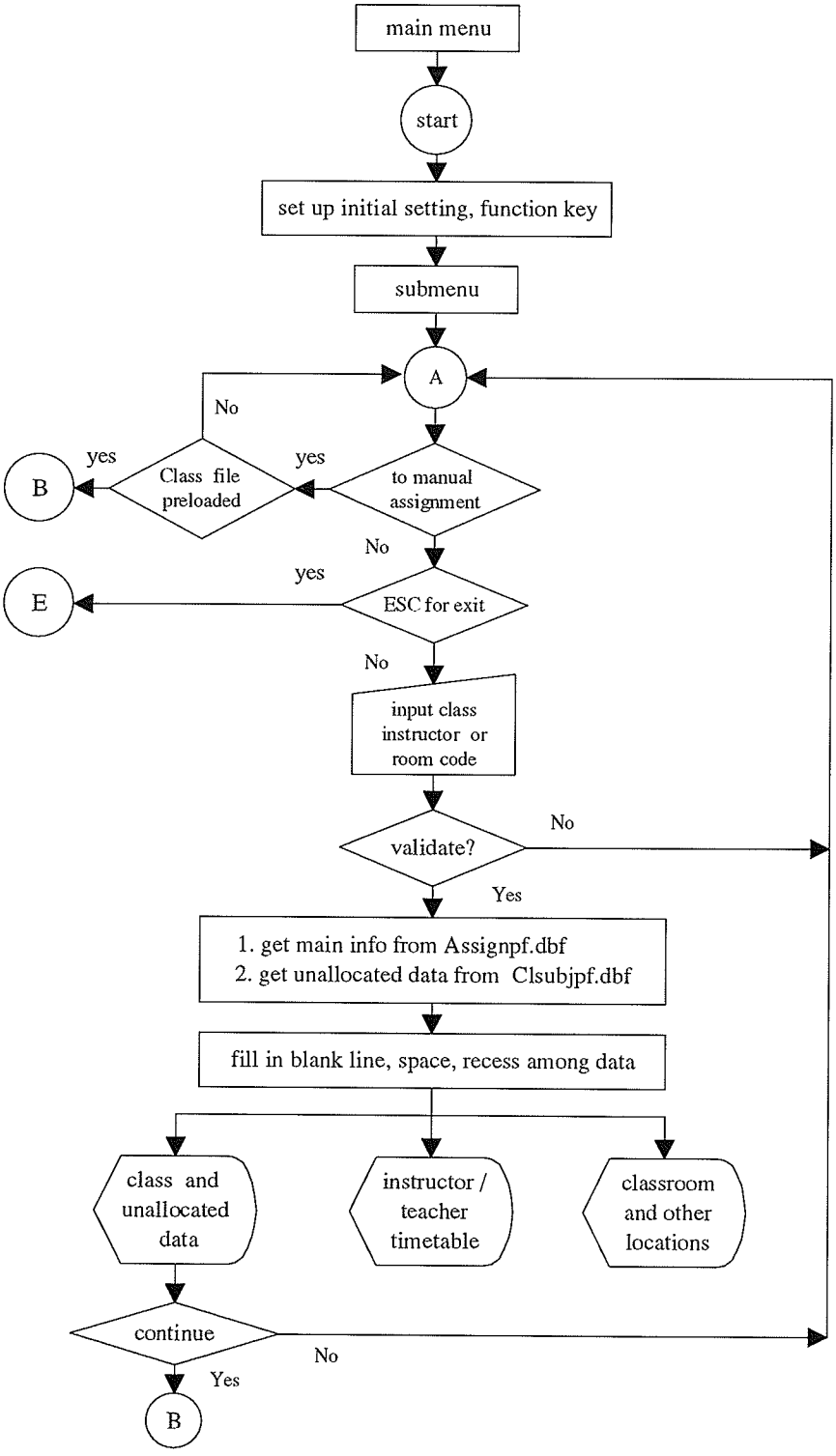


Figure 8a, Flow Chart for Inquiry / Manual Adjustment Module

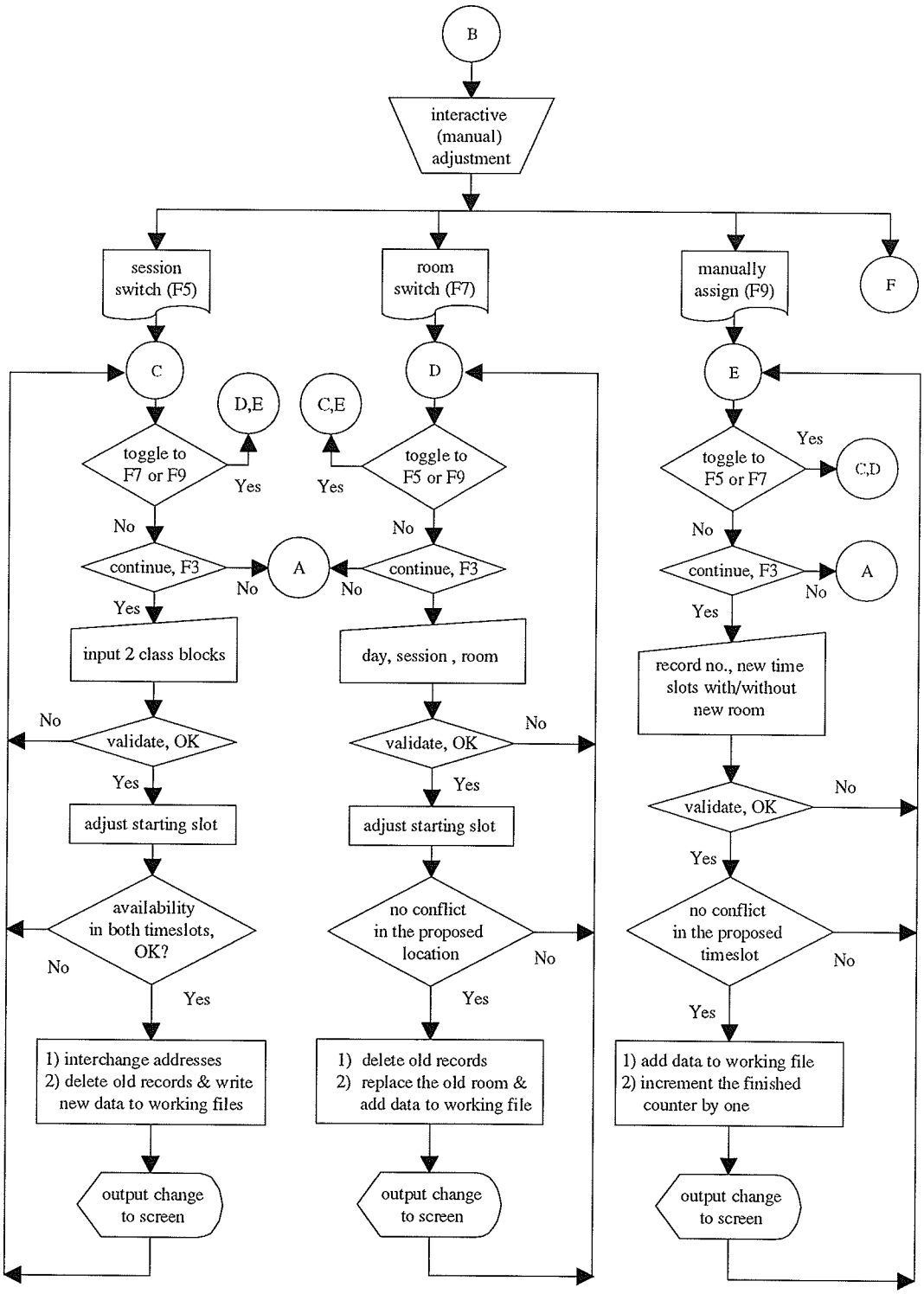


Figure 8b, Flow Chart for Inquiry / Manual Adjustment Module

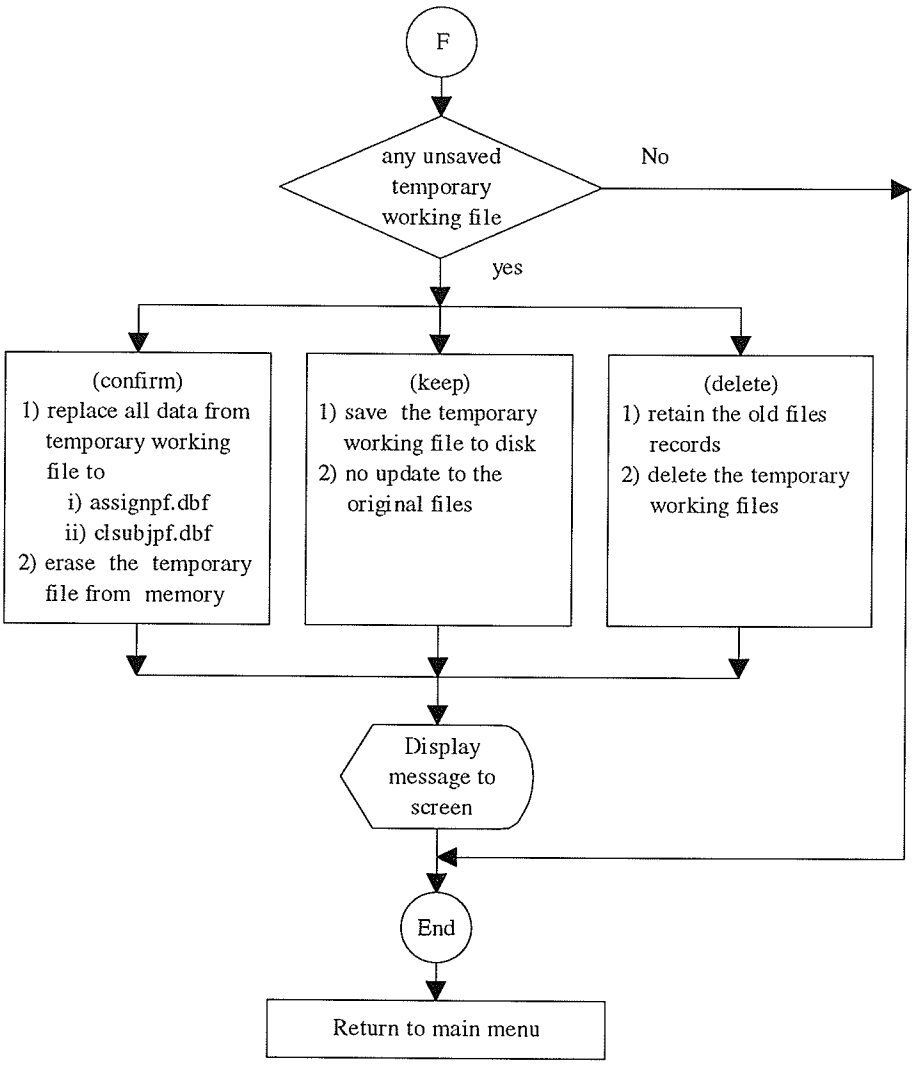


Figure 8c, Flow Chart for Inquiry / Manual Adjustment Module

5.2.4 Output to Printer / File Module

This is the output procedure controlled by program module (TTPRINT.PRG) where its flow chart is given in Figure 9. The algorithm will direct the output data to

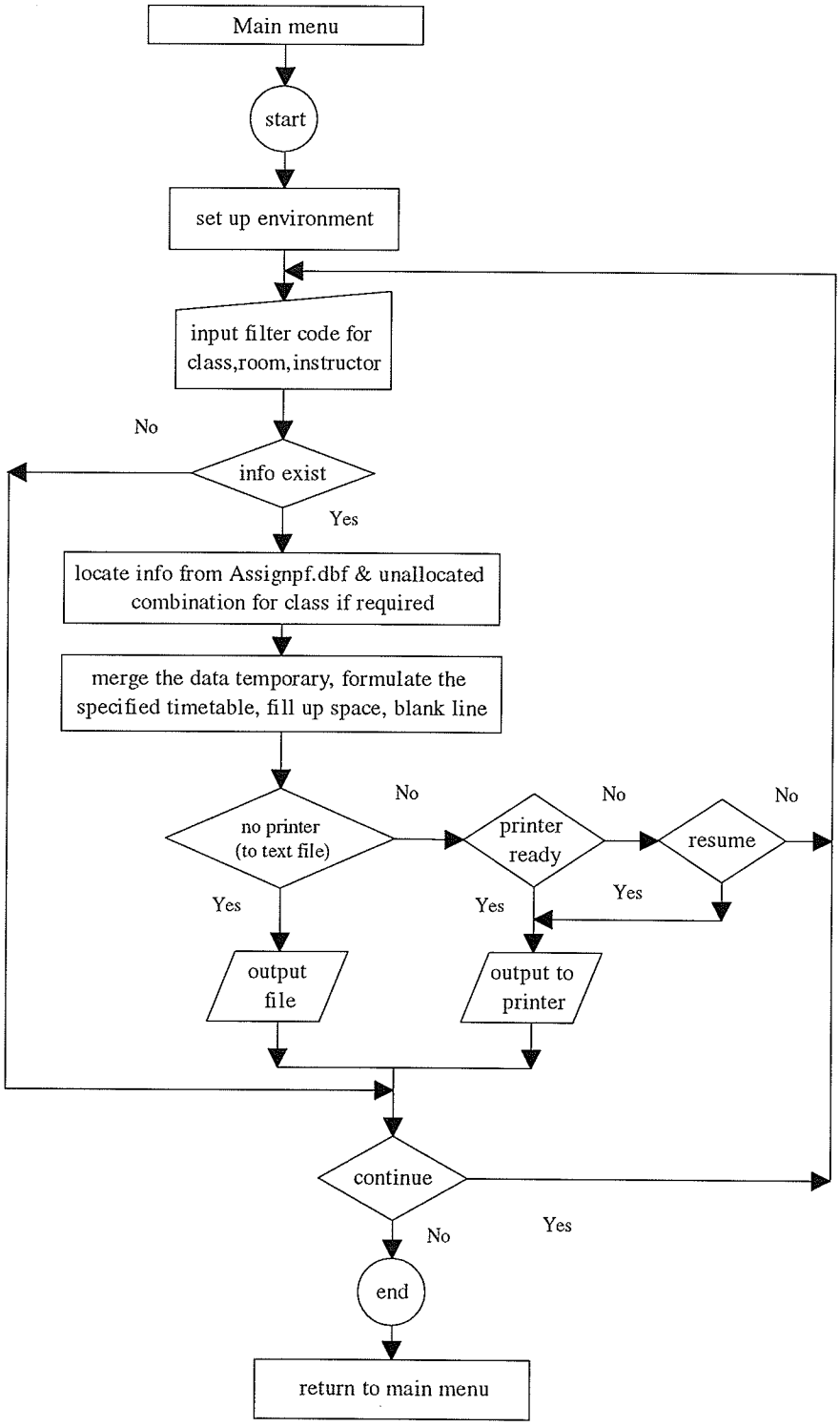


Figure 9, Flow Chart for the Output Module

the default device e.g., an output file, or a printer that is configured in the initial setting. User may choose to produce all or individual class, instructor and room timetables. For each class period, the class timetables show the subject name, instructor abbreviation and room. The instructor timetables show the subject abbreviation, class and room. Whereas the classroom timetables indicate subject, instructor abbreviation and the class. The procedure is similar to that in the inquiry, except these times are for detailed hard copy printouts.

5.2.5 Set Up / Change Setting Module

The module (TTSETTNG.PRG) is used for initialization of memory variables and to define the output destination. Basic school structure parameters include the number of school days in a curricular cycle, the number of sessions and breaks for a day, the daily break structure and the maximum daily teaching hours. The procedure enables an option to suppress the Split Block subroutine which allows same subject to repeat once in the same day. In addition, it also requires the information to access the output either to a text file or to a printer. Detailed flow chart is illustrated in Figure 10.

In simplicity, the model at present does not consider any unusual school structure within its scope. For most common school cycles in Hong Kong, they will be varied from 5 to 8 school days. Each day consists of identical number of sessions, usually in the range of 6 to 10. Moreover, it will be realistic to confine the number of school days in a curricular cycle and the sessions per day at a reasonable value. Both values are bounded between 2 and 15. In every school day, it allows a maximum of

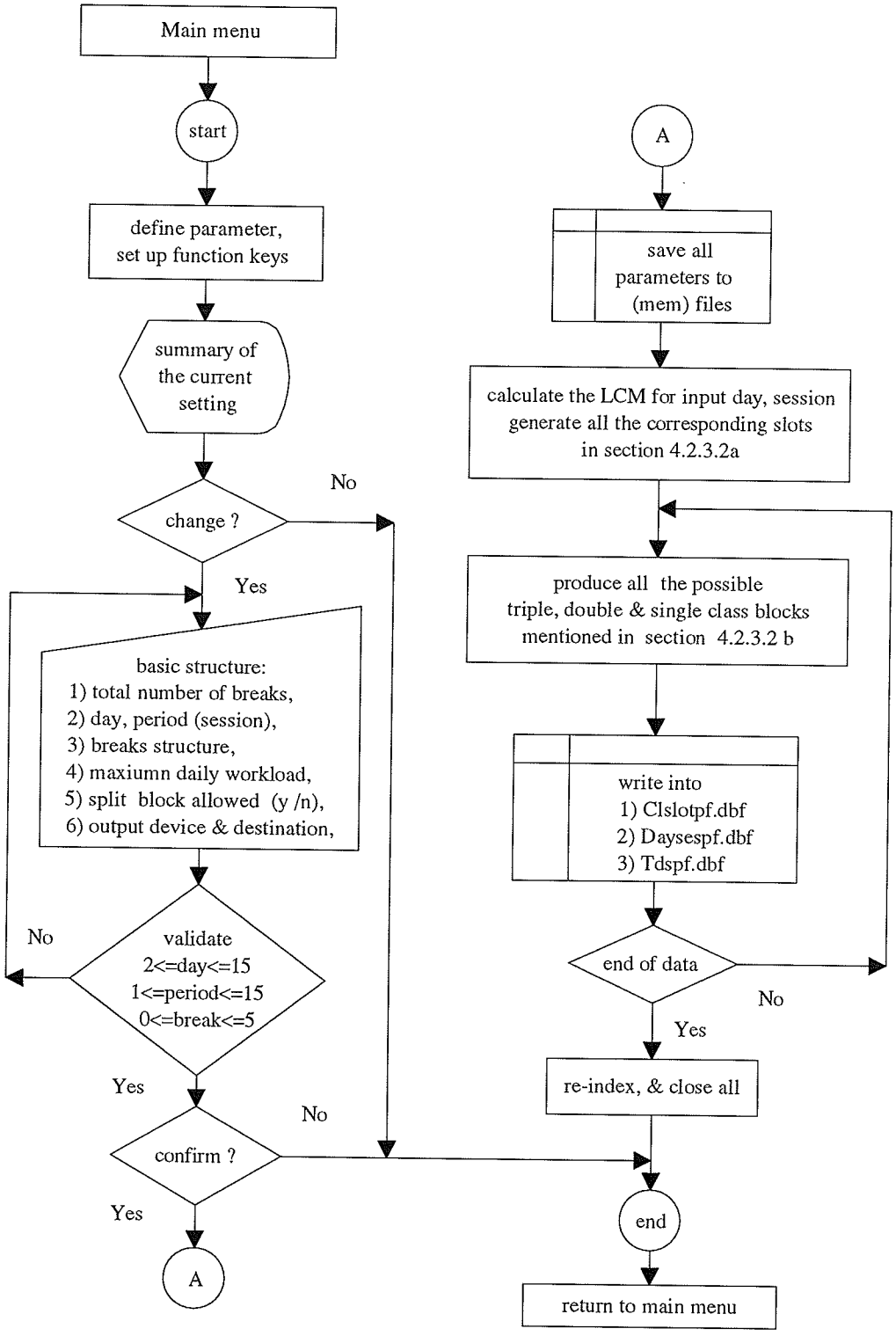


Figure 10, Flow Chart for Set Up / Change Setting Module

5 breaks in between. Before the input information can proceed to the next stage, it will be first validated and stored permanently in memory variable file with an extension (MEM). These parameters will be applied to produce the default data pattern in Daysespf.dbf, Tdspf.dbf and Clslotpf.dbf. This module should be redone each time if there is a structural change in the school, such as the substitution of a new class type, a change in the curricular cycle or in the break structure.

5.2.6 Update File Module

The last module (TTUPDATE.PRG) allows the user to update and to renew all the old slot values in Preassignment and Assignment files once the school structure is changed. Error message will be appeared on screen if there exists any incompatibility to the new setting. One of the good examples will be a consecutive class block straddled by a break. In addition, the module enables the user to pass the information from assignment to preassignment file, so that the timetabling process can be continued in stages. The flow chart in Figure 11 demonstrates the details of the module.

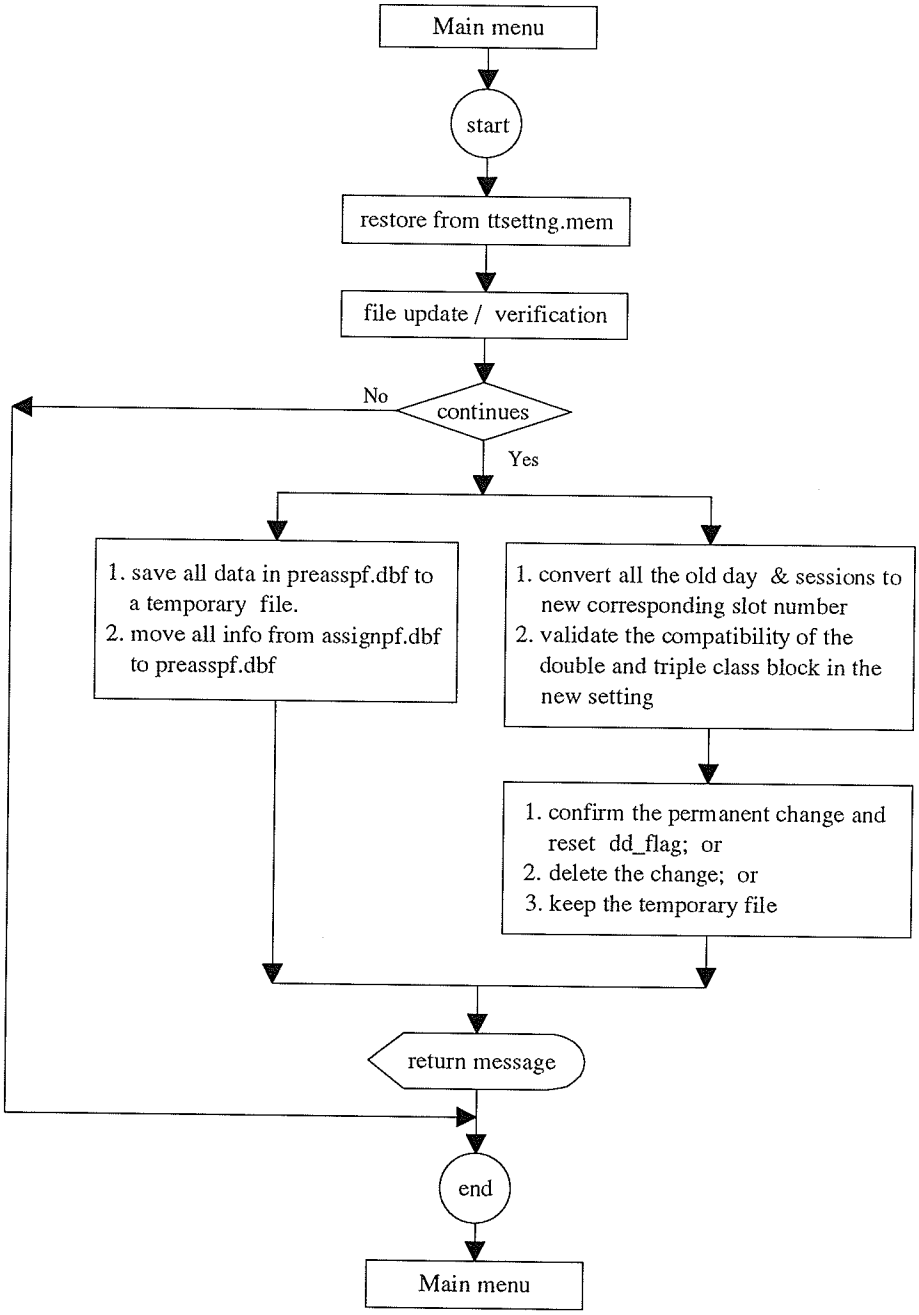


Figure 11, Flow Chart for Update File Module

Chapter 6

SYSTEM EXPERIMENTATION, CONCLUSIONS, AND RECOMMENDATIONS

In this chapter, we demonstrate the use of our prototype program for constructing the timetables in an adult educational training school in Hong Kong. Conclusions, limitations and recommendations related to our study are also presented.

6.1 Implementation and Computational Results

To illustrate the use of the proposed system, the manual timetabling system in the adult education training school mentioned in chapter three is further investigated. The objective of the school is to provide the opportunities for continuing education for those who need practical training either in their work or to extend what they learnt in school. In fact, it was recognized as one of the prominent organizations that provided continuing adult education in the early sixties.

Unlike other conventional institutions, this school offers regular grammar school modules and other vocational training courses. At present, the school has 23 classes which can be categorized into 7 individual modules. Each module will take a different duration of school hours which are in the range of 22 to 28 sessions per scheduling week. The weekly teaching hours for each full-time staff are confined in between 22 and 26, depending on what the subjects one teaches. The timetabling

process will involve about 900 students, 24 full-time and 3 part-time instructors. Each year, about 600 course-instructor pairs are required to schedule into 12 classrooms and 4 functional rooms. The present timetable frame is a 10-period daily schedule with 2-6-2 break structure, and 5 days per scheduling week. It will be revised if new module is introduced. Classes are divided into the morning and the afternoon sections with compact type timetables. However, the time to release a class in the morning section is not fixed. While for the afternoon section, the class may start the lecture as early as in the third period. A lunch break will be located randomly in between the fifth and seventh periods. Thus the school structure makes the present timetabling system more difficult to formulate. As we compare to the traditional timetabling structure, every class in the school will use up the entire timetable.

In order to solve the problem, we construct the timetable in two iterations. First, the timetable is configured as a 5-day by 6-period with 2-4 break structure. The first database containing 12 morning classes are put into the program for computation. The automatic part, that is first draft timetables, are found in about 4 minutes. After having some minor adjustments to the output data, the resulting information is saved to the preassignment file. We continue our next iteration by restoring the timetable structure to its full scale, defining the maximum teaching workload. Usually we keep the workload limit to not more than 6 or 7 teaching hours per day. The preassignment file will also be updated. In the second iteration, the system will first duplicate all the information from the preassignment file, and then schedule the remaining classes in the afternoon section to the unused classroom

blocks. The package has taken approximately 10 minutes to accomplish the task and is not considered excessive at this stage.

The program is constructed and run on an 80486-25SX personal computer without math co-processor. Preassignments are easily handled by placing the information into the preassignment file. In appendix A, we show a collection of the screen outputs generated by the timetabling system.

The automated part of the system has proved to be an easy method to produce the draft timetables under a reasonable amount of time. The experiment has indicated that there are some isolated lessons. These could however be assessed with simple permutations by the users, to get expected compact schedules for all classes. There appears a total of 6 unassigned course-instructor pairs; one of the pairs in the morning section and the rest are found in the afternoon section. These pairs can be reassigned interactively through the manual adjustment procedure to obtain the final schedules. In this case, the computer based support system developed here is considered valuable, which eliminates the repetitive manual checking process that the timetable planners need to do. The scheduling time is shortened and the solution is conflict-free. Although we do not identify the actual amount of time used in the manual adjustments, it is reasonably believe that the entire timetabling process can be accomplished within a working day. It used to take weeks for a group of non-experienced teaching staffs or administrative workers to obtain a less satisfying timetable. In addition, it also requires days for a secretary to produce the hard copies of all the class, instructor and room timetables. A collection of the course structures, input and output databases, and sample listings of timetables are illustrated in Appendices B to D.

6.2 Conclusions

The research work presented in this thesis has described the development of a prototype computer-based support system for constructing school timetable. The original framework introduced by Johnson [1993] is further extended and incorporated into the domain of school timetable planning. The scheme pinpoints on the existing resources that are available in a school. Efforts have been made on the elaboration of a practical and efficient design of producing school timetables in a low-end personal computer system. We also develop our program using dBase, which is a popular command language for data management applications and can be easily obtained in most of the school computer system.

School timetabling system is practically different from that of the universities not only in view of the structures, but also in the limitations of the computer devices and other facilities. It will be difficult to adopt a university system for a school system, since most of their formulated rules and procedures are case-specific. Mathematical programming is recognized as the backbone in the Operations Research. However, if we conceive the manner in which the combinatorial problems grow with the increase number of elements, the computational time may be discouraging. In particular, it is often unrealistic to incorporate all the requirements that a school could possibly wish for into a timetabling system. Any school timetable is necessarily a compromise. Lions [1967] emphasized the fact that a computer based system for producing school timetables could not be assessed solely from objective criteria.

The aim of our system is to perform the supporting role to the school timetablers, while retaining their decision making position during the timetable planning. In the automated component of the system, it covers the basic criteria that the school timetablers will frequently encounter during their planning. The conditions which include subject clustering, consecutive class block assignment, split block option and maximum daily teaching workload are incorporated explicitly into the algorithms. A timetabler may select the appropriate settings through the setup procedure to obtain the required timetable format. The module uses a 'One-Pass' strategy to obtain the first draft timetable so as to shorten the CPU time. As we discussed in the chapter 2, the drawback of this strategy might yield an incomplete schedule once an infeasibility has arisen. Chahal and De Werra [1989] have drawn the fact that it is not essential to get an entire schedule directly out of the personal computer in one shot. In reality, the timetablers often involve certain manual rearrangements in the preliminary stage so that any subjective request can be organized. In this case, an interactive component of the system has provided an efficient tool, which can also be used successfully by non-specialists with a substantial saving in time. The system also performs the bookkeeping task by printing the class, instructor and classroom schedules to alleviate the clerical burden.

The degree of success on the real timetable problem gives ground for hope that our approach is on the right trail, although it is not as large a size as that of university timetables handled by computers. More importantly, it demonstrates how the application of relatively simple device and software can give a much needed tool. The systems discussed in this study can be easily implemented to other school timetabling problems, as they only require a one-step process. It is hoped that the

efforts made herein can be developed as an effective tool for dealing with other important planning in this area.

6.3 Limitations, Future Works and Recommendations

The development of good software involves a lengthy and continuing process known as the software's life cycle. This process begins with a design, includes the writing, testing and debugging of programs, and continues for years to involve revisions and improvements in the original software. Future work on current prototype is to improve the compatibility to the operating system; perhaps the use of the latest version of Clipper compiler or other compilable DBMS language may be an alternative. Another interesting direction is to reduce the execution time of the program. The system partly adopts the sequential searching technique to assess the database, which causes a slight delay in data inquiry or retrieval when the database is becoming larger. Other possible improvements include more elaborate messages and instructions for the user, more elegant data presentation for the print out, better indication of appropriate slot for inquiry and a more sophisticated user input system. An on-line help and instruction manual are also recommended for consideration in future work.

Timetabling by computer is obviously influenced by the devices at hand. A higher performance computer can allow the programmer to accommodate more flexible constraints into the design. We feel that recent developments in computer technology and software engineering have furnished a proper environment for timetable programming. Nevertheless, it has been the objective of the author to

design a flexible support system for the timetabling process that could be adopted by various school systems. Within the scope, the system has proved its flexibility and efficiency. It is strongly recommended that the school administrators would adopt the prototype model developed here. A good practice is to allow this system to run in parallel with the manual system concurrently, so that any discrepancy can be amended appropriately. Once the timetabling components in the proposed design are acquainted with, the system can provide a faster and inexpensive way to help a school timetable planner to obtain more effective control on timetabling problem. The framework can be put forward as a starting point for improvements of the procedures and methods that are currently in use.

REFERENCES

Abramson, D., (1991) "Constructing School Timetable using simulated annealing Sequential and parallel Algorithm.", *Management Sciences*, Vol. 37, pp. 98-113.

Akkoyunlu, E. A., (1973) "A Linear algorithm for computing the optimal university timetable", *Computer Journals*, Vol. 16, pp. 347-350.

Almond, M., (1965) "An Algorithm for Constructing University Timetables", *Computer Journals*, Vol. 8, pp. 331-340.

Almond, M., (1969) "A University Faculty Timetable", *Computer Journals*, Vol. 12, pp. 215-217.

Appleby, J. S., Blake, D. V., and Newman, E. A., (1961) "Techniques for Producing School Timetables on a Computer and their Application to other Scheduling Problems", *Computer Journals*, Vol. 3, pp. 237.

Arani, T. and Lotfi, V., (1989) "A three-phased approach to final examination scheduling", *IIE Trans*, Vol. 21, pp. 86-96.

Aust, R. J., (1976) "An improvement algorithm for school timetabling", *Computer Journals*, Vol. 19, pp. 339-343.

Barham, A., and Westwood, J., (1978) "A simple heuristic to facilitate course timetabling", *Journal of Operational Research Society*, Vol. 29, pp. 1055-1060.

Barraclough, Elizabeth D., (1965) "The application of a digital computer to the construction of timetables", *Computer Journals*, Vol. 8, pp. 136-146.

Bharucha, Kerman D., (1989) "dBase IV: A comprehensive user's manual for non-programmers", Blue Ridge Summit, P.A., Windcrest.

Bloomfield, S. D. and McSharry M. M., (1979) "Preferential course scheduling", *Interfaces*, Vol. 9, pp. 24-31.

Brittan, J. N. G. and Farley, F. J. M., (1971) "College timetable construction by computer", *Computer Journals*, Vol. 14, pp. 361-365.

Brookes, J. (1980) "Timetable Planning", Royal Institute of Public Administration, UK.

Busam, V. A., (1967) "An algorithm for class scheduling with section preference", *ACM Com*, Vol. 10, No. 9, pp. 567-569.

Carter, M. W., (1986) "A survey of practical applications of Examination timetabling algorithms", *Operations Research*, Vol. 34, pp. 193-202.

Carter, M. W., (1989) "A Lagrangian Relaxation Approach to the Classroom Assignment Problem", *INFOR*, Vol. 27, No. 2, pp. 230-245.

Chahal, N., and De Werra, D., (1989) "An interactive system for constructing timetables on a PC", *European Journal of Operational Research*, Vol. 40, pp. 32-37.

Chou, George T. D., (1989) "dBase IV, Trips, Tricks, and Traps", Carmel, IN: Que Corp., 2nd Edition.

Chou, George T. D. and Tiley, Edward W., (1989) "dBase IV handbook", Carmel, IN: Que Corp.

Csima, J., and Gotlieb, C. C., (1964) "Test on a computer method for constructing school timetables", *ACM Com*, Vol. 7, No. 3, pp. 160-163.

Cuneo, K., and Bond, Gary C., (1988) " dBase III plus: quick reference handbook", Center for Professional, Computer and Education, New York, Wiley.

De Gans, O. B., (1981) "A computer timetabling system for secondary schools in the Netherlands", *European Journal of Operational Research*, Vol. 7, pp. 175-182.

De Werra, D., (1985) "An introduction to timetabling (Invited review)", *European Journal of Operational Research*, Vol. 19, pp. 151-162.

Dempsey, Richard A., and Traverso Henry P., (1983) "Scheduling the secondary school", National Association of Secondary School Principals.

Dhar, V., and Ranganathan, N., (1990) "Integer programming vs. expert systems: An experimental comparison. ", *ACM Com*, Vol. 33, No. 3, pp. 323-336.

Dinkel, J. J., Mote, J. and Venkataramanan, M. A., (1989) "An efficient decision support system for academic course scheduling", *Operations Research*, Vol. 37, pp. 853-864.

Dowsland, A. Kathryn, (1990) "A Timetabling problem in which Clashes are Inevitable", *Journal of Operational Research Society*, Vol. 41, pp. 907-918.

Dyer, J. S. and Mulvey, J. M., (1976) "The implementation of an integrated optimization / information system for academic departmental planning.", *Management Sciences*, The Institute of Management Science, Vol. 22, No. 12, pp. 1332-1341.

Eglese, R. W., and Rand, G. K., (1987) "Conference Seminar Timetabling", *Journal of Operational Research Society*, Vol. 38, pp. 591-598.

Even, S., Itai, A., and Shamir, (1976) "On the complexity of timetable and multicommodity flow problems", *SIAM Journal on Computing*, Vol. 5, pp. 691-703.

Ferland, J. A., and Roy, S., (1982) "Course scheduling and classroom assignment in a university", Publication 459, University of Montreal

Ferland, J. A., and Roy, S., (1985) "Timetabling problem for university as assignment of activities to resources", *Computers and Operations Research*, Vol. 12, No. 2, pp. 207-218.

Fisher, L., (1981) "The Lagrangian Relaxation Method for Solving Integer Programming Problems", *Management Science*, Vol. 27, No. 1, pp. 1-16.

Glassey, C. R. and Mizrach, M., (1986) "A Decision Support System for assigning classes to rooms", *Interfaces*, Vol. 16, No. 5, pp. 92-101.

Gosselin, K., and Truchon, M., (1986) "Allocation of classrooms by linear programming", *Journal of Operational Research Society*, Vol. 37, No. 6, pp. 561-569

Gotlieb, C. C., (1962) "The Construction of Class - Teacher Timetables", *Proceedings of the I.F.I.P. Congress*, North-Holland Publishing Co., Amsterdam, Vol. 7, pp. 73.

Grimmett, G. R., and McDiarmid, C. J. H., (1975) "On Colouring random Graphs", *Math. Proc. Camb. Phil. Soc.*, Vol. 77, pp. 313-324.

Harwood, B., and Lawless, W., (1975) "Optimizing Organizational goals in assigning faculty teaching scheduling", *Decision Science*, Vol. 6, pp. 513-524.

Hergent, D., (1987) "dBase III plus: The Microsoft reference guide to all commands, function and features", Redmond, Wash: Microsoft Press [New York].

Hillier, Frederick S., and Lieberman, Gerald J., (1990) "Introduction to Operations Research", McGraw-Hill Publishing Company, Singapore.

Hossein Bidgoli, (1991) "Information systems literacy and software productivity tools, dBase III plus", Macmillan Publishing Company, New York.

Johnson David, (1993) "A database approach to course timetabling", *Journal of Operational Research Society*, Vol. 44, pp. 425-433.

Junginger, W., (1986) "Timetabling in Germany - a survey", *Interfaces*, Vol. 16, No. 4, pp. 66-74.

Kang, L. and White, G., (1992) "A Logic Approach to the Resolution of Constraints in Timetabling" , *European Journal of Operational Research*, Vol. 61, pp. 306-317.

Karp, R. M., (1972) "Reducibility among combinatorial problems", Complexity of Computer Computations, Plenum Press, New York, pp. 85-104.

Kovacic, M., (1993) "Timetable construction with Markovian neural network", *European Journal of Operational Research*, Elsevier Science Publishers B. V., Vol. 69, pp. 92-96.

Lawrie, N., (1969) "An integer programming model of school timetabling problem.", *Computer Journals*, Vol. 12, pp. 307-316.

Lawrie, N., and Veitch, H., (1975) "Timetabling and Organization in secondary school", NFER Publishing Company, UK.

Leblood, Geoffrey T., (1989) "dBase IV: The complete reference ", Berkeley, Calif.: Osborne McGraw Hill.

Lewis, C. F., (1961) "The School Time Table", Cambridge University Press.

Li, W. M. and Wong, H. Y., (1994) "A computerized timetabling system ", Unpublished Case Study.

Lions, J., (1967) "The Ontario school scheduling program", *Computer Journals*, Vol. 10, pp. 14-21.

Loo, E. H., Goh, T. N. and Ong, H. L., (1986) "A Heuristic Approach to Scheduling University Timetables", *Computer and Education*, Vol. 10, No. 3, pp. 379-88.

Lotfi, V. and Cerveny, R., (1991) "A Final-exam-scheduling Package", *Journal of Operational Research Society*, Vol. 3, pp. 205-216.

McClure, H., and Wells, E., (1984) "A mathematical programming model for faculty course assignment", *Decision Science*, Vol. 15, pp. 409-419.

Mulvey, M., (1982) "A classroom / Time assignment model", *European Journal of Operational Research Society*, Vol. 9, pp. 64-70.

Papoulias, D. B., (1980) "The assignment-to-days problem in a school timetable, a heuristic approach", *European Journal of Operational Research*, Vol. 4, pp. 31-41.

Sabin, G. C. W. and Winter, G. K., (1986) "The impact of automated timetabling on Universities - A Case study", *Journal of Operational Research Society*, Vol. 37, No. 7, pp. 689-693.

Salazar, A. and Oakford, R. V. (1974) "A Graph Formulation of a School Scheduling Algorithm", *Communications of the ACM*, Vol. 17, No. 12, pp. 696-698

Salt, F. B. (1978) "Timetabling models for secondary schools", NFER Publishing Company, UK.

Schmidt, G., and Strohlein, T., (1980) "Timetable construction - an annotated bibliography", *Computer Journals*, Vol. 23, No. 4, pp. 307-316.

Selim, S. M., (1983) "An algorithm for producing course and lecture timetables", *Computer and Education*, Vol. 7, No. 2, pp. 101-108.

Shih, W. and Sullivan, James A., (1977) "Dynamic course scheduling for college faculty via zero-one programming", *Decision Science*, Vol. 8, pp. 711-721.

Townsend, J. J., (1992) "Introduction to Databases", Carmel, IN: Que Corp.

Tripathy, A., (1980) "A Lagrangean Relaxation Approach to Course Timetabling", *Journal of Operational Research Society*, Vol. 31, pp. 599-603.

Tripathy, A., (1984) "School - timetabling", *Management Science*, Vol. 30, No. 7 pp. 1473-1489.

White, G. M. and Wong, S. K. S., (1988) "Interactive timetabling in Universities", *Computer and Education*, Vol. 12, No. 4, pp. 521-529.

Winters, W. K., (1971) "A scheduling algorithm for a computer assisted registration system", *ACM Com*, Vol. 14, No. 3, pp. 166-171.

Wood, D. C. (1969) "A technique for colouring a graph applicable to large scale timetabling problems', *Computer Journals*, Vol. 12, pp. 317-319

Wong, H. Y., (1993) "A computerized timetabling system in an adult education institution", Unpublished Case Study.

APPENDIX A

Screen Output of the Timetabling System

APPENDIX A

TIMETABLING SYSTEM MENU

OPERATIONS

1. AUTO ASSIGNMENT
2. INQUIRY / MANUAL ADJUSTMENT
3. OUTPUT TO FILE / PRINTER

MAINTENANCE

51. SET UP / CHANGE SETTING
52. UPDATE FILE

Enter Option : __

ESC = CANCEL

U P D A T E F I L E U T I L I T Y

1. PREASSIGNMENT FILE
2. ASSIGNMENT FILE
3. ASSIGNMENT → PREASSIGNMENT FILE

Enter Option : __

ESC = CANCEL

TIME TABLE - INQUIRY

Legend: Subject / Teacher / Room
CLASS : 5E TEACHER : __ ROOM : __

FORM 5E

	Day 1	Day 2	Day 3	Day 4	Day 5
1					
2					
3					
4	11/YW/B1	11/YW/B1			
5	17/KL/B5		**17/KL/B5**		
6	**15/WY/B1**		**17/KL/B5**	14/SC/B1	
7	**15/WY/B1**	**14/SC/B1**	**9 /CY/B1**	**11/YW/CL**	**9 /CY/B1**
8	9 /CY/B1	**14/SC/B1**	**9 /CY/B1**	**11/YW/CL**	**9 /CY/B1**
9	**90/YH/A5**	19/YK/B1	15/WY/B5	**22/CF/T1**	**14/SC/LL**
10	**90/YH/A5**	17/KL/B1	14/SC/B1	**22/CF/T1**	**14/SC/LL**

Record	Subject	Type	Teacher	Room	# of unalloc session
88	15/MATHS	S	WY/WHY	B1	1

ESC=Cancel F5=Session Switch F7=Room Switch F9=Manually Assign

---

TIME TABLE - PRINT

CLASS : ___
TEACHER : ___
ROOM : ___

- use '*' as wildcard character
  e.g. enter '4*' to match all codes start with '4'
  enter '*' to match all codes (i.e. PRINT ALL)
- e.g. to print Time Table for all Form 4 classes, for all teachers, and for all special room, please enter '4*', '* ' and '* ' respectively

ESC = Cancel

A SUMMARY TO THE CURRENT SETTING

Number of Days ..... (2..15) : 5

Number of Periods in a Day ..... (2..15) : 10

Total Number of Recess and Break ..... (0.. 5) : 2

The Recess and Break Structure ..... (1..15) : 2 - 6 - 2

Maximum Daily Teaching Work Load ..... (1..15) : 6

Split Block Allowed, Same Single Period May Repeat Once (Y / N) : N

Choice of Printing Device : C:\TT1\TT3.OUT
The present of '@' means to hide command in batch file : @

CHANGE THE CURRENT SETTING (Y / N) : __

APPENDIX B

Summary of Course Modules and their Descriptions in the Sample School

APPENDIX B

Table 1, A General Description for Subject Allocation in Each Module.

AF60

ABBREV	UE	CHI_C	ACCTN*	MSTAT	COM_U	RS	ECON*			TOTAL
DL	1				1					
D	2	1	3	1			3			
S		2		2	1	1				24

BF40 and BF50

ABBREV	ENG_L	CHI_C	P_ACC	MATHS	COM_S*	RS	TYPE	COMM	CHIST*	TOTAL
DL	1				1		1			
D	1	2	2	1				1	1	
S	2	1		2	2	1				26 - 28

CSC130A

ABBREV	B_ENG	P_CHI	P_ACC	MSTAT	W_PRO	RS	ECON			TOTAL
DL					1					
D	2		3	1			1			
S	2	1		2		1	2			24

CSC130B

ABBREV	B_ENG	P_CHI	BK	BMATH	W_PRO	RS	TYPE	COMM		TOTAL
DL					1		2			
D	1		2					1		
S	2	1	2	2		1				22

D140

ABBREV	B_ENG	P_CHI	BK	SH	W_PRO	RS	TYPE	SEC_P		TOTAL
DL					1					
D	2			3			2	1		
S		1	2	1		1				23

ESC110 **

ABBREV	ENG_L	ASTAT	ACCTN	RS	COM_S					TOTAL
DL					2					
D	2	1	3		1					
S	1	2		1	2					24

ESC180

ABBREV	ENG_L	ASTAT	ACCTN	RS	W_PRO	ECON				TOTAL
DL					1					
D	2	2	3			2				
S	2			1		1				24

Table 2, A general description for each category

Category	General Description	Total number of classes
AF60	Form 6	2
BF50	Form 5	5
BF40	Form 4	7
CSC130A	Commercial Studies IA	2
CSC130B	Commercial Studies IB	2
D140	Secretarial Training	2
ESC110 **	Business Computing	1
ESC180 #	Commercial Studies II	2
TOTAL		23

* Either one of the subjects will be offered in a class. Except in AF60, both subjects will be conducted in parallel at the same time. Student may take either one of those.

** The last year offered module.

Second year course for CSC130A and CSC130B.

DL, D, S be Double class conducted in specially equipped room, Double class and Single class respectively.

APPENDIX C

Input, Output Databases

APPENDIX C: CLSLOTPF.DBF

09/12/94

CLASS	TYPE	SLOT
4A	S	1
4A	S	2
4A	S	3
4A	S	4
4A	S	5
4A	S	6
4A	S	7
4A	S	8
4A	S	9
4A	S	10
4A	S	11
4A	S	12
4A	S	13
4A	S	14
4A	S	15
4A	S	16
4A	S	17
4A	S	18
4A	S	19
4A	S	20
4A	S	21
4A	S	22
4A	S	23
4A	S	24
4A	S	25
4A	S	26
4A	S	27
4A	S	28
4A	S	29
4A	S	30
4A	S	31
4A	S	32
4A	S	33
4A	S	34
4A	S	35
4A	S	36
4A	S	37
4A	S	38
4A	S	39
4A	S	40
4A	S	41
4A	S	42
4A	S	43
4A	S	44
4A	S	45
4A	S	46
4A	S	47
4A	S	48
4A	S	49

CLASS	TYPE	SLOT
4A	S	50
4A	T	3
4A	T	4
4A	T	5
4A	T	6
4A	T	13
4A	T	14
4A	T	15
4A	T	16
4A	T	23
4A	T	24
4A	T	25
4A	T	26
4A	T	33
4A	T	34
4A	T	35
4A	T	36
4A	T	43
4A	T	44
4A	T	45
4A	T	46
4A	D	1
4A	D	3
4A	D	4
4A	D	5
4A	D	6
4A	D	7
4A	D	9
4A	D	11
4A	D	13
4A	D	14
4A	D	15
4A	D	16
4A	D	17
4A	D	19
4A	D	21
4A	D	23
4A	D	24
4A	D	25
4A	D	26
4A	D	27
4A	D	29
4A	D	31
4A	D	33
4A	D	34
4A	D	35
4A	D	36
4A	D	37
4A	D	39
4A	D	41
4A	D	43
4A	D	44
4A	D	45
4A	D	46
4A	D	47
4A	D	49

APPENDIX C: DAYSESPF.DBF

09/12/94

SLOT	DAY	SESSION
1	1	1
2	2	2
3	3	3
4	4	4
5	5	5
6	1	6
7	2	7
8	3	8
9	4	9
10	5	10
11	2	1
12	3	2
13	4	3
14	5	4
15	1	5
16	2	6
17	3	7
18	4	8
19	5	9
20	1	10
21	3	1
22	4	2
23	5	3
24	1	4
25	2	5
26	3	6
27	4	7
28	5	8
29	1	9
30	2	10
31	4	1
32	5	2
33	1	3
34	2	4
35	3	5
36	4	6
37	5	7
38	1	8
39	2	9
40	3	10
41	5	1
42	1	2
43	2	3
44	3	4
45	4	5
46	5	6
47	1	7
48	2	8
49	3	9
50	4	10

APPENDIX C: TDSPF.DBF

09/12/94

TYPE	SLOT1	SLOT2	SLOT3
S	1		
S	2		
S	3		
S	4		
S	5		
S	6		
S	7		
S	8		
S	9		
S	10		
S	11		
S	12		
S	13		
S	14		
S	15		
S	16		
S	17		
S	18		
S	19		
S	20		
S	21		
S	22		
S	23		
S	24		
S	25		
S	26		
S	27		
S	28		
S	29		
S	30		
S	31		
S	32		
S	33		
S	34		
S	35		
S	36		
S	37		
S	38		
S	39		
S	40		
S	41		
S	42		
S	43		
S	44		
S	45		
S	46		
S	47		
S	48		
S	49		
S	50		

TYPE	SLOT1	SLOT2	SLOT3
T	3	44	35
T	4	45	36
T	5	46	37
T	6	47	38
T	13	4	45
T	14	5	46
T	15	6	47
T	16	7	48
T	23	14	5
T	24	15	6
T	25	16	7
T	26	17	8
T	33	24	15
T	34	25	16
T	35	26	17
T	36	27	18
T	43	34	25
T	44	35	26
T	45	36	27
T	46	37	28
D	1	42	
D	3	44	
D	4	45	
D	5	46	
D	6	47	
D	7	48	
D	9	50	
D	11	2	
D	13	4	
D	14	5	
D	15	6	
D	16	7	
D	17	8	
D	19	10	
D	21	12	
D	23	14	
D	24	15	
D	25	16	
D	26	17	
D	27	18	
D	29	20	
D	31	22	
D	33	24	
D	34	25	
D	35	26	
D	36	27	
D	37	28	
D	39	30	
D	41	32	
D	43	34	
D	44	35	
D	45	36	
D	46	37	
D	47	38	
D	49	40	

APPENDIX C: CLASSPF.DBF

9/12/94

CLASS	ABBREV	DESC
4A	F4A	FORM 4A
4B	F4B	FORM 4B
4C	F4C	FORM 4C
4D	F4D	FORM 4D
4E	F4E	FORM 4E
4F	F4F	FORM 4F
4G	F4G	FORM 4G
5A	F5A	FORM 5A
5B	F5B	FORM 5B
5C	F5C	FORM 5C
5D	F5D	FORM 5D
5E	F5E	FORM 5E
6A	F6A	FORM 6A
6B	F6B	FORM 6B
B1	C_A1	COM. STUDIES A1
B2	C_A2	COM. STUDIES A2
C1	C_B1	COM. STUDIES B1
C2	C_B1	COM. STUDIES B2
GA	C_IIA	COM. STUDIES 2A
GB	C_IIB	COM. STUDIES 2B
P2	BCOMP	BUS. COMPUTING
S1	SEC_A	SEC. TRAINING A
S2	SEC_B	SEC. TRAINING B

APPENDIX C: ROOMPF.DBF

09/12/94

ROOM	ABBREV	DESC
A1	A_2A	RM 2A, BLOCK A
A2	A_2B	RM 2B, BLOCK A
A3	A_2C	RM 2C, BLOCK A
A4	A_2D	RM 2D, BLOCK A
A5	A_2E	RM 2E, BLOCK A
A6	A_6A	RM 6A, BLOCK A
A7	A_6B	RM 6B, BLOCK A
B1	B_4A	RM 4A, BLOCK B
B2	B_5B	RM 5B, BLOCK B
B3	B_5C	RM 5C, BLOCK B
B4	B_5D	RM 5D, BLOCK B
B5	B_7A	RM 7A, BLOCK B
CL	COM_L	COMPUTER ROOM
H1	HALL	HALL
LL	LAN_L	LANGUAGE LABORATORY
GP	GP_RM	G.P.ROOM
T1	TYP_A	TYPING ROOM A
T2	TYP_B	TYPING ROOM B

APPENDIX C: SUBJPF.DBF

09/12/94

SUBJECT	ABBREV	DESC
1	ACCTN	ACCOUNTING
2	ASSEM	ASSEMBLY
3	BK	BOOKKEEPING
4	BMATH	BUSINESS MATHMATICS
5	BSTAT	STATISTICS
6	B_ENG	BUSINESS ENGLISH
7	CHILC	C.LANGUAGE & CULTURE
8	CHIST	CHINESE HISTORY
9	CHI_L	CHINESE LANGUAGE
10	COMM	COMMERCE
11	COM_S	COMPUTER STUDIES
12	COM_U	COMPUTER USAGES
13	ECON	ECONOMICS
14	ENG_L	ENGLISH LANGUAGE
15	MATHS	MATHMATICS
16	MSTAT	MATHS & STATISTICS
17	P_ACC	PRINCIPLE OF ACCOUNT
18	P_CHI	PRACTICAL CHINESE
19	RS	RELIGOUS STUDIES
20	SEC_P	SECRETARIAL PRACTICE
21	SH	SHORT HAND
22	TYPE	TYPING
23	U_ENG	USE OF ENGLISH
24	W_PRO	WORD PROCESSING
25	ASTAT	ADVANCED STATISTICS
90	CO_CO	COMMERCE/COMMERCE
91	EC_AC	ECONOMICS/ACCOUNTING
99	OTHER	OTHERS

APPENDIX C: TCHERPF.DBF

09/12/94

TCHER	ABBREV	DESC
CF	CYF	MS. CHEUNG Y. F.
CK	CKK	MR. CHAN K. K.
WY	WHY	MR. WONG H. Y.
CY	HCY	MR. HO C. Y.
FC	WFC	MS. WONG F. C.
FS	WFS	MR. WU F. S.
HT	LHT	MR. LIN H.T.
HY	CHY	MR. CHANG H. Y.
KF	CKF	MR. CHARM K.F.
KK	TKK	MR. TAM K. K.
KL	NKL	MS. NG K. L.
KS	WKS	MR. WONG K. S.
KU	KU	MR. KU
LH	LSH	MS. LAU S. H.
LS	CLS	MS. CHEUNG L. S.
MW	LMW	MS. LO M. W.
SC	MSC	MR. MA S. C
SH	HSH	MR. HEUNG S. H.
SL	CSL	MR. CHAN S. L.
SM	WSM	MS. WONG S. M.
WS	SWS	MS. SO. W. S.
TW	WSF	MS. WONG S. F.
LW	LWS	MS. LEUNG W.S.
Y1	WYH	MS. WONG Y. H.
YF	CYF	MS. CHOI Y. F.
YH	WYH	MS. WONG Y. H.
YK	YKL	MR. YU K. L.
YW	YYW	MR. YAU Y.W.

APPENDIX C: ALTRMPF.DBF

09/12/94

ROOM	ALTRM
A1	A2
A2	A1
A3	A4
A3	A5
A4	A5
A6	A7
A7	A6
B1	B5
B2	B3
B2	B4
B2	B5
B3	B5
B3	B4
B4	B3
B4	B5
T1	T2
T2	T1

APPENDIX C: CLSUBJPF.DBF

09/12/94

SEQ	PRIORITY	CLASS	SUBJECT	TCHER	ROOM	CLASSTYPE	REPEAT	FINISHED
1	4	P2	14	SM	B4	D	2	2
2	6	P2	14	SM	B4	S	1	1
3	3	P2	11	YW	CL	DL	2	2
4	4	P2	11	YW	B4	D	1	1
5	6	P2	11	YW	B4	S	2	2
6	4	P2	1	KS	B4	D	3	3
7	4	P2	25	WY	B4	D	1	1
8	6	P2	25	WY	B4	S	2	2
9	6	P2	19	YK	B4	S	1	1
10	4	B2	13	KF	B5	D	1	1
11	6	B2	13	KF	B5	S	2	2
12	3	B2	24	HY	CL	D	1	1
13	4	B2	6	LW	B5	D	2	2
14	6	B2	6	LW	B5	S	2	0
15	4	B2	17	KS	B5	D	3	3
16	4	B2	5	FS	B5	D	1	1
17	6	B2	5	FS	B5	S	2	0
18	6	B2	19	YK	B5	S	1	1
19	6	B2	18	KU	B5	S	1	1
20	4	4A	9	FC	A6	D	2	2
21	6	4A	9	FC	A6	S	1	1
22	4	4A	15	HY	A6	D	1	1
23	6	4A	15	HY	A6	S	2	2
24	4	4A	17	FS	A6	D	2	2
25	3	4A	11	SH	CL	DL	1	1
26	6	4A	11	SH	A6	S	2	2
27	4	4A	14	WS	LL	DL	1	1
28	4	4A	14	WS	A6	D	1	1
29	6	4A	14	WS	A6	S	2	2
30	4	4A	22	TW	T2	DL	1	1
31	6	4A	10	YF	A6	S	2	2
32	4	4A	19	YK	A6	S	1	1
33	4	4C	9	FC	A7	D	2	2
34	6	4C	9	FC	A7	S	1	1
35	4	4C	15	HY	A7	D	1	1
36	6	4C	15	HY	A7	S	2	2
37	4	4C	17	FS	A7	D	2	2
38	3	4C	11	SH	CL	DL	1	1
39	6	4C	11	SH	A7	S	2	2
40	4	4C	14	WS	LL	DL	1	1
41	4	4C	14	WS	A7	D	1	1
42	6	4C	14	WS	A7	S	2	2
43	4	4C	22	TW	T2	DL	1	1
44	6	4C	10	YH	A7	S	2	2
45	4	4C	19	YK	A7	S	1	1
46	4	4D	9	HT	A4	D	2	2
47	6	4D	9	HT	A4	S	1	1
48	4	4D	15	MW	A4	D	1	1
49	6	4D	15	MW	A4	S	2	2
50	4	4D	17	YF	A4	D	2	2

09/12/94

SEQ	PRIORITY	CLASS	SUBJECT	TCHER	ROOM	CLASSTYPE	REPEAT	FINISHED
51	3	4D	11	SH	CL	DL	1	1
52	6	4D	11	SH	A4	S	2	2
53	4	4D	14	WS	LL	DL	1	1
54	4	4D	14	WS	A4	D	1	1
55	6	4D	14	WS	A4	S	2	2
56	4	4D	22	YH	T2	DL	1	1
57	6	4D	10	YH	A4	S	2	2
58	4	4D	19	YK	A4	S	1	1
59	4	5A	14	LH	LL	DL	1	1
60	4	5A	14	LH	A5	D	1	1
61	6	5A	14	LH	A5	S	2	2
62	4	5A	15	MW	A5	D	1	1
63	6	5A	15	MW	A5	S	2	2
64	4	5A	8	HT	T2	D	1	1
65	6	5A	19	YK	A5	S	1	1
66	4	5A	10	YF	LL	D	1	1
67	4	5A	9	HT	A5	D	2	2
68	6	5A	9	HT	A5	S	1	1
69	4	5A	17	KS	A5	D	2	2
70	4	5A	22	TW	T2	D	1	1
71	4	5C	14	LH	LL	D	1	1
72	4	5C	14	LH	A2	D	1	1
73	6	5C	14	LH	A2	S	2	2
74	4	5C	15	MW	A2	D	1	1
75	6	5C	15	MW	A2	S	2	2
76	4	5C	11	SH	CL	DL	1	1
77	6	5C	11	SH	A2	S	2	2
78	7	5C	19	YK	A2	S	1	1
80	4	5C	9	CY	A2	D	2	2
81	6	5C	9	CY	A2	S	1	1
82	4	5C	17	KL	A2	D	2	2
83	4	5C	22	CF	T1	D	1	1
84	4	5E	14	SC	LL	D	1	1
85	4	5E	14	SC	B1	D	1	1
86	6	5E	14	SC	B1	S	2	2
87	4	5E	15	WY	B1	D	1	1
88	6	5E	15	WY	B1	S	2	1
89	4	5E	11	YW	CL	DL	1	1
90	6	5E	11	YW	B1	S	2	2
91	7	5E	19	YK	B1	S	1	1
93	4	5E	9	CY	B1	D	2	2
94	6	5E	9	CY	B1	S	1	1
95	4	5E	17	KL	B1	D	1	1
96	4	5E	22	CF	T1	D	1	1
97	6	5E	17	KL	B1	S	2	2
98	4	C1	3	KL	A1	D	2	2
99	6	C1	3	KL	A1	S	2	2
100	4	C1	24	SL	CL	DL	1	1
101	4	C1	6	LW	A1	D	1	1
102	6	C1	6	LW	A1	S	2	2

09/12/94

SEQ	PRIORITY	CLASS	SUBJECT	TCHER	ROOM	CLASSTYPE	REPEAT	FINISHED
103	4	C1	22	CF	T2	D	2	2
104	6	C1	4	HY	A1	S	2	2
105	4	C1	10	YF	A1	D	1	1
106	6	C1	19	YK	A1	S	1	1
107	6	C1	18	KU	A1	S	1	1
108	4	C2	3	KL	B3	D	2	2
109	6	C2	3	KL	B3	S	2	2
110	4	C2	24	SL	CL	DL	1	1
111	4	C2	6	CK	B3	D	1	1
112	6	C2	6	CK	B3	S	2	2
113	4	C2	22	CF	T1	D	2	2
114	6	C2	4	HY	B3	S	2	2
115	4	C2	10	YF	B3	D	1	1
116	6	C2	19	YK	B3	S	1	1
117	6	C2	18	KU	B3	S	1	1
118	6	6B	19	YK	A3	S	1	1
119	4	6B	16	WY	A3	D	1	1
120	6	6B	16	WY	A3	S	2	2
121	5	6B	23	SC	LL	DL	1	1
122	4	6B	23	SC	A3	D	2	2
123	4	6B	7	CY	A3	D	1	1
124	6	6B	7	CY	A3	S	2	2
125	3	6B	12	YW	CL	D	1	1
126	6	6B	12	YW	A3	S	1	1

APPENDIX C: ASSIGNPF.DBF

09/12/94

CLASS	SLOT	TCHER	ROOM	SUBJECT	DAY	SESSION	TYPE	TYPE1
4G	1	SL	CL	11	1	1	D	1
4G	42	SL	CL	11	1	2	D	2
4E	3	SL	CL	11	3	3	D	1
4E	44	SL	CL	11	3	4	D	2
S2	13	SL	CL	24	4	3	D	1
S2	4	SL	CL	24	4	4	D	2
6A	1	LS	A4	1	1	1	D	1
6A	42	LS	A4	1	1	2	D	2
6A	3	LS	A4	1	3	3	D	1
6A	44	LS	A4	1	3	4	D	2
GA	5	LS	B3	1	5	5	D	1
GA	46	LS	B3	1	5	6	D	2
GA	13	LS	B3	1	4	3	D	1
GA	4	LS	B3	1	4	4	D	2
GA	15	LS	B3	1	1	5	D	1
GA	6	LS	B3	1	1	6	D	2
GB	21	LS	B2	1	3	1	D	1
GB	12	LS	B2	1	3	2	D	2
GB	23	LS	B2	1	5	3	D	1
GB	14	LS	B2	1	5	4	D	2
B1	1	KF	A2	13	1	1	D	1
B1	42	KF	A2	13	1	2	D	2
GA	11	KF	B3	13	2	1	D	1
GA	2	KF	B3	13	2	2	D	2
GB	13	KF	B2	13	4	3	D	1
GB	4	KF	B2	13	4	4	D	2
4G	2	SC	B4	14	2	2	S	1
4G	4	SC	B4	14	4	4	S	1
5D	1	SC	B1	14	1	1	D	1
5D	42	SC	B1	14	1	2	D	2
5D	5	SC	B1	14	5	5	S	1
5D	11	SC	B1	14	2	1	S	1
GA	1	KK	B3	14	1	1	D	1
GA	42	KK	B3	14	1	2	D	2
GA	21	KK	B3	14	3	1	D	1
GA	12	KK	B3	14	3	2	D	2
GB	3	KK	B2	14	3	3	D	1
GB	44	KK	B2	14	3	4	D	2
GB	11	KK	B2	14	2	1	D	1
GB	2	KK	B2	14	2	2	D	2
5B	1	LH	A6	14	1	1	D	1
5B	42	LH	A6	14	1	2	D	2
4F	1	HY	A5	15	1	1	D	1
4F	42	HY	A5	15	1	2	D	2
5D	3	MW	B1	15	3	3	D	1
5D	44	MW	B1	15	3	4	D	2
5D	2	MW	B1	15	2	2	S	1
4B	11	FS	A7	17	2	1	S	1
4B	2	FS	A7	17	2	2	S	2
4F	3	YH	A5	17	3	3	D	1
4F	44	YH	A5	17	3	4	D	2
4G	21	YH	B4	17	3	1	D	1

CLASS	SLOT	TCHER	ROOM	SUBJECT	DAY	SESSION	TYPE	TYPE1
4G	12	YH	B4	17	3	2	D	2
B1	11	KS	A2	17	2	1	D	1
B1	2	KS	A2	17	2	2	D	2
5D	21	KL	B1	17	3	1	D	1
5D	12	KL	B1	17	3	2	D	2
4E	11	YF	A3	17	2	1	D	1
4E	2	YF	A3	17	2	2	D	2
S2	1	YK	B5	19	1	1	S	1
S2	42	FS	B5	3	1	2	S	1
S1	1	YF	A1	20	1	1	D	1
S1	42	YF	A1	20	1	2	D	2
S1	3	TW	A1	21	3	3	D	1
S1	44	TW	A1	21	3	4	D	2
S1	13	TW	A1	21	4	3	D	1
S1	4	TW	A1	21	4	4	D	2
S2	11	TW	B5	21	2	1	D	1
S2	2	TW	B5	21	2	2	D	2
S2	21	TW	B5	21	3	1	D	1
S2	12	TW	B5	21	3	2	D	2
6A	21	LH	A4	23	3	1	D	1
6A	12	LH	A4	23	3	2	D	2
GB	1	WY	B2	25	1	1	D	1
GB	42	WY	B2	25	1	2	D	2
GB	31	WY	B2	25	4	1	D	1
GB	22	WY	B2	25	4	2	D	2
B1	13	FS	A2	5	4	3	D	1
B1	4	FS	A2	5	4	4	D	2
B1	21	FS	A2	5	3	1	D	1
B1	12	FS	A2	5	3	2	D	2
S1	31	LW	A1	6	4	1	D	1
S1	22	LW	A1	6	4	2	D	2
S2	33	LW	B5	6	1	3	D	1
S2	24	LW	B5	6	1	4	D	2
B1	23	LW	A2	6	5	3	D	1
B1	14	LW	A2	6	5	4	D	2
4F	21	FC	A5	9	3	1	D	1
4F	12	FC	A5	9	3	2	D	2
4F	23	FC	A5	9	5	3	D	1
4F	14	FC	A5	9	5	4	D	2
5D	25	CY	B1	9	2	5	D	1
5D	16	CY	B1	9	2	6	D	2
5D	31	CY	B1	9	4	1	D	1
5D	22	CY	B1	9	4	2	D	2
5B	11	HT	A6	9	2	1	D	1
5B	2	HT	A6	9	2	2	D	2
4E	31	FC	A3	9	4	1	D	1
4E	22	FC	A3	9	4	2	D	2
5B	25	LH	LL	14	2	5	D	1
5B	16	LH	LL	14	2	6	D	2
5D	33	CF	T2	22	1	3	D	1
5D	24	CF	T2	22	1	4	D	2
6A	31	LH	LL	23	4	1	D	1
6A	22	LH	LL	23	4	2	D	2
GA	31	SL	CL	24	4	1	D	1
GA	22	SL	CL	24	4	2	D	2

CLASS	SLOT	TCHER	ROOM	SUBJECT	DAY	SESSION	TYPE	TYPE1
GB	33	SL	CL	24	1	3	D	1
GB	24	SL	CL	24	1	4	D	2
5B	22	YF	A6	10	4	2	S	1
4F	13	KK	A5	14	4	3	S	1
5B	32	LH	A6	14	5	2	S	1
5B	23	YK	A6	19	5	3	S	1
B1	43	LW	A2	6	2	3	S	1
4G	31	HT	B4	9	4	1	S	1
S1	41	TW	A1	21	5	1	D	1
S1	32	TW	A1	21	5	2	D	2
S1	11	LW	A1	6	2	1	D	1
S1	2	LW	A1	6	2	2	D	2
5B	41	YW	A6	11	5	1	S	1
S1	21	YW	CL	24	3	1	D	1
S1	12	YW	CL	24	3	2	D	2
S1	43	FS	A1	3	2	3	D	1
S1	34	FS	A1	3	2	4	D	2
S1	25	TW	T1	22	2	5	D	1
S1	16	TW	T1	22	2	6	D	2
S2	41	LW	B5	6	5	1	D	1
S2	32	LW	B5	6	5	2	D	2
S2	31	FS	B5	3	4	1	S	1
S2	43	YF	B5	20	2	3	D	1
S2	34	YF	B5	20	2	4	D	2
S1	33	TW	T1	22	1	3	D	1
S1	24	TW	T1	22	1	4	D	2
S2	23	TW	B5	21	5	3	D	1
S2	14	TW	B5	21	5	4	D	2
S2	15	CF	T2	22	1	5	D	1
S2	6	CF	T2	22	1	6	D	2
S1	14	YK	A1	19	5	4	S	1
S1	23	KU	A1	18	5	3	S	1
S1	15	TW	A1	21	1	5	S	1
6A	41	LS	A4	1	5	1	D	1
6A	32	LS	A4	1	5	2	D	2
6A	11	YW	CL	12	2	1	D	1
6A	2	YW	CL	12	2	2	D	2
4B	41	FC	A7	9	5	1	D	1
4B	32	FC	A7	9	5	2	D	2
4F	31	YK	A5	19	4	1	S	1
4F	43	SL	A5	11	2	3	S	1
4F	41	YH	A5	17	5	1	D	1
4F	32	YH	A5	17	5	2	D	2
5B	31	YF	A6	10	4	1	S	1
5B	21	KS	A6	17	3	1	D	1
5B	12	KS	A6	17	3	2	D	2
5B	15	KS	A6	17	1	5	D	1
5B	6	KS	A6	17	1	6	D	2
5B	13	HT	A6	9	4	3	D	1
5B	4	HT	A6	9	4	4	D	2
B1	41	SL	CL	24	5	1	D	1
B1	32	SL	CL	24	5	2	D	2
B1	31	KS	A2	17	4	1	D	1
B1	22	KS	A2	17	4	2	D	2
B1	33	KS	A2	17	1	3	D	1
B1	24	KS	A2	17	1	4	D	2
B1	3	LW	A2	6	3	3	D	1

CLASS	SLOT	TCHER	ROOM	SUBJECT	DAY	SESSION	TYPE	TYPE1
B1	44	LW	A2	6	3	4	D	2
B1	15	KF	A2	13	1	5	S	1
4G	41	HT	B4	9	5	1	D	1
4G	32	HT	B4	9	5	2	D	2
4G	3	HT	B4	9	3	3	D	1
4G	44	HT	B4	9	3	4	D	2
4G	15	SC	LL	14	1	5	D	1
4G	6	SC	LL	14	1	6	D	2
4G	33	YH	B4	17	1	3	D	1
4G	24	YH	B4	17	1	4	D	2
4G	23	SC	B4	14	5	3	D	1
4G	14	SC	B4	14	5	4	D	2
4G	43	HY	B4	15	2	3	D	1
4G	34	HY	B4	15	2	4	D	2
4B	33	YF	A7	10	1	3	S	1
6A	33	LH	A4	23	1	3	D	1
6A	24	LH	A4	23	1	4	D	2
6A	15	YW	A4	12	1	5	S	1
6A	35	WY	A4	16	3	5	S	1
5D	41	SH	B1	11	5	1	S	1
5B	44	LH	A6	14	3	4	S	1
5B	3	YW	A6	11	3	3	S	1
5B	14	MW	A6	15	5	4	S	1
5D	32	YK	B1	19	5	2	S	1
5D	43	SH	CL	11	2	3	D	1
5D	34	SH	CL	11	2	4	D	2
5D	23	KL	B1	17	5	3	D	1
5D	14	KL	B1	17	5	4	D	2
5D	15	CY	B1	9	1	5	S	1
GA	34	KK	B3	14	2	4	S	1
GA	43	KK	B3	14	2	3	S	1
GA	23	KF	B3	13	5	3	D	1
GA	14	KF	B3	13	5	4	D	2
GA	41	WY	B3	25	5	1	D	1
GA	32	WY	B3	25	5	2	D	2
GA	3	WY	B3	25	3	3	D	1
GA	44	WY	B3	25	3	4	D	2
GB	41	KF	B2	13	5	1	D	1
GB	32	KF	B2	13	5	2	D	2
GB	43	LS	B2	1	2	3	D	1
GB	34	LS	B2	1	2	4	D	2
GB	45	KK	B2	14	4	5	S	1
GB	15	YK	B2	19	1	5	S	1
GB	25	KK	B2	14	2	5	S	1
GB	35	KF	B2	13	3	5	S	1
4E	35	YF	A3	10	3	5	S	1
4E	41	KK	A3	14	5	1	D	1
4E	32	KK	A3	14	5	2	D	2
4E	21	MW	A3	15	3	1	D	1
4E	12	MW	A3	15	3	2	D	2
4E	1	FC	A3	9	1	1	D	1
4E	42	FC	A3	9	1	2	D	2
5B	43	HT	A6	9	2	3	S	1
5B	34	MW	A6	15	2	4	S	1
5B	33	MW	A6	15	1	3	D	1
5B	24	MW	A6	15	1	4	D	2

CLASS	SLOT	TCHER	ROOM	SUBJECT	DAY	SESSION	TYPE	TYPE1
4E	43	MW	A3	15	2	3	S	1
4E	36	KK	A3	14	4	6	S	1
4B	6	HY	A7	15	1	6	S	1
4B	1	WS	A7	14	1	1	D	1
4B	42	WS	A7	14	1	2	D	2
4B	31	SH	A7	11	4	1	S	1
4B	22	SH	A7	11	4	2	S	1
4F	25	CF	T2	22	2	5	D	1
4F	16	CF	T2	22	2	6	D	2
4G	11	SL	B4	11	2	1	S	1
4G	5	HY	B4	15	5	5	S	1
4F	11	HY	A5	15	2	1	S	1
4G	46	SL	B4	11	5	6	S	1
5B	45	YW	CL	11	4	5	D	1
5B	36	YW	CL	11	4	6	D	2
4F	5	KK	LL	14	5	5	D	1
4F	46	KK	LL	14	5	6	D	2
4F	35	SL	CL	11	3	5	D	1
4F	26	SL	CL	11	3	6	D	2
4G	45	CF	T1	22	4	5	D	1
4G	36	CF	T1	22	4	6	D	2
4G	25	YH	B5	10	2	5	D	1
4G	16	YH	B5	10	2	6	D	2
5D	47	SC	LL	14	1	7	D	1
5D	38	SC	LL	14	1	8	D	2
5D	37	YH	A7	10	5	7	D	1
5D	28	YH	A7	10	5	8	D	2
5C	29	YH	A5	90	1	9	D	1
5C	20	YH	A5	90	1	10	D	2
5E	29	YH	A5	90	1	9	D	1
5E	20	YH	A5	90	1	10	D	2
6B	29	LS	A2	91	1	9	D	1
6B	20	LS	A2	91	1	10	D	2
6B	17	LS	A2	91	3	7	D	1
6B	8	LS	A2	91	3	8	D	2
6B	9	LS	A2	91	4	9	D	1
6B	50	LS	A2	91	4	10	D	2
6B	29	KF	A3	91	1	9	D	1
6B	20	KF	A3	91	1	10	D	2
6B	17	KF	A3	91	3	7	D	1
6B	8	KF	A3	91	3	8	D	2
6B	9	KF	A3	91	4	9	D	1
6B	50	KF	A3	91	4	10	D	2
4A	7	SH	CL	11	2	7	D	1
4A	48	SH	CL	11	2	8	D	2
4D	17	SH	CL	11	3	7	D	1
4D	8	SH	CL	11	3	8	D	2
6B	19	YW	CL	12	5	9	D	1
6B	10	YW	CL	12	5	10	D	2
B2	25	HY	CL	24	2	5	D	1
B2	16	HY	CL	24	2	6	D	2
P2	7	KS	B5	1	2	7	D	1
P2	48	KS	B5	1	2	8	D	2
C2	6	YF	B4	10	1	6	D	1
C2	47	YF	B4	10	1	7	D	2

CLASS	SLOT	TCHER	ROOM	SUBJECT	DAY	SESSION	TYPE	TYPE1
5C	37	SH	CL	11	5	7	D	1
5C	28	SH	CL	11	5	8	D	2
P2	17	SM	B4	14	3	7	D	1
P2	8	SM	B4	14	3	8	D	2
P2	19	SM	B4	14	5	9	D	1
P2	10	SM	B4	14	5	10	D	2
4A	5	WS	A6	14	5	5	D	1
4A	46	WS	A6	14	5	6	D	2
4C	17	WS	A7	14	3	7	D	1
4C	8	WS	A7	14	3	8	D	2
4D	9	WS	LL	14	4	9	D	1
4D	50	WS	LL	14	4	10	D	2
4D	6	WS	A4	14	1	6	D	1
4D	47	WS	A4	14	1	7	D	2
5C	17	LH	LL	14	3	7	D	1
5C	8	LH	LL	14	3	8	D	2
5C	6	LH	A2	14	1	6	D	1
5C	47	LH	A2	14	1	7	D	2
5E	19	SC	LL	14	5	9	D	1
5E	10	SC	LL	14	5	10	D	2
5E	7	SC	B1	14	2	7	D	1
5E	48	SC	B1	14	2	8	D	2
4A	9	HY	A6	15	4	9	D	1
4A	50	HY	A6	15	4	10	D	2
4C	19	HY	A7	15	5	9	D	1
4C	10	HY	A7	15	5	10	D	2
4D	7	MW	A4	15	2	7	D	1
4D	48	MW	A4	15	2	8	D	2
5A	9	MW	A5	15	4	9	D	1
5A	50	MW	A5	15	4	10	D	2
5C	19	MW	A2	15	5	9	D	1
5C	10	MW	A2	15	5	10	D	2
5E	6	WY	B1	15	1	6	D	1
5E	47	WY	B1	15	1	7	D	2
B2	17	KS	B5	17	3	7	D	1
B2	8	KS	B5	17	3	8	D	2
B2	19	KS	B5	17	5	9	D	1
B2	10	KS	B5	17	5	10	D	2
4A	17	FS	A6	17	3	7	D	1
4A	8	FS	A6	17	3	8	D	2
4A	19	FS	A6	17	5	9	D	1
4A	10	FS	A6	17	5	10	D	2
4C	5	FS	A7	17	5	5	D	1
4C	46	FS	A7	17	5	6	D	2
4D	19	YF	A4	17	5	9	D	1
4D	10	YF	A4	17	5	10	D	2
4D	25	YF	A4	17	2	5	D	1
4D	16	YF	A4	17	2	6	D	2
5A	25	KS	A5	17	2	5	D	1
5A	16	KS	A5	17	2	6	D	2
5A	35	KS	A5	17	3	5	D	1
5A	26	KS	A5	17	3	6	D	2
5C	7	KL	A2	17	2	7	D	1
5C	48	KL	A2	17	2	8	D	2

CLASS	SLOT	TCHER	ROOM	SUBJECT	DAY	SESSION	TYPE	TYPE1
5C	9	KL	A1	17	4	9	D	1
5C	50	KL	A1	17	4	10	D	2
5E	35	KL	B5	17	3	5	D	1
5E	26	KL	B5	17	3	6	D	2
4A	16	YK	A6	19	2	6	S	1
4C	18	YK	A7	19	4	8	S	1
5E	9	CF	T1	22	4	9	D	1
5E	50	CF	T1	22	4	10	D	2
C2	17	CF	T2	22	3	7	D	1
C2	8	CF	T2	22	3	8	D	2
6B	27	SC	A4	23	4	7	D	1
6B	18	SC	A4	23	4	8	D	2
C1	29	SL	CL	24	1	9	D	1
C1	20	SL	CL	24	1	10	D	2
C2	39	SL	CL	24	2	9	D	1
C2	30	SL	CL	24	2	10	D	2
C1	17	KL	A1	3	3	7	D	1
C1	8	KL	A1	3	3	8	D	2
C2	19	KL	B3	3	5	9	D	1
C2	10	KL	B3	3	5	10	D	2
C2	25	KL	B3	3	2	5	D	1
C2	16	KL	B3	3	2	6	D	2
B2	29	LW	B5	6	1	9	D	1
B2	20	LW	B5	6	1	10	D	2
B2	39	LW	B5	6	2	9	D	1
B2	30	LW	B5	6	2	10	D	2
6B	6	CY	A3	7	1	6	D	1
6B	47	CY	A3	7	1	7	D	2
5A	17	HT	T1	8	3	7	D	1
5A	8	HT	T1	8	3	8	D	2
4A	27	FC	A6	9	4	7	D	1
4A	18	FC	A6	9	4	8	D	2
4A	29	FC	A6	9	1	9	D	1
4A	20	FC	A6	9	1	10	D	2
4C	37	FC	A6	9	5	7	D	1
4C	28	FC	A6	9	5	8	D	2
4C	39	FC	A6	9	2	9	D	1
4C	30	FC	A6	9	2	10	D	2
4D	27	HT	A5	9	4	7	D	1
4D	18	HT	A5	9	4	8	D	2
4D	29	HT	A4	9	1	9	D	1
4D	20	HT	A4	9	1	10	D	2
5A	19	HT	A5	9	5	9	D	1
5A	10	HT	A5	9	5	10	D	2
5A	39	HT	A5	9	2	9	D	1
5A	30	HT	A5	9	2	10	D	2
5C	27	CY	A1	9	4	7	D	1
5C	18	CY	A1	9	4	8	D	2
5E	17	CY	B1	9	3	7	D	1
5E	8	CY	B1	9	3	8	D	2
5E	37	CY	B1	9	5	7	D	1
5E	28	CY	B1	9	5	8	D	2
4A	26	YF	A6	10	3	6	S	1
4C	26	YH	A7	10	3	6	S	1

CLASS	SLOT	TCHER	ROOM	SUBJECT	DAY	SESSION	TYPE	TYPE1
4C	27	YH	A7	10	4	7	S	1
P2	26	YW	B3	11	3	6	S	1
4A	35	SH	A7	11	3	5	S	1
4A	36	SH	A6	11	4	6	S	1
4C	40	SH	A7	11	3	10	S	1
4D	30	SH	A4	11	2	10	S	1
4D	38	SH	A4	11	1	8	S	1
5C	16	SH	A2	11	2	6	S	1
6B	7	YW	A3	12	2	7	S	1
B2	36	KF	B5	13	4	6	S	1
B2	40	KF	B5	13	3	10	S	1
4A	30	WS	A7	14	2	10	S	1
4C	29	WS	A7	14	1	9	S	1
4D	26	WS	A4	14	3	6	S	1
4D	28	WS	A4	14	5	8	S	1
5C	30	LH	A2	14	2	10	S	1
4A	40	HY	A6	15	3	10	S	1
4C	35	HY	A6	15	3	5	S	1
4C	38	HY	A6	15	1	8	S	1
4D	37	MW	A4	15	5	7	S	1
5A	28	MW	A5	15	5	8	S	1
5E	15	KL	B5	17	1	5	S	1
5E	30	KL	B1	17	2	10	S	1
B2	28	KU	B5	18	5	8	S	1
C1	10	KU	A1	18	5	10	S	1
P2	20	YK	B4	19	1	10	S	1
C1	19	YK	A1	19	5	9	S	1
C1	18	KL	A2	3	4	8	S	1
C1	28	KL	A2	3	5	8	S	1
C1	26	HY	A2	4	3	6	S	1
C1	27	HY	A2	4	4	7	S	1
C2	18	HY	B3	4	4	8	S	1
C2	28	HY	B3	4	5	8	S	1
6B	48	CY	A3	7	2	8	S	1
4A	25	FC	A6	9	2	5	S	1
4C	47	FC	A6	9	1	7	S	1
4D	40	HT	A4	9	3	10	S	1
5E	38	CY	B1	9	1	8	S	1
5E	39	YK	B1	19	2	9	S	1
P2	49	YW	CL	11	3	9	D	1
P2	40	YW	CL	11	3	10	D	2
P2	47	YW	CL	11	1	7	D	1
P2	38	YW	CL	11	1	8	D	2
P2	37	KS	B4	1	5	7	D	1
P2	28	KS	B4	1	5	8	D	2
P2	29	SM	B4	14	1	9	S	1
5B	35	TW	T1	22	3	5	D	1
5B	26	TW	T1	22	3	6	D	2
5E	5	YW	CL	11	5	5	D	1
5E	46	YW	CL	11	5	6	D	2
B2	27	KS	B2	17	4	7	D	1
B2	18	KS	B2	17	4	8	D	2
5A	29	TW	T2	22	1	9	D	1
5A	20	TW	T2	22	1	10	D	2
5A	27	LH	LL	14	4	7	D	1

CLASS	SLOT	TCHER	ROOM	SUBJECT	DAY	SESSION	TYPE	TYPE1
5A	18	LH	LL	14	4	8	D	2
5A	48	LH	A5	14	2	8	S	1
5A	49	YF	LL	10	3	9	D	1
5A	40	YF	LL	10	3	10	D	2
5A	47	MW	A5	15	1	7	S	1
5A	38	HT	A5	9	1	8	S	1
5C	49	CF	T1	22	3	9	D	1
5C	40	CF	T1	22	3	10	D	2
6B	39	SC	A3	23	2	9	D	1
6B	30	SC	A3	23	2	10	D	2
5C	39	LH	A2	14	2	9	S	1
C1	49	LW	A1	6	3	9	D	1
C1	40	LW	A1	6	3	10	D	2
C1	39	CF	T2	22	2	9	D	1
C1	30	CF	T2	22	2	10	D	2
5C	38	MW	A2	15	1	8	S	1
5C	3	CY	A3	9	3	3	D	1
5C	44	CY	A3	9	3	4	D	2
4C	15	SH	CL	11	1	5	D	1
4C	6	SH	CL	11	1	6	D	2
4C	9	TW	T2	22	4	9	D	1
4C	50	TW	T2	22	4	10	D	2
4C	49	SH	A7	11	3	9	S	1
4C	20	WS	A7	14	1	10	S	1
4D	39	YH	A4	10	2	9	S	1
4D	49	YH	A4	10	3	9	S	1
4F	22	YH	A5	10	4	2	S	1
5E	18	YW	B1	11	4	8	S	1
5E	45	SC	B1	14	4	5	S	1
P2	27	WY	B4	25	4	7	D	1
P2	18	WY	B4	25	4	8	D	2
P2	39	WY	B4	25	2	9	S	1
P2	9	YW	B4	11	4	9	D	1
P2	50	YW	B4	11	4	10	D	2
S2	22	TW	B5	21	4	2	S	1
B1	25	KF	A2	13	2	5	S	1
C1	9	YF	A4	10	4	9	D	1
C1	50	YF	A4	10	4	10	D	2
C2	36	KL	B3	3	4	6	S	1
5C	46	CY	A2	9	5	6	S	1
6B	40	WY	A3	16	3	10	S	1
6B	35	SC	LL	23	3	5	D	1
6B	26	SC	LL	23	3	6	D	2
6B	37	WY	A3	16	5	7	D	1
6B	28	WY	A3	16	5	8	D	2
5E	40	SC	B1	14	3	10	S	1
6B	46	YK	A3	19	5	6	S	1
6B	38	WY	A3	16	1	8	S	1
6B	49	CY	A3	7	3	9	S	1
5E	49	WY	B1	15	3	9	S	1
5E	36	WY	B1	15	4	6	S	1
5E	27	YW	B1	11	4	7	S	1
C2	48	CK	B3	6	2	8	S	1
C1	5	CF	T1	22	5	5	D	1
C1	46	CF	T1	22	5	6	D	2
C2	9	CK	B3	6	4	9	D	1

CLASS	SLOT	TCHER	ROOM	SUBJECT	DAY	SESSION	TYPE	TYPE1
C2	50	CK	B3	6	4	10	D	2
C2	29	CF	T1	22	1	9	D	1
C2	20	CF	T1	22	1	10	D	2
C2	37	KL	B3	3	5	7	S	1
C2	27	KU	B3	18	4	7	S	1
B2	50	FS	B5	5	4	10	S	1
B2	9	LW	B5	6	4	9	S	1
C1	48	LW	B2	6	2	8	S	1
4B	15	HY	A7	15	1	5	S	1
C2	49	YK	B4	19	3	9	S	1
C2	40	CK	B3	6	3	10	S	1
4A	47	TW	T2	22	1	7	D	1
4A	38	TW	T2	22	1	8	D	2
4A	49	HY	A6	15	3	9	S	1
4A	39	WS	A7	14	2	9	S	1
B2	49	FS	B5	5	3	9	S	1
B2	26	LW	B4	6	3	6	S	1
B2	48	YK	B4	19	2	8	S	1
B1	5	KU	A2	18	5	5	S	1
B1	45	LW	A2	6	4	5	S	1
B1	34	YK	A2	19	2	4	S	1
S2	45	KU	B4	18	4	5	S	1
P2	4	KS	B5	1	4	4	D	1
P2	45	KS	B5	1	4	5	D	2
P2	16	YW	B4	11	2	6	S	1
P2	30	WY	B4	25	2	10	S	1
4D	35	YK	A2	19	3	5	S	1
4A	13	WS	LL	14	4	3	D	1
4A	4	WS	LL	14	4	4	D	2
4B	21	HY	A7	15	3	1	D	1
4B	12	HY	A7	15	3	2	D	2
4B	37	CF	T2	22	5	7	D	1
4B	28	CF	T2	22	5	8	D	2
4B	3	YF	A7	10	3	3	S	1
S2	3	CF	T1	22	3	3	D	1
S2	44	CF	T1	22	3	4	D	2
4B	44	FS	A7	17	3	4	S	1
4B	24	FS	A7	17	1	4	S	1
4C	7	FS	A7	17	2	7	D	1
4C	48	FS	A7	17	2	8	D	2
4C	45	WS	LL	14	4	5	D	1
4C	36	WS	LL	14	4	6	D	2
4B	25	YK	A7	19	2	5	S	1
4B	16	FC	A7	9	2	6	S	1
C1	47	KL	A1	3	1	7	D	1
C1	38	KL	A1	3	1	8	D	2
B2	47	FS	B5	5	1	7	D	1
B2	38	FS	B5	5	1	8	D	2
B2	46	KF	B5	13	5	6	D	1
B2	37	KF	B5	13	5	7	D	2
4B	43	WS	LL	14	2	3	D	1
4B	34	WS	LL	14	2	4	D	2
4B	19	WS	A3	14	5	9	S	1
4B	10	WS	A3	14	5	10	S	1
4A	28	YF	A1	10	5	8	S	1

CLASS	SLOT	TCHER	ROOM	SUBJECT	DAY	SESSION	TYPE	TYPE1
4B	13	FC	A7	9	4	3	D	1
4B	4	FC	A7	9	4	4	D	2
4B	23	SH	CL	11	5	3	D	1
4B	14	SH	CL	11	5	4	D	2
5D	13	MW	B1	15	4	3	S	1
5D	4	SH	B1	11	4	4	S	1
5A	7	LH	A5	14	2	7	S	1
5C	45	SH	A1	11	4	5	S	1
4G	22	HY	B4	15	4	2	S	1
4G	13	YK	B4	19	4	3	S	1
4E	15	SL	A3	11	1	5	S	1
4F	2	YH	A5	10	2	2	S	1
4F	34	SL	A5	11	2	4	S	1
4F	33	FC	A5	9	1	3	S	1
5A	36	YK	A5	19	4	6	S	1
4F	45	HY	A5	15	4	5	S	1
4F	4	KK	A5	14	4	4	S	1
4F	24	KK	A5	14	1	4	D	1
4F	15	KK	A5	14	1	5	D	2
4E	45	MW	A3	15	4	5	S	1
4D	46	MW	A4	15	5	6	S	1
4D	45	YH	T2	22	4	5	D	1
4D	36	YH	T2	22	4	6	D	2
C1	36	LW	A2	6	4	6	S	1
5C	36	MW	A1	15	4	6	S	1
5A	46	LH	A5	14	5	6	D	1
5A	37	LH	A5	14	5	7	D	2
4E	23	YF	A3	10	5	3	S	1
4E	34	FC	A3	9	2	4	S	1
GA	24	KF	B3	13	1	4	S	1
4E	24	YK	A3	19	1	4	S	1
GA	33	YK	B3	19	1	3	S	1
4E	33	KK	A3	14	1	3	S	1
4E	14	SL	A3	11	5	4	S	1
4E	5	YH	T2	22	5	5	D	1
4E	46	YH	T2	22	5	6	D	2
4E	13	YF	A3	17	4	3	D	1
4E	4	YF	A3	17	4	4	D	2
4E	25	KK	A3	14	2	5	D	1
4E	16	KK	A3	14	2	6	D	2
5C	26	YK	A3	19	3	6	S	1
6A	13	CY	A4	7	4	3	D	1
6A	4	CY	A4	7	4	4	D	2
6A	23	WY	A4	16	5	3	D	1
6A	14	WY	A4	16	5	4	D	2
6A	45	YK	A4	19	4	5	S	1
6A	5	CY	A4	7	5	5	S	1
6A	43	CY	A4	7	2	3	S	1
6A	34	WY	A4	16	2	4	S	1

APPENDIX D

Sample Listing of Timetables
(Class, Instructor, Location)

APPENDIX D

Class Time Table 09/13/94
00:33:11

Class : [6A] FORM 6A

	Day 1	Day 2	Day 3	Day 4	Day 5
1	ACCTN CLS A_2D	COM_U YYW COM_L	U_ENG LSH A_2D	U_ENG LSH LAN_L	ACCTN CLS A_2D
2	ACCTN CLS A_2D	COM_U YYW COM_L	U_ENG LSH A_2D	U_ENG LSH LAN_L	ACCTN CLS A_2D
	B R E A K				
3	U_ENG LSH A_2D	CHILC HCY A_2D	ACCTN CLS A_2D	CHILC HCY A_2D	MSTAT WHY A_2D
4	U_ENG LSH A_2D	MSTAT WHY A_2D	ACCTN CLS A_2D	CHILC HCY A_2D	MSTAT WHY A_2D
5	COM_U YYW A_2D		MSTAT WHY A_2D	RS YKL A_2D	CHILC HCY A_2D
6					
7					
8					
	B R E A K				
9					
10					

Class Time Table

09/13/94
00:33:14

Class : [5E] FORM 5E

	Day 1	Day 2	Day 3	Day 4	Day 5
1					
2					
	B R E A K				
3					
4					
5	P_ACC NKL B_7A		P_ACC NKL B_7A	ENG_L MSC B_4A	COM_S YYW COM_L
6	MATHS WHY B_4A		P_ACC NKL B_7A	MATHS WHY B_4A	COM_S YYW COM_L
7	MATHS WHY B_4A	ENG_L MSC B_4A	CHI_L HCY B_4A	COM_S YYW B_4A	CHI_L HCY B_4A
8	CHI_L HCY B_4A	ENG_L MSC B_4A	CHI_L HCY B_4A	COM_S YYW B_4A	CHI_L HCY B_4A
	B R E A K				
9	CO_CO WYH A_2E	RS YKL B_4A	MATHS WHY B_4A	TYPE CYF TYP_A	ENG_L MSC LAN_L
10	CO_CO WYH A_2E	P_ACC NKL B_4A	ENG_L MSC B_4A	TYPE CYF TYP_A	ENG_L MSC LAN_L

Instructor Time Table

09/13/94
00:34:02

Instructor : [MW] MS. LO M. W.

	Day 1	Day 2	Day 3	Day 4	Day 5
1			MATHS F4E A_2C		
2		MATHS F5D B_4A	MATHS F4E A_2C		
	B R E A K				
3	MATHS F5B A_6A	MATHS F4E A_2C	MATHS F5D B_4A	MATHS F5D B_4A	
4	MATHS F5B A_6A	MATHS F5B A_6A	MATHS F5D B_4A		MATHS F5B A_6A
5				MATHS F4E A_2C	
6				MATHS F5C A_2A	MATHS F4D A_2D
7	MATHS F5A A_2E	MATHS F4D A_2D			MATHS F4D A_2D
8	MATHS F5C A_2B	MATHS F4D A_2D			MATHS F5A A_2E
	B R E A K				
9				MATHS F5A A_2E	MATHS F5C A_2B
10				MATHS F5A A_2E	MATHS F5C A_2B

Instructor Time Table 09/13/94
00:34:13

Instructor : [WY] MR. WONG H. Y.

	Day 1	Day 2	Day 3	Day 4	Day 5
1	ASTAT C_IIB B_5B			ASTAT C_IIB B_5B	ASTAT C_IIA B_5C
2	ASTAT C_IIB B_5B			ASTAT C_IIB B_5B	ASTAT C_IIA B_5C
	B R E A K				
3			ASTAT C_IIA B_5C		MSTAT F6A A_2D
4		MSTAT F6A A_2D	ASTAT C_IIA B_5C		MSTAT F6A A_2D
5			MSTAT F6A A_2D		
6	MATHS F5E B_4A			MATHS F5E B_4A	
7	MATHS F5E B_4A			ASTAT BCOMP B_5D	MSTAT F6B A_2C
8	MSTAT F6B A_2C			ASTAT BCOMP B_5D	MSTAT F6B A_2C
	B R E A K				
9		ASTAT BCOMP B_5D	MATHS F5E B_4A		
10		ASTAT BCOMP B_5D	MSTAT F6B A_2C		

Classroom Time Table

09/13/94
00:34:43

Location : [A7] RM 6B, BLOCK A

	Day 1	Day 2	Day 3	Day 4	Day 5
1	ENG_L SWS F4B	P_ACC WFS F4B	MATHS CHY F4B	COM_S HSH F4B	CHI_L WFC F4B
2	ENG_L SWS F4B	P_ACC WFS F4B	MATHS CHY F4B	COM_S HSH F4B	CHI_L WFC F4B
	B R E A K				
3	COMM CYF F4B		COMM CYF F4B	CHI_L WFC F4B	
4	P_ACC WFS F4B		P_ACC WFS F4B	CHI_L WFC F4B	
5	MATHS CHY F4B	RS YKL F4B	COM_S HSH F4A		P_ACC WFS F4C
6	MATHS CHY F4B	CHI_L WFC F4B	COMM WYH F4C		P_ACC WFS F4C
7		P_ACC WFS F4C	ENG_L SWS F4C	COMM WYH F4C	COMM WYH F5D
8		P_ACC WFS F4C	ENG_L SWS F4C	RS YKL F4C	COMM WYH F5D
	B R E A K				
9	ENG_L SWS F4C	ENG_L SWS F4A	COM_S HSH F4C		MATHS CHY F4C
10	ENG_L SWS F4C	ENG_L SWS F4A	COM_S HSH F4C		MATHS CHY F4C

Classroom Time Table

09/13/94
00:35:00

Location : [CL] COMPUTER ROOM

	Day 1	Day 2	Day 3	Day 4	Day 5
1	COM_S CSL F4G	COM_U YYW F6A	W_PRO YYW SEC_A	W_PRO CSL C_IIA	W_PRO CSL C_A1
2	COM_S CSL F4G	COM_U YYW F6A	W_PRO YYW SEC_A	W_PRO CSL C_IIA	W_PRO CSL C_A1
	B R E A K				
3	W_PRO CSL C_IIB	COM_S HSH F5D	COM_S CSL F4E	W_PRO CSL SEC_B	COM_S HSH F4B
4	W_PRO CSL C_IIB	COM_S HSH F5D	COM_S CSL F4E	W_PRO CSL SEC_B	COM_S HSH F4B
5	COM_S HSH F4C	W_PRO CHY C_A2	COM_S CSL F4F	COM_S YYW F5B	COM_S YYW F5E
6	COM_S HSH F4C	W_PRO CHY C_A2	COM_S CSL F4F	COM_S YYW F5B	COM_S YYW F5E
7	COM_S YYW BCOMP	COM_S HSH F4A	COM_S HSH F4D		COM_S HSH F5C
8	COM_S YYW BCOMP	COM_S HSH F4A	COM_S HSH F4D		COM_S HSH F5C
	B R E A K				
9	W_PRO CSL C_B1	W_PRO CSL C_B1	COM_S YYW BCOMP		COM_U YYW F6B
10	W_PRO CSL C_B1	W_PRO CSL C_B1	COM_S YYW BCOMP		COM_U YYW F6B

APPENDIX E

Listing of (PRG) Programs

E1: TTMAIN.PRG

E2: TTASSIGN.PRG

E3: TTINQ.PRG

E4: TTPRINT.PRG

E5: TTSETTNG.PRG

E6: TTUPDATE.PRG

APPENDIX E1

**** PROGRAM NAME : TTMAIN.PRG**
**** LAST CHANGED : JULY 23, 1994**
**** DESCRIPTION : TIME TABLE MENU MODULE**

```
RESTORE FROM TTSETTNG

IF (.NOT. FILE("TTMENU.BAT"))
  W_STATUS=MEMOWRIT("TTMENU.BAT",D_HIDE+"TTMAIN.EXE"+CHR(13)+CHR(10)+  ;
                D_HIDE+ "TTNXTPGM.BAT")
  W_STATUS=MEMOWRIT("TTNXTPGM.BAT"," ")
  ? CHR(7)+"Please use 'TTMENU' to access the 'TIME TABLE SYSTEM'"
  RETURN
ENDIF
*-------------------------------------------------

set talk off
set bell off
set scoreboard off
set exact on
set confirm on

W_MSG=SPACE(80)

DO WHILE .T.
  G_OPTION=0

  SET DEVICE TO SCREEN
  @ 0,0 SAY " "
  IF (W_MSG<>SPACE(80))
    ? CHR(7)
  ENDIF
  SET CURSOR ON
  CLEAR
  SET COLOR TO W/N
  @ 0,72 SAY DATE()
  @ 1,72 SAY TIME()
  SET COLOR TO W/N
  @ 3,0 TO 18,78 DOUBLE
  SET COLOR TO W+/N
  @ 4,15 SAY "T I M E T A B L I N G  S Y S T E M  M E N U"
  SET COLOR TO W/N
  @ 5,1 TO 5,77
  SET COLOR TO W+/N
  @ 6,15 SAY "OPERATIONS"
  SET COLOR TO W/N
  @ 8,15 SAY "1. AUTO ASSIGNMENT "
  @ 9,15 SAY "2. INQUIRY / MANUAL ADJUSTMENT "
  @ 10,15 SAY "3. OUTPUT TO FILE / PRINTER "
  SET COLOR TO W/N
  @ 12,1 TO 12,77
  SET COLOR TO W+/N
  @ 13,15 SAY "MAINTENANCE"
  SET COLOR TO W/N
```

```
   @ 15,15 SAY "51. SET UP / CHANGE SETTING"
   @ 16,15 SAY "52. UPDATE  FILE"
   @19, 4 SAY "Enter Option : " GET G_OPTION PICTURE '@Z 99'
   @23, 0 SAY "ESC=CANCEL"
   @24, 0 CLEAR TO 24,80
   @24, 0 SAY W_MSG

   READ

   W_READKEY=READKEY(0)
   W_MSG=SPACE(80)

 * ESC KEY PRESSED DURING 'READ' (268=FIELD UPDATED; 12=FIELD NOT UPDATED)
   IF (W_READKEY=12) .OR. (W_READKEY=268)

     W_STATUS=MEMOWRIT("TTNXTPGM.BAT",D_HIDE+"DEL TTNXTPGM.BAT")
     CLEAR ALL
     CLEAR
     RETURN

   ENDIF  && W_READKEY=12/268 == ESC

   W_OK=.T.
   DO CASE
     CASE G_OPTION=1   && AUTO ASSIGNMENT
        W_PGM="TTASSIGN"
     CASE G_OPTION=2   && INQUIRY / MANUAL ADJUSTMENT
        W_PGM="TTINQ"
     CASE G_OPTION=3   && PRINT
        W_PGM="TTPRINT"
      CASE G_OPTION=51  && MAINTENANCE-CHANGE SETTINGS
        W_PGM="TTSETTNG"
     CASE G_OPTION=52  && UPDATE FILE
        W_PGM="TTUPDATE"

     CASE G_OPTION=0
        LOOP
     OTHERWISE
        W_MSG="Option not valid"

        LOOP && ERROR, LOOP AGAIN
     ENDCASE
     && PREPARE NEXT PGM TO CALL
     W_STATUS=MEMOWRIT("TTNXTPGM.BAT",D_HIDE+W_PGM+".EXE"+CHR(13)+CHR(10)+   ;
                    D_HIDE+  "TTMENU.BAT")

     QUIT   && QUIT TO DOS AND LET PROGRAM IN 'TTNXTPGM.BAT' TO BE EXECUTED

ENDDO  && .T.

CLOSE ALL
SET CURSOR ON
RETURN  && MAIN LINE
```

APPENDIX E2

**** PROGRAM NAME : TTASSIGN.PRG**
**** LAST CHANGED : August 16, 1994**
**** DESCRIPTION : TIME TABLE - AUTO ASSIGNMENT MODULE**
******

```
set cursor off
set talk off
set bell off
set scoreboard off
set exact on
set confirm on

L_FKEY=0  && 1=F1, 2=F2, ..., 11=F11, ..., 30=ESC, 31=PAGEDOWN, 32=PAGEUP
CALL SETKEY  && RETURN: L_FKEY   EXTERNAL PROGRAM TO HANDLE FUNCTION KEYS

DD_SPLIT="N"
DD_MAX = 0
DD_MFLAG=.F.
RESTORE FROM TTSETNG ADDITIVE

W_OK1=.T.
DO PREASS
IF (W_OK1=.F.)
 CLEAR ALL
 RETURN
ENDIF
@ 3, 0 CLEAR TO 19,79
@ 23,0 SAY "Time Table Assignment Is Under Progress ... "
DO INITIAL

SELECT 1
USE ASSIGNPF INDEX ASSIGNL1, ASSIGNL2, ASSIGNL3
REINDEX
SELECT 2
USE CLSUBJPF INDEX CLSUBJL1
REPLACE ALL FINISHED WITH 0  && RESET COUNTER TO ZERO
SELECT 3
USE ALTRMPF INDEX ALTRML1
SELECT 4
USE TDSPF
SELECT 5
USE CLSLOTPF INDEX CLSLOTL1
SELECT 6
USE DAYSESPF
** ---------------------------------------

SELECT CLSUBJPF && CLASS SUBJECT RECORD
GO TOP
W_COUNT=0
W_RECCOUNT=RECCOUNT()
DO WHILE (.NOT. EOF())
  W_COUNT=W_COUNT+1
```

```
  @24,0 SAY ALLTRIM(STR(W_COUNT))+'/'+ALLTRIM(STR(W_RECCOUNT))+ ;
        " Class="+CLASS+"  Subject="+SUBJECT+"  Teacher="+TCHER+ ;
        " Room="+ROOM+"  Classtype="+CLASSTYPE
  W_TYPE = LEFT(CLASSTYPE,1)  && GENERAL CLASS TYPE (Triple/Double/Single)
  W_REPEAT = 0
  DO WHILE (W_REPEAT < REPEAT)
    W_REPEAT = W_REPEAT + 1
    W_OK = .F.
    W_CLASS = CLASS
    W_TCHER = TCHER
    W_ROOM  = ROOM
    W_CLASSTYPE = CLASSTYPE
    W_SUBJECT = SUBJECT
    W_FIRST = .T. && GET FIRST SLOT FOR THE SPECIFIED CLASS+TYPE
    W_ENDSLOT = .F.
    W_SLOT1 = ' '
    W_SLOT2 = ' '
    W_SLOT3 = ' '

    DO WHILE ((.NOT. W_OK) .AND. (.NOT. W_ENDSLOT))
      DO NEXTSLOT WITH W_CLASS, W_TYPE, W_FIRST, W_ENDSLOT, W_SLOT1, W_SLOT2, W_SLOT3
       IF (.NOT. W_ENDSLOT)
       DO CHECKSLOT WITH W_CLASS, W_TCHER, W_ROOM, W_SUBJECT, W_TYPE, ;
                W_SLOT1, W_SLOT2, W_SLOT3, W_OK
       IF (W_OK)
         DO ASSIGNSLOT WITH W_CLASS, W_TCHER, W_ROOM, W_SUBJECT, ;
                  W_TYPE, W_SLOT1, W_SLOT2, W_SLOT3
         REPLACE CLSUBJPF->FINISHED WITH (CLSUBJPF->FINISHED + 1)
       ENDIF && W_OK
       ENDIF && (.NOT. W_ENDSLOT)
    ENDDO  && DO WHILE ((.NOT. W_OK) .AND. (.NOT. W_ENDSLOT))
  SELECT CLSUBJPF
  ENDDO  && DO WHILE (W_REPEAT < REPEAT)
SELECT CLSUBJPF
SKIP
ENDDO  && DO WHILE (.NOT. EOF())
CLOSE ALL
SET CURSOR ON
RETURN   && MAIN LINE
** ----------------------------------------

PROCEDURE INITIAL
** INITIALIZE CONTENT OF 'CLSLOTPF'
** (CLSLOTPF CONTAINS A RECORD OF STARTING SLOT FOR EACH CLASS AND CLASSTYPE)

SELECT 1
USE TDSPF
SELECT 2
USE CLSLOTPF
ZAP
COPY STRUCTURE TO TEMP
SELECT 3
USE TEMP

SELECT TDSPF
GO TOP
DO WHILE (.NOT. EOF())
  SELECT TEMP
  APPEND BLANK
  REPLACE TYPE WITH TDSPF->TYPE, SLOT WITH TDSPF->SLOT1
  SELECT TDSPF
  SKIP
ENDDO  && .NOT. EOF()
SELECT TEMP
USE
```

```
SELECT 1
USE CLASSPF
GO TOP
DO WHILE ((.NOT. EOF()) .AND. (CLASS<>' '))
  SELECT CLSLOTPF
  APPEND FROM TEMP
  REPLACE ALL CLASS WITH CLASSPF->CLASS FOR CLASS=' '
  SELECT CLASSPF
  SKIP
ENDDO  && .NOT. EOF() .AND. CLASS<>' '
SELECT CLSLOTPF
SET INDEX TO CLSLOTL1
REINDEX

CLOSE DATABASES
RUN DEL TEMP.DBF
RETURN  && INITIAL
** ----------------------------------------

PROCEDURE NEXTSLOT
PARAMETERS P_CLASS, P_TYPE, P_FIRST, P_ENDSLOT, P_SLOT1, P_SLOT2, P_SLOT3

** GET NEXT SLOT# AVAILABLE FOR THE SPECIFIED CLASS AND TYPE
** 'CLSLOTPF' SHOULD CONTAINS RECORDS FOR EACH UN-ALLOCATED SLOTS FOR EACH CLASS & TYPE

SELECT CLSLOTPF
SET DELETED ON
IF (P_FIRST)
  GO TOP
  SEEK P_CLASS+P_TYPE
ELSE
  SKIP
ENDIF && P_FIRST
SET DELETED OFF
P_FIRST = .F. && DO NOT GET THE FIRST RECORD ON NEXT ACCESS

IF EOF() .OR. (CLASS<>P_CLASS) .OR. (TYPE<>P_TYPE)
  P_ENDSLOT = .T.
  P_SLOT1 = ' '
  P_SLOT2 = ' '
   P_SLOT3 = ' '
  GO BOTTOM
ELSE
  P_ENDSLOT = .F.
  P_SLOT1 = SLOT
  SELECT TDSPF
  LOCATE FOR TYPE=P_TYPE .AND. SLOT1=P_SLOT1
  IF (FOUND())
    P_SLOT2 = SLOT2
    P_SLOT3 = SLOT3
  ELSE
    P_SLOT2 = ' '
    P_SLOT3 = ' '
  ENDIF && FOUND()
ENDIF  && (.NOT. FOUND()) .OR. (CLASS<>P_CLASS) .OR. (TYPE<>P_TYPE)
RETURN  && NEXTSLOT
** ----------------------------------------

PROCEDURE CHECKSLOT
PARAMETERS P_CLASS, P_TCHER, P_ROOM, P_SUBJECT, P_TYPE, ;
        P_SLOT1, P_SLOT2, P_SLOT3, P_OK

** TO CHECK THE AVAILABLITY OF THE GIVEN SLOTS FOR THE GIVEN CLASS+TEACHER+ROOM
P_OK = .T.
```

```
** CHECK AVAILABLITY OF A SINGLE SLOT
DO CHECKSLOT2 WITH P_CLASS, P_TCHER, P_ROOM, P_SLOT1, P_OK

IF (P_OK) .AND. (VAL(P_SLOT2)<>0)
  DO CHECKSLOT2 WITH P_CLASS, P_TCHER, P_ROOM, P_SLOT2, P_OK

  IF (P_OK) .AND. (VAL(P_SLOT3)<>0)
    DO CHECKSLOT2 WITH P_CLASS, P_TCHER, P_ROOM, P_SLOT3, P_OK

  ENDIF && P_OK .AND. VAL(P_SLOT3)<>0
ENDIF && P_OK .AND. VAL(P_SLOT2)<>0

** WHEN ASSIGNING DOUBLE, OR TRIPLE CLASS BLOCKS
** CHECK CONSTRAINT FOR SAME SUBJECT IN THE SAME DAY
** SHOULD CONTAIN NO SINGLE, DOUBLE OR TRIPLE CLASS BLOCK OF SAME SUBJECT
** SHOULD NOT EXIST THE MAXIMUM WORKLOAD

If W_OK
 DO CASE
   CASE (P_TYPE='T')
      SELECT DAYSESPF
      LOCATE FOR SLOT=P_SLOT1
      SELECT ASSIGNPF

      W_COUNT=0
      COUNT TO W_COUNT FOR CLASS=P_CLASS .AND. SUBJECT=P_SUBJECT .AND. ;
          DAY=DAYSESPF->DAY
      W_OK=(W_COUNT<1)

      IF (W_OK) .AND. (W_COUNT=0)
        LOCATE FOR CLASS=P_CLASS .AND. SUBJECT=P_SUBJECT .AND.  ;
            DAY=DAYSESPF->DAY
        W_OK=(.NOT. FOUND())
      ENDIF && (W_OK) AND W_COUNT=0

      IF ((W_OK) .AND. (.NOT. DD_MFLAG))
      W_COUNT=0
      COUNT TO W_COUNT FOR TCHER=P_TCHER .AND. DAY=DAYSESPF->DAY
      W_OK=(W_COUNT<=(DD_MAX-3))
      ENDIF && (W_OK)

  CASE (P_TYPE='D')
      SELECT DAYSESPF
      LOCATE FOR SLOT=P_SLOT1
      SELECT ASSIGNPF

      W_COUNT=0
      COUNT TO W_COUNT FOR CLASS=P_CLASS .AND. SUBJECT=P_SUBJECT .AND. ;
          DAY=DAYSESPF->DAY
      W_OK=(W_COUNT<1)


      IF (W_OK) .AND. (W_COUNT=0)
       LOCATE FOR CLASS=P_CLASS .AND. SUBJECT=P_SUBJECT .AND. ;
       DAY=DAYSESPF->DAY
       W_OK=(.NOT. FOUND())
      ENDIF && (W_OK) AND W_COUNT=0

      IF ((W_OK).AND.(.NOT. DD_MFLAG))
      W_COUNT=0
      COUNT TO W_COUNT FOR TCHER=P_TCHER .AND. DAY=DAYSESPF->DAY
      W_OK=(W_COUNT<=(DD_MAX-2))
      ENDIF && (W_OK)

** CHECK CONSTRAINT FOR SAME SUBJECT IN THE SAME DAY
```

```
** SHOULD NOT EXIST THE MAXIMUM WORKLOAD
** IT ALLOWS A SINGLE SESSION (PERIOD) REPEATS ONCE (UNDER REQUEST)
** PROVIDED THAT THERE IS NO DOUBLE OR TRIPLE BLOCK IN THE SAME DAY

   CASE (P_TYPE='S')
      SELECT DAYSESPF
      LOCATE FOR SLOT=P_SLOT1
      SELECT ASSIGNPF

      W_COUNT=0
      COUNT TO W_COUNT FOR CLASS=P_CLASS .AND. SUBJECT=P_SUBJECT .AND. ;
         DAY=DAYSESPF->DAY .AND. TYPE='S'

      IF (DD_SPLIT = 'Y')
        W_OK=(W_COUNT<=1)
        IF (W_OK) .AND. (W_COUNT<=1)
          LOCATE FOR CLASS=P_CLASS .AND. SUBJECT=P_SUBJECT .AND. ;
            DAY=DAYSESPF->DAY .AND. (TYPE='D' .OR. TYPE='T')
          W_OK=(.NOT. FOUND())
        ENDIF && (W_OK) AND W_COUNT=1
      ELSE
        W_OK=(W_COUNT<1)
        IF (W_OK) .AND. (W_COUNT=0)
          LOCATE FOR CLASS=P_CLASS .AND. SUBJECT=P_SUBJECT .AND. ;
            DAY=DAYSESPF->DAY .AND. (TYPE='D' .OR. TYPE='T')
          W_OK=(.NOT. FOUND())
        ENDIF && (W_OK) AND W_COUNT=0
      ENDIF

      IF ((W_OK) .AND. (.NOT. DD_MFLAG))
      W_COUNT=0
      COUNT TO W_COUNT FOR TCHER=P_TCHER .AND. DAY=DAYSESPF->DAY
      W_OK=(W_COUNT<=(DD_MAX-1))
      ENDIF && (W_OK)

 ENDCASE
ENDIF && W_OK
RETURN  && CHECKSLOT
** ----------------------------------------

PROCEDURE CHECKSLOT2
PARAMETERS P_CLASS, P_TCHER, P_ROOM, P_SLOTW2, P_OKW2

** TO CHECK THE AVAILABLITY OF A GIVEN SLOT FOR THE GIVEN CLASS+TEACHER+ROOM

P_OKW2=.T.
SELECT ASSIGNPF
SET ORDER TO 1
SEEK P_CLASS+P_SLOTW2 && CHECK AVAILABLITY OF CLASS IN THE SLOT

IF FOUND()
  P_OKW2=.F.
ELSE
  SET ORDER TO 2
  SEEK P_TCHER+P_SLOTW2 && CHECK AVAILABLITY OF TEACHER IN THE SLOT

  IF FOUND()
    P_OKW2=.F.
  ELSE
    IF (P_ROOM<>' ')  && CHECK AVAILABLITY OF SPECIAL CLASS ROOM IN THE SLOT
      SET ORDER TO 3
      SEEK P_ROOM+P_SLOTW2

      IF FOUND()
        P_OKW2=.F.
```

```
        SELECT ALTRMPF  && CHECK FOR ALTERNATE ROOM AVAILABLE
        GO TOP
        DO WHILE (.NOT. EOF()) .AND. (.NOT. P_OKW2)
          IF (ALTRMPF->ROOM=P_ROOM)
          SELECT ASSIGNPF
          SEEK ALTRMPF->ALTRM+P_SLOTW2
           IF (.NOT. FOUND())
            P_OKW2=.T.
            P_ROOM=ALTRMPF->ALTRM
           ELSE
            P_OKW2=.F.
            SELECT ALTRMPF
            SKIP
           ENDIF
          ELSE
            SELECT ALTRMPF
            SKIP
          ENDIF  && .NOT. FOUND()
        ENDDO  && (.NOT. EOF()) .AND. (.NOT. P_OKW2)
      ENDIF  && FOUND()
    ENDIF && P_ROOM<>' '
  ENDIF && FOUND()
ENDIF && FOUND()
RETURN  && CHECKSLOT2
** ----------------------------------------

PROCEDURE ASSIGNSLOT
PARAMETERS P_CLASS, P_TCHER, P_ROOM, P_SUBJECT, P_TYPE, P_SLOT1, P_SLOT2, P_SLOT3

** TO DO THE ASSIGNMENT OF SUBJECT TO THE GIVEN CLASS, TEACHER, ROOM AND SLOTS

IF (P_ROOM=' ')
  P_ROOM=P_CLASS
ENDIF  && P_ROOM=' '
SELECT DAYSESPF
LOCATE FOR SLOT=P_SLOT1

SELECT ASSIGNPF
APPEND BLANK
REPLACE SLOT WITH P_SLOT1, CLASS WITH P_CLASS, TCHER WITH P_TCHER, ;
     ROOM WITH P_ROOM, SUBJECT WITH P_SUBJECT, TYPE WITH P_TYPE, ;
     TYPE1 WITH '1', DAY WITH DAYSESPF->DAY, SESSION WITH DAYSESPF->SESSION

IF (VAL(P_SLOT2)<>0)
  SELECT DAYSESPF
  LOCATE FOR SLOT=P_SLOT2

  SELECT ASSIGNPF
  APPEND BLANK
  REPLACE SLOT WITH P_SLOT2, CLASS WITH P_CLASS, TCHER WITH P_TCHER, ;
       ROOM WITH P_ROOM, SUBJECT WITH P_SUBJECT, TYPE WITH P_TYPE, ;
       TYPE1 WITH '2', DAY WITH DAYSESPF->DAY, SESSION WITH DAYSESPF->SESSION
ENDIF  && P_SLOT2<>' '

IF (VAL(P_SLOT3)<>0)
  SELECT DAYSESPF
  LOCATE FOR SLOT=P_SLOT3
  SELECT ASSIGNPF
  APPEND BLANK
  REPLACE SLOT WITH P_SLOT3, CLASS WITH P_CLASS, TCHER WITH P_TCHER, ;

       ROOM WITH P_ROOM, SUBJECT WITH P_SUBJECT, TYPE WITH P_TYPE, ;
       TYPE1 WITH '3', DAY WITH DAYSESPF->DAY, SESSION WITH DAYSESPF->SESSION
ENDIF  && P_SLOT3<>' '
```

```
SELECT CLSLOTPF  && DELETE SLOTS WHICH HAS BEEN ASSIGNED WITH SUBJECT
DELETE ALL FOR (CLASS=P_CLASS) .AND. ;
        ((SLOT=P_SLOT1) .OR. (SLOT=P_SLOT2) .OR. (SLOT=P_SLOT3))
RETURN  && ASSIGNSLOT
** ----------------------------------------

PROCEDURE PREASS
SET DEVICE TO SCREEN
SELECT 1
USE ASSIGNPF
ZAP
CLOSE DATABASES
IF FILE('PREASSPF.DBF')
 STORE ' ' TO W_OPT
 SET CURSOR ON
 @19,0 CLEAR TO 21,79
  IF (DD_FLAG=.T.)
   @20,4 SAY "Changes Made In Settings, Update The Preassignment File."
  ENDIF
  @21,4 SAY "Combine The Preassignment File Now (Y/N) : "  ;
  GET W_OPT PICTURE '@Z N'
  READ
  IF (L_FKEY=30) .OR. (UPPER(W_OPT)<>'N' .AND. UPPER(W_OPT)<>'Y')
   W_OK1=.F.
   RETURN
  ENDIF  && L_FKEY=30 (ESC) OR OTHER KEYS TO ABORT
  SET CURSOR OFF
  IF UPPER(W_OPT)='Y'
   @22,0 CLEAR TO 22,79
   SELECT 2
   USE ASSIGNPF
   SELECT 1
   USE PREASSPF
   W_OK2=(TYPE("DAY")='C')
   W_OK3=(TYPE("SESSION")='C')
   GO TOP
   DO WHILE (.NOT. EOF())
    @22,4 SAY "Preassignment Data Processing, Please Wait...."
    SELECT 2
    APPEND BLANK
     IF (W_OK2=.T.)
      T_DAY=VAL(A->DAY)
     ELSE
      T_DAY=(A->DAY)
     ENDIF
     IF (W_OK3=.T.)
      T_SESSION=VAL(A->SESSION)
     ELSE
      T_SESSION=(A->SESSION)
     ENDIF
     REPLACE CLASS WITH TRIM(A->CLASS), SLOT WITH TRIM(A->SLOT), ;
          TCHER WITH TRIM(A->TCHER), ROOM WITH TRIM(A->ROOM)
     REPLACE SUBJECT WITH TRIM(A->SUBJECT), DAY WITH TRIM(STR(T_DAY,2)), ;
          SESSION WITH TRIM(STR(T_SESSION,2)), TYPE WITH TRIM(A->TYPE), ;
          TYPE1 WITH TRIM(A->TYPE1)
    SELECT 1
    SKIP
   ENDDO
  CLOSE DATABASES
 ENDIF
ENDIF
@20,0 CLEAR TO 24,79
RETURN  && END PREASSIGNMENT
```

APPENDIX E3

**** PROGRAM NAME : TTINQ.PRG**
**** LAST CHANGED : SEPTEMBER 15, 1994**
**** DESCRIPTION : TIME TABLE - INQUIRY, SWITCH SESSION & MANUAL ADJUSTMENT MODULE**
******

```
set talk off
set bell off
set scoreboard off
set exact on
set confirm on

RESTORE FROM TTSETTNG ADDITIVE
W_SPACE=SPACE(9*D_DAY) && D_DAY: # OF DAYS IN THE TIMETABLE
W_LASTSESS=STR(D_SESSION+1,2)
W_LINE1=W_SPACE
W_RECESS=REPLICATE('-',(9*D_DAY))
W_DTITLE=""
W_DAY=0
DO WHILE (W_DAY<D_DAY)
  W_DAY=W_DAY+1
  W_DTITLE=W_DTITLE+" Day "+STR(W_DAY,2)+"  "
ENDDO  && DO WHILE (W_DAY<D_DAY)
W_DASH=REPLICATE("-",9*D_DAY)

L_FKEY=0
CALL SETKEY && RETURN: L_FKEY;0=ENTER,1-20=F1-F20,30=ESC,31=PAGEUP,32=PAGEDOWN

* USE TEMP FILE IF EXIST
IF FILE('ASSTEMP.DBF')
  W_ASSPF='ASSTEMP'
ELSE
  W_ASSPF='ASSIGNPF'
ENDIF
IF FILE('CLSTEMP.DBF')
  W_CLSPF='CLSTEMP'
  W_CLSL1='CLSTEMP1'
ELSE
  W_CLSPF='CLSUBJPF'
  W_CLSL1='CLSUBJL1'
ENDIF
W_MSG=SPACE(80)
ROW=0
G_CLASS="  "
G_TCHER="  "
G_ROOM="  "
W_FILTER1=' '
W_FILTER2=' '
W_FILTER3=' '
W_OK = .F.
CLEAR
DO WHILE .T.
  CLOSE DATABASES
  W_PREV=G_CLASS+G_TCHER+G_ROOM  && PREV SCREEN INPUT
  IF (W_MSG<>SPACE(80))
    ? CHR(7)
```

```
 ENDIF
 IF (W_ASSPF='ASSTEMP')
   SET COLOR TO W+/N
   @ 0,0 SAY '(WORK FILE IN USE)'
 ENDIF
 SET CURSOR ON
 SET COLOR TO N/W
 @ 0,30 SAY "TIME TABLE - INQUIRY"
 SET COLOR TO W/N
 @ 0,72 SAY DATE()
 @ 1,72 SAY TIME()
 @ 2, 0 CLEAR TO 2,78
 @ 2, 0 SAY "CLASS :"   GET G_CLASS PICTURE 'XX'
 @ 2,14 SAY "TEACHER :" GET G_TCHER PICTURE 'XX'
 @ 2,30 SAY "ROOM :"    GET G_ROOM  PICTURE 'XX'
 @23, 0 SAY "ESC=Cancel   F5=Session Switch  F7=Room Switch  F9=Manually Assign  F10=Split"
 @24, 0 CLEAR TO 24,80
 @24, 0 SAY W_MSG
 W_MSG=SPACE(80)
 L_FKEY=0
 READ  && L_FKEY: FUNCTION KEY RETURN

* ESC KEY PRESSED DURING 'READ'
 IF (L_FKEY=30)  && ESC
   IF FILE("ASSTEMP.DBF")
     G_OPT=' '
     @22,0 CLEAR TO 24,79
     @22,0 TO 24,79 DOUBLE
     @23,3 SAY 'ADJUSTMENT: (C)onfirm/(K)eep till next session/(D)elete ' ;
         GET G_OPT PICTURE '!'
     L_FKEY=0
     READ
     DO CASE
       CASE UPPER(G_OPT)='C'
          @23,3 CLEAR TO 23,78
          @23,3 SAY 'Work File(s) Saved and Deleted'
          COPY FILE ASSTEMP.DBF TO ASSIGNPF.DBF
          ERASE ASSTEMP.DBF
          IF FILE('CLSTEMP.DBF')
            COPY FILE CLSTEMP.DBF TO CLSUBJPF.DBF
            ERASE CLSTEMP.DBF
            ERASE CLSTEMP1.NDX
          ENDIF
       CASE UPPER(G_OPT)='D'
          @23,3 CLEAR TO 23,78
          @23,3 SAY 'Work File(s) Deleted. All Changes Lost'
          ERASE ASSTEMP.DBF
          ERASE CLSTEMP.DBF
          ERASE CLSTEMP1.NDX
       OTHERWISE
          @23,3 CLEAR TO 23,78
          @23,3 SAY 'Work File(s) Has Been Kept For Later Use'
     ENDCASE
     SET CURSOR ON
   ENDIF  && FILE('ASSTEMP.DBF')
   RETURN
 ENDIF  && L_FKEY=30 ESC

 IF (W_PREV<>G_CLASS+G_TCHER+G_ROOM)
   IF (G_CLASS="  ") .AND. (G_TCHER="  ") .AND. (G_ROOM="  ")
    LOOP  && BLANK INPUT, LOOP AGAIN
   ENDIF  && G_CLASS=' ' & G_TCHER=' ' & G_ROOM=' '
   W_COUNT=0
   IF (G_CLASS<>"  ")
     W_COUNT=W_COUNT+1
```

```
 ENDIF
 IF (G_TCHER<>"  ")
   W_COUNT=W_COUNT+1
 ENDIF
 IF (G_ROOM<>"  ")
   W_COUNT=W_COUNT+1
 ENDIF
 IF (W_COUNT>1)
   W_MSG='Please Enter Only One Entry'
   LOOP  && BLANK INPUT, LOOP AGAIN
 ENDIF

 W_OK=.T.
 W_FILTER1=' '
 IF (G_CLASS<>"  ")
   G_CLASS=UPPER(G_CLASS)
   W_FILTER1="CLASS='"+G_CLASS+"'"

   IF (W_FILTER1<>" ")
     SELECT 1
     USE &W_ASSPF ALIAS ASSIGNPF
     LOCATE FOR &W_FILTER1
     IF (.NOT. FOUND())
       W_OK=.F.
       W_MSG="Class code '"+G_CLASS+"' not found in assignment file"
       LOOP  && ERROR, LOOP AGAIN
     ENDIF  && .NOT. FOUND()
   ENDIF  && W_FILTER1<>" "
 ENDIF  && G_CLASS<>' '

 W_FILTER2=' '
 IF (G_TCHER<>"  ") .AND. W_OK
   G_TCHER=UPPER(G_TCHER)
   W_FILTER2="TCHER='"+G_TCHER+"'"

   IF (W_FILTER2<>" ")
     SELECT 1
     USE &W_ASSPF ALIAS ASSIGNPF
     LOCATE FOR &W_FILTER2
     IF (.NOT. FOUND())
       W_OK=.F.
       W_MSG="Teacher code '"+G_TCHER+"' not found in assignment file"
       LOOP  && ERROR, LOOP AGAIN
     ENDIF  && .NOT. FOUND()
   ENDIF  && W_FILTER2<>" "
 ENDIF  && G_TCHER<>' '

 W_FILTER3=' '
 IF (G_ROOM<>"  ")  .AND. W_OK
   G_ROOM=UPPER(G_ROOM)
   W_FILTER3="ROOM='"+G_ROOM+"'"

   IF (W_FILTER3<>" ")
     SELECT 1
     USE &W_ASSPF ALIAS ASSIGNPF
     LOCATE FOR &W_FILTER3
     IF (.NOT. FOUND())
       W_OK=.F.
       W_MSG="Room code '"+G_ROOM+"' not found in assignment file"
       LOOP  && ERROR, LOOP AGAIN
     ENDIF  && .NOT. FOUND()
   ENDIF  && W_FILTER3<>" "
 ENDIF  && G_ROOM<>' '
ENDIF  && W_PREV<>G_CLASS+G_TCHER+G_ROOM
```

```
DO CASE
  CASE (L_FKEY= 0)  && ENTER
  CASE (L_FKEY= 5)  && SWITCH SESSION
     IF (W_FILTER1=" ")
       W_MSG='Switch Session is valid only while CLASS TIME TABLE is in displayed'
       LOOP
     ENDIF  && W_FILTER1=' '
     IF  W_PREV=' '
       DO PRINT
     ENDIF
     W_PREV_RTN='MAIN'
     DO SWITCH
     LOOP

  CASE (L_FKEY= 7)  && SWITCH ROOM
    IF (W_FILTER1=" ")
       W_MSG='Switch Room is valid only while CLASS TIME TABLE is in displayed'
       LOOP
     ENDIF  && W_FILTER1=' '
     IF W_PREV=' '
       DO PRINT
     ENDIF
     W_PREV_RTN='MAIN'
     DO SWITCHROOM
     LOOP

  CASE (L_FKEY= 9)  && MANUALLY ASSIGN
     IF (W_FILTER1=" ")
       W_MSG='Manually Assign is valid only while CLASS TIME TABLE is in displayed'
       LOOP
     ENDIF  && W_FILTER1=' '
     IF W_PREV=' '
       DO PRINT
     ENDIF
     W_PREV_RTN='MAIN'
     DO MANUALASGN
     LOOP

  CASE (L_FKEY=10)  && SPLITTING MULTIPLE CLASS BLOCK INTO SINGLE
     IF (W_FILTER1=" ")
       W_MSG='Split Class Block is valid only while CLASS TIME TABLE is in displayed'
       LOOP
     ENDIF  && W_FILTER1=' '

     IF W_PREV=' '
       DO PRINT
     ENDIF
     W_PREV_RTN='MAIN'
     DO SPLITBLOCK
     LOOP

  CASE (L_FKEY=30)  && ESC
  CASE (L_FKEY=31)  && PAGEUP
  CASE (L_FKEY=32)  && PAGEDOWN
  CASE (L_FKEY=30)  && ESC
  CASE (L_FKEY>=1) .AND. (L_FKEY<=9) && F1-F9
     W_MSG="Function Key 'F"+STR(L_FKEY,1)+"' is not assigned"
     LOOP
  CASE (L_FKEY>=10) .AND. (L_FKEY<=20) && F10-F20
     W_MSG="Function Key 'F"+STR(L_FKEY,2)+"' not valid"
     LOOP
  OTHERWISE
     W_MSG="Function Key not valid"
     LOOP
ENDCASE
```

```
  IF (W_OK=.T.)
    W_MSG='Inquiry in progress...'
    @24, 0 CLEAR TO 24,80
    @24, 0 SAY W_MSG
    W_MSG=SPACE(80)

    DO PRINT
  ELSE
    W_MSG ="Please Enter A Class / Teacher / Room Code."
    LOOP
  ENDIF  && W_OK
ENDDO  && .T.

W_STATUS=MEMOWRIT("TTNXTPGM.BAT",D_HIDE+"TTMAIN.EXE")
CLOSE ALL
ERASE TEMP.NDX
SET CURSOR ON
RETURN  && MAIN LINE
** ----------------------------------------

**** PART A, INQUIRY

PROCEDURE PRINT
SET CURSOR OFF
IF (G_CLASS<>"  ")
  SELECT 1
  USE &W_ASSPF ALIAS ASSIGNPF
  IF (W_FILTER1<>"  ")
    SET FILTER TO &W_FILTER1
  ENDIF
  DO PRINTCLASS
ENDIF  && G_CLASS<>"  "
IF (G_TCHER<>"  ")
  SELECT 1
  USE &W_ASSPF ALIAS ASSIGNPF
  IF (W_FILTER2<>"  ")
    SET FILTER TO &W_FILTER2
  ENDIF
  INDEX ON TCHER+SESSION+DAY TO TEMP
  SET INDEX TO TEMP
  DO PRINTTCHER
ENDIF  && G_TCHER<>"  "
IF (G_ROOM<>"  ")
  SELECT 1
  USE &W_ASSPF ALIAS ASSIGNPF
  IF (W_FILTER3<>"  ")
    SET FILTER TO &W_FILTER3
  ENDIF
  INDEX ON ROOM+SESSION+DAY TO TEMP
  SET INDEX TO TEMP
  DO PRINTROOM
ENDIF  && G_ROOM<>"  "
RETURN   && PRINT
** ----------------------------------------

PROCEDURE PRINTCLASS
PRIVATE W_PRVPRTSS, W_PRVCLASS, W_PRVSESS, W_COL, W_SABBREV, W_TABBREV, W_LINE1

* PRINT OUT TIMETABLE FOR ALL SPECIFIED CLASSES AND LIST OF
* UNALLOCATED SUBJECTS FOLLOWING THE CLASS

SELECT 1
USE &W_ASSPF ALIAS ASSIGNPF
IF (W_FILTER1<>"  ")
  SET FILTER TO &W_FILTER1
```

```
 ENDIF
 INDEX ON CLASS+SESSION+DAY TO TEMP
 SET INDEX TO TEMP
 SELECT 2
 USE &W_CLSPF ALIAS CLSUBJPF INDEX &W_CLSL1
 IF (W_FILTER1<>" ")
   SET FILTER TO &W_FILTER1
 ENDIF

 @1,0 SAY "Legend: Subject/Teacher/Room "
 W_LINE1=W_SPACE
 W_LINE2=SPACE(D_DAY)
 SELECT 3
 USE CLASSPF
 SELECT 4
 USE SUBJPF
 SELECT 5
 USE TCHERPF
 SELECT ASSIGNPF
 GO TOP
 IF (.NOT. EOF())
   W_PRVPRTSS="00"  && SESSION PREVIOUSLY PRINTED
   W_PRVCLASS=CLASS
   W_PRVSESS=SESSION
   DO NEWCLASS WITH W_PRVCLASS
   SELECT ASSIGNPF
 ENDIF
 DO WHILE (.NOT. EOF())
   IF (ASSIGNPF->CLASS<>W_PRVCLASS)
     DO BLANKSESS  WITH W_PRVPRTSS, W_PRVSESS
     DO PRTSESSION WITH W_PRVSESS, W_LINE1, W_LINE2
     DO BLANKSESS  WITH W_PRVSESS, W_LASTSESS
     DO PRTUNALLOC WITH W_PRVCLASS
     SELECT ASSIGNPF
     W_PRVPRTSS="00"
     W_PRVSESS=SESSION
     W_PRVCLASS=CLASS
     DO NEWCLASS WITH W_PRVCLASS
     W_LINE1=W_SPACE
     W_LINE2=SPACE(D_DAY)
   ENDIF  && (CLASS<>W_PRVCLASS)

   IF (ASSIGNPF->SESSION<>W_PRVSESS)  && SESSION CHANGED
     DO BLANKSESS  WITH W_PRVPRTSS, W_PRVSESS  && PRINT ANY BLANK SESSION IN BETWEEN
     DO PRTSESSION WITH W_PRVSESS, W_LINE1, W_LINE2
     W_PRVPRTSS=W_PRVSESS
     W_LINE1=W_SPACE
     W_LINE2=SPACE(D_DAY)
     W_PRVSESS=ASSIGNPF->SESSION
   ENDIF  && (SESSION<>W_PRVSESS)

   W_COL=3+9*(VAL(ASSIGNPF->DAY)-1)
   SELECT ASSIGNPF
   W_LINE1=STUFF(W_LINE1,W_COL,5,SUBJECT+'/'+TCHER)
   SELECT ASSIGNPF
   IF (CLASS<>ROOM)
     W_LINE1=STUFF(W_LINE1,W_COL+5,3,'/'+ROOM)
   ENDIF  && (CLASS<>ROOM)
   W_LINE2=STUFF(W_LINE2,VAL(DAY),1,TYPE)
   SKIP
 ENDDO  && (.NOT. EOF())

 DO BLANKSESS  WITH W_PRVPRTSS, W_PRVSESS
 DO PRTSESSION WITH W_PRVSESS, W_LINE1, W_LINE2   && LAST LINE
 DO BLANKSESS  WITH W_PRVSESS, W_LASTSESS
```

```
ROW=ROW+1
@ ROW, 4 SAY W_DASH
DO PRTUNALLOC WITH W_PRVCLASS
RETURN  && PRINTCLASS
** ----------------------------------------

PROCEDURE NEWCLASS
PARAMETERS P_CLASS
PRIVATE W_DESC

** PRINT TITLE FOR NEW CLASS

SELECT CLASSPF
LOCATE FOR CLASS=P_CLASS
W_DESC=" "
IF FOUND()
  W_DESC=DESC
ENDIF

SET COLOR TO N/W
@2,24 SAY "  "
@2,37 SAY "  "
SET COLOR TO W/N
@3, 0 CLEAR TO 22,79
@0,72 SAY DATE()
@1,72 SAY TIME()
@3,8  SAY W_DESC
@4,4  SAY W_DTITLE
@5,4  SAY W_DASH
ROW=5
RETURN  && NEWCLASS
** ----------------------------------------

PROCEDURE  PRTUNALLOC
PARAMETERS P_CLASS
PRIVATE W_FILTER, W_LINE1, W_LINE2

* PRINT OUT THE LIST OF SUBJECTS IN THE CLASS WHICH ARE NOT ABLE TO ALLOCATE

SELECT 2
USE
W_FILTER="REPEAT>FINISHED .AND. CLASS='"+P_CLASS+"'"
USE &W_CLSPF ALIAS CLSUBJPF INDEX &W_CLSL1
SET FILTER TO &W_FILTER
GO TOP
IF EOF()
  USE
  RETURN  && NO UNALLOCATED CLASS
ENDIF

@ ROW+1, 4 SAY "Record  Subject  Type  Teacher  Room  # of unalloc session"
@ ROW+2, 4 SAY "----------------------------------------------------------"
ROW=ROW+2

DO WHILE (.NOT. EOF())
  SELECT SUBJPF
  LOCATE FOR SUBJECT=CLSUBJPF->SUBJECT
  IF FOUND()
    W_LINE1=ABBREV
  ELSE
    W_LINE1=SUBJECT
  ENDIF  && FOUND()

  SELECT TCHERPF
  LOCATE FOR TCHER=CLSUBJPF->TCHER
```

```
  IF FOUND()
    W_LINE2=ABBREV
  ELSE
    W_LINE2=TCHER
  ENDIF  && FOUND()

  SELECT CLSUBJPF
  ROW=ROW+1
  @ ROW, 5 SAY SEQ PICTURE '@Z9999'
  @ ROW,12 SAY SUBJECT+'/'+W_LINE1
  @ ROW,22 SAY CLASSTYPE
  @ ROW,27 SAY TCHER+'/'+W_LINE2
  @ ROW,36 SAY ROOM
  @ ROW,49 SAY REPEAT-FINISHED PICTURE "99"
  IF (ROW>=21)
    EXIT     && NOT ENOUGH LINE ON SCREEN
  ENDIF

  SKIP
ENDDO  && DO WHILE .NOT. EOF()
ROW=ROW+1
@ ROW, 4 SAY "------------------------------------------------------------"
SELECT CLSUBJPF
USE
RETURN  && PRTUNALLOC
** ----------------------------------------

PROCEDURE PRINTTCHER
PRIVATE W_PRVPRTSS, W_PRVTCHER, W_PRVSESS, W_COL, W_SABBREV, W_LINE1, W_LINE2

* PRINT OUT TIMETABLE FOR ALL SPECIFIED TEACHERS

@1,0 SAY "Legend: Subject/Class/Room   "
W_LINE1=W_SPACE
W_LINE2=SPACE(D_DAY)

SELECT 4
USE SUBJPF
SELECT 5
USE TCHERPF
SELECT ASSIGNPF
GO TOP

IF (.NOT. EOF())
  W_PRVPRTSS="00"  && SESSION PREVIOUSLY PRINTED
  W_PRVTCHER=TCHER
  W_PRVSESS=SESSION
  DO NEWTCHER WITH W_PRVTCHER
  SELECT ASSIGNPF
ENDIF
DO WHILE (.NOT. EOF())
  IF (ASSIGNPF->TCHER<>W_PRVTCHER)  && ROOM CHANGED
    DO BLANKSESS  WITH W_PRVPRTSS, W_PRVSESS
    DO PRTSESSION WITH W_PRVSESS, W_LINE1, W_LINE2
    DO BLANKSESS  WITH W_PRVSESS, W_LASTSESS  && PRINT BLANK SESSION TILL LAST
    SELECT ASSIGNPF
    W_PRVPRTSS="00"
    W_PRVTCHER=TCHER
    W_PRVSESS=SESSION
    DO NEWTCHER WITH W_PRVTCHER
    SELECT ASSIGNPF
    W_LINE1=W_SPACE
    W_LINE2=SPACE(D_DAY)
  ENDIF  && (TCHER<>W_PRVTCHER)
```

```
  IF (ASSIGNPF->SESSION<>W_PRVSESS)  && SESSION CHANGED
    DO BLANKSESS WITH W_PRVPRTSS, W_PRVSESS  && PRINT ANY BLANK SESSION IN BETWEEN
    DO PRTSESSION WITH W_PRVSESS, W_LINE1, W_LINE2
    W_PRVPRTSS=W_PRVSESS
    W_LINE1=W_SPACE
    W_LINE2=SPACE(D_DAY)
    W_PRVSESS=ASSIGNPF->SESSION
  ENDIF  && (SESSION<>W_PRVSESS)
  W_COL=3+9*(VAL(ASSIGNPF->DAY)-1)
  SELECT ASSIGNPF
  W_LINE1=STUFF(W_LINE1,W_COL,5,SUBJECT+'/'+CLASS)
  IF (CLASS<>ROOM)
    W_LINE1=STUFF(W_LINE1,W_COL+5,3,'/'+ROOM)
  ENDIF
  W_LINE2=STUFF(W_LINE2,VAL(DAY),1,TYPE)
  SKIP
ENDDO  && (.NOT. EOF())

DO BLANKSESS  WITH W_PRVPRTSS, W_PRVSESS
DO PRTSESSION WITH W_PRVSESS, W_LINE1, W_LINE2   && LAST LINE
DO BLANKSESS  WITH W_PRVSESS, W_LASTSESS
ROW=ROW+1
@ ROW, 4 SAY W_DASH
RETURN  && PRINTTCHER
** ----------------------------------------

PROCEDURE NEWTCHER
PARAMETERS P_TCHER
PRIVATE W_DESC

** PRINT TITLE FOR NEW TEACHER

SELECT TCHERPF
LOCATE FOR TCHER=P_TCHER
W_DESC=" "
IF FOUND()
  W_DESC=DESC
ENDIF

SET COLOR TO N/W
@2, 8 SAY "  "
@2,37 SAY "  "
SET COLOR TO W/N
@3, 0 CLEAR TO 22,79
@0,72 SAY DATE()
@1,72 SAY TIME()
@3,24 SAY W_DESC
@4,4  SAY W_DTITLE
@5,4  SAY W_DASH
ROW=5
RETURN  && NEWTCHER
** ----------------------------------------

PROCEDURE PRINTROOM
* PRINT OUT TIMETABLE FOR ALL SPECIFIED ROOMS
PRIVATE W_PRVPRTSS, W_PRVROOM, W_PRVSESS, W_COL, W_SABBREV, W_TABBREV, W_LINE1, W_LINE2

@1,0 SAY "Legend: Subject/Teacher/Class"
W_LINE1=W_SPACE
W_LINE2=SPACE(D_DAY)

SELECT 3
USE ROOMPF
SELECT 4
USE SUBJPF
```

```
SELECT 5
USE TCHERPF
SELECT ASSIGNPF
GO TOP

IF (.NOT. EOF())
  W_PRVPRTSS="00"  && SESSION PREVIOUSLY PRINTED
  W_PRVROOM=ROOM
  W_PRVSESS=SESSION
  DO NEWROOM WITH W_PRVROOM
  SELECT ASSIGNPF
ENDIF
DO WHILE (.NOT. EOF())
  IF (ASSIGNPF->ROOM<>W_PRVROOM)  && ROOM CHANGED
    DO BLANKSESS  WITH W_PRVPRTSS, W_PRVSESS
    DO PRTSESSION WITH W_PRVSESS, W_LINE1, W_LINE2
    DO BLANKSESS  WITH W_PRVSESS, W_LASTSESS  && PRINT BLANK SESSION TILL LAST
    SELECT ASSIGNPF
    W_PRVPRTSS="00"
    W_PRVROOM=ROOM
    W_PRVSESS=SESSION
    DO NEWROOM WITH W_PRVROOM
    SELECT ASSIGNPF
    W_LINE1=W_SPACE
    W_LINE2=SPACE(D_DAY)
  ENDIF  && (ROOM<>W_PRVROOM)

  IF (ASSIGNPF->SESSION<>W_PRVSESS)  && SESSION CHANGED
    DO BLANKSESS WITH W_PRVPRTSS, W_PRVSESS  && PRINT ANY BLANK SESSION IN BETWEEN
    DO PRTSESSION WITH W_PRVSESS, W_LINE1, W_LINE2
    W_PRVPRTSS=W_PRVSESS
    W_LINE1=W_SPACE
    W_LINE2=SPACE(D_DAY)
    W_PRVSESS=ASSIGNPF->SESSION
  ENDIF  && (SESSION<>W_PRVSESS)

  W_COL=3+9*(VAL(ASSIGNPF->DAY)-1)
  SELECT ASSIGNPF
  W_LINE1=STUFF(W_LINE1,W_COL,5,SUBJECT+'/'+TCHER)
  IF (CLASS<>ROOM)
    W_LINE1=STUFF(W_LINE1,W_COL+5,3,'/'+CLASS)
  ENDIF
  W_LINE2=STUFF(W_LINE2,VAL(DAY),1,TYPE)
  SKIP
ENDDO  && (.NOT. EOF())

DO BLANKSESS  WITH W_PRVPRTSS, W_PRVSESS
DO PRTSESSION WITH W_PRVSESS, W_LINE1, W_LINE2  && LAST LINE
DO BLANKSESS  WITH W_PRVSESS, W_LASTSESS
ROW=ROW+1
@ ROW, 4 SAY W_DASH
RETURN  && PRINTROOM
** ----------------------------------------

PROCEDURE NEWROOM
PARAMETERS P_ROOM
PRIVATE W_DESC

** PRINT TITLE FOR NEW ROOM

SELECT ROOMPF
LOCATE FOR ROOM=P_ROOM
W_DESC=" "
IF FOUND()
 W_DESC=DESC
```

```
 ENDIF

 SET COLOR TO N/W
 @2, 8 SAY "  "
 @2,24 SAY "  "
 SET COLOR TO W/N
 @3, 0 CLEAR TO 22,79
 @0,72 SAY DATE()
 @1,72 SAY TIME()
 @3,37 SAY W_DESC
 @4,4  SAY W_DTITLE
 @5,4  SAY W_DASH
 ROW=5
 RETURN  && NEWROOM
 ** ----------------------------------------

 PROCEDURE BLANKSESS
 PARAMETERS P_PRVSESS, P_NEXTSESS
 PRIVATE   W_PS, W_S, W_SESS

 * PRINT OUT ROWS OF BLANK SESSION WHICH ARE NOT ASSIGNED TO

 W_PS=VAL(P_PRVSESS)
 W_S=VAL(P_NEXTSESS)
 DO WHILE (W_PS+1<W_S)
   W_PS=W_PS+1
   W_SESS=STR(W_PS,2)
   DO PRTSESSION WITH W_SESS, W_SPACE, W_SPACE  && BLANK SESSION LINE
 ENDDO   && (P_PS+1<P_S)
 RETURN  && BLANKSESS
 ** ----------------------------------------

 PROCEDURE PRTSESSION
 PARAMETERS P_SESSION, P_LINE1, P_LINE2
 PRIVATE W_SESS, W_DAY, W_COL

 * PRINT OUT A SINGLE SESSION OF THE TIMETABLE

 W_SESS=VAL(P_SESSION)
 IF (W_SESS=(D_BASTRUC1+1)) .OR. (W_SESS=(D_BASTRUC1+D_BASTRUC2+1)) ;
   .OR. (W_SESS=(D_BASTRUC1+D_BASTRUC2+D_BASTRUC3+1)) ;
   .OR. (W_SESS=(D_BASTRUC1+D_BASTRUC2+D_BASTRUC3+D_BASTRUC4+1))  ;
   .OR. (W_SESS=(D_BASTRUC1+D_BASTRUC2+D_BASTRUC3+D_BASTRUC4+D_BASTRUC5+1))
   DO PRTRECESS   && SESSION AFTER RECESS
 ENDIF
 ROW=ROW+1
 @ ROW,0 SAY P_SESSION PICTURE "XX"
 W_COL=4
 W_DAY=0
 DO WHILE (W_DAY<D_DAY)
   W_DAY=W_DAY+1
   DO CASE
     CASE (SUBSTR(P_LINE2,W_DAY,1)='T')
        SET COLOR TO N/W
     CASE (SUBSTR(P_LINE2,W_DAY,1)='D')
        SET COLOR TO W+/N
   ENDCASE
   @ ROW,W_COL SAY SUBSTR(P_LINE1,(W_DAY-1)*9+3,9)
   W_COL=W_COL+9
   SET COLOR TO W/N
 ENDDO  && (W_DAY<D_DAY)
 RETURN  && PRTSESSION
 ** ----------------------------------------

 PROCEDURE PRTRECESS
```

```
* PRINT RECESS LINE

@ ROW+1,4 SAY W_RECESS
ROW=ROW+1
RETURN  && PRTRECESS
** ----------------------------------------

**** PART B, ROOM OR SESSION SWITCH, MANUAL ASSIGNMENT, BLOCK SPLITTING
PROCEDURE SWITCH
PRIVATE G_FDAY, G_FSESS, G_TDAY, G_TSESS, W_MSG, ;
      W_FTYPE, W_FSLOT1, W_FSLOT2, W_FSLOT3, ;
      W_TTYPE, W_TSLOT1, W_TSLOT2, W_TSLOT3
* DO MANUAL ADJUSTMENT
W_MSG=SPACE(80)
DO WHILE .T.
  CLOSE DATABASES
  IF (W_MSG<>SPACE(80))
    ? CHR(7)
  ENDIF
  IF (W_ASSPF='ASSTEMP')
    SET COLOR TO W+/N
    @ 0,0 SAY '(WORK FILE IN USE)'
    SET COLOR TO W/N
  ENDIF
  SET CURSOR ON
  G_FDAY=' '
  G_FSESS=' '
  G_TDAY=' '
  G_TSESS=' '
  @22,0  CLEAR TO 24,79
  @22,0  SAY 'Switch : Day '
  @22,13 GET G_FDAY PICTURE '99'
  @22,16 SAY 'Session '
  @22,24 GET G_FSESS PICTURE '99'
  @22,30 SAY 'and   Day '
  @22,40 GET G_TDAY PICTURE '99'
  @22,43 SAY 'Session '
  @22,51 GET G_TSESS PICTURE '99'
  @23,0  SAY 'F3=Cancel    F7=Room Switch   F9=Manually Assign   F10=Split'
  @24,0  SAY W_MSG
  L_FKEY=0
  READ   && L_FKEY: FUNCTION KEY #

  W_MSG=SPACE(80)
  DO CASE
    CASE (L_FKEY= 0) .OR. ;  && ENTER
       (L_FKEY= 5) .OR. ;  && MANUAL ADJUSTMENT
       (L_FKEY=31) .OR. ;  && PAGEUP
       (L_FKEY=32)         && PAGEDOWN
       G_FDAY =RIGHT("  "+ALLTRIM(G_FDAY),2)
       G_FSESS=RIGHT("  "+ALLTRIM(G_FSESS),2)
       G_TDAY =RIGHT("  "+ALLTRIM(G_TDAY),2)
       G_TSESS=RIGHT("  "+ALLTRIM(G_TSESS),2)
       SELECT 1
       USE &W_ASSPF ALIAS ASSIGNPF
       W_FTYPE=' '
       W_FSLOT1=' '
       W_FSLOT2=' '
       W_FSLOT3=' '
       W_TTYPE=' '
       W_TSLOT1=' '
       W_TSLOT2=' '
       W_TSLOT3=' '
       && VALIDATE INPUTs & ADJUST TO STARTING SESSION #
       DO VALIDATE WITH W_FTYPE, W_FSLOT1, W_FSLOT2, W_FSLOT3, ;
```

```
                W_TTYPE, W_TSLOT1, W_TSLOT2, W_TSLOT3
        IF (W_MSG<>SPACE(80))  && ERROR
          LOOP
        ENDIF
        && TRY TO SWITCH
        DO SESSWITCH WITH W_FSLOT1, W_FSLOT2, W_FSLOT3, ;
                W_TSLOT1, W_TSLOT2, W_TSLOT3
        IF (W_MSG<>SPACE(80))  && ERROR
          LOOP
        ENDIF

      CASE (L_FKEY=7)  && ROOM SWITCHING
        IF (W_PREV_RTN='SWITCHROOM')  && CALLER OF CURRENT INSTANT IS 'SWITCHROOM'
          RETURN
        ENDIF
        W_PREV_RTN='SWITCH'
        DO SWITCHROOM
      CASE (L_FKEY=9)  && MANUALLY ASSIGN
        IF (W_PREV_RTN='MANUALASGN')  && CALLER OF CURRENT INSTANT IS 'MANUALASGN'
          RETURN
        ENDIF
        W_PREV_RTN='SWITCH'
        DO MANUALASGN
      CASE (L_FKEY=10)  && SPLITTING MULTIPLE CLASS BLOCK
        IF (W_PREV_RTN='SPLITBLOCK')  && CALLER OF CURRENT INSTANT IS 'SPLITBLOCK'
          RETURN
        ENDIF
        W_PREV_RTN='SWITCH'
        DO SPLITBLOCK
      CASE (L_FKEY=30)  && ESC
        @22,0 CLEAR TO 22,79
        EXIT
      CASE (L_FKEY=3)   && FUNCTION F3
        @22,0 CLEAR TO 22,79
        EXIT
      CASE (L_FKEY>=1) .AND. (L_FKEY<=9) && F1-F9
        W_MSG="Function Key 'F"+STR(L_FKEY,1)+"' is not assigned"
        LOOP
      CASE (L_FKEY>=10) .AND. (L_FKEY<=20) && F10-F20
        W_MSG="Function Key 'F"+STR(L_FKEY,2)+"' not valid"
        LOOP
      OTHERWISE
        W_MSG="Function Key not valid"
    ENDCASE
  ENDDO  && .T.
  @23, 0 SAY "ESC=Cancel    F5=Session Switch   F7=Room Switch   F9=Manually Assign  F10=Split"
  RETURN  && SWITCH
  ** ----------------------------------------

  PROCEDURE VALIDATE
  PARAMETERS P_FTYPE, P_FSLOT1, P_FSLOT2, P_FSLOT3, ;
        P_TTYPE, P_TSLOT1, P_TSLOT2, P_TSLOT3
  PRIVATE W_FFOUND, W_TFOUND, W_OK, W_TEMP
  * VALIDATE INPUT DAYs AND SESSIONs AND ADJUST THE INPUTs TO THE
  * CORRESPONDING STARTING SESSION

  W_MSG=' '
  DO VALIDATE2 WITH G_FDAY, G_FSESS
  IF (W_MSG<>' ')
    RETURN
  ENDIF
  DO VALIDATE2 WITH G_TDAY, G_TSESS
  IF (W_MSG<>' ')
    RETURN
  ENDIF
```

```
IF (G_FDAY=G_TDAY) .AND. (G_FSESS=G_TSESS)
  W_MSG='Day and session must not be the same'
  RETURN
ENDIF

P_FTYPE=' '
W_FFOUND=.F.
DO SESSADJUST WITH G_FDAY, G_FSESS, P_FTYPE, W_FFOUND && SESSION# ADJUSTMENT
P_TTYPE=' '
W_TFOUND=.F.
DO SESSADJUST WITH G_TDAY, G_TSESS, P_TTYPE, W_TFOUND

IF (.NOT. W_FFOUND) .AND. W_TFOUND
  W_TEMP=G_FDAY  && SWITCH BLANK 'FROM' SESSION TO 'TO' SESSION
  G_FDAY=G_TDAY
  G_TDAY=W_TEMP
  W_TEMP=G_FSESS
  G_FSESS=G_TSESS
  G_TSESS=W_TEMP
  W_TEMP=P_FTYPE
  P_FTYPE=P_TTYPE
  P_TTYPE=W_TEMP
  W_TEMP=W_FFOUND
  W_FFOUND=W_TFOUND
  W_TFOUND=W_TEMP
ENDIF
P_FSLOT1=' '
P_FSLOT2=' '
P_FSLOT3=' '
* GET SLOT#s (RETURN SLOT# OF AVAILABLE BLANK SLOTs IF DAY/SESSION IS BLANK)
DO GETSLOT WITH G_FDAY, G_FSESS, P_FTYPE, P_FSLOT1, P_FSLOT2, P_FSLOT3
P_TSLOT1=' '
P_TSLOT2=' '
P_TSLOT3=' '
DO GETSLOT WITH G_TDAY, G_TSESS, P_TTYPE, P_TSLOT1, P_TSLOT2, P_TSLOT3

DO CASE
  CASE (.NOT. W_FFOUND) .AND. (.NOT. W_TFOUND)
     W_MSG='Cannot do switching on empty sessions'
     RETURN

  CASE W_FFOUND .AND. (.NOT. W_TFOUND)
* CHECK WHETHER OR NOT ENOUGH BLANK 'TO' SESSION(s) TO FIT THE 'P_FTYPE' SESSION
     IF ((P_FTYPE='T') .AND. (P_TSLOT3=' ')) .OR. ;
       ((P_FTYPE='D') .AND. (P_TSLOT2=' '))
       W_MSG='Not enough blank session(s) to accomodate the subject'
       RETURN
     ENDIF
     IF (P_FTYPE='S')
       P_TSLOT2=' '
       P_TSLOT3=' '
     ENDIF
     IF (P_FTYPE='D')
       P_TSLOT3=' '
     ENDIF

  CASE W_FFOUND .AND. W_TFOUND
     IF (G_FDAY=G_TDAY) .AND. (G_FSESS=G_TSESS)
       W_MSG='Day and session must not be the same'
       RETURN
     ENDIF
     IF (P_FTYPE<>P_TTYPE)
       W_MSG='Session Types not match'
       RETURN
     ENDIF
```

```
  ENDCASE
 RETURN  && VALIDATE
 ** ----------------------------------------

 PROCEDURE SESSADJUST
 PARAMETERS P_DAY, P_SESS, P_TYPE, P_FOUND
 PRIVATE W_COND
 * ADJUST USER ENTERED SESSION # TO A STARTING SESSION # IF ASSIGNMENT IS FOUND
 P_TYPE=' '
 SELECT ASSIGNPF
 LOCATE FOR CLASS=G_CLASS .AND. DAY=P_DAY .AND. SESSION=P_SESS
 P_FOUND=FOUND()
 IF P_FOUND
   P_TYPE=TYPE
   DO CASE
     CASE TYPE='D' .AND. TYPE1='2'
        P_SESS=STR(VAL(P_SESS)-1,2) && ADJUST TO STARTING OF DOUBLE SESSION
     CASE TYPE='T' .AND. TYPE1='2'
        P_SESS=STR(VAL(P_SESS)-1,2) && ADJUST TO STARTING OF TRIPLE SESSION
     CASE TYPE='T' .AND. TYPE1='3'
        P_SESS=STR(VAL(P_SESS)-2,2) && ADJUST TO STARTING OF TRIPLE SESSION
   ENDCASE
 ENDIF  && P_FOUND
 RETURN  && SESSADJUST
 ** ----------------------------------------

 PROCEDURE GETSLOT
 PARAMETERS P_DAY, P_SESS, P_TYPE, P_SLOT1, P_SLOT2, P_SLOT3
 PRIVATE   W_TYPE
 * GET SLOT#s FOR SLOT ON P_DAY & P_SESS
 * IF SLOT ON P_DAY & P_SESS IS BLANK, THIS ROUTINE WILL TRY TO RETURN
 * AT MOST THREE EMPTY SLOT#s WHICH FOLLOW

 SELECT 7
 USE DAYSESPF
 LOCATE FOR DAY=P_DAY .AND. SESSION=P_SESS
 IF (.NOT. FOUND())
   W_MSG="Corresponding SLOT NO. on Day "+LTRIM(P_DAY)+" Session "+ ;
      LTRIM(P_SESS)+" in 'DAYSESPF' not found"
   RETURN
 ENDIF
 P_SLOT1=SLOT
 USE TDSPF
 IF P_TYPE=' '
   W_TYPE='T'
 ELSE
   W_TYPE=P_TYPE
 ENDIF
 LOCATE FOR TYPE=W_TYPE .AND. SLOT1=P_SLOT1
 IF (.NOT. FOUND())
   LOCATE FOR TYPE='D' .AND. SLOT1=P_SLOT1
   IF (.NOT. FOUND())
     LOCATE FOR TYPE='S' .AND. SLOT1=P_SLOT1
     IF (.NOT. FOUND())
       W_MSG="Record with '"+P_TYPE+"' Session Type and Starting Slot no. '"+;
          LTRIM(P_SLOT1)+"' in 'TDSPF' not found"
       RETURN
     ENDIF
   ENDIF
 ENDIF
 P_SLOT2=SLOT2
 P_SLOT3=SLOT3
 USE
 DO CASE
   CASE P_TYPE='S'
```

```
      P_SLOT2=' '
      P_SLOT3=' '
   CASE P_TYPE='D'
      P_SLOT3=' '
   CASE P_TYPE=' ' && GIVEN DAY & SESSION IS EMPTY ON ASSIGNPF
      SELECT ASSIGNPF
      LOCATE FOR CLASS=G_CLASS .AND. SLOT=P_SLOT2
      IF FOUND()
        P_SLOT2=' '
        P_SLOT3=' '
      ELSE
        LOCATE FOR CLASS=G_CLASS .AND. SLOT=P_SLOT3
        IF FOUND()
          P_SLOT3=' '
        ENDIF
      ENDIF
ENDCASE
RETURN  && GETSLOT
** ----------------------------------------

PROCEDURE SESSWITCH
PARAMETERS P_FSLOT1, P_FSLOT2, P_FSLOT3, P_TSLOT1, P_TSLOT2, P_TSLOT3
PRIVATE W_COND, W_FOUND, W_TYPE, ;
      W_FSLOT, W_FTCHER, W_FROOM, W_FSUBJECT, W_FDAY, W_FSESSION, ;
      W_TSLOT, W_TTCHER, W_TROOM, W_TSUBJECT, W_TDAY, W_TSESSION, ;
      W_FFOUND, W_TFOUND, W_DAY, W_SESS
* TRY TO SWITCH THE SELECTED SESSIONs

W_FTCHER=' '
W_FROOM=' '
W_FSUBJECT=' '
W_FDAY=' '
W_FSESSION=' '
W_TTCHER=' '
W_TROOM=' '
W_TSUBJECT=' '
W_TDAY=' '
W_TSESSION=' '
W_TYPE=' '

* CREATE WORK FILE IF NOT ALREADY EXIST
SELECT ASSIGNPF
USE
IF (.NOT. FILE("ASSTEMP.DBF"))
  W_ASSPF="ASSTEMP"
  COPY FILE ASSIGNPF.DBF TO ASSTEMP.DBF
ENDIF
USE &W_ASSPF ALIAS ASSIGNPF
W_COND="CLASS='"+G_CLASS+"' .AND. SLOT='"+P_FSLOT1+"'"
LOCATE FOR &W_COND
W_FFOUND=FOUND()
IF W_FFOUND
  W_FTCHER=TCHER
  W_FROOM=ROOM
  W_FSUBJECT=SUBJECT
  W_FDAY=DAY
  W_FSESSION=SESSION
  W_TYPE=TYPE
ENDIF

W_COND="CLASS='"+G_CLASS+"' .AND. SLOT='"+P_TSLOT1+"'"
LOCATE FOR &W_COND
W_TFOUND=FOUND()
IF W_TFOUND
 W_TTCHER=TCHER
```

```
   W_TROOM=ROOM
   W_TSUBJECT=SUBJECT
   W_TDAY=DAY
   W_TSESSION=SESSION
   W_TYPE=TYPE
  ENDIF

  W_FSLOT=" "+P_FSLOT1+" "+P_FSLOT2+" "+P_FSLOT3+" "
  W_TSLOT=" "+P_TSLOT1+" "+P_TSLOT2+" "+P_TSLOT3+" "
  IF (ALLTRIM(W_TSLOT)+' '=' ')
    W_TSLOT=' '
  ENDIF
  W_COND="CLASS='"+G_CLASS+"' .AND. ((SLOT+' ') $ '"+W_FSLOT+W_TSLOT+"')"
  DELETE ALL FOR &W_COND

  * DO CHECKING IF TO-SLOT IS NOT BLANK
  W_FOUND=.F.
  * CHECK AVAILABLITY OF TEACHER
  DO CHECK WITH "T", W_FTCHER, P_TSLOT1, W_FOUND
  IF W_FOUND
    RECALL ALL
    RETURN
  ENDIF
  IF (P_TSLOT2<>' ')
    DO CHECK WITH "T", W_FTCHER, P_TSLOT2, W_FOUND
    IF W_FOUND
      RECALL ALL
      RETURN
    ENDIF
    IF (P_TSLOT3<>' ')
      DO CHECK WITH "T", W_FTCHER, P_TSLOT3, W_FOUND
      IF W_FOUND
        RECALL ALL
        RETURN
      ENDIF
    ENDIF
  ENDIF

  IF (W_TDAY<>' ')
    DO CHECK WITH "T", W_TTCHER, P_FSLOT1, W_FOUND
    IF W_FOUND
      RECALL ALL
      RETURN
    ENDIF
    IF (P_FSLOT2<>' ')
      DO CHECK WITH "T", W_TTCHER, P_FSLOT2, W_FOUND
      IF W_FOUND
        RECALL ALL
        RETURN
      ENDIF
      IF (P_FSLOT3<>' ')
        DO CHECK WITH "T", W_TTCHER, P_FSLOT3, W_FOUND
        IF W_FOUND
          RECALL ALL
          RETURN
        ENDIF
      ENDIF
    ENDIF
  ENDIF (W_TDAY<>' ')

  * CHECK AVAILABLITY OF ROOM
  DO CHECK WITH "R", W_FROOM, P_TSLOT1, W_FOUND
  IF W_FOUND
    RECALL ALL
    RETURN
```

```
 ENDIF
 IF (P_TSLOT2<>' ')
  DO CHECK WITH "R", W_FROOM, P_TSLOT2, W_FOUND
  IF W_FOUND
   RECALL ALL
   RETURN
  ENDIF
  IF (P_TSLOT3<>' ')
   DO CHECK WITH "R", W_FROOM, P_TSLOT3, W_FOUND
   IF W_FOUND
    RECALL ALL
    RETURN
   ENDIF
  ENDIF
 ENDIF

 IF (W_TDAY<>' ')
  DO CHECK WITH "R", W_TROOM, P_FSLOT1, W_FOUND
  IF W_FOUND
   RECALL ALL
   RETURN
  ENDIF
  IF (P_FSLOT2<>' ')
   DO CHECK WITH "R", W_TROOM, P_FSLOT2, W_FOUND
   IF W_FOUND
    RECALL ALL
    RETURN
   ENDIF
   IF (P_FSLOT3<>' ')
    DO CHECK WITH "R", W_TROOM, P_FSLOT3, W_FOUND
    IF W_FOUND
     RECALL ALL
     RETURN
    ENDIF
   ENDIF
  ENDIF
 ENDIF  && W_TDAY<>' '

 SELECT ASSIGNPF
 * DO ACTUAL SWITCH OPERATION
 IF (W_TDAY<>' ')   && 'TO' SESSION IS NOT BLANK
 *  'TO' SESSION TO 'FROM' SESSION
  APPEND BLANK
  REPLACE CLASS WITH G_CLASS, SLOT WITH P_FSLOT1, TCHER WITH W_TTCHER, ;
       ROOM  WITH W_TROOM, SUBJECT WITH W_TSUBJECT, DAY WITH W_FDAY, ;
       SESSION WITH W_FSESSION, TYPE WITH W_TYPE, TYPE1 WITH '1'
  IF (P_FSLOT2<>' ')
   APPEND BLANK
   REPLACE CLASS WITH G_CLASS, SLOT WITH P_FSLOT2, TCHER WITH W_TTCHER, ;
        ROOM  WITH W_TROOM, SUBJECT WITH W_TSUBJECT, DAY WITH W_FDAY, ;
        SESSION WITH STR(VAL(W_FSESSION)+1,2), TYPE WITH W_TYPE, TYPE1 WITH '2'
  ENDIF
  IF (P_FSLOT3<>' ')
   APPEND BLANK
   REPLACE CLASS WITH G_CLASS, SLOT WITH P_FSLOT3, TCHER WITH W_TTCHER, ;
        ROOM  WITH W_TROOM, SUBJECT WITH W_TSUBJECT, DAY WITH W_FDAY, ;
        SESSION WITH STR(VAL(W_FSESSION)+2,2), TYPE WITH W_TYPE, TYPE1 WITH '3'
  ENDIF
 ELSE
  SELECT 7
  USE DAYSESPF
  LOCATE FOR SLOT=P_TSLOT1
  W_TDAY=DAY
  W_TSESSION=SESSION
  USE
```

```
 ENDIF && W_TDAY<>' '

 * 'FROM' SESSION TO 'TO' SESSION
 SELECT ASSIGNPF
 APPEND BLANK
 REPLACE CLASS WITH G_CLASS, SLOT WITH P_TSLOT1, TCHER WITH W_FTCHER, ;
      ROOM  WITH W_FROOM, SUBJECT WITH W_FSUBJECT, DAY WITH W_TDAY, ;
      SESSION WITH W_TSESSION, TYPE WITH W_TYPE, TYPE1 WITH '1'
 IF (P_TSLOT2<>' ')
   APPEND BLANK
   REPLACE CLASS WITH G_CLASS, SLOT WITH P_TSLOT2, TCHER WITH W_FTCHER, ;
        ROOM  WITH W_FROOM, SUBJECT WITH W_FSUBJECT, DAY WITH W_TDAY, ;
        SESSION WITH STR(VAL(W_TSESSION)+1,2), TYPE WITH W_TYPE, TYPE1 WITH '2'
 ENDIF
 IF (P_TSLOT3<>' ')
   APPEND BLANK
   REPLACE CLASS WITH G_CLASS, SLOT WITH P_TSLOT3, TCHER WITH W_FTCHER, ;
        ROOM  WITH W_FROOM, SUBJECT WITH W_FSUBJECT, DAY WITH W_TDAY, ;
        SESSION WITH STR(VAL(W_TSESSION)+2,2), TYPE WITH W_TYPE, TYPE1 WITH '3'
 ENDIF
 PACK
 DO PRINTCLASS  && RE-DISPLAY THE CLASS
 RETURN  && SESSWITCH
 ** ----------------------------------------

 PROCEDURE CHECK
 PARAMETERS P_TR, P_TRVAL, P_SLOT, P_FOUND
 PRIVATE   W_COND
 * CHECK EXISTENCY OF RECORD ('T', TCHER, SLOT or 'R', ROOM, SLOT)

 SELECT ASSIGNPF
 SET DELETE ON
 IF (P_TR='T')
   W_COND="TCHER='"+P_TRVAL+"' .AND. SLOT='"+P_SLOT+"'"
 ELSE
   W_COND="ROOM='"+P_TRVAL+"' .AND. SLOT='"+P_SLOT+"'"
 ENDIF
 LOCATE FOR &W_COND
 P_FOUND=FOUND()
 SET DELETE OFF
 IF P_FOUND
   IF (P_TR='T')
     W_MSG="Teacher '"+ALLTRIM(P_TRVAL)+"' on Day "+ALLTRIM(DAY)+"-"+ ;
        ALLTRIM(SESSION)+" already assigned to Class '"+ALLTRIM(CLASS)+ ;
        "', Subject '"+ALLTRIM(SUBJECT)+"'"
     IF CLASS<>ROOM
       W_MSG=W_MSG+", Room '"+ALLTRIM(ROOM)+"'"
     ENDIF
   ELSE
      W_MSG="Room '"+ALLTRIM(P_TRVAL)+"' on Day "+ALLTRIM(DAY)+"-"+ALLTRIM(SESSION)+" already assigned
to"
     IF CLASS<>ROOM
       W_MSG=W_MSG+" Class '"+ALLTRIM(CLASS)+"',"
     ENDIF
     W_MSG=W_MSG+" Subject '"+ALLTRIM(SUBJECT)+"', Teacher '"+ALLTRIM(TCHER)+"'"
   ENDIF
 ENDIF  && P_FOUND
 RETURN  && CHECK
 ** ----------------------------------------

 PROCEDURE SWITCHROOM
 PRIVATE G_DAY, G_SESS, G_ROOM, W_MSG, W_TCHER, W_SUBJECT, W_SLOT, ;
      W_TYPE, W_SLOT1, W_SLOT2, W_SLOT3, W_FOUND, W_COND;
 **DO ROOM SWITCHING
 W_MSG=SPACE(80)
```

```
DO WHILE .T.
  CLOSE DATABASES
  IF (W_MSG<>SPACE(80))
    ? CHR(7)
  ENDIF
  IF (W_ASSPF='ASSTEMP')
    SET COLOR TO W+/N
    @ 0,0 SAY '(WORK FILE IN USE)'
    SET COLOR TO W/N
  ENDIF
  SET CURSOR ON
  G_DAY=' '
  G_SESS=' '
  G_ROOM=' '
  @22,0  CLEAR TO 24,79
  @22,0  SAY 'Room Switch : Day '
  @22,19 GET G_DAY PICTURE '99'
  @22,22 SAY 'Session '
  @22,30 GET G_SESS PICTURE '99'
  @22,36 SAY 'To New Room : '
  @22,51 GET G_ROOM PICTURE 'XX'
  @23,0  SAY 'F3=Cancel    F5=Session Switch   F9=Manually Assign   F10=Split'
  @24,0  SAY W_MSG
  L_FKEY=0
  READ   && L_FKEY: FUNCTION KEY #
  W_MSG=SPACE(80)
  DO CASE
    CASE (L_FKEY= 0) .OR. ;  && ENTER
       (L_FKEY= 7) .OR. ;  && ROOM SWITCH
       (L_FKEY=31) .OR. ;  && PAGEUP
       (L_FKEY=32)         && PAGEDOWN
       G_ROOM=UPPER(G_ROOM)
       G_DAY =RIGHT("  "+ALLTRIM(G_DAY),2)
       G_SESS=RIGHT("  "+ALLTRIM(G_SESS),2)
       G_ROOM=RIGHT("  "+ALLTRIM(G_ROOM),2)
       DO VALIDATE2 WITH G_DAY, G_SESS   && VALIDATE DAY,SESSION INPUTs
       IF (W_MSG<>SPACE(80))  && ERROR
         LOOP
       ENDIF
       IF (G_ROOM<>' ')
       DO VALIDATE3 WITH G_ROOM         && VALIDATE ROOM INPUTs
       IF (W_MSG<>SPACE(80)) && ERROR
         LOOP
       ENDIF
       ELSE
        W_MSG="Enter A Room Code."
        LOOP
       ENDIF

       W_TCHER=' '
       W_SUBJECT=' '
       W_TYPE=' '
       W_SLOT1=' '
       W_SLOT2=' '
       W_SLOT3=' '
       W_FOUND=.F.

       SELECT 1
       * CREATE WORK FILE IF NOT ALREADY EXIST
       USE
       IF (.NOT. FILE("ASSTEMP.DBF"))
         W_ASSPF="ASSTEMP"
         COPY FILE ASSIGNPF.DBF TO ASSTEMP.DBF
       ENDIF
```

```
USE &W_ASSPF ALIAS ASSIGNPF
LOCATE FOR CLASS=G_CLASS.AND.DAY=G_DAY.AND.SESSION=G_SESS
IF FOUND()
  W_TCHER=TCHER
  W_SUBJECT=SUBJECT
ELSE
  W_MSG=" Data: Class "+LTRIM(G_CLASS)+" on day "+LTRIM(G_DAY)+" Session "+ ;
      LTRIM(G_SESS)+" is not valid."
  LOOP
ENDIF

* ADJUST THE STARTING SESSION FOR REQUIRED CLASS BLOCK
DO SESSADJUST WITH G_DAY, G_SESS, W_TYPE, W_FOUND
IF (.NOT. W_FOUND)
  W_MSG="Given Day,"+LTRIM(G_DAY)+" Session "+LTRIM(G_SESS)+" and CLASSTYPE "+ ;
      LTRIM(W_TYPE)+" is not matched."
  LOOP
ENDIF
* GET SLOT#s (RETURN SLOT# OF AVAILABLE BLANK SLOTs IF DAY/SESSION IS BLANK)
DO GETSLOT WITH G_DAY, G_SESS, W_TYPE, W_SLOT1, W_SLOT2, W_SLOT3
IF (W_MSG<>SPACE(80))  && ERROR
  LOOP
ENDIF

IF (W_TYPE='D') .AND. (W_SLOT2=' ') .OR. ;
  (W_TYPE='T') .AND. (W_SLOT3=' ')
  W_MSG="Not enough empty slot(s) to assign the '"+W_TYPE+"' session subject"
  LOOP
ENDIF

W_FOUND=.F.
DO CHECK WITH "R", G_ROOM, W_SLOT1, W_FOUND
IF (W_FOUND)
  W_MSG=" Room "+LTRIM(G_ROOM)+" in Slot "+LTRIM(W_SLOT1)+" is Occupied."
  LOOP
ENDIF
IF (W_SLOT2<>' ')
  DO CHECK WITH "R", G_ROOM, W_SLOT2, W_FOUND
  IF (W_FOUND)
    W_MSG=" Room "+LTRIM(G_ROOM)+" in Slot "+LTRIM(W_SLOT2)+" is Occupied."
    LOOP
  ENDIF
ENDIF
IF (W_SLOT3<>' ')
  DO CHECK WITH "R", G_ROOM, W_SLOT3, W_FOUND
  IF (W_FOUND)
    W_MSG=" Room "+LTRIM(G_ROOM)+" in Slot "+LTRIM(W_SLOT3)+" is Occupied."
    LOOP
  ENDIF
ENDIF

IF (.NOT. W_FOUND)
W_SLOT=" "+W_SLOT1+" "+W_SLOT2+" "+W_SLOT3+" "
W_COND="CLASS='"+G_CLASS+"' .AND. ((SLOT+' ') $ '"+W_SLOT+"')"
DELETE ALL FOR &W_COND
ENDIF

&& ROOM SWITCH
SELECT ASSIGNPF
IF (W_SLOT1<>' ')
APPEND BLANK
REPLACE CLASS WITH G_CLASS, SLOT WITH W_SLOT1, TCHER WITH W_TCHER, ;
    ROOM  WITH G_ROOM, SUBJECT WITH W_SUBJECT, DAY WITH G_DAY, ;
    SESSION WITH STR(VAL(G_SESS),2), TYPE WITH W_TYPE, TYPE1 WITH '1'
ENDIF
```

```
        IF (W_SLOT2<>' ')
          APPEND BLANK
          REPLACE CLASS WITH G_CLASS, SLOT WITH W_SLOT2, TCHER WITH W_TCHER, ;
               ROOM  WITH G_ROOM, SUBJECT WITH W_SUBJECT, DAY WITH G_DAY, ;
               SESSION WITH STR(VAL(G_SESS)+1,2), TYPE WITH W_TYPE, TYPE1 WITH '2'
        ENDIF
        IF (W_SLOT3<>' ')
          APPEND BLANK
          REPLACE CLASS WITH G_CLASS, SLOT WITH W_SLOT3, TCHER WITH W_TCHER, ;
               ROOM  WITH G_ROOM, SUBJECT WITH W_SUBJECT, DAY WITH G_DAY, ;
               SESSION WITH STR(VAL(G_SESS)+2,2), TYPE WITH W_TYPE, TYPE1 WITH '3'
        ENDIF
        PACK
        DO PRINTCLASS

      CASE (L_FKEY=5)  && SESSION SWITCH
        IF (W_PREV_RTN='SWITCH')  && CALLER OF CURRENT INSTANT IS 'MANUALASGN'
          RETURN
        ENDIF
        W_PREV_RTN='SWITCHROOM'
        DO SWITCH

      CASE (L_FKEY=9)  && MANUALLY ASSIGN
        IF (W_PREV_RTN='MANUALASGN')  && CALLER OF CURRENT INSTANT IS 'MANUALASGN'
          RETURN
        ENDIF
        W_PREV_RTN='SWITCHROOM'
        DO MANUALASGN

      CASE (L_FKEY=10)  && SPLITTING MULTIPLE CLASS BLOCK
        IF (W_PREV_RTN='SPLITBLOCK')  && CALLER OF CURRENT INSTANT IS 'SPLITBLOCK'
          RETURN
        ENDIF
        W_PREV_RTN='SWITCHROOM'
        DO SPLITBLOCK

      CASE (L_FKEY=30)  && ESC
        @22,0 CLEAR TO 22,79
        EXIT

      CASE (L_FKEY=3)  && FUNCTION F3
        @22,0 CLEAR TO 22,79
        EXIT

      CASE (L_FKEY>=1) .AND. (L_FKEY<=9) && F1-F9
        W_MSG="Function Key 'F"+STR(L_FKEY,1)+"' not valid"
        LOOP

      CASE (L_FKEY>=10) .AND. (L_FKEY<=20) && F10-F20
        W_MSG="Function Key 'F"+STR(L_FKEY,2)+"' not valid"
        LOOP
      OTHERWISE
        W_MSG="Function Key not valid"
    ENDCASE
  ENDDO  && .T.

  @23, 0 SAY "ESC=Cancel   F5=Session Switch  F7=Room Switch  F9=Manually Assign"
  RETURN  && SWITCH ROOM
  ** ----------------------------------------

  PROCEDURE MANUALASGN
  PRIVATE G_SEQ, G_DAY, G_SESS, W_TYPE, W_SLOT1, W_SLOT2, W_SLOT3, ;
       W_SUBJECT, W_TCHER, W_ROOM, W_CLASSTYPE, W_FOUND, G_ROOM

  * DO MANUALLY ASSIGN OF SUBJECT TO EMPTY SLOT
```

```
W_MSG=SPACE(80)
DO WHILE .T.
  CLOSE DATABASES
  IF (W_MSG<>SPACE(80))
    ? CHR(7)
  ENDIF
  IF (W_ASSPF='ASSTEMP')
    SET COLOR TO W+/N
    @ 0,0 SAY '(WORK FILE IN USE)'
    SET COLOR TO W/N
  ENDIF
  SET CURSOR ON
  G_SEQ=0
  G_DAY='  '
  G_SESS='  '
  G_ROOM='  '
  @22,0  CLEAR TO 24,79
  @22,0  SAY 'Assign : Record '
  @22,16 GET G_SEQ PICTURE '@Z 9999'
  @22,21 SAY ' to  Day '
  @22,30 GET G_DAY PICTURE '99'
  @22,32 SAY ' Session '
  @22,41 GET G_SESS PICTURE '99'
  @22,43 SAY ' New Room '
  @22,55 GET G_ROOM PICTURE 'XX'
  @23,0  SAY 'F3=Cancel      F5=Session Switch   F7=Room Switch   F10=Split'
  @24,0  SAY W_MSG
  L_FKEY=0
  READ   && L_FKEY: FUNCTION KEY #

  W_MSG=SPACE(80)
  DO CASE
    CASE (L_FKEY=5)        && SWITCH SESSION
      IF (W_PREV_RTN='SWITCH')  && CALLER OF CURRENT INSTANT IS 'SWITCH'
        RETURN
      ENDIF
      W_PREV_RTN='MANUALASGN'
      DO SWITCH
    CASE (L_FKEY=7)        && SWITCH SESSION
      IF (W_PREV_RTN='SWITCHROOM')  && CALLER OF CURRENT INSTANT IS 'SWITCH'
        RETURN
      ENDIF
      W_PREV_RTN='MANUALASGN'
      DO SWITCHROOM
    CASE (L_FKEY=10)  && SPLITTING MULTIPLE CLASS BLOCK
      IF (W_PREV_RTN='SPLITBLOCK')  && CALLER OF CURRENT INSTANT IS 'SPLITBLOCK'
        RETURN
      ENDIF
      W_PREV_RTN='MANUALASGN'
      DO SPLITBLOCK
    CASE (L_FKEY= 0) .OR. ;  && ENTER
      (L_FKEY=31) .OR. ;  && PAGEUP
      (L_FKEY=32)       && PAGEDOWN
      G_DAY =RIGHT("  "+ALLTRIM(G_DAY),2)
      G_SESS=RIGHT("  "+ALLTRIM(G_SESS),2)
      SELECT 2
      USE
      IF (.NOT. FILE("CLSTEMP.DBF"))
        W_CLSPF='CLSTEMP'
        W_CLSL1='CLSTEMP1'
        COPY FILE CLSUBJPF.DBF TO CLSTEMP.DBF
        COPY FILE CLSUBJL1.NDX TO CLSTEMP1.NDX
      ENDIF
      USE &W_CLSPF ALIAS CLSUBJPF
```

```
 LOCATE FOR CLASS=G_CLASS .AND. SEQ=G_SEQ
 IF (.NOT. FOUND())
   W_MSG="Course '"+ALLTRIM(STR(G_SEQ))+"' not found in 'CLSUBJPF"'
   LOOP
 ENDIF
 IF (REPEAT<=FINISHED)
   W_MSG="Course '"+ALLTRIM(STR(G_SEQ))+"' already fully assigned"
   LOOP
 ENDIF
 W_SUBJECT=SUBJECT
 W_TCHER=TCHER

 IF (G_ROOM<>' ')
  G_ROOM=UPPER(G_ROOM)
  W_ROOM=G_ROOM
 ELSE
  W_ROOM=ROOM
 ENDIF
 IF (W_ROOM=' ')
   W_ROOM=CLASS
 ENDIF
 W_CLASSTYPE=CLASSTYPE
 IF (G_ROOM<>' ')
 DO VALIDATE3 WITH G_ROOM        && VALIDATE ROOM INPUTs
  IF (W_MSG<>SPACE(80)) && ERROR
    LOOP
  ENDIF
 ENDIF
 DO VALIDATE2 WITH G_DAY, G_SESS   && VALIDATE INPUTs
 IF (W_MSG<>SPACE(80))  && ERROR
   LOOP
 ENDIF

 SELECT 1
 * CREATE WORK FILE IF NOT ALREADY EXIST
 USE
 IF (.NOT. FILE("ASSTEMP.DBF"))
   W_ASSPF="ASSTEMP"
   COPY FILE ASSIGNPF.DBF TO ASSTEMP.DBF
 ENDIF
 USE &W_ASSPF ALIAS ASSIGNPF
 LOCATE FOR CLASS=G_CLASS .AND. DAY=G_DAY .AND. SESSION=G_SESS
 IF FOUND()
   W_MSG='Slot on Day '+ALLTRIM(G_DAY)+'-'+ALLTRIM(G_SESS)+" for Class '"+G_CLASS+"' already assigned"
   LOOP
 ENDIF

 W_TYPE=' '
 IF (W_CLASSTYPE<>' ')
   W_TYPE=W_CLASSTYPE
 ENDIF
 W_SLOT1=' '
 W_SLOT2=' '
 W_SLOT3=' '
 * GET SLOT#s (RETURN SLOT# OF AVAILABLE BLANK SLOTs IF DAY/SESSION IS BLANK)
 DO GETSLOT WITH G_DAY, G_SESS, W_TYPE, W_SLOT1, W_SLOT2, W_SLOT3
 IF (W_MSG<>SPACE(80))  && ERROR
   LOOP
 ENDIF

 IF (W_CLASSTYPE='D') .AND. (W_SLOT2=' ') .OR. ;
   (W_CLASSTYPE='T') .AND. (W_SLOT3=' ')
   W_MSG="Not Enough Empty Slot(s) to Assign the '"+W_CLASSTYPE+"' session subject"
   LOOP
 ENDIF
```

```
      W_FOUND=.F.
      DO CHECK WITH "T", W_TCHER, W_SLOT1, W_FOUND
      IF W_FOUND
        LOOP
      ENDIF
      IF (W_SLOT2<>' ')
        DO CHECK WITH "T", W_TCHER, W_SLOT2, W_FOUND
        IF W_FOUND
          LOOP
        ENDIF
      ENDIF
      IF (W_SLOT3<>' ')
        DO CHECK WITH "T", W_TCHER, W_SLOT3, W_FOUND
        IF W_FOUND
          LOOP
        ENDIF
      ENDIF
      DO CHECK WITH "R", W_ROOM, W_SLOT1, W_FOUND
      IF W_FOUND
        LOOP
      ENDIF
      IF (W_SLOT2<>' ')
        DO CHECK WITH "R", W_ROOM, W_SLOT2, W_FOUND
        IF W_FOUND
          LOOP
        ENDIF
      ENDIF
      IF (W_SLOT3<>' ')
        DO CHECK WITH "R", W_ROOM, W_SLOT3, W_FOUND
        IF W_FOUND
          LOOP
        ENDIF
      ENDIF

      && DO THE ASSIGNMENT
      SELECT CLSUBJPF
      REPLACE FINISHED WITH FINISHED+1 FOR CLASS=G_CLASS .AND. SEQ=G_SEQ
      USE
      SELECT ASSIGNPF
      APPEND BLANK
      REPLACE CLASS WITH G_CLASS, SLOT WITH W_SLOT1, TCHER WITH W_TCHER, ;
           ROOM  WITH W_ROOM, SUBJECT WITH W_SUBJECT, DAY WITH G_DAY, ;
           SESSION WITH G_SESS, TYPE WITH W_CLASSTYPE, TYPE1 WITH '1'
      IF (W_SLOT2<>' ')
        APPEND BLANK
        REPLACE CLASS WITH G_CLASS, SLOT WITH W_SLOT2, TCHER WITH W_TCHER, ;
             ROOM  WITH W_ROOM, SUBJECT WITH W_SUBJECT, DAY WITH G_DAY, ;
             SESSION WITH STR(VAL(G_SESS)+1,2), TYPE WITH W_CLASSTYPE, TYPE1 WITH '2'
      ENDIF
      IF (W_SLOT3<>' ')
        APPEND BLANK
        REPLACE CLASS WITH G_CLASS, SLOT WITH W_SLOT3, TCHER WITH W_TCHER, ;
             ROOM  WITH W_ROOM, SUBJECT WITH W_SUBJECT, DAY WITH G_DAY, ;
             SESSION WITH STR(VAL(G_SESS)+2,2), TYPE WITH W_CLASSTYPE, TYPE1 WITH '3'
      ENDIF
      DO PRINTCLASS
    CASE (L_FKEY=30) .OR. (L_FKEY=3)  && ESC and F3
      @22,0 CLEAR TO 22,79
      EXIT
    CASE (L_FKEY>=1) .AND. (L_FKEY<=20) && F1-F20
      W_MSG="Function Key 'F"+LTRIM(STR(L_FKEY,2))+"' not valid"
      LOOP
    OTHERWISE
      W_MSG="Function Key not Valid"
  ENDCASE
```

```
ENDDO  && .T.
@23, 0 SAY "ESC=Cancel    F5=Session Switch   F7=Room Switch   F9=Manually Assign"
RETURN  && MANUALASGN
** ----------------------------------------

PROCEDURE SPLITBLOCK
PRIVATE G_DAY, G_SESS, G_CONFIRM, W_MSG, W_TCHER, W_ROOM, W_SUBJECT, W_SLOT, ;
     W_TYPE, W_SLOT1, W_SLOT2, W_SLOT3, W_FOUND, W_COND;
**DO ROOM SWITCHING
W_MSG=SPACE(80)
DO WHILE .T.

 CLOSE DATABASES

 IF (W_MSG<>SPACE(80))
   ? CHR(7)
 ENDIF
 IF (W_ASSPF='ASSTEMP')
   SET COLOR TO W+/N
   @ 0,0 SAY '(WORK FILE IN USE)'
   SET COLOR TO W/N
 ENDIF
 SET CURSOR ON
 G_DAY=' '
 G_SESS=' '
 G_CONFIRM=' '
 @22,0  CLEAR TO 24,79
 @22,0  SAY 'Split Block : Day '
 @22,19 GET G_DAY PICTURE '99'
 @22,22 SAY 'Session '
 @22,30 GET G_SESS PICTURE '99'
 @23,0  SAY 'F3=Cancel    F5=Session Switch   F7=Room Switch   F9=Manually Assign'
 @24,0  SAY W_MSG
 L_FKEY=0
 READ  && L_FKEY: FUNCTION KEY #
 W_MSG=SPACE(80)
 DO CASE
   CASE (L_FKEY= 0) .OR. ;  && ENTER
      (L_FKEY=10) .OR. ;  && ROOM SWITCH
      (L_FKEY=31) .OR. ;  && PAGEUP
      (L_FKEY=32)       && PAGEDOWN
      G_ROOM=UPPER(G_ROOM)
      G_DAY =RIGHT("  "+ALLTRIM(G_DAY),2)
      G_SESS=RIGHT("  "+ALLTRIM(G_SESS),2)
      G_ROOM=RIGHT("  "+ALLTRIM(G_ROOM),2)

      DO VALIDATE2 WITH G_DAY, G_SESS   && VALIDATE DAY,SESSION INPUTs
      IF (W_MSG<>SPACE(80))  && ERROR
        LOOP
      ENDIF
      W_TCHER=' '
      W_ROOM=' '
      W_SUBJECT=' '
      W_TYPE=' '
      W_SLOT1=' '
      W_SLOT2=' '
      W_SLOT3=' '
      W_FOUND=.F.
      SELECT 1
      * CREATE WORK FILE IF NOT ALREADY EXIST
      USE
      IF (.NOT. FILE("ASSTEMP.DBF"))
        W_ASSPF="ASSTEMP"
        COPY FILE ASSIGNPF.DBF TO ASSTEMP.DBF
      ENDIF
```

```
USE &W_ASSPF ALIAS ASSIGNPF
LOCATE FOR CLASS=G_CLASS.AND.DAY=G_DAY.AND.SESSION=G_SESS
IF FOUND()
  W_TCHER=TCHER
  W_ROOM=ROOM
  W_SUBJECT=SUBJECT
ELSE
  W_MSG=" Data: Class "+LTRIM(G_CLASS)+" on day "+LTRIM(G_DAY)+" Session "+ ;
      LTRIM(G_SESS)+" is not valid."
  LOOP
ENDIF

* ADJUST THE STARTING SESSION FOR REQUIRED CLASS BLOCK
DO SESSADJUST WITH G_DAY, G_SESS, W_TYPE, W_FOUND
IF (.NOT. W_FOUND)
  W_MSG="Given Day,"+LTRIM(G_DAY)+" Session "+LTRIM(G_SESS)+" and CLASSTYPE "+ ;
      LTRIM(W_TYPE)+" is not matched."
  LOOP
ENDIF
* GET SLOT#s (RETURN SLOT# OF AVAILABLE BLANK SLOTs IF DAY/SESSION IS BLANK)
DO GETSLOT WITH G_DAY, G_SESS, W_TYPE, W_SLOT1, W_SLOT2, W_SLOT3
IF (W_MSG<>SPACE(80))  && ERROR
  LOOP
ENDIF

IF (W_TYPE='S')
  W_MSG="Already a Single Session."
  LOOP
ENDIF

IF (W_TYPE='D') .AND. (W_SLOT2='  ') .OR. ;
  (W_TYPE='T') .AND. (W_SLOT3='  ')
  W_MSG="Not Enough Empty Slot(s) to Assign the '"+W_TYPE+"' Session subject"
  LOOP
ENDIF

IF (.NOT. W_FOUND)
  W_SLOT=" "+W_SLOT1+" "+W_SLOT2+" "+W_SLOT3+" "
  W_COND="CLASS='"+G_CLASS+"' .AND. ((SLOT+' ') $ '"+W_SLOT+"')"
  DELETE ALL FOR &W_COND
ENDIF

&& CLASS BLOCK SPLIT
SELECT ASSIGNPF
IF (W_SLOT1<>'  ')
APPEND BLANK
REPLACE CLASS WITH G_CLASS, SLOT WITH W_SLOT1, TCHER WITH W_TCHER, ;
     ROOM  WITH W_ROOM, SUBJECT WITH W_SUBJECT, DAY WITH G_DAY, ;
     SESSION WITH STR(VAL(G_SESS),2), TYPE WITH 'S', TYPE1 WITH '1'
ENDIF
IF (W_SLOT2<>'  ')
  APPEND BLANK
  REPLACE CLASS WITH G_CLASS, SLOT WITH W_SLOT2, TCHER WITH W_TCHER, ;
       ROOM  WITH W_ROOM, SUBJECT WITH W_SUBJECT, DAY WITH G_DAY, ;
       SESSION WITH STR(VAL(G_SESS)+1,2), TYPE WITH 'S', TYPE1 WITH '2'
ENDIF
IF (W_SLOT3<>'  ')
  APPEND BLANK
  REPLACE CLASS WITH G_CLASS, SLOT WITH W_SLOT3, TCHER WITH W_TCHER, ;
       ROOM  WITH W_ROOM, SUBJECT WITH W_SUBJECT, DAY WITH G_DAY, ;
       SESSION WITH STR(VAL(G_SESS)+2,2), TYPE WITH 'S', TYPE1 WITH '3'
ENDIF
PACK
DO PRINTCLASS
```

```
    CASE (L_FKEY=5)  && SESSION SWITCH
       IF (W_PREV_RTN='SWITCH')  && CALLER OF CURRENT INSTANT IS 'MANUALASGN'
         RETURN
       ENDIF
       W_PREV_RTN='SPLITBLOCK'
       DO SWITCH
    CASE (L_FKEY=7)  && SWITCH SESSION
       IF (W_PREV_RTN='SWITCHROOM')  && CALLER OF CURRENT INSTANT IS 'SWITCH'
         RETURN
       ENDIF
       W_PREV_RTN='SPLITBLOCK'
       DO SWITCHROOM
    CASE (L_FKEY=9)  && MANUALLY ASSIGN
       IF (W_PREV_RTN='MANUALASGN')  && CALLER OF CURRENT INSTANT IS 'MANUALASGN'
         RETURN
       ENDIF
       W_PREV_RTN='SPLITBLOCK'
       DO MANUALASGN
    CASE (L_FKEY=30)  && ESC
       @22,0 CLEAR TO 22,79
       EXIT
    CASE (L_FKEY=3)   && FUNCTION F3
       @22,0 CLEAR TO 22,79
       EXIT
    CASE (L_FKEY>=1) .AND. (L_FKEY<=9) && F1-F9
       W_MSG="Function Key 'F"+STR(L_FKEY,1)+"' not valid"
       LOOP
    CASE (L_FKEY>=10) .AND. (L_FKEY<=20) && F10-F20
       W_MSG="Function Key 'F"+STR(L_FKEY,2)+"' not valid"
       LOOP
    OTHERWISE
       W_MSG="Function Key not valid"
  ENDCASE
ENDDO  && .T.

@23, 0 SAY "ESC=Cancel    F5=Session Switch   F7=Room Switch   F9=Manually Assign"
RETURN  && SPLIT CLASS BLOCK
** ----------------------------------------

PROCEDURE VALIDATE2
PARAMETERS P_DAY, P_SESS
* VALIDATE A DAY AND A SESSION
IF (VAL(P_DAY)<1) .OR. (VAL(P_DAY)>D_DAY)
  W_MSG="Day Entered Must Be Between 1 and "+STR(D_DAY,2)
  RETURN
ENDIF  && P_DAY<1 OR D_DAY>D_DAY
IF (VAL(P_SESS)<1) .OR. (VAL(P_SESS)>D_SESSION)
  W_MSG="Session Entered Must Be Between 1 and "+STR(D_SESSION,2)
  RETURN
ENDIF  && P_SESS<1 OR P_SESS<D_SESSION
RETURN  && VALIDATE2
** ----------------------------------------

PROCEDURE VALIDATE3
PARAMETERS P_ROOM
* VALIDATE A ROOM
SELECT 3
USE ROOMPF
LOCATE FOR ROOM=P_ROOM
IF (.NOT. FOUND())
  W_MSG="Room Provided Not Found."
  RETURN
ENDIF && CHECK GIVEN P_ROOM
RETURN && VALIDATE3
```

APPENDIX E4

**** PROGRAM NAME : TTPRINT.PRG**
**** LAST CHANGED : AUGUST 8, 1994**
**** DESCRIPTION : TIME TABLE - PRINTING MODULE**

```
set talk off
set bell off
set scoreboard off
set exact on
set confirm on

RESTORE FROM TTSETTNG ADDITIVE
W_SPACE=SPACE(9*D_DAY) && D_DAY: # OF DAYS IN THE TIMETABLE
W_LASTSESS=STR(D_SESSION+1,2)
W_LINE1=W_SPACE
W_LINE2=W_SPACE
W_LINE3=W_SPACE
W_RECESS=SPACE((9*D_DAY-12)/2)
W_RECESS=W_RECESS+"B R E A K"+W_RECESS
W_TCHTBL=SPACE((9*D_DAY-20)/2)
W_TCHTBL=W_TCHTBL+"Instructor Time Table"
W_CLSTBL=SPACE((9*D_DAY-16)/2)
W_CLSTBL=W_CLSTBL+"Class Time Table"
W_ROMTBL=SPACE((9*D_DAY-20)/2)
W_ROMTBL=W_ROMTBL+"Classroom Time Table"
W_EOFLST=SPACE((9*D_DAY-18)/2)
W_EOFLST=W_EOFLST+"*** END OF LIST ***"
W_DTITLE=""
W_DAY=0
DO WHILE (W_DAY<D_DAY)
  W_DAY=W_DAY+1
  W_DTITLE=W_DTITLE+"  Day "+STR(W_DAY,2)+" "
ENDDO  && DO WHILE (W_DAY<D_DAY)
W_DASH=REPLICATE("-",9*D_DAY)

W_MSG=SPACE(80)

DO WHILE .T.
  G_CLASS="  "
  G_TCHER="  "
  G_ROOM="  "

  SET DEVICE TO SCREEN
  @ 0,0 SAY " "
  IF (W_MSG<>SPACE(80))
    ? CHR(7)
  ENDIF
  SET CURSOR ON
  CLEAR
  SET COLOR TO N/W
  @ 0,30 SAY "TIME TABLE - PRINT"
  SET COLOR TO W/N
  @ 0,72 SAY DATE()
  @ 1,72 SAY TIME()
  @ 5,30 SAY "CLASS  : " GET G_CLASS PICTURE 'XX'
```

```
 @ 7,30 SAY "TEACHER: " GET G_TCHER PICTURE 'XX'
 @ 9,30 SAY "ROOM   : " GET G_ROOM  PICTURE 'XX'
 @17, 0 SAY "- use '*' as wildcard character"
 @18, 0 SAY "  e.g. enter '4*' to match all codes start with '4'"
 @19, 0 SAY "       enter '* ' to match all codes (i.e. PRINT ALL)"
 @20, 0 SAY "- e.g. to print Time Table for all Form 4 classes, for all teachers, and for"
 @21, 0 SAY "       all special rooms, please enter '4*', '* ' and '* ' respectively"
 @23, 0 SAY "ESC=Cancel"
 @24, 0 CLEAR TO 24,80
 @24, 0 SAY W_MSG

 READ

 W_READKEY=READKEY(0)
 W_MSG=SPACE(80)

* ESC KEY PRESSED DURING 'READ' (268=FIELD UPDATED; 12=FIELD NOT UPDATED)
 IF (W_READKEY=12) .OR. (W_READKEY=268)

   RETURN

 ENDIF  && W_READKEY=12/268 == ESC

 IF (G_CLASS="  ") .AND. (G_TCHER="  ") .AND. (G_ROOM="  ")

   LOOP  && BLANK INPUT, LOOP AGAIN

 ENDIF  && G_CLASS=' ' & G_TCHER=' ' & G_ROOM=' '


 W_OK=.T.
 IF (G_CLASS<>"  ")
   G_CLASS=UPPER(G_CLASS)
   W_POS=AT("*",G_CLASS)
   DO CASE
     CASE G_CLASS="* " .OR. G_CLASS=" *" .OR. G_CLASS="**"
        W_FILTER1=" "
     CASE W_POS=1  && G_CLASS="*?"
        W_FILTER1="SUBSTR(CLASS,2,1)=SUBSTR("'+G_CLASS+"',2,1)"
     CASE W_POS=2  && G_CLASS="?*"
        W_FILTER1="SUBSTR(CLASS,1,1)=SUBSTR("'+G_CLASS+"',1,1)"
     OTHERWISE
        W_FILTER1="CLASS="'+G_CLASS+""
   ENDCASE

   IF (W_FILTER1<>" ")
     SELECT 1
     USE ASSIGNPF
     LOCATE FOR &W_FILTER1
     IF (.NOT. FOUND())
        W_OK=.F.
       W_MSG="Class code "'+G_CLASS+"' not found in assignment file"
       LOOP  && ERROR, LOOP AGAIN
     ENDIF  && .NOT. FOUND()
   ENDIF  && W_FILTER1<>" "
 ENDIF  && G_CLASS<>' '


 IF (G_TCHER<>"  ") .AND. W_OK
   G_TCHER=UPPER(G_TCHER)

   W_POS=AT("*",G_TCHER)
   DO CASE
     CASE G_TCHER="* " .OR. G_TCHER=" *" .OR. G_TCHER="**"
        W_FILTER2=" "
```

```
    CASE W_POS=1  && G_TCHER="*?"
       W_FILTER2="SUBSTR(TCHER,2,1)=SUBSTR('"+G_TCHER+"',2,1)"
    CASE W_POS=2  && G_TCHER="?*"
       W_FILTER2="SUBSTR(TCHER,1,1)=SUBSTR('"+G_TCHER+"',1,1)"
    OTHERWISE
       W_FILTER2="TCHER='"+G_TCHER+"'"
  ENDCASE

  IF (W_FILTER2<>" ")
    SELECT 1
    USE ASSIGNPF
    LOCATE FOR &W_FILTER2
    IF (.NOT. FOUND())
      W_OK=.F.
      W_MSG="Teacher code '"+G_TCHER+"' not found in assignment file"
      LOOP  && ERROR, LOOP AGAIN
    ENDIF  && .NOT. FOUND()
  ENDIF  && W_FILTER2<>" "
ENDIF  && G_TCHER<>' '


IF (G_ROOM<>"  ")  .AND. W_OK
  G_ROOM=UPPER(G_ROOM)

  W_POS=AT("*",G_ROOM)
  DO CASE
    CASE G_ROOM="* " .OR. G_ROOM=" *" .OR. G_ROOM="**"
       W_FILTER3=" "
    CASE W_POS=1  && G_ROOM="*?"
       W_FILTER3="SUBSTR(ROOM,2,1)=SUBSTR('"+G_ROOM+"',2,1)"
    CASE W_POS=2  && G_ROOM="?*"
       W_FILTER3="SUBSTR(ROOM,1,1)=SUBSTR('"+G_ROOM+"',1,1)"
    OTHERWISE
       W_FILTER3="ROOM='"+G_ROOM+"'"
  ENDCASE

  IF (W_FILTER3<>" ")
    SELECT 1
    USE ASSIGNPF
    LOCATE FOR &W_FILTER3
    IF (.NOT. FOUND())
      W_OK=.F.
      W_MSG="Room code '"+G_ROOM+"' not found in assignment file"
      LOOP  && ERROR, LOOP AGAIN
    ENDIF  && .NOT. FOUND()
  ENDIF  && W_FILTER3<>" "
ENDIF  && G_ROOM<>' '

IF W_OK
  W_MSG='Printing in progress...'
  @24, 0 CLEAR TO 24,80
  @24, 0 SAY W_MSG


   SET PRINTER TO &D_LPT
  SET DEVICE TO PRINTER
  SET CURSOR OFF

  IF (G_CLASS<>"  ")
    SELECT 1
    USE ASSIGNPF
    IF (W_FILTER1<>" ")
      SET FILTER TO &W_FILTER1
    ENDIF
    INDEX ON CLASS+SESSION+DAY TO TEMP
```

```
      SET INDEX TO TEMP
      SELECT 2
      USE CLSUBJPF INDEX CLSUBJL1
      IF (W_FILTER1<>" ")
        SET FILTER TO &W_FILTER1
      ENDIF

      DO PRINTCLASS

    ENDIF  && G_CLASS<>" "

    IF (G_TCHER<>" ")
      SELECT 1
      USE ASSIGNPF
      IF (W_FILTER2<>" ")
        SET FILTER TO &W_FILTER2
      ENDIF
      INDEX ON TCHER+SESSION+DAY TO TEMP
      SET INDEX TO TEMP

      DO PRINTTCHER

    ENDIF  && G_TCHER<>" "

    IF (G_ROOM<>" ")
      SELECT 1
      USE ASSIGNPF
      IF (W_FILTER3<>" ")
         SET FILTER TO &W_FILTER3
      ENDIF
      INDEX ON ROOM+SESSION+DAY TO TEMP
      SET INDEX TO TEMP

      DO PRINTROOM

    ENDIF  && G_ROOM<>" "

  ENDIF  && W_OK

  W_MSG="Printing finished"
  CLOSE DATABASES
ENDDO  && .T.

W_STATUS=MEMOWRIT("TTNXTPGM.BAT",D_HIDE+"TTMAIN.EXE")
CLOSE ALL
ERASE TEMP.NDX
SET CURSOR ON
RETURN  && MAIN LINE


** ----------------------------------------

PROCEDURE PRINTCLASS
PRIVATE W_PRVPRTSS, W_PRVCLASS, W_PRVSESS, W_COL, W_SABBREV, W_TABBREV, ;
     W_RABBREV, W_LINE1, W_LINE2, W_LINE3

* PRINT OUT TIMETABLE FOR ALL SPECIFIED CLASSES AND LIST OF ALL
* UNALLOCATED SUBJECTS FOLLOWING THE CLASS

W_LINE1=W_SPACE
W_LINE2=W_SPACE
W_LINE3=W_SPACE


SELECT 3
```

```
USE CLASSPF
SELECT 4
USE SUBJPF
SELECT 5
USE TCHERPF
SELECT 7
USE ROOMPF
SELECT ASSIGNPF
GO TOP

IF (.NOT. EOF())
  W_PRVPRTSS="00"  && SESSION PREVIOUSLY PRINTED
  W_PRVCLASS=CLASS
  W_PRVSESS=SESSION
  DO NEWCLASS WITH W_PRVCLASS
  SELECT ASSIGNPF
ENDIF
DO WHILE (.NOT. EOF())
  IF (ASSIGNPF->CLASS<>W_PRVCLASS)
    DO BLANKSESS  WITH W_PRVPRTSS, W_PRVSESS
    DO PRTSESSION WITH W_PRVSESS, W_LINE1, W_LINE2, W_LINE3
    DO BLANKSESS  WITH W_PRVSESS, W_LASTSESS
    DO PRTUNALLOC WITH W_PRVCLASS
    DO ENDOFLIST  WITH .F.
    SELECT ASSIGNPF

    W_PRVPRTSS="00"
    W_PRVSESS=SESSION
    W_PRVCLASS=CLASS

    DO NEWCLASS WITH W_PRVCLASS

    W_LINE1=W_SPACE
    W_LINE2=W_SPACE
    W_LINE3=W_SPACE
  ENDIF  && (CLASS<>W_PRVCLASS)

  IF (ASSIGNPF->SESSION<>W_PRVSESS)  && SESSION CHANGED
    DO BLANKSESS  WITH W_PRVPRTSS, W_PRVSESS  && PRINT ANY BLANK SESSION IN BETWEEN
    DO PRTSESSION WITH W_PRVSESS, W_LINE1, W_LINE2, W_LINE3
    W_PRVPRTSS=W_PRVSESS
    W_LINE1=W_SPACE
    W_LINE2=W_SPACE
    W_LINE3=W_SPACE
    W_PRVSESS=ASSIGNPF->SESSION
  ENDIF  && (SESSION<>W_PRVSESS)

  W_COL=3+9*(VAL(ASSIGNPF->DAY)-1)
  SELECT SUBJPF
  LOCATE FOR SUBJECT=ASSIGNPF->SUBJECT
  IF FOUND()
    W_SABBREV=ABBREV
  ELSE
    W_SABBREV=SUBJECT+"  "
  ENDIF
  SELECT TCHERPF
  LOCATE FOR TCHER=ASSIGNPF->TCHER
  IF FOUND()
    W_TABBREV=ABBREV
  ELSE
    W_TABBREV=TCHER+"  "
  ENDIF
  SELECT ROOMPF
  LOCATE FOR ROOM=ASSIGNPF->ROOM
  IF FOUND()
```

```
    W_RABBREV=ABBREV
  ELSE
    W_RABBREV=ROOM+"  "
  ENDIF

  SELECT ASSIGNPF
  W_LINE1=STUFF(W_LINE1,W_COL,5,W_SABBREV)
  W_LINE2=STUFF(W_LINE2,W_COL,5,W_TABBREV)
  W_LINE3=STUFF(W_LINE3,W_COL,5,W_RABBREV)
  SKIP
ENDDO  && (.NOT. EOF())

DO BLANKSESS  WITH W_PRVPRTSS, W_PRVSESS
DO PRTSESSION WITH W_PRVSESS, W_LINE1, W_LINE2, W_LINE3  && LAST LINE
DO BLANKSESS  WITH W_PRVSESS, W_LASTSESS
DO PRTUNALLOC WITH W_PRVCLASS
DO ENDOFLIST  WITH .F.
RETURN  && PRINTCLASS
** ----------------------------------------

PROCEDURE NEWCLASS
PARAMETERS P_CLASS
PRIVATE W_DESC

** PRINT TITLE FOR NEW CLASS

DO CHKPRINTER
SELECT CLASSPF
LOCATE FOR CLASS=P_CLASS
W_DESC=" "
IF FOUND()
  W_DESC=DESC
ENDIF

@0,5 SAY W_CLSTBL
@0,50 SAY DATE()
@1,50 SAY TIME()
@2,0  SAY "Class : "+"["+P_CLASS+"]"+"  "+W_DESC
@4,5  SAY W_DTITLE
@5,5  SAY W_DASH
RETURN  && NEWCLASS
** ----------------------------------------

PROCEDURE  PRTUNALLOC
PARAMETERS P_CLASS
PRIVATE W_FILTER, W_LINE1, W_LINE2

** PRINT OUT THE LIST OF SUBJECTS IN THE CLASS WHICH ARE NOT ABLE TO ALLOCATE

DO CHKPRINTER
SELECT 6
USE
W_FILTER="REPEAT>FINISHED .AND. CLASS="'+P_CLASS+'""
USE CLSUBJPF INDEX CLSUBJL1
SET FILTER TO &W_FILTER
GO TOP
IF EOF()
  USE
  RETURN  && NO UNALLOCATED CLASS
ENDIF

@ PROW()+2, 5 SAY W_DASH
@ PROW()+3, 5 SAY "List of Unallocated Subject(s):"
@ PROW()+2, 5 SAY "       Class                Number of"
@ PROW()+1, 5 SAY "Subject  Type  Teacher  Room  unallocated session"
```

```
@ PROW()+1, 5 SAY "----------------------------------------------------"
@ PROW()+1, 5 SAY " "

DO WHILE (.NOT. EOF())
  SELECT SUBJPF
  LOCATE FOR SUBJECT=CLSUBJPF->SUBJECT
  IF FOUND()
    W_LINE1=ABBREV
  ELSE
    W_LINE1=SUBJECT+"  "
  ENDIF  && FOUND()

  SELECT TCHERPF
  LOCATE FOR TCHER=CLSUBJPF->TCHER
  IF FOUND()
    W_LINE2=ABBREV
  ELSE
    W_LINE2=TCHER+"  "
  ENDIF  && FOUND()

  DO CHKPRINTER
  SELECT CLSUBJPF
  @ PROW()+1, 5 SAY W_LINE1
  @ PROW()  ,16 SAY CLASSTYPE
  @ PROW()  ,20 SAY W_LINE2
  @ PROW()  ,29 SAY ROOM
  @ PROW()  ,42 SAY REPEAT-FINISHED PICTURE "99"

  SKIP
ENDDO  && DO WHILE .NOT. EOF()
@ PROW()+2, 5 SAY "----------------------------------------------------"

SELECT CLSUBJPF
USE
RETURN  && PRTUNALLOC
** ----------------------------------------

PROCEDURE PRINTTCHER
PRIVATE W_PRVPRTSS, W_PRVTCHER, W_PRVSESS, W_COL, W_SABBREV, W_CABBREV, ;
     W_RABBREV, W_LINE1, W_LINE2, W_LINE3

* PRINT OUT TIMETABLE FOR ALL SPECIFIED TEACHERS

W_LINE1=W_SPACE
W_LINE2=W_SPACE
W_LINE3=W_SPACE

SELECT 3
USE CLASSPF
SELECT 4
USE SUBJPF
SELECT 5
USE TCHERPF
SELECT 7
USE ROOMPF
SELECT ASSIGNPF
GO TOP

IF (.NOT. EOF())
  W_PRVPRTSS="00"  && SESSION PREVIOUSLY PRINTED
  W_PRVTCHER=TCHER
  W_PRVSESS=SESSION
  DO NEWTCHER WITH W_PRVTCHER
  SELECT ASSIGNPF
ENDIF
```

```
DO WHILE (.NOT. EOF())
  IF (ASSIGNPF->TCHER<>W_PRVTCHER)  && ROOM CHANGED
    DO BLANKSESS  WITH W_PRVPRTSS, W_PRVSESS
    DO PRTSESSION WITH W_PRVSESS, W_LINE1, W_LINE2, W_LINE3
    DO BLANKSESS  WITH W_PRVSESS, W_LASTSESS  && PRINT BLANK SESSION TILL LAST
    DO ENDOFLIST  WITH .T.
     SELECT ASSIGNPF

    W_PRVPRTSS="00"
    W_PRVTCHER=TCHER
    W_PRVSESS=SESSION

    DO NEWTCHER WITH W_PRVTCHER
    SELECT ASSIGNPF

    W_LINE1=W_SPACE
    W_LINE2=W_SPACE
    W_LINE3=W_SPACE
  ENDIF  && (TCHER<>W_PRVTCHER)

  IF (ASSIGNPF->SESSION<>W_PRVSESS)  && SESSION CHANGED
    DO BLANKSESS WITH W_PRVPRTSS, W_PRVSESS  && PRINT ANY BLANK SESSION IN BETWEEN
    DO PRTSESSION WITH W_PRVSESS, W_LINE1, W_LINE2, W_LINE3
    W_PRVPRTSS=W_PRVSESS
    W_LINE1=W_SPACE
    W_LINE2=W_SPACE
    W_LINE3=W_SPACE
    W_PRVSESS=ASSIGNPF->SESSION
  ENDIF  && (SESSION<>W_PRVSESS)

  W_COL=3+9*(VAL(ASSIGNPF->DAY)-1)
  SELECT SUBJPF
  LOCATE FOR SUBJECT=ASSIGNPF->SUBJECT
  IF FOUND()
    W_SABBREV=ABBREV
  ELSE
    W_SABBREV=SUBJECT+"  "
  ENDIF
  SELECT CLASSPF
  LOCATE FOR CLASS=ASSIGNPF->CLASS
  IF FOUND()
    W_CABBREV=ABBREV
  ELSE
    W_CABBREV=ROOM+"  "
  ENDIF
  SELECT ROOMPF
  LOCATE FOR ROOM=ASSIGNPF->ROOM
  IF FOUND()
    W_RABBREV=ABBREV
  ELSE
    W_RABBREV=ROOM+"  "
  ENDIF

  SELECT ASSIGNPF
  W_LINE1=STUFF(W_LINE1,W_COL,5,W_SABBREV)
  W_LINE2=STUFF(W_LINE2,W_COL,5,W_CABBREV)
  W_LINE3=STUFF(W_LINE3,W_COL,5,W_RABBREV)
  SKIP
ENDDO  && (.NOT. EOF())

DO BLANKSESS  WITH W_PRVPRTSS, W_PRVSESS
DO PRTSESSION WITH W_PRVSESS, W_LINE1, W_LINE2, W_LINE3  && LAST LINE
DO BLANKSESS  WITH W_PRVSESS, W_LASTSESS
DO ENDOFLIST  WITH .T.
RETURN  && PRINTTCHER
```

```
** ----------------------------------------

PROCEDURE NEWTCHER
PARAMETERS P_TCHER
PRIVATE W_DESC

** PRINT TITLE FOR NEW TEACHER

DO CHKPRINTER
SELECT TCHERPF
LOCATE FOR TCHER=P_TCHER
W_DESC=" "
IF FOUND()
  W_DESC=DESC
ENDIF

@0,5 SAY W_TCHTBL
@0,50 SAY DATE()
@1,50 SAY TIME()
@2,0  SAY "Instructor : "+"["+P_TCHER+"]"+"  "+W_DESC
@4,5  SAY W_DTITLE
@5,5  SAY W_DASH
RETURN  && NEWTCHER
** ----------------------------------------

PROCEDURE PRINTROOM
* PRINT OUT TIMETABLE FOR ALL SPECIFIED ROOMS
PRIVATE W_PRVPRTSS, W_PRVROOM, W_PRVSESS, W_COL, W_SABBREV, W_TABBREV, ;
      W_CABBREV, W_LINE1, W_LINE2, W_LINE3


W_LINE1=W_SPACE
W_LINE2=W_SPACE
W_LINE3=W_SPACE

SELECT 3
USE ROOMPF
SELECT 4
USE SUBJPF
SELECT 5
USE TCHERPF
SELECT 7
USE CLASSPF
SELECT ASSIGNPF
GO TOP

IF (.NOT. EOF())
  W_PRVPRTSS="00"  && SESSION PREVIOUSLY PRINTED
  W_PRVROOM=ROOM
  W_PRVSESS=SESSION
  DO NEWROOM WITH W_PRVROOM
  SELECT ASSIGNPF
ENDIF
DO WHILE (.NOT. EOF())
  IF (ASSIGNPF->ROOM<>W_PRVROOM)  && ROOM CHANGED
    DO BLANKSESS  WITH W_PRVPRTSS, W_PRVSESS
    DO PRTSESSION WITH W_PRVSESS, W_LINE1, W_LINE2, W_LINE3
    DO BLANKSESS  WITH W_PRVSESS, W_LASTSESS  && PRINT BLANK SESSION TILL LAST
    DO ENDOFLIST  WITH .T.
    SELECT ASSIGNPF

    W_PRVPRTSS="00"
    W_PRVROOM=ROOM
    W_PRVSESS=SESSION
```

```
    DO NEWROOM WITH W_PRVROOM
    SELECT ASSIGNPF

    W_LINE1=W_SPACE
    W_LINE2=W_SPACE
    W_LINE3=W_SPACE
  ENDIF  && (ROOM<>W_PRVROOM)

  IF (ASSIGNPF->SESSION<>W_PRVSESS)  && SESSION CHANGED
    DO BLANKSESS WITH W_PRVPRTSS, W_PRVSESS  && PRINT ANY BLANK SESSION IN BETWEEN
     DO PRTSESSION WITH W_PRVSESS, W_LINE1, W_LINE2, W_LINE3
    W_PRVPRTSS=W_PRVSESS
    W_LINE1=W_SPACE
    W_LINE2=W_SPACE
    W_LINE3=W_SPACE
    W_PRVSESS=ASSIGNPF->SESSION
  ENDIF  && (SESSION<>W_PRVSESS)

  W_COL=3+9*(VAL(ASSIGNPF->DAY)-1)
  SELECT SUBJPF
  LOCATE FOR SUBJECT=ASSIGNPF->SUBJECT
  IF FOUND()
    W_SABBREV=ABBREV
  ELSE
    W_SABBREV=SUBJECT+"  "
  ENDIF
  SELECT TCHERPF
  LOCATE FOR TCHER=ASSIGNPF->TCHER
  IF FOUND()
    W_TABBREV=ABBREV
  ELSE
    W_TABBREV=TCHER+"  "
  ENDIF
  SELECT CLASSPF
  LOCATE FOR CLASS=ASSIGNPF->CLASS
  IF FOUND()
    W_CABBREV=ABBREV
  ELSE
    W_CABBREV=CLASS+"  "
  ENDIF

  SELECT ASSIGNPF
  W_LINE1=STUFF(W_LINE1,W_COL,5,W_SABBREV)
  W_LINE2=STUFF(W_LINE2,W_COL,5,W_TABBREV)
  W_LINE3=STUFF(W_LINE3,W_COL,5,W_CABBREV)
  SKIP
ENDDO  && (.NOT. EOF())

DO BLANKSESS  WITH W_PRVPRTSS, W_PRVSESS
DO PRTSESSION WITH W_PRVSESS, W_LINE1, W_LINE2, W_LINE3  && LAST LINE
DO BLANKSESS  WITH W_PRVSESS, W_LASTSESS
DO ENDOFLIST  WITH .T.
RETURN  && PRINTROOM
** ----------------------------------------


PROCEDURE NEWROOM
PARAMETERS P_ROOM
PRIVATE W_DESC

** PRINT TITLE FOR NEW ROOM

DO CHKPRINTER
SELECT ROOMPF
LOCATE FOR ROOM=P_ROOM
```

```
W_DESC=" "
IF FOUND()
  W_DESC=DESC
ENDIF
@0,5 SAY W_ROMTBL
@0,50 SAY DATE()
@1,50 SAY TIME()
@2,0  SAY "Location : "+"["+P_ROOM+"]"+"  "+W_DESC
@4,5  SAY W_DTITLE
@5,5  SAY W_DASH
RETURN  && NEWROOM
** ----------------------------------------

PROCEDURE ENDOFLIST
PARAMETERS P_DASH
* PRINT END OF LIST MESSAGE OF PRINTOUT

DO CHKPRINTER
IF (P_DASH)
 @ PROW()+2, 5 SAY W_DASH
ENDIF
@ PROW()+3, 5 SAY W_EOFLST
@ PROW()+1, 0 SAY " "
@ 0,0 SAY " "  && EJECT TO NEW PAGE
RETURN  && ENDOFLIST
** ----------------------------------------

PROCEDURE BLANKSESS
PARAMETERS P_PRVSESS, P_NEXTSESS
PRIVATE   W_PS, W_S, W_SESS

* PRINT OUT ROWS OF BLANK SESSION WHICH ARE NOT ASSIGNED TO

W_PS=VAL(P_PRVSESS)
W_S=VAL(P_NEXTSESS)
DO WHILE (W_PS+1<W_S)
  W_PS=W_PS+1
  W_SESS=STR(W_PS,2)
  DO PRTSESSION WITH W_SESS, W_SPACE, W_SPACE, W_SPACE  && BLANK SESSION LINE
ENDDO  && (P_PS+1<P_S)
RETURN  && BLANKSESS
** ----------------------------------------

PROCEDURE PRTSESSION
PARAMETERS P_SESSION, P_LINE1, P_LINE2, P_LINE3
PRIVATE W_SESS

* PRINT OUT A SINGLE SESSION OF THE TIMETABLE

DO CHKPRINTER
W_SESS=VAL(P_SESSION)
IF (W_SESS=(D_BASTRUC1+1)) .OR. (W_SESS=(D_BASTRUC1+D_BASTRUC2+1)) ;
  .OR. (W_SESS=(D_BASTRUC1+D_BASTRUC2+D_BASTRUC3+1)) ;
  .OR. (W_SESS=(D_BASTRUC1+D_BASTRUC2+D_BASTRUC3+D_BASTRUC4+1))  ;
  .OR. (W_SESS=(D_BASTRUC1+D_BASTRUC2+D_BASTRUC3+D_BASTRUC4+D_BASTRUC5+1))
  DO PRTRECESS   && SESSION AFTER RECESS
ENDIF
@ PROW()+2, 1 SAY P_SESSION PICTURE "XX"
@ PROW()  , 5 SAY P_LINE1
@ PROW()+1, 5 SAY P_LINE2
IF (P_LINE3<>" ")
  @ PROW()+1, 5 SAY P_LINE3
ENDIF
RETURN  && PRTSESSION
** ----------------------------------------
```

```
PROCEDURE PRTRECESS
* PRINT RECESS LINE

@ PROW()+2, 5 SAY W_DASH
@ PROW()+1, 5 SAY W_RECESS
@ PROW()+1, 5 SAY W_DASH
RETURN  && PRTRECESS
** ----------------------------------------


PROCEDURE CHKPRINTER
PRIVATE W_QUIT
* CHECK WHETHER THE PRINTER IS PROPERLY CONNECTED

IF (.NOT. (D_LPT $ "LPT1LPT2LPT3COM1COM2"))
  RETURN  && NOT NECCESARY TO CHECK PRINTER WHILE DOING OUTPUT TO DISK FILE
ENDIF

SET PRINTER TO &D_LPT
SET DEVICE TO PRINT
DO WHILE .T.
  IF ISPRINTER()
    RETURN   && PRINTER IS OK
  ELSE
    SET DEVICE TO SCREEN
    @10,50 SAY " "
    ? CHR(7)
    @24,0 SAY "Printer not ready. Enter 'Q' to quit or any key to continue"
    SET CURSOR ON
    @24,63 SAY ' '
    W_QUIT=INKEY()
    DO WHILE W_QUIT=0
      W_QUIT=INKEY()
    ENDDO

    IF W_QUIT=ASC('q') .OR. W_QUIT=ASC('Q')
      QUIT
    ENDIF
    @ 24,63 SAY UPPER(CHR(W_QUIT))

    SET PRINTER TO &D_LPT
    SET CURSOR OFF
  ENDIF  && .NOT. ISPRINTER()
ENDDO && DO WHILE .T.
RETURN  && CHKPRINTER
```

APPENDIX E5

**** PROGRAM NAME : TTSETTNG.PRG**
**** LAST CHANGED : August 13, 1994**
**** DESCRIPTION : MAINTENANCE - TIME TABLE SYSTEM**
**** SET UP/ CHANGE SETTINGS MODULE**

```
set talk off
set bell off
set scoreboard off
set exact on
set confirm on
clear

L_FKEY=0  && 1=F1, 2=F2, ..., 11=F11, ..., 30=ESC, 31=PAGEDOWN, 32=PAGEUP
CALL SETKEY  && RETURN: L_FKEY   EXTERNAL PROGRAM TO HANDLE FUNCTION KEYS

D_BREAK=0
D_DAY=0
D_SESSION=0
D_BASTRUC1=0
D_BASTRUC2=0
D_BASTRUC3=0
D_BASTRUC4=0
D_BASTRUC5=0
D_BASTRUC6=0
D_LPT="LPT1"
D_HIDE="@"
D_MAX=0
D_SPLIT="N"
DD_FLAG=.F.
DD_MAX=0
DD_MFLAG=.F.
DD_SPLIT="N"
RESTORE FROM TTSETTNG ADDITIVE
RESTORE FROM TTSETNG ADDITIVE

** A SUMMARY TO THE CURRENT SETTING
W_OPTION=' '
@ 0,72 SAY DATE()
@ 1,72 SAY TIME()
@ 2, 0 TO 4,47 DOUBLE
@ 3, 8 SAY "A SUMMARY TO THE CURRENT SETTING"
@ 6, 0 SAY "Number of Days ..... (2..15):"
@ 6,45 SAY TRIM(STR(D_DAY,2))
@ 8, 0 SAY "Number of Periods in a Day..... (2..15):"
@ 8,45 SAY TRIM(STR(D_SESSION,2))
@10, 0 SAY "Total Number of Recess and Break ... (0...5):"
@10,45 SAY TRIM(STR(D_BREAK,2))
@12, 0 SAY "The Recess and Break Structure .....(1..15):"
@12,45 SAY TRIM(STR(D_BASTRUC1,2))
IF (D_BASTRUC2 >0)
@12,48 SAY "-"
@12,49 SAY TRIM(STR(D_BASTRUC2,2))
ENDIF
IF (D_BASTRUC3 >0)
```

```
 @12,52 SAY "-"
 @12,53 SAY TRIM(STR(D_BASTRUC3,2))
 ENDIF
 IF (D_BASTRUC4 >0)
 @12,56 SAY "-"
 @12,57 SAY TRIM(STR(D_BASTRUC4,2))
 ENDIF
 IF (D_BASTRUC5 >0)
 @12,60 SAY "-"
 @12,61 SAY TRIM(STR(D_BASTRUC5,2))
 ENDIF
 IF (D_BASTRUC6 >0)
 @12,64 SAY "-"
 @12,65 SAY TRIM(STR(D_BASTRUC5,2))
 ENDIF

 @14, 0 SAY "Maximum Daily Teaching Work Load.....(1..15):"
 @14,45 SAY TRIM(STR(D_MAX,2))
 @16, 0 SAY "Split Block Allowed, Same Single Period May Repeat Once (Y/N): "
 @16,65 SAY TRIM(D_SPLIT)
 @18, 0 SAY "Choice of Printing Device: "
 @18,29 SAY TRIM(D_LPT)
 @19, 0 SAY "The present of '@' means to hide command in batch file :"
 @19,58 SAY D_HIDE
 @22, 0 SAY "CHANGE THE CURRENT SETTING (Y/N) :" GET W_OPTION PICTURE '@Z N'
  READ
  IF UPPER(W_OPTION)<>'Y'
   CLEAR ALL
   RETURN
  ELSE
   DO WHILE .T.
    W_MSG=SPACE(80)
    @20,0  CLEAR TO 22,79
    @22,0 SAY "HOW MANY RECESS(ES) AND BREAK(S) IN A DAY (0,1,2,3,4 OR 5):";
    GET D_BREAK PICTURE '@Z 99'
    READ
     IF (L_FKEY=30)
      CLEAR ALL
      RETURN
     ENDIF
     IF (D_BREAK <0) .OR. (D_BREAK >5)
      @24,0 CLEAR TO 24,80
      W_MSG='DATA INVALID, MUST BETWEEN 0 AND 5, OR PRESS ESC TO ABORT.'
      @24,0 SAY W_MSG
      @0 ,0 SAY ' '
       ?CHR(7)
      LOOP
     ELSE
      EXIT
     ENDIF
    ENDDO
ENDIF

** SYSTEM SETTING RE-CONFIGURATION

W_MSG=SPACE(80)
W_SAVE=.F.
DO WHILE .T.
  D_LPT=D_LPT+SPACE(50)
  D_LPT=LEFT(D_LPT,50)
  SET DEVICE TO SCREEN
  @ 0,0 SAY " "
  IF (W_MSG<>SPACE(80))
    ? CHR(7)
  ENDIF
```

```
SET CURSOR ON
CLEAR
SET COLOR TO N/W
@ 0,22 SAY "TIME TABLE - SET UP / CHANGE SETTINGS"
SET COLOR TO W/N
@ 0,72 SAY DATE()
@ 1,72 SAY TIME()
@ 5, 0 SAY "Number of Days ..... (2..15):"  ;
     GET D_DAY PICTURE '@Z 99'
@ 7, 0 SAY "Number of Periods ..... (2..15):"  ;
     GET D_SESSION PICTURE '@Z 99'
@ 9, 0 CLEAR TO 9,79
DO CASE
 CASE (D_BREAK=0)
  D_BASTRUC2=0
  D_BASTRUC3=0
  D_BASTRUC4=0
  D_BASTRUC5=0
  D_BASTRUC6=0
  @ 9, 0 SAY "No Break, Re-enter Total Number of Periods to Confirm (1..15):";
       GET D_BASTRUC1 PICTURE '@Z 99'
 CASE (D_BREAK=1)
  D_BASTRUC3=0
  D_BASTRUC4=0
  D_BASTRUC5=0
  D_BASTRUC6=0
  @ 9, 0 SAY "Choice of Basic Structure (1..15):" ;
       GET D_BASTRUC1 PICTURE '@Z 99'
  @ 9,37 SAY "-"
  @ 9,38 GET D_BASTRUC2 PICTURE '@Z 99'
 CASE (D_BREAK=2)
  D_BASTRUC4=0
  D_BASTRUC5=0
  D_BASTRUC6=0
  @ 9, 0 SAY "Choice of Basic Structure (1..15):" ;
       GET D_BASTRUC1 PICTURE '@Z 99'
  @ 9,37 SAY "-"
  @ 9,38 GET D_BASTRUC2 PICTURE '@Z 99'
  @ 9,40 SAY "-"
  @ 9,41 GET D_BASTRUC3 PICTURE '@Z 99'
 CASE (D_BREAK=3)
  D_BASTRUC5=0
  D_BASTRUC6=0
  @ 9, 0 SAY "Choice of Basic Structure (1..15):" ;
       GET D_BASTRUC1 PICTURE '@Z 99'
  @ 9,37 SAY "-"
  @ 9,38 GET D_BASTRUC2 PICTURE '@Z 99'
  @ 9,40 SAY "-"
  @ 9,41 GET D_BASTRUC3 PICTURE '@Z 99'
  @ 9,43 SAY "-"
  @ 9,44 GET D_BASTRUC4 PICTURE '@Z 99'
 CASE (D_BREAK=4)
  D_BASTRUC6=0
  @ 9, 0 SAY "Choice of Basic Structure (1..15):" ;
       GET D_BASTRUC1 PICTURE '@Z 99'
  @ 9,37 SAY "-"
  @ 9,38 GET D_BASTRUC2 PICTURE '@Z 99'
  @ 9,40 SAY "-"
  @ 9,41 GET D_BASTRUC3 PICTURE '@Z 99'
  @ 9,43 SAY "-"
  @ 9,44 GET D_BASTRUC4 PICTURE '@Z 99'
  @ 9,46 SAY "-"
  @ 9,47 GET D_BASTRUC5 PICTURE '@Z 99'
 CASE (D_BREAK=5)
 @ 9, 0 SAY "Choice of Basic Structure (1..15):" ;
```

```
      GET D_BASTRUC1 PICTURE '@Z 99'
   @ 9,37 SAY "-"
   @ 9,38 GET D_BASTRUC2 PICTURE '@Z 99'
   @ 9,40 SAY "-"
   @ 9,41 GET D_BASTRUC3 PICTURE '@Z 99'
   @ 9,43 SAY "-"
   @ 9,44 GET D_BASTRUC4 PICTURE '@Z 99'
   @ 9,46 SAY "-"
   @ 9,47 GET D_BASTRUC5 PICTURE '@Z 99'
   @ 9,49 SAY "-"
   @ 9,50 GET D_BASTRUC6 PICTURE '@Z 99'
  ENDCASE

 @10, 3 SAY "(Enter Number of Periods between Recess(es) and Break(s))"
 @12, 0 SAY "Maximum Number of Teaching Periods (1..15):" ;
     GET D_MAX PICTURE '@Z 99'
 @14, 0 SAY "Split Block Allow, Same Single Period May Repeat Once: " ;
     GET D_SPLIT PICTURE '!'
 @16, 0 SAY "Choice of Printing Device: " GET D_LPT ;
        PICTURE '!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!'
 @17, 3 SAY "(LPT1 or LPT2 or C:\path\filename.ext)"
 @19, 0 SAY "Enter '@' to hide command in batch file :" GET D_HIDE PICTURE '!'

 IF (W_SAVE=.T.)
  @20,0 CLEAR TO 20,79
  @21,0 SAY "All Parameters Validate, Press F6 To Save The Setting."
  @23,0 CLEAR TO 23,79
  @23,0 SAY "ESC=ABORT       F6=SAVE"
 ELSE
  @23,0 SAY "ESC=Cancel"
 ENDIF
 @24, 0 CLEAR TO 24,80
 @24, 0 SAY W_MSG
 W_MSG=SPACE(80)
 W_SAVE=.F.
 L_FKEY=0
 READ

* ESC KEY PRESSED DURING 'READ' (RETURNED BY 'SETKEY')
 IF (L_FKEY=30)
   CLEAR ALL
   RETURN

 ENDIF  && L_FKEY=30  (ESC)

* VALIDATION
 IF (D_DAY < 2) .OR. (D_DAY > 15)
   W_SAVE=.F.
   W_MSG='Number of Days must be between 2 and 15'
   LOOP
 ELSE
   W_SAVE=.T.
 ENDIF
 IF (D_SESSION < 2) .OR. (D_SESSION > 15)
   W_SAVE=.F.
   W_MSG='Number of Periods must be between 1 and 15'
   LOOP
 ELSE
   W_SAVE=.T.
 ENDIF

 W_OK=.T.
 DO CASE
 CASE (D_BREAK=0)
  IF ((D_BASTRUC1 < 1) .OR. (D_BASTRUC1 > 15))
```

```
   W_OK =.F.
 ENDIF
 CASE (D_BREAK=1)
 IF ((D_BASTRUC1 < 1) .OR. (D_BASTRUC1 > 15)) .OR. ;
   ((D_BASTRUC2 < 1) .OR. (D_BASTRUC2 > 15))
   W_OK =.F.
 ENDIF
 CASE (D_BREAK=2)
 IF ((D_BASTRUC1 < 1) .OR. (D_BASTRUC1 > 15)) .OR. ;
   ((D_BASTRUC2 < 1) .OR. (D_BASTRUC2 > 15)) .OR. ;
   ((D_BASTRUC3 < 1) .OR. (D_BASTRUC3 > 15))
   W_OK = .F.
 ENDIF
 CASE(D_BREAK=3)
 IF ((D_BASTRUC1 < 1) .OR. (D_BASTRUC1 > 15)) .OR. ;
   ((D_BASTRUC2 < 1) .OR. (D_BASTRUC2 > 15)) .OR. ;
   ((D_BASTRUC3 < 1) .OR. (D_BASTRUC3 > 15)) .OR. ;
   ((D_BASTRUC4 < 1) .OR. (D_BASTRUC4 > 15))
   W_OK = .F.
 ENDIF
CASE(D_BREAK=4)
 IF ((D_BASTRUC1 < 1) .OR. (D_BASTRUC1 > 15)) .OR. ;
   ((D_BASTRUC2 < 1) .OR. (D_BASTRUC2 > 15)) .OR. ;
   ((D_BASTRUC3 < 1) .OR. (D_BASTRUC3 > 15)) .OR. ;
   ((D_BASTRUC4 < 1) .OR. (D_BASTRUC4 > 15)) .OR. ;
   ((D_BASTRUC5 < 1) .OR. (D_BASTRUC5 > 15))
   W_OK = .F.
 ENDIF
CASE(D_BREAK=5)
 IF ((D_BASTRUC1 < 1) .OR. (D_BASTRUC1 > 15)) .OR. ;
   ((D_BASTRUC2 < 1) .OR. (D_BASTRUC2 > 15)) .OR. ;
   ((D_BASTRUC3 < 1) .OR. (D_BASTRUC3 > 15)) .OR. ;
   ((D_BASTRUC4 < 1) .OR. (D_BASTRUC4 > 15)) .OR. ;
   ((D_BASTRUC5 < 1) .OR. (D_BASTRUC5 > 15)) .OR. ;
   ((D_BASTRUC6 < 1) .OR. (D_BASTRUC6 > 15))
   W_OK = .F.
 ENDIF
ENDCASE

IF (W_OK = .F.)
 W_SAVE=.F.
 W_MSG='Basic Structure breakdowns must be between 1 and 15'
 LOOP
ELSE
 W_SAVE=.T.
ENDIF

IF (D_BASTRUC1+D_BASTRUC2+D_BASTRUC3+D_BASTRUC4+D_BASTRUC5 ;
  +D_BASTRUC6 <> D_SESSION)
  W_SAVE=.F.
  W_MSG='Sum of Basic Structure breakdowns must be equal to '+ALLTRIM(STR(D_SESSION))
  LOOP
ELSE
  W_SAVE=.T.
ENDIF

IF (D_MAX > D_SESSION)
  W_SAVE=.F.
  W_MSG='Maximum Daily Teaching Periods Should Not Exceed Total Sessions'
  LOOP
ELSE
  W_SAVE=.T.
  IF (D_MAX = 0)
    D_MAX = D_SESSION
    W_MSG='Maximum Daily Teaching Periods Set To Total Sessions'
```

```
    ENDIF
  ENDIF

  IF UPPER(D_SPLIT)<>'Y'
     D_SPLIT='N'
  ENDIF

  D_LPT=ALLTRIM(D_LPT)
  IF (.NOT. (D_LPT $ 'LPT1LPT2LPT3COM1COM2'))
    IF (.NOT. FILE('&D_LPT'))
      * RETURN .T. IF ABLE TO WRITE TO THE FILE->DIRECTORY EXIST
      IF (.NOT. MEMOWRIT('&D_LPT','TTSETTNG-TEST EXISTENCE OF DIRECTORY'))
        W_SAVE=.F.
        W_MSG="File '"+D_LPT+"' not valid"
        LOOP
      ELSE
        W_SAVE=.T.
      ENDIF
      ERASE &D_LPT
    ENDIF
  ENDIF
  IF (D_HIDE<>'@')
    D_HIDE=' '
  ENDIF

  IF (.NOT. (D_LPT $ 'LPT1LPT2LPT3COM1COM2')) .AND. (.NOT. ('.' $ D_LPT))
    D_LPT=D_LPT+'.'
  ENDIF

  DO CASE
    CASE (L_FKEY=0)   && ENTER
    CASE (L_FKEY=6)   && F6
       SAVE TO TTSETTNG ALL LIKE D_*
       DD_FLAG=.T.
       DD_SPLIT=D_SPLIT
       DD_MAX = D_MAX
       DD_MFLAG=(D_MAX=D_SESSION)
       SAVE TO TTSETNG ALL LIKE DD_*
       EXIT
    CASE (L_FKEY=30)  && ESC
    CASE (L_FKEY=31)  && PAGEUP/PAGEDOWN
    CASE (L_FKEY=32)  && PAGEUP
    CASE (L_FKEY>=1) .AND. (L_FKEY<=9)
       W_MSG="Function Key 'F"+STR(L_FKEY,1)+"' not valid"
    OTHERWISE
       W_MSG="Function Key 'F"+STR(L_FKEY,2)+"' not valid"
  ENDCASE
ENDDO && DO WHILE .T.
DO GEN_FILES
CLOSE ALL
RETURN  && MAIN LINE
** ----------------------------------------

PROCEDURE GEN_FILES
* GENERATE CONTENT FOR FILEs 'DAYSESPF' & 'CLSLOTPF' ACCORDING TO D_DAY & D_SESSION

@20,0 CLEAR TO 24,79
?CHR(7)
@21,0 SAY "Setting Change, Update The Preassignment File (If Required)."
@24,0 CLEAR TO 24,79
@24,0 SAY "Updating Parameter Files, Please Wait ..."
```

```
SET CURSOR OFF
SELECT 1
USE DAYSESPF INDEX DAYSESL1
ZAP
SELECT 2
USE CLSLOTPF
ZAP

* CALCULATE THE L.C.M. OF D_DAY AND D_SESSION
* EACH MULTIPLE OF L.C.M. ARE POSITION OF THE ADJUSTMENT SLOT
W_A=D_DAY
W_B=D_SESSION
IF (W_A<W_B)
  W_A=D_SESSION
  W_B=D_DAY
ENDIF
W_REMINDER=2
DO WHILE (W_REMINDER>1)
  W_REMINDER=MOD(W_A, W_B)
  W_A=W_B
  W_B=W_REMINDER
ENDDO
IF (W_REMINDER=0)
  W_LCM=(D_DAY/W_A)*D_SESSION
ENDIF
IF (W_REMINDER=1)
  W_LCM=D_DAY*D_SESSION
ENDIF

W_DAY=0
W_SESSION=0
FOR W_ORDER=1 TO (D_DAY * D_SESSION)
  W_DAY=W_DAY+1
  IF (W_DAY>D_DAY)
    W_DAY=1
  ENDIF
  IF (W_ORDER>1) .AND. (MOD(W_ORDER, W_LCM)=1)
    W_DAY=W_DAY+1
  ENDIF
  W_SESSION=W_SESSION+1
  IF (W_SESSION>D_SESSION)
    W_SESSION=1
  ENDIF
  SELECT DAYSESPF
  APPEND BLANK
  REPLACE SLOT WITH STR(W_ORDER,2), DAY WITH STR(W_DAY,2), SESSION WITH STR(W_SESSION,2)
  SELECT CLSLOTPF
  APPEND BLANK
  REPLACE CLASS WITH ' ', TYPE WITH 'S', SLOT WITH STR(W_ORDER,2)
NEXT

**BUILD UP CONSECUTIVE BLOCKS, T=TRIPLE, D=DOUBLE, S=SINGLE

SELECT DAYSESPF
USE
SELECT CLSLOTPF
USE
COPY FILE CLSLOTPF.DBF TO CLSLOT$.DBF
USE CLSLOT$
REPLACE ALL TYPE WITH 'T'

W_T=' '
FOR W_A=1 TO (D_BASTRUC1 -2)
  W_T=W_T+STR(W_A,3)
NEXT
```

```
IF (D_BREAK>=1)
FOR W_A=1 TO (D_BASTRUC2 -2)
   W_T=W_T+STR(D_BASTRUC1+W_A,3)
NEXT
ENDIF
IF (D_BREAK>=2)
FOR W_A=1 TO (D_BASTRUC3 -2)
   W_T=W_T+STR(D_BASTRUC1 +D_BASTRUC2 +W_A,3)
NEXT
ENDIF
IF (D_BREAK>=3)
FOR W_A=1 TO (D_BASTRUC4 -2)
   W_T=W_T+STR(D_BASTRUC1 +D_BASTRUC2 +D_BASTRUC3 +W_A,3)
NEXT
ENDIF
IF (D_BREAK>=4)
FOR W_A=1 TO (D_BASTRUC5 -2)
   W_T=W_T+STR(D_BASTRUC1 +D_BASTRUC2 +D_BASTRUC3 +D_BASTRUC4 +W_A,3)
NEXT
ENDIF
IF (D_BREAK=5)
FOR W_A=1 TO (D_BASTRUC6 -2)
   W_T=W_T+STR(D_BASTRUC1 +D_BASTRUC2 +D_BASTRUC3 +D_BASTRUC4  ;
      +D_BASTRUC5 +W_A,3)
NEXT
ENDIF

W_T=W_T+' '
SELECT 3
USE DAYSESPF
SET FILTER TO SESSION+' ' $ W_T
GO TOP
W_SLOT=' '
DO WHILE (.NOT. EOF())
  W_SLOT=W_SLOT+SLOT+' '
  SKIP
ENDDO
SELECT 2
USE CLSLOTPF
APPEND FROM CLSLOT$.DBF FOR SLOT+' ' $ W_SLOT

W_D=' '
FOR W_A=1 TO (D_BASTRUC1 -1)
   W_D=W_D+STR(W_A,3)
NEXT
IF (D_BREAK>=1)
FOR W_A=1 TO (D_BASTRUC2 -1)
   W_D=W_D+STR(D_BASTRUC1 +W_A,3)
NEXT
ENDIF
IF (D_BREAK>=2)
FOR W_A=1 TO (D_BASTRUC3 -1)
   W_D=W_D+STR(D_BASTRUC1 +D_BASTRUC2 +W_A,3)
NEXT
ENDIF
IF (D_BREAK>=3)
FOR W_A=1 TO (D_BASTRUC4 -1)
   W_D=W_D+STR(D_BASTRUC1 +D_BASTRUC2 +D_BASTRUC3 +W_A,3)
NEXT
ENDIF
IF (D_BREAK>=4)
FOR W_A=1 TO (D_BASTRUC5 -1)
   W_D=W_D+STR(D_BASTRUC1 +D_BASTRUC2 +D_BASTRUC3 +D_BASTRUC4 +W_A,3)
NEXT
ENDIF
```

```
IF (D_BREAK=5)
FOR W_A=1 TO (D_BASTRUC6 -1)
   W_D=W_D+STR(D_BASTRUC1 +D_BASTRUC2 +D_BASTRUC3 +D_BASTRUC4  ;
       +D_BASTRUC5 +W_A,3)
NEXT
ENDIF
W_D=W_D+' '
SELECT 3
USE DAYSESPF
SET FILTER TO SESSION+' ' $ W_D
GO TOP
W_SLOT=' '
DO WHILE (.NOT. EOF())
  W_SLOT=W_SLOT+SLOT+' '
  SKIP
ENDDO
SELECT 2
USE CLSLOT$
REPLACE ALL TYPE WITH 'D'
USE CLSLOTPF
APPEND FROM CLSLOT$.DBF FOR SLOT+' ' $ W_SLOT
USE
COPY FILE CLSLOTPF.DBF TO CLSLOT$.DBF
USE CLSLOTPF
SELECT 3
USE TDSPF
ZAP
SELECT 4
USE DAYSESPF
SELECT CLSLOTPF
GO TOP
DO WHILE (.NOT. EOF())
  W_SLOT2=' '
  W_SLOT3=' '
  IF (CLSLOTPF->TYPE='D') .OR. (CLSLOTPF->TYPE='T')
    SELECT DAYSESPF
    LOCATE FOR SLOT=CLSLOTPF->SLOT
    IF (.NOT. FOUND())
      @24,0 SAY "(TTSETTNG): Record not found in 'DAYSESPF'. File 'CLSLOTPF' corrupted"
      EXIT
    ENDIF
    W_DAY=DAY
    W_SESSION=VAL(SESSION)
    LOCATE FOR DAY=W_DAY .AND. SESSION=STR(W_SESSION+1,2)
    IF (.NOT. FOUND())
      @24,0 SAY "(TTSETTNG): Record not found in 'DAYSESPF'. File 'CLSLOTPF' corrupted"
      EXIT
    ENDIF
    W_SLOT2=SLOT
  ENDIF
  IF (CLSLOTPF->TYPE='T')
    LOCATE FOR DAY=W_DAY .AND. SESSION=STR(W_SESSION+2,2)
    IF (.NOT. FOUND())
      @24,0 SAY "(TTSETTNG): Record not found in 'DAYSESPF'. File 'CLSLOTPF' corrupted"
      EXIT
    ENDIF
    W_SLOT3=SLOT
  ENDIF
  SELECT TDSPF
  APPEND BLANK
  REPLACE TYPE WITH CLSLOTPF->TYPE, SLOT1 WITH CLSLOTPF->SLOT, ;
       SLOT2 WITH W_SLOT2, SLOT3 WITH W_SLOT3
  SELECT CLSLOTPF
  SKIP
ENDDO
```

```
ZAP
SELECT 3
USE CLASSPF
GO TOP
DO WHILE (.NOT. EOF())
  SELECT CLSLOTPF
  APPEND FROM CLSLOT$.DBF
  REPLACE ALL CLASS WITH CLASSPF->CLASS FOR CLASS=' '
  SELECT CLASSPF
  SKIP
ENDDO
SELECT CLSLOTPF
SET INDEX TO CLSLOTL1
REINDEX
CLOSE ALL
ERASE CLSLOT$.DBF
SET CURSOR ON
RETURN  && GEN_FILES
```

APPENDIX E6

**** PROGRAM NAME : TTUPDATE.PRG**
**** LAST CHANGED : August 15, 1994**
**** DESCRIPTION : TIME TABLE FILE MAINTENNANCE**
**** PEPREASSIGNMENT/ASSIGNMENT FILE**
**** UPDATE MODULE**

```
set talk off
set bell off
set scoreboard off
set exact on
set confirm on
set cursor on

CLEAR ALL

L_FKEY=0  && 1=F1, 2=F2, ..., 11=F11, ..., 30=ESC, 31=PAGEDOWN, 32=PAGEUP
CALL SETKEY  && RETURN: L_FKEY  EXTERNAL PROGRAM TO HANDLE FUNCTION KEYS
RESTORE FROM TTSETNG ADDITIVE
STORE ' ' TO T_OPT
STORE ' ' TO T_OPT1
CLEAR
@ 0,72 SAY DATE()
@ 1,72 SAY TIME()
@ 3, 0 CLEAR TO 19,79
@12, 4 SAY "File Will Be Updated And Verified (Y/N) : " ;
  GET T_OPT PICTURE '@Z X'
READ
 IF (L_FKEY=30) .OR. (UPPER(T_OPT)<>'Y')
  RETURN
 ENDIF  && L_FKEY=30 (ESC) OR OTHER KEYS TO ABORT

STORE ' ' TO T_OPT
@ 3, 0 CLEAR TO 21,79
@ 4, 4 TO 16,46 DOUBLE
SET COLOR TO W+/N
@ 6, 8 SAY "U P D A T E  F I L E  U T I L I T Y"
SET COLOR TO W/N
@ 8, 5 TO 8,45
@10, 8 SAY "1. PREASSIGNMENT FILE "
@12, 8 SAY "2. ASSIGNMENT FILE "
@14, 8 SAY "3. ASSIGNMENT --> PREASSIGNMENT FILE "
@18, 8 SAY "Enter Option : " GET T_OPT PICTURE '@Z N'
@23, 0 SAY "ESC=CANCEL "
READ
 DO CASE
  CASE (T_OPT='1')
     COPY FILE PREASSPF.DBF TO PASSTMP2.DBF

  CASE (T_OPT='2')
     COPY FILE ASSIGNPF.DBF TO PASSTMP2.DBF
```

```
   CASE (T_OPT='3')
      @18, 0 CLEAR TO 23,0
      @18, 8 SAY "Permanent Update !"
      @19, 8 SAY "Data Transferred from Assignment to Preassignment File (Y/N): " ;
        GET T_OPT1 PICTURE '@Z X'
        READ
        IF (UPPER(T_OPT1)='Y')
         @ 18, 0 CLEAR TO 22,78
         @ 22, 0 SAY "Data in Preassignment File is now replaced !"
         COPY FILE PREASSPF.DBF TO PASSTMP1.DBF
         COPY FILE ASSIGNPF.DBF TO PREASSPF.DBF
         ?CHR(7)
        ENDIF
      RETURN
   OTHERWISE
      RETURN
 ENDCASE

@ 23,0 SAY "Working...."

SET CURSOR OFF
SELECT 3
USE CLSLOTPF.DBF INDEX CLSLOTL1
SELECT 2
USE DAYSESPF.DBF
INDEX ON DAY+SESSION TO DSTEMP
SET ORDER TO 1
SELECT 1
USE PASSTMP2.DBF
DELE FOR CLASS=" "
PACK
REPLACE ALL SLOT WITH 'X' FOR CLASS <> ' '
GO TOP
DO WHILE (.NOT. EOF())
@ 0,72 SAY DATE()
@ 1,72 SAY TIME()
 IF (TYPE("DAY")='N')
  T_DAY=DAY
 ELSE
  T_DAY=VAL(DAY)
 ENDIF
 IF (TYPE("SESSION")='N')
  T_SESSION=SESSION
 ELSE
  T_SESSION=VAL(SESSION)
 ENDIF
 T_CLASS=CLASS
 T_SLOT='X'
 T_TYPE=TYPE
 T_TYPE1=TYPE1
SELECT 2
SEEK TRIM(STR(T_DAY,2)+STR(T_SESSION,2))
IF FOUND()
  T_SLOT = (B->SLOT)
  IF ((T_TYPE='S').OR.(T_TYPE1<>'1'))
   REPLACE A->SLOT WITH B->SLOT
  ELSE
   SELECT 3
   SEEK (TRIM(T_CLASS+T_TYPE+T_SLOT))
    IF FOUND()
     REPLACE A->SLOT WITH B->SLOT
    ENDIF
  ENDIF
ENDIF
SELECT 1
```

```
 SKIP
ENDDO  && DO WHILE

@3, 0 CLEAR TO 18,78
SELECT 1
GO TOP
IF (.NOT. EOF())
LOCATE FOR SLOT = 'X'
 IF FOUND()
  ?CHR(7)
  @13, 4 SAY "Date Structure Incompatible With The New Setting."
  ELSE
  @13, 4 SAY "Conversion Compatible With The New Setting."
 ENDIF
ELSE
 @ 13, 20 SAY "File  Empty !!"
 ?CHR(7)
ENDIF

 SET CURSOR ON
 T_OPT1=' '
 @22,0 CLEAR TO 24,79
 @22,0 TO 24,79 DOUBLE
 @23,4 SAY "Permanent Update: (C)onfirm/(D)elete/(U)nchanged & Keep Work File." ;
  GET T_OPT1 PICTURE '@Z !'
  L_FKEY=0
   READ
   IF (L_FKEY=30)  && ESC
    RETURN
   ELSE
    CLOSE DATABASES
      DO CASE
        CASE UPPER(T_OPT1)='C'
            @23,3 CLEAR TO 23,78
            @23,3 SAY 'Work file saved and deleted'
            IF (T_OPT)='1'
             COPY FILE PASSTMP2.DBF TO PREASSPF.DBF
             DD_FLAG=.F.
            ELSE
             IF (T_OPT)='2'
             COPY FILE PASSTMP2.DBF TO ASSIGNPF.DBF
             SELECT 4
             USE ASSIGNPF.DBF INDEX ASSIGNL1,ASSIGNL2,ASSIGNL3,ASSIGNL4,TEMP
             REINDEX
             ENDIF
            ENDIF
            ERASE PASSTMP2.DBF
            SAVE TO TTSETNG ALL LIKE DD_*
        CASE UPPER(T_OPT1)='D'
            @23,3 CLEAR TO 23,78
            @23,3 SAY 'Work File deleted. All Changes Lost'
            ERASE PASSTMP2.DBF
        CASE UPPER(T_OPT1)='U'
            @23,3 CLEAR TO 23,78
            @23,3 SAY 'Work File (PASSTMP2) has been kept for later use.'
      ENDCASE
   ENDIF  && FILE('PASSTMP2.DBF')

CLEAR
CLOSE ALL
ERASE DSTEMP.NDX
RETURN  && MAIN LINE
```