

DESIGN AND IMPLEMENTATION OF AN  
AUTOMATED REGISTRATION SYSTEM

A MASTER'S THESIS  
in  
Computer Engineering  
Middle East Technical University

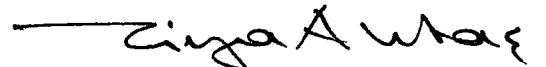
T. C.  
Yükseköğretim Kurulu  
Dokümantasyon Merkezi

By  
Argun C. ATASAGUN  
September 1987


Approval of the Graduate School of Natural and Applied Sciences.

  
Prof. Dr. Halim Dogrusöz  
-----  
Director

I certify that this thesis satisfies all the requirements as a thesis for the degree of Master of Science in Computer Engineering.

  
Prof. Dr. Ziya Aktas  
-----  
Chairman of the Department

We certify that we have read this thesis and that in our opinion it is fully adequate, in scope and quality, as a thesis for the degree of Master of Science in Computer Engineering.

  
Asst. Prof. Bülent Epir  
-----  
Supervisor

Examining Committee in Charge

Prof. Dr. Ziya Aktas (Chairman)

Assoc. Prof. Dr. Ünal Yarımagan

Assoc. Prof. Dr. Müren Gökeri

Asst. Prof. Dr. Faruk Tokdemir

Asst. Prof. Bülent Epir

ABSTRACT

DESIGN AND IMPLEMENTATION OF AN  
AUTOMATED REGISTRATION SYSTEM

ATASAGUN, Argun C.  
M.S. in Computer Eng.  
Supervisor: Asst. Prof. Bülent Epir  
September 1987, 208 pages

In this study, a Registration System, which is one of the fundamental sections of an information system, carrying out an examination organization automatically from the application of the candidates up to the determination of the results is designed and implemented. The main function of the Registration System is to obtain the information about the candidates and the executive staff systematically and to print the documents to be sent to candidates and to be used in the examination.

The Registration System is designed to communicate with the Examination System which is the other section of the examination organization. These two systems form an information system called "Registration and Examination System", or REX. REX is aimed to be modular, structural, executable by multi-user in an interactive environment and directing the users by menus. In developing REX, the system in ÖSYM, which carries out various examination organizations, has been examined and the experience obtained from this organization has been utilised.

Key words: Examination Organization System, Registration System, structured system analysis, program writing programs.

## ÖZET

### ÖZDEVİNİMLİ BİR KAYIT SİSTEMİNİN TASARIMI VE GERÇEKLEŞTİRİMİ

ATASAGUN, Argun C.  
Yüksek Lisans Tezi, Bilgisayar Müh. Bölümü  
Tez Yöneticisi: Y.Doc. Bülent Epir  
Eylül 1987, 208 sahife

Bu çalışmada, bir sınav organizasyonunu aday başvurularından başlayarak sınav sonuçlarının belirlenmesine kadar otomatik olarak yürütecek bir bilişim sisteminin temel kesimlerinden biri olan Kayıt Sistemi tasarlanmış ve gerçekleştirilmiştir. Kayıt Sisteminin temel işlevi, sınava girecek adaylara ve sınavı uygulayacak görevlilere ait bilgileri sistematik olarak elde etmek ile adaylara gönderilecek ve sınavda kullanılacak dökümlerin yapılmasını sağlamaktır.

Kayıt Sistemi sınav organizasyonun diğer kesimi olan Sınav Sistemi ile iletişim kuracak biçimde tasarlanmıştır. Bu iki sistem "Registration and Examination System" olarak adlandırılan ve kısa adı REX olan bir bilişim sistemi oluşturmaktadır. REX'in modüler ve yapısal olması, etkileşimli bir ortamda aynı anda birden çok kullanıcı tarafından çalıştırılabilmesi ve kullanıcıları menülerle yönlendirmesi öngörülmüştür. Gerçekleştirimde çeşitli sınav organizasyonlarının uygulandığı bir kurum olan ÖSYM'deki mevcut sistem incelenmiş ve burada edinilen deneyimlerden yararlanılmıştır.

Anahtar sözcükler: Sınav Organizasyon Sistemi, Kayıt Sistemi, yapısal sistem çözümlenme, program yazan programlar.

## ACKNOWLEDGEMENT

The writer expresses his sincere gratitude to Asst.Prof. Bülent Epir for his excellent guidance and patience in directing this study. Deep appreciation is extended to Assoc.Prof.Dr. Ünal Yarımagan for his valuable suggestions and counselling. Special appreciation is also extended to Prof.Dr. Altan Günalp for his permission to implement this study in ÖSYM and to use the resources of the center. The writer also wishes to express his gratefulness to Mehmet Altunay for his assistance. Thanks also due to my family for their unfailing support and encouragement. The following people have had a direct interest in this study which was helpful: Dr. Atalay Yunusoglu, Giray Berberoglu, Tamer Tüfekçi, Eyüp Onat and Suat Canpolat. Finally, I would like to thank my colleague Mustafa Tütüncü for his cooperation.

## TABLE OF CONTENTS

	Page
ABSTRACT .....	iii
ÖZET .....	iv
ACKNOWLEDGEMENT .....	v
LIST OF TABLES .....	x
LIST OF FIGURES .....	xi
1. INTRODUCTION .....	1
1.1. Purpose and Scope of REX .....	2
1.2. Subsystems of REX .....	4
1.3. Implementation Notes on REX .....	4
2. EXAMINATION AND REGISTRATION CONCEPTS .....	6
2.1. Examination Concept .....	6
2.2. Registration Concept .....	7
3. PROPOSED EXAMINATION ORGANIZATION SYSTEM .....	9
3.1. Files .....	9
3.2. Sources and Sinks .....	11
3.3. Data Flows .....	12
3.4. The Subsystems of the Examination Organization System .....	14
3.4.1. Registration Supporting Subsystem ....	14
3.4.2. Examination Supporting Subsystem ....	16
3.4.3. Test Preparing Subsystem .....	16
3.4.4. REX Subsystem .....	16
3.4.4.1. Concepts of REX .....	18
3.4.4.1.1. User System .....	18
3.4.4.1.2. Examination Center ...	19
3.4.4.1.3. Session .....	19

3.4.4.1.4. Area .....	20
3.4.4.1.5. Subarea .....	20
3.4.4.2. Relations Between the Concepts	20
3.4.4.3. Subsystems of REX .....	26
3.4.4.3.1. Registration Subsystem	26
3.4.4.3.2. Examination Subsystem	26
3.4.4.3.3. Supporting Subsystem	27
3.4.4.4. Physical Implementation of REX	27
3.4.4.5. How to Run REX .....	28
3.4.4.6. Properties of REX .....	31
3.4.4.7. System Requirements of REX ....	33
4. LOGICAL DESIGN OF REGISTRATION SYSTEM .....	34
4.1. Registration Supporting Subsystem .....	38
4.2. REX Registration Subsystem .....	44
4.2.1. Processes Related with the Candidates	44
4.2.2. Processes Related with the Staff .....	51
5. PHYSICAL DESIGN OF REGISTRATION SYSTEM .....	57
5.1. Data Files .....	57
5.1.1. Candidate File .....	57
5.1.2. Room Files .....	61
5.1.3. Building File .....	64
5.1.4. Staff Files .....	65
5.1.5. Registration Optical File .....	68
5.1.6. Staff Information File .....	68
5.2. General Purpose Programs .....	69
5.2.1. Common Characteristics .....	70
5.2.2. File Load Program .....	77
5.2.3. Print Program .....	82
5.2.4. Distribution Program .....	86
5.3. Parameter File .....	90
5.4. Layout Files .....	93
5.5. Screen Files .....	94

5.6. Libraries .....	96
5.7. Synchronization Files .....	97
5.8. Job Files .....	98
6. IMPLEMENTATION OF REGISTRATION SYSTEM .....	99
6.1. Preparation of the Candidate File Layout ....	99
6.2. Creation of the Candidate File .....	101
6.3. Printing the Application Forms .....	101
6.4. Updating the Candidate File .....	102
6.5. Determination of the Duplicate Applications	104
6.6. Printing the Incomplete/Invalid Information	105
6.7. Interactive Inquiry/Updating .....	105
6.8. Center/Session/Area Distribution .....	107
6.9. Printouts Concerning the Candidates .....	108
6.10. Sorting the Room Files .....	110
6.11. Creation of the Staff Files .....	110
6.12. Printing the Staff Notification Schedule ...	112
6.13. Transferring the Staff Data to the Files ...	112
6.14. Printouts Concerning the Staff .....	112
7. TESTING OF REX .....	116
8. CONCLUSION .....	117
REFERENCES .....	119
APPENDICES	
A. STUDENT SELECTION AND PLACEMENT CENTER (ÖSYM) ....	120
A.1. ÖSYM Chronicle .....	120
A.2. Organization Schema of ÖSYM .....	121
A.3. Activities of ÖSYM .....	125
B. EXISTING EXAMINATION ORGANIZATION SYSTEM .....	126
B.1. Data processing Unit .....	126
B.1.1. General Supporting Subunit .....	126
B.1.2. Candidate Registration Subunit .....	126
B.1.3. Examination Evaluation Subunit .....	130
B.1.4. Examination Execution Subunit .....	132
B.1.5. Research Subunit .....	133



B.2. Execution of the Examination .....	134
B.2.1. Staff .....	134
B.2.2. Examination Places .....	139
C. STRUCTURED TOOLS .....	140
C.1. Data Flow Diagram (DFD) .....	140
C.2. Warnier-Orr Diagram .....	142
D. GLOSSARY .....	144
E. PROGRAM LISTINGS .....	145
F. PRINTOUTS OF THE SAMPLE RUN .....	201



LIST OF TABLES

Table		Page
5.1	Title Codes of the Staff .....	67
5.2	Duty Codes of the Staff .....	67
5.3	Screen Files of REX .....	95



## LIST OF FIGURES

Figure	Page
3.1 DFD of Examination Organization System .....	10
3.2 DFD of Subsystems of Examination Organization	15
3.3 DFD of Subsystems of REX .....	17
3.4 Relations between the REX Concepts .....	21
3.5 Possible States of the Moduls .....	29
3.6 REX_Main Menu .....	30
3.7 REX_Existing User Systems Form .....	30
3.8 REX_User System Parameters Form .....	32
3.9 REX_Subsystems Form .....	32
4.1 Examination Organization System Structure .....	35
4.2 DFD of Registration System .....	36
4.3 DFD of Registration Supporting System .....	40
4.4 DFD of Identification of Staff Process .....	43
4.5 DFD of REX Registration Subsystem .....	45
4.6 DFD of Processes Related to Candidates .....	47
4.7 DFD of Printouts Concerning the Candidates ....	50
4.8 DFD of Processes Related to Staff .....	52
4.9 DFD of Printouts Concerning the Staff .....	55
5.1 Warnier-Orr Diagram of Candidate File Layout ..	58
5.2 Warnier-Orr Diagram of Room File Layout .....	62
5.3 Warnier-Orr Diagram of Building File Layout ...	64
5.4 Warnier-Orr Diagram of Staff File Layout .....	66
5.5 Program Name Form .....	73
5.6 Job Starting Form .....	73
5.7 Layouts Form .....	74
5.8 Commands Form .....	74
5.9 Output Definition Form .....	76
5.10 File Attributes Form .....	76
5.11 File Load_Main Menu .....	80
5.12 File Load_Specifications Form .....	80
5.13 File Load_Warnings Form .....	81
5.14 Print_Main Menu .....	84

5.15	Print_Page Definition Form .....	84
5.16	Sample Layout of the Distribution .....	87
5.17	Distribution_Main Menu .....	89
5.18	Distribution_Line/Column Definition Form .....	89
5.19	Warnier-Orr Diagram of Parameter File .....	91
6.1	Registration_Main Menu .....	100
6.2	Optional Fields Form .....	100
6.3	Candidate Information Form .....	106
6.4	Examination Information Form .....	106
6.5	Candidate Printouts Form .....	109
6.6	Sort Room Files Form .....	109
6.7	Staff Printouts Form .....	113
A.1	Organization Schema of OSYM .....	122
B.1	Functional Schema of Data Processing Unit .....	127
B.2	Overall Schema of Computer System .....	128
B.3	Overall Schema of Optical Reader Systems .....	129

## 1. INTRODUCTION

Registration and Examination concepts are terms being frequently encountered in daily life. In order to develop an information system based on these concepts, two master theses studies complementing each other have been implemented. These theses are designed for the applications in the Student Selection and Placement Center (ÖSYM) in Turkey.

The main purpose of ÖSYM is selection and placement of students for higher education institutions in Turkey. For this purpose a project named "Student Selection and Placement System (ÖSYS)" is being carried out [ÖSYM, 1980] [ÖSYM, 1984] [PAYASLIOĞLU, A., 1985] [TOKER, F., GÜNALP A., 1982]. In addition to this basic project, ÖSYM carries out many other projects. Most of these projects are the requests of examination organizations coming from various institutions.

These projects are being carried out by adapting the programs which were developed for the ÖSYS project. This project requires a complex organization in addition to systematic and fairly ordered data processing methods and programs. For this reason, it is not suitable to use a project directed to a specific application, when the examination organizations which have different properties.

Besides that, the existing examination organization is executed by various subunits whose duties are predefined [ÖSYM, 1985a]. However, in this structure, any failure in one of these subunits, effects the entire project. That is to say, the subunits are too much dependent on each other. Whereas, in this kind of project, the subunits must be independent as much as possible and must be conceptually whole. That means, the subunits must be functionally cohesive [AKTAŞ, A.Z., 1987] [BROOKS, C.H.P. et al., 1982] [WEINBERG, V., 1980].

For this purpose, it was necessary to implement an information system which is modular, general purpose, structured and functionally cohesive that can be used in any type of examination organizations within the ÖSYM.

Taking these requirements as a starting point, phases of the information system life cycle which are analysis, logical design, physical design and implementation have been followed [AKTAŞ, A.Z., 1987] [BROOKS, C.H.P. et al., 1982] [BURCH, J.G., STRATER, F.R., 1984] [WEINBERG, V., 1980]. During understanding the existing system, many documents prepared by ÖSYM for various examination organizations have been analyzed [ÖSYM, 1985a] [ÖSYM, 1985b] [ÖSYM, 1985c] and the proposed system has been developed by means of two master theses.

The first master thesis, which discusses the registration of the candidates and determination of the executive staff necessary for the examination is titled "Design and Implementation of an Automated Registration System".

The second master thesis, which complements the first one and discusses on the prearrangements and evaluation of the examination is titled "Design and Implementation of an Automated Examination System" [TÜTÜNCÜ, M., 1987].

These two theses together are forming an information system titled "Registration and Examination System" (REX).

### 1.1. Purpose and Scope of REX

The purpose of REX is to develop software that implements the system which will perform the examination organization automatically beginning from the application of the candidates up to the determination and announcement of the examination results.

REX improves the ÖSYS project as it was developed as a package program which can be used for every kind of examination organization of ÖSYM.

During the development of the REX software a structured system analysis study was performed. Data Flow Diagram (DFD) and Warnier-Orr Diagrams were selected as tools [AKTAŞ, A.Z., 1987] [BROOKS, C.H.P. et al., 1982] [WEINBERG, V., 1980].

As a result of the analysis of the existing system, some new concepts were developed and the proposed Examination Organization System was constructed on these concepts. These concepts are the user name which identifies the examination, the examination centers in which the examination will be performed, the sessions which are the parts of the examination, the areas which are the study groups and the subareas which are the group of questions in the area.

The subsystems which comprise the Examination Organization System are: Registration Supporting Subsystem, Examination Supporting Subsystem, Test Preparation Subsystem and REX Subsystem. Supporting Subsystems are the interfaces between external environment and REX Subsystem. Test Preparation Subsystem produces the tests which will be used in the examination. REX Subsystem is the software section which processes the the data and produces the necessary printouts of an examination organization.

REX consists of subsystems that may transfer data one to another. Some of the programs in the subsystems are of specific, and others of general purpose type. The general purpose programs are not interpreters. They are software that have the properties of "program writing programs", which run in an interactive environment. All the programs guide the users with menus at each stage while the system is running.

## 1.2. Subsystems of REX

REX consists of three main subsystems.

i)Registration Subsystem: A Candidate File is created from the information given by the candidates having applied for admission to the examination. Some information controls are performed on this file in order to correct the errors. Staff Files are created to record the information belonging to the persons that will be charged with the responsibility of the examination. Using the information of these files the necessary actions are performed.

ii)Examination Subsystem: Using the information of the Candidate File, the Building, Room and Examination Files are created. The rooms in which the candidates will take the examination are determined. Necessary actions are performed to transfer the documents belonging to the staff which was determined by the Registration Subsystem and to transfer the documents to be used in the examination. After the examination, the evaluation process is performed and the results are obtained.

iii)Supporting Subsystem: The purpose of this subsystem is to perform the necessary activities to ensure the use of system program packages. It also takes the necessary security measures and impedes the formation of inconsistent structures.

## 1.3. Implementation Notes on REX

REX is developed on Burroughs mainframe systems and since many parts of the software are machine dependent it is not portable.



Generally REX is implemented by the PL/I programming language. In some cases ALGOL is used to support REX, especially to synchronize the programs which can be run as independent tasks.

Classic file structures and access methods have been used.

According to the characteristics of the application some special algorithms have been developed.

The total size of the software is more than 5000 statements.

Approximately two years have been spent developing this system.

## 2. EXAMINATION AND REGISTRATION CONCEPTS

Since REX is closely related to the registration and examination concepts, general definitions of these concepts will be presented.

### 2.1. Examination Concept

The process of determining the aptitude of individuals within a community, relative to each other, in one or more subject areas, or determining the aptitude of each individual within the community, independently of the other ones, according to some predefined level, is called examination.

Examinations are performed for two purposes.

- i) To select
- ii) To measure

In the examinations in which the purpose is selection, the answers given by each of the candidates are graded and the scores put in a descending order. According to the purpose of the examination, a specified number of candidates are designated successful beginning from the highest score. The score of the last candidate in the list is called "lower limit score". The lower limit score has the following properties.

1) This score can be determined only after the scores of all the candidates have been calculated.

2) When an examination consisting of the same questions is applied to two different communities, the obtained scores will be different because of dissimilarities between the levels of knowledge of the people entering the examination.

In the examinations in which the purpose is to measure, the lower limit score is predetermined. This score is compared with the score obtained from the answers of each candidate and the ones who can not reach the limit are deemed to be unsuccessful.

The point which must be emphasised here is that the lower limit score is determined after the examination in selection type examinations and before the examination in measurement type examinations.

Basically, examinations are performed by using one or more of the following methods.

- i) Written
- ii) Oral
- iii) Applied

Written examinations may be either essay or fixed-answer type. REX is designed for fixed-answer type examinations [PAYASLIOĞLU, A., 1985] [ÖZÇELİK, D.A., 1981]. The reasons are as follows.

-This is the most practical method to be applied for examinations having a large number of applicants like in the case of ÖSYS application in Turkey.

-Test type examinations are more suitable for measuring the knowledge and skills of candidates as they are more objective, standard and reliable when compared with essay type examinations.

-Test results can be evaluated rapidly and correctly via optical readers and computers.

## 2.2. The Registration Concept

In order to carry out the organization of an examination, it is necessary to determine the candidates who will take the examination. Application forms which are designed for this purpose are sent to the candidates. The candidates make their application by filling these forms.

The process of recording the information about candidates obtained from the application forms to be used for examination or other purposes is called, the registration.

The registration process can be handled in two ways depending on whether the candidates to be registered is known or not.

i) If the candidates to be registered is known then the application forms are prepared only for these candidates and sent to them. Renewal registrations in the universities can be an example for this type of registrations.

ii) If the candidates to be registered is not known then the candidates take the application forms from predefined centers and the ones who fill and give the forms are said to be registered. University selection examination registrations can be an example for this type of registrations.

The registration process can be handled in two ways if the application periods are considered.

i) The registrations in which the application period is predetermined. The candidates must apply within the predetermined application period. Registrations for employment exams of firms can be an example for this type of registrations.

ii) The registrations in which the application period is not defined but comes into force after the application of the candidate. Registrations for drivers licence applications can be an example for this type of registrations.

### 3. PROPOSED EXAMINATION ORGANIZATION SYSTEM

Proposed Examination Organization System schema is shown in Figure 3.1 by using Data Flow Diagrams (DFD) together with the files, sources/sinks and data flows. All files, sources/sinks and data flows are described in detail in the following sections.

#### 3.1. Files

All the files used in REX are described below.

**Candidate File:** It is the file in which the information, concerning the candidates having applied to take the examination, are stored. This file is used, to record the information gathered with the Application Form from the candidates by the Registration Subsystem; to transfer the assignment information gathered by the Examination Subsystem prior the examination and the evaluations information after the examination.

**Examination file:** This file is created by selecting the candidates that will be admitted to the examination from the Candidate File, after the completion of the registration. The evaluation of the examination and the determination of the scores is performed by using this file. The scores obtained after the evaluation, are transferred to the Candidate File.

**Room Files:** These are the files that has been formed by choosing sufficient number of rooms, from the General Room File, for the candidates that will be admitted to the examination. There is one Room File for each session of the examination. Room Files are created and used during the room assignment.

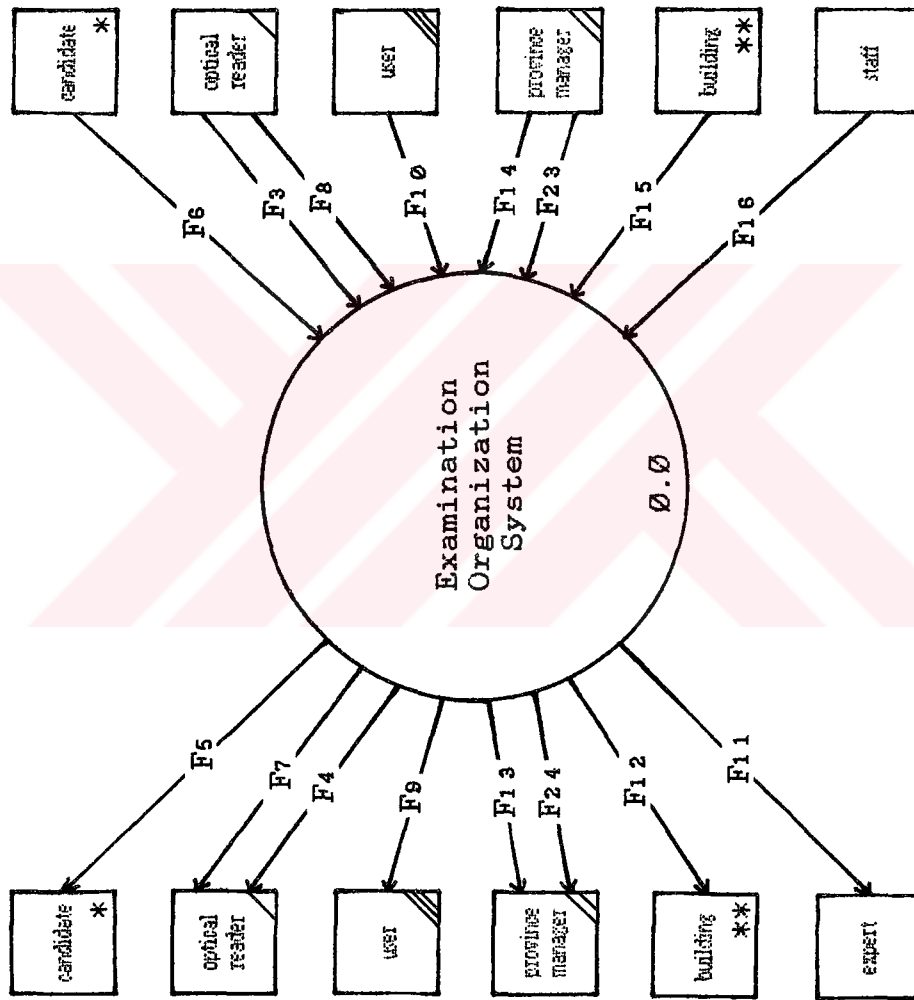


Figure 3.1 DFD of Examination Organization System

Staff Files: The staff files consist of the information concerning the Proctor, Reserve Proctor, Room Director, Building Examination Director, Assistant Building Examination Director and Building Manager appointed in the examination. The staff appearing in these files are determined by the Province Examination Manager. There is one Staff File for each session of the examination. Staff Files are created and used during the staff assignment.

Building File: It is the file in which all the information about the buildings utilized in the examination are stored.

Optical Files: These are the magnetic tape files formed in order to transfer the information of the optical forms read on an Optical Reader to REX for registration, examination or any other purpose.

### 3.2. Sources and Sinks

Candidate: Is the person who wishes to take the examination. The candidates make applications for registration. The places in which they will take the examination and results are declared to the candidates.

User: The person who uses REX is called the user. The user enters commands to REX. In return, REX provides reports on performed studies and guides the user with menus.

Province Examination Management: It is the center which lends the examination documents to the buildings the documents related with the staff to them; and which sends back the examination documents from the buildings.

Building: They are the centers where the documents coming from the Province Examination Management are distributed and collected.

Staff:They are the people who are responsible from the examination execution. They get Duty Assignment Form, Identification Card and Staff Cheque from the Province Examination Management.

Expert:They are the persons who have specialized on their areas and who prepare the questions used in the examination. The examinations which are prepared by these persons, are sent to the Province Examination Management.

Optical Reader:They are the electronic devices on which the optical readings of Application Forms and Answer Sheets are performed.

### 3.3. Data Flows

F<sub>1</sub> representation has been used on the DFD while data flows were being coded. The meanings of data flows are as follows.

F<sub>1</sub>:The data transferred to the Optical File from Optical Reader.

F<sub>2</sub>:The data transferred from the Optical File to the Examination Organization System.

F<sub>3</sub>:Application Forms which are to be read by means of an Optical Reader.

F<sub>4</sub>:The Application Forms which have been damaged during the optical reading process.

F<sub>5</sub>:The Application Forms received from the candidates.

F<sub>6</sub>:The Application Forms, Examination Entrance Forms and Examination Results Forms which are sent to the candidates.



F7:Faulty Answer Sheets.

F8:Answer Sheets which are to be read by means of an Optical Reader.

F9:The commands that the user enters to the system to inform about his/her demand.

F10:The responses, reports and menus that the system offers to the user.

F11:The questions prepared by the experts to be used in the examination.

F12:The examination documents which are to be sent to the system from the building after the examination.

F13:The examination documents which are to be sent to the system from the Province Examination Management after the examination.

F14:The examination documents to be used in examination process.

F15:The examination documents which are sent from the system to the buildings.

F16:The documents which are sent from the system to the examination staff.

F17:Information about the examination staff.

F18:Information about the buildings that are used for the examination.

F19:Information about the rooms that are used for the examination.

F20:Information about the candidates related to the evaluation of the examination.

F21:Information about the registered candidates.

F22:Information which defines and controls the examination organization.

F23:Staff Notification Schedules sent to the Province Examination Management.

F24:Staff Notification Schedules returned from the Province Examination Management.

### 3.4. The Subsystems of the Examination Organization System

The subsystems which form an Examination Organization System, files and sources/sinks related with these subsystems and data flows between them are shown in Figure 3.2 by using DFD.

#### 3.4.1. Registration Supporting Subsystem

It is designed to form a Registration Supporting Subsystem which is formed by manpower and assisting machines to examine the documents which come from the candidates, to correct and determine the faulty cases, to transfer the information on these documents to REX, to regulate and send the documents prepared by REX for candidates. It also sends the Staff Notification Schedules to the Province Examination Management and transfers the information about the staff to the REX. This Supporting Subsystem is communicating with Optical Reader, Province Examination Management and Optical File.

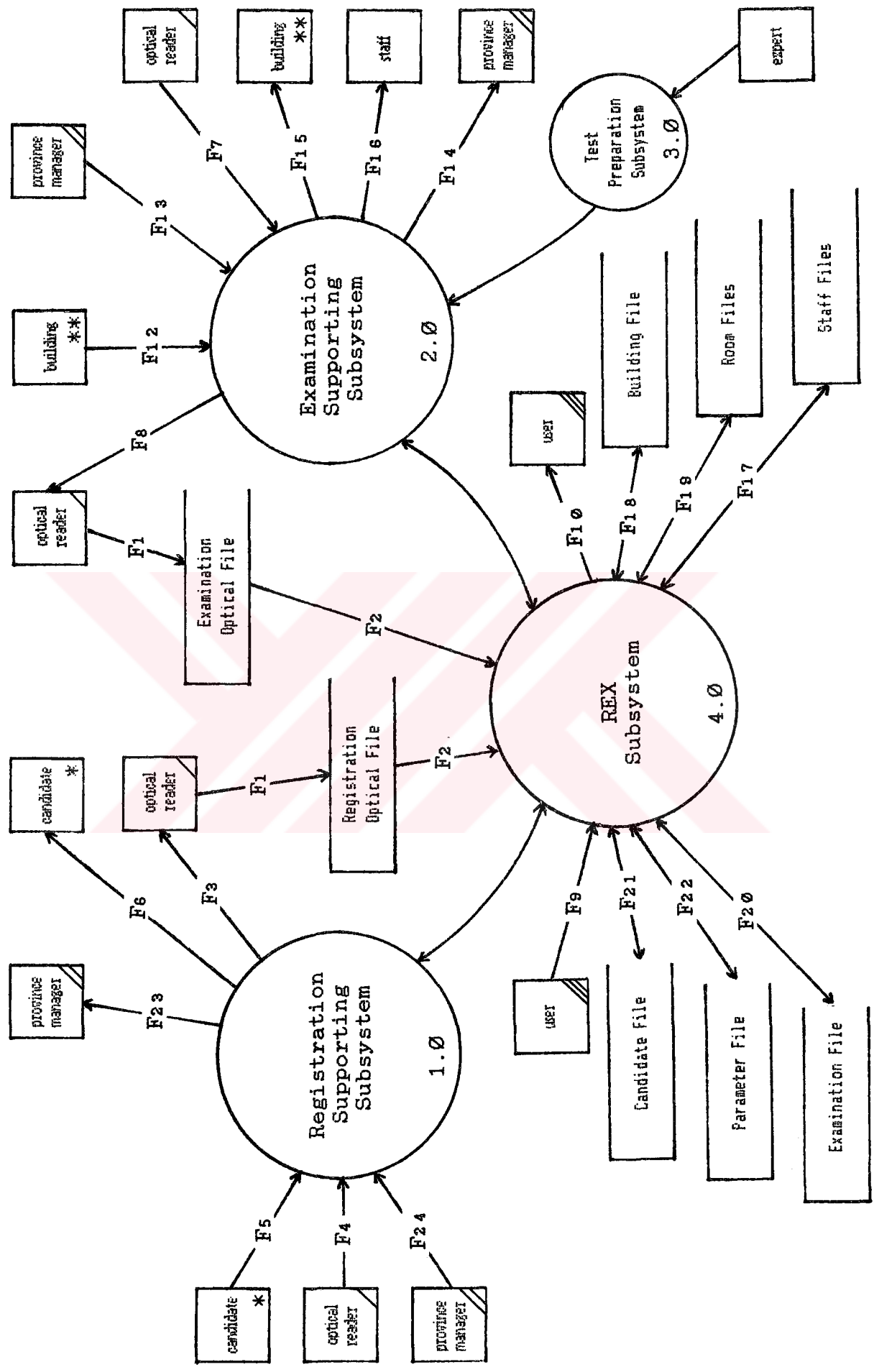


Figure 3.2 DFD of Subsystems of Examination Organization

### 3.4.2. Examination Supporting Subsystem

It is designed to form an Examination Supporting Subsystem which is formed by manpower assisting and machines to prepare all printed documents to be sent to the Province Examination Management before the examination, to receive the documents after examination and to transfer the candidates answers to REX. This Supporting Subsystem is communicating with the Province Examination Management, Building, Staff, Optical Reader and Optical File.

### 3.4.3. Test Preparing Subsystem

It is the subsystem which prepares and prints the questions to be used in the examination, in the booklet form. This subsystem transfers the question booklets to the Examination Supporting Subsystem to be sent to the Province Examination Management.

### 3.4.4. REX Subsystem

This is the subsystem in which all data processing activities related to the examination organization is carried out. This subsystem processes the information coming from the supporting subsystems to obtain necessary printouts. These printouts are transferred to sources/sinks via the Supporting Subsystems.

The subsystems forming REX, files and sources/sinks related with these subsystems and data flows between them are shown in Figure 3.3 by DFD.

In this section, the concepts, subsystems, implementation, properties and requirements of REX will be explained.

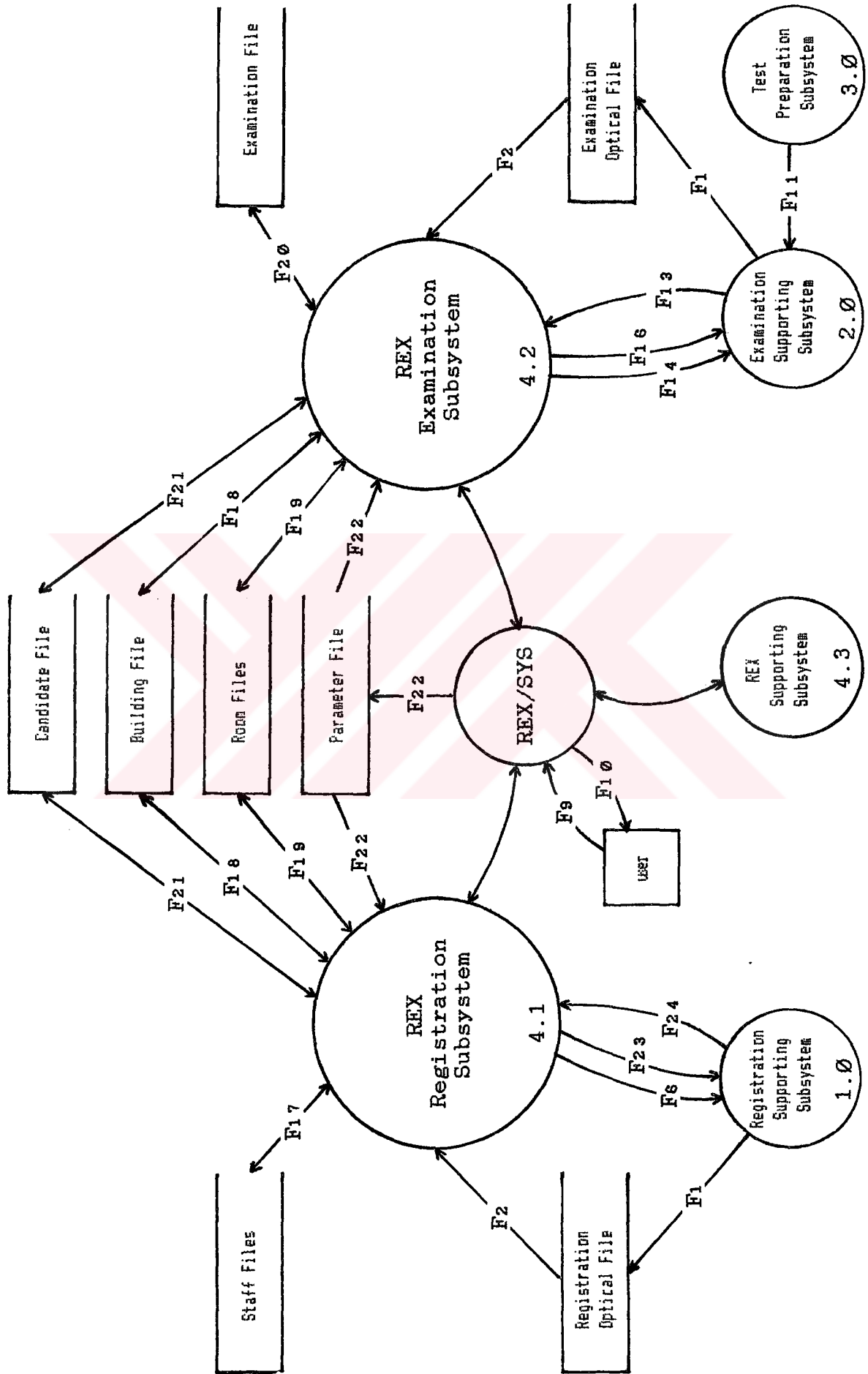


Figure 3.3 DFD of Subsystems of REX

#### 3.4.4.1. Concepts of REX

As a result of the analysis of the existing system it has been conceived that the implementation of a general purpose examination organization may be settled on the following five concepts.

- i) User system name which identifies the examination
- ii) Number of centers in which the examination will be performed
- iii) Number of sessions in the examination
- iv) Number of areas in each session
- v) Number of subareas in each area

The information system, which takes these aspects on a general purpose basis, will be of structured and modular type. Since these concepts are forming a basis for the information system, they have been analyzed in details.

##### 3.4.4.1.1. User System

All data files and user programs which are formed by REX's studies on an examination organization are kept as a directory, under a certain name. Its reason is to distinguish the structures of each examination while working on more than one examination organizations at the same time. The name that defines on examination organization is called the User System. This name consist of five alphanumeric characters.

These names may be meaningless, such as ABX2Y. However, it shall be more useful to use abbreviated names indicating the performed examination organization and generally known for this examination.

ÖSS for Student Selection Examination, ÖYS for Student Placement Examination, AÖF for student examinations of Open Education Faculty, YÖS for Foreign Student Examination, SSYB for Ministry of Health and Social Assistance examination can be examples for User System names.

#### 3.4.4.1.2. Examination Center

This is the residence center as province, district etc. in which the examination will be performed. The examination organization is executed in the same manner in each center. A Province Examination Manager is present in each center and is responsible to execute the examination in conformity with the rules and to take the precautions that will resolve the problems that may arise.

#### 3.4.4.1.3. Session

These are the parts of an examination which are performed in different time portions. According to its purpose, an examination may be performed either in one session or in more than one session. In some examinations, the previously performed sessions are used to determine the candidates that will be admitted to the following ones. In those type of examinations the candidates may be allowed to enter in more than one session according to their success. In some other type of examinations the candidates may be admitted into more than one session, without any condition and according to the areas chosen by the candidate.

#### 3.4.4.1.4. Area

The study field, for which it is necessary to prepare different questions is named area. Some of these areas are forming a group according to the subjects they are consisting. The property of these groups is that, the candidates have to take the examination only in one of the subject within the area. Though the candidates may choose one of the group in the area, they may take the examination in more than one area by choosing among the groups of some other areas. For this reason it is not possible to perform an examination for two different groups at the same time. The part of the examination reserved for one group is executed in one session.

#### 3.4.4.1.5. Subarea

The group of questions which are of specific type within the questions of a area is called subarea. The candidate who takes an examination in one area is considered to be admitted to all of the subareas within this area.

#### 3.4.4.2. Relations Between The Concepts

- The user system name identifies the examination.
- The examination is performed in one or more than one center.
- The examination is executed in one or more than one session in each center.
- In each session there is one or more than one area which the candidate may take the examination in only one of them.
- Each area may be divided into subareas.
- The number of subareas are equal for each of the areas in one session.

The relations between the concepts are shown in Figure 3.4.



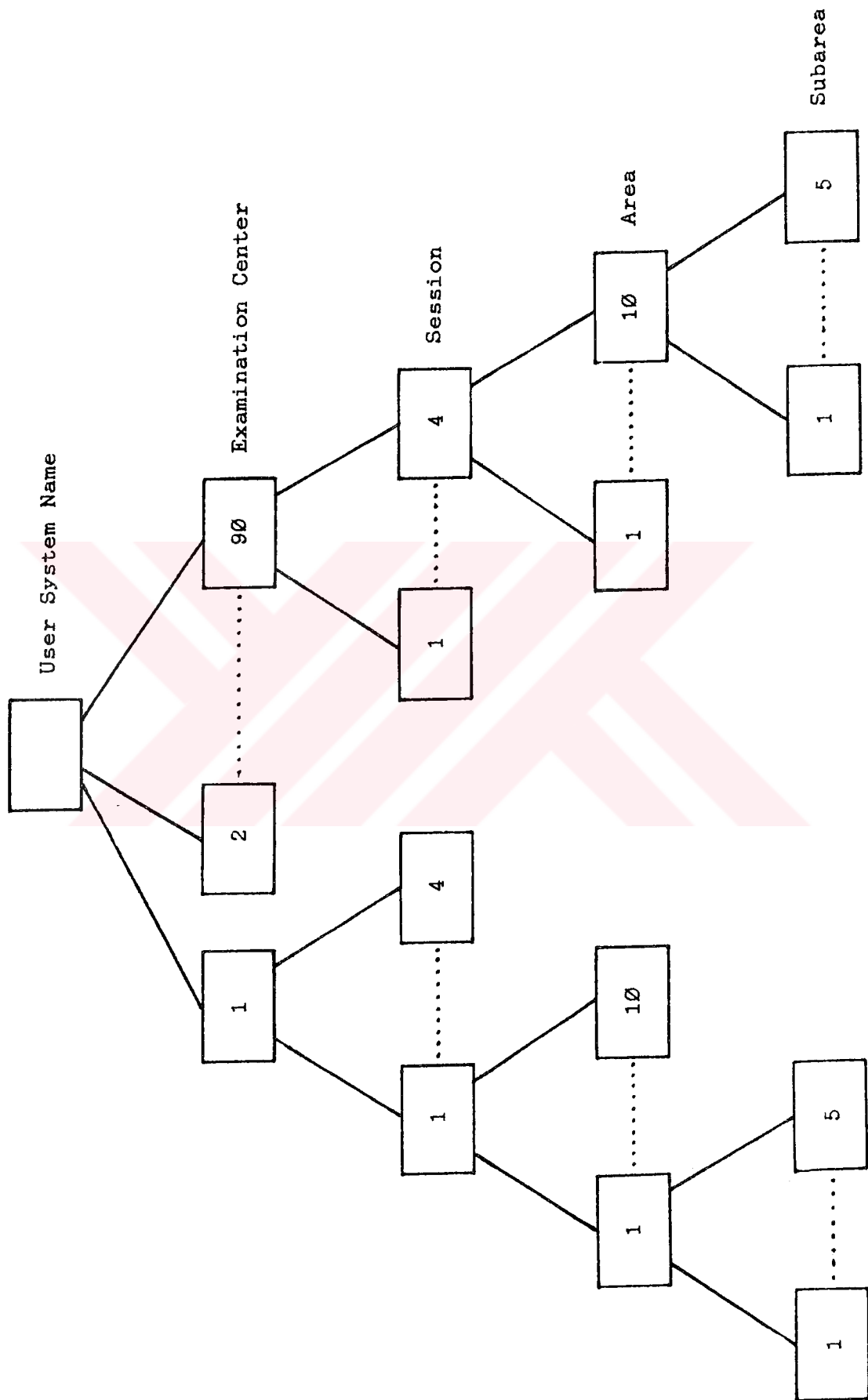


Figure 3.4 Relations between the REX Concepts

It will be useful to examine the four concepts that their descriptions and relations to each other is mentioned above, on the actual applications at ÖSYM.

i) Application 1

In this application, the examination which is utilized for foreign students in the admittance of higher education programs (YÖS) shall be introduced.

This examination is performed in seven centers, six of them are in Turkey.

It is performed in one session.

All the candidates take the same examination, it is to say, there is one area.

This area is divided into two subareas.

- Basic Learning Skills test
- Turkish Language Proficiency test

ii) Application 2

Assistanship examination of Ministry of Health and Social Assistance shall be introduced in this application.

This examination is performed in Ankara, only in one center.

It is applied in two sessions, one is on Saturday and the other on Sunday.

In the first session, there are six areas as English, French, German, Italian, Spanish and Russian.

In the second session, there are three areas as General Medicine, General Microbiology and General Biochemistry.

All the candidates take one of the areas of the first session. They can be admitted for the second session only if they have succeeded in the first session's test.

In this application there are no subareas.

iii) Application 3

In this application, the student placement examination (ÖYS) which is performed for placing students to Turkish higher education programs, shall be introduced.

This examination is performed presently in twenty one examination centers, twenty of them are in Turkish provinces, one is in Lefkoşa, KKTC.

It is applied in one session.

All the candidates take the same examination, that is to say, there is only one area. This area is divided into five subareas as Turkish, Mathematics, Science, Social Science and Foreign Language.

iv) Application 4

In this application, the midterm, final and make-up examinations of Eskişehir Anadolu University Open Education Faculty's (AÖF) students, shall be introduced.

Instruction period of AÖF is four years. The faculty has two programs as Economics and Business Administration. It is required for the students to take eight courses in each year. In the first two years, all

the students have to take the general courses without distinction. Program distinctions take place in third and fourth years. In these classes, five courses are taken in common and three courses with program separations. So, there is a total of thirty eight courses to be studied.

Promotion rule is applied in AÖF. To be able to promote, a student has to succeed all courses or must have been failed in maximum two of the courses. A student who has failed in one or two courses is sentenced to be on probation and takes these courses together with the eight courses of the next class. A student who has failed in more than two courses, repeats these courses. In this case, minimum three, maximum ten courses can be taken. This rule excludes only last year students. A last year student who has failed in one or two courses can not promote to a higher class, so, he/she takes one or two courses in the next year. A student who has completed all courses, becomes a graduate. If no amnesty law is applied, a student who has failed twice in one course without any intervals is dismissed.

After these explanations, the concepts given for REX can be considered as follows for this application.

There are eighteen centers for this examination.

The examination is applied in four sessions as on Saturday morning, Saturday afternoon, Sunday morning and Sunday afternoon.

First and third year students take examinations for the first four courses on Saturday morning and the rest four courses on Sunday morning, second and fourth year students take examinations for the first four courses on Saturday afternoon and the rest four courses on Sunday afternoon.

All students take the examination in two sessions. However, the students who are on probation from the previous class, can be exposed to participate the third and fourth sessions for their probational courses. For the same reason, repeat students, can take the examination in one session, if the related courses are given in the same session. In summary, the students can take the examination in maximum four sessions, depending on their courses.

A total of twelve areas exist in these four sessions. These areas and sessions are as follows.

- 1) First year, first four courses: Saturday morning.
- 2) First year, second four courses: Sunday morning.
- 3) Second year, first four courses: Saturday afternoon.
- 4) Second year, second four courses: Sunday afternoon.
- 5) Third year, first four courses in economics: Saturday morning.
- 6) Third year, second four courses in economics: Sunday morning.
- 7) Third year, first four courses in business administration: Saturday morning.
- 8) Third year, second four courses in business administration: Sunday morning.
- 9) Fourth year, first four courses in economics: Saturday afternoon.
- 10) Fourth year, second four courses in economics: Sunday afternoon.
- 11) Fourth year, first four courses in business administration: Saturday afternoon.
- 12) Fourth year, second four courses in business administration: Sunday afternoon.

### 3.4.4.3. Subsystems of REX

#### 3.4.4.3.1 Registration Subsystem

The actions executed by this subsystem are listed below.

-It forms the candidate file depending on the information gathered from the candidates and to the parameters showing the number of center, area and session in which the examination will be executed.

-It prints the Application Forms.

-It records the candidate information to the file.

-It prints the control listings.

-It records the updates caused by faulty information.

-It supplies the printouts and the statistical schedules for the future stages at the completion of the transactions on the candidate file.

-It warns the Examination Supporting Subsystem for the printing of the Examination Entrance Form and Examination Results Form.

-It sends the list of all the buildings and rooms to be used in the examination in order to determine the staff to the Province Examination Management.

-It forms the staff file using the information concerning the staff.

-It prints all the documents related to the candidates and the staff.

#### 3.4.4.3.2. Examination Subsystem

The actions executed by this subsystem are listed below.

-It forms the Building, Room and Examination Files.

-It makes the room assignment using the distributions taken from the candidate file delivered by the Registration Subsystem.

-It prints all the documents related to execution of the examination.

-It transfers the information concerning the places in which the candidates have taken the examination to the candidate file, at the end of the room assignment.

-It supplies the raw, standard and weighted scores by evaluating the examination.

-It transfers the examination results to the candidate file.

#### 3.4.4.3.3. Supporting Subsystem

The actions executed by this subsystem are listed below.

-It supplies the necessary logs.

-It takes the necessary precautions towards the facts that may arise in case of a rupture.

-It warns the user when a rupture has occurred during the previous operations and lead the system to the first encountered unfaulty point.

-It takes the necessary measures in order to ensure the security in the file accessing and program operating levels, while the system is used.

-It ensures the operation of some program packages concerning REX in the system.

-It performs the works like copying, deleting, listing, checking the presence, analyzing in an interactive environment, changing the name of the files organized with REX.

#### 3.4.4.4. Physical Implementation of REX

Rex consists of three modules which run under the control of a main program. The name of the main program in the system is REX/SYS. The three modules start the Registration, Examination, Supporting systems and they are named as REX/REGISTRATION, REX/EXAMINATION, REX/SUPPORTING respectively.

When REX is used, the main program is activated and according to the choice of the user, one of the three modules is run. When one of the modules began to run, the main program enters to the waiting state. When the module has terminated its task, the main program continues to run. The module which is run under the control of the main program may activate another submodule if necessary. In this case the module activated from the main program will wait till the task of the submodule is terminated. So the main program waits the module, and the module waits the submodule. The schema concerning this case is shown in Figure 3.5.

#### 3.4.4.5. How to Run REX

The user who wants to activate REX, must initiate a session with the usercode REX and run the program REX/SYS from a terminal.

When REX/SYS is run, the form/menu shown in Figure 3.6 is displayed on the screen. On the fields of this form/menu, the <user system name> and the <password> must be entered. However, the user may want to see the present user systems by pressing the SPCFY key. In this case, using the form given in Figure 3.7 all the user system names are displayed on the screen. It is also possible to terminate the session by entering the word QUIT.

When a <user system name> and a <password> is entered, it is checked whether this user system has been previously created or not. If it has been previously created, then the session is directly started under this user system. In this stage, the <station number> is added to the Parameter File named <user system name>/PARAMS and this file is copied with the name <station number>/PARAMS. This is done, in order to get the general information about the user system, by means of the <station number>.



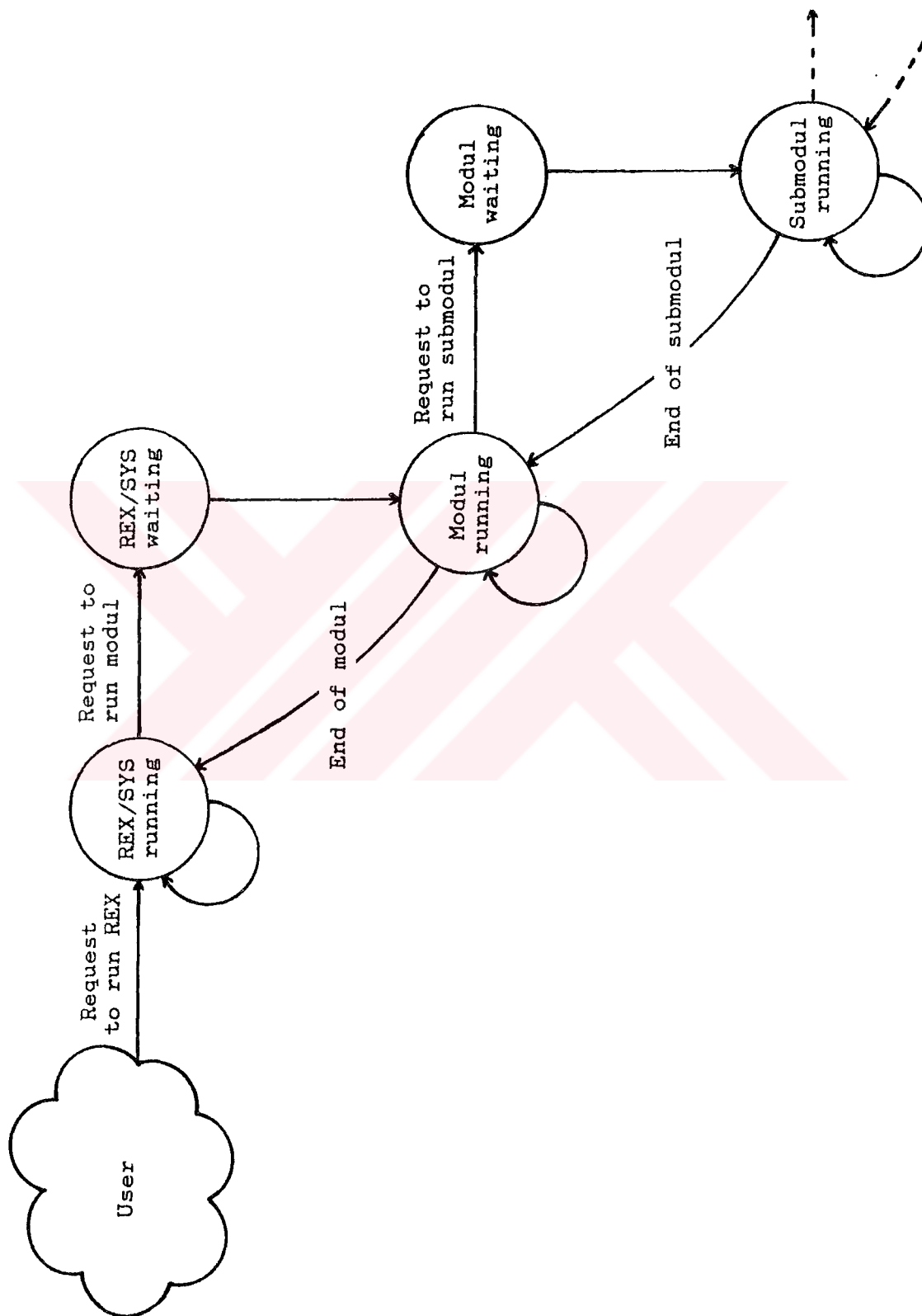


Figure 3.5 Possible States of the Moduls

```

RRRRRRRRR      EEEEEEEEEEE      XXX      XXX
RRR   RRR      EEE                XXX   XXX
RRR   RRR      EEE                XXX   XXX
RRRRRRRRR      EEEEEEEEEEE                XXXX
RRR   RRR      EEE                XXX   XXX
RRR   RRR      EEE                XXX   XXX
RRR   RRR      EEEEEEEEEEE                XXX   XXX

REGISTRATION & EXAMINATION SYSTEM

(C) 1986, Tutuncu & Argun

Kullanıcı Sistemi adı ) (
                        Password ) (

-----
Mevcut "Kullanıcı Sistemlerini" görmek için SPCFY tuşuna basınız.
Bitirmek için QUIT giriniz.

```

Figure 3.6 REX\_Main Menu

REX - (KULLANICI SİSTEMLERİ)

---

	<u>Kullanıcı Sistemi adı</u>	<u>Yaratma tarihi</u>	<u>Son erişim tarihi</u>
1	) (	) (	) (
2	) (	) (	) (
3	) (	) (	) (
4	) (	) (	) (
5	) (	) (	) (
6	) (	) (	) (
7	) (	) (	) (
8	) (	) (	) (
9	) (	) (	) (
10	) (	) (	) (
11	) (	) (	) (
12	) (	) (	) (
13	) (	) (	) (
14	) (	) (	) (
15	) (	) (	) (

---

Figure 3.7 REX\_Existing User Systems Form

If the <user system name> entered using the form/menu given in Figure 3.6 is not present in the system, a Parameter File is created using the form shown in Figure 3.8. After this stage, the menu shown in Figure 3.9 is used to activate one of the subsystems of REX and all the functions are executed in the chosen subsystem.

#### 3.4.4.6. Properties of REX

-Some menus are used in order to give an idea about how to proceed on, to the users who do not know the characteristics of the software.

-It is assumed that, if some structures are formed in REX, these must be known by the users.

-REX is designed to be reentrant for multi user access.

-The subsystems of REX consist of more than one programs running synchronously.

-Special structures are used to obtain synchronization between programs.

-The users shall be able to stop working and continue from the point where they left working.

-REX can handle the examination for at most 100.000 persons.

-The examination can be performed in maximum 90 centers, 4 sessions and 10 areas in each sessions.

-Each area can be divided into 5 subareas.

-In every subarea, maximum 200 questions can be asked, however the total number of questions per area can not exceed 500.

REX - (SINAV PARAMETRELERİ)

---

Sınavın açık adı ) (

Sınav Merkezi sayısı ) ( Sınav Merkezi kodları

1	)	)	)	)	)	)	)	)	)	)	)	)	)	)	)	)
16	)	)	)	)	)	)	)	)	)	)	)	)	)	)	)	)
31	)	)	)	)	)	)	)	)	)	)	)	)	)	)	)	)
46	)	)	)	)	)	)	)	)	)	)	)	)	)	)	)	)
61	)	)	)	)	)	)	)	)	)	)	)	)	)	)	)	)
76	)	)	)	)	)	)	)	)	)	)	)	)	)	)	)	)

Oturum sayısı ) ( Alan sayısı Alt alan sayısı Sınav tarihi Sınav saati

1	)	(	)	(	)	(	)	(	)	(	)	(	)	(
2	)	(	)	(	)	(	)	(	)	(	)	(	)	(
3	)	(	)	(	)	(	)	(	)	(	)	(	)	(
4	)	(	)	(	)	(	)	(	)	(	)	(	)	(

Beklenen aday sayısı ) ( GERİ DÖNÜŞ ) (

---

Figure 3.8 REX\_User System Parameters Form

REX - (ALTSİSTEMLER)

---

) ( KAYIT altsistemi

) ( SINAV altsistemi

) ( DESTEK altsistemi

Altsistemlerden birini seçerek SPCFY tuşuna basınız.

GERİ DÖNÜŞ ) (

---

Figure 3.9 REX\_Subsystems Form

-An individual score for each subarea and a success score obtained from the combination of subarea shall be calculated.

#### 3.4.4.7. System Requirements of REX

-REX is designed for Burroughs Mainframe Systems.

-It can operate with operating systems like MCP 3.5 and later versions.

-The terminals to be used must be ET series or equivalent.

-The MCS to be used must be \*SYSTEM/CANDE.

-The value of the LAISSEFILE, one of the operating parameters of \*SYSTEM/CANDE, must be 6.

-A usercode as REX must be defined in the system. The password of this usercode must be REX, too.

-The following softwares need to be present in the system in order to activate the Supporting Subsystem:

\*SMFII/LOGCONSOLIDATOR,  
\*SMFII/QUERY,  
\*SYSTEM/EDITOR,  
\*SYSTEM/DUMPALL

#### 4. LOGICAL DESIGN OF REGISTRATION SYSTEM

The structure chart of the Examination Organization System is shown in Figure 4.1. REX Registration Subsystem with the Registration Supporting Subsystem form the Registration System.

REX Registration Subsystem which is one of the three subsystems of REX, is responsible from the processes related to the candidates such as collection, storage and confirmation of candidate information and printing of the documents to be sent to the candidates. Besides these, determination of examination executive staff and processing of their data related to the examination and preparation of some printouts to be used in the examination is also performed by this subsystem.

REX Registration Subsystem is continuously communicating with the Registration Supporting Subsystem which is a subsystem of the Examination Organization System. Registration Supporting Subsystem provides the connection of REX with the sources/sinks. The processes like sending the documents to the candidates and Province Examination Management and transferring the information on the returned documents to REX, after checking, are provided by this subsystem.

The general logical structure of the Registration System with the files used in this structure, sources/sinks, data flows between the subsystems are shown in Figure 4.2 by using Data Flow Diagrams (DFD). The data flows are coded by using  $D_i$  notation. The same representation is used in the lower levels of the logical structure. The meanings of data flows are as follows.

$D_1$ : The Application Forms which are sent to the candidates for application.

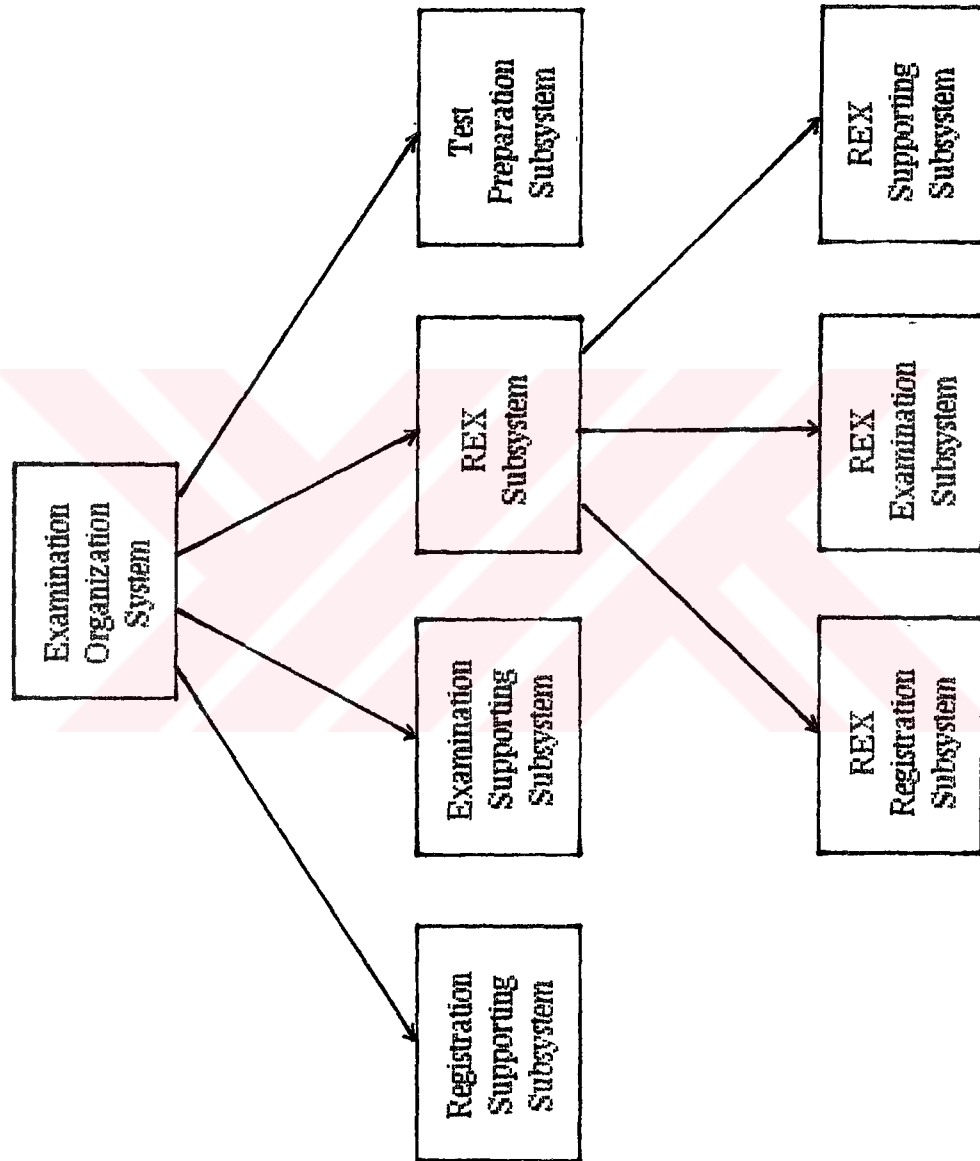


Figure 4.1 Examination Organization System Structure

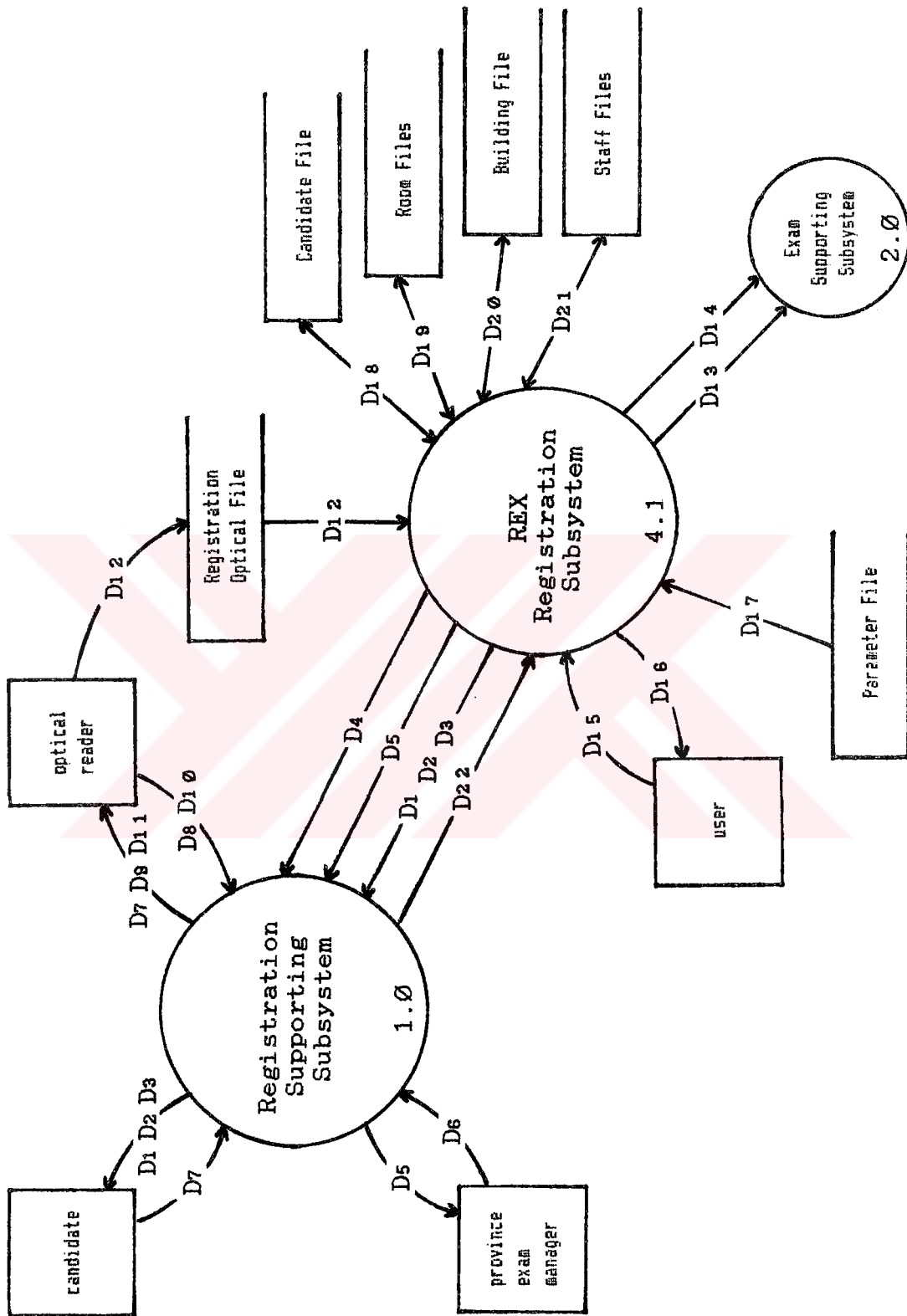


Figure 4.2 DFD of Registration System



D2: The Examination Entrance Forms sent to the candidates to indicate their examination places.

D3: The Examination Results Forms which are sent to the candidates to show their examination results.

D4: The Control Lists which are printed by processing the information on Application Forms via REX and are used for determination of faulty cases.

D5: The Staff Notification Schedules which are sent to the Province Examination Management for identification of appointed staff of examination.

D6: The Staff Notification Schedules which are sent from the Province Examination Management after the identification of the staff.

D7: The Application Forms which will be optically read for the first time.

D8: The separated Application Forms which are damaged or their faults are identified during the optical reading process.

D9: The Application Forms which are to be read again after separation and correction of faulty parts.

D10: The Application Forms which are read optically with no errors.

D11: The Application Forms which are to be read again after their correction by means of comparing with the Control Lists.

D12: The candidate information which is provided by reading of the Application Forms.

D13: The center\_session\_area distribution which is obtained to be sent to the Examination System after the completion of registration process.

D14: The documents which are related to the staff, printed to be sent to the Examination System after processing of staff information.

D15: The commands which are entered to REX by the user.

D16: The responses and reports given by REX to the user.

D17: The examination describing information read from the Parameter File.

D18: The candidate information written on the Candidate File or read from this file.

D19: The room information written on the Room Files or read from this file.

D20: The building information written on the Building File or read from this file.

D21: The staff information written on the Staff Files or read from this file.

D22: The staff information on the Staff Notification Schedules.

#### 4.1. Registration Supporting Subsystem

Most of the activities in the Registration Supporting Subsystem are performed by manpower with the help of assisting machines. The functions of this subsystem can be summarized as follows.

- Sending the Application Forms to the candidates.
- Receiving the forms coming from the candidates.
- Preparing the forms for optical reading by checking.
- Executing the optical reading of the forms.
- Repairing the damaged forms during optical reading.
- Comparing the optically read forms with the Control Lists.
- Correction of the forms which are understood to have errors.
- Archiving the forms which are read with no errors.
- Sending the Examination Entrance Forms and Examination Results Forms to the candidates.
- Sending the Staff Notification Schedules to the Province Examination Management.
- Receiving the schedules which are sent back from the Province Examination Management.
- Transferring the staff information to REX.

The Registration Supporting Subsystem is divided into various processes which perform one or more of these duties. These processes, sources/sinks, files and the data flows between them are shown in Figure 4.3 by using DFD. The function of each process is as follows.

#### i) Sending the Forms

The function of this process is to provide the mailing of Application Forms, Examination Entrance Forms and Examination Results Forms to the candidates. As these documents are printed on continuous forms, they are separated first and then if required they are matched with the documents like guide or manual etc.

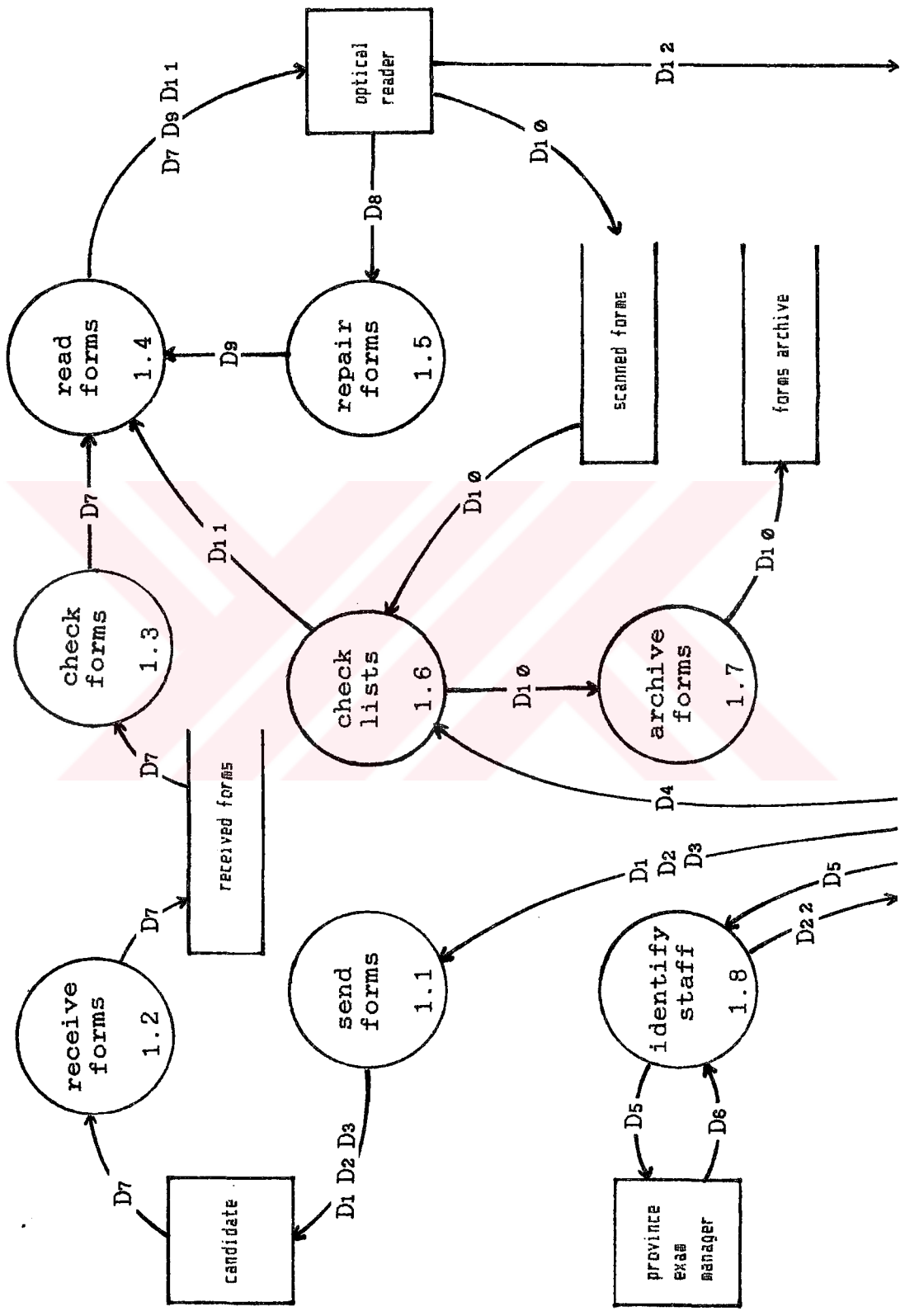


Figure 4.3 DFD of Registration Supporting System

#### ii) Receiving the Forms

The Application Forms which are filled out and sent back by the candidates are received and collected by this process in order to be transferred to Optical Readers. The number of received forms are stored as lists to be compared with the number of candidates recorded by REX Registration Subsystem.

#### iii) Checking the Forms

Before optical reading, the Application Forms are checked visually. This is performed for correcting the coding errors; correcting the inconsistent information; preparing the worn, dirty or faulty forms (filled with pen, not pencil etc.) for optical reading.

#### iv) Optical Reading

In this process, transferring the information on the Application Forms to the magnetic tapes by using Optical Readers, are performed. Optical Readers are computer controlled devices. The fields on the forms, their kinds and order of recording on tape is defined by using a special purpose software. Also, by the use of a high level programming language, some information checks over the form may be performed. The forms which are read correctly by Optical Readers are stored in boxes in order to be compared by the Control Lists which are prepared during the registration of the candidates. However, the forms which are damaged by the candidates can be selected during optical reading process. These forms are transferred for correction to the process which is charged with this duty.

#### v) Reparation of the Forms

The Application Forms which are damaged or which are defined to have errors via software control are again transferred for optical reading by making fair copies or by correcting on the same sheets.

#### vi) Checking the Lists

The Control Lists which are prepared by processing the candidate information transferred to REX are compared with the stored Application Forms. The forms which are determined to have errors by checking the warnings on the Control Lists, are corrected. During the control of the lists, also the unread forms are determined and sent to optical reading with corrected ones.

#### vii) Archiving the Forms

In order to provide an access to the Application Forms which are defined to have no errors by list controls, they are sorted due to the application numbers and put in boxes of thousand.

#### viii) Identification of the Staff

This process is executed by two sections shown in Figure 4.4 by using DFD. Their functions are as follows.

##### a) Sending the Staff Notification Schedules

In this section, the Staff Notification Schedules which are prepared for identification of staff by the Province Examination Management, are sent to the examination centers.

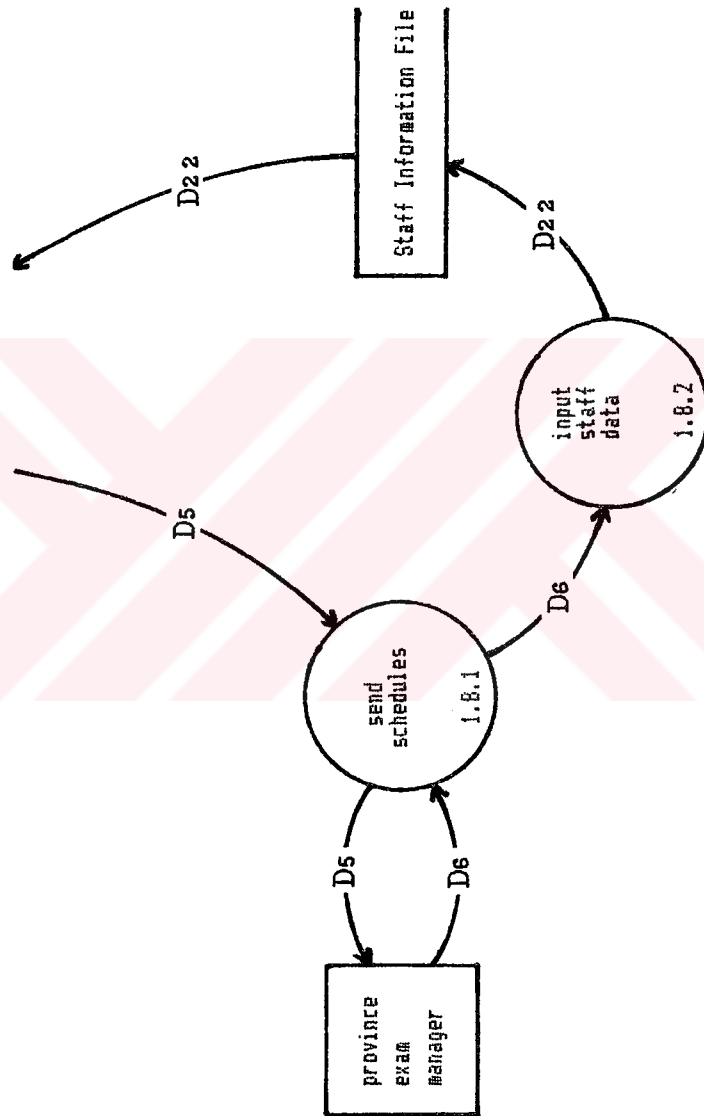


Figure 4.4 DFD of Identification of Staff Process

## b) Inputing the Staff Data

In this section, the process of transferring the information on the Staff Notification Schedules coming from the examination centers, by recording on tapes is performed.

### 4.2. REX Registration Subsystem

REX Registration Subsystem has two main functions. The first of these functions is the execution of processes related to the candidates and the second one is related to the staff. The relation of these processes with other subsystems, files and data flows are shown in Figure 4.5 by using DFD.

#### 4.2.1. Processes Related to the Candidates

The activities performed for the execution of processes related to the candidates by REX Registration Subsystem are summarized below.

- Creation of Candidate File on which the candidate information will be recorded.

- Printing the Application Forms which provides the application of candidates for the examination.

- Transferring the candidate information which is obtained from optical reading to the Candidate File.

- Printing the Control Lists which are used for checking the information recorded on the Candidate File.

- Execution of necessary corrections on the Candidate File.

- Determination and correction of duplicate applications.

- Preparing the distribution which will be used for room assignment.

- Obtaining the printouts related to the candidates.



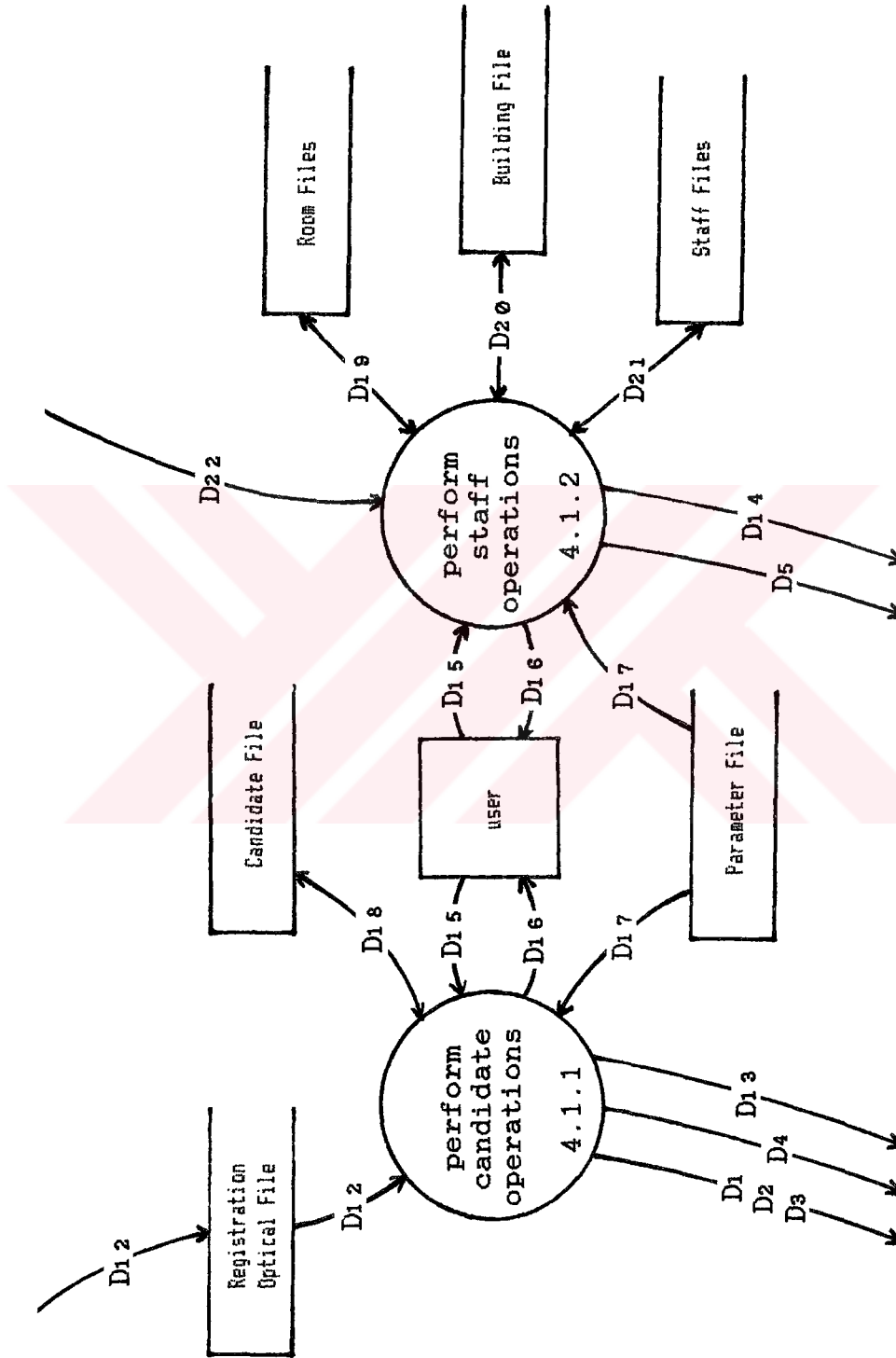


Figure 4.5 DFD of REX Registration Subsystem

These tasks are performed by various processes of REX Registration Subsystem. The schema showing these processes and sources/sinks are given in Figure 4.6 by using DFD. The function of each process is given below.

#### i) Creation of the Candidate File

Candidate File is created by loading the value of 0 (zero) to the numeric fields and space to the alphanumeric fields of the records. The number of expected candidates is used for the size of the file. The application number of each candidate to be registered is entered to the related field during the creation of the file.

#### ii) Printing the Application Forms

The Application Form is used to enable the applications of the candidates for the examination. This form consists of three parts: Candidate Declaration Form, Registration Card and Identification Card.

The Candidate Declaration Form provides the direct transfer of candidate information to the computer by means of Optical Readers. Because of this reason it should be filled with pencil. Identification Card is used for controlling the ID's of candidates in the examination rooms. Registration Card is used to avoid information conflicts with that of the Candidate Declaration Form, by using the information written by the own handwriting of the candidate.

During printing of the Application Forms, the application numbers which have been recorded on the Candidate File before, are printed on three parts of the form.

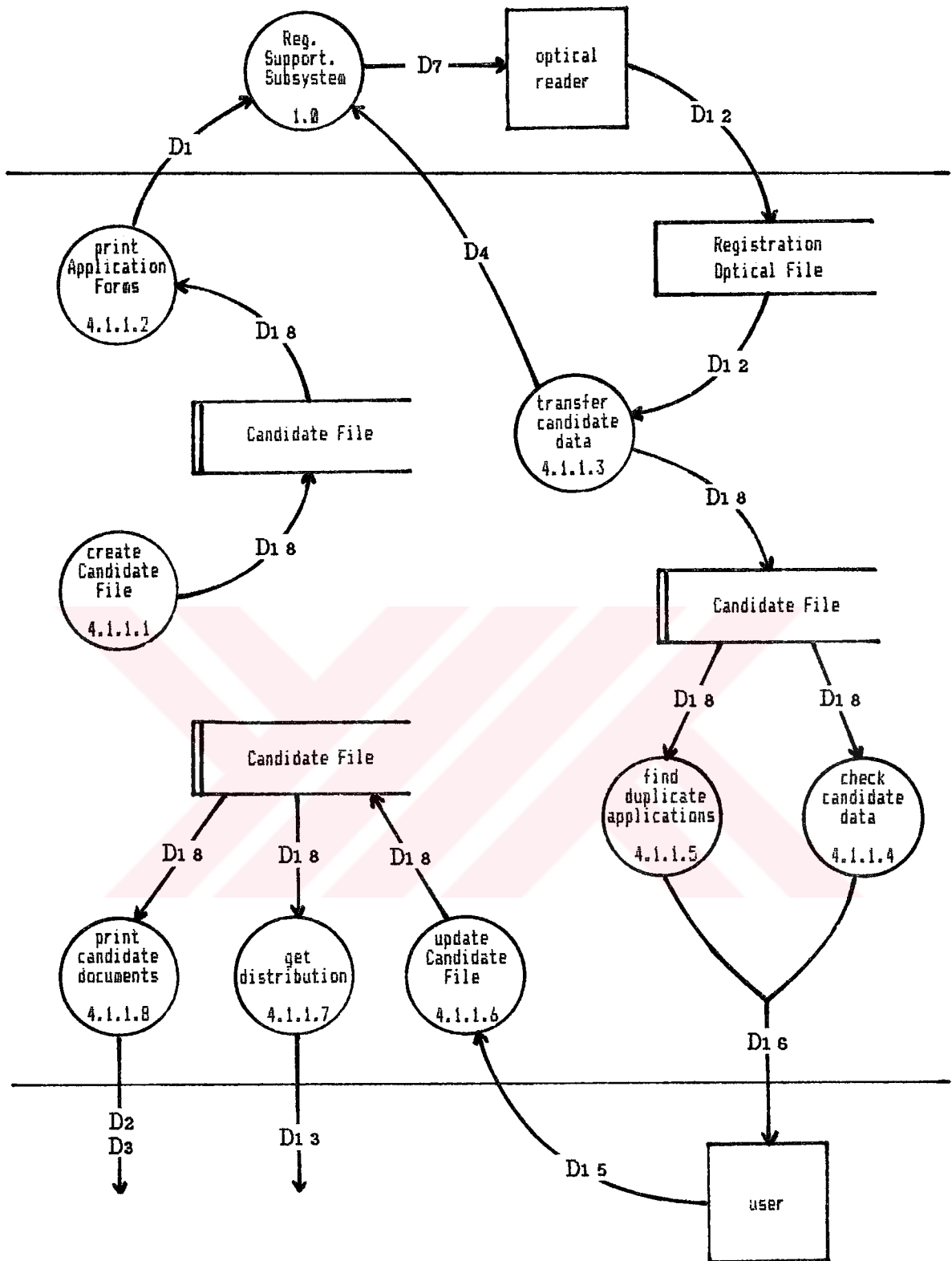


Figure 4.6 DFD of Processes Related to Candidates

### iii) Transfer of the Candidate Data

In this process, the candidate information recorded to the magnetic tapes by means of Optical Readers are transferred to the Candidate File. The Candidate File is the main file to be used during the whole examination and to be kept after the examination, so the information that it contains must be in full and correct. For this purpose, the following controls must be carried out during the transfer of these information to the file.

-Double alphanumeric marks which occur because of marking more than one in each column of the Candidate Declaration Form. These columns contain "\*" (star) signs as Optical Reader outputs.

-Blank fields which are left unfilled by candidates.

-The invalid codes for the fields to which a numeric code is assigned and the full names are given in the tables of the Application Manual (e.g. Examination Center Code).

The other function of this process is the printing of Control Lists. The Control Lists contain the information about all candidates recorded on the Candidate File and the errors determined by the above mentioned controls.

### iv) Identification of the Faulty Cases

In order to eliminate the chance of errors which are not noticed during the control of the lists, after recording all the candidates to the Candidate File, a final control on the fields which are critical for the examination performance is carried out and such cases are listed.

v) Determination of the Duplicate Applications

The Candidate File is sorted by names and surnames, and the ones having same names and surnames are listed. The determination of these ones whether they are different people or duplicate applications is executed by examining of their Application Forms.

vi) Updating the Candidate File

In this process, except for the valid one, the duplicate applications are deleted and the faulty cases are corrected. This process is realized in an interactive environment using a form which consists all information of the candidate.

vii) Preparation of the Center\_Session\_Area  
Distribution

After the controls, the Candidate File is updated by correcting the necessary information and the distribution of candidates for each center, session and area is obtained. This distribution shall be used for room assignment by the Examination System.

viii) Printing the Documents Related to the  
Candidates

In this process, a series of printouts which are to be sent to the candidates or to be used to show all the information of the candidates, are obtained. The related files of these printouts are shown in Figure 4.7 by using DFD. Description of the printouts are as follows.

-Examination Entrance Form: It is printed after that the room assignment has been realized by the Examination system, and it shows where the candidates will take the examination for each session.

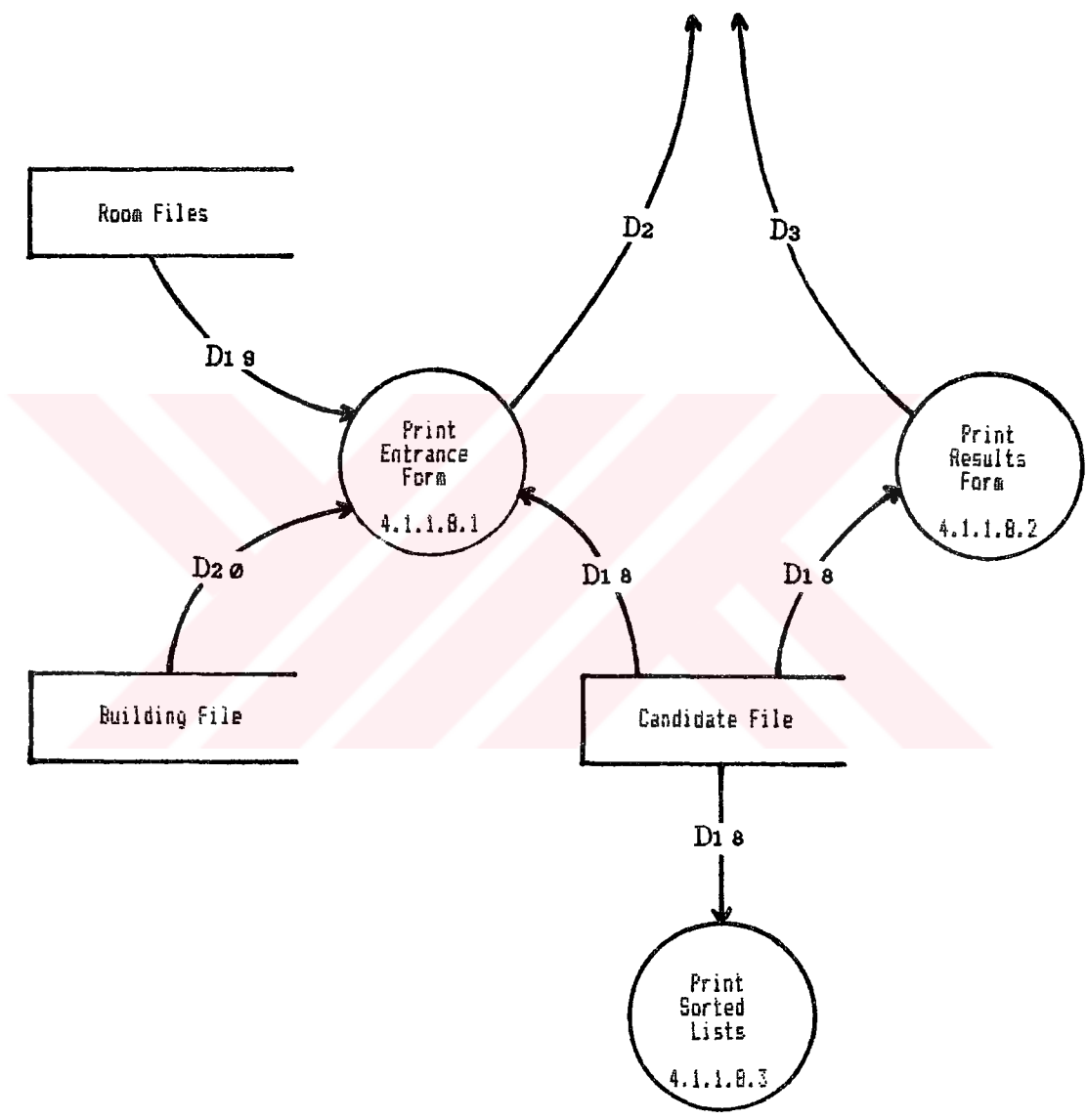


Figure 4.7 DFD of Printouts Concerning the Candidates

-Examination Results Form: The printout which is obtained after that the evaluation of the examination has been realized by the Examination System, is used to show the weighted score and the scores of each subarea that the candidate has got in one session.

-Candidate Lists: These lists are printed separately by sorting according to the application number, weighted score and surname of the candidates and they are used to analyze the scores of all the candidates.

#### 4.2.2. Processes Related to the Staff

The activities performed during the execution of the processes related to the staff by the REX Registration Subsystem, are summarized below.

-Determination of the number of staff necessary for the execution of the examination in each examination center.

-Sorting of the Room Files in order to execute the processes related to the staff.

-Creation of the Staff Files in order to record the staff information.

-Recording of the addresses of the staff file into the Room and Building Files.

-Printing of the Staff Notification Schedules.

-Transferring the staff information to the Staff Files.

-Obtaining of the printouts related to the staff.

These studies are performed by different processes of the REX Registration Subsystem. The schema representing these processes, are shown in Figure 4.8 by using DFD. The function of each process is stated below.

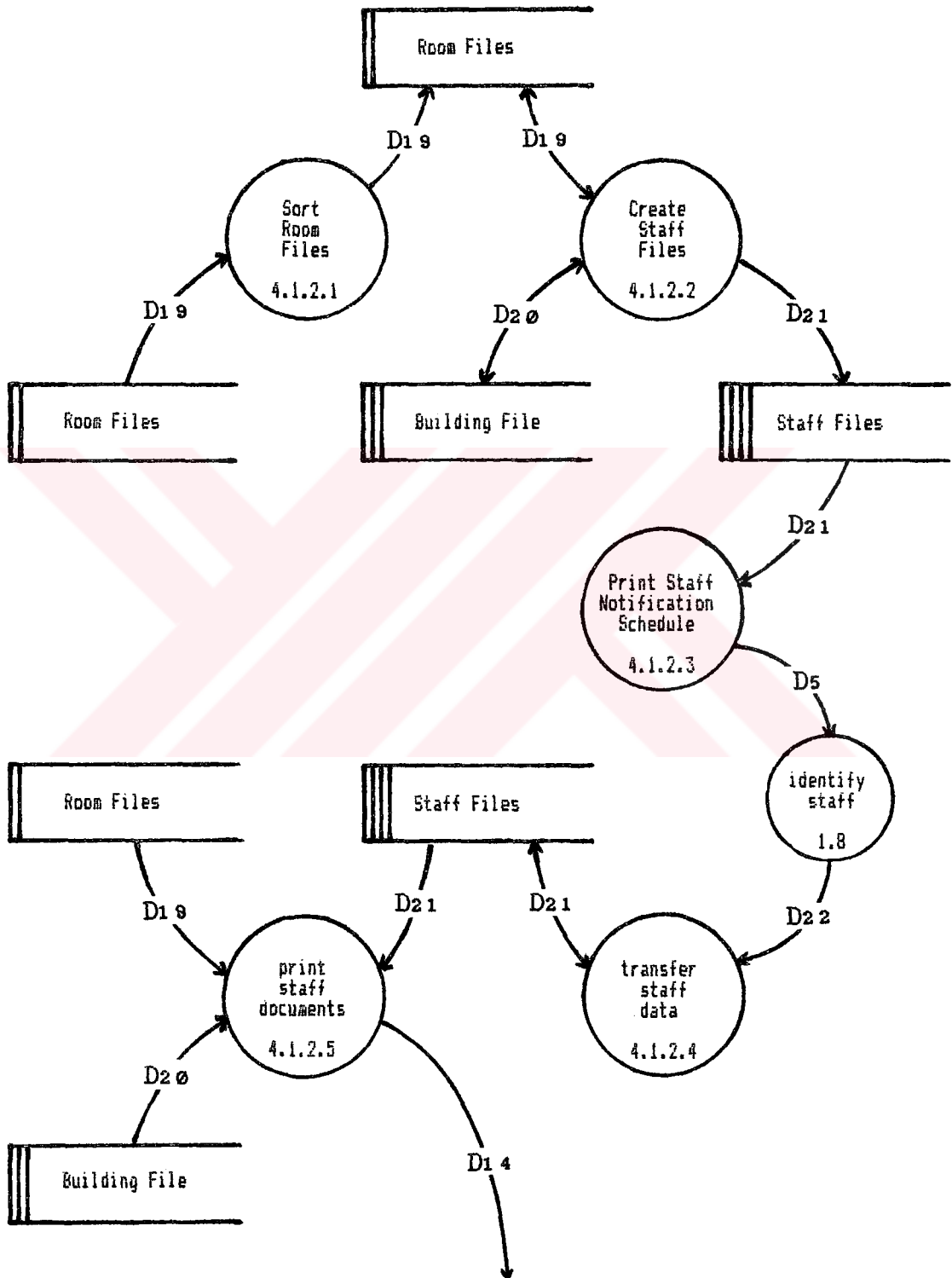


Figure 4.8 DFD of Processes Related to Staff



#### i) Sorting the Room Files

The Room Files created by the REX Examination Subsystem, separately for each of the session, are sorted by the examination center, area code and assignment priority for the room assignment. However the procedures related to the staff and the realization of the printouts requires the Room Files which are sorted by room number within building number. For that reason each Room File is sorted again according to this order.

#### ii) Creation of the Staff Files

In this process, for each center of the examination, the number of staff necessary for each room and building is determined, the addresses of the staff are recorded on the Room and Building Files and a record including the duty code, room and building addresses is added to the Staff Files.

#### iii) Printing the Staff Notification Schedule

These schedules are used by the Province Examination Managements in order to determine the staff to be used. The Staff Notification Schedules are printed separately for each examination center and following the order of the room numbers within the building.

#### iv) Recording the Staff Data

The staff information stated in the schedules send back by the Province Examination Managements after the determination of the staff are transferred to the magnetic tapes. Using these files, the name, surname, title and working institutions of the staff, are recorded on the Staff files.

v) Preparation of the Documents Related to the Staff

In this process, a series of printouts which are to be sent to the examination centers together with the other documents for the Examination System. The related files of these printouts are shown in Figure 4.9 by using DFD. The description of the printouts is as follows.

-Duty Assignment Form: This is used to inform the staff about their duties.

-Identification Card: This is the card which the staff will keep on their collars during the examination.

-Staff Cheque: This is used for the payment of staff after the examination.

-Building Staff List: It displays the staff who are on duty in the buildings and rooms.

-Room Document Receipt/Delivery Report: It is used when the Room Director receives/delivers any document from Building Examination Director.

-Cheque Distribution Report: It is used during the distribution of the staff cheques.

-Cheque Envelope: This is the envelope in which the Staff Cheques and the Cheque Distribution Reports are kept.

-Staff and Duty Sites List: It is the list on which the names and surnames of the staff of each examination center is displayed.

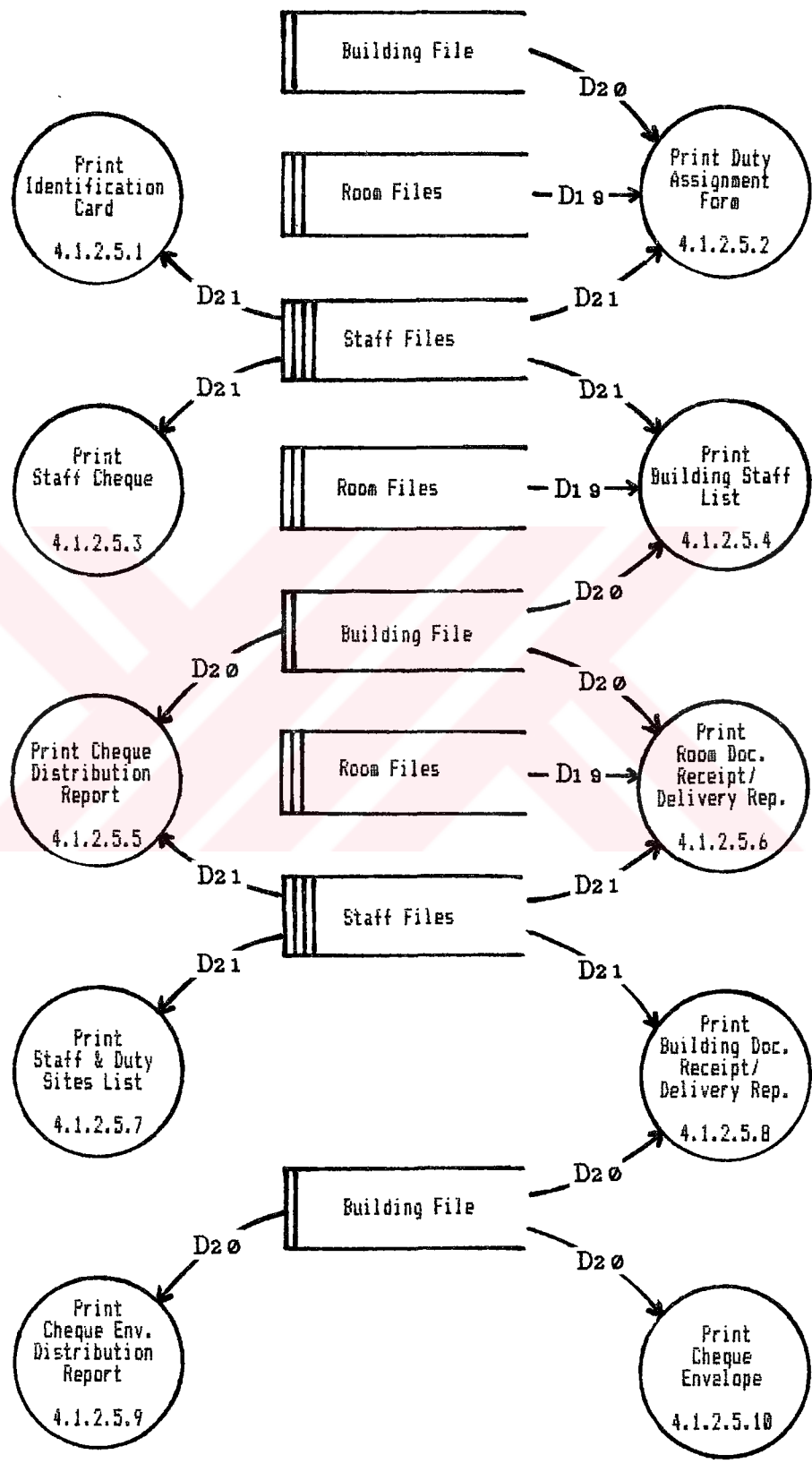


Figure 4.9 DFD of Printouts Concerning the Staff

-Building Document Receipt/Delivery Report:It is used when the Building Examination Director receives/delivers any examination document from the Province Exam Manager.

-Cheque Envelope Distribution Report:It is used during the distribution of Cheque Envelopes to the buildings.



## 5. PHYSICAL DESIGN OF REGISTRATION SYSTEM

The processes, which form the logical structure of the Registration System and their relations to the files, were shown in the previous section. In this section, the data files used by the Registration System will be introduced and then the programs, supporting software and system files which provides the operations on these files will be determined.

### 5.1. Data Files

There are six main data files used in the operation stages of the examination organization's Registration System. The description of these files and the features of their data fields are as follows.

#### 5.1.1. Candidate File

This is the file on which the candidate information is recorded. In this file, the information about candidates obtained from the Application Forms; the information related to the room assignment; the information obtained by the evaluation of results after the examination are recorded and stored.

Record layout of the Candidate File is given in Figure 5.1 by using Warnier-Orr. The description of the data fields is as follows.

Application Number (Type=Integer, Length=6)

It is the unique number given to the candidate. The first 5 digits of this number are used as the file access key and the sixth one is the check digit. It is calculated as follows.

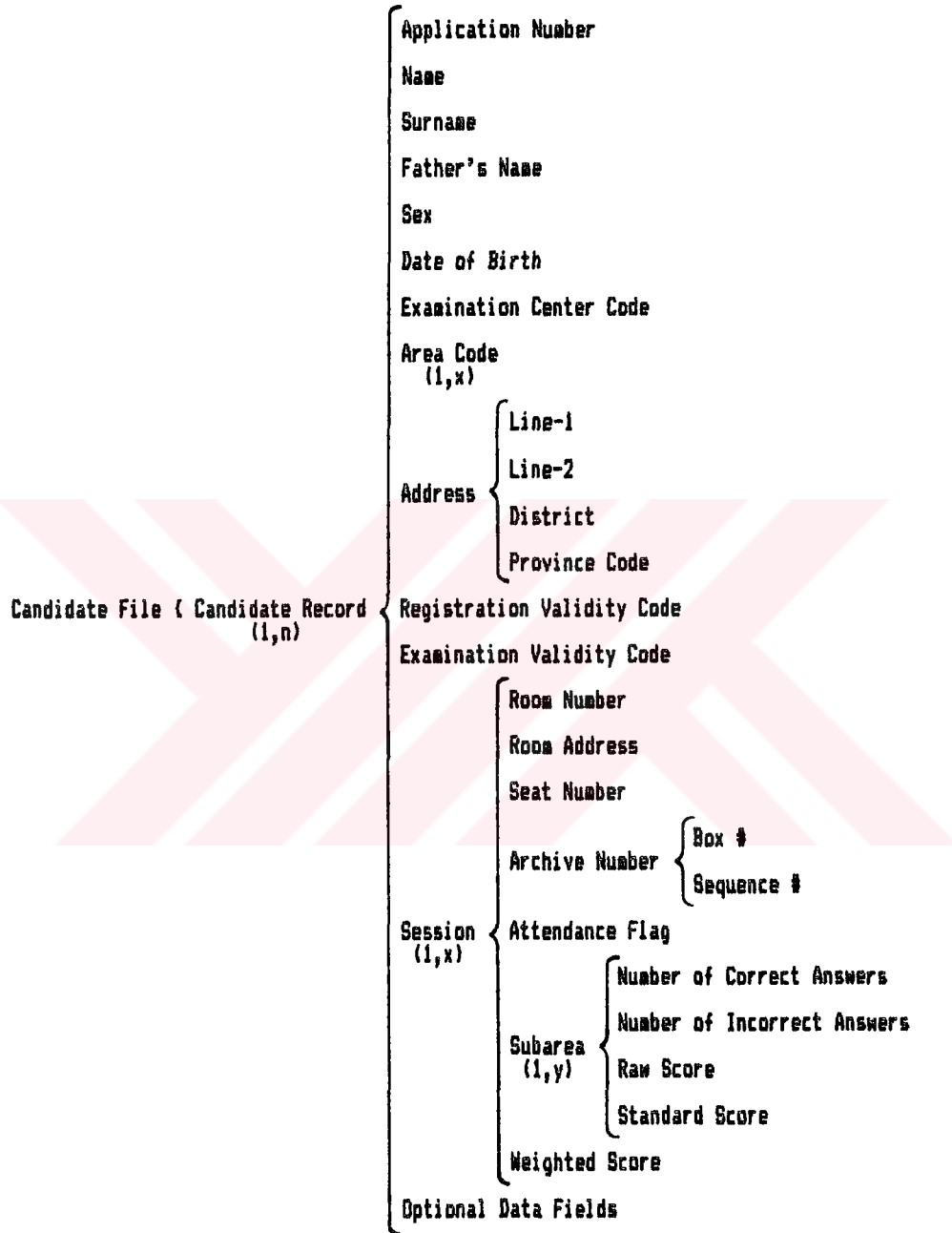


Figure 5.1 Warnier-Orr Diagram of Candidate File Layout

	A1	A2	A3	A4	A5	A6
	+-----+	+-----+	+-----+	+-----+	+-----+	+-----+
Application Number	!	!	!	!	!	!
	+-----+	+-----+	+-----+	+-----+	+-----+	+-----+

$$A6 = 9 - (A1*2+A2*3+A3*4+A4*5+A5*6) \text{ mod } 9$$

Name (Type=Alphabetic, Length=12)

Surname (Type=Alphabetic, Length=14)

Father's Name (Type=Alphabetic, Length=12)

Sex (Type=Integer, Length=1)

Date of Birth (Type=Integer, Length=6)

These are the identification data of the candidate.

Examination Center Code (Type=Integer, Length=2)

It defines the center that the candidate wishes to take the examination. For the examinations executed in the provinces, the province traffic code, for other centers the codes larger than 67 are used.

Area Code (Type=Integer, Length=2)

It defines the area that the candidate is going to take the examination in each session. For each session, the area codes start from 1 and continues up to the number of areas.

Address: Line-1 (Type=Alphabetic, Length=23)

Line-2 (Type=Alphabetic, Length=23)

District (Type=Alphabetic, Length=12)

Province Code (Type=Integer, Length=2)

Mailing Address of the candidate. For printing the address correctly on the document, this field is divided into 4 pieces. The first two of them include the information like street names and house numbers, third one includes district information and the last one includes the address province traffic code.

Registration Validity Code (Type=Integer, Length=1)

It indicates whether the application is valid or canceled because of the insufficient conditions.

Examination Validity Code (Type=Integer, Length=1)

It indicates whether the examination of the candidate is valid or canceled because of inconforming the examination rules.

Room Number (Type=Integer, Length=5)

It indicates the room in which the candidate shall take the examination. Room number is unique for each room.

Room Address (Type=Integer, Length=4)

It is the address of the room in the Room File of the related session.

Seat Number (Type=Integer, Length=3)

It indicates the seat number on which the candidate shall take the examination in the room. The seat number starts from 1 and continues up to the capacity of the room.

Archive Number: Box # (Type=Integer, Length=2)

Sequence # (Type=Integer, Length=3)

It is used by the Examination System to access the answer sheets.

Attendance Flag (Type=Integer, Length=1)

It indicates whether the candidate attended or not attended the examination.

Number of Correct Answers (Type=Integer, Length=3)

It is the number of correct answers given by the candidate for each subarea.



#### Number of Incorrect Answers

(Type=Integer, Length=3)

It is the number of incorrect answers given by the candidate for each subarea.

#### Raw Score (Type=Signed Real, Length=6.3)

It is obtained by subtracting the 1/4 of incorrect answers from the number of correct answers.

#### Standard Score (Type=Signed Real, Length=6.3)

It is obtained by the transformation of raw score to the scores having a mean of 50 and standard deviation 10.

#### Weighted Score (Type=Signed Real, Length=6.3)

It is obtained by multiplying the standard scores of each subarea with definite coefficients and their summation.

#### Optional Data Fields

In some examinations, it might be necessary to appoint some data fields for some reasons other than the predefined data fields. For this purpose, the user may define new data fields in the Candidate File. These fields are completely under the control of user and can be named in any form.

#### 5.1.2. Room Files

These are the files in which the information about the rooms of examination is stored. There is one Room File for each examination session. This file is used for specifying the number of staff required in the rooms, recording the staff addresses and preparation of the printouts related to the staff.

The record layout of the Room Files is shown in Figure 5.2 by using Warnier-Orr. The definition of the data fields is as follows.

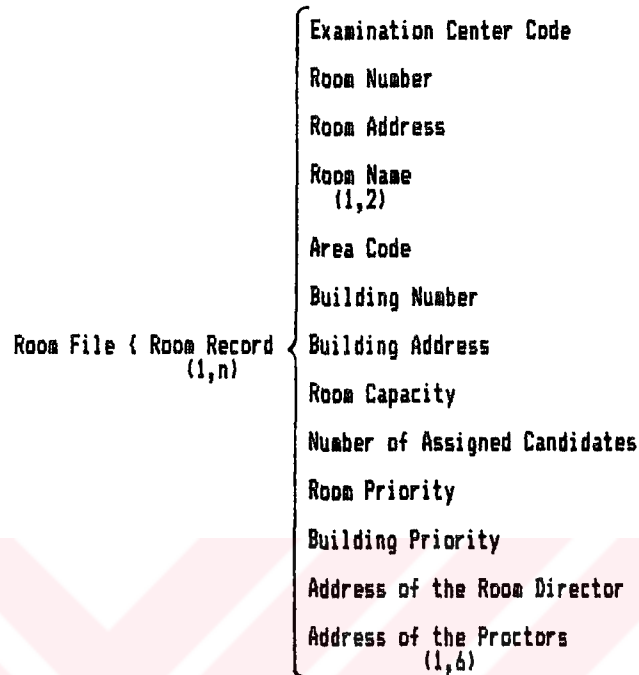


Figure 5.2 Warnier-Orr Diagram of Room File Layout

Examination Center Code (Type=Integer, Length=2)  
It defines the examination center of the room.

Room Number (Type=Integer, Length=5)  
It is the unique number which indicates the room.

Room Address (Type=Integer, Length=4)  
Used for access to the Room File.

Room Name (Type=Alphabetic, Length=2x30)  
Full name of the room. It is divided into two pieces to ease the printouts.

Area Code (Type=Integer, Length=2)  
It defines the area of the related session for which room will be used.

Building Number (Type=Integer, Length=5)

It defines the building that the room is located.

Building Address (Type=Integer, Length=4)

It is the address of the building in the Building File, that the room is located.

Room Capacity (Type=Integer, Length=3)

It defines the capacity of the room. It is used during the room assignment by the Examination System.

Number of Assigned Candidates

(Type=Integer, Length=3)

It is the number of candidates confirmed to take the examination.

Room Priority (Type=Integer, Length=1)

It defines the availability of the room for examination. It is used during the room assignment by the Examination System.

Building Priority (Type=Integer, Length=3)

It defines whether the building has higher priority than the the other buildings in an examination center or not. It is used during the room assignment by the Examination System.

Address of the Room Director

(Type=Integer, Length=4)

It is the address of the appointed Room Director in the staff file.

Address of the Proctors (Type=Integer, Length=4)

These are the addresses of the appointed Room Proctors in the Staff Files.

### 5.1.3. Building File

It is the file in which the information about the buildings to be used in the examination is stored. This file is used to determine the necessary number of staff, to record the staff addresses and to prepare the printouts related to the staff.

The record layout of the Building File is shown in Figure 5.3 by using Warnier-Orr. The description of the data fields is as follows.

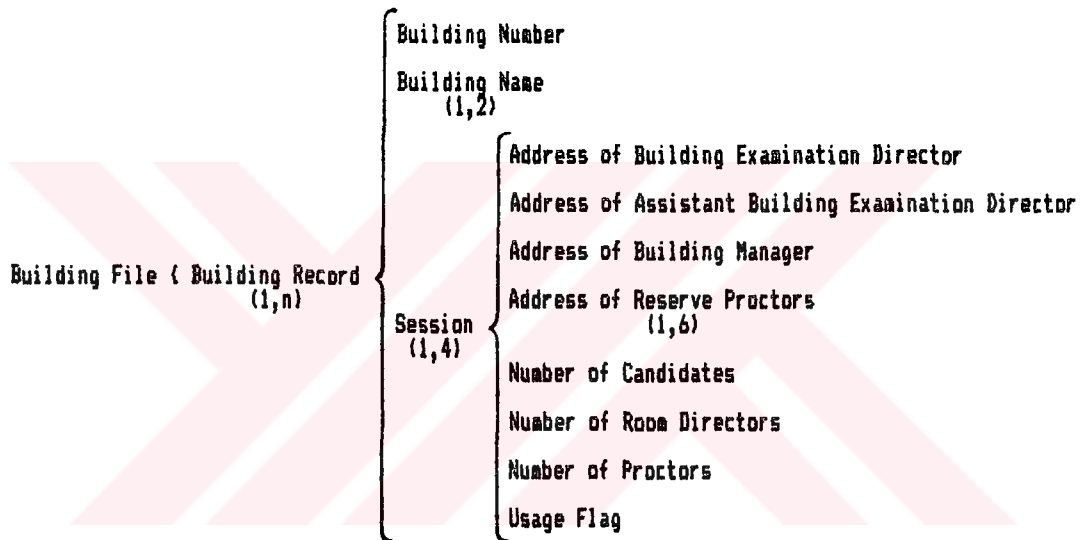


Figure 5.3 Warnier-Orr Diagram of Building File Layout

Building Number (Type=Integer, Length=5)

It identifies the examination center and the order of the building within this center.

Building Name (Type=Alphabetic, Length=2x30)

The name of the building in full. It is divided into 2 pieces to ease the printouts.

Address of Building Examination Director,  
Address of Assistant Building Examination Director,  
Address of Building Manager,  
Address of Reserve Proctors  
(Type=Integer, Length=4)

The addresses of the building staff appointed for each session of the examination in the Staff Files.

Number of Candidates (Type=Integer, Length=4)

It is the number of candidates taking the examination in the building for the related session.

Number of Room Directors (Type=Integer, Length=4)

It is the number of Room Directors which are on duty in the building of the related session.

Number of Proctors (Type=Integer, Length=4)

It is the number of Proctors which are on duty in the building of the related session.

Usage Flag (Type=Integer, Length=1)

It shows whether the building was used or not in the related session.

#### 5.1.4. Staff Files

These are the files on which the information about the Proctor, Reserve Proctor, Room Director, Building Examination Director, Assistant Building Examination Director and Building Manager responsible for each session is stored.

The record layout for the Staff Files is shown in Figure 5.4 by using Warnier-Orr. The description of the data fields is as follows.

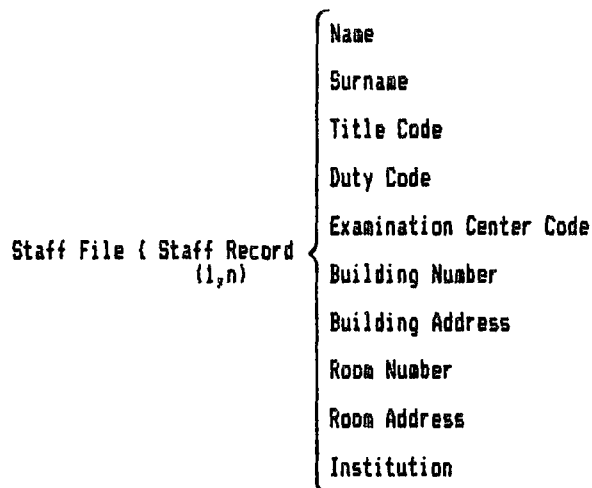


Figure 5.4 Warnier-Orr Diagram of Staff File Layout

Name (Type=Alphabetic, Length=12)

Surname (Type=Alphabetic, Length=14)

The name and surname of the staff.

Title code (Type=Integer, Length=2)

It defines the title of the staff. Title codes are shown in Table 5.1.

Duty Code (Type=Integer, Length=1)

It defines the task of the staff in the examination. Duty codes are shown in Table 5.2.

Examination Center Code (Type=Integer, Length=2)

It defines the center on which the staff is appointed.

Building Number (Type=Integer, Length=5)

It defines the building on which the staff is appointed.

Building Address (Type=Integer, Length=4)

It is the address of the building that the staff is appointed, in the Building File.

Table 5.1 Title Codes of the Staff

Title Code	Title
1	Professor Dr.
2	Professor
3	Associate Professor Dr.
4	Associate Professor
5	Assistant Professor Dr.
6	Assistant Professor
7	Instructor Dr.
8	Instructor
9	Research Assistant Dr.
10	Research Assistant
11	Expert Dr.
12	Expert
13	Teaching Assistant Dr.
14	Teaching Assistant
15	Translator
16	Director
17	Assistant Director
18	Inspector
19	Teacher
20	Officer
21	Others

Table 5.2 Duty Codes of the Staff

Duty Code	Duty
1	Building Examination Manager
2	Assistant Building Examination Manager
3	Building Director
4	Reserve Proctor
5	Room Director
6	Proctor

Room Number (Type=Integer, Length=5)

It defines the room that the Room Director and Proctors are appointed.

Room Address (Type=Integer, Length=4)

It is the address of the room in the Room File that the Room Director and Proctors are appointed.

Institution (Type=Alphabetic, Length=30)

It is the full name of the institution that the staff works for.

#### 5.1.5. Registration Optical File

These are the magnetic tape files on which the candidate information have been recorded by the optical reading of Application Forms. These files consist of the application number of the candidate, the identity information, the candidate's preference about the examination center, area codes and the optional data fields.

#### 5.1.6. Staff Information File

These are files obtained by transferring the staff information on the Staff Notification Schedules to the magnetic tapes by the Registration Supporting Subsystem. In these files there is the information of the staff name and surname, his title, his duty in the examination, the name of the organization he is working and the address to be used for recording these information on the Staff File.



## 5.2. General Purpose Programs

The processes which can be applied without any changes in every examination organization, can be realized by using the programs written only for this purpose. However, some processes are not same for every examination and they might require modifications. For example, some examinations are performed in one session, some are in four sessions. In such cases, the repetition of the data fields in the Candidate File must be different. It is not a good solution to define fields in this file for four sessions, because during the whole examination organization, the Candidate File needs to be accessed frequently. Instead of this, it is better to form the record layout of the Candidate File after the determination of the number of examination sessions.

In the Candidate File, some optional data fields can be reserved for placing the candidates to their preferences after the examination or for statistical data collection, in a manner not influencing the examination organization. The changes in the record layouts causes the rearrangement of the programs which provides the processes of the file's above mentioned fields. Furthermore, except for the predefined processes, new printouts and statistical tables, depending on the application's characteristics may be required, during an examination organization. In this case, there are three methods that can be followed.

- i) To write new programs or to modify the old ones
- ii) Interpreter programs
- iii) Program writing programs

To write new programs or to modify the old ones is the method followed in the existing system. During the preparation of examination organization, all required programs are written or the programs for the previous

applications are modified. This means to use the same efforts once more and it is one of the reasons of the requirement to build a general purpose Registration System.

Interpreter programs require a fine defined series of rules and data structures for all possible processes. The user is unable to use any other structure than the existing ones when faces a situation which was not forecasted.

Program writing programs make it possible for the user to add some commands of a high level language to some parts of the source program. So, the user can create programs more easily without being limited with the frames of the interpreter program. This is the method used in Registration System. There are three programs developed by the author using this method and they are called the general purpose programs. In general, these programs are used to execute various processes on the files. The general purpose programs and their functions are as follows.

1)File Load program:This program performs the creation or updating processes of a file.

2)Print Program:This program enables the printing of a file's certain data fields in the desired form.

3)Distribution Program:This program prepares statistical tables, showing repeating density of values in data fields of a file.

#### 5.2.1. Common characteristics

The common characteristics of the general purpose programs are as follows.

-They operate in interactive environment. The users fill the fields of the forms on the screen or they change the values of these fields.

-Their aim is to produce a source program in PL/I programming language. This source program may be compiled and run automatically, whenever the user wishes.

-These programs can be used by the main program of the Registration System and they can be used independently, as well.

-The information entered to the fields of the forms on the screen is stored in a parameter file.

-There are two uses of general purpose programs.

i) All fields of the forms on the screen are empty and they are filled by the user. This method is used in the cases when the general purpose programs are used for a situation which is not required before the realization of the examination.

ii) The form is displayed with the fields filled by the values using a parameter file which was created before. So the user has the possibility of making modifications on these fields. When the general purpose programs are used for executing the tasks to be done during the examination organization by the Registration System they operate in this manner. When the user make a choice among the options from the main menu of the Registration System which was defined to be done by using the general purpose programs, the concerned one is run automatically by means of the following commands.

```
RUN <general purpose program>; VALUE=1;  
FILE F=<user system name>/<program name>/PARAMS
```

Here <program name> is the source program to be generated by the general purpose program. When this program has been compiled and run, the requirement of the user is fulfilled.

-There are various screen forms which the general purpose programs use commonly. The definition and utilization purpose of them are stated below.

i)Program Name Form:When a general purpose program is run, this form appears firstly on the screen (Figure 5.5). The <program name> to be entered in the first field of the form, ensures the source program which will be created to be referred as <user system name>/<program name>. Besides, the information entered to the fields of the subsequent forms by the user are recorded in <user system name>/<program name>/PARAMS file. It is also possible to enter some comments, stating the function of the source program. These comments will be added to the source program for explanatory purpose.

ii)Job Starting Form:If the source program has been created before, this form will be displayed secondly (Figure 5.6). It ensures to modify the source program; to compile and execute it; or to run the previously generated object code. According to the selection, a file consisting of the job flow language statements is being formed. These statements ensure the source program to be compiled or run. By transferring the control to the WFL compiler, the calling program is kept in the waiting state.

iii)Layouts Form:The purpose of the form is to ensure the selection of the record layouts that will be required within the source program to be generated (Figure 5.7). These layouts are recorded in the files of the directory <user system name>/LAYOUT. The names of all these files are displayed on the screen assigning a code

REX - (PROGRAM ADI)

---

Yaratılacak program adı

---

)CREATE/CANDIDATE (

Açıklamalar

---

) BU PROGRAM ADAYLAR KUTUGUNU BOS OLARAK YARATIR. (<  
) (<  
) (<  
) (<  
) (<  
) (<

---

Figure 5.5 Program Name Form

REX - (iř BAřLATMA)

---

) ( Program üretme  
) ( Program derleme  
) ( Program çalıştırma  
) ( Derleme ve çalıştırma

Seęiminizi yaparak SPCFY tuřuna basınız.

---

Figure 5.6 Job Starting Form

REX - (GÖRÜNÜMLER)	
1 LAYOUT/CANDIDATE	2 LAYOUT/REGOPTIC
3 LAYOUT/ROOM	4 LAYOUT/BUILDING
5 LAYOUT/STAFF	6 LAYOUT/STAFFINF
7 LAYOUT/TITLES	8 LAYOUT/OUTIES
9 LAYOUT/AREACODES	10 LAYOUT/PROVINCES
11	12
13	14
15	16
17	18
19	20
21	22
23	24
25	26
27	28

Kullanmak istediğiniz görünüm numaralarını virgüllerle ayrılmış olarak giriniz

---

>1

Figure 5.7 Layouts Form

REX - (KOMUTLAR)	
- SIRA -	DEYİMLER
>000100(	)IF N(6) Ü= 9-MOD(N(1)*2+N(2)*3+N(3)*4+N(4)*5+N(5)*6,9)
>000200(	) THEN DO;
>000300(	) MESSAGE(1);
>000400(	) INVALIDKEY;
>000500(	) END; ELSE
>000600(	) KEY=KEY-1;
>000700(	)
>000800(	)
>000900(	)
>001000(	)
>001100(	)
>001200(	)
>001300(	)
>001400(	)
>001500(	)
>001600(	)
>999999(	)

Bitirmek için SIRA kolonuna 999999 giriniz.

Figure 5.8 Commands Form

number each of them. The user will enter the code numbers by separating with commas.

iv)Commands Form:This form enables the user to enter the PL/I statements into the source program to be created (Figure 5.8). These statements are requested in two places of the general structure of the source program. The first one is, the beginning of the program for those to be executed only once. The second one is the inside of the main loop, for those to be executed several times. The form is arranged as two columns. The first column is entitled SEQUENCE, and the second column STATEMENTS. The SEQUENCE column is displayed on the screen with values beginning from 100 and with the increment of 100. The PL/I language statements must be entered appropriate to the syntax into the STATEMENT column. The SEQUENCE column is used whenever a new statement is wished to be added between the previously entered statements. In this case the SEQUENCE number must be updated. Whenever a 999999 is entered to the SEQUENCE column the statements will be sorted according to their sequence numbers.

v)Output Definition Form:This form is used in various stages of the File Load and Print programs (Figure 5.9). The purpose of the program is to generate the printing commands, directed to the line printer of the source program to be created. These are entitled as LINE, COLUMN, DATA LIST and FORMAT. The LINE value ensures the vertical control of the printer. This value has to be filled whenever it is wished to proceed on a new line. The COLUMN value ensures the horizontal control of the printer. This column has to be filled if a skip on the same line is wished. The DATA LIST column ensures the entry of the variables and constants to be printed. It is not necessary to specify a FORMAT for each element of DATA LIST as long as the LINE and COLUMN values are left empty. However the FORMAT of the elements of the DATA LIST entered up that moment have to be specified in case





that the LINE and COLUMN fields are filled. This form repeats till a 99 is entered to the LINE column.

vi)File Attributes Form:This form, is used to determine the physical attributes of the files (Figure 5.10). If these files are present in the system during the running of the general purpose programs, their attributes are obtained by the program automatically, otherwise they will be entered to the first and second fields of the form by specifying the record size and block size in terms of byte. The following fields are reserved for the input of I/O Subsystem parameters except KIND, MAXRECSIZE, BLOCKSIZE, FRAMESIZE, MYUSE and TITLE.

### 5.2.2. File Load Program

By means of the File Load program, a source program ensuring the creation and updating of a file, is generated. This source program is used for five different functions.

TYPE 1-Creating by constant values:It has the purpose of creating a file in required number of records and physical properties by loading some predefined values.

TYPE 2-Updating by constant values:It has the purpose of modifying some of the data fields of a previously created file.

TYPE 3-Creating by using another file:It has the purpose of creating a new file with the same configuration of another file by using some of its data fields.

TYPE 4-Updating by sequential access from another file:It has the purpose of updating the data fields of a file which is read in the same sequence of another file.

TYPE 5-Updating by direct access from another file:It has the purpose of updating a file by accessing directly to another file by using one of the data fields as key.

Two optional properties may be present in the source program created by the File Load program.

i)Report property:A report may be issued for each record while the creation or updating of the file. This report has one or more lines and the type of information it will consist is determined by the user. The information transferred to the file may be controlled using this report.

ii)Warning property:It is possible to give warnings related to some conditions specified by the user. These warnings may be used to examine the validity of the information transferred to the loaded file.

The running of the file load program and the order in which the forms are displayed to the screen, the function of the forms and the meaning of the fields on them are shown below.

1)By means of the "Program Name" form the program to be created will be entitled.

2)The compilation or executing of the previously generated source program can be done by using "Job Starting" form. Otherwise the following steps will be executed.

3)By means of the "Layouts" form the record layouts to be used in the source program, will be specified.

4)File Load\_Main Menu:By means of this form, which one of the 5 basic functions of the File Load program will be used is determined (Figure 5.11). The user makes his selection by pressing the SPCFY key.

5)File Load\_Specifications form:By means of this form, the files which will be used in the source program, are specified (Figure 5.12). In the first and second fields of the form, the name of the file which will be created or updated and the record variable of this file is entered. The third field of the form is used for TYPE=1 and determines the number of records to be created. The fourth field of the form is used for TYPE=5 and determines the name of the key variable which will ensure the direct access. The fifth and sixth fields of the form determines the name of the record variable and the name of the file for TYPE=2. The seventh and eighth fields of the form determines whether the optional properties are present or not.

6)If TYPE=1 or TYPE=3 the physical attributes of the file to be created is determined by the "File Attributes" form. Among the other type of usage of the File Load program, in case that the file to be updated is not present in the system, the form is utilized directed to this file.

7)If TYPE>2 and the loading file is not present in the system, the "File Attributes" form is displayed once more and the loading file properties are determined.

8)If the warnings property is desired to be used, the message that will be presented in case of an error, will be determined by means of the form entitled "File Loading\_Warnings" (Figure 5.13). The realization of this message will be ensured by means of the special command MESSAGE(n) within the "Commands" form.

REX - KÜTÜK YÜKLEME (ANA MENÜ)

---

	<u>Yaratma</u>	<u>Günleme</u>
Değişmez değerler yükleyerek . . . . .	}	( . . . ) (
Başka bir kütükten sıradan erişimle . . . . .	}	( . . . ) (
Başka bir kütükten doğrudan erişimle . . . . .	}	(

Seçiminizi yaparak SPCFY tuşuna basınız

---

Figure 5.11 File Load\_Main Menu

REX - KÜTÜK YÜKLEME (PARAMETRELER)

---

Yüklenecek kütük adı	}	_____	(
Tutanak adı	}	_____	(
Yaratılacak tutanak sayısı	}	_____	(
Doğrudan erişim anahtarı	}	_____	(

---

Yükleyen kütük adı	}	_____	(
Tutanak adı	}	_____	(

---

Rapor üretme	}	(	(1:Evet, 2:Hayır)
Hataları Listeleme	}	(	(1:Evet, 2:Hayır)

---

Figure 5.12 File Load\_Specifications Form

REX - KÜTÜK YÜKLEME (UYARILAR)	
UYARI İLETİŞİ	UYARI İLETİŞİ
01 )*** BASVURU NO GECERSİZ ***(<	02 )*** ÖNCE DEN KAYITLI ADAY ***(<
03 )*** KAYITLI OLMAYAN ADAY ***(<	04 )ADI : KODLANMAMIS (<
05 )ADI : ÇİFT İSARET (<	06 )SOYADI : KODLANMAMIS (<
07 )SOYADI : ÇİFT İSARET (<	08 )BABA ADI : KODLANMAMIS (<
09 )BABA ADI : ÇİFT İSARET (<	10 )CİNSİYET : GECERSİZ (<
11 )D.TARİHİ : KODLANMAMIS (<	12 )D.TARİHİ : ÇİFT İSARET (<
13 )SINAV M. : GECERSİZ (<	14 )ALAN KODU: GECERSİZ (1.OTURUM)<
15 )ALAN KODU: GECERSİZ (2.OTURUM)<	16 )ALAN KODU: GECERSİZ (3.OTURUM)<
17 )ALAN KODU: GECERSİZ (4.OTURUM)<	18 )ADRES : KODLANMAMIS (<
19 )ADRES : ÇİFT İSARET (<	20 )ŞEHT/İLCE: ÇİFT İSARET (<
21 )İL KODU : GECERSİZ (<	22 ) (<
23 ) (<	24 ) (<
25 ) (<	26 ) (<
27 ) (<	28 ) (<
29 ) (<	30 ) (<
31 ) (<	32 ) (<
33 ) (<	34 ) (<

Figure 5.13 File Load\_Warnings Form

9) If the report property to desired to be used the title that will be printed on the top of the report's page, will be determined using the "Output Definition" form.

10) With the "Commands" form, the PL/I commands to be executed prior the main loop of the source program, are entered.

11) If TYPE=5, in other words if an updating action has to be taken by means of direct access from a file, then the PL/I Commands which will ensure the definition of the access key will be entered using the "Commands" form. The updating process on the loaded file may be canceled by means of a special command named INVALIDKEY.

12)Using once again the "Commands" form it is possible, this time, to enter the PL/I commands to be executed in the main loop of the source program. These commands shall be of assignment type which will load values to the data fields of the loaded file. In order to cancel writing to the loaded file it is possible to use the special command SKIPTONEXT. Besides it is also possible to give a message, whenever the special command MESSAGE(n) is used and if the conditions determined by the user take place. These two special commands might be used with the structure as

```
IF <cond 1> THEN DO;
    SKIPTONEXT; MESSAGE(1);
END; ELSE
    IF <cond 2> THEN DO;
        <commands>
        MESSAGE(2);
    END;
```

13)If the report property is used, the definition of the lines to be printed will be defined by the "Output Definition" form.

### 5.2.3. Print Program

The Print program generates a source program which ensures the printing of some data fields of a file, in a desired format. The printout will consist the following properties.

-The page length is determined by the user.

-The lines to be printed for header and footer of each page may be determined.

-The printing may be realized with channel skipping method. In that case the first character of each line is used for the channel control.

-Contiguous document printing: This property ensures to print on the documents being contiguously arranged because of the narrow width of the continuous forms.

-Grouped printing: It ensures to proceed on a new page, when the value of the control variable changes.

The running of the Print program and the order in which the forms are displayed to the screen, the function of these forms and the meanings of the fields on them are stated below.

1) By means of the "Program Name" form the program to be created will be entitled.

2) The compilation or executing of the previously generated source program can be done by using "Job Starting" form. Otherwise the following steps will be executed.

3) By means of the "Layouts" form the record layouts to be used in the source program will be specified.

4) Print\_Main Menu: By means of this form, the properties of the printout and the file to be used in the printing are determined (Figure 5.14). In the first two fields of the form, the name of the file and the record variable of the file are entered. The third field of the form specifies whether the channel skipping method will be used or not. In the fourth field of the form, the number of contiguous documents to be printed are entered. The fifth field of the file shows the name of the control variable if a grouped printing type is used, the sixth field of the file shows the format of this variable.

```

REX - DÖKÜM (ANA MENÜ)
-----
Dökülecek kütük adı   } _____ (
Tutanak adı          } _____ (
-----
Satır/sayfa denetimi } (  1: Satır atlayarak
                       } (  2: Kanal atlayarak
Yanyana belge sayısı  } 1(
-----
                (Gruplayarak döküm yapılacaksa)
Kontrol değişkeni adı } _____ (
Format              } _____ (
-----

```

Figure 5.14 Print\_Main Menu

```

REX - DÖKÜM (SAYFA TANIMI)
-----
Sayfaya yazılacak toplam satır sayısı } (
Sayfa başı için ayrılan satır sayısı  } (
Sayfa sonu için ayrılan satır sayısı  } (
Her tutanak için ayrılan satır sayısı } (
-----

```

Figure 5.15 Print\_Page Definition Form



5)Print\_Page Definition form:By means of this form the layout of the printout is determined (Figure 5.15). The fields on the form determines respectively the total number of lines to be printed on a page, the number of lines allocated for the header and footer of the page and the number of line to be printed for each record of the file to be printed.

6)If the file to be printed is not present in the system, by means of a "File Attributes" form the physical properties of the file are determined.

7)With the "Commands" form, the PL/I commands to be executed prior the main loop of the source program, are entered.

8)The lines to be printed as footer of the page are determined by means of the "Output Definition" form.

9)The lines to be printed as header of the page are determined by means of the "Output Definition" form. In the DATA LIST column of this form, the special command REXHEADER may be entered. This command, provides the printing of the user system name, hour, date and page number on each page of the printout.

10)With the "Commands" form, the PL/I commands, to be executed in the main loop of the source program, are entered. This commands are for the preparation of some of the information that will appear on the printouts. It is possible to make use of the special command SKIPTONEXT in order to skip the records which will not be included in the printout.

11)Using once again the "Output Definition" form it is possible to determine which information will be included in the printout for each record of the file.

#### 5.2.4. Distribution Program

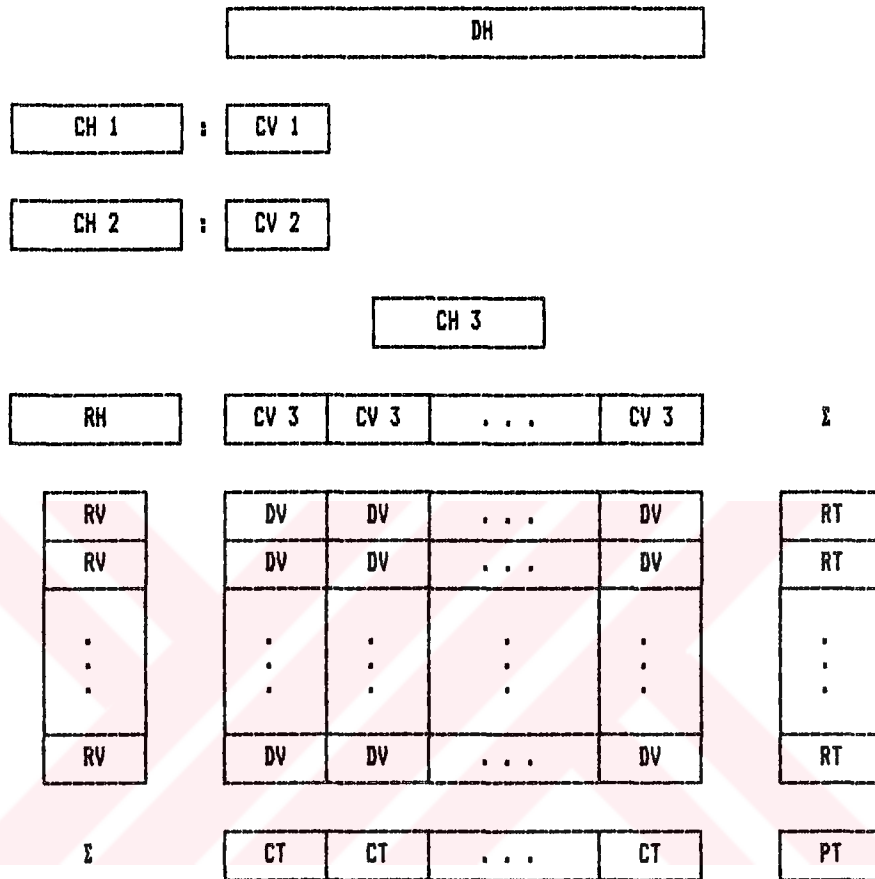
The Distribution program generates a source program which enables to form a table showing the repeating density of values in some data fields of a file. These tables are arranged as a two dimensional matrix depending on one row and one column parameter. Each element of the matrix shows the number of records having these row and column values in the related data fields. Row level is limited by 1 but the column level can be increased up to three. In this case, the matrix will be arranged as four dimensional and will be dependent on the four data fields of the file records. The table is arranged according to the upper and lower limits of the row and column variables. Only one of the row or column variable can be subscripted. In this case, the value of each element takes place in this distribution. When the subscripted variable is used, one dimension of the distribution can be formed by its index variable.

The layout of a page for the distribution with column level three is shown in Figure 5.16.

The running of the distribution program and the order in which the forms are displayed to the screen, the function of these forms and the meanings of the fields on them are stated below.

1)By means of the "Program Name" form the program to be created will be entitled.

2)The compilation or executing of the previously generated source program can be done by using "Job Starting" form. Otherwise the following steps will be executed.



- DH : Distribution Heading
- CH : Column Heading
- RH : Row Heading
- CV : Column Values
- RV : Row Values
- DV : Distribution Values
- CT : Column Total
- RT : Row Total
- PT : Page Total
- 1,2,3 : Column Levels

Figure 5.16 Sample Layout of the Distribution

3)By means of the "Layouts" form the record layouts to be used in the source program, will be specified.

4)Distribution\_Main Menu:By means of this form, the properties of the file and distribution are determined (Figure 5.17). In the first field of the form, the name of the file, and in the second field of the form the name of the file's record variable are entered. In the third field of the form, the heading of the distribution is entered. This title will be printed on each page of the distribution. The fourth field of the form determines the column level.

5)Distribution\_Line/Column Definition form:This form is used to determine the line/column characteristics of the distribution (Figure 5.18). It provides the definition of the distribution row, when it comes to the screen for the first time. Then it is repeated as much as the column level for definition of the columns. In the first field of the form, the names of the row/column variables (distribution variables) are entered. If one dimension of the distribution consists of a subscripted variable and if the word INDEX is entered to this field, then this dimension of the distribution is assigned for the subscript. The second field of the form identifies whether the distribution variable is subscripted or not. The third field of the form identifies the upper and lower limit values of the distribution variables, so a matrix dependent on these values is formed. The row and column titles of the distribution is entered to the fourth field of the form. The last field determines whether the numeric values between the upper and lower limits of the distribution variables or the full names obtained from an array accessed with these values, will be printed under these titles. If the name of an array is entered to this field, then the values of the row/column variables are obtained here.

REX - DAĞILIM (ANA MENÜ)

---

İşlenecek kütük adı } \_\_\_\_\_ {

Tutanak adı } \_\_\_\_\_ {

---

Dağılım başlığı } \_\_\_\_\_ {

Kolon düzeyi }1{ (1-3)

---

Figure 5.17 Distribution\_Main Menu

REX - DAĞILIM (SATIR/KOLON TANIMI)

---

Satır/Kolon değişkeni } \_\_\_\_\_ {

Değişken türü }1{ (1:Basit, 2:Dizimli)

Alt sınırı } \_\_\_\_\_ { Üst sınırı } \_\_\_\_\_ {

---

Satır/Kolon başlığı } \_\_\_\_\_ {

Satır/Kolon değerleri dizisi } \_\_\_\_\_ {

---

Figure 5.18 Distribution\_Line/Column Definition Form

6)With the "Commands" form, the PL/I commands to be executed prior the main loop of the source program, are entered.

7)By the "Commands" form, the PL/I commands to be executed in the main loop of the source program are entered. These commands are for grouping the values of distribution variables and making some transformations. In this form, it is possible to use the special command SKIPTONEXT in order to skip the records which shall not be included in the distribution.

### 5.3. Parameter File

This file is used to record, the detailed information about the user system, the number of candidates and the number of active stations. The name of the Parameter File is

<user system name>/PARAMS

When a user starts to use REX, this file will be copied as

<station number>/PARAMS

If the user system is formed for the first time, no parameter file is present and it has to be created at that moment. If this file is present in the system, this implies that a transaction has been made previously. This file consists of five records differing in appearance from each other. The structure of the Parameter File is given in Figure 5.19 by using Warnier-Orr. The data fields in these records are given below.

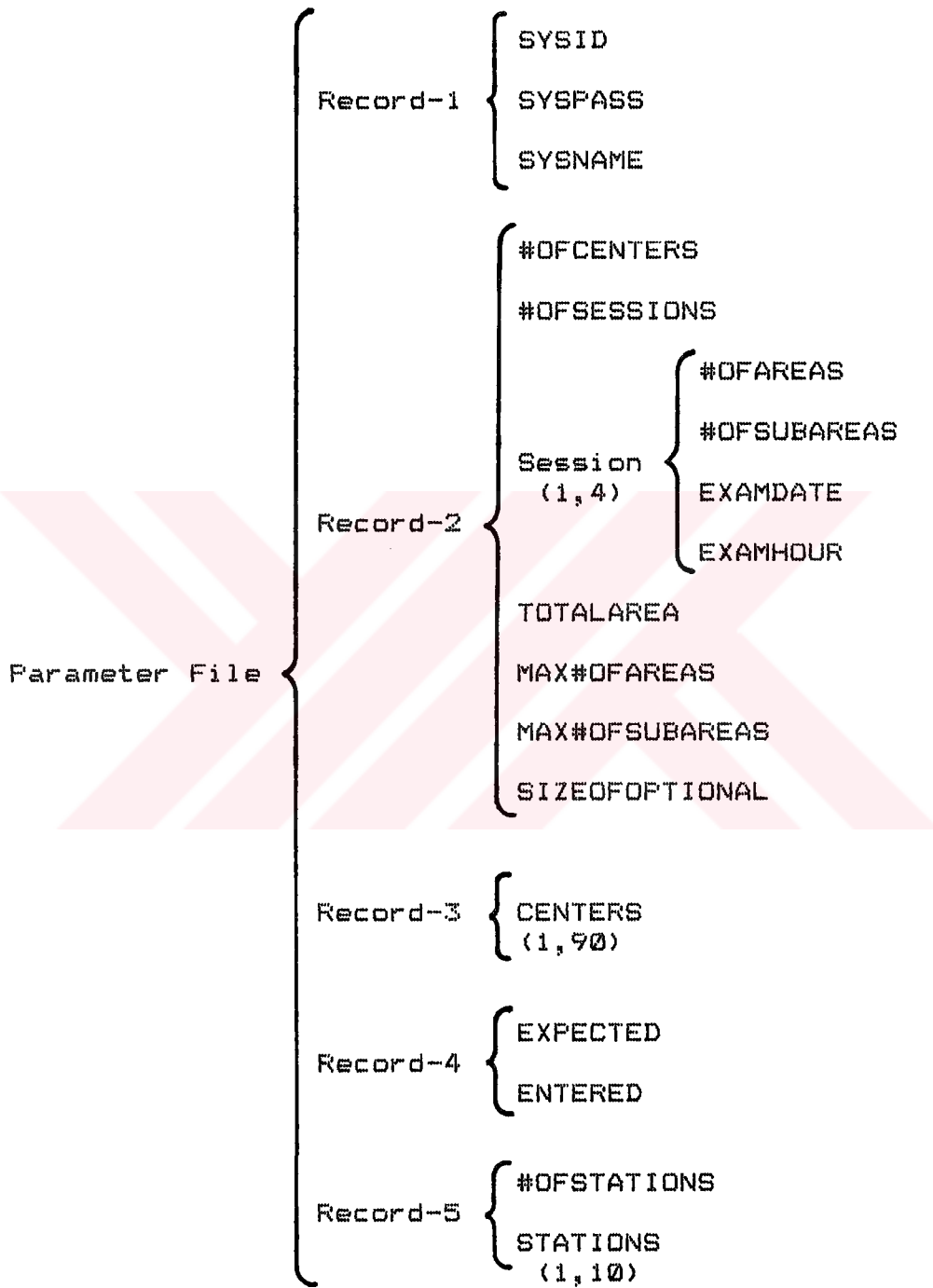


Figure 5.19 Warnier-Orr Diagram of Parameter File

Record 1

SYSID: This is a word specifying the user system which has a maximum length of five character. The Parameter File name will be <user system name>/PARAMS".

SYSPASS: This is the password necessary to access the User System structures.

SYSNAME: Name given to the examination.

Record 2

#OFCENTERS: It determines the number of centers in which the examination will be performed.

#OFSESSIONS: It determines the number of sessions in which the examination will be performed.

#OFAREAS: It determines the number of areas in each session.

#OFSUBAREAS: It determines the number of the subareas present in each area of a session.

EXAMDATE: It determines the date of each session of the examination.

EXAMHOUR: It determines the hour of each session of the examination.

TOTALAREA: It determines the total of the areas of all sessions.

MAX#OFAREAS: It determines the maximum number of the areas through the sessions.



MAX#OFSUBAREAS:It determines the maximum number of the subareas through the sessions.

SIZEOFOPTIONAL:It determines the size of the optional fields of the Candidate File in bytes.

#### Record 3

CENTERS:They are the codes given to the centers where the examination shall be executed. If the center is a province, then provincial traffic codes are used. For the centers other than provinces, the numbers larger than 67 are used.

#### Record 4

EXPECTED:The number of candidates expected to make registration.

ENTERED:The number of candidates whose admittance for the examination is ensured.

#### Record 5

#OFSTATIONS:The number of jobs on the same User System at a certain time.

STATIONS:The numbers of terminals which are used in the same user system at a certain time.

### 5.4. Layout Files

These files include the definitions of the data files record layouts and the full name of the coded information used by REX. The data files have been mentioned in Section 5.1. The layout files belonging to them, are in the form of

<user system name>/LAYOUT/<data file name>

In the layouts, the record variable of the file has been named as <data file name>\_REC.

The table definitions are stored in the files named

<user system name>/LAYOUT/<table name>.

These tables are used in order to obtain the examination center, the areas on which the examination is executed, the position of the staff and the full name of their duties in the examination. In implementation, new tables may be defined according to the characteristic of the examination, if it is required.

#### 5.5. Screen Files

When REX is running, all the forms displayed to the screen are stored in the files named

REX/SCREENS/x (x, is an integer of a minimum length of 1, and a maximum length of 3)

The forms in these files are loaded as initial values to the variables named SCREENx like strings of 1923 length, and they are displayed to the screen from here. The structure in which the fields will be read is named STRUCTx. REX includes several forms that are used in various stages and for different purposes. Their name and the screen file numbers are shown in Table 5.3.

There is another file named REX/SCREENS/DECLARE in the same directory of the screen files. This file includes the file definition and control codes required for the communication between the screen and the programs.

Table 5.3 Screen Files of REX

Screen	Screen Name
Ø	REX_Main Menu
1	REX_Existing User Systems Form
2	REX_User System Parameters Form
4	REX_Subsystems Form
1Ø	File Load_Main Menu
11	File Load_Specifications Form
12	File Load_Warnings Form
2Ø	Print_Main Menu
21	Print_Page Definition Form
3Ø	Distribution_Main Menu
31	Distribution_Line/Column Definition Form
5Ø	Program Name Form
51	Layouts Form
52	File Attributes Form
53	Commands Form
54	Output Definition Form
55	Job Starting Form
1ØØ	Registration_Main Menu
1Ø1	Candidate Printouts Form
1Ø2	Staff Printouts Form
1Ø3	Candidate Information Form
1Ø4	Examination Information Form
1Ø5	Optional Fields Form
1Ø6	Sort Room Files Form
> 199	REX Examination Subsystem Forms

## 5.6. Libraries

The libraries permit to write procedures compiled independently from the calling program. A library may consist of more than one procedure. Whatever the programming language they have been written, these procedures may be called with appropriate parameters by means of programs written in a desired language. Since all the programs of REX have been written in PL/I, the functions not implemented in this language have been realized with procedures written in ALGOL. These procedures are stored in files named

REX/LIBRARY/<library name>

The libraries in REX are as follows.

i)SUBSYSTEM:It enables to activate one of the subsystems (Registration, Examination, Supporting) from the main program of REX, by means of an integer parameter.

ii)STARTJOB:It enables to execute the work flow language commands in a program. It takes the file name as parameter in which these commands are stored.

iii)FILEMANAGER:It includes four procedures explained below.

a)COPY:It has two parameters. It ensures, the file defined by the first parameter to be copied by the name in the second parameter.

b)REMOVE:It ensures to remove a file. It takes the name of the file to be removed as parameter.

c)CHANGE:It has two parameters. It ensures to change the name of the file defined by the first parameter to the name of the file in the second parameter.

d)SEARCH:It has two parameters. The first parameter is a directory name. All the files in this directory are found and recorded to the file determined by the second parameter.

### 5.7. Synchronization Files

In REX, a program may initiate another one. Under this condition, the calling program shall be taken into waiting state till the completion of the execution of other program. This process requires a synchronization. In order to ensure the synchronization in REX a special structure has been used. The called program creates a file at the end of its job. Meanwhile the calling program waits till the creation of this file. These files are named as synchronization files. The general name of the synchronization files are in the form of

<station number>/WAITn

The presence of the <station number> prohibit to effect the users when REX is running with multi users. WAITn determines the program level called within a program. When a program initiate another one, it waits the file named WAIT1. Furthermore if the called program initiate another one too, then it waits the synchronization file WAIT2. Similarly it is possible to go much more deeper levels. When the execution of the called program is completed and so, returned to the calling program, the synchronization files are removed automatically.

### 5.8. Job files

These files are used to enable the procedure named STARTJOB within the library REX/LIBRARY/STARTJOB, to start a new job in the system. This process is used to run a program from another one or to compile automatically a source program created by a general purpose program. The job files are created by the program that will start the job and they are transferred as parameters to the STARTJOB program. It is automatically removed when the started job is terminated.



## 6. IMPLEMENTATION OF REGISTRATION SYSTEM

The functions of the REX Registration Subsystem stated in Section 4.2. are executed by the program named REX/REGISTRATION. This program is started by a procedure named SUBSYSTEM of the REX/LIBRARY/SUBSYSTEM library, after that the registration subsystem has been selected from the menu named "Subsystems" of REX/SYS. Meanwhile REX/SYS passes to waiting state, using the synchronization file named <station number>/WAIT1. Some of the functions in REX/REGISTRATION are executed by internal procedures and the other ones by external programs. When such programs are run, REX/REGISTRATION passed to waiting state using the synchronization file named <station number>/WAIT2. The user has made his choice among a menu where all the functions in the scope of the registration are being listed by pressing the SPCFY key. This menu is shown in Figure 6.1. When a choice is made from the menu, the procedure that will perform the concerned function will be activated. In this stage the functions are being executed either in this procedure or by an independent program started by this procedure. The functions in the main menu of the REX Registration Subsystem, the procedures executed and their tasks are given below.

### 6.1. Preparation of the Candidate File Layout

This function is implemented in the CREATE\_CANDIDATES\_LAYOUT procedure. The layout of the Candidate File has been shown in Section 5.1.1. by using Warnier-Orr. Though much of the data fields in this layout are same for each examination organization, some fields are still related to some variables. The layout is obtained by determining these fields according to the #OFSESSIONS and MAX#OFSUBAREAS of the Parameter File. The layout of the Candidate File is created with the name of <user system name>/LAYOUT/CANDIDATES. If there are some optional type data fields in the Candidate File these can

REX - KAYIT (ANA MENÜ)

- > < 1. ADAYLAR Kütüğü Görünümünün Hazırlanması
- > < 2. Boş ADAYLAR Kütüğünün Yaratılması
- > < 3. Başvuru Belgesi Dökümü
- > < 4. Aday Kayıt/Günleme işlemleri
- > < 5. Çift Başvuruların Belirlenmesi
- > < 6. Eksik/Hatalı Bilgilerin Belirlenmesi
- > < 7. Etkileşimli Sorgu/Günleme
- > < 8. Oturum/Merkez/Alan Dağılımının Alınması
- > < 9. Adaylara Yönelik Dökümlerin Yapılması
  
- > < 10. SALON Kütüklerinin Sıralanması
- > < 11. GÖREVLİ Kütüklerinin Yaratılması
- > < 12. Görevli Bildirim çizelgesi Dökümü
- > < 13. Görevli Bilgilerinin Kütüklere İşlenmesi
- > < 14. Görevlilere Yönelik Dökümlerin Yapılması

Seçiminizi yaparak SPCFY tuşuna basınız.

GERİ DÖNÜŞ ) <

Figure 6.1 Registration\_Main Menu

REX - (SEÇENEKLi ALANLAR)

Seçenekli alanların toplam boyu ) 11< (Byte)

DÜZEY	DEĞİŞKEN ADI	FORMAT
>2< ) OKULKODU	( )	( )
>3< ) IL	( )	>P'99'
>3< ) TUR	( )	>P'99'
>3< ) SIRA	( )	>P'99'
>2< ) ANKET(5)	( )	>CHAR(1)
>2< )	( )	( )
>2< )	( )	( )
>2< )	( )	( )
>2< )	( )	( )
>2< )	( )	( )
>2< )	( )	( )
>2< )	( )	( )
>2< )	( )	( )
>2< )	( )	( )

Bitirmek için DÜZEY alanına \* giriniz.

Figure 6.2 Optional Fields Form



be defined by using the form titled "Optional Fields" (Figure 6.2). The first field of the form specifies the size of the optional fields. This value is stored in the SIZEOFOPTIONAL variable of the Parameter File. The following lines of the form are used to enter the level, name and format of the optional fields. These definitions are automatically added to the end of the layout.

### 6.2. Creation of the Candidate File

This process is implemented by the procedure named CREATE\_CANDIDATE\_FILE by running general purpose File Load program. The File Load program is mentioned in Section 5.2.2. This program is run so that TYPE=1, in order to create the Candidate File. Number of records in the Candidate File is obtained by the EXPECTED variable of the Parameter File. The commands, prior the main loop of the source program to be created are in the form of initial values to be loaded to the data fields within the records of the Candidate File. The commands repeated in the loop enables to calculate the check digit of the application number. If some optional type data fields have been specified in the layout of the Candidate File, then the user may add some commands that will ensure to load some initial values to these fields. Using the File Load program a source program named CREATE\_CANDIDATES is obtained.

### 6.3. Printing the Application Forms

This function is implemented by the procedure named PRINT\_APPLICATION\_FORMS by running a general purpose Print program. The Print program has been mentioned in Section 5.2.3. By means of this program a source program named PRINT\_APPLICATION is being created. This source program ensures to print the application numbers to each of the three parts of the Application Form. The application numbers are obtained from the Candidate File. Since the Application Forms are not standard for each

examination organization, the columns in which the application number will be printed are organized by the user.

#### 6.4. Updating the Candidate File

The transfer of the candidate information to the Candidate File correctly and completely, is the most important function of the Registration System. The information of a candidate are transferred to the file for two different purposes.

i) Loading the information for the first time: No information is present in the Candidate File about this candidate. The Application Form of the candidate is read for the first time by the Optical Reader and the candidate will be recorded.

ii) Updating of the information: The candidate has been previously registered. But since some of the information are found to be incorrect, the Application Form will be read once again and these information will be corrected.

The basic steps of the registration process are given below.

1. Get the function code. (1: New registration, 2: Updating)
2. Read a record from the Registration Optical File.
3. Check the validity of the application number.
4. If the application number is correct, access to the Candidate File, if not give a warning as "incorrect number"; Go to the 2. step.
5. If the function code=1 and registration validity=1, give a warning as "previously recorded"; Go to the 2. step.

6. If the function code=2 and registration validity=Ø, give a warning as "not recorded"; Go to the 2. step.
7. Load the value 1 to the registration validity code.
8. Make the information checks.
9. Transfer the correct information to the Candidate File.
10. Print the candidate information to the Control List.
11. Give warnings for the incorrect information.
12. Go to the 2. step.

These functions are implemented by a procedure named UPDATE\_CANDIDATES\_FILE in order to run a general purpose File Load program. This program is run for TYPE=5, by means of to update a file by accessing directly from another file.

Here the updating file is the Registration Optical File and the updated is the Candidate File. The access from the Registration Optical File to the Candidate File is performed by the application number. The source program to be created shall be able to print the Control List for verifying the validity of each information incoming from the Registration Optical File, and to list the errors occurred. In this program firstly the validity of the application number incoming from the Registration Optical File is being verified. If the application number is correct then by accessing to the Candidate File the following two conditions are examined.

i) If the candidate is registered for the first time, but there is another candidate in the file with the same candidate number, this means either that the application number has been used by two different candidates or the Application Form has been read twice on the optical reader.

ii) If the candidate information has to be updated and meanwhile the candidate not recorded to the file, this means that either the Application Form has not been read or the candidate has not been previously recorded because that the application number is found to be incorrect.

In each case a detailed research must be done about the candidate. For that reason the candidate is not registered to the file. If one of this condition is not present, the information of the candidate can be transferred to the Candidate File. Therefore some controls will be done, in order to check whether any coding error has been done while the candidate was filling his Application Form. By checking them, the blank fields or the double marks are determined. Besides, the code of the examination center and the area are controlled by means of CENTERS array and #OFAREAS for each session in the Parameter File. For the information found to be invalid, a warning message appears on the Control List.

In order to implement all these functions a source program named UPDATE\_CANDIDATES is created. If some optional data fields are defined in the Candidate File, then the user will be able to determine the type of the controls, the warnings to be informed and the configuration of the control lists, by means of the organization that they will perform on the forms displayed to the screen.

#### **6.5. Determination of the Duplicate Applications**

This process is implemented by the procedure named DUPLICATE\_APPLICATIONS. The Candidate File is sorted according to the surname and the names of the candidates, by taking into account Turkish Characters. Therefore ones who have the same surname and name are listed using an intermediate file and then this file is removed.

## 6.6. Printing the Incomplete/Invalid Information

This function is implemented by the procedure named CHECK\_CANDIDATES\_DATA in order to prevent the presence of incomplete or invalid information of the Candidate File that may effect the application of the examination. For this purpose the Candidate File is read sequentially and the following checks are done and the inconsistent ones are reported.

i)The code of the examination center must be one of the #OFCENTERS element in the CENTERS array of the Parameter File.

ii)The area code for each session of the examination must be in the range of 1 and #OFAREAS value of the Parameter File. If the candidate takes the examination in the concerning session of the examination, the area code must be between 1 and this value.

## 6.7. Interactive Inquiry/Updating

This function is implemented in the procedure named INTER\_INQUIRY\_UPDATE. Two forms shown in Figure 6.3 and Figure 6.4 consisting of the data fields in the Candidate File is displayed to the screen. After that the user enters the application number of the candidate that he wants to inquiry/update on the right top field of the first form, all the requested information concerning the candidate is displayed to the screen. If the user wants only to inquiry the information concerning the candidate then by entering a new application number he/she may continue to the process. If the user wants to update one or more than one information, then he writes the modification down and enters the update code to the bottom line of the form. So the new information on the screen is transferred to the Candidate File.

REX - KAYIT (ADAY BİLGİLERİ)	
	Başvuru Numarası ) (
Adı .....	) (
Soyadı .....	) (
Baba Adı .....	) (
Cinsiyeti .....	) (
Doğum Tarihi .....	) (
Sınav Merkezi Kodu .....	) (
Alan Kodu .....	) ( (1) ) ( (2) ) ( (3) ) ( (4)
Adres .....	) (
	) (
Semt/ilçe .....	) (
il Trafik Kodu .....	) (
Kayıt Geçerlilik Kodu ...	) (
Sınav Geçerlilik Kodu ...	) (
İŞLEM KODU ) ( (1:GERİ DÖNÜŞ, 2:SINAV BİLGİLERİ FORMUNA GEÇİŞ, 3:GÜNLEME)	

Figure 6.3 Candidate Information Form

REX - KAYIT (SINAV BİLGİLERİ)				
	Oturum ... ) (			
Salon Numarası .....	) (			
Salon Adresi .....	) (			
Sıra Numarası .....	) (			
Arşiv Numarası .....	) (			
Girdi/Girmedi .....	) (			
Ağırlıklı Puan .....	) (			
<u>Alt Alan</u>	<u>Doğru Cevap</u>	<u>Yanlış Cevap</u>	<u>Ham Puan</u>	<u>Standart Puan</u>
1	) (	) (	) (	) (
2	) (	) (	) (	) (
3	) (	) (	) (	) (
4	) (	) (	) (	) (
5	) (	) (	) (	) (
İŞLEM KODU ) ( (1:GERİ DÖNÜŞ, 2:SONRAKİ OTURUMUN FORMUNA GEÇİŞ)				

Figure 6.4 Examination Information Form

## 6.8. Center/Session/Area Distribution

This function is implemented by means of a procedure named CANDIDATES\_DISTRIBUTION by running the general purpose Distribution program. The Distribution program is stated in Section 5.2.4. The distribution of the Center/Session/Area shows the number of candidates that will enter the examination for each center, for each session and for each area. This distribution is used for the determination of the requested room capacities during the room assignment by the REX Examination Subsystem. The first dimension of the distribution shows the examination center while the second dimension shows the area code values. The number of elements of the area code array of the Candidate File varies according to the number of session in which the examination will be performed. So the row of the distribution is selected as the index of the area code array and takes at least a value of 1 and a maximum value of #OFSESSIONS. The examination center codes are in the CENTERS array of the Parameter File. In the first dimension of the distribution the values of the #OFCENTERS element of the array shall be present. The area code number is not the same for each session and they are determined by the #OFAREAS array of the Parameter File. A place will be allocated in the second dimension of the distribution large enough to cover the higher value in the array. Therefore the size of the matrix that will be used to determine the number of candidates that will take the examination in each center, each session and each area, will be as shown below.

TABLE (1:#OFSESSIONS,  
CENTERS(1):CENTERS(#OFCENTERS),  
1:MAX#OFAREAS)



## 6.9. Printouts Concerning the Candidates

These functions are implemented in the procedure named CANDIDATES\_PRINTOUTS. The printouts concerning the candidates are presented to the user in the form of a menu. This menu is shown in Figure 6.5. The printing will be started when he/she has made the choice by pressing the SPCFY key. These printouts are as follows.

-Examination Entrance Form: This form is printed separately for each session of the examination, and they are mailed to the addresses of the candidates. On this form the following information are being printed: the application number, name, surname, full name of the area in which the candidate will take the examination in the session, the date and hour of the examination, the room number, the seat number, examination center, the full name of the room and building. The room name is obtained by accessing to the concerned Room File from the Candidate File and the building name is obtained by accessing from the Room File to the Building File by the building address.

-Examination Results Form: This form being mailed to the addresses of the candidates is printed in order to acknowledge the candidates about their weighted scores obtained in each session of the examination. In this form the following information are printed: the application number, the name, surname of the candidate and the weighted scores obtained for each session of the examination.

-Whole Candidate List: These lists have the purpose to show the examination results of the candidates as a whole. Each of these printouts are as a separate matter in the menu shown in the Figure 6.5 and when one of them is chosen among the menu the Candidate File is sorted according to the the application number, surname or the



REX - KAYIT (ADAY DÖKÜMLERİ)

---

) ( 1. Sınava Giriş Belgesi  
)  
) ( 2. Sınav Sonuç Belgesi  
) ( 3. Tüm Adaylar Listesi  
) ( - Numara Sırasında  
) ( - Soyadı Sırasında  
) ( - Ağırlıklı Puan Sırasında  
) ( \* Birinci Oturum  
) ( \* İkinci Oturum  
) ( \* Üçüncü Oturum  
) ( \* Dördüncü Oturum

---

Seçiminizi yaparak SPCFY tuşuna basınız.

GERİ DÖNÜŞ ) (

---

Figure 6.5 Candidate Printouts Form

REX - (SALON KÜTÜKLERİNİ SIRALA)

---

) ( Salon adresine göre sıralama  
) ( Bina içinde salon numarasına göre sıralama

Seçiminizi yaparak SPCFY tuşuna basınız

GERİ DÖNÜŞ ) (

---

Figure 6.6 Sort Room Files Form

weighted score obtained for each session. In the printout, the candidate application number, the name and surname and the weighted score for each session are printed.

#### 6.10. Sorting the Room Files

This function is implemented in the procedure named SORT\_ROOM\_FILES. In order to determine the staff necessary for the execution of the examination, the Room Files created separately for each session by the REX Examination Subsystem has to be sorted according to the building number and the room number. These files are used to create Staff Files. After the creation of the Staff Files, this time the Room Files are sorted according to the room address in order to make it possible to get the information about the rooms. This process is called "back sort". The sorted Room Files are used for the printouts concerning the staff. The type of the sort can be selected by the menu given in Figure 6.6.

#### 6.11. Creation of the Staff Files

There is a Staff File for each session of the examination. These files are created under the name

<user system name>/STAFFn (n, means the session)

Creation of the Staff Files is being performed by a procedure named CREATE\_STAFF\_FILES. The method applied in the procedure is as follows.

The Staff Files are created by using the Room Files sorted by building number and the room number. So the staff information in the each Staff File will be in the following order.

Building staff of the first building  
Room staff of the first room of the first building  
. . .  
Room staff of the last room of the first building  
Building staff of the second building  
. . .  
Room staff of the last room of the last building

While creating the Staff File, the Room File of the concerned session is being read sequentially and when the building of the related room is changed the Building File is accessed by means of the building address. In the Building File the addresses of the Building Examination Director, Assistant Building Examination Director, Building Manager and Reserve Proctors are recorded. Then, to the records accessed by means of these addresses, the examination center code, duty code, building number and building address are loaded to the Staff File. In this stage, the required number of Proctor for each room is determined. There should be one Proctor for each 25 candidates. Then the addresses of the Room Director, and Proctors are recorded for each room of the building to the Room File. And finally by means of this addresses accessing to the Staff File the examination center code, duty code, building number, building address, room number and room addresses are being loaded. In this manner, all the processes related to all the rooms of a building are terminated and when proceeded to a new building, the number of candidates, the number of the room directors and the number of supervisors of the previous building is recorded to the Building File.

### 6.12. Printing the Staff Notification Schedule

These schedules are printed separately for each session of the examination by a procedure named PRINT\_STAFF\_SCHEDULE. In the printout there will be a line for each record of the Staff File that will consist the staff address and its duty in the examination. In this line large enough place is allocated for the staff name, surname, his/her position, and the institution they are working for, which will be filled by the Province Examination Management.

### 6.13. Transferring the Staff Data to the Files

This function is implemented in the procedure named PROCESS\_STAFF\_DATA. The information on the Staff Notification Schedules returned back after determination of the staff by the Province Examination Management, are being transferred to the Staff Information File by means of the Registration Supporting Subsystem. Using the staff address field in this file, the Staff File is accessed and the name, surname, position and institution of the staff is recorded.

### 6.14. Printouts Concerning the Staff

These functions are implemented in the procedure named STAFF\_PRINTOUTS. All the printouts concerning the staff, are presented to the user in the form of a menu. This menu is shown in Figure 6.8. The printing will be started when he/she has made the choice by pressing the SPCFY key. These printouts are as follows.

-Identification Card: This card is printed using the Staff File and related to the staff. On the Identification Card, the name, surname, duty, building number or the room number are being printed.

REX - KAYIT (GÖREVLİ DOKÜMLERİ)	
) (	1. Yaka Kartı
) (	2. Görevlendirme Belgesi
) (	3. Görevli çeki
) (	4. Bina Salon Görevli Bildirim çizelgesi
) (	5. Salon Sınav Evrakı Alındı Verildi Tutanağı
) (	6. çek Dağıtım Tutanağı
) (	7. çek Zarfı
) (	8. Görevliler ve Görev Yerleri Listesi
) (	9. Bina Sınav Evrakı Alındı Verildi Tutanağı
) (	10. çek Zarfı Dağıtım Tutanağı

---

Seçiminizi yaparak SPCFY tuşuna basınız.

GERİ DÖNÜŞ ) (

Figure 6.7 Staff Printouts Form

-Duty Assignment Form: This form is printed using the Staff, Room and Building files and related to the staff. On the Duty Assignment Form, all the information in the Staff File are being printed. Besides, for the room staff, the full name of the room and building: for the building staff, the full name of the building are being printed. Besides that, for the Room Director, the Building Examination Director and the other staff in the room; for the Proctors the Building Examination Director and the Room Director; and for the building staff, the other staff concerned in the management of the building are being specified.

-Staff Cheque: This cheque is performed using the Staff File and related to the staff. On the Staff Cheque the name and surname of the staff and the amount to be paid is printed.

-Building Staff List: This list is printed using the Staff, Room and Building Files and related to the buildings. In the list, the building number, the full name of the building and building staff information are being specified together with the room number for each room in the building, full name of the room, the room capacity and information related to the staff appointed in the room. Besides this, the Reserve Proctor for each building and the total number of candidates, Room Directors and Proctors in the building are stated as well.

-Room Document Receiving/Delivery Report: This report has similar properties of the Building Staff List. However for each room a place is left empty on which the Room Directors will sign about they have received and delivered the room documents.

-Cheque Distribution Report: This envelope is printed by using the Building and Staff file and related to the buildings. On the Cheque Distribution Report, the building number, full name of the building and all the information about the staff appointed in the building and rooms, the amount to be paid and a signature place is present.

-Cheque Envelope: This printout is printed by using the Building file and related to the Province Examination Management. On the Cheque Envelope, the number of the building, the full name of the building, using the number of room directors and supervisors the amount of Staff Cheque is being printed.

-Staff and Duty Sites List: For this printout temporary files are created by sorting the Staff File of each session according to the surname and name. In the list the name, surname, position, duty, room or building number and the institution of the staff is being printed.

-Building Document Receiving/Delivery Report:This report is printed using the Building and Staff file and related to the Province Examination Management. In the printout the building number and the full name for each of the buildings and the name and surname of the Building Examination Director are being specified. Besides this a place is left empty in order to be signed while the receipt and delivery of the examination document.

-Cheque Envelope Distribution Report:This report is printed using the Building File and related to the Province Examination Management. In the printout the building number of the buildings in a province center is being specified as well as the full name of the building and the amount of the Staff Cheques.

## 7. TESTING OF REX

REX has been tested on a sample Examination Organization with the following properties.

- User System Name is "ORNEK".
- Number of the Examination Centers is 2.
- Examination Center codes are 01 and 06.
- Number of Sessions is 2.
- First Session has 4 Areas.
- Second Session has 5 Areas.
- Each Area has 1 Subarea.
- Number of the Candidates is 308.
- In the first Examination Center,
  - 1 Building, 7 Rooms, 25 Staff  
have been used for the first Session.
  - 1 Building, 9 Rooms, 29 Staff  
have been used for the second Session.
- In the second Examination Center,
  - 1 Building, 6 Rooms, 25 Staff  
have been used for the first Session.
  - 2 Buildings, 8 Rooms, 38 Staff  
have been used for the second Session.

Some of the printouts related to this Examination Organization are given in Appendix F.



## 8. CONCLUSION

In this study a software for Registration System is designed and implemented to be used for the examination organizations of ÖSYM. Registration System is divided into two sections. The first one is a software section which performs the data processing activities by means of a computer. The second one ensures the connection between the external environment and the software section. These sections are entitled REX Registration Subsystem and Registration Supporting Subsystem respectively. Both of these subsystems have been designed, the REX Registration Subsystem has been implemented. The presence of the Registration Supporting Subsystem is necessary for several functions of the REX Registration Subsystem. However, its implementation which involves manpower and machines is not the concern of this thesis.

REX is the name of the information system which executes the whole examination organization. It has three subsystems. In addition to the REX Registration Subsystem, which mentioned above, these are the REX Examination Subsystem and the REX Supporting Subsystem. REX Examination Subsystem has been developed as a separate master thesis. REX Supporting Subsystem aims to ease some operations for the user, but it is not the focus of this thesis.

REX Registration Subsystem has two main functions. The first is directed to the candidates and the second to the staff. The activities performed for the staff are similar for every examination organization. But there may be differences for the activities related to the candidates. According to the characteristic of the application there may be a need to get additional information about the candidates. Therefore the layout of the file, which is used for recording the candidate information, will not be the same for every application. The additional information may take place in printouts

and tables. For these types of processes, program writing programs have been developed. These programs can be used for operations like transferring data from one file to another, obtaining new printouts which were not definite before and obtaining distributions of one or more variable. According to the experience gained from the real life applications, it has been inferred that all the processes can not be well-defined before the execution of the examination organization. Therefore these programs can be useful for unexpected situations.

During the development process of REX, the main task of ÖSYM, which is the selection and placement of students to the higher education programs, has been analyzed. As a result of this analysis, an original study has been actualized. The developed information system will be used for examination organizations which are less complex and have small number of applicants. Further work on this system could be directed to increasing the complexity of the organization and the number of the applicants it can handle.

## REFERENCES

- AKTAŞ, A.Z., Structured Analysis and Design of Information Systems, Prentice-Hall, 1987
- BROOKS, C.H.P. et al., Information Systems Design, Prentice-Hall, 1982
- BURCH, J.G., STRATER, F.R., Information Systems: Theory and Practice, John Wiley & Sons, 1984
- ÖSYM, İki Aşamalı Üniversitelerarası Öğrenci Seçme ve Yerleştirme Sistemi, 1980
- ÖSYM, The System of Student Selection and Placement in Higher Education Institutions in Turkey, 1984
- ÖSYM, 1982-1985 ÖSYM Çalışma Raporu, 1985a
- ÖSYM, 1985 Öğrenci Seçme ve Yerleştirme Sınavı Başvuru Kılavuzu, 1985b
- ÖSYM, 1985 Öğrenci Seçme ve Yerleştirme Sınavı Uygulama Yönergesi, 1985c
- ÖZÇELİK, D.A., Test Hazırlama Kılavuzu, ÖSYM, 1981
- PAYASLIOĞLU, A., Türkiyede Yükseköğretim Kurumlarına Öğrenci Seçme ve Yerleştirme Sistemi, ÖSYM, 1985
- TOKER, F., GÜNALP A., Recent Changes in the System of Higher Education in Turkey, ÖSYM, 1982
- TÜTÜNCÜ, M., Design and Implementation of an Automated Examination System, Master's Thesis, 1987
- WEINBERG, V., Structured Analysis, Prentice-Hall, 1980
- YILDIRIM, C., Eğitimde Ölçme ve Değerlendirme, ÖSYM, 1983
- YOK, Yükseköğretimle İlgili Kanun ve Yönetmelikler, 1987



**APPENDICES**

## A. STUDENT SELECTION AND PLACEMENT CENTER (ÖSYM)

REX is designed by the analysis of examination organizations which are applied at ÖSYM and implemented to be used in this center. So, a brief introduction of ÖSYM might be useful.

### A.1. ÖSYM Chronicle [ÖSYM, 1984]

ÖSYM, previously called the Interuniversity Student Selection and Placement Center (USYM) was established on 22.11.1974 by the Interuniversity Board in accordance with article 52 of the University Law, No.175Ø. In accordance with the Higher Education Law which came into effect as from 1981, the Center has been attached to the Higher Education Council and its name has been changed to the Student Selection and Placement Center (ÖSYM). According to article 1Ø of the Higher Education Law: "The Student Selection and Placement Center determines, in the context of fundamentals established by the Higher Education Council, the examination principles of the students to be admitted to the institutions of higher education, it prepares the tests, administers them, evaluates them on the basis of their results and the principles determined by the Higher Education Council and in the light of student demands, effects the placement of student candidates in universities and other higher educational institutions, taking into account, as it does so, the students' own preferences, and carries out research related to these activities."

The President of ÖSYM is appointed by the President of the Higher Education Council and is responsible for the administration of the Center. The President presides over the Executive and Advisory Committees. The Executive Committee consists of three members selected for three years by the President of the Higher Education Council among six candidates proposed by the President of ÖSYM. The Advisory Committee is set up as follows: one member

is selected by the Higher Education Council from among two candidates proposed by each university rector and four members are assigned by the Ministry of National Education Youth and Sports. ÖSYM is composed of the following units: The Presidency, The General Secretariat, Information Data Processing, Research, Development and Evaluation, The Examination Services, Library and Documentation, Legal Advisory, and Maintenance. At present a total of 70 full-time academic staff work at the Center and there are 35 specialists employed on a part-time basis. Financially ÖSYM is self-supporting. Candidates for places in the higher education institutions pay a registration and an examination fee for the selection and placement examinations. Otherwise, ÖSYM has no other sources of income.

## A.2. Organization Schema of ÖSYM

Organization Schema of ÖSYM is shown in Figure A.1. The functions of the units shown in organization schema are as follows [YÖK, 1987].

### i) General Secretariat

-To provide the conformability of the center's administrative departments and offices activities concerning the entire center with the related laws, regulations, bylaws and directives.

-To execute the president's correspondences.

-To regulate all protocol, visits and ceremonies of the president.

### ii) Data Processing Unit

-To design, regulate and apply the evaluation, classification, selection and placement process, which are performed for higher education student selection and placement.

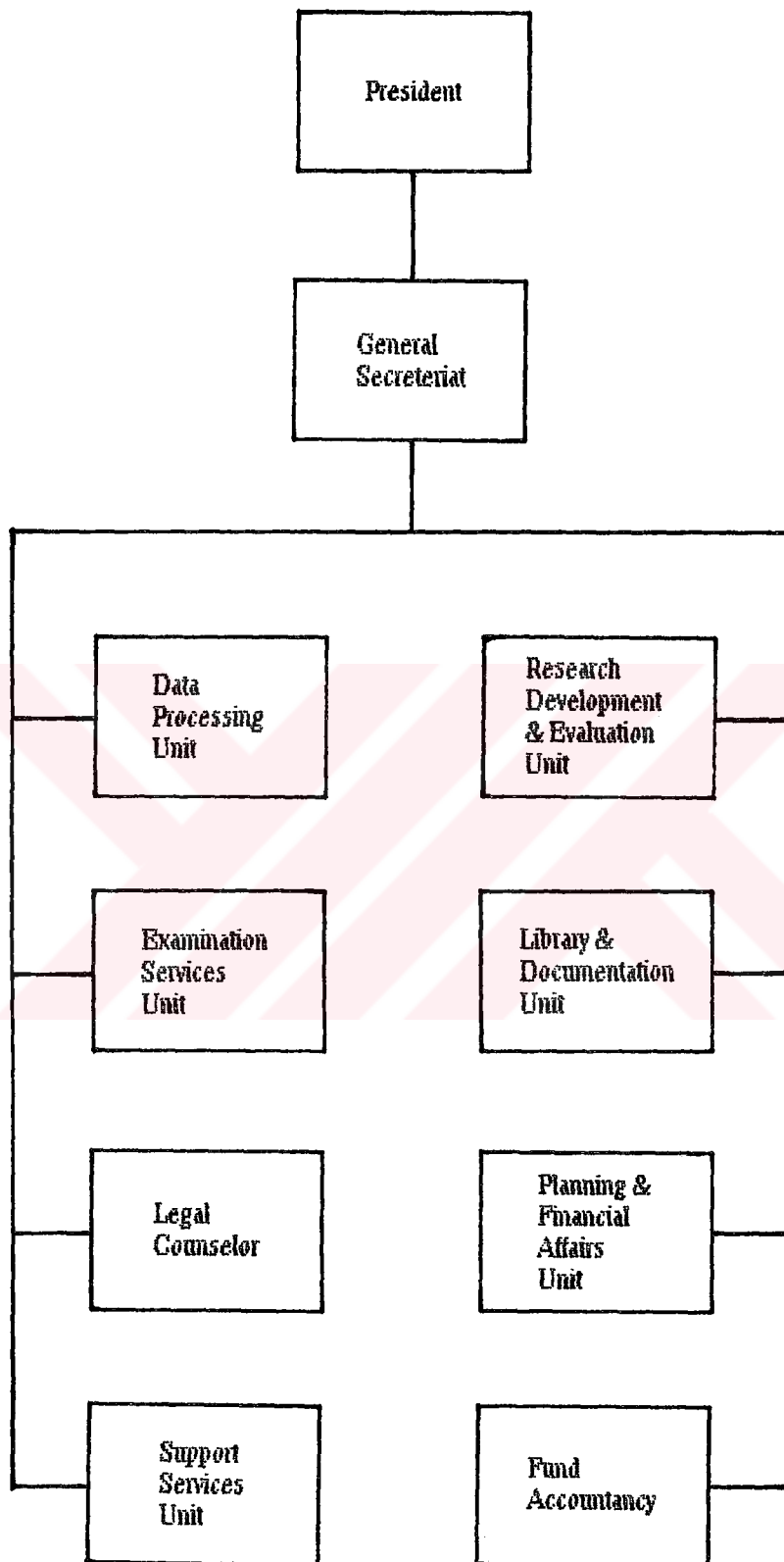


Figure A.1 Organization Schema of ÖSYM

-To provide all necessary data processing services to realize the statistical analysis and researches related with student selection and placement activities.

#### iii) Research, Development and Evaluation Unit

-To prepare, perform and evaluate the results of research studies to be the basis of higher education institutions' decisions on student selection and placement; or to help or support this kind of research projects.

-To cooperate with the other institutions concerning with the research projects and to realize the education and publication activities related with research results.

-To prepare the tests and similar measuring tools which are to be used for student selection and placement activities and examinations.

-To perform research and development studies in order to increase the validity and reliability of the prepared tests and measurement tools.

-To prepare proficiency tests on various purposes and success tests on intermediate school, high school and undergraduate levels.

-To perform education and publication studies related with the measurement, evaluation and development of these tests.

#### iv) Examination Services Unit

-To execute application process, examination organization and application services for student selection and placement activities for higher education.

-To perform archives and microfilm duties of documents.

-To handle all kinds of correspondence with the candidates.



v) Library and Documentation Unit

-To execute library activities.

-To collect and protect all kinds of information and documents related with the examinations; and to serve for the users whenever required.

vi) Legal Counselor

-To consult Student Selection and Placement Center in legal matters.

-To take the side of the center in trials.

vii) Planning and Financial Affairs Unit

-To perform the activities on center's administrative and economic subjects.

-To prepare the budgetary forecasts of the center.

-To apply the final budgets.

-To fulfill the requirements of the purchasing regulations.

viii) Support Services Unit

-To execute the related activities of Student Selection and Placement Center personnel's personal rights.

-To perform all technical services of the center, such as maintenance, repairment etc.

-To perform the center's civil defence, security and environment control activities.

-To perform printing and graphics services.

-To carry out lighting, heating, cleaning and similar duties.

ix) Fund Accountancy

-To conduct the Student Selection and Placement Center fund.

-To provide the execution of all activities related with the fund's accountancy.

A.3. Activities of ÖSYM

The services offered by ÖSYM are as follows [ÖSYM, 1984] [PAYASLIOĞLU, A., 1985].

-Student Selection and Placement Examinations for the universities (ÖSYS).

-Foreign Student Examinations (YÖS).

-Registration and examination activities for the Open University (AÖF).

-Central Foreign Proficiency Examinations for Associate Professorship.

-Preparation of special tests for some institutions and evaluation of the results.

-Personnel selection examinations for some public institutions.

-Collection and publication of statistical information on subjects of professional interest.

-Selection examinations for students in the graduate class of high schools in order to give them the facility of education in foreign countries (AFS).

-Selection examinations of the faculty of medicine graduates for specialization studies on medicine (TUS).

-Collection and processing of all statistical data of academic staff and students in the higher education institutions.

## B. EXISTING EXAMINATION ORGANIZATION SYSTEM

The activities of an Examination Organization System related with data processing are carried out by the Data Processing Unit of ÖSYM that the organization schema was given Figure A.1. All the documents used in the examination are prepared by this unit. Therefore the functions of Data Processing Unit will be introduced firstly, then the execution of the examination will be presented.

### B.1. Data Processing Unit

Functional schema of Data Processing Unit is shown in Figure B.1, overall schema of the Computer System which is installed in the unit is shown in Figure B.2 and overall schema of Optical Reader Systems is shown in Figure B.3. According to the schema B.1 the Data Processing Unit is divided into five subunits. The descriptions of these units are given below.

#### B.1.1. General Supporting Subunit

The functions of this subunit can be outlined as follows.

- To find solutions for the problems which happens during the use of the computer system by other subunits.
- To take precautions required for more efficient use of the computer system.
- Ensuring the use of any kind of program packages.
- Developing system programs on the Optical Reader.

#### B.1.2. Candidate Registration Subunit

It is the subunit which performs the activities prior to the examination. The outline of the activities performed by this subunit is as follows.

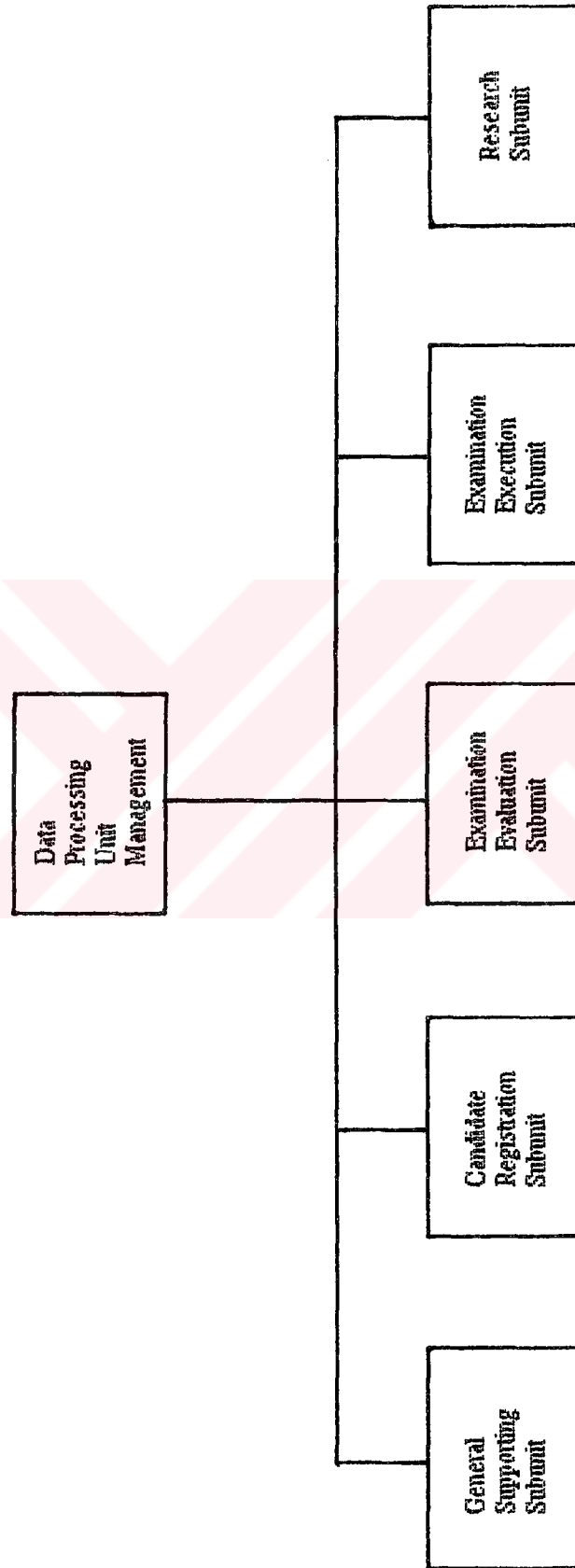


Figure B.1 Functional Schema of Data Processing Unit

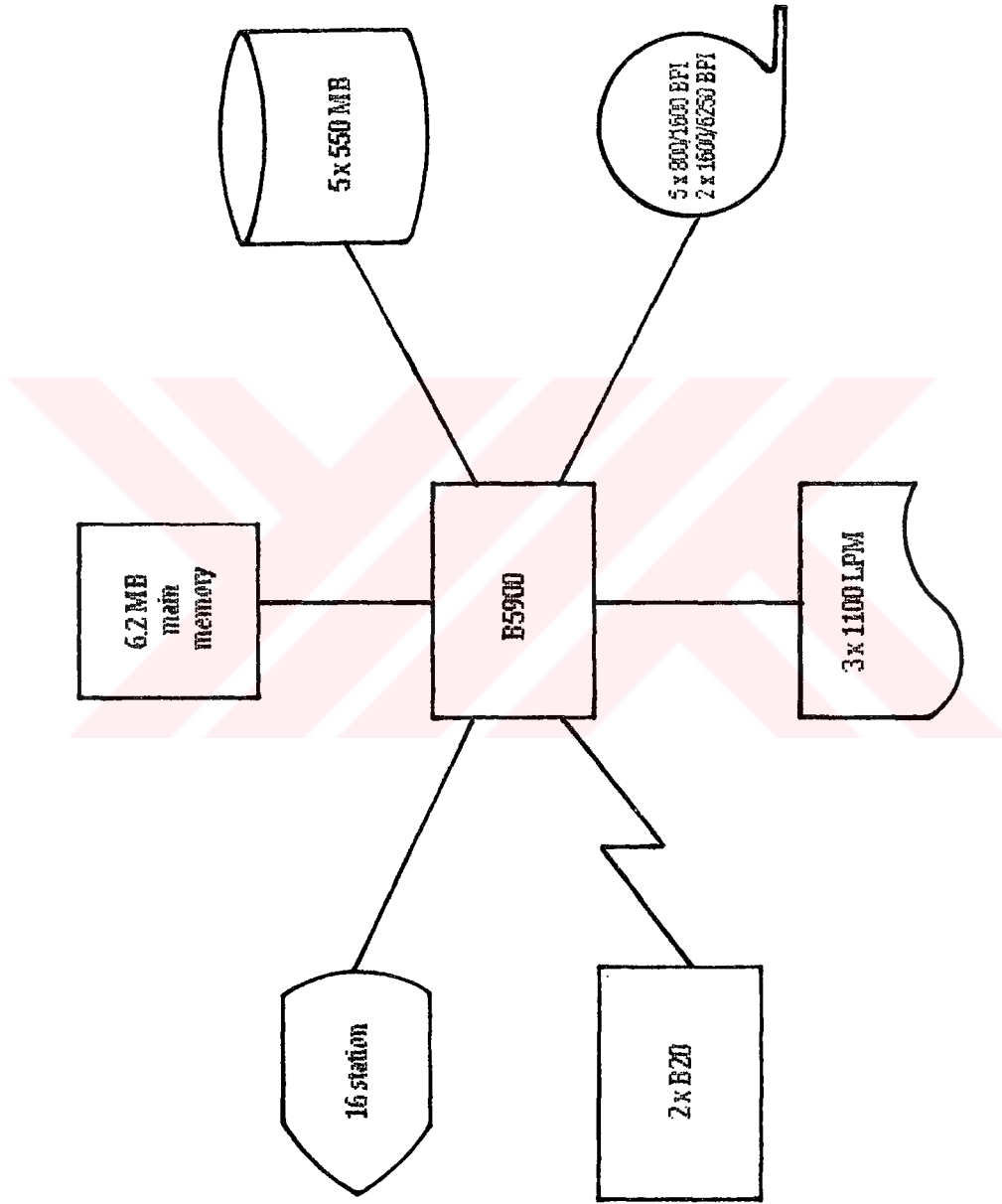


Figure B.2 Overall Schema of Computer System

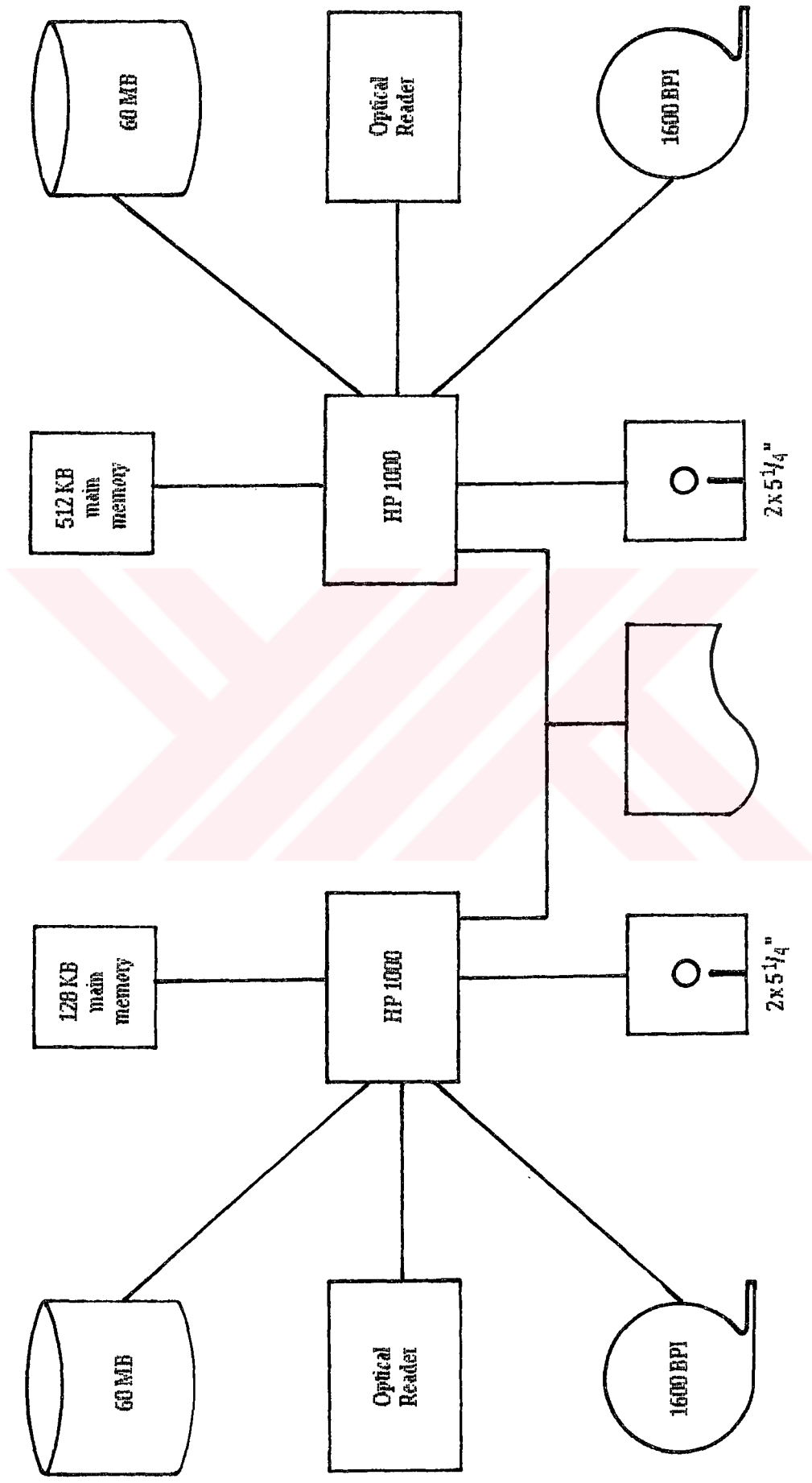


Figure B.3 Overall Schema of Optical Reader Systems

- Design and drawing of the Application Form.
- Preparation of the Application Manual including the basic principles, application conditions concerning the examination and including the information concerning the filling of the Application Form, the documents that will be sent to the candidates, the rules that will be obeyed during the examination, how the examination is evaluated and the means by which the results are announced.
- Printing the application numbers to the Application Forms by means of a computer.
- Ensuring that the Application Manual and Forms has been reached to the candidates.
- Creation of an empty Candidate File.
- Optical reading of the Application Forms incoming from the candidates.
- Transfer of the candidate information obtained with the optical reading to the Candidate File.
- Controlling the information by listing the candidate information recorded in the Candidate File.
- Correction on the Application Forms, and sending back to the optical reading of the documents in which some incomplete or erroneous information has been encountered during the information control process.
- Proceeding to the information controls till the Candidate File will be completely correct.
- Printing of the Examination Entrance Forms (SGB) at the end of the room assignment process realized by the Examination Execution and Evaluation Subunits.
- Printing of the Examination Results Forms (SSB), after that the examination has been realized and the results obtained.

### B.1.3. Examination Evaluation Subunit

It is the subunit which performs the room assignment before the examination and evaluation activities after the examination, respectively. The outline of the activities performed by this subunit is as follows.

-Creating the Examination File by means of the Candidate File, in order to assign the candidates to the rooms.

-Supplying the statistical information of the candidates to be assigned in order to ensure the creation of the Room and Building Files by the Execution Subunit.

-Designation of the building and rooms in the examination centers, in which the candidates will take the examination (Room Assignment).

-Transferring the assignment information (Room #, Seat #) at the end of the process to the Candidate File in order to obtain the Examination Entrance Forms by the Candidate Registration Subunit.

-Printing of the Answer Sheets belonging to the candidates that will take the examination, by means of a computer.

-Optical reading of the blank Answer Sheets in order to check whether any kind of fault or dirt are present or not.

-Printing of the Answer Sheet Receipt List in order to ensure the receipt of the Answer Sheets from the examination centers, at the end of the examination.

-Obtaining the answers given by the candidates in the examination, by optical reading of the Answer Sheets received.

-Second optical reading and repairing of the Answer Sheets which have been damaged during the optical reading process.

-Determination of the raw scores, finding the number of the correct and incorrect answers for each test, by confronting the correct answers with those given by the candidates during the examination.

-Calculation of the mean and standard deviation for each test and obtaining the raw score distribution.

-Calculation of the standard scores for each test using the mean and standard deviation.



-Transferring the number of correct and incorrect answers to the Candidate File in order to obtain the raw, standard and weighted scores for printing the Examination Results Form.

#### B.1.4. Examination Execution Subunit

It executes the activities concerning the staff which performs the examination and the activities belonging to the preparation of the documents used in the examination. The outline of the activities performed by this subunit is as follows.

-Determination of sufficient amount of conformable room and building for each examination center after that the number of the candidates taking the examination has been settled by Examination Evaluation Subunit.

-Sending the lists concerning the information about the characteristics and the number of staff required for the examination buildings and rooms, to the Province Examination Managers after the realization of the room assignment process by the Examination Evaluation Subunit.

-Creating the Staff File after that the staff has been determined by the Province Examination Manager.

-Preparation of the Examination Execution Manual consisting of the rules of the examination and the principles directed to the execution of the examination, in order to be supplied to the examination staff.

-Preparation of the following printouts.

##### 1) Printouts Concerning the Staff

1-Duty Assignment Form

2-Identification Card

3-Official cheque

ii) Printouts concerning the Rooms

- 1-Room Identification Card
- 2-Room Candidate Attendance List
- 3-Room Examination Report

iii) Printouts Concerning the Buildings

- 1-Building Room List
- 2-Building Staff List
- 3-Room Document Receipt/Delivery Report
- 4-Extra Document Envelope
- 5-Building Extra Document Utilization Schedule
- 6-Cheque Distribution Report
- 7-Cheque Envelope

iv) Printouts Concerning the Province Exam Management

- 1-Staff and Duty Site List
- 2-Building Document Receipt/Delivery Report
- 3-Province Extra Document Utilization Schedule
- 4-Cheque Envelope Distribution Report

**B.1.5. Research Subunit**

The outline of the activities performed by this subunit is as follows.

-Evaluation of the inquiries concerning the performed examinations.

-Realization of the subject analysis in order to determine whether the test questions asked to the candidates are appropriate to their aims or not.

-Performing the statistical researches whenever required.

## B.2. Execution of the Examination

In the examination organization several documents are used for different purposes in different times. Some of these documents are directed to the candidates and some other to the staff. The execution of the examination is performed by the staff which is liable to perform several tasks. The staff perform their tasks on different site during the execution of the examination [ÖSYM, 1985c].

### B.2.1. Staff

The person who has the ultimate authority and responsibility in the examination, is the ÖSYM President. The other authorities in the hierarchy in charge of the execution of the examination are listed below.

#### i) Province Examination Manager

He has the top level authority and responsibility for the organization and execution of the examination in the province. All the staff related to the examination are appointed by the Province Examination Manager except those assigned by the ÖSYM Presidency.

The Province Examination Manager declares the duties of people whom he appointed by means of a Duty Assignment Form. Besides this, he provides the staff to get the Examination Execution Manual and the Identification Cards on which the name, surname, type and place of work information is written.

#### ii) Assistant Province Examination Manager

He is the closest companion of the Province Examination Manager. He assists the Province Examination Manager in organization and application of the examination.

iii) Building Examination Director

He is responsible from the preparation of the building, rooms and execution of the examination in all rooms in accordance with the examination terms.

On the examination date, he receives the Building Examination Document from City Examination Courier in the examination building. He gives back the same documents to the City Examination Courier in the examination building after the examination. These documents are as follows.

- Room Document of the Building
- Building Document Receipt/Delivery Report
- Extra Question Booklet Package
- Extra Answer Sheets Package
- Building Staff List
- Examination Execution Manual
- Cheque Envelope

Duties of Building Examination Director can be summarized as follows.

a)He provides the preparation of rooms for the examination by contacting with the Building Manager on time.

b)He provides the placement of Room Identification Cards that he gets from the Province Examination Manager on room doors, at least two days before the examination.

c)He ensures that the Examination Execution Manual is read by all Room Directors in his building.

iv) Assistant Building Examination Director

In the buildings which has more than twenty rooms or more than a thousand candidates, an assistant for the Building Examination Director is appointed. This assistant, cooperates with the Building Examination

Director in all aspects and attends all receiving and returning processes of Building Examination Document.

v) Building Manager

The Building Managers are generally selected from the directors or managers of the institution in the building that the examination will take place. He ensures the preparation of the building and rooms in accordance with examination terms. He helps the Building Examination Director in delivery works of examination documents.

vi) Room Director

The authority and responsibility of executing the examination in accordance with the room rules is given to the Room Director.

His/Her duties can be summarized as follows.

a) To examine the examination room with the Examination Proctors.

b) To assign seat numbers to the seats of the room.

c) To admit the candidates to the examination room by controlling their identification cards.

d) To bring the related examination documents to the room after receiving them from the Building Examination Director. These documents are as follows.

- Answer Sheets Package
- Room Candidate Attendance List
- Room Examination Report
- Question Booklets Package
- Examination Execution Manual

e) To check the attendance by using Room Candidate Attendance List.

f)To provide the distribution of Answer Sheets to the candidates.

g)To read the rules written on the back side of the Examination Execution Manual before starting the examination.

h)To provide the distribution of Question Booklets to the candidates.

i)To inform the candidates about the duration of the examination.

j)To report the necessary points on the Room Examination Report about the candidates who are cheating or the ones who disagree to obey the examination rules during the examination.

k)To provide the collection of Answer Sheets and Question Booklets after the examination.

l)To deliver the Room Examination Document to the Building Examination Director.

#### vii) Proctor/Reserve Proctor

Under the authority and responsibility of the Room Director, he supervises the execution of the examination in his room in conformance with examination rules. He assists the Room Director in identification card controls, distribution and collection of examination documents to the candidates.

#### viii) ÖSYM Representative

A representative from ÖSYM is appointed to standardize the examination organization and to help all staff. ÖSYM representatives prepare a report including their observations, evaluations and suggestions and they send these reports to the ÖSYM Presidency.

The ÖSYM Representative is responsible before the ÖSYM President and the Province Examination Manager. If necessary, he can enter each room and control all examination details and candidates' identification cards.

If the ÖSYM representative reports that the examination is not performed in conformance with the examination rules or collective cheating intentions have been occurred, then ÖSYM can cancel the results by using its discretionary power.

ix) Province Examination Courier

Appointed by the Province Examination Manager. He receives the examination document bags of his own distribution or collection group's building from the Province Examination Manager on the examination date's morning. He delivers these documents to the Building Examination Director. After the examination, he receives the same documents from the Building Examination Directors and delivers to the Province Examination Management. Delivery process is carried out by signing the Building Document Receipt/Delivery Report which is prepared by ÖSYM and sent to Province Examination Managements.

x) Examination Documents Distribution Courier

He takes the examination documents in closed and sealed trucks from ÖSYM and takes them to the Province Examination Managers. After the examination, he receives the same examination documents from the Province Examination managers and delivers to ÖSYM.

xi) Examination Documents Security Courier

He takes his place in the examination center to which he is appointed on the day before the examination and cooperate with the Province Examination Manager for the security of the examination documents. He assists the Province Examination Manager in collection of examination documents and their delivery to the Examination Documents Distribution Couriers after the examination.

### B.2.2. Examination Places

The examination is performed in examination centers. There is one Province Examination Manager for each examination center. The examination is executed in the buildings which are in the center boundaries. There is one Examination Director and Manager for each building. The candidates take the examination in rooms of the building. The Room Director is responsible from the execution of the examination in the room.





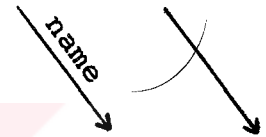
## C. STRUCTURED TOOLS [AKTAŞ, A.Z., 1987]

### C.1. Data Flow Diagram (DFD)

Data Flow Diagrams can be used either to describe an existing system or a proposed system at the logical level without considering the physical environment. It can be defined as a logical model that describes a system as a network of processes connected to each other and to data stores and also to sources/sinks. The basic symbols that are used in a DFD are as follows.

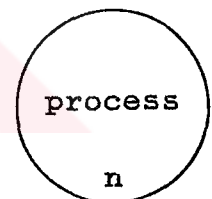
#### a) Data Flow

An arrow is used to represent a flow of data. The name of the data flow is written through or next to the line.



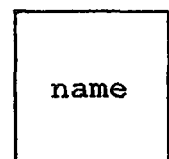
#### b) Process

A circle represents an automated or manual task or process. It identifies input data that flows into the circle, the transformation that input data undergoes, and the output that flows out of the circle. A brief descriptive statement and a reference number for the process is written inside the circle.



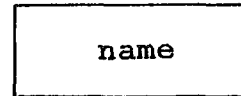
#### c) Source/Sink

A square is used as an external entity symbol to represent an area where data originates (i.e. source) or terminates (i.e. sink). The name of the entity is written inside the square, in singular. A source produces data flows that our system processes and a sink receives data flows that our system produces.



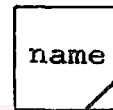
d) Storage

An open ended rectangle represents a store of information or objects, irrespective of the physical storage medium. The name of the store is written inside the symbol. The store symbol identifies a time delay for its content. If data elements do not flow from one process to the next directly; that is, they are stored for a period of time, this symbol can be used to show such delays.



e) Duplication Symbols

Some sources/sinks or data stores can be drawn more than once on the same DFD in order to minimize the crossing of the data flow lines. The duplicate sources/sinks are identified by a "/" or a "\*" symbol. The duplicate data stores are identified by a "|" or a "\*" symbol.



f) Additional Symbols

By defining some relational operators, capabilities of a DFD can be improved. These operators are:

\* denotes a logical AND connection

⊕ denotes an exclusive OR connection

○ denotes an inclusive OR connection

## C.2. Warnier/Orr Diagram

Warnier/Orr Diagrams are used to represent data structures as well as processes. The main tool in a Warnier/Orr Diagram is the brace '{' and it shows decomposition of the system. Things that do not decompose further are called elements. If a structure is represented by a Warnier/Orr Diagram, elements are data elements; and if process of a system is expressed, then elements are elementary operations. General form of structures using Warnier/Orr Diagrams are as follows.

### a) Hierarchy

The general form of hierarchy is

$$\text{aaa} \{ \text{bb} \{ \text{c}$$

which means 'aaa' consists of 'bb' and 'bb' consists of 'c'.

### b) Sequence

The general form of sequence is

$$\text{aaa} \left\{ \begin{array}{l} \text{aa} \\ \text{bb} \\ \text{cc} \end{array} \right.$$

which means 'aaa' consists of 'aa' followed by 'bb' and followed by 'cc'.

### c) Repetition

The general form of repetition is

$$\text{aaa} \{ \text{bb} \\ (1, N)$$

which means 'aaa' occurs one to N times.

d) Selection

The general form of selection is

$$aaa \left\{ \begin{array}{l} bb \{ \\ (\emptyset, 1) \{ \\ \emptyset \\ cc \{ \\ (\emptyset, 1) \{ \end{array} \right.$$

e) Concurrency

The general form of concurrency is

$$aaa \left\{ \begin{array}{l} bb \\ + \\ cc \end{array} \right.$$

which means 'aaa' consists of both 'bb' and 'cc' and their order is not important.

f) Recursion

The general form of recursion is

$$aaa \left\{ \begin{array}{l} bb \\ aaa \{ \end{array} \right.$$

which means 'aaa' consists of 'bb' and itself.

## **D. GLOSSARY**

Başvuru Belgesi	Application Form
Başvuru Kılavuzu	Application Manual
Bina Salon Görevli Gösterim Çizelgesi	Building Staff List
Bina Sınav Evrakı Alındı/Verildi Tutanağı	Building Document Receipt/Delivery Report
Bina Sınav Sorumlusu	Building Examination Director
Bina Sınav Sorumlusu Yardımcısı	Assistant Building Examination Director
Bina Yedek Sınav Evrakı Kullanım Çizelgesi	Building Extra Document Utilization Schedule
Bina Yöneticisi	Building Manager
Binadaki Salonlar Listesi	Building Room List
Cevap Kağıdı	Answer Sheet
Çek Dağıtım Tutanağı	Cheque Distribution Report
Çek Zarfı	Cheque Envelope
Çek Zarfı Dağıtım Tutanağı	Cheque Envelope Distribution Report
Görevli Çeki	Staff Cheque
Görevli Gösterim Çizelgesi	Staff Notification Schedule
Görevlendirme Belgesi	Duty Assignment Form
Görevliler ve Görev Yerleri Listesi	Staff and Duty Sites List
Gözetmen	Proctor
İl Sınav Kuryesi	Province Examination Courier
İl Sınav Yöneticisi	Province Examination Manager
öSYM Temsilcisi	öSYM Representative
Salon Aday Yoklama Listesi	Room Candidate Attendance List
Salon Başkanı	Room Director
Salon Sınav Evrakı Alındı/Verildi Tutanağı	Room Document Receipt/Delivery Report
Salon Sınav Tutanağı	Room Examination Report
Salon Tanıtım Kartı	Room Identification Card
Sınav Evrakı Dağıtım Kuryesi	Examination Document Distribution Courier
Sınav Evrakı Gönderme Listesi	Examination Document Delivery List
Sınav Evrakı Koruma Kuryesi	Examination Document Security Courier
Sınav Evrakı Teslim Alma Listesi	Examination Document Receipt list
Sınav Sonuç Belgesi	Examination Results Form
Sınav Uygulama Yönergesi	Examination Execution Manual
Sınava Giriş Belgesi	Examination Entrance Form
Yaka Kartı	Identification Card
Yedek Gözetmen	Reserve Proctor
Yedek Sınav Evrakı Zarfı	Extra Document Envelope

APPENDIX E



```

/*          - - - - -          */
/*          R E X / S Y S      */
/*          - - - - -          */
SYSTEM:PROC OPTIONS(MAIN);

```

```

DCL R FILE RECORD ENV(KIND='DC',MAXRECSIZE=1923,FRAMESIZE=8,
                      BLOCKSIZE=1923,MYUSE='IO');

```

```

DCL
( NUL,SOH,STX,ETX,EOT,ENQ,ACK,BEL,BS,HT,LF,VT,CLS/* FF */,
  CR,REVERSE/* SO */,UNDERLINE/* SI */,DLE,DC1,DC2,DC3,DC4,
  NAK,SYN,NORMAL/* ETB */,BLINK/* CAN */,SECURE/* EM */,
  BRIGHT/* SUB */,ESC,FS/* WRITE ONLY */,GS/* RIGHT JUST */,
  RS/* END FIELD */,US/* LEFT JUST */ ) CHAR(1);

```

```

UNSPEC(      NUL)='00000000'; UNSPEC(      SOH)='00000001';
UNSPEC(      STX)='00000010'; UNSPEC(      ETX)='00000011';
UNSPEC(      EOT)='00111110'; UNSPEC(      ENQ)='00101101';
UNSPEC(      ACK)='00101110'; UNSPEC(      BEL)='00101111';
UNSPEC(      BS)='00010110'; UNSPEC(      HT)='00000101';
UNSPEC(      LF)='00100101'; UNSPEC(      VT)='00001011';
UNSPEC(      CLS)='00001100'; UNSPEC(      CR)='00001101';
UNSPEC( REVERSE)='00001110'; UNSPEC(UNDERLINE)='00001111';
UNSPEC(      DLE)='00010000'; UNSPEC(      DC1)='00010001';
UNSPEC(      DC2)='00010010'; UNSPEC(      DC3)='00010011';
UNSPEC(      DC4)='00111100'; UNSPEC(      NAK)='00111101';
UNSPEC(      SYN)='00110010'; UNSPEC(      NORMAL)='00100110';
UNSPEC(      BLINK)='00011000'; UNSPEC(      SECURE)='00011001';
UNSPEC(      BRIGHT)='00111111'; UNSPEC(      ESC)='00100111';
UNSPEC(      FS)='00011100'; UNSPEC(      GS)='00011101';
UNSPEC(      RS)='00011110'; UNSPEC(      US)='00011111';

```

```

DCL DC2ETX CHAR(2); DC2ETX=DC2!!ETX;

```

```

DCL ASCII CHAR(80)
INIT(' !"#$%&'()*+,-./0123456789:;<=>?@ABCDEFGH*!!
      'HIJKLMNOPQRSTUVWXYZ[\]^_`abcdefghijklmno'),
POS(80) CHAR(1) DEF ASCII POS(1);

```

```

% INCLUDE 'REX/SCREENS/0.';
% INCLUDE 'REX/SCREENS/1.';
% INCLUDE 'REX/SCREENS/2.';
% INCLUDE 'REX/SCREENS/4.';

```

```

DCL U FILE RECORD ENV(KIND='DISK',MAXRECSIZE=180,AREAS=5,
                      AREASIZE=1);

```

```

DCL 1 RECORD0,
    2 SYSID          CHAR(5),
    2 SYSPASS       CHAR(5),
    2 SYSNAME       CHAR(54);

```

```

DCL 1 RECORD1,
    2 #OFCENTERS    PIC'99',
    2 #OFSESSIONS   PIC'9',
    2 SESSIONS(4),
    3 #OFAREAS      PIC'99',
    3 #OFSUBAREAS   PIC'9',
    3 EXAMDATE      CHAR(8),

```

```

        3 EXAMHOUR          CHAR(5),
        2 TOTALAREA        PIC'99',
        2 MAX#OFAREAS      PIC'99',
        2 MAX#OFSUBAREAS   PIC'9',
        2 SIZEOFOPTIONAL   PIC'999';
DCL 1 RECORD2,
        2 CENTERS(90)      PIC'99';
DCL 1 RECORD3,
        2 EXPECTED        PIC'99999',
        2 ENTERED         PIC'99999';
DCL 1 RECORD4,
        2 #OFSTATIONS     PIC'99',
        2 STATIONS(10)    PIC'9999';

```

```
ON RECORD(U);
```

```

GETUSERPARAMS:PROC(SYSID);
  DCL SYSID CHAR(*);
  TITLE(U)=SYSID !! '/PARAMETERS';
  OPEN FILE(U) INPUT;
  READ FILE(U) INTO(RECORD0);
  READ FILE(U) INTO(RECORD1);
  READ FILE(U) INTO(RECORD2);
  READ FILE(U) INTO(RECORD3);
  READ FILE(U) INTO(RECORD4);
  CLOSE FILE(U) ENV(LOCK);
END GETUSERPARAMS;

```

```

DCL LIB1 LIBRARY(TITLE='OBJECT/REX/LIBRARY/SUBSYSTEM. '),
  LIB2 LIBRARY(TITLE='OBJECT/REX/LIBRARY/FILEMANAGER. ');

```

```

DCL SUBSYSTEM ENTRY(FIXED) RETURNS(BIT(1))
  OPTIONS(LIBRARY=LIB1),
  ( COPY      ENTRY(CHAR(*),CHAR(*)) RETURNS(BIT(1)),
    CHANGE    ENTRY(CHAR(*),CHAR(*)) RETURNS(BIT(1)),
    SEARCH    ENTRY(CHAR(*),CHAR(*)) RETURNS(BIT(1)),
    REMOVE    ENTRY(CHAR(*)) RETURNS(BIT(1)) )
  OPTIONS(LIBRARY=LIB2);

```

```

DCL 1 STRUCT0,
        2 SYSID            CHAR(5),
        2 SYSPASS          CHAR(5);
DCL 1 STRUCT2,
        2 SYSNAME          CHAR(54),
        2 #OFCENTERS       PIC'99',
        2 CENTERS(90)      PIC'99',
        2 #OFSESSIONS     PIC'9',
        2 SESSIONS(4),
          3 #OFAREAS       PIC'99',
          3 #OFSUBAREAS    PIC'9',
          3 EXAMDATE       CHAR(8),
          3 EXAMHOUR       CHAR(5),
        2 EXPECTED        PIC'99999',
        2 CONTROL          CHAR(1);
DCL 1 STRUCT4,
        2 SPCFY(4)         CHAR(1);

```



```

DCL MESSAGE          CHAR(200) VAR,
MYSTA                PIC'9999',
( CONTINUEFLAG,
  NOTSPCFYFLAG,
  RETURNFLAG )      BIT(1),
MYSELF              BUILTIN;

ON RECORD(R);

MYSTA=STATION(MYSELF) * -1;

DO WHILE(1);
  CONTINUEFLAG='1'B;
  DO WHILE(CONTINUEFLAG);
    WRITE FILE(R) FROM(SCREEN0);
    READ FILE(R) INTO(STRUCT0);
    IF STRUCT0.SYSID = 'QUIT' THEN DO;
      CALL REMOVE(MYSTA !! '/PARAMETERS');
      CALL DELSTATION(MYSTA);
      STOP;
    END;
    IF SUBSTR(STRUCT0.SYSID,1,1) = ESC THEN DO;
      CALL FINDUSERSYS;
      WRITE FILE(R) FROM(SCREEN1);
      READ FILE(R) INTO(MESSAGE);
    END; ELSE
      CONTINUEFLAG='0'B;
  END;
  RETURNFLAG='0'B;
  TITLE(U)=EXCEPT(' ',STRUCT0.SYSID) !! '/PARAMETERS';
  IF ~ RESIDENT(U) THEN DO;
    WRITE FILE(R) FROM(SCREEN2);
    READ FILE(R) INTO(STRUCT2);
    IF CONTROL = ' ' THEN DO;
      OPEN FILE(U) OUTPUT;
      RECORD0=STRUCT0, BY NAME;
      RECORD0=STRUCT2, BY NAME;
      WRITE FILE(U) FROM(RECORD0);
      TOTALAREA=SUM(STRUCT2.#OFAREAS);
      MAX#OFAREAS=MAX(
        STRUCT2.#OFAREAS(1),STRUCT2.#OFAREAS(2),
        STRUCT2.#OFAREAS(3),STRUCT2.#OFAREAS(4));
      MAX#OFSUBAREAS=MAX(
        STRUCT2.#OFSUBAREAS(1),STRUCT2.#OFSUBAREAS(2),
        STRUCT2.#OFSUBAREAS(3),STRUCT2.#OFSUBAREAS(4));
      SIZEOPTIONAL=0;
      RECORD1=STRUCT2, BY NAME;
      WRITE FILE(U) FROM(RECORD1);
      RECORD2=STRUCT2, BY NAME;
      WRITE FILE(U) FROM(RECORD2);
      ENTERED=0;
      RECORD3=STRUCT2, BY NAME;
      WRITE FILE(U) FROM(RECORD3);
      #OFSTATIONS=0;
      STATIONS(*)=0;
      WRITE FILE(U) FROM(RECORD4);
      CLOSE FILE(U) ENV(LOCK);
    END;
  END;
END;

```

```

        END; ELSE
            RETURNFLAG='1'B;
    END; ELSE DO;
        OPEN FILE(U) INPUT;
        READ FILE(U) INTO(RECORD0);
        IF STRUCT0.SYSPASS ^= RECORD0.SYSPASS THEN
            RETURNFLAG='1'B;
        CLOSE FILE(U);
    END;
    IF ~ RETURNFLAG THEN DO;
        CALL COPY(EXCEPT(' ',STRUCT0.SYSID) !! '/PARAMETERS',
            MYSTA !! '/PARAMETERS');
        CALL ADDSTATION(MYSTA);
    END;
    DO WHILE( ~ RETURNFLAG);
        NOTSPCFYFLAG='1'B;
        DO WHILE(NOTSPCFYFLAG);
            NOTSPCFYFLAG='0'B;
            WRITE FILE(R) FROM(SCREEN4);
            READ FILE(R) INTO(STRUCT4);
            IF SPCFY(1) = ESC THEN
                IF SPCFY(3) = POS(31) &
                    SPCFY(4) = POS(7) THEN TYPE=1; ELSE
                IF SPCFY(3) = POS(31) &
                    SPCFY(4) = POS(9) THEN TYPE=2; ELSE
                IF SPCFY(3) = POS(31) &
                    SPCFY(4) = POS(11) THEN TYPE=3; ELSE
                IF SPCFY(3) = POS(79) &
                    SPCFY(4) = POS(22) THEN RETURNFLAG='1'B;
            ELSE DO;
                SUBSTR(SCREEN4,1842,80)=BLINK!!
                'Gecersiz konumda SPCFY tusuna bastiniz.';
                NOTSPCFYFLAG='1'B;
            END;
        ELSE DO;
            SUBSTR(SCREEN4,1842,80)=BLINK!!
            'XMIT yerine SPCFY tusunu kullaniniz.';
            NOTSPCFYFLAG='1'B;
        END;
    END;
    SUBSTR(SCREEN4,1842,80)=' ';
    IF ~ RETURNFLAG THEN DO;
        MESSAGE=CLS!!ESC!!"2-!!BLINK!!
            'Altsisteme gecis icin lutfen bekleyiniz!!'
            ESC!!"o7'!!DC2ETX;
        WRITE FILE(R) FROM(MESSAGE);
        CLOSE FILE(R);
        CALL SUBSYSTEM(TYPE);
    END;
END;
END;
END;

ADDSTATION:PROC(STA#);
DCL STA# PIC'9999',
    FLAG BIT(1);
OPEN FILE(U) UPDATE;
READ FILE(U) INTO(RECORD4) INDEX(4);

```

```

FLAG='1'B;
DO I=1 TO 10 WHILE(FLAG);
  IF STATIONS(I)=0 THEN DO;
    #OFSTATIONS=#OFSTATIONS+1;
    STATIONS(I)=STA#;
    FLAG='0'B;
  END;
END;
REWRITE FILE(U) FROM(RECORD4);
CLOSE FILE(U) ENV(LOCK);
END ADDSTATION;

DELSTATION:PROC(STA#);
DCL STA# PIC'9999',
      FLAG BIT(1);
OPEN FILE(U) UPDATE;
READ FILE(U) INTO(RECORD4) INDEX(4);
FLAG='1'B;
DO I=1 TO 10 WHILE(FLAG);
  IF STATIONS(I)=STA# THEN DO;
    #OFSTATIONS=#OFSTATIONS-1;
    STATIONS(I)=0;
    FLAG='0'B;
  END;
END;
REWRITE FILE(U) FROM(RECORD4);
CLOSE FILE(U) ENV(LOCK);
END DELSTATION;

END SYSTEM;

```

```

/*          - - - - -
/*          R E X / R E G I S T R A T I O N
/*          - - - - -
REGISTRATION:PROC OPTIONS(MAIN);

DCL CANDIDATE    FILE RECORD ENV(KIND='DISK',FILETYPE=7),
  SORTEDCAND    FILE RECORD ENV(KIND='DISK',FILETYPE=7),
  BUILDING      FILE RECORD ENV(KIND='DISK',MAXRECSIZE=261),
  ( ROOM,ROOM1,ROOM2,ROOM3,ROOM4 )
    FILE RECORD ENV(KIND='DISK',MAXRECSIZE=120),
  ( STAFF,STAFF1,STAFF2,STAFF3,STAFF4 )
    FILE RECORD ENV(KIND='DISK',MAXRECSIZE=79),
  STAFFINF      FILE RECORD ENV(KIND='DISK',MAXRECSIZE=63);

DCL SWROOM(4)    FILE INIT(ROOM1,ROOM2,ROOM3,ROOM4),
  SWSTAFF(4)     FILE INIT(STAFF1,STAFF2,STAFF3,STAFF4);

DCL U FILE RECORD ENV(KIND='DISK',MAXRECSIZE=180,AREAS=5,
  AREASIZE=1);

DCL 1 RECORD0,
  2 SYSID        CHAR(5),
  2 SYSPASS      CHAR(5),
  2 SYSNAME      CHAR(54);
DCL 1 RECORD1,
  2 #OFCENTERS   PIC'99',
  2 #OFSESSIONS  PIC'9',
  2 SESSIONS(4),
  3 #OFAREAS     PIC'99',
  3 #OFSUBAREAS  PIC'9',
  3 EXAMDATE     CHAR(8),
  3 EXAMHOUR     CHAR(5),
  2 TOTALAREA    PIC'99',
  2 MAX#OFAREAS  PIC'99',
  2 MAX#OFSUBAREAS PIC'9',
  2 SIZEOFOPTIONAL PIC'999';
DCL 1 RECORD2,
  2 CENTERS(90)  PIC'99';
DCL 1 RECORD3,
  2 EXPECTED     PIC'99999',
  2 ENTERED      PIC'99999';
DCL 1 RECORD4,
  2 #OFSTATIONS  PIC'99',
  2 STATIONS(10) PIC'9999';

ON RECORD(U);

GETUSERPARAMS:PROC(SYSID);
  DCL SYSID CHAR(*);
  TITLE(U)=SYSID !! '/PARAMETERS';
  OPEN FILE(U) INPUT;
  READ FILE(U) INTO(RECORD0);
  READ FILE(U) INTO(RECORD1);
  READ FILE(U) INTO(RECORD2);
  READ FILE(U) INTO(RECORD3);
  READ FILE(U) INTO(RECORD4);
  CLOSE FILE(U) ENV(LOCK);

```

END GETUSERPARAMS;

/\*  
/\* LAYOUT OF CANDIDATE FILE \*/  
/\*

DCL 1 CANDIDATE\_REC CONTROLLED,  
2 APPLICATION# PIC'(6)9',  
2 NAME CHAR(12),  
2 SURNAME CHAR(14),  
2 FATHER CHAR(12),  
2 SEX PIC'9',  
2 BIRTH PIC'(6)9',  
2 CENTER PIC'99',  
2 AREA(#OFSESSIONS) PIC'99',  
2 ADDRESS,  
3 LINE1 CHAR(23),  
3 LINE2 CHAR(23),  
3 DISTRICT CHAR(12),  
3 PROVINCE PIC'99',  
2 REG\_VALIDITY PIC'9',  
2 EXAM\_VALIDITY PIC'9',  
2 SESSION(#OFSESSIONS),  
3 ROOM# PIC'(5)9',  
3 ROOM@ PIC'(4)9',  
3 SEAT# PIC'999',  
3 ARCHIVE#,  
4 BOX# PIC'99',  
4 BOXSEQ# PIC'999',  
3 ATTENDANCE PIC'9',  
3 SUBAREA(MAX#OFSUBAREAS),  
4 CORRECT PIC'999',  
4 INCORRECT PIC'999',  
4 RAW PIC'-999.V999',  
4 STANDARD PIC'-999.V999',  
3 WEIGHTED PIC'-999.V999',  
2 OPTIONAL CHAR(SIZEOFOPTIONAL);

/\*  
/\* LAYOUT OF BUILDING FILE \*/  
/\* MAXRECSIZE=261 BLOCKSIZE=261 FRAMESIZE=8 \*/  
/\*

DCL 1 BUILDING\_REC,  
3 BUILDING# PIC'99999',  
3 BUILDING\_NAME,  
4 LINE1 CHAR(30),  
4 LINE2 CHAR(30),  
3 SESSION(4),  
4 @OFBUILDING\_DIRECTOR PIC'9999',  
4 @OFASST\_DIRECTOR PIC'9999',  
4 @OFBUILDING\_MANAGER PIC'9999',  
4 @OFSTANDBY(6) PIC'9999',  
4 #OFCANDIDATES PIC'9999',  
4 #OFRROOM\_DIRECTOR PIC'9999',  
4 #OFSUPERVISOR PIC'9999',  
4 USAGE\_FLAG PIC'9' ;

/\*

```

/*                                 LAYOUT OF ROOM FILE                                 */
/*             MAXRECSIZE=120 BLOCKSIZE=120 FRAMESIZE=8                             */
/******%*****%*****%*****%*****%*****%*****%*****%*****%*****%*****%*****%*****%*****%*****%*/
DCL 1 ROOM_REC,
      2 CENTER                    PIC'99',
      2 ROOM#                      PIC'99999',
      2 ROOM@                      PIC'9999',
      2 ROOM_NAME,
          3 LINE1                   CHAR(30),
          3 LINE2                   CHAR(30),
      2 AREA                       PIC'99',
      2 BUILDING#                  PIC'99999',
      2 BUILDING@                  PIC'9999',
      2 ROOM_CAPACITY              PIC'999',
      2 #OFASSIGNED               PIC'999',
      2 ROOM_PRIORITY              PIC'9',
      2 BUILD_PRIORITY            PIC'999',
      2 @OFROOM_DIRECTOR          PIC'9999',
      2 @OFSUPERVISOR(6)         PIC'9999';

```

```

/******%*****%*****%*****%*****%*****%*****%*****%*****%*****%*****%*****%*****%*****%*****%*/
/*                                 LAYOUT OF STAFF FILE                                 */
/*             MAXRECSIZE=79                                                       */
/******%*****%*****%*****%*****%*****%*****%*****%*****%*****%*****%*****%*****%*****%*****%*/
DCL 1 STAFF_REC,
      2 STAFF_NAME                 CHAR(12),
      2 STAFF_SURNAME             CHAR(14),
      2 STAFF_TITLE               PIC'99',
      2 DUTY                      PIC'9',
      2 CENTER                   PIC'99',
      2 BUILDING#                 PIC'99999',
      2 BUILDING@                 PIC'9999',
      2 ROOM#                     PIC'99999',
      2 ROOM@                     PIC'9999',
      2 INSTITUTION              CHAR(30);

```

```

/******%*****%*****%*****%*****%*****%*****%*****%*****%*****%*****%*****%*****%*****%*****%*/
/*                                 LAYOUT OF STAFF INFORMATION FILE                 */
/*             MAXRECSIZE=65                                                       */
/******%*****%*****%*****%*****%*****%*****%*****%*****%*****%*****%*****%*****%*****%*****%*/
DCL 1 STAFF_INF_REC,
      2 @OFSTAFF                  PIC'9999',
      2 INF_NAME                  CHAR(12),
      2 INF_SURNAME              CHAR(14),
      2 INF_TITLE                PIC'99',
      2 INF_DUTY                 PIC'9',
      2 INF_INSTITUTION          CHAR(30),
      2 INF_CENTER               PIC'99';

```

```

DCL GOREV(06) CHAR(20) INIT('BINA SINAV SORUMLUSU',
                             'BINA SINAV SOR. YRD.',
                             'BINA YONETICISI',
                             'YEDEK GOZETMEN',
                             'SALON BASKANI',
                             'GOZETMEN');

```

```

DCL UNVAN(21) CHAR(10) INIT('PROF.DR ');

```

```

' PROF          ',
' DOC_DR        ',
' DOC           ',
' YRD_DOC_DR   ',
' YRD_DOC       ',
' OGR_GOR_DR   ',
' OGR_GOR       ',
' ARS_GOR_DR   ',
' ARS_GOR       ',
' UZMAN DR     ',
' UZMAN         ',
' OKUTMAN DR   ',
' OKUTMAN       ',
' CEVIRICI     ',
' MUDUR        ',
' MUDUR YRD    ',
' M.E.MUF      ',
' OGRETMEN     ',
' MEMUR        ',
' DIGER       ');

```

```

DCL LIB1 LIBRARY(TITLE='OBJECT/REX/LIBRARY/STARTJOB. '),
LIB2 LIBRARY(TITLE='OBJECT/REX/LIBRARY/FILEMANAGER. '),
LIB3 LIBRARY(TITLE='OBJECT/REX/LIBRARY/REGISTRATION. ');

```

```

DCL STARTJOB ENTRY(CHAR(*)) RETURNS(BIT(1))
                                OPTIONS(LIBRARY=LIB1),
(COPY ENTRY(CHAR(*), CHAR(*)) RETURNS(BIT(1)),
 REMOVE ENTRY(CHAR(*)) RETURNS(BIT(1)),
 CHANGE ENTRY(CHAR(*), CHAR(*)) RETURNS(BIT(1)),
 SEARCH ENTRY(CHAR(*), CHAR(*)) RETURNS(BIT(1)) )
                                OPTIONS(LIBRARY=LIB2),
STARTREG ENTRY(FIXED, CHAR(*)) RETURNS(BIT(1))
                                OPTIONS(LIBRARY=LIB3);

```

```

% INCLUDE 'REX/SCREENS/DECLARE. ';
% INCLUDE 'REX/SCREENS/100. ';
% INCLUDE 'REX/SCREENS/101. ';
% INCLUDE 'REX/SCREENS/102. ';
% INCLUDE 'REX/SCREENS/103. ';
% INCLUDE 'REX/SCREENS/104. ';
% INCLUDE 'REX/SCREENS/105. ';
% INCLUDE 'REX/SCREENS/106. ';

```

```

DCL PROCEDURES(14) ENTRY INIT(CREATE_CANDIDATE_LAYOUT,
                                CREATE_CANDIDATE_FILE,
                                PRINT_APPLICATION_FORMS,
                                UPDATE_CANDIDATE_FILE,
                                DUPLICATE_APPLICATIONS,
                                CHECK_CANDIDATE_DATA,
                                INTER_INQUIRY_UPDATE,
                                CANDIDATE_DISTRIBUTION,
                                CANDIDATE_PRINTOUTS,
                                SORT_ROOM_FILES,
                                CREATE_STAFF_FILES,
                                PRINT_STAFF_SCHEDULE,
                                PROCESS_STAFF_DATA,

```



STAFF\_PRINTOUTS);

```
DCL 1 STRUCT105,  
  2 TOTALSIZE          CHAR(3),  
  2 ARRAY(14),  
    3 LEVEL            CHAR(1),  
    3 VARIABLE         CHAR(36),  
    3 FORMAT           CHAR(24);  
DCL 1 STRUCTSPCFY,  
  2 SPCFY(4)           CHAR(1);  
  
DCL MESSAGE            CHAR(200) VAR,  
  MESSAGE2            CHAR(200) VAR,  
  SQUASHID             CHAR(5)  VAR,  
  RJUSTID              CHAR(5),  
  COL73T080           PIC'(8)9' INIT(3000),  
  MYSTA                PIC'9999',  
  S#                   PIC'9',  
  ( CONTINUEFLAG,  
    NOTSPCFYFLAG )    BIT(1),  
  MYSELF              BUILTIN;
```

ON RECORD(R);

```
MYSTA=STATION(MYSELF) * -1;  
CALL GETUSERPARAMS(MYSTA);  
SQUASHID=EXCEPT(' ',SYSID);  
LEN=LENGTH(SQUASHID);  
IF LEN = 5 THEN RJUSTID=SQUASHID;  
  ELSE RJUSTID=COPY(' ',5-LEN)!!SQUASHID;  
ALLOCATE CANDIDATE_REC;  
TITLE(CANDIDATE)=SQUASHID !! '/CANDIDATE';  
TITLE(SORTEDCAND)=SQUASHID !! '/SORTEDCAND';  
TITLE(STAFF1)=SQUASHID !! '/STAFF/1';  
TITLE(STAFF2)=SQUASHID !! '/STAFF/2';  
TITLE(STAFF3)=SQUASHID !! '/STAFF/3';  
TITLE(STAFF4)=SQUASHID !! '/STAFF/4';  
TITLE(ROOM1)=SQUASHID !! '/ROOM/1';  
TITLE(ROOM2)=SQUASHID !! '/ROOM/2';  
TITLE(ROOM3)=SQUASHID !! '/ROOM/3';  
TITLE(ROOM4)=SQUASHID !! '/ROOM/4';  
TITLE(BUILDING)=SQUASHID !! '/BUILDING';  
  
CONTINUEFLAG='1'B;  
DO WHILE(CONTINUEFLAG);  
  NOTSPCFYFLAG='1'B;  
  DO WHILE(NOTSPCFYFLAG);  
    NOTSPCFYFLAG='0'B;  
    WRITE FILE(R) FROM(SCREEN100);  
    READ FILE(R) INTO(STRUCTSPCFY);  
    IF SPCFY(1) = ESC THEN  
      IF ( SPCFY(3) = POS(16) ) &  
        ( SPCFY(4) = POS(4)  ! SPCFY(4) = POS(5)  !  
          SPCFY(4) = POS(6)  ! SPCFY(4) = POS(7)  !  
          SPCFY(4) = POS(8)  ! SPCFY(4) = POS(9)  !  
          SPCFY(4) = POS(10) ! SPCFY(4) = POS(11) !  
          SPCFY(4) = POS(12) ! SPCFY(4) = POS(14) !
```



```

        SPCFY(4) = POS(15) ! SPCFY(4) = POS(16) !
        SPCFY(4) = POS(17) ! SPCFY(4) = POS(18) )
    THEN DO;
        TYPE=INDEX(ASCII,SPCFY(4))-3;
        IF TYPE > 9 THEN TYPE=TYPE-1;
    END; ELSE
        IF SPCFY(3) = POS(79) & SPCFY(4) = POS(22) THEN
            CONTINUEFLAG='0'B;
        ELSE DO;
            SUBSTR(SCREEN100,1842,80)=BLINK!!
            'Gecersiz konumda SPCFY tusuna bastiniz.';
            NOTSPCFYFLAG='1'B;
        END;
    ELSE DO;
        SUBSTR(SCREEN100,1842,80)=BLINK!!
        'XMIT yerine SPCFY tusunu kullaniniz.';
        NOTSPCFYFLAG='1'B;
    END;
END;
SUBSTR(SCREEN100,1842,80)=' ';
IF CONTINUEFLAG THEN DO;
    CALL PROCEDURES(TYPE);
    MESSAGE=CLS!!ESC!!'"2-!!BLINK!!
        MESSAGE !! ' SONA ERDI. XMIT''E BASINIZ.' !!
        ESC!!'"o7'!!DC2ETX;
    WRITE FILE(R) FROM(MESSAGE);
    READ FILE(R) INTO(MESSAGE);
END;
END;

CREATE_CANDIDATE_LAYOUT:PROC;
    DCL L FILE RECORD ENV(KIND='DISK',MAXRECSIZE=15,
        BLOCKSIZE=420,FRAMESIZE=48);
    DCL S CHAR(90);
    CALL COPY('LAYOUT/CANDIDATE',
        SQUASHID!!'/LAYOUT/CANDIDATE');
    WRITE FILE(R) FROM(SCREEN105);
    READ FILE(R) INTO(STRUCT105);
    TITLE(L)=SQUASHID !! '/LAYOUT/CANDIDATE';
    OPEN FILE(L) UPDATE;
    READ FILE(L) INTO(S) INDEX(4);
    SUBSTR(S,20,24)=' ' !! (23) ' ' ;
    REWRITE FILE(L) FROM(S);
    READ FILE(L) INTO(S) INDEX(12);
    SUBSTR(S,14,12)=#OFSESSIONS !! ' )' !! (10) ' ' ;
    REWRITE FILE(L) FROM(S);
    READ FILE(L) INTO(S) INDEX(20);
    SUBSTR(S,17,13)=#OFSESSIONS !! ' ),' !! (10) ' ' ;
    REWRITE FILE(L) FROM(S);
    READ FILE(L) INTO(S) INDEX(28);
    SUBSTR(S,19,16)=MAX#OFSUBAREAS !! ' ),' !! (13) ' ' ;
    REWRITE FILE(L) FROM(S);
    IF TOTALSIZE ^= 0 THEN DO;
        EOF=LASTRECORD(L);
        DO I=1 TO 14 WHILE(LEVEL(I) ^= '*');
            S=(6) ' !!LEVEL(I)!!' !!VARIABLE(I)!!
            FORMAT(I)!!',';
        END;
    END;

```

```

        SUBSTR(S,73,8)=COL73T080;
        COL73T080=COL73T080+100;
        WRITE FILE(L) FROM(S) INDEX(EOF);
        EOF=EOF+1;
    END;
    SUBSTR(S,69,1)='';
    WRITE FILE(L) FROM(S) INDEX(EOF-1);
END; ELSE DO;
    READ FILE(L) INTO(S) INDEX(34);
    SUBSTR(S,38,16)='1)'; !! (13) ' ';
    REWRITE FILE(L) FROM(S);
    TOTALSIZE=1;
END;
CLOSE FILE(L) ENV(LOCK);
TITLE(U)=SQUASHID !! '/PARAMETERS';
OPEN FILE(U) UPDATE;
READ FILE(U) INTO(RECORD1) INDEX(1);
SIZEOFOPTIONAL=TOTALSIZE;
REWRITE FILE(U) FROM(RECORD1);
CLOSE FILE(U) ENV(LOCK);
MESSAGE='CREATE-CANDIDATE-LAYOUT';
END;

CREATE_CANDIDATE_FILE:PROC;
    DCL P FILE RECORD ENV(KIND='DISK',MAXRECSIZE=1080);
    DCL T CHAR(1080);
    DCL (RCD,BLK) PIC'(5)9';
    TITLE(P)=SQUASHID !! '/CREATE/CANDIDATE/PARAMS';
    IF ~ RESIDENT(P) THEN DO;
        CALL COPY('REX/CREATE/CANDIDATE/PARAMS',
                SQUASHID!!'/CREATE/CANDIDATE/PARAMS');
        OPEN FILE(P) UPDATE;
        READ FILE(P) INTO(T) INDEX(0);
        SUBSTR(T,1,5)=RJUSTID;
        REWRITE FILE(P) FROM(T);
        READ FILE(P) INTO(T) INDEX(3);
        SUBSTR(T,1,5)=RJUSTID;
        SUBSTR(T,73,6)=EXPECTED;
        REWRITE FILE(P) FROM(T);
        READ FILE(P) INTO(T) INDEX(4);
        RCD=115+#OFSESSIONS*(28+MAX#OF SUBAREAS*22)+
            SIZEOFOPTIONAL;
        BLK=RCD*10;
        SUBSTR(T,1,5)=RCD;
        SUBSTR(T,6,5)=BLK;
        REWRITE FILE(P) FROM(T);
        CLOSE FILE(P) ENV(LOCK);
    END;
    CLOSE FILE(R);
    CALL STARTREG(1,SQUASHID!!'/CREATE/CANDIDATE/PARAMS');
    MESSAGE='CREATE-CANDIDATE-FILE';
END;

PRINT_APPLICATION_FORMS:PROC;
    CALL STARTREG(2,SQUASHID!!'/PRINT/APPLICATION/PARAMS');
    MESSAGE='PRINT-APPLICATION-FORMS';
END;

```

```

UPDATE_CANDIDATE_FILE:PROC;
  DCL P FILE RECORD ENV(KIND='DISK',MAXRECSIZE=1080);
  DCL T CHAR(1080);
  DCL (RCD,BLK) PIC'(5)9';
  TITLE(P)=SQUASHID !! '/UPDATE/CANDIDATE/PARAMS';
  IF ~ RESIDENT(P) THEN DO;
    CALL COPY('REX/UPDATE/CANDIDATE/PARAMS',
             SQUASHID!!'/UPDATE/CANDIDATE/PARAMS');
    OPEN FILE(P) UPDATE;
    READ FILE(P) INTO(T) INDEX(0);
    SUBSTR(T,1,5)=RJUSTID;
    REWRITE FILE(P) FROM(T);
    READ FILE(P) INTO(T) INDEX(3);
    SUBSTR(T,1,5)=RJUSTID;
    SUBSTR(T,115,5)=RJUSTID;
    REWRITE FILE(P) FROM(T);
    READ FILE(P) INTO(T) INDEX(4);
    RCD=115+#OFSESSIONS*(28+MAX#OFSUBAREAS*22)+
      SIZEOFOPTIONAL;
    BLK=RCD*10;
    SUBSTR(T,1,5)=RCD;
    SUBSTR(T,6,5)=BLK;
    REWRITE FILE(P) FROM(T);
    READ FILE(P) INTO(T) INDEX(8);
    SUBSTR(T,291,5)=RJUSTID;
    REWRITE FILE(P) FROM(T);
    CLOSE FILE(P) ENV(LOCK);
  END;
  CLOSE FILE(R);
  CALL STARTREG(1,SQUASHID!!'/UPDATE/CANDIDATE/PARAMS');
  MESSAGE='UPDATE-CANDIDATE-FILE';
END;

```

```

DUPLICATE_APPLICATIONS:PROC;
  DCL STR1 CHAR(31)
    INIT((1)' _ABCDEFGHIJKLMNOPQRSTUVWXYZ12');
  DCL STR2 CHAR(31)
    INIT((1)' _ABC@DEFG_HI[IJKLMNO\PRS]TU~VYZ');
  DCL (PNAM,PFAT) CHAR(12) INIT((12)'*'),
    PSUR CHAR(14) INIT((14)'*'),
    (PAPP,PBIR) PIC'(6)9' INIT(0);
  DCL (CNT1,CNT2) FIXED(6);
  OPEN FILE(SYSPRINT) ENV(KIND='PRINTER',MAXRECSIZE=132)
    TAB(1,15,30,45,60);
  PUT PAGE LIST(' ',' ','CIFT BASVURU LISTESI');
  PUT SKIP(2) LIST
    ('BAS.NO','SOYADI','ADI','BABA ADI','D.TARIHI');
  PUT SKIP LIST('====','=====','=====','
    =====','=====');
  SORT CANDIDATE_REC ON ASCENDING(SURNAME,NAME)
    INPUT(INPROC) OUTPUT(OUTPROC);
  CLOSE FILE(SYSPRINT);
  INPROC:PROC(A) RETURNS(BIT(1)) OPTIONS(SORTINPUT);
  DCL A CHAR(*);
  ON ENDFILE(CANDIDATE) GO TO XIT;
CONT:

```

```

READ FILE(CANDIDATE) INTO(CANDIDATE_REC);
IF REG_VALIDITY ^= 1 THEN GO TO CONT;
NAME=TRANSLATE(NAME,STR1,STR2);
SURNAME=TRANSLATE(SURNAME,STR1,STR2);
CNT1=CNT1+1;
RETURN('0'B);
XIT:
RETURN('1'B);
END INPROC;
OUTPROC:PROC(B,A) OPTIONS(SORTOUTPUT);
DCL B BIT(1),A CHAR(*);
CNT2=CNT2+1;
IF CNT2 > CNT1 THEN DO;
IF LAST = 1 THEN
PUT SKIP LIST(PAPP,PSUR,PNAM,PFAT,PBIR);
B='1'B;
RETURN;
END; ELSE DO;
NAME=TRANSLATE(NAME,STR2,STR1);
SURNAME=TRANSLATE(SURNAME,STR2,STR1);
IF NAME = PNAM & SURNAME = PSUR THEN DO;
PUT SKIP LIST(PAPP,PSUR,PNAM,PFAT,PBIR);
LAST=1;
END; ELSE
IF LAST = 1 THEN DO;
PUT SKIP LIST(PAPP,PSUR,PNAM,PFAT,PBIR);
PUT SKIP LIST(' ');
END;
PAPP=APPLICATION#;
PNAM=NAME; PSUR=SURNAME;
PFAT=FATHER; PBIR=BIRTH;
B='0'B;
RETURN;
END;
END OUTPROC;
MESSAGE='DUPLICATE-APPLICATIONS';
END;

CHECK_CANDIDATE_DATA:PROC;
DCL (EOF,FLAG) BIT(1);
ON ENDFILE(CANDIDATE) EOF='1'B;
OPEN FILE(SYSPRINT) ENV(KIND='PRINTER');
PUT PAGE EDIT('SON DENETIM LISTESI') (X(10),A);
OPEN FILE(CANDIDATE) INPUT;
READ FILE(CANDIDATE) INTO(CANDIDATE_REC);
EOF='0'B;
DO WHILE(~ EOF);
FLAG='1'B;
DO S#=1 TO #OFCENTERS WHILE(FLAG);
IF CANDIDATE_REC.CENTER = CENTERS(S#) THEN
FLAG='0'B;
END;
IF FLAG = '1'B THEN
PUT SKIP(2) EDIT(APPLICATION#,
': SINAV MERKEZI HATALI')
(P'(6)9',A);
DO S#=1 TO #OFSESSIONS;

```

```

                IF CANDIDATE_REC.AREA(S#) > #OFAREAS(S#) THEN
                    PUT SKIP(2) EDIT(APPLICATION#,
                                     ': ALAN KODU HATALI')
                                     (P'(6)9',A);
                END;
                READ FILE(CANDIDATE) INTO(CANDIDATE_REC);
            END;
            CLOSE FILE(CANDIDATE);
            CLOSE FILE(SYSPRINT);
            MESSAGE='CHECK-CANDIDATE-DATA';
        END;

INTER_INQUIRY_UPDATE:PROC;
DCL 1 STRUCT103,
    2 APPLICATION#    PIC'(6)9',
    2 NAME            CHAR(12),
    2 SURNAME        CHAR(14),
    2 FATHER         CHAR(12),
    2 SEX            PIC'9',
    2 BIRTH          PIC'(6)9',
    2 CENTER         PIC'99',
    2 MAXAREA(4)     PIC'99',
    2 ADDRESS,
    3 LINE1          CHAR(23),
    3 LINE2          CHAR(23),
    3 DISTRICT       CHAR(12),
    3 PROVINCE       PIC'99',
    2 REG_VALIDITY   PIC'9',
    2 EXAM_VALIDITY  PIC'9',
    2 TASK103        PIC'9',
    2 EXIT103        CHAR(1);
DCL 1 STRUCT104,
    2 ROOM#          PIC'(5)9',
    2 ROOM@          PIC'(4)9',
    2 SEAT#          PIC'999',
    2 ARCHIVE#,
    3 BOX#           PIC'99',
    3 BOXSEQ#        PIC'999',
    2 ATTENDANCE     PIC'9',
    2 WEIGHTED       PIC'-999.V999',
    2 MAXSUBAREA(5),
    3 UNICORRECT     PIC'999',
    3 UNIINCORRECT  PIC'999',
    3 UNIRAW         PIC'-999.V999',
    3 UNISTANDARD   PIC'-999.V999',
    2 TASK104        PIC'9',
    2 EXIT104        CHAR(1);

DCL APP# PIC'(6)9',
    KEY PIC'(5)9' DEF APP# POS(1),
    N(6) PIC'9' DEF APP# POS(1),
    CONFIGURE CHAR(18);

CONFIGURE=ESC!!'RH00A0010Z'!!ESC!!'RC';
WRITE FILE(R) FROM(CONFIGURE);
READ FILE(R) INTO(MESSAGE);
IF SUBSTR(MESSAGE,1,6) = '*****' THEN DO;

```

```

        DISPLAY('CONFIGURATION ERROR'); STOP; END;

OPEN FILE(CANDIDATE) INPUT;
L1:WRITE FILE(R) FROM(DC2ETX);
    WRITE FILE(R) FROM(SCREEN103);
    READ FILE(R) INTO(STRUCT103);
    IF TASK103 = 1 THEN GO TO L3;
    APP#=STRUCT103.APPLICATION#;
L2:IF N(6) = 9-MOD(N(1)*2+N(2)*3+N(3)*4+N(4)*5+N(5)*6,9)
    THEN GO TO L1;
    ELSE DO;
        READ FILE(CANDIDATE) INTO(CANDIDATE_REC) INDEX(KEY-1);
        STRUCT103=CANDIDATE_REC, BY NAME;
        MAXAREA=0;
        DO S#=1 TO #OFSESSIONS;
            MAXAREA(S#)=CANDIDATE_REC.AREA(S#);
        END;
        TASK103=2; EXIT103=ETX;
        WRITE FILE(R) FROM(DC2ETX);
        WRITE FILE(R) FROM(SCREEN103);
        WRITE FILE(R) FROM(STRUCT103);
        READ FILE(R) INTO(STRUCT103);
        IF TASK103 = 1 THEN GO TO L3; ELSE
        IF TASK103 = 2 THEN
            DO S#=1 TO #OFSESSIONS;
                STRUCT104=CANDIDATE_REC.SESSION(S#), BY NAME;
                UNICORRECT,UNIINCORRECT=0;
                UNIRAW,UNISTANDARD=0;
                DO I=1 TO #OFSUBAREAS(S#);
                    UNICORRECT=CORRECT(S#,I);
                    UNIINCORRECT=INCORRECT(S#,I);
                    UNIRAW=RAW(S#,I);
                    UNISTANDARD=STANDARD(S#,I);
                END;
                TASK104=3; EXIT104=ETX;
                WRITE FILE(R) FROM(DC2ETX);
                SUBSTR(SCREEN104,285,1)=S#;
                WRITE FILE(R) FROM(SCREEN104);
                WRITE FILE(R) FROM(STRUCT104);
                READ FILE(R) INTO(STRUCT104);
                IF TASK104 = 1 THEN GO TO L3; ELSE
                IF TASK104 = 2 THEN GO TO L1; ELSE;
            END;
        ELSE GO TO L2;
    END;
GO TO L1;
L3:
CLOSE FILE(CANDIDATE);
CONFIGURE=ESC!!'RH00A00100'!!ESC!!'RC';
WRITE FILE(R) FROM(CONFIGURE);
READ FILE(R) INTO(MESSAGE);
MESSAGE='INTER-INQUIRY-UPDATE';
END;

CANDIDATE_DISTRIBUTION:PROC;
CALL STARTREG(3,SQUASHID!!'/GET/DISTRIBUTION/PARAMS');
MESSAGE='CANDIDATE-DISTRIBUTION';

```

END;

CANDIDATE\_PRINTOUTS:PROC;

```
DCL PROCEDURES(8) ENTRY INIT(PRINT_ENTRANCE,  
                              PRINT_RESULTS,  
                              PRINT_SORTED_APP#,  
                              PRINT_SORTED_NAME,  
                              PRINT_SORTED_SCORE1,  
                              PRINT_SORTED_SCORE2,  
                              PRINT_SORTED_SCORE3,  
                              PRINT_SORTED_SCORE4);
```

```
DCL ( CONTINUEFLAG,  
      NOTSPCFYFLAG ) BIT(1);
```

```
CONTINUEFLAG='1'B;  
DO WHILE(CONTINUEFLAG);  
  NOTSPCFYFLAG='1'B;  
  DO WHILE(NOTSPCFYFLAG);  
    NOTSPCFYFLAG='0'B;  
    WRITE FILE(R) FROM(SCREEN101);  
    READ FILE(R) INTO(STRUCTSPCFY);  
    IF SPCFY(1) = ESC THEN  
      IF ( SPCFY(3) = POS(16) ) &  
        ( SPCFY(4) = POS(4) ! SPCFY(4) = POS(6) !  
          SPCFY(4) = POS(10) ! SPCFY(4) = POS(11) !  
          SPCFY(4) = POS(15) ! SPCFY(4) = POS(16) !  
          SPCFY(4) = POS(17) ! SPCFY(4) = POS(18) )  
        THEN DO;  
          TYPE=INDEX(ASCII,SPCFY(4))-3;  
          IF TYPE = 3 THEN TYPE=2;  
          IF TYPE > 6 THEN TYPE=TYPE-4;  
          IF TYPE > 7 THEN TYPE=TYPE-3;  
        END; ELSE  
          IF SPCFY(3) = POS(79) & SPCFY(4) = POS(22)  
            THEN CONTINUEFLAG='0'B;  
          ELSE DO;  
            SUBSTR(SCREEN101,1842,80)=BLINK!!  
            'Gecersiz konumda SPCFY tusuna bastiniz.';  
            NOTSPCFYFLAG='1'B;  
          END;  
        ELSE DO;  
          SUBSTR(SCREEN101,1842,80)=BLINK!!  
          'XMIT yerine SPCFY tusunu kullaniniz.';  
          NOTSPCFYFLAG='1'B;  
        END;  
      END;  
    SUBSTR(SCREEN101,1842,80)=' ';  
    IF CONTINUEFLAG THEN DO;  
      IF TYPE > 2 THEN DO;  
        DCL FA FIXED(5);  
        FA=115+#OFSESSIONS*(28+MAX#OFSUBAREAS*22)+  
          SIZEOFOPTIONAL;  
        MAXRECSIZE(SORTEDCAND)=FA;  
        BLOCKSIZE(SORTEDCAND)=FA;  
        FRAMESIZE(SORTEDCAND)=8;
```



```

END;
CALL PROCEDURES(TYPE);
MESSAGE2=CLS!!ESC!!"2-!!BLINK!!
      MESSAGE2!!' SONA ERDI. XMIT'E BASINIZ.!!
      ESC!!"67!!DC2ETX;
WRITE FILE(R) FROM(MESSAGE2);
READ FILE(R) INTO(MESSAGE2);
END;
END;
MESSAGE='CANDIDATE-PRINTOUTS';
END;

PRINT_ENTRANCE:PROC;
DCL EOF BIT(1);
ON ENDFILE(CANDIDATE) EOF='1'B;
ON ENDPAGE(SYSPRINT);
OPEN FILE(SYSPRINT) ENV(KIND='PRINTER',MAXRECSIZE=132);
OPEN FILE(BUILDING) INPUT;
DO S#=1 TO #OFSESSIONS;
  OPEN FILE(CANDIDATE) INPUT;
  OPEN FILE(SWROOM(S#)) INPUT;
  READ FILE(CANDIDATE) INTO(CANDIDATE_REC);
  EOF='0'B;
  DO WHILE(~ EOF);
    IF REG_VALIDITY = 1 THEN DO;
      READ FILE(SWROOM(S#)) INTO(ROOM_REC)
        INDEX(CANDIDATE_REC.ROOM@(S#)-1);
      READ FILE(BUILDING) INTO(BUILDING_REC)
        INDEX(ROOM_REC.BUILDING@);
      PUT SKIP(8) EDIT(
        SYSNAME,' SINAVA GIRIS BELGESI',
        APPLICATION#,SURNAME!!' !!!NAME,' ',
        ILLER(CANDIDATE_REC.CENTER),
        NAME!!' !!!SURNAME,
        BUILDING_NAME.LINE1,ADDRESS.LINE1,
        BUILDING_NAME.LINE2,ADDRESS.LINE2,
        ROOM_NAME.LINE1,(11)' !!!DISTRICT,
        ROOM_NAME.LINE2,(11)' !!!
        ILLER(PROVINCE),EXAMDATE(S#),
        EXAMHOUR(S#),CANDIDATE_REC.ROOM@(S#),
        CANDIDATE_REC.SEAT@(S#)
        (2(COL(22),A,SKIP),SKIP,COL(15),P'(6)9',
        SKIP,2(COL(15),A,SKIP),SKIP(2),
        5(COL(15),A,COL(71),A,SKIP),SKIP(3),
        COL(7),A,COL(20),A,COL(29),P'(5)9',
        COL(40),P'999');
    END;
    READ FILE(CANDIDATE) INTO(CANDIDATE_REC);
  END;
  CLOSE FILE(SWROOM(S#));
  CLOSE FILE(CANDIDATE);
END;
CLOSE FILE(BUILDING);
CLOSE FILE(SYSPRINT);
MESSAGE2='PRINT-ENTRANCE';
END;

```



```

PRINT_RESULTS:PROC;
  DCL SCORE(4) CHAR(25) INIT((4)(25)' '),
  EOF BIT(1);
  ON ENDFILE(CANDIDATE) EOF='1'B;
  ON ENDPAGE(SYSPRINT);
  OPEN FILE(SYSPRINT) ENV(KIND='PRINTER',MAXRECSIZE=132);
  OPEN FILE(CANDIDATE) INPUT;
  READ FILE(CANDIDATE) INTO(CANDIDATE_REC);
  EOF='0'B;
  DO WHILE(~ EOF);
    IF REG_VALIDITY = 1 THEN DO;
      DO S#=1 TO #OFSESSIONS;
        PUT STRING(SCORE(S#)) EDIT(S#,'. OTURUM PUANI:',
          WEIGHTED(S#)) (P'9',A,P'-ZZ9.V999');
      END;
      PUT SKIP(8) EDIT(SYSNAME,' SINAV SONUC BELGESI',
        APPLICATION#,SURNAME!!' !!!NAME,
        ' ',' ',NAME!!' !!!SURNAME,
        SCORE(1),ADDRESS.LINE1,
        ' ',ADDRESS.LINE2,
        SCORE(2),DISTRICT,
        ' ',(11)' !!!ILLER(PROVINCE),
        SCORE(3),SCORE(4))
        (2(COL(22),A,SKIP),SKIP,COL(15),P'(6)9',
        SKIP,2(COL(15),A,SKIP),SKIP(2),
        5(COL(15),A,COL(71),A,SKIP),
        2(SKIP,COL(15),A,SKIP));
    END;
    READ FILE(CANDIDATE) INTO(CANDIDATE_REC);
  END;
  CLOSE FILE(CANDIDATE);
  CLOSE FILE(SYSPRINT);
  MESSAGE2='PRINT-RESULTS';
END;

PRINT_SORTED_APP#:PROC;
  CALL COPY(SQUASHID !! '/CANDIDATE',
    SQUASHID !! '/SORTEDCAND');
  CALL PRINT_SORTED_FILE;
  MESSAGE2='PRINT-SORTED-APP#';
END;

PRINT_SORTED_NAME:PROC;
  DCL STR1 CHAR(31)
  INIT((1)' _ABCDEFGHIJKLMNPNRSTUVWXYZ012');
  DCL STR2 CHAR(31)
  INIT((1)' _ABC@DEFG_HI[JKLMNO\PRS]TU~VYZ');
  DCL (CNT1,CNT2) FIXED(6);
  SORT CANDIDATE_REC ON ASCENDING(SURNAME,NAME)
    INPUT(INPROC) OUTPUT(OUTPROC);
  CLOSE FILE(SORTEDCAND) ENV(LOCK);
  CALL PRINT_SORTED_FILE;
  INPROC:PROC(A) RETURNS(BIT(1)) OPTIONS(SORTINPUT);
  DCL A CHAR(*);
  ON ENDFILE(CANDIDATE) GO TO XIT;
CONT:
  READ FILE(CANDIDATE) INTO(CANDIDATE_REC);

```

```

        IF REG_VALIDITY = 1 THEN GO TO CONT;
        NAME=TRANSLATE(NAME,STR1,STR2);
        SURNAME=TRANSLATE(SURNAME,STR1,STR2);
        CNT1=CNT1+1;
        RETURN('0'B);
XIT:
        RETURN('1'B);
END INPROC;
OUTPROC:PROC(B,A) OPTIONS(SORTOUTPUT);
        DCL B BIT(1),A CHAR(*);
        CNT2=CNT2+1;
        IF CNT2 > CNT1 THEN DO;
                B='1'B;
                RETURN;
        END; ELSE DO;
                NAME=TRANSLATE(NAME,STR2,STR1);
                SURNAME=TRANSLATE(SURNAME,STR2,STR1);
                WRITE FILE(SORTEDCAND) FROM(CANDIDATE_REC);
                B='0'B;
                RETURN;
        END;
END OUTPROC;
        MESSAGE2='PRINT-SORTED-NAME';
END;

PRINT_SORTED_SCORE1:PROC;
        SORT CANDIDATE_REC ON ASCENDING(WEIGHTED(1))
                USING FILE(CANDIDATE) GIVING FILE(SORTEDCAND);
        CALL PRINT_SORTED_FILE;
        MESSAGE2='PRINT-SORTED-SCORE1';
END;

PRINT_SORTED_SCORE2:PROC;
        SORT CANDIDATE_REC ON ASCENDING(WEIGHTED(2))
                USING FILE(CANDIDATE) GIVING FILE(SORTEDCAND);
        CALL PRINT_SORTED_FILE;
        MESSAGE2='PRINT-SORTED-SCORE2';
END;

PRINT_SORTED_SCORE3:PROC;
        SORT CANDIDATE_REC ON ASCENDING(WEIGHTED(3))
                USING FILE(CANDIDATE) GIVING FILE(SORTEDCAND);
        CALL PRINT_SORTED_FILE;
        MESSAGE2='PRINT-SORTED-SCORE3';
END;

PRINT_SORTED_SCORE4:PROC;
        SORT CANDIDATE_REC ON ASCENDING(WEIGHTED(4))
                USING FILE(CANDIDATE) GIVING FILE(SORTEDCAND);
        CALL PRINT_SORTED_FILE;
        MESSAGE2='PRINT-SORTED-SCORE4';
END;

PRINT_SORTED_FILE:PROC;
        DCL EOF BIT(1);
        ON ENDFILE(SORTEDCAND) BEGIN;
                CLOSE FILE(SORTEDCAND) ENV(LOCK);

```

```

CALL REMOVE(SQUASHID !! '/SORTEDCAND');
CLOSE FILE(SYSPRINT);
EOF='1'B; END;
ON ENDPAGE(SYSPRINT)
  PUT PAGE EDIT(SYSNAME,'SIRALI ADAY LISTESI',
    'BAS.NO', 'ADI', 'SOYADI',
    (I,'. OTURUM PUANI' DO I=1 TO #OFSESSIONS),
    (6)'=',(12)'=',(14)'=',
    ((15)'=' DO I=1 TO #OFSESSIONS))
    (COL(1),A,A,SKIP(2),COL(1),A,COL(9),A,
    COL(23),A,(#OFSESSIONS)(COL(22+I*17),P'9',A),
    SKIP,COL(1),A,COL(9),A,COL(23),A,
    (#OFSESSIONS)(COL(22+I*17),A));
OPEN FILE(SYSPRINT) ENV(KIND='PRINTER',MAXRECSIZE=132,
  PAGESIZE=60);
OPEN FILE(SORTEDCAND) INPUT;
SIGNAL ENDPAGE(SYSPRINT);
EOF='0'B;
READ FILE(SORTEDCAND) INTO(CANDIDATE_REC);
DO WHILE(~ EOF);
  IF REG_VALIDITY = 1 THEN
    PUT SKIP EDIT(APPLICATION#,NAME,SURNAME,
      (WEIGHTED(S#) DO S#=1 TO #OFSESSIONS))
      (COL(1),P'(6)9',2(X(2),A),X(5),(#OFSESSIONS)
      (P'-ZZ9.V999',X(9)));
    READ FILE(SORTEDCAND) INTO(CANDIDATE_REC);
  END;
END;

SORT_ROOM_FILES:PROC;
WRITE FILE(R) FROM(SCREEN106);
READ FILE(R) INTO(STRUCTSPCFY);
SORTTYPE=0;
IF SPCFY(1) = ESC & SPCFY(3) = POS(22) THEN
  IF SPCFY(4) = POS(8) THEN SORTTYPE=1; ELSE
  IF SPCFY(4) = POS(10) THEN SORTTYPE=2;
IF SORTTYPE > 0 THEN
  DO S#=1 TO #OFSESSIONS;
  MESSAGE=CLS!!ESC!!'"2-'!!BLINK!!
    S#!!'. OTURUMUN KUTUGU SIRALANIYOR.'!!
    ESC!!'"07'!!DC2ETX;
  WRITE FILE(R) FROM(MESSAGE);
  IF SORTTYPE = 1 THEN
    SORT ROOM_REC ON ASCENDING(ROOM_REC.ROOM#)
      USING FILE(SWROOM(S#))
      GIVING FILE(SWROOM(S#));
  ELSE
    SORT ROOM_REC ON ASCENDING
      (ROOM_REC.BUILDING#,ROOM_REC.ROOM#)
      USING FILE(SWROOM(S#))
      GIVING FILE(SWROOM(S#));
  END;
MESSAGE='SORT-ROOM-FILES';
END;

CREATE_STAFF_FILES:PROC;
DCL PBUILDING PIC'(5)9',

```

```

        ADR          PIC'9999',
        FLAG        BIT(1);
ON ENDFILE(ROOM) BEGIN;
    REWRITE FILE(BUILDING) FROM(BUILDING_REC);
    CLOSE FILE(ROOM) ENV(LOCK);
    CLOSE FILE(STAFF) ENV(LOCK);
    FLAG='0'B;
END;
STAFF_NAME=(12)' '; STAFF_SURNAME=(14)' ';
STAFF_REC.INSTITUTION=(30)' '; STAFF_TITLE=0;
TITLE(BUILDING)=SQUASHID !! '/BUILDING';
OPEN FILE(BUILDING) UPDATE;
DO S#=1 TO #OFSESSIONS;
    MESSAGE=CLS!!ESC!!'"2-'!!BLINK!!
           S#!! '. OTURUMUN KUTUGU YARATILİYOR.'!!
           ESC!!'"o7'!!DC2ETX;
    WRITE FILE(R) FROM(MESSAGE);
    TITLE(ROOM)=SQUASHID !! '/ROOM/' !! S#;
    OPEN FILE(ROOM) UPDATE;
    TITLE(STAFF)=SQUASHID !! '/STAFF/' !! S#;
    OPEN FILE(STAFF) OUTPUT;
    PBUILDING=0; ADR=0; FLAG='1'B;
    READ FILE(ROOM) INTO(ROOM_REC);
    DO WHILE(FLAG);
        IF ROOM_REC.BUILDING# /= PBUILDING THEN DO;
            PBUILDING=ROOM_REC.BUILDING#;
            IF ADR > 0 THEN
                REWRITE FILE(BUILDING) FROM(BUILDING_REC);
            READ FILE(BUILDING) INTO(BUILDING_REC)
                INDEX(ROOM_REC.BUILDING@);
            #OFCANDIDATES(S#)=0;
            #OFRROOM_DIRECTOR(S#)=0;
            #OFSUPERVISOR(S#)=0;
            STAFF_REC=ROOM_REC, BY NAME;
            STAFF_REC.DUTY=1;
            WRITE FILE(STAFF) FROM(STAFF_REC);
            @OFBUILDING_DIRECTOR(S#)=ADR; ADR=ADR+1;
            STAFF_REC.DUTY=2;
            WRITE FILE(STAFF) FROM(STAFF_REC);
            @OFASST_DIRECTOR(S#)=ADR; ADR=ADR+1;
            STAFF_REC.DUTY=3;
            WRITE FILE(STAFF) FROM(STAFF_REC);
            @OFBUILDING_MANAGER(S#)=ADR; ADR=ADR+1;
            DO I=1 TO 6;
                STAFF_REC.DUTY=4;
                WRITE FILE(STAFF) FROM(STAFF_REC);
                @OFSTANDBY(S#,I)=ADR; ADR=ADR+1;
            END;
        END;
        STAFF_REC=ROOM_REC, BY NAME;
        STAFF_REC.DUTY=5;
        WRITE FILE(STAFF) FROM(STAFF_REC);
        @OFRROOM_DIRECTOR=ADR; ADR=ADR+1;
        DO I=1 TO CEIL(#OFASSIGNED/25);
            STAFF_REC.DUTY=6;
            WRITE FILE(STAFF) FROM(STAFF_REC);
            @OFSUPERVISOR(I)=ADR; ADR=ADR+1;

```

```

        #OFSUPERVISOR(S#)=#OFSUPERVISOR(S#)+1;
    END;
    #OFCANDIDATES(S#)=#OFCANDIDATES(S#)+#OFASSIGNED;
    #OFRROOM_DIRECTOR(S#)=#OFRROOM_DIRECTOR(S#)+1;
    REWRITE FILE(ROOM) FROM(ROOM_REC);
    READ FILE(ROOM) INTO(ROOM_REC);
END;
END;
CLOSE FILE(BUILDING) ENV(LOCK);
MESSAGE='CREATE-STAFF-FILES';
END;

PRINT_STAFF_SCHEDULE:PROC;
    DCL ROOM_BUIL(2) CHAR(30),
        CAPACITY CHAR(10),
        STAFFCOUNT FIXED(4),
        PLACE PIC'(5)9',
        PBUIL PIC'(5)9',
        TEMP1 PIC'ZZZ9',
        TEMP2 PIC'ZZ9',
        EOF BIT(1);
    ON ENDFILE(STAFF) BEGIN;
        CLOSE FILE(STAFF); CLOSE FILE(ROOM);
        EOF='1'B; END;
    ON ENDPAGE(SYSPRINT) BEGIN;
        PUT PAGE EDIT(SYSNAME,' GOREVLI BILDIRIM CIZELGESI',
            ' (OTURUM:',S#,')', 'SALON/BINA NUMARASI,ADI',
            'KAPASITE', 'GOREV TURU', 'UNV.',
            'GOREVLININ ADI,SOYADI', 'CALISTIGI KURUM',
            (36)'-',(8)'-',(20)'-',(4)'-',(27)'-',(20)'-')
            (COL(1),A,A,A,P'9',A,SKIP(2),
            2(SKIP,COL(8),A,COL(46),A,COL(56),A,
            COL(78),A,COL(84),A,COL(113),A));
    END;
    OPEN FILE(SYSPRINT) ENV(KIND='PRINTER',MAXRECSIZE=132,
        PAGESIZE=59);
    OPEN FILE(BUILDING) INPUT;
    DO S#=1 TO #OFSESSIONS;
        MESSAGE=CLS!!ESC!!"2~!!BLINK!!
            S#!!'. OTURUMUN CIZELGESI DOKULUYOR.!!
            ESC!!"o7!!DCZETX;
        WRITE FILE(R) FROM(MESSAGE);
        STAFFCOUNT=0;
        TITLE(STAFF)=SQUASHID !! '/STAFF/' !! S#;
        OPEN FILE(STAFF) INPUT;
        TITLE(ROOM)=SQUASHID !! '/ROOM/' !! S#;
        OPEN FILE(ROOM) INPUT;
        SIGNAL ENDPAGE(SYSPRINT);
        READ FILE(STAFF) INTO(STAFF_REC);
        PBUIL=STAFF_REC.BUILDING#;
        EOF='0'B;
        DO WHILE(~ EOF);
            IF PBUIL ~ = STAFF_REC.BUILDING# THEN DO;
                SIGNAL ENDPAGE(SYSPRINT);
                PBUIL=STAFF_REC.BUILDING#;
            END;
            STAFFCOUNT=STAFFCOUNT+1;

```

```

IF STAFF_REC.DUTY > 4 THEN DO;
PLACE=STAFF_REC.ROOM#;
READ FILE(ROOM) INTO(ROOM_REC)
INDEX(STAFF_REC.ROOM@-1);

TEMP2=#OFASSIGNED;
CAPACITY=' ' !! TEMP2 !! ' ADAY';
ROOM_BUIL(1)=ROOM_NAME.LINE1;
ROOM_BUIL(2)=ROOM_NAME.LINE2;
END; ELSE DO;
PLACE=STAFF_REC.BUILDING#;
READ FILE(BUILDING) INTO(BUILDING_REC)
INDEX(STAFF_REC.BUILDING@);

TEMP1=#OFROOM_DIRECTOR(S#);
CAPACITY=TEMP1 !! ' SALON';
ROOM_BUIL(1)=BUILDING_NAME.LINE1;
ROOM_BUIL(2)=BUILDING_NAME.LINE2;
END;
PUT SKIP EDIT(STAFFCOUNT,PLACE,ROOM_BUIL(1),
CAPACITY,GOREV(STAFF_REC.DUTY),STAFF_TITLE,
STAFF_NAME!!' ' !! STAFF_SURNAME,
SUBSTR(INSTITUTION,1,20),ROOM_BUIL(2),
(36)'-',(8)'-',(20)'-',(4)'-',
(27)'-',(20)'-')
(COL(1),F(4),COL(8),P'(5)9',X(1),A,
COL(44),A,COL(56),A,COL(79),P'Z9',
COL(84),A,COL(113),A,SKIP,COL(14),A,SKIP,
COL(8),A,COL(46),A,COL(56),A,COL(78),A,
COL(84),A,COL(113),A);
READ FILE(STAFF) INTO(STAFF_REC);
END;
END;
CLOSE FILE(SYSPRINT);
CLOSE FILE(BUILDING);
MESSAGE='PRINT-STAFF-SCHEDULE';
END;

PROCESS_STAFF_DATA:PROC;
DCL EOF BIT(1);
ON ENDFILE(STAFFINF) BEGIN;
CLOSE FILE(STAFFINF);
CLOSE FILE(STAFF) ENV(LOCK);
EOF='1'B; END;
DO S#=1 TO #OFSESSIONS;
MESSAGE=CLS!!ESC!!'"2-!!BLINK!!
S# !! ' _ OTURUMUN GOREVLILERI ISLENIYOR.' !!
ESC!!'"o7'!!DC2ETX;
WRITE FILE(R) FROM(MESSAGE);
TITLE(STAFFINF)=SQUASHID !! '/STAFFINF/' !! S#;
OPEN FILE(STAFFINF) INPUT;
TITLE(STAFF)=SQUASHID !! '/STAFF/' !! S#;
OPEN FILE(STAFF) UPDATE;
READ FILE(STAFFINF) INTO(STAFF_INF_REC);
EOF='0'B;
DO WHILE(~ EOF);
READ FILE(STAFF) INTO(STAFF_REC)
INDEX(STAFF_INF_REC.@OFSTAFF-1);
STAFF_NAME=INF_NAME;

```



```

        STAFF_SURNAME=INF_SURNAME;
        STAFF_TITLE=INF_TITLE;
        INSTITUTION=INF_INSTITUTION;
        REWRITE FILE(STAFF) FROM(STAFF_REC);
        READ FILE(STAFFINF) INTO(STAFF_INF_REC);
    END;
END;
MESSAGE='PROCESS-STAFF-DATA';
END;

STAFF_PRINTOUTS:PROC;

    DCL PROCEDURES(10) ENTRY INIT(PRINT_ID_CARD,
                                  PRINT_DUTY_FORM,
                                  PRINT_CHEQUE,
                                  PRINT_STAFF_LIST,
                                  PRINT_ROOM_REC_DEL,
                                  PRINT_CHEQUE_DIST,
                                  PRINT_CHEQUE_ENV,
                                  PRINT_DUTY_SITES,
                                  PRINT_BUIL_REC_DEL,
                                  PRINT_ENV_DIST);

    DCL ( CONTINUEFLAG,
          NOTSPCFYFLAG ) BIT(1);

    CONTINUEFLAG='1'B;
    DO WHILE(CONTINUEFLAG);
        NOTSPCFYFLAG='1'B;
        DO WHILE(NOTSPCFYFLAG);
            NOTSPCFYFLAG='0'B;
            WRITE FILE(R) FROM(SCREEN102);
            READ FILE(R) INTO(STRUCTSPCFY);
            IF SPCFY(1) = ESC THEN
                IF ( SPCFY(3) = POS(16) ) &
                    ( SPCFY(4) = POS(6) ! SPCFY(4) = POS(7) !
                      SPCFY(4) = POS(8) ! SPCFY(4) = POS(9) !
                      SPCFY(4) = POS(10) ! SPCFY(4) = POS(11) !
                      SPCFY(4) = POS(12) ! SPCFY(4) = POS(13) !
                      SPCFY(4) = POS(14) ! SPCFY(4) = POS(15) )
                THEN DO;
                    TYPE=INDEX(ASCII,SPCFY(4))-5;
                END; ELSE
                    IF SPCFY(3) = POS(79) & SPCFY(4) = POS(22)
                    THEN CONTINUEFLAG='0'B;
                    ELSE DO;
                        SUBSTR(SCREEN102,1842,80)=BLINK!!
                        'Gecersiz konumda SPCFY tusuna bastiniz.';
                        NOTSPCFYFLAG='1'B;
                    END;
                ELSE DO;
                    SUBSTR(SCREEN102,1842,80)=BLINK!!
                    'XMIT yerine SPCFY tusunu kullaniniz.';
                    NOTSPCFYFLAG='1'B;
                END;
            END;
            SUBSTR(SCREEN102,1842,80)=' ';

```

```

IF CONTINUEFLAG THEN DO;
CALL PROCEDURES(TYPE);
MESSAGE2=CLS!!ESC!!'"2-!!BLINK!!
MESSAGE2!!' SONA ERDI. XMIT''E BASINIZ.'!!
ESC!!'"o7!!DCZETX;
WRITE FILE(R) FROM(MESSAGE2);
READ FILE(R) INTO(MESSAGE2);
END;
END;
MESSAGE='STAFF-PRINTOUTS';
END;

```

```

PRINT_ID_CARD:PROC;
DCL ADR PIC'9999',
EOF BIT(1);
ON ENDFILE(STAFF) BEGIN;
CLOSE FILE(STAFF);
EOF='1'B; END;
ON ENDPAGE(SYSPRINT);
OPEN FILE(SYSPRINT) ENV(KIND='PRINTER');
DO S#=1 TO #OFSESSIONS;
TITLE(STAFF)=SQUASHID !! '/STAFF/' !! S#;
OPEN FILE(STAFF) INPUT;
READ FILE(STAFF) INTO(STAFF_REC);
EOF='0'B; ADR=0;
DO WHILE(~ EOF);
ADR=ADR+1;
PUT SKIP(19) EDIT(SYSID,EXAMDATE(S#),EXAMHOUR(S#),
STAFF_NAME!!' '!!STAFF_SURNAME,
ADR,GOREV(STAFF_REC.DUTY),
ILLER(STAFF_REC.CENTER))
(COL(23),A,SKIP(5),COL(16),A,
COL(29),A,SKIP(2),COL(10),A,SKIP(2),
COL(10),P'9999',SKIP(2),COL(10),A,
SKIP(2),COL(10),A);
READ FILE(STAFF) INTO(STAFF_REC);
END;
END;
CLOSE FILE(SYSPRINT);
MESSAGE2='PRINT-ID-CARD';
END;

```

```

PRINT_DUTY_FORM:PROC;
DCL PLACE CHAR(55),
ADR PIC'9999',
EOF BIT(1);
ON ENDFILE(STAFF) BEGIN;
CLOSE FILE(STAFF); CLOSE FILE(ROOM);
EOF='1'B; END;
ON ENDPAGE(SYSPRINT);
OPEN FILE(SYSPRINT) ENV(KIND='PRINTER',MAXRECSIZE=132);
OPEN FILE(BUILDING);
DO S#=1 TO #OFSESSIONS;
TITLE(STAFF)=SQUASHID !! '/STAFF/' !! S#;
OPEN FILE(STAFF) INPUT;
TITLE(ROOM)=SQUASHID !! '/ROOM/' !! S#;
READ FILE(STAFF) INTO(STAFF_REC);

```



```

EOF='0'B; ADR=0;
DO WHILE(¬ EOF);
  READ FILE(ROOM) INTO(ROOM_REC)
    INDEX(STAFF_REC.ROOM@-1);
  READ FILE(BUILDING) INTO(BUILDING_REC)
    INDEX(ROOM_REC.BUILDING@);

  ADR=ADR+1;
  IF STAFF_REC.DUTY > 4 THEN
    PUT STRING(PLACE) EDIT(STAFF_REC.ROOM#,(9)' ',
      #OFASSIGNED,' KISILIK SALONDA ',
      GOREV(STAFF_REC.DUTY))
      (P'(5)9',A,P'ZZ9',A,A);
  ELSE
    PUT STRING(PLACE) EDIT
      (STAFF_REC.BUILDING#,(9)' ',
      #OFROOM_DIRECTOR(S#),' SALONLU BINADA ',
      GOREV(STAFF_REC.DUTY))
      (P'(5)9',A,P'ZZZ9',A,A);
    PUT SKIP(15) EDIT(SYSID,ADR,STAFF_NAME!!!' !!!
      STAFF_SURNAME,EXAMDATE(S#)!!!' GUNU,!!!
      ' SAAT !!!EXAMHOUR(S#)!!!' DAKI !!!
      SYSNAME,PLACE,BUILDING_NAME.LINE1,
      BUILDING_NAME.LINE2,
      ILLER(STAFF_REC.CENTER))
      (COL(37),A,SKIP(2),COL(59),P'9999',SKIP(2),
      COL(15),A,SKIP(2),COL(7),A,SKIP(3),
      COL(11),A,SKIP(5),2(COL(15),A,SKIP),
      SKIP,COL(15),A);
    READ FILE(STAFF) INTO(STAFF_REC);
  END;
END;
CLOSE FILE(BUILDING);
CLOSE FILE(SYSPRINT);
MESSAGE2='PRINT-DUTY-FORM';
END;

END REGISTRATION;

```

```

/*          - - - - -          */
/*          R E X / F I L E L O A D          */
/*          - - - - -          */
FILELOAD:PROC OPTIONS(MAIN);

DCL LIB LIBRARY(TITLE='OBJECT/REX/LIBRARY/STARTJOB. ');
DCL STARTJOB ENTRY(CHAR(*)) RETURNS(BIT(1))
      OPTIONS(LIBRARY=LIB);

DCL D FILE RECORD ENV(KIND='DISK',FILETYPE=7),
  F FILE RECORD ENV(KIND='DISK',MAXRECSIZE=80,
      FRAMESIZE=8,FILEKIND='PLISYMBOL'),
  P FILE RECORD ENV(KIND='DISK',MAXRECSIZE=1080,
      AREASIZE=6),
  Z FILE RECORD ENV(KIND='DISK',MAXRECSIZE=15,
      BLOCKSIZE=420,FRAMESIZE=48,AREAS=1,
      AREASIZE=1008,FILEKIND='JOBSYMBOL');

DCL 1 STRUCT11,
  2 LOADINGFILE CHAR(36),
  2 LOADINGREC CHAR(36),
  2 FILESIZE CHAR(6),
  2 ACCESSKEY CHAR(36),
  2 LOADERFILE CHAR(36),
  2 LOADERREC CHAR(36),
  2 REPORT CHAR(1),
  2 WARNING CHAR(1),
  2 EXIT11 CHAR(1);
DCL 1 STRUCT12,
  2 ARRAY(34) CHAR(30),
  2 EXIT12 CHAR(1);
DCL 1 STRUCT50,
  2 FILEID CHAR(36),
  2 COMMENT(6) CHAR(68),
  2 EXIT50 CHAR(1);
DCL 1 STRUCT51,
  2 NUMBERS CHAR(78),
  2 EXIT51 CHAR(1);
DCL 1 STRUCT52,
  2 ARRAY(17),
  3 SEQNUMBER CHAR(06),
  3 COMMAND CHAR(60),
  2 EXIT52 CHAR(1);
DCL 1 STRUCT53,
  2 ARRAY(17),
  3 ROW PIC'99',
  3 COL PIC'999',
  3 DATA CHAR(36),
  3 FORMAT CHAR(24),
  2 EXIT53 CHAR(1);
DCL 1 STRUCT54,
  2 RECORDLEN CHAR(05),
  2 BLOCKLEN CHAR(05),
  2 IOSUBPARAMS(5),
  3 PARAMETER CHAR(16),
  3 MNEMONIC CHAR(16),
  2 EXIT54 CHAR(1);

```

```

DCL 1 STRUCTSPCFY,
    2 SPCFY(4)          CHAR(1);

DCL STORE(1000)        LIKE STRUCT52.ARRAY,
    MESSAGE            CHAR(200) VAR,
    FILEVAR            CHAR(36)  VAR,
    CONFIGURE          CHAR(18),
    COL73T080          PIC'(8)9' INIT(1000),
    COL83T090          PIC'(8)9' INIT(100),
    LASTSEQ            PIC'(6)9',
    MYSTA              PIC'9999',
    MESSAGECOUNT      PIC'99',
    NOTSPCFYFLAG       BIT(1),
    CONTINUEFLAG       BIT(1),
    OBJECTFLAG         BIT(1),
    REPORTFLAG         BIT(1),
    SOURCEFLAG         BIT(1),
    WARNINGFLAG        BIT(1),
    MYSELF              BUILTIN;

% INCLUDE 'REX/SCREENS/DECLARE.';
% INCLUDE 'REX/SCREENS/50.';
% INCLUDE 'REX/SCREENS/51.';
% INCLUDE 'REX/SCREENS/52.';
% INCLUDE 'REX/SCREENS/53.';
% INCLUDE 'REX/SCREENS/54.';
% INCLUDE 'REX/SCREENS/55.';
% INCLUDE 'REX/SCREENS/10.';
% INCLUDE 'REX/SCREENS/11.';
% INCLUDE 'REX/SCREENS/12.';

FILEDEFINE:PROC(STRING,USAGE,FILE);
    DCL (STRING,USAGE,FILE) CHAR(*),
        (RCDSIZE,BLKSIZE,FRMSIZE) PIC'(5)9';
    TITLE(D)=FILE;
    IF RESIDENT(D) THEN DO;
        OPEN FILE(D);
        RCDSIZE=MAXRECSIZE(D);
        BLKSIZE=BLOCKSIZE(D);
        FRMSIZE=FRAMESIZE(D);
        CLOSE FILE(D);
        IF FRMSIZE = 48 THEN DO;
            RCDSIZE=RCDSIZE*6;
            BLKSIZE=BLKSIZE*6;
        END;
        SUBSTR(SCREEN54,604,5)=RCDSIZE;
        SUBSTR(SCREEN54,764,5)=BLKSIZE;
    END;
    SUBSTR(SCREEN54,432,19)=STRING;
    IF OPERATION > 0 THEN WRITE FILE(R) FROM(DCZETX);
    WRITE FILE(R) FROM(SCREEN54);
    IF OPERATION > 0 THEN DO; READ FILE(P) INTO(STRUCT54);
        EXIT54=ETX;
        WRITE FILE(R) FROM(STRUCT54); END;
    READ FILE(R) INTO(STRUCT54);
    IF OPERATION = 0 THEN WRITE FILE(P) FROM(STRUCT54);
    CALL GEN(9,'MAXRECSIZE='!!EXCEPT(' ','RECORDLEN)!!','');

```

```

CALL GEN(9,'BLOCKSIZE = '!!EXCEPT(' ',BLOCKLEN)!!','');
CALL GEN(9,'FRAMESIZE =8,');
CALL GEN(9,'MYUSE='''!!USAGE!''','');
DO I=1 TO 5;
    IF PARAMETER(I) ^= ' ' THEN
        CALL GEN(9,PARAMETER(I)!!='!!MNEMONIC(I)!!','');
    END;
CALL GEN(9,'TITLE='''!!EXCEPT(' ',FILE)!!'. '');');
SUBSTR(SCREEN54,604,5)='    ';
SUBSTR(SCREEN54,764,5)='    ';
END FILEDEFINE;

```

```

JOBTYPE:PROC;
TITLE(D)=FILEVAR;
IF RESIDENT(D) THEN SOURCEFLAG='1'B;
TITLE(D)='OBJECT/'!! FILEVAR;
IF RESIDENT(D) THEN OBJECTFLAG='1'B;
IF ~ SOURCEFLAG & ~ OBJECTFLAG THEN RETURN;
IF SOURCEFLAG & ~ OBJECTFLAG THEN
    SUBSTR(SCREEN55,802,80)=(80)' ';
NOTSPCFYFLAG='1'B;
DO WHILE(NOTSPCFYFLAG);
    NOTSPCFYFLAG='0'B;
    IF OPERATION > 0 THEN WRITE FILE(R) FROM(DC2ETX);
    WRITE FILE(R) FROM(SCREEN55);
    READ FILE(R) INTO(STRUCTSPCFY);
    IF SPCFY(1) = ESC THEN
        IF ( SPCFY(3) = POS(28) ) &
            ( SPCFY(4) = POS(7) ! SPCFY(4) = POS(9) !
              SPCFY(4) = POS(11) ! SPCFY(4) = POS(13) )
        THEN
            TYPE=INDEX(ASCII,SPCFY(4))-6;
        ELSE DO;
            SUBSTR(SCREEN55,1762,80)=
                'Gecers{z konumda SPCFY tusuna bast{n{z.';
            NOTSPCFYFLAG='1'B;
        END;
    ELSE DO;
        SUBSTR(SCREEN55,1762,80)=
            'XMIT yer{ne SPCFY tusunu kullan{n{z.';
        NOTSPCFYFLAG='1'B;
    END;
END;
IF TYPE = 1 THEN RETURN;
TITLE(Z)=FILEVAR!!'/ZIPFILE';
CALL ZIP('?BEGIN JOB;USER=REX/REX');
CALL ZIP(';STATION='!!MYSTA);
CALL ZIP(';TASK T');
CALL ZIP(';FILE D(KIND=DISK,TITLE='!!MYSTA!!'/WAIT2)');
IF TYPE = 3 ! TYPE = 7 THEN DO;
    CALL ZIP(';COMPILE OBJECT/'!!FILEVAR!!
        ' WITH PL/IETJ LIBRARY');
    CALL ZIP(';PL/I FILE CARD(KIND=DISK,TITLE='!!
        FILEVAR!!')');
    CALL ZIP(';IF T IS COMPILEDOK');
    CALL ZIP(' THEN DISPLAY("COMPILE OK")');
    CALL ZIP(' ELSE DISPLAY("SYNTAX ERROR")');

```

```

END;
IF TYPE = 5 THEN
  CALL ZIP(';RUN OBJECT/'!!FILEVAR);
IF TYPE = 7 THEN
  CALL ZIP(';IF T IS COMPILEDOK THEN RUN OBJECT/'!!
    FILEVAR);
CALL ZIP(';D(MYUSE=OUT); OPEN(D); LOCK(D)');
CALL ZIP(';END JOB');
CLOSE FILE(Z) ENV(LOCK);
CALL STARTJOB(FILEVAR!!'/ZIPFILE');
SIGNAL ENDFILE(R);
END JOBTYP;

GEN:PROC(SPACE,STRING);
  DCL STRING CHAR(*),
  RECORD CHAR(80);
  RECORD=COPY(' ',SPACE)!!STRING!!
    COPY(' ',80-SPACE-LENGTH(STRING));
  SUBSTR(RECORD,73,8)=COL73T080;
  COL73T080=COL73T080+1000;
  WRITE FILE(F) FROM(RECORD);
END GEN;

COMMANDS:PROC(SPACE,MENUTYPE);
  DCL MENUTYPE CHAR(9);
  SUBSTR(SCREEN52,19,9)=MENUTYPE;
  LASTSEQ=100; POINTER=0;
  CONTINUEFLAG='1'B;
  DO WHILE(CONTINUEFLAG);
    DO I=1 TO 17;
      SUBSTR(SCREEN52,246+I*80,6)=LASTSEQ;
      LASTSEQ=LASTSEQ+100;
    END;
    IF OPERATION > 0 THEN WRITE FILE(R) FROM(DCZETX);
    WRITE FILE(R) FROM(SCREEN52);
    IF OPERATION > 0 THEN DO; READ FILE(P) INTO(STRUCT52);
      EXIT52=ETX;
      WRITE FILE(R) FROM(STRUCT52); END;
    READ FILE(R) INTO(STRUCT52);
    IF OPERATION = 0 THEN WRITE FILE(P) FROM(STRUCT52);
    DO I=1 TO 17 WHILE(CONTINUEFLAG);
      IF STRUCT52.COMMAND(I) = ' ' THEN DO;
        POINTER=POINTER+1;
        STORE(POINTER)=STRUCT52.ARRAY(I) , BY NAME;
      END;
      IF STRUCT52.SEQNUMBER(I) = '999999' THEN DO;
        DO J=1 TO POINTER;
          CALL GEN(SPACE,STORE.COMMAND(J));
        END;
        CONTINUEFLAG='0'B;
      END;
    END;
  END;
END;
END;
END;
END COMMANDS;

PUTITEMS:PROC(SPACE,MENUTYPE,LASTR0W);
  DCL STACK(1000) CHAR(48) VAR,

```

```

        MENUTYPE      CHAR(14),
        LASTROW      PIC'99';
SUBSTR(SCREEN53,23,14)=MENUTYPE;
LASTROW=1; POINTER=0;
CONTINUEFLAG='1'B;
DO WHILE(CONTINUEFLAG);
    IF OPERATION > 0 THEN WRITE FILE(R) FROM(DC2ETX);
    WRITE FILE(R) FROM(SCREEN53);
    IF OPERATION > 0 THEN DO; READ FILE(P) INTO(STRUCT53);
        EXIT53=ETX;
        WRITE FILE(R) FROM(STRUCT53); END;
    READ FILE(R) INTO(STRUCT53);
    IF OPERATION ^= 0 THEN WRITE FILE(P) FROM(STRUCT53);
    DO I=1 TO 17 WHILE(ROW(I) ^= 99);
        IF ROW(I) ^= 0 THEN
            IF ROW(I) = LASTROW & LASTROW ^= 1 THEN
                CALL EDITLIST('SKIP(0),');
            ELSE IF ROW(I) > LASTROW THEN DO;
                CALL EDITLIST('SKIP('!!ROW(I)-LASTROW!!'),');
                LASTROW=ROW(I);
            END;
        IF COL(I) ^= 0 THEN
            CALL EDITLIST('COL('!!COL(I)!!'),');
        IF DATA(I) ^= ' ' THEN DO;
            DO J=36 TO 1 BY -1
                WHILE(SUBSTR(DATA(I),J,1) = ' '); END;
            CALL GEN(SPACE,SUBSTR(DATA(I),1,J));
            END;
        IF FORMAT(I) ^= ' ' THEN
            CALL EDITLIST(EXCEPT(' ',FORMAT(I))!!',');
    END;
    IF I < 18 THEN CONTINUEFLAG='0'B;
END;
CALL GEN(SPACE,'''''');
CALL GEN(SPACE+3,'(!!STACK(1));
DO I=2 TO POINTER;
    CALL GEN(SPACE+3,STACK(I));
END;
CALL GEN(SPACE+3,'A);');

EDITLIST:PROC(S);
    DCL S CHAR(*);
    POINTER=POINTER+1;
    STACK(POINTER)=S;
END EDITLIST;
END PUTITEMS;

ZIP:PROC(STRING);
    DCL STRING CHAR(*),
        RECORD CHAR(90);
    RECORD=STRING !! COPY(' ',90-LENGTH(STRING));
    SUBSTR(RECORD,83,8)=COL83T090;
    COL83T090=COL83T090+100;
    WRITE FILE(Z) FROM(RECORD);
END ZIP;

INITIALIZE:PROC;

```

```

OPERATION=TASKVALUE(MYSELF);
IF OPERATION > 0 THEN DO;
  OPEN FILE(P) UPDATE;
  CONFIGURE=ESC!!'RH00A00102'!!ESC!!'RC';
  WRITE FILE(R) FROM(CONFIGURE);
  READ FILE(R) INTO(MESSAGE);
  IF SUBSTR(MESSAGE,1,6) ^= '*****' THEN STOP;
END;
MYSTA=STATION(MYSELF) * -1;
CALL GETUSERPARAMS(MYSTA);

IF OPERATION > 0 THEN WRITE FILE(R) FROM(DCZETX);
WRITE FILE(R) FROM(SCREEN50);
IF OPERATION > 0 THEN DO; READ FILE(P) INTO(STRUCT50);
  EXIT50=ETX;
  WRITE FILE(R) FROM(STRUCT50); END;
READ FILE(R) INTO(STRUCT50);
IF OPERATION ^= 0 THEN WRITE FILE(P) FROM(STRUCT50);
TITLE(F)=FILEID;
FILEVAR=EXCEPT(' ',FILEID);
IF OPERATION >= 0 THEN CALL JOBTYP;
CALL GEN(0,EXCEPT(' /',BEFORE(FILEID,''))!!
  ':PROC OPTIONS(MAIN);');

SUBSTR(SCREEN51,246,36)=SYSID !! '/LAYOUT/CANDIDATE';
SUBSTR(SCREEN51,286,36)=SYSID !! '/LAYOUT/REGOPTIC';
SUBSTR(SCREEN51,326,36)='LAYOUT/ROOM';
SUBSTR(SCREEN51,366,36)='LAYOUT/BUILDING';
SUBSTR(SCREEN51,406,36)='LAYOUT/STAFF';
SUBSTR(SCREEN51,446,36)='LAYOUT/STAFFINF';
SUBSTR(SCREEN51,486,36)='LAYOUT/TITLES';
SUBSTR(SCREEN51,526,36)='LAYOUT/DUTIES';
SUBSTR(SCREEN51,566,36)=SYSID !! '/LAYOUT/AREACODES';
SUBSTR(SCREEN51,606,36)='LAYOUT/PROVINCES';
IF OPERATION > 0 THEN WRITE FILE(R) FROM(DCZETX);
WRITE FILE(R) FROM(SCREEN51);
IF OPERATION > 0 THEN DO; READ FILE(P) INTO(STRUCT51);
  EXIT51=ETX;
  WRITE FILE(R) FROM(STRUCT51); END;
READ FILE(R) INTO(STRUCT51);
IF OPERATION ^= 0 THEN WRITE FILE(P) FROM(STRUCT51);
IF NUMBERS ^= ' ' THEN DO;
  DO WHILE(INDEX(NUMBERS,',') > 0);
  CALL GEN(1,'% INCLUDE '!!EXCEPT(' ',SUBSTR(SCREEN51,
    206+40*BEFORE(NUMBERS,','),36))!!'. ');
  NUMBERS=AFTER(NUMBERS,',');
  END;
  CALL GEN(1,'% INCLUDE '!!EXCEPT(' ',SUBSTR(SCREEN51,
    206+40*BEFORE(NUMBERS,','),36))!!'. ');
  END;
END INITIALIZE;

% INCLUDE 'REX/GETUSERPARAMS.';

ON RECORD(P);
ON RECORD(R);
ON ENDFILE(R) BEGIN;

```



```

CLOSE FILE(R);
IF OPERATION > 0 THEN DO;
    CONFIGURE=ESC!!'RH00ADD100'!!ESC!!'RC';
    WRITE FILE(R) FROM(CONFIGURE);
    READ FILE(R) INTO(MESSAGE);
END;
CLOSE FILE(F) ENV(LOCK);
IF OPERATION = 0 THEN CLOSE FILE(P) ENV(LOCK);
STOP;
END;
ON ERROR SIGNAL ENDFILE(R);

CALL INITIALIZE;

NOTSPCFYFLAG='1'B;
DO WHILE(NOTSPCFYFLAG);
    NOTSPCFYFLAG='0'B;
    IF OPERATION > 0 THEN
        READ FILE(P) INTO(STRUCTSPCFY);
    ELSE DO;
        WRITE FILE(R) FROM(SCREEN10);
        READ FILE(R) INTO(STRUCTSPCFY);
        IF OPERATION < 0 THEN WRITE FILE(P) FROM(STRUCTSPCFY);
    END;
    IF SPCFY(1) = ESC THEN
        IF SPCFY(3) = POS(57) & SPCFY(4) = POS(9)
            THEN TYPE=1; ELSE
        IF SPCFY(3) = POS(71) & SPCFY(4) = POS(9)
            THEN TYPE=2; ELSE
        IF SPCFY(3) = POS(57) & SPCFY(4) = POS(11)
            THEN TYPE=3; ELSE
        IF SPCFY(3) = POS(71) & SPCFY(4) = POS(11)
            THEN TYPE=4; ELSE
        IF SPCFY(3) = POS(71) & SPCFY(4) = POS(13)
            THEN TYPE=5;
        ELSE DO;
            SUBSTR(SCREEN10,1762,80)=
                'Gecersiz konumda SPCFY tusuna bast{n{z.';
            NOTSPCFYFLAG='1'B;
        END;
    ELSE DO;
        SUBSTR(SCREEN10,1762,80)=
            'XMIT yer{ne SPCFY tusunu kullan{n{z.';
        NOTSPCFYFLAG='1'B;
    END;
END;

IF TYPE > 1 THEN SUBSTR(SCREEN11,642,1)=BRIGHT;
IF TYPE < 5 THEN SUBSTR(SCREEN11,802,1)=BRIGHT;
IF TYPE < 3 THEN DO;
    SUBSTR(SCREEN11,1042,1)=BRIGHT;
    SUBSTR(SCREEN11,1202,1)=BRIGHT;
END;

IF OPERATION > 0 THEN WRITE FILE(R) FROM(DC2ETX);
WRITE FILE(R) FROM(SCREEN11);
IF OPERATION > 0 THEN DO; READ FILE(P) INTO(STRUCT11);
EXIT11=ETX;

```



```

WRITE FILE(R) FROM(STRUCT11); END;
READ FILE(R) INTO(STRUCT11);
IF OPERATION ^= 0 THEN WRITE FILE(P) FROM(STRUCT11);
IF REPORT = 1 THEN REPORTFLAG='1'B;
IF WARNING = 1 THEN WARNINGFLAG='1'B;

CALL GEN(3,'DCL YUKLENEK FILE RECORD ENV(');
CALL GEN(9,'KIND='DISK',');
IF TYPE = 2 ! TYPE > 4 THEN
    CALL FILEDEFINE(' Gunlenecek kutugun','IO',LOADINGFILE);
ELSE
    CALL FILEDEFINE('Yaratilacak kutugun','OUT',LOADINGFILE);

IF TYPE > 2 THEN DO;
    CALL GEN(3,'DCL YUKLEYEN FILE RECORD ENV(');
    CALL GEN(9,'KIND='DISK',');
    CALL FILEDEFINE(' Yukleyen kutugun ','IN',LOADERFILE);
END;

CALL GEN(3,'DCL ISLE BIT(1);');
IF TYPE = 5 THEN DO;
    CALL GEN(3,'DCL ERIS BIT(1);');
    CALL GEN(3,'DCL GECERSIZ FIXED(5);');
END;
CALL GEN(3,'DCL ISLENEN FIXED(5);');
CALL GEN(3,'DCL ATLANAM FIXED(5);');

IF WARNINGFLAG THEN DO;
    IF OPERATION > 0 THEN WRITE FILE(R) FROM(DCZETX);
    WRITE FILE(R) FROM(SCREEN12);
    IF OPERATION > 0 THEN DO; READ FILE(P) INTO(STRUCT12);
        EXIT12=ETX;
        WRITE FILE(R) FROM(STRUCT12); END;
    READ FILE(R) INTO(STRUCT12);
    IF OPERATION ^= 0 THEN WRITE FILE(P) FROM(STRUCT12);
    CONTINUEFLAG='1'B;
    DO I=34 TO 1 BY -1 WHILE(CONTINUEFLAG);
        IF STRUCT12.ARRAY(I) ^= ' ' THEN DO;
            MESSAGECOUNT=I;
            CONTINUEFLAG='0'B;
        END;
    END;
    IF MESSAGECOUNT > 0 THEN DO;
        CALL GEN(3,'DCL ILETI('!!MESSAGECOUNT!!
            ') CHAR(30) INIT(');
        DO I=1 TO MESSAGECOUNT-1;
            CALL GEN(9,'!!!STRUCT12.ARRAY(I)!!!',');
        END;
        CALL GEN(9,'!!!STRUCT12.ARRAY(MESSAGECOUNT)!!!
            ');');
    END;
END;

IF TYPE = 5 THEN DO;
    CALL GEN(3,'DCL ILETITUT('!!MESSAGECOUNT!!) BIT(1);');
    CALL GEN(1,'% DCL INVALIDKEY CHAR; !!!
        '% INVALIDKEY='ERIS='!!!0!!!B!!!');

```

```

END;
CALL GEN(1,'% DCL SKIPTONEXT CHAR; !!!
        '% SKIPTONEXT=' 'ISLE=' '0' 'B' ');

IF WARNINGFLAG THEN
    CALL GEN(1,'% DCL MESSAGE CHAR; !!!
            '% MESSAGE=' 'CALL UYARI' ');

IF TYPE > 2 THEN DO;
    CALL GEN(3,'ON ENDFILE(YUKLEYEN) BEGIN;');
    CALL GEN(6,'CLOSE FILE(YUKLEENEN) ENV(LOCK);');
    CALL GEN(6,'DISPLAY(' 'YUKLEENEN TUTANAK SAYISI=' '!!!
                    ISLEENEN);');
    CALL GEN(6,'DISPLAY(' 'ATLANAN TUTANAK SAYISI=' '!!!
                    ATLANAN);');
IF TYPE = 5 THEN
    CALL GEN(6,'DISPLAY(' 'GECERSİZ ANAHTAR SAYISI=' '!!!
                    GECERSİZ);');
    CALL GEN(6,'STOP;');
    CALL GEN(3,'END;');
END;

IF REPORTFLAG THEN DO;
    CALL GEN(3,'ON ENDPAGE(SYSPRINT) BEGIN;');
    CALL GEN(6,'DCL (DATE,TIME) BUILTIN;');
    CALL GEN(6,'DCL TARİH CHAR(6) INIT(DATE());');
    CALL GEN(6,'DCL SAAT CHAR(9) INIT(TIME());');
    CALL GEN(6,'SAYFA=SAYFA+1;');
    CALL GEN(6,'PUT PAGE EDIT(');
    CALL GEN(12,'(25)' ' '!!!' 'K O N T R O L ' ' ');
    CALL GEN(12,' 'L I S T E S I' '!!!(25)' ' ');
    CALL GEN(12,' 'SAAT: ' ',SUBSTR(SAAT,1,2),' ': ' ');
    CALL GEN(12,'SUBSTR(SAAT,3,2),' ': ' ',SUBSTR(SAAT,5,2),' ');
    CALL GEN(12,' 'TARİH: ' ',SUBSTR(TARİH,5,2),' '/' ' ');
    CALL GEN(12,'SUBSTR(TARİH,3,2),' '/'19' '!!!
            'SUBSTR(TARİH,1,2),' ');
    CALL GEN(12,' 'SAYFA: ' ',SAYFA)');
    CALL GEN(15,'(15(A),F(4));');
    CALL GEN(6,'PUT SKIP(2) EDIT(');
    CALL PUTITEMS(12,' SAYFA BASI ' ',LASTROW);
    CALL GEN(3,'END;');
END;

CALL COMMANDS(3,'BASLANGIC');

IF REPORTFLAG THEN DO;
    CALL GEN(3,'OPEN FILE(SYSPRINT) ENV(KIND=' 'PRINTER' ',');
    CALL GEN(12,'MAXRECSIZE=132,PAGESIZE=60);');
    CALL GEN(3,'SIGNAL ENDPAGE(SYSPRINT);');
END;

IF TYPE > 2 THEN
    CALL GEN(3,'DO WHILE(' '1' 'B);');
ELSE DO;
    IF TYPE = 1 THEN
        CALL GEN(3,'KUTUKSONU=' '!!!EXCEPT(' ' ',FILESIZE)!!!');
    ELSE

```

```

        CALL GEN(3,'KUTUKSONU=LASTRECORD(YUKLELEN)');
        CALL GEN(3,'DO SAYAC = 1 TO KUTUKSONU');
    END;

    IF WARNINGFLAG THEN
        CALL GEN(6,'ILETITUT='0'B');

    CALL GEN(6,'ISLE='1'B);
    IF TYPE = 5 THEN CALL GEN(6,'ERIS='1'B);

    IF TYPE > 2 THEN DO;
        CALL GEN(6,'READ FILE(YUKLEYEN) INTO('!!
            EXCEPT(' ',LOADERREC)!!');
        IF TYPE = 5 THEN DO;
            CALL COMMANDS(6,' ANAHTAR ');
            CALL GEN(6,'IF ~ ERIS THEN');
            CALL GEN(9,'GECERSIZ=GECERSIZ+1');
            CALL GEN(6,'ELSE DO;');
            CALL GEN(9,'READ FILE(YUKLELEN) INTO('!!
                EXCEPT(' ',LOADINGREC)!!');
            CALL GEN(15,'INDEX('!!EXCEPT(' ',ACCESSKEY)!!');
        END; ELSE
        IF TYPE = 4 THEN
            CALL GEN(6,'READ FILE(YUKLELEN) INTO('!!
                EXCEPT(' ',LOADINGREC)!!');
        ELSE;
    END; ELSE
    IF TYPE = 2 THEN
        CALL GEN(6,'READ FILE(YUKLELEN) INTO('!!
            EXCEPT(' ',LOADINGREC)!!');

    CALL COMMANDS(6,'YINELELEN');

    CALL GEN(6,'IF ISLE THEN DO;');
    IF TYPE < 5 THEN DO;
        CALL GEN(9,'WRITE FILE(YUKLELEN) FROM('!!
            EXCEPT(' ',LOADINGREC)!!');
    END; ELSE DO;
        CALL GEN(9,'WRITE FILE(YUKLELEN) FROM('!!
            EXCEPT(' ',LOADINGREC)!!');
        CALL GEN(15,'INDEX('!!EXCEPT(' ',ACCESSKEY)!!');
    END;
    CALL GEN(9,'ISLENEN=ISLENEN+1');
    CALL GEN(6,'END; ELSE');
    CALL GEN(9,'ATLANAN=ATLANAN+1');
    IF TYPE = 5 THEN CALL GEN(6,'END;');

    IF REPORTFLAG THEN DO;
        CALL GEN(6,'PUT SKIP EDIT(');
        CALL PUTITEMS(12,'VERI SATIRLARI',LASTROW);
    END;

    IF WARNINGFLAG THEN DO;
        CALL GEN(6,'IF SUM(ILETITUT) > 0 THEN DO;');
        CALL GEN(9,'PUT SKIP EDIT('ILETILER:') (COL(10),A);');
        CALL GEN(9,'DO I=1 TO '!!MESSAGECOUNT!!');
        CALL GEN(12,'IF ILETITUT(I) = '1'B THEN');

```

```
CALL GEN(15,'PUT EDIT(ILETI(I)) (COL(22),A);');
CALL GEN(9,'END;');
CALL GEN(6,'END;');
END;

CALL GEN(3,'END;');

IF TYPE < 3 THEN DO;
CALL GEN(3,'CLOSE FILE(YUKLELEN) ENV(LOCK);');
CALL GEN(3,'DISPLAY(''YUKLELEN TUTANAK SAYISI=!!!
ISLELEN);');
CALL GEN(3,'DISPLAY(''ATLANAN TUTANAK SAYISI=!!!
ATLANAN);');
END;

IF WARNINGFLAG THEN
CALL GEN(0,'UYARI:PROC(P); ILETITUT(P)=''1''8; END;');

CALL GEN(0,'END;');
SIGNAL ENDFILE(R);

END FILELOAD;
```

```

/*          - - - - -          */
/*          R E X / P R I N T          */
/*          - - - - -          */
PRINT:PROC OPTIONS(MAIN);

DCL LIB LIBRARY(TITLE='OBJECT/REX/LIBRARY/STARTJOB. ');
DCL STARTJOB ENTRY(CHAR(*)) RETURNS(BIT(1))
      OPTIONS(LIBRARY=LIB);

DCL D FILE RECORD ENV(KIND='DISK',FILETYPE=7),
  F FILE RECORD ENV(KIND='DISK',MAXRECSIZE=80,
                    FRAMESIZE=8,FILEKIND='PLISYMBOL'),
  P FILE RECORD ENV(KIND='DISK',MAXRECSIZE=1080,
                    AREASIZE=6),
  Z FILE RECORD ENV(KIND='DISK',MAXRECSIZE=15,
                    BLOCKSIZE=420,FRAMESIZE=48,AREAS=1,
                    AREASIZE=1008,FILEKIND='JOBSYMBOL');

DCL 1 STRUCT20,
  2 PRINTFILE          CHAR(36),
  2 PRINTREC          CHAR(36),
  2 SKIPCONTROL        PIC'9',
  2 CONTIGUOUS         PIC'9',
  2 GROUPITEM          CHAR(36),
  2 GROUPFORMAT        CHAR(24),
  2 EXIT20             CHAR(10);

DCL 1 STRUCT21,
  2 PAGESIZE           PIC'99',
  2 HEADINGSIZE        PIC'99',
  2 FOOTINGSIZE        PIC'99',
  2 #OFLINES           PIC'99',
  2 EXIT21             CHAR(1);

DCL 1 STRUCT50,
  2 FILEID             CHAR(36),
  2 COMMENT(6)         CHAR(68),
  2 EXIT50             CHAR(1);

DCL 1 STRUCT51,
  2 NUMBERS            CHAR(78),
  2 EXIT51             CHAR(1);

DCL 1 STRUCT52,
  2 ARRAY(17),
  3 SEQNUMBER          CHAR(06),
  3 COMMAND            CHAR(60),
  2 EXIT52             CHAR(1);

DCL 1 STRUCT53,
  2 ARRAY(17),
  3 ROW                PIC'99',
  3 COL                PIC'999',
  3 DATA              CHAR(36),
  3 FORMAT             CHAR(24),
  2 EXIT53             CHAR(1);

DCL 1 STRUCT54,
  2 RECORDLEN          CHAR(05),
  2 BLOCKLEN           CHAR(05),
  2 IOSUBPARAMS(5),
  3 PARAMETER          CHAR(16),
  3 MNEMONIC           CHAR(16);

```

```

      2 EXIT54          CHAR(1);
DCL 1 STRUCTSPCFY,    CHAR(1);
      2 SPCFY(4)
DCL STORE(1000)      LIKE STRUCT52.ARRAY,
  MESSAGE            CHAR(200) VAR,
  FILEVAR            CHAR(36)  VAR,
  CONFIGURE          CHAR(18),
  COL73T08D          PIC'(8)9' INIT(1000),
  COL83T09D          PIC'(8)9' INIT(100),
  LASTSEQ            PIC'(6)9',
  MYSTA              PIC'9999',
  MESSAGECOUNT     PIC'99',
  NOTSPCFYFLAG       BIT(1),
  OBJECTFLAG         BIT(1),
  SOURCEFLAG         BIT(1),
  MYSELF             BUILTIN;

% INCLUDE 'REX/SCREENS/DECLARE.';
% INCLUDE 'REX/SCREENS/50.';
% INCLUDE 'REX/SCREENS/51.';
% INCLUDE 'REX/SCREENS/52.';
% INCLUDE 'REX/SCREENS/53.';
% INCLUDE 'REX/SCREENS/54.';
% INCLUDE 'REX/SCREENS/55.';
% INCLUDE 'REX/SCREENS/20.';
% INCLUDE 'REX/SCREENS/21.';

% INCLUDE 'REX/PUTITEMS.';
% INCLUDE 'REX/GEN.';
% INCLUDE 'REX/FILEDEFINE.';
% INCLUDE 'REX/COMMANDS.';
% INCLUDE 'REX/ZIP.';
% INCLUDE 'REX/JOBTYPE.';
% INCLUDE 'REX/GETUSERPARAMS.';
% INCLUDE 'REX/INITIALIZE.';

ON RECORD(P);
ON RECORD(R);
ON ENDFILE(R) BEGIN;
  CLOSE FILE(R);
  IF OPERATION > 0 THEN DO;
    CONFIGURE=ESC!!'RH00A00100'!!ESC!!'RC';
    WRITE FILE(R) FROM(CONFIGURE);
    READ FILE(R) INTO(MESSAGE);
  END;
  CLOSE FILE(F) ENV(LOCK);
  IF OPERATION = 0 THEN CLOSE FILE(P) ENV(LOCK);
  STOP;
END;
ON ERROR SIGNAL ENDFILE(R);

CALL INITIALIZE;

IF OPERATION > 0 THEN WRITE FILE(R) FROM(DC2ETX);
WRITE FILE(R) FROM(SCREEN20);
IF OPERATION > 0 THEN DO; READ FILE(P) INTO(STRUCT20);

```

```

EXIT20=ETX;
WRITE FILE(R) FROM(STRUCT20); END;
READ FILE(R) INTO(STRUCT20);
IF OPERATION = 0 THEN WRITE FILE(P) FROM(STRUCT20);
IF OPERATION > 0 THEN WRITE FILE(R) FROM(DC2ETX);
WRITE FILE(R) FROM(SCREEN21);
IF OPERATION > 0 THEN DO; READ FILE(P) INTO(STRUCT21);
EXIT21=ETX;
WRITE FILE(R) FROM(STRUCT21); END;
READ FILE(R) INTO(STRUCT21);
IF OPERATION = 0 THEN WRITE FILE(P) FROM(STRUCT21);
TYPE=0;
IF CONTIGUOUS > 1 THEN TYPE=TYPE+1;
IF GROUPITEM = 0 THEN TYPE=TYPE+2;

CALL GEN(3,'DCL DOKULEN FILE RECORD ENV(');
CALL GEN(9,'KIND="'DISK"',');
CALL FILEDEFINE(' Dokulecek kutugun ','IN',PRINTFILE);

CALL GEN(3,'DCL ERIS BIT(1);');
CALL GEN(3,'DCL ISLENEN FIXED(5);');
CALL GEN(3,'DCL ATLANAN FIXED(5);');

IF TYPE > 1 THEN
CALL GEN(3,'DCL !!!EXCEPT(' ',GROUPITEM)!!!' !!!
EXCEPT(' ',GROUPFORMAT)!!!');

CALL GEN(1,'% DCL SKIPTONEXT CHAR; !!!
%' SKIPTONEXT="'ISLE="'0''"B'';');

CALL GEN(3,'ON ENDFILE(DOKULEN) BEGIN;');
IF TYPE = 1 ! TYPE = 3 THEN DO;
CALL GEN(6,'IF GOSTERGE = 1 THEN DO;');
CALL GEN(9,'CALL SATIRYAZ;');
CALL GEN(9,'DISPLAY('!!CONTIGUOUS!!'-GOSTERGE+1);');
CALL GEN(6,'END;');
END;
CALL GEN(6,'CALL SAYFASONU(1);');
CALL GEN(6,'DISPLAY('DOKULEN TUTANAK SAYISI='!!!
ISLENEN);');
CALL GEN(6,'DISPLAY('ATLANAN TUTANAK SAYISI='!!!
ATLANAN);');
CALL GEN(6,'STOP;');
CALL GEN(3,'END;');

CALL GEN(3,'SAYFABOYU =!!!PAGE SIZE!!!');
CALL GEN(3,'ILKSATIRLAR=!!!HEADINGSIZE!!!');
CALL GEN(3,'SONSATIRLAR=!!!FOOTINGSIZE!!!');
CALL GEN(3,'SATIRARTISI=!!!#OFLINES!!!');
CALL GEN(3,'SAYFASAYACI=1;');

CALL COMMANDS(3,'BASLANGIC');

CALL GEN(3,'OPEN FILE(SYSPRINT) ENV(KIND="'PRINTER"',!!!
'MAXRECSIZE=132,');
IF SKIPCONTROL = 2 THEN
CALL GEN(9,'UNITS="'CHARACTERS"',INTMODE="'EBCDIC"',!!!

```



```

        'CARRIAGECONTROL=' 'CTLASA' ');');
ELSE
    CALL GEN(9,'PAGESIZE=' '!!PAGESIZE!!' ');');

CALL GEN(0,'SAYFASONU:PROC(DONUS)');
IF FOOTINGSIZE > 0 THEN DO;
    CALL GEN(3,'PUT LINE(SAYFABOYU-SONSATIRLAR+1) EDIT(');
    CALL PUTITEMS(9,' SAYFA SONU ',LASTROW);
END;
CALL GEN(3,'IF DONUS = 1 THEN RETURN;');

CALL GEN(0,'SAYFABASI:ENTRY;');
IF HEADINGSIZE > 0 THEN DO;
    CALL GEN(3,'PUT PAGE EDIT(');
    CALL PUTITEMS(9,' SAYFA BASI ',LASTROW);
END;
CALL GEN(3,'SATIRSAYACI=ILKSATIRLAR;');
CALL GEN(3,'SAYFASAYACI=SAYFASAYACI+1;');
CALL GEN(0,'END SAYFASONU;');

CALL GEN(3,'DO WHILE(' '1' 'B);');
CALL GEN(6,'ERIS=' '1' 'B);');
RECORDVAR=EXCEPT(' ',PRINTREC);
IF TYPE = 1 ! TYPE = 3 THEN DO;
    CALL GEN(6,'GOSTERGE=1;');
    CALL GEN(6,'DO WHILE(GOSTERGE <= ' '!!CONTIGUOUS!!' ');');
    RECORDVAR=RECORDVAR!! '(GOSTERGE)';
END;
CALL GEN(9,'READ FILE(DOKULEN) INTO(' '!!RECORDVAR!!' ');');
CALL COMMANDS(9,'YINELLEN');
CALL GEN(9,'IF ERIS THEN DO;');
CALL GEN(12,'ISLENEN=ISLENEN+1;');
CALL GEN(12,'IF ISLENEN = 1 THEN DO;');
CALL GEN(15,'CALL SAYFABASI;');
IF TYPE > 1 THEN DO;
    CALL GEN(15,'SAKLA=' '!!EXCEPT(' ',GROUPITEM)!!' ');');
    CALL GEN(12,'END; ELSE');
    CALL GEN(15,'IF SAKLA = ' '!!EXCEPT(' ',GROUPITEM)!!'
        ' THEN DO;');
    CALL GEN(18,'SAKLA=' '!!EXCEPT(' ',GROUPITEM)!!' ');');
    IF TYPE = 2 THEN CALL GEN(18,'CALL SAYFASONU(0);');
    CALL GEN(15,'END;');
    IF TYPE = 3 THEN CALL GEN(12,'GOSTERGE=GOSTERGE+1;');
    IF TYPE = 2 THEN CALL GEN(12,'CALL SATIRYAZ;');
END; ELSE DO;
    CALL GEN(12,'END;');
    IF TYPE = 1 THEN CALL GEN(12,'GOSTERGE=GOSTERGE+1;');
    ELSE CALL GEN(12,'CALL SATIRYAZ;');
END;
CALL GEN(9,'END; ELSE');
CALL GEN(12,'ATLANAN=ATLANAN+1;');
IF TYPE = 1 ! TYPE = 3 THEN DO;
    CALL GEN(6,'END;');
    CALL GEN(6,'CALL SATIRYAZ;');
END;
CALL GEN(3,'END;');

```



```
CALL GEN(0,'SATIRYAZ:PROC;');
CALL GEN(3,'IF SATIRSAYACI > SAYFABOYU-SONSATIRLAR-!!!
          'SATIRARTISI');
CALL GEN(6,'THEN CALL SAYFASONU(0);');
CALL GEN(3,'PUT SKIP EDIT(');
CALL PUTITEMS(9,'VERI SATIRLARI',LASTROW);
CALL GEN(3,'SATIRSAYACI=SATIRSAYACI+SATIRARTISI;');
CALL GEN(0,'END SATIRYAZ;');

CALL GEN(0,'END;');
SIGNAL ENDFILE(R);

END PRINT;
```

```

/*          - - - - -
/*          R E X / D I S T R I B U T I O N
/*          - - - - -
DISTRIBUTION:PROC OPTIONS(MAIN);

```

```

DCL LIB LIBRARY(TITLE='OBJECT/REX/LIBRARY/STARTJOB. ');
DCL STARTJOB ENTRY(CHAR(*)) RETURNS(BIT(1))
    OPTIONS(LIBRARY=LIB);

```

```

DCL D FILE RECORD ENV(KIND='DISK',FILETYPE=7),
F FILE RECORD ENV(KIND='DISK',MAXRECSIZE=80,
    FRAMESIZE=8,FILEKIND='PLISYMBOL'),
P FILE RECORD ENV(KIND='DISK',MAXRECSIZE=1080,
    AREASIZE=6),
Z FILE RECORD ENV(KIND='DISK',MAXRECSIZE=15,
    BLOCKSIZE=420,FRAMESIZE=48,AREAS=1,
    AREASIZE=1008,FILEKIND='JOBSYMBOL');

```

```

DCL 1 STRUCT30,
    2 INPUTFILE      CHAR(36),
    2 INPUTREC       CHAR(36),
    2 DISTHEADING    CHAR(50),
    2 COLLEVEL       PIC'9',
    2 EXIT30         CHAR(1);

```

```

DCL 1 STRUCT31,
    2 ROWVAR         CHAR(36),
    2 ROWTYPE        PIC'9',
    2 ROWMIN         PIC'(6)9',
    2 ROWMAX         PIC'(6)9',
    2 ROWHEADING     CHAR(20),
    2 ROWVECTOR      CHAR(36),
    2 EXIT31         CHAR(1);

```

```

DCL 1 STRUCT32(3),
    2 COLVAR         CHAR(36),
    2 COLTYPE        PIC'9',
    2 COLMIN         PIC'(6)9',
    2 COLMAX         PIC'(6)9',
    2 COLHEADING     CHAR(20),
    2 COLVECTOR      CHAR(36),
    2 EXIT32         CHAR(1);

```

```

DCL 1 STRUCT50,
    2 FILEID         CHAR(36),
    2 COMMENT(6)     CHAR(68),
    2 EXIT50         CHAR(1);

```

```

DCL 1 STRUCT51,
    2 NUMBERS        CHAR(78),
    2 EXIT51         CHAR(1);

```

```

DCL 1 STRUCT52,
    2 ARRAY(17),
        3 SEQNUMBER  CHAR(06),
        3 COMMAND    CHAR(60),
    2 EXIT52         CHAR(1);

```

```

DCL 1 STRUCT54,
    2 RECORDLEN     CHAR(05),
    2 BLOCKLEN      CHAR(05),
    2 IOSUBPARAMS(5),
        3 PARAMETER  CHAR(16),

```

```

          3 MNEMONIC          CHAR(16),
          2 EXITS4            CHAR(1);
DCL 1 STRUCTSPCFY,
          2 SPCFY(4)          CHAR(1);

DCL STORE(1000)
MESSAGE          CHAR(200) VAR,
FILEVAR          CHAR(36)  VAR,
COLSTR           CHAR(6)   VAR,
Y                CHAR(4)   VAR,
CONFIGURE        CHAR(18),
INDEXPTR         FIXED(1)  INIT(-1),
COL73T080        PIC'(8)9' INIT(1000),
COL83T090        PIC'(8)9' INIT(100),
COLCOUNT        PIC'(6)9',
LASTSEQ          PIC'(6)9',
MYSTA            PIC'9999',
MESSAGECOUNT    PIC'99',
X                PIC'9',
INDEXFLAG        BIT(1),
NOTSPCFYFLAG     BIT(1),
OBJECTFLAG       BIT(1),
SOURCEFLAG       BIT(1),
MYSELF           BUILTIN;

% INCLUDE 'REX/SCREENS/DECLARE.';
% INCLUDE 'REX/SCREENS/50.';
% INCLUDE 'REX/SCREENS/51.';
% INCLUDE 'REX/SCREENS/52.';
% INCLUDE 'REX/SCREENS/54.';
% INCLUDE 'REX/SCREENS/55.';
% INCLUDE 'REX/SCREENS/30.';
% INCLUDE 'REX/SCREENS/31.';

% INCLUDE 'REX/COMMANDS.';
% INCLUDE 'REX/FILEDEFINE.';
% INCLUDE 'REX/JOBTYPE.';
% INCLUDE 'REX/ZIP.';
% INCLUDE 'REX/GEN.';
% INCLUDE 'REX/GETUSERPARAMS.';
% INCLUDE 'REX/INITIALIZE.';

ON RECORD(P);
ON RECORD(R);
ON ENDFILE(R) BEGIN;
  CLOSE FILE(R);
  IF OPERATION > 0 THEN DO;
    CONFIGURE=ESC!!'RH00A00100'!!ESC!!'RC';
    WRITE FILE(R) FROM(CONFIGURE);
    READ FILE(R) INTO(MESSAGE);
  END;
  CLOSE FILE(F) ENV(LOCK);
  IF OPERATION = 0 THEN CLOSE FILE(P) ENV(LOCK);
  STOP;
END;
ON ERROR SIGNAL ENDFILE(R);

```

```

CALL INITIALIZE;

IF OPERATION > 0 THEN WRITE FILE(R) FROM(DC2ETX);
WRITE FILE(R) FROM(SCREEN30);
IF OPERATION > 0 THEN DO; READ FILE(P) INTO(STRUCT30);
EXIT30=ETX;
WRITE FILE(R) FROM(STRUCT30); END;

READ FILE(R) INTO(STRUCT30);
IF OPERATION ^= 0 THEN WRITE FILE(P) FROM(STRUCT30);

CALL GEN(3,'DCL DAGILIM FILE RECORD ENV(');
CALL GEN(9,'KIND='DISK',');
CALL FILEDEFINE(' Islenecek kutugun', 'IN', INPUTFILE);

CALL GEN(3,'DCL ISLE BIT(1);');
CALL GEN(3,'DCL ISLENEN FIXED(5);');
CALL GEN(3,'DCL ATLANAN FIXED(5);');
CALL GEN(3,'DCL DAGILIMADI CHAR(50) INIT(');
CALL GEN(9,'!!!DISTHEADING!!!');
CALL GEN(1,'% DCL SKIPTONEXT CHAR; !!!
          % SKIPTONEXT='ISLE='O'B',');

IF OPERATION > 0 THEN WRITE FILE(R) FROM(DC2ETX);
WRITE FILE(R) FROM(SCREEN31);
IF OPERATION > 0 THEN DO; READ FILE(P) INTO(STRUCT31);
EXIT31=ETX;
WRITE FILE(R) FROM(STRUCT31); END;

READ FILE(R) INTO(STRUCT31);
IF OPERATION ^= 0 THEN WRITE FILE(P) FROM(STRUCT31);
IF EXCEPT(' ',ROWVAR) = 'INDEX' THEN DO;
ROWVAR='GOSTERGE';
INDEXPTR=0;
END; ELSE
IF ROWTYPE = 2 THEN INDEXFLAG='1'B;
CALL GEN(3,'DCL SAYI('!!!ROWMIN!!!: '!!!ROWMAX!!!,');
DO X=COLLEVEL TO 1 BY -1;
SUBSTR(SCREEN31,273,18)='KOLON DUZEYI: '!!X;
SUBSTR(SCREEN31, 414,5)='Kolon';
SUBSTR(SCREEN31,1216,5)='Kolon';
SUBSTR(SCREEN31,1447,5)='Kolon';
IF INDEXFLAG THEN SUBSTR(SCREEN31,642,1)=BRIGHT;
IF OPERATION > 0 THEN WRITE FILE(R) FROM(DC2ETX);
WRITE FILE(R) FROM(SCREEN31);
IF OPERATION > 0 THEN DO; READ FILE(P) INTO(STRUCT32(X));
EXIT32(X)=ETX;
WRITE FILE(R) FROM(STRUCT32(X)); END;
READ FILE(R) INTO(STRUCT32(X));
IF OPERATION ^= 0 THEN WRITE FILE(P) FROM(STRUCT32(X));
IF EXCEPT(' ',COLVAR(X)) = 'INDEX' THEN DO;
COLVAR(X)='GOSTERGE';
INDEXPTR=X;
END; ELSE
IF COLTYPE(X) = 2 THEN INDEXFLAG='1'B;
IF X ^= 1 THEN
CALL GEN(12,COLMIN(X)!!!: '!!!COLMAX(X)!!!,');
ELSE DO;
CALL GEN(12,COLMIN(X)!!!: '!!!COLMAX(X)!!!

```

```

        ') FIXED(5) INIT(0);');
    COLCOUNT=1+COLMAX(X)-COLMIN(X);
END;
END;

CALL GEN(3,'ON ENDPAGE(SYSPRINT) BEGIN;');
CALL GEN(6,'PUT PAGE EDIT(DAGILIMADI) (COL(15),A);');
DO X=COLLEVEL TO 2 BY -1;
    IF COLVECTOR(X) <= ' ' THEN DO;
        CALL GEN(6,'PUT SKIP(2) EDIT(''!!!COLHEADING(X)!!
            ' : ',');
        CALL GEN(12,EXCEPT(' ',COLVECTOR(X))!!'(K!!!X!!!)');
        CALL GEN(18,'(COL(1),A,A)');
    END; ELSE
        CALL GEN(6,'PUT SKIP(2) EDIT(''!!!COLHEADING(X)!!
            ' : ',!!!'K!!!X!!!');
        CALL GEN(18,'(COL(1),A,P"ZZZZZ9")');
    END;
CALL GEN(6,'PUT SKIP(2) EDIT(''!!!COLHEADING(1)!!"')');
CALL GEN(18,'(COL(!!!12+5*COLCOUNT!!!),A)');
CALL GEN(6,'PUT SKIP EDIT(COPY("'-"',!!!COLCOUNT+10-2!!
    '))!!!'(COL(23),A)');
IF COLVECTOR(1) <= ' ' THEN DO;
    CALL GEN(6,'PUT SKIP EDIT(''!!!ROWHEADING!!!"');
    CALL GEN(12,'(!!!EXCEPT(' ',COLVECTOR(1))!!'(I)');
    CALL GEN(12,'DO I=!!!COLMIN(1)!!! TO !!!COLMAX(1)!!!,!!!
        !!! TOPLAM"')');
    CALL GEN(18,'(COL(1),A,COL(23),(!!!COLCOUNT!!
        ')(A(8),X(2))');
END; ELSE DO;
    CALL GEN(6,'PUT SKIP EDIT(''!!!ROWHEADING!!!"');
    CALL GEN(12,'(I DO I=!!!COLMIN(1)!!! TO !!!COLMAX(1)!!!
        '),!!!"TOPLAM"')');
    CALL GEN(18,'(COL(1),A,COL(24),(!!!COLCOUNT!!!)!!!
        '(P"ZZZZZ9",X(4)),A)');
END;
CALL GEN(6,'PUT SKIP EDIT((20)"'-!!!COPY(" '!!!
    (8)"'-"',!!!COLCOUNT+1!!!) (COL(1),A)');
CALL GEN(3,'END;');

CALL GEN(3,'ON ENDFILE(DAGILIM) BEGIN;');
COLSTR='';
DO X=COLLEVEL TO 2 BY -1;
    CALL GEN(6,'DO K!!!X!!!=!!!COLMIN(X)!!! TO !!!COLMAX(X)!!!
        ');
    COLSTR=COLSTR!!!'K!!!X!!!',';
END;
CALL GEN(6,'SIGNAL ENDPAGE(SYSPRINT);');
CALL GEN(6,'DO I=!!!ROWMIN!!! TO !!!ROWMAX!!!');
CALL GEN(9,'IF SUM(SAYI(I,!!!COLSTR!!!*)) > 0 THEN');
IF ROWVECTOR <= ' ' THEN DO;
    CALL GEN(12,'PUT SKIP EDIT('!!!EXCEPT(' ',ROWVECTOR)!!
        '(I),');
    CALL GEN(18,'(SAYI(I,!!!COLSTR!!!K1) DO K1=!!!COLMIN(1)!!!
        ' TO !!!COLMAX(1)!!!),');
    CALL GEN(18,'SUM(SAYI(I,!!!COLSTR!!!*)))');
    CALL GEN(24,'(COL(1),A(20),X(3),(!!!COLCOUNT+1!!!)');

```

```

CALL GEN(24,'(P'ZZZZZ9',X(4)));');
END; ELSE DO;
CALL GEN(12,'PUT SKIP EDIT(I,(SAYI(I,'!!COLSTR!!'K1)'));
CALL GEN(18,'DO K1='!!COLMIN(1)!!' TO '!!COLMAX(1)!!
'),'!!'SUM(SAYI(I,'!!COLSTR!!'*)))');
CALL GEN(24,'(COL(8),P'ZZZZZ9',X(10),(!!COLCOUNT+1!!
)'));
CALL GEN(24,'(P'ZZZZZ9',X(4)));');
END;
CALL GEN(6,'END;');
CALL GEN(6,'PUT SKIP(2) EDIT('TOPLAM',!!
'(SUM(SAYI(*,'!!COLSTR!!'K1)) ');
CALL GEN(12,'DO K1='!!COLMIN(1)!!' TO '!!COLMAX(1)!!'),'!!
'SUM(SAYI(*,'!!COLSTR!!'*)))');
CALL GEN(18,'(COL(8),A,X(10),(!!COLCOUNT+1!!)'')!!
'(P'ZZZZZ9',X(4)));');
DO X=COLLEVEL TO 2 BY -1;
CALL GEN(6,'END;');
END;
CALL GEN(6,'STOP;');
CALL GEN(3,'END;');

CALL COMMANDS(3,'BASLANGIC');

CALL GEN(3,'OPEN FILE(SYSPRINT) ENV(KIND='PRINTER',!!
'MAXRECSIZE=132,PAGESIZE=60)');');

CALL GEN(3,'DO WHILE('1'B);');
CALL GEN(6,'ISLE='1'B);');
CALL GEN(6,'READ FILE(DAGILIM) INTO(!!
EXCEPT(' ',INPUTREC)!!)');');
CALL COMMANDS(6,'YINLELEN');
CALL GEN(6,'IF ISLE THEN DO;');
CALL GEN(9,'ISLELEN=ISLELEN+1;');
IF INDEXPTR >= 0 THEN
  IF INDEXPTR = 0 THEN
    CALL GEN(9,'DO GOSTERGE='!!ROWMIN!!' TO '!!ROWMAX!!
    ');');
  ELSE DO;
    CALL GEN(9,'DO GOSTERGE='!!COLMIN(INDEXPTR)!!' TO '!!
    COLMAX(INDEXPTR)!!');');
  END;
DO W=1 TO 2;
  IF ROWTYPE = 2 THEN DO;
    CALL GEN(9,'SAYI('!!EXCEPT(' ',ROWVAR));
    CALL GEN(17,'(GOSTERGE),');
    DO X=COLLEVEL TO 1 BY -1;
      Y=',';
      IF X = 1 THEN IF W = 1 THEN Y=')='; ELSE Y=')+1';
      CALL GEN(14,EXCEPT(' ',COLVAR(X))!!Y);
    END;
  END; ELSE DO;
    CALL GEN(9,'SAYI('!!EXCEPT(' ',ROWVAR)!!',');
    DO X=COLLEVEL TO 1 BY -1;
      Y=',';
      IF X = 1 THEN IF W = 1 THEN Y=')='; ELSE Y=')+1';
      IF COLTYPE(X) = 2 THEN DO;

```

```
        CALL GEN(14,EXCEPT(' ',COLVAR(X)));
        CALL GEN(17,'(GOSTERGE)!!Y);
    END; ELSE
        CALL GEN(14,EXCEPT(' ',COLVAR(X))!!Y);
    END;
END;
END;
END;
IF INDEXFLAG THEN CALL GEN(9,'END;');
CALL GEN(6,'END; ELSE');
CALL GEN(9,'ATLANAN=ATLANAN+1;');
CALL GEN(3,'END;');

CALL GEN(0,'END;');
SIGNAL ENDFILE(R);

END DISTRIBUTION;
```



```

/*          - - - - - - - - - - - - - - - - - - - -          */
/*          O R N E K / C R E A T E / C A N D I D A T E          */
/*          - - - - - - - - - - - - - - - - - - - -          */
ORNEKCREATCANDIDATE:PROC OPTIONS(MAIN);
% INCLUDE 'ORNEK/LAYOUT/CANDIDATE.';
DCL YUKLENEN FILE RECORD ENV(
    KIND='DISK',
    MAXRECSIZE=00216,
    BLOCKSIZE =00216,
    FRAMESIZE =8,
    MYUSE='OUT',
    TITLE='ORNEK/CANDIDATE. ');
DCL ISLE BIT(1);
DCL ISLENEN  FIXED(5);
DCL ATLANAN  FIXED(5);
% DCL SKIPTONEXT CHAR; % SKIPTONEXT='ISLE='0'B';
DCL APP# PIC'(6)9'; APP#=0;
DCL NUM  PIC'(5)9' DEF APP# POS(1);
DCL N(6) PIC'9' DEF APP# POS(1);
NAME,FATHER=(12)' '; SURNAME=(14)' ';
SEX,BIRTH,CENTER=0; AREA(*)=0;
LINE1,LINE2=(23)' '; DISTRICT=(12)' '; PROVINCE=0;
REG_VALIDITY,EXAM_VALIDITY=0;
ROOM#(*),ROOM@(*),SEAT#(*)=0;
BOX#(*),BOXSEQ#(*)=0; ATTENDANCE(*)=0;
CORRECT(*,*),INCORRECT(*,*)=0;
RAW(*,*),STANDARD(*,*)=0; WEIGHTED(*)=0;
KUTUKSONU=00400;
DO SAYAC = 1 TO KUTUKSONU;
    ISLE='1'B;
    NUM=NUM+1;
    N(6)=9-MOD(N(1)*2+N(2)*3+N(3)*4+N(4)*5+N(5)*6,9);
    APPLICATION#=APP#;
    IF ISLE THEN DO;
        WRITE FILE(YUKLENEN) FROM(CANDIDATE_REC);
        ISLENEN=ISLENEN+1;
    END; ELSE
        ATLANAN=ATLANAN+1;
END;
CLOSE FILE(YUKLENEN) ENV(LOCK);
DISPLAY('YUKLENEN TUTANAK SAYISI='!!ISLENEN);
DISPLAY('ATLANAN TUTANAK SAYISI='!!ATLANAN);
END;

```



```

/*      - - - - -                                     */
/*      O R N E K / U P D A T E / C A N D I D A T E   */
/*      - - - - -                                     */
ORNEKUPDATECANDIDATE:PROC OPTIONS(MAIN);
% INCLUDE 'ORNEK/LAYOUT/CANDIDATE.';
% INCLUDE 'ORNEK/LAYOUT/REGOPTIC.';
DCL YUKLELENEN FILE RECORD ENV(
    KIND='DISK',
    MAXRECSIZE=00216,
    BLOCKSIZE =00216,
    FRAMESIZE =8,
    MYUSE='IO',
    TITLE='ORNEK/CANDIDATE. ');
DCL YUKLEYEN FILE RECORD ENV(
    KIND='DISK',
    MAXRECSIZE=122,
    BLOCKSIZE =122,
    FRAMESIZE =8,
    MYUSE='IN',
    TITLE='ORNEK/REGOPTIC. ');
DCL ISLE BIT(1);
DCL ERIS BIT(1);
DCL GECERSIZ FIXED(5);
DCL ISLENEN  FIXED(5);
DCL ATLANAN  FIXED(5);
DCL ILETI(21) CHAR(30) INIT(
    '>>>> BASVURU NO GECERSIZ <<<<',
    '>>>> ONCEDEN KAYITLI ADAY <<<<',
    '>>>> KAYITLI OLMAYAN ADAY <<<<',
    'ADI      : KODLANMAMIS      ',
    'ADI      : CIFT ISARET      ',
    'SOYADI   : KODLANMAMIS      ',
    'SOYADI   : CIFT ISARET      ',
    'BABA ADI : KODLANMAMIS      ',
    'BABA ADI : CIFT ISARET      ',
    'CINSIYET : GECERSIZ         ',
    'D.TARIHI : KODLANMAMIS      ',
    'D.TARIHI : CIFT ISARET      ',
    'SINAV M. : GECERSIZ         ',
    'ALAN KODU: GECERSIZ (1.OTURUM)',
    'ALAN KODU: GECERSIZ (2.OTURUM)',
    'ALAN KODU: GECERSIZ (3.OTURUM)',
    'ALAN KODU: GECERSIZ (4.OTURUM)',
    'ADRES    : KODLANMAMIS      ',
    'ADRES    : CIFT ISARET      ',
    'SEMT/ILCE: CIFT ISARET      ',
    'IL KODU  : GECERSIZ         ');
DCL ILETITUT(21) BIT(1);
% DCL INVALIDKEY CHAR; % INVALIDKEY='ERIS=''0''B';
% DCL SKIPTONEXT CHAR; % SKIPTONEXT='ISLE=''0''B';
% DCL MESSAGE CHAR; % MESSAGE='CALL UYARI';
ON ENDFILE(YUKLEYEN) BEGIN;
CLOSE FILE(YUKLELENEN) ENV(LOCK);
DISPLAY('YUKLELENEN TUTANAK SAYISI='!!ISLENEN);
DISPLAY('ATLANAN TUTANAK SAYISI='!!ATLANAN);
DISPLAY('GECERSIZ ANAHTAR SAYISI='!!GECERSIZ);
STOP;

```

```

END;
ON ENDPAGE(SYSPRINT) BEGIN;
  DCL (DATE,TIME) BUILTIN;
  DCL TARIH CHAR(6) INIT(DATE());
  DCL SAAT CHAR(9) INIT(TIME());
  SAYFA=SAYFA+1;
  PUT PAGE EDIT(
    (25)' '!!'K O N T R O L ' ,
    'L I S T E S I'!!(25)' ' ,
    'SAAT: ',SUBSTR(SAAT,1,2),':',
    SUBSTR(SAAT,3,2),':',SUBSTR(SAAT,5,2),
    ' TARIH: ',SUBSTR(TARIH,5,2),'/',
    SUBSTR(TARIH,3,2),'/19',SUBSTR(TARIH,1,2),
    ' SAYFA: ',SAYFA)
    (15(A),F(4));
  PUT SKIP(2) EDIT(
    'BAS.NO ADI..... ' ,
    'SOYADI..... BABA ADI.... ' ,
    'C D.TAR. SM A1 A2 A3 A4 ' ,
    'ADRES SATIRI-1..... ' ,
    'ADRES SATIRI-2.....' ,
    'SEMT/ILCE... IL' ,
    '')
    (SKIP( 1),
    COL( 1),
    (5)A,
    SKIP( 1),
    COL( 85),
    A,
    A);
END;
DCL APP# PIC'(6)9';
DCL KEY PIC'(5)9' DEF APP# POS(1);
DCL N(6) PIC'9' DEF APP# POS(1);
% INCLUDE 'REX/GETUSERPARAMS.';
CALL GETUSERPARAMS('ORNEK');
DISPLAY('ISLEM TURU(1:KAYIT,2:GUNLEME)') REPLY(FUNCTION);
OPEN FILE(SYSPRINT) ENV(KIND='PRINTER',
  MAXRECSIZE=132,PAGESIZE=60);
SIGNAL ENDPAGE(SYSPRINT);
DO WHILE('1'B);
  ILETITUT='0'B;
  ISLE='1'B;
  ERIS='1'B;
  READ FILE(YUKLEYEN) INTO(REGOPTIC_REC);
  APP#=OPT_APP#;
  IF N(6) = 9-MOD(N(1)*2+N(2)*3+N(3)*4+N(4)*5+N(5)*6,9)
  THEN DO;
    MESSAGE(1);
    INVALIDKEY;
  END; ELSE
    KEY=KEY-1;
  IF = ERIS THEN
    GECERSIZ=GECERSIZ+1;
  ELSE DO;
    READ FILE(YUKLELEN) INTO(CANDIDATE_REC)
    INDEX(KEY);

```

```

IF FUNCTION = 1 & REG_VALIDITY = 1 THEN DO;
  MESSAGE(2); SKIPTONEXT; END;
IF FUNCTION = 2 & REG_VALIDITY = 0 THEN DO;
  MESSAGE(3); SKIPTONEXT; END;
IF OPT_NAME = (12)' ' THEN MESSAGE(4); ELSE
IF INDEX(OPT_NAME,'*') > 0 THEN MESSAGE(5);
  ELSE NAME=OPT_NAME;
IF OPT_SURNAME = (14)' ' THEN MESSAGE(6); ELSE
IF INDEX(OPT_SURNAME,'*') > 0 THEN MESSAGE(7); ELSE
  SURNAME=OPT_SURNAME;
IF OPT_FATHER = (12)' ' THEN MESSAGE(8); ELSE
IF INDEX(OPT_FATHER,'*') > 0 THEN MESSAGE(9);
  ELSE FATHER=OPT_FATHER;
IF OPT_SEX ^= 1 & OPT_SEX ^= 2 THEN MESSAGE(10);
  ELSE SEX=OPT_SEX;
IF OPT_BIRTH = (6)' ' THEN MESSAGE(11); ELSE
IF INDEX(OPT_BIRTH,'*') > 0 THEN MESSAGE(12);
  ELSE BIRTH=OPT_BIRTH;

OK=1;
DO I=1 TO #OFCENTERS WHILE(OK);
  IF OPT_CENTER = CENTERS(I) THEN OK=0;
END;
  IF OK = 1 THEN MESSAGE(13);
  ELSE CENTER=OPT_CENTER;

DO I=1 TO 4;
  IF I <= #OFSESSIONS THEN
    IF OPT_AREA(I) > #OFAREAS(I)
      THEN MESSAGE(13+I);
      ELSE AREA(I)=OPT_AREA(I);
    ELSE OPT_AREA(I)=0;
  END;
IF OPT_LINE1 = (23)' ' &
OPT_LINE2 = (23)' ' THEN MESSAGE(18); ELSE
IF INDEX(OPT_LINE1,'*') > 0 !
INDEX(OPT_LINE2,'*') > 0 THEN MESSAGE(19); ELSE
  DO; LINE1=OPT_LINE1;
    LINE2=OPT_LINE2;
  END;
IF INDEX(OPT_DISTRICT,'*') > 0 THEN MESSAGE(20); ELSE
  DISTRICT=OPT_DISTRICT;
IF OPT_PROVINCE < '01' !
OPT_PROVINCE > '67' THEN MESSAGE(21); ELSE
  PROVINCE=OPT_PROVINCE;

REG_VALIDITY=1;
IF ISLE THEN DO;
  WRITE FILE(YUKLENEN) FROM(CANDIDATE_REC)
    INDEX(KEY);
  ISLENEN=ISLENEN+1;
END; ELSE
  ATLANAN=ATLANAN+1;
END;
PUT SKIP EDIT(
  (132)'-',
  REGOPTIC_REC,
  '')
  (COL( 1),
  A,

```

```

        SKIP(      1),
        COL(  1),
        P'(6)9',X(3),
        3(A,X(2)),
        P'9',X(2),A,
        5(X(2),P'99'),
        2(X(2),A),
        SKIP(      1),
        COL( 85),
        A,X(2),A,A,
        A);
    IF SUM(ILETITUT) > 0 THEN DO;
        PUT SKIP EDIT('ILET[LER: ') (COL(10),A);
        DO I=1 TO 21;
            IF ILETITUT(I) = '1'B THEN
                PUT EDIT(ILETI(I)) (COL(22),A);
        END;
    END;
END;
UYARI:PROC(P); ILETITUT(P)='1'B; END;
END;

```

```

/*          -----          */
/*          CENTER/SESSION/AREA DISTRIBUTION          */
/*          -----          */
ORNEKDISTRIBUTION:PROC OPTIONS(MAIN);
% INCLUDE 'ORNEK/LAYOUT/CANDIDATE.';
DCL DAGILIM FILE RECORD ENV(
    KIND='DISK',
    MAXRECSIZE=00216,
    BLOCKSIZE =00216,
    FRAMESIZE =8,
    MYUSE='IN',
    TITLE='ORNEK/CANDIDATE. ');
DCL ISLE BIT(1);
DCL ISLENEN FIXED(5);
DCL ATLANAN FIXED(5);
DCL DAGILIMADI CHAR(50) INIT(
    'MERKEZ/OTURUM/ALAN DAGILIMI');
% DCL SKIPTONEXT CHAR; % SKIPTONEXT='ISLE=' '0' 'B';
DCL SAYI(
    0: 5,
    1: 6,
    1: 2) FIXED(5) INIT(0);
ON ENDPAGE(SYSPRINT) BEGIN;
    PUT PAGE EDIT(DAGILIMADI) (COL(15),A);
    PUT SKIP(2) EDIT('          MERKEZ          : ',K2)
        (COL(1),A,P'ZZZZZ9');
    PUT SKIP(2) EDIT('          OTURUM          ')
        (COL(
            22),A);
    PUT SKIP EDIT(COPY('-',          18))(COL(23),A);
    PUT SKIP EDIT('          ALAN          ',
        (I DO I= 1 TO 2), 'TOPLAM')
        (COL(1),A,COL(24),(000002)(P'ZZZZZ9',X(4)),A);
    PUT SKIP EDIT((20)'-!!!COPY(' !!!(8)'-',          3))
        (COL(1),A);
END;
ON ENDFILE(DAGILIM) BEGIN;
    DO K2= 1 TO 6;
    SIGNAL ENDPAGE(SYSPRINT);
    DO I= 0 TO 5;
        IF SUM(SAYI(I,K2,*)) > 0 THEN
            PUT SKIP EDIT(I,(SAYI(I,K2,K1)
                DO K1= 1 TO 2),SUM(SAYI(I,K2,*)))
                (COL(8),P'ZZZZZ9',X(10),(          3)
                (P'ZZZZZ9',X(4)));
            END;
            PUT SKIP(2) EDIT('TOPLAM',(SUM(SAYI(*,K2,K1))
                DO K1= 1 TO 2),SUM(SAYI(*,K2,*)))
                (COL(8),A,X(10),(          3)(P'ZZZZZ9',X(4)));
            END;
    STOP;
END;
OPEN FILE(SYSPRINT) ENV(KIND='PRINTER',MAXRECSIZE=132,
    PAGESIZE=60);
DO WHILE('1'B);
    ISLE='1'B;
    READ FILE(DAGILIM) INTO(CANDIDATE_REC);
    IF REG_VALIDITY ^= 1 THEN SKIPTONEXT;
    IF ISLE THEN DO;

```

```
ISLENEN=ISLENEN+1;
DO GOSTERGE= 1 TO 2;
  SAYI(AREA
    (GOSTERGE),
    CENTER,
    GOSTERGE)=
  SAYI(AREA
    (GOSTERGE),
    CENTER,
    GOSTERGE)+1;
END;
END; ELSE
  ATLANAN=ATLANAN+1;
END;
END;
```



APPENDIX F



## K O N T R O L L İ S T E S İ

SAAT: 23:40:53 TARİH: 11/09/1087 SAYFA: 1

BAS.NO	ADI.....	SOYADI.....	BABA ADI....	C	D.TAR.	SM	A1	A2	A3	A4	ADRES SATIRI-1.....	ADRES SATIRI-2.....
000606	İSMAIL	ŞEKER	HASAN	1	*0957	06	01	03	00	00	SEMT/İLCE... MERKEZ SAĞLIK OCAĞI LOJ ŞUBUK 06.	
	İLETİLEP:	D.TARİHİ :	CİFT İSAPET									
000073	MEVLUT	ASILAYAN	RIFAT	1	010150	06	04	03	00	00	KOCAMUSTAFAPAŞA C.NO:94 /3 CERRAHPAŞA 34.	
000456	CEZMİ TÜRK	TIRAŞ	CIHAT	1	010152	06	01	04	00	00	EMEK MAHALLESİ 83.SOKAK 2/6 06.	
	İLETİLER:	>>>>	BASVURU NO GECERSİZ <<<<									
000471	HASAN	KUŞUKŞE	YAHYA	1	010152	06	03	02	00	00	8 SOKAK NO:6 DAİRE:8 BAHCELİEVLER 06.	ÖYMEN APT
002304	ÜŞLER	KISA	NECİP	0	010156	16	04	06	00	00	ŞİİLTEPE SB LOJMANLARI MAMAK 06.	TEYMUR AP NO:0
	İLETİLER:	CİNSİYET :	GEÇERSİZ									
		SINAV M. :	GEÇERSİZ									
		ALAN KODU:	GEÇERSİZ (2.OTURUM)									
000151	ERDAL	COŞKUN	MEHMET ZEK*	1	010157	06	04	01	00	00	AKAT SOKAK UĞUR APT.:18/ CEBEÇİ 06.	
	İLETİLEP:	BABA ADI :	CİFT İSARET									
001905	SABAHA	ÜNLÜ	ALİ	2	010159	01	01	02	00	00	BAGLAR CADDESİ HACIYOLU 55/6 Ş.BAGLARI 06.	
001095	HÜLYA	ATALAY	MUSTAFA	2	010159	01	02	03	00	00	BAHCELİEVLER MAHALLESİ YALOVA 34.	KOŞUYOLU CAĞIRIŞI NO:14*
	İLETİLEP:	ADRES :	CİFT İSARET									
000532	HASAN	ÖZKAN	AHMET	1	010160	06	01	03	00	00	BILLUR SOKAK 9/15 KAVAKLIDERE 06.	
000575	SONGÜL	AKKAŞ	AHMET	2	010161	01	04	03	00	00	MERUTİYET C.32/11 KIZILAY 06.	
002961	ATILLA	KAYA	MAHMET	1	010161	06	01	03	00	00	KUTLUGÜN SOKAK NO:18/1 İÇEBECİ 06.	
002533	TAHIR	ÇAKIR	MEHMET	1	010254	06	02	03	00	00	SAMUR SOKAK NO 23/14 KURTULUŞ 06.	



MERKEZ/OTURUM/ALAN DAGILIMI

MERKEZ : 1

OTURUM

ALAN	1	2	TOPLAM
1	31	25	56
2	24	26	50
3	48	75	123
4	38	6	44
5	0	9	9
TOPLAM	141	141	282

MERKEZ/OTURUM/ALAN DAGILIMI

MERKEZ : 6

OTURUM

ALAN	1	2	TOPLAM
1	48	37	85
2	29	38	67
3	71	80	151
4	19	8	27
5	0	4	4
TOPLAM	167	167	334



ORNEK KULLANICI SISTEMI  
SINAVA GIRIS BELGESI

Aday Numarası : 000181  
Soyadı, Adı : MUTLU  
Alanı :

ÇİĞDEM

Sınava gireceği

İl : ANKARA  
Bina : A.Ü.ZİRAAT FAK.  
Salon : DIŞKAPI  
MUHLİS KÜTÜPHANESİ  
ÜST SALON

Sınav Tarihi

Saati

Sınav Salon Numarası

Sıra Numarası

14/09/87

14:00

12339

037

ÇİĞDEM MUTLU  
BAŞCAVUŞ S 89/7  
KÜÇÜKESAT  
ANKARA



ORNEK KULLANICI SISTEMI  
SINAV SONUC BELGESI

Aday Numarası : 000208

Soyadı, Adı : KARAASLAN ŞULE

1. OTURUM PUANI: 46.957

2. OTURUM PUANI: 97.368

ŞULE KARAASLAN  
MENEKŞE SOKAK NO:32/6

KIZILAY ANKARA

## ORNEK KULLANICI SISTEMI


## SIRALI ADAY LISTESI

BAS.NO	ADI	SOYADI	1. OTURUM PUANI	2. OTURUM PUANI
001857	BEYHAN	ABAY	92.284	44.267
000757	NİLÜFER	ADAL	44.631	65.766
000822	SEYDAN	ADALI	98.941	93.501
002273	ERDAL	AFACAN	66.335	57.563
000761	VİLDAN	AĞA	79.556	7.377
000329	HİKMET FATİH	AĞALAR	62.779	98.137
000562	RIFAT MURAT	AKAL	99.911	64.916
000865	BİLGEHAN	AKALIN	8.837	71.103
001948	ŞENEL	AKGÖZ	40.605	20.921
002243	MEHMET KAZIM	AKGÜL	56.009	8.420
003023	ALİ HAN	AKIN	94.740	5.409
000575	SONGÜL	AKKAŞ	86.837	62.265
003079	ÖMER TAYLAN	AKKAYA	33.467	31.741
001524	MÜBECCEL	AKSU	65.412	2.782
002957	MELEK	AKTAŞ	47.067	45.811
002503	MUSTAFA K	ALADAĞLI	29.166	66.736
001325	AZMİYE	ALAN	32.613	57.789
002148	HÜSNAVERÖMÜR	ALEV	47.642	75.050
002559	RECEPSERDAR	ALPAN	30.916	5.985
001571	MERAL	ALPASLAN	17.812	19.572
001082	ALİ OĞUZ	ALTER	79.189	86.928
002031	PERİHAN	ALTUN	39.029	21.557
001537	AZİZ	ARSLAN	81.484	90.844
002394	OSMAN	ARSLAN	85.840	48.277
000835	ŞEMSİ	ARSLAN	81.105	20.066
000073	MEVLUT	AŞILAYAN	31.852	16.220
001095	HÜLYA	ATALAY	60.653	16.816
002901	MUSTAFA	ATALAY	27.492	85.564
002239	AYŞE	AVŞAR	36.647	84.357
001507	ZEYNEP SERAP	AYKUT	43.667	82.253
001186	DEVLET S	AYRANCI	4.449	71.728
002853	SEDAT	AZAK	11.545	1.258
002589	MEHMET TUFAN	BABAYİĞİT	79.000	58.114
000999	MESTAN UĞUR	BAŞIŞLAR	21.115	51.072
001978	ÜMMÜHAN Ş	BAHAR	10.021	20.536
003019	MEHMET	BAKAR	20.197	67.022
001753	ŞÜKRÜ	BAKIOĞLU	4.518	67.012
000285	ESRA	BAKTİROĞLU	1.362	27.790
001416	NEJLA	BALCI	40.181	96.925
000818	KAMİL CENGİZ	BALİ	3.089	66.511
002014	İLKNUR NEJLA	BAŞARAN	81.824	72.846
002546	NERMİN	BAŞOL	59.590	42.642
002563	FİGEN	BATIOĞLU	0.431	8.445
002152	AHMET	BAYRAKÇIOĞLU	98.261	55.212
000939	CAHİT	BAYRAKTAROĞLU	53.628	17.278
001355	YAVUZ	BENLİ	81.571	9.813
000515	MUSTAFA	BERKER	5.593	24.820
002091	AYDIN	BEYDİLLİ	47.028	78.751
001463	FATİH	BİYİK	90.266	29.238
000909	AHMET	BİLGE	12.759	96.508
000134	ILMAY	BİLGE	13.037	62.199
002321	MEHMET	BİLGİN	34.949	98.875
001476	MERAL	BİLGİN	37.093	82.934
001891	ŞEVKET	BİLİCİ	92.175	90.521
002135	CAN	BİRİNCİ	50.345	6.319
002182	LEYLA NUR	BOZAY	79.946	51.470


## ORNEK KULLANICI SISTEMI

## GOREVLI BILDIRIM CIZELGESI (OTURUM:2)

	SALON/BINA NUMARASI,ADI	KAPASITE	GCREV TURU	UNV.	GOREVLININ ADI,SOYADI	CALISTIGI KURUM
30	06001 A.Ü.ZİRAAT FAK. DIŞKAPI	6 SALON	BINA SINAV SORUMLUSU	1	ZEKİ HAFIZOĞULLARI	AÜ HUKUK FAK.
31	06001 A.Ü.ZİRAAT FAK. DIŞKAPI	6 SALON	BINA SINAV SOR. YRD.	16	TURAN ATEŞ	AÜ HUKUK FAK.
32	06001 A.Ü.ZİRAAT FAK. DIŞKAPI	6 SALON	BINA YONETICISI	9	HALUK KONURALP	AÜ HUKUK FAK.
33	06001 A.Ü.ZİRAAT FAK. DIŞKAPI	6 SALON	YEDEK GCZETMEN	10	CÜNEYT ÖZANSOY	AÜ HUKUK FAK.
34	06001 A.Ü.ZİRAAT FAK. DIŞKAPI	6 SALON	YEDEK GCZETMEN	10	CİVAN TURMANGİL	AÜ HUKUK FAK.
35	06001 A.Ü.ZİRAAT FAK. DIŞKAPI	6 SALON	YEDEK GCZETMEN	19	M ALPER AYVAZ	KURTULUŞ LİSESİ
36	06001 A.Ü.ZİRAAT FAK. DIŞKAPI	6 SALON	YEDEK GCZETMEN	5	İRFAN YAZMAN	AÜ HUKUK FAK.
37	06001 A.Ü.ZİRAAT FAK. DIŞKAPI	6 SALON	YEDEK GCZETMEN	10	SÜHA TANRIVER	AÜ HUKUK FAK.
38	06001 A.Ü.ZİRAAT FAK. DIŞKAPI	6 SALON	YEDEK GCZETMEN	10	AHMET MERİH ÖZEN	AÜ HUKUK FAK.
39	12337 DERSANE 5 NOLU	30 ADAY	SALON BASKANI	5	NERPİN BERKİ	AÜ HUKUK FAK.
40	12337 DERSANE 5 NOLU	30 ADAY	GCZETMEN	19	HALİL ÇATABAS	KURTULUŞ LİSESİ
41	12337 DERSANE 5 NOLU	30 ADAY	GCZETMEN	10	MUHARREM ÖZEN	AÜ HUKUK FAK.
42	12338 MUHLİS KÜTÜPHANESİ ALT SALON	30 ADAY	SALON BASKANI	10	TÜRKAN YALÇIN	AÜ HUKUK FAK.
43	12338 MUHLİS KÜTÜPHANESİ ALT SALON	30 ADAY	GCZETMEN	19	MÜBERRA YILMAZ	KURTULUŞ LİSESİ
44	12338 MUHLİS KÜTÜPHANESİ ALT SALON	30 ADAY	GCZETMEN	8	NECDET KÖKSAL	ODTU MADEN MÜH.

ÖSYM ÖSYM ÖSYM ÖSYM ÖSYM ÖSYM ÖSYM ÖSYM ÖSYM ÖSYM ÖSYM ÖSYM ÖSYM	
 <b>ÖSYM</b> YÜKSEKÖĞRETİM KURULU ULUSLARARASI ÖSYM MERKEZİ ULUSLARARASI ÖSYM MERKEZİ	ORNEK <b>GÖREVLENDİRME BELGESİ</b> ORNEK NO. <input type="text" value="0025"/>
SAYIN <b>05 HAN</b> <b>BÜYÜKALACA</b> <b>14/09/87 GÜNÜ, SAAT 14:00 DAKI CRNEK KULLANICI SİSTEMİ</b> SINAVINDA AŞAĞIDA BELİRTİLEN	
<input type="text" value="10011"/> NUMARALI <b>13 KISILIK SALONDA GCZETMEN</b> OLARAK GÖREVLENDİRİLDİĞİNİZİ BİLDİRİR, BASARILAR DİLERİM. SAYGILARIMLA, GÖREV YERİNİZ :	
BINA <b>ÇUKUROVA ÜNİVERSİTESİ ANFİLER GRUBU</b> <b>BALCALI KAMPÜSÜ</b> ADANA	
ÖSYM ÖSYM ÖSYM ÖSYM ÖSYM ÖSYM ÖSYM ÖSYM ÖSYM ÖSYM ÖSYM ÖSYM ÖSYM	

Not : Belgenin öbür yüzündeki açıklamaları okuyunuz!

ÖSYM ÖSYM ÖSYM ÖSYM ÖSYM ÖSYM ÖSYM ÖSYM ÖSYM ÖSYM ÖSYM ÖSYM ÖSYM	
 <b>ÖSYM</b> YÜKSEKÖĞRETİM KURULU ULUSLARARASI ÖSYM MERKEZİ ULUSLARARASI ÖSYM MERKEZİ	CRNEK <b>GÖREVLENDİRME BELGESİ</b> GÖREV NO. <input type="text" value="0026"/>
SAYIN <b>ZAFER</b> <b>İLBARS</b> <b>14/09/87 GÜNÜ, SAAT 14:00 DAKI CRNEK KULLANICI SİSTEMİ</b> SINAVINDA AŞAĞIDA BELİRTİLEN	
<input type="text" value="06001"/> NUMARALI <b>6 SALONLU BINADA BINA SINAV SORUMLUSU</b> OLARAK GÖREVLENDİRİLDİĞİNİZİ BİLDİRİR, BASARILAR DİLERİM. SAYGILARIMLA, GÖREV YERİNİZ :	
BINA <b>A.Ü. ZİRAAT FAK.</b> <b>DEŞKAPI</b> ANKARA	
ÖSYM ÖSYM ÖSYM ÖSYM ÖSYM ÖSYM ÖSYM ÖSYM ÖSYM ÖSYM ÖSYM ÖSYM ÖSYM	

Not : Belgenin öbür yüzündeki açıklamaları okuyunuz!



ÖRNEK

GÖREV KARTI

14/09/87 14:00

Adı Soyadı : ORHAN

BÜYÜKALACA

Görev No. 0025

Görevi GÖZETMEN

Sınav İli ADANA

METEKSAN 209-12-10381

ÖSYM-BİB / UYGULAMA-FORM: 87 / 525



ÖRNEK

GÖREV KARTI

14/09/87 14:00

Adı Soyadı : ZAFER

İLBARS

Görev No. 0026

Görevi BINA SINAV SORUMLUSU

Sınav İli ANKARA

METEKSAN 209-12-10381

ÖSYM-BİB / UYGULAMA-FORM: 87 / 525

Y. Ö.  
Yükseköğretim Kurulu Başkanlığı  
Öğrenci Seçme ve Yerleştirme Merkezi