NEAR EAST UNIVERSITY

Faculty Of Engineering

Department of Computer Engineering

STUDENT TRACKING SYSTEM USING DELPHI PROGRAMMING

Graduation Project

COM-400

Student: Zişan Çavuşoğlu

Supervisor: Assoc.Prof.Dr.Rahib Abiyev

Nicosia-2003

ACKNOWLEDGEMENT

I am greatly indebted to my supervisor Assoc Prof. Dr. Rahib Abiyev and never forget the scarifying job that did in teaching and guiding me through his courses and work on this project.

I would like to thank my family for their endless support from the first day they I've started my high education life till today. I will never forget the things that my friend Ms. Ayşegül Yılmaz did for me during the last four years. Her unique friendship and good memories of sharing the same house with her made she very special person in my life indeed.

My sincere thanks to Mr. Kenan Aksoy my friend who helped me in realising applied Delphi programming, Prof. Dr. Fakhreddin Mamedov the Dean of Engineering Faculty, all teaching staff, specially Mr. Tayseer and Ms. Filiz Alashnableh and finally to Mr. Mehrdad Khaledi for all they did for us.

I promise to do my best to be an honourable ambassador for my teachers and Near East University in future.

ABSTRACT

This aim of this project was to prepare a suitable registration program. The program was prepared by using Delphi and normally consists of so many menus. The main menu of the program is designed for login of four different groups who are listed under the "User Group" title and authorised to reach to student data. These are Admin, Advisor, Secretary and Accountancy. An individual who is working in any of these groups, can login to the program by using a predefined password. After login there will be a main-form, which has four subtitles as Student, Definitions, Shut Down and Exit. The authority of the users to reach, do changes and update the information in this program is limited with respect to the position and the relation of the people who are working with the data. For instant a person who works and responsible for Accountancy group has nothing to do with defining new courses or academic records of students. Meanwhile the secretary can not change the grades of the students till it is not approved by administration. These are simply expressing how the program was designed to use in a proper and secure way. The program provides the main personal details such as name, photo, the admission date and more about students. Additionally the disciplinary situation, Academic semesters, which they were enrolled, courses they have taken and their payments and their instalments are available in different screens of the program.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

INTRODUCTION

As a registration program is necessary but different for all education institutes, in the project it was aimed to write a program considering the problems that we were faced till today in our university. The main structure of the program was designed to apply to the registration process in all faculties and not only the engineering school. The program is user friendly and very simply adaptable to different education institutes with simple changes. Using the enormous advantages of Delphi program gives the chance to update this code in future due to academic needs. In the following chapters the main structures and menus of the program are explained in details and finally the source code of the program is presented

CHAPTER ONE:

STUDENT TRACKING SYSYTEM INFORMATION

1.1.Student Tracking Program Main Structure

In all universities and higher education institutes there is a need for a registration program which is suitable for their system and facilities of that organisation. Accordingly a common program that directly responds to registration formalities in all of these type organisations can not be prepared easily.

Although there are some common features that can be defined for all students, advisors and administrators in education system that can be noticed in preparing the main flowchart of type programs. In this chapter it was tried to explain some of these common features for a private university very likely adaptable to our university.

1.2.Explanation of Main steps in Student Registration

In a private university the first step of registration for any student is tuition fee payment at the beginning of an academic semester. After payment is done and if there is a special case such as scholarship etc. the student would be able to apply to the department that he/she is enrolled and meet the academic advisor.

Consequently if it done manually a paper is going with student showing the proof of registration or the computer program should give a chance to the advisor to check whether the students did pay for that specific semester or not. This is necessary for the secretaries and other administrative authorities of that department however if not a computerised, this would take time so the related forms and papers pass through all these people. The next step and very important one is course registration and update of academic records of the student.

1.3.Course Registration

After the advisor observes the proof of registration the previous academic records of student are necessary so new courses can be offered for the new academic semester. Consequently the whole details about the courses that the student failed or passed and additionally the GPA and CGPA of the student should be ready when the details are analysed by the advisor.

Any student is able to try a course for a short period at the beginning of the academic semester and if face any problem there is a procedure that is called ADD/DROP for changing the course. The advisor then should be able to apply this procedure for any student at the permitted days. It is simply the change of the courses that student does not want to continue with the new one. It is very important that advisor should definitely be able to see all courses which are opened in that semester. The calculations of GPA and CGPA for any university are due to same principles but different mathematically as the letter grades and their multiplying factors for them and additionally the credit of the courses may be different from university to university. In a computerised registration system this can be designed simply by a multi-user program so the information could be upload and download and a data base which is able to keep very large amount of the records about the students, courses, calculations and similar important data. There is no doubt that comparison of a good program with using only human power for this stage of registration is not only logical but also waist of time.

1.4.End of the Semester Procedures and

The procedure applied at the end of the semester is normally upload the letter grades of the students in all courses and calculation their GPA and CGPA. The Registration office of the university indeed will be informed about these and there is a short period that gives the chance for grade changing and considering the objections of the students.

1.5.Why a Data Base Program is Necessary?

Doing all explained in previous sections would very long and not effective if a proper program is not using by all units of higher education institute in harmony. Normally the number students per advisors in a best and reliable system can not be less than 10 so

keeping their records and calculations of GPA and CGPA would be take a long time. The advantage of using a proper program at the end of the semester is that, it may give a chance to analyse the academic situation of the student and offer some courses that can be taken by at the following semester. An advanced registration program the Time-Table of the courses taken by the students are also automatically prepared so at the beginning the students would be aware about the clashes and conflict between the timing of the lectures and laboratories. For all these purposes and desired features then a good visual programming language should be selected. Delphi with its significant language features would be the best choice as this language is contains tools to make programming for Windows easy. Delphi code is compiled; therefore, the compiled code runs quick. It is object oriented so objects keep the simple, organised and protected. It is very easy to read and well structured. One can actually have their code easily & efficiently proofed by a third party.

Delphi's editor works and the component system is efficient and easy to use. Delphi is single platform. While this may not be strength in some arguments, it means that (if you're developing for windows) the tools are very mature & uniquely suited to the job at hand. Delphi is very fast. . Not only is the generated code very tight, but the compiler is orders of magnitude faster then most compilers. Considering all of these advantages it can be very easy to create such a program for registration purpose in a higher education.

CHAPTER TWO: DATABASE STRUCTURE

2.1. General Informatics' Structure

General structure of the program is given in figure 2.1. As is shown program mainly contain sub problems: Student and definition.

In student sub menu the registration of students, courses, payments are considered.

Definition sub menu includes definition of penalty,educator,payment,course, department and faculty

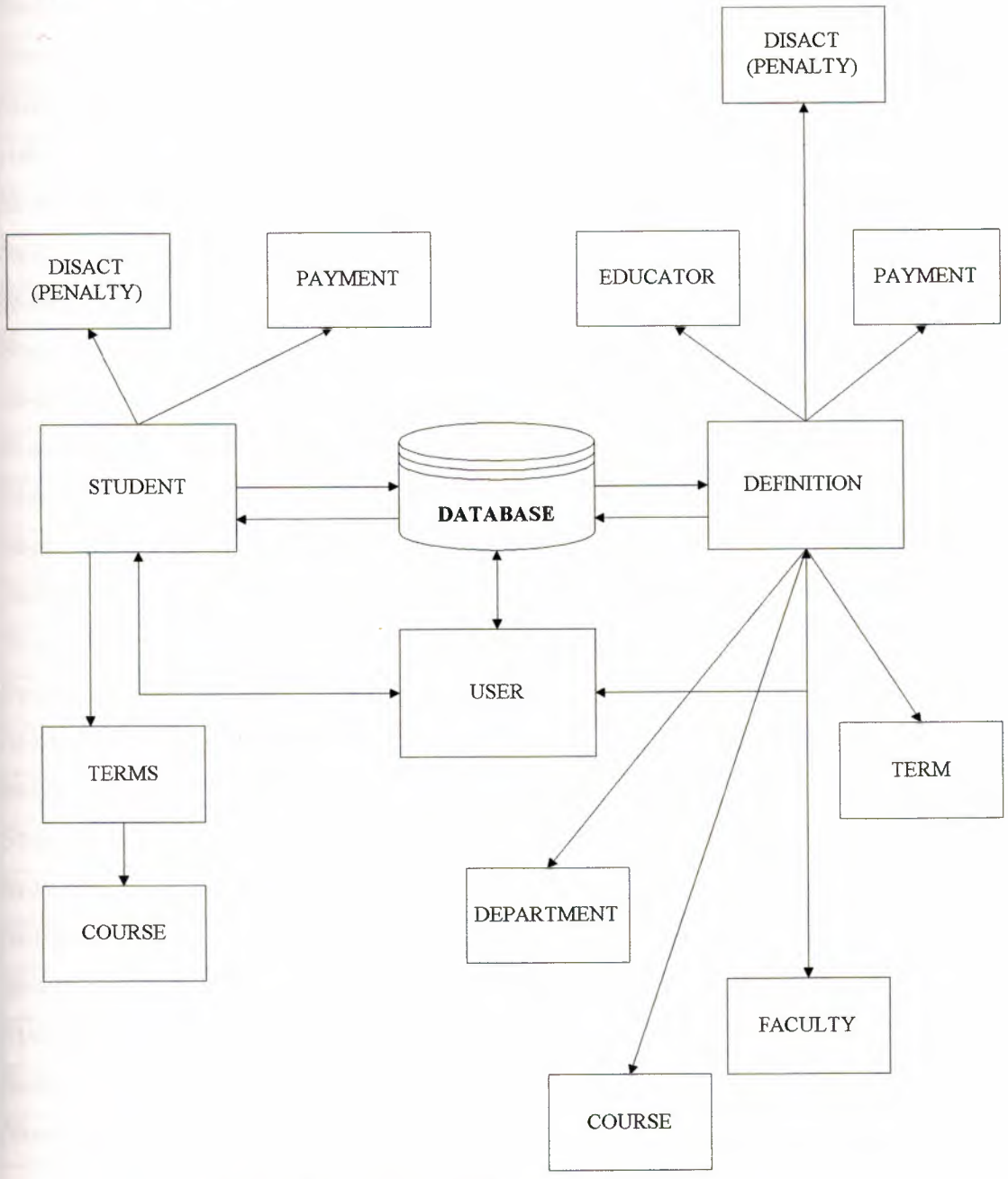


Figure 2.1. General Informatics' Structure

2.2. Database Structure

Program Database includes eighteen tables. Some tables are given below.

Student database table includes about the information student registration.

STUDENT.DB			
Field Name	**Type**	**Size**	**Key**
Id	+		*
St-photo	O		
St-id	A	10	*
St-firstname	A	10	
St-midname	A	20	
St-surname	D		
Admissiondate	A	6	
St-sex	A	20	
St-fathername	A	20	
St-mothername	A	15	
St-placeofbirth	D		
St-dateofbirth	A	50	
St-newaddr	A	50	
St-oldaddr	A	15	
St-pphone	A	15	
St-mphone	A	15	
St-country	A	15	
St-province	A	20	
St-nationality	A	25	
St-highschool	D		
St-gradeofdateH	A	25	
St-Hbranch	N		
St-Hgpa	A	30	
St-e-mail	A	20	
St-dept	A	35	
St-faculty	A	35	
Numberid	N		

Table2.1 :Student Database Table

Course.db			
Field Name	**Type**	**Size**	**Key**
Id	+		*
C-DEPTCODE	A	15	*
C-CODE	A	7	
C-CREDIT	N		
C-NAME	A	20	
C-REF	A	7	
C-YEAR	A	15	
C-TYPE	A	5	
C-CONTENT	A	20	
C-TEACHER	A	25	
C-DEPTNAME	A	35	
C-FACULTY	A	35	

Table2.2 : Course Database Table

Stcourse.db			
Field Name	**Type**	**Size**	**Key**
Tid	N		*
Id	+		*
Stid	A	10	
Ccode	A	7	
Cname	A	20	
Ccredit	N		
Tname	A	20	
Tsurname	A	25	
Tdeptcode	A	15	
Tdeptname	A	35	
Tfacultycode	N	15	
Tfacultyname	A	35	

Table2.3:Student-Course Database Table

Stcograde.db			
Field Name	**Type**	**Size**	**Key**
Cid	N		*
Id	+		*
Stid	A	10	
Grade	N		
GradeA	A	2	
Explanition	A		
Avarage	N		

Table2.4 :Student-Course Grade Database Table

Educator.db			
Field Name	**Type**	**Size**	**Key**
ID	+		*
E-NAME	A	20	*
E-SURNAME	A	25	
E-DEPT	A	35	
E-FUNCTION	A	25	
E-COUNTRY	A	15	
E-CITY	A	20	
E-AGE	N		
E-PPHONE	A	15	
E-MPHONE	*A*	*15*	
E-GENDER	A	6	
E-FACULTY	A	35	

Table2.5: Educator Database Table

Educourse.db			
Field Name	**Type**	**Size**	**Key**
Educatorid	N		*
Id	+		*
Educatorname	A	20	
Educatorsurname	A	25	
Facultycode	A	15	
Facultyname	A	35	
Deptcode	A	15	
Deptname	A	35	
Coursecode	A	7	
Couresename	A	20	
Tid	N		
Tyear	A	15	
Ccredit	N		

Table2.6: Educator-Course Database Table

Teach.db			
Field Name	**Type**	**Size**	**Key**
ID	+		*
T-NAME	A	20	*
T-SURNAME	A	25	
T-DEPT	A	35	
T-FUNCTION	A	25	
T-COUNTRY	A	15	
T-CITY	A	20	
T-AGE	N		
T-PHONENO	A	15	

Table2.7: Teacher Database Table

groups.db			
Field Name	**Type**	**Size**	**Key**
Id	+		
Groupname	A	25	
Explanation	A	50	

Table2.8 :Groups Database Table

USERS.db			
Field Name	**Type**	**Size**	**Key**
ID	+		
USERNAME	A	25	
PASSWORD	A	10	
GROUPNAME	A	15	

Table2.9. Users Database Table

Faculty.db			
Field Name	**Type**	**Size**	**Key**
Id	+		
Code	A	15	
Name	A	30	

Table2.10: Faculty Database Table

Dept.db			
Field Name	**Type**	**Size**	**Key**
Id	+		
Code	A	15	
Name	A	35	
Fcode	A	15	
Fname	A	35	

Table2.11. Department Database Table

Terms.db			
Field Name	**Type**	**Size**	**Key**
ID	+		
T-ID	I		
T-YEAR	A	15	
T-BDATE	D		
T-EDATE	D		
C-CODE	A	7	

Table2.12. Terms Database Table

Termact.db			
Field Name	**Type**	**Size**	**Key**
Stid	N		*
Id	+		*
Tid	N		
Tyear	A	15	
Tbdate	D		
Tedate	D		

Table2.13. Termact Database Table

Disact.db			
Field Name	**Type**	**Size**	**Key**
Sid	N		*
Id	+		*
Ddate	D		
Dcode	A	15	
Dname	A	35	
Explanation	A	90	
Result	A	50	

Table2.14.Disact(Penalty) Database Table

Disactdef.db			
Field Name	**Type**	**Size**	**Key**
Id		+	
Dcode		A	15
Dname		A	30

Table2.15.Disact (Penalty) Database Table

Paydefinition.db			
Field Name	**Type**	**Size**	**Key**
Id	+		
Pcode	A	10	
Pname	A	30	

Table2.16. Payment Definition Database Table

PAYMENT.DB			
Field Name	**Type**	**Size**	**Key**
Id	+		*
Pcode	A	10	*
Sid	A	10	*
Pname	A	30	
Pdate	D		
Inscount	N		
Pamount	N		
Paid	A	3	
Tax	N		

Table2.17.Payment Database Table

Pdetail.db			
Field Name	**Type**	**Size**	**Key**
Id	+		*
Payid	N		*
Pamount	N		
Ppamount	N		
Tax	N		
Pdate	D		
Ppdate	D		
Ptype	A	15	
Pdelay	N		
Paid	A	3	

Table2.18.Payment Detail Database Table

Relationship Between the Tables

eighteen tables. The structures and relation between tables are given in

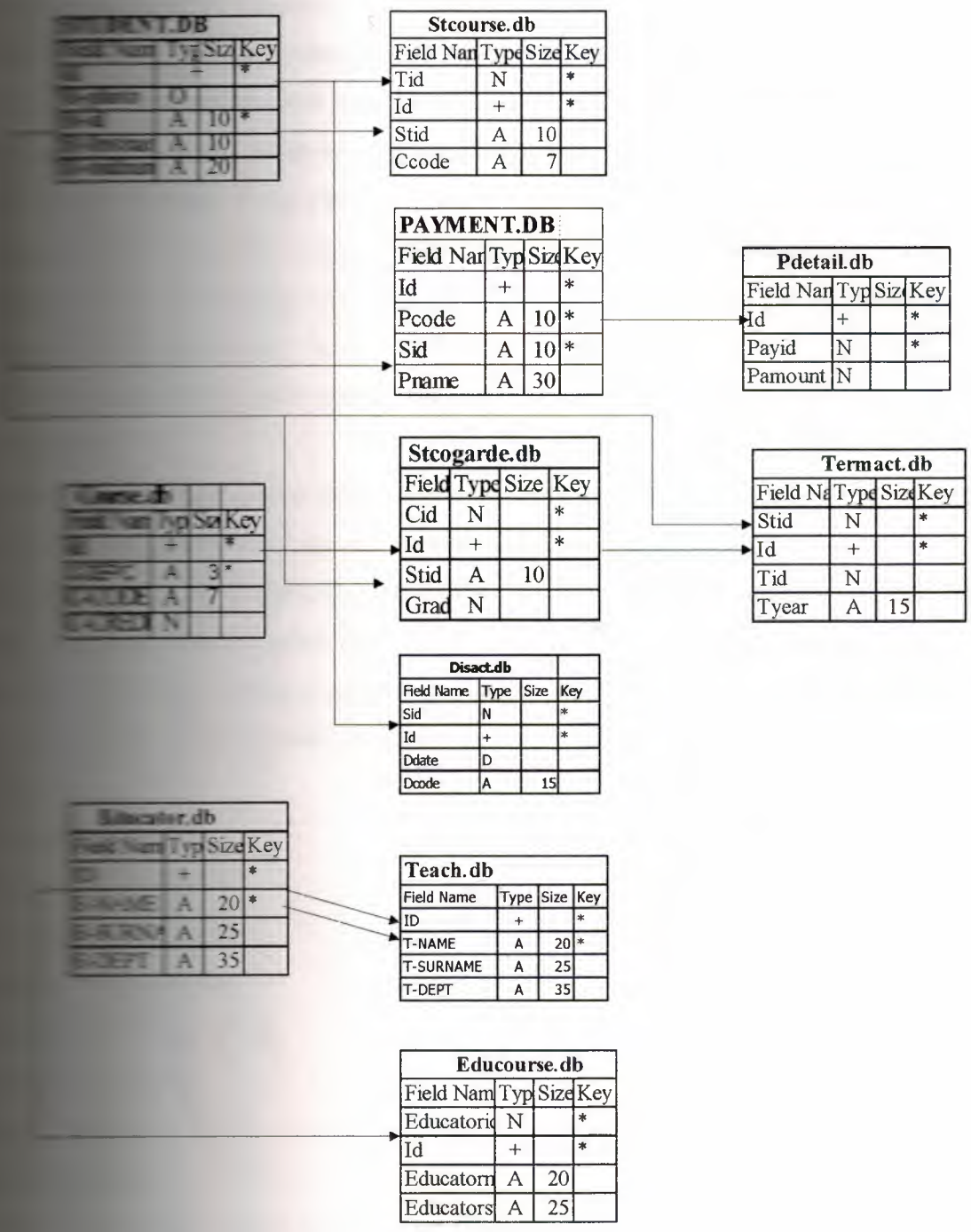


Figure 2.2 Relationship Between the Tables

2.4.Working with SQL

SQL (pronounced "ess-que-el") stands for Structured Query Language. SQL is used to communicate with a database. According to ANSI (American National Standards Institute), it is the standard language for relational database management systems. SQL statements are used to perform tasks such as update data on a database, or retrieve data from a database. Some common relational database management systems that use SQL are: Oracle, Sybase, Microsoft SQL Server, Access, Ingress, etc. Although most database systems use SQL, most of them also have their own additional proprietary extensions that are usually only used on their system. However, the standard SQL commands such as "Select", "Insert", "Update", "Delete", "Create", and "Drop" can be used to accomplish almost everything that one needs to do with a database.

Table Basics

A relational database system contains one or more objects called tables. The data or information for the database are stored in these tables. Tables are uniquely identified by their names and are comprised of columns and rows. Columns contain the column name, data type, and any other attributes for the column. Rows contain the records or data for the columns. Here is a sample table called "weather".

city, state, high, and low are the columns. The rows contain the data for this table:

Weather

city state high low

Phoenix Arizona 105 90

Tucson Arizona 101 92

Flagstaff Arizona 88 69

San Diego California 77 60

Albuquerque New

Mexico 80 72

Selecting Data

The select statement is used to query the database and retrieve selected data that match the criteria that you specify. Here is the format of a simple select statement:

select "column1"[,"column2",etc] from "tablename" [where "condition"];

[] = optional

The column names that follow the select keyword determine which columns will be returned in the results. You can select as many column names that you'd like, or you can use a "*" to select all columns.

The table name that follows the keyword from specifies the table that will be queried to retrieve the desired results.

The where clause (optional) specifies which data values or rows will be returned or displayed, based on the criteria described after the keyword where.

Conditional selections used in the where clause:

Operator	Purpose
=	Equality test
!=, ^=, <>	Inequality test.
> <	"Greater than" and "less than" tests
>= <=	"Greater than or equal to" and "less than or equal to" tests
IN	"Equal to any member of" test. Equivalent to "=ANY"
NOT IN	Equivalent to "!=ALL". Evaluates to FALSE if any member of the set is NULL
BETWEEN	Greater than or equal to x and less than or equal to y
x [NOT] LIKE y	TRUE if x does [not] match the pattern y. Within y, the character '%' matches any string of zero or more characters except null. The character '_' matches any single character.

LIKE *See note below

The LIKE pattern matching operator can also be used in the conditional selection of the where clause. Like is a very powerful operator that allows you to select only rows that are "like" what you specify. The percent sign "%" can be used as a wild card to match

any possible character that might appear before or after the characters specified. For example:

select first, last, city from empinfo where first LIKE 'Er%';

This SQL statement will match any first names that start with 'Er'. Strings must be in single quotes.

Or you can specify,

select first, last from empinfo where last LIKE '%s';

This statement will match any last names that end in a 's'.

select * from empinfo where first = 'Eric';

This will only select rows where the first name equals 'Eric' exactly.

Sample Table: empinfo

first last id age city state

John Jones 99980 45 Payson Arizona

Mary Jones 99982 25 Payson Arizona

Eric Edwards 88232 32 San Diego California

Mary Ann Edwards 88233 32 Phoenix Arizona

Ginger Howell 98002 42 Cottonwood Arizona

Sebastian Smith 92001 23 Gila Bend Arizona

Gus Gray 22322 35 Bagdad Arizona

Mary Ann May 32326 52 Tucson Arizona

Erica Williams 32327 60 Show Low Arizona

Leroy Brown 32380 22 Pinetop Arizona

Elroy Cleaver 32382 22 Globe Arizona

select first, last, city from empinfo;

select last, city, age from empinfo where age > 30;

select first, last, city, state from empinfo where first LIKE 'J%';

select first, last, age from empinfo where last LIKE '%illia%';

select * from empinfo where first = 'Eric';

Updating Records

The update statement is used to update or change records that match a specified criteria. This is accomplished by carefully constructing a where clause.

update "tablename" set "columnname" = "newvalue" [,"nextcolumn" = "newvalue2"...]
where "columnname" OPERATOR "value" [and|or "column" OPERATOR "value"];
[] = optional
update phone_book set area_code = 623 where prefix = 979;
update phone_book set last_name = 'Smith', prefix=555, suffix=929
where last_name = 'Jones';

Deleting Records

The delete statement is used to delete records or rows from the table.

delete from "tablename" where "columnname" OPERATOR "value" [and|or "column"
OPERATOR "value"];[ ] = optional
delete from employee;
Note: if you leave off the where clause, all records will be deleted!
delete from employee where lastname = 'May';
delete from employee where firstname = 'Mike' or firstname = 'Eric';

To delete an entire record/row from a table, enter "delete from" followed by the table name, followed by the where clause which contains the conditions to delete. If you leave off the where clause, all records will be deleted.

Drop a Table

The drop table command is used to delete a table and all rows in the table.

To delete an entire table including all of its rows, issue the drop table command followed by the table name. Drop table is different from deleting all of the records in the table. Deleting all of the records in the table leaves the table including column and constraint information. Dropping the table removes the table definition as well as all of its rows.

drop table "tablename"
Example: drop table employee;

CHAPTER THREE: FLOW-CHARTS OF PROGRAM MODULS

3.1 Flow-Chart of Main program

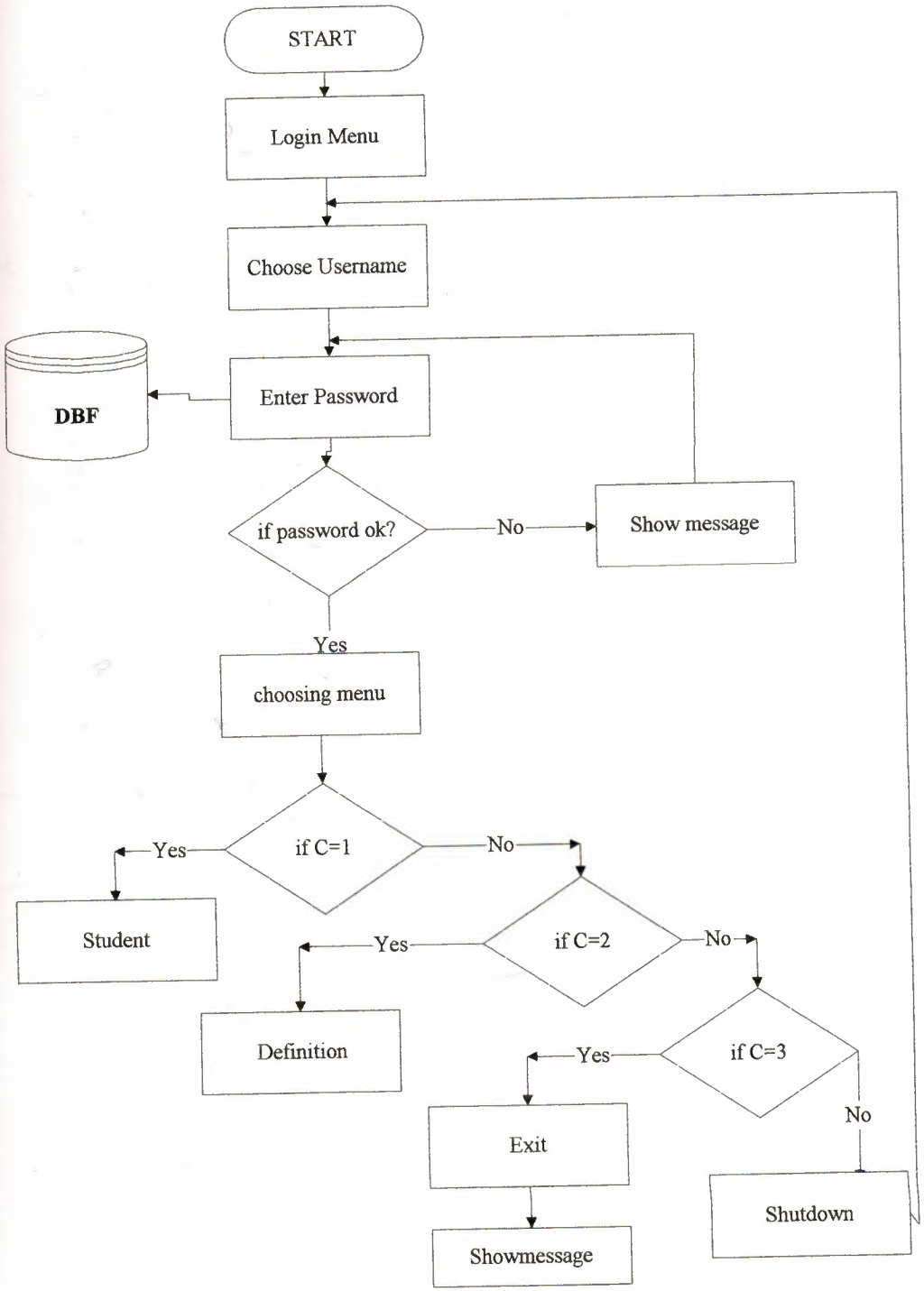


Figure 3.1.Main Menu Flow-Chart

3.2.Flow-Chart for Student Menu(Admin)

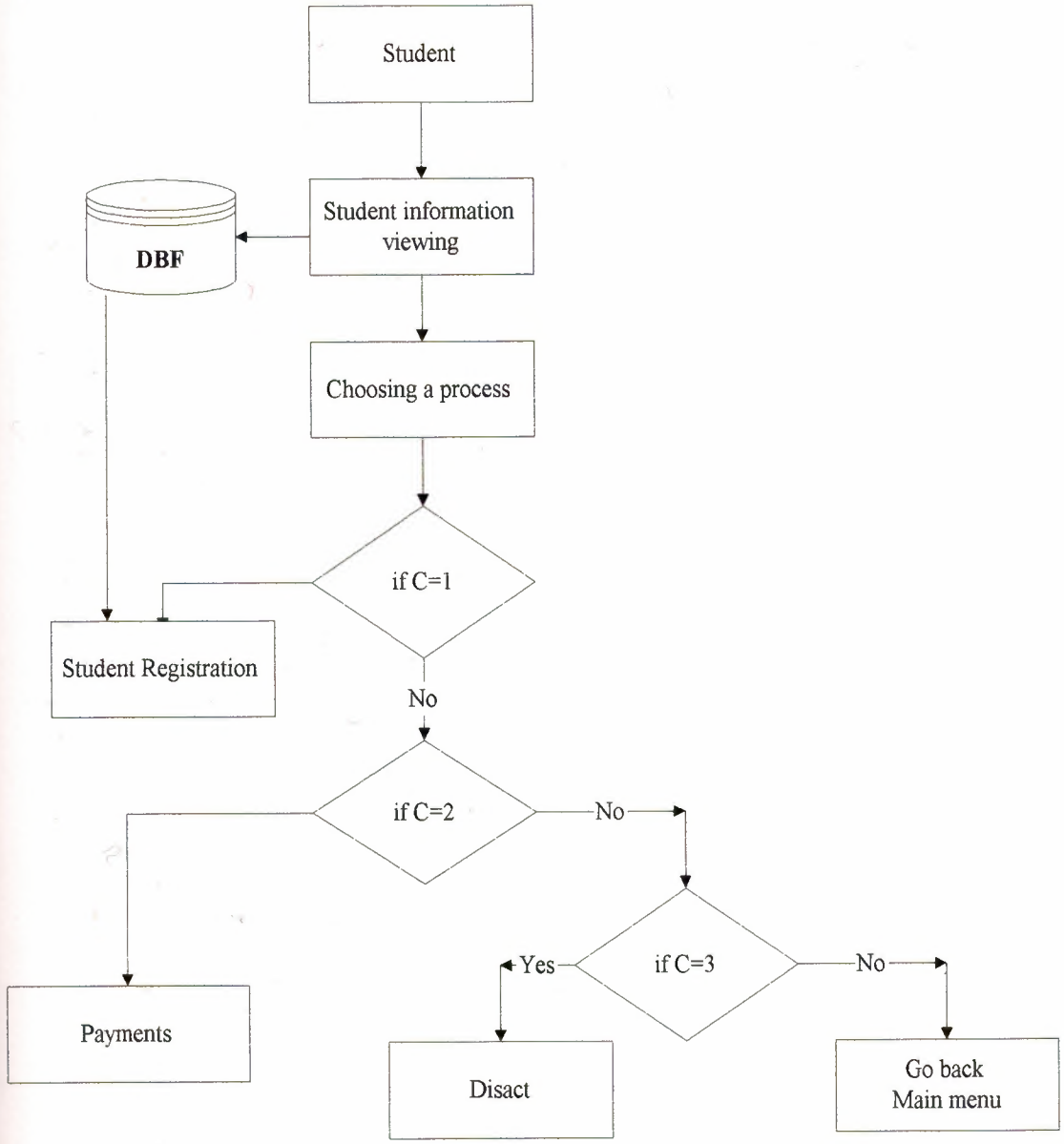


Figure 3.2.Student Menu Flow-Chart

3.2.1.Flow-Chart for Disact (for Student Admin Menu)

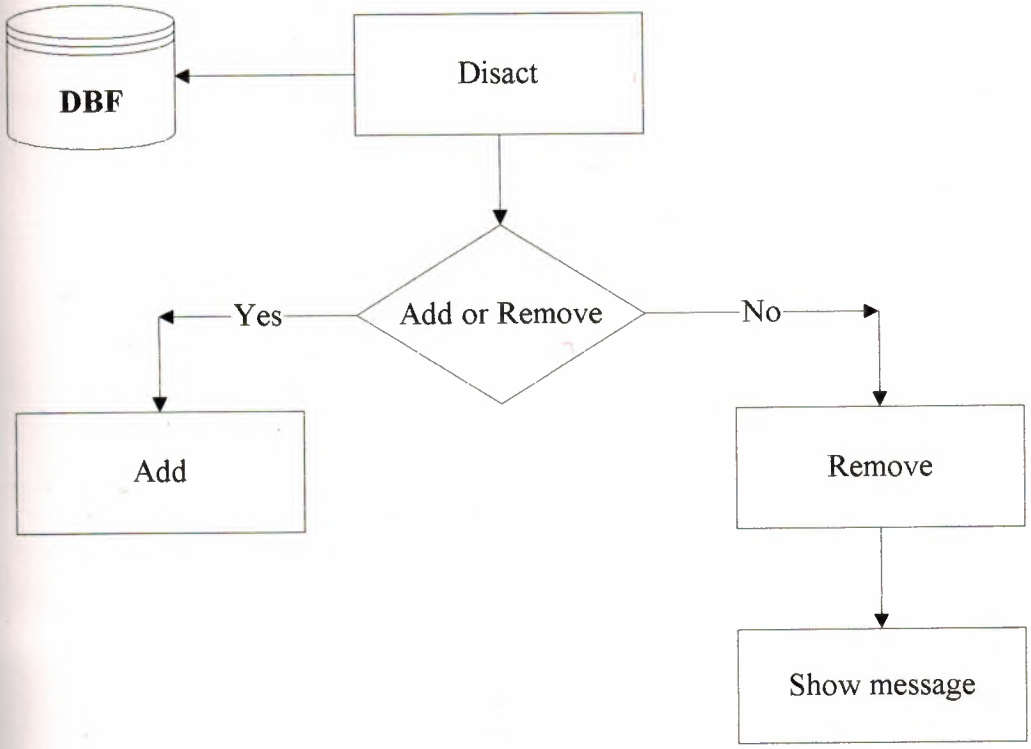


Figure 3.3.Disact Menu Flow-Chart

3.2.2.Flow-Chart for Payment (for Student Admin Menu)

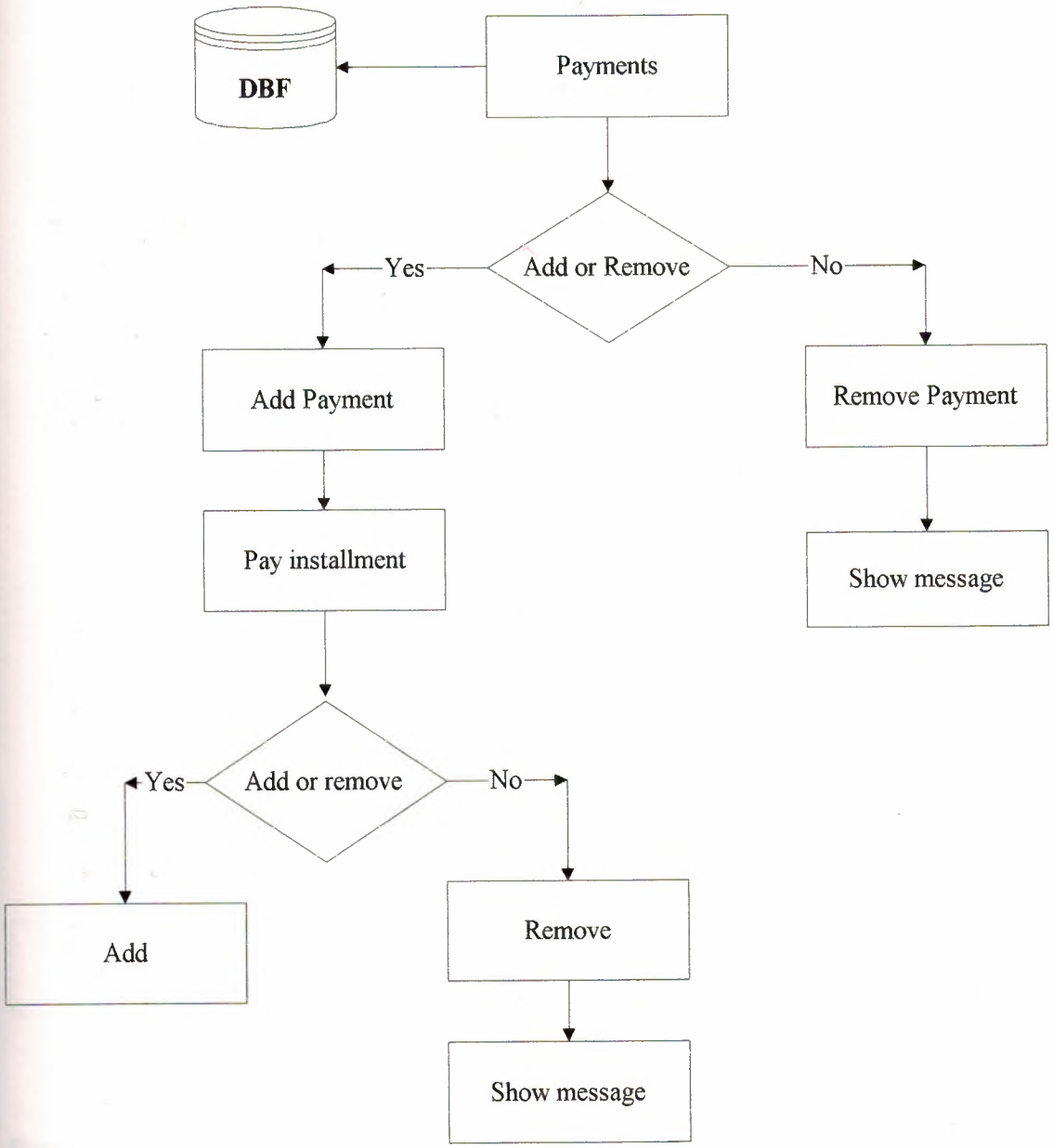


Figure 3.4.Payment Menu Flow-Chart

3.3.Flow-Chart for Student Menu (Secretary and Advisor)

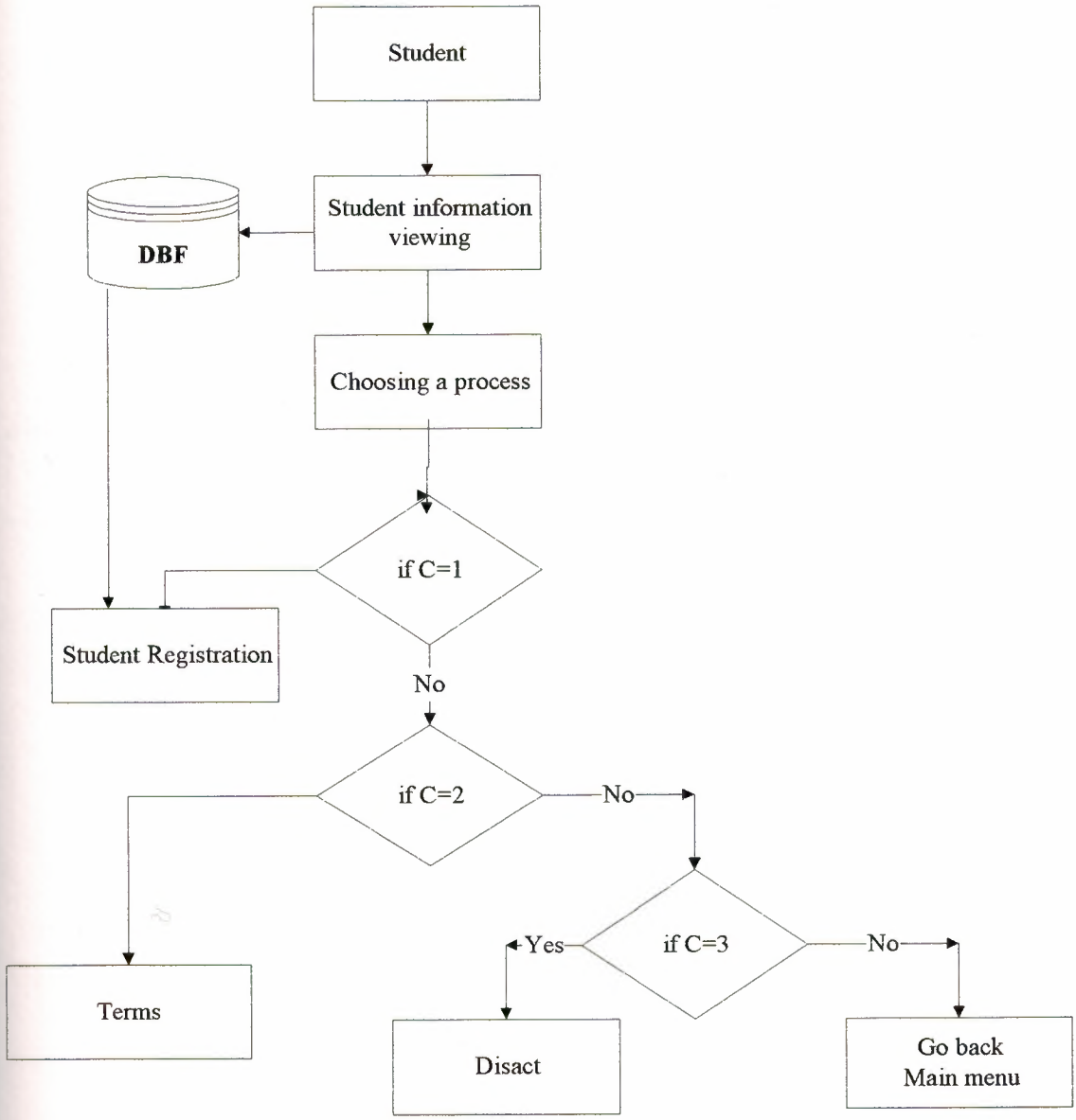


Figure 3.5. Student Menu Flow-Chart (Secretary and Advisior)

3.3.1.Flow-Chart for Terms (Secretary and Advisor)

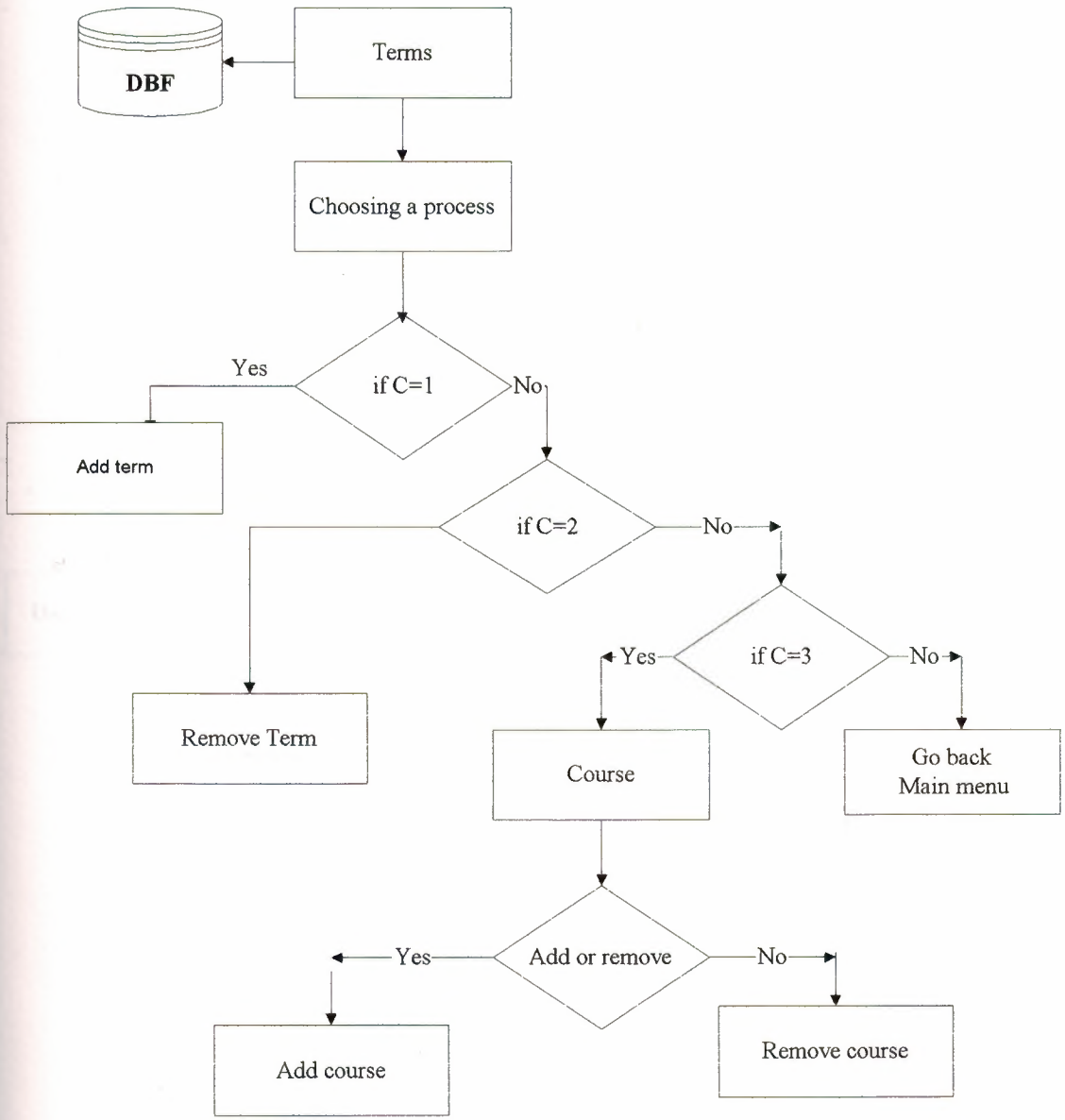


Figure 3.6.Terms Flow-Chart

3.4.Flow-Chart for Definition (for Admin Menu)

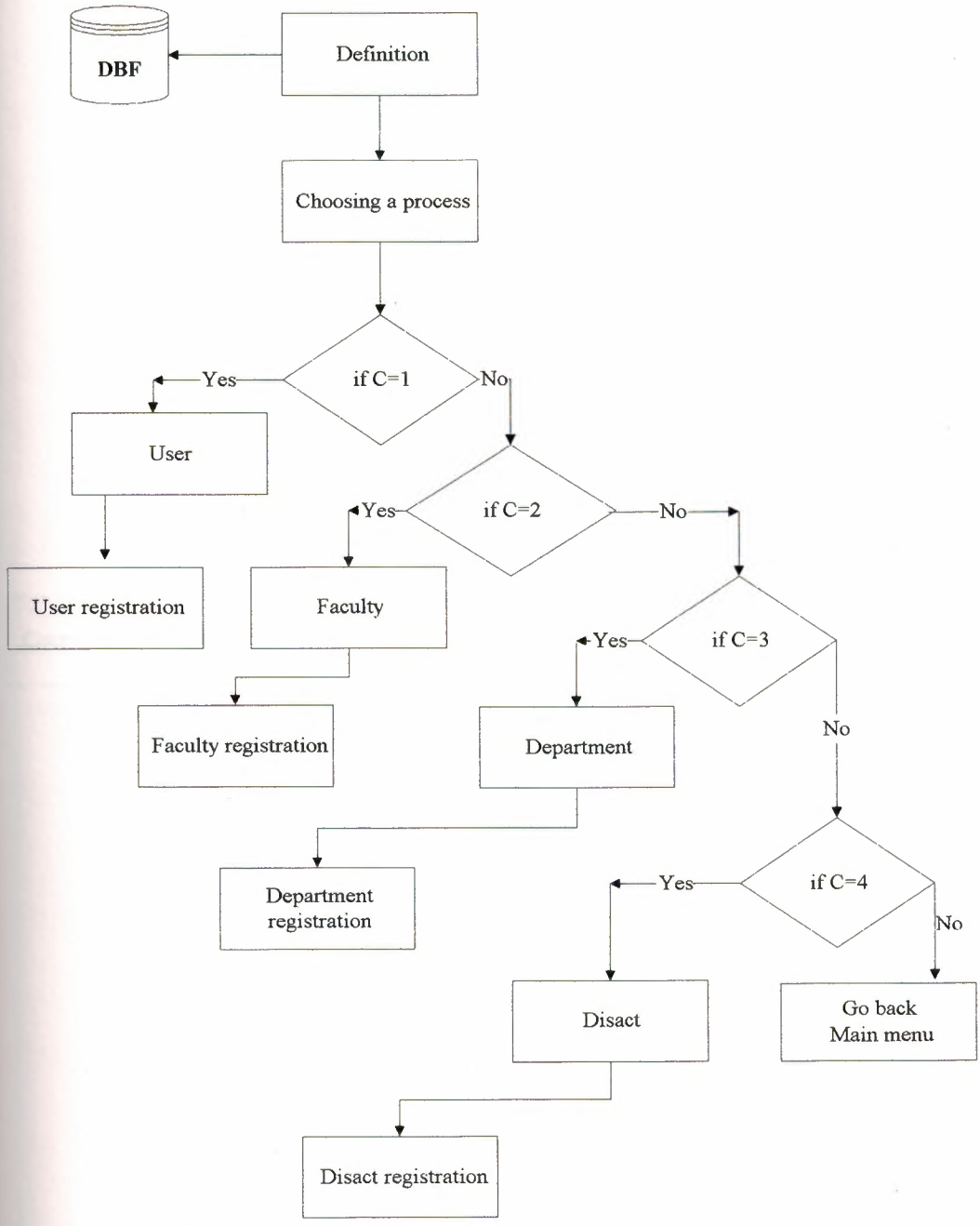


Figure 3.7.Definition Flow-Chart(admin)

3.5. Flow-Chart for Definition (for Secretary and Advisor)

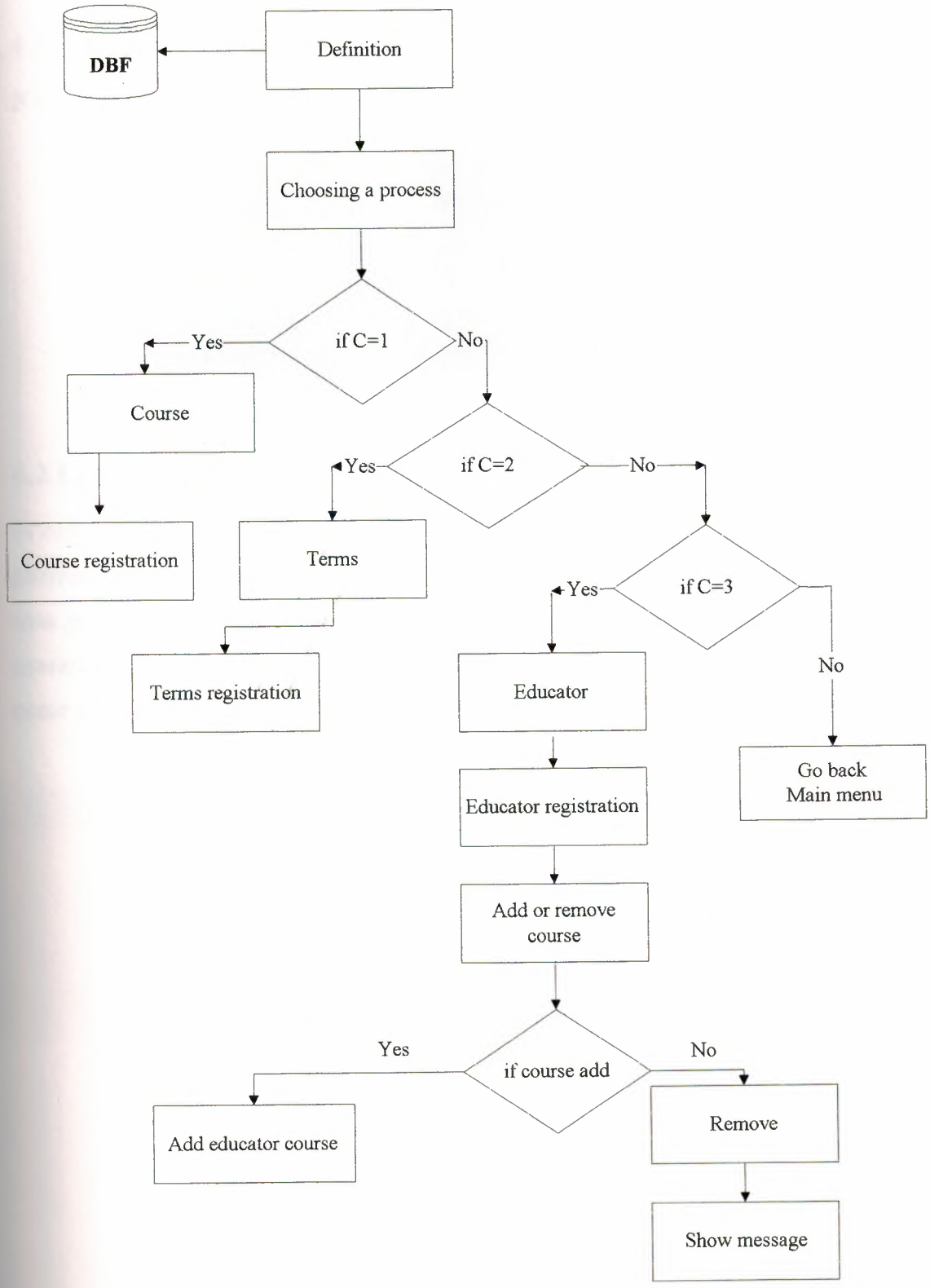


Figure 3.6.Definition Flow-Chart(secretary and advisor)

CHAPTER FOUR: DEVELOPMENT OF PROGRAM MODULES OF STUDENT TACKING SYSTEM

4.1.Starting Screen

If the program is running then start screen show.

Figure 4.1.Starting Screen

4.2.Log-in Screen

In order to protect our software a high level of security must be applied, so when the program runs it ask the operator to choose the group and user name to enter the his /her own password to accomplish the entrance process. When the program recognise the operator the main menu screen will accrue. In case of unrecognising alert message will occur tells that "Invalid password, try again!"

Figure 4.2.wrong message screen

Shows this screen user choose the group

Figure 4.3.Log-in Screen

After choose

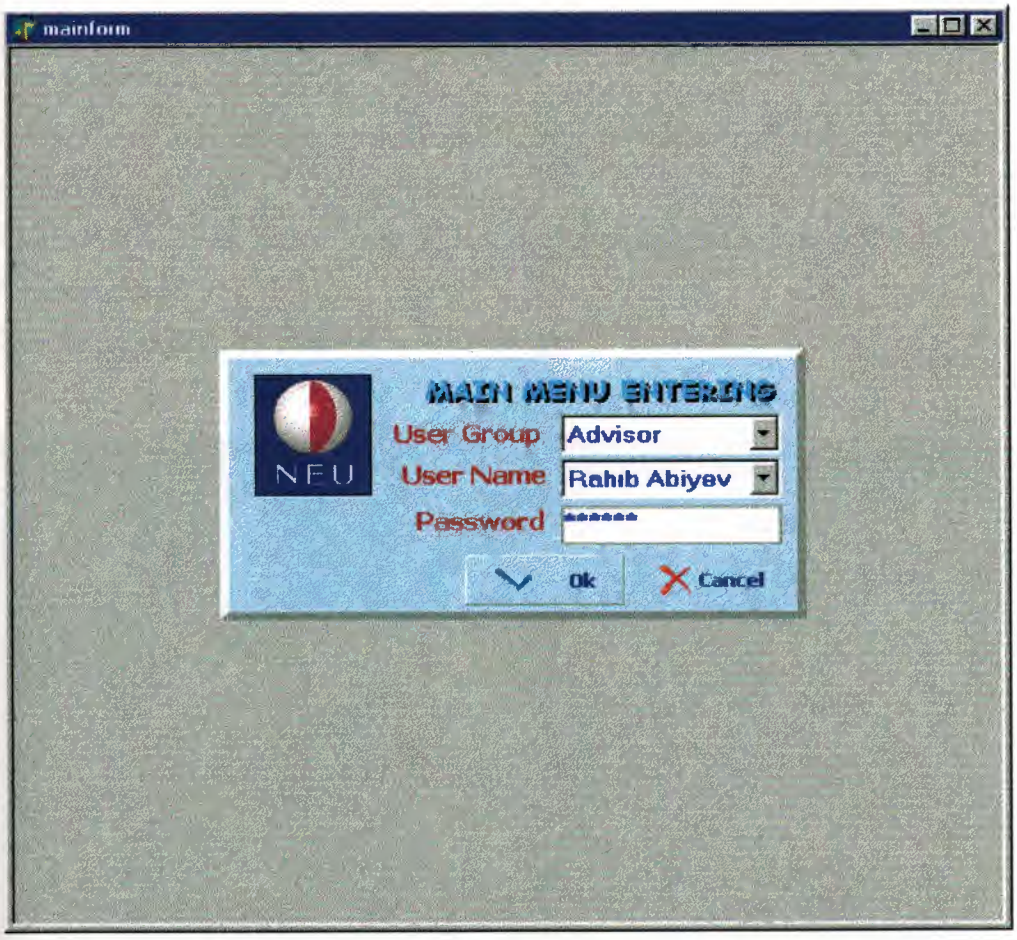


Figure 4.4. Log-in Screen

4.3.Main Menu Screen

Main menu is same for each user. There are four buttons on the main menu. Each button it's own obligation. The buttons are on the main menus are;

1-Student Button:

Some operation can be done according to authority of the users. These operations are;

- User can register information of a new student or user can see the information of registered student
- User can register the students to the course if the student registered to the semester, and user can delete course registration according to the authority.
- User can add or remove disact (penalty) and see the given penalties

User can do the money operations also according to the authority.

2-Definitions Button:

There are different menus according to the user's authority. Definition button is in the definition menu as a structure

3-Search Button: This button using search the searching data, by name by id

4-Shut Down Button:

This button provides us to exit from the main menu and return the login menu.

5-Exit Button:

This button provide us to exit from the program

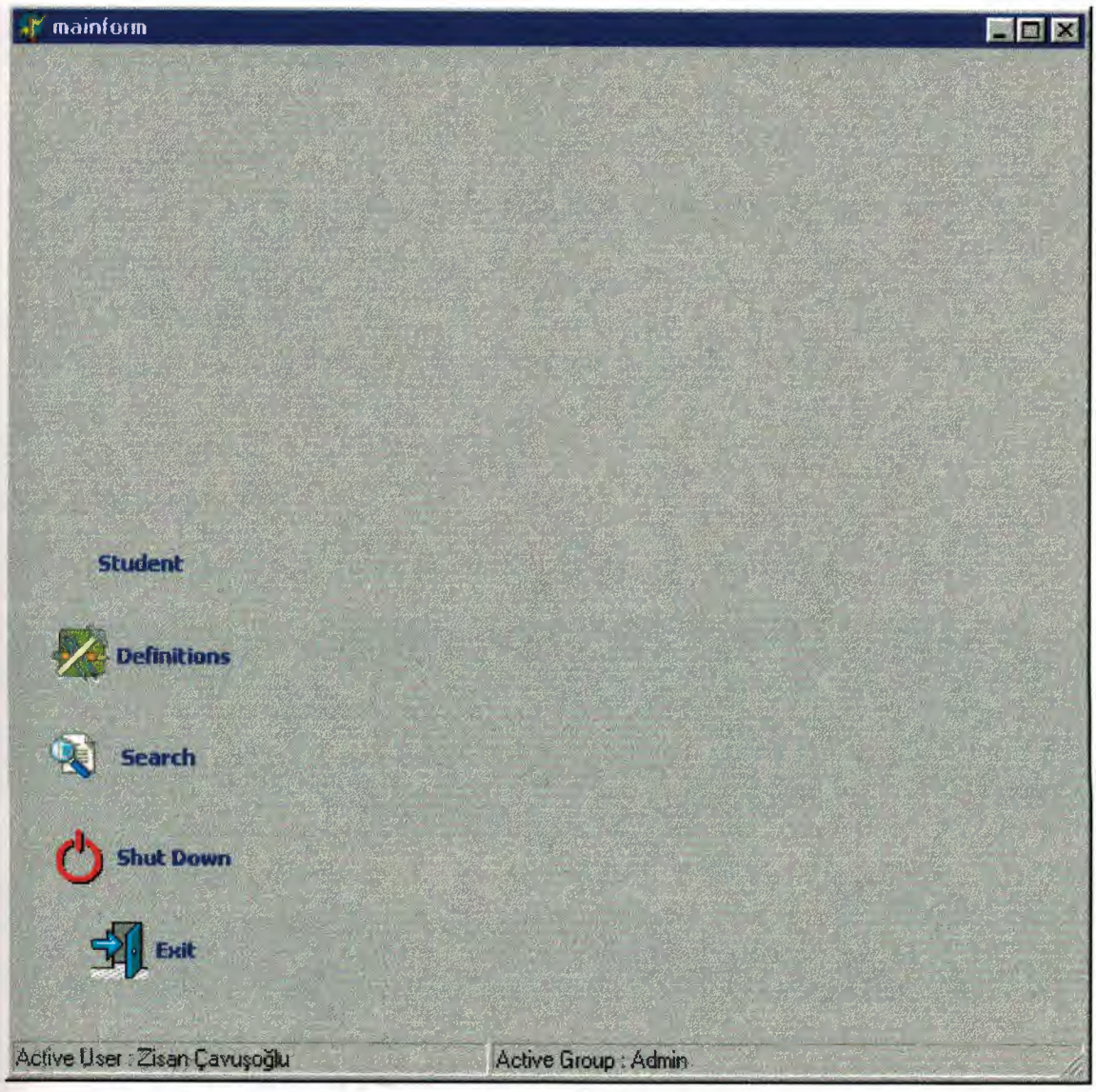


Figure4.5.Main Menu Screen

4.4.Student Screen

General information and details with a picture are listed here about each student and also extra details could be hidden due to the request of the user. The sub-lists at the bottom of the page are designed according to the need and authority of the users

If the user is,

1-ADMİN:

- Hide details button
- Main menu button
- Disact page
- Payment page

1-Hide Details Button:

This button is active then shows the student information detail. (NOTE: If this button is pressed then the details will be hidden)

2-Main Menu Button:

This button provides us to exit from the Student menu and return the Main menu.

3.Disact Page:

1-Add Disact Button:

If the student take a punishment we can add it with using this button.

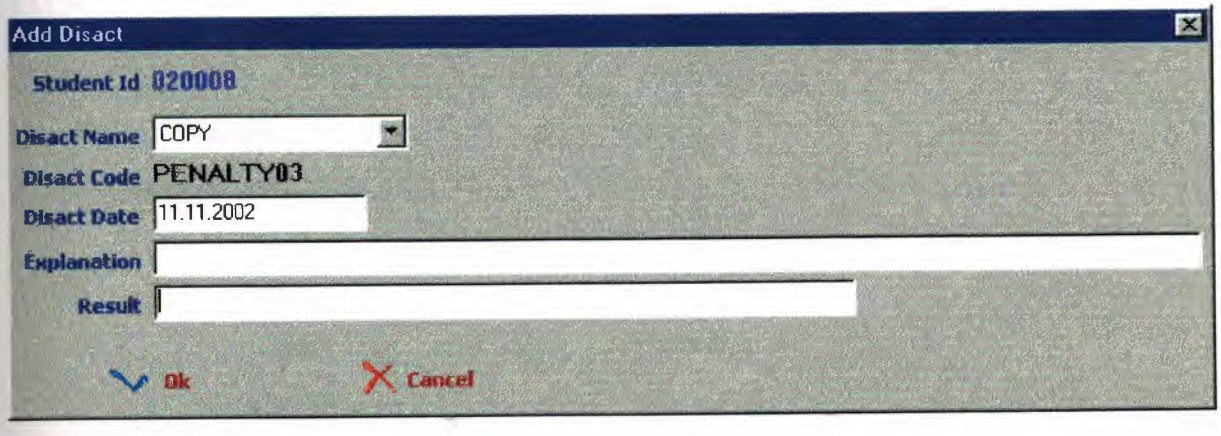


Figure4.6.Add Disact Menu Screen

2- Remove Disact:

If admin want to remove the punishment given to the student can removed by admin. When we click this button a message "Record is deleting, Are you sure?" appears on the screen. If we click OK the punishment become deleted.

4-Payment Page:

They are authorized to change them for each student with Add and Remove buttons here. Meanwhile there is an another part, as "Revenue Installments" and a chance for canceling them here. This is when the payments are not going to be paid on time by the student and the university gives them a chance to pay in a longer time period.

1-Add Payment Button:

In this section we can list the payment of the students. If we click this button we can see add payment menu.

4.4.1.Add Payment Screen

Payment Name Combobox:

We can see payment titles, which are written in the admin definition. for instance social activity payment, school semester payment... When we choose a title it's code automatically seen on the board.

Tax Rate:

If some one pay the money late, he/she must pay interest.

Instalment Number:

If an instalment will be applied then user write it to this section and click enter.

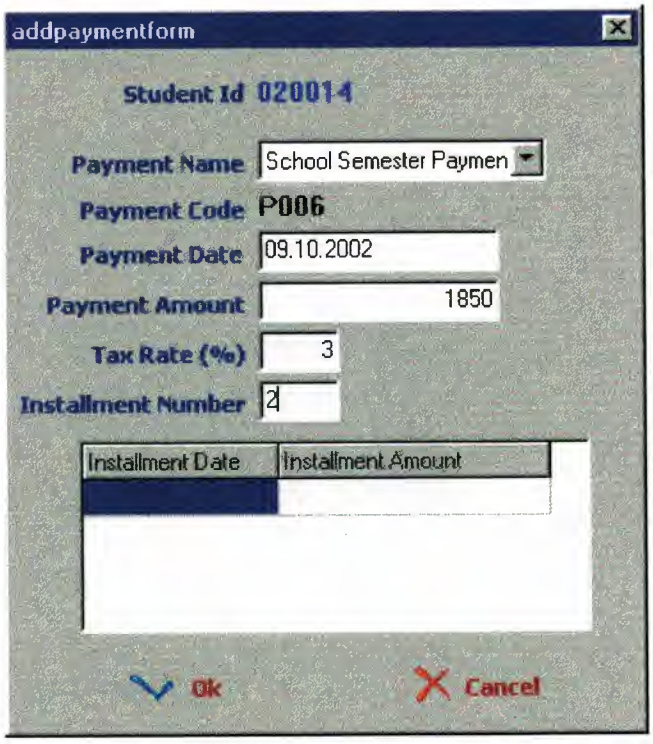


Figure4.7.Add payment form

After that the amount of each install appears according to the number of the payment next to he installs.

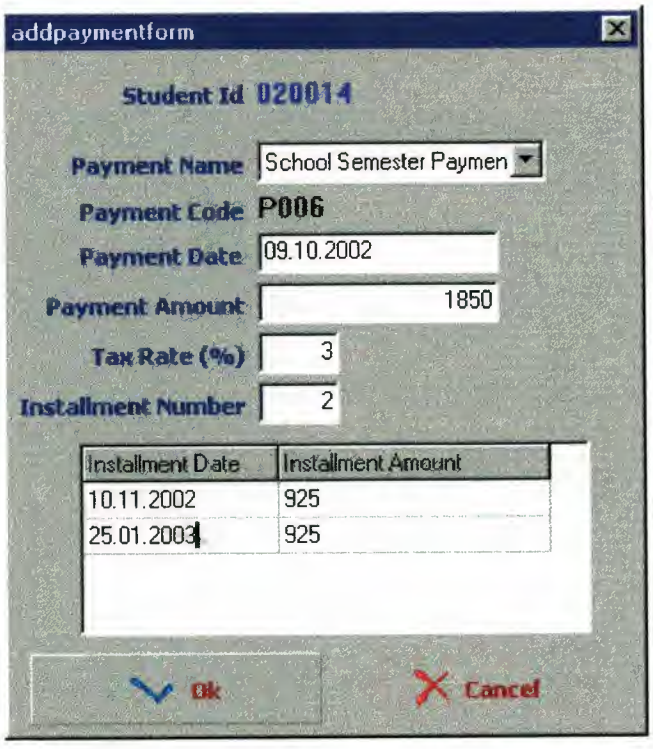


Figure 4.8.Addpayment Screen

2-Remove Payment:

If it is necessary payments can deleted. If user click this button a message "Record is deleting, Are you sure?" appear on the screen. If user click **Yes** then the registration become deleted.

Figure 4.9.Remove Payment Screen

3-Revenue Instalment:

If the user click this button then the pay instalment menu, which is the sub menu of the payment page, opens

4.4.2. Pay Instalment Screen :

We use this menu to pay the money on the date which is determined in the add payment menu.

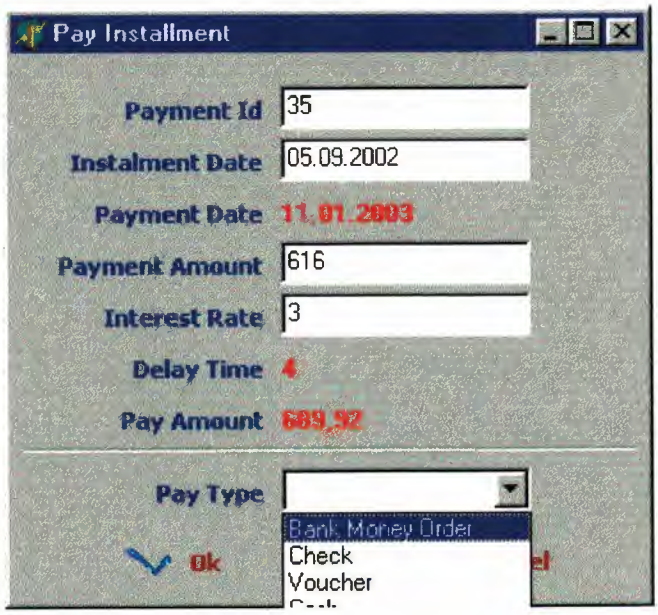


Figure 4.10.Pay instalment Screen

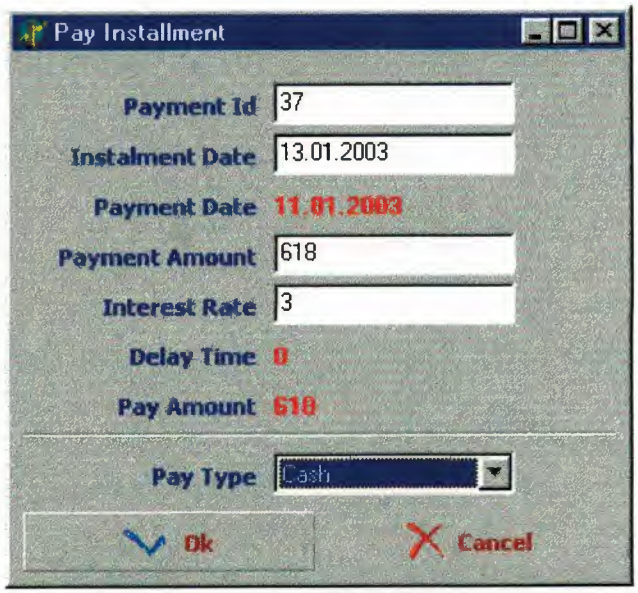


Figure 4.11.Pay instalment Screen (after choose)

If the users are;

2- Secretary and Advisor

- Student Registration
- Picture Button
- Main Menu Button
- Disact Page
- Term Page
- Hide detail Button

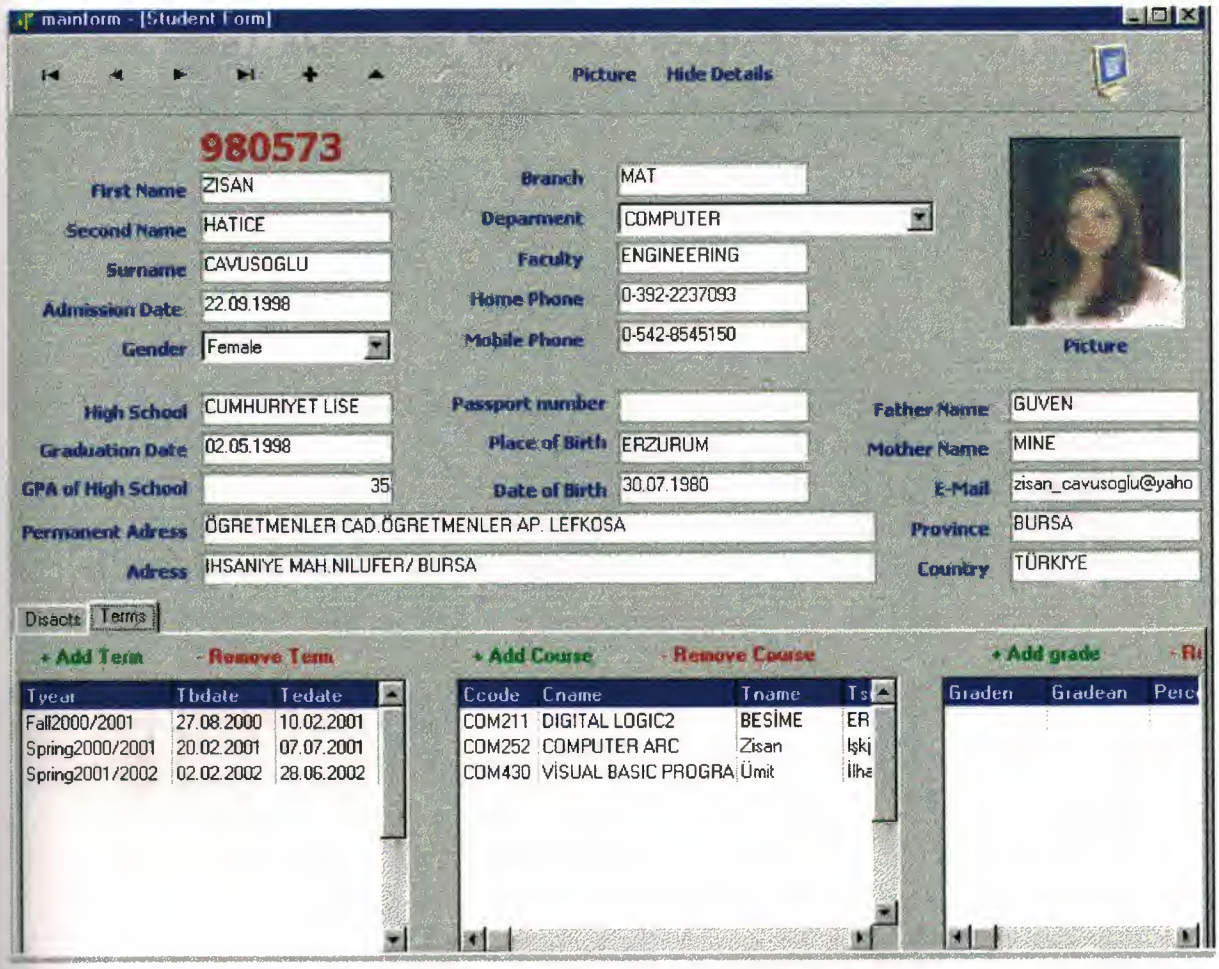


Figure 4.12.Student Menu (Secretary and Advisor)

Hide detail Button-Show Details

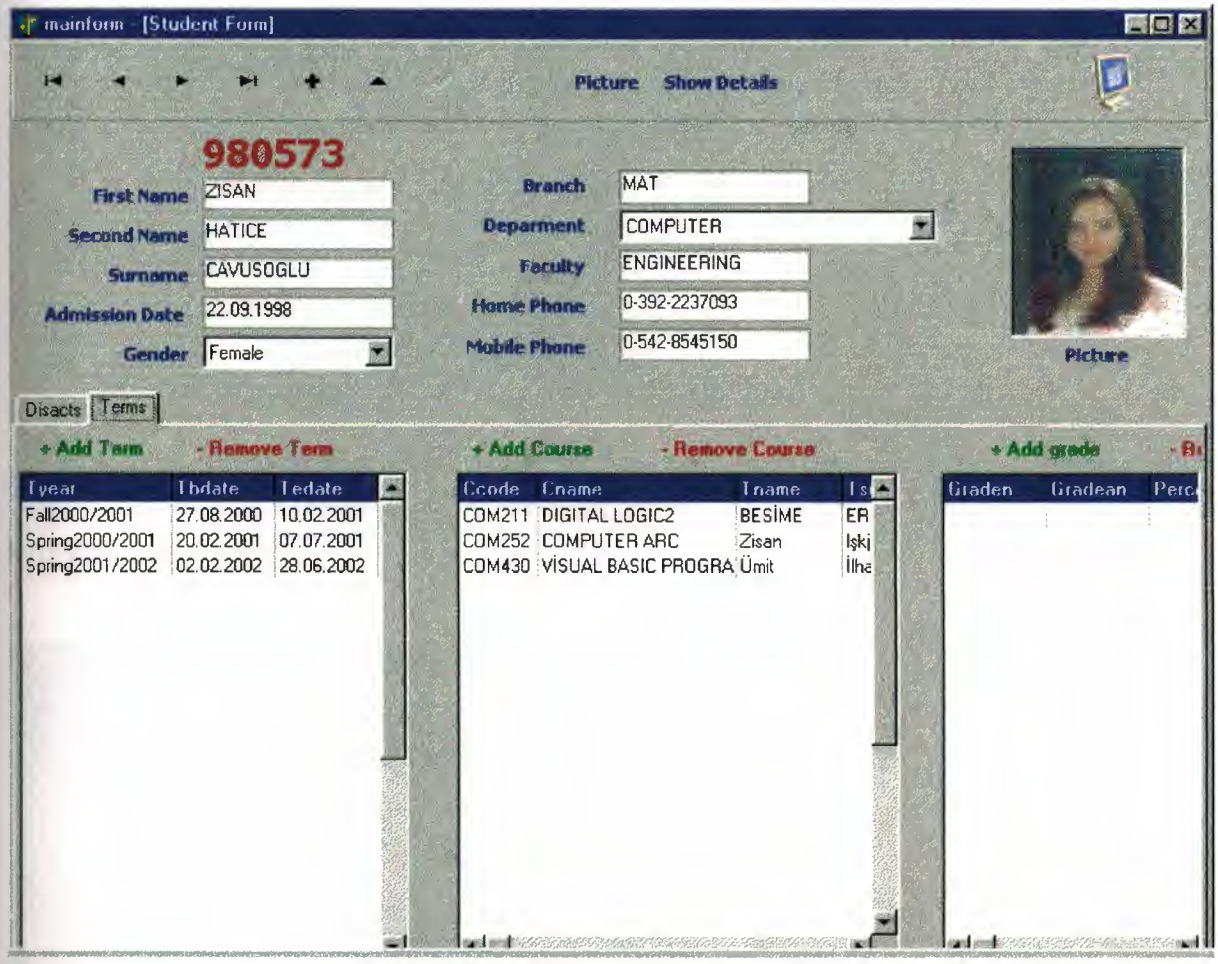


Figure 4.13.Student Menu (hide details button)

4.4.3. Terms Screen

The Term section. The secretary here is able to add the academic semester that any student is registered and additionally the details about the courses that student is registered and finally at the end of the semester it is giving a chance to entering the letter grade for each course.

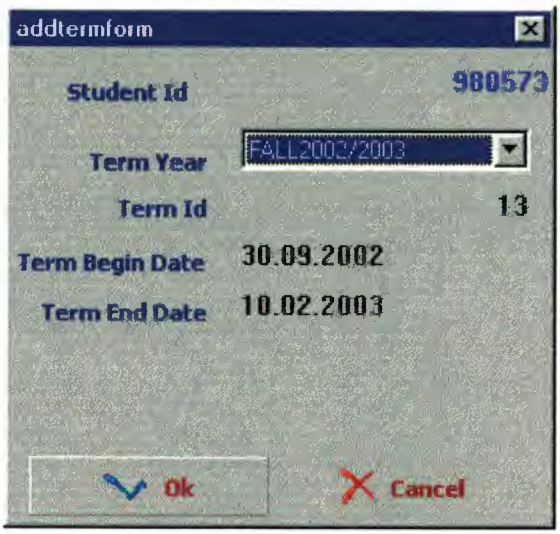


Figure 4.14. Add Term Screen

2. Add Course Button

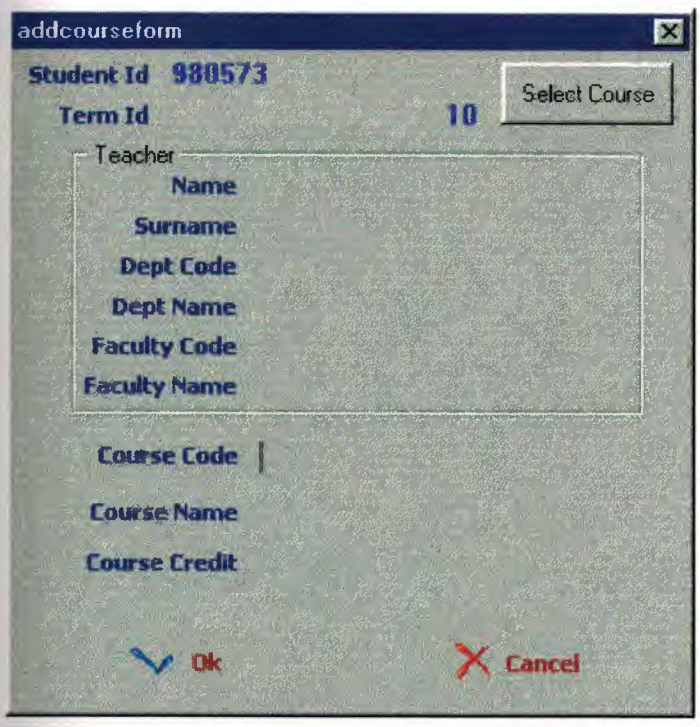
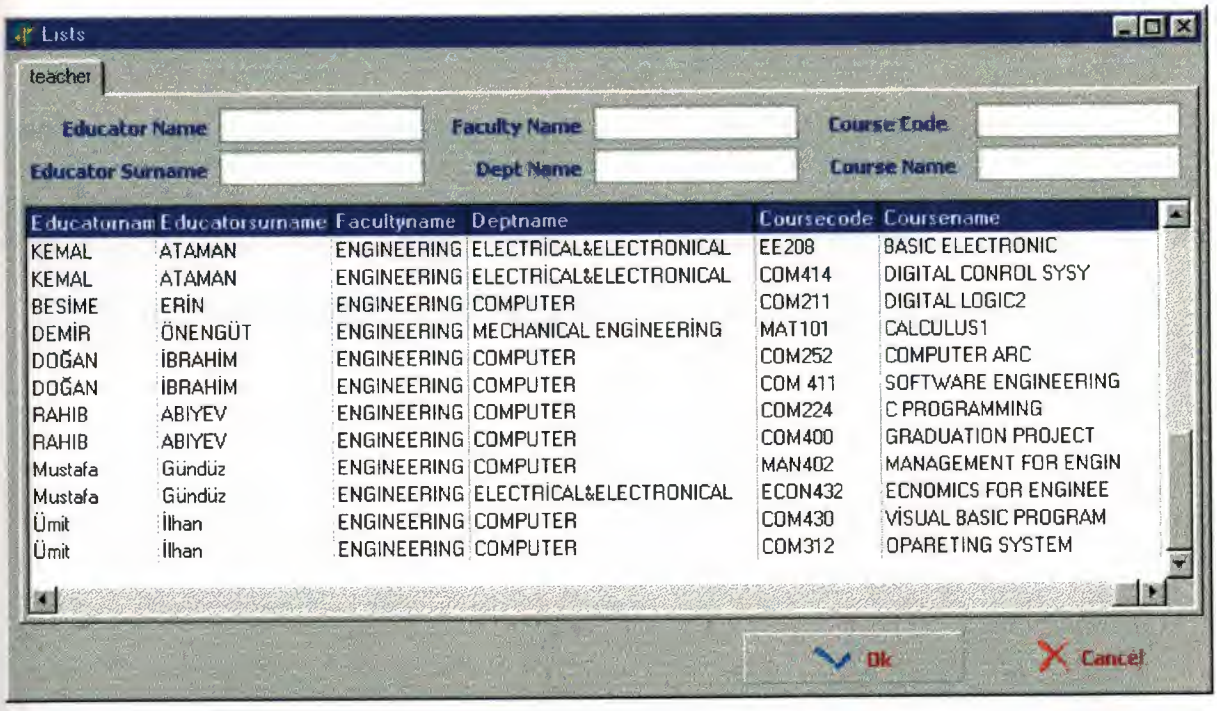


Figure 4.15. Add Course Screen



Educatornam	Educatorsurname	Facultyname	Deptname	Coursecode	Coursename
KEMAL	ATAMAN	ENGINEERING	ELECTRİCAL&ELECTRONICAL	EE208	BASIC ELECTRONIC
KEMAL	ATAMAN	ENGINEERING	ELECTRİCAL&ELECTRONICAL	COM414	DIGITAL CONROL SYSY
BESİME	ERİN	ENGINEERING	COMPUTER	COM211	DIGITAL LOGIC2
DEMİR	ÖNENGÜT	ENGINEERING	MECHANICAL ENGİNEERİNG	MAT101	CALCULUS1
DOĞAN	İBRAHİM	ENGINEERING	COMPUTER	COM252	COMPUTER ARC
DOĞAN	İBRAHİM	ENGINEERING	COMPUTER	COM 411	SOFTWARE ENGINEERING
RAHIB	ABIYEV	ENGINEERING	COMPUTER	COM224	C PROGRAMMING
RAHIB	ABIYEV	ENGINEERING	COMPUTER	COM400	GRADUATION PROJECT
Mustafa	Gündüz	ENGINEERING	COMPUTER	MAN402	MANAGEMENT FOR ENGIN
Mustafa	Gündüz	ENGINEERING	ELECTRİCAL&ELECTRONICAL	ECON432	ECNOMICS FOR ENGINEE
Ümit	İlhan	ENGINEERING	COMPUTER	COM430	VİSUAL BASIC PROGRAM
Ümit	İlhan	ENGINEERING	COMPUTER	COM312	OPARETING SYSTEM

Figure 4.16. List Secreen

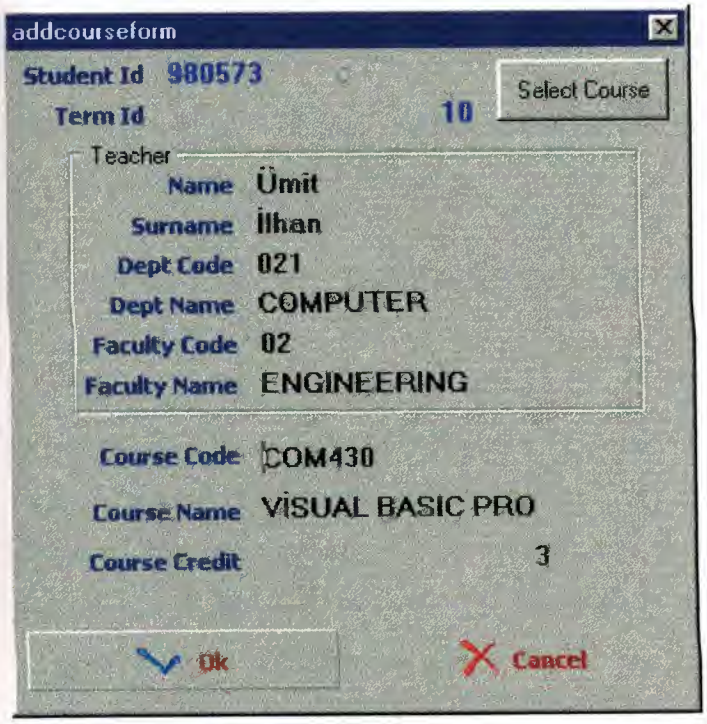


Figure 4.17. Add Course Screen (after choose the list)

Searching course

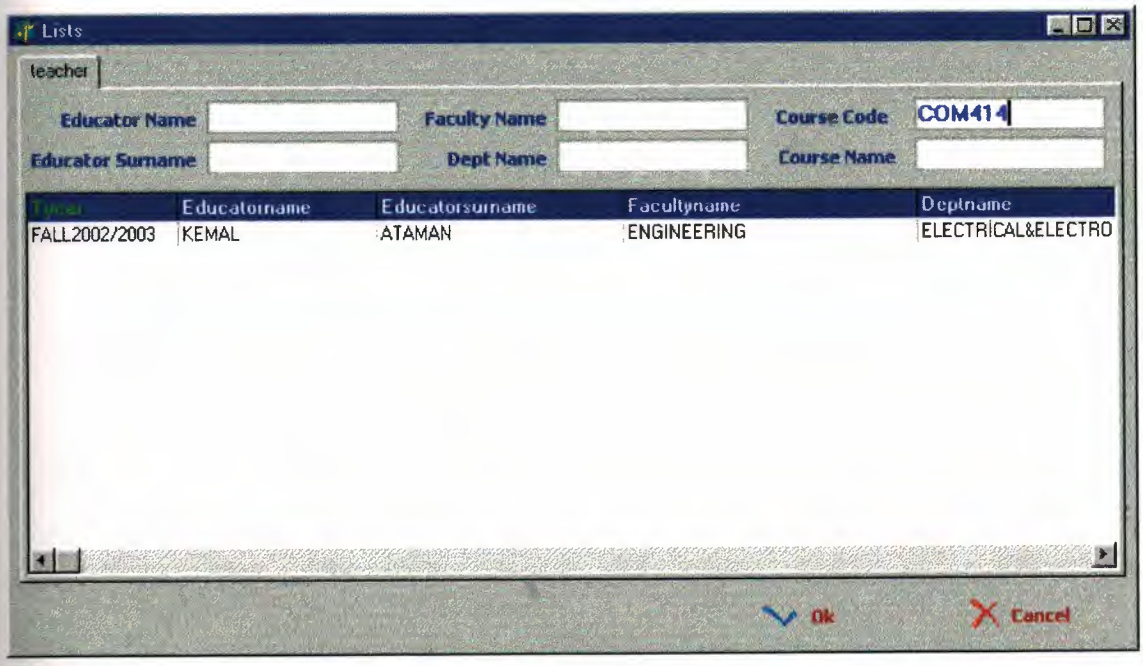


Figure 4.18. List Search Screen

The Disacts page: The secretary only can screen the disciplinary decisions about any student here and not change them.

If the user is in "Advisor" group, additionally they see the followings:

Any Advisor is able to reach and change the details about any student exactly as it is explained for the users in the secretary groups above. Precisely if the program is going to work on a network there can a control mechanism for important data that once they changed, they are updated for all user groups and security and harmony should be noticed in that case. The members of Accountancy group do not have a chance to reach to student screen in this program.

4.6.Definition Screen

Each user can reach to this screen but their authorities are lemmatised due to groups that they are registered in.

If the user is in "Admin" group:

1. **User Button:** The definitions of the users and their group are done here. The name of the user is defined in a related group and a password is introduced to the users so they can login to the system.
2. **Faculty Button:** The definition of the Faculties that are active in the university is done here. For example Engineering, Law, Business Administration etc with a specific code defined by the university administration.
3. **Department Button:** Here the names of the departments are defined and a code is given to them with respect to the faculty that they are related to.
4. **Disact Button:** Here the type of the disciplinary penalties that could be faced by any student during their academic life is defined and a code is given respectively.

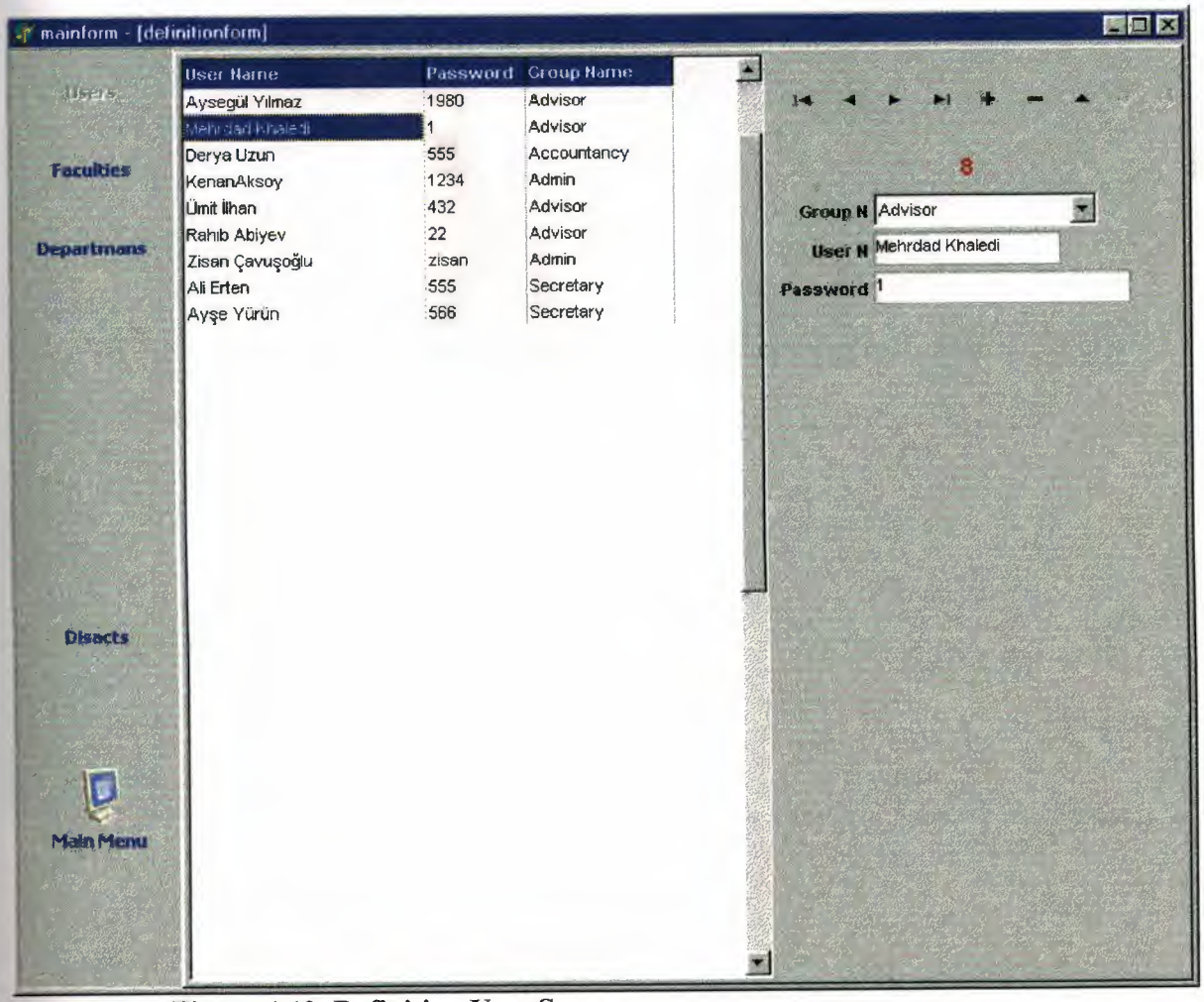


Figure 4.19. Definition User Screen

If the user is in "Secretary" group:

1. **Terms Button:** The details about the academic semesters are entered here. The academic year, the starting date and ending date of the semester are some of them for instant.
2. **Course Button:** The course code, course name, its credit and description, the related department which offer this course and the pre-requisite and similar information is defined here.
3. **Educator Button:** This is for definition of the details of the teaching staff. The instructor for any course is identified here and personal information and additional contact numbers are uploaded to the program. Add Course here is for adding a course to the semester load of any educator and Remove Course is for removing a course from the semester load of any educator

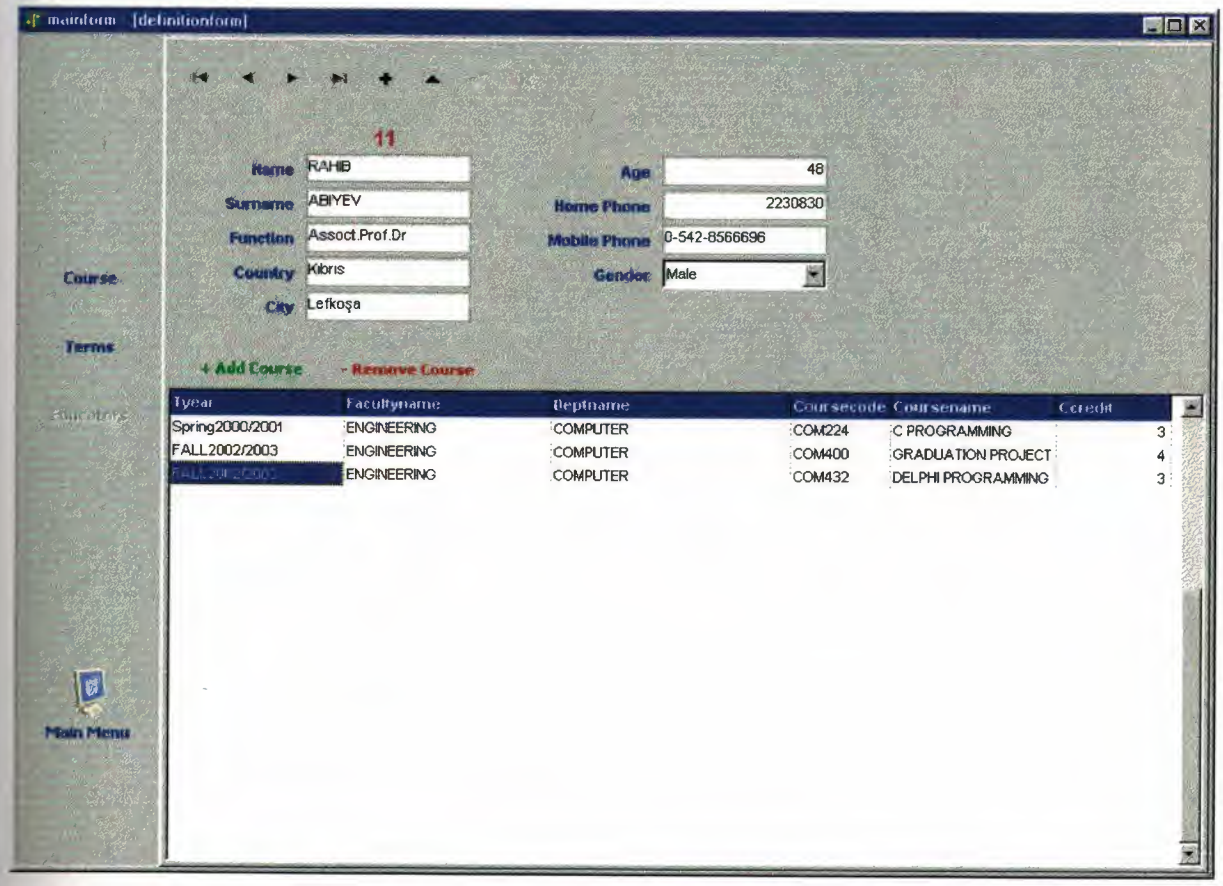


Figure 4.20.Definition Educator Screen

If the user is in "Advisor" group:

They have the whole buttons and list available for the secretary except the "Educator Button", as they do not deal with this part of academic work while the registration period is running. As the main step of registration is while student is with advisor then the algorithm of the program was designed so it gives the most suitable facilities to reach the required data when they want to register any student.

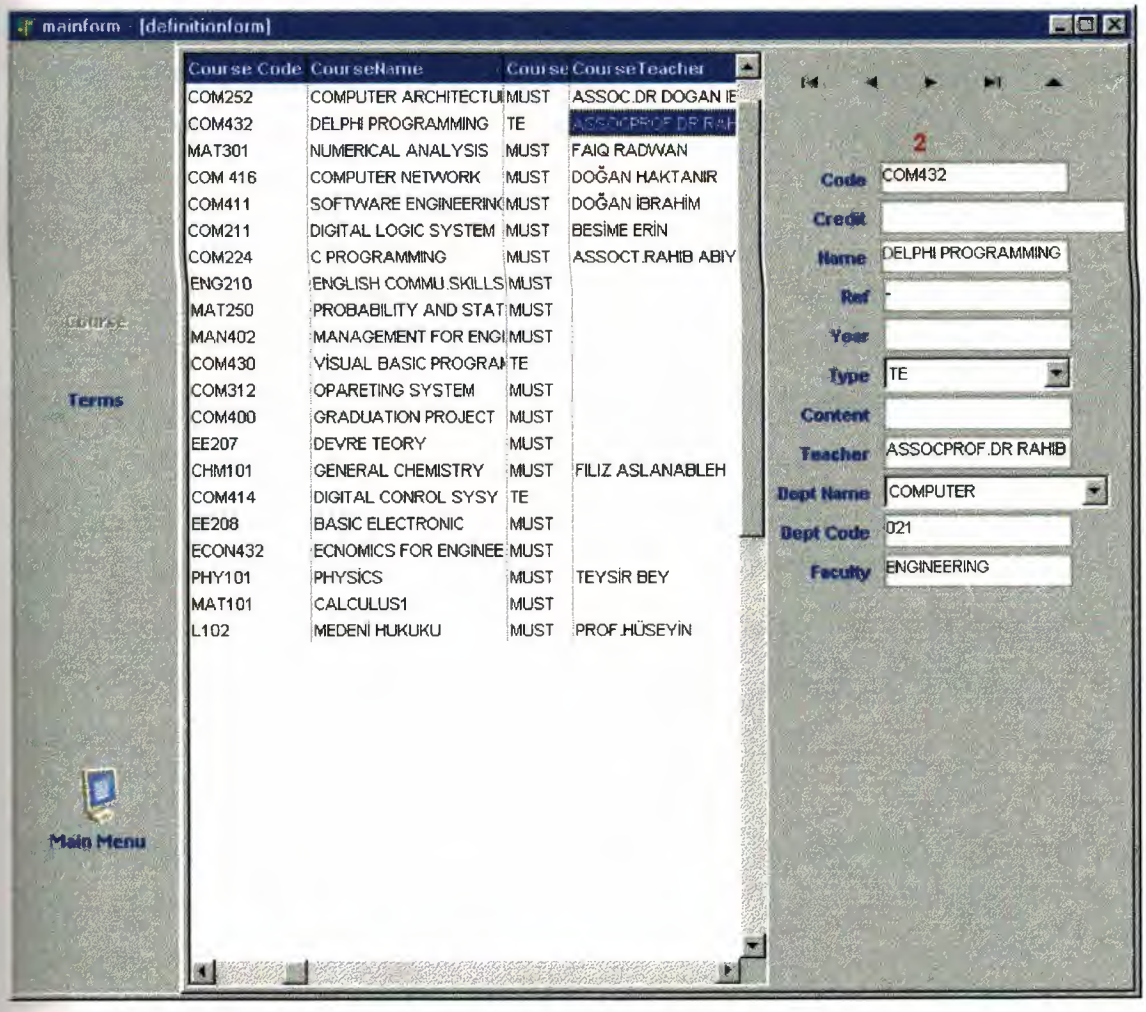


Figure 4.21.Definition Course Screen

CONCLUSION

In this graduation project Delphi programming was used to create a simple registration programme that can be used in our faculty. The development of student tracking system includes such problems as; student and course registration, educator registration, and student payment following.

For each block the special menu is designed and allows any user to easily realise, update and apply the searching process. The advantage of Delphi as an object-oriented programme particularly allows the programmer to create a perfect view and easily control the database.

The security of this programme and the authorisation for reaching the data of course was the most important factor considered in all steps of creation of this programme. Precisely automation in all aspects of our life is good, but in any education system the back up units and printed records of old data is absolutely a need. In this programme currently the print comment and its required facilities are not available and should be added in future. The other important thing that was not added here was the timetable. It should be possible to upload the weekly timetable of a faculty to the program so after registration of each student automatically the timetable could be printed and hand over to the students.

Finally it should be mentioned that this programme can be updated in future and extra units and menus can be added due to factors that may be was ignored or due to necessities that newly asked by the administration.

REFERENCES

1. Jeff Dunteman, Jim Mischell, Don Taylor. *Delphi Programming Explorer*. The Coriolis Group. USA 1995.
2. Neil Rubenking. Delphi Programming Problem Solver. IDG Books Worldwide.Inc. USA 1996.
3. A research guide for Delphi. December 24, 2002 from the World Wide Web *"http://www.borland.com/delphi* "
4. Delphi Driving Tomorrow Techonology.November 15, 2002. from the World Wide Web *"http://www.delphi.com* "
5. Delphi Developers Informatıon and Components. January 05, 2002 from the World Wide Web *"http://www.magsys.co.uk/delphi* "

APPENDIX

```
unit addcourseunit;
interface
uses
  Windows, Messages, SysUtils, Variants, Classes, Graphics, Controls, Forms,
  Dialogs, Buttons, DB, DBTables, StdCtrls, Mask, DBCtrls, ExtCtrls;
type
  Taddcourseform = class(TForm)
    SpeedButton1: TSpeedButton;
    SpeedButton2: TSpeedButton;
    Label3: TLabel;
    Label4: TLabel;
    Label11: TLabel;
    DBEdit1: TDBEdit;
    DBEdit8: TDBEdit;
    DBEdit2: TDBEdit;
    SpeedButton3: TSpeedButton;
    Label1: TLabel;
    DBEdit4: TDBEdit;
    GroupBox1: TGroupBox;
    Label5: TLabel;
    DBEdit3: TDBEdit;
    Label2: TLabel;
    DBEdit5: TDBEdit;
    Label6: TLabel;
    DBEdit6: TDBEdit;
    Label7: TLabel;
    DBEdit7: TDBEdit;
    Label8: TLabel;
    DBEdit9: TDBEdit;
    Label9: TLabel;
    DBEdit10: TDBEdit;
    Label10: TLabel;
    DBEdit11: TDBEdit;
    Query1: TQuery;
    procedure SpeedButton3Click(Sender: TObject);
    procedure FormClose(Sender: TObject; var Action: TCloseAction);
    procedure SpeedButton1Click(Sender: TObject);
    procedure SpeedButton2Click(Sender: TObject);
  private
    { Private declarations }
  public
    { Public declarations }
  end;
var
  addcourseform: Taddcourseform;
implementation
```

```
uses listunit,studentunit;
{$R *.dfm}

procedure Taddcourseform.SpeedButton3Click(Sender: TObject);
begin
  if not Assigned (listform) then listform := Tlistform.Create(Application);
  listunit.goingto := 0;
  listform.tquery.Open;
  listform.ShowModal;
end;
procedure Taddcourseform.FormClose(Sender: TObject;
  var Action: TCloseAction);
begin
  action := caFree;
  addcourseform := nil;
end;
procedure Taddcourseform.SpeedButton1Click(Sender: TObject);
begin
  if DBEdit3.Text <> '' then
    begin
      studentform.Table7.Post;
      addcourseform.Close;
    end
  else Showmessage('You must select a course !!!');
end;

procedure Taddcourseform.SpeedButton2Click(Sender: TObject);
begin
  studentform.Table7.Cancel;
  addcourseform.Close;
end;
end.

unit adddisactunit;
interface
uses
  Windows, Messages, SysUtils, Variants, Classes, Graphics, Controls, Forms,
  Dialogs, DB, DBTables, StdCtrls, Mask, DBCtrls, Buttons;
type
  Tadddisactform = class(TForm)
    SpeedButton1: TSpeedButton;
    SpeedButton2: TSpeedButton;
    Label3: TLabel;
    Label5: TLabel;
    Label4: TLabel;
    Label6: TLabel;
    Label8: TLabel;
    Label11: TLabel;
    DBLookupComboBox1: TDBLookupComboBox;
    DBEdit1: TDBEdit;
```

```
    DBEdit2: TDBEdit;
    DBEdit3: TDBEdit;
    DBEdit5: TDBEdit;
    DBEdit8: TDBEdit;
    Table2: TTable;
    DataSource2: TDataSource;
    Table2Id: TAutoIncField;
    Table2Dcode: TStringField;
    Table2Dname: TStringField;
    procedure FormClose(Sender: TObject; var Action: TCloseAction);
    procedure DBLookupComboBox1CloseUp(Sender: TObject);
    procedure SpeedButton1Click(Sender: TObject);
    procedure SpeedButton2Click(Sender: TObject);
  private
    { Private declarations }
  public
    { Public declarations }
  end;
var
  adddisactform: Tadddisactform;
implementation
uses studentunit;
{$R *.dfm}
procedure Tadddisactform.FormClose(Sender: TObject;
  var Action: TCloseAction);
begin
  Action := caFree;
  adddisactform := nil;
end;
procedure Tadddisactform.DBLookupComboBox1CloseUp(Sender: TObject);
begin
DBEdit1.Text := Table2Dcode.AsString;
end;
procedure Tadddisactform.SpeedButton1Click(Sender: TObject);
begin
  if (DBEdit1.Text <> '') and (DBEdit2.Text <> '') and (DBEdit5.Text <> '') then
    begin
      studentform.Table5.Post;
      adddisactform.Close;
    end
  else ShowMessage('You must enter data for all fields !');
end;
procedure Tadddisactform.SpeedButton2Click(Sender: TObject);
begin
  studentform.Table5.Cancel;
  adddisactform.Close;
end;
end.
```

```
unit addgradeunit;
interface
uses
  Windows, Messages, SysUtils, Variants, Classes, Graphics, Controls, Forms,
  Dialogs, Buttons, DB, DBTables;
type
  Taddgradeform = class(TForm)
    SpeedButton1: TSpeedButton;
    SpeedButton2: TSpeedButton;
    Table1: TTable;
    DataSource1: TDataSource;
    procedure FormClose(Sender: TObject; var Action: TCloseAction);
    procedure SpeedButton2Click(Sender: TObject);
  private
    { Private declarations }
  public
    { Public declarations }
  end;
var
  addgradeform: Taddgradeform;
implementation
{$R *.dfm}
procedure Taddgradeform.FormClose(Sender: TObject;
  var Action: TCloseAction);
begin
action:= cafree;
addgradeform := nil;
end;
procedure Taddgradeform.SpeedButton2Click(Sender: TObject);
begin
table1.Close;
addgradeform.Close;
end;
end.

unit addpaymentunit;
interface
uses
  Windows, Messages, SysUtils, Variants, Classes, Graphics, Controls, Forms,
  Dialogs, DB, Grids, DBTables, StdCtrls, Mask, DBCtrls, Buttons;
type
  Taddpaymentform = class(TForm)
    Label3: TLabel;
    Label5: TLabel;
    Label4: TLabel;
    Label6: TLabel;
    Label7: TLabel;
    Label8: TLabel;
    SpeedButton1: TSpeedButton;
    SpeedButton2: TSpeedButton;
```

```
    Label11: TLabel;
    DBLookupComboBox1: TDBLookupComboBox;
    DBEdit1: TDBEdit;
    DBEdit2: TDBEdit;
    DBEdit3: TDBEdit;
    DBEdit4: TDBEdit;
    DBEdit5: TDBEdit;
    DBEdit8: TDBEdit;
    Query1: TQuery;
    taxgrid: TStringGrid;
    Table2: TTable;
    DataSource2: TDataSource;
    Table2Id: TAutoIncField;
    Table2Pcode: TStringField;
    Table2Pname: TStringField;
    procedure FormCreate(Sender: TObject);
    procedure DBEdit4KeyPress(Sender: TObject; var Key: Char);
    procedure FormClose(Sender: TObject; var Action: TCloseAction);
    procedure SpeedButton1Click(Sender: TObject);
    procedure SpeedButton2Click(Sender: TObject);
    procedure DBLookupComboBox1CloseUp(Sender: TObject);
  private
    { Private declarations }
  public
    { Public declarations }
  end;
var
  addpaymentform: Taddpaymentform;
implementation
uses studentunit;
var int,int1 : integer;
{$R *.dfm}
procedure Taddpaymentform.FormCreate(Sender: TObject);
begin
taxgrid.Cells[0,0] := 'Installment Date';
taxgrid.Cells[1,0] := 'Installment Amount';
end;
procedure Taddpaymentform.DBEdit4KeyPress(Sender: TObject; var Key: Char);
var
money,em : double;
insno : integer;
tmp : string;
begin
  if Key = #13 then
    begin
      money := StrToFloat(DBEdit3.Text);
      insno := StrToInt(DBEdit4.Text);
      em := money/insno;
      tmp := FloatToStr(em);
      while Pos(',',tmp) <> 0 do Delete(tmp,Pos(',',tmp),Length(tmp)-Pos(',',tmp)+1); o
```

```
    while Pos('.',tmp) <> 0 do Delete(tmp,Pos('.',tmp),Length(tmp)-Pos('.',tmp)+1);
    em := StrToFloat(tmp);
    for int := 1 to taxgrid.RowCount-1 do
     begin
      for int1 := 0 to taxgrid.ColCount-1 do taxgrid.Cells[int1,int] := '';
     end;
    taxgrid.RowCount := insno+1;
    for int := 1 to insno do
     begin
      taxgrid.Cells[1,int] := FloatToStr(em);
     end;
    taxgrid.Cells[1,insno] := FloatToStr(money-(em*(insno-1)));
   end;
end;
procedure Taddpaymentform.FormClose(Sender: TObject;
 var Action: TCloseAction);
begin
 Action := caFree;
 addpaymentform := nil;
end;
procedure Taddpaymentform.SpeedButton1Click(Sender: TObject);
begin
 studentform.Table3.Post;  studentform.Table3.Last;
 Query1.Close;
 Query1.SQL.Clear;
 Query1.SQL.Add('insert into pdetail(Payid,Pamount,Tax,Pdate,Paid)
Values(:deger0,:deger1,:deger2,:deger3,:deger4)');
 for int := 1 to taxgrid.RowCount-1 do
  begin
   Query1.Params.Items[0].AsFloat  := studentform.Table3Id.AsFloat;
   Query1.Params.Items[1].AsFloat  := StrtoFloat(taxgrid.Cells[1,int]);
   Query1.Params.Items[2].AsFloat  := StrToFloat(DBEdit5.Text);
   Query1.Params.Items[3].AsDate   := StrToDate(taxgrid.Cells[0,int]);
   Query1.Params.Items[4].AsString := 'N';
   Query1.ExecSQL;
  end;
 addpaymentform.Close;
 studentform.Table4.Close; studentform.Table4.Open;
end;
procedure Taddpaymentform.SpeedButton2Click(Sender: TObject);
begin
 studentform.Table3.Cancel;
 addpaymentform.Close;
end;
procedure Taddpaymentform.DBLookupComboBox1CloseUp(Sender: TObject);
begin
 DBEdit1.Text := Table2Pcode.AsString;
end;
end.
```

```
unit addtermunit;
interface

uses
  Windows, Messages, SysUtils, Variants, Classes, Graphics, Controls, Forms,
  Dialogs, DB, DBTables, StdCtrls, Mask, DBCtrls, Buttons;

type
  Taddtermform = class(TForm)
    SpeedButton1: TSpeedButton;
    SpeedButton2: TSpeedButton;
    Label3: TLabel;
    Label5: TLabel;
    Label4: TLabel;
    Label6: TLabel;
    Label11: TLabel;
    DBLookupComboBox1: TDBLookupComboBox;
    DBEdit1: TDBEdit;
    DBEdit8: TDBEdit;
    Table2: TTable;
    DataSource2: TDataSource;
    DBEdit2: TDBEdit;
    DBEdit3: TDBEdit;
    Table2ID: TAutoIncField;
    Table2TID: TFloatField;
    Table2TYEAR: TStringField;
    Table2TBDATE: TDateField;
    Table2TEDATE: TDateField;
    Table2CCODE: TStringField;
    procedure FormClose(Sender: TObject; var Action: TCloseAction);
    procedure DBLookupComboBox1CloseUp(Sender: TObject);
    procedure SpeedButton1Click(Sender: TObject);
    procedure SpeedButton2Click(Sender: TObject);
  private
    { Private declarations }
  public
    { Public declarations }
  end;

var
  addtermform: Taddtermform;
implementation
uses studentunit;
{$R *.dfm}

procedure Taddtermform.FormClose(Sender: TObject;
  var Action: TCloseAction);
begin
  Action := caFree;
  addtermform := nil;end;
```

```
procedure Taddtermform.DBLookupComboBox1CloseUp(Sender: TObject);
begin
  DBEdit1.Text := Table2TID.AsString;
  DBEdit2.Text := Table2TBDATE.AsString;
  DBEdit3.Text := Table2TEDATE.AsString;
end;

procedure Taddtermform.SpeedButton1Click(Sender: TObject);
begin
 if DBLookupcombobox1.Text <> '' then
   begin
     studentform.Table6.Post;
     addtermform.Close;
   end
 else Showmessage('You must select a term');
end;

procedure Taddtermform.SpeedButton2Click(Sender: TObject);
begin
  studentform.Table6.Cancel;
  addtermform.Close;
end;
end.

unit definitionunit;
interface
uses
  Windows, Messages, SysUtils, Variants, Classes, Graphics, Controls, Forms,
  Dialogs, Buttons, ExtCtrls, Grids, DBGrids, ComCtrls, DB, DBTables,
  StdCtrls, Mask, DBCtrls;

type
  Tdefinitionform = class(TForm)
    pc: TPageControl;
    faculty: TTabSheet;
    dept: TTabSheet;
    terms: TTabSheet;
    users: TTabSheet;
    Panel2: TPanel;
    SpeedButton1: TSpeedButton;
    DBText2: TDBText;
    Label9: TLabel;
    Label10: TLabel;
    DBGrid3: TDBGrid;
    DBNavigator2: TDBNavigator;
    DBEdit9: TDBEdit;
    DBEdit10: TDBEdit;
    Table2: TTable;
    DataSource2: TDataSource;
```

Table2ID: TAutoIncField;
Table2USERNAME: TStringField;
Table2PASSWORD: TStringField;
Table2GROUPNAME: TStringField;
Label11: TLabel;
DBLookupComboBox1: TDBLookupComboBox;
usersbt: TSpeedButton;
DBText3: TDBText;
Label1: TLabel;
Label3: TLabel;
DBGrid4: TDBGrid;
DBNavigator3: TDBNavigator;
DBEdit1: TDBEdit;
DBEdit2: TDBEdit;
Table3: TTable;
DataSource3: TDataSource;
Table3Id: TAutoIncField;
Table3Code: TStringField;
Table3Name: TStringField;
DBText4: TDBText;
Label2: TLabel;
Label4: TLabel;
DBGrid1: TDBGrid;
DBNavigator4: TDBNavigator;
DBEdit3: TDBEdit;
DBEdit4: TDBEdit;
Table4: TTable;
Table4Id: TAutoIncField;
Table4Code: TStringField;
Table4Name: TStringField;
DataSource4: TDataSource;
Table4Fcode: TStringField;
Table4Fname: TStringField;
DBLookupComboBox3: TDBLookupComboBox;
Label6: TLabel;
DBEdit5: TDBEdit;
facultybt: TSpeedButton;
deptbt: TSpeedButton;
coursebt: TSpeedButton;
termsbt: TSpeedButton;
course: TTabSheet;
DBText5: TDBText;
Label5: TLabel;
Label12: TLabel;
Label13: TLabel;
DBGrid5: TDBGrid;
DBNavigator5: TDBNavigator;
DBEdit6: TDBEdit;
DBEdit11: TDBEdit;
DBEdit12: TDBEdit;

DataSource5: TDataSource;
Table5: TTable;
Table5ID: TAutoIncField;
Table5TID: TFloatField;
Table5TYEAR: TStringField;
Table5TBDATE: TDateField;
Table5TEDATE: TDateField;
Table5CCODE: TStringField;
DBEdit13: TDBEdit;
Label14: TLabel;
Label15: TLabel;
DBText6: TDBText;
Label16: TLabel;
Label17: TLabel;
Label18: TLabel;
DBGrid6: TDBGrid;
DBNavigator6: TDBNavigator;
DBEdit14: TDBEdit;
DBEdit15: TDBEdit;
DBEdit16: TDBEdit;
DBEdit17: TDBEdit;
Label19: TLabel;
DataSource6: TDataSource;
Table6: TTable;
DBEdit18: TDBEdit;
Label20: TLabel;
DBEdit19: TDBEdit;
Label21: TLabel;
DBEdit20: TDBEdit;
Label22: TLabel;
DBEdit21: TDBEdit;
Label23: TLabel;
Label24: TLabel;
DBComboBox1: TDBComboBox;
DBLookupComboBox2: TDBLookupComboBox;
Label25: TLabel;
educator: TTabSheet;
DBGrid7: TDBGrid;
educatorbt: TSpeedButton;
DataSource7: TDataSource;
Table7: TTable;
Table7ID: TAutoIncField;
Table7ENAME: TStringField;
Table7ESURNAME: TStringField;
Table7EDEPT: TStringField;
Table7EFUNCTION: TStringField;
Table7ECOUNTRY: TStringField;
Table7ECITY: TStringField;
Table7EAGE: TFloatField;
Table7EPHONENO: TFloatField;

Table7EMPHONE: TStringField;
Table7EGENDER: TStringField;
Table7EFACULTY: TStringField;
DataSource8: TDataSource;
Table8: TTable;
Table8Educatorid: TFloatField;
Table8Id: TAutoIncField;
Table8Educatorname: TStringField;
Table8Educatorsurname: TStringField;
Table8Facultycode: TStringField;
Table8Facultyname: TStringField;
Table8Deptcode: TStringField;
Table8Deptname: TStringField;
Table8Coursecode: TStringField;
Table8Coursename: TStringField;
Table8Tid: TFloatField;
Table8Tyear: TStringField;
Panel3: TPanel;
DBText7: TDBText;
Label26: TLabel;
Label27: TLabel;
Label28: TLabel;
Label29: TLabel;
Label30: TLabel;
Label31: TLabel;
Label32: TLabel;
Label33: TLabel;
Label34: TLabel;
DBNavigator7: TDBNavigator;
DBEdit22: TDBEdit;
DBEdit23: TDBEdit;
DBEdit24: TDBEdit;
DBEdit25: TDBEdit;
DBEdit26: TDBEdit;
DBEdit27: TDBEdit;
DBEdit28: TDBEdit;
DBEdit29: TDBEdit;
DBComboBox2: TDBComboBox;
Label35: TLabel;
DBEdit30: TDBEdit;
Table6CDEPTCODE: TStringField;
Table6ID: TAutoIncField;
Table6CDEPTNAME: TStringField;
Table6CCODE: TStringField;
Table6CNAME: TStringField;
Table6CTYPE: TStringField;
Table6CTEACHER: TStringField;
Table6CCREDIT: TFloatField;
Table6CREF: TStringField;
Table6CYEAR: TStringField;

```
Table6CCONTENT: TStringField;
Table6CFACULTY: TStringField;
SpeedButton7: TSpeedButton;
SpeedButton8: TSpeedButton;
DataSource1: TDataSource;
Table1: TTable;
Table1Id: TAutoIncField;
Table1Groupname: TStringField;
Table1Explanation: TStringField;
payments: TTabSheet;
DBText1: TDBText;
Label7: TLabel;
Label8: TLabel;
DBGrid2: TDBGrid;
DBNavigator1: TDBNavigator;
DBEdit7: TDBEdit;
DBEdit8: TDBEdit;
paymentsbt: TSpeedButton;
DataSource9: TDataSource;
Table9: TTable;
Table9Id: TAutoIncField;
Table9Pcode: TStringField;
Table9Pname: TStringField;
disactsbt: TSpeedButton;
disacts: TTabSheet;
DBText8: TDBText;
Label36: TLabel;
Label37: TLabel;
DBGrid8: TDBGrid;
DBNavigator8: TDBNavigator;
DBEdit31: TDBEdit;
DBEdit32: TDBEdit;
DataSource10: TDataSource;
Table10: TTable;
Table10Id: TAutoIncField;
Table10Dcode: TStringField;
Table10Dname: TStringField;
Table8Ccredit: TFloatField;
Label38: TLabel;
procedure pageselect(pagecode : integer);
procedure SpeedButton1Click(Sender: TObject);
procedure FormClose(Sender: TObject; var Action: TCloseAction);
procedure DBLookupComboBox3CloseUp(Sender: TObject);
procedure DBLookupComboBox2CloseUp(Sender: TObject);
procedure SpeedButton7Click(Sender: TObject);
procedure SpeedButton8Click(Sender: TObject);
procedure FormCreate(Sender: TObject);
procedure usersbtClick(Sender: TObject);
```

```
  private
    { Private declarations }
  public
    { Public declarations }
  end;

var
  definitionform: Tdefinitionform;

implementation
uses main, educourseunit;
{$R *.dfm}

procedure Tdefinitionform.pageselect(pagecode : integer);
begin
  usersbt.Enabled    := true;
  facultybt.Enabled  := true;
  deptbt.Enabled     := true;
  coursebt.Enabled   := true;
  termsbt.Enabled    := true;
  educatorbt.Enabled := true;
  paymentsbt.Enabled := true;
  disactsbt.Enabled := true;
  case pagecode of
    1 : begin pc.ActivePage := users;    usersbt.Enabled    := false; end;
    2 : begin pc.ActivePage := faculty;  facultybt.Enabled  := false; end;
    3 : begin pc.ActivePage := dept;     deptbt.Enabled     := false; end;
    4 : begin pc.ActivePage := course;   coursebt.Enabled   := false; end;
    5 : begin pc.ActivePage := terms;    termsbt.Enabled    := false; end;
    6 : begin pc.ActivePage := educator; educatorbt.Enabled := false; end;
    7 : begin pc.ActivePage := payments; paymentsbt.Enabled := false; end;
    8 : begin pc.ActivePage := disacts;  disactsbt.Enabled  := false; end;
  end;

end;

procedure Tdefinitionform.SpeedButton1Click(Sender: TObject);
begin
  mainform.Panel1.Visible := true;
  Close;
end;

procedure Tdefinitionform.FormClose(Sender: TObject;
  var Action: TCloseAction);
begin
  Action := caFree;
  definitionform := nil;
end;
```

```
procedure Tdefinitionform.DBLookupComboBox3CloseUp(Sender: TObject);
begin
  DbEdit5.Text := Table3Code.AsString;
end;

procedure Tdefinitionform.DBLookupComboBox2CloseUp(Sender: TObject);
begin
 DBEdit30.Text := Table4Code.AsString;
 DBEdit19.Text := Table4Fname.AsString;
end;

procedure Tdefinitionform.SpeedButton7Click(Sender: TObject);
begin
 if not Assigned (educourseform) then educourseform :=
Teducourseform.Create(Application);
 Table8.Insert;
 educourseform.DBEdit8.Text := Table7Id.AsString;
 educourseform.DBEdit6.Text := Table7Ename.AsString;
 educourseform.DBEdit7.Text := Table7Esurname.AsString;
 educourseform.Table1.Open;
 educourseform.ShowModal;
end;

procedure Tdefinitionform.SpeedButton8Click(Sender: TObject);
begin
 if Table8.RecordCount > 0 then
   begin
    if Application.MessageBox('Record is deleting !. Are you sure ?','Attention
!!',mb_YESNO) = IDYES then
      begin
        Table8.Delete;
      end;
   end;
end;

procedure Tdefinitionform.FormCreate(Sender: TObject);
begin

 faculty.TabVisible  := false;
 dept.TabVisible     := false;
 terms.TabVisible    := false;
 users.TabVisible    := false;
 course.TabVisible   := false;
 educator.TabVisible := false;
 payments.TabVisible := false;
 disacts.TabVisible  := false;
end;
```

```
procedure Tdefinitionform.usersbtClick(Sender: TObject);
begin
 pageselect(TPanel(Sender).Tag);
end;
end.

unit educourseunit;
interface
uses
  Windows, Messages, SysUtils, Variants, Classes, Graphics, Controls, Forms,
  Dialogs, Buttons, StdCtrls, Mask, DBCtrls, DB, DBTables;
type
  Teducourseform = class(TForm)
    Label1: TLabel;
    Label2: TLabel;
    Label3: TLabel;
    Label5: TLabel;
    DBLookupComboBox1: TDBLookupComboBox;
    DBLookupComboBox2: TDBLookupComboBox;
    DBEdit1: TDBEdit;
    DBEdit2: TDBEdit;
    DBEdit3: TDBEdit;
    DBEdit4: TDBEdit;
    Label4: TLabel;
    Label6: TLabel;
    Label7: TLabel;
    Label8: TLabel;
    DBLookupComboBox3: TDBLookupComboBox;
    DBEdit5: TDBEdit;
    Label9: TLabel;
    Label10: TLabel;
    DBEdit6: TDBEdit;
    DBEdit7: TDBEdit;
    SpeedButton1: TSpeedButton;
    SpeedButton2: TSpeedButton;
    Label11: TLabel;
    DBEdit8: TDBEdit;
    Query1: TQuery;
    Table1: TTable;
    DataSource1: TDataSource;
    Table1CDEPTCODE: TStringField;
    Table1ID: TAutoIncField;
    Table1CDEPTNAME: TStringField;
    Table1CCODE: TStringField;
    Table1CNAME: TStringField;
    Table1CTYPE: TStringField;
    Table1CTEACHER: TStringField;
    Table1CCREDIT: TFloatField;
    Table1CREF: TStringField;
```

```
    Table1CYEAR: TStringField;
    Table1CCONTENT: TStringField;
    Table1CFACULTY: TStringField;
    Label12: TLabel;
    DBEdit9: TDBEdit;
    procedure FormClose(Sender: TObject; var Action: TCloseAction);
    procedure DBLookupComboBox1CloseUp(Sender: TObject);
    procedure DBLookupComboBox2CloseUp(Sender: TObject);
    procedure DBLookupComboBox3CloseUp(Sender: TObject);
    procedure SpeedButton1Click(Sender: TObject);
    procedure SpeedButton2Click(Sender: TObject);
  private
    { Private declarations }
  public
    { Public declarations }
  end;
var
  educourseform: Teducourseform;
implementation
uses definitionunit;
{$R *.dfm}
procedure Teducourseform.FormClose(Sender: TObject;
  var Action: TCloseAction);
begin
  Action := caFree;
  educourseform := nil;
end;
procedure Teducourseform.DBLookupComboBox1CloseUp(Sender: TObject);
begin
  DBEdit1.Text := definitionform.Table5ID.AsString;
end;
procedure Teducourseform.DBLookupComboBox2CloseUp(Sender: TObject);
begin
  DBEdit2.Text := definitionform.Table4Code.AsString;
  DBEdit3.Text := definitionform.Table4Fname.AsString;
  DBEdit4.Text := definitionform.Table4Fcode.AsString;
end;
procedure Teducourseform.DBLookupComboBox3CloseUp(Sender: TObject);
begin
  DBEdit5.Text := Table1CCODE.AsString;
  DBEdit9.Text := Table1ccredit.AsString;
end;
procedure Teducourseform.SpeedButton1Click(Sender: TObject);
begin

Query1.Close;
  Query1.Params.Items[0].AsFloat  := StrToFloat(DBEdit8.Text);
  Query1.Params.Items[1].AsFloat  := StrToFloat(DBEdit1.Text);
  Query1.Params.Items[2].AsString := DBEdit5.Text;
  Query1.Open;
```

```
 if Query1.RecordCount > 0 then showmessage('The course which you selected before
that time selected for same term !. You must select another course  or term.')
  else
   begin
    definitionform.Table8.Post;
    table1.Close;
    educourseform.Close;
   end;
end;
procedure Teducourseform.SpeedButton2Click(Sender: TObject);
begin
 table1.Close;
 definitionform.Table8.Cancel;
 educourseform.Close;
end;
end.

unit searchunit;
interface
uses
 Windows, Messages, SysUtils, Variants, Classes, Graphics, Controls, Forms,
 Dialogs, DB, DBTables, Buttons;
type
 Tsearchform = class(TForm)
  DataSource1: TDataSource;
  searchtable: TTable;
  SpeedButton2: TSpeedButton;
  procedure FormClose(Sender: TObject; var Action: TCloseAction);
  procedure SpeedButton2Click(Sender: TObject);
 private
  { Private declarations }
 public
  { Public declarations }
 end;
var
 searchform: Tsearchform;
implementation
uses main;
{$R *.dfm}
procedure Tsearchform.FormClose(Sender: TObject; var Action: TCloseAction);
begin
 Action := caFree;
searchform := nil;
end;
procedure Tsearchform.SpeedButton2Click(Sender: TObject);
begin
searchtable.Close;
searchform.Close;
mainform.Panel1.visible:= true;
end;end.
```

```
unit listunit;
interface
uses
  Windows, Messages, SysUtils, Variants, Classes, Graphics, Controls, Forms,
  Dialogs, DB, Grids, DBGrids, ExtCtrls, DBTables, Buttons, ComCtrls,
  StdCtrls;
type
  Tlistform = class(TForm)
    PageControl1: TPageControl;
    teacher: TTabSheet;
    SpeedButton1: TSpeedButton;
    SpeedButton2: TSpeedButton;
    tquery: TQuery;
    DataSource1: TDataSource;
    Panel1: TPanel;
    tgrid: TDBGrid;
    tqueryEducatorid: TFloatField;
    tqueryId: TIntegerField;
    tqueryEducatorname: TStringField;
    tqueryEducatorsurname: TStringField;
    tqueryFacultycode: TStringField;
    tqueryFacultyname: TStringField;
    tqueryDeptcode: TStringField;
    tqueryDeptname: TStringField;
    tqueryCoursecode: TStringField;
    tqueryCoursename: TStringField;
    tqueryTid: TFloatField;
    tqueryTyear: TStringField;
    Label11: TLabel;
    Label1: TLabel;
    Label2: TLabel;
    Label3: TLabel;
    Label4: TLabel;
    Label5: TLabel;
    Edit1: TEdit;
    Edit2: TEdit;
    Edit3: TEdit;
    Edit4: TEdit;
    Edit5: TEdit;
    Edit6: TEdit;
    tqueryCcredit: TFloatField;
    procedure search(where : integer);
    procedure tgridTitleClick(Column: TColumn);
    procedure FormCreate(Sender: TObject);
    procedure Edit1KeyPress(Sender: TObject; var Key: Char);
    procedure FormClose(Sender: TObject; var Action: TCloseAction);
    procedure SpeedButton2Click(Sender: TObject);
    procedure SpeedButton1Click(Sender: TObject);
```

```
private
  { Private declarations }
public
   { Public declarations }
 end;
var
   listform: Tlistform;
  goingto : integer;
implementation
uses addcourseunit;
var
tqry,tfield,topposide : string;
int : integer;
{$R *.dfm}
procedure Tlistform.search(where : integer);
var qry,value : string;
begin
  case where of
    0 :
      begin
        qry := '';
        value := edit1.Text;
        if value <> '' then begin while Pos('*',value) > 0 do value[Pos('*',value)] := '%';
 qry := qry+' where Educatorname LIKE '"+value+"'"; end
         else qry := qry+' where Educatorname <> "x" ';
        value := edit2.Text;
        if value <> '' then begin while Pos('*',value) > 0 do value[Pos('*',value)] := '%';
 qry := qry+' and Educatorsurname LIKE '"+value+"'"; end;
        value := edit3.Text;
        if value <> '' then begin while Pos('*',value) > 0 do value[Pos('*',value)] := '%';
 qry := qry+' and Facultyname LIKE '"+value+"'"; end;
        value := edit4.Text;
        if value <> '' then begin while Pos('*',value) > 0 do value[Pos('*',value)] := '%';
 qry := qry+' and Deptname LIKE '"+value+"'"; end;
        value := edit5.Text;
        if value <> '' then begin while Pos('*',value) > 0 do value[Pos('*',value)] := '%';
 qry := qry+' and Coursecode LIKE '"+value+"'"; end;
        value := edit6.Text;
        if value <> '' then begin while Pos('*',value) > 0 do value[Pos('*',value)] := '%';
 qry := qry+' and Coursename LIKE '"+value+"'"; end;
        tqry := 'select* from educourse'+qry;
        tquery.Close;
        tquery.SQL.Clear;
        tquery.SQL.Add(tqry);
        tquery.SQL.Add('Order By '+tfield+topposide);
        tquery.Open;
       end;
   end;
 end;
```

```
procedure Tlistform.tgridTitleClick(Column: TColumn);
begin
  for int := 0 to tGrid.Columns.Count-1 do tgrid.Columns.Items[int].Title.Font.Color :=
clWhite;
  Column.Title.Font.Color := clLime;
  tfield := Column.FieldName;
  tquery.Close;
  tquery.SQL.Clear;
  tquery.SQL.Add(tqry);
  tquery.SQL.Add('Order By '+tfield+topposide);
  tquery.Open;
  if topposide <> '' then topposide := '' else topposide := ' DESC ';
end;
procedure Tlistform.FormCreate(Sender: TObject);
begin
  tqry := 'Select* from educourse';
  tfield := 'Tyear';
  topposide := ' DESC ';
end;
procedure Tlistform.Edit1KeyPress(Sender: TObject; var Key: Char);
begin
  if Key = #13 then search(TListbox(Sender).Tag);
end;
procedure Tlistform.FormClose(Sender: TObject; var Action: TCloseAction);
begin
  Action := caFree;
  listform := nil;
end;
procedure Tlistform.SpeedButton2Click(Sender: TObject);
begin
  listform.Close;
end;

procedure Tlistform.SpeedButton1Click(Sender: TObject);
begin
  case goingto of
    0 :
      begin
        if tquery.RecordCount > 0 then
          begin
            addcourseform.DBEdit3.Text                                    :=
tquery.FieldByName('Educatorname').AsString;
            addcourseform.DBEdit5.Text                                    :=
tquery.FieldByName('Educatorsurname').AsString;
            addcourseform.DBEdit6.Text := tquery.FieldByName('Deptcode').AsString;
            addcourseform.DBEdit7.Text := tquery.FieldByName('Deptname').AsString;
            addcourseform.DBEdit9.Text := tquery.FieldByName('Facultycode').AsString;
            addcourseform.DBEdit10.Text := tquery.FieldByName('Facultyname').AsString;
            addcourseform.DBEdit1.Text := tquery.FieldByName('Coursecode').AsString;
            addcourseform.DBEdit2.Text := tquery.FieldByName('Coursename').AsString;
```

```
        addcourseform.DBEdit11.Text := tquery.FieldByName('Ccredit').AsString;
        listform.Close;
       end;
     end;
  end;
end;
end.

unit main;
interface
uses
  Windows, Messages, SysUtils, Variants, Classes, Graphics, Controls, Forms,
  Dialogs, StdCtrls, Buttons, ExtCtrls, DB, DBCtrls, Mask, DBTables,
  ComCtrls, Menus;
type
  Tmainform = class(TForm)
    Panel1: TPanel;
    Panel2: TPanel;
    user: TComboBox;
    Label1: TLabel;
    Label2: TLabel;
    Label3: TLabel;
    usercode: TEdit;
    Query1: TQuery;
    ComboBox1: TComboBox;
    st1: TStatusBar;
    Label4: TLabel;
    Label5: TLabel;
    okbutton: TSpeedButton;
    cancbutton: TSpeedButton;
    definitionbutton: TSpeedButton;
    exitbutton: TSpeedButton;
    SpeedButton1: TSpeedButton;
    Image1: TImage;
    adminpanel: TPanel;
    secretarypanel: TPanel;
    advisorpanel: TPanel;
    SpeedButton2: TSpeedButton;
    SpeedButton3: TSpeedButton;
    accountancypanel: TPanel;
    shutdownbt: TSpeedButton;
    SpeedButton4: TSpeedButton;
    SpeedButton5: TSpeedButton;
    SpeedButton6: TSpeedButton;
    SpeedButton7: TSpeedButton;
    SpeedButton8: TSpeedButton;
    SpeedButton9: TSpeedButton;
    SpeedButton10: TSpeedButton;
    SpeedButton11: TSpeedButton;
    SpeedButton12: TSpeedButton;
```

```
    SpeedButton13: TSpeedButton;
    SpeedButton14: TSpeedButton;
    SpeedButton15: TSpeedButton;
    SpeedButton16: TSpeedButton;
    SpeedButton17: TSpeedButton;
    SpeedButton18: TSpeedButton;
    SpeedButton19: TSpeedButton;
    SpeedButton20: TSpeedButton;
    procedure shutdown;
    procedure FormCreate(Sender: TObject);
    procedure ComboBox1Change(Sender: TObject);
    procedure definitionbuttonClick(Sender: TObject);
    procedure okbuttonClick(Sender: TObject);
    procedure cancbuttonClick(Sender: TObject);
    procedure exitbuttonClick(Sender: TObject);
    procedure usercodeKeyPress(Sender: TObject; var Key: Char);
    procedure SpeedButton1Click(Sender: TObject);
    procedure FormResize(Sender: TObject);
    procedure shutdownbtClick(Sender: TObject);
    procedure FormClose(Sender: TObject; var Action: TCloseAction);
    procedure SpeedButton17Click(Sender: TObject);
private
    { Private declarations }
  public
    { Public declarations }
  end;

var
  mainform: Tmainform;
  groupcode : integer;
implementation
uses studentunit,definitionunit, searchunit;var
int,int1 : integer;
{$R *.dfm}
procedure Tmainform.SpeedButton1Click(Sender: TObject);
begin
  if not Assigned (studentform) then studentform := Tstudentform.Create(Application);
  Panel1.Visible := false;
  studentform.Table1.Open;
  studentform.Table2.Open;
  studentform.Table3.Open;
  studentform.Table4.Open;
  studentform.Table5.Open;
  studentform.Table6.Open;
  studentform.Table7.Open;
  studentform.Table8.Open;
```

```
case groupcode of
  0 :
   begin
     studentform.payments.TabVisible := true;
     studentform.PageControl1.ActivePage := studentform.disacts;
     studentform.SpeedButton4.Visible := true;
     studentform.terms.TabVisible := false ;end;
   1:
   begin
   studentform.payments.TabVisible := false;
   studentform.PageControl1.ActivePage := studentform.terms;
   studentform.terms.TabVisible := true;
   studentform.DBNavigator1.VisibleButtons                                    :=
[nbFirst,nbPrior,nbNext,nbLast,nbInsert,nbEdit,nbPost,nbCancel];
   studentform.SpeedButton5.Visible :=false;
   studentform.SpeedButton6.Visible :=false;
   end;
   2:
   begin
   studentform.payments.TabVisible := false;
   studentform.PageControl1.ActivePage := studentform.terms;
   studentform.terms.TabVisible := true;
   studentform.SpeedButton5.Visible :=false;
   studentform.SpeedButton6.Visible :=false;
   studentform.DBNavigator1.VisibleButtons                                    :=
[nbFirst,nbPrior,nbNext,nbLast,nbInsert,nbEdit,nbPost,nbCancel];
end;
  end;
  panel1.Visible:=false;
    studentform.Show;
end;
procedure Tmainform.shutdown;
begin
 adminpanel.Visible := false;    secretarypanel.Visible := false;
 advisorpanel.Visible := false;  accountancypanel.Visible := false;
 Panel1.Visible := false;
 Panel2.Visible := true;
 combobox1.Items.Clear; user.Items.Clear;
 usercode.Text := '';
 Query1.Close;
 Query1.SQL.Clear;
 Query1.SQL.Add('select* from groups');
 Query1.Open;
 if Query1.RecordCount > 0 then
  begin
   for int := 0 to Query1.RecordCount-1 do
    begin
     ComboBox1.Items.Add(Query1.FieldByName('Groupname').AsString);
     Query1.Next;
    end;
```

```
  end;
 if combobox1.Items.Count > 0 then
  begin
    combobox1.ItemIndex := 0;
    Query1.Close;
    Query1.SQL.Clear;
    Query1.SQL.Add('select*  from  users  where  Groupname  =
'"+combobox1.Items.Strings[0]+"');
    Query1.Open;
    if Query1.RecordCount > 0 then
      begin
       for int := 0 to Query1.RecordCount-1 do
        begin
          user.Items.Add(Query1.FieldByName('UserName').AsString);
          Query1.Next;
        end;
      end;
   end;
 end;
 procedure Tmainform.FormCreate(Sender: TObject);
 var side,top : variant;
 begin
  adminpanel.Align      := alClient;
  secretarypanel.Align  := alClient;
  advisorpanel.Align    := alClient;
  accountancypanel.Align := alClient;
  side := (mainform.Width/2)-(Panel2.Width/2);
  top := (mainform.ClientHeight/2)-(Panel2.Height/2);
 Panel2.Left := side;
  Panel2.Top := top;
  shutdown;
 end;
 procedure Tmainform.ComboBox1Change(Sender: TObject);
 begin
   user.Items.Clear;
   Query1.Close;
   Query1.SQL.Clear;
   Query1.SQL.Add('select* from users where Groupname = '"+combobox1.Text+"');
   Query1.Open;
  if Query1.RecordCount > 0 then
    begin
      for int := 0 to Query1.RecordCount-1 do
       begin
         user.Items.Add(Query1.FieldByName('UserName').AsString);
         Query1.Next;  query1.Refresh;
       end;
    end;
  end;
```

```
procedure Tmainform.definitionbuttonClick(Sender: TObject);
begin
 if not Assigned(definitionform) then
definitionform=Tdefinitionform.Create(Application);

 definitionform.Table1.Open;
 definitionform.Table2.Open;
 definitionform.Table3.Open;
 definitionform.Table4.Open;
 definitionform.Table5.Open;
 definitionform.Table6.Open;
 definitionform.Table7.Open;
 definitionform.Table8.Open;
 definitionform.Table9.Open;
 definitionform.Table10.Open;
 definitionform.usersbt.Visible    := false;
 definitionform.facultybt.Visible  := false;
 definitionform.deptbt.Visible     := false;
 definitionform.coursebt.Visible   := false;
 definitionform.termsbt.Visible    := false;
 definitionform.educatorbt.Visible := false;
 definitionform.paymentsbt.Visible := false;
 definitionform.disactsbt.Visible  := false;
 case groupcode of
  0 :
   begin
    definitionform.usersbt.Visible    := true;
    definitionform.facultybt.Visible  := true;
    definitionform.deptbt.Visible     := true;
    definitionform.disactsbt.Visible := true;
    definitionform.usersbt.Click;
   end;
  1 :
   begin
    definitionform.coursebt.Visible   := true;
    definitionform.termsbt.Visible    := true;
    definitionform.educatorbt.Visible := true;
    definitionform. DBNavigator5.VisibleButtons :=
[nbFirst,nbPrior,nbNext,nbLast,nbInsert,nbEdit,nbPost,nbCancel];
    definitionform. DBNavigator6.VisibleButtons :=
[nbFirst,nbPrior,nbNext,nbLast,nbInsert,nbEdit,nbPost,nbCancel];
    definitionform. DBNavigator7.VisibleButtons :=
[nbFirst,nbPrior,nbNext,nbLast,nbInsert,nbEdit,nbPost,nbCancel];
    definitionform.coursebt.Click;
   end;
   2:
   begin
    definitionform.coursebt.Visible   := true;
    definitionform.termsbt.Visible    := true;
    definitionform.educatorbt.Visible := false;
```

```
      definitionform.                DBNavigator5.VisibleButtons                :=
[nbFirst,nbPrior,nbNext,nbLast,nbEdit,nbPost,nbCancel];
      definitionform.                DBNavigator6.VisibleButtons                :=
[nbFirst,nbPrior,nbNext,nbLast,nbEdit,nbPost,nbCancel];
      definitionform.coursebt.Click;
     end;
   3 :
    begin
     definitionform.paymentsbt.Visible  := true;
     definitionform.paymentsbt.Click;
    end;
 end;

 Panel1.Visible := false;
 definitionform.Show;
end;
procedure Tmainform.okbuttonClick(Sender: TObject);
var
opts : TLocateOptions;
 pass : string;
begin
 Query1.Close;
 Query1.SQL.Clear;
 Query1.SQL.Add('select* from users');
 Query1.Open;
 if ComboBox1.Items.Count > 0 then
  begin
   opts := [loCaseInsensitive];
   if Query1.Locate('UserName',user.Text,opts) then
   begin
    pass := Query1.FieldByName('PASSWORD').AsString;
    if usercode.Text = pass then
     begin
      panel2.Visible := false;
      Panel1.Visible := true;
      st1.Panels[0].Text          :=          'Active          User          :
'+Query1.FieldByName('UserName').AsString;
      st1.Panels[1].Text          :=          'Active          Group          :
'+Query1.FieldByName('Groupname').AsString;
      if Query1.FieldByName('Groupname').AsString = 'Admin'       then groupcode
:= 0;
      if Query1.FieldByName('Groupname').AsString = 'Secretary'   then groupcode
:= 1;
      if Query1.FieldByName('Groupname').AsString = 'Advisor'     then groupcode
:= 2;
      if Query1.FieldByName('Groupname').AsString = 'Accountancy' then groupcode
:= 3;
```

```
    case groupcode of
            0 : adminpanel.Visible      := true;
            1 : secretarypanel.Visible  := true;
            2 : advisorpanel.Visible    := true;
            3 : accountancypanel.Visible := true;
          end;
        end
      else begin ShowMessage("Invalid password,Try again !!!'); user.SetFocus; end;
    end else showmessage('Invalid user name !');
  end else showmessage('You must select an user!');
end;
procedure Tmainform.cancbuttonClick(Sender: TObject);
begin
  mainform.Close;
end;
procedure Tmainform.exitbuttonClick(Sender: TObject);
begin
  mainform.Close;
end;
procedure Tmainform.usercodeKeyPress(Sender: TObject; var Key: Char);
begin
  if Key = #13 then okbutton.Click;
end;

procedure Tmainform.FormResize(Sender: TObject);
var side,top : variant;
begin

 side := (mainform.Width/2)-(Panel2.Width/2);
 top := (mainform.ClientHeight/2)-(Panel2.Height/2);
 Panel2.Left := side;
 Panel2.Top := top;
end;

procedure Tmainform.shutdownbtClick(Sender: TObject);
begin
  shutdown;
end;
procedure Tmainform.FormClose(Sender: TObject; var Action: TCloseAction);
begin
  if    Application.MessageBox('The    program    will    close!    Are    you    sure
?','Attention',mb_yesno) = idno then Abort;
 end;
```

```
procedure Tmainform.SpeedButton17Click(Sender: TObject);
begin
 if not Assigned (searchform) then searchform := Tsearchform.Create(Application);
 panel1.Visible := false;
 searchform.searchtable.open;
 searchform.Show;
 end;
end.

unit payinsunit;
interface
uses
 Windows, Messages, SysUtils, Variants, Classes, Graphics, Controls, Forms,
 Dialogs, ExtCtrls, StdCtrls, DBCtrls, Mask, Buttons;
type
 Tpayinsform = class(TForm)
  newaddedit: TLabel;
  Label1: TLabel;
  Label2: TLabel;
  Label3: TLabel;
  Label4: TLabel;
  Label5: TLabel;
  Label6: TLabel;
  Label7: TLabel;
  SpeedButton1: TSpeedButton;
  SpeedButton2: TSpeedButton;
  DBComboBox1: TDBComboBox;
  Bevel1: TBevel;
  Edit1: TEdit;
  Edit2: TEdit;
  Edit3: TEdit;
  Edit4: TEdit;
  DBText1: TDBText;
  DBText2: TDBText;
  DBText3: TDBText;
  DBEdit1: TDBEdit;
  DBEdit2: TDBEdit;
  DBEdit3: TDBEdit;
  DBEdit4: TDBEdit;
  procedure FormClose(Sender: TObject; var Action: TCloseAction);
  procedure FormCreate(Sender: TObject);
  procedure SpeedButton1Click(Sender: TObject);
  procedure SpeedButton2Click(Sender: TObject);
 private
  { Private declarations }
 public
  { Public declarations }
 end;
var
 payinsform: Tpayinsform;
```

```
implementation
uses studentunit;
{$R *.dfm}
procedure Tpayinsform.FormClose(Sender: TObject; var Action: TCloseAction);
begin
  Action := caFree;
  payinsform := nil;
end;
procedure Tpayinsform.FormCreate(Sender: TObject);
begin
  DBCombobox1.ItemIndex := 0;
end;
procedure Tpayinsform.SpeedButton1Click(Sender: TObject);
begin
  if DBCombobox1.ItemIndex > -1 then
    begin
      studentform.Table4.Post;
      payinsform.Close;
    end
  else ShowMessage('You must select a Payment Place');
end;
procedure Tpayinsform.SpeedButton2Click(Sender: TObject);
begin
  studentform.Table4.Cancel;
  payinsform.Close;
end;
end.

nit studentunit;
interface
uses
  Windows, Messages, SysUtils, Variants, Classes, Graphics, Controls, Forms,
  Dialogs, StdCtrls, ComCtrls, Buttons, ExtCtrls, DBCtrls, DBTables, DB,
  Mask, ExtDlgs, Grids, DBGrids;

type
  Tstudentform = class(TForm)
    Panel2: TPanel;
    anamenubutton: TSpeedButton;
    imagebt: TSpeedButton;
    DataSource1: TDataSource;
    Table1: TTable;
    Query1: TQuery;
    DBNavigator1: TDBNavigator;
    photodlg: TOpenPictureDialog;
    Table2: TTable;
    DataSource2: TDataSource;
    Table2Id: TAutoIncField;
    Table2Code: TStringField;
```

Table2Name: TStringField;
Table2Fcode: TStringField;
Table2Fname: TStringField;
PageControl1: TPageControl;
Panel1: TPanel;
Label1: TLabel;
Label2: TLabel;
Label3: TLabel;
Label4: TLabel;
Label5: TLabel;
Label20: TLabel;
Label22: TLabel;
DBText1: TDBText;
Label23: TLabel;
Label12: TLabel;
Label10: TLabel;
DBEdit1: TDBEdit;
DBEdit2: TDBEdit;
DBEdit3: TDBEdit;
DBEdit4: TDBEdit;
DBEdit12: TDBEdit;
DBEdit13: TDBEdit;
DBEdit22: TDBEdit;
DBComboBox1: TDBComboBox;
DBImage1: TDBImage;
DBLookupComboBox1: TDBLookupComboBox;
payments: TTabSheet;
disacts: TTabSheet;
terms: TTabSheet;
detailspanel: TPanel;
Label14: TLabel;
Label18: TLabel;
Label19: TLabel;
Label21: TLabel;
Label7: TLabel;
Label8: TLabel;
Label6: TLabel;
Label9: TLabel;
Label11: TLabel;
newaddedit: TLabel;
Label13: TLabel;
Label15: TLabel;
DBEdit17: TDBEdit;
DBEdit18: TDBEdit;
DBEdit20: TDBEdit;
DBEdit21: TDBEdit;
DBEdit6: TDBEdit;
DBEdit7: TDBEdit;
DBEdit8: TDBEdit;
DBEdit9: TDBEdit;

```
    DBEdit10: TDBEdit;
    DBEdit11: TDBEdit;
    DBEdit14: TDBEdit;
    DBEdit15: TDBEdit;
    detailsbt: TSpeedButton;
    Panel3: TPanel;
    DBGrid1: TDBGrid;
    Splitter1: TSplitter;
    Panel4: TPanel;
    DataSource3: TDataSource;
    Table3: TTable;
    SpeedButton1: TSpeedButton;
    SpeedButton2: TSpeedButton;
    DataSource4: TDataSource;
    Table4: TTable;
    Table4Payid: TFloatField;
    Table4Id: TAutoIncField;
    Table4Pamount: TFloatField;
    Table4Ppamount: TFloatField;
    Table4Tax: TFloatField;
    Table4Pdate: TDateField;
    Table4Ppdate: TDateField;
    Table4Paid: TStringField;
    Table4Ptype: TStringField;
    Table3Sid: TStringField;
    Table3Id: TAutoIncField;
    Table3Pcode: TStringField;
    Table3Pname: TStringField;
    Table3Pdate: TDateField;
    Table3Inscount: TFloatField;
    Table3Pamount: TFloatField;
    Table3Paid: TStringField;
    Table3Tax: TFloatField;
    Table4Pdelay: TFloatField;
    Panel5: TPanel;
    DBGrid2: TDBGrid;
    Panel6: TPanel;
    SpeedButton3: TSpeedButton;
    SpeedButton4: TSpeedButton;
    Panel7: TPanel;
    SpeedButton5: TSpeedButton;
    SpeedButton6: TSpeedButton;
    DBGrid3: TDBGrid;
    DataSource5: TDataSource;
    Table5: TTable;
    Table5Sid: TStringField;
    Table5Id: TAutoIncField;
    Table5Ddate: TDateField;
    Table5Dcode: TStringField;
    Table5Dname: TStringField;
```

```
    Table5Explanation: TStringField;
    Table5Result: TStringField;
    Panel9: TPanel;
    DBGrid4: TDBGrid;
    Panel8: TPanel;
    SpeedButton7: TSpeedButton;
    SpeedButton8: TSpeedButton;
    Splitter2: TSplitter;
    Panel10: TPanel;
    Panel11: TPanel;
    SpeedButton9: TSpeedButton;
    SpeedButton10: TSpeedButton;
    DBGrid5: TDBGrid;
    DataSource6: TDataSource;
    Table6: TTable;
    DataSource7: TDataSource;
    Table7: TTable;
    Table6Stid: TFloatField;
    Table6Id: TAutoIncField;
    Table6Tid: TFloatField;
    Table6Tyear: TStringField;
    Table6Tbdate: TDateField;
    Table6Tedate: TDateField;
    Splitter3: TSplitter;
    Panel12: TPanel;
    Panel13: TPanel;
    SpeedButton11: TSpeedButton;
    SpeedButton12: TSpeedButton;
    DBGrid6: TDBGrid;
    DataSource8: TDataSource;
    Table8: TTable;
    Table8Cid: TFloatField;
    Table8Id: TAutoIncField;
    Table8Tid: TFloatField;
    Table8Stid: TStringField;
    Table8Explanation: TStringField;
    Table8Grade: TFloatField;
    Table8GradeA: TStringField;
    Table8Percent: TFloatField;
    Table7Tid: TFloatField;
    Table7Id: TAutoIncField;
    Table7Stid: TStringField;
    Table7Ccode: TStringField;
    Table7Cname: TStringField;
    Table7Ccredit: TFloatField;
    Table7Tname: TStringField;
    Table7Tsurname: TStringField;
    Table7Tdeptcode: TStringField;
    Table7Tdeptname: TStringField;
    Table7Tfacultycode: TStringField;
```

```
    Table7Tfacultyname: TStringField;
    Table7Caverage: TFloatField;
    Table7Cgrade: TStringField;
    DBEdit5: TDBEdit;
    Label24: TLabel;
    DBEdit19: TDBEdit;
    Label17: TLabel;
    Table1Stid: TStringField;
    Table1Id: TAutoIncField;
    Table1Stphoto: TBlobField;
    Table1Stfirstname: TStringField;
    Table1Stmidname: TStringField;
    Table1Stsurname: TStringField;
    Table1Admissiondate: TDateField;
    Table1Stsex: TStringField;
    Table1Stfathername: TStringField;
    Table1Stmothername: TStringField;
    Table1Stplaceofbirth: TStringField;
    Table1Stdateofbirth: TDateField;
    Table1Stnewaddr: TStringField;
    Table1Stoldaddr: TStringField;
    Table1Stpphone: TStringField;
    Table1Stmphone: TStringField;
    Table1Stcountry: TStringField;
    Table1Stprovince: TStringField;
    Table1Stnationality: TStringField;
    Table1Sthighschool: TStringField;
    Table1StgradeofdateH: TDateField;
    Table1StHbranch: TStringField;
    Table1StHgpa: TFloatField;
    Table1Stemail: TStringField;
    Table1Stdept: TStringField;
    Table1Stfaculty: TStringField;
    Table1Numberid: TFloatField;
    Table1Stpassnum: TStringField;
    procedure hesapla;
    procedure Table1BeforePost(DataSet: TDataSet);
    procedure anamenubuttonClick(Sender: TObject);
    procedure FormClose(Sender: TObject; var Action: TCloseAction);
    procedure Table1AfterInsert(DataSet: TDataSet);
    procedure imagebtClick(Sender: TObject);
    procedure DBLookupComboBox1CloseUp(Sender: TObject);
    procedure detailsbtClick(Sender: TObject);
    procedure SpeedButton1Click(Sender: TObject);
    procedure Table3BeforePost(DataSet: TDataSet);
    procedure SpeedButton2Click(Sender: TObject);
    procedure SpeedButton3Click(Sender: TObject);
    procedure SpeedButton4Click(Sender: TObject);
    procedure SpeedButton5Click(Sender: TObject);
    procedure SpeedButton6Click(Sender: TObject);
```

```
    procedure Table1BeforeInsert(DataSet: TDataSet);
    procedure SpeedButton7Click(Sender: TObject);
    procedure SpeedButton9Click(Sender: TObject);
    procedure SpeedButton8Click(Sender: TObject);
    procedure SpeedButton10Click(Sender: TObject);
    procedure SpeedButton11Click(Sender: TObject);
  private
    { Private declarations }
  public
    { Public declarations }
  end;

var
  studentform: Tstudentform;

implementation
uses main,addpaymentunit, payinsunit, adddisactunit, addtermunit,
  addcourseunit, addgradeunit;
var
int,int1,howmanytax : integer;
tmp,tmp1 : string;
{$R *.dfm}

function gunbul(ay,yil : integer) : integer;
var
ayingunu : integer;
tt : TDate;
tts,ttp : string;
begin
  if (ay = 1 ) or ( ay = 3 ) or ( ay = 5 ) or ( ay = 7 ) or ( ay = 8 ) or ( ay = 10 ) or ( ay =
12) then ayingunu := 31;
  if (ay = 4 ) or ( ay = 6 ) or ( ay = 9 ) or ( ay = 11 ) then ayingunu := 30;
  if ay = 2 then
   begin
     tt  := StrToDate('01.03.'+IntToStr(yil));
     tt  := tt-1;
     tts := DateToStr(tt);
     ttp := Copy(tts,1,2);
     ayingunu := StrToInt(ttp);
   end;
 result :=  ayingunu;
end;

procedure Tstudentform.hesapla;

var
gg,kdvs,fborcus,alinans,kalans,birimucs,tarih,tarih1,vtarih,fdur  : string;
year,mounth,day,ayingunu,eklenecekay,gun,gun1,ay,ay1,yil,yil1,sure : integer;
vtarihi,cikistar,teskeretarih,ktarih : TDate;
normalt : boolean;
```

```
begin
   kdvs := '';fborcus := '';alinans := '';kalans := '';birimucs := '';tarih := '';tarih1 := '';vtarih
:= '';fdur := '';
   eklenecekay := 0;gun := 0;gun1 := 0;ay := 0;ay1 := 0;yil := 0;yil1 := 0;
    cikistar := Table4Pdate.AsDateTime;
    vtarihi  := Date();
    tarih    := DateToStr(vtarihi);
    if vtarihi > cikistar then
     begin
       sure := 0;
       vtarih := DateToStr(vtarihi);
       gun := StrToInt(Copy(vtarih,1,2));
       ay  := StrToInt(Copy(vtarih,4,2));
       yil := StrToInt(Copy(vtarih,7,4));
       tarih1 := DateToStr(cikistar);
       gun1 := StrToInt(Copy(tarih1,1,2));
       ay1  := StrToInt(Copy(tarih1,4,2));
       yil1 := StrToInt(Copy(tarih1,7,4));
       year   := yil1;
       mounth := ay1;
       day    := gun1;
       ayingunu := gunbul(mounth,year);
       if day > ayingunu then gg := IntToStr(ayingunu)  else gg := IntToStr(day);
       teskeretarih := StrToDate(gg+'.'+IntToStr(mounth)+'.'+IntToStr(year));
      ktarih := Table4Pdate.AsDateTime;
      if (ktarih > mezuniyettarih) or (ktarih = mezuniyetttarih) then
       begin
         tarih1 := DateToStr(Table4Pdate.AsDateTime);
         gun1 := StrToInt(Copy(tarih1,1,2));
         ay1  := StrToInt(Copy(tarih1,4,2));
         yil1 := StrToInt(Copy(tarih1,7,4));
         sure := 0;
       end;
       normalt := true;
       if yil > yil1 then begin normalt := true; eklenecekay := ((12-ay1)+ay+(12*(yil-
yil1-1)))- sure; end;
       if yil = yil1 then
         begin
           if ay>ay1 then begin normalt := true; eklenecekay := (ay-ay1)-sure; end
            else begin normalt := false; eklenecekay := 0;end;
           if ay = ay1 then begin normalt := true; end;
         end;
       if yil1 > yil then begin normalt := false; eklenecekay := 0; end;
       if eklenecekay < 0 then begin eklenecekay := 0; normalt := false;end;
       if (normalt = true) and (eklenecekay = 0) then
         begin
           if gun > gun1 then begin eklenecekay := eklenecekay+1;end
         end;
```

```
//          showmessage('eklenecek ay : '+IntToStr(eklenecekay));
     howmanytax := eklenecekay
   end else howmanytax := 0;
 end;


 procedure Tstudentform.Table1BeforePost(DataSet: TDataSet);
 begin
  if Table1StId.AsString = '' then
    begin
      tmp := dbedit4.Text;
      delete(tmp,1,8);
      Query1.Close;
      Query1.SQL.Clear;
      Query1.SQL.Add('select max(Id) as nmr from student');
      Query1.Open;
      if Query1.RecordCount > 0 then
        begin
          int := Query1.FieldByName('nmr').AsInteger+1;
          tmp1 := IntToStr(int);
          for int1 := 0 to 3-Length(tmp1) do insert('0',tmp1,1);
          tmp := tmp+tmp1;
          Table1StId.AsString := tmp;
        end;
    end;
 end;

 procedure Tstudentform.anamenubuttonClick(Sender: TObject);
 begin
  mainform.Panel1.Visible := true;
  Close;
 end;

 procedure Tstudentform.FormClose(Sender: TObject;
  var Action: TCloseAction);
 begin
  Action := caFree;
  studentform := nil;
 end;

 procedure Tstudentform.Table1AfterInsert(DataSet: TDataSet);
 begin
  DBEdit4.Text := DateToStr(Date());
 end;

 procedure Tstudentform.imagebtClick(Sender: TObject);
 begin
  if photodlg.Execute then    begin
    if Table1.State <> dsInsert then Table1.Edit;
    DbImage1.Picture.Bitmap.LoadFromFile(photodlg.FileName);   end;end;
```

```
procedure Tstudentform.DBLookupComboBox1CloseUp(Sender: TObject);
begin
  DBEdit22.Text := Table2Fname.AsString;
end;

procedure Tstudentform.detailsbtClick(Sender: TObject);
begin
  if detailspanel.Visible  then
    begin detailspanel.Visible := false; detailsbt.Caption := 'Show Details'; end
  else begin detailspanel.Visible := true; detailsbt.Caption := 'Hide Details'; end;
end;

procedure Tstudentform.SpeedButton1Click(Sender: TObject);
begin
  if not Assigned (addpaymentform) then addpaymentform :=
Taddpaymentform.Create(Application);
  Table3.Insert;
  addpaymentform.DBEdit8.Text := DBText1.Caption;
  addpaymentform.Table2.Open;
  addpaymentform.ShowModal;
end;

procedure Tstudentform.Table3BeforePost(DataSet: TDataSet);
begin
//Table3Sid.AsFloat := Table1Stid.AsFloat;
end;

procedure Tstudentform.SpeedButton2Click(Sender: TObject);
begin
  if Table3.RecordCount > 0 then
    begin
      if Application.MessageBox('Record is deleting !. Are you sure ?','Attention
!!',mb_YESNO) = IDYES then
        begin
          Query1.Close;
          Query1.SQL.Clear;
          Query1.SQL.Add('delete from pdetail where Payid = '+Table3Id.AsString);
          Query1.ExecSQL;
          Table4.Close; Table4.Open;
          Table3.Delete;
        end;
    end;
end;
```

```
procedure Tstudentform.SpeedButton3Click(Sender: TObject);
var
mny,pmny : double;
tax,ksy : real;
pday,today : TDate;
begin
 if Table4.Active then
  begin
   if Table4Paid.AsString = 'N' then
    begin
     mny  := Table4Pamount.AsFloat;
     tax  := Table4Tax.AsFloat;
     pday := Table4Pdate.AsDateTime;
     today := Date();
     if today > pday then
      begin
       hesapla;
       ksy := (mny*tax)/100;
       pmny := mny+(ksy*howmanytax);
      end
     else
      begin
       howmanytax := 0;
       pmny := mny;
      end;
     if not Assigned (payinsform) then payinsform :=
Tpayinsform.Create(Application);
     Table4.Edit;
     payinsform.Edit1.Text     := Table4Id.AsString;
     payinsform.Edit2.Text     := Table4Pdate.AsString;
     payinsform.DBText1.Caption := DateToStr(Date());
     payinsform.Edit3.Text     := Table4Pamount.AsString;
     payinsform.Edit4.Text     := Table4Tax.AsString;
     payinsform.DBText2.Caption := IntToStr(howmanytax);
     payinsform.DBText3.Caption := FloatToStr(pmny);
     payinsform.DBEdit1.Text   := DateToStr(Date());
     payinsform.DBEdit2.Text   := IntToStr(howmanytax);
     payinsform.DBEdit3.Text   := FloatToStr(pmny);
     payinsform.DBEdit4.Text   := 'Y';
     payinsform.ShowModal;
    end;
  end;
end;
procedure Tstudentform.SpeedButton4Click(Sender: TObject);
begin
 if Table4.RecordCount > 0 then
  begin
   if Table4Paid.AsString = 'Y' then
```

```
  begin
      if Application.MessageBox('This Instalment is canceling !. Are you sure
?','Attention !!',mb_YESNO) = IDYES then
        begin
          Table4.Edit;
          Table4Ppdate.AsDateTime := Table4Pdate.AsDateTime;
          Table4Pdelay.AsInteger  := 0;
          Table4Ppamount.AsFloat  := 0;
          Table4Paid.AsString     := 'N';
          Table4Ptype.AsString    := '';
          Table4.Post;
        end;
     end;
   end;
end;

procedure Tstudentform.SpeedButton5Click(Sender: TObject);
begin
 if not Assigned (adddisactform) then adddisactform :=
Tadddisactform.Create(Application);
 adddisactform.Table2.Open;
 Table5.Insert;
 adddisactform.DBEdit8.Text := Table1stid.AsString;
 adddisactform.ShowModal;
end;

procedure Tstudentform.SpeedButton6Click(Sender: TObject);
begin
 if Table5.RecordCount > 0 then
   begin
    if Application.MessageBox('Record is deleting !. Are you sure ?','Attention
!!',mb_YESNO) = IDYES then
      begin
        Table5.Delete;
      end;
   end;
end;
procedure Tstudentform.Table1BeforeInsert(DataSet: TDataSet);
begin
 if main.groupcode = 0 then
   begin
    ShowMessage('You can not record a student. Because you aren''t an authorized !');
    Abort;
   end;
end;
procedure Tstudentform.SpeedButton7Click(Sender: TObject);
begin
 if not Assigned (addtermform) then addtermform :=
Taddtermform.Create(Application);
 addtermform.Table2.Open;
```

```
 Table6.Insert;
 addtermform.DBEdit8.Text := Table1Stid.AsString;
 addtermform.ShowModal;
end;
procedure Tstudentform.SpeedButton9Click(Sender: TObject);
begin
 if not Assigned(addcourseform) then addcourseform :=
Taddcourseform.Create(Application);
 Table7.Insert;
 addcourseform.DBEdit8.Text := Table1Stid.AsString;
 addcourseform.DBEdit4.Text := Table6Id.AsString;
 addcourseform.ShowModal;
end;
procedure Tstudentform.SpeedButton8Click(Sender: TObject);
begin
if Table6.RecordCount > 0 then
   begin
    if Application.MessageBox('Record is deleting !. Are you sure ?','Attention
!!',mb_YESNO) = IDYES then
      begin
       Table6.Delete;
      end;
end;
 end;
procedure Tstudentform.SpeedButton10Click(Sender: TObject);
begin
  if Table7.RecordCount > 0 then
   begin
    if Application.MessageBox('Record is deleting !. Are you sure ?','Attention
!!',mb_YESNO) = IDYES then
      begin
       Table7.Delete;
      end;
      end;
      end;
procedure Tstudentform.SpeedButton11Click(Sender: TObject);
begin
 if not Assigned (addgradeform) then addgradeform :=
Taddgradeform.Create(Application);
 addgradeform.Table1.Open;
 addgradeform.ShowModal;
end;
end.
```