# NEAR EAST UNIVERSITY

## Faculty of Engineering

## Department of Computer Engineering

# SCHOOL INFORMATION SYSTEM

Graduation Project
COM-400

Student: ilkkan Alpar Serimer

Supervisor: Elbrus Imanov

Nicosia - 2008

## ACKNOWLEDGEMENTS

" First, i would like to thank my supervisor Assist. Professor Dr Elbrus Imanov for his invaluable advice and blief in my work and myself over the course of this Gradiation Project...

Second, I would like to Express my gratitude to Near East Universty fort he scholarship that made work possible.

Third,I thank my family for their constant encouragement and support during the preparation of this project.

Fourth,I would like to thank NEU Computer Engineering Department academicians for their invaluable advice and support.

Finnally, I would also like to thank my friends Ensar Bereket,Mehmet Akif Gursoy and Ayçim Atik for their advice and support"

## ABSTRACT

Automation and management software is generally any software program that help a easy manage to school and easy to gather information about student,teacher,departmens meaning all of university. The term covers a large varation of uses within school enviroment,and categorized by using msall,medium and large databases.

This program write for only one target easy but usefull program for university. All part of university like this program because what they want, it is in the program.

In program you can add new faculties with its students, teachers and courses and you can manage them and its relations each other in this program have five aplication student,teacher,department,courese and course registration applications.

The main problem of school management program is tracking data on time and store datas safely and quickly . In my Project there will be easy to reach and easy to store of all datas with high security.

The aim of Project is to develop a school information system that the user can add and update student, add and update teacher , add and update course , add and update departments and register student to course ,add or drop course and grade course and more.

School information simply enables school to manage and control their personels and courses.

## TABLE OF CONTENTS

# INTRODUCTION

The purpose of the project is to design user friendly school information software product. This school information software is designed to help school improve while reducing time for information.since it is time saving and need minimal effort to run and follow the school with the use of such kind of software, i preferred to develop this fort he user who would like utilize or make use of this product which server simplicity. For the successful execution of a project effective planning is essential

While planning what necessary things and features should be in the content of the software and what functions should be fulfilled in the software, more and more functions seemed to be considered to design a proper and well working information system software product. In order to develop a product which can serve its users, all the details should be considerd in depth. In this software, all neccessary things wew thought in detail and the software, all necessary things were thought in detail and the software which can also be suited to all universities was designed

The aim of this project is to develop a simple database management system for university. The project consists of introduction,three chapters and conculusion.

Chapter one describes general terms of delphi programing and specific details about delphi components and coding structures.

Chapter two describes the main lines of borland delphi databases and controls. İt includes the borland database engine descriptions. And information about used database Microsoft Access 2007. And this chapter has information about programs database tables and relations of each other

# CHAPTER ONE

# DELPHI PROGRAMING LANGUAGE

## 1.1 Introduction

Object (or Delphi) Pascal, a set of object-oriented extensions to standard Pascal, is the language of Delphi. Delphi Pascal is a high-level, compiled, strongly typed language that supports structured and object-oriented design. Its benefits include easy-to-read code, quick compilation, and the use of multiple unit files for modular programming.

Borland Delphi is a sophisticated Windows programming environment, suitable for beginners and professional programmers alike. Using Delphi you can easily create self-contained, user friendly, highly efficient Windows applications in a very short time - with a minimum of manual coding.

Delphi provides all the tools you need to develop, test and deploy Windows applications, including a large number of so-called reusable components.Borland Delphi, in it's latest version, provides a cross platform solution when used with Borland Kylix - Borland's RAD tool for the Linux platform. Borland Delphi (1/2/3/4/5) is a development tool for Microsoft Windows applications. Delphi is powerful and easy to use tool for generating stand-alone graphical user interface (GUI) programs or 32-bit console applications (programs that have no GUI presence but instead run in what is commonly referred to as a "DOS box.") when paired with Borland Kylix, Delphi users can build single-source applications for both Windows and Linux, which opens new opportunities and increases the potential return on development investments. Use the Cross-platform CLX component library and visual designers to build high-performance portable applications for Windows that can be easily re-compiled on Linux.

Delphi is the first programming language to shatter the barrier between high-level, easy-to-use rapid application development environments and low-level bits-and-bytes power tools. Delphi ships in a variety of configurations aimed at both departmental and enterprise needs. With Delphi, you can write Windows programs more quickly and more easily than was possible ever before

Delphi can access many types of databases. Using forms and reports that you create, the BDE (Borland Database Engine) can access local databases, like Paradox and DBase, network SQL server databases, like InterBase, and SysBase, and any data source accessible though ODBC (open database connectivity).

**1.2 what is delphi**

Borland Delphi is a high-level, compiled, strongly typed language that supports structured and object-oriented design. Delphi language is based on Object Pascal. Today, Delphi is much more than simply "Object Pascal language".

Borland Delphi is the first rapid application development environment for Windows that fully supports new and emerging Web Services. With Delphi, corporate or individual developers can create next-generation e-business applications quickly and easily.

**1.3 History of delphi**

This chapter gives information about history of delphi borland delphi

***1.3.1 Pascal And Delphi's History***

The origin of Pascal owes much of its design to Algol - the first high-level language with a readable, structured, and systematically defined syntax.

In the late sixties (196X), several proposals for an evolutionary successor to Algol were developed. The most successful one was Pascal, defined by Prof. Niklaus Wirth. Wirth published the original definition of Pascal in 1971. It was implemented in 1973 with some modifications. Many of the features of Pascal came from earlier languages. The case statement, and value-result parameter passing came from Algol, and the records structures were similar to Cobol and PL 1. Besides cleaning up or leaving out some of Algol's more obscure features, Pascal added the capability to define new data types out of simpler existing ones. Pascal also supported dynamic data structures; i.e., data structures which can grow and shrink while a program is running. The language was designed to be a teaching tool for students of programming classes.

In 1975, Wirth and Jensen produced the ultimate Pascal reference book "Pascal User Manual and Report". Wirth stopped its work on Pascal in 1977 to create a new language, Modula - the successor to Pascal.

With the release (November 1983) of Turbo Pascal 1.0, Borland started its journey into the world of development environments and tools. To create Turbo Pascal 1.0 Borland licensed the fast and inexpensive Pascal compiler core, written by Anders Hejlsberg. Turbo Pascal introduced an Integrated Development Environment (IDE) where you could edit the code, run the compiler, see the errors, and jump back to the lines containing those errors. Turbo Pascal compiler has been one of the best-selling series of compilers of all time, and made the language particularly popular on the PC platform.

In 1995 Borland revived its version of Pascal when it introduced the rapid application development environment named Delphi - turning Pascal into a visual programming language. The strategic decision was to make database tools and connectivity a central part of the new Pascal product.

After the release of Turbo Pascal 1, Anders joined the company as an employee and was the architect for all versions of the Turbo Pascal compiler and the first three versions of Delphi. As a chief architect at Borland, Hejlsberg secretly turned Turbo Pascal into an object-oriented application development language, complete with a truly visual environment and superb database-access features: Delphi.

### *1.3.2 Delphi Versions*

#### 1.3.2.1 Delphi 1 (Codename :" Delphi" - Relased 1995)

Delphi, Borland's powerful Windows programming development tool first appeared in 1995. Delphi 1 extended the Borland Pascal language by providing object-orientated and form-based approach, extremely fast native code compiler, visual two-way tools and great database support, close integration with Windows and the component technology.

Here's the Visual Component Library First Draft

Slogan of Delphi 1 : Delphi and Delphi Client/Server are the only development tools that provide the Rapid Application Development (RAD) benefits of visual component-based design, the power of an optimizing native code compiler and a scalable client/server solution.

Here's what were the "7 Top Reasons to Buy Borland Delphi 1.0 Client/Server"

#### 1.3.2.2 Delphi 2 (Codename : "Polaris" – Relased 1996)

Delphi 2 is the only Rapid Application Development tool that combines the performance of the world's fastest optimizing 32-bit native-code compiler, the productivity of visual component-based design, and the flexibility of scalable database architecture in a robust object-oriented environment.

Delphi 2, beside being developed for the Win32 platform (full Windows 95 support and integration), brought improved database grid, OLE automation and variant data type support, the long string data type and Visual Form Inheritance. Delphi 2: "the Ease of VB with the Power of C++"

#### 1.3.2.3 Delphi 3(Codename : "Ivory" – Relased 1997)

The most comprehensive set of visual, high-performance, client and server development tools for creating distributed enterprise and Web-enabled applications.

Delphi 3 introduced new features and enhancements in the following areas: the code insight technology, DLL debugging, component templates, the DecisionCube and TeeChart components, the WebBroker technology, ActiveForms, component packages, and integration with COM through interfaces.

#### 1.3.2.4 Delphi 4 (Codename: "Allegro" –Relased 1998)

Delphi 4 is a comprehensive set of professional and client/server development tools for building high productivity solutions for distributed computing. Delphi provides Java interoperability, high performance database drivers, CORBA development, and Microsoft BackOffice support. You've never had a more productive way to customize, manage, visualize and update data. With Delphi, you deliver robust applications to production, on time and on budget.

Delphi 4 introduced docking, anchoring and constraining components. New features included the AppBrowser, dynamic arrays, method overloading, Windows 98 support, improved OLE and COM support as well as extended database support.

#### 1.3.2.5 Delphi 5 (Codename: "Argus" – Relased 1999)

High-productivity development for the Internet

Delphi 5 introduced many new features and enhancements. Some, among many others, are: various desktop layouts, the concept of frames, parallel development, translation capabilities, enhanced integrated debugger, new Internet capabilities (XML), more database power (ADO support), etc.

Then, in 2000, Delphi 6 was the first tool to fully supports new and emerging Web Services

#### 1.3.2.6 Delphi 6 (Codename: "Iliad" – Relased 2000)

Borland Delphi is the first rapid application development environment for Windows that fully supports new and emerging Web Services. With Delphi, corporate or individual developers can create next-generation e-business applications quickly and easily.

Delphi 6 introduced new features and enhancements in the following areas: IDE, Internet, XML, Compiler, COM/Active X, Database support… What's more, Delphi 6 added the support for cross-platform development – thus enabling the same code to be compiled with Delphi (under Windows) and Kylix (under Linux).

More enhancements included: support for Web Services, the DBExpress engine, new components and classes…

#### 1.3.2.7 Delphi 7 (Codename: " Aurora" – Relased 2001)

Borland Delphi 7 Studio provides the migration path to Microsoft .NET that developers have been waiting for. With Delphi, the choices are always yours: you're in control of a complete e-business development studio — with the freedom to easily take your solutions cross-platform to Linux.

#### 1.3.2.8 Delphi 8 (Codename: "Octane" – Relased 2003)

For the 8th anniversary of Delphi, Borland prepared the most significant Delphi release: Delphi 8 continues to provide Visual Component Library (VCL) and Component Library for Cross-platform (CLX) development for Win32 (and Linux) as well as new features and continued framework, compiler, IDE, and design time enhancements.

## 1.4 Some Extra Information About Delphi

### 1.4.1 Why The Name Delphi ?

As explained in the Borland's Museum article, project codenamed Delphi hatched in mid 1993. Why Delphi? It was simple: "If you want to talk to [the] Oracle, go to Delphi". When it came time to pick a retail product name, after an article in Windows Tech Journal about a product that will change the life of programmers, the proposed (final) name was AppBuilder. Since Novell released its Visual AppBuilder, the guys at Borland needed to pick another name; it became a bit of a comedy: the harder people tried to dismiss "Delphi" for the product name, the more it gained support.

### 1.4.2 Advantage Of The Delphi

- Suitable for Rapid Application Development (RAD)
- Based on a well-designed language, high-level and strongly typed, but able to use low-level code for hardware access and performance
- A large community on Usenet and the web
- Can compile to a single executable, simplifying distribution and eliminating DLL version issues
- Many VCL (Visual Component Library) and third-party components (usually available with full source code) and tools (documentation, debug tools, etc.)
- Quick optimizing compiler also able to use assembler code
- Multiple platform native code from the same source code
- High level of source compatibility between versions
- CrossKylix - a now discontinued third-party toolkit which allows native Kylix/Linux applications to be compiled from the Windows Delphi IDE, enabling dual-platform development and deployment
- CrossFPC - a sister project to CrossKylix, not released and now inactive, which enables Windows Delphi applications to be cross-compiled from the Delphi IDE for platforms supported by the Free Pascal compiler.
- Class helpers to bridge functionality available natively in the Delphi RTL

- The language's object orientation features only class- and interface-based polymorphism
- Delphi 2005, Delphi 2006 and Delphi 2007 all support advanced refactoring features such as Method Extraction, etc. [1]
- Metaclasses are first class objects
- There are dedicated string types (as well as null-terminated strings). Strings can be added by using the '+' sign, rather than using functions.
- Objects are actually references to the objects (like in Java), which Delphi implicitly dereferences
- Delphi is strongly type-based.
- Delphi's compiler is extremely efficient and fast.

### 1.4.2 Disadvantages Of The Delphi

- Produces code for machines running Microsoft Windows only. Kylix, which allowed Delphi code to be ported to Linux relatively easily, discontinued. CrossKylix discontinued and CrossFPC not released and inactive.
- A reluctance to break any code has led to some convoluted language design choices, and orthogonality and predictability have suffered.

## 1.5 Delphi Programing Pheriphals

### 1.5.1 Using Project Files

Since it is quite common for Delphi applications to share code or previously customized forms, Delphi organizes applications into what is called projects. A project is made up of the visual interface along with the code that activates the interface. Each project can have multiple forms, allowing us to build applications that have multiple windows. The code that is needed for a form in our project is stored in a separate Unit file that Delphi automatically associates to the form. General code that we want to be shared by all the forms in our application is placed in unit files as well. Simply put, a Delphi project is a collection of files that make up an application. What this means is that each project is made of one or more Form files (files with the *.dfm* extension) and one or more Unit files (*.pas* extension). We can also add resource files, and they are compiled into .RES files and linked when we compile the project.

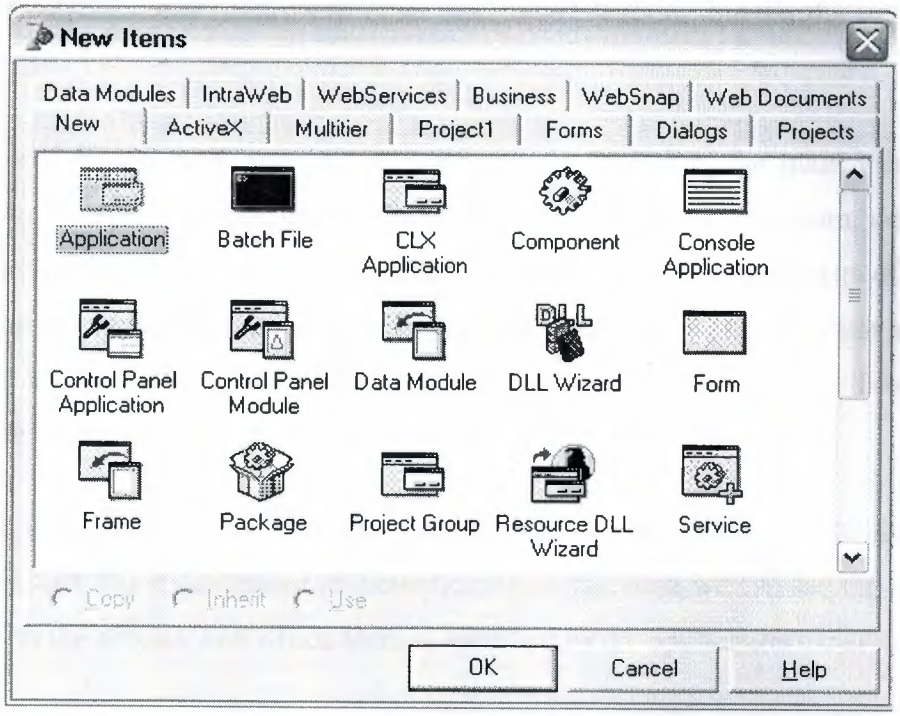



Figure 1.5.1.1 New Items View

Each project is made up of a single project file (.dpr). Project files contain directions for building an application. This is normally a set of simple routines which open the main form and any other forms that are set to be opened automatically and then starts the program by calling the *Initialize*, *CreateForm* and *Run* methods of the global Application object (which is actually a form of zero width and height, so it never actually appears on the screen).

### 1.5.2 Project Unit

A program is constructed from source-code modules called units. Each unit is stored in its own file and compiled separately; compiled units (.DCU files) are linked to create an application. Units allow you to:

- divide large programs into modules that can be edited separately.
- create libraries that you can share among programs.

- distribute libraries to other developers without making the source code available.

In traditional Pascal programming, all source code, including the main program, is stored in .PAS files. Delphi uses a project (.DPR) file to store the "main" program, while most other source code resides in unit (.PAS) files. Each application-or project-consists of a single project file and one or more unit files. (Strictly speaking, you needn't explicitly use any units in a project, but all programs automatically use the System unit.) To build a project, the compiler needs either a source file or a previously compiled DCU for each unit.

Although you can look and edit the Project File, in most cases, you'll let Delphi maintain the DPR file. The main reason to view the project file is so we can see the units and forms that make up the project, and which form is specified as the application's main form.

Another reason to work with the project file is when we are creating a DLL rather than a stand-alone application or need some start-up code, such as a splash screen before the main form is created by Delphi.

Here is the default project file for a new application (containing one form: "Form1"):

```
Program Project1;

  Uses

   Forms,

   Unit1          in          'Unit1.pas'          {Form1};
{SR                                                 *.RES}
begin
 Application.Initialize;
 Application.CreateForm (TForm1, Form1) ;

   Application.Run; end.
```

The **program** [link url=/od/delphiprogrammingglossary/g/reservedword.htm]keyword identifies this unit as a program's main source unit. You can see that the unit name, Project1, follows the program keyword (Delphi gives the project a default name until you save the project with a more meaningful name). When we run a project file from the IDE, Delphi uses the name of the Project file for the name of the EXE file that it creates.

Delphi reads the **uses** clause of the project file to determine which units are part of a project.

The .dpr file is linked with the .pas file with the compile directive *{$R *.RES}* (in this case '*' represents the root of the .pas filename rather than "any file"). This compiler directive tells Delphi to include this project's resource file. The project's resource file contains such items as the project's icon image.

The **begin..end** block is the main source-code block for the project.

Although **Initialize** is the first method called in the main project source code, it is not the first code that is executed in an application. The application first executes the **"initialization" section of all the units** used by the application.

The **Application.CreateForm** statement loads the form specified in its argument. Delphi adds an Application.CreateForm statement to the project file for each form you add to the project. This code's job is to first allocate memory for the form. The statements are listed in the order the forms are added to the project. This is the order that the forms will be created in memory at runtime. If you want to change this order, do not edit the project source code. Use the Project|Options menu command.

The **Application.Run** statement starts your application. This instruction tells the predeclared object called Application to begin processing the events that occur during the run of a program.

Note: To add an additional form to Delphi Project, we select file → New Form.

There are, of course, other ways to add a "new" form to a Delphi Project.

### 1.5.3 Data Types And Variables

A type is essentially a name for a kind of data. When we declare a variable we must specify its type, which determines the set of values the variable can hold and the operations that can be performed on it. Every expression returns data of a particular type, as does every function. Most functions and procedures require parameters of specific types.

The Delphi language is a 'strongly typed' language, which means that it distinguishes a variety of data types and

does not always allow you to substitute one type for another. This is usually beneficial because it lets the compiler

treat data intelligently and validate your code more thoroughly, preventing hard-to-diagnose runtime errors. When

you need greater flexibility, however, there are mechanisms to circumvent strong typing. These include typecasting,

pointers, Variants, Variant parts in records, and absolute addressing of variables.

There are several ways to categorize Delphi data types:

- Some types are predefined (or built-in); the compiler recognizes these automatically, without the need for a declaration. Almost all of the types documented in this language reference are predefined. Other types are created by declaration; these include user-defined types and the types defined in the product libraries.

- Types can be classified as either fundamental or generic. The range and format of a fundamental type is the same in all implementations of the Delphi language, regardless of the underlying CPU and operating system. The range and format of a generic type is platform-specific and could vary across different implementations. Most predefined types are fundamental, but a handful of integer, character, string, and pointer types are generic. It's a good idea to use generic types when possible, since they provide optimal performance and portability. However, changes in storage format from one implementation of a generic type to the next could cause compatibility problems - for example, if you are streaming content to a file as raw, binary data, without type and versioning information.

- Types can be classified as simple, string, structured, pointer, procedural, or variant. In addition, type dentifiers themselves can be regarded as belonging to a special 'type' because they can be passed as parameters to certain functions (such as High, Low, and SizeOf).

The outline below shows the taxonomy of Delphi data types.

- simple
- ordinal
- integer
- character
- Boolean
- enumerated
- subrange
- real
- string
- structured
- set
- array
- record
- file
- class
- class reference
- interface
- pointer
- procedural
- Variant
- type identifier

The standard function SizeOf operates on all variables and type identifiers. It returns an integer representing the amount of memory (in bytes) required to store data of the specified type. For example, SizeOf(Longint) returns 4, since a Longint variable uses four bytes of memory.

Type declarations are illustrated in the topics that follow. For general information about type declarations, see Declaring types.

### 1.5.4 Procedures And Functions

Procedures and functions, referred to collectively as *routines*, are self-contained statement blocks that can be called from different locations in a program. A function is a routine that returns a value when it executes. A procedure is a routine that does not return a value.

Function calls, because they return a value, can be used as expressions in assignments and operations. For example,

```
I := SomeFunction(X);
```

calls SomeFunction and assigns the result to I. Function calls cannot appear on the left side of an assignment statement.

Procedure calls - and, when extended syntax is enabled ({$X+}), function calls - can be used as complete statements. For example,

```
DoSomething;
```

calls the DoSomething routine; if DoSomething is a function, its return value is discarded.

Procedures and functions can call themselves recursively.

When you declare a procedure or function, you specify its name, the number and type of parameters it takes, and, in the case of a function, the type of its return value; this part of

The declaration is sometimes called the prototype, heading, or header. Then you write a block of code that executes whenever the procedure or function is called; this part is sometimes called the routine's body or block.

A procedure declaration :

```
procedure procedureName(parameterList); directives;
        localDeclarations;
                begin
                statements
                end;
```

where *procedureName* is any valid identifier, *statements* is a sequence of statements that execute when the procedure is called, and *(parameterList)*, *directives;*, and *localDeclarations;* are optional.

Here is an example of a procedure declaration:

```
procedure NumString(N: Integer; var S: string);
var
        V: Integer;
begin
 V := Abs(N);
          S := '';
repeat
        S := Chr(V mod 10 + Ord('0')) + S;
        V := V div 10;
        until V = 0;
        if N < 0 then S := '-' + S;
        end;
```

Given this declaration, you can call the NumString procedure like this:

```
NumString(17, MyString);
```

This procedure call assigns the value '17' to MyString (which must be a string variable).

Within a procedure's statement block, you can use variables and other identifiers declared in the *localDeclarations* part of the procedure. You can also use the parameter names from the parameter list (like N and S in the previous example); the parameter list defines a set of local variables, so don't try to redeclare the parameter names in the *localDeclarations* section. Finally, you can use any identifiers within whose scope the procedure declaration falls.

Most procedure and function headers include a parameter list. For example, in the header

function Power(X: Real; Y: Integer): Real;

the parameter list is (X: Real; Y: Integer).

A parameter list is a sequence of parameter declarations separated by semicolons and enclosed in parentheses. Each declaration is a comma-delimited series of parameter names, followed in most cases by a colon and a type identifier, and in some cases by the = symbol and a default value. Parameter names must be valid identifiers. Any declaration can be preceded by var, const, or out. Examples:

```
(X, Y: Real)
(var S: string; X: Integer)
(HWnd: Integer; Text, Caption: PChar; Flags: Integer)
(const P; I: Integer)
```

The parameter list specifies the number, order, and type of parameters that must be passed to the routine when it is called. If a routine does not take any parameters, omit the identifier list and the parentheses in its declaration:

```
procedure UpdateRecords;
begin
...
end;
```

Within the procedure or function body, the parameter names (X and Y in the first example) can be used as local variables. Do not redeclare the parameter names in the local declarations section of the procedure or function body.

Calling Procedures:

When you call a procedure or function, program control passes from the point where the call is made to the body of the routine. You can make the call using the routine's declared name (with or without qualifiers) or using a procedural variable that points to the routine. In either case, if the routine is declared with parameters, your call to it must pass parameters that correspond in order and type to the routine's parameter list. The parameters you pass to a routine are called actual parameters, while the parameters in the routine's declaration are called formal parameters.

When calling a routine, remember that

- expressions used to pass typed const and value parameters must be assignment-compatible with the corresponding formal parameters.
- expressions used to pass var and out parameters must be identically typed with the corresponding formal parameters, unless the formal parameters are untyped.
- only assignable expressions can be used to pass var and out parameters.
- if a routine's formal parameters are untyped, numerals and true constants with numeric values cannot be used as actual parameters.

When you call a routine that uses default parameter values, all actual parameters following the first accepted default must also use the default values; calls of the form SomeFunction(,,X) are not legal.

You can omit parentheses when passing all and only the default parameters to a routine. For example, given the procedure.

procedure DoSomething(X: Real = 1.0; I: Integer = 0; S: string = '');

the following calls are equivalent.

DoSomething();

DoSomething;

### 1.5.5 Classes And Object

A class, or class type, defines a structure consisting of fields, methods, and properties. Instances of a class type are called objects. The fields, methods, and properties of a class are called its components or members.

- A field is essentially a variable that is part of an object. Like the fields of a record, a class' fields represent data items that exist in each instance of the class.

- A method is a procedure or function associated with a class. Most methods operate on objectsthat is, instances of a class. Some methods (called class methods) operate on class types themselves.

- A property is an interface to data associated with an object (often stored in a field). Properties have Access specifiers, which determine how their data is read and modified. From other parts of a programoutside of the object itselfa property appears in most respects like a field.

Objects are dynamically allocated blocks of memory whose structure is determined by their class type. Each object has a unique copy of every field defined in the class, but all instances of a class share the same methods. Objects are created and destroyed by special methods called constructors and destructors.

A variable of a class type is actually a pointer that references an object. Hence more than one variable can refer to the same object. Like other pointers, class-type variables can hold the value nil. But you don't have to explicitly dereference a class-type variable to access the object it points to. For example, SomeObject.Size := 100 assigns the value 100 to the Size

property of the object referenced by SomeObject; you would not write this as SomeObject^.Size := 100.

**Class Types**

A class type must be declared and given a name before it can be instantiated. (You cannot define a class type within a variable declaration.) Declare classes only in the outermost scope of a program or unit, not in a procedure or function declaration.

A class type declaration has the form

```
type
    className = class (ancestorClass)
    memberList
end;
```

where *className* is any valid identifier, *(ancestorClass)* is optional, and *memberList* declares members - that is, fields, methods, and properties - of the class. If you omit *(ancestorClass)*, then the new class inherits directly from the predefined TObject class. If you include *(ancestorClass)* and *memberList* is empty, you can omit end. A class type declaration can also include a list of interfaces implemented by the class; see Implementing Interfaces.

Methods appear in a class declaration as function or procedure headings, with no body. Defining declarations for each method occur elsewhere in the program.

For example, here is the declaration of the TMemoryStream class from the Classes *unit*.

```
type TMemoryStream = class(TCustomMemoryStream)
 private
     FCapacity: Longint;
     procedure SetCapacity(NewCapacity: Longint);
 protected
     function Realloc(var NewCapacity: Longint): Pointer; virtual;
     property Capacity: Longint read FCapacity write SetCapacity;
```

```
public
    destructor Destroy; override;
    procedure Clear;
    procedure LoadFromStream(Stream: TStream);
    procedure LoadFromFile(const FileName: string);
    procedure SetSize(NewSize: Longint); override;
    function Write(const Buffer; Count: Longint): Longint; override;
end;
```

TMemoryStream descends from TCustomMemoryStream (in the Classes unit), inheriting most of its members. But it defines - or redefines - several methods and properties, including its destructor method, Destroy. Its constructor, Create, is inherited without change from TObject, and so is not redeclared. Each member is declared as private, protected, or public (this class has no published members)

Given this declaration, you can create an instance of TMemoryStream as follows:

```
var stream: TMemoryStream;
stream := TMemoryStream.Create;
```

**Fields**

A field is like a variable that belongs to an object. Fields can be of any type, including class types. (That is, fields can hold object references.) Fields are usually private

To define a field member of a class, simply declare the field as you would a variable. All field declarations must ocur before any property or method declarations. For example, the following declaration creates a class called TNumber whose only member, other than the methods is inherits from TObject, is an integer field called Int.

```
type TNumber = class
```

```
    Int: Integer;
  end;
```

Fields are statically bound; that is, references to them are fixed at compile time. To see what this means, consider the following code.

```
type
    TAncestor = class
      Value: Integer;
    end;
    TDescendant = class(TAncestor)
       Value: string; // hides the inherited Value field
    end;
var
    MyObject: TAncestor;
begin
    MyObject := TDescendant.Create;
    MyObject.Value := 'Hello!' // error
    (MyObject as TDescendant).Value := 'Hello!' // works!
end;
```

Although MyObject holds an instance of TDescendant, it is declared as TAncestor. The compiler therefore interprets MyObject.Value as referring to the (integer) field declared in TAncestor. Both fields, however, exist in the TDescendant object; the inherited Value is hidden by the new one, and can be accessed through a typecast

### 1.5.6 Libraries And Packages

A dynamically loadable library is a dynamic-link library (DLL) on Win32, and an assembly (also a DLL) on the .NET platform. It is a collection of routines that can be called by applications and by other DLLs or shared objects. Like units, dynamically loadable libraries contain sharable code or resources. But this type of library is a separately compiled executable that is linked at runtime to the programs that use it.

**Calling Dynammically Loadable Libaries**

You can call operating system routines directly, but they are not linked to your application until runtime. This means that the library need not be present when you compile your program. It also means that there is no compile-time validation of attempts to import a routine.

Before you can call routines defined in DLL or assembly, you must import them. This can be done in two ways: by declaring an external procedure or function, or by direct calls to the operating system. Whichever method you use, the routines are not linked to your application until runtime.

The Delphi language does not support importing of variables from DLLs or assemblies.

**Static Loading**

The simplest way to import a procedure or function is to declare it using the external directive. For example,

procedure DoSomething; external 'MYLIB.DLL';

If you include this declaration in a program, MYLIB.DLL is loaded once, when the program starts. Throughout execution of the program, the identifier DoSomething always refers to the same entry point in the same shared library.

Declarations of imported routines can be placed directly in the program or unit where they are called. o simplify maintenance, however, you can collect external declarations into a separate "import unit" that also contains any constants and types required for interfacing with the library. Other modules that use the import unit can call any routines declared in it.

# CHAPTER TWO

# DATABASE

## 2.1 Borland Database Engine

### 2.1.1 What is BDE

**Borland Database Engine** (*BDE*) is 32-bit Windows-based core database engine and connectivity software behind Borland Delphi, C++Builder, IntraBuilder, Paradox for Windows, and Visual dBASE for Windows.

### 2.1.2 History Of BDE

Borland's Turbo Pascal included a "database" Toolbox, it was the beginning of the Borland compiler add-ons that facilitated database connectivity. Then came the Paradox Engine for Windows – PXENGWIN – which could be compiled into a program to facilitate connectivity to Paradox tables.

The first DLL-based connectivity engine was **ODAPI** (Open Database API). It represented Borland's attempt to centralise connectivity in its suite of applications which included the brand-new Paradox for Windows 4 and Quattro. With version 4.5 / 5.0 of Paradox for Windows, this database engine was crystallised as IDAPI.

In 2000, Borland introduced a new SQL driver architecture called dbExpress, which deprecated BDE SQL links technology.

### 2.1.3 BDE Design

The included set of database drivers enables consistent access to standard data sources: Paradox, dBASE, FoxPro, Access, and text databases. You can add Microsoft ODBC drivers

as needed to the built-in ODBC socket. Optionally, Borland's SQL Links product provides access to a range of SQL servers, including Informix, DB2, InterBase, Oracle, and Sybase.

BDE is object-oriented in design. At runtime, application developers interact with BDE by creating various BDE objects. These runtime objects are then used to manipulate database entities, such as tables and queries. BDE's application program interface (API) provides direct C and C++ optimized access to the database engine, as well as BDE's built-in drivers for dBASE, Paradox, FoxPro, Access, and text databases.

The core database engine files consist of a set of DLLs that are fully re-entrant and thread-safe. Included with BDE are a set of supplemental tools and examples with sample code.

BDE system is configured using the BDE Administrator (BDEADMIN.EXE).

Included with BDE is Borland's Local SQL, a subset of ANSI-92 SQL enhanced to support Paradox and dBASE (standard) naming conventions for tables and fields (called "columns" in SQL). Local SQL lets you use SQL to query "local" standard database tables that do not reside on a database server as well as "remote" SQL servers. Local SQL is also essential to make multi-table queries across both local standard tables and those on remote SQL servers.

The older name for the BDE API is the "Integrated Database Application Program Interface" or "IDAPI".

## 2.2 Microsoft Access (Microsoft Office Access)

### 2.2.1 General Information About Microsoft Access

**Microsoft Office Access**, previously known as **Microsoft Access**, is a relational database management system from Microsoft that combines the relational Microsoft Jet Database Engine with a graphical user interface and software development tools. It is a member of the 2007 Microsoft Office system.

Access can use data stored in Access/Jet, Microsoft SQL Server, Oracle, or any ODBC-compliant data container (including MySQL and PostgreSQL). Skilled software

developers and data architects use it to develop application software. Relatively unskilled programmers and non-programmer "power users" can use it to build simple applications. It supports some object-oriented techniques but falls short of being a fully object-oriented development tool.

Access was also the name of a communications program from Microsoft, meant to compete with ProComm and other programs. This proved a failure and was dropped.[1] Years later Microsoft reused the name for its database software.

### 2.2.2 History Of Acess

Access version 1.0 was released in November 1992, followed in May of 1993 by an Access 1.1 release to improve compatibility with other Microsoft products. Significantly, these early versions were priced at a $99 introductory price, designed to promote trial without requiring CIO-level sign-off; at the time, database management systems typically were priced at $795. Since that time, the following versions have been released: 2.0, 95, 97, 2000, 2002 (also called XP), 2003, and the latest, 2007, all at prices consistent with other Microsoft Office products.

Microsoft specified the minimum operating system for Version 2.0 as Microsoft Windows v3.0 with 4 MB of RAM. 6 MB RAM was recommended along with a minimum of 8 MB of available hard disk space (14 MB hard disk space recommended). The product was shipped on seven 1.44 MB diskettes. The manual shows a 1993 copyright date.

The software worked well with very large records sets but testing showed some circumstances caused data corruption. For example, file sizes over 700 MB were problematic (note that most hard disks were smaller than 700 MB at the time this was in wide use). The *Getting Started* manual warns about a number of circumstances where obsolete device drivers or incorrect configurations can cause data loss.

Access's initial codename was Cirrus; the forms engine was called Ruby. This was before Visual Basic - Bill Gates saw the prototypes and decided that the BASIC language component should be co-developed as a separate expandable application, a project called Thunder. The two projects were developed separately as the underlying forms engines were incompatible with each other; however, these were merged together again after VBA.

## 2.3 Database And Relations

### 2.3.1 Database

#### 2.3.1.1 *Tables*

**Course Table**

| CourseId | CourseCode | DepartmentId | CourseName | CourseContent | TeacherId | Credit | ProcessUser | ProcessDate | UpdateUser | UpdateDate |
|---|---|---|---|---|---|---|---|---|---|---|
| 39 | COM111 | CENG | INTRO. TO COMPUTE | | 24 | 3 | Elbrus | 03.06.2008 | | |
| 40 | com434 | CENG | 123123 | 13221 | 24 | 3 | Elbrus | 03.06.2008 | | |
| Yeni | | | | | 0 | 0 | | | | |

Figure 2.1 Course Table

Course table stores that values: Courseid, CourseCode, DepartmendId, CourseName, CourseContent, TeacherId, Credit, ProcessUser, ProcessDate, UpdateUser, UpdateDate

**Department Table**

| DepartmentId | DepartmentName | ProcessUser | ProcessDate | UpdateUser | UpdateDate |
|---|---|---|---|---|---|
| 6 | geee | Elbrus | 31.05.2008 | | |
| ACC | ACCOUNTING | | | | |
| ARC | ARCHITECTURE | Alpar | 31.05.2008 | | |
| BESYO | BEDEN EGITIM SPOR MESLEK | Alpar | 31.05.2008 | Alpar | 02.06.2008 |
| BF | BANKING AND FINANCE | | | | |
| BIO | BIOLOGY 2 | Alpar | 31.05.2008 | Alpar | 02.06.2008 |
| CENG | COMPUTER ENGINEERING | Alpar | 31.05.2008 | Alpar | 02.06.2008 |
| CHEM | CHEMISTRY | | | | |
| CIS | COMPUTER INFORMATION SYSTEMS | | | Alpar | 02.06.2008 |
| ECON | ECONOMY | | | | |
| IE | INDUSTRIAL ENGINNERING | Alpar | 31.05.2008 | | |
| MAN | BUSINESS ADMINISTRATION | | | | |
| MARK | MARKETING DEPARTMENT | | | | |
| MATH | MATHEMATICS | | | | |
| PHYS | PHY | | | | |

Figure 2.2 Department Table

Department table stores that values:DepartmentId, DepartmentName, ProcessUser, ProcessDate, UpdateUser, UpdateDate

**Login Table**

| LoginId | LoginName | PassWord | AccessLevel | Department |
|---|---|---|---|---|
| 20 | Admin | 1 | User | CENG |
| 23 | user | 1 | User | CENG |
| 24 | Elbrus | 1 | Admin | CENG |
| 25 | Alpar | 1984 | Admin | CENG |

Figure 2.3 Login table

Login Table stores that values: LoginId, LoginName, Password, AcessLevel, DepartmentId

**Student Table**

| StudentId | Department | name | surname | sex | Address | City | Country | postcode | Gsm | FatherName |
|---|---|---|---|---|---|---|---|---|---|---|
| 20081977 | MAN | ALPAR | SERIMER | Male | | | TR | | | |
| 20084544 | ACC | ASD | DASDAS | Male | | | FRE | | | |
| | | | | Male | | | | | | |

Figure 2.4 Student Table – 1

| MotherNam | BirthPlace | BirthDate | OsymNo | Email | RegisterDat | phone | isGraduatec | Graduation( | Graduation( | IsDismised | Dismisdat |
|---|---|---|---|---|---|---|---|---|---|---|---|
| | | 11.12.1988 | | | 31.05.2008 | | Hayir | | | ☐ | |
| | | 11.12.1988 | | | 31.05.2008 | | Hayir | | | ☐ | |
| | | | | | | | Hayir | | | ☐ | |

Figure 2.5 Student Table – 2

| Dismisdate | DismisComr | ProcessUser | ProcessDate | UpdateUser | UpdateDate |
|---|---|---|---|---|---|
| | | Elbrus | | | |
| | | Elbrus | | | |
| | | | | | |

Figure 2.6 Student Table – 3

Student Table stores that values : StudentId, DepartmentId, Name, Surname, Sex, Address, City, Country, Postcode, GSM, FatherName, MotherName, BİrthPlace, BirthDate, OgrNo, Email, RegisterDate, phone, isGraduated(choise), GradationDate, GradationComment, isDismised(choise), DismisDate, DismisComment,ProcessUser,ProcessDate,UpdateUser,UpdateDate

**Teacher Table**

| TeacherId | Tname | Tsurname | Address | Gsm | Sex | Phone | Email | DepartmentId | BirthDate | Title | ProcessUser | ProcessDate | UpdateUser | UpdateDate |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 24 | OKAN | DONANGİL | Memo1 | | Male | | | CENG | 12.12.1978 | Teacher | Elbrus | 02.06.2008 | | |
| Yeni | | | | | | | | | | | | | | |

Figure 2.7 Teacher Table

Teacher Table store that values : TeacherId, Tname, Tsurname, Adress, GSM, Phone,Email, DepartmentId, BirthDate, Title, ProcessUser, ProcessDate, UpdateUser, UpdateDate

**Transcript Table**

| TransId | CourseId | StudentId | CourseRegD | Grade | GradeDate | AddDate | DropDate | ProcessUser | ProcessDate | UpdateUser |
|---|---|---|---|---|---|---|---|---|---|---|
| 269 | 39 | 20081977 | 03.06.2008 | | | | | Elbrus | 03.06.2008 | |
| 270 | 40 | 20081977 | 03.06.2008 | | | | | Elbrus | 03.06.2008 | |
| Yeni | 0 | | | | | | | | | |

Figure 2.8 Transcript Table

Transcript Table store that Values : TransId, CourseId, StudentId, CourseRegDate, Grade, Gradedate, AddDate, DropDate,ProcessUser,ProcessDate,UpdateUser,UpdateDate

### *2.3.1.2 Values In The Table*

- CourseId : value shows the place of the data in the course table. Database Automatically input to this data.
- CourseCode : value shows Code of course (for example: COM301,COM333,MAT101 ).This Data will input by user manually to the database.
- DepartmentId : valu showsDepartment code of system (for example : CENG,ACC,BIO) . This Data will input by user manually to the database.
- CourseName : value shows Name of the course. This Data will input by user manually to the database.
- CourseContent : value shows Content of the course. This Data will input by user manually to the database
- TeacherId : value shows Teachers code of Identification like student number. This Data will input by user manually to the database.

- Credit : value shows the how many credit that the lecture have. This Data will input by user manually to the database.
- ProcessUser : value shows which user add this values. Database Automatically input to this data.
- ProcessDate: value shows when user add this values. Database Automatically input to this data.
- UpdateUser: value shows which user update values. Database Automatically input to this data.
- UpdateDate: value shows when user update values. Database Automatically input to this data.
- DepartmentId: value shows the place of the data in the Department table . This Data will input by user manually to the database.
- LoginId : value shows the place of the data in the Login table. This Data will input by user manually to the database.
- LoginName:value shows code of Logins name. This Data will input by user manually to the database.
- Password: value shows password of Login. This Data will input by user manually to the database.
- AcessLevel: value show user can where enter or where not. This Data will input by user manually to the database.
- StudentId : values show student number and the place of the data in the student table. This Data will input by user manually to the database.
- Name : value show student name. This Data will input by user manually to the database.
- Surname : value show student surname. This Data will input by user manually to the database.
- Sex : value show persons sex. This Data will input by user manually to the database.
- Adress : value show persons adress of home. This Data will input by user manually to the database.
- City : value show person where is he/she form in his/her country. This Data will input by user manually to the database.

- Country : value show where is person from. This Data will input by user manually to the database.
- Postcode : value show persons city's post codes. This Data will input by user manually to the database.
- GSM : value show persons mobie phone number. This Data will input by user manually to the database.
- FatherName : value show persons father's name. This Data will input by user manually to the database.
- MotherName : value show persons mother's name. This Data will input by user manually to the database.
- BirthPlace: value show persons where was born. This Data will input by user manually to the database.
- BirthDate: value show persons when was born. This Data will input by user manually to the database.
- OsymNo : value show student's OSYM no. This Data will input by user manually to the database.
- Email : value show persons e-mail adres. This Data will input by user manually to the database.
- RegisterDate : value show when person register. This Data will input by user manually to the database.
- Phone : value show persons phone number. This Data will input by user manually to the database.
- isGraduate: user can choise person graduate or not. This Data will input by user manually to the database.
- GraduationDate : value show when student graduate. This Data will input by user manually to the database.
- GraduationComment: value show any comment about graduated student. This Data will input by user manually to the database.
- isDismised: user can choise person dismised or not. This Data will input by user manually to the database.
- DismisedDate: value show when student dismised. This Data will input by user manually to the database.

- DismisedComment: value show why student dismised. This Data will input by user manually to the database.
- Tname: value show teacher's name. This Data will input by user manually to the database.
- Tsurname : value show teacher's surname. This Data will input by user manually to the database.
- TransId: value show
- Grade : value show student's grade. This Data will input by user manually to the database.
- CourseRegDate : value show when student decide to take this course. This Data will input by user manually to the database.
- GradeDate: value show when student take his/her grade. Database Automatically input to this data.
- AddDate : value show when student decide to add newcourse. Database Automatically input to this data.
- DropDate : value show when student decide to drop course. Database Automatically input to this data.

### Relations And Keys

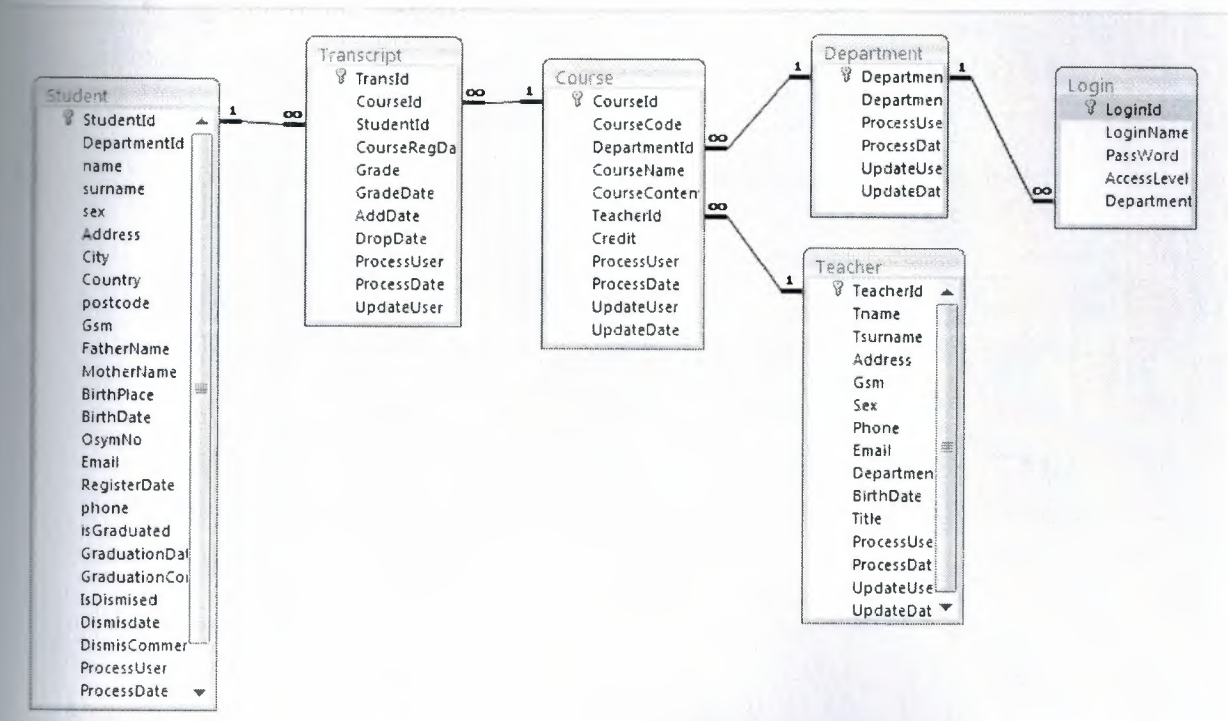



Figure 2.9 Relations

Program has 6 table and 5 relations those are :

1. Student Table → Transcript Table relations on StudentId key.
2. Transcript Table → Course Table relations on CourseId key.
3. Course Table → Department Table relations on DepartmentId key.
4. Course Table → Teacher Table relations on TeacherId key.
5. Department Table → Login Table DepartmentId key

So primary keys of tabeles are

- Student Table : StudentId
- Transcript Table : TranscriptId
- Course Table : CourseId
- Department Table : DepartmentId

- Teacher Table : TeacherId
- Login Table : LoginID

# CHAPTER THREE

# USERS MANUAL

After exacuting the main program thsi following page welcomes us (figure 3.1)


Figure 3.1 Login Menu

This is the Control interface of program. The user select a name and enter password.

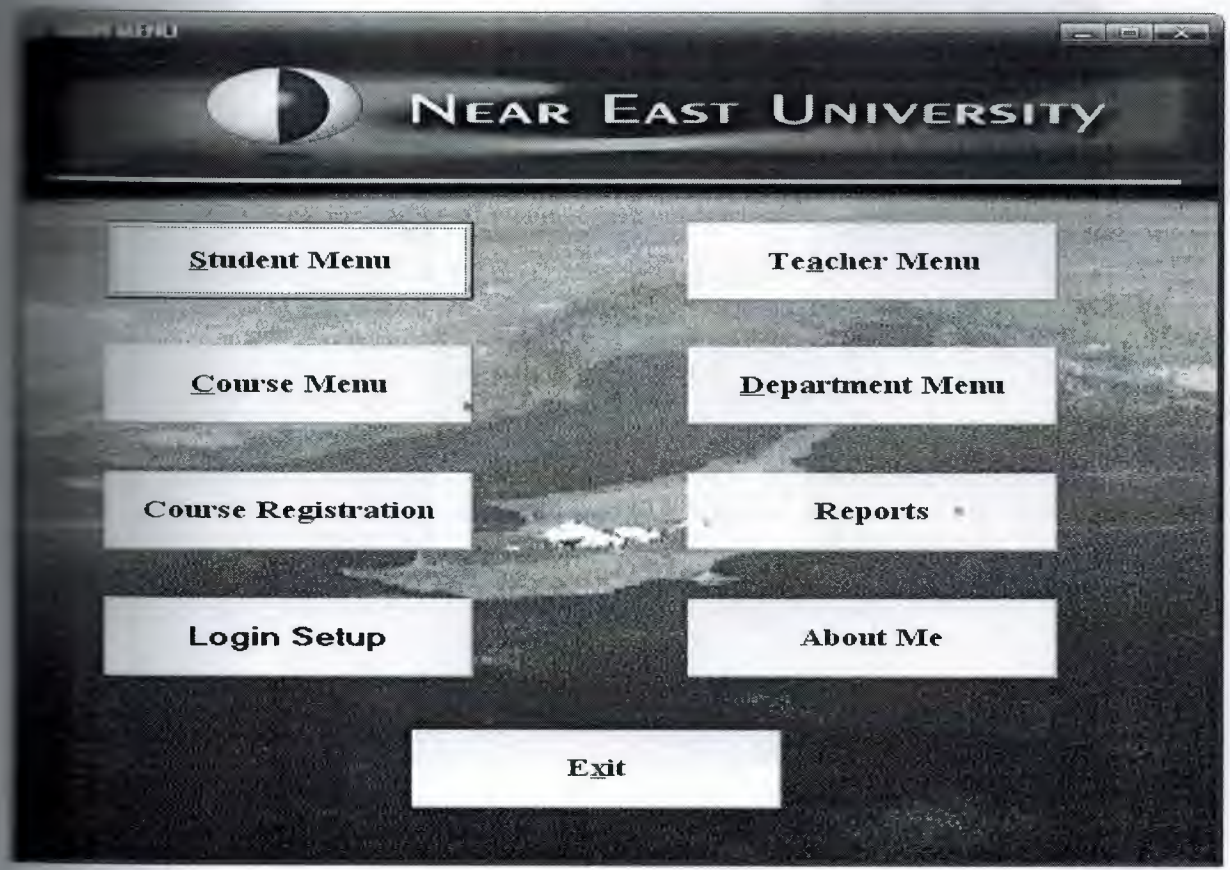



Figure 3.2 Main Menu

This is the main view of the program that we can select the operations.

We have 6 operation buttons and 2 information buttons. These are "**Student Menu**" for student operations, "**Teacher Menu**" for teacher operations, " **Course Menu**" for course operations, "**Department Menu**" for department operations, "**Course Registration** " for Course and Grade relations, "**Login Setup**" for user and user's Access level, "**Reports**" for report screens of datas,

"**About Me**" for programmers information page, "**Exit**" for Exit


Figure 3.3 Student Menu

When you select "**Student Menu**" button on main menu, the student menu window in figure 3.3 will open automatically. This menu has 3 buttons, "**Add Student**" for adding new student into our Student list. "**Update Student**" for updating student informaition of added student in the student list.

When you select "**Add Student**" buton on "**Student Menu**" , the Add student form window in figure 3.4 will be opened automatically. After that you have to write student id and click "**Search**" button to enter the details of new student, then you have to click "**Save**" button to add student to database safely.

If you press the "**Cancel**" button then the program will cancel the record and everything you wrote will be truncated automatically. If you press the "**Student Menu**" button then the program will return to student menu. If you press the "**Main Menu**" button then the program will return to main menu.

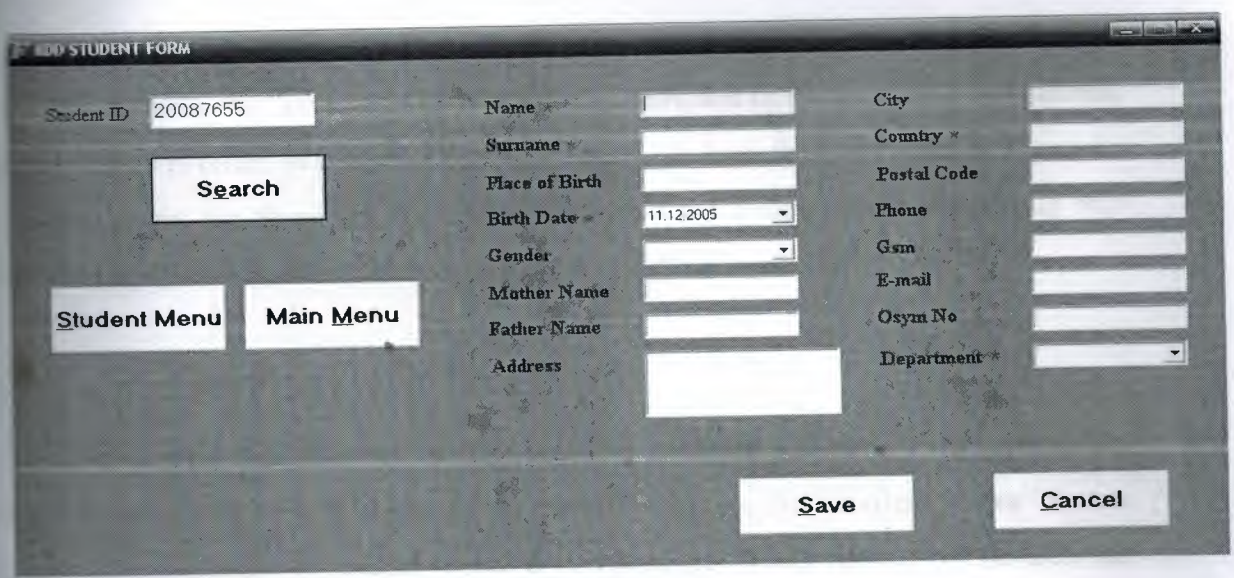



Figure 3.4 Add Student

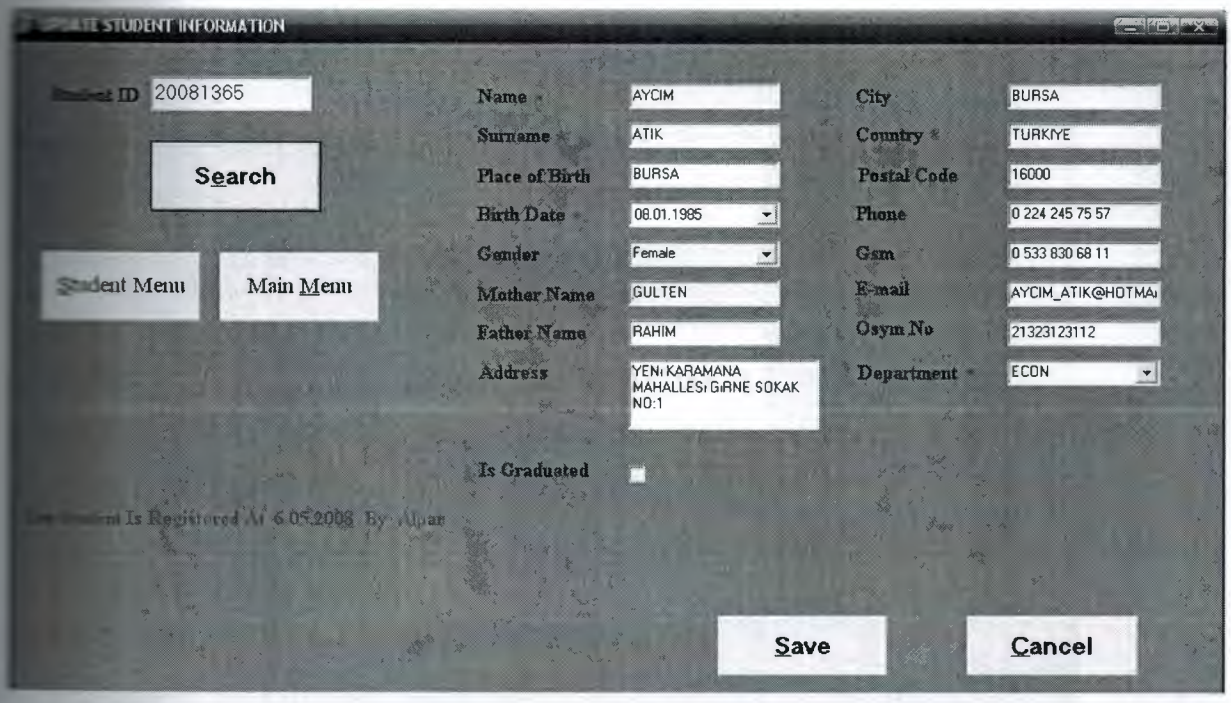



Figure 3.5 Update Student

In figure 3.5 we see our " Update Student Information" window after pressing the " **Update Student** " in Student Menu. İn this window you can select student and edit students details which already added to database once a time ago.

When you press the "save" button you will be redirected to the Student ID input box to input new values in order. Whenever you fill all boxes, click on the "save" buton. Then you will be warned about the dbase saved. "The student Updated Succesfully" will pop up into your screen in popup box and then the record will be saved automatically.

If you press the "**Cancel**" button then the program will cancel the record and everything you wrote will be truncated automatically.If you press the "**Student Menu**" button then the program will return to student menu. If you pres the "**Main Menu**" button then the program will return to main menu.

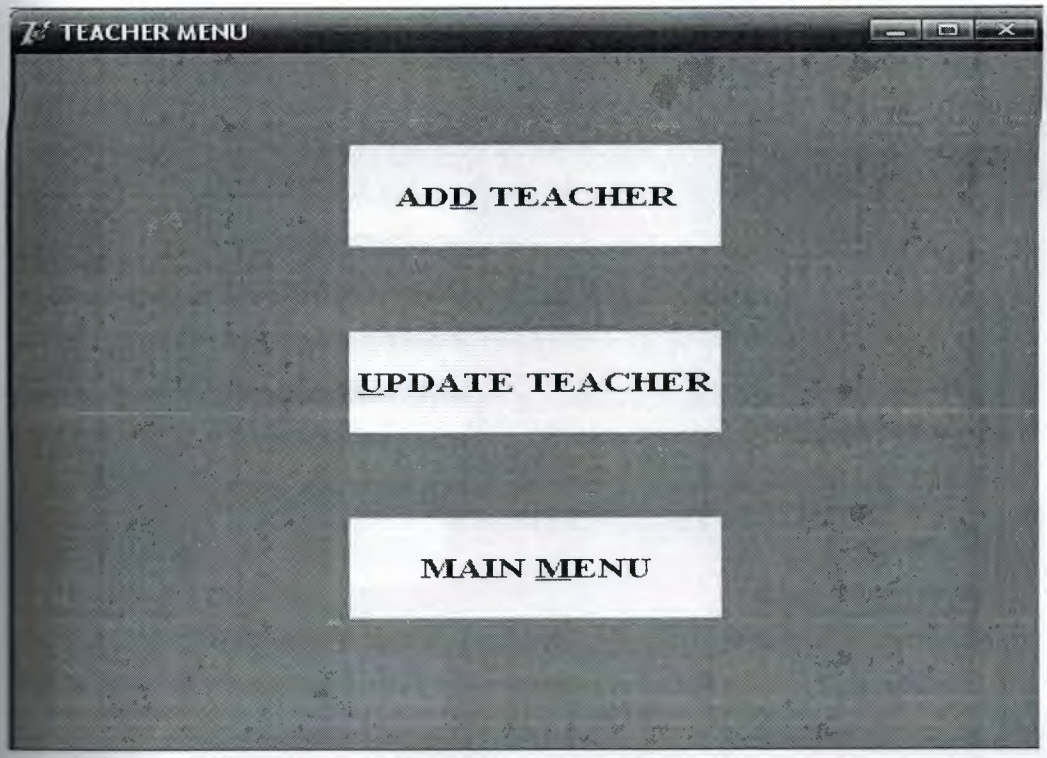



Figure 3.6 Teacher Menu

When you Select "**Teacher Menu**" button on main menu, the teacher menu window in figure 3.6 will open automatically. This menu has 3 buttons "**Add Teacher**" for adding new teacher into our teacher list. "**Update Teacher**" for updating teacher information of selected teacher in teacher list.

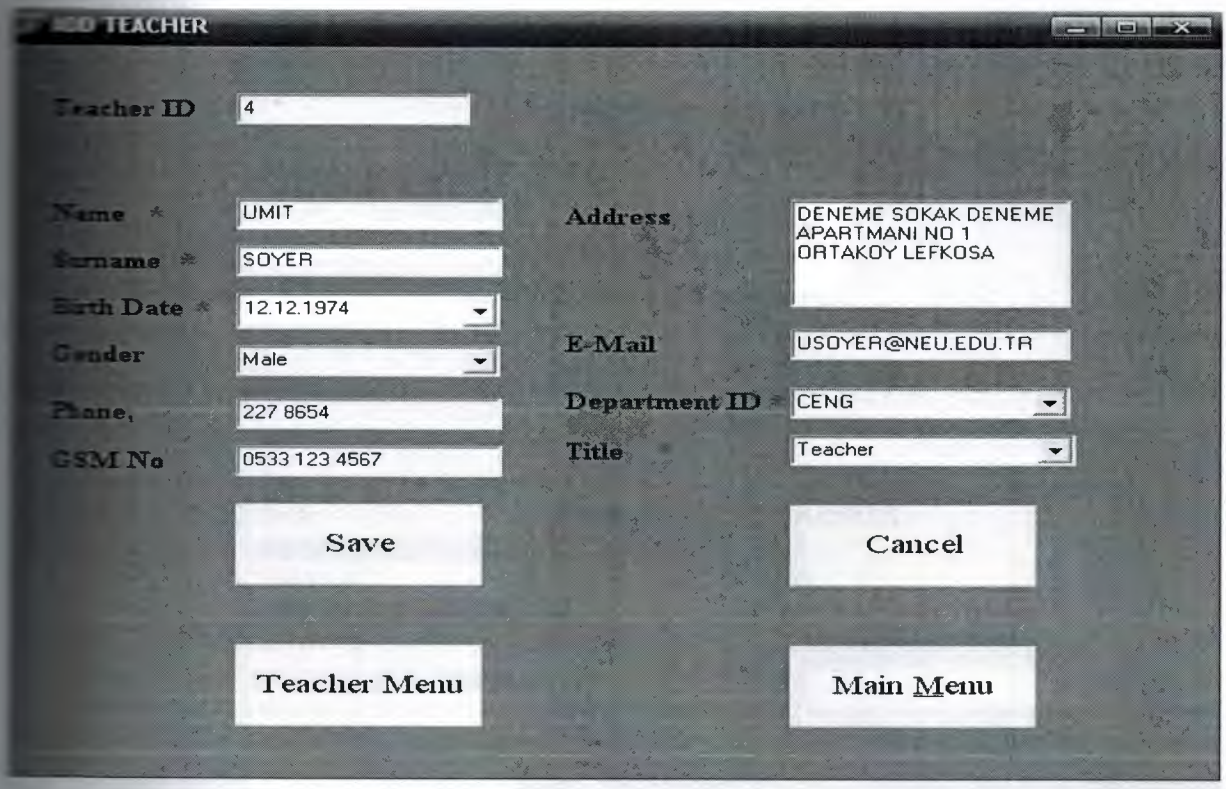



Figure 3.7 Add Teacher Window

When you select "**Add Teacher**" buton on "**Teacher Menu**", Add teacher form window in figure 3.7 will be opened automatically. After that you have to write teacher id and other details of new teacher,then you have to click "**Save**" buton to add teacher with database easily.

If you press the "**Cancel**" button then the program will cancel the record and everything you wrote will be truncated automatically. If you press the "**Teacher Menu**" button then the program will return to teacher menu. If you pres the "**Main Menu**" button then the program will return to main menu.

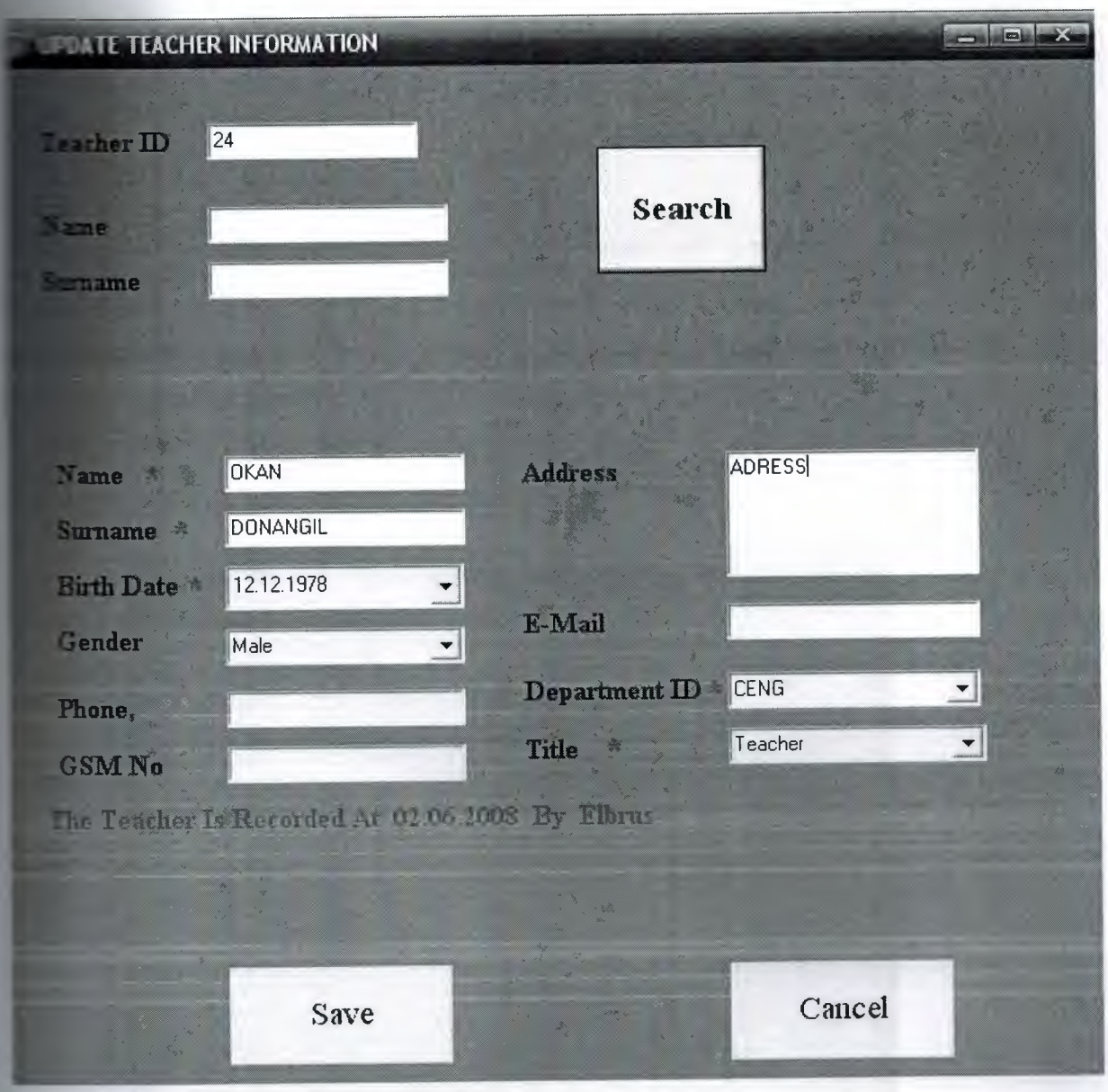



Figure 3.8 Update Teacher Window

In figure 3.8 we see our " Update Teacher Information" window after pressing the " **Update Teacher** " in Teacher Menu. İn this window you can select teacher and edit teacher details which already added to databese once a time ago.

When you press the "save" button you will be redirected to the TeacherId input box to input new values in order. Whenever you fill all boxes, click on the "save" buton. Then you

about the dbase saved. "The Teacher Information Recorded Succesfully" will screen in popup box and then the record will be saved automatically.

press the "**Cancel**" button then the program will cancel the record and you wrote will be truncated automatically.If you press the "**Teacher Menu**" program will return to Teacher menu. If you pres the "**Main Menu**" button, the return to main menu.

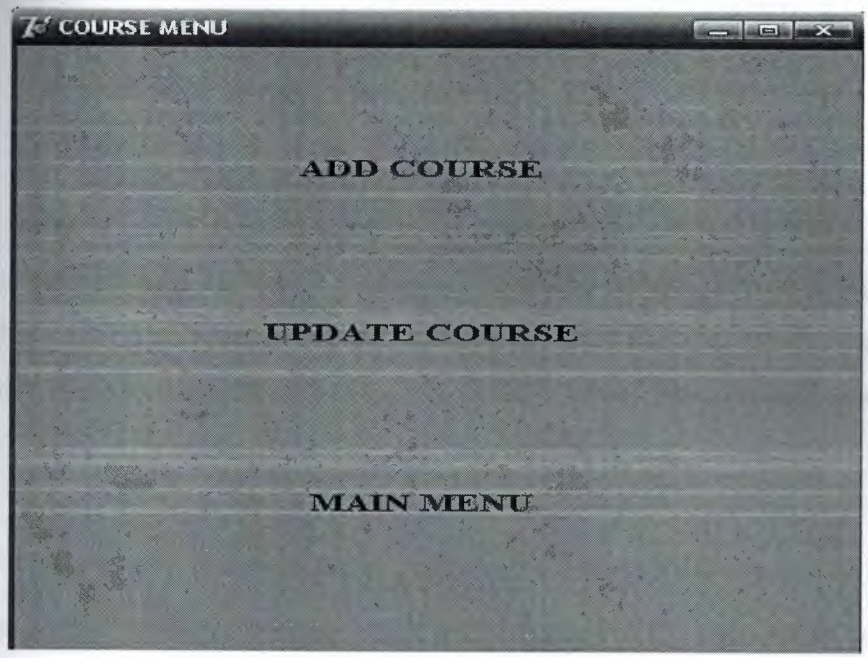



Figure 3.9 Course Menu

When you Select "**Course Menu**" buton on main menu then the teacher menu window in figure 3.9 will open automatically. This menu has 3 buttons "**Add Course**" for new course into our course list. "**Update Course**" for updating teacher informaition added course in course list.

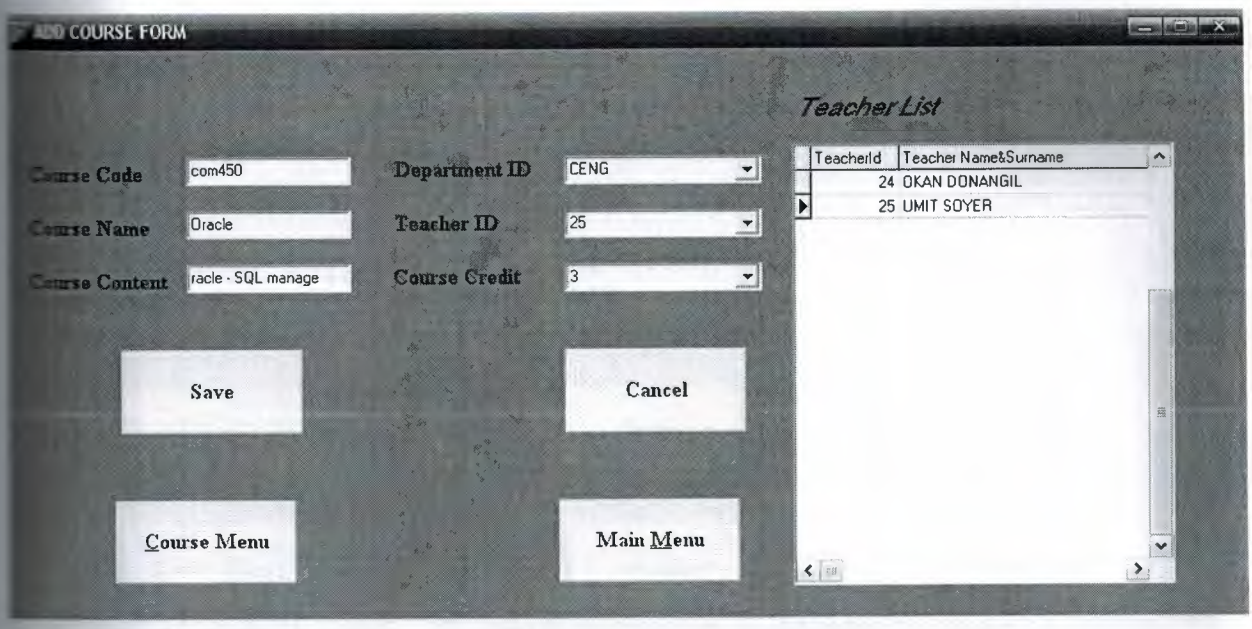



Figure 3.10 Add Course Menu

When you select "**Add Course**" buton on "**Course Menu**" then the Add Course form window in figure 3.10 will be opened automatically , After this you have to write course id and enter details of new course,then you have to click "**Save**" buton to add course with database safety.

If you press the "**Cancel**" button then the program will cancel the record and everything you wrote will be truncated automatically.If you press the "**Course Menu**" button then the program will return to course menu. If you pres the "**Main Menu**" button then the program will return to main menu.

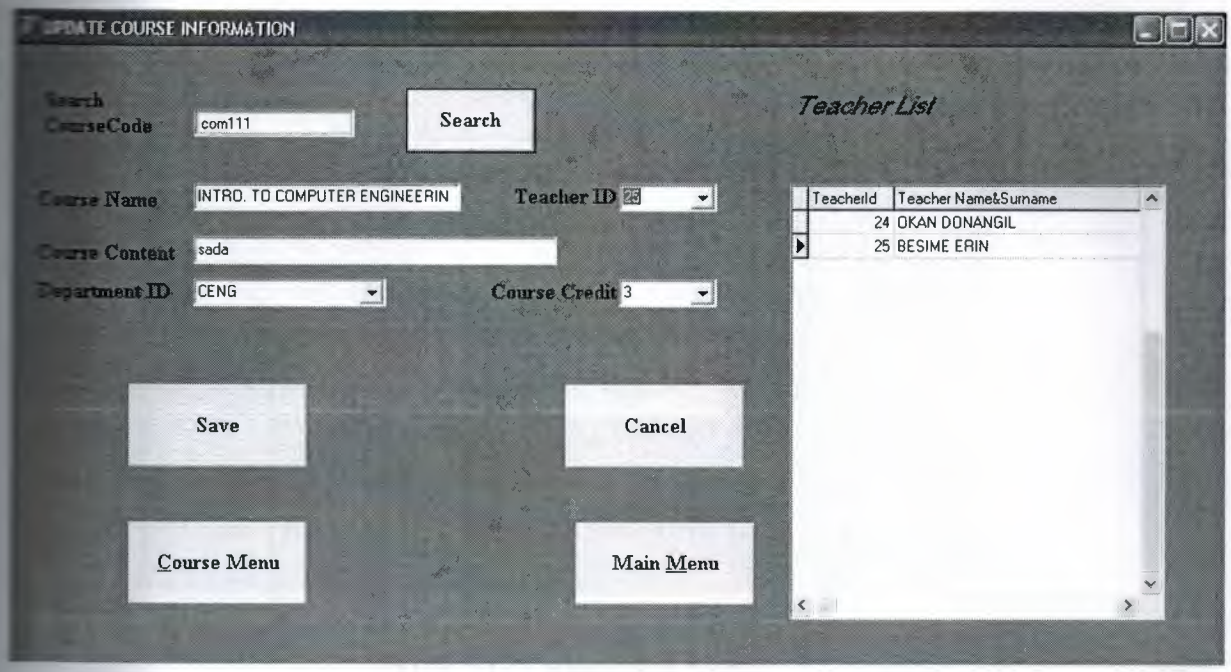



Figure 3.11 Update Course Menu

In figure 3.11 we see our " Update Course Information" window after pressing the " **Course** " in course Menu. İn this window you can select course and edit course which already added to database once a time ago.

When you press the "save" button you will be redirected to the CourseCode input box new values in order. Whenever you fill all boxes, click on the "save" buton. Then you be warned about the dbase saved. "The Course Updated Succesfully" will pop up into screen in popup box and then the record will be saved automatically.

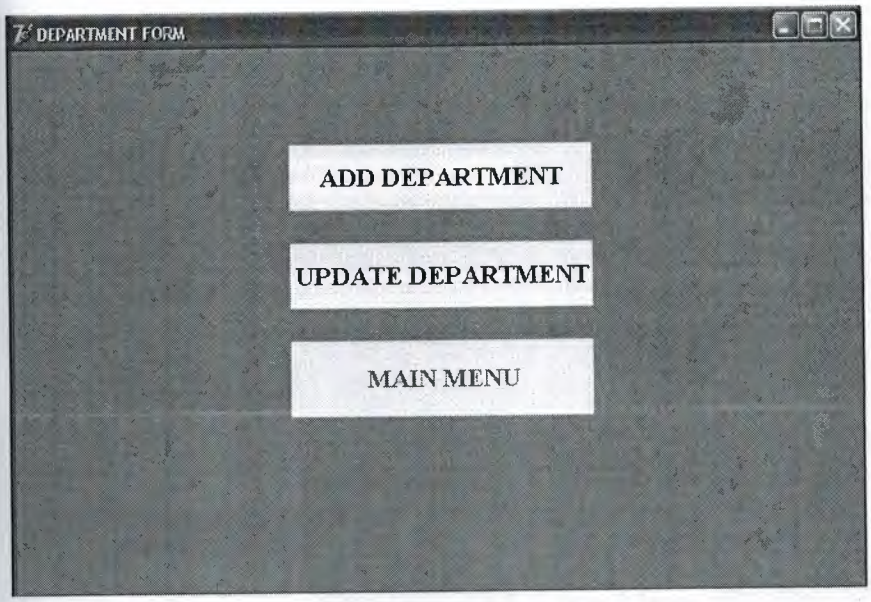



Figure 3.12 Department Menu

When you Select "**Department Menu**" buton on main menu then the department menu window in figure 3.12 will be Automatically.So this menu has 3 button "**Add Department**" for Add new department into our department list. "**Update Department**" for update department informaition of added department in department list.

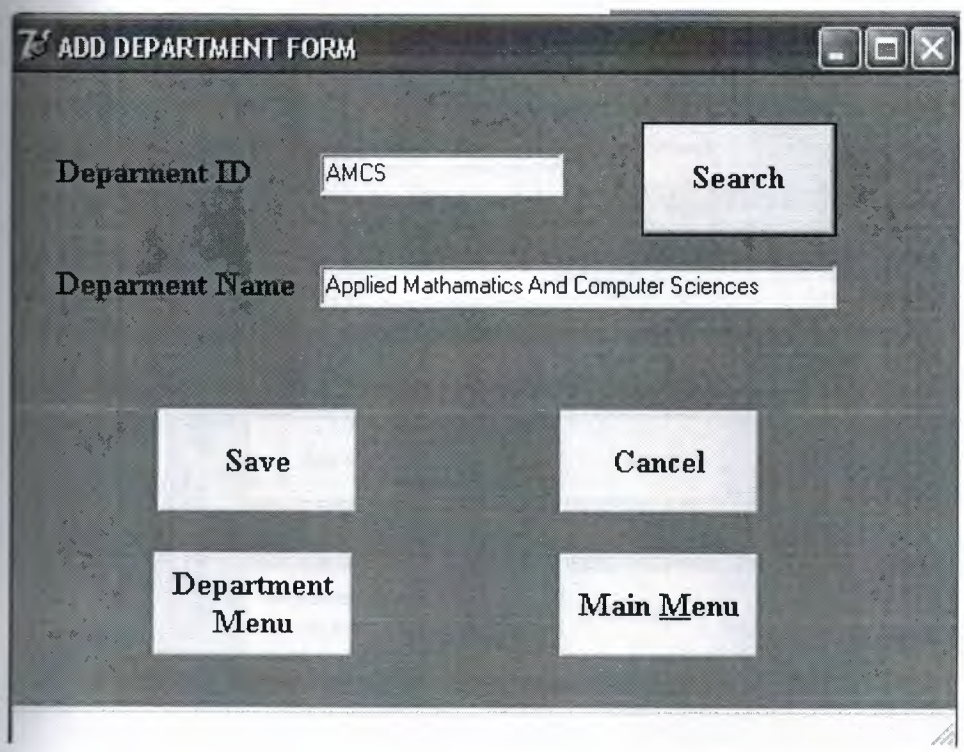



Figure 3.13 Add Department Menu

When you select "**Add Course**" buton on "**Course Menu**" then the Add Course [illegible] window in figure 3.10 will be opened automatically , After this you have to write course [illegible] and enter details of new course,then you have to click "**Save**" buton to add course with [illegible] safety.

If you press the "**Cancel**" button then the program will cancel the record and [illegible] you wrote will be truncated automatically.If you press the "**Department Menu**" [illegible] then the program will return to department menu. If you pres the "**Main Menu**" [illegible] then the program will return to main menu.

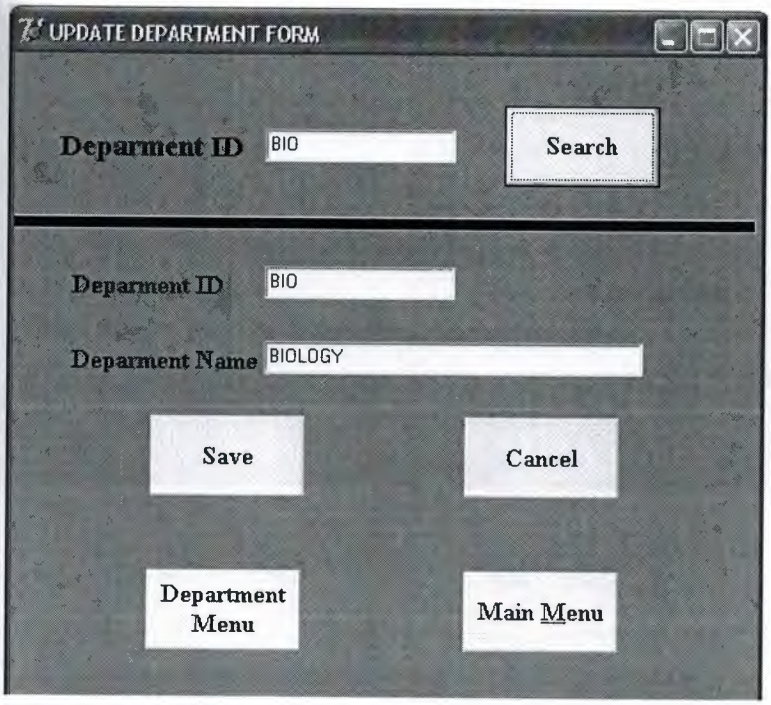



Figure 3.14 Update Department Window

Figure 3.11 Update Course Menu

In figure 3.14 we see our " Update Department " window after pressing the " **Update Departmen** " in Department Menu. İn this window you can select department and edit department details which already added to database once a time ago.

When you press the "save" button you will be redirected to the DepartmentId input box to input new values in order. Whenever you fill all boxes, click on the "save" buton. Then you will be warned about the dbase saved. "The Department Information Recorded Succesfully" will pop up into your screen in popup box and then the record will be saved automatically.

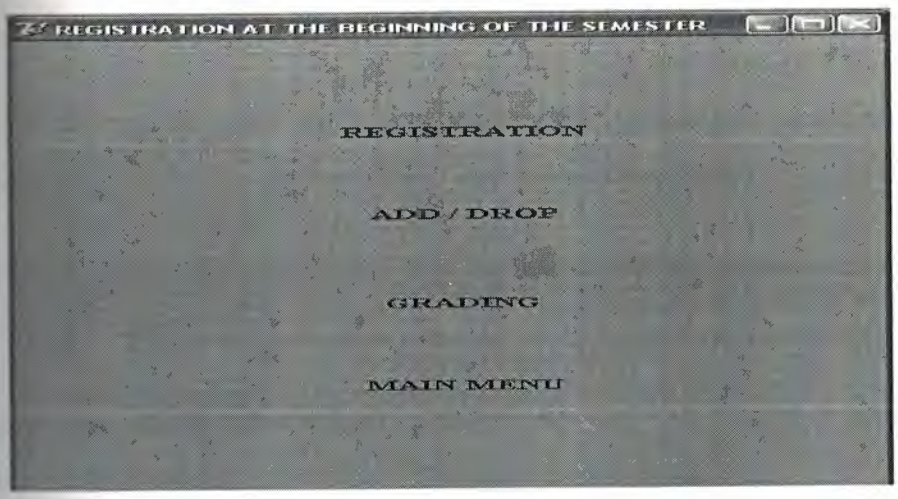
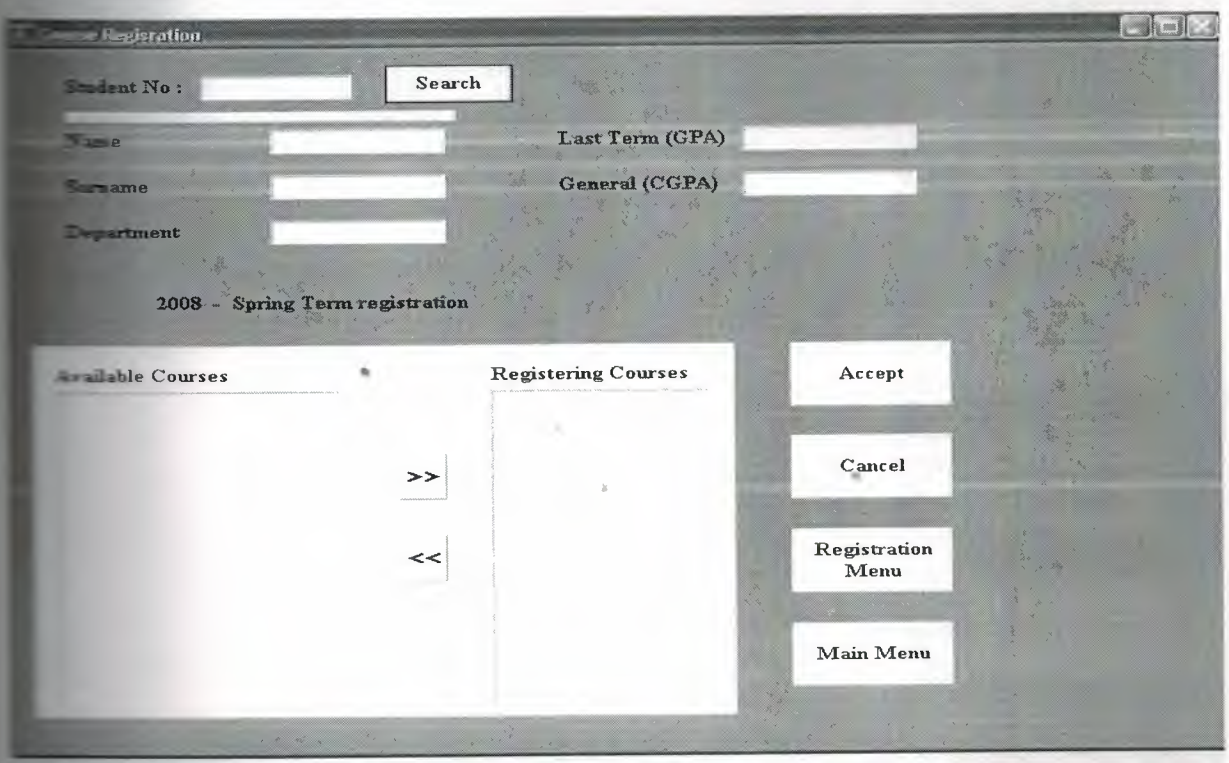



Figure 3.15 Course Registration Menu

Course registration menu has 4 buttons, these are Registration,**Add/Drop,Grading,Main Menu.** Main menu is used to return to main menu as mentioned before.




3.16 Course Registration Menu

In this Menu when we enter studentid program lists available courses and shows Last term GPA and CGPA ,so we adding course for new semester,

The program reads the student id and brings the information about the student which is surname, GPA and CGPA.

The user(advisor) determines the courses that the student will take next semester and add the information to the student's transcript by saving it to database.

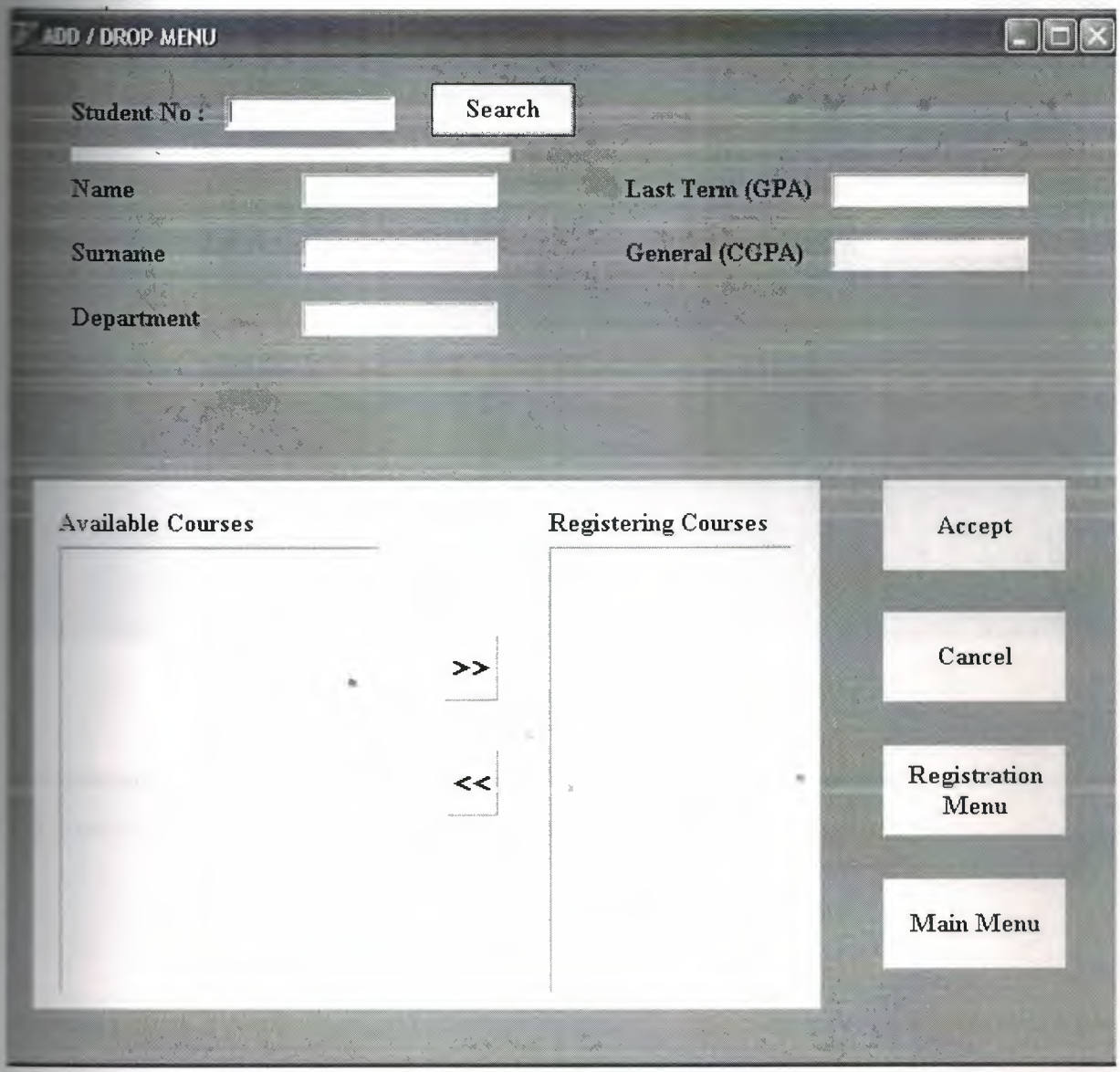



Figure 3.17 Add /Drop Menu

Bu menude registration menuye benzer tek farkı daha önceden database girilen transcript tablosundan çağırım yeniden işlememize (yani update ) etmemizi sağlar

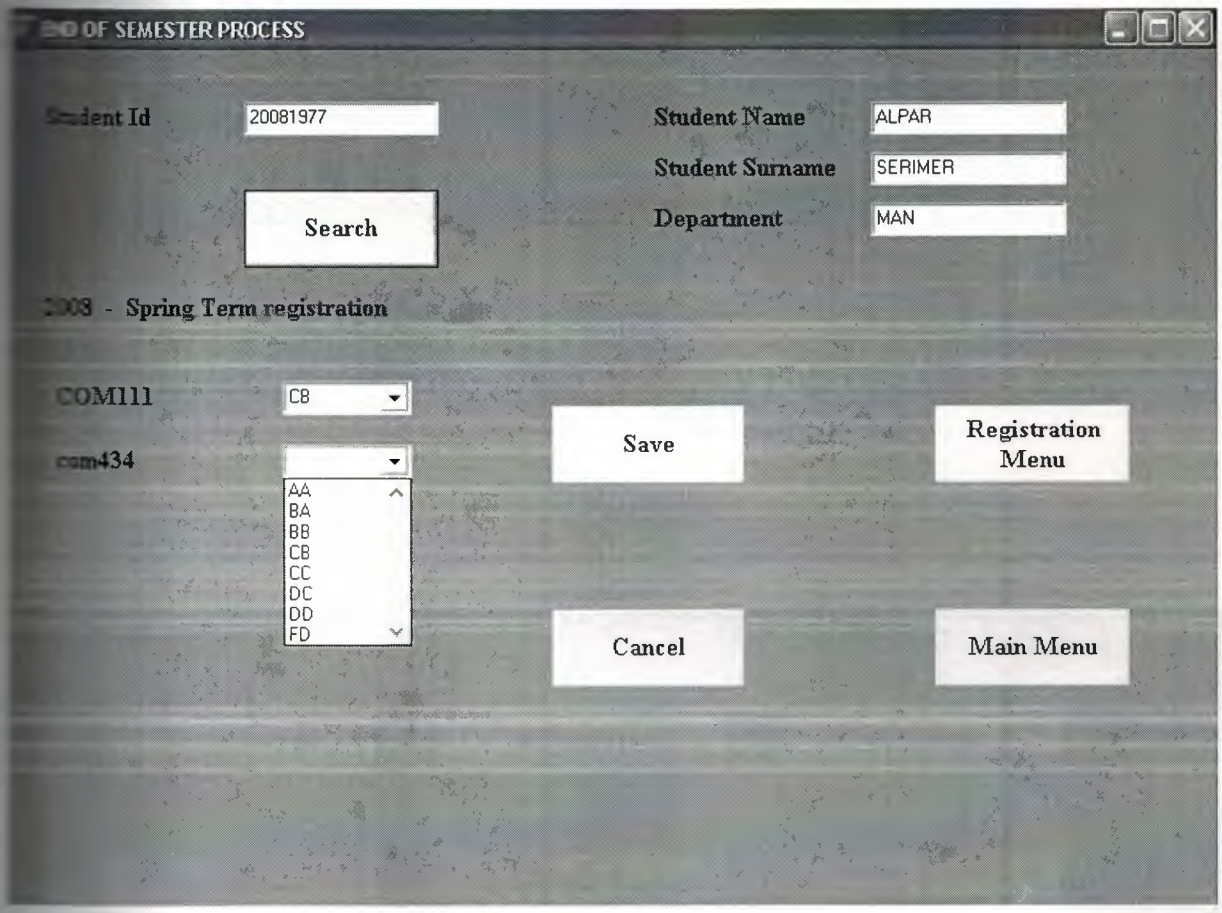



Figure 3.18 Grade Menu

In this menu data which we call from student the course which are belong to we add to we points to them. I mean we can point the activity of student in that part.

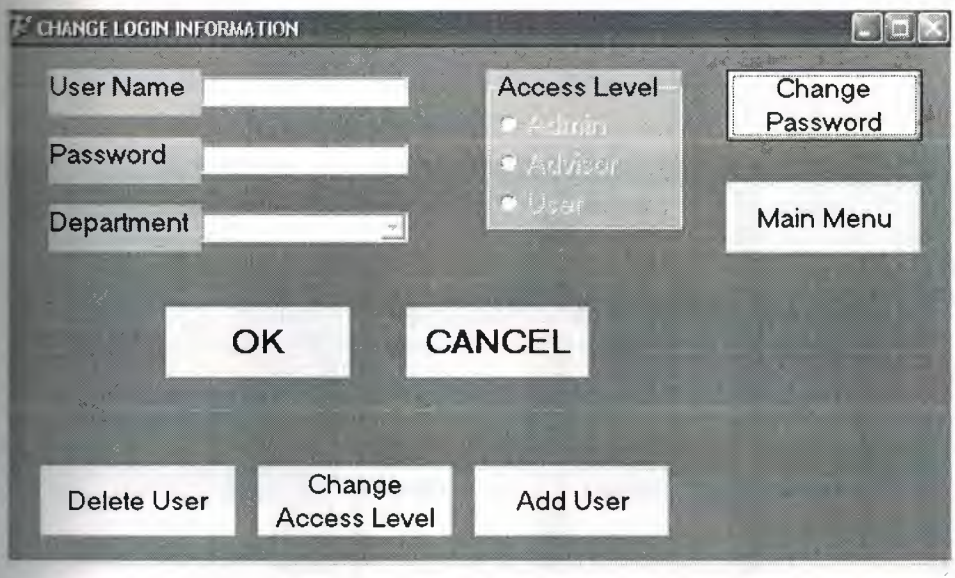



Figure 3.19 User Configration Menu

This program points out that the user's position and it points the parts which users can

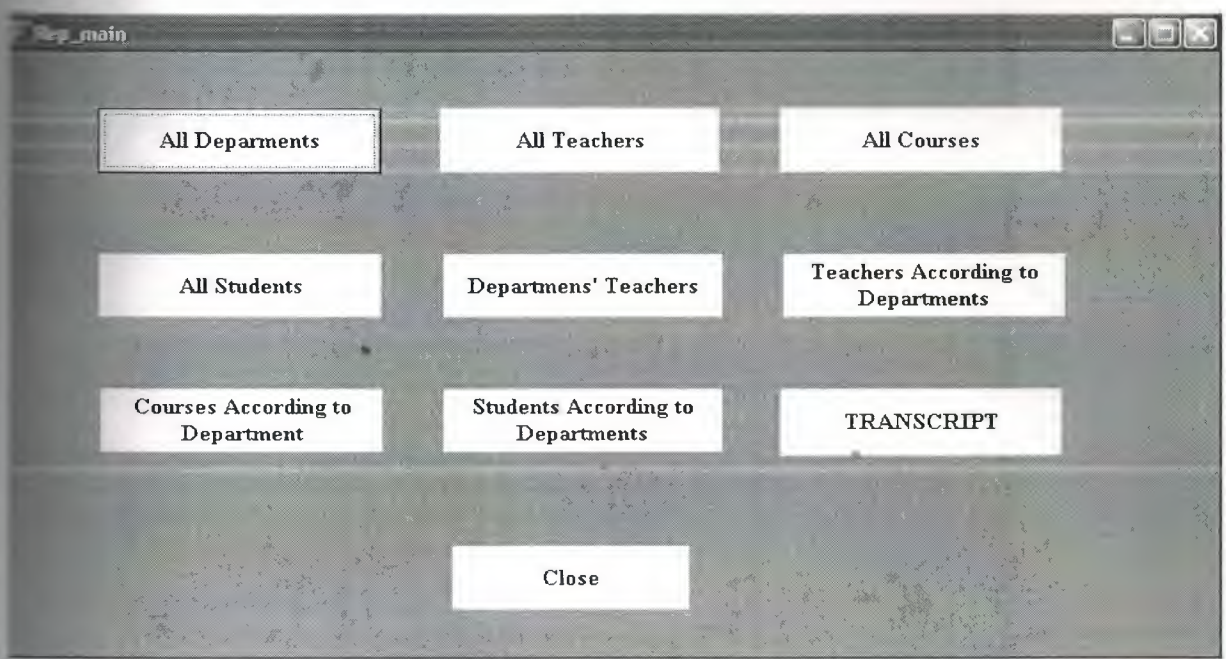



Figure 3.20 Report Menu

It's possible to see all databases and print out information from this menu. These 8 menus works out similarly but the transcript menu has a different function.

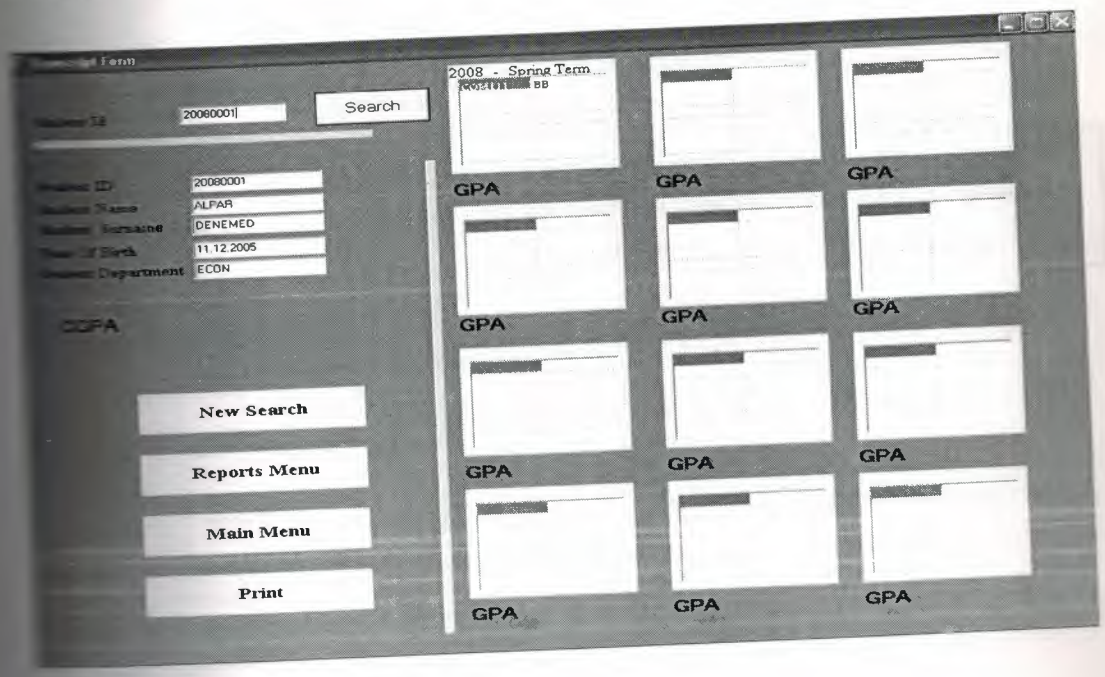



Figure 3.21 Transcript Form

In transcript menu, it shows all courses and grades of students added each semester. This menu is an output form of transcript table.

Diagrams

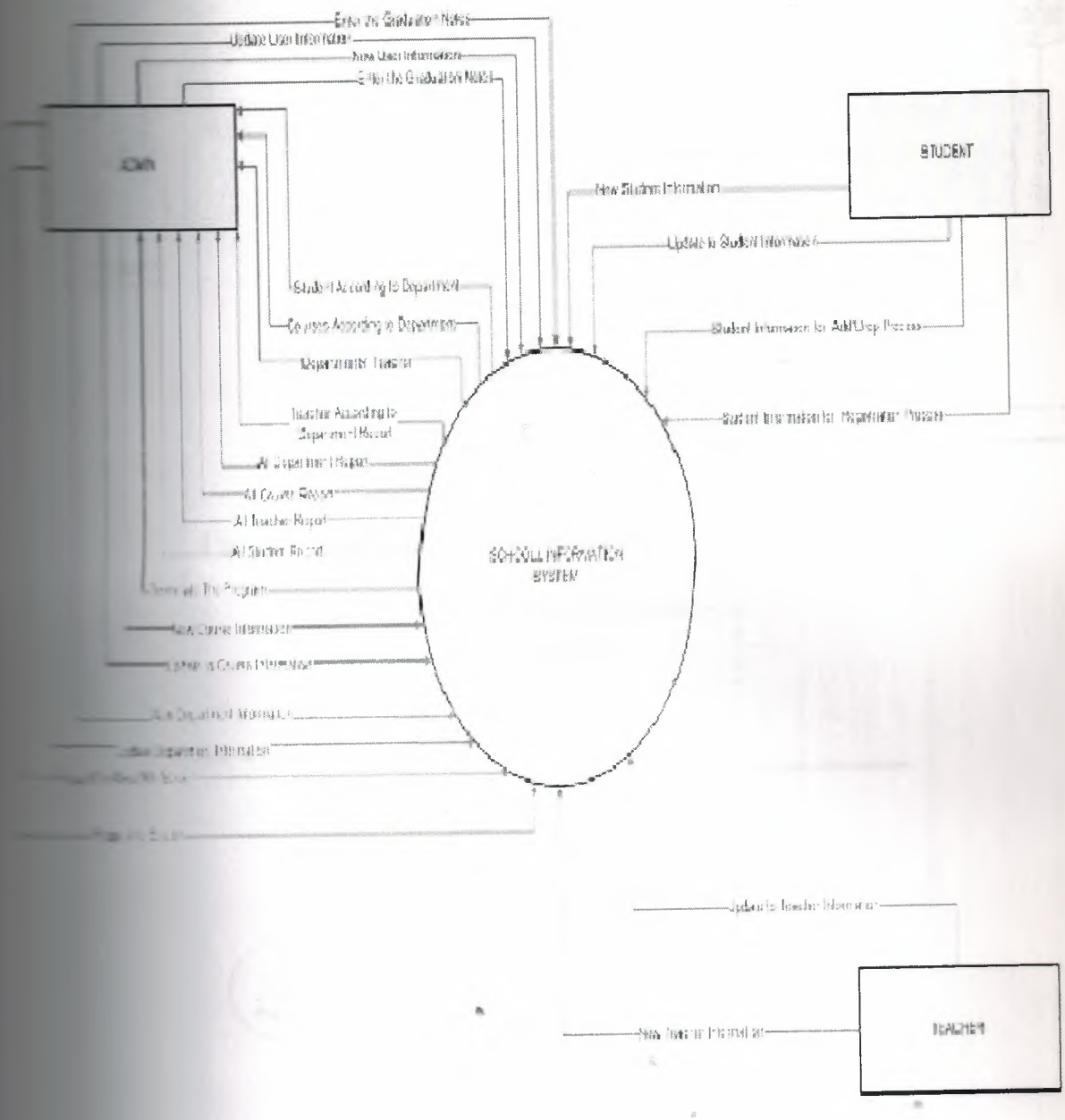



Figure 3.22 School Information System For Contex Diagram

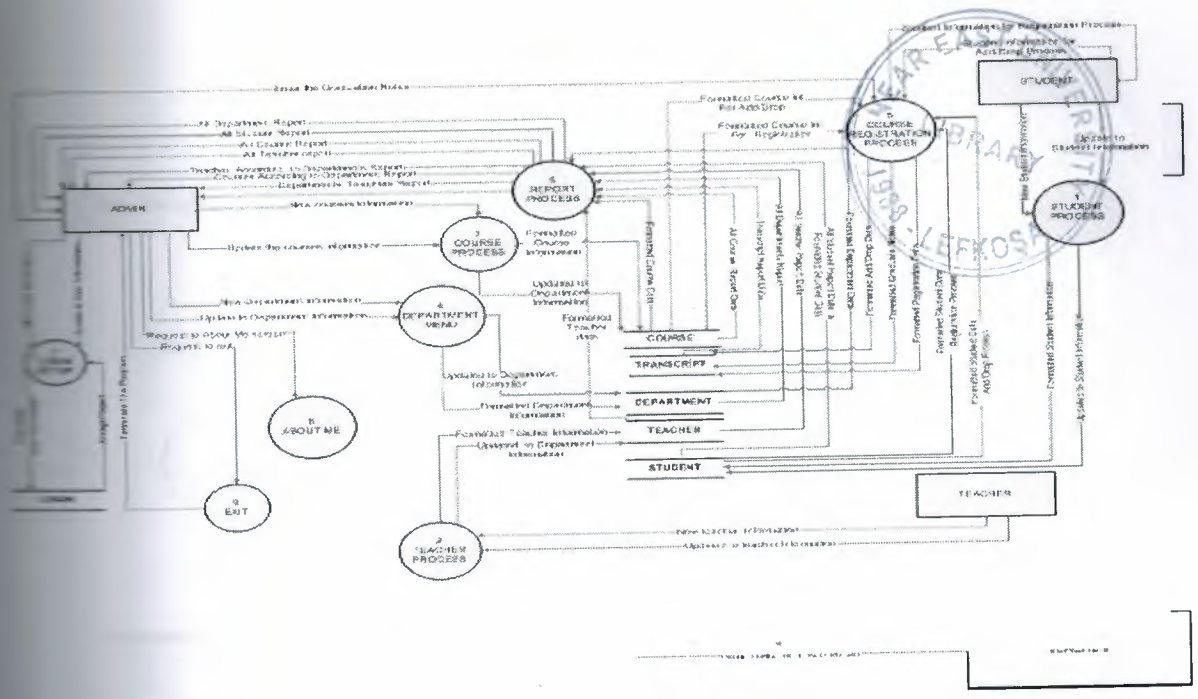



Figure 3.24 School information system top Level Diagram

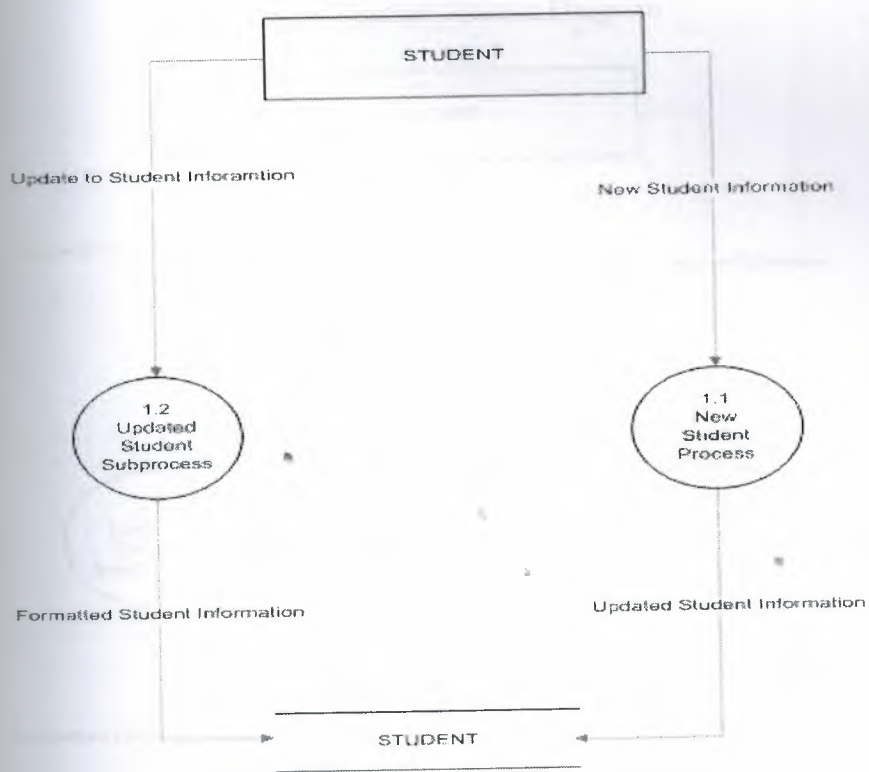



Figure 3.25 Student Process Diagram

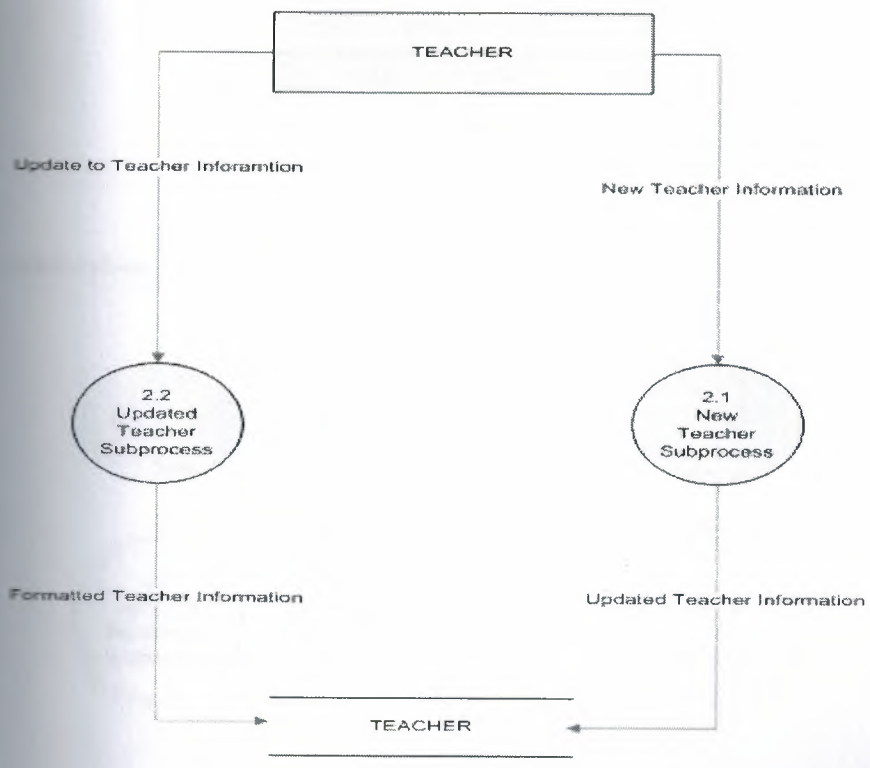



Figure 3.26 Teacher Process Diagram

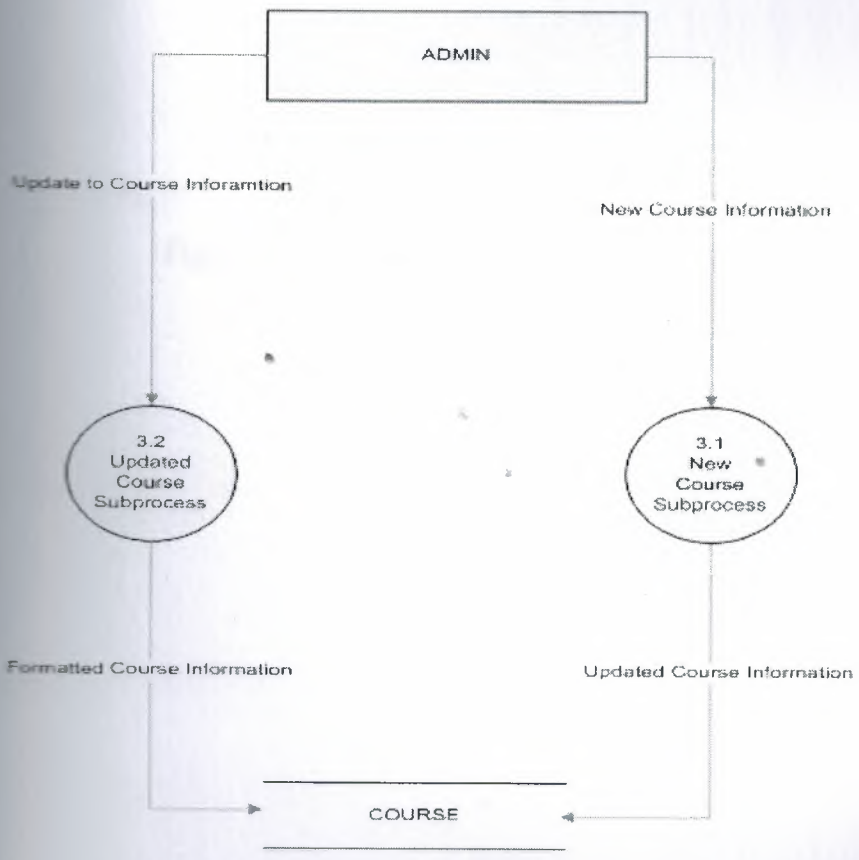



Figure 3.27 Course Process Diagram

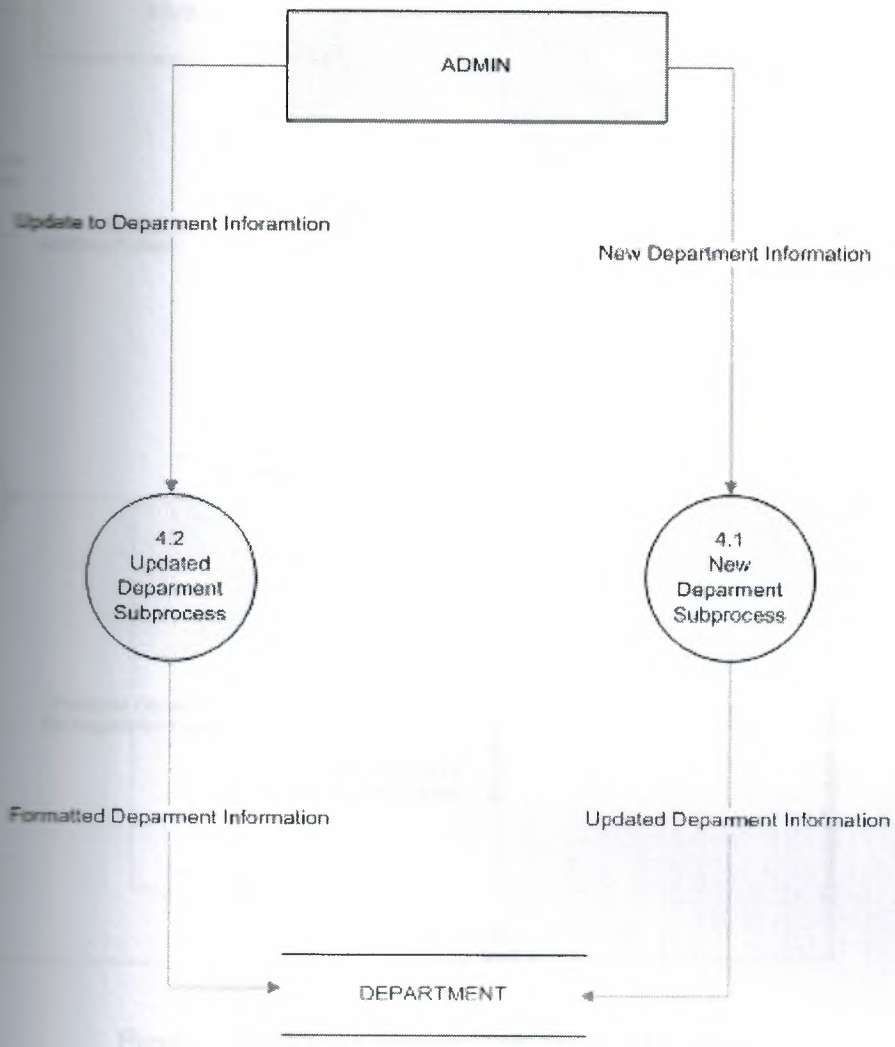



Figure 3.28 Department Process Diagram

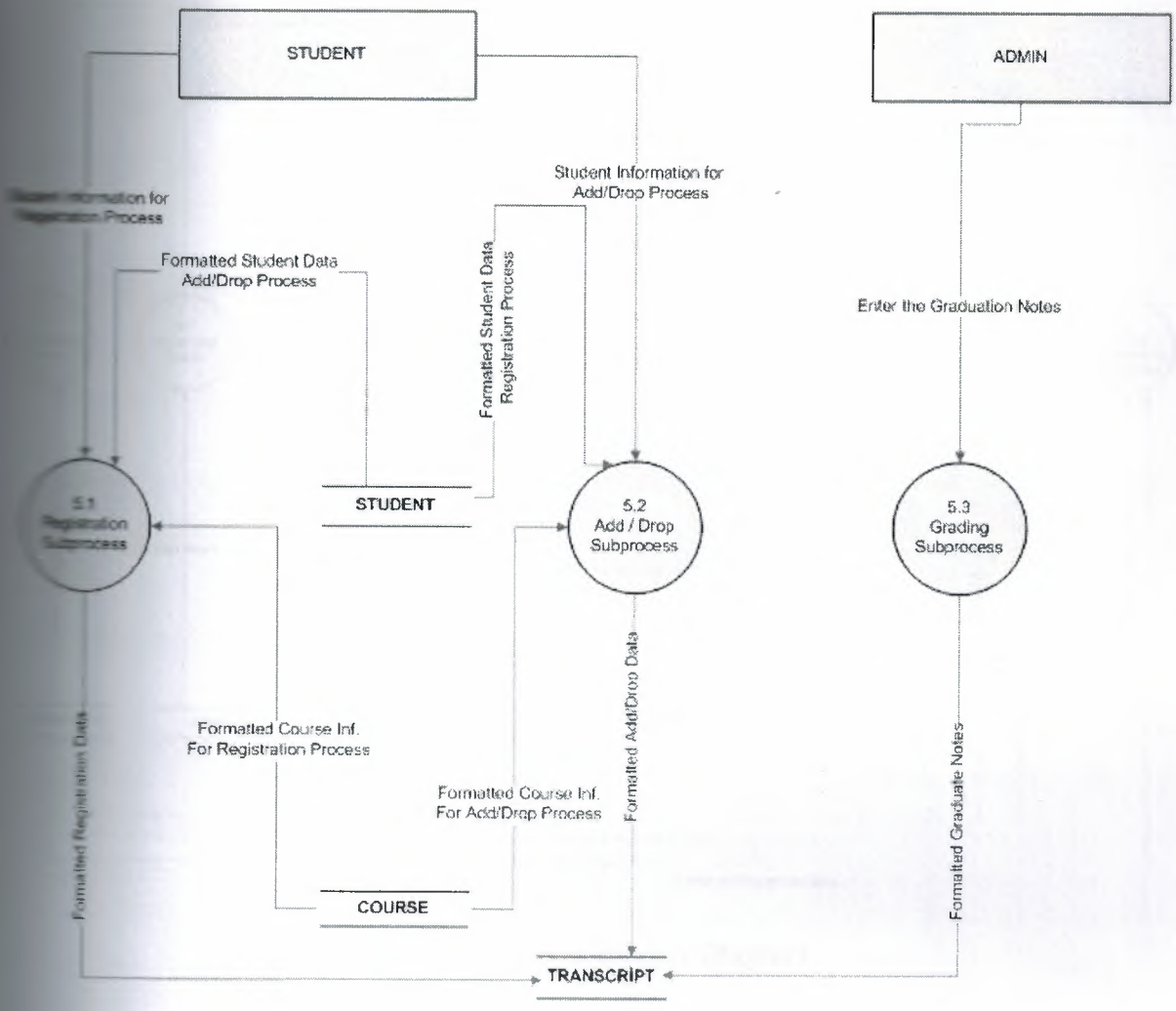



Figure 3.29 Course Registration Process Diagram

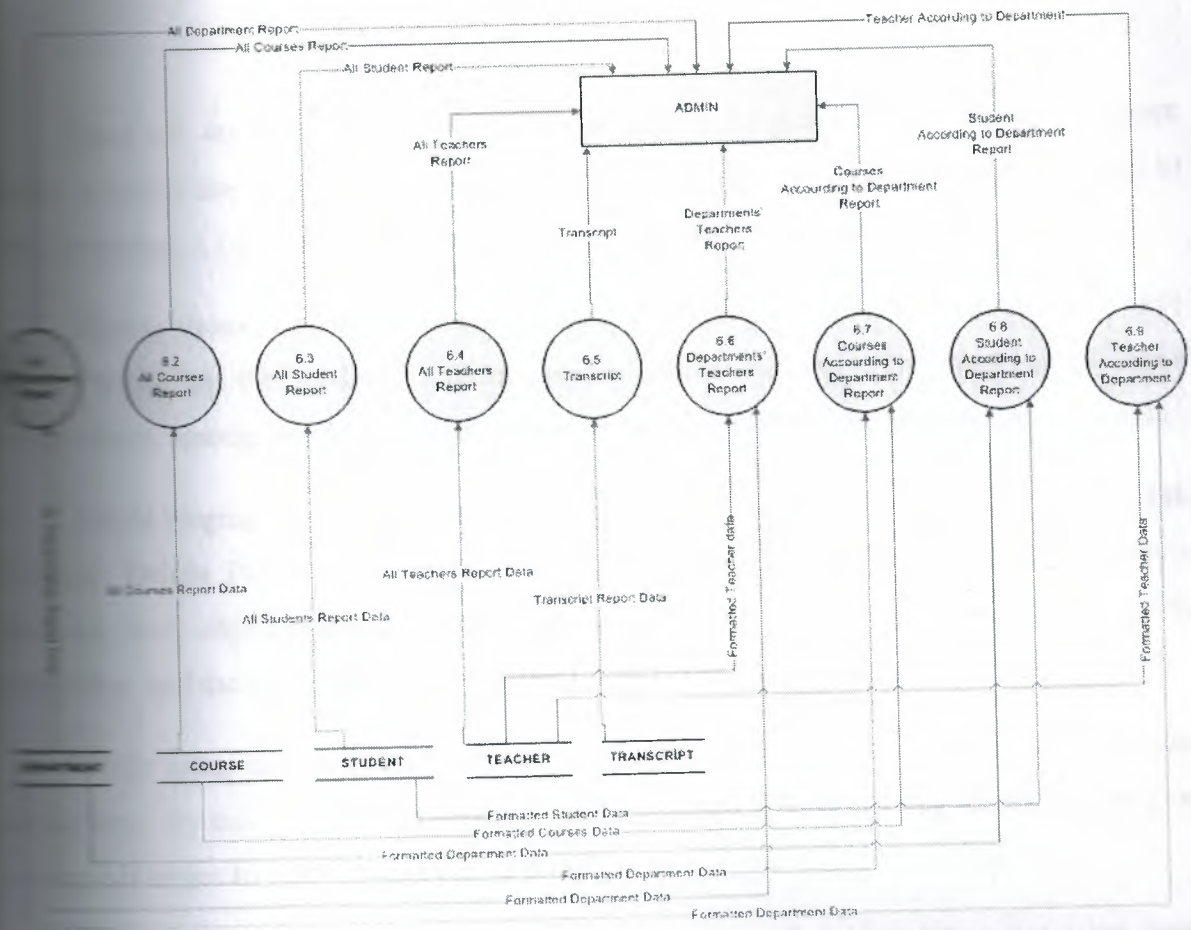



Figure 3.30 Report Process Diagram

## CONCLUSION

There are 40 forms and 6 database tables used in the program,this program has been written to enable the students, teachers,departments to Access and Exchange information to easily. Nowadays it has been a must for schools to have such kind of programs.

Schools should be careful when choosing their program. Interface should be simple and easy to use for the student , academician and whole university. The interface will be a popular choice among people due to its been simple and well–thought till the smallest details.

Delphi language, a set of object-oriented extensions to standard Pascal, is the language of Delphi. Delphi Pascal is a high-level, compiled, strongly typed language that supports structured and object-oriented design. Its benefits include easy-to-read code, quick compilation, and the use of multiple unit files for modular programming.

While using this program you can save your datas safely and retrieve them without wasting time. By this way you can exacute your operation very fast. This progfam can give so many advantges to teachers and the students.

At least I want to say that this program support with Delphi and Pascal, I mean with this two base program I tried to show that it is very easy to use and have lots of advantages.

Delphi is a general purpose development environment for all versions of Windows. It is the most powerful tool available for software developers working on the Windows platform. It can produce any kind of Windows application, including Service and Console applications, IIS extensions, etc, but in the specific case of GUI applications and database access, it is the best tool available, bar none.

It can create programs that run on any version of Windows, access every major kind of Database, including ORACLE, SQL Server, DB2, Interbase, Firebird, and supports n-tier and Client/Server architectures. It has always supported native (Win32) development but also provides support for .Net development and, soon according to the current roadmap, will include support for native 64 bit and the newest .Net releases.

Interestingly, the underlying architecture of the Delphi product and its object-oriented VCL framework (Visual Control Library) has allowed it to bring its considerable strengths to .Net development as well. Experienced Delphi programmers can be productive in .Net

immediately because they can continue to use the same VCL framework classes and powerful IDE design environment they have used for native-code development, yet the resulting .Net executable is a 100% managed .Net application. Existing application code, perhaps dating all the way back to Delphi's first version in 1995, can often migrate with little or even no change into .Net where it can be leveraged by being able to introduce new features and abilities found in the .Net platform. New development for .Net can leverage this VCL platform, or use the Windows Forms framework provided by Microsoft, and Delphi provides ASP.Net development too.

# References


[1] Mastering Borland Delphi 2005 (mastering) by Marco Cantu' (paperback Aug 19, 2005)

[2] Inside Delphi 2006 (wordware Delphi Developer's Library) by ivan Hladni (paperback – Jan 25 2005)

[3] www.programlama.com

[4] www.delphiturkiye.com

[5] www.ceviz.net

[6] www.google.com

[7] www.wikipedia.org

# APPENDIX

## PROGRAM CODE

```
Windows, Messages, SysUtils, Variants, Classes, Graphics, Controls, Forms,
Dialogs, DB, DBTables, ADODB, StdCtrls, ExtCtrls, jpeg;
type
  Tfm_main = class(TForm)
    Image1: TImage;
    Image2: TImage;
    Button1: TButton;
    Button2: TButton;
    Button4: TButton;
    Button5: TButton;
    Button7: TButton;
    DataSource1: TDataSource;
    ADOTable1: TADOTable;
    ADOTable2: TADOTable;
    ADOTable3: TADOTable;
    ADOTable5: TADOTable;
    ADOTable6: TADOTable;
    DataSource2: TDataSource;
```

# APPENDIX

## PROGRAM CODE

Messages, SysUtils, Variants, Classes, Graphics, Controls, Forms,

DB, DBTables, ADODB, StdCtrls, ExtCtrls, jpeg;

= class(TForm)

TImage;

TImage;

TButton;

TButton;

TButton;

TButton;

TButton;

TDataSource;

TADOTable;

TADOTable;

TADOTable;

TADOTable;

TADOTable;

TDataSource;

DataSource3: TDataSource;

DataSource5: TDataSource;

DataSource6: TDataSource;

ADOQuery1: TADOQuery;

DataSource7: TDataSource;

ADOQuery2: TADOQuery;

DataSource8: TDataSource;

ADOQuery3: TADOQuery;

DataSource9: TDataSource;

ADOQuery4: TADOQuery;

DataSource10: TDataSource;

ADOQuery5: TADOQuery;

DataSource11: TDataSource;

ADOQuery6: TADOQuery;

DataSource12: TDataSource;

Button8: TButton;

Button6: TButton;

Button10: TButton;

Button11: TButton;

ADOTable4: TADOTable;

DataSource4: TDataSource;

DataSource13: TDataSource;

procedure Button7Click(Sender: TObject);

procedure Button1Click(Sender: TObject);

procedure Button2Click(Sender: TObject);

procedure Button4Click(Sender: TObject);

```
procedure Button5Click(Sender: TObject);
procedure Button8Click(Sender: TObject);
procedure Button3Click(Sender: TObject);
procedure Button6Click(Sender: TObject);
procedure Button10Click(Sender: TObject);
procedure Button11Click(Sender: TObject);
private
  { Private declarations }
public
  { Public declarations }
end;

var
  frm_main: Tfrm_main;

implementation

uses Unit2, Unit5, Unit8, Unit11, Unit22, Unit14, Unit15, Unit16, Unit40,
  Unit27, unit26, Unit28;

{$R *.dfm}

procedure Tfrm_main.Button7Click(Sender: TObject);
begin
  Application.Terminate;
end;

procedure Tfrm_main.Button1Click(Sender: TObject);
begin
  if level='Admin' then
```

```
begin
 frm_std.show;
 frm_main.hide;
end
else
messagedlg('You are not an accessible person to this part!!',mtinformation,[mbok],0);

end;
procedure Tfrm_main.Button2Click(Sender: TObject);
begin
if level='Admin' then
begin
 frm_teacher.show;
 frm_main.Hide;
end
else
messagedlg('You are not an accessible person to this part!!',mtinformation,[mbok],0);
end;
procedure Tfrm_main.Button4Click(Sender: TObject);
begin
if level='Admin' then
begin
 frm_course.show;
 frm_main.hide;
end
else
```

```
messagedlg('You are not an accessible person to this part!!',mtinformation,[mbok],0);
end;

procedure Tfrm_main.Button5Click(Sender: TObject);
begin
if level='Admin' then
begin
frm_dept.show;
frm_main.Hide;
end
else
messagedlg('You are not an accessible person to this part!!',mtinformation,[mbok],0);
end;

procedure Tfrm_main.Button8Click(Sender: TObject);
begin
if (level='Advisor') or (level='Admin') then
begin
frm_main.Hide;
frm_beginsemester.show;
end
else
messagedlg('You are not an accessible person to this part!!',mtinformation,[mbok],0);
end;

procedure Tfrm_main.Button3Click(Sender: TObject);
begin
if (level='Admin') or (level='User') then
begin
```

```
frm_trans.show;
frm_main.Hide;
end
else
MessageDlg('You are not an accessible person to this part!!',mtinformation,[mbok],0);
end;
procedure Tfrm_main.Button6Click(Sender: TObject);
begin
if (level='Admin') or (level='User') then
begin
frm_main.show;
frm_main.Hide;
end
else
MessageDlg('You are not an accessible person to this part!!',mtinformation,[mbok],0);
end;
procedure Tfrm_main.Button10Click(Sender: TObject);
begin
frm_about.show;
frm_main.hide;
end;
procedure Tfrm_main.Button11Click(Sender: TObject);
begin
frm_login.show;
frm_main.Hide;
end;
```

```
unit

UNIT 2
unit Unit2;

interface

uses

Windows, Messages, SysUtils, Variants, Classes, Graphics, Controls, Forms,

Dialogs, Buttons, jpeg, ExtCtrls;

type

Tfrm_std = class(TForm)

SpeedButton1: TSpeedButton;

SpeedButton3: TSpeedButton;

SpeedButton2: TSpeedButton;

procedure FormClose(Sender: TObject; var Action: TCloseAction);

procedure SpeedButton3Click(Sender: TObject);

procedure SpeedButton1Click(Sender: TObject);

procedure SpeedButton2Click(Sender: TObject);

private

{ Private declarations }

public

{ Public declarations }

end;

var

frm_std: Tfrm_std;

implementation

uses Unit1, Unit3, Unit4, unit26;

{$R *.dfm}

procedure Tfrm_std.FormClose(Sender: TObject; var Action: TCloseAction);
```

```
begin
 frm_main.show;
end;

procedure Tfrm_std.SpeedButton3Click(Sender: TObject);
begin
 frm_std.Close;
end;


procedure Tfrm_std.SpeedButton1Click(Sender: TObject);
begin
if level='Admin' then
 begin
 frm_addstd.show;
 frm_std.Hide;
 end;
end;

procedure Tfrm_std.SpeedButton2Click(Sender: TObject);
begin
if level='Admin' then
 begin
 frm_updstd.Show;
 frm_std.Hide;
 end;
end;
end.
```

**UNIT 3**

```
unit Unit3;
```

```
interface

uses
  Windows, Messages, SysUtils, Variants, Classes, Graphics, Controls, Forms,
  Dialogs, StdCtrls, Mask, DBCtrls, ExtCtrls, jpeg, strutils, dateutils,
  ComCtrls;

type
  Tfrm_addstd = class(TForm)
    Label1: TLabel;
    Label2: TLabel;
    Label3: TLabel;
    Label4: TLabel;
    Label5: TLabel;
    Label6: TLabel;
    Label7: TLabel;
    Label8: TLabel;
    Label9: TLabel;
    Label10: TLabel;
    Label11: TLabel;
    Label12: TLabel;
    Label13: TLabel;
    Label14: TLabel;
    Label15: TLabel;
    Button1: TButton;
    Button2: TButton;
    Label16: TLabel;
    Label17: TLabel;
```

Edit1: TEdit;

Edit2: TEdit;

Edit3: TEdit;

Edit4: TEdit;

Edit5: TEdit;

Edit6: TEdit;

Edit7: TEdit;

Edit8: TEdit;

Edit9: TEdit;

Edit10: TEdit;

Edit11: TEdit;

Edit12: TEdit;

Edit13: TEdit;

Memo1: TMemo;

ComboBox1: TComboBox;

Button3: TButton;

Button4: TButton;

Button5: TButton;

Label18: TLabel;

Label19: TLabel;

Label20: TLabel;

Label21: TLabel;

Label22: TLabel;

ComboBox2: TComboBox;

DateTimePicker1: TDateTimePicker;

StatusBar1: TStatusBar;

```
    procedure Button1Click(Sender: TObject);
    procedure Button2Click(Sender: TObject);
    procedure FormShow(Sender: TObject);
    procedure FormClose(Sender: TObject; var Action: TCloseAction);
    procedure Edit1KeyPress(Sender: TObject; var Key: Char);
    procedure Button5Click(Sender: TObject);
    procedure Button4Click(Sender: TObject);
    procedure Button3Click(Sender: TObject);
    procedure Edit2KeyPress(Sender: TObject; var Key: Char);
    procedure Memo1KeyPress(Sender: TObject; var Key: Char);
    procedure ComboBox2KeyPress(Sender: TObject; var Key: Char);
    procedure Button2MouseMove(Sender: TObject; Shift: TShiftState; X,
      Y: Integer);
    procedure FormMouseMove(Sender: TObject; Shift: TShiftState; X,
      Y: Integer);
  private
    { Private declarations }
  public
    { Public declarations }
  end;
var
  frm_addstd: Tfrm_addstd;
implementation
uses unit1, Unit2, unit26;
{$R *.dfm}
procedure Tfrm_addstd.Button1Click(Sender: TObject);
```

```
begin
  frm_std.show;
  frm_addstd.Close;
end;

procedure Tfrm_addstd.Button2Click(Sender: TObject);
begin
  frm_main.show;
  frm_addstd.hide;
end;

procedure clearing(Sender:Tobject);
begin
 frm_addstd.Edit1.Clear;
 frm_addstd.Edit2.Clear;
 frm_addstd.Edit3.Clear;
 frm_addstd.Edit4.Clear;
 frm_addstd.Edit5.Clear;
 frm_addstd.Edit6.Clear;
 frm_addstd.Edit7.Clear;
 frm_addstd.Edit8.Clear;
 frm_addstd.Edit9.Clear;
 frm_addstd.Edit10.Clear;
 frm_addstd.Edit11.Clear;
 frm_addstd.Edit12.Clear;
 frm_addstd.Edit13.Clear;
 frm_addstd.combobox2.Clear;
 frm_addstd.combobox2.Items.Clear;
```

```
frm_addstd.Memo1.Clear;
frm_addstd.combobox1.Clear;
frm_addstd.combobox1.Items.Clear;
end;

procedure invisibility(Sender:Tobject);
begin
frm_addstd.Edit2.Visible:=false;
frm_addstd.Edit3.Visible:=false;
frm_addstd.Edit4.Visible:=false;
frm_addstd.Edit5.Visible:=false;
frm_addstd.Edit6.Visible:=false;
frm_addstd.Edit7.Visible:=false;
frm_addstd.Edit8.Visible:=false;
frm_addstd.Edit9.Visible:=false;
frm_addstd.Edit10.Visible:=false;
frm_addstd.Edit11.Visible:=false;
frm_addstd.Edit12.Visible:=false;
frm_addstd.Edit13.Visible:=false;
frm_addstd.DateTimePicker1.Visible:=false;
frm_addstd.combobox2.Visible:=false;
frm_addstd.memo1.Visible:=false;
frm_addstd.combobox1.Visible:=false;
frm_addstd.button3.Visible:=false;
frm_addstd.button4.Visible:=false;
end;

procedure visibility(Sender:Tobject);
```

```
begin
  frm_addstd.Edit2.Visible:=true;
  frm_addstd.Edit3.Visible:=true;
  frm_addstd.Edit4.Visible:=true;
  frm_addstd.Edit5.Visible:=true;
  frm_addstd.Edit6.Visible:=true;
  frm_addstd.Edit7.Visible:=true;
  frm_addstd.Edit8.Visible:=true;
  frm_addstd.Edit9.Visible:=true;
  frm_addstd.Edit10.Visible:=true;
  frm_addstd.Edit11.Visible:=true;
  frm_addstd.Edit12.Visible:=true;
  frm_addstd.Edit13.Visible:=true;
  frm_addstd.DateTimePicker1.Visible:=true;
  frm_addstd.memo1.Visible:=true;
  frm_addstd.combobox1.Visible:=true;
  frm_addstd.combobox2.Visible:=true;
  frm_addstd.button3.Visible:=true;
  frm_addstd.button4.Visible:=true;
end;

procedure enabling(Sender:Tobject);
begin
  frm_addstd.edit2.enabled:=true;
  frm_addstd.edit3.enabled:=true;
  frm_addstd.edit4.enabled:=true;
  frm_addstd.edit5.enabled:=true;
```

frm_addstd.edit6.enabled:=true;

frm_addstd.edit7.enabled:=true;

frm_addstd.edit8.enabled:=true;

frm_addstd.edit9.enabled:=true;

frm_addstd.edit10.enabled:=true;

frm_addstd.edit11.enabled:=true;

frm_addstd.edit12.enabled:=true;

frm_addstd.edit13.enabled:=true;

frm_addstd.DateTimePicker1.enabled:=true;

frm_addstd.combobox1.enabled:=true;

frm_addstd.combobox2.enabled:=true;

frm_addstd.memo1.enabled:=true;

frm_addstd.button3.enabled:=true;

frm_addstd.button4.enabled:=true;

end;

procedure disabling(Sender:Tobject);

begin

frm_addstd.edit2.enabled:=false;

frm_addstd.edit3.enabled:=false;

frm_addstd.edit4.enabled:=false;

frm_addstd.edit5.enabled:=false;

frm_addstd.edit6.enabled:=false;

frm_addstd.edit7.enabled:=false;

frm_addstd.edit8.enabled:=false;

frm_addstd.edit9.enabled:=false;

```
frm_addstd.edit10.enabled:=false;
frm_addstd.edit11.enabled:=false;
frm_addstd.edit12.enabled:=false;
frm_addstd.edit13.enabled:=false;
frm_addstd.edit13.enabled:=false;
frm_addstd.DateTimePicker1.enabled:=false;
frm_addstd.combobox1.enabled:=false;
frm_addstd.combobox2.enabled:=false;
frm_addstd.memo1.enabled:=false;
frm_addstd.button3.enabled:=false;
frm_addstd.button4.enabled:=false;
end;

procedure Tfrm_addstd.FormShow(Sender: TObject);
begin
 edit1.ReadOnly:=false;
 clearing(sender);
 edit1.SetFocus;
 disabling(sender);
 invisibility(sender);
 frm_addstd.Position:=poscreencenter;
end;


procedure Tfrm_addstd.FormClose(Sender: TObject; var Action: TCloseAction);
begin
edit1.ReadOnly:=false;
```

```
disabling(sender);
frm_std.Show;
end;

procedure Tfrm_addstd.Edit1KeyPress(Sender: TObject; var Key: Char);
begin
if not((key in ['0'..'9'])or(key=#8) or (key=#13)) then
begin
key:=#0;
messagedlg('Please Enter a Numeric Character ',mtwarning,[mbok],0);
end;
if key=#13 then
button5.Click;
end;

procedure Tfrm_addstd.Button5Click(Sender: TObject);
begin
if edit1.text<>'' then
begin
ShortDateFormat := 'yyyy';

if Ansileftstr(edit1.Text,4)=datetostr(date) then
begin
 if length(edit1.Text)=8 then
 begin
  with frm_main.ADOquery1 do
```

```
begin
 sql.clear;
 sql.Add('Select StudentId from student where StudentId="'+edit1.text+'"');
 open;
end;
if frm_main.DataSource7.DataSet.RecordCount=0 then
begin
 shortdateformat:='m/d/yyyy';
 enabling(sender);
 visibility(sender);
 edit2.SetFocus;
 combobox2.Items.Add('Male');
 combobox2.Items.Add('Female');
 edit1.ReadOnly:=true;
 with frm_main.ADOquery1 do
 begin
  sql.clear;
  sql.Add('Select DepartmentId from Department');
  open;
 end;
frm_main.datasource7.DataSet.First;
while not frm_main.datasource7.DataSet.Eof do
begin
 combobox1.Items.Add(frm_main.datasource7.DataSet.FieldValues['DepartmentId']);
 frm_main.DataSource7.DataSet.Next;
end;
```

```
epicker1.Date:=strtodate('01/01/1985');

edgedlg('This Student Number Is Already Recorded !',mtwarning,[mbok],0);
.SetFocus;

edgedlg('Student Number Must Be 8 Digits !',mtwarning,[mbok],0);
.SetFocus;

edgedlg('First 4 digit must be same with current year !',mtwarning,[mbok],0);
.Clear;
.SetFocus;

edgedlg('Please Enter Number For New Student ! ',mtwarning,[mbok],0);
.Clear;
.SetFocus;
```

```
end;

procedure Tfrm_addstd.Button4Click(Sender: TObject);
begin
clearing(sender);
disabling(sender);
visibility(sender);
edit1.SetFocus;
edit1.Clear;
edit1.ReadOnly:=false;
end;

procedure Tfrm_addstd.Button3Click(Sender: TObject);
begin
if(edit2.Text<>'')and(edit3.Text<>'')and(edit8.Text<>'')and(combobox1.Text<>'')) then
begin
with frm_main.DataSource1 do
begin
edit1.ReadOnly:=false;
dataset.Edit;
dataset.Append;
dataset.FieldValues['StudentId']:=edit1.Text;
if combobox1.text<>'' then
dataset.FieldValues['DepartmentId']:=combobox1.Text;
dataset.FieldValues['name']:=edit2.Text;
```

```
dataset.FieldValues['surname']:=edit3.Text;
if combobox2.text<>'' then
dataset.FieldValues['sex']:=combobox2.Text;
if memo1.Text<>'' then
dataset.FieldValues['Address']:=memo1.Text;
if edit7.text<>'' then
dataset.FieldValues['City']:=edit7.Text;
if edit8.text<>'' then
dataset.FieldValues['Country']:=edit8.Text;
if edit9.text<>'' then
dataset.FieldValues['postcode']:=edit9.Text;
if edit11.text<>'' then
dataset.FieldValues['Gsm']:=edit11.Text;
if edit6.text<>'' then
dataset.FieldValues['FatherName']:=edit6.Text;
if edit5.text<>'' then
dataset.FieldValues['MotherName']:=edit5.Text;
if edit4.text<>'' then
dataset.FieldValues['BirthPlace']:=edit4.Text;

dataset.FieldValues['BirthDate']:=datetimepicker1.Date;
if edit13.text<>'' then
dataset.FieldValues['OsymNo']:=edit13.Text;
if edit12.text<>'' then
dataset.FieldValues['Email']:=edit12.Text;
```

```
FieldValues['RegisterDate']:=Date;

text<>'' then
FieldValues['Phone']:=edit10.Text;

FieldValues['ProcessUser']:=username;

Post;
dlg('The student recorded successfully !',mtwarning,[mbok],0);
(sender);
(sender);
ility(sender);
SetFocus;


dlg('Please Fill All The Required Fields !',mtwarning,[mbok],0);
SetFocus;


procedure Tfrm_addstd.Edit2KeyPress(Sender: TObject; var Key: Char);
```

```
(key) in ['A'..'Z'])or(key=#8) or (key=#13)) then

;

(Please Enter a Character ',mtwarning,[mbok],0);

(key);

Tfrm_addstd.Memo1KeyPress(Sender: TObject; var Key: Char);

(key);

Tfrm_addstd.ComboBox2KeyPress(Sender: TObject; var Key: Char);

;

Tfrm_addstd.Button2MouseMove(Sender: TObject; Shift: TShiftState;
: Integer);

bar1.SimpleText:=(Sender as Tbutton).Hint;
```

cedure Tfrm_addstd.FormMouseMove(Sender: TObject; Shift: TShiftState; X,

Y: Integer);

egin

usbar1.SimpleText:='';

nd;

nd.

**NIT 4**

nit Unit4;

erface

es

Windows, Messages, SysUtils, Variants, Classes, Graphics, Controls, Forms,

Dialogs, StdCtrls, DBCtrls, ExtCtrls, Mask, jpeg, ComCtrls;

pe

Tfrm_updstd = class(TForm)

Button1: TButton;

Button2: TButton;

Label1: TLabel;

Label2: TLabel;

Label3: TLabel;

Label4: TLabel;

Label5: TLabel;

Label6: TLabel;

Label7: TLabel;

Label8: TLabel;

Label9: TLabel;

Label10: TLabel;

Label11: TLabel;

Label12: TLabel;

Label13: TLabel;

Label14: TLabel;

Label15: TLabel;

Label16: TLabel;

Label17: TLabel;

Label18: TLabel;

Label19: TLabel;

Label20: TLabel;

Label21: TLabel;

Label22: TLabel;

Edit1: TEdit;

Edit2: TEdit;

Edit3: TEdit;

Edit4: TEdit;

Edit5: TEdit;

Edit6: TEdit;

Edit7: TEdit;

Edit8: TEdit;

Edit9: TEdit;

Edit10: TEdit;

Edit11: TEdit;

```
    Edit2: TEdit;
    Edit1: TEdit;
    Memo1: TMemo;
    ComboBox1: TComboBox;
    Button3: TButton;
    Button4: TButton;
    Button5: TButton;
    ComboBox2: TComboBox;
    DateTimePicker1: TDateTimePicker;
    StatusBar1: TStatusBar;
    Label3: TLabel;
    Label4: TLabel;
    Label5: TLabel;
    CheckBox1: TCheckBox;
    Label6: TLabel;
    procedure Button1Click(Sender: TObject);
    procedure Button2Click(Sender: TObject);
    procedure FormClose(Sender: TObject; var Action: TCloseAction);
    procedure FormShow(Sender: TObject);
    procedure Button5Click(Sender: TObject);
    procedure Button4Click(Sender: TObject);
    procedure Button3Click(Sender: TObject);
    procedure Edit2KeyPress(Sender: TObject; var Key: Char);
    procedure Edit1KeyPress(Sender: TObject; var Key: Char);
    procedure Button3MouseMove(Sender: TObject; Shift: TShiftState; X,
      Y: Integer);
```

MouseMove(Sender: TObject; Shift: TShiftState; X,

KeyPress(Sender: TObject; var Key: Char);

declarations }

declarations }

Tfrm_updstd;

Unit2, unit26, unit3;

clearing(Sender:Tobject);

frm_updstd.Edit1.Clear;

frm_updstd.Edit2.Clear;

frm_updstd.Edit3.Clear;

frm_updstd.Edit4.Clear;

frm_updstd.Edit5.Clear;

frm_updstd.Edit6.Clear;

frm_updstd.Edit7.Clear;

frm_updstd.Edit8.Clear;

frm_updstd.Edit9.Clear;

frm_updstd.Edit10.Clear;

frm_updstd.Edit11.Clear;

frm_updstd.Edit12.Clear;

frm_updstd.Edit13.Clear;

frm_updstd.combobox2.Clear;

frm_updstd.combobox2.Items.Clear;

frm_updstd.Memo1.Clear;

frm_updstd.combobox1.Clear;

frm_updstd.combobox1.Items.Clear;

frm_updstd.Label23.Caption:='';

frm_updstd.Label24.Caption:='';

frm_updstd.CheckBox1.Checked:=false;

frm_updstd.Label26.Caption:='';

end;

procedure invisibility(Sender:Tobject);

begin

frm_updstd.Edit2.Visible:=false;

frm_updstd.Edit3.Visible:=false;

frm_updstd.Edit4.Visible:=false;

frm_updstd.Edit5.Visible:=false;

frm_updstd.Edit6.Visible:=false;

frm_updstd.Edit7.Visible:=false;

frm_updstd.Edit8.Visible:=false;

frm_updstd.Edit9.Visible:=false;

frm_updstd.Edit10.Visible:=false;

frm_updstd.Edit11.Visible:=false;

frm_updstd.Edit12.Visible:=false;

frm_updstd.Edit13.Visible:=false;

frm_updstd.DateTimePicker1.Visible:=false;

frm_updstd.combobox2.Visible:=false;

frm_updstd.memo1.Visible:=false;

frm_updstd.combobox1.Visible:=false;

frm_updstd.button3.Visible:=false;

frm_updstd.button4.Visible:=false;

frm_updstd.CheckBox1.Visible:=false;

end;

procedure visibility(Sender:Tobject);

begin

frm_updstd.Edit2.Visible:=true;

frm_updstd.Edit3.Visible:=true;

frm_updstd.Edit4.Visible:=true;

frm_updstd.Edit5.Visible:=true;

frm_updstd.Edit6.Visible:=true;

frm_updstd.Edit7.Visible:=true;

frm_updstd.Edit8.Visible:=true;

frm_updstd.Edit9.Visible:=true;

frm_updstd.Edit10.Visible:=true;

frm_updstd.Edit11.Visible:=true;

```
.Edit12.Visible:=true;
.Edit13.Visible:=true;
.DateTimePicker1.Visible:=true;
.memo1.Visible:=true;
.combobox1.Visible:=true;
.combobox2.Visible:=true;
.button3.Visible:=true;
.button4.Visible:=true;
.CheckBox1.Visible:=true;


enabling(Sender:Tobject);

.edit2.enabled:=true;
.edit3.enabled:=true;
.edit4.enabled:=true;
.edit5.enabled:=true;
.edit6.enabled:=true;
.edit7.enabled:=true;
.edit8.enabled:=true;
.edit9.enabled:=true;
.edit10.enabled:=true;
.edit11.enabled:=true;
.edit12.enabled:=true;
.edit13.enabled:=true;
```

frm_updstd.DateTimePicker1.enabled:=true;

frm_updstd.combobox1.enabled:=true;

frm_updstd.combobox2.enabled:=true;

frm_updstd.memo1.enabled:=true;

frm_updstd.button3.enabled:=true;

frm_updstd.button4.enabled:=true;

frm_updstd.CheckBox1.Enabled:=true;

end;

procedure disabling(Sender:Tobject);

begin

frm_updstd.edit2.enabled:=false;

frm_updstd.edit3.enabled:=false;

frm_updstd.edit4.enabled:=false;

frm_updstd.edit5.enabled:=false;

frm_updstd.edit6.enabled:=false;

frm_updstd.edit7.enabled:=false;

frm_updstd.edit8.enabled:=false;

frm_updstd.edit9.enabled:=false;

frm_updstd.edit10.enabled:=false;

frm_updstd.edit11.enabled:=false;

frm_updstd.edit12.enabled:=false;

frm_updstd.edit13.enabled:=false;

frm_updstd.edit13.enabled:=false;

frm_updstd.DateTimePicker1.enabled:=false;

```
frm_updstd.combobox1.enabled:=false;
frm_updstd.combobox2.enabled:=false;
frm_updstd.memo1.enabled:=false;
frm_updstd.button3.enabled:=false;
frm_updstd.button4.enabled:=false;
frm_updstd.CheckBox1.Enabled:=false;
end;

procedure Tfrm_updstd.Button1Click(Sender: TObject);
begin
frm_std.show;
frm_updstd.hide;
end;

procedure Tfrm_updstd.Button2Click(Sender: TObject);
begin
frm_main.Show;
frm_updstd.hide
end;

procedure Tfrm_updstd.FormClose(Sender: TObject; var Action: TCloseAction);
begin
edit1.ReadOnly:=false;
```

sender);

Show;

Tfrm_updstd.FormShow(Sender: TObject);

ReadOnly:=false;

sender);

SetFocus;

sender);

sender);

Tfrm_updstd.Button5Click(Sender: TObject);

text<>'' then

ShortDateFormat := 'm/dd/yyyy';

length(edit1.Text)=8 then

with frm_main.ADOquery1 do

sql.clear;

sql.Add('Select * from student where StudentId='''+edit1.text+'''');

open;

```
end;

if frm_main.DataSource7.DataSet.RecordCount<>0 then
begin

editing(sender);

validity(sender);

id.SetFocus;

combobox2.Items.Add('Male');

combobox2.Items.Add('Female');

id.ReadOnly:=true;

with frm_main.ADOquery1 do

begin

sql.clear;

sql.Add('Select DepartmentId from Department');

open;

end;


frm_main.datasource7.DataSet.First;

while not frm_main.datasource7.DataSet.Eof do

begin

combobox1.Items.Add(frm_main.datasource7.DataSet.FieldValues['DepartmentId']);

frm_main.DataSource7.DataSet.Next;

end;


with frm_main.ADOquery1 do

begin

sql.clear;
```

```
Select * from student where StudentId="'+edit1.text+'");

frm_main.DataSource7.DataSet.FieldValues['name']<>null then
Text:=frm_main.DataSource7.DataSet.FieldValues['name'];
frm_main.DataSource7.DataSet.FieldValues['surname']<>null then
Text:=frm_main.DataSource7.DataSet.FieldValues['surname'];
frm_main.DataSource7.DataSet.FieldValues['BirthPlace']<>null then
Text:=frm_main.DataSource7.DataSet.FieldValues['BirthPlace'];
frm_main.DataSource7.DataSet.FieldValues['MotherName']<>null then
Text:=frm_main.DataSource7.DataSet.FieldValues['MotherName'];
frm_main.DataSource7.DataSet.FieldValues['fathername']<>null  then
Text:=frm_main.DataSource7.DataSet.FieldValues['fathername'];
frm_main.DataSource7.DataSet.FieldValues['city']<>null then
Text:=frm_main.DataSource7.DataSet.FieldValues['city'];
frm_main.DataSource7.DataSet.FieldValues['country']<>null then
Text:=frm_main.DataSource7.DataSet.FieldValues['country'];
frm_main.DataSource7.DataSet.FieldValues['postcode']<>null then
Text:=frm_main.DataSource7.DataSet.FieldValues['postcode'];
frm_main.DataSource7.DataSet.FieldValues['phone']<>null then
Text:=frm_main.DataSource7.DataSet.FieldValues['phone'];
frm_main.DataSource7.DataSet.FieldValues['Gsm']<>null then
Text:=frm_main.DataSource7.DataSet.FieldValues['Gsm'];
frm_main.DataSource7.DataSet.FieldValues['Email']<>null then
Text:=frm_main.DataSource7.DataSet.FieldValues['Email'];
```

```
if frm_main.DataSource7.DataSet.FieldValues['OsymNo']<>null then
edit13.Text:=frm_main.DataSource7.DataSet.FieldValues['OsymNo'];
if frm_main.DataSource7.DataSet.FieldValues['DepartmentId']<>null then
combobox1.Text:=frm_main.DataSource7.DataSet.FieldValues['DepartmentId'];
if frm_main.DataSource7.DataSet.FieldValues['sex']<>null then
combobox2.Text:=frm_main.DataSource7.DataSet.FieldValues['sex'];
if frm_main.DataSource7.DataSet.FieldValues['BirthDate']<>null then
datetimepicker1.Date:=frm_main.DataSource7.DataSet.FieldValues['BirthDate'];
if frm_main.DataSource7.DataSet.FieldValues['Address']<>null then
memo1.Text:=frm_main.DataSource7.DataSet.FieldValues['Address'];
if frm_main.DataSource7.DataSet.FieldValues['IsGraduated']=true then
begin
checkbox1.Checked:=true;
label26.Caption:='The Student has been graduated At
'+DateToStr(frm_main.DataSource7.DataSet.FieldValues['GraduationDate']);
end
else
checkbox1.Checked:=false;
//label23.Caption:='The Student has been graduated At
'+DateToStr(frm_main.DataSource7.DataSet.FieldValues['GraduationDate']);
label23.Caption:='The Student Is Registered At
'+DateToStr(frm_main.DataSource7.DataSet.FieldValues['RegisterDate'])+' By
'+frm_main.DataSource7.DataSet.FieldValues['ProcessUser'];
if frm_main.DataSource7.DataSet.FieldValues['UpdateUser']<>null then
label24.Caption:='The Student Record Was last Updated At
'+DateToStr(frm_main.DataSource7.DataSet.FieldValues['UpdateDate'])+' By
'+frm_main.DataSource7.DataSet.FieldValues['UpdateUser'];
end
```

```
essagedlg('Student Number Is Not Found !',mtwarning,[mbok],0);
dit1.SetFocus;
nd;

nd;

egin

essagedlg('Student Number Must Be 8 Digits !',mtwarning,[mbok],0);
dit1.SetFocus;
nd;

nd;

egin

essagedlg('Please Enter Number For Student To Update ! ',mtwarning,[mbok],0);
dit1.Clear;
dit1.SetFocus;
nd;

nd;

rocedure Tfrm_updstd.Button4Click(Sender: TObject);
```

```
(sender);
(sender);
ility(sender);
SetFocus;
Clear;
ReadOnly:=false;


Tfrm_updstd.Button3Click(Sender: TObject);

Text<>")and(edit3.Text<>")and(edit8.Text<>")and(combobox1.Text<>")) then

frm_main.ADOquery1 do
 begin
 sql.clear;
 sql.Add('Select * from student where StudentId="'+edit1.text+'"');
 open;
 end;
frm_main.DataSource7 do
begin
edit1.ReadOnly:=false;
dataset.Edit;

if combobox1.text<>'' then
dataset.FieldValues['DepartmentId']:=combobox1.Text;
```

```
dataset.FieldValues['name']:=edit2.Text;
dataset.FieldValues['surname']:=edit3.Text;
if combobox2.text<>'' then
dataset.FieldValues['sex']:=combobox2.Text;
if memo1.Text<>'' then
dataset.FieldValues['Address']:=memo1.Text;
if edit7.text<>'' then
dataset.FieldValues['City']:=edit7.Text;
if edit8.text<>'' then
dataset.FieldValues['Country']:=edit8.Text;
if edit9.text<>'' then
dataset.FieldValues['postcode']:=edit9.Text;
if edit11.text<>'' then
dataset.FieldValues['Gsm']:=edit11.Text;
if edit6.text<>'' then
dataset.FieldValues['FatherName']:=edit6.Text;
if edit5.text<>'' then
dataset.FieldValues['MotherName']:=edit5.Text;
if edit4.text<>'' then
dataset.FieldValues['BirthPlace']:=edit4.Text;


dataset.FieldValues['BirthDate']:=datetimepicker1.Date;
if edit13.text<>'' then
dataset.FieldValues['OsymNo']:=edit13.Text;
if edit12.text<>'' then
dataset.FieldValues['Email']:=edit12.Text;
```

```
checkbox1.Checked then
begin
dataset.FieldValues['IsGraduated']:=true;
dataset.FieldValues['Graduationdate']:=date;

end;


dataset.FieldValues['UpdateDate']:=Date;


edit10.text<>'' then
dataset.FieldValues['Phone']:=edit10.Text;
datetimepicker1.Enabled:=True;
dataset.FieldValues['UpdateUser']:=username;

dataset.Post;
messagedlg('The student recorded successfully !',mtwarning,[mbok],0);
clearing(sender);
disabling(sender);
invisibility(sender);
edit1.SetFocus;
end;


end
else
begin
messagedlg('Please Fill All The Required Fields !',mtwarning,[mbok],0);
```

```
Edit2.SetFocus;

end;

end;

procedure Tfrm_updstd.Edit2KeyPress(Sender: TObject; var Key: Char);

begin

if (upcase(key) in ['A'..'Z'])or(key=#8) or (key=#13)) then

begin

key:=#0;

messagedlg('Please Enter a Character ',mtwarning,[mbok],0);

end

else

key:=upcase(key);

end;


procedure Tfrm_updstd.Edit1KeyPress(Sender: TObject; var Key: Char);

begin

if (key in ['0'..'9'])or(key=#8) or (key=#13)) then

begin

key:=#0;

messagedlg('Please Enter a Numeric Character ',mtwarning,[mbok],0);

end;

if key=#13 then

button5.Click;

end;
```

procedure Tfrm_updstd.Button3MouseMove(Sender: TObject; Shift: TShiftState;

Y: Integer);

begin

StatusBar1.SimpleText:=(Sender as Tbutton).Hint;

end;

procedure Tfrm_updstd.FormMouseMove(Sender: TObject; Shift: TShiftState; X,

Y: Integer);

begin

StatusBar1.SimpleText:='';

end;

procedure Tfrm_updstd.Memo1KeyPress(Sender: TObject; var Key: Char);

begin

key:=upcase(key);

end;

end.

UNIT 5

unit Unit5;

interface

uses

Windows, Messages, SysUtils, Variants, Classes, Graphics, Controls, Forms,

Buttons, jpeg, ExtCtrls;

type
Tfrm_teacher = class(TForm)
SpeedButton3: TSpeedButton;
SpeedButton2: TSpeedButton;
SpeedButton1: TSpeedButton;
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
frm_teacher: Tfrm_teacher;

implementation

uses Unit1, Unit6, Unit7;

{$R *.dfm}

```
procedure Tfrm_teacher.SpeedButton3Click(Sender: TObject);
begin
  frm_main.show;
  frm_teacher.Close;
end;

procedure Tfrm_teacher.FormClose(Sender: TObject;
  var Action: TCloseAction);
begin
  frm_main.show;
end;

procedure Tfrm_teacher.SpeedButton1Click(Sender: TObject);
begin
  frm_addteacher.show;
  frm_teacher.Hide;
end;

procedure Tfrm_teacher.SpeedButton2Click(Sender: TObject);
begin
  frm_updteacher.show;
  frm_teacher.Hide;
end;

end.
```

Messages, SysUtils, Variants, Classes, Graphics, Controls, Forms,

StdCtrls, ExtCtrls, DBCtrls, Mask, ComCtrls;

acher = class(TForm)

TLabel;

TLabel;

TLabel;

TLabel;

TLabel;

TLabel;

TLabel;

TLabel;

TLabel;

TLabel;

TButton;

TButton;

TLabel;

TLabel;

TEdit;

TEdit;

TEdit;

TEdit;

TEdit;

TEdit;

TMemo;

ComboBox1: TComboBox;

ComboBox2: TComboBox;

TButton;

TButton;

DateTimePicker1: TDateTimePicker;

ComboBox3: TComboBox;

TLabel;

TLabel;

TLabel;

TLabel;

TLabel;

procedure FormClose(Sender: TObject; var Action: TCloseAction);

procedure Button2Click(Sender: TObject);

procedure Button1Click(Sender: TObject);

procedure FormShow(Sender: TObject);

procedure Button4Click(Sender: TObject);

procedure Button3Click(Sender: TObject);

procedure Edit2KeyPress(Sender: TObject; var Key: Char);

procedure Memo1KeyPress(Sender: TObject; var Key: Char);

procedure ComboBox3KeyDown(Sender: TObject; var Key: Word;

Shift: TShiftState);

declarations }

declarations }

Tfrm_addteacher;

Unit1,unit26;

clearing(Sender:Tobject);

Edit2.Clear;

Edit3.Clear;

Edit4.Clear;

Edit5.Clear;

Edit6.Clear;

DateTimePicker1.Date:=StrToDate('01/01/1970');

combobox2.Clear;

combobox2.Items.Clear;

Memo1.Clear;

combobox1.Clear;

```
[illegible]acher.combobox1.Items.Clear;
[illegible]acher.combobox3.Clear;
[illegible]acher.combobox3.Items.Clear;

end;

[illegible] invisibility(Sender:Tobject);
begin
[illegible]acher.Edit2.Visible:=false;
[illegible]acher.Edit3.Visible:=false;
[illegible]acher.Edit4.Visible:=false;
[illegible]acher.Edit5.Visible:=false;
[illegible]acher.Edit6.Visible:=false;
[illegible]acher.DateTimePicker1.Visible:=false;
[illegible]acher.combobox2.Visible:=false;
[illegible]acher.memo1.Visible:=false;
[illegible]acher.combobox1.Visible:=false;
[illegible]acher.combobox3.visible:=false;

end;

[illegible] visibility(Sender:Tobject);
begin
[illegible]acher.Edit2.Visible:=true;
[illegible]acher.Edit3.Visible:=true;
[illegible]acher.Edit4.Visible:=true;
```

```
...addteacher.Edit5.Visible:=true;
...addteacher.Edit6.Visible:=true;

...addteacher.DateTimePicker1.Visible:=true;
...addteacher.memo1.Visible:=true;
...addteacher.combobox1.Visible:=true;
...addteacher.combobox2.Visible:=true;
...addteacher.combobox3.visible:=true;
end;

procedure enabling(Sender:Tobject);
begin
...addteacher.edit2.enabled:=true;
...addteacher.edit3.enabled:=true;
...addteacher.edit4.enabled:=true;
...addteacher.edit5.enabled:=true;
...addteacher.edit6.enabled:=true;

...addteacher.DateTimePicker1.enabled:=true;
...addteacher.combobox1.enabled:=true;
...addteacher.combobox2.enabled:=true;
...addteacher.combobox3.enabled:=true;
...addteacher.memo1.enabled:=true;
end;

procedure disabling(Sender:Tobject);
```

```
er.edit2.enabled:=false;
er.edit3.enabled:=false;
er.edit4.enabled:=false;
er.edit5.enabled:=false;
er.edit6.enabled:=false;

er.DateTimePicker1.enabled:=false;
er.combobox1.enabled:=false;
er.combobox2.enabled:=false;
er.combobox3.enabled:=false;
er.memo1.enabled:=false;


Tfrm_addteacher.FormClose(Sender: TObject;
: TCloseAction);

.show;


Tfrm_addteacher.Button2Click(Sender: TObject);

show;
.Hide;
```

```
procedure Tfrm_addteacher.Button1Click(Sender: TObject);
begin

frm_teacher.Show;
frm_addteacher.Hide;
end;


procedure Tfrm_addteacher.FormShow(Sender: TObject);
begin
inherited;
edit1.ReadOnly:=true;
edit2.SetFocus;

with frm_main.ADOquery1 do
begin
close;
sql.Add('Select Max(TeacherId) as id from Teacher');
open;
end;

edit1.Text:=IntToStr(frm_main.DataSource7.DataSet.FieldValues['id']+1);
combobox1.Items.Add('Male');
combobox1.Items.Add('Female');
combobox2.Items.Add('Prof.Dr.');
combobox2.Items.Add('Assoc.Prof.Dr.');
combobox2.Items.Add('Asist.Prof.Dr.');
combobox2.Items.Add('Teacher');
with frm_main.ADOquery1 do
```

Select DepartmentId from Department');

source7.DataSet.First;

frm_main.datasource7.DataSet.Eof do

3.Items.Add(frm_main.datasource7.DataSet.FieldValues['DepartmentId']);

DataSource7.DataSet.Next;

Tfrm_addteacher.Button4Click(Sender: TObject);

sender);

Focus;

Tfrm_addteacher.Button3Click(Sender: TObject);

Text<>")and(edit3.Text<>")and(combobox2.Text<>")and(combobox3.Text<>"))

```
with form_main.DataSource3 do
begin
  dataset.Edit;
  dataset.Append;
  dataset.FieldValues['TeacherId']:=edit1.Text;
  if combobox1.text<>'' then
  dataset.FieldValues['Sex']:=combobox1.Text;

  dataset.FieldValues['Tname']:=edit2.Text;

  dataset.FieldValues['Tsurname']:=edit3.Text;

  if combobox2.text<>'' then
  dataset.FieldValues['Title']:=combobox2.Text;

  if combobox3.text<>'' then
  dataset.FieldValues['DepartmentId']:=combobox3.Text;
  if memo1.Text<>'' then
  dataset.FieldValues['Address']:=memo1.Text;
  if edit5.text<>'' then
  dataset.FieldValues['Gsm']:=edit5.Text;
  if edit6.text<>'' then
  dataset.FieldValues['Email']:=edit6.Text;
  if edit4.text<>'' then
  dataset.FieldValues['Phone']:=edit4.Text;
```

```
FieldValues['BirthDate']:=datetimepicker1.Date;

FieldValues['ProcessDate']:=Date;
FieldValues['ProcessUser']:=username;

Post;
dlg('The Teacher recorded successfully !',mtwarning,[mbok],0);
sender);
SetFocus;
ReadOnly:=false;
frm_main.ADOquery2 do

clear;
Add('Select Max(TeacherId) as id from Teacher');


frm_main.DataSource7.DataSet:=frm_main.ADOQuery1;
Text:=IntToStr(frm_main.DataSource8.DataSet.FieldValues['id']+1);
ReadOnly:=true;


dlg('Please Fill All The Required Fields !',mtwarning,[mbok],0);
SetFocus;
```

```
procedure Tfrm_addteacher.Edit2KeyPress(Sender: TObject; var Key: Char);
begin
if (upcase(key) in ['A'..'Z'])or(key=#8) or (key=#13)) then
begin
exit;
messagedlg('Please Enter a Character ',mtwarning,[mbok],0);
end
else
key:=upcase(key);
end;

procedure Tfrm_addteacher.Memo1KeyPress(Sender: TObject; var Key: Char);
begin
key:=upcase(key);
end;

procedure Tfrm_addteacher.ComboBox3KeyDown(Sender: TObject; var Key: Word;
Shift: TShiftState);
begin
if (key=vk_down)or(key=vk_up)) then
key:=vk_up;
end;
```

Messages, SysUtils, Variants, Classes, Graphics, Controls, Forms,

StdCtrls, ExtCtrls, DBCtrls, Mask, ComCtrls;

teacher = class(TForm)

TLabel;

TLabel;

TLabel;

TLabel;

TLabel;

TLabel;

TLabel;

TLabel;

TLabel;

TLabel;

TLabel;

TLabel;

TLabel;

TLabel;

TLabel;

TLabel;

TButton;

TButton;

TEdit;

TEdit;

TEdit;

TEdit;

TEdit;

TEdit;

TMemo;

Box1: TComboBox;

Box2: TComboBox;

TButton;

TButton;

TimePicker1: TDateTimePicker;

Box3: TComboBox;

TLabel;

TLabel;

TEdit;

TEdit;

TButton;

TLabel;

TLabel;

ure FormClose(Sender: TObject; var Action: TCloseAction);

ure Button1Click(Sender: TObject);

```
procedure Button2Click(Sender: TObject);
procedure Edit1KeyPress(Sender: TObject; var Key: Char);
procedure FormShow(Sender: TObject);
procedure Button5Click(Sender: TObject);
procedure Button4Click(Sender: TObject);
procedure Button3Click(Sender: TObject);
procedure Edit2KeyPress(Sender: TObject; var Key: Char);
procedure Memo1KeyPress(Sender: TObject; var Key: Char);
private
  { Private declarations }
public
  { Public declarations }
end;


var
  frm_updteacher: Tfrm_updteacher;

implementation

uses Unit1, Unit5, unit26;

{$R *.dfm}


procedure clearing(Sender:Tobject);
begin
frm_updteacher.Edit1.Clear;
```

```
.Edit2.Clear;
.Edit3.Clear;
.Edit4.Clear;
.Edit5.Clear;
.Edit6.Clear;
.Edit7.Clear;
.Edit8.Clear;
.DateTimePicker1.Date:=StrToDate('01/01/1970');
.combobox2.Clear;
.combobox2.Items.Clear;
.Memo1.Clear;
.combobox1.Clear;
.combobox1.Items.Clear;
.combobox3.Clear;
.combobox3.Items.Clear;
.Label23.Caption:='';
.Label24.Caption:='';


invisibility(Sender:Tobject);


.Edit4.Visible:=false;
.Edit5.Visible:=false;
.Edit6.Visible:=false;
```

```
frm_updteacher.Edit7.Visible:=false;
frm_updteacher.Edit8.Visible:=false;
frm_updteacher.DateTimePicker1.Visible:=false;
frm_updteacher.combobox2.Visible:=false;
frm_updteacher.memo1.Visible:=false;
frm_updteacher.combobox1.Visible:=false;
frm_updteacher.combobox3.visible:=false;
frm_updteacher.Button3.Visible:=false;
frm_updteacher.Button4.Visible:=false;
end;

procedure visibility(Sender:Tobject);
begin

frm_updteacher.Edit4.Visible:=true;
frm_updteacher.Edit5.Visible:=true;
frm_updteacher.Edit6.Visible:=true;
frm_updteacher.Edit7.Visible:=true;
frm_updteacher.Edit8.Visible:=true;
frm_updteacher.DateTimePicker1.Visible:=true;
frm_updteacher.memo1.Visible:=true;
frm_updteacher.combobox1.Visible:=true;
frm_updteacher.combobox2.Visible:=true;
frm_updteacher.combobox3.visible:=true;
frm_updteacher.Button3.Visible:=true;
frm_updteacher.Button4.Visible:=true;
```

```
procedure enabling(Sender:Tobject);
begin

frmteacher.edit4.enabled:=true;
frmteacher.edit5.enabled:=true;
frmteacher.edit6.enabled:=true;
frmteacher.edit7.enabled:=true;
frmteacher.edit8.enabled:=true;
frmteacher.DateTimePicker1.enabled:=true;
frmteacher.combobox1.enabled:=true;
frmteacher.combobox2.enabled:=true;
frmteacher.combobox3.enabled:=true;
frmteacher.memo1.enabled:=true;
end;


procedure disabling(Sender:Tobject);
begin

frmteacher.edit4.enabled:=false;
frmteacher.edit5.enabled:=false;
frmteacher.edit6.enabled:=false;
frmteacher.edit7.enabled:=false;
frmteacher.edit8.enabled:=false;
frmteacher.DateTimePicker1.enabled:=false;
```

```
frm_updteacher.combobox1.enabled:=false;
frm_updteacher.combobox2.enabled:=false;
frm_updteacher.combobox3.enabled:=false;
frm_updteacher.memo1.enabled:=false;
end;

procedure Tfrm_updteacher.FormClose(Sender: TObject; var Action: TCloseAction);
begin
frm_main.Show;
end;

procedure Tfrm_updteacher.Button1Click(Sender: TObject);
begin
frm_teacher.show;
frm_updteacher.Hide;
end;

procedure Tfrm_updteacher.Button2Click(Sender: TObject);
begin
frm_updteacher.Hide;
frm_main.Show;
end;
```

```
_main.ADOquery1 do

;

Select DepartmentId from Department');

;

;

datasource7.DataSet.First;

frm_main.datasource7.DataSet.Eof do

;

box3.Items.Add(frm_main.datasource7.DataSet.FieldValues['DepartmentId']);

DataSource7.DataSet.Next;

;


_main.ADOquery1 do

;

;

Select *  from Teacher where teacherId='+edit1.text);

;

;


_main.DataSource7.DataSet.RecordCount>0 then

;

(sender);

(sender);

_main.DataSource7.DataSet.FieldValues['Tname']<>null then

Text:=frm_main.DataSource7.DataSet.FieldValues['Tname'];
```

```
frm_main.DataSource7.DataSet.FieldValues['Tsurname']<>null  then
Text:=frm_main.DataSource7.DataSet.FieldValues['Tsurname'];
frm_main.DataSource7.DataSet.FieldValues['Phone']<>null then
Text:=frm_main.DataSource7.DataSet.FieldValues['phone'];
frm_main.DataSource7.DataSet.FieldValues['Gsm']<>null then
Text:=frm_main.DataSource7.DataSet.FieldValues['Gsm'];
frm_main.DataSource7.DataSet.FieldValues['Email']<>null then
Text:=frm_main.DataSource7.DataSet.FieldValues['Email'];
frm_main.DataSource7.DataSet.FieldValues['BirthDate']<>null then
timepicker1.Date:=frm_main.DataSource7.DataSet.FieldValues['BirthDate'];
frm_main.DataSource7.DataSet.FieldValues['Sex']<>null then
combobox1.Text:=frm_main.DataSource7.DataSet.FieldValues['Sex'];
frm_main.DataSource7.DataSet.FieldValues['Title']<>null then
combobox2.Text:=frm_main.DataSource7.DataSet.FieldValues['Title'];
frm_main.DataSource7.DataSet.FieldValues['DepartmentId']<>null then
combobox3.Text:=frm_main.DataSource7.DataSet.FieldValues['DepartmentId'];
frm_main.DataSource7.DataSet.FieldValues['Address']<>null then
memo1.Text:=frm_main.DataSource7.DataSet.FieldValues['Address'];


frm_main.DataSource7.DataSet.FieldValues['ProcessUser']<>null then
label23.Caption:='The Teacher Is Recorded At
DateToStr(frm_main.DataSource7.DataSet.FieldValues['ProcessDate'])+'  By
frm_main.DataSource7.DataSet.FieldValues['ProcessUser'];

frm_main.DataSource7.DataSet.FieldValues['UpdateUser']<>null then
label24.Caption:='The Teacher Record Was last Updated At
DateToStr(frm_main.DataSource7.DataSet.FieldValues['UpdateDate'])+'  By
frm_main.DataSource7.DataSet.FieldValues['UpdateUser'];
```

```
SetFocus;
ReadOnly:=true;
ReadOnly:=true;
ReadOnly:=true;

g('Teacher Id is not found!!',mtinformation,[mbok],0);
sender);
SetFocus;


text='')and((edit2.Text<>'')and(edit3.Text<>''))) then

box1.Items.Add('Male');
box1.Items.Add('Female');
box2.Items.Add('Prof.Dr.');
box2.Items.Add('Assoc.Prof.Dr.');
box2.Items.Add('Asist.Prof.Dr.');
box2.Items.Add('Teacher');
frm_main.ADOquery1 do
```

```
Select DepartmentId from Department');

datasource7.DataSet.First;

frm_main.datasource7.DataSet.Eof do

3.Items.Add(frm_main.datasource7.DataSet.FieldValues['DepartmentId']);

DataSource7.DataSet.Next;


_main.ADOquery1 do

Select *  from Teacher where Tname="'+edit2.text+'" and
="+edit3.Text+"");


_main.DataSource7.DataSet.RecordCount>0 then

(sender);

(sender);


Text:=IntToStr(frm_main.DataSource7.DataSet.FieldValues['TeacherId']);

_main.DataSource7.DataSet.FieldValues['Tname']<>null then

Text:=frm_main.DataSource7.DataSet.FieldValues['Tname'];
```

```
frm_main.DataSource7.DataSet.FieldValues['Tsurname']<>null  then
Text:=frm_main.DataSource7.DataSet.FieldValues['Tsurname'];
frm_main.DataSource7.DataSet.FieldValues['Phone']<>null then
Text:=frm_main.DataSource7.DataSet.FieldValues['phone'];
frm_main.DataSource7.DataSet.FieldValues['Gsm']<>null then
Text:=frm_main.DataSource7.DataSet.FieldValues['Gsm'];
frm_main.DataSource7.DataSet.FieldValues['Email']<>null then
Text:=frm_main.DataSource7.DataSet.FieldValues['Email'];
frm_main.DataSource7.DataSet.FieldValues['BirthDate']<>null then
timepicker1.Date:=frm_main.DataSource7.DataSet.FieldValues['BirthDate'];
frm_main.DataSource7.DataSet.FieldValues['Sex']<>null then
combobox1.Text:=frm_main.DataSource7.DataSet.FieldValues['Sex'];
frm_main.DataSource7.DataSet.FieldValues['Title']<>null then
combobox2.Text:=frm_main.DataSource7.DataSet.FieldValues['Title'];
frm_main.DataSource7.DataSet.FieldValues['DepartmentId']<>null then
combobox3.Text:=frm_main.DataSource7.DataSet.FieldValues['DepartmentId'];
frm_main.DataSource7.DataSet.FieldValues['Address']<>null then
memo1.Text:=frm_main.DataSource7.DataSet.FieldValues['Address'];


frm_main.DataSource7.DataSet.FieldValues['ProcessUser']<>null then
label23.Caption:='The Teacher Is Recorded At
DateStr(frm_main.DataSource7.DataSet.FieldValues['ProcessDate'])+'  By
frm_main.DataSource7.DataSet.FieldValues['ProcessUser'];
frm_main.DataSource7.DataSet.FieldValues['UpdateUser']<>null then
label24.Caption:='The Teacher Record Was last Updated At
DateStr(frm_main.DataSource7.DataSet.FieldValues['UpdateDate'])+'  By
frm_main.DataSource7.DataSet.FieldValues['UpdateUser'];
```

```
edit4.SetFocus;
edit1.ReadOnly:=true;
edit2.ReadOnly:=true;
edit3.ReadOnly:=true;
end
else
begin
messagedlg('Teacher Name and Surname are not found!!',mtinformation,[mbok],0);
clearing(sender);
edit1.SetFocus;
edit1.ReadOnly:=FALSE;
edit2.ReadOnly:=FALSE;
edit3.ReadOnly:=FALSE;
end;



end
else
begin
messagedlg('Please Enter Teacher ID or Teacher Name And Surname
information,[mbok],0);
edit1.SetFocus;
clearing(sender);


end;
```

Tfrm_updteacher.Button4Click(Sender: TObject);

sender);

sender);

sender);

Focus;

ReadOnly:=false;

ReadOnly:=false;

ReadOnly:=false;

Tfrm_updteacher.Button3Click(Sender: TObject);

Text<>")and(edit5.Text<>")and(combobox2.Text<>")and(combobox3.Text<>"))

frm_main.ADOquery1 do

clear;

Add('Select * from Teacher where TeacherId='+edit1.text);

_main.DataSource7 do

ReadOnly:=false;

ReadOnly:=false;

ReadOnly:=false;

Edit;

box1.text<>'' then

FieldValues['sex']:=combobox1.Text;

FieldValues['Tname']:=edit4.Text;

FieldValues['Tsurname']:=edit5.Text;

box2.text<>'' then

FieldValues['Title']:=combobox2.Text;

box3.text<>'' then

FieldValues['DepartmentId']:=combobox3.Text;

1.Text<>'' then

FieldValues['Address']:=memo1.Text;

ext<>'' then

FieldValues['Gsm']:=edit7.Text;

ext<>'' then

FieldValues['Email']:=edit8.Text;

FieldValues['BirthDate']:=Datetimepicker1.Date;

ext<>'' then

```
FieldValues['Phone']:=edit6.Text;
picker1.Enabled:=True;
FieldValues['UpdateUser']:=username;
FieldValues['UpdateDate']:=Date;


Post;
MessageDlg('The Teacher Information Recorded successfully !',mtInformation,[mbok],0);
sender);
sender);
ility(sender);
SetFocus;
end;

end;

else

MessageDlg('Please Fill All The Required Fields !',mtwarning,[mbok],0);
SetFocus;
end;
```

Tfrm_updteacher.Edit2KeyPress(Sender: TObject; var Key: Char);

upcase(key) in ['A'..'Z'])or(key=#8) or (key=#13)) then

edlg('Please Enter a Character ',mtwarning,[mbok],0);

case(key);

Tfrm_updteacher.Memo1KeyPress(Sender: TObject; var Key: Char);

case(key);

Messages, SysUtils, Variants, Classes, Graphics, Controls, Forms,

Buttons, ExtCtrls;

```
type
  Tfrm_course = class(TForm)
    SpeedButton1: TSpeedButton;
    SpeedButton2: TSpeedButton;
    SpeedButton4: TSpeedButton;
    procedure FormClose(Sender: TObject; var Action: TCloseAction);
    procedure SpeedButton4Click(Sender: TObject);
    procedure SpeedButton1Click(Sender: TObject);
    procedure SpeedButton2Click(Sender: TObject);
  private
    { Private declarations }
  public
    { Public declarations }
  end;

var
  frm_course: Tfrm_course;

implementation

uses Unit1, Unit9, Unit10;

{$R *.dfm}

procedure Tfrm_course.FormClose(Sender: TObject; var Action: TCloseAction);
begin
```

```
frm_main.show;
end;

procedure Tfrm_course.SpeedButton4Click(Sender: TObject);
begin
frm_course.Close;
end;

procedure Tfrm_course.SpeedButton1Click(Sender: TObject);
begin
frm_addcourse.show;
frm_course.Hide;
end;

procedure Tfrm_course.SpeedButton2Click(Sender: TObject);
begin
frm_updcourse.show;
frm_course.Hide;
end;

end.
UNIT 9
unit Unit9;

interface
```

Messages, SysUtils, Variants, Classes, Graphics, Controls, Forms,

ExtCtrls, DBCtrls, StdCtrls, Mask, jpeg, Grids, DBGrids,

;

addcourse = class(TForm)

TLabel;

TLabel;

TLabel;

TLabel;

TLabel;

TLabel;

TButton;

TButton;

TEdit;

TEdit;

TEdit;

Box1: TComboBox;

Box2: TComboBox;

Box3: TComboBox;

TButton;

TButton;

Bar1: TStatusBar;

TLabel;

TDBGrid;

```
    procedure FormClose(Sender: TObject; var Action: TCloseAction);
    procedure Button1Click(Sender: TObject);
    procedure Button2Click(Sender: TObject);
    procedure FormShow(Sender: TObject);
    procedure ComboBox1Exit(Sender: TObject);
    procedure Button4Click(Sender: TObject);
    procedure Button3Click(Sender: TObject);
  private
    { Private declarations }
  public
    { Public declarations }
  end;

var
  frm_addcourse: Tfrm_addcourse;

implementation

uses Unit8, Unit1, ADODB, unit26;

{$R *.dfm}

procedure clearing(Sender:Tobject);
begin
frm_addcourse.Edit2.Clear;
frm_addcourse.Edit3.Clear;
```

frm_addcourse.Edit1.Clear;

frm_addcourse.combobox2.Clear;

frm_addcourse.combobox2.Items.Clear;

frm_addcourse.combobox1.Clear;

frm_addcourse.combobox1.Items.Clear;

frm_addcourse.combobox3.Clear;

frm_addcourse.combobox3.Items.Clear;

frm_addcourse.combobox3.Items.Add(inttostr(0));

frm_addcourse.combobox3.Items.Add(Inttostr(1));

frm_addcourse.combobox3.Items.Add(Inttostr(2));

frm_addcourse.combobox3.Items.Add(Inttostr(3));

frm_addcourse.combobox3.Items.Add(Inttostr(4));

frm_addcourse.combobox3.Items.Add(Inttostr(5));

frm_addcourse.combobox3.Items.Add(Inttostr(6));

procedure invisibility(Sender:Tobject);

begin

frm_addcourse.Edit2.Visible:=false;

frm_addcourse.Edit3.Visible:=false;

frm_addcourse.Edit1.Visible:=false;

frm_addcourse.combobox2.Visible:=false;

frm_addcourse.combobox1.Visible:=false;

frm_addcourse.combobox3.visible:=false;

end;

procedure visibility(Sender:Tobject);

begin

frm_addcourse.Edit2.Visible:=true;

frm_addcourse.Edit3.Visible:=true;

frm_addcourse.Edit1.Visible:=true;

frm_addcourse.combobox1.Visible:=true;

frm_addcourse.combobox2.Visible:=true;

frm_addcourse.combobox3.visible:=true;

end;

procedure enabling(Sender:Tobject);

begin

frm_addcourse.edit2.enabled:=true;

frm_addcourse.edit3.enabled:=true;

frm_addcourse.edit1.enabled:=true;

frm_addcourse.combobox1.enabled:=true;

frm_addcourse.combobox2.enabled:=true;

frm_addcourse.combobox3.enabled:=true;

```
disabling(Sender:Tobject);

course.edit2.enabled:=false;
course.edit3.enabled:=false;
course.edit1.enabled:=false;
course.combobox1.enabled:=false;
course.combobox2.enabled:=false;
course.combobox3.enabled:=false;


Tfrm_addcourse.FormClose(Sender: TObject;
: TCloseAction);

course.Show;


Tfrm_addcourse.Button1Click(Sender: TObject);

addcourse.Close;


Tfrm_addcourse.Button2Click(Sender: TObject);
```

```
Show;

Tfrm_addcourse.FormShow(Sender: TObject);

sender);
sender);
Focus;
frm_main.ADOquery1 do

'Select DepartmentId from Department');

frm_main.datasource7.DataSet.First;
frm_main.datasource7.DataSet.Eof do

box1.Items.Add(frm_main.datasource7.DataSet.FieldValues['DepartmentId']);
frm_main.DataSource7.DataSet.Next;
```

```
Tfrm_addcourse.ComboBox1Exit(Sender: TObject);

_main.ADOquery4 do

Select TeacherId,Tname+" "+Tsurname As [Teacher Name&Surname] from
where DepartmentId="'+combobox1.Text+'");

DataSource:=frm_main.DataSource10;

2.Items.Clear;

.datasource10.DataSet.First;

not frm_main.datasource10.DataSet.Eof do

box2.Items.Add(frm_main.datasource10.DataSet.FieldValues['TeacherId']);

.DataSource10.DataSet.Next;

Tfrm_addcourse.Button4Click(Sender: TObject);

(sender);

frm_main.ADOquery1 do
```

```
begin
  with
  clear;
  add('Select DepartmentId from Department');
  open;
  end;
  frm_main.datasource7.DataSet.First;
  while not frm_main.datasource7.DataSet.Eof do
  begin
  combobox1.Items.Add(frm_main.datasource7.DataSet.FieldValues['DepartmentId']);
  frm_main.DataSource7.DataSet.Next;
  end;
  SetFocus;
  DataSource:=frm_main.DataSource13;
  Repaint;
  end;


procedure Tfrm_addcourse.Button3Click(Sender: TObject);
begin
  with frm_main.ADOquery3 do
  begin
  sql.clear;
  sql.Add('Select CourseCode from course where coursecode="'+edit1.text+'"');
  open;
  end;
  if frm_main.DataSource9.DataSet.RecordCount=0 then
  begin
```

```
Text<>")and(edit3.Text<>")and(combobox1.Text<>")and(combobox2.Text<>")and(co
Text<>")) then
egin
with frm_main.DataSource6 do
egin
dataset.Edit;
dataset.Append;
dataset.FieldValues['Coursecode']:=edit1.Text;
if combobox1.text<>'' then
dataset.FieldValues['DepartmentId']:=combobox1.Text;

dataset.FieldValues['Coursename']:=edit2.Text;

dataset.FieldValues['Coursecontent']:=edit3.Text;

if combobox2.text<>'' then
dataset.FieldValues['TeacherId']:=StrToint(combobox2.Text);

if combobox3.text<>'' then
dataset.FieldValues['Credit']:=Strtoint(combobox3.Text);
dataset.FieldValues['ProcessDate']:=Date;
dataset.FieldValues['ProcessUser']:=username;

dataset.Post;
messagedlg('The Course recorded successfully !',mtwarning,[mbok],0);
clearing(sender);
```

```
edit1.SetFocus;

with frm_main.ADOquery1 do
begin
sql.clear;
sql.Add('Select DepartmentId from Department');
open;
end;
frm_main.datasource7.DataSet.First;
while not frm_main.datasource7.DataSet.Eof do
begin
combobox1.Items.Add(frm_main.datasource7.DataSet.FieldValues['DepartmentId']);
frm_main.DataSource7.DataSet.Next;
end;
end;
end
else
begin
messagedlg('Please Fill All The Required Fields !',mtwarning,[mbok],0);
edit2.SetFocus;
end;

end
else
begin
messagedlg('This Course Is Already Recorded !',mtwarning,[mbok],0);
```

Edit1.SetFocus;

end;

end;

end;

UNIT 10

unit Unit11;

interface

uses

Windows, Messages, SysUtils, Variants, Classes, Graphics, Controls, Forms,

Dialogs, Buttons, jpeg, ExtCtrls;

type

Form_dept = class(TForm)

SpeedButton1: TSpeedButton;

SpeedButton2: TSpeedButton;

SpeedButton3: TSpeedButton;

procedure FormClose(Sender: TObject; var Action: TCloseAction);

procedure SpeedButton3Click(Sender: TObject);

procedure SpeedButton1Click(Sender: TObject);

procedure SpeedButton2Click(Sender: TObject);

private

{ Private declarations }

```
public
  { Public declarations }
end;

var
  frm_dept: Tfrm_dept;

implementation

uses Unit1, Unit12, Unit13;

{$R *.dfm}

procedure Tfrm_dept.FormClose(Sender: TObject; var Action: TCloseAction);
begin
  frm_main.show;
end;

procedure Tfrm_dept.SpeedButton3Click(Sender: TObject);
begin
  frm_dept.Close;
end;

procedure Tfrm_dept.SpeedButton1Click(Sender: TObject);
begin
  frm_adddept.show;
```

frm_dept.Hide;

end;

procedure Tfrm_dept.SpeedButton2Click(Sender: TObject);

begin

frm_upddept.show;

frm_dept.Hide;

end;

end.

## UNIT 12

unit Unit12;

interface

uses

Windows, Messages, SysUtils, Variants, Classes, Graphics, Controls, Forms,

Dialogs, StdCtrls, ExtCtrls, DBCtrls, Mask, ComCtrls;

type

Tfrm_adddept = class(TForm)

Label1: TLabel;

Label2: TLabel;

Button1: TButton;

Button2: TButton;

Edit1: TEdit;

Edit2: TEdit;

```
    TButton;
    TButton;
    TButton;
    TStatusBar;
    Button1Click(Sender: TObject);
    FormClose(Sender: TObject; var Action: TCloseAction);
    Button2Click(Sender: TObject);
    FormShow(Sender: TObject);
    Button5Click(Sender: TObject);
    Button3Click(Sender: TObject);
    Button4Click(Sender: TObject);
    Button2MouseMove(Sender: TObject; Shift: TShiftState; X,
    Integer);
    FormMouseMove(Sender: TObject; Shift: TShiftState; X,
    Integer);
  private
    { Private declarations }
  public
    { Public declarations }
  end;

var
  frm_adddept: Tfrm_adddept;

implementation
```

```
Unit1, Unit11,unit26;

*.dfm}

dure clearing(Sender:Tobject);

_adddept.Edit1.Clear;
_adddept.Edit2.Clear;

dure invisibility(Sender:Tobject);

_adddept.Edit2.Visible:=false;
_adddept.button3.Visible:=false;
_adddept.button4.Visible:=false;

dure visibility(Sender:Tobject);

_adddept.Edit2.Visible:=true;
_adddept.button3.Visible:=true;
_adddept.button4.Visible:=true;

dure enabling(Sender:Tobject);
```

_adddept.Edit2.Enabled:=true;

_adddept.button3.Enabled:=true;

_adddept.button4.Enabled:=true;

dure disabling(Sender:Tobject);

dept.Edit2.Enabled:=false;

*tton3.Enabled:=false;*

dept.button4.Enabled:=false;

Tfrm_adddept.Button1Click(Sender: TObject);

hide;

Tfrm_adddept.FormClose(Sender: TObject;

TCloseAction);

Tfrm_adddept.Button2Click(Sender: TObject);

```
Close;

Tfrm_adddept.FormShow(Sender: TObject);

Tfrm_adddept.Button5Click(Sender: TObject);

<>" then

frm_main.ADOquery1 do

clear;
Add('Select DepartmentId from Department where DepartmentId="'+edit1.text+'"');

main.DataSource7.DataSet.RecordCount=0 then

sibility(sender);
bling(sender);
edit2.SetFocus;
```

edlg('This Department ID Is Already Recorded !',mtwarning,[mbok],0);

SetFocus;

Clear;

edlg('Please Enter Department Id For Te New Department ! ',mtwarning,[mbok],0);

Clear;

SetFocus;

```
Tfrm_adddept.Button3Click(Sender: TObject);

Text<>")and(edit2.Text<>")) then

frm_main.DataSource5 do

Edit;

Append;

edit2.text<>" then

set.FieldValues['DepartmentId']:=edit1.Text;

edit2.text<>" then

set.FieldValues['DepartmentName']:=edit2.Text;

set.FieldValues['ProcessUser']:=username;

set.FieldValues['ProcessDate']:=Date;

set.Post;

messagedlg('The Department recorded successfully !',mtwarning,[mbok],0);

clearing(sender);

disabling(sender);

invisibility(sender);

edit1.SetFocus;
```

```
g('Please Fill All Fields !',mtwarning,[mbok],0);
SetFocus;


Tfrm_adddept.Button4Click(Sender: TObject);

sender);
sender);
ility(sender);
SetFocus;


Tfrm_adddept.Button2MouseMove(Sender: TObject;
TShiftState; X, Y: Integer);

1.SimpleText:=(sender as tbutton).Hint;


Tfrm_adddept.FormMouseMove(Sender: TObject; Shift: TShiftState;
Integer);
```

SimpleText:='';

Messages, SysUtils, Variants, Classes, Graphics, Controls, Forms,

StdCtrls, ExtCtrls, DBCtrls, Mask, ComCtrls;

= class(TForm)

: TLabel;

: TLabel;

: TButton;

: TButton;

: TLabel;

: TPanel;

: TEdit;

Bar1: TStatusBar;

: TButton;

: TEdit;

: TEdit;

: TButton;

```
    : TButton;
    FormClose(Sender: TObject; var Action: TCloseAction);
    Button2Click(Sender: TObject);
    Button1Click(Sender: TObject);
    FormShow(Sender: TObject);
    Button3Click(Sender: TObject);
    Button4Click(Sender: TObject);
    Button5Click(Sender: TObject);

    { Private declarations }

    { Public declarations }

var
  frm_upddept: Tfrm_upddept;

implementation

uses Unit11, Unit1, unit26;

{$R *.dfm}

procedure Tfrm_upddept.FormClose(Sender: TObject; var Action: TCloseAction);
begin

frm_dept.show;
```

Tfrm_upddept.Button2Click(Sender: TObject);

dept.hide;

dept.show;

Tfrm_upddept.Button1Click(Sender: TObject);

dept.Close;

main.Show ;

Tfrm_upddept.FormShow(Sender: TObject);

Clear;

Clear;

Clear;

ReadOnly:=false;

SetFocus;

Visible:=false;

Visible:=false;

4.Visible:=false;

5.Visible:=false;

```
Tfrm_upddept.Button3Click(Sender: TObject);

Text<>'') then

frm_main.ADOquery1 do

clear;
Add('Select *  from Department where DepartmentId='''+edit1.text+'''');

_main.DataSource7.DataSet.RecordCount>0 then

Visible:=true;
Visible:=true;
4.Visible:=true;
5.Visible:=true;

_main.DataSource7.DataSet.FieldValues['DepartmentId']<>null then
2.Text:=frm_main.DataSource7.DataSet.FieldValues['DepartmentId'];
_main.DataSource7.DataSet.FieldValues['DepartmentName']<>null  then
3.Text:=frm_main.DataSource7.DataSet.FieldValues['DepartmentName'];
```

```
ReadOnly:=true;

MessageDlg('Department ID is not found !',mtinformation,[mbok],0);
SetFocus;

MessageDlg('Please Enter Department ID !',mtinformation,[mbok],0);
SetFocus;
Clear;

procedure Tfrm_upddept.Button4Click(Sender: TObject);

Edit1.Text<>edit2.Text then
```

```
begin
with frm_main.ADOquery2 do
begin
sql.clear;
sql.Add('Select departmentID from Department where DepartmentId="'+edit2.text+'"');
open;
end;
frm_main.DataSource8.DataSet.RecordCount=0 then
begin
with frm_main.DataSource7 do
begin
edit1.ReadOnly:=false;
dataset.Edit;
if edit2.Text<>'' then
dataset.FieldValues['DepartmentId']:=edit2.Text;
dataset.FieldValues['UpdateUser']:=username;
dataset.FieldValues['Updatedate']:=date;
if edit3.text<>'' then
dataset.FieldValues['DepartmentName']:=Edit3.Text;
dataset.post;
 with frm_main.ADOquery3 do
begin
 sql.clear;
 sql.Add('Select * from TEACHER where DepartmentId="'+edit1.text+'"');
 open;
end;
```

```
frm_main.DataSource9.DataSet.RecordCount>0 then
begin
with frm_main.DataSource9 do
begin
dataset.edit;
dataset.First;
while not (dataset.Eof) do
begin
 dataset.FieldValues['DepartmentId']:=edit2.Text;
 dataset.Next;
end;
dataset.Post;
END;

messagedlg('The Department Information in All Tables are Updated Successfully
information,[mbok],0);

edit1.SetFocus;
edit1.Clear;
edit2.Clear;
edit3.Clear;
edit2.Visible:=false;
edit3.Visible:=false;
button4.Visible:=false;
button5.Visible:=false;
end; }
end;
```

```
essagedlg('Duplicate Department ID is not allowed !',mtinformation,[mbok],0);
dit2.SetFocus;
nd

nd

egin

th fm_main.DataSource7 do

egin

dataset.edit;

dataset.FieldValues['UpdateUser']:=username;

dataset.FieldValues['Updatedate']:=date;

Edit3.text<>'' then

dataset.FieldValues['DepartmentName']:=Edit3.Text;

dataset.post;

essagedlg('The Department Information Recorded Successfully
nformation,[mbok],0);

dit1.SetFocus;

dit1.ReadOnly:=false;

dit1.Clear;

dit2.Clear;

dit3.Clear;

dit2.Visible:=false;

dit3.Visible:=false;

utton4.Visible:=false;
```

Visible:=false;

Tfrm_upddept.Button5Click(Sender: TObject);

Focus;

ReadOnly:=false;

Clear;

Clear;

Clear;

Visible:=false;

Visible:=false;

Visible:=false;

Visible:=false;

NIT 14

Unit14;

face

Messages, SysUtils, Variants, Classes, Graphics, Controls, Forms,

ExtCtrls, StdCtrls, Mask, DBCtrls, Grids, DBGrids;

= class(TForm)

TLabel;

TEdit;

TButton;

TPanel;

TLabel;

TLabel;

TLabel;

TLabel;

TLabel;

TPanel;

TPanel;

TPanel;

TPanel;

TPanel;

TPanel;

TPanel;

TPanel;

TPanel;

TPanel;

TLabel;

TLabel;

TLabel;

TLabel;

TLabel;

TLabel;

TLabel;

TLabel;

TButton;

TButton;

TEdit;

TEdit;

TEdit;

TEdit;

TEdit;

Grid1: TStringGrid;

Grid2: TStringGrid;

Grid3: TStringGrid;

Grid4: TStringGrid;

Grid5: TStringGrid;

Grid6: TStringGrid;

Grid7: TStringGrid;

Grid8: TStringGrid;

16: TLabel;

17: TLabel;

18: TLabel;

19: TLabel;

20: TLabel;

21: TLabel;

TLabel;

TLabel;

TLabel;

TLabel;

TLabel;

TLabel;

TLabel;

TLabel;

TLabel;

TLabel;

TLabel;

TLabel;

TLabel;

TLabel;

TPanel;

TLabel;

StringGrid9: TStringGrid;

TLabel;

TLabel;

TPanel;

TLabel;

StringGrid10: TStringGrid;

Label39: TLabel;

Label40: TLabel;

Panel13: TPanel;

Label41: TLabel;

TStringGrid;

TLabel;

TLabel;

TPanel;

TLabel;

TStringGrid;

TButton;

TButton;

TLabel;

TLabel;

TLabel;

FormClose(Sender: TObject; var Action: TCloseAction);

Button3Click(Sender: TObject);

Button1Click(Sender: TObject);

Edit1KeyPress(Sender: TObject; var Key: Char);

Button2Click(Sender: TObject);

Button5Click(Sender: TObject);

Button4Click(Sender: TObject);

FormShow(Sender: TObject);

{ Private declarations }

{ Public declarations }

Tfrm_trans;

Unit16;

clearing(Sender:Tobject);

_trans do

.Clear;

.Clear;

.Clear;

.Clear;

.Clear;

.Clear;

.Caption:='';

.Caption:='';

.Caption:='';

.Caption:='';

.Caption:='';

.Caption:='';

.Caption:='';

.Caption:='';

.Caption:='';

.Caption:='';

.Caption:='';

.Caption:='';

.Caption:='';

.Caption:='';

.Caption:='';

.Caption:='';

.Caption:='';

.Caption:='';

.Caption:='';

.Caption:='';

.Caption:='';

.Caption:='';

.Caption:='';

.Caption:='';

.Caption:='';

.Caption:='';

to 6 do

rid1.Cells[0,i]:='';

rid1.Cells[1,i]:='';

```
to 6 do
Cells[0,i]:='';
Cells[1,i]:='';
to 6 do
Cells[0,i]:='';
Cells[1,i]:='';
to 6 do
Cells[0,i]:='';
Cells[1,i]:='';
to 6 do
Cells[0,i]:='';
```

```
.Cells[1,i]:='';
end;

end;
=1 to 6 do
begin
Grid6.Cells[0,i]:='';
Grid6.Cells[1,i]:='';
end;

end;
=1 to 6 do
begin
Grid7.Cells[0,i]:='';
Grid7.Cells[1,i]:='';
end;

end;
=1 to 6 do
begin
Grid8.Cells[0,i]:='';
Grid8.Cells[1,i]:='';
end;

end;
=1 to 6 do
```

```
begin
  stringgrid9.Cells[0,i]:='';
  stringgrid9.Cells[1,i]:='';
end;

end;
for i:=1 to 6 do
begin
  stringgrid10.Cells[0,i]:='';
  stringgrid10.Cells[1,i]:='';
end;

end;
for i:=1 to 6 do
begin
  stringgrid11.Cells[0,i]:='';
  stringgrid11.Cells[1,i]:='';
end;

end;
for i:=0 to 6 do
begin
  stringgrid12.Cells[0,i]:='';
  stringgrid12.Cells[1,i]:='';
end;
```

Focus;

Tfrm_trans.FormClose(Sender: TObject; var Action: TCloseAction);

main.show;

Tfrm_trans.Button3Click(Sender: TObject);

Close;

Tfrm_trans.Button1Click(Sender: TObject);

sumcredit,sumgencredit,varcorId:integer;

sumgrade,sumgengrade:real;

Tdate;

regmonth,mpart,ypart,season,t:string;

```
DataSource7.DataSet:=frm_main.ADOQuery1;


text<>null) then
with frm_main.ADOQuery1 do
SQL.Clear;
add('SELECT * FROM Student WHERE StudentID='''+edit1.text+'''');
open;

frm_main.DataSource7.DataSet.RecordCount=0 then
messagedlg('Student is not found !!',mtwarning,[mbok],0);
clearing(sender);
edit2.Text:=frm_main.DataSource7.DataSet.FieldValues['StudentId'];
edit3.Text:=frm_main.DataSource7.DataSet.FieldValues['name'];
edit4.Text:=frm_main.DataSource7.DataSet.FieldValues['Surname'];
```

```
Text:=datetostr(frm_main.DataSource7.DataSet.FieldValues['Birthdate']);

Text:=frm_main.DataSource7.DataSet.FieldValues['DepartmentId'];


main.DataSource7.DataSet.FieldValues['Isgraduated']=true then

Caption:='Graduated At
frm_main.DataSource7.DataSet.FieldValues['Graduationdate']);



write semester names


frm_main.ADOQuery1 do

Clear;

add('SELECT t.courseregdate,t.grade,c.coursecode,c.credit FROM transcript as t
course as c on t.courseId=c.courseId WHERE t.StudentID="'+edit1.text+'" and
is null and t.grade is not null order by t.transId');

open;

```

```
frm_main.DataSource7.DataSet.First;

label45.Caption:=inttostr(frm_main.DataSource7.DataSet.RecordCount);


realdate:=frm_main.DataSource7.DataSet.FieldValues['courseregdate'];

shortdateformat:=('yyyy');

regyear:=datetostr(realdate);

ypart:=regyear;
```

```
format:=('mm');
=datetostr(realdate);
=regmonth;
=0;
=0;

((abs(strtoint(regmonth)-
))>2)or(regyear<>ypart)or(frm_main.DataSource7.DataSet.eof))do

='01')or(regmonth='12')or(regmonth='11')or(regmonth='10')or(regmonth='09')or(re
)) then
='Fall'
:='Spring';

grid1.Cells[0,i]:=frm_main.DataSource7.DataSet.FieldValues['Coursecode'];
grid1.Cells[1,i]:=frm_main.DataSource7.DataSet.FieldValues['grade'];
=i+1;
dateformat:=('yyyy');
7.Caption:=REGYEAR+' - '+season+' Term';
dateformat:=('dd/mm/yyyy');

credit:=sumcredit+frm_main.DataSource7.dataset.FieldValues['credit'];
```

```
if frm_main.DataSource7.dataset.FieldValues['grade']='AA' then
sumgrade:=sumgrade+frm_main.DataSource7.dataset.FieldValues['credit']*4;
if frm_main.DataSource7.dataset.FieldValues['grade']='BA' then
sumgrade:=sumgrade+frm_main.DataSource7.dataset.FieldValues['credit']*3.5;
if frm_main.DataSource7.dataset.FieldValues['grade']='BB' then
sumgrade:=sumgrade+frm_main.DataSource7.dataset.FieldValues['credit']*3;
if frm_main.DataSource7.dataset.FieldValues['grade']='CB' then
sumgrade:=sumgrade+frm_main.DataSource7.dataset.FieldValues['credit']*2.5;
if frm_main.DataSource7.dataset.FieldValues['grade']='CC' then
sumgrade:=sumgrade+frm_main.DataSource7.dataset.FieldValues['credit']*2;
if frm_main.DataSource7.dataset.FieldValues['grade']='DC' then
sumgrade:=sumgrade+frm_main.DataSource7.dataset.FieldValues['credit']*1.5;
if frm_main.DataSource7.dataset.FieldValues['grade']='DD' then
sumgrade:=sumgrade+frm_main.DataSource7.dataset.FieldValues['credit']*1;
if frm_main.DataSource7.dataset.FieldValues['grade']='FD' then
sumgrade:=sumgrade+frm_main.DataSource7.dataset.FieldValues['credit']*0.5;
if frm_main.DataSource7.dataset.FieldValues['grade']='FF' then
sumgrade:=sumgrade+frm_main.DataSource7.dataset.FieldValues['credit']*0;


frm_main.DataSource7.DataSet.Next;
realdate:=frm_main.DataSource7.DataSet.FieldValues['courseregdate'];
shortdateformat:=('yyyy');
regyear:=datetostr(realdate);
shortdateformat:=('mm');
regmonth:=datetostr(realdate);
```

```
end;
if(sumcredit>0)) then
begin
t:=(sumgrade/sumcredit):3:2,t);
label16.Caption:=t;
end;
end of first stringgrid
start of second stringgrid

realdate:=frm_main.DataSource7.DataSet.FieldValues['courseregdate'];
shortdateformat:=('yyyy');
regyear:=datetostr(realdate);
ypart:=regyear;
shortdateformat:=('mm');
regmonth:=datetostr(realdate);
mpart:=regmonth;
i:=0;
sumcredit:=0;
sumgrade:=0;

while not(((abs(strtoint(regmonth)-
strtoint(mpart)))>2)or(regyear<>ypart)or(frm_main.DataSource7.DataSet.eof))do
begin
```

```
=01')or(regmonth='12')or(regmonth='11')or(regmonth='10')or(regmonth='09')or(re
=08')) then

:='Fall'

:='Spring';


rid2.Cells[0,i]:=frm_main.DataSource7.DataSet.FieldValues['Coursecode'];

rid2.Cells[1,i]:=frm_main.DataSource7.DataSet.FieldValues['grade'];

:=i+1;

ateformat:=('yyyy');

.Caption:=REGYEAR+' - '+season+' Term';

ateformat:=('dd/mm/yyyy');


redit:=sumcredit+frm_main.DataSource7.dataset.FieldValues['credit'];


f frm_main.DataSource7.dataset.FieldValues['grade']='AA' then

grade:=sumgrade+frm_main.DataSource7.dataset.FieldValues['credit']*4;

f frm_main.DataSource7.dataset.FieldValues['grade']='BA' then

grade:=sumgrade+frm_main.DataSource7.dataset.FieldValues['credit']*3.5;

f frm_main.DataSource7.dataset.FieldValues['grade']='BB' then

grade:=sumgrade+frm_main.DataSource7.dataset.FieldValues['credit']*3;

f frm_main.DataSource7.dataset.FieldValues['grade']='CB' then

grade:=sumgrade+frm_main.DataSource7.dataset.FieldValues['credit']*2.5;

f frm_main.DataSource7.dataset.FieldValues['grade']='CC' then

grade:=sumgrade+frm_main.DataSource7.dataset.FieldValues['credit']*2;

f frm_main.DataSource7.dataset.FieldValues['grade']='DC' then
```

```
sumgrade:=sumgrade+frm_main.DataSource7.dataset.FieldValues['credit']*1.5;
frm_main.DataSource7.dataset.FieldValues['grade']='DD' then
sumgrade:=sumgrade+frm_main.DataSource7.dataset.FieldValues['credit']*1;
frm_main.DataSource7.dataset.FieldValues['grade']='FD' then
sumgrade:=sumgrade+frm_main.DataSource7.dataset.FieldValues['credit']*0.5;
frm_main.DataSource7.dataset.FieldValues['grade']='FF' then
sumgrade:=sumgrade+frm_main.DataSource7.dataset.FieldValues['credit']*0;



frm_main.DataSource7.DataSet.Next;
realdate:=frm_main.DataSource7.DataSet.FieldValues['courseregdate'];
formatdateformat:=('yyyy');
regyear:=datetostr(realdate);
formatdateformat:=('mm');
regmonth:=datetostr(realdate);


end;
if (sumcredit>0)) then
begin
str(sumgrade/sumcredit):3:2,t);
Label17.Caption:=t;
end;


end of second stringgrid
end of 3. stringgrid
```

```
=frm_main.DataSource7.DataSet.FieldValues['courseregdate'];
ateformat:=('yyyy');
=datetostr(realdate);
=regyear;
ateformat:=('mm');
:=datetostr(realdate);
=regmonth;

edit:=0;
ade:=0;


ile not(((abs(strtoint(regmonth)-
mpart)))>2)or(regyear<>ypart)or(frm_main.DataSource7.DataSet.eof))do
gin



f
th='01')or(regmonth='12')or(regmonth='11')or(regmonth='10')or(regmonth='09')or(re
th='08')) then
 ason:='Fall'
se
 ason:='Spring';


inggrid3.Cells[0,i]:=frm_main.DataSource7.DataSet.FieldValues['Coursecode'];
inggrid3.Cells[1,i]:=frm_main.DataSource7.DataSet.FieldValues['grade'];
=i+1;
ortdateformat:=('yyyy');
el9.Caption:=REGYEAR+'  -  '+season+' Term';
```

dateformat:=('dd/mm/yyyy');

sumcredit:=sumcredit+frm_main.DataSource7.dataset.FieldValues['credit'];

if frm_main.DataSource7.dataset.FieldValues['grade']='AA' then
sumgrade:=sumgrade+frm_main.DataSource7.dataset.FieldValues['credit']*4;
if frm_main.DataSource7.dataset.FieldValues['grade']='BA' then
sumgrade:=sumgrade+frm_main.DataSource7.dataset.FieldValues['credit']*3.5;
if frm_main.DataSource7.dataset.FieldValues['grade']='BB' then
sumgrade:=sumgrade+frm_main.DataSource7.dataset.FieldValues['credit']*3;
if frm_main.DataSource7.dataset.FieldValues['grade']='CB' then
sumgrade:=sumgrade+frm_main.DataSource7.dataset.FieldValues['credit']*2.5;
if frm_main.DataSource7.dataset.FieldValues['grade']='CC' then
sumgrade:=sumgrade+frm_main.DataSource7.dataset.FieldValues['credit']*2;
if frm_main.DataSource7.dataset.FieldValues['grade']='DC' then
sumgrade:=sumgrade+frm_main.DataSource7.dataset.FieldValues['credit']*1.5;
if frm_main.DataSource7.dataset.FieldValues['grade']='DD' then
sumgrade:=sumgrade+frm_main.DataSource7.dataset.FieldValues['credit']*1;
if frm_main.DataSource7.dataset.FieldValues['grade']='FD' then
sumgrade:=sumgrade+frm_main.DataSource7.dataset.FieldValues['credit']*0.5;
if frm_main.DataSource7.dataset.FieldValues['grade']='FF' then
sumgrade:=sumgrade+frm_main.DataSource7.dataset.FieldValues['credit']*0;

frm_main.DataSource7.DataSet.Next;
realdate:=frm_main.DataSource7.DataSet.FieldValues['courseregdate'];

```
dateformat:=('yyyy');
=datetostr(realdate);
dateformat:=('mm');
:=datetostr(realdate);


credit>0)) then

grade/sumcredit):3:2,t);
18.Caption:=t;


3. stringgrid


of 4 stringgrid


:=frm_main.DataSource7.DataSet.FieldValues['courseregdate'];
dateformat:=('yyyy');
:=datetostr(realdate);
:=regyear;
dateformat:=('mm');
:=datetostr(realdate);
:=regmonth;

credit:=0;
grade:=0;
```

```
(((abs(strtoint(regmonth)-
))))>2)or(regyear<>ypart)or(frm_main.DataSource7.DataSet.eof))do

='01')or(regmonth='12')or(regmonth='11')or(regmonth='10')or(regmonth='09')or(re
='08')) then
:='Fall'
:='Spring';
ringgrid4.Cells[0,i]:=frm_main.DataSource7.DataSet.FieldValues['Coursecode'];
ringgrid4.Cells[1,i]:=frm_main.DataSource7.DataSet.FieldValues['grade'];
:=i+1;
dateformat:=('yyyy');
abel10.Caption:=REGYEAR+' - '+season+' Term';
dateformat:=('dd/mm/yyyy');

umcredit:=sumcredit+frm_main.DataSource7.dataset.FieldValues['credit'];

if frm_main.DataSource7.dataset.FieldValues['grade']='AA' then
umgrade:=sumgrade+frm_main.DataSource7.dataset.FieldValues['credit']*4;
if frm_main.DataSource7.dataset.FieldValues['grade']='BA' then
umgrade:=sumgrade+frm_main.DataSource7.dataset.FieldValues['credit']*3.5;
if frm_main.DataSource7.dataset.FieldValues['grade']='BB' then
umgrade:=sumgrade+frm_main.DataSource7.dataset.FieldValues['credit']*3;
```

```
frm_main.DataSource7.dataset.FieldValues['grade']='CB' then
umgrade:=sumgrade+frm_main.DataSource7.dataset.FieldValues['credit']*2.5;
frm_main.DataSource7.dataset.FieldValues['grade']='CC' then
umgrade:=sumgrade+frm_main.DataSource7.dataset.FieldValues['credit']*2;
frm_main.DataSource7.dataset.FieldValues['grade']='DC' then
umgrade:=sumgrade+frm_main.DataSource7.dataset.FieldValues['credit']*1.5;
frm_main.DataSource7.dataset.FieldValues['grade']='DD' then
umgrade:=sumgrade+frm_main.DataSource7.dataset.FieldValues['credit']*1;
frm_main.DataSource7.dataset.FieldValues['grade']='FD' then
umgrade:=sumgrade+frm_main.DataSource7.dataset.FieldValues['credit']*0.5;
frm_main.DataSource7.dataset.FieldValues['grade']='FF' then
umgrade:=sumgrade+frm_main.DataSource7.dataset.FieldValues['credit']*0;


frm_main.DataSource7.DataSet.Next;
realdate:=frm_main.DataSource7.DataSet.FieldValues['courseregdate'];
shortdateformat:=('yyyy');
regyear:=datetostr(realdate);
shortdateformat:=('mm');
regmonth:=datetostr(realdate);


end;
if ((sumcredit>0)) then
begin
str((sumgrade/sumcredit):3:2,t);
label19.Caption:=t;
```

```
ext;

ol of 4. stringgrid


er of 5. stringgrid


ealdate:=frm_main.DataSource7.DataSet.FieldValues['courseregdate'];

ormdateformat:=('yyyy');

egyear:=datetostr(realdate);

part:=regyear;

ormdateformat:=('mm');

egmonth:=datetostr(realdate);

part:=regmonth;

el;

mcredit:=0;

mgrade:=0;


hile not(((abs(strtoint(regmonth)-
f mpart)))>2)or(regyear<>ypart)or(frm_main.DataSource7.DataSet.eof))do

egin


nth='01')or(regmonth='12')or(regmonth='11')or(regmonth='10')or(regmonth='09')or(re
='08')) then

eason:='Fall'

se

eason:='Spring';
```

```
ggrid5.Cells[0,i]:=frm_main.DataSource7.DataSet.FieldValues['Coursecode'];
ggrid5.Cells[1,i]:=frm_main.DataSource7.DataSet.FieldValues['grade'];
=i+1;
dateformat:=('yyyy');
ell1.Caption:=REGYEAR+' - '+season+' Term';
dateformat:=('dd/mm/yyyy');

redit:=sumcredit+frm_main.DataSource7.dataset.FieldValues['credit'];

f frm_main.DataSource7.dataset.FieldValues['grade']='AA' then
grade:=sumgrade+frm_main.DataSource7.dataset.FieldValues['credit']*4;
f frm_main.DataSource7.dataset.FieldValues['grade']='BA' then
grade:=sumgrade+frm_main.DataSource7.dataset.FieldValues['credit']*3.5;
f frm_main.DataSource7.dataset.FieldValues['grade']='BB' then
grade:=sumgrade+frm_main.DataSource7.dataset.FieldValues['credit']*3;
f frm_main.DataSource7.dataset.FieldValues['grade']='CB' then
grade:=sumgrade+frm_main.DataSource7.dataset.FieldValues['credit']*2.5;
f frm_main.DataSource7.dataset.FieldValues['grade']='CC' then
grade:=sumgrade+frm_main.DataSource7.dataset.FieldValues['credit']*2;
f frm_main.DataSource7.dataset.FieldValues['grade']='DC' then
grade:=sumgrade+frm_main.DataSource7.dataset.FieldValues['credit']*1.5;
f frm_main.DataSource7.dataset.FieldValues['grade']='DD' then
grade:=sumgrade+frm_main.DataSource7.dataset.FieldValues['credit']*1;
f frm_main.DataSource7.dataset.FieldValues['grade']='FD' then
grade:=sumgrade+frm_main.DataSource7.dataset.FieldValues['credit']*0.5;
```

frm_main.DataSource7.dataset.FieldValues['grade']='FF' then

sumgrade:=sumgrade+frm_main.DataSource7.dataset.FieldValues['credit']*0;

frm_main.DataSource7.DataSet.Next;

realdate:=frm_main.DataSource7.DataSet.FieldValues['courseregdate'];

formatdateformat:=('yyyy');

regyear:=datetostr(realdate);

formatdateformat:=('mm');

regmonth:=datetostr(realdate);

end;

if (sumcredit>0)) then

begin

str(sumgrade/sumcredit):3:2,t);

Label20.Caption:=t;

end;

end of 5. stringgrid

start of 6. stringgrid

realdate:=frm_main.DataSource7.DataSet.FieldValues['courseregdate'];

formatdateformat:=('yyyy');

regyear:=datetostr(realdate);

regstart:=regyear;

```
dateformat:=('mm');
nth:=datetostr(realdate);
=regmonth;

redit:=0;
grade:=0;

ile not(((abs(strtoint(regmonth)-
part)))>2)or(regyear<>ypart)or(frm_main.DataSource7.DataSet.eof))do
gin


='01')or(regmonth='12')or(regmonth='11')or(regmonth='10')or(regmonth='09')or(re
='08')) then
son:='Fall'
se
son:='Spring';

inggrid6.Cells[0,i]:=frm_main.DataSource7.DataSet.FieldValues['Coursecode'];
inggrid6.Cells[1,i]:=frm_main.DataSource7.DataSet.FieldValues['grade'];
=i+1;
ortdateformat:=('yyyy');
el12.Caption:=REGYEAR+' - '+season+' Term';
ortdateformat:=('dd/mm/yyyy');

mcredit:=sumcredit+frm_main.DataSource7.dataset.FieldValues['credit'];
```

frm_main.DataSource7.dataset.FieldValues['grade']='AA' then

sumgrade:=sumgrade+frm_main.DataSource7.dataset.FieldValues['credit']*4;

frm_main.DataSource7.dataset.FieldValues['grade']='BA' then

sumgrade:=sumgrade+frm_main.DataSource7.dataset.FieldValues['credit']*3.5;

frm_main.DataSource7.dataset.FieldValues['grade']='BB' then

sumgrade:=sumgrade+frm_main.DataSource7.dataset.FieldValues['credit']*3;

frm_main.DataSource7.dataset.FieldValues['grade']='CB' then

sumgrade:=sumgrade+frm_main.DataSource7.dataset.FieldValues['credit']*2.5;

frm_main.DataSource7.dataset.FieldValues['grade']='CC' then

sumgrade:=sumgrade+frm_main.DataSource7.dataset.FieldValues['credit']*2;

frm_main.DataSource7.dataset.FieldValues['grade']='DC' then

sumgrade:=sumgrade+frm_main.DataSource7.dataset.FieldValues['credit']*1.5;

frm_main.DataSource7.dataset.FieldValues['grade']='DD' then

sumgrade:=sumgrade+frm_main.DataSource7.dataset.FieldValues['credit']*1;

frm_main.DataSource7.dataset.FieldValues['grade']='FD' then

sumgrade:=sumgrade+frm_main.DataSource7.dataset.FieldValues['credit']*0.5;

frm_main.DataSource7.dataset.FieldValues['grade']='FF' then

sumgrade:=sumgrade+frm_main.DataSource7.dataset.FieldValues['credit']*0;

frm_main.DataSource7.DataSet.Next;

realdate:=frm_main.DataSource7.DataSet.FieldValues['courseregdate'];

shortdateformat:=('yyyy');

regyear:=datetostr(realdate);

shortdateformat:=('mm');

regmonth:=datetostr(realdate);

```
credit>0)) then
grade/sumcredit):3:2,t);
21.Caption:=t;
6. stringgrid
7. stringgrid
:=frm_main.DataSource7.DataSet.FieldValues['courseregdate'];
dateformat:=('yyyy');
:=datetostr(realdate);
=regyear;
dateformat:=('mm');
onth:=datetostr(realdate);
=regmonth;
credit:=0;
grade:=0;
while not(((abs(strtoint(regmonth)-
mpart)))>2)or(regyear<>ypart)or(frm_main.DataSource7.DataSet.eof))do
begin
```

```
if
gmonth='01')or(regmonth='12')or(regmonth='11')or(regmonth='10')or(regmonth='09')or(re
gmonth='08')) then
 season:='Fall'
else
 season:='Spring';

stringgrid7.Cells[0,i]:=frm_main.DataSource7.DataSet.FieldValues['Coursecode'];
stringgrid7.Cells[1,i]:=frm_main.DataSource7.DataSet.FieldValues['grade'];
i:=i+1;
shortdateformat:=('yyyy');
label13.Caption:=REGYEAR+' - '+season+' Term';
shortdateformat:=('dd/mm/yyyy');

sumcredit:=sumcredit+frm_main.DataSource7.dataset.FieldValues['credit'];

if frm_main.DataSource7.dataset.FieldValues['grade']='AA' then
 sumgrade:=sumgrade+frm_main.DataSource7.dataset.FieldValues['credit']*4;
if frm_main.DataSource7.dataset.FieldValues['grade']='BA' then
 sumgrade:=sumgrade+frm_main.DataSource7.dataset.FieldValues['credit']*3.5;
if frm_main.DataSource7.dataset.FieldValues['grade']='BB' then
 sumgrade:=sumgrade+frm_main.DataSource7.dataset.FieldValues['credit']*3;
if frm_main.DataSource7.dataset.FieldValues['grade']='CB' then
 sumgrade:=sumgrade+frm_main.DataSource7.dataset.FieldValues['credit']*2.5;
if frm_main.DataSource7.dataset.FieldValues['grade']='CC' then
 sumgrade:=sumgrade+frm_main.DataSource7.dataset.FieldValues['credit']*2;
```

```
frm_main.DataSource7.dataset.FieldValues['grade']='DC' then
sumgrade:=sumgrade+frm_main.DataSource7.dataset.FieldValues['credit']*1.5;
frm_main.DataSource7.dataset.FieldValues['grade']='DD' then
sumgrade:=sumgrade+frm_main.DataSource7.dataset.FieldValues['credit']*1;
frm_main.DataSource7.dataset.FieldValues['grade']='FD' then
sumgrade:=sumgrade+frm_main.DataSource7.dataset.FieldValues['credit']*0.5;
frm_main.DataSource7.dataset.FieldValues['grade']='FF' then
sumgrade:=sumgrade+frm_main.DataSource7.dataset.FieldValues['credit']*0;


frm_main.DataSource7.DataSet.Next;
realdate:=frm_main.DataSource7.DataSet.FieldValues['courseregdate'];
formatdateformat:=('yyyy');
regyear:=datetostr(realdate);
formatdateformat:=('mm');
regmonth:=datetostr(realdate);

end;
if (sumcredit>0)) then
begin
str(sumgrade/sumcredit):3:2,t);
label22.Caption:=t;
end;

end of 7. stringgrid
```

art of 8. stringgrid

```
aldate:=frm_main.DataSource7.DataSet.FieldValues['courseregdate'];
hortdateformat:=('yyyy');
egyear:=datetostr(realdate);
part:=regyear;
hortdateformat:=('mm');
egmonth:=datetostr(realdate);
part:=regmonth;
redit:=0;
grade:=0;

hile not(((abs(strtoint(regmonth)-
mpart)))>2)or(regyear<>ypart)or(frm_main.DataSource7.DataSet.eof))do
egin


 f
egmonth='01')or(regmonth='12')or(regmonth='11')or(regmonth='10')or(regmonth='09')or(re
onth='08')) then
 eason:='Fall'
 lse
 eason:='Spring';

tringgrid8.Cells[0,i]:=frm_main.DataSource7.DataSet.FieldValues['Coursecode'];
tringgrid8.Cells[1,i]:=frm_main.DataSource7.DataSet.FieldValues['grade'];
=i+1;
```

```
shortdateformat:=('yyyy');
label14.Caption:=REGYEAR+'  -  '+season+' Term';
shortdateformat:=('dd/mm/yyyy');

sumcredit:=sumcredit+frm_main.DataSource7.dataset.FieldValues['credit'];

if frm_main.DataSource7.dataset.FieldValues['grade']='AA' then
sumgrade:=sumgrade+frm_main.DataSource7.dataset.FieldValues['credit']*4;
if frm_main.DataSource7.dataset.FieldValues['grade']='BA' then
sumgrade:=sumgrade+frm_main.DataSource7.dataset.FieldValues['credit']*3.5;
if frm_main.DataSource7.dataset.FieldValues['grade']='BB' then
sumgrade:=sumgrade+frm_main.DataSource7.dataset.FieldValues['credit']*3;
if frm_main.DataSource7.dataset.FieldValues['grade']='CB' then
sumgrade:=sumgrade+frm_main.DataSource7.dataset.FieldValues['credit']*2.5;
if frm_main.DataSource7.dataset.FieldValues['grade']='CC' then
sumgrade:=sumgrade+frm_main.DataSource7.dataset.FieldValues['credit']*2;
if frm_main.DataSource7.dataset.FieldValues['grade']='DC' then
sumgrade:=sumgrade+frm_main.DataSource7.dataset.FieldValues['credit']*1.5;
if frm_main.DataSource7.dataset.FieldValues['grade']='DD' then
sumgrade:=sumgrade+frm_main.DataSource7.dataset.FieldValues['credit']*1;
if frm_main.DataSource7.dataset.FieldValues['grade']='FD' then
sumgrade:=sumgrade+frm_main.DataSource7.dataset.FieldValues['credit']*0.5;
if frm_main.DataSource7.dataset.FieldValues['grade']='FF' then
sumgrade:=sumgrade+frm_main.DataSource7.dataset.FieldValues['credit']*0;
```

```
frm_main.DataSource7.DataSet.Next;
realdate:=frm_main.DataSource7.DataSet.FieldValues['courseregdate'];
shortdateformat:=('yyyy');
regyear:=datetostr(realdate);
shortdateformat:=('mm');
regmonth:=datetostr(realdate);

end;

if (sumcredit>0)) then
begin
t:=floattostrf((sumgrade/sumcredit):3:2,t);
label23.Caption:=t;
end;


end of 8. stringgrid


end of 9. stringgrid


realdate:=frm_main.DataSource7.DataSet.FieldValues['courseregdate'];
shortdateformat:=('yyyy');
regyear:=datetostr(realdate);
xpart:=regyear;
shortdateformat:=('mm');
regmonth:=datetostr(realdate);
mpart:=regmonth;
i:=0;
```

```
sumcredit:=0;
sumgrade:=0;

while not(((abs(strtoint(regmonth)-
ypart)))>2)or(regyear<>ypart)or(frm_main.DataSource7.DataSet.eof))do
begin

if
regmonth='01')or(regmonth='12')or(regmonth='11')or(regmonth='10')or(regmonth='09')or(re
gmonth='08')) then
 season:='Fall'
else
 season:='Spring';

stringgrid10.Cells[0,i]:=frm_main.DataSource7.DataSet.FieldValues['Coursecode'];
stringgrid10.Cells[1,i]:=frm_main.DataSource7.DataSet.FieldValues['grade'];
i:=i+1;
shortdateformat:=('yyyy');
label38.Caption:=REGYEAR+'  -  '+season+' Term';
shortdateformat:=('dd/mm/yyyy');

sumcredit:=sumcredit+frm_main.DataSource7.dataset.FieldValues['credit'];

if frm_main.DataSource7.dataset.FieldValues['grade']='AA' then
sumgrade:=sumgrade+frm_main.DataSource7.dataset.FieldValues['credit']*4;
if frm_main.DataSource7.dataset.FieldValues['grade']='BA' then
sumgrade:=sumgrade+frm_main.DataSource7.dataset.FieldValues['credit']*3.5;
```

```
frm_main.DataSource7.dataset.FieldValues['grade']='BB' then
sumgrade:=sumgrade+frm_main.DataSource7.dataset.FieldValues['credit']*3;
frm_main.DataSource7.dataset.FieldValues['grade']='CB' then
sumgrade:=sumgrade+frm_main.DataSource7.dataset.FieldValues['credit']*2.5;
frm_main.DataSource7.dataset.FieldValues['grade']='CC' then
sumgrade:=sumgrade+frm_main.DataSource7.dataset.FieldValues['credit']*2;
frm_main.DataSource7.dataset.FieldValues['grade']='DC' then
sumgrade:=sumgrade+frm_main.DataSource7.dataset.FieldValues['credit']*1.5;
frm_main.DataSource7.dataset.FieldValues['grade']='DD' then
sumgrade:=sumgrade+frm_main.DataSource7.dataset.FieldValues['credit']*1;
frm_main.DataSource7.dataset.FieldValues['grade']='FD' then
sumgrade:=sumgrade+frm_main.DataSource7.dataset.FieldValues['credit']*0.5;
frm_main.DataSource7.dataset.FieldValues['grade']='FF' then
sumgrade:=sumgrade+frm_main.DataSource7.dataset.FieldValues['credit']*0;


frm_main.DataSource7.DataSet.Next;
realdate:=frm_main.DataSource7.DataSet.FieldValues['courseregdate'];
shortdateformat:=('yyyy');
regyear:=datetostr(realdate);
shortdateformat:=('mm');
regmonth:=datetostr(realdate);

end;
if ((sumcredit>0)) then
begin
```

```
grade/sumcredit):3:2,t);
Caption:=t;

stringgrid

stringgrid

=frm_main.DataSource7.DataSet.FieldValues['courseregdate'];
eformat:=('yyyy');
=datetostr(realdate);
=regyear;
eformat:=('mm');
th:=datetostr(realdate);
=regmonth;

edit:=0;
grade:=0;

while not(((abs(strtoint(regmonth)-
part)))>2)or(regyear<>ypart)or(frm_main.DataSource7.DataSet.eof))do
begin

if
nth='01')or(regmonth='12')or(regmonth='11')or(regmonth='10')or(regmonth='09')or(re
h='08')) then
season:='Fall'
```

```
else
season:='Spring';

stringgrid9.Cells[0,i]:=frm_main.DataSource7.DataSet.FieldValues['Coursecode'];
stringgrid9.Cells[1,i]:=frm_main.DataSource7.DataSet.FieldValues['grade'];
i:=i+1;
shortdateformat:=('yyyy');
label33.Caption:=REGYEAR+' - '+season+' Term';
shortdateformat:=('dd/mm/yyyy');

sumcredit:=sumcredit+frm_main.DataSource7.dataset.FieldValues['credit'];

if frm_main.DataSource7.dataset.FieldValues['grade']='AA' then
sumgrade:=sumgrade+frm_main.DataSource7.dataset.FieldValues['credit']*4;
if frm_main.DataSource7.dataset.FieldValues['grade']='BA' then
sumgrade:=sumgrade+frm_main.DataSource7.dataset.FieldValues['credit']*3.5;
if frm_main.DataSource7.dataset.FieldValues['grade']='BB' then
sumgrade:=sumgrade+frm_main.DataSource7.dataset.FieldValues['credit']*3;
if frm_main.DataSource7.dataset.FieldValues['grade']='CB' then
sumgrade:=sumgrade+frm_main.DataSource7.dataset.FieldValues['credit']*2.5;
if frm_main.DataSource7.dataset.FieldValues['grade']='CC' then
sumgrade:=sumgrade+frm_main.DataSource7.dataset.FieldValues['credit']*2;
if frm_main.DataSource7.dataset.FieldValues['grade']='DC' then
sumgrade:=sumgrade+frm_main.DataSource7.dataset.FieldValues['credit']*1.5;
if frm_main.DataSource7.dataset.FieldValues['grade']='DD' then
sumgrade:=sumgrade+frm_main.DataSource7.dataset.FieldValues['credit']*1;
```

```
if frm_main.DataSource7.dataset.FieldValues['grade']='FD' then
sumgrade:=sumgrade+frm_main.DataSource7.dataset.FieldValues['credit']*0.5;
if frm_main.DataSource7.dataset.FieldValues['grade']='FF' then
sumgrade:=sumgrade+frm_main.DataSource7.dataset.FieldValues['credit']*0;


frm_main.DataSource7.DataSet.Next;
realdate:=frm_main.DataSource7.DataSet.FieldValues['courseregdate'];
shortdateformat:=('yyyy');
regyear:=datetostr(realdate);
shortdateformat:=('mm');
regmonth:=datetostr(realdate);

end;
if ((sumcredit>0)) then
begin
str((sumgrade/sumcredit):3:2,t);
label33.Caption:=t;
end;

end of 10. stringgrid

start of 11. stringgrid

realdate:=frm_main.DataSource7.DataSet.FieldValues['courseregdate'];
shortdateformat:=('yyyy');
```

```
=datetostr(realdate);
=regyear;
dateformat:=('mm');
nth:=datetostr(realdate);
=regmonth;

redit:=0;
rade:=0;


e not(((abs(strtoint(regmonth)-
part)))>2)or(regyear<>ypart)or(frm_main.DataSource7.DataSet.eof))do
gin


th='01')or(regmonth='12')or(regmonth='11')or(regmonth='10')or(regmonth='09')or(re
h='08')) then
 ason:='Fall'
se
 ason:='Spring';


inggrid12.Cells[0,i]:=frm_main.DataSource7.DataSet.FieldValues['Coursecode'];
inggrid12.Cells[1,i]:=frm_main.DataSource7.DataSet.FieldValues['grade'];
=i+1;
ortdateformat:=('yyyy');
bel44.Caption:=REGYEAR+' - '+season+' Term';
ortdateformat:=('dd/mm/yyyy');
```

```
sumcredit:=sumcredit+frm_main.DataSource7.dataset.FieldValues['credit'];

if frm_main.DataSource7.dataset.FieldValues['grade']='AA' then
sumgrade:=sumgrade+frm_main.DataSource7.dataset.FieldValues['credit']*4;
if frm_main.DataSource7.dataset.FieldValues['grade']='BA' then
sumgrade:=sumgrade+frm_main.DataSource7.dataset.FieldValues['credit']*3.5;
if frm_main.DataSource7.dataset.FieldValues['grade']='BB' then
sumgrade:=sumgrade+frm_main.DataSource7.dataset.FieldValues['credit']*3;
if frm_main.DataSource7.dataset.FieldValues['grade']='CB' then
sumgrade:=sumgrade+frm_main.DataSource7.dataset.FieldValues['credit']*2.5;
if frm_main.DataSource7.dataset.FieldValues['grade']='CC' then
sumgrade:=sumgrade+frm_main.DataSource7.dataset.FieldValues['credit']*2;
if frm_main.DataSource7.dataset.FieldValues['grade']='DC' then
sumgrade:=sumgrade+frm_main.DataSource7.dataset.FieldValues['credit']*1.5;
if frm_main.DataSource7.dataset.FieldValues['grade']='DD' then
sumgrade:=sumgrade+frm_main.DataSource7.dataset.FieldValues['credit']*1;
if frm_main.DataSource7.dataset.FieldValues['grade']='FD' then
sumgrade:=sumgrade+frm_main.DataSource7.dataset.FieldValues['credit']*0.5;
if frm_main.DataSource7.dataset.FieldValues['grade']='FF' then
sumgrade:=sumgrade+frm_main.DataSource7.dataset.FieldValues['credit']*0;


frm_main.DataSource7.DataSet.Next;
realdate:=frm_main.DataSource7.DataSet.FieldValues['courseregdate'];
shortdateformat:=('yyyy');
regyear:=datetostr(realdate);
```

```
dateformat:=('mm');
month:=datetostr(realdate);

end;
if(sumcredit>0)) then
begin
str(sumgrade/sumcredit):3:2,t);
label42.Caption:=t;
end;

end of 11. stringgrid

start of 12. stringgrid

realdate:=frm_main.DataSource7.DataSet.FieldValues['courseregdate'];
dateformat:=('yyyy');
regyear:=datetostr(realdate);
ypart:=regyear;
dateformat:=('mm');
regmonth:=datetostr(realdate);
mpart:=regmonth;
i:=0;
sumcredit:=0;
sumgrade:=0;

while not(((abs(strtoint(regmonth)-
strtoint(mpart)))>2)or(regyear<>ypart)or(frm_main.DataSource7.DataSet.eof))do
```

```
begin


 if
regmonth='01')or(regmonth='12')or(regmonth='11')or(regmonth='10')or(regmonth='09')or(re
nth='08')) then
  season:='Fall'
 else
  season:='Spring';

 stringgrid11.Cells[0,i]:=frm_main.DataSource7.DataSet.FieldValues['Coursecode'];
 stringgrid11.Cells[1,i]:=frm_main.DataSource7.DataSet.FieldValues['grade'];
 i:=i+1;
 shortdateformat:=('yyyy');
 label41.Caption:=REGYEAR+' - '+season+' Term';
 shortdateformat:=('dd/mm/yyyy');

 sumcredit:=sumcredit+frm_main.DataSource7.dataset.FieldValues['credit'];

 if frm_main.DataSource7.dataset.FieldValues['grade']='AA' then
 sumgrade:=sumgrade+frm_main.DataSource7.dataset.FieldValues['credit']*4;
 if frm_main.DataSource7.dataset.FieldValues['grade']='BA' then
 sumgrade:=sumgrade+frm_main.DataSource7.dataset.FieldValues['credit']*3.5;
 if frm_main.DataSource7.dataset.FieldValues['grade']='BB' then
 sumgrade:=sumgrade+frm_main.DataSource7.dataset.FieldValues['credit']*3;
 if frm_main.DataSource7.dataset.FieldValues['grade']='CB' then
 sumgrade:=sumgrade+frm_main.DataSource7.dataset.FieldValues['credit']*2.5;
```

```
frm_main.DataSource7.dataset.FieldValues['grade']='CC' then
sumgrade:=sumgrade+frm_main.DataSource7.dataset.FieldValues['credit']*2;
frm_main.DataSource7.dataset.FieldValues['grade']='DC' then
sumgrade:=sumgrade+frm_main.DataSource7.dataset.FieldValues['credit']*1.5;
frm_main.DataSource7.dataset.FieldValues['grade']='DD' then
sumgrade:=sumgrade+frm_main.DataSource7.dataset.FieldValues['credit']*1;
frm_main.DataSource7.dataset.FieldValues['grade']='FD' then
sumgrade:=sumgrade+frm_main.DataSource7.dataset.FieldValues['credit']*0.5;
frm_main.DataSource7.dataset.FieldValues['grade']='FF' then
sumgrade:=sumgrade+frm_main.DataSource7.dataset.FieldValues['credit']*0;


frm_main.DataSource7.DataSet.Next;
realdate:=frm_main.DataSource7.DataSet.FieldValues['courseregdate'];
formatdateformat:=('yyyy');
regyear:=datetostr(realdate);
formatdateformat:=('mm');
regmonth:=datetostr(realdate);

end;
if (sumcredit>0)) then
begin
str((sumgrade/sumcredit):3:2,t);
label39.Caption:=t;
end;
```

```
end of 12. stringgrid

start of cgpa
 sumgencredit:=0;
 sumgengrade:=0;


 with frm_main.ADOQuery4 do
 begin
  sql.Clear;
  sql.add('Select t.GradeDate,t.studentId,t.grade,t.courseId,c.credit from transcript as t
inner join course as c on t.courseId=c.courseId where t.studentId="'+edit1.Text+'" and t.grade
is not null');
  open;
 end;


frm_main.DataSource10.DataSet.First;


 while not(frm_main.DataSource10.DataSet.Eof) do
 begin
  with frm_main.DataSource10 do
  begin
   varcorId:=dataset.fieldvalues['courseId'];
   label46.caption:=inttostr(varcorId);
   with frm_main.ADOQuery6 do
   begin
    sql.Clear;
```

```
add('Select t.GradeDate,t.studentId,t.grade,t.courseID,c.credit from transcript as t
course as c on t.courseID=c.courseID where t.studentId='"+edit1.Text+'" and
'+label46.caption+' order by gradedate desc');

open;

end;


frm_main.DataSource12.DataSet.First;


if
FieldValues['gradedate']=frm_main.DataSource12.DataSet.FieldValues['gradedate']


begin

sumgencredit:=sumgencredit+dataset.FieldValues['credit'];


if dataset.FieldValues['grade']='AA' then

sumgengrade:=sumgengrade+dataset.FieldValues['credit']*4;

if dataset.FieldValues['grade']='BA' then

sumgengrade:=sumgengrade+dataset.FieldValues['credit']*3.5;

if dataset.FieldValues['grade']='BB' then

sumgengrade:=sumgengrade+dataset.FieldValues['credit']*3;

if dataset.FieldValues['grade']='CB' then

sumgengrade:=sumgengrade+dataset.FieldValues['credit']*2.5;

if dataset.FieldValues['grade']='CC' then

sumgengrade:=sumgengrade+dataset.FieldValues['credit']*2;

if dataset.FieldValues['grade']='DC' then

sumgengrade:=sumgengrade+dataset.FieldValues['credit']*1.5;

if dataset.FieldValues['grade']='DD' then

sumgengrade:=sumgengrade+dataset.FieldValues['credit']*1;
```

```
if dataset.FieldValues['grade']='FD' then
 sumgengrade:=sumgengrade+dataset.FieldValues['credit']*0.5;
if dataset.FieldValues['grade']='FF' then
 sumgengrade:=sumgengrade+dataset.FieldValues['credit']*0;
dataset.Next;
end
else
begin
 dataset.Next;
end;    // end of if

end;    //end of with

end;    //end of while

if ((sumgencredit>0) and (sumgengrade>0)) then
begin
 str((sumgengrade/sumgencredit):3:2,t);
 label15.Caption:=t;
end;



shortdateformat:=('dd/mm/yyyy');

end of cgpa....
```

```
end

end

else

begin

messagedlg('Please Enter student number',mtwarning,[mbok],0);

clearing(sender);

exit;

end

*****************************************************/}


end


procedure Tfrm_trans.Edit1KeyPress(Sender: TObject; var Key: Char);

begin

if(key in ['0'..'9'])) then

key:=#0;

end


procedure Tfrm_trans.Button2Click(Sender: TObject);

var i,j:integer;

begin
```

```
frm_trans.Print;
sender);

Tfrm_trans.Button5Click(Sender: TObject);

frm_main.show;
frm_trans.hide;

Tfrm_trans.Button4Click(Sender: TObject);

sender);
SetFocus;

Tfrm_trans.FormShow(Sender: TObject);

sender);
```

## UNIT 15

```
Unit15;

face
```

```
uses
  Windows, Messages, SysUtils, Variants, Classes, Graphics, Controls, Forms,
  Dialogs, StdCtrls, DBCtrls, Grids, DBGrids, DB, ADODB;

type
  Tfrm_semend = class(TForm)
  Edit1: TEdit;
  Button1: TButton;
  Label1: TLabel;
  Label2: TLabel;
  Label3: TLabel;
  Label4: TLabel;
  Label5: TLabel;
  Label6: TLabel;
  Label7: TLabel;
  Label8: TLabel;
  ComboBox1: TComboBox;
  ComboBox2: TComboBox;
  ComboBox3: TComboBox;
  ComboBox4: TComboBox;
  ComboBox5: TComboBox;
  ComboBox6: TComboBox;
  ComboBox7: TComboBox;
  Button2: TButton;
  Button3: TButton;
  Button4: TButton;
```

```
Button5: TButton;
Edit2: TEdit;
Edit3: TEdit;
Edit4: TEdit;
Label9: TLabel;
Label10: TLabel;
Label11: TLabel;
Label12: TLabel;
ADOQuery1: TADOQuery;
Label13: TLabel;
Label14: TLabel;
Label15: TLabel;
Label16: TLabel;
Label17: TLabel;
Label18: TLabel;
Label19: TLabel;
procedure FormShow(Sender: TObject);
procedure Button1Click(Sender: TObject);
procedure FormClose(Sender: TObject; var Action: TCloseAction);
procedure Button4Click(Sender: TObject);
procedure Button5Click(Sender: TObject);
procedure Button3Click(Sender: TObject);
procedure Button2Click(Sender: TObject);
private
{ Private declarations }
public
```

Public declarations }

end;

var

frm_semend: Tfrm_semend;

implementation

uses Unit1,unit26, Unit22, Unit28;

{$R *.dfm}

procedure Tfrm_semend.FormShow(Sender: TObject);

begin

Edit1.SetFocus;

Edit1.Clear;

Edit2.Clear;

Edit3.Clear;

Edit4.Clear;

Edit1.ReadOnly:=false;

Label12.Caption:='';

Label2.Visible:=false;

Label2.Caption:='';

Label3.Visible:=false;

```
abel3.Caption:='';
abel4.Visible:=false;
abel4.Caption:='';
abel5.Visible:=false;
abel5.Caption:='';
abel6.Visible:=false;
abel6.Caption:='';
abel7.Visible:=false;
abel7.Caption:='';
abel8.Visible:=false;
abel8.Caption:='';

ombobox1.Text:='';
ombobox1.Visible:=false;
ombobox2.Text:='';
ombobox2.Visible:=false;
ombobox3.Text:='';
ombobox3.Visible:=false;
ombobox4.Text:='';
ombobox4.Visible:=false;
ombobox5.Text:='';
ombobox5.Visible:=false;
ombobox6.Text:='';
ombobox6.Visible:=false;
ombobox7.Text:='';
ombobox7.Visible:=false;
```

```
ure Tfrm_semend.Button1Click(Sender: TObject);
eger;
:integer;
:string;
MonthNames:string;

ext<>'' then

DateFormat := 'm/dd/yyyy';
gth(edit1.Text)=8 then

frm_main.ADOquery1 do

clear;
Add('Select * from student where StudentId='''+edit1.text+'''');
pen;

frm_main.DataSource7.DataSet.RecordCount<>0 then

edit1.ReadOnly:=true;
edit2.Text:= frm_main.DataSource7.DataSet.FieldValues['name'];
edit3.Text:= frm_main.DataSource7.DataSet.FieldValues['surname'];
```

```
edit4.Text:= frm_main.DataSource7.DataSet.FieldValues['departmentID'];

// term caption
shortdateformat:=('mm');
ShortMonthNames:=datetostr(date);
 label12.Caption:=ShortMonthNames;
if
ShortMonthNames='01')or(ShortMonthNames='12')or(ShortMonthNames='11')or(ShortMo
thNames='10')or(ShortMonthNames='09')or(ShortMonthNames='08')) then
 mpart:='Fall'
else
 mpart:='Spring';

shortdateformat:=('yyyy');
 ypart:=date;
label12.Caption:=datetostr(date)+'  -  '+mpart+' Term registration ';

shortdateformat:=('dd/mm/yyyy');

with frm_main.ADOQuery2 do      //Find if already registered
begin
 sql.Clear;
 sql.Add('SELECT  c.coursecode,c.courseId,t.grade,t.gradedate,t.updateuser FROM
Transcript as t inner join course as c on t.courseId=c.courseId  where
StudentId="'+edit1.Text+'" and t.grade is null and t.dropdate is null');
 open;
end;
```

```
if frm_main.DataSource8.DataSet.RecordCount=0 then
 begin
 messagedlg('This Student Has Not Been Registered Yet, Use Registration Menu To
Course Registration ..!',mtwarning,[mbok],0);
 Edit1.Clear;
 Edit2.Clear;
 Edit3.Clear;
 Edit4.Clear;
 edit1.ReadOnly:=False;
 exit;
 end
 else
 begin
 with frm_main.DataSource8 do
 begin
  dataset.First;
  i:=1;
  if not(dataset.Eof) then
   begin
    label2.Visible:=true;
    combobox1.Visible:=true;
    label2.Caption:=dataset.FieldValues['Coursecode'];
    label13.Caption:=dataset.FieldValues['CourseId'];
    dataset.Next;
   end;
  if not(dataset.Eof) then
   begin
```

```
  label3.Visible:=true;
  combobox2.Visible:=true;
  label3.Caption:=dataset.FieldValues['Coursecode'];
  label14.Caption:=dataset.FieldValues['CourseId'];
  dataset.Next;
 end;


if not(dataset.Eof) then
 begin
 label4.Visible:=true;
 combobox3.Visible:=true;
 label4.Caption:=dataset.FieldValues['Coursecode'];
 label15.Caption:=dataset.FieldValues['CourseId'];
 dataset.Next;
end;

if not(dataset.Eof) then
 begin
 label5.Visible:=true;
 combobox4.Visible:=true;
 label5.Caption:=dataset.FieldValues['Coursecode'];
 label16.Caption:=dataset.FieldValues['CourseId'];
 dataset.Next;
end;
```

```
if not(dataset.Eof) then
 begin
 label6.Visible:=true;
 combobox5.Visible:=true;
 label6.Caption:=dataset.FieldValues['Coursecode'];
 label17.Caption:=dataset.FieldValues['CourseId'];
 dataset.Next;
 end;

if not(dataset.Eof) then
 begin
 label7.Visible:=true;
 combobox6.Visible:=true;
 label7.Caption:=dataset.FieldValues['Coursecode'];
 label18.Caption:=dataset.FieldValues['CourseId'];
 dataset.Next;
 end;

if not(dataset.Eof) then
 begin
 label8.Visible:=true;
 combobox7.Visible:=true;
 label8.Caption:=dataset.FieldValues['Coursecode'];
 label19.Caption:=dataset.FieldValues['CourseId'];
end;
```

```
    end;
  end;

  with frm_main.ADOquery1 do
  begin
   sql.clear;
   sql.Add('Select * from student where StudentId="'+edit1.text+'"');
   open;
  end;

 end
else
begin
 messagedlg('Student Number Is Not Found !',mtwarning,[mbok],0);
```

```
  Edit1.SetFocus;
  exit
 end
 else
 begin
  messagedlg('Student Number Must Be 8 Digits !',mtwarning,[mbok],0);
  Edit1.SetFocus;
  exit
 end
end
else
begin
 messagedlg('Please Enter Number For Student To Update ! ',mtwarning,[mbok],0);
 Edit1.Clear;
 Edit1.SetFocus;
 exit
end


procedure Tfrm_semend.FormClose(Sender: TObject; var Action: TCloseAction);
begin
 frm_beginsemester.show;
end


procedure Tfrm_semend.Button4Click(Sender: TObject);
begin
 frm_semend.Close;
```

```
re Tfrm_semend.Button5Click(Sender: TObject);

.Show;
end.Hide;


re Tfrm_semend.Button3Click(Sender: TObject);

end.Show;


re Tfrm_semend.Button2Click(Sender: TObject);


frm_main.ADOQuery2 do          //Find if already registered

.Clear;
.Add('SELECT t.grade,t.studentId,t.gradedate,t.updateuser FROM transcript as t inner
course as c on t.courseId=c.courseId  where t.StudentId="'+edit1.Text+'" and
rsecode="'+label1.caption+'" and t.grade is null and t.dropdate is null');

en;


  }


frm_main.ADOQuery2 do          //Find if already registered
```

```
  Clear;
  Add('SELECT top 1 grade,studentId,gradedate,updateuser,courseId FROM transcript
  studentId="'+edit1.Text+'" and courseId=(SELECT courseId from course where
  coursecode="'+label2.Caption+'") order by courseregdate desc');
  Open;
end;

with frm_main.DataSource8 do
begin
DataSet.First;
Label19.caption:=inttostr(DataSet.RecordCount);
if (DataSet.RecordCount=0)and((combobox1.text='')or(combobox1.Text=null))) then
begin
DataSet.Edit;
DataSet.fieldvalues['grade']:=combobox1.Text;
DataSet.FieldValues['gradedate']:=date;
DataSet.FieldValues['updateuser']:=username;
DataSet.FieldValues['StudentId']:=edit1.Text;
DataSet.FieldValues['courseId']:=strtoint(label13.Caption);
DataSet.post;

end;
end;

with frm_main.ADOQuery2 do        //Find if already registered
```

```
begin
 sql.Clear;
 sql.Add('SELECT top 1 grade,studentId,gradedate,updateuser,courseId FROM transcript
where studentId="'+edit1.Text+'" and courseId=(SELECT courseId from course where
coursecode="'+label3.Caption+'") order by courseregdate desc');
 open;
end;



with frm_main.DataSource8 do
begin
 dataset.First;


 if not((DataSet.RecordCount=0)and((combobox2.text='')or(combobox2.Text=null))) then
  begin
   dataset.Edit;
   DataSet.FieldValues['grade']:=combobox2.Text;
   DataSet.FieldValues['gradedate']:=date;
   DataSet.FieldValues['updateuser']:=username;
   DataSet.FieldValues['StudentId']:=edit1.Text;
   DataSet.FieldValues['courseId']:=strtoint(label14.Caption);
   DataSet.post;
   //dataset.Next;
  end;
end;


with frm_main.ADOQuery2 do          //Find if already registered
```

```
begin
 sql.Clear;
 sql.Add('SELECT top 1 grade,studentId,gradedate,updateuser,courseId FROM transcript
 where studentId="'+edit1.Text+'" and courseId=(SELECT courseId from course where
 coursecode="'+label4.Caption+'") order by courseregdate desc');
 open;
 end;


with frm_main.DataSource8 do
begin
 dataset.First;

if not((DataSet.RecordCount=0)and((combobox3.text='')or(combobox3.Text=null))) then
 begin
  dataset.Edit;
  DataSet.FieldValues['grade']:=combobox4.Text;
  DataSet.FieldValues['gradedate']:=date;
  DataSet.FieldValues['updateuser']:=username;
  DataSet.FieldValues['StudentId']:=edit1.Text;
  DataSet.FieldValues['courseId']:=strtoint(label15.Caption);
  DataSet.post;
  dataset.Next;
 end;
end;


with frm_main.ADOQuery2 do        //Find if already registered
```

```
begin
 sql.Clear;
 sql.Add('SELECT top 1 grade,studentId,gradedate,updateuser,courseId FROM transcript
where studentId="'+edit1.Text+'" and courseId=(SELECT courseId from course where
coursecode="'+label5.Caption+'") order by courseregdate desc');
 open;
end;


with frm_main.DataSource8 do
begin
// dataset.First;

if not((DataSet.RecordCount=0)and((combobox4.text='')or(combobox4.Text=null))) then
begin
 dataset.Edit;
 DataSet.FieldValues['grade']:=combobox4.Text;
 DataSet.FieldValues['gradedate']:=date;
 DataSet.FieldValues['updateuser']:=username;
 DataSet.FieldValues['StudentId']:=edit1.Text;
 DataSet.FieldValues['courseId']:=strtoint(label16.Caption);
 DataSet.post;
// dataset.Next;
 end;
end;


with frm_main.ADOQuery2 do        //Find if already registered
```

```
begin
 sql.Clear;
 sql.Add('SELECT top 1 grade,studentId,gradedate,updateuser,courseId FROM transcript
where studentId="'+edit1.Text+'" and courseId=(SELECT courseId from course where
coursecode="'+label6.Caption+'") order by courseregdate desc');
 open;
end;


frm_main.ADOQuery2.DataSource.DataSet.FieldValues
with frm_main.DataSource8 do
begin
  dataset.First;

if not((DataSet.RecordCount=0)and((combobox5.text='')or(combobox5.Text=null))) then
 begin
  dataset.Edit;
  DataSet.FieldValues['grade']:=combobox5.Text;
  DataSet.FieldValues['gradedate']:=date;
  DataSet.FieldValues['updateuser']:=username;
  DataSet.FieldValues['StudentId']:=edit1.Text;
  DataSet.FieldValues['courseId']:=strtoint(label17.Caption);
  DataSet.post;
    dataset.Next;
 end;
end;
```

```
with frm_main.ADOQuery2 do          //Find if already registered
begin
  SQL.Clear;
  SQL.Add('SELECT top 1 grade,studentId,gradedate,updateuser,courseId FROM transcript
  where studentId="'+edit1.Text+'" and courseId=(SELECT courseId from course where
  courseCode="'+label7.Caption+'") order by courseregdate desc');
  Open;
end;


with frm_main.DataSource8 do
begin
  dataset.First;
  label19.caption:=inttostr(DataSet.RecordCount);
  if not((DataSet.RecordCount=0)and((combobox6.text='')or(combobox6.Text=null))) then
  begin
   dataset.Edit;
   DataSet.fieldvalues['grade']:=combobox6.Text;
   DataSet.FieldValues['gradedate']:=date;
   DataSet.FieldValues['updateuser']:=username;
   DataSet.FieldValues['StudentId']:=edit1.Text;
   DataSet.FieldValues['courseId']:=strtoint(label18.Caption);
   DataSet.post;

  end;
end;
```

```
with frm_main.ADOQuery2 do        //Find if already registered
begin
sql.Clear;
sql.Add('SELECT top 1 grade,studentId,gradedate,updateuser,courseId FROM transcript
where studentId="'+edit1.Text+'" and courseId=(SELECT courseId from course where
coursecode="'+label8.Caption+'") order by courseregdate desc');
open;
end;


with frm_main.DataSource8 do
begin
dataset.First;
label19.caption:=inttostr(DataSet.RecordCount);
if not((DataSet.RecordCount=0)and((combobox7.text='')or(combobox7.Text=null))) then
begin
dataset.Edit;
DataSet.fieldvalues['grade']:=combobox7.Text;
DataSet.FieldValues['gradedate']:=date;
DataSet.FieldValues['updateuser']:=username;
DataSet.FieldValues['StudentId']:=edit1.Text;
DataSet.FieldValues['courseId']:=strtoint(label19.Caption);
DataSet.post;

end;
end;
```

```
with frm_main.ADOQuery2 do          //Find if already registered
begin
sql.Clear;
sql.Add('UPDATE transcript ()');//SELECT top 1
studentId,gradedate,updateuser,courseId FROM transcript where
studentId="'+edit1.Text+'" and courseId=(SELECT courseId from course where
coursecode="'+label6.Caption+'") order by courseregdate desc');
open;
end;
with frm_main.ADOQuery2 do          //Find if already registered
begin
sql.Clear;
sql.Add('SELECT t.grade,t.gradedate,t.updateuser,t.studentId FROM transcript as t inner
join course as c on t.courseId=c.courseId  where t.StudentId="'+edit1.Text+'" and
coursecode="'+label1.caption+'" and t.grade is null and t.dropdate is null');
open;
end;


with frm_main.DataSource8 do
begin
dataset.Edit;
dataset.First;
if not((dataset.Eof)and(combobox6.text<>'')and(combobox6.Visible)) then
```

```
begin
 dataset.FieldValues['grade']:=combobox6.Text;
 dataset.FieldValues['gradedate']:=date;
 dataset.FieldValues['updateuser']:=username;
 dataset.FieldValues['StudentId']:=edit1.Text;
 dataset.post;
 dataset.Next;
end;
end;

with frm_main.ADOQuery2 do        //Find if already registered
begin
sql.Clear;
sql.Add('SELECT t.grade,t.gradedate,t.updateuser,t.StudentId FROM transcript as t inner
course as c on t.courseId=c.courseId  where t.StudentId="'+edit1.Text+'" and
coursecode="'+label1.caption+'" and t.grade is null and t.dropdate is null');
open;
end;


with frm_main.DataSource8 do
begin
dataset.Edit;
dataset.First;
if not((dataset.Eof)and(combobox7.text<>")and(combobox7.Visible)) then
begin
 dataset.FieldValues['grade']:=combobox7.Text;
```

```
dataset.FieldValues['gradedate']:=date;
dataset.FieldValues['updateuser']:=username;
dataset.FieldValues['StudentId']:=edit1.Text;
dataset.post;
 dataset.Next;
end;
end;


messagedlg('Student Grades Are Recorded Successfully ..',mtinformation,[mbok],0);
Edit1.Clear;
Edit2.Clear;
Edit3.Clear;
Edit4.Clear;
edit1.ReadOnly:=False;
combobox1.Text:='';
combobox2.Text:='';
combobox3.Text:='';
combobox4.Text:='';
combobox5.Text:='';
combobox6.Text:='';
combobox7.Text:='';


end;


end.
```

## NIT 16

```
nit Unit16;

nterface

ses
 Windows, Messages, SysUtils, Variants, Classes, Graphics, Controls, Forms,
 Dialogs, StdCtrls;

ype
 TRep_main = class(TForm)
  Button1: TButton;
  Button2: TButton;
  Button3: TButton;
  Button4: TButton;
  Button5: TButton;
  Button6: TButton;
  Button7: TButton;
  Button8: TButton;
  Button9: TButton;
  Button10: TButton;
  procedure Button1Click(Sender: TObject);
  procedure Button2Click(Sender: TObject);
  procedure Button3Click(Sender: TObject);
  procedure Button4Click(Sender: TObject);
  procedure Button5Click(Sender: TObject);
  procedure Button6Click(Sender: TObject);
```

```
  procedure Button7Click(Sender: TObject);
  procedure Button8Click(Sender: TObject);
  procedure Button9Click(Sender: TObject);
  procedure FormClose(Sender: TObject; var Action: TCloseAction);
  procedure Button10Click(Sender: TObject);
private
  { Private declarations }
public
  { Public declarations }
end;

var
Rep_main: TRep_main;

implementation

uses Unit17, Unit18, Unit19, Unit20, Unit21, Unit23, Unit24, Unit25, Unit1,
 Unit14;

{$R *.dfm}

procedure TRep_main.Button1Click(Sender: TObject);
begin

Rep_all_depts.show;
Rep_main.Hide;
```

```
end;

procedure TRep_main.Button2Click(Sender: TObject);
begin
 rep_main.close;
end;


procedure TRep_main.Button3Click(Sender: TObject);
begin


rep_all_teacs.show;


end;


procedure TRep_main.Button4Click(Sender: TObject);
begin
Rep_teacs_dept.show;
end;


procedure TRep_main.Button5Click(Sender: TObject);
begin
Rep_dept_teacs.show;
end;


procedure TRep_main.Button6Click(Sender: TObject);
begin
```

```
Rep_all_courses.show;
end;

procedure TRep_main.Button7Click(Sender: TObject);
begin
Rep_dept_cor.show;
end;

procedure TRep_main.Button8Click(Sender: TObject);
begin
Rep_all_stu.show;
end;

procedure TRep_main.Button9Click(Sender: TObject);
begin
Rep_dept_stu.show;
end;

procedure TRep_main.FormClose(Sender: TObject; var Action: TCloseAction);
begin
frm_main.show;
end;

procedure TRep_main.Button10Click(Sender: TObject);
begin
frm_trans.show;
```

p_main.hide;

nd;

nd.

## NIT 17

t Unit17;

terface

es

ndows, Messages, SysUtils, Variants, Classes, Graphics, Controls, Forms,

alogs, DB, ADODB, Grids, DBGrids, StdCtrls;

pe

Rep_all_depts = class(TForm)

DBGrid1: TDBGrid;

ADOQuery1: TADOQuery;

DataSource1: TDataSource;

Label1: TLabel;

Button1: TButton;

Button2: TButton;

procedure FormCreate(Sender: TObject);

procedure Button1Click(Sender: TObject);

procedure Button2Click(Sender: TObject);

procedure FormClose(Sender: TObject; var Action: TCloseAction);

vate

{ Private declarations }

```
public
  { Public declarations }
end;

var
  Rep_all_depts: TRep_all_depts;

implementation

uses Unit16;

{$R *.dfm}

procedure TRep_all_depts.FormCreate(Sender: TObject);
begin
   with ADOQuery1 do
   begin
    SQL.Clear;
    sql.add('SELECT * FROM department');
    open;
   end;

end;

procedure TRep_all_depts.Button1Click(Sender: TObject);
begin
```

```
Rep_all_depts.close;
end;

procedure TRep_all_depts.Button2Click(Sender: TObject);
begin
Rep_all_depts.Print;
end;

procedure TRep_all_depts.FormClose(Sender: TObject;
 var Action: TCloseAction);
begin
 rep_main.show;
end;

end.
```

## UNIT 18

```
unit Unit18;

interface

uses
 Windows, Messages, SysUtils, Variants, Classes, Graphics, Controls, Forms,
 Dialogs, StdCtrls, Grids, DBGrids, DB, ADODB;

type
 TRep_all_teacs = class(TForm)
```

```
  ADOQuery1: TADOQuery;
  DataSource1: TDataSource;
  Label1: TLabel;
  DBGrid1: TDBGrid;
  Button1: TButton;
  Button2: TButton;
  procedure Button2Click(Sender: TObject);
  procedure Button1Click(Sender: TObject);
  procedure FormCreate(Sender: TObject);
  procedure FormClose(Sender: TObject; var Action: TCloseAction);
private
  { Private declarations }
public
  { Public declarations }
end;

var
 Rep_all_teacs: TRep_all_teacs;

implementation

uses Unit16;

{$R *.dfm}

procedure TRep_all_teacs.Button2Click(Sender: TObject);
```

```
_all_teacs.close;


ure TRep_all_teacs.Button1Click(Sender: TObject);
_all_teacs.Print;

dure TRep_all_teacs.FormCreate(Sender: TObject);
  with ADOQuery1 do
  begin
   SQL.Clear;
   sql.add('SELECT * FROM teacher');
   open;
  end;

;

cedure TRep_all_teacs.FormClose(Sender: TObject;
 var Action: TCloseAction);
 rep_main.show;
```

NIT 19

Unit19;

erface

Windows, Messages, SysUtils, Variants, Classes, Graphics, Controls, Forms,

Dialogs, StdCtrls, Grids, DBGrids, DB, ADODB;

pe

TRep_teacs_dept = class(TForm)

ADOQuery1: TADOQuery;

DataSource1: TDataSource;

DBGrid1: TDBGrid;

Button1: TButton;

Button2: TButton;

Label1: TLabel;

procedure FormCreate(Sender: TObject);

procedure Button2Click(Sender: TObject);

private

{ Private declarations }

public

{ Public declarations }

end;

```
var
  Rep_teacs_dept: TRep_teacs_dept;

implementation

{$R *.dfm}

procedure TRep_teacs_dept.FormCreate(Sender: TObject);
begin
    with ADOQuery1 do
    begin
     SQL.Clear;
     sql.add('SELECT tname,departmentId FROM teacher order by departmentId');
     open;
    end;

end;

procedure TRep_teacs_dept.Button2Click(Sender: TObject);
begin
Rep_teacs_dept.Free;
end;

end.
```

## UNIT 20

```
unit Unit20;
```

```
nterface

es
Windows, Messages, SysUtils, Variants, Classes, Graphics, Controls, Forms,
Dialogs, StdCtrls, DBCtrls, DB, ADODB, Grids, DBGrids;

pe
TRep_dept_teacs = class(TForm)
 ADOQuery1: TADOQuery;
 DataSource1: TDataSource;
 Label1: TLabel;
 Button1: TButton;
 Button2: TButton;
 DBGrid1: TDBGrid;
 DataSource2: TDataSource;
 ADOQuery2: TADOQuery;
 ComboBox1: TComboBox;
 Label2: TLabel;
 procedure FormCreate(Sender: TObject);
 procedure Button1Click(Sender: TObject);

 procedure ComboBox1Click(Sender: TObject);
private
 { Private declarations }
public
```

```
  { Public declarations }
end;

var
  Rep_dept_teacs: TRep_dept_teacs;

implementation

uses unit26;
{$R *.dfm}

procedure TRep_dept_teacs.FormCreate(Sender: TObject);
begin

 with ADOQuery1 do
  begin
   SQL.Clear;
   sql.add('SELECT departmentId FROM department order by departmentId');
   open;
  end;

DataSource1.DataSet.First;
while not (DataSource1.DataSet.Eof) do
begin
 combobox1.Items.add(DataSource1.DataSet.FieldValues['DepartmentID']);
```

```
DataSource1.DataSet.Next;
end;

end;

procedure TRep_dept_teacs.Button1Click(Sender: TObject);
begin
Rep_dept_teacs.Close;
end;

procedure TRep_dept_teacs.ComboBox1Click(Sender: TObject);
begin

  with ADOQuery2 do
   begin
    SQL.Clear;
    sql.add('SELECT Tname,Tsurname FROM teacher where
departmentID="'+combobox1.Text+'" order by tname,tsurname');
    open;
   end;
```

**NIT 21**

Unit21;

erface

es

Windows, Messages, SysUtils, Variants, Classes, Graphics, Controls, Forms,

Dialogs, StdCtrls, Grids, DBGrids, DB, ADODB;

pe

TRep_all_courses = class(TForm)

ADOQuery1: TADOQuery;

DataSource1: TDataSource;

DBGrid1: TDBGrid;

Button1: TButton;

Button2: TButton;

Label1: TLabel;

procedure Button2Click(Sender: TObject);

procedure Button1Click(Sender: TObject);

procedure FormCreate(Sender: TObject);

procedure FormClose(Sender: TObject; var Action: TCloseAction);

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

var
Rep_all_courses: TRep_all_courses;

implementation

uses Unit16,unit26;

{$R *.dfm}

procedure TRep_all_courses.Button2Click(Sender: TObject);
begin
 Rep_all_courses.Close;
end;

procedure TRep_all_courses.Button1Click(Sender: TObject);
begin
Rep_all_courses.Print;
end;

procedure TRep_all_courses.FormCreate(Sender: TObject);
```

```
egin
  with ADOQuery1 do
  begin
   SQL.Clear;
   sql.add('SELECT * FROM course order by coursename ');
   open;
  end;


nd;


rocedure TRep_all_courses.FormClose(Sender: TObject;
ar Action: TCloseAction);
egin
 rep_main.show;
nd;


nd.
```

**NIT 22**

```
nit Unit22;


nterface


ses
 Windows, Messages, SysUtils, Variants, Classes, Graphics, Controls, Forms,
 Dialogs, StdCtrls, ExtCtrls, strutils, ExtDlgs;


ype
```

```
Tfrm_correg = class(TForm)
Edit1: TEdit;
Button1: TButton;
Edit2: TEdit;
Edit3: TEdit;
Edit4: TEdit;
Panel1: TPanel;
Label1: TLabel;
Label2: TLabel;
Label3: TLabel;
Label4: TLabel;
Panel2: TPanel;
ListBox1: TListBox;
ListBox2: TListBox;
Label5: TLabel;
Label6: TLabel;
Button2: TButton;
Button3: TButton;
Label7: TLabel;
Label8: TLabel;
Edit5: TEdit;
Edit6: TEdit;
Button4: TButton;
Button5: TButton;
Button6: TButton;
Label9: TLabel;
```

```
    Button7: TButton;
    Label10: TLabel;
    Label11: TLabel;
    Label12: TLabel;
    Label13: TLabel;
    Label14: TLabel;
    Label15: TLabel;
    Label16: TLabel;
    procedure Button6Click(Sender: TObject);
    procedure FormClose(Sender: TObject; var Action: TCloseAction);
    procedure Button1Click(Sender: TObject);
    procedure Button2Click(Sender: TObject);
    procedure FormShow(Sender: TObject);
    procedure Button3Click(Sender: TObject);
    procedure Button4Click(Sender: TObject);
    procedure Button7Click(Sender: TObject);
  private
    { Private declarations }
  public
    { Public declarations }
  end;

var
  frm_correg: Tfrm_correg;

implementation
```

```
Unit1, ADODB,unit26, Unit28;

*.dfm}

procedure Tfrm_correg.Button6Click(Sender: TObject);
begin
frm_main.show;
frm_correg.hide;
end;

procedure Tfrm_correg.FormClose(Sender: TObject; var Action: TCloseAction);
begin
 frm_main.Show;
end;

procedure Tfrm_correg.Button1Click(Sender: TObject);
var sumcredit:integer;
var sumgrade,gpa:real;
var sumgencredit:integer;
var sumgengrade,cgpa:real;
var cid:integer;

 var ypart:integer;
 var mpart,t:string;
 var ShortMonthNames:string;
```

```
begin
if edit1.text<>null then
begin
with frm_main.ADOQuery1 do
begin
sql.Clear;
sql.Add('SELECT  * FROM student WHERE StudentId="'+edit1.Text+'"');
open;
end;
if not(frm_main.DataSource7.DataSet.eof) then
begin
edit2.Text:=frm_main.DataSource7.DataSet.FieldValues['name'];
edit3.Text:=frm_main.DataSource7.DataSet.FieldValues['surname'];
edit4.Text:=frm_main.DataSource7.DataSet.FieldValues['departmentID'];
if frm_main.DataSource7.DataSet.FieldValues['IsGraduated']=true then
begin
messagedlg('This Student Has Been Graduated... Can Not Be Registered
Anymore!',mtinformation,[mbok],0);
Edit1.Clear;
Edit2.Clear;
Edit3.Clear;
Edit4.Clear;
Edit5.Clear;
Edit6.Clear;
Listbox1.Items.Clear;
```

```
    listbox2.Items.Clear;
    edit1.ReadOnly:=false;
    edit1.SetFocus;
    exit;
    end;

   edit1.ReadOnly:=true;

   shortdateformat:=('mm');
   ShortMonthNames:=datetostr(date);
   label12.Caption:=ShortMonthNames;
   if
ShortMonthNames='01')or(ShortMonthNames='12')or(ShortMonthNames='11')or(ShortMo
thNames='10')or(ShortMonthNames='09')or(ShortMonthNames='08')) then
   mpart:='Fall'
  else
   mpart:='Spring';

  shortdateformat:=('yyyy');
  ypart:=date;
  label9.Caption:=datetostr(date)+'  -  '+mpart+' Term registration ';

  shortdateformat:=('dd/mm/yyyy');

// To calculate gpa.....

sumcredit:=0;
```

```
sumgrade:=0;
gpa:=0;
sumgencredit:=0;
sumgengrade:=0;
cgpa:=0;

with frm_main.ADOQuery4 do
begin
sql.Clear;
sql.add('Select t.GradeDate,t.studentId,t.grade,t.courseID,c.credit from transcript as t
join course as c on t.courseID=c.courseID where t.studentId="'+edit1.Text+'" and
grade is not null');
open;
end;

with frm_main.ADOQuery5 do
begin
sql.Clear;
sql.add('Select MAX(Gradedate) as maxdate from transcript WHERE
StudentId="'+edit1.text+'"');
open;
end;

frm_main.DataSource10.DataSet.First;
if not(frm_main.DataSource11.DataSet.Eof) then
begin
while not(frm_main.DataSource10.DataSet.Eof) do
begin
```

```
with frm_main.DataSource10 do
begin
 if
set.FieldValues['GradeDate']>=frm_main.DataSource11.DataSet.FieldValues['maxdate'])
=
  begin
   sumcredit:=sumcredit+dataset.FieldValues['credit'];

    if dataset.FieldValues['grade']='AA' then
     sumgrade:=sumgrade+dataset.FieldValues['credit']*4;
    if dataset.FieldValues['grade']='BA' then
     sumgrade:=sumgrade+dataset.FieldValues['credit']*3.5;
    if dataset.FieldValues['grade']='BB' then
     sumgrade:=sumgrade+dataset.FieldValues['credit']*3;
    if dataset.FieldValues['grade']='CB' then
     sumgrade:=sumgrade+dataset.FieldValues['credit']*2.5;
    if dataset.FieldValues['grade']='CC' then
     sumgrade:=sumgrade+dataset.FieldValues['credit']*2;
    if dataset.FieldValues['grade']='DC' then
     sumgrade:=sumgrade+dataset.FieldValues['credit']*1.5;
    if dataset.FieldValues['grade']='DD' then
     sumgrade:=sumgrade+dataset.FieldValues['credit']*1;
    if dataset.FieldValues['grade']='FD' then
     sumgrade:=sumgrade+dataset.FieldValues['credit']*0.5;
    if dataset.FieldValues['grade']='FF' then
     sumgrade:=sumgrade+dataset.FieldValues['credit']*0;
```

```
      dataset.Next;
    end
    else
     begin
      dataset.Next;
     end;
    end;  // with

  end;  //while
  if ((sumcredit>0) and (sumgrade>0)) then
  begin
  str((sumgrade/sumcredit):3:2,t);
  edit5.Text:=t;
  end;
     edit6.Text:=floattostr(sumgrade);
  end
  else
  begin
   messagedlg('This Student Never Registered Before...',mtinformation,[mbok],0);
   gpa:=0;
  end;

//end of gpa

// start of cgpa
   sumgencredit:=0;
```

```
sumgengrade:=0;
cgpa:=0;

with frm_main.ADOQuery4 do
begin
sql.Clear;
sql.add('Select t.GradeDate,t.studentId,t.grade,t.courseID,c.credit from transcript as t
inner join course as c on t.courseID=c.courseID where t.studentId="'+edit1.Text+'" and
grade is not null');
open;
end;

frm_main.DataSource10.DataSet.First;

while not(frm_main.DataSource10.DataSet.Eof) do
begin
with frm_main.DataSource10 do
begin
label10.Caption:=frm_main.ADOQuery4.FieldValues['courseId'];
with frm_main.ADOQuery6 do
begin
sql.Clear;
sql.add('Select t.GradeDate,t.studentId,t.grade,t.courseID,c.credit from transcript as t
inner join course as c on t.courseID=c.courseID where t.studentId="'+edit1.Text+'" and
t.courseId='+label10.Caption+' order by gradedate desc');
open;
end;
```

```
frm_main.DataSource12.DataSet.First;

if
dataset.FieldValues['gradedate']=frm_main.DataSource12.DataSet.FieldValues['gradedate']
then

begin
sumgencredit:=sumgencredit+dataset.FieldValues['credit'];

if dataset.FieldValues['grade']='AA' then
sumgengrade:=sumgengrade+dataset.FieldValues['credit']*4;
if dataset.FieldValues['grade']='BA' then
sumgengrade:=sumgengrade+dataset.FieldValues['credit']*3.5;
if dataset.FieldValues['grade']='BB' then
sumgengrade:=sumgengrade+dataset.FieldValues['credit']*3;
if dataset.FieldValues['grade']='CB' then
sumgengrade:=sumgengrade+dataset.FieldValues['credit']*2.5;
if dataset.FieldValues['grade']='CC' then
sumgengrade:=sumgengrade+dataset.FieldValues['credit']*2;
if dataset.FieldValues['grade']='DC' then
sumgengrade:=sumgengrade+dataset.FieldValues['credit']*1.5;
if dataset.FieldValues['grade']='DD' then
sumgengrade:=sumgengrade+dataset.FieldValues['credit']*1;
if dataset.FieldValues['grade']='FD' then
sumgengrade:=sumgengrade+dataset.FieldValues['credit']*0.5;
if dataset.FieldValues['grade']='FF' then
sumgengrade:=sumgengrade+dataset.FieldValues['credit']*0;
dataset.Next;
```

```
  end
  else
   begin
     dataset.Next;
   end;      //  end of if

  end;    //end of with

end;    //end of while


if ((sumgencredit>0) and (sumgengrade>0)) then
  begin
  str((sumgengrade/sumgencredit):3:2,t);
  edit6.Text:=t;
  end;

 label10.Caption:=Inttostr(sumgencredit);
 label11.Caption:=floattostr(sumgengrade);



// end of cgpa....
   with frm_main.ADOQuery2 do        //Find if already registered
    begin
```

```
sql.Clear;
sql.Add('SELECT  grade FROM transcript where StudentId="'+edit1.Text+'" and grade
null and dropdate is null');
open;
end;

if frm_main.DataSource8.DataSet.RecordCount>0 then
begin
messagedlg('This Student Has Already Been Registered, Use Add/Drop Menu To update
Course Registration ..!',mtwarning,[mbok],0);
Edit1.Clear;
Edit2.Clear;
Edit3.Clear;
Edit4.Clear;
Edit5.Clear;
Edit6.Clear;
Listbox1.Items.Clear;
listbox2.Items.Clear;
edit1.ReadOnly:=false;
edit1.SetFocus;
exit;

end;
```

```
with frm_main.ADOQuery2 do          //Find failed courses
begin
sql.Clear;
sql.Add('SELECT  c.coursecode,t.grade,t.courseId FROM course as c inner join
ranscript as t');
sql.add('on c.courseId=t.courseId WHERE StudentId="'+edit1.Text+'");// and ((t.grade
 null) or(t.grade="FF") or');
 sql.Add('(t.grade="FD"))');
open;
end;


frm_main.DataSource8.DataSet.first;
while not(frm_main.DataSource8.DataSet.eof) do
begin
if ((frm_main.DataSource8.DataSet.FieldValues['grade']='FF') or
frm_main.DataSource8.DataSet.FieldValues['grade']='FD')) then
 begin
 label15.Caption:=inttostr(frm_main.DataSource8.DataSet.FieldValues['courseId']);
 with frm_main.ADOQuery5 do          //Find if passed
  begin
  sql.Clear;
  sql.Add('SELECT  grade FROM transcript WHERE StudentId="'+edit1.Text+'" and
courseId='+label15.caption+' order by gradedate desc');
  open;
  end;
```

```
    if
frm_main.DataSource11.DataSet.FieldValues['grade']='FF')OR(frm_main.DataSource11.Da
taSet.FieldValues['grade']='FD')) THEN

     begin

      listbox1.Font.Color:=clred;

      listbox1.Items.Add(' '+frm_main.DataSource8.DataSet.FieldValues['coursecode']+' ---
- >'+frm_main.DataSource8.DataSet.FieldValues['grade']);

      frm_main.DataSource8.DataSet.next;

      end

     else

      begin

       frm_main.DataSource8.DataSet.next;


        end;

      end

     else

      begin

       frm_main.DataSource8.DataSet.Next;

      end;


    end;

   with frm_main.ADOQuery2 do          //find unregistered courses

    begin

     sql.Clear;

     sql.Add('SELECT  coursecode FROM course ');

     sql.add('WHERE courseId not in (SELECT  c.courseID FROM course as c inner join
transcript as t');
```

```
 sql.Add('on c.courseId=t.courseId WHERE StudentId="'+edit1.Text+'")');// and ((t.grade
is null) or(t.grade="FF") or(t.grade="FD")))');

 open;

 end;


frm_main.DataSource8.DataSet.first;

while not(frm_main.DataSource8.DataSet.eof) do

begin

 listbox1.Font.Color:=clblack;

 listbox1.Items.Add(' '+frm_main.DataSource8.DataSet.FieldValues['coursecode']);

 frm_main.DataSource8.DataSet.next;

end ;

with frm_main.ADOQuery2 do          //Already passed courses

begin

 sql.Clear;

 sql.Add('SELECT  c.coursecode,t.grade FROM course as c inner join transcript as t');

 sql.add('on c.courseId=t.courseId WHERE StudentId="'+edit1.Text+'" and t.grade is not
null');// and ((t.grade is null) or(t.grade="FF") or');

   sql.Add('(t.grade="FD"))');

 open;

 end;


frm_main.DataSource8.DataSet.first;

while not(frm_main.DataSource8.DataSet.eof) do

 begin

  if not((frm_main.DataSource8.DataSet.FieldValues['grade']='FF') or
frm_main.DataSource8.DataSet.FieldValues['grade']='FD')) then

   begin
```

```
  listbox1.Font.Color:=clred;

  listbox1.Items.Add(' '+frm_main.DataSource8.DataSet.FieldValues['coursecode']+'  ----
+frm_main.DataSource8.DataSet.FieldValues['grade']+' --- PASSED');

  frm_main.DataSource8.DataSet.next;

 end

else

 begin


  frm_main.DataSource8.DataSet.Next;

 end;

end;


nd

se

egin

nessagedlg('Student ID is not Found !!',mtinformation,[mbok],0);

dit1.SetFocus;

nd;


d;
```

```
procedure Tfrm_correg.Button2Click(Sender: TObject);
var i:integer;
begin
 if ((listbox2.Items.Count<7) and (listbox1.ItemIndex>=0)) then
  begin
   i:=listbox1.ItemIndex;
   listbox2.Items.Add(' '+listbox1.Items.ValueFromIndex[listbox1.itemindex]);//
listbox1..SelCount.ValueFromIndex[listbox1.Items.)
   listbox1.Items.Delete(listbox1.ItemIndex);


  end
  else
  begin
   messagedlg('You can not register any more course !!!',mtwarning,[mbok],0);
   button5.setfocus;
  end;
end;

procedure Tfrm_correg.FormShow(Sender: TObject);
begin
```

```
Edit1.Clear;
Edit2.Clear;
Edit3.Clear;
Edit4.Clear;
Edit5.Clear;
Edit6.Clear;
Listbox1.Items.Clear;
stbox2.Items.Clear;
dit1.ReadOnly:=false;
dit1.SetFocus;


nd;

rocedure Tfrm_correg.Button3Click(Sender: TObject);

egin
 f listbox2.ItemIndex>=0 then
 egin
  listbox1.Items.Add(' '+listbox2.Items.ValueFromIndex[listbox2.itemindex]);//
stbox1..SelCount.ValueFromIndex[listbox1.Items.)
  listbox2.Items.Delete(listbox2.ItemIndex);
 end;
end;

procedure Tfrm_correg.Button4Click(Sender: TObject);
var i:integer;
```

```
egin
:=0;
while (i<(listbox2.Count)) do
begin
with frm_main do
begin

Datasource2.Edit;
datasource2.DataSet.Append;
datasource2.DataSet.FieldValues['StudentId']:=edit1.Text;
adotable2.FieldValues['StudentId']:=edit1.Text;
adotable2.FieldValues['coursecode']:=listbox2.Items.ValueFromIndex[i];
if pos(' ',listbox2.Items.ValueFromIndex[i])>0 then
begin
with frm_main.ADOQuery4 do          //find unregistered courses
begin
sql.Clear;
sql.Add('SELECT  courseId FROM course where
coursecode="'+(midstr(listbox2.items.ValueFromIndex[i],1,(pos('-
ansireplacestr(listbox2.items.ValueFromIndex[i],' ','')))))+'"');
open;
end;

datasource2.DataSet.FieldValues['courseid']:=datasource10.DataSet.FieldValues['courseid'];
end
else
begin
```

```
    with frm_main.ADOQuery4 do          //find unregistered courses
     begin
      sql.Clear;
      sql.Add('SELECT  courseId FROM course where
coursecode="'+listbox2.Items.ValueFromIndex[i]+'"');
      open;
     end;

atasource2.DataSet.FieldValues['courseid']:=datasource10.DataSet.FieldValues['courseid'];
    end;
   datasource2.DataSet.FieldValues['courseregdate']:=date;
   datasource2.DataSet.FieldValues['processuser']:=username;
   datasource2.DataSet.FieldValues['processdate']:=date;
   datasource2.DataSet.Post;
    messagedlg('Couse Registration completed successfully!!',mtinformation,[mbok],0);


   adotable2.Post;
   adotable2.Next;
   datasource2.DataSet.Next;
  end;
 i:=i+1;
end;
if listbox2.Count>0 then
 begin
  messagedlg(' Couse Registration completed successfully!!',mtinformation,[mbok],0);
  Edit1.Clear;
```

```
  Edit2.Clear;
  Edit3.Clear;
  Edit4.Clear;
  Edit5.Clear;
  Edit6.Clear;
  Listbox1.Items.Clear;
  listbox2.Items.Clear;
  edit1.SetFocus;
  edit1.ReadOnly:=false;
 end
 else
  messagedlg('Please Select courses to register!!',mtwarning,[mbok],0);
end;

procedure Tfrm_correg.Button7Click(Sender: TObject);
begin
frm_correg.Hide;
frm_beginsemester.show;
end;

end.
```

**UNIT 23**

```
unit Unit23;

interface

uses
```

```
Windows, Messages, SysUtils, Variants, Classes, Graphics, Controls, Forms,
Dialogs, StdCtrls, Grids, DBGrids, DB, ADODB;

type
  TRep_dept_cor = class(TForm)
    ADOQuery1: TADOQuery;
    DataSource1: TDataSource;
    ComboBox1: TComboBox;
    Label1: TLabel;
    Label2: TLabel;
    DataSource2: TDataSource;
    ADOQuery2: TADOQuery;
    DBGrid1: TDBGrid;
    Button1: TButton;
    Button2: TButton;
    procedure FormCreate(Sender: TObject);
    procedure Button1Click(Sender: TObject);
    procedure Button2Click(Sender: TObject);
    procedure ComboBox1Click(Sender: TObject);
    procedure FormClose(Sender: TObject; var Action: TCloseAction);
  private
    { Private declarations }
  public
    { Public declarations }
  end;
```

```
var
Rep_dept_cor: TRep_dept_cor;

implementation

uses Unit16,unit26;

{$R *.dfm}

procedure TRep_dept_cor.FormCreate(Sender: TObject);
begin

 with ADOQuery1 do
  begin
   SQL.Clear;
   sql.add('SELECT departmentId FROM department order by departmentId');
   open;
  end;

 DataSource1.DataSet.First;
 while not (DataSource1.DataSet.Eof) do
 begin
  combobox1.Items.add(DataSource1.DataSet.FieldValues['DepartmentID']);

  DataSource1.DataSet.Next;
 end;
```

```
end;

procedure TRep_dept_cor.Button1Click(Sender: TObject);
begin
Rep_dept_cor.print;
end;

procedure TRep_dept_cor.Button2Click(Sender: TObject);
begin
 Rep_dept_cor.Close;
end;

procedure TRep_dept_cor.ComboBox1Click(Sender: TObject);
begin

  with ADOQuery2 do
   begin
    SQL.Clear;
    sql.add('SELECT * FROM course where departmentID="'+combobox1.Text+'" order by
coursename');
    open;
   end;

end;

procedure TRep_dept_cor.FormClose(Sender: TObject;
```

var Action: TCloseAction);

egin

rep_main.show;

nd;

nd.

## NIT 24

nit Unit24;

nterface

ses

Windows, Messages, SysUtils, Variants, Classes, Graphics, Controls, Forms,

Dialogs, StdCtrls, Grids, DBGrids, DB, ADODB;

ype

TRep_all_stu = class(TForm)

Label1: TLabel;

ADOQuery1: TADOQuery;

DataSource1: TDataSource;

DBGrid1: TDBGrid;

Button1: TButton;

Button2: TButton;

procedure Button2Click(Sender: TObject);

procedure Button1Click(Sender: TObject);

procedure FormCreate(Sender: TObject);

procedure FormClose(Sender: TObject; var Action: TCloseAction);

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

var
Rep_all_stu: TRep_all_stu;

implementation

uses Unit16,unit26;

{$R *.dfm}

procedure TRep_all_stu.Button2Click(Sender: TObject);
begin
Rep_all_stu.Close;
end;

procedure TRep_all_stu.Button1Click(Sender: TObject);
begin
Rep_all_stu.Print;
end;

procedure TRep_all_stu.FormCreate(Sender: TObject);
```

```
egin
  with ADOQuery1 do
  begin
   SQL.Clear;
   sql.add('SELECT * FROM student');
   open;
  end;

nd;

rocedure TRep_all_stu.FormClose(Sender: TObject;
 ar Action: TCloseAction);
egin
 ep_main.show;
nd;

nd.
```

**UNIT 25**

```
nit Unit25;

nterface

ses
  Windows, Messages, SysUtils, Variants, Classes, Graphics, Controls, Forms,
  Dialogs, Grids, DBGrids, DB, ADODB, StdCtrls;

ype
```

```
TRep_dept_stu = class(TForm)
 Button1: TButton;
 Button2: TButton;
 ComboBox1: TComboBox;
 Label1: TLabel;
 Label2: TLabel;
 ADOQuery1: TADOQuery;
   ADOQuery2: TADOQuery;
   DataSource1: TDataSource;
   DataSource2: TDataSource;
   DBGrid1: TDBGrid;
   procedure Button2Click(Sender: TObject);
   procedure Button1Click(Sender: TObject);
   procedure FormCreate(Sender: TObject);
   procedure ComboBox1Click(Sender: TObject);
   procedure FormClose(Sender: TObject; var Action: TCloseAction);
  private
    { Private declarations }
  public
    { Public declarations }
  end;

 var
   Rep_dept_stu: TRep_dept_stu;

 implementation
```

```
uses Unit16,unit26;

{$R *.dfm}

procedure TRep_dept_stu.Button2Click(Sender: TObject);
begin
 Rep_dept_stu.Close;
end;

procedure TRep_dept_stu.Button1Click(Sender: TObject);
begin
 Rep_dept_stu.Print;
end;

procedure TRep_dept_stu.FormCreate(Sender: TObject);
begin
 with ADOQuery1 do
   begin
    SQL.Clear;
    sql.add('SELECT departmentId FROM department order by departmentId');
    open;
   end;

  DataSource1.DataSet.First;
  while not (DataSource1.DataSet.Eof) do
```

```
begin
  combobox1.Items.add(DataSource1.DataSet.FieldValues['DepartmentID']);

  DataSource1.DataSet.Next;
 end;end;

procedure TRep_dept_stu.ComboBox1Click(Sender: TObject);
begin

  with ADOQuery2 do
   begin
    SQL.Clear;
    sql.add('SELECT * FROM student where departmentID="'+combobox1.Text+'" order by
name,surname');
    open;
   end;
 end;

procedure TRep_dept_stu.FormClose(Sender: TObject;
 var Action: TCloseAction);
begin
 rep_main.show;
end;

end.
```

## UNIT 26

```
unit Unit26;
```

```
interface

uses
 Windows, Messages, SysUtils, Variants, Classes, Graphics, Controls, Forms,
 Dialogs, StdCtrls, DB, ADODB, jpeg, ExtCtrls;

type
 Tfrm_password = class(TForm)
  Image1: TImage;
  Label1: TLabel;
  Label2: TLabel;
  Edit2: TEdit;
  Button1: TButton;
  EXIT: TButton;
  ComboBox1: TComboBox;
  ADOTable1: TADOTable;
  DataSource1: TDataSource;
  ADOQuery1: TADOQuery;
  DataSource2: TDataSource;
  procedure FormClose(Sender: TObject; var Action: TCloseAction);
  procedure EXITClick(Sender: TObject);
  procedure FormCreate(Sender: TObject);
  procedure Button1Click(Sender: TObject);
 private
  { Private declarations }
```

```
public
 { Public declarations }
end;

var
 frm_password: Tfrm_password;
 level:string;
 username:string;
 trial:integer;
implementation

uses Unit1;

{$R *.dfm}

procedure Tfrm_password.FormClose(Sender: TObject;
 var Action: TCloseAction);
begin
 application.Terminate;
end;

procedure Tfrm_password.EXITClick(Sender: TObject);
begin
 frm_password.Close;
end;
```

```
procedure Tfrm_password.FormCreate(Sender: TObject);
begin
trial:=0;
ADOTable1.Close;
ADOTable1.Open;
datasource1.DataSet.First;
while not datasource1.DataSet.Eof do
begin
combobox1.Items.Add(datasource1.DataSet.FieldValues['LoginName']);
datasource1.DataSet.Next;
end;
end;

procedure Tfrm_password.Button1Click(Sender: TObject);
begin
 with adoquery1 do
 begin
  sql.Clear;
  sql.Add('Select * from login where LoginName="'+combobox1.Text+'"');
  open;
 end;


   begin
   if edit2.text=datasource2.dataset.fieldvalues['password'] then
   begin
```

```
  level:=datasource2.DataSet.FieldValues['AccessLevel'];
  username:=datasource2.DataSet.FieldValues['LoginName'];
  frm_password.Hide;
  frm_main.Show;
 end
 else
 begin
  edit2.Text:='';
  trial:=trial+1;
  MessageDlg('Wrong Password. Please Enter again. '+IntToStr(3-trial)+' Trials Left',
mtInformation,[mbOk], 0);
  edit2.SetFocus;
 end;
end;
if trial=3 then
begin
 MessageDlg('You Are Not An Accessible Person To This System', mtInformation,[mbOk],
0);
 application.Terminate;
end;
end;


end.
```

## UNIT 27

```
unit Unit27;


interface
```

```
uses
 Windows, Messages, SysUtils, Variants, Classes, Graphics, Controls, Forms,
 Dialogs, Buttons, StdCtrls, ExtCtrls, ComCtrls, jpeg;

type
 Tfrm_login = class(TForm)
  StatusBar1: TStatusBar;
  Label1: TLabel;
  Label2: TLabel;
  Label3: TLabel;
  Edit1: TEdit;
  Edit2: TEdit;
  ComboBox1: TComboBox;
  RadioGroup1: TRadioGroup;
  Button1: TButton;
  Button2: TButton;
  Button3: TButton;
  Button4: TButton;
  Button5: TButton;
  Button6: TButton;
  ComboBox2: TComboBox;
  Button7: TButton;
  procedure FormClose(Sender: TObject; var Action: TCloseAction);
  procedure BitBtn1Click(Sender: TObject);
  procedure Button6Click(Sender: TObject);
```

```
  procedure Button1MouseMove(Sender: TObject; Shift: TShiftState; X,
    Y: Integer);
  procedure FormMouseMove(Sender: TObject; Shift: TShiftState; X,
    Y: Integer);
  procedure FormShow(Sender: TObject);
  procedure Button3Click(Sender: TObject);
  procedure FormCreate(Sender: TObject);
  procedure Button2Click(Sender: TObject);
  procedure Button4Click(Sender: TObject);
  procedure ComboBox2Select(Sender: TObject);
  procedure ComboBox2KeyPress(Sender: TObject; var Key: Char);
  procedure ComboBox1KeyPress(Sender: TObject; var Key: Char);
  procedure Button7Click(Sender: TObject);
  procedure Button1Click(Sender: TObject);
  procedure Button5Click(Sender: TObject);
 private
  { Private declarations }
 public
  { Public declarations }
 end;

var
 frm_login: Tfrm_login;
 process:string;
implementation
```

```
uses Unit1,unit26;

{$R *.dfm}

procedure Tfrm_login.FormClose(Sender: TObject; var Action: TCloseAction);
begin
 frm_main.show;
 combobox2.Visible:=false;
end;

procedure Tfrm_login.BitBtn1Click(Sender: TObject);
begin
 frm_login.Close;
end;

procedure Tfrm_login.Button6Click(Sender: TObject);
begin
 frm_login.Close;
end;

procedure Tfrm_login.Button1MouseMove(Sender: TObject; Shift: TShiftState;
  X, Y: Integer);
begin
  statusbar1.SimpleText:=(sender as tbutton).Hint;
end;
```

```
procedure Tfrm_login.FormMouseMove(Sender: TObject; Shift: TShiftState; X,
 Y: Integer);
 begin
  statusbar1.SimpleText:='';
 end;

 procedure Tfrm_login.FormShow(Sender: TObject);
 begin
   edit1.Clear;
   edit1.Visible:=true;
   edit2.Clear;
   combobox1.Clear;
   combobox2.Clear;
   combobox2.Visible:=false;
   edit1.Enabled:=false;
   edit2.Enabled:=false;
   combobox1.Enabled:=false;
   radiogroup1.Enabled:=false;
   button3.Enabled:=false;
   button4.Enabled:=false;
   button5.Enabled:=false;
   if level='Admin' then
   begin
     button3.Enabled:=true;
     button4.Enabled:=true;
     button5.Enabled:=true;
```

```
id;
d;

ocedure Tfrm_login.Button3Click(Sender: TObject);
gin
dit1.Enabled:=true;
dit1.Visible:=true;
ombobox2.Visible:=false;
dit1.Clear;
dit2.Clear;
combobox1.Clear;
combobox2.Clear;
combobox2.Enabled:=false;
with frm_main.ADOquery1 do
begin
 sql.clear;
 sql.Add('Select DepartmentId from Department');
 open;
end;
frm_main.datasource7.DataSet.First;
while not frm_main.datasource7.DataSet.Eof do
begin
  combobox1.Items.Add(frm_main.datasource7.DataSet.FieldValues['DepartmentId']);
  frm_main.DataSource7.DataSet.Next;
end;
```

```
.Enabled:=true;
.Enabled:=true;
obobox1.Enabled:=true;
bobox2.Enabled:=false;
1.SetFocus;
ogroup1.Enabled:=true;
1.Clear;
2.Clear;
iogroup1.ItemIndex:=2;
cess:='add';  //to inform save button


cedure Tfrm_login.FormCreate(Sender: TObject);
in
ocess:='';
;


ocedure Tfrm_login.Button2Click(Sender: TObject);
gin
(process='add') then
egin
f
rim(edit1.text)<>'')and(trim(edit2.text)<>'')and(trim(combobox1.text)<>'')and(radiogroup1.
emIndex>=0)) then
oegin
 if process='add' then
  begin
```

```
with frm_main.ADOQuery2 do
begin
 sql.Clear;
 sql.Add('select loginname from login where loginname="'+edit1.Text+'"');
 open;
end;
//  frm_main.DataSource8.DataSet.First;
if frm_main.DataSource8.DataSet.RecordCount>0 then
begin
 messagedlg('Please select another user name!!',mtinformation,[mbok],0);
 edit1.Clear;
 edit2.Clear;
 exit;
end
else
 begin
  with frm_password.DataSource1 do
  begin
   dataset.Edit;
   dataset.Append;
   dataset.FieldValues['LoginName']:=edit1.Text;
   dataset.FieldValues['Password']:=edit2.Text;
   dataset.FieldValues['departmentId']:=combobox1.text;
   case radiogroup1.ItemIndex of
    0:dataset.FieldValues['AccessLevel']:='Admin';
    1:dataset.FieldValues['AccessLevel']:='Advisor';
```

```
2:dataset.FieldValues['AccessLevel']:='User';
nd;
ataset.post;
dit1.Clear;
dit2.Clear;
ombobox1.Clear;
ombobox2.Clear;
dit1.Enabled:=false;
dit2.Enabled:=false;
ombobox1.Enabled:=false;
adiogroup1.Enabled:=false;
process:='';
end;
nd;
nd;
d
se
egin
nessagedlg('Please Fill All Fields And Select Access Level',mtinformation,[mbok],0);
nd;
d;
process='delete' then
egin
f combobox1.text='' then
begin
```

```
 edit1.Visible:=true;
 combobox2.Visible:=false;
 combobox2.Enabled:=false;
 process:='';
end
else
 begin
 edit1.Enabled:=true;
  edit2.Enabled:=true;
  combobox1.Enabled:=true;
  combobox2.Visible:=false;
   combobox2.Enabled:=false;

   frm_password.DataSource2.DataSet.Delete;
   messagedlg('The record successfully  deleted',mtinformation,[mbok],0);
   edit1.Clear;
   edit2.Clear;
   combobox1.Clear;
   combobox2.Clear;
   edit1.Visible:=true;
   edit1.Enabled:=false;
   edit2.Enabled:=false;
   combobox1.Enabled:=false;
   combobox2.Enabled:=false;
   combobox2.Visible:=false;
//     button3.Enabled:=false;
```

```
//    button4.Enabled:=false;
//    button5.Enabled:=false;
  end;
 end;
if process='changepassword' then
 begin
  if trim(edit2.text)<>'' then
   begin
    frm_password.DataSource2.DataSet.edit;
    frm_password.DataSource2.DataSet.FieldValues['password']:=edit2.Text;
    frm_password.DataSource2.DataSet.Post;
    edit1.clear;
    edit2.Clear;
    combobox1.Clear;
    process:='';
   end
  else
   messagedlg('Blank Password Is Not Allowed..!!',mtinformation,[mbok],0);
   edit2.SetFocus;
  end;


   if process='changeaccess' then
    begin
     with frm_password.ADOquery1 do
      begin
```

```
    sql.clear;
    sql.Add('Select * from Login where Loginname="'+username+'"');
    open;
   end;
   frm_password.DataSource2.DataSet.edit;
   case radiogroup1.ItemIndex of
    0:frm_password.DataSource2.dataset.FieldValues['AccessLevel']:='Admin';
    1:frm_password.DataSource2.dataset.FieldValues['AccessLevel']:='Advisor';
    2:frm_password.DataSource2.dataset.FieldValues['AccessLevel']:='User';
   end;
  frm_password.DataSource2.DataSet.Post;
  edit1.clear;
  edit2.Clear;
  combobox1.Clear;
  combobox2.Items.Clear;
  combobox2.Clear;
  combobox2.Visible:=false;
  edit1.visible:=true;
  radiogroup1.Enabled:=falsE;
  process:='';
  end;


end;
```

```
procedure Tfrm_login.Button4Click(Sender: TObject);
begin
  edit1.Clear;
  edit2.Clear;
  combobox1.Clear;
  combobox2.Clear;
  edit1.Enabled:=false;
  combobox2.Enabled:=true;
  with frm_password.ADOquery1 do
  begin
    sql.clear;
    sql.Add('Select LoginName from Login');
    open;
   end;
   frm_password.datasource2.DataSet.First;
   while not frm_password.datasource2.DataSet.Eof do
   begin
     combobox2.Items.Add(frm_password.datasource2.DataSet.FieldValues['LoginName']);
     frm_password.DataSource2.DataSet.Next;
    end;
   // edit2.Enabled:=true;
   // combobox1.Enabled:=true;
    edit1.Visible:=false;
```

```
end;

procedure Tfrm_login.ComboBox2KeyPress(Sender: TObject; var Key: Char);
begin
  KEY:=#0;
end;

procedure Tfrm_login.ComboBox1KeyPress(Sender: TObject; var Key: Char);
 begin
   KEY:=#0;
 end;

 procedure Tfrm_login.Button7Click(Sender: TObject);
 begin
   edit1.Clear;
   edit2.Clear;
   combobox1.Text:='';
   combobox2.Text:='';
   combobox1.Items.Clear;
   combobox2.Items.Clear;
   radiogroup1.Enabled:=false;
   process:='';
  end;
```

```
procedure Tfrm_login.Button1Click(Sender: TObject);
begin
 process:='changepassword';
 combobox2.Visible:=false;
 edit1.Visible:=true;
 combobox2.Enabled:=false;
 edit1.Enabled:=false;
 edit2.Enabled:=true;
 edit2.Visible:=true;
 combobox1.Enabled:=false;
 combobox2.Text:=username;
 edit1.Text:=username;
 edit2.SetFocus;

 with frm_password.ADOquery1 do
 begin
  sql.clear;
  sql.Add('Select * from Login where Loginname="'+username+'"');
  open;
 end;
 combobox1.Text:=frm_password.DataSource2.DataSet.FieldValues['DepartmentId'];

// radiogroup1.ItemIndex:=frm_password.DataSource2.DataSet.FieldValues['AccessLevel'];
end;
```

```
rocedure Tfrm_login.Button5Click(Sender: TObject);
egin
 process:='changeaccess';
 combobox2.Visible:=true;
 combobox2.Enabled:=true;
 edit1.Enabled:=false;
 edit2.Enabled:=true;
 edit2.Visible:=true;
 combobox1.Enabled:=false;
 combobox2.Text:=username;


 with frm_password.ADOquery1 do
 begin
  sql.clear;
  sql.Add('Select * from Login where Loginname="'+username+'"');
  open;
 end;
 radiogroup1.Enabled:=true;
 if frm_password.DataSource2.DataSet.FieldValues['AccessLevel']='Admin' then
  radiogroup1.ItemIndex:=0;
 if frm_password.DataSource2.DataSet.FieldValues['AccessLevel']='Advisor' then
  radiogroup1.ItemIndex:=1;
 if frm_password.DataSource2.DataSet.FieldValues['AccessLevel']='User' then
  radiogroup1.ItemIndex:=2;
 combobox1.Text:=frm_password.DataSource2.DataSet.FieldValues['DepartmentId'];
```

```
with frm_password.ADOquery1 do
begin
 sql.clear;
 sql.Add('Select * from Login');
 open;
end;
frm_password.datasource2.DataSet.First;
while not frm_password.datasource2.DataSet.Eof do
begin
 combobox2.Items.Add(frm_password.datasource2.DataSet.FieldValues['LoginName']);
 frm_password.DataSource2.DataSet.Next;
end;


end;


end.
```

## UNIT 28

```
unit Unit28;


interface


uses
 Windows, Messages, SysUtils, Variants, Classes, Graphics, Controls, Forms,
 Dialogs, Buttons;
```

```
e
frm_beginsemester = class(TForm)
SpeedButton1: TSpeedButton;
SpeedButton2: TSpeedButton;
SpeedButton3: TSpeedButton;
SpeedButton4: TSpeedButton;
procedure SpeedButton3Click(Sender: TObject);
procedure FormClose(Sender: TObject; var Action: TCloseAction);
procedure SpeedButton1Click(Sender: TObject);
procedure SpeedButton2Click(Sender: TObject);
procedure SpeedButton4Click(Sender: TObject);
rivate
{ Private declarations }
ublic
{ Public declarations }
nd;

r
rm_beginsemester: Tfrm_beginsemester;

nplementation

es Unit1, Unit22, Unit29, Unit15;

SR *.dfm}
```

```
procedure Tfrm_beginsemester.SpeedButton3Click(Sender: TObject);
begin
 frm_beginsemester.Close;
end;

procedure Tfrm_beginsemester.FormClose(Sender: TObject;
 var Action: TCloseAction);
begin
 frm_main.show;
end;

procedure Tfrm_beginsemester.SpeedButton1Click(Sender: TObject);
begin
 frm_correg.show;
 frm_beginsemester.Hide;
end;

procedure Tfrm_beginsemester.SpeedButton2Click(Sender: TObject);
begin
 frm_adddrop.show;
 frm_beginsemester.Hide;
end;

procedure Tfrm_beginsemester.SpeedButton4Click(Sender: TObject);
begin
```

```
 frm_beginsemester.Hide;
 frm_semend.show;
end;

end.
```

## UNIT 29

```
unit Unit29;

interface

uses
  Windows, Messages, SysUtils, Variants, Classes, Graphics, Controls, Forms,
  Dialogs, StdCtrls, ExtCtrls,strutils;

type
  Tfrm_adddrop = class(TForm)
    Label1: TLabel;
    Label2: TLabel;
    Label3: TLabel;
    Label4: TLabel;
    Label7: TLabel;
    Label8: TLabel;
    Label9: TLabel;
    Label10: TLabel;
    Label11: TLabel;
    Edit1: TEdit;
    Button1: TButton;
```

```
Edit2: TEdit;
Edit3: TEdit;
Edit4: TEdit;
Panel1: TPanel;
Panel2: TPanel;
Label5: TLabel;
Label6: TLabel;
ListBox1: TListBox;
ListBox2: TListBox;
Button2: TButton;
Button3: TButton;
Edit5: TEdit;
Edit6: TEdit;
Button4: TButton;
Button5: TButton;
Button6: TButton;
Button7: TButton;
Label12: TLabel;
Label15: TLabel;
procedure FormClose(Sender: TObject; var Action: TCloseAction);
procedure Button6Click(Sender: TObject);
procedure Button7Click(Sender: TObject);
procedure Button1Click(Sender: TObject);
procedure Button2Click(Sender: TObject);
procedure Button3Click(Sender: TObject);
procedure Button5Click(Sender: TObject);
```

```
procedure FormShow(Sender: TObject);
procedure Button4Click(Sender: TObject);
rivate
{ Private declarations }
ublic
{ Public declarations }
nd;

ar
frm_adddrop: Tfrm_adddrop;

nplementation

ses Unit28, Unit1,unit26;

$R *.dfm}

rocedure Tfrm_adddrop.FormClose(Sender: TObject;
var Action: TCloseAction);
egin
frm_beginsemester.show;
nd;

procedure Tfrm_adddrop.Button6Click(Sender: TObject);
begin
frm_adddrop.Hide;
```

```
 frm_main.show;
end;

procedure Tfrm_adddrop.Button7Click(Sender: TObject);
begin
 frm_adddrop.Close;
end;

procedure Tfrm_adddrop.Button1Click(Sender: TObject);
var sumcredit:integer;
var sumgrade,gpa:real;
var sumgencredit:integer;
var sumgengrade,cgpa:real;
var cid:integer;

var ypart:integer;
var mpart:string;
var ShortMonthNames:string;

begin
  if edit1.text<>null then
   begin
    with frm_main.ADOQuery1 do
     begin
      sql.Clear;
      sql.Add('SELECT  * FROM student WHERE StudentId="'+edit1.Text+'"');
```

```
 open;
end;
if not(frm_main.DataSource7.DataSet.eof) then
begin
 edit2.Text:=frm_main.DataSource7.DataSet.FieldValues['name'];
 edit3.Text:=frm_main.DataSource7.DataSet.FieldValues['surname'];
 edit4.Text:=frm_main.DataSource7.DataSet.FieldValues['departmentID'];

 edit1.ReadOnly:=true;

 shortdateformat:=('mm');
 ShortMonthNames:=datetostr(date);
 label12.Caption:=ShortMonthNames;
 if
((ShortMonthNames='01')or(ShortMonthNames='12')or(ShortMonthNames='11')or(ShortMo
nthNames='10')or(ShortMonthNames='09')or(ShortMonthNames='08')) then
 mpart:='Fall'
 else
 mpart:='Spring';

 shortdateformat:=('yyyy');
//  ypart:=date;
 label9.Caption:=datetostr(date)+'  -  '+mpart+' Term registration ';

 shortdateformat:=('dd/mm/yyyy');

 // To calculate gpa.....
```

```
sumcredit:=0;
sumgrade:=0;
gpa:=0;
sumgencredit:=0;
sumgengrade:=0;
cgpa:=0;

with frm_main.ADOQuery4 do
begin
sql.Clear;
sql.add('Select t.GradeDate,t.studentId,t.grade,t.courseID,c.credit from transcript as t
ner join course as c on t.courseID=c.courseID where t.studentId="'+edit1.Text+'" and
grade is not null');
open;
end;

with frm_main.ADOQuery5 do
begin
sql.Clear;
sql.add('Select MAX(Gradedate) as maxdate from transcript');
open;
end;

frm_main.DataSource10.DataSet.First;
if not(frm_main.DataSource11.DataSet.Eof) then
begin
```

```
while not(frm_main.DataSource10.DataSet.Eof) do
 begin
  with frm_main.DataSource10 do
   begin
    if
ataset.FieldValues['GradeDate']>=frm_main.DataSource11.DataSet.FieldValues['maxdate'])
en
     begin
      sumcredit:=sumcredit+dataset.FieldValues['credit'];

      if dataset.FieldValues['grade']='AA' then
       sumgrade:=sumgrade+dataset.FieldValues['credit']*4;
      if dataset.FieldValues['grade']='BA' then
       sumgrade:=sumgrade+dataset.FieldValues['credit']*3.5;
      if dataset.FieldValues['grade']='BB' then
       sumgrade:=sumgrade+dataset.FieldValues['credit']*3;
      if dataset.FieldValues['grade']='CB' then
       sumgrade:=sumgrade+dataset.FieldValues['credit']*2.5;
      if dataset.FieldValues['grade']='CC' then
       sumgrade:=sumgrade+dataset.FieldValues['credit']*2;
      if dataset.FieldValues['grade']='DC' then
       sumgrade:=sumgrade+dataset.FieldValues['credit']*1.5;
      if dataset.FieldValues['grade']='DD' then
       sumgrade:=sumgrade+dataset.FieldValues['credit']*1;
      if dataset.FieldValues['grade']='FD' then
       sumgrade:=sumgrade+dataset.FieldValues['credit']*0.5;
      if dataset.FieldValues['grade']='FF' then
```

```
  sumgrade:=sumgrade+dataset.FieldValues['credit']*0;


  dataset.Next;
end
lse
begin
 dataset.Next;
end;
nd;  // with


d;  //while
(sumcredit>0) and (sumgrade>0)) then
it5.Text:=floattostr(sumgrade/sumcredit);
dit6.Text:=floattostr(sumgrade);
l


gin
essagedlg('This Student Never Registered Before...',mtinformation,[mbok],0);
a:=0;
d;


f gpa


of cgpa
ngencredit:=0;
ngengrade:=0;
```

```
cgpa:=0;

with frm_main.ADOQuery4 do
 begin
  sql.Clear;
  sql.add('Select t.GradeDate,t.studentId,t.grade,t.courseID,c.credit from transcript as t
ner join course as c on t.courseID=c.courseID where t.studentId="'+edit1.Text+'" and
rade is not null');
  open;
 end;

frm_main.DataSource10.DataSet.First;

while not(frm_main.DataSource10.DataSet.Eof) do
 begin
  with frm_main.DataSource10 do
   begin
   label10.Caption:=frm_main.ADOQuery4.FieldValues['courseId'];
    with frm_main.ADOQuery6 do
     begin
      sql.Clear;
      sql.add('Select t.GradeDate,t.studentId,t.grade,t.courseID,c.credit from transcript as t
ner join course as c on t.courseID=c.courseID where t.studentId="'+edit1.Text+'" and
courseId='+label10.Caption+' order by gradedate desc');
      open;
     end;

    frm_main.DataSource12.DataSet.First;
```

```
FieldValues['gradedate']=frm_main.DataSource12.DataSet.FieldValues['gradedate']

egin
sumgencredit:=sumgencredit+dataset.FieldValues['credit'];

f dataset.FieldValues['grade']='AA' then
 sumgengrade:=sumgengrade+dataset.FieldValues['credit']*4;
f dataset.FieldValues['grade']='BA' then
 sumgengrade:=sumgengrade+dataset.FieldValues['credit']*3.5;
if dataset.FieldValues['grade']='BB' then
 sumgengrade:=sumgengrade+dataset.FieldValues['credit']*3;
if dataset.FieldValues['grade']='CB' then
 sumgengrade:=sumgengrade+dataset.FieldValues['credit']*2.5;
if dataset.FieldValues['grade']='CC' then
 sumgengrade:=sumgengrade+dataset.FieldValues['credit']*2;
if dataset.FieldValues['grade']='DC' then
 sumgengrade:=sumgengrade+dataset.FieldValues['credit']*1.5;
if dataset.FieldValues['grade']='DD' then
 sumgengrade:=sumgengrade+dataset.FieldValues['credit']*1;
if dataset.FieldValues['grade']='FD' then
 sumgengrade:=sumgengrade+dataset.FieldValues['credit']*0.5;
if dataset.FieldValues['grade']='FF' then
 sumgengrade:=sumgengrade+dataset.FieldValues['credit']*0;
dataset.Next;
```

```
    end
   else
    begin
      dataset.Next;
    end;      //  end of if
  end;    //end of with
 end;    //end of while
if ((sumgencredit>0) and (sumgengrade>0)) then
   edit6.Text:=floattostr(sumgengrade/sumgencredit);
 label10.Caption:=Inttostr(sumgencredit);
 label11.Caption:=floattostr(sumgengrade);
end of cgpa....

 with frm_main.ADOQuery2 do         //Find if already registered
  begin
   sql.Clear;
   sql.Add('SELECT  c.coursecode FROM transcript as t inner join course as c on
courseId=c.courseId  where t.StudentId="'+edit1.Text+'" and t.grade is null  and dropdate is
ll');
    open;
   end;

  if frm_main.DataSource8.DataSet.RecordCount=0 then
    begin
    messagedlg('This Student Has Not Been Registered Yet, Use Registration Menu To
lake Course Registration ..!',mtwarning,[mbok],0);
     Edit1.Clear;
     Edit2.Clear;
```

```
t3.Clear;
t4.Clear;
t5.Clear;
t6.Clear;
tbox1.Items.Clear;
box2.Items.Clear;
t1.ReadOnly:=false;
t1.SetFocus;
t;


in
h frm_main.DataSource8 do
gin
ataset.First;
hile not(dataset.Eof) do
egin
listbox2.Items.Add(' '+dataset.FieldValues['Coursecode']);
dataset.Next;
end;
nd;
d;


h frm_main.ADOQuery2 do        //Find failed courses
gin
l.Clear;
```

```
sql.Add('SELECT  c.coursecode,t.grade,t.courseId FROM course as c inner join
script as t');

sql.add('on c.courseId=t.courseId WHERE StudentId="'+edit1.Text+'");// and ((t.grade
ull) or(t.grade="FF") or');

  sql.Add('(t.grade="FD"))');

open;

end;


frm_main.DataSource8.DataSet.first;

while not(frm_main.DataSource8.DataSet.eof) do

begin

 if ((frm_main.DataSource8.DataSet.FieldValues['grade']='FF') or
m_main.DataSource8.DataSet.FieldValues['grade']='FD')) then

  begin

   label15.Caption:=inttostr(frm_main.DataSource8.DataSet.FieldValues['courseId']);

   with frm_main.ADOQuery5 do        //Find if passed

    begin

     sql.Clear;

     sql.Add('SELECT  grade FROM transcript WHERE StudentId="'+edit1.Text+'" and
ourseId='+label15.caption+' order by gradedate desc');

     open;

    end;


   if
(frm_main.DataSource11.DataSet.FieldValues['grade']='FF')OR(frm_main.DataSource11.Da
aSet.FieldValues['grade']='FD')) THEN

    begin

     listbox1.Font.Color:=clred;
```

```
stbox1.Items.Add(' '+frm_main.DataSource8.DataSet.FieldValues['coursecode']+'  ---
n_main.DataSource8.DataSet.FieldValues['grade']);

rm_main.DataSource8.DataSet.next;

ıd

se

egin

rm_main.DataSource8.DataSet.next;


ıd;

d

e

gin

m_main.DataSource8.DataSet.Next;

d;


;

ı frm_main.ADOQuery2 do          //find unregistered courses

gin

l.Clear;

l.Add('SELECT  coursecode FROM course ');

l.add('WHERE courseId not in (SELECT  c.courseID FROM course as c inner join
ipt as t');

l.Add('on c.courseId=t.courseId WHERE StudentId="'+edit1.Text+'")');// and ((t.grade
 or(t.grade="FF") or(t.grade="FD")))');

ben;

d;


n_main.DataSource8.DataSet.first;
```

```
e not(frm_main.DataSource8.DataSet.eof) do

in

tbox1.Font.Color:=clblack;

tbox1.Items.Add(' '+frm_main.DataSource8.DataSet.FieldValues['coursecode']);

m_main.DataSource8.DataSet.next;

d ;

h frm_main.ADOQuery2 do        //Already passed courses

gin

ql.Clear;

ql.Add('SELECT  c.coursecode,t.grade FROM course as c inner join transcript as t');

ql.add('on c.courseId=t.courseId WHERE StudentId="'+edit1.Text+'");// and ((t.grade
l) or(t.grade="FF") or');

 sql.Add('(t.grade="FD"))');

open;

end;


rm_main.DataSource8.DataSet.first;

while not(frm_main.DataSource8.DataSet.eof) do

 begin

  if
(frm_main.DataSource8.DataSet.FieldValues['grade']=null)or(frm_main.DataSource8.Dat
t.FieldValues['grade']='FF') or
n_main.DataSource8.DataSet.FieldValues['grade']='FD')) then

    begin

     listbox1.Font.Color:=clred;

     listbox1.Items.Add(' '+frm_main.DataSource8.DataSet.FieldValues['coursecode']+'  ----
>'+frm_main.DataSource8.DataSet.FieldValues['grade']+' --- PASSED');

     frm_main.DataSource8.DataSet.next;
```

```
      end
     else
      begin

       frm_main.DataSource8.DataSet.Next;
      end;
     end;

   end
  else
   begin
    messagedlg('Student ID is not Found !!',mtinformation,[mbok],0);
    edit1.SetFocus;
   end;

  end;

end;

procedure Tfrm_adddrop.Button2Click(Sender: TObject);

begin
  if listbox2.Items.Count<7 then
   begin

    listbox2.Items.Add(' '+listbox1.Items.ValueFromIndex[listbox1.itemindex]);//
listbox1..SelCount.ValueFromIndex[listbox1.Items.)
```

```
ox1.Items.Delete(listbox1.ItemIndex);



n

sagedlg('You can not register any more course !!!',mtwarning,[mbok],0);

on5.setfocus;



dure Tfrm_adddrop.Button3Click(Sender: TObject);


tbox2.ItemIndex>=0 then

n

tbox1.Items.Add(' '+listbox2.Items.ValueFromIndex[listbox2.itemindex]);//
ox1..SelCount.ValueFromIndex[listbox1.Items.)

tbox2.Items.Delete(listbox2.ItemIndex);

;



edure Tfrm_adddrop.Button5Click(Sender: TObject);

n
it1.Clear;
it2.Clear;
lit3.Clear;
lit4.Clear;
lit5.Clear;
lit6.Clear;
```

```
stbox1.Items.Clear;
tbox2.Items.Clear;
it1.ReadOnly:=false;
it1.SetFocus;
;

cedure Tfrm_adddrop.FormShow(Sender: TObject);
gin
dit1.Clear;
dit2.Clear;
dit3.Clear;
dit4.Clear;
dit5.Clear;
dit6.Clear;
istbox1.Items.Clear;
istbox2.Items.Clear;
dit1.ReadOnly:=false;
dit1.SetFocus;
d;

rocedure Tfrm_adddrop.Button4Click(Sender: TObject);
ar i,t:integer;
ar a:string;
egin
vith frm_main.ADOQuery2 do          //Find if already registered
begin
```

```
 sql.Clear;

 sql.Add('SELECT  c.coursecode,t.processdate,t.processuser,t.dropdate FROM transcript as t
nner join course as c on t.courseId=c.courseId  where t.StudentId="'+edit1.Text+'" and
.grade is null and t.dropdate is null order by t.courseregdate desc');

 open;

end;

a:=' ';


frm_main.DataSource8.DataSet.First;

while not (frm_main.DataSource8.DataSet.Eof) do

begin

 i:=0;

 t:=0;

 while (i<=(listbox2.Count-1)) do

  begin

   if
frm_main.DataSource8.DataSet.FieldValues['coursecode']=listbox2.Items.ValueFromIndex[i
) then //(midstr(listbox2.items.ValueFromIndex[i],1,(pos('-
ansireplacestr(listbox2.items.ValueFromIndex[i],' ',''))))) then

    begin

     frm_main.DataSource8.DataSet.Edit;

     frm_main.DataSource8.DataSet.FieldValues['ProcessDate']:=date;

     frm_main.DataSource8.DataSet.FieldValues['ProcessUser']:=username;

     frm_main.DataSource8.DataSet.Post;

     t:=1;

     i:=i+1;          //unchanged course

    end

   else
```

```
egin
:=i+1;
nd;
d; //end of 2nd while

=0 then       //dropped course
gin
rm_main.DataSource8.DataSet.Edit;
rm_main.DataSource8.DataSet.FieldValues['ProcessDate']:=date;
rm_main.DataSource8.DataSet.FieldValues['ProcessUser']:=username;
rm_main.DataSource8.DataSet.FieldValues['dropdate']:=date;
rm_main.DataSource8.DataSet.Post;


nd;
m_main.DataSource8.DataSet.Next;


d;   // end of while

////////////////7
0;


n_main.DataSource8.DataSet.First;
ile (i<=(listbox2.Count-1)) do
gin
=0;
```

```
ıain.DataSource8.DataSet.First;
 not (frm_main.DataSource8.DataSet.Eof) do

ain.DataSource8.DataSet.FieldValues['coursecode']=leftstr(listbox2.Items.ValueFromI
,ansiindexstr(listbox2.Items.ValueFromIndex[i],a))) then
in
ı_main.DataSource8.DataSet.Next;
1;                  //unchanged course

in
n_main.DataSource8.DataSet.Next;
l;
 //end of 2nd while

0 then       //add course
in
ı_main.DataSource2.DataSet.Edit;
ı_main.DataSource2.DataSet.Append;
ı_main.DataSource8.DataSet.FieldValues['StudentId']:=edit1.Text;
ı_main.DataSource8.DataSet.FieldValues['courseregdate']:=date;
ı_main.DataSource8.DataSet.FieldValues['adddate']:=date;
n_main.DataSource8.DataSet.FieldValues['ProcessUser']:=username;
n_main.DataSource8.DataSet.FieldValues['processdate']:=date;
```

```
h frm_main.ADOQuery4 do        //Find courseID
gin
ql.Clear;
ql.Add('SELECT  courseId FROM course where
ecode="'+leftstr(listbox2.Items.ValueFromIndex[i],ansiindexstr(listbox2.Items.ValueFro
ex[i],a))+'"');
pen;
nd;

main.DataSource8.DataSet.FieldValues['courseId']:=frm_main.DataSource10.DataSet.Fi
alues['courseId'];
m_main.DataSource8.DataSet.post;
d;
n_main.DataSource8.DataSet.Next;
;   // end of while
```

**IT 40**

```
Unit40;

rface

s

indows, Messages, SysUtils, Variants, Classes, Graphics, Controls, Forms,
alogs, ExtCtrls, StdCtrls, Buttons, jpeg;
```

```
type
 Tfrm_about = class(TForm)
  Image1: TImage;
  Panel1: TPanel;
  Memo2: TMemo;
  a: TMemo;
  SpeedButton1: TSpeedButton;
  procedure SpeedButton1Click(Sender: TObject);
  procedure FormClose(Sender: TObject; var Action: TCloseAction);
 private
  { Private declarations }
 public
  { Public declarations }
 end;

var
 frm_about: Tfrm_about;

implementation

uses Unit1,unit26;

{$R *.dfm}

procedure Tfrm_about.SpeedButton1Click(Sender: TObject);
```

```
_about.Close;


edure Tfrm_about.FormClose(Sender: TObject; var Action: TCloseAction);
n
_main.Show;
```