UNIVERSITI
KEBANGSAAN
MALAYSIA
The National University of Malaysia

TU2983: Advanced Databases

Lab Notes 2:

Crash Course in Visual Basic .NET
and
Data Access Programming Fundamentals

Written by:

Hafiz Mohd Sarim

Centre of Artificial Intelligence Technology
Fakulti Teknologi dan Sains Maklumat
Universiti Kebangsaan Malaysia

Table of Contents

PART 1: Introduction

A. Lab Objectives

1. This TU2983 lab module aims to fulfil the following objectives:

   i. To provide students with a quick tutorial on using the Visual Basic .NET (VB) programming language for creating Windows form applications and executable. This tutorial will familiarize students with the fundamental VB programming syntax and covers variable declaration, control statements, functions and sub-procedures, Windows form objects and object events programming.

   ii. To teach students data access programming fundamentals using the OLEDB programming extensions, which will provide VB programs access to data that is stored in Microsoft Access databases.

2. Students must already be familiar with general programming concepts, as the lab modules for TU2983 from this point forward is not intended to be a full-fledged programming course. However, these lab notes were written in such a way that a novice and inexperienced programmer can immediately use it as a guide for building two-tier Windows programs that can access databases, also known as "**database applications**".

3. The programming language used for the TU2983 lab modules is Visual Basic .NET, while database used to hold project data will be either of the following
   a. Microsoft Access 2007 or above as the portable database
   b. IBM DB2 Enterprise Server Edition v9.7 or above as the server database.

   Students will be instructed on the exact database to use for each lab exercise and lab assignment.

4. These lab notes were written using the "**Visual Studio 2010 Ultimate Edition**" integrated development environment (IDE) for the **Visual Basic .NET** programming language. However, the steps demonstrated in these lab notes can also be used for all versions of Visual Basic .NET in Visual Studio IDEs starting from Visual Studio .NET 2002 and above.

B. IMPORTANT: Installing and Configuring Visual Studio 2019

1. Download the “Visual Studio 2019 Community - ZIP File” from the ‘REQUIRED SOFTWARE’ section on UKMfolio, and extract the ‘vs_community.exe’. Alternatively, you can also download Visual Studio Community directly from Microsoft at the following link and click the “Download Community 2019” button:

   https://docs.microsoft.com/en-us/visualstudio/releases/2019/history

2. Run the ‘vs_community.exe’ file.

3. When you reach the “WORKLOADS” window, tick only the following box, and click ‘INSTALL’

   .NET desktop development

4. When installation completes, run "Visual Studio 2019" from the the Windows Start Menu.

5. At the "Sign in to Visual Studio" screen, log in with your Microsoft-registered account.

6. Exit Visual Studio.

C. *** VERY IMPORTANT ***: Installing the Required Database Drivers

***** VERY IMPORTANT ***: You may need to install the Database Drivers to prevent the following error when trying to connect to your Microsoft Access Database:**

"The Microsoft.ACE.OLEDB.12.0 provider is not registered on the local machine"

The error above is caused by the failure of Microsoft Office / Microsoft 365 in registering the database drivers in the Windows Registry during initial installation. Follow these steps exactly to FORCE INSTALL the required database drivers to your PC.

1. Make sure Microsoft Access and Visual Studio are closed.

2. Download the **"Microsoft Access Database Engine 2016 Redistributable"** file (i.e. the database drivers) from Microsoft at the following link:

   https://www.microsoft.com/en-us/download/details.aspx?id=54920

3. Click the 'DOWNLOAD' button.

4. At the 'Choose the download you want' window, tick **'accessdatabaseengine.exe'** and click 'Download' (this will download the **32-bit database drivers**. See the NOTE on the following page to determine if you need to install the 64-bit database drivers instead)

5. In 'Windows Explorer, create a temporary folder C:\TEMP

6. Copy the "accessdatabaseengine.exe" file to the C:\TEMP folder.

7. Open a "Command Prompt" window as an "ADMINISTRATOR" (i.e. In your 'Windows Search' bar, type 'cmd' and when the 'Command Prompt' app menu appears, click 'Run As Administrator').

8. In the "Command Prompt" window. type the following line in full at the C:\> prompt (this will FORCE INSTALL the database drivers to your PC):

   ```
   C:\TEMP\accessdatabaseengine.exe /quiet
   ```

9. Nothing will happen for a few seconds. When you receive the the C:\> prompt again, close your "Command Prompt" window and run Visual Studio.

NOTE: The majority of TU2983 students will only require the 32-bit database drivers (i.e. the "accessdatabaseengine.exe" file). if you need to install the 64-bit database drivers, repeat the steps above with the following changes (all other steps are the same as before):

- In Step (**4**) above, download the **"accessdatabaseengine_X64.exe"** file instead.

- In Step (**8**) above, type the following line in full at the C:\> prompt instead:

  ```
  C:\TEMP\accessdatabaseengine_X64.exe /quiet
  ```

D. Visual Basic Object Naming conventions

1. Standardized prefixes and suffixes will be appended to the names of Visual Basic objects, according to the type of object created, for two purposes:

   - To ease the identification of a VB windows form object through its name alone, when referenced in VB-code.

   - To prove originality of a student's work and ownership of the VB project through the student's matric number.

2. The Visual Basic object naming conventions are as follows:

   i. All Visual Basic object names should be in lowercase (no capital letters) and must not contain any 'spaces', punctuations (**! ? , ; : .** etc.), mathematical operators (**+ - * /** etc.), or any other special characters. All 'spaces' must be replaced by the 'underscore' (_) character.

   ii. All object names must begin with a **prefix** to indicate the object type, and end with a **suffix** to indicate originality and ownership, as follows:

Object Type	Name Prefix	Name Suffix	Name Format	Example
VB project file	**prj_**	**_<matric>**	**prj_<name>_<matric>**	prj_university_a123456
Form	**frm_**	**_<matric>**	**frm_<name>_<matric>**	frm_registration_a123456
Module	**mod_**	**_<matric>**	**mod_<name>_<matric>**	mod_globals_a123456
DataGridView	**grd_**	*none*	**grd_<name>**	grd_students
Button	**btn_**	*none*	**btn_<name>**	btn_search
Textbox	**txt_**	*none*	**txt_<name>**	txt_studentname
Label	**lbl_**	*none*	**lbl_<name>**	lbl_formtitle
ComboBox	**cmb_**	*none*	**cmb_<name>**	cmb_department
ListBox	**lst_**	*none*	**lst_<name>**	lst_courses
PictureBox	**pic_**	*none*	**pic_<name>**	pic_studentphoto
CheckBox	**chk_**	*none*	**chk_<name>**	chk_examresults
RadioButton	**rad_**	*none*	**rad_<name>**	rad_gender

IMPORTANT MANDATORY RULE :

All VB Project file names, Form names, and Module names MUST end with the student's matric number as the suffix to indicate the creator of the VB object, as shown in the example above.

E. Student Requirements for this Lab

1. All students must bring the following equipment for the lab:

    - **MANDATORY REQUIREMENT:** One USB hard drive / thumb drive pen drive (to save your work)

2. All students must have the following requisite knowledge:

    - **MANDATORY REQUIREMENT:** Using the Microsoft Windows operating system.

    - **MANDATORY REQUIREMENT:** Understand the fundamentals of programming and recognize the different control statements and data structures that are generally available in all programming languages.

    - **MANDATORY REQUIREMENT:** Writing Structured Query Language code, or SQL.

    - **MANDATORY REQUIREMENT:** Creating a sample database in Microsoft Access (see TU2983 Lab Notes 1).

PART 2: Creating a Visual Basic .NET Project

A. Creating a new VB project

1. Open your working folder or create a new working folder.

    - If you have previously completed **TU2983 Lab Notes 1**, we will use the previous working folder that you used for creating a Microsoft Access database, i.e. **'a123456_facultyrecords'**, which you have stored in your USB drive.

    - If you have not, in the desktop, click **'Start' > 'Computer' > 'C:'**. Create a folder or directory with the name '**<matric>_<your project name>**'. For this example, type **'c:\a123456_facultyrecords'**.

    Remember this folder location as it will be used throughout these lab notes.

2. Start Visual Studio. In the Windows desktop, click **'Start' > 'All Programs' > 'Microsoft Visual Studio 2010' > 'Microsoft Visual Studio 2010'**.

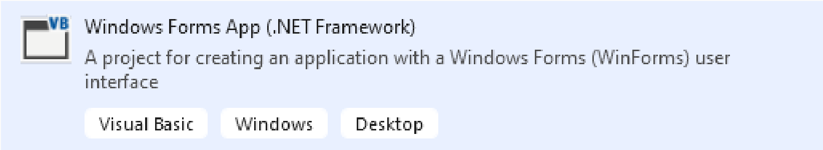
3. In the Visual Studio start page, click **'New Project...'**. The 'New Project window will open.

4. On the **'Installed Templates'** panel to the left of the 'New Project' window , you should see **'Visual Basic'** at the top. If you do not see **'Visual Basic'**, then click **'Other Languages' > 'Visual Basic'**.

5. Click **'Visual Basic' > 'Windows' > 'Windows Forms App (.NET Framework)'**.

IMPORTANT! In Visual Studio 2019, there is more than one project type called Windows Forms App

Choose the project type called **"Windows Forms App <u>(.NET Framework)".</u>** This is the CORRECT project type that has complete OLEDB data access features:

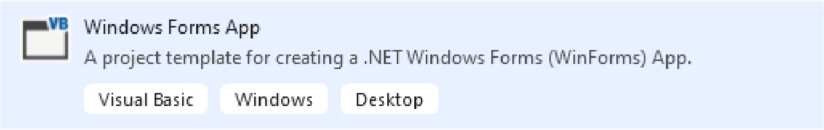


Windows Forms App (.NET Framework)

A project for creating an application with a Windows Forms (WinForms) user interface

← **CORRECT**

Do NOT choose the project type **"Windows Forms App".** This is the WRONG project type that does not have OLEDB data access features:



Windows Forms App

A project template for creating a .NET Windows Forms (WinForms) App.

← **WRONG**

6. In the **'Name'** field, type **'prj_facultyrecords_a123456'**, and click **'OK'**. The form design will open.

B. Renaming a Form File and Form Name

1. In the **'Solution Explorer'** panel on the top-right side of the form design window, you will see the objects that have been created automatically for your project, which is your project folder **'prj_facultyrecords_a123456'** and a blank form called **'Form1.vb'**

2. To rename the form file, click on **'Form1.vb'** in the **'Solution Explorer'**. In the **'Properties'** panel on the lower-right side of your form design window, click on the **'File Name'** property, and replace the current **'Form1.vb'** form name with **'frm_splashscreen_a123456.vb'**.

3. Click anywhere outside of the **'Properties'** panel, and the form name in your **'Solution Explorer**' panel will automatically change to **'frm_splashscreen_a123456.vb'**.

4. On the **'frm_splashscreen_a123456.vb [Design]'** tab in the main window, click on the form object. You will see that the **'Properties'** panel will switch to the properties for this form. Scroll down the properties list to the **(Name)** property (located under **'Design'**) and you will see that the form's name has been automatically changed to **'frm_splashscreen_a123456'**.

C. Saving your VB project

1. To save your VB project and all its objects and programming source code, on the main menu click **'File' > 'Save All'**. Alternatively you can click on the **'Save all'** button (multiple diskettes) on your tool bar.

2. The **'Save Project'** window will open. Leave the **'Name'** and **'Solution Name'** field as it is.

3. For the **'Location'** field, click on the **'Browse'** button, select your working folder from the folder list (e.g. **c:\a123456_facultyrecords**), and click **'Select Folder'**. Click **'Save'** in the **'Save Project'** window.

D. Running and Stopping your VB program

1. To run your program, click on the **'Start Debugging'** button (green arrow) or **'Play'** button on your toolbar. A blank Windows form, titled 'Form1' will appear.

2. Alternatively, on your main menu, you can click **'Debug' > 'Start Debugging'** to run your program. You can also just press the **'F5'** key on your keyboard to run your program.

3. To stop your program, click the **'X'** button on your form window. You will be returned to the form design view.

4. Alternatively, on your main menu, you can click **'Debug' > 'Stop Debugging'** to stop your program. You can also click the **'Stop Debugging'** button (blue square) or **'Stop'** button on your toolbar to stop your program.

5. You must make sure to stop your program completely before you can edit your form design, or edit your VB source code again. If the title bar of your Visual Studio IDE says **'(Running)'** then you have not stopped your program completely.

E. Adding Form Objects

1. The left-side of your form design window is the **'Toolbox'** panel, which contains a list of form objects that you can place onto your forms, either as display objects, input objects, or control objects.

2. From the toolbox, click-and-drag the **'Label'** object, and release-click it in your **'frm_splashscreen_a123456'** form. Labels are generally used to display static text that cannot be edited directly by the user during runtime.

3. From the toolbox, click-and-drag the **'TextBox'** object, and release-click it in your **'frm_splashscreen_a123456'** form. TextBoxes are used as input field for alphanumeric data entered by the user during runtime.

4. From the toolbox, click-and-drag the **'Button'** object, and release-click click it in your **'frm_splashscreen_a123456'** form. Buttons are used as control objects that can be used to execute user functions or control form transitions.

F. Editing the Properties of Form Objects

1. In the **'frm_splashscreen_a123456.vb [Design]'** tab, click once on the label you have just added, **'Label1'**.
   - In the '**Properties'** panel, scroll to the **'(Name)'** property and rename it to **'lbl_name'**.
   - In the **'Properties'** panel, scroll to the **'Text'** property and type in **'Please enter your name below:'**.

2. In the **'frm_splashscreen_a123456.vb [Design]'** tab, click once on the TextBox you have just added, **'TextBox1'**.
   - In the **'Properties'** panel, scroll to the **'(Name)'** property and rename it to **'txt_name'**.
   - In the **'Properties'** panel, scroll to the **'Text'** property and type in **'Anonymous'**.

3. In the **'frm_splashscreen_a123456.vb [Design]'** tab, click once on the Button you have just added, **'Button1'**.
   - In the **'Properties'** panel, scroll to the **'(Name)'** property and rename it to **'btn_start'**.
   - In the **'Properties'** panel, scroll to the **'Text'** property and type in **'START'**.

4. In the **'frm_splashscreen_a123456.vb [Design]'** tab, click once on the Form itself, **'frm_splashscreen_a123456'**.
   - In the **'Properties'** panel, scroll to the **'Text'** property and type in **'UKM Faculty Records'**.
   - In the **'Properties'** panel, scroll to the **'BackColor'** property and click the drop-arrow next to the property. In the drop-list, click on the **'Web'** tab, and click the **'Yellow'** color.

5. Run your program to view the results. Afterwards, stop your program to return to the form design window.

6. Move and arrange your form objects to make some space above the **'lbl_name'** label.

7. Add another label above **'lbl_name'**, and click once on the label you have just added.
   - In the **'Properties'** panel, scroll to the **'(Name)'** property and rename it to **'lbl_title'**.
   - In the **'Properties'** panel, scroll to the **'Text'** property and type in **'UKM Faculty Records'**.

- In the **'Properties'** panel, scroll to the **'Font'** property, and click on the ellipses **(...)** button. In the **'Font'** window, change the **'Font'** to **'Times New Roman'**, change the **'Font Style'** to **'Bold'**, and change the **'Font size'** to **'24'**. Click **'OK'**.
- In the **'Properties'** panel, scroll to the **'ForeColor'** property and click the drop-arrow next to the property. In the drop-list, click on the **'Web'** tab, and click the **'Blue'** color.

8. Run your program again to view the results, then stop your program completely.

TIP: It is a good idea to try changing the properties of any form object to see its effects on the form. Some object properties require a specific format, or valid range of values. To view the description of each property and what it does, click on the property and press the **'F1'** key on your keyboard. This will open the **'Help File'** which describes the property and also the valid range of values.

TIP: The Visual Studio **'Help File'** requires an Internet connection to connect to the main **MSDN TechNet Knowledge Base**. Accessing the **'Help File'** is faster if it has been installed locally in your computer. The option for installing the Help File will appear at the end of the standard installation of Visual Studio.

PART 3: Editing Visual Basic Source Code and Using Form Events

A. Viewing VB Source Code

1. All user-editable visual basic source code is contained in the form that the code is written for. To view the source code, in the **'Solution Explorer'** tab, right-click on the form **'frm_splashscreen_a123456.vb'** and click **'View Code'**.

2. Alternatively, you can click on the form name in the **'Solution Explorer'** and press the **'F7'** key on your keyboard.

3. The main window will switch to the source code in **'frm_splashscreen_a123456'**. You will see the automatically generated **'code snippet'** for this form as shown below:

```
Public Class frm_splashscreen_a123456

End Class
```

INFO: This VB **'code snippet'** represents the only user editable area of the VB source code in a particular form. It is not the only source code, as Visual Studio will automatically generate other source code for controlling the Windows form display and other programming artefacts. Users are advised to edit source code only in these VB code snippets, and not alter any of the automatically generated code. Alteration of any of the automatically generated code will cause code corruption, which is usually irreversible. The only way to recover from the code corruption is to delete the entire form and restart from the beginning.

B. Automatically Generating Code Snippets for Form Object Events

1. Visual Basic code is normally executed during **'Form Events'** or **'Object Events'**. An **'Event'** is an **'activity', 'action' or 'time'** that will trigger the execution of user-entered code.

2. Double clicking on any form object while in the form design window will automatically generate the code snippet for the object's default event.

3. The default event for the 'Form' object, is the **'Form_Load'** event. To generate its code snippet, switch to the form design window, and double click on any empty area of the **'frm_splashscreen_a123456'** form. The following code will be automatically added:

```
Public Class frm_splashscreen_a123456

        Private Sub frm_splashscreen_a123456_Load(sender As System.Object, e as
        System.EventArgs) Handles MyBase.Load

        End Sub

End Class
```

4. Any code entered in between the **'Private Sub...'** and **'... End Sub'** of this **'Form_Load'** event will be executed exactly when the form is opened, or just before the form is displayed.

TIP: At the top of the main window, you will see the **'Load'** event with a lightning icon. Additional code snippets for other events can be automatically generated by clicking on the drop-list next to this icon, and selecting another event. It is a good idea to try any of the other events to see when exactly the code that you write within the event is executed.

C. Using Message Boxes as System Prompts

1. A **'Message Box'** or **'MsgBox'** is a dialog box that pops-up to alert the user each time it is called.

2. To create a message box that appears before the **'frm_splashscreen_a123456'** form is loaded, type the following in between its **'Form_Load'** code snippet:

```
Private Sub frm_splashscreen_a123456_Load(sender As System.Object, e as System.EventArgs)
Handles MyBase.Load

        MsgBox("Welcome!")

End Sub
```

3. Run the program to see the results. You will see the **'Welcome!'** message box pop-up before the form is displayed.

4. Click **'OK'** to close the message box and stop the program completely after you are done.

D. Commenting Out Code

1. If you have code that you no longer want VB to execute, but you do not feel like deleting the code, you can comment the code out.

2. Commenting in VB is done by placing the **apostrophe** (') symbol or **single-quote** in the beginning of the code line, as follows:

```
'MsgBox("Welcome!")
```

3. Alternatively, if you have multiple lines of code that you would like comment out, highlight all the unwanted lines, and click the **'Comment'** button on your toolbar.

4. To uncomment, or to recover commented code, simply remove the apostrophe from the beginning of the code line, or press the **'Uncomment'** button on your toolbar.

> TIP: It is a good programming practice to place comments in your code whenever you write your own functions, sub-procedures, or whenever you are coding a complex process flow, to remind yourself of what your function or code will do, what are the necessary inputs, and other things that you might forget in the future. Adding comments will not add to the execution load of your program.

E. Getting User Input from Object Properties

1. User input is obtained directly from the current properties of a particular object.

2. For example, you have previously changed the **'Text'** property for the TextBox **'txt_name'**, i.e. **'txt_name.Text'**, to **'Anonymous'**. You can use this property reference directly as user input.

3. In the **'frm_splashscreen_a123456.vb [Design]'** tab, double click on the **'btn_start'** Button to automatically generate the code snippet for the default events for Buttons, which is the **'Button_Click'** event, as shown below:

```
Private Sub btn_start_Click(sender As System.Object, e as System.EventArgs) Handles
btn_start.Click

End Sub
```

4. Type the code to call a message box whenever the **'btn_start'** Button is clicked. This message box will use the current value of the **'txt_name.Text'** property as part of its displayed message:

```
Private Sub btn_start_Click(sender As System.Object, e as System.EventArgs) Handles btn_start.Click

        MsgBox("Welcome " & txt_name.Text & " to the UKM Faculty Record System")

End Sub
```

5. The **ampersand (&)** symbol concatenates or joins together two or more text strings. In the code above, the **'txt_name.Text'** property value will be joined in between the other strings in the message box.

6. Run the program, and click the **'START'** button. The message box should display:

```
Welcome Anonymous to the UKM Faculty Record System
```

7. Click **'OK'** to close the message box.

8. Now delete **'Anonymous'** from the text box, and replace it with your actual name.

9. Click the **'START'** button again. The message box should now display the welcome message which now has your name in it.

10. Stop the program completely to return to the source code.

F. Spanning Code Over Multiple Lines

1. Visual Basic does not have a termination character like, C, C++, or Java, to indicate to the compiler when a line of code ends.

2. A line of code in VB terminates when the line ends.

3. This can cause some lines of VB code to become very long horizontally, and in some cases, unreadable.

4. To span VB code across multiple lines to make it more readable, use the **"Space Underscore"** (**' _'**) to break a line of VB code into multiple lines.

5. For example, you can break the message box code we wrote previously to two lines, as follows:

```
MsgBox("Welcome " & txt_name.Text _
            & " to the UKM Faculty Record System")
```

> TIP: You cannot break VB code to span multiple lines in the middle of a string text. In case you need to do this anyway, you will need to break the string into two or more strings, and join them together using ampersands (**&**). The "Space Underscore" (**_**) goes between the ampersands. For example:
>
> ```
>             "Hello there"
> ```
>
> ... can be broken in two, as follows:
>
> ```
>             "Hello " & _
>                         & "there"
> ```

G. Creating and Adding a New Form

1. In the **'Solution Explorer'** panel, right-click on the project name **'prj_facultyrecords_a123456' > 'Add' > 'Windows Form...'**.

2. The **'Add New Item'** window will open. In the **'Name:'** field, type **'frm_mainmenu_a123456.vb'.** Click **'Add'**.

3. Your new form will be added as an object under **'prj_facultyrecords_a123456'** in the Solution Explorer. If you view the properties of the form, the **(Name)** property and **Text** property will have automatically been set to **'frm_mainmenu_a123456'**.

4. Add a Label to the top of your **'frm_mainmenu_a123456'** form, and rename this label as **'lbl_menutitle'**. Change the Text property of **'lbl_menutitle'** to **'MAIN MENU'** and change the design of the label and the form as you feel fit.

5. Save your VB project.

H. Performing Form Operations

1. In the **'frm_splashscreen_a123456.vb'** tab, comment out or delete the MsgBox code.

2. Add the following code to the **'Button_Click'** event for **'btn_start'**:

```
Private Sub btn_start_Click(sender As System.Object, e as System.EventArgs) Handles btn_start.Click

        frm_mainmenu_a123456.Show()
        Me.Hide

End Sub
```

3. Run your program and click the **'START'** button. You initial Windows form, **'frm_splashscreen_a123456'** will close, and the new form **'frm_mainmenu_a123456'** you have just created will open.

INFO: **'.Show'** and **'.Hide'** are **'Methods'** that can be used to perform form operations. You may notice that the when using a method within a form, any reference to an external form must use the full form name. On the other hand, any reference to the form you are currently in, i.e. the form that is executing the form operation, must use **'Me'** instead of the form name as reference.

4. Stop the program completely to return to the source code.

TIP: When you execute form operations, such as switching between one form to another form, closing the form window using the **'X'** button only closes the current form but does not stop your project completely. This is because there may be a form that is just **'hidden'** instead of being **closed**, such as is the case here. To stop the program completely, click the **'Stop Debugging'** or **'Stop'** button on your toolbar.

5. Switch to the **'frm_mainmenu_a123456.vb [Design]'** tab. In the main form design window, add a Button at the bottom of the form.

6. Rename this new button as **'btn_exit'**. Change the Text property of the **'btn_exit'** Button to **'EXIT PROGRAM'**.

7. Double-click **'btn_exit'** to produce the code snippet for its **'Button_Click'** event.

8. Type the following the **'btn_exit_Click'** code snippet:

```
Private Sub btn_exit_Click(sender As System.Object, e as System.EventArgs) Handles btn_start.Click

        End

End Sub
```

9. Run your program and click **'START'** on the first window. When the second window opens, click **'EXIT'**. You will notice that the program stops completely this time.

I. Declaring Local Variables

1. Variables within VB events, sub-procedures and functions are declared using the **'Dim'** statement as follows (without the **< >** brackets):

```
Dim <variablename> As <datatype>
```

2. The variable name must begin with a character and can contain numbers. The variable name must not contain any 'spaces', punctuations (**! ? , ; : .** etc.), mathematical operators (**+ - * /** etc.), or any other special characters. All 'spaces' must be replaced by the 'underscore' (_) character.

3. Examples of commonly used variable declarations, according to the data type are as follows:

Data Type	Data Domain	Declaration Example	Declaration with Initial Value Example
String	Any text string, sentence, paragraph.	**Dim student_name As String**	Dim student_name As String **= "Ahmad"**
Char	A single letter, number, or symbol on a standard U.S. keyboard.	**Dim grade as Char**	Dim grade as Char **= "B"**
Integer	Any positive or negative integer between negative and positive 2,147,483,648	**Dim credit_hours As Integer**	Dim credit_hours As Integer **= 4**
Double	Any floating point number with double-precision	**Dim cgpa As Double**	Dim cgpa As Double **= 3.21**
Boolean	True or False	**Dim can_continue as Boolean**	Dim can_continue as Boolean **= True**

4. Variables declared using the **'Dim'** statement are accessible only within the structure, event, sub-procedure, function, or class where the declaration is made, or its members. The following is a rough guide of the accessibility of a variable declared using the **'Dim'** statement:

   - If a variable is declared within a **control structure**, such as a **'For...Next'** loop, the variable can only be used inside the control structure, or **'For...Next'**, loop itself.

   - If a variable is declared in an **event, sub-procedure, or function**, the variable can only be used by any code and control structure within the same event, sub-procedure, or function.

   - If a variable is declared in a **form class** or any other **class structure**, the variable can only be used anywhere within the same form class or class structure.

J. Using Local Variables

1. In the **'frm_mainmenu_a123456.vb [Design]'** tab, add a Label near the lower-right corner of the form, and rename it as **'lbl_date'**.

2. Double-click in any empty space in the **'frm_mainmenu_a123456'** form to generate its **'Form_Load'** code snippet.

3. In the **'frm_mainmenu_a123456_Load'** code snippet, type in the following:

```
Private Sub frm_mainmenu_a123456_Load(sender As System.Object, e as System.EventArgs) Handles
MyBase.Load

            Dim current_date As String = Date.Now

            lbl_date.Text = current_date

End Sub
```

4. Run your program and click **'START'** on the first window. When the second window opens, you will see the **'lbl_date'** Label change to show the current date and time.

> TIP: In the example above, you do not necessarily need to use the **'current_date'** variable, and you can assign the value of the Label directly by typing **lbl_date.Text = Date.Now** . You would generally only use a variable when more than one program statement or control structure needs to use the same variable value. Using a variable will also allow such code to give the same value any time it is referenced.

K. Creating Modules and Declaring Global Variables

1. If you want to create a variable that can be accessed by more than one form, you cannot declare the variable within a form. The variable must be declared as a global variable in a **'Module'**.

2. A **'Module'** is a blank VB code page that does not come with a Form or form objects. Modules can be used to place any global variables, global sub-procedures, or global functions.

3. In the **'Solution Explorer'** panel, right-click on the project name **'prj_facultyrecords_a123456' > 'Add' > 'Module...'.**

4. The **'Add New Item'** window will open. In the **'Name:'** field, type **'mod_globals_a123456.vb'**. Click **'Add'**.

5. Your new module will be added as an object under **'prj_facultyrecords_a123456'** in the **'Solution Explorer'**. Note that modules do not have any significant properties other than its **'File Name'** property.

6. In the **Solution Explorer** panel, double-click on **'mod_globals_a123456.vb'** to automatically generate its code snippet.

7. To declare a global variable, you must use a publically accessible variable declaration method such as **'Public'**:

```
Public <variablename> As <datatype>
```

8. By using **'Public'** instead of **'Dim'**, this variable can be accessed by any form, form object, or programming structure within the same project file.

9. In the **'mod_globals_a123456'** code snippet, type the following:

```
Module mod_globals_a123456

        Public username as string

End Module
```

10. Switch to the **'frm_splashscreen_a123456.vb '** tab, and add the following line of code inside the **'btn_start_Click'** code snippet, before any existing code:

```
Private Sub btn_start_Click(sender As System.Object, e as System.EventArgs) Handles btn_start.Click

        username = txt_name.Text

        frm_mainmenu_a123456.Show()
        Me.Hide

End Sub
```

11. Switch to the **'frm_mainmenu_a123456.vb [Design] '** tab, and add a new Label underneath the **'lbl_mainmenu'** Label. Rename this new Label as **'lbl_welcome'**. You may need to expand the width of your form.

12. Switch to the **'frm_mainmenu_a123456.vb '** tab, and add the following line of code inside the **'frm_mainmenu_a123456_Load'** code snippet, after all existing code:

```
Private Sub frm_mainmenu_a123456_Load(sender As System.Object, e as System.EventArgs) Handles
MyBase.Load

        Dim current_date As String = Date.Now

        lbl_date.Text = current_date

        lbl_welcome.Text = "Welcome " & username & " to the UKM Faculty Records System"

End Sub
```

13. Run your program and type your name in the first form, and click **'START'**. In the second form, you will note that the name your typed in the first form has successfully passed to the second form, when you used it in **'lbl_welcome.Text'**.

14. Click **'EXIT PROGRAM'** to stop your program completely.

15. Save your project.

L. Changing the Startup Form Properties

1. In your Solution Explorer panel, add a new Form to your **'prj_facultyrecords_a123456'** project, and name your new form **'frm_studentlist_a123456.vb'**.

2. Add a new Label in this form. Rename the Label as **'lbl_title'**, and change its **'Text'** property to **'Student List'**.

3. Add a **DataGridView** object to this form, and close any context window that appears. Expand the **DataGridView** object to fill almost the entire form. Rename the **DataGridView** object as **'grd_students'**.

4. In your **Solution Explorer** panel, double-click on the **'My Project'** object. This **'My Project'** view will be shown as a tab in the main window. It contains the configurations and settings that will be used whenever you compile, execute, or even publish your VB project.

5. In the **'Application'** tab, find the **'Startup Form:'** setting. Use the drop-list to change the starting form to **'frm_studentlist_a123456'** form.

6. Close this **'My Project'** view by closing the **'prj_facultyrecords_a123456'** tab in the main window.

7. Run your program. You will notice that the program will now start with the newly created **'frm_studentlist_a123456'** form.

8. Stop the program completely to return to the **'frm_studentlist_a123456 [Design]'** tab.

PART 4: Data Access Fundamentals

A. Relocating a Database File to the Project Folder

1. Save your project.

2. Minimize Visual Studio to show the Windows desktop.

3. Find the database that you created in Lab Notes 1. In the taskbar, click **'Start' > 'Computer' > 'C:' > 'a123456_facultyrecords'**. Right-click on the **'DB_FACULTYRECORDS_A123456.accdb'** file, and click **'Cut'**.

4. Open your VB project's **'Bin/Debug'** folder. To find your VB project's **'Bin/Debug'** folder, in the taskbar, click **'Start' > 'Computer' > 'C:' > 'a123456_facultyrecords' > 'prj_facultyrecords_a123456' > 'prj_facultyrecords_a123456' > 'bin' > 'Debug'**. Right-click in any empty space in this folder list and click **'Paste'**.

5. Your **'DB_FACULTYRECORDS_A123456.accdb'** database file is now relocated to your project folder. Remember this **'Bin/Debug'** folder location since you will be editing your database file directly from here from now on.

6. Maximize the Visual Studio window again.

> **IMPORTANT!:** You need to relocate your database file to your project **'Bin/Debug'** folder in order to eliminate any local directory path reference from your VB code. If you have any local directory path reference, for example:
>
> **'c:\a123456_facultyrecords\DB_FACULTYRECORDS_A123456.accdb'**
>
> ... this will create problems later on when you need to package and deploy your project into an **Installer**. By placing your database file in your **'Bin/Debug'** folder, you will only need to reference the database file name, and do not need to reference the path.

B. General Data Access Structure for Reading Data from a Database

1. For this course, we will be using OLEDB programming extensions within our VB code to connect to a Microsoft Access database, which should now be located in your project's **'Bin/Debug'** folder.

2. The minimum data structures needed to access data contained in the database are as follows:

   i. **The connection string**: This is a string that is specifically formatted to access a particular type of database or read a specific type of database file. [*Also applicable for any other programming language*]

   ii. **The SQL statement**: This is the Structured Query Language (SQL) operation that you will execute to read a specific set of data from the database. [*Also applicable for any other programming language*]

iii. **A DataTable object**: This is the temporary variable data structure that will be used to store the results of the SQL statement in memory. [*Only applicable for VB*]

iv. **A cursor object or Reader**: This is the core data structure that allows connectivity to the database file. [*Dependent on the programming language*]

v. **A Read or Fill command**: This is the command that will use the cursor to read through the database tables, and load the results of the SQL statement into memory. [*Dependent on the programming language*]

vi. **A display object or Writer**: This is the command that will show the results of the SQL statement in the user interface. [*Dependent on the programming language*]

C. Finding the Correct Connection String

1. Open the **Google** search engine, and type the following key words in the search field:

```
connection string accdb
```

2. Click on the first result, which is usually from **'www.connectionstring.com'**, that can be found on the following link:

```
https://www.connectionstrings.com/access-2007/
```

3. On the linked page, look for the connection string stated under **'Standard Security'**. It will show the following:

```
Provider=Microsoft.ACE.OLEDB.12.0;Data Source=C:\myFolder\myAccessFile.accdb;Persist
Security Info=False;
```

4. Copy this connection string, and temporarily paste it in **Notepad**. Edit is so that it is only in a single line.

5. Note that this connection string points to a sample database in **'C:\myFolder\myAccessFile.accdb'**. Since you do not have this sample database, replace this path with you database file name **'DB_FACULTYRECORDS_A123456'**. Your connection string should now look like the following:

```
Provider=Microsoft.ACE.OLEDB.12.0;Data Source=DB_FACULTYRECORDS_A123456.accdb;Persist
Security Info=False;
```

6. **IMPORTANT NOTE:** The spelling, spacing, and punctuation of your connection string must be exactly as shown above. Any misspelling will cause an error to occur whenever you try to run your code.

D. Preparing your SQL Statement

1. You will need to tailor your SQL statement to suit what you want to view as your database output.

2. For this tutorial, we will use a simple **'SELECT ALL'** statement to read the entire contents of the **TBL_STUDENTS_A123456** table from our database, as follows:

```
SELECT * FROM TBL_STUDENTS_A123456
```

3. **IMPORTANT NOTE:** The spelling, spacing, and punctuation of your SQL statement, especially attribute names, and table names must be correct. Any misspelling will cause an error to occur whenever you try to run your code.

TIP: In case you are rusty in your SQL-writing skills, you can use the **QBE (Query-By-Example)** or **Query Designer** in Microsoft Access to generate the SQL for you. Just design the query using the Query Designer and then switch to the SQL view. You can copy the entire line of SQL code displayed. Just leave out the final semicolon (;). The SQL code generated by Microsoft Access should be fully compatible with Visual Basic. However, the generated SQL code is usually much longer than normal, since Microsoft Access fully-qualifies all attribute references.

E. Reading Data from a Database into a VB Form

1. Copy your connection string from **Notepad**.

2. Re-open Visual Studio, and switch to the **'frm_studentlist_a123456.vb [Design] '** tab, and double click on any empty form area, to generate its **'Form_Load'** event code snippet.

3. In the **'frm_studentlist_a123456_Load'** code snippet, type in the following:

```
Private Sub frm_studentlist_a123456_Load(sender As System.Object, e as System.EventArgs) Handles
MyBase.Load

        Dim myconnection as String = "Provider=Microsoft.ACE.OLEDB.12.0;Data
        Source=DB_FACULTYRECORDS_A123456.accdb;Persist Security Info=False;"

        Dim mysql as String = "SELECT * FROM TBL_STUDENTS_A123456"

        Dim mydatatable as New DataTable

        Dim myreader as New OleDb.OleDbDataAdapter(mysql, myconnection)

        myreader.Fill(mydatatable)

        grd_students.DataSource = mydatatable

End Sub
```

4. Run your program. When the **'frm_studentlist_a123456'** form opens, the results of your SQL statement will be displayed in the form's grid.

5. Save your project.

TIP: The Connection String rarely changes in a VB project since you will most likely only use one database file. Therefore the connection string can be safely moved to the module **'mod_globals_a123456'** and be made into a global variable using the following variable declaration:

```
Module mod_globals_a123456

        Public username as string

        Public myconnection as String = "Provider=Microsoft.ACE.OLEDB.12.0;Data
        Source=DB_FACULTYRECORDS_A123456.accdb;Persist Security Info=False;"

End Module
```

You can then safely remove any local variable declaration for the same connection string (Dim mysql...) from all your form code. Later, if you need to change your database file name, you will only need to do it at one place.

EXERCISE 1:	Experiment with the code above to view different results by just altering the SQL statement written in the **'mysql'** variable. You do not need to change any other line of code, but you must make sure that your SQL statement is syntactically correct in structure and spelling.
EXERCISE 2:	Change your **Startup Form** properties back to **'frm_splashscreen_a123456'**. Add a new button in the **'frm_mainmenu_a123456'** form to open the **'frm_studentlist_a123456'** form. Rename the button and change its **'Text'** property as you see fit. Run the program to view the results.
EXERCISE 3:	Create additional forms to view each table in your **'DB_FACULTYRECORDS_A123456.accdb'** database. Add buttons to the main menu form to open each one.

Part 5: Before Leaving the Lab

1. Copy your entire working folder, e.g **'c:\a123456_facultyrecords'**, to your USB drive. This folder will be used again for the next lab session.

2. Delete the existing **'c:\a123456_facultyrecords'** folder in your lab computer's **'C:\'** drive.

Additional Lab Exercises

1. Modify the ‘Splash Screen’ form to display your full name and Matric Number

2. Customize your forms by modifying the following.
    - Use different Fonts for all your text and titles.
    - Change the text style and colour of all text fonts.
    - Change the colour of your form background and buttons.
    - Use pictures for the background of each of your forms.

3. Modify the overall appearance of all your forms so that your program no longer looks like a dull grey windows application.

4. Add useful and informative Labels throughout your forms that tell the user how to use each form in your program.

5. Change the text in a TextBox or Label when you hover your mouse over a button, picture, or text box.

6. Add a ‘Goodbye’ message when the user closes the program.