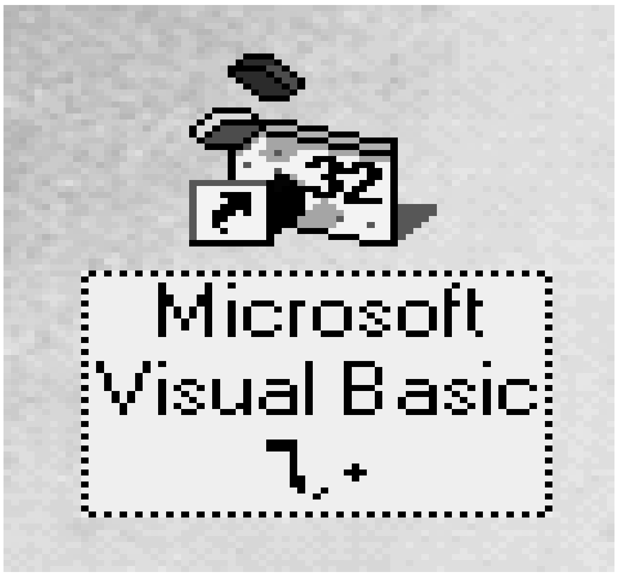
Microsoft
Visual Basic
6.0



Applied Science

Second class

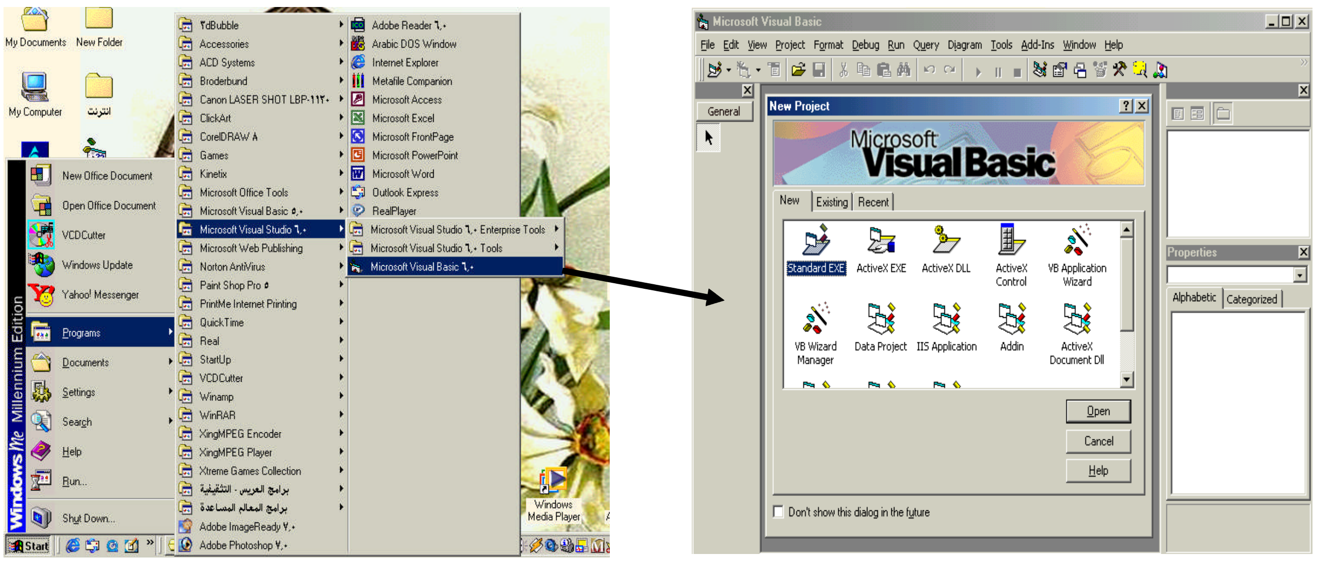
Prepared By
Assist. Lect. Alaa Omer Najim

1. Introduction

Visual Basic implements graphical user interface that allows the use of graphics for different applications. It provides visual interactive windows with user, like Dialogue box for (color, font ...), Input box, and Output box. Also it is able to create menu to simplify user application.

To run this program on user computer:
Start>programs>Microsoft Visual Studio 6.0>Microsoft Visual Basic 6.0.

It will appear on the computer screen as in the following picture.



To exit from Visual Basic and return to Windows is like exit from most Windows applications. There are three ways to close the Visual Basic as stated below.

1- Click on close button icon that appears in the upper-left corner of the screen.
2- Press Alt+F4
3- Select File >Exit

1.2-The Importance of Visual Basic Program

Languages like Basic and Pascal depend on variables and procedures to build the applications .This is why it is called procedural languages. The new approach is called object programming for visual programs like Visual Basic and Visual C++ and others. In this programming approach every thing (form, command buttons, controls) is an object.

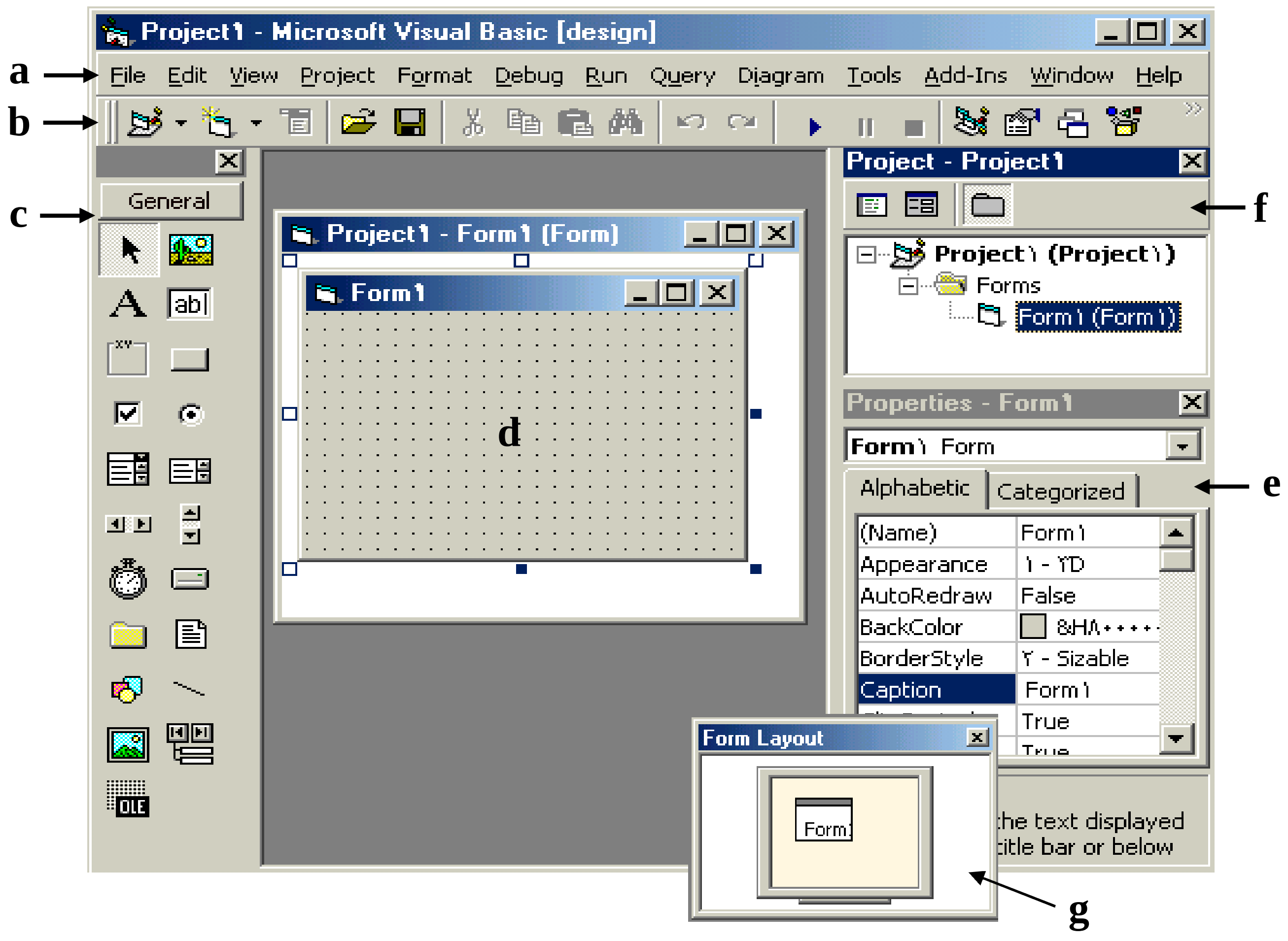
The reasons for of implementing Visual Basic program are listed as follows:

1- It uses integrated development environment (IDE) which is easier for the user to minimize code writing.

2- All visual programs follow the same concepts, therefore the user will become more familiar with visual approach for other visual languages.
3- It provides Input box and Output box as an interactive windows with user.
4- It is able to connect to Internet, and to call Explorer.

1.3- Elements of the Integrated Development Environment (IDE)

The IDE environment consists of many elements. Some elements are displayed when Visual Basic is started (By default) as in the following figure. Other elements are displayed if the user requires them. We will list some of these elements.



(Name)	Form١
Appearance	١ - ٣D
AutoRedraw	False
BackColor	&H٨٠٠٠٠٠…
BorderStyle	٢ - Sizable
Caption	Form١
	True
	True

a- Menu Bar: It contains a standard command like: File, Edit, View, Window, Help menus, and specific command such as: Project, Format, or Debug menus.

b- Toolbar: it contains the most commonly used commands (button), if clicked an action represented by that button is carried out.

c- ToolBox: it contains a collection of tools that are needed for project design.

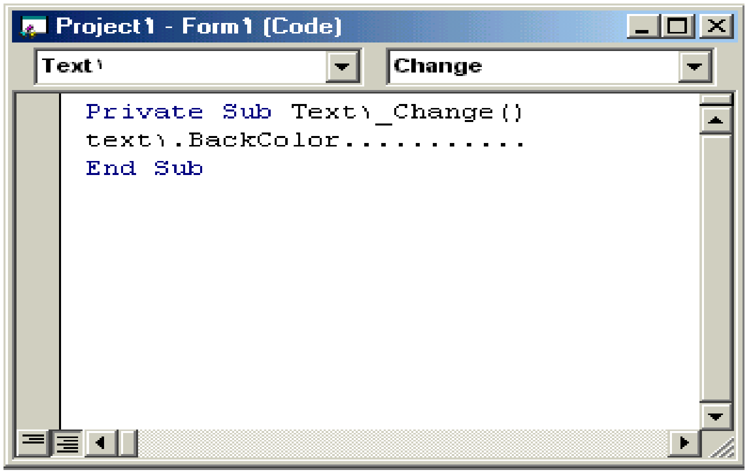
d- form Designer: it is a window for each form to customize the designed interface of the application. Using the form designer, the user can add controls, graphics, and text to create the desired form appearance.

e- Properties Window: it is a List of properties settings for a selected form or a control. These properties are characteristics (such as size, visible, or color) of the selected object it provides an easy way to set properties.

f- Project Explorer Window: it is a list of the forms and modules for the current projects. lt is a hierarchical tree- branch structure, where the project at top of tree and other parts like forms ,modules) descend from this tree.

g- Form Layout Window: The Form Layout window is a small screen. Which is used to reposition the form of the application so that it appears in proper place when project is run.

h- Code Editor Window: Code Editor Window is used to write a VB code for an application. For each form there is a separate code editor window. It is displayed when user clicks on form or object in form.



```
Private Sub Text1_Change()
text1.BackColor..........
End Sub
```

To Create an Application

The title of program includes the name of project, and when the user first starts the program it takes a defaulted value (projectl).It also includes resize icons. The following steps are required to create an application in Visual Basic 6.0:

1. Select type of project New or Exciting. A form automatically appears in the form design .The basis for any application's interface is the form that user should create. User can add other forms to the project (to add another form select project menu>add form).
2. To add objects (controls) to the form use the ToolBox.
3. Set the properties for the objects through properties window.
4. Write code. The Visual Basic Code consists of statements, and declarations. The code for an application can be written on the Code Editor window.In this window user can view and edit quickly any of the code.
5. Run the Application. To run the application, click the Start button onthe toolbar, or press F5.
6. Stop. To stop running the application and return to visual basic program click on stop button in tool bar.
7. Check if there is an error, return to step 3 ,otherwise continue.
8. Save project.
9. Exit.

Project

Project is a program designed to user application that may be simple (like calculator program) or complex (like word program). Visual basic program can create many types of projects. The most important or usual project is the standard project (for window applications) and the DHTML project (for internet).

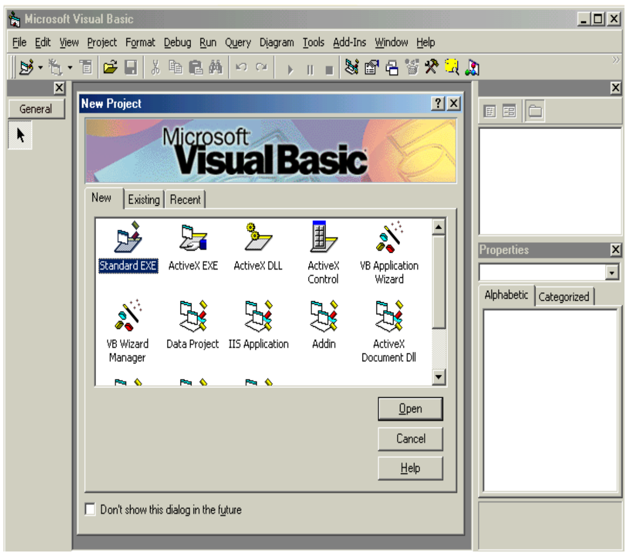
Working with Standard Projects:

The following working steps (create, save, add, open and delete) could be done:

a) To create project:

When program starts, project box appears-select Standard EXE > Project window appears.

OR: File> New project> Box (select Standard EXE)> Project window appears



b) To add project: Any number can be added.
Project icon> Select Standard EXE> Project window appears.

Note: Usually first project runs first, but user can change that by:
Selecting project from project window > mouse list > Set as startup.

c) To open an existing project:
It is previously designed and saved on disc in a folder.
File> Open project> Box (select existing and look for the project) > Project window

d) To delete a project:
Select project in Project window > Mouse list > Remove project.

e) To save project:
The visual basic can save the project on disc in two ways, as an executable type or a non- executable type.

I- for project in non execution stage:
There are many types of files summarized as follows:

1- Project file: it consists of all files which are related to specific project, also some other information with it. This could be saved with extension (.VBP)

2- The form Files: this contains form description and any Object or program related to it .This is saved with extension (.frm).

To save project for first time:
File>Save project (group) as>Box (project name)> forms saved then projects group saved.

To resave project: to save previously saved project in same place
File>save project (group)

Note: If a form is modified it should be saved. To save a form:
Select a form from Project window>File>Save project form1 as > Save box (select form name). OR: File>Save project forml.

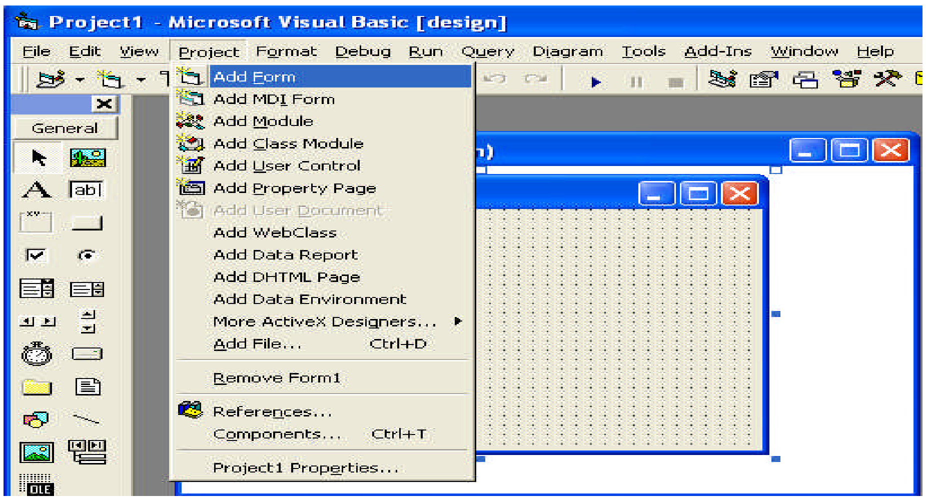
II- project for execution: This is the final stage so that it could be opened and run by Windows and no need for Visual Basic program. File> Make project.exe.

Item		**Action steps**	**Remarks**
Create project	New	File>New project	The user can open any number of projects.
	Exist	File>Open project	Project was already designed and saved.
	Recent	File>Open project	Project was recently designed and saved.
Save project		File>Save project group as	Visual Basic can deal with it (open and modify).
		File>Make projectl.exe	For execution by window.
Delete project		File> Remove project	Select project before remove.

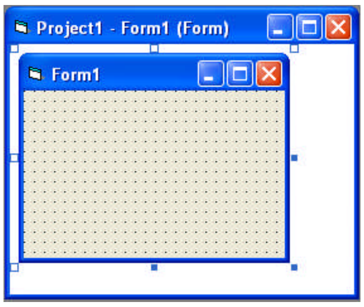
Forms

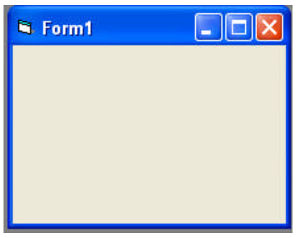
1) Introduction to form

The form is the most important visible object,without it no control can be displayed. It is a window that can be designed and modified to fit user applications. In the standard project the form Designer creates and modifies visual forms .When user starts visual Basic program a form is automatically displayed in Designer window. The designer can add any number of forms to the project of his application by pressing: add form from project menu.



There are two modes: design mode and running mode. User can interchange between them, by pressing on start icon ▶ or stop icon ■ on tool bar.





The forms also have properties and events.

2) Form properties

Properties list has a predefined value (numeric or string) and could be changed, some properties could be rewritten like caption, and some could be selected from option list by pressing on down arrow on the side.

Others could be rewritten or by browsing the computer files when the user clicks on the dotted button on the right side a dialogue box appears. The browsing button appears when the user clicks inside the box.

The most important properties of the form are listed in the following table:

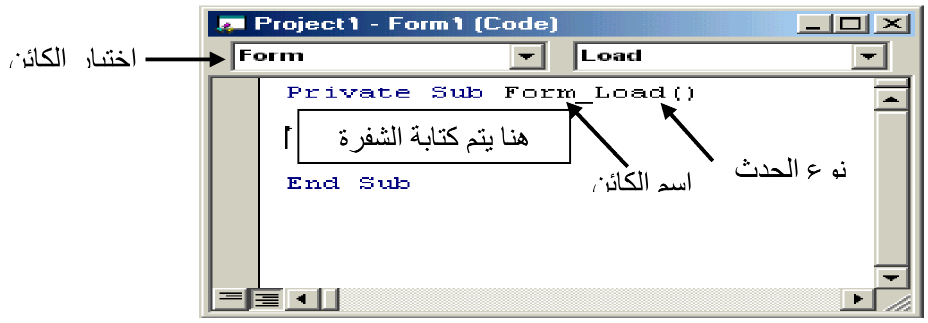
Property name	**Objective**	**code**	**Stage of Changing**
Name	Used to represent name of form in code		Design
Caption	String appear in title of form	$\text{Form}_{\text{no.}}$.caption= "any name"	Design and run
Backcolor	Background color for form.	$\text{Form}_{\text{no.}}$.Backcolor=Qbcolor(no.)	Design and run
Forecolor	Color of text written on form.	$\text{Form}_{\text{no.}}$.forecolor=Qbcolor(no.)	Design and run
Font	Font style, type and size.	Size: $\text{Form}_{\text{no.}}$.fontsize= no. Style: $font \begin{cases} italic \\ bold \\ underline \end{cases}$ Type: Form1.FontName = "arial"	Design and run
borderstyle	0-None borderless and captionless 1-Fixed Single a nonresizable form 2-Sizable (default), creates a resizable window, 3-Fixed Dialog: nonresizable form without Minimize and Maximize buttons 4-Fixed Tool window for a floating toolbox like form, 5-Sizable Tool window		Design
Enabled	The tools enable or disable.	$\text{Form}_{\text{no.}}$. Enabled =true or false	Design and run

Min button
Max button
	=true. The Minimize and Maximize buttons are enabled.
	=false. The Minimize and Maximize button are disabled.
		Form1.MaxButton = True or
		= false
		Form1.MinButton = True or
		= false
			Run

Start up position	0- Manual ,use form layer window to position Form 1- Center owner 2- at Center Screen 3- Windowdefault.		Design
movable	True or false to make form movable or unmovable		Design
Hide	To hide the form	$\text{Form}_{\text{no.}}$.hide	Run
show	To show the form	$\text{Form}_{\text{no.}}$.Show	Run
icon	Change the icon on title bar of form (the icon must have the extension ico or cur)		

3) Code form

The code is written in code Form and it will be edited quickly by code editor .The codes are of two categories:
1- Declaration is written before any procedure in the code
2- Statements. The user selects the required event then code statements are written inside these event procedures.



```
Private Sub Form_Load()

End Sub
```

5) Events:

Events are like electrical switches. The electrical switches are of many types, so are the events.

The forms and controls support events (generation, interaction with mouse and keyboard). The most important events for the form are described in the following table.

Event	Action taken when
Click	Single click on object.
DbClick	Double click on object.
load	Loading the object

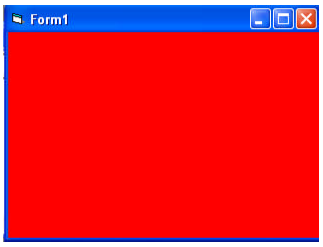
Examples:

1- Design a form such that: in event load, when project runs, the form backcolor property changed (chose any color).

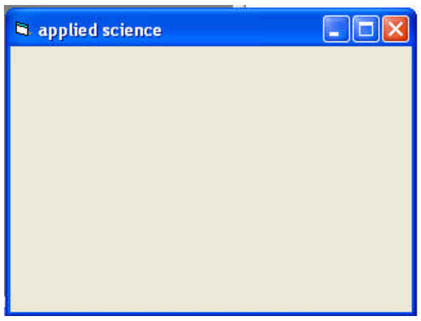
sol:

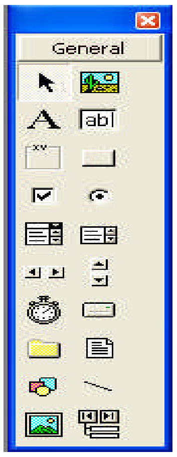
code:

```
Private Sub Form_Load()
Form1.BackColor = QBColor(12)
End Sub
```



2- Design a form such that: in event click on form, when project runs, the title of the form changed to applied science.





Toolbox

It is a window box that contains tools which could be used in the project. Tools are objects that could be selected from Toolbox to be placed on form. To show the toolbox, Press toolbox icon > the toolb ppear as in the following diagram.

The toolbox includes many tools and in general they are:

1) Pointer (not control) sed to select tools already on form
2) Picture box : Used to display images in any of the following formats: BMP, DIB (bitmap), ICO (icon), CUR (cursor), WMF (metafile), EMF (enhanced metafile), GIF, and JPEG.
3) Label Fixed text appears on form for remark.
4) Textbox For text edit .Like note pad.
5) Frame To group tools together (container).
6) Command button Used as a switch (such as OK and Cancel) buttons. Code is written in the *Click* event procedure of this control
7) Check box For a yes/ no (true /false) selection.
8) Option button : For selection as group. Many options are placed inside container (grouped) (a Frame control). One control is selected from the group all others of the group are automatically deselected.
9) Combo box onsists of (list and arrow when clicked a small a list appears), if user selects item from the list, it will be displayed in TextBox. Vertical size is fixed.
10) List box For a list, user adds to and deletes from this list. It takes any size.
11) Horizontal Scrollbar Create stand-alone Horizontal scroll bars.
12) Vertical Scrollbar Create stand-alone vertical scroll bars.
13) Timer Used to control object movement.
14) Drive List Box t is a special ListBox filled automatically with names, of the files in a specified directory. It is a list invariant.
15) Dir List Box s a special ListBox filled with drives (Hard disc, Flopy, CD) in the system. It is an invariant.
16) File List Box It is a special ListBox filled automatically with the names of all DirListBox. It is a list invariant.
17) Shape Used only to display rectangles, circles, and ovals on the forms. Never raises any events
18) Line Used only to display lines on the forms. It never raises any events.
19) Image : Used instead of PictureBox because it consumes fewer system resources.
20) Data used for data base.
21) OLE used for joining with another programs.

Tool Box and Form

The user can place the tool on form and then work with the tool. To place the tool on form:

Click on tool >Draw tool to Form> the tool appears on Form.

Or: double click on it.

Notes:

a) Each tool has a property window .To see this window: Click on tool on form> Property window appears.

b) Property can be changed manually or by code and the effect of code appears in the run time (when user runs project).

c) To put code for tool action:

Double click on tool > code sheet of the Form appears (with code of corresponding tool is written) > User write the desired code inside tool event, or outside in Form event.

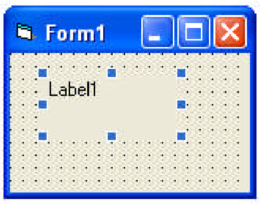
Working With Tools

The user can work with tool in the design stage.

- **To add tool**: double click on tool. Tool appears on form or drags it to design part of page and draw it in the desired size.
- **To delete**: click on element in page> press delete key of the key board or right click on object for mouse list> choose delete.
- **To display tool properties window**: click on element> properties window appear.
- **To display code form:** double click on tool code form for that element.

Label:

It is used to display fixed text on form



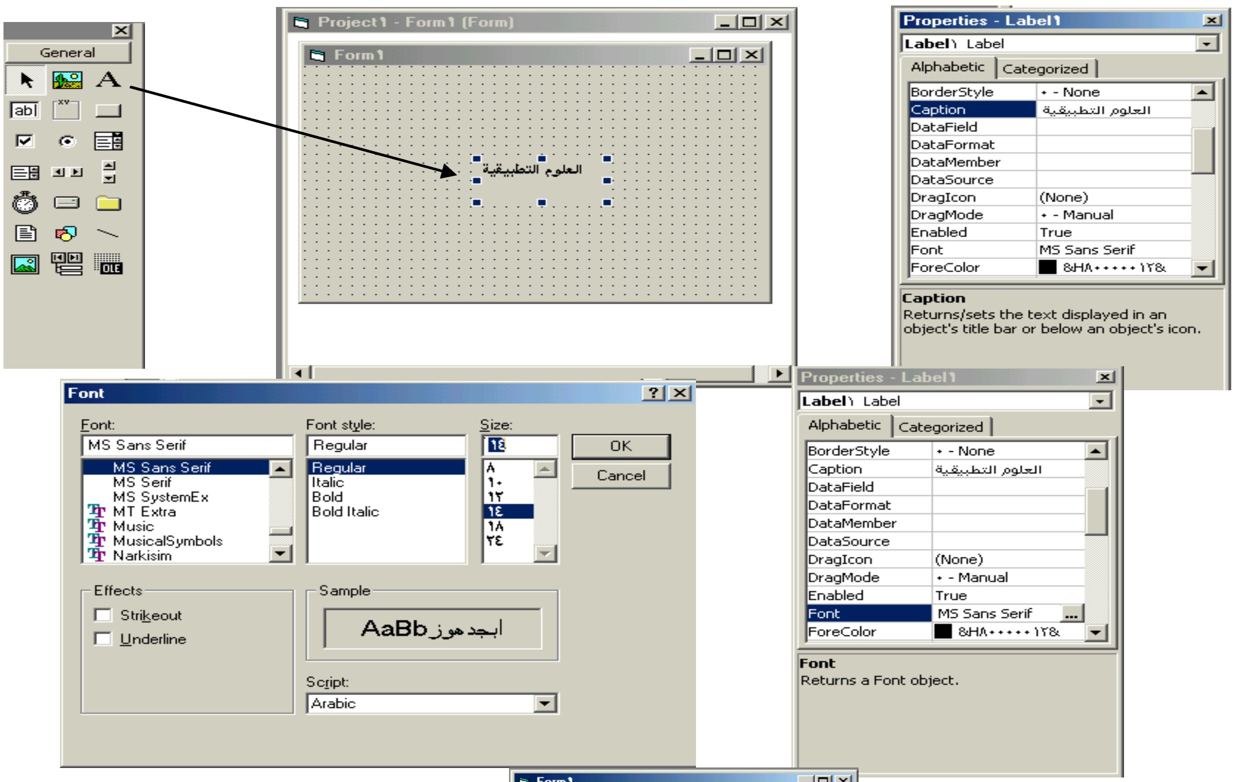
Property name	Objective	Code	Stage of Changing
Caption	String appear on label	$\text{label}_{\text{no.}}$.caption= "any name"	Design and run
Autosize	To resize tool to fit text	$\text{label}_{\text{no.}}$.autosize= true or false	Design and run
Backcolor	Background color for label	$\text{label}_{\text{no.}}$.Backcolor=Qbcolor(no.)	Design and run
Forecolor	Color of text written on label	$\text{label}_{\text{no.}}$.forecolor=Qbcolor(no.)	Design and run
Font	Font style, type and size	Size: $\text{label}_{\text{no.}}$.fontsize= no. Style: *font* { *italic* / *bold* / *underline* Type: label.FontName = "arial"	Design and run
visible	The label appear or disappear	Label_{no}.visible= true or false	Design and run
Enabled	The label enable or disable.	label $_{\text{no.}}$. Enabled =true or false	Design and run

Note: The available color numbers that used with QBcolor is the integers 0 to 15 only.

Example: Design a form contains label "العلوم التطبيقية" in size 14.

Sol: the properties are:

Label1	
caption	Applied science
fontsize	14



Properties - Label1

Label1 Label

Alphabetic | Categorized

BorderStyle	0 - None
Caption	العلوم التطبيقية
DataField	
DataFormat	
DataMember	
DataSource	
DragIcon	(None)
DragMode	0 - Manual
Enabled	True
Font	MS Sans Serif
ForeColor	&H٨٠٠٠٠٠١٢&

Caption
Returns/sets the text displayed in an object's title bar or below an object's icon.

Font

Font: MS Sans Serif — MS Sans Serif, MS Serif, MS SystemEx, MT Extra, Music, MusicalSymbols, Narkisim

Font style: Regular — Regular, Italic, Bold, Bold Italic

Size: ١٤ — ٨, ١٠, ١٢, ١٤, ١٨, ٢٤

OK | Cancel

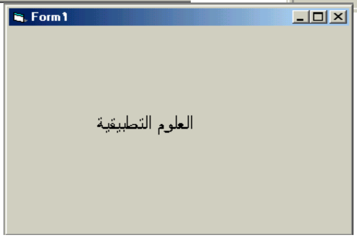
Effects: Strikeout, Underline

Sample: AaBb ابجد هوز

Script: Arabic

Font
Returns a Font object.

Running stage:



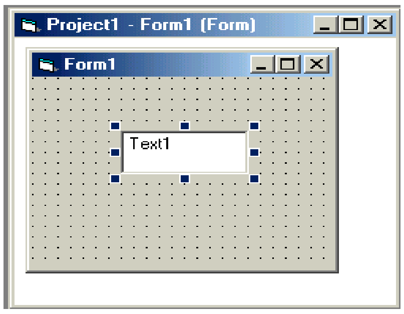
Textbox

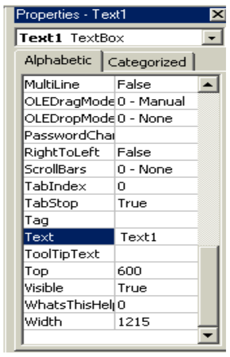
The textbox is a box for entering and displaying text (characters or values) in user project. This tool is used frequently in most of the application. The textbox has property window, with no caption, but with space for text. The most important property of this tool is the text content which is described in the following:

Property name	Objective	Code	Stage of Changing
Text	String appear on textbox	text $_{no.}$. text = "any name"	Design and run
multiline	To enter more than one line	true or false	Design
Backcolor	Background color for textbox.	text $_{no.}$.Backcolor=Qbcolor(no.)	Design and run
Forecolor	Color of text written on textbox.	text $_{no.}$.forecolor=Qbcolor(no.)	Design and run

Font	Font style, type and size.	Size: $\text{text}_{no.}$.fontsize= no. Style: $font\begin{cases} italic \\ bold \\ underline \end{cases}$ Type: label.FontName = "arial"	Design and run
visible	The textbox appear or disappear	text_{no}.visible= true or false	Design and run
Enabled	The textbox enable or disable.	$\text{text}_{no.}$. Enabled =true or false	Design and run
passwordchar	A row of symbols appear instead of letters	Text_{no}.passwordchar=(symbol)	Design and run
Setfocus	Put the focus on the specified textbox	Text_{no}.setfocus	Run

Change text manually: change text property from property window, click inside textbox and add text.




Properties - Text1
Text1 TextBox
Alphabetic | Categorized

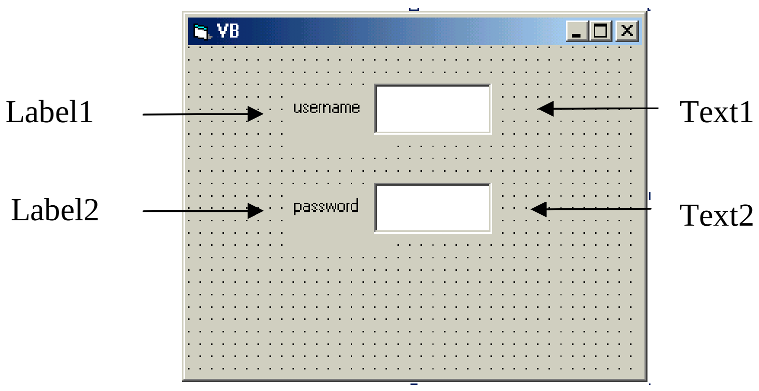
MultiLine	False
OLEDragMode	0 - Manual
OLEDropMode	0 - None
PasswordChar	
RightToLeft	False
ScrollBars	0 - None
TabIndex	0
TabStop	True
Tag	
Text	Text1
ToolTipText	
Top	600
Visible	True
WhatsThisHel	0
Width	1215

Change text by code:

1- Text1.text=" "
2- Text1.text=" نص "
3-Text1.text=text2.text
4- Text1.text=label1.caption
5- Text1.text = inputbox ("نص")

Example: Design a form to enter username and password such that the title of the form is VB.

Sol: design stage



Form1	
caption	V.B
Text1	
text	
Text2	
Text	
Label1	
caption	username
Label2	
caption	password

Example: Design a form with one textbox, set the text properties so that this massage appears when project runs (welcome to visual basic world).

Sol: There are two methods:

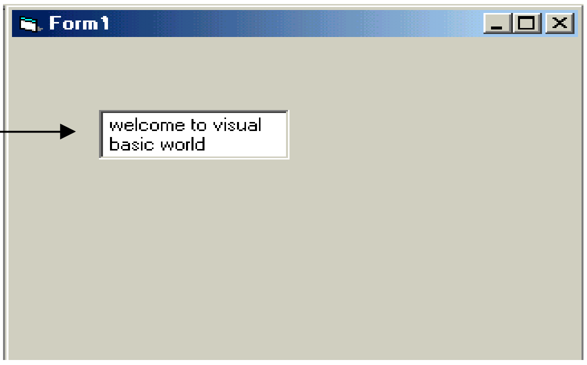
First method: changing property by code:

```
Private Sub Form_Load()
Text1.Text = "welcome to visual basic world"
End Sub
```

Second method: by properties window

Text1	
text	Welcome to visual basic world

Running stage

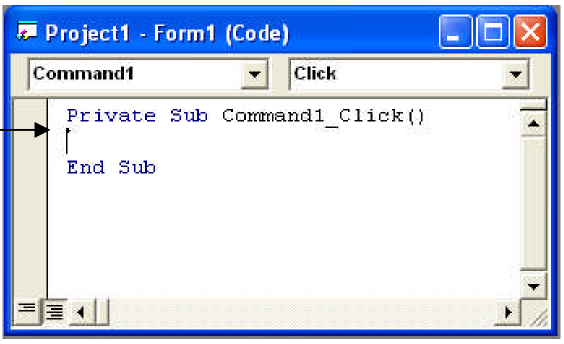


welcome to visual basic world

Command button

It acts as a switch. To deal with tool property> click on command button> property window appear> change setting of any desired property. Usually change set its caption property to a suitable string.

To make the button functional, the user should add some code. To do this: click on command tool> code form appears with click event procedure. Write code in this event or other events like press key event.



```
Private Sub Command1_Click()

End Sub
```

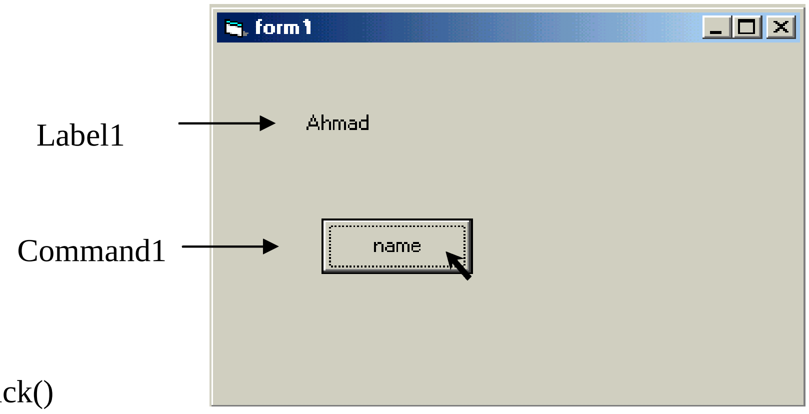
The most familiar properties that are needed for the command button are stated in the table below.

Property name	Objective	Code	Stage of Changing
Caption	String appear on command	command$_{no.}$.caption="any name"	Design and run
style	Determine the style of command	1-graphical 0-standard	Design
Backcolor	Background color for command	command$_{no.}$.Backcolor=Qbcolor(no.)	Design and run
Forecolor	Color of text written on command	command$_{no.}$.forecolor=Qbcolor(no.)	Design and run
Font	Font style, type and size	Size: command$_{no.}$.fontsize= no. Style: $font\begin{cases}italic\\bold\\underline\end{cases}$ Type: command$_{no}$.FontName = "arial"	Design and run
visible	The command appear or disappear	command$_{no}$.visible= true or false	Design and run
Enabled	The command enable or disable.	command$_{no.}$. Enabled =true or false	Design and run

Example: Design a form with label, such that when click on the command button "name" your name appears on label (at running stage).

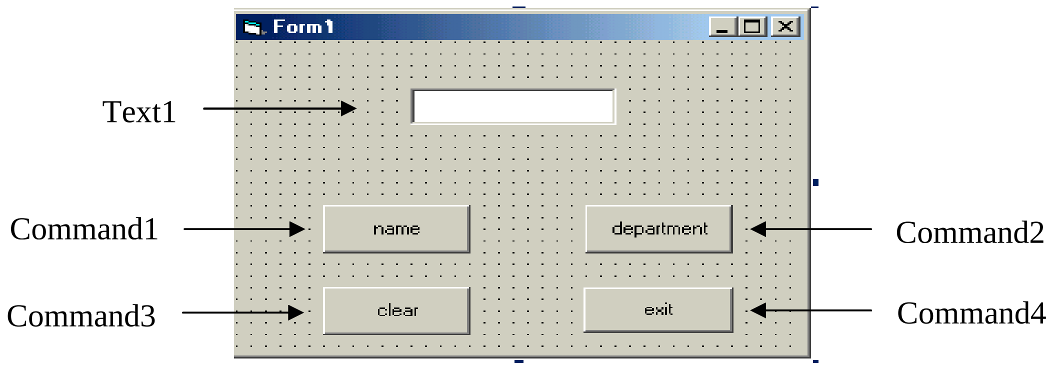
sol:

Label1	
caption	
Command1	
caption	name



```
Private Sub Command1_Click()
Label1.Caption = "Ahmad"
End Sub
```

Example: Design a form to appear your name and department in textbox, when click on command button "name" and "department" respectively so that you can clear these informations when click on command "clear" and stop project when click on command "exit".



Text1	
text	
Command1	
caption	name
Command2	
caption	department
Command3	
caption	Clear
Command4	
caption	exit

```
Private Sub Command1_Click()
Text1.text="Muna"
End Sub

Private Sub Command2_Click()
Text1.text="Science"
End Sub

Private Sub Command3_Click()
Text1.text=" "
End Sub

Private Sub Command4_Click()
end
End Sub
```

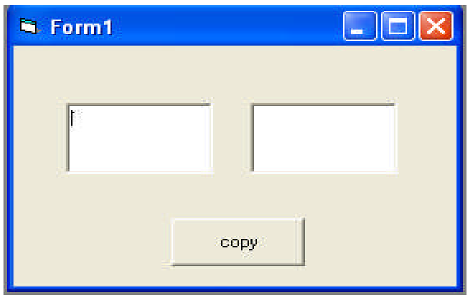
Example: Design a form contains two textbox so that when click on command button "copy" the text copied from first textbox to the second textbox but in size (28).

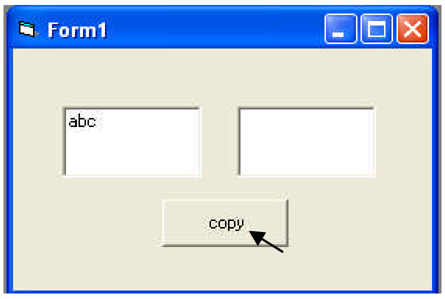
Sol:

Text1	
text	
Text2	
Text	
Command1	
caption	copy

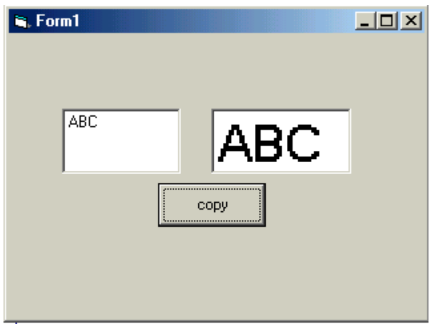
```
Private Sub Command1_Click()
Text2.Text = Text1.Text
Text2.FontSize = 28
End Sub
```

At run stage this window appear





If the user enter by example the text (ABC) in first textbox and click on command (copy) the same text appear on the second textbox but in size 28.

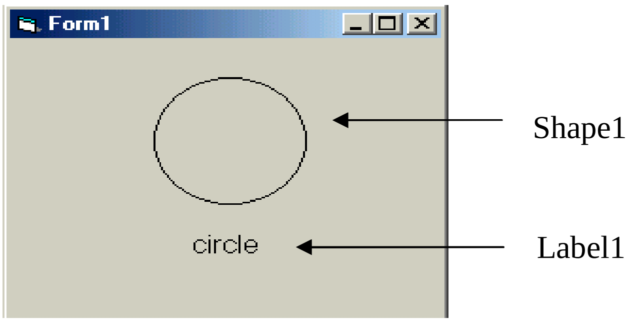


Shape: Shape is a tool used to draw geometric shape (circle, rectangle, square ,etc). It has property window. **It has no events like other tools** (such as click, dblclick, etc.).

Property name	Objective	Stage of Changing
shape	To determine a specific shape: 0-rectangle 1-square 2-oval 3-circle 4-rounded rectangle 5-rounded square	Design and run
backstyle	0-trancsparence 1-opaque	Design
Backcolor	colored the Background for shape, appear after backstyle changed to value 1-opaque	Design and run
bordercolor	Colored the Border of shape	Design and run

Example: Design a form contains a specific shape then write the name of this shape on form.

Label1	
caption	circle
Shape1	
shape	3- circle

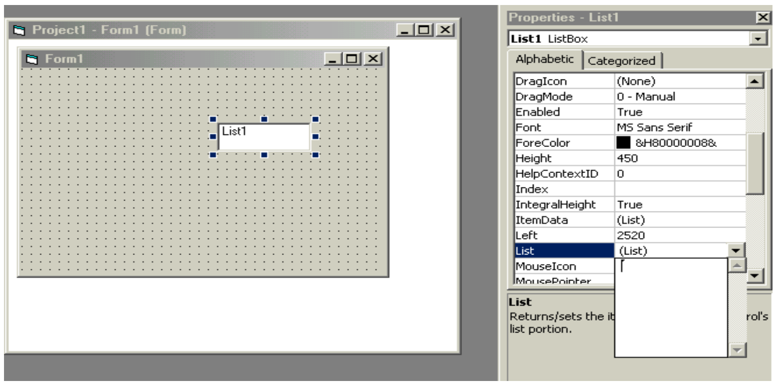


List box: The user can't write directly in ListBox . He can add item to the ListBox property or by code in the form.

Property name	Objective and code
sorted	True , to sort the elements alphabetically False , elements without sort.
Style	To determine the style of list: 0-standard 1-checkbox
Clear list	To clear all elements of the list: List_{no}.Clear

__Add items to list:__

a) Change property list from properties window. When click on arrow, write items (elements).



DragIcon	(None)
DragMode	0 - Manual
Enabled	True
Font	MS Sans Serif
ForeColor	■ &H80000008&
Height	450
HelpContextID	0
Index	
IntegralHeight	True
ItemData	(List)
Left	2520
List	(List)
MouseIcon	
MousePointer	

List
Returns/sets the it rol's list portion.

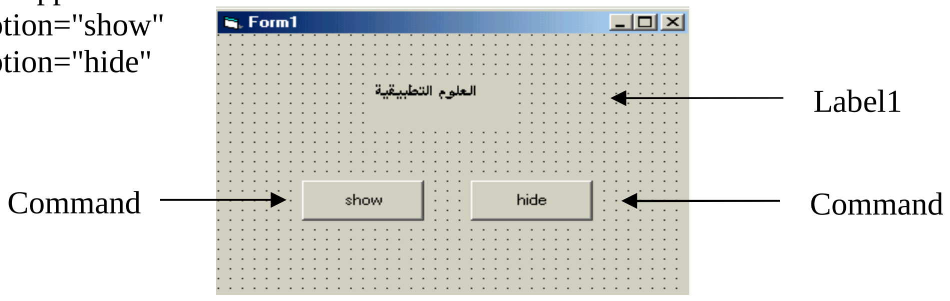
b) Add elements by code using the property additem as follows::

List$_{no}$.additem ("first element ")
List$_{no}$.additem ("second element ")
.
.
.
List$_{no}$.additem ("last element")

Example: Design a form contains label to display your department and two command buttons "show" and "hide" such that when click on command1, form2 appears and when click on command2, form2 disappears. In form2 design a list to contain the name of departments branches which appears after click on command button "display".

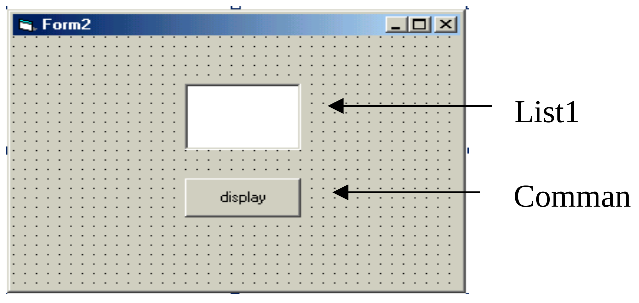
Form1

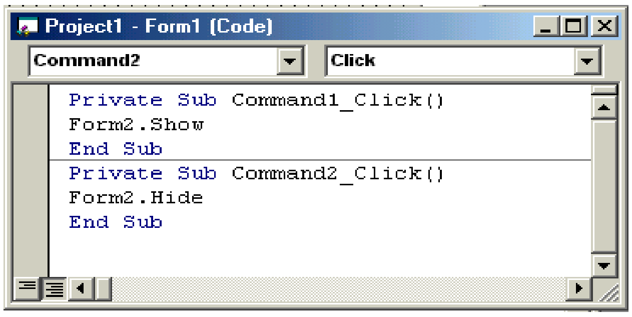
Label1: caption="applied science"
Command1: caption="show"
Command2: caption="hide"



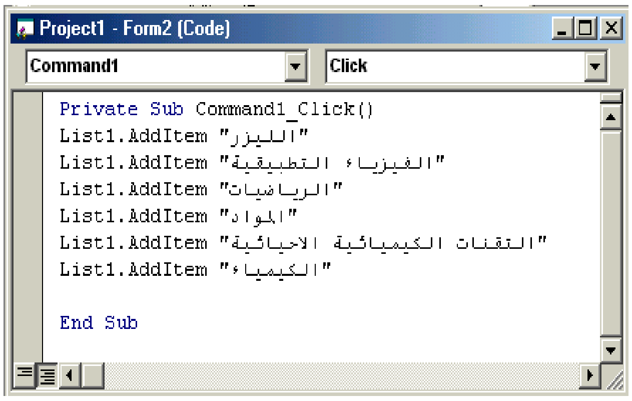
Form2

Command1: caption="display"
List1: list= فارغ





```
Private Sub Command1_Click()
Form2.Show
End Sub
Private Sub Command2_Click()
Form2.Hide
End Sub
```

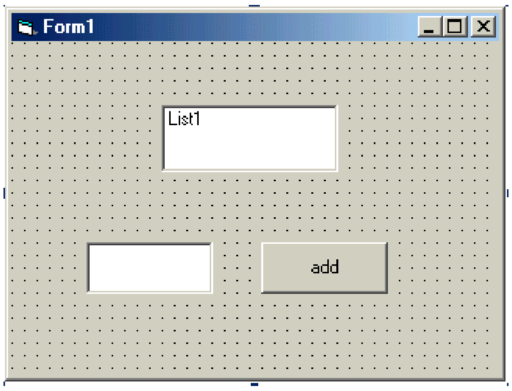


```
Private Sub Command1_Click()
List1.AddItem "الليزر"
List1.AddItem "الفيزياء التطبيقية"
List1.AddItem "الرياضيات"
List1.AddItem "المواد"
List1.AddItem "التقنات الكيميائية الاحيائية"
List1.AddItem "الكيمياء"

End Sub
```

Example: Design a form contains a sorted list alphabetically such that the user can add the item from text to the list after click on command button "add".

Sol:

List1	
list	
sorted	true
command1	
caption	add
Text1	
text	



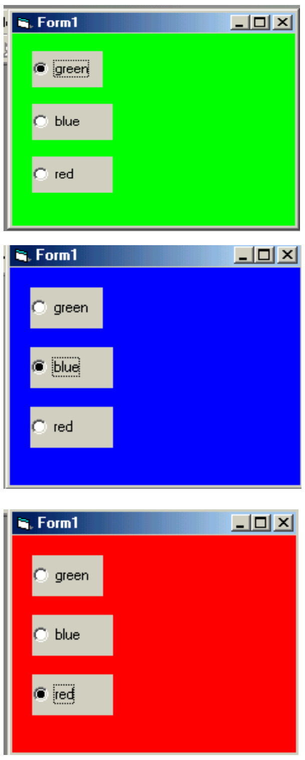
```
Private Sub Command1_Click()
list1.AddItem (Text1.Text)
Text1.Text = "   "
End Sub
```

Option button: Used only as a group of buttons. When the user selects one of them the others are deselected automatically.

All other properties of this control are similar to those in form and command button where they are fully discussed which are caption, font, enabled, backcolor and visible beside an important property which is value that takes true or false and it used with if statement. The option button usually takes click event.

Example: Design a form with three option buttons " red ", " green " and " blue " such that when we click on options the color of the form colored by red, green and blue respectively.

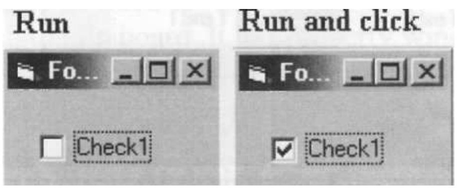
option1:caption	green
option2: caption	blue
option3: caption	red



```
Private Sub Option1_Click()
Form1.BackColor = vbGreen
End Sub

Private Sub Option2_Click()
Form1.BackColor = vbBlue
End Sub

Private Sub Option3_Click()
Form1.BackColor = vbRed
End Sub
```

Check box:

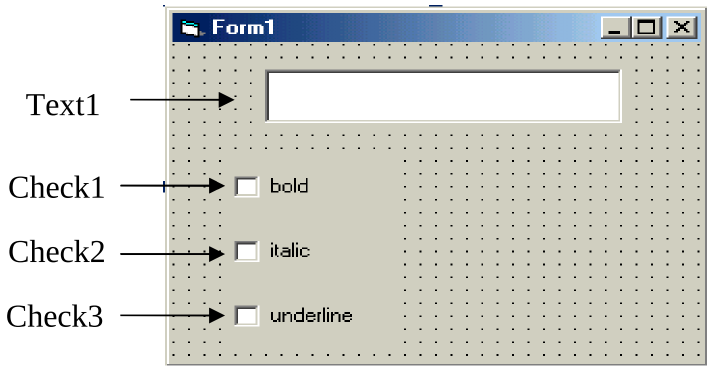
Any number of check boxes can be used on a form. They work independently. Its Property value could be changed in design stage manually, or in running stage by code.



Example: Design a form with one text box and three check boxes such that when click on boxes the following is done: change typing to bold, italic, underline.

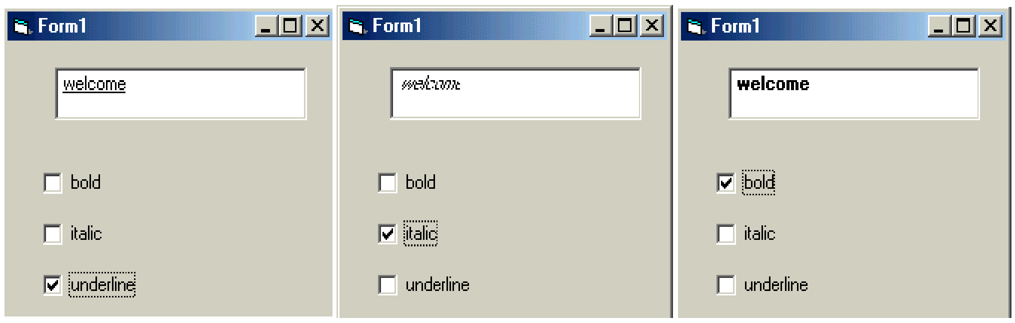
Sol:

Text1	
Text	فارغ
Check1	
caption	Bold
Check2	
caption	Italic
Check3	
caption	underline



```
Private Sub Check1_Click()
Text1.FontBold = Check1.Value
End Sub
Private Sub Check2_Click()
Text1.FontItalic = Check2.Value
End Sub
Private Sub Check3_Click()
Text1.FontUnderline = Check3.Value
End Sub
```

Run stage:



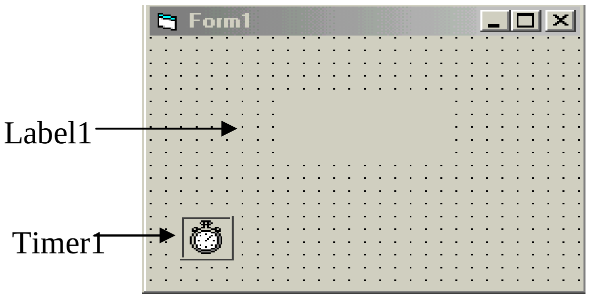
Timer
Timer returns the time in millisecond. It may be used to measure execution time of code (program efficiency).

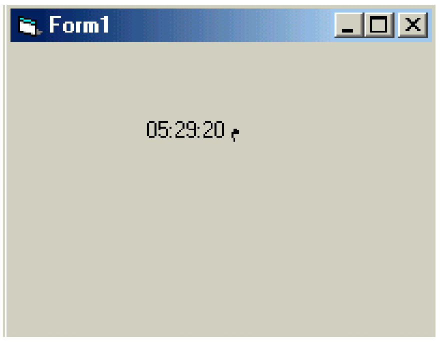
Property name	Objective and code
interval	To repeat the code according to event. It takes an integer values (0-65535) and measured in millisecond
enabled	timer$_{no.}$. Enabled =true or false

Ex: design electronic clock to display the time in seconds.
sol:

Timer1	
interval	1000
Label1	
Caption	

```
Private Sub Timer1_Timer()
Label1.Caption = Time
End Sub
```





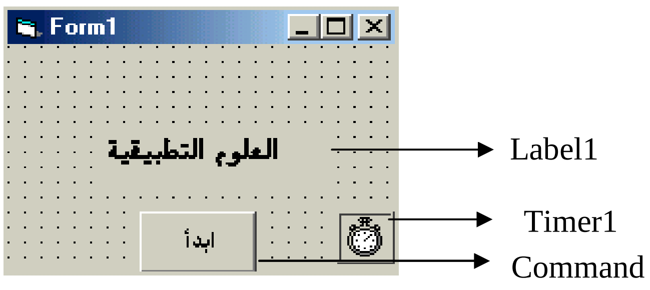
05:29:20 م

Example: Design a form to display "applied science" such that when click on command button "start" the color of "applied science" changed randomly every second.

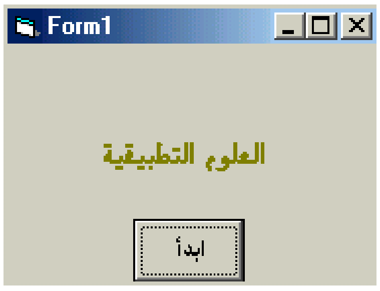
Sol:

Timer1	
interval	1000
enabled	false
Label1	
Caption	العلوم التطبيقية
Command1	
caption	أبدا

```
Private Sub Command1_Click()
Timer1.Enabled = True
End Sub
Private Sub Timer1_Timer()
t = Rnd * 15
Label1.ForeColor = QBColor(CInt(t))
```



End Sub
Run stage:
When click on command button ابدأ the color of the font will be changed every second randomly in integer no. (0-15).

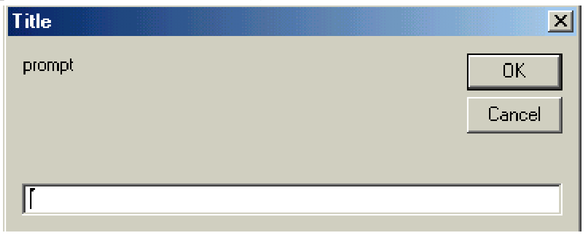


Note: the function (Cint) used to convert to integer no.
And (Rnd) used to generate a random no. in a range (0-1)

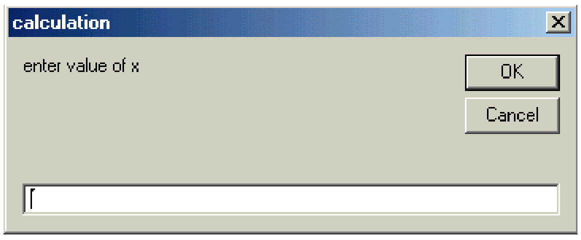
Input - output boxes

There are two types of dialog boxes which are inputbox and messagebox. The first is used to input variable and the second to output variable or message. Both needs code and appear at run time.

a) Inputbox
Inputbox used to input value or characters for one variable from keyboard at running stage.
This box needs a code in code sheet and could be written in any event or command
X=inputbox(" prompt or remark", "title")



Example: enter value of x using inputbox
Sol:



```
Private Sub Form_Load()
X=Inputbox("enter value of x", "calculation")
End Sub
```

Message box
It is used to output a message to the user (at running stage) the code needed could be written in code sheet and in any event or command.

The available icons for message box

structure	value	icon
vbcritical	16	❌
vbquestion	32	❓
vbexclamation	48	⚠
vbinformation	64	ℹ

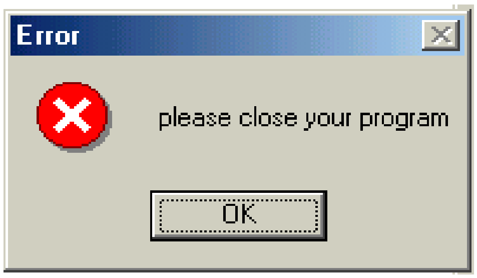
The available commands for message box

structure	value	Commands
Vbokonly	0	Ok
Vbokcancel	1	Ok, Cancel
vbAbortRetryIgnor	2	Abort, Retry, Ignore
vbYesNoCancel	3	Yes, No, Cancel
vbYesNo	4	Yes, No
vbRetryCancel	5	Retry, Cancel

For example if we write the following statement then a message box will be appear as shown below

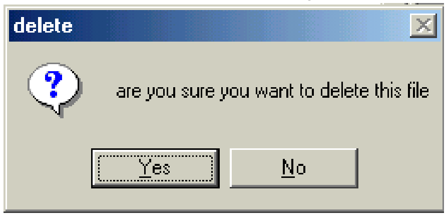
MsgBox "please close your program", 16, "Error"

or
MsgBox "please close your program", vbcritical, "Error"


Error
please close your program

Example: show what appear after running the following statement
MsgBox "are you sure you want to delete this file", 32 + 4, "delete"

or
MsgBox "are you sure you want to delete this file",vbQuestion+vbYesNo,"delete"
Sol:

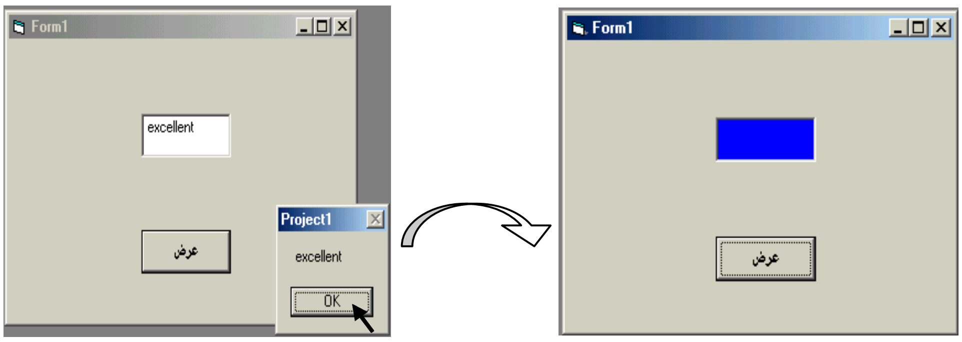

delete
are you sure you want to delete this file

Example: write a program to move the text (excellent) from textbox to message box and change the color of the text after click on command button (display).
Sol:

Text1: text="excellent"

Command1: caption="عرض"

```
Private Sub Command1_Click()
MsgBox (Text1.Text)
Text1.BackColor = QBColor(9)
Text1.Text = "  "
End Sub
```



Or we can write the following code:

```
Private Sub Command1_Click()
X= Text1.Text
MsgBox (X)
Text1.BackColor = QBColor(9)
Text1.Text = "  "
End Sub
```

Visual basic statements

In visual basic program (code) there are four basic parts, i.e. it is contains the following statements:

1- Declaration of variables and constants
2- Inputting variables
3- Operators for variables
4- Outputting variables

1- **Declaration of a variable and constants**
The declaration means defining the data type (variable or constant).

- **Variables**

A variable is a space in memory filled with data (value, character, time or date).

Notes:
- Variable name must start with character (not number or function) and maximum length 256 character, and does not contain point or symbol.
- Variable name must not repeat for other values.

The variable has to be declared. Variable type is defined by its content .The content may be data as numeric or character or string or Boolean or date, or any type of data (called variant), these types declared as:

Dim variable name **as** type
Or
Global variable name **as** type

Note: The **Dim** declaration written in general part of the form or in any place in form or sub procedure which used for one form. While **Global** declaration used for all forms

The types of variables that are allowed in visual basic are stated in the table below.

Types of variables

Type	**Value range**	**Declaration**
1-Integer	-32768<x<32768	Dim x as integer
2-Long	-2.1 e+009<x<2.1 e+009	Dim x as long
3-Single	1.4e-045 <|x|<3.4e+038	Dim x as single
4-Double	4.9e-324<x<1.79e+308	Dim x as double

5-String	65535 characters	Dim x as string
6-Boolean	True or false	Dim x as Boolean
7-Date	Computer time and date Jan 100<x< 31 Dec 9999	Dim x as date

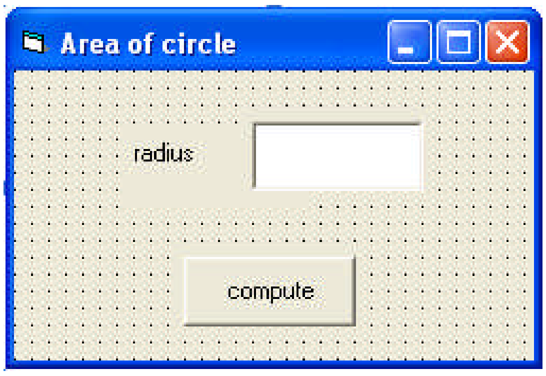
- **Constants**

It is a space in memory filled with fixed value that will not be changed. Constant may be declared as:

Const constant name **=** value

Example: Declare x as a constant (P), then compute the area of a circle. Put suitable design.

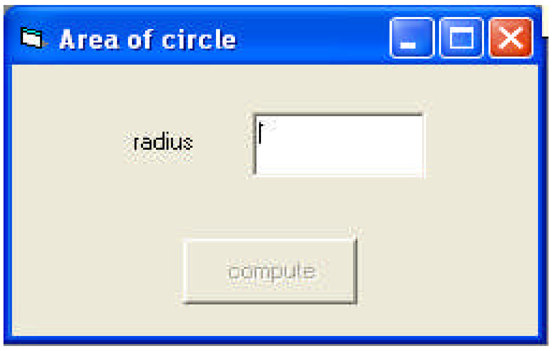
Sol:



Form1	
caption	Area of a circle
label1	
Caption	radius
Text1	
text	فارغ
Command1	
caption	compute
Enabled	false

code stage:

```
Const p = 3.14159
Dim a, r As Single

Private Sub Text1_Change()
Command1.Enabled = True
End Sub

Private Sub Command1_Click()
r = Val (Text1.Text)
a = r ^ 2 * p
MsgBox ("area=" & a)
Text1.Text = " "
Text1.SetFocus
End Sub
```



2- Inputting variables

There are methods to input variable x as stated in the following:

Method of input	For all type of variable
In text tool	X=text$_{no}$.text
In input box	X=inputbox("prompt","title")

Note: To enter many variables we usually use the second method with loop.

3- Operators for variables

The operators that are used for variable are described in the following table

Arithmetic operators	+	addition
	-	subtraction
	*	multiplication
	/	division
	mod	Modulus –rest of division
	^	exponent
Relational operators	=	equal
	<	Less than
	<=	Less or equal
	>	Greater than
	>=	Greater or equal
	<>	Not equal

Note: The order of operations when executing arithmetic operation is:
Exponentiation - multiplication division and mod - finally addition and subtraction.

The mathematical representation must be written as visual basic representation in the code as following examples:

Mathematical representation	Programming representation
3(x+4y)	3*(x+4*y)
$X^2 + 4 \div 2$	X^2+4/2
$\sqrt[4]{16} + 3^3 + 10 - 5 \times 4 \div 3^2 - 2^3$	16^(1/4)+3^3+10-(5*4)/3^2-2^3
$\frac{5y}{x^2-4} + x - 1$	(5*y)/(x^2-4)+x-1
$\frac{e^{2x}}{\cos(2x)+\sin(x)}$	Exp(2*x)/(cos(2*x)+sin(x))

Assignment statement

There are many statements ways to fill a variable as follows:
Variable = expression

Expression may include variables, operations and functions as follows:

1- Numerical variable. For example: i=3
2- Mathematical relation. For example: x=a/b
3- Characters variable (string). For example: t="abc"
4- Boolean variable (logical). For example: p=true

Functions for variables

The numeric and string variables are the most common used variables in programming, therefore V.B provides the user with many functions to be used with a variable to perform certain operations or type convention. The most common functions for numerical variable x

Function	Description
Abs(x)	Absolute of x
Sqr(x)	Square root of x
Int(x)	Integer of x
Exp(x)	Exponential of x (e^x)
Fix(x)	Take the integer part
Sin(x), cos(x), tan(x)	Trigonometric functions
Log(x)	Natural logarithms
Len(x)	Number of character of variable x
Lcase(x)	Change the text x to small letters
Ucase(x)	Change the text x to capital letters
Cint(x)	Convert x to integer
Clong(x)	Convert x to long integer
Cdbl(x)	Convert x to double precision
Cstr(x)	Convert variable x to string
Val(x)	Convert string x to numerical variable

Note: the last five functions are called conversion functions.

The following functions for different x are given for comparison.

Function	output
X=lcase("MY NAME IS")	my name is
X=ucase("my name is")	MY NAME IS
int(2.5)	2
Int(-2.5)	-3
Fix(2.5)	2
Fix(-2.5)	-2

4- Outputting variables

There are methods to output variable x as stated in the following:

Method of output	For all type of variable
On form	Print x Note: in load event we must use the statement: (form1.show)
to text tool	text$_{no}$.text =X
to label tool	Label$_{no}$.caption=x
By message box	msgbox (x) Or msgbox ("remark"& x)

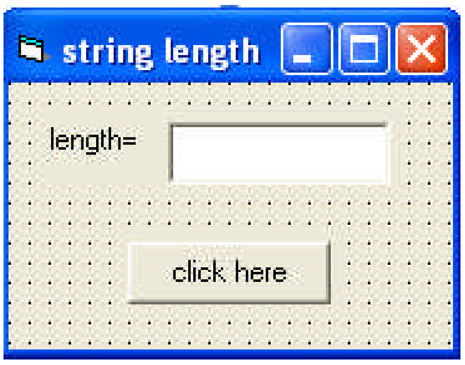
The instruction print could be very helpful to display data and used as follows:

Code	Description	example
print	To leave one line and print on next	
Print "a", "b", "c"	Use (,) to print a distance between outputs	a b c
Print "a"; "b"; "c"	Use (;) to print the outputs adjacent	abc
Print "a","b"; Print "c"	Print a, b then print c on the same line	abc

Example1: write a program to enter any text and compute its length. Put suitable design.
Sol:

Design stage:

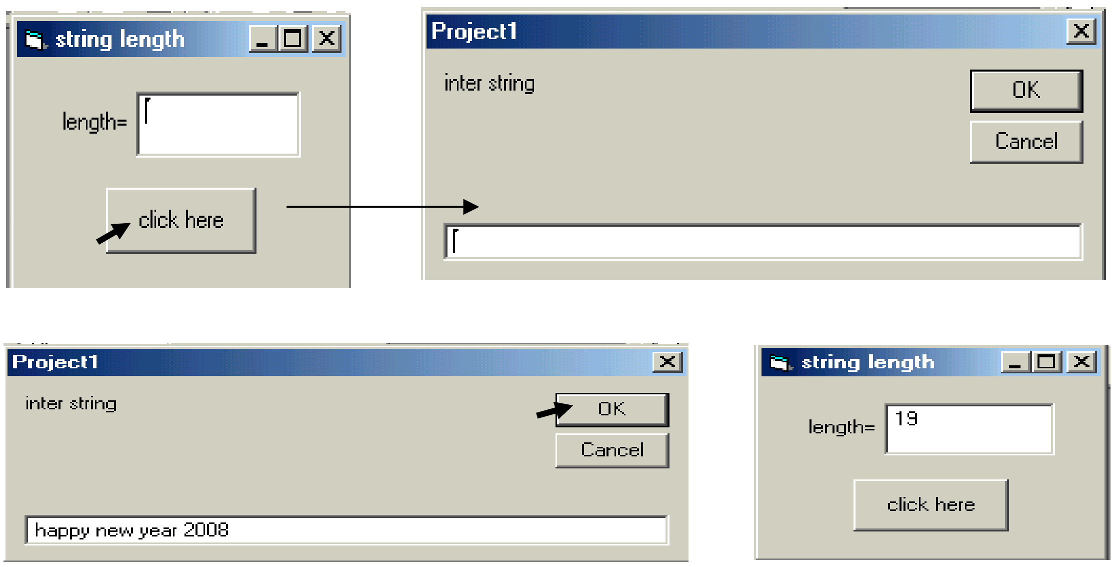
Form1	
Caption	String length
Command1	
caption	Click here
Label1	
Caption	Length=
Text1	
Text	



Code stage:

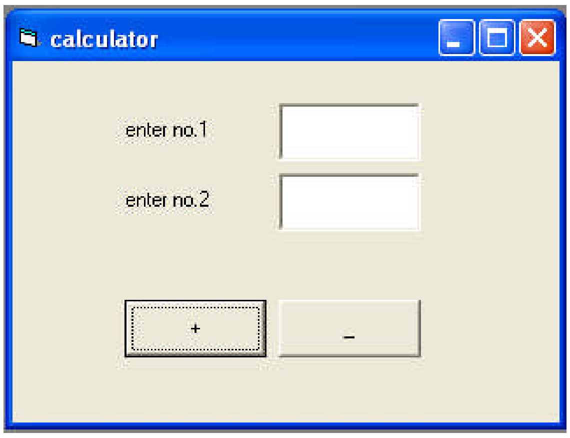
```
Dim s As String
Private Sub Command1_Click()
s = InputBox("inter string")
L = Len(s)
Text1.Text = CStr(L)
End Sub
```

Running stage:



Example2: write a program to add and subtract two integer numbers after putting a suitable design. Use message box for outputting.

Design stage:



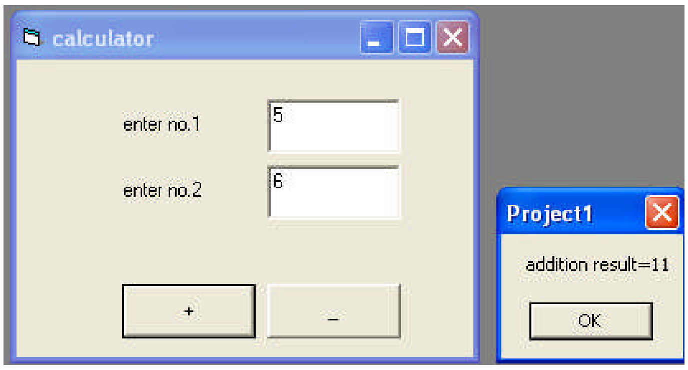
form	
caption	calculator
Command1	
caption	+
Command2	
Caption	-
Label1	
Caption	Enter no.1
Label2	
Caption	Enter no.2
text ونفرغ text1, text2 من خاصية	

Code stage:

```
Dim x, y, z as integer
Private sub command1_click ()
X=val(text1.text)
Y=val(text2.text)
Z=x + y
Msgbox("addition result="&z)
End sub

Private sub command2_click ()
X=val(text1.text)
Y=val(text2.text)
Z=x - y
Msgbox("subtraction result="&z)
End sub
```

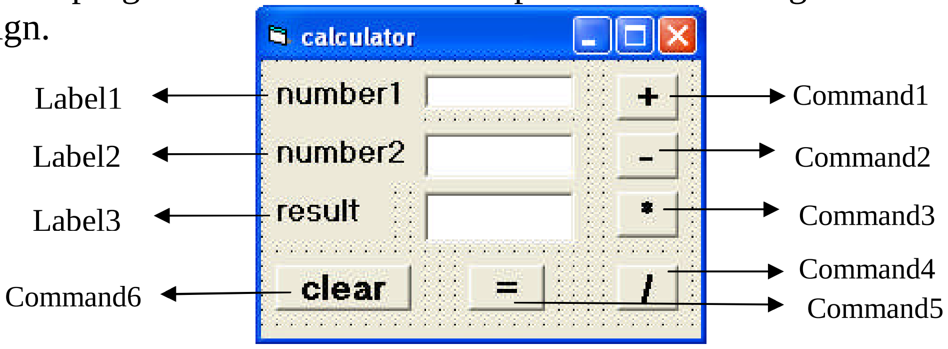
Running stage

Enter two values in text1 and text2. When click on command (+) or (-) the addition or subtraction result appears in message box.



addition result=11

Example3: write a program to execute the four operations according to the following design.

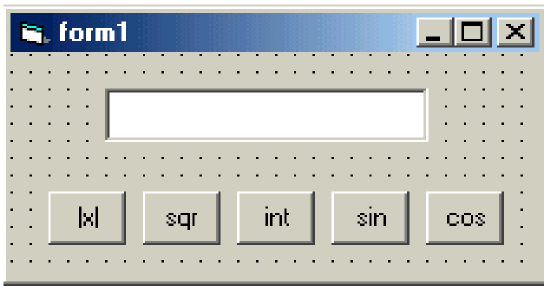


Code stage:

```
Dim a,b, c as single
Private sub command1_click ()
a=val(text1.text)
b=val(text2.text)
c=a + b
End sub
```

```
Private sub command2_click ()
a=val(text1.text)
b=val(text2.text)
c=a - b
End sub

Private sub command3_click ()
a=val(text1.text)
b=val(text2.text)
c=a * b
End sub

Private sub command4_click ()
a=val(text1.text)
b=val(text2.text)
c=a / b
End sub

Private sub command5_click ()
Text3.text=cstr(c)
End sub

Private sub command6_click ()
Text1.text=" "
Text2.text=" "
Text3.text=" "
End sub
```

Example4: write a program to compute the functions: sine, cosine, integer value, square, absolute value.
sol:

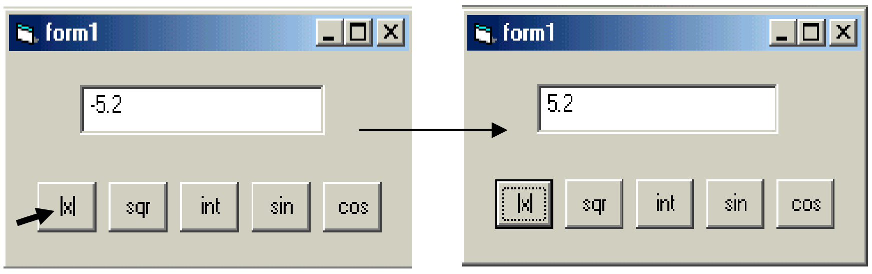


<u>Design stage:</u>

```
Dim x, y As Single
Private Sub command1_click()
x = Val(Text1.Text)
y = Abs(x)
Text1.Text = CStr(y)
End Sub

Private Sub Command2_Click()
x = Val(Text1.Text)
y = Sqr(x)
Text1.Text = CStr(y)
End Sub
```

```
Private Sub Command3_Click()
x = Val(Text1.Text)
y = Int(x)
Text1.Text = CStr(y)
End Sub

Private Sub Command4_Click()
x = Val(Text1.Text)
y = Sin(x * 3.14159 / 180)
Text1.Text = CStr(y)
End Sub

Private Sub Command5_Click()
x = Val(Text1.Text)
y = Cos(x * 3.14159 / 180)
Text1.Text = CStr(y)
End Sub
```

<u>Running stage</u>



Menu

The menu is a bar at the top of the form. The standard form is display without menu, but the user can add it. This menu could be included in form using menu editor. In next section the menu editor and the required code will be discussed.

Menu Editor

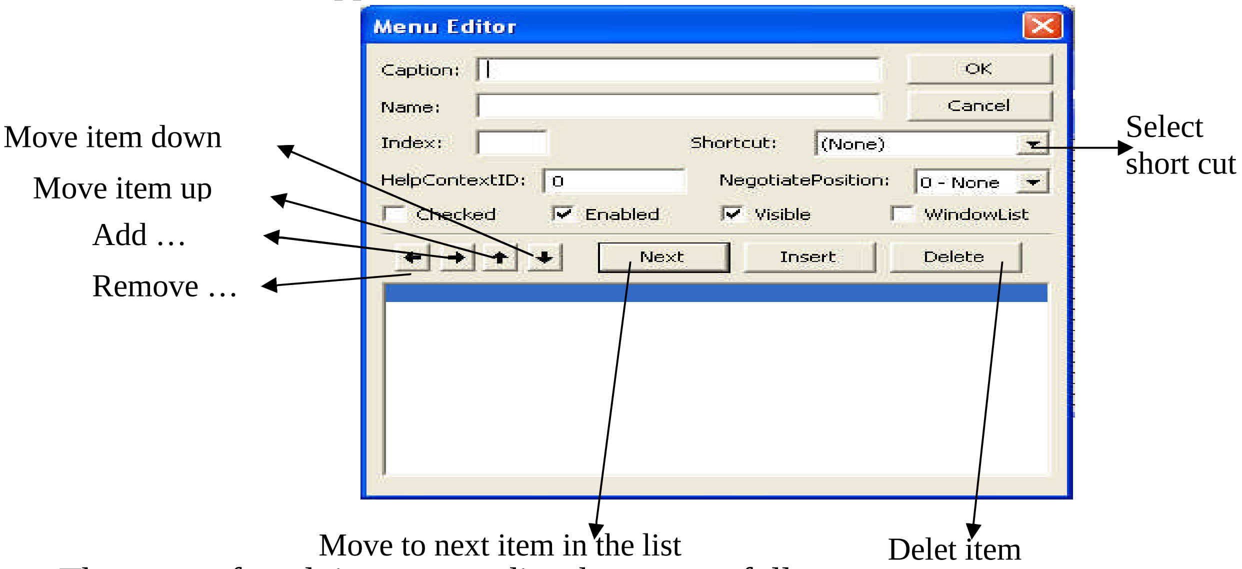
To use menu there are three ways:

1- Press menu icon from toolbar .

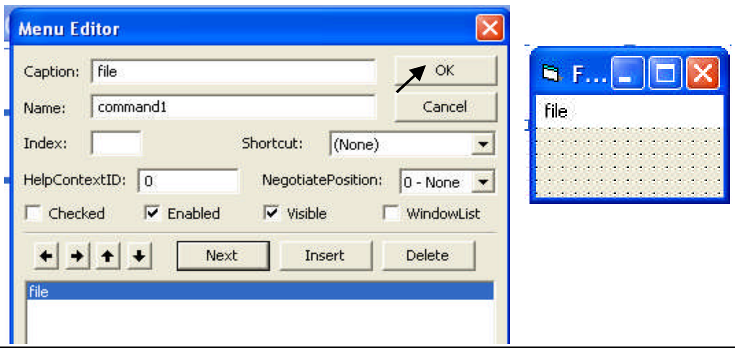
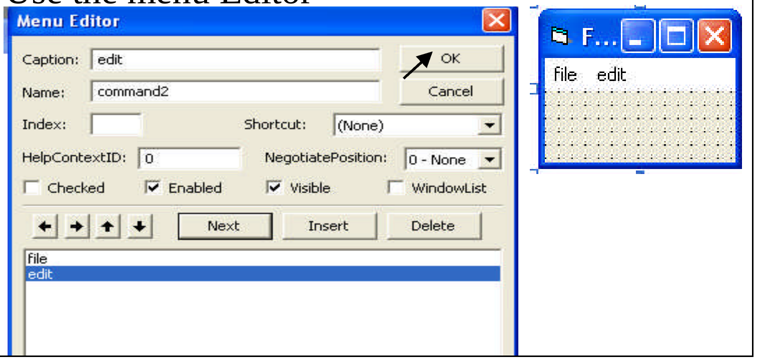
2- press (ctrl+E) from key board.

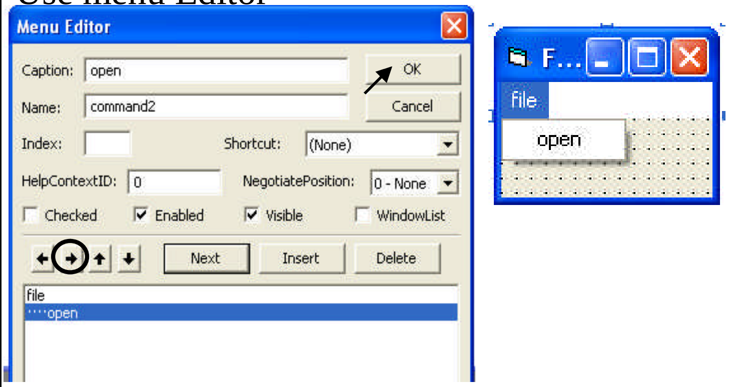
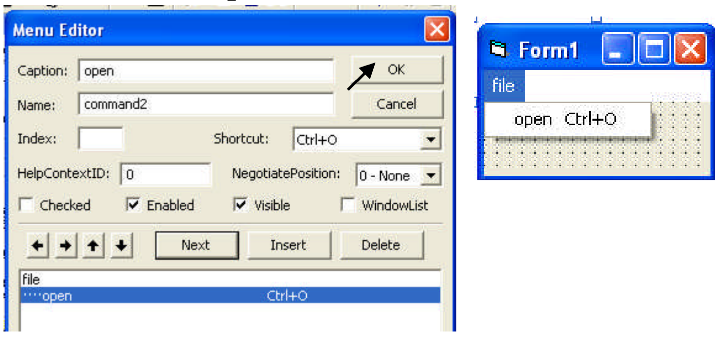
3- click on: tools>Menu Editor.

Menu editor box appears as shown below.



The steps of applying menu editor box are as follows:

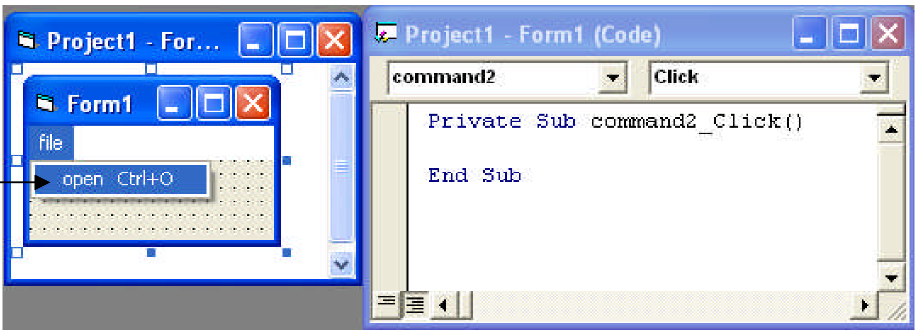
To add item	Use menu editor as follows  Caption: file; Name: command1; Index: ; Shortcut: (None); HelpContextID: 0; NegotiatePosition: 0 - None; Checked: no; Enabled: yes; Visible: yes; WindowList: no; List: file
To add many items	Use the menu Editor  Caption: edit; Name: command2; Index: ; Shortcut: (None); HelpContextID: 0; NegotiatePosition: 0 - None; Checked: no; Enabled: yes; Visible: yes; WindowList: no; List: file, edit

Create sub Menu list	Use menu Editor  Caption: open Name: command2 Index: Shortcut: (None) HelpContextID: 0 NegotiatePosition: 0 - None ☐ Checked ☑ Enabled ☑ Visible ☐ WindowList file ····open Menu Editor OK Cancel Next Insert Delete F... file open
Create Shortcut	Use & with caption and select shortcut  Caption: open Name: command2 Index: Shortcut: Ctrl+O HelpContextID: 0 NegotiatePosition: 0 - None ☐ Checked ☑ Enabled ☑ Visible ☐ WindowList file ····open Ctrl+O Menu Editor OK Cancel Next Insert Delete Form1 file open Ctrl+O

Code for menu items

Each item in menu or sub menu is considered as a command which takes the event click only. The user can add code for each item: click on item>code for that item appears on code sheet. Also code can be added to form: click on item >code for that item appears on code sheet. This is described in the following figure:

Click on item

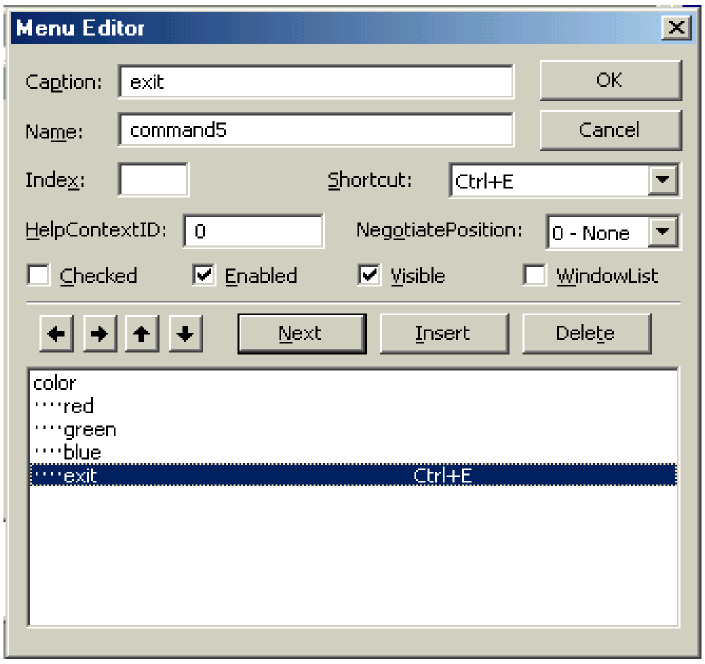


```
Private Sub command2_Click()

End Sub
```

Example: Design a form with menu and a label with a specific title. The menu contains one item color with sub menu items: red, green, blue and exit, to color the label in red, green, blue then exit from the program.
Sol: put label1 with any caption for example (hello)

Caption: color Name: command1	Create standard menu (color) from menu editor>next
Caption: red Name:command2	Add sub menu items by pressing → then enter the caption and name>next.

Caption: green Name:command3	Do the same thinks for other items>ok
Caption: blue Name:command4	
Caption: exit Name:command5 Shortcut: ctrl+E	


Menu Editor

Caption: exit
Name: command5
Index:
Shortcut: Ctrl+E
HelpContextID: 0
NegotiatePosition: 0 - None
☐ Checked ☑ Enabled ☑ Visible ☐ WindowList

```
color
····red
····green
····blue
····exit                    Ctrl+E
```

To programming these commands click on each one to open its code window and write the following code:

```
Private sub command2_click ()
Label1.backcolor=vbred
End Sub

Private sub command3_click ()
Label1.backcolor=vbgreen
End Sub

Private sub command4_click ()
Label1.backcolor=vbblue
End Sub

Private sub command5_click ()
End
End Sub
```

Conditional statements

There are two types of conditional statements:

1- If statement
2- Select case

1- **If** statement: The comparison operations are used with conditional statements.

The comparison operations are: (<, <=, >, >=, =, <>, and, or)

There are four structures for if statement.

a) Simple structure **If.. then**:

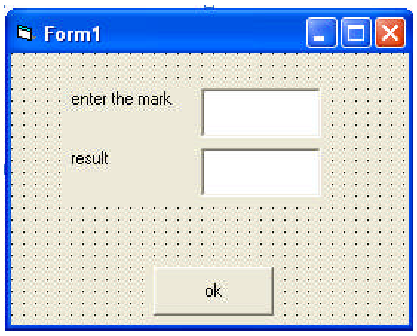
Used for running one programming statement only if the required condition satisfied.

The general form is: **If** condition **then** statement

Example 1: write a program to enter a mark of a student then print (pass) if he successful.

Sol:

```
Dim x as integer
Private sub command1_click()
X= cint(text1.text)
If x>= 50 then text2.text= "pass"
End sub
```



b) **If block** structure: Used for running many programming statements if the required condition satisfied.

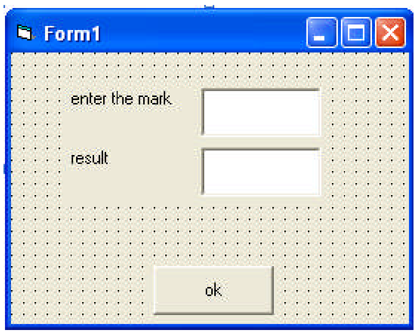
The general form is:

If condition **then**
Statements
End if

Example 2: write a program to enter a mark of a student then print (pass) in size 18 if he successful.

Sol:

```
Dim x as integer
Private sub command1_click()
X= cint(text1.text)
If x>= 50 then
text2.text= "pass"
text2.fontsize=18
end if
End sub
```



c) **If.. Then.. Else** structure: Used for running many programming statements if the required condition satisfied. And running another programming statements (after else) if the required condition not satisfied.

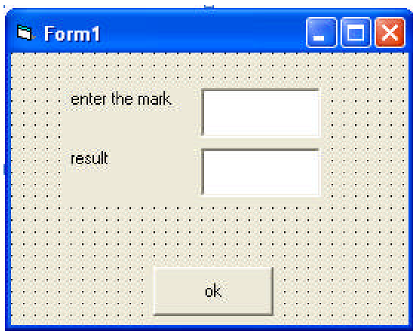
The general form is:

```
If condition then
Statements
Else
Statements
End if
```

Example3: write a program to enter a mark of a student then print (pass) if he successful and print (fail) otherwise.

Sol:

```
Dim x As Integer
Private Sub command1_click()
x = CInt(Text1.Text)
If x >= 50 Then
Text2.Text = "pass"
Else
Text2.Text = "fail"
End If
End Sub
```



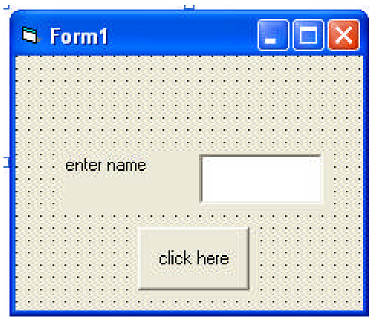
d) **If.. Then.. Elseif.. Else** structure:

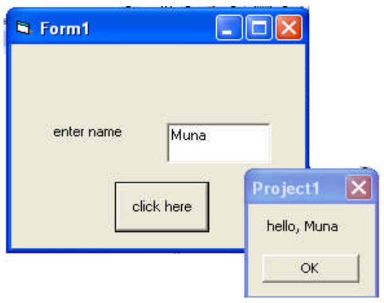
Used if we have many conditions to be satisfied

Example 4: write a program to enter a user name and display the message (hello) three times. The first one for (Muna), the second one for (Maha) and the third for any user as a guest.

Sol:

```
Dim x As String
Private Sub command1_click()
x = Text1.Text
If x = "Muna" Then
MsgBox "hello, Muna"
ElseIf x = "Maha" Then
MsgBox "hello,Maha"
Else
MsgBox "hello, guest"
End If
End Sub
```

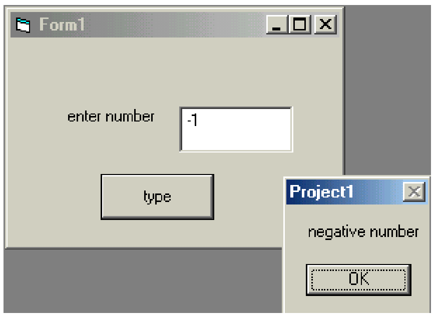


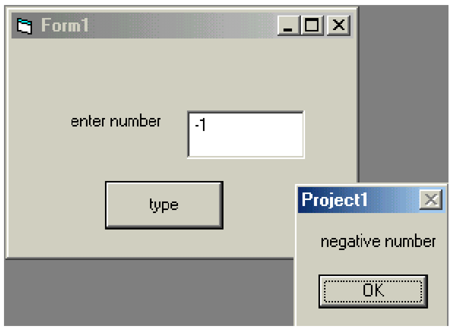


Example 5: Write a program to classify any entered number according to its sign and display the phrase (negative number) when the number is negative and the phrase (positive number) when the number is positive, otherwise display the phrase (neither positive nor negative).

Sol:

```
Dim x As Single
Private Sub command1_click ()
x = Val(Text1.Text)
If x > 0 Then
MsgBox "positive number"
ElseIf x < 0 Then
MsgBox "negative number"
Else
MsgBox "neither positive nor negative"
End If
End Sub
```


Form1
enter number -1
type
Project1
negative number
OK


Form1
enter number -1
type
Project1
negative number
OK

Nested If statement:

It can be takes the following structure:

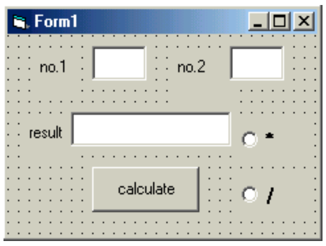
```
If  condition  then
      If condition then
            Statements        } structure2      } structure 1
      End if
End if
```

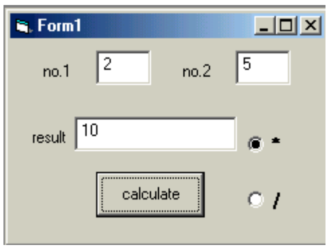
Note: Any structure of if structures can be used insteade of structure 1 and 2 above.

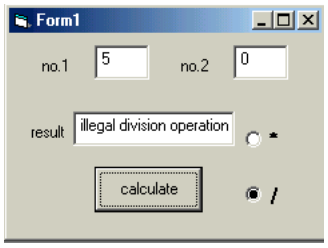
Example 6: Write a program to enter two numbers and compute multiplication and division operations using option button with display the phrase (illegal division operation) when the denominator is zero.

Sol:

```
Dim a, b, c As Single
Private Sub command1_click()
a = Val(Text1.Text)
b = Val(Text2.Text)
If Option1.Value Then
  c = a * b
  Text3.Text = CStr(c)
Else
```


Form1
no.1 no.2
result
calculate
○ *
○ /


Form1
no.1 2 no.2 5
result 10
calculate
● *
○ /


Form1
no.1 5 no.2 0
result illegal division operation
calculate
○ *
● /

```
  If b <> 0 Then
    c = a / b
    Text3.Text = CStr(c)
  Else
    Text3.Text = "illegal division operation"
  End If
End If
End Sub
```

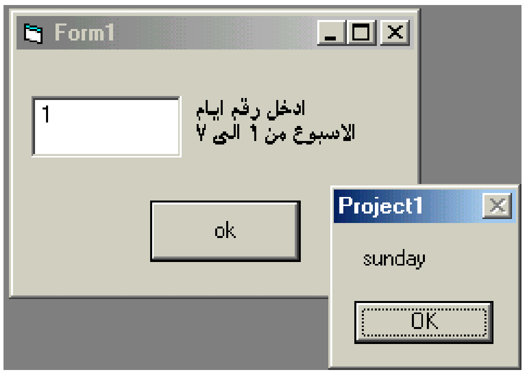
Select statement

Used for applying many statements depending on one variable. The general form is:

```
Select case variable
Case value1
statements
Case value2
Statements
.
.
.
Case value n
Statements
Case else
Statements
End select
```

Example 7: write a program to print the days of the week when we enter its number

Sol:



1

ادخل رقم ايام الاسبوع من ١ الى ٧

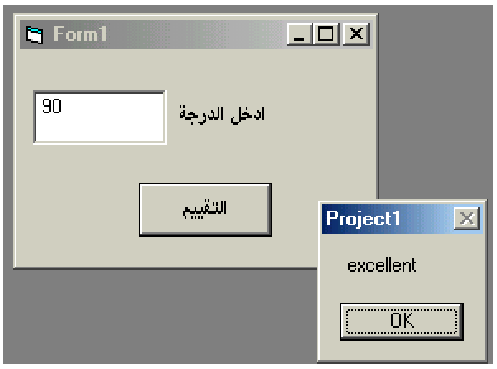
sunday

```
Dim x As Integer
Private Sub Command1_Click()
x = CInt(Text1.Text)
Select Case x
Case 1
MsgBox ("Sunday")
Case 2
MsgBox ("Monday")
Case 3
MsgBox ("Tuesday")
Case 4
MsgBox ("Thursday")
Case 5
MsgBox ("Wednesday")
```

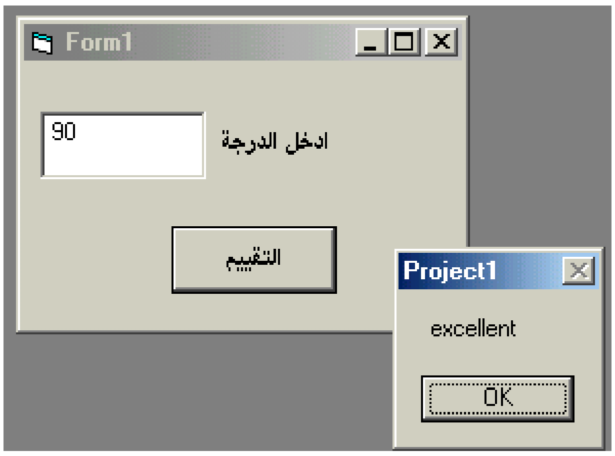
```
Case 6
MsgBox ("Friday")
Case 7
MsgBox ("Saturday")
End Select
End Sub
```

Example 8: write a program to give the evaluation for different marks as follows:

mark	evaluation
90-100	Excellent
80-89	Very good
70-79	Good
60-69	Medium
50-59	Pass
0-49	Fail



```
Dim x As Integer
Private Sub Command1_Click()
x = CInt(Text1.Text)
Select Case x
Case 90 To 100
MsgBox ("excellent")
Case 80 To 89
MsgBox ("very good")
Case 70 To 79
MsgBox ("good")
Case 60 To 69
MsgBox ("medium")
Case 50 To 59
MsgBox ("pass")
Case 0 To 49
MsgBox ("fail")
Case Else
MsgBox "the range is 0-100", vbCritical, "error"
End Select
End Sub
```



Loop statement:

Visual basic supports statement to perform loops. The loops statements could have different structures as follows:

1- Counter loop.
2- Conditional loop.

1- Counter loop:

Loops apply programming statements for fixed number of times using counter (for… next) statement.

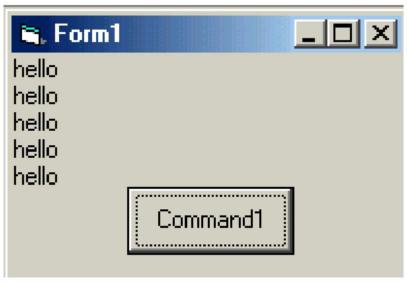
The general form is:

```
For variable = start value  to  end value  step  step value
Statements
Next variable
```

Example1: Write a program to print (hello) five times.

Sol:

```
Dim i as integer
Private Sub Command1_Click ()
For i = 1 To 5
Print "hello"
Next i
End Sub
```

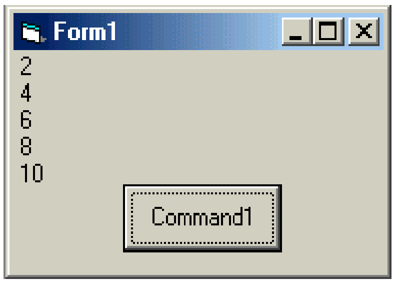


```
hello
hello
hello
hello
hello
```

Example2: Write a program to print even numbers from 1 to 10.

Sol:

```
Dim i as integer
Private Sub Command1_Click ()
For i = 2 To 10 step 2
Print i
Next i
End Sub
```



```
2
4
6
8
10
```

Notes:

1-The variable's value that we use as counter must be integer value (integer, long).
2- If we don’t determined the step value then the assumed value is 1.
3- If the final value smallest than the initial value, then the step value must be negative.

2- Conditional loop

Loops repeat programming statements according to specific condition. There are two types of conditional loop:

1- Do while
2- Do until

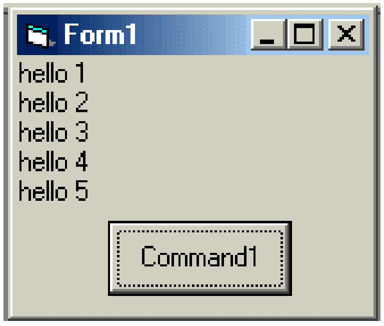
1-**Do while loop:** In this loop the statements will be implemented and repeated when ever the condition satisfied. The general form is:

```
Do while condition
Statements
Loop
```

Example3: Write a program to print (hello) five times with its numbering using do while loop.

Sol:

```
Dim i as integer
Private Sub Command1_Click ()
i = 1
Do while i <= 5
Print "hello"; i
i = i + 1
Loop
End Sub
```

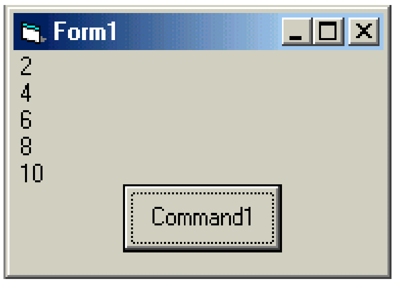


```
hello 1
hello 2
hello 3
hello 4
hello 5
```

Example4: Write a program to print even numbers from 1 to 10.

Sol:

```
Dim i as integer
Private Sub Command1_Click ()
i = 2
Do while i <= 10
Print i
i = i + 2
Loop
End Sub
```



```
2
4
6
8
10
```

2-Do until loop: In this loop the statements will be implemented and repeated when ever the condition not satisfied, (i.e) the loop will be stopped when the condition satisfied. The general form is:

```
Do until condition
Statements
Loop
```

Example5: Write a program to print (hello) five times with its numbering using do until loop.

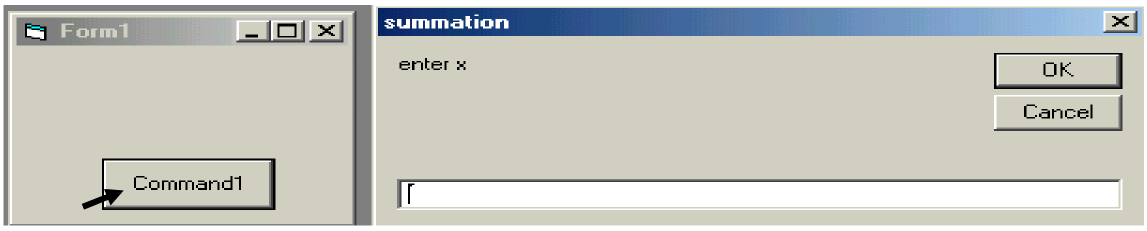
Sol:

```
Dim i as integer
Private Sub Command1_Click ()
i = 1
Do until i > 5
Print "hello"; i
i = i + 1
Loop
End Sub
```

Example6: Write a program to find the summation of undetermined number of positive numbers such that the program will be stopped when we enter negative number.

Sol:

```
Dim x, sum As Single
Private Sub command1_click()
sum = 0
x = Val(InputBox("enter x", "summation"))
Do While x >= 0
sum = sum + x
x = Val(InputBox("enter x", "summation "))
Loop
MsgBox (CStr(sum))
End Sub
```



Example7: Write a program to find the summation of the numbers from 5 to 15.

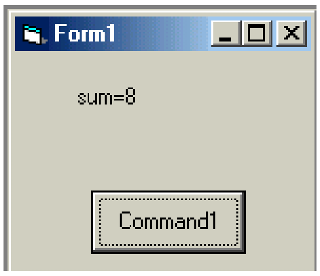
Sol:

```
Dim I, sum as integer
Private Sub command1_click ()
sum = 0
For i = 5 to 15
Sum = sum + i
Next i
Label1.caption = "sum ="&cstr(sum)
End Sub
```

Example8: Write a program to find the summation of 10 numbers.

Sol:

```
Dim i as integer
Dim x, sum as double
Private Sub command1_click ()
sum = 0
For i = 1 to 10
x = val(inputbox ("enter number"))
Sum = sum + x
Next i
Label1.caption = "sum="& cstr(sum)
End Sub
```



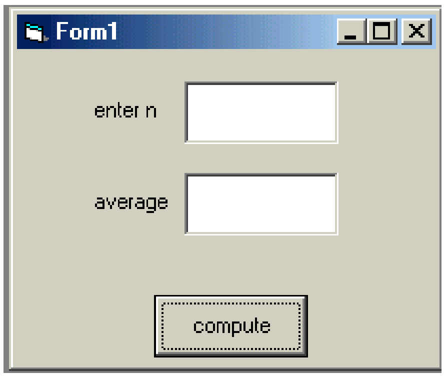
Running stage:

For example if we entered the numbers: 1, 5, -1, 3, 2, 0, -1, 3, 0, -4 then sum=8

Example9: Write a program to find the average of n numbers.

Sol:

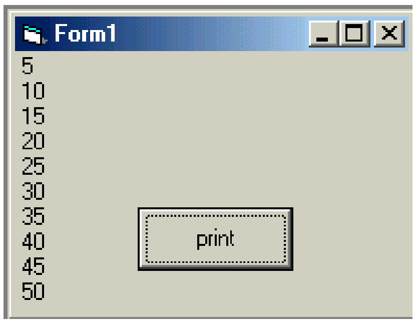
```
Dim i as integer
Dim x, sum, av as Double
Private Sub command1_click ()
i = 1: sum = 0
n = cint (text1.text)
Do while i <= n
x = val(inputbox ("enter number"))
Sum = sum + x
i = i + 1
Loop
Av = sum/n
Text2.text = cstr(av)
End Sub
```



Example10: Write a program to print multipliers of 5 (from 5 to 50)

Sol:

```
 Dim i as integer
 Private Sub Command1_Click ()
 i = 5
 Do until i > 50
 Print i
 i = i + 5
 Loop
End Sub
```



```
5
10
15
20
25
30
35
40
45
50
```

Example11: write a program to find the average of numbers that dividable by 3 (with out remainder) from 3 to 99.

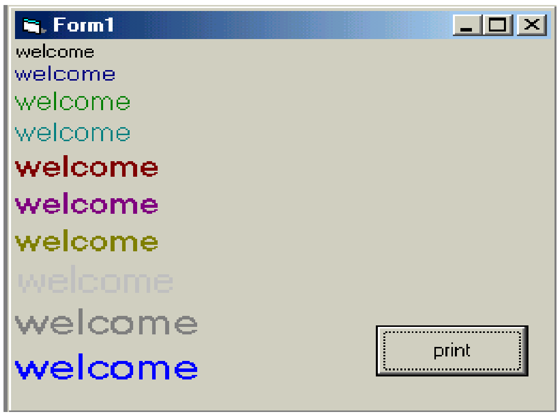
Sol:

```
  Dim I, n, sum as integer
  Dim av as Double
  Private Sub command1_click ()
  i = 3 : n = 0
  sum = 0
  Do while i <= 99
  Sum = sum + i
  i = i + 3
  n = n + 1
  Loop
  Av = sum/n
  Print "av ="; av
  End Sub
```

Example12: write a program to print (welcome) ten times, the first one with the ordinary size and color. Then make the color changed and the size bigger at each time.

Sol:

```
Dim i As Integer
Private Sub Command1_Click()
Print "welcome"
For i = 1 To 9
FontSize = 10 + i
ForeColor = QBColor(i)
Print "welcome"
Next i
End Sub
```



```
welcome
welcome
welcome
welcome
welcome
welcome
welcome
welcome
welcome
welcome
```

Series:

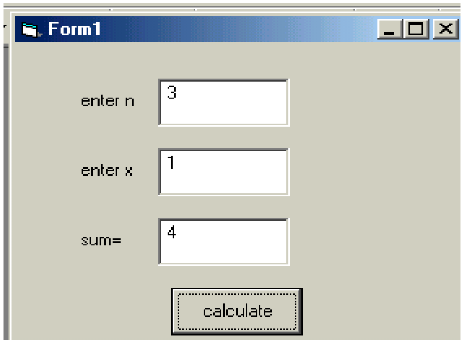
To compute the value of series, we use suitable loop statements according to the boundaries (limits) of each series.

Example13: Find

Sum=1+x+x^2+x^3+…+x^n , where x is an integer.

Sol:

```
Dim I, n, x, sum as integer
Private Sub command1_click ()
sum = 1
n=cint(text1.text)
x=val(text2.text)
For i = 1 To n
Sum = sum +x^i
Next i
Text3.text=cstr(sum)
End Sub
```



enter n 3

enter x 1

sum= 4

Nested loop:

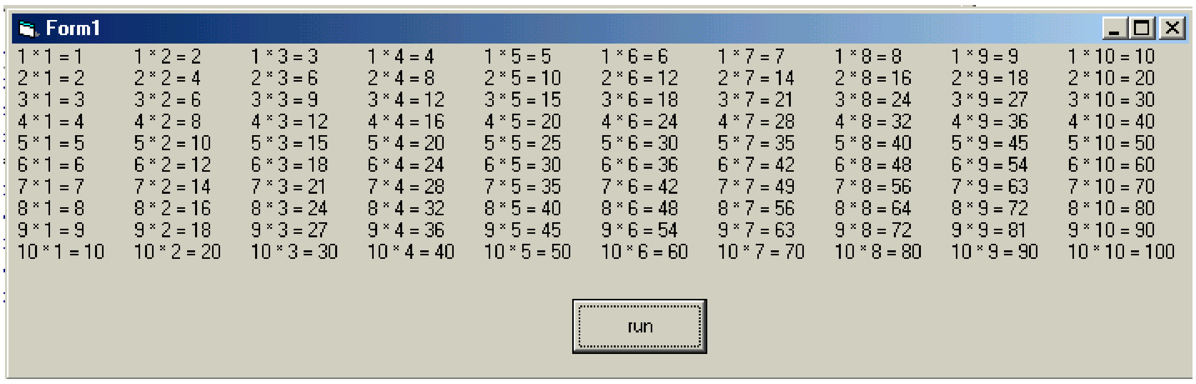
The nested loops are the loops that are placed inside each other. The most inner loop will be executed first, then the outer ones. These loops should neither intersect, nor have the same index. As follows:

```
For i = 1 To n
     For j = 1 To m
          Statements
     Next j
Next i
```

Example14: write a program to print the multiplication table.

Sol:

```
Dim I, j As Integer
Private Sub command1_click()
For I = 1 To 10
For j = 1 To 10
p = I * j
Print I; "*"; j; "="; p,
Next j
Print
Next I
End Sub
```



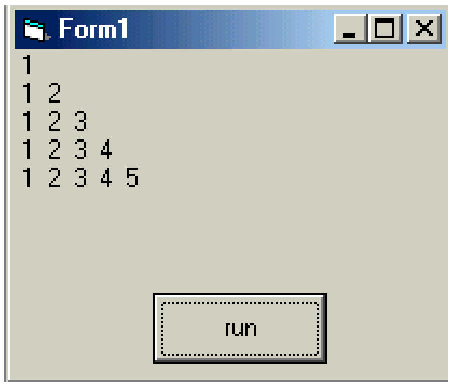
```
1 * 1 = 1    1 * 2 = 2    1 * 3 = 3    1 * 4 = 4    1 * 5 = 5    1 * 6 = 6    1 * 7 = 7    1 * 8 = 8    1 * 9 = 9    1 * 10 = 10
2 * 1 = 2    2 * 2 = 4    2 * 3 = 6    2 * 4 = 8    2 * 5 = 10   2 * 6 = 12   2 * 7 = 14   2 * 8 = 16   2 * 9 = 18   2 * 10 = 20
3 * 1 = 3    3 * 2 = 6    3 * 3 = 9    3 * 4 = 12   3 * 5 = 15   3 * 6 = 18   3 * 7 = 21   3 * 8 = 24   3 * 9 = 27   3 * 10 = 30
4 * 1 = 4    4 * 2 = 8    4 * 3 = 12   4 * 4 = 16   4 * 5 = 20   4 * 6 = 24   4 * 7 = 28   4 * 8 = 32   4 * 9 = 36   4 * 10 = 40
5 * 1 = 5    5 * 2 = 10   5 * 3 = 15   5 * 4 = 20   5 * 5 = 25   5 * 6 = 30   5 * 7 = 35   5 * 8 = 40   5 * 9 = 45   5 * 10 = 50
6 * 1 = 6    6 * 2 = 12   6 * 3 = 18   6 * 4 = 24   6 * 5 = 30   6 * 6 = 36   6 * 7 = 42   6 * 8 = 48   6 * 9 = 54   6 * 10 = 60
7 * 1 = 7    7 * 2 = 14   7 * 3 = 21   7 * 4 = 28   7 * 5 = 35   7 * 6 = 42   7 * 7 = 49   7 * 8 = 56   7 * 9 = 63   7 * 10 = 70
8 * 1 = 8    8 * 2 = 16   8 * 3 = 24   8 * 4 = 32   8 * 5 = 40   8 * 6 = 48   8 * 7 = 56   8 * 8 = 64   8 * 9 = 72   8 * 10 = 80
9 * 1 = 9    9 * 2 = 18   9 * 3 = 27   9 * 4 = 36   9 * 5 = 45   9 * 6 = 54   9 * 7 = 63   9 * 8 = 72   9 * 9 = 81   9 * 10 = 90
10 * 1 = 10  10 * 2 = 20  10 * 3 = 30  10 * 4 = 40  10 * 5 = 50  10 * 6 = 60  10 * 7 = 70  10 * 8 = 80  10 * 9 = 90  10 * 10 = 100
```

Example15: write a program to generate the numbers in the following form.

```
1
1  2
1  2  3
1  2  3  4
1  2  3  4  5
```

Sol:

```
Dim I, j As Integer
Private Sub command1_click()
For I = 1 To 5
For j = 1 To i
Print  j;
Next j
Print
Next I
End Sub
```



```
1
1  2
1  2  3
1  2  3  4
1  2  3  4  5
```