

COURSE OUTLINE

1. INTRODUCTION TO VISUAL BASIC PROGRAMMING LANGUAGES

Description of V.B

Example of V.P.L

Visual basic

Visual c++

Delphi

Hardware and software constructions for V.P

2. VISUAL ENVIRONMENT

Description of visual environment

Description of intergrated development environment (IDE)

Visual objects

Form window

Properties window

Immediate window

Code window


Others

3. PROGRAM STRUCTURE


Format of Visual Program

Data types

Operators

 Arithmetic

 Logical

 Comparison

 Others

Variables

4. PROGRAM WRITING

Creating an application

Compilation

Debugging

Testing

Execution

5. CONTROL STRUCTURES

Types of control structures

Definition

Types:

Sequence

Selection

Iteration / repetition

Implementation of data structures

6. ERROR HANDLING

Types of errors

- Syntax

- Run time
- Semantics
- Logical

Error handling techniques .

Writing error handlers

On error resume

On error to go

Error object

Debugging tools

7. SUB-PROGRAM

Meaning of sub-program

Types of sub-programs

Private sub program

Public sub-program

Scope of variables

Local variables

Global variables

8. Data structures

Data structures

Types of data structures

- Arrays
- Records
- Pointer
- Stacks and queues
- Sets
- Files

Sort Techniques

- Bubble shell
- Selection

Search Techniques

- Binary search
- Linear search

9. Linking to databases

- Database controls
- Data control
- Ms data bound controls
- Active data object (ADO)

Reports

Data reports

10. Emerging Trends In Visual Basic.

Emerging trends in visual programming

Challenges of emerging trends in Visual Basic

Coping with challenges of emerging trends in visual programming.

Content

1. Introduction to visual Basic 6.0

Visual Basic

It is a programming language and development environment that allows you to create programs that run under windows Os. With VB you can create programs that interact with the Databases, interact with internet and even with hardware.

Event Driven Programming

V.B is event driven ,meaning the code remains idle until called upon to respond to some event. An event can be a mouse click, a key press , menu selection e.t.c . V.B is governed by an event processor. Nothing happens until an event is detected .Once an event is detected the code corresponding to that event (event procedure) is executed.

Terminologies

a)Forms

Are windows that you create for user interface .A form generally has many different controls placed upon it.

b) Controls

Are graphical features drawn on forms to allow user interaction, examples are textboxes labels, scrollbars command buttons e.t.c. Forms and controls are sometimes collectively known as objects.

c) Properties

Every characteristic of a form or control is specified by a property, examples of properties include name, caption ,size, colour ,position and content.

d) Methods

Is a built-in procedure that can be invoked to impart some action to a particular object e.g **print** method. Example: Me.print sum ‘ displays value held by the variable sum in the current form.

e)Event Procedure

Is a code related to some object. This is the code that is executed when a certain event occurs.

f)General Procedure

Is code not related to objects. This code must be invoked by the application.

g) Module

It is a collection of general procedures, variable declaration and constant definitions used by the application.

h)Application

Is a collection of objects that work together to accomplish something useful. In Vb the application is called a project. A project could be the management of a video store , calculation of mortgages e.t.c

i)Object

Is a piece of s/w that has properties and functions that can be manipulated.

2. VISUAL OBJECTS

Visual Basic Integrated Development Environment (I.D.E)

I.D.E is a VB screen in which one can develop, run, test and debug applications. It contains the elements with which you can develop the user interface of your application.

Elements/components/parts of IDE (Visual Objects)

1. Menu bar

It displays the commands you need to work with VB. Menus are provided for standard actions like : file , edit , view , window and help and to access functions specific to programming such as: project, format and debug

2.Standard Toolbar

The standard toolbar has the standard icons that are shortcut command to the menu bar commands. They provide quick access to commonly used commands in the programming environment.

3.Toolbox

It contains a set of tools that can be used at design time to place controls on a form

4.Project Window

This window lists the forms and modules in your current project.

5. Properties window

This window lists the property settings for the selected form or control. The properties are characteristics of an object such as name, size, colour, caption e.t.c

6. Form layouts window

It shows where (upon program execution) your form will be displayed relative to your monitor's screen.

7. Code editor window

The code editor window is used for entering and editing the programming code for the application. Each form has its own code editor window .By double clicking on an object the corresponding code editor window will open.

8. Immediate , local and watch windows

This windows are included in the IDE for use in debugging your application. They are available only when you are running your application.

Controls

Are graphical features drawn on forms to allow user interaction .it is a tool used to display information or retrieve information from the user.

Control types

1. Standard controls

Are controls always present in the toolbox

2. Custom controls

Are brought into the VB environment when needed and they extend one tool box.

Description of the roles of various controls

Pointer

Is a tool used to select an object to be worked on

Picture box

Is used to display graphical objects or text and to initiate event actions. It is similar to an image box but has more properties , redraw slower and cannot be stretched.

Label

Displays a text on a form. The text cannot be reassigned during program execution though its appearance can be altered during the program execution though its appearance can be altered

Text box

Provides a means of entering or displaying text. The text may be already available or can be entered during the program execution.

Frame

Used as a container of a group of controls

Command button

Used to activate an event driven procedure.

Check box

Used if more than one selection are to be made.

Option button

Are used when one selection must be made before an action is carried out.

Combo box

Is a special type of list box that combines the capabilities of a text box and a list box. It provides a list of text item for selection by the user during program execution. Items can also be added during program execution.

List box

Provides a list of text items for the selection by the user.

Horizontal scroll

Is used for drawing horizontal scrollbars on a form.

Vertical scrollbar

Is used for drawing vertical scrollbar on a form.

Timer

Allows timed events to occur repeatedly at a specific time interval

Drive list box

Provides a way of selecting a drive from a list of existing drivers

Directory list box

Provides a way of selecting a directory from a list

File list box

Provides a way of selecting files within a group of files .

Shape tool

Is used to draw ; circles, eclipses , squares and rectangles within forms

Line tool

Is used when drawing straight lines within forms.

Image box

Is used to display graphical objects

Data control

It provides a means of displaying information from existing database.

N/B :The number of tools available on the tool box depends on edition and release of VB you are using.

Saving a VB project

Saving a VB project is different from other applications because VB operation involves saving multiple files

- a) To save a VB project for the first time, select the **save project as** command from the file menu. The **save file as** dialogue box appear. You will have to enter a form name and then click the same button ,the form is saved with an extension .frm
- b) After this the **save project As** dialogue box appears. Type the name of the project and it will be saved as file with extension .vbp

Opening an existing project

You can open a project when VB starts or from the file menu. To open a project during start up from the new project dialogue box that appears when VB is starting, click the existing tab. A list of existing projects will be displayed. If VB is already running close all other projects then click **open project** command on the file menu. The open project dialogue box appears from which you can select the existing tab

3. PROGRAM STRUCTURE

VB FUNDAMENTAL CONCEPTS

Data types

Data types	Description
Integer	A whole number with no fraction part. Integer range from -32768 to 32767
Long integer	Numbers which are integers but have bigger values and range. They range from -2,147,483,648 to 2,147,483,647
Single	A single precision real constant includes a fractional part. The largest value is 3.4×10^{38}
Double	It includes a fractional part but has far much larger magnitude than the single real number constant. However it cannot be larger than 1.2×10^{308}
String	Characters enclosed in quotation marks
Boolean	Data types that have only two logical states i.e true or false

Constants

A constant is a value that remains the same. It does not change during program execution. There are three types of constants; string, numeric and named constants.

String constants

Is a sequence of characters enclosed in quotation marks.They are used to write non-numeric values like telephone numbers, addresses and names

Numeric constants

Are either whole numbers (integer), double or single

Named constants

Is identified by name rather than its actual values, const keyword is used declare this type of constants

VARIABLES

A variable is a memory location referred to by name used to hold a value that is subject to change during program execution. In VB the following rules should be followed when naming variables:

1. The variable name must not have more than 255 characters
2. The variable name must start with a letter.
3. The letter case is not important when declaring variables
4. A variable name must not be a reserved word
5. Variable names should not have spaces.

A variable is declared using the reserved word DIM the short form of the word dimension (size). It is used to associate a variable with a specific data type e.g Dim First_Name as String means that the variable first_name is of the type string.

VARIANTS

Is a variable whose data types has not been explicitly declared by the program. They are called variants because their data types keeps on changing with the values they hold

Scope

Refer to the level of the program that a variable, a constant or a procedure is recognized. The scope of a variable constant is said to be global or local. The scope of the procedure, variable or constant is set by declaring it as either private or public. The following terms are used in this regard:

1. Private sub-procedure
2. Public
3. Global variables / constants
4. Local variables

A public procedure can be accessed from any module or form in the program while private procedure is accessed in the module or form within which it is declared.

A variable or constant that is declared outside the standard module but within the same project is said to be global.

User defined data types.

Is a data type whose individual components are standard data items

Using suffixes to declare variables and constants.

Suffixes are special symbols appended to the end of a variable constant name in order to associate them with a particular data type

Examples

Suffix	Data type	Long Declaration	Short declaration
%	Integer	Dim A as integer	A%
!	Single	Dim X as single	X!
\$	String	Dim A as string	Q\$
&	Long integer	Dim C as long integer	C&

#	Double	Dim P as double	P#
---	--------	-----------------	----

Operators

They include;

- i. Arithmetic
- ii. Relational
- iii. Logical

Arithmetic operators

Are special symbols that are used to write arithmetic expressions.

Symbol	Name	Operation
+	Plus sign	Addition
-	Minus sign	Subtraction
*	Asterisk	Multiplication
/	Slash	Division
^	Caret	Exponentiation
\	Backslash	Integer division
MOD	Modulus	Integer remainder

N/B: The last two rows can be explained as follows

1. In an integer division, each of the numbers being divided is first rounded to become an integer then the quotient is truncated to an integer.
2. The MOD operator returns the remainder of an integer division.

Operators Precedence / Hierachy

Arithmetic operators have the following order of precedence when used in a program
Exponentiation, Multiplication and Division , integer division, integer remainder, addition and subtraction.

i. Relational Operators

Is used in an expression that returns a true or false value when evaluated. The operators can compare numeric variables, constants or expressions.

Operator	Name
=	Equal to
<>	Not equal to
>	Greater than
<	Less than
<=	Less than or equal to
>=	Greater than or equal to

Logical operators

operator	Operation
----------	-----------

And	Results in a condition that is true if both expressions are true
Or	Results in a condition that is true if one of the condition is true or both are true
Xor	Results in a condition that is true if one of the condition is true and the other is false
Not	Negates the value of a logic expression

The Assignment Statement

An assignment statement is an executable statement that assigns whatever is on the right of the assignment operator to the variable.

The Print Statement

The print statement displays data on the active form of your VB project. It can be used to display processing results of a program. The structure of the print statement is :

Print variable name (this prints out the value held in a variable). The use of comma to separate the various elements to be output by the print statement makes them to be widely spaced on the screen to have a compact display with the various outputs close to each other, then use a semi colon to separate the various elements.

Library functions/In-built functions

Are modules that have been pre-written and included in the visual Basic Language. A function performs single task like calculating Mathematical expressions then returns a value. A function will usually have a name and can be called when needed to perform a particular task.

A library function is accessed by stating its name followed by the information that must be supplied to the function enclosed in parenthesis. This process is called calling a function. The values held in the parenthesis are called arguments. When a function is called, it uses the supplied arguments to perform an operation and return a value.

Function	Call	Description
Date	Z= Date ()	Return the current date
Exp	Z = Exp (W)	Returns the value of e^w
Sqr	Z= Sqr (w)	Returns the square root of w
Log	Z = Log (w)	Returns the natural logarithm of w
Str	Z = Str (w)	Returns a string whose characters are w e.g str (6.80)="6.80"
Cos	Z =Cos (w)	Returns the cosine of w
Abs	Z = Abs (w)	Return absolute value of w e.g Abs (.8)
Ucase	Z = Ucase (w)	Returns uppercase equivalent of w
Int	Z = Int (w)	Returns the largest integer that is algebraically less than w e.g Int (2.2) = 2
Time	Z = Time	Return the current system time
Format	Z = Format	Formats the output to appear as statements

Using the format function

The format functions helps the programmer to display data in many different formats.

Expression

Print format (16.778994, “##.##”)	16.78 (notice the rounding)
Print format (Now, “mm-dd-yy”)	1-20-20105
Print Format (15678; “##,###.00”)	15,678.00
Label 1. Caption = Format\$(sprice, “#####.##”)	10,630.75

N/B: The dollar sign may be used with the format function depending on the type of output required. In the last row of the table the formatted output is displayed on a label control.

Program comments

Comments are English statements included in the code to document a program. The comments are not executed when the program is running. In Visual Basic, a comment starts with a single apostrophe (‘) followed by the comment statement e.g X = Sin (P) ‘ find the Sine of P and store it in X

Converting numeric string to a value

If you enter a number, in a text box, the best way to convert it into a value is to use the Val function. This is because the computer treats anything typed in the text box as a string. The val function operates on the string and returns a numerical value.

Converting a value to a string

Assuming a particular variable has a numerical value and you wish to display the value in a text box, then you have to convert the value to a string first. Suppose the name of the text box is txt Area and the integer variable is X then ; txt Area .text = str(X) display the value in X as string in the txtArea.

4. PROGRAM WRITING

Project

Create a program that can be used to calculate the area of a rectangle. The program should prompt the user to enter the length and the width of the rectangle .To create the project proceed as follows;

1. Open visual basic and select standard .EXE from the new project dialog box. A blank form will be displayed.
2. Place three text boxes and two command buttons. Double click the text box tool on the tool box then command button. Add another command button .Use the shape tool to place a rectangle on the form. Drag the controls to the desired location on the form and repeat the process until you have the interface below.
3. The next step is to define the set of properties for each control. To do this, right click on the control then click properties. Set each control in the properties window.

N/B: The conventional way of setting the name property of any object in VB is to use object prefixes. Such as txt for text or boxes, lbl for labels cmd for command buttons etc.

4. After setting the properties, we can now write the event procedures that calculates the area of the rectangle once values are entered in the txtlength and txtWidth textboxes. To write the code double click calculate button and write the code listing below.

```
Private sub cmdcalculate_click()
```

```

Dim L as single, W as single, A as single
L = val(txtlength.text)
W = val(txtWidth.text)
A = L * W
TxtArea= Str (A)
End sub
Private sub cmdExit_Click()
‘ exit from the program
End
End sub.

```

5. Run the project. Enter a value in the length and another in the txtWidth text boxes then click the calculate button. The area of the rectangle is displayed.

Naming Standards For Controls

A very important property for each object is its name, name is used by VB to refer to a particular object in code. A convention has been established for naming VB objects. This convention is to use a three letter prefix (depending on the object) followed by a name you assign.

Object	Prefix	Example
Form	Frm	Frmemployee
Label	Lbl	Lblstart, lblend
Command button	Cmd,btn	Cmdsave,cmdexit
Text box	Txt	Txtnum,txtName
Menu	Mnu	Mnuexit,mnusave
Check box	Clck	Chkchoice
Combo box	Cbo	Cbostates
Data control	Dat	Datstudent
Directory list box	Dir	Dirsources
Drive list box	Drv	Drvuserdrives
File list box	File	Filesources
Frame	Fra	Fracards
Grid	Grd	Grdpartlist
Horizontal scrollbar	Hsb	Hsbpartlist
Vertical scrollbar	Vsb	Vsbpartlist
Image box	Img	Imgicon
Line	Lin	Linseparate
List box	Lst	Lstcountries
Option button	Opt	Optsex
Picture box	Pic	Picresult
Shape	Shp	Shprectangle
Time	Tmr	tmrAlarm

Declaration of variables

There are two ways of declaring variables

i. Explicit declaration

The variable is declared in the declaration section of the program or at the beginning of a procedure e.g

Dim A As integer 'explicit declaration'

Dim name as string ' explicit declaration.

ii. Implicit declaration

The variable is declared on the fly. Its data type is deduced from the declaration of other variables e.g

Dim A as integer ' explicit

Dim B As integer 'explicit

C = A + B 'Implicit declaration.

Project Examples

Example

Calculation of area of a rectangle, length and Width entered by the user.

a).User Interface/UI

Frmarea	
Calculation of area	
Enter length	Txtlen
<input type="text"/>	<input type="text"/>
Enter width	Txtwid
<input type="text"/>	<input type="text"/>
Area is	Txt A
<input type="text"/>	<input type="text"/>
Exit	calculate
<input type="text"/>	<input type="text"/>
cmdexit	cmdcalculate

b).Code

```
Private sub cmdexit_click()
```

```
End
```

```
End sub
```

```

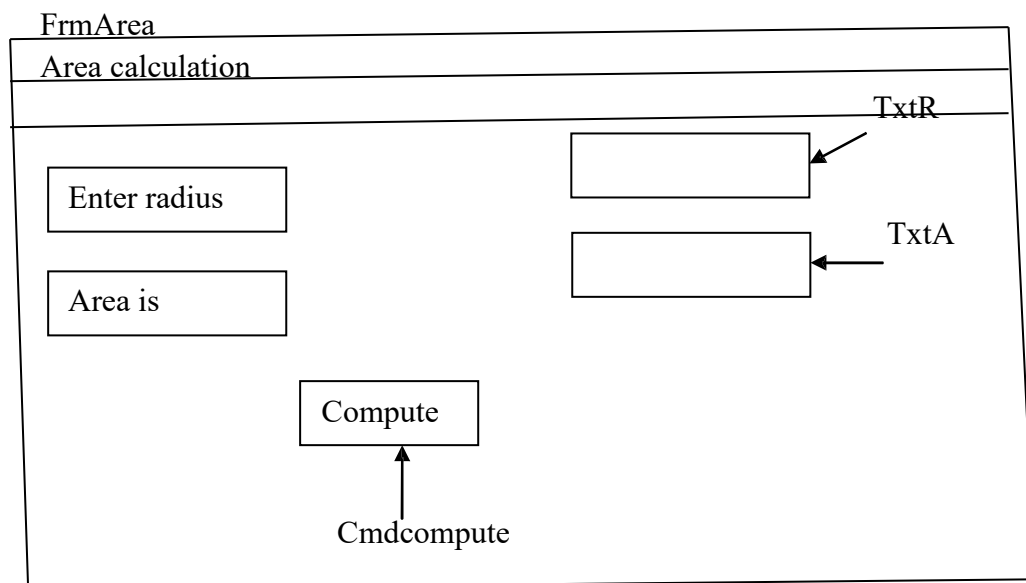
Private sub cmdcalculate_click()
Dim L as integer
Dim W as integer
Dim A as integer
L =val(txtlen.text)
W = val(txtwid.text)
A = L * W
TxtA.text = str (A)
End sub

```

Example

A Vb program to accept the radius of a circle, calculate and display its area

a).UI



b).Code

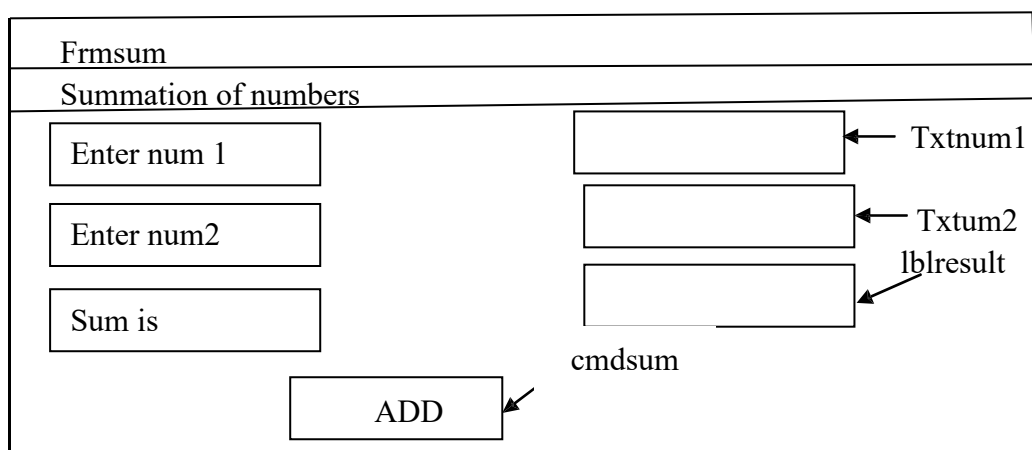
```

Private sub cmdcompute_click()
Dim R as single, Area as single
Const pie as single = 3.142
R = val (txtR.text)
Area = pie * R * R
TxtA.text = Str(Area)
End sub

```

Example

Project on calculation of sum of two numbers



Code

```
Private sub cmdsum_click()  
Dim a as integer, b as integer  
Dim s as integer  
a= val(txtnum1.text)  
b=val(txtnum2.text)  
s = a + b  
lblresult.caption=s  
End Sub
```

Exercise

Write a VB that accept the height and radius of a closed cylinder, compute and display its surface area. Attach your code to a command click event.

```
Private sub cmdsurfacearea_click()  
Const pie as single = 3.142  
Dim height as integer, radius as integer  
Dim surfaceArea as single  
height = val(txth.text)  
radius = val (txtr.text)  
SurfaceArea = 2 * pie * radius * radius + (pie *2*radius * height)  
Lblarea.caption = surfaceArea  
End sub
```

Frmsarea	
Surface area	
Enter radius	<input type="text"/> Txtr
Enter height	<input type="text"/> Txth
Surface area is	<input type="text"/> Lblarea
<input type="button" value="CALCULATE"/> Cmdsurface	

CONTROL STRUCTURES

a. Selection control structures

i. If.....else construct

Used to write programs that require branching depending on a test on some condition

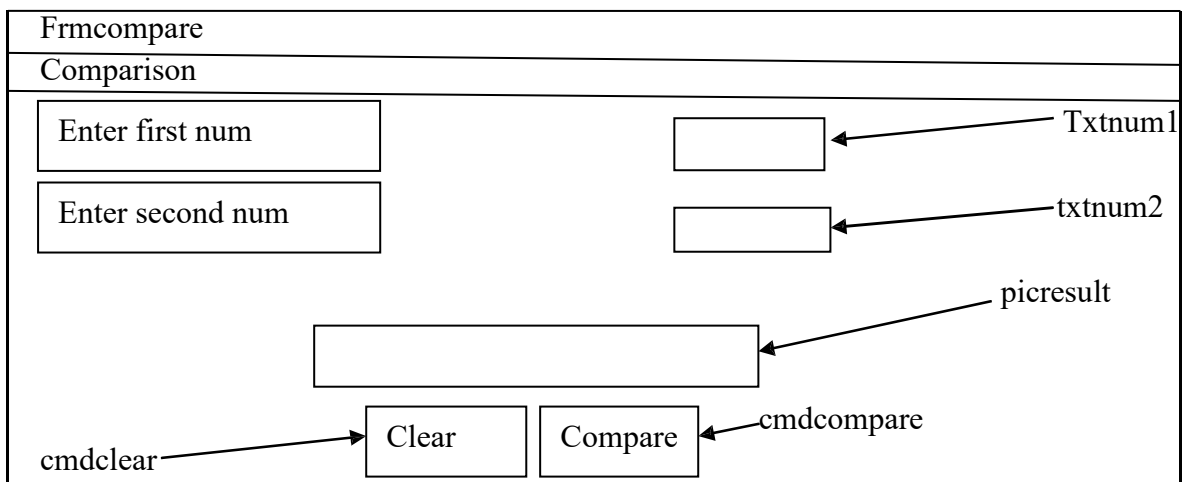
-Syntax is:

```
If (condition) then
    Statement(s)
Else
    Statement(s)
End IF
```

Example

A program 2 accept two numbers and determine the greatest

- .User interface
-



```
private sub cmdcompare-click()
dim num1 as integer
dim num2 as integer
num1 = val (txtnum1.text)
num2 = val (txtnum2.text)
if (num1>num2)THEN
picresult.print "num1 is the greatest"
else
picResult.print "num2 is the greatest"
End if
end sub
private sub cmdclear_click()
txtnum1.text =
" " txtnum2.text="""
End sub.
```

If with Else If

Used in situations where we have more than one condition to test in the program.

-Its Syntax is:

```
If (condition1) Then
```



```

Statement1
Else if (condition2) Then
    Statement2
Else if (condition3)
    Statement3
    :
    .
Else
    StatementN
End if

```

Example

A program to accept student position in class and display category as per the table below

Position	Category
1-3	Top 3 student
4-5	Top 5 student
6-10	Top 10 student

a) User interface

```

private sub cmdcategorize_click()
dim position as integer
position = val(txtpos.text)
if (position >=1) and (position <=3)Then
picresult.print "top 3 student"
Else if (position >= 4) and (position <= 5)Then
PicResult.print" top 5 student"
Else if (position >= 6) and ( position<=10) Then
Picresult.print "top 10 student"
Else
Picresult.print "You are not categorized"
End if
End sub

```

Exercise

Write a VB program to accept student marks in class and display grade as per the table below.

Marks	Grade
70-100	A
60-69	B
50-59	C
40-49	D
<40	F

a).User interface

Frmgrade	
Grade marks	
Enter marks	<input type="text"/> ← txtmarks
<input type="text"/> ← picresult	
<input type="button" value="GRADE"/> ← cmdgrade	

```
private sub cmdgrade_Click()  
Dim marks as integer  
Marks = val(txtmarks.text)  
If (marks >=70) and (marks <=100) Then  
Picresult.print "A"  
Else if (marks >=60) and (marks<= 69) Then  
Picresult.print"B"  
Else if (marks >=50) and (marks<=59) Then  
Picresult.print "C"  
Else if (marks >= 40 ) and (marks <= 49) Then  
Picresult.print "D"  
Else if (marks<40) Then  
Picresult.print "F"  
Else  
Picresult.print "Invalid entry"  
End if  
End sub
```

USING OPTION AND CHECKBOX WITH IF CONSTRUCT SELECTION CONTROL STRUCTURE

OPTION BUTTON

-Control used to make a single selection out of several options available.

VALUE PROPERTY

The value property is used to indicate whether the option button is selected or not.

A value property of TRUE means the option button is selected and a value property of FALSE means no selection made.

PROGRAM EXAMPLE:

Program to prompt user to enter principal amount, calculate and display interest based on the gender selected. The criteria for interest rates are given in the table below.

GENDER	INTEREST RATE
Male	Award 20% of Principal amount
Female	Award 15% of principal amount

a).User Interface and properties

The screenshot shows a Visual Basic form titled 'Form1' with the following components:

- ENTER PRINCIPAL**: A label above a text box labeled **txtA**.
- SELECT GENDER**: A label above a group box containing two radio buttons.
 - MALE**: A radio button labeled **optmale**.
 - FEMALE**: A radio button labeled **optfemale**.
- PicResult**: A large rectangular area for displaying the result.
- cmdCompute**: A button labeled **COMPUTE**.

b).Code

```
Private Sub cmdcompute_Click()  
Dim Amount As Double, interest As Double  
Amount = Val(txta.Text)  
Picresult.Cls  
If (Optmale.Value = True) And (Optfemale.Value = False) Then  
interest = 20 / 100 * Amount  
Picresult.Print "Interest = ", interest  
ElseIf (Optfemale.Value = True) And (Optmale.Value = False) Then  
interest = 15 / 100 * Amount  
Picresult.Print "Interest = ", interest
```

```

Else
MsgBox "Please select Gender"
End If
End Sub

```

CHECKBOX

-Control used to make more than one selection out of several options available.

VALUE PROPERTY

The value property is used to indicate whether the checkbox is selected or not.

A value property of 1 means the checkbox is selected and a value property of 0 means no selection made.

PROGRAM EXAMPLE:

Program to prompt user to enter salary earned, calculate and display tax based on the status selected. The criteria for tax rates are given in the table below.

STATUS	TAX RATE
Married only	20% of salary
Studying only	15% of salary
Married and Studying	10% of salary

a).User Interface and properties

The screenshot shows a Visual Basic form titled 'Form1'. It contains the following elements:

- A label 'ENTER SALARY' above a text box labeled 'txts'.
- A label 'SELECT STATUS' above a group box containing two checkboxes: 'MARRIED' (labeled 'chkM') and 'STUDYING' (labeled 'chkS').
- A label 'PicResult' above a text box.
- A 'COMPUTE' button labeled 'cmdCompute'.

b).Code

```

Private Sub CMDCOMPUTE_Click()
Dim SALARY As Double, tax As Double
SALARY = Val(TXTS.Text)
PicResult.Cls
If (Chkm.Value = 1) And (chkS.Value = 0) Then

```

```

tax = 20 / 100 * SALARY
PicResult.Print "Tax = ", tax
ElseIf (chkS.Value = 1) And (Chkm.Value = 0) Then
tax = 15 / 100 * SALARY
PicResult.Print "Tax = ", tax
ElseIf (Chkm.Value = 1) And (chkS.Value = 1) Then
tax = 10 / 100 * SALARY
PicResult.Print "Tax = ", tax
Else
MsgBox "Please select status"
End If
End Sub

```

ii. CASE CONTRUCT

Programs are more simple to read and understand when a case construct instead of if construct is used to write programs which involve decisions.

Its Syntax is:

```

Select Case expression
Case expression1
    Statement1
Case expression2
    Statement2
    ⋮
Case else
    StatementN
End select

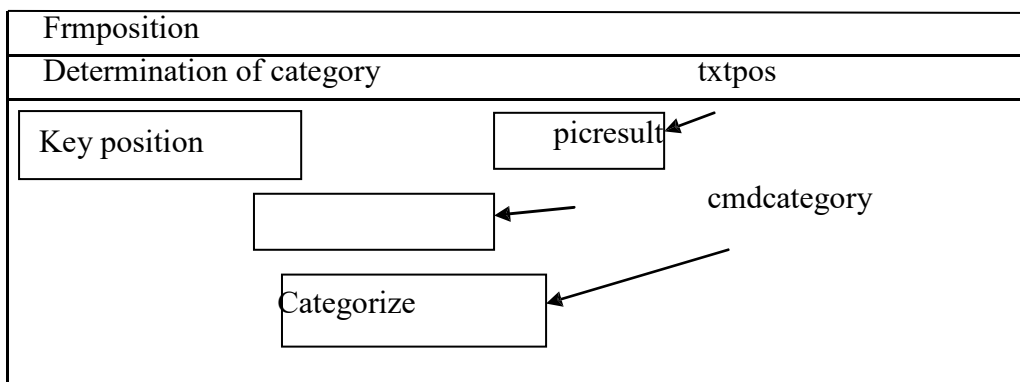
```

Example

A program to accept the position of a student in a class and display his category as per the table below.

Position	Category
1-3	Top 3 students
4-5	Top 5 students
6-10	Top 10 students

a).User interface



b).Code

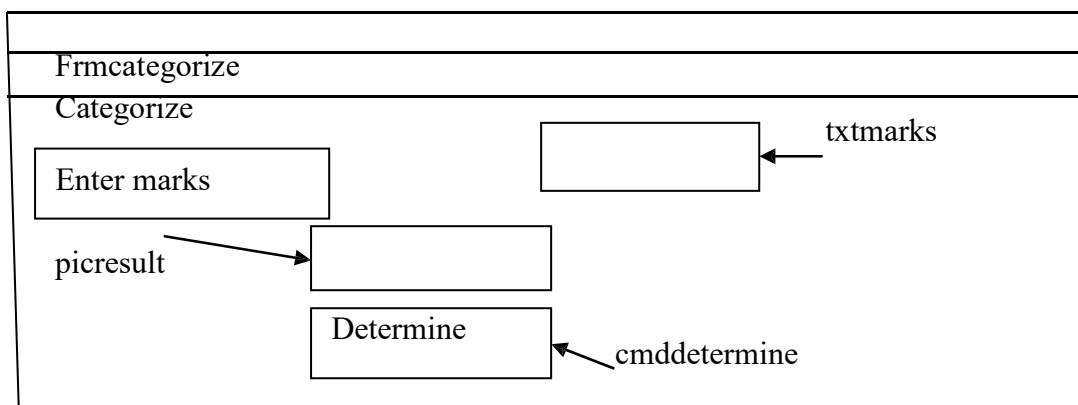
```
Private sub cmdcategory – Click ()  
dim position as integer  
position = val (txtpos.text)  
SELECT CASE position  
Case 1 to 3  
Picresult.print “top 3 student”  
Case 4 to 5  
Picresult.print “top 5 student”  
Case 6 to 10  
Picresult.print “top 10 student”  
Case else  
Picresult.print ”You entered valid position”  
End select  
End sub
```

Exercise

1. Rewrite a program to grade marks using the case construct
2. Write a vb program that accepts the initials of the color and display its name in full as per table.

Initials	Colour name in full
R	Red
B	Blue
G	Green

a).User interface



b).Code

```
private sub cmddetermine_click()  
dim marks as integer  
marks =val(txtmark.text)  
SELECT CASE marks  
Case 70 to 100  
Picresult.print ”A”
```

```

Case 60 to 69
Picresult.print "B"
Case 50 to 59
Picresult.print "C"
Case 40 to 49
Picresult.print "D"
Case 0 to 39
Picresult.print "F"
Case else
Picresult.print "You entered invalid marks"
End select
End sub

```

b).Repetition / iteration control structure

i. The For...Next loop

Used to execute one or more statements for predetermined number of times.

-Syntax is:

For control_variable = initial_value to final_value

Statement(s)

Next control_variable ' increment value of control_variable

-You can also make the for loop to skip some iterations e.g

a) For I = 1 to 10 step 2

Picresult.print i

Next i

b) For i= 10 to 1 step -2

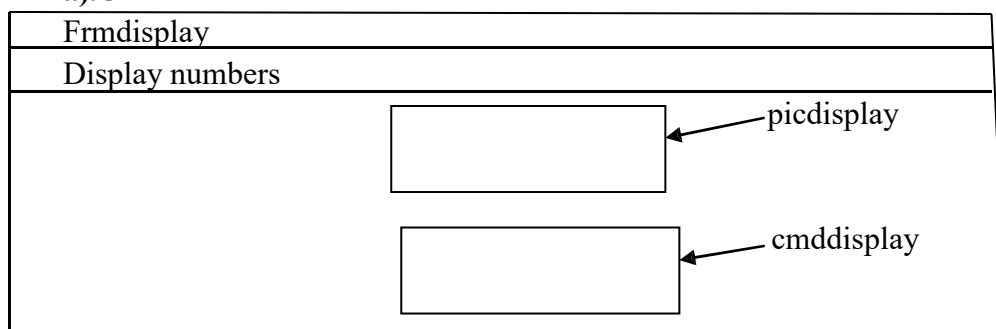
Picresult.print i

Next i

Example

A program to display numbers 1 to 5

a).UI



b).Code

```

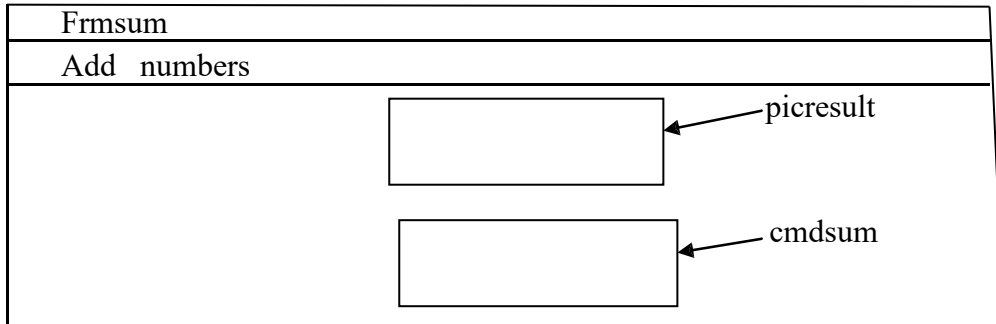
Private sub cmddisplay_click()
Dim I as integer
For I = 1 to 5
Picdisplay.print i

```

Next i
End sub

Example 2

A program to find sum of three numbers entered by the user.



b).Code

```
Private sub cmdsum_click()
Dim I as integer
Dim sum as integer
Dim num as integer
Sum = 0
For I = 1 to 3
num = inputbox (" Key in number", " addition")
sum = sum + num
Next I
Picresult print "sum of numbers =", sum
End sub
```

ii) do and while loops

Used to execute one or more statemets based on a test on some condition

a) Do loops

- | | |
|-------------------------|-------------------------|
| 1. Do until (condition) | 2. Do while (condition) |
| Statements(s) | Statements(s) |
| Loop | Loop |
| 3. Do | 4. Do |
| Statements (s) | Statements(s) |
| Until (condition) | While (condition) |

-With loops 1 and 2 the condition is tested before the statements inside the loop are executed

-With loops 3 and 4 the statements are executed at least once before the condition is tested.

b).While loop

-It syntax is:

While (condition)

Statement(s)

Wend

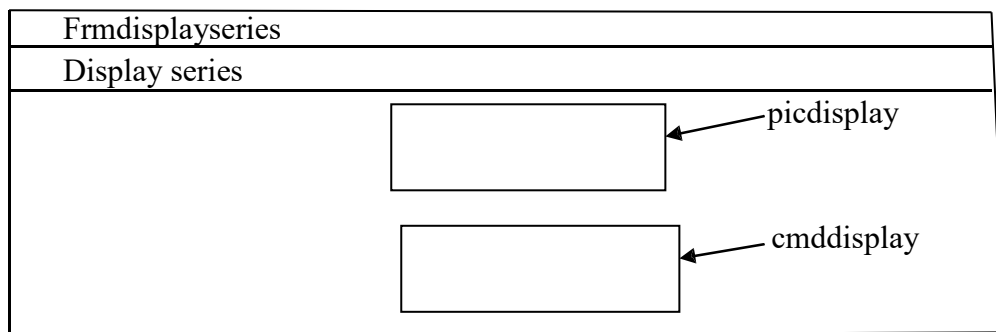
-Exactly the same as the Do while..Loop, and as with the Do while..loop, the While..Wend loop executes statements inside it as long as the condition set is true.

Example

Program using while..wend loop that displays the number series:

3,7,11,15

a).UI



b).Code

```
Private sub cmddisplay_click()  
Dim num as integer  
num=3  
While (num<=15)  
Picdisplay.print num  
num=num+4  
Wend  
End sub
```

SUBPROGRAMS IN VISUAL LANGUAGES

Structured program design requires that problems be broken into small problems to be solved one at a time. VB has a device, sub procedures and function procedures that are used to break problems into manageable chunks, sub procedures eliminate repetitive code, can be reused in other programs and allow a team of programmers to work on a single program

Sub procedures

Is a part of a program that performs one or more related tasks, has its own name and it is written as a separate part of the program.

-Its syntax is:

```
Private sub procedurename()
```

```
Statement(s)
```

```
End sub
```

A sub procedure is invoked with a statement of the form:

```
call ProcedureName()
```

Example:

A program tha accepts and calculates the sum of two numbers using a sub procedure.

a).User interface

frmProcedureadd	
Addition procedure	
Enter num1	<input type="text"/> ← Txtn 1
Enter num2	<input type="text"/> ← Txtn 2
Sum is	<input type="text"/> ← Txts
Add	<input type="button" value="Add"/> ← cmdadd

b).Code

```
Private sub cmdadd- click
```

```
Dim x as single, y as single
```

```
X = val(txtn1.text)
```

```
Y = val(txtn2. Text)
```

```
Call add(x,y)
```

```
End sub
```

```
Private sub add (n1 as single, n2 as single)
```

```
Dim S as single
```

```
S = n1 + n2
```

```
Txts.text = str(S)
```

```
End sub
```

-Sub procedures make a program easy to read, modify and debug. Sub procedures can also be called several times during the execution of the program.

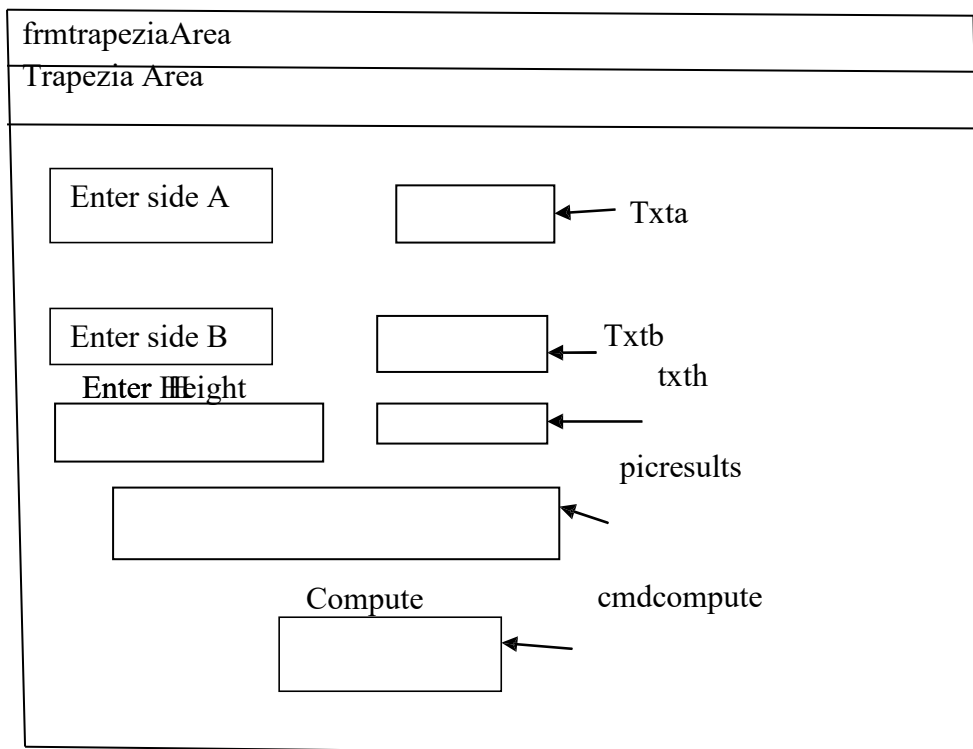
Passing (transmitting values to a sub procedure)

The variables n1 and n2 appearing in the sub procedure are called parameters. They are place holders for values passed to the procedure.

Exercise

Write a Vb program using two procedures. The first procedure displays a message showing what the program does. The second accepts the dimensions of a trapezium, calculates and displays its area. Both procedures display information in the same picture box.

a).UI



b).Code

```
private sub cmdcompute()  
Dim area as single  
a = val (txta. Text)  
b = val (txtb. Text)  
h = val (txth.text)  
call procdisplay()  
call CalcArea(a,b,h)  
End sub
```

```
private sub procdisplay()
picresults.Print "This program calculates area of Trapezium"
End sub
```

```
private sub CalcArea(a single, b as single, h as single)
dim area as single
area = 0.5 * (a + b) * h
picresults.Print "area is ", area
End sub
```

PARAMETER PASSING

PASSING A VARIABLE BY REFERENCE TO A SUB PROCEDURE Suppose a variable call it **arg** appears as an argument in a call statement and its corresponding parameter in the sub statement is **par**. After the sub procedure is executed, arg will have whatever value par had in the sub procedure hence not only is the value of arg passed to par but the value of par is passed back to arg.

Example

```
Private sub cmddisplay-click()
Dim amt as single
picresult.cls
amt = 2
picresults.print amt;
call triple (amt)
picresult.print amt;
End sub

Private sub triple (byref num as single)
picresult.Print num;
num = 3 * num
picresult.print num;
End sub
```

Output

2 2 6 6

-It provides a vehicle for passing values form a sub procedure back to the place from which the sub procedure was called. The variable amt is said to be passed by reference.

Scope of variables

a)Local variables

when thesame variable name appears in two different sub procedures or sub procedure and an event procedure, visual basic gives the variable separate identities and treats them as two different variables. A value assigned to a variable in one part of the program will not affect the value of the other like named variable in the other part the

program, unless of course the values are passed by a call statement. Also each time a sub procedure is called, all declared variables that are not parameters assume their default values (numeric variables have default value 0 and string variables default is empty).

Example

```
Private sub cmddisplay – click()
Dim x as single
'Demonstrate the local value of variables
Picresults.cls
X = 2
Picresults.print x
Call trivial
PicResults. Print x
Call trivial
PicResults. Print x
Call trivial
PicResults,Print x;
End sub
```

```
Private sub trivial ()
Dim x as single
'Do something trivial
PicResults.Print x,
x= 3
PicResults.Print x,
End sub
```

Out put

2 0 3 2 0 3 2

b) Form level variables (Global)

VB provides a way to make a variable visible to every procedure in a form's code without being passed. Such a variable is called form-level variable. They reside in the (declaration) section of (general)

Example

The following program contains the form level variables num1 and num2 . Their Dim statement does not appear in a procedure.

```
Dim num1 as single, num2 as single
'In (declaration) section of (General)
Private sub cmddisplay_click()
'Display the sum of the numbers
Num1 = 2
```

```

Num2 = 3
PicResults.cls
Call AddAndIncrement
PicResults.print

PicResults.Print "num1 =", num1
PickResults.Print "num2 = ", num2
End sub

```

```

Private sub addandincrement()
'Display numbers and their sum
PicResults.print"sum of "; num1; "and"; num2; "is";num1 + num2
Num1 = num1 + 1
Num2 = num2 + 1
End sub

```

Output

```

Sum of 2 and 3 is 5
Num1 = 3
Num2 = 4

```

FUNCTIONS PROCEDURES

A function is simply a general procedure that returns one value to the function that calls it. Although the input can involve several values, the output always consists of a single value. The items inside the parenthesis can be constants, variables or expressions. Function procedures are defined by function blocks of the form:

```

Private function functionname (var1 as type1, var2 as type2) as data type
Statement(s)
Functionname = expression
End function

```

The variables in the top line are called parameters and variables in the function block that are not parameters have local scope. Function names should be suggestive of the role performed and most conform to the rules for naming variables.

Example of function procedures

```

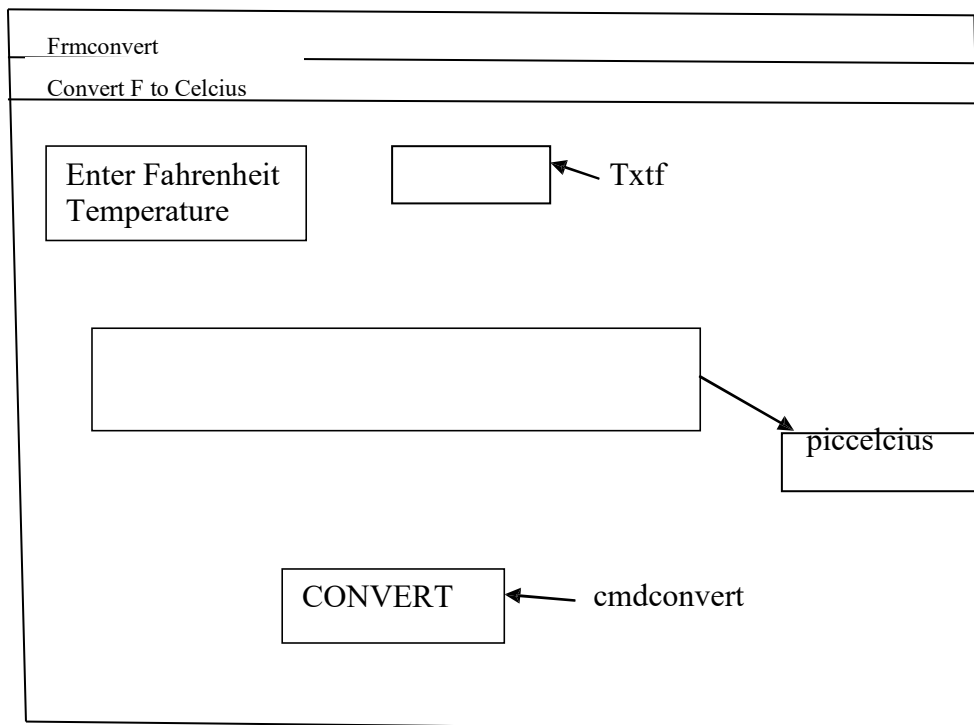
Private function FtoC (t as single) as single 'convert Fahrenheit temperature to Celcius
F toC = 5/9 *( t - 32 )
End function

```

Example

The following program uses the function FtoC to convert a Fahrenheit temperature to Celcius temperature.

a).UI



b).Code

```
Private sub cmconvert – click()
```

```
Dim Celcius As single
```

```
Dim f as single
```

```
f = val(txtf.text)
```

```
Celcius=FtoC(f) ‘call the function FtoC
```

```
Piccelcius.print “celcius temp is “, Celcius
```

```
End sub
```

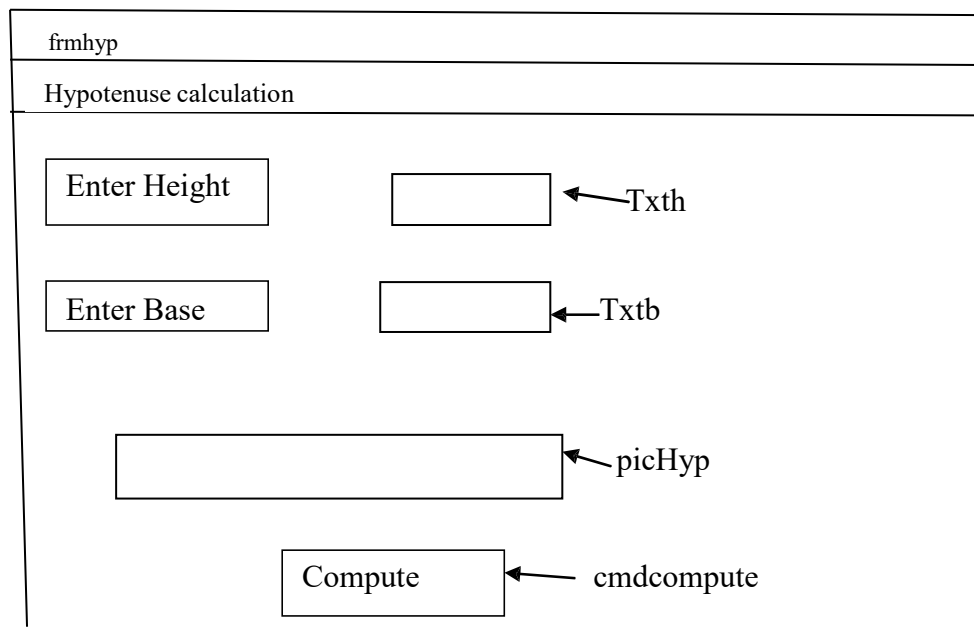
```
Private Function FtoC(t as single) as single
```

```
FtoC = (5/9) * (t – 32)
```

```
End function
```

Example

A program to calculate the hypotenuse of a right angle triangle given its base and height.



Code

```
Private sub and compute and click( )  
Dim a as single, Hyp as single  
Dim b as single  
a = val (txth. Text)  
b = val (txtb. Text)  
Hyp = Hypotenuse (a, b)  
Pichyp.Print "Hypotenuse is ", Hyp  
End sub
```

```
Private function hypotenuse (height as single, base as single) as single  
hypotenuse = sqr(height^2 + base^2)  
End function.
```

Exercise

Write a VB program using a function that accepts the dimensions of a closed cylinder, calculates and returns its surface area for display.

DATA STRUCTURES

Arrays

Is a data structure that stores several data items of the same type. The figure below shows how data of type integer is stored in an array.

20	30	80	120	200	250
index → 0	1	2	3	4	5

The array has cells. The numbers 0 to 5 are called array indices or subscripts. In V.B an array starts from cell 0 as shown in the table.

One dimensional array

Declaring an array

To declare an array use the statement `Dim ArrayName (n) As Datatype`, where n stands for the number of elements in the array **e.g**

`Dim scores (4) as integer` is an array that will hold five integer values

Assigning values to an array

Example

To **assign** a numeric value to location 4 of the array scores we use the statement `scores (4) = 90`

To **display** a value held in location 3 of the array scores we use the statement `picresult.print scores (3)`

NB: if the array is very large and you wish to read or enter values in it, it could be very tedious entering a value cell by cell i.e `score (0) = 10` `score(1) = 2..` `score (n) = x`
To avoid this, the for loop can be used. Remember this loop is used where the number of iterations are predetermined hence its suitability for use with arrays.

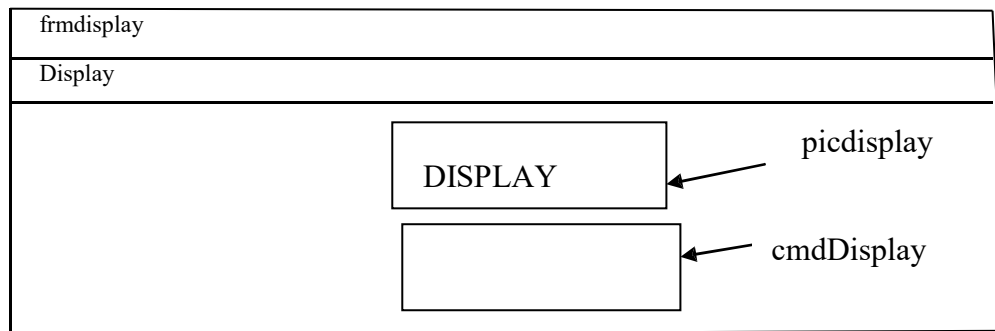
NB: The array declaration can also be adjusted as follows :

Dim A(1 to 5) as integer

In this case the array variable A is declared to hold five elements but we have forced the first available location in the array to be 1.

Example (on one dimension array)

A V.B program to prompt the user for five integer values to be stored in array then display them.



b).Code

```
Private sub cmdDisplay – click( )
```

```
Dim A(4) as integer
```

```
Dim i as integer, j as integer
```

```
For i = 0 to 4
```

```
A (i) = inputBox (“enter array value” ,“arrays”)
```

```
Next i
```

```
For j = 0 to 4
```

```
Picdisplay.Print A(j)
```

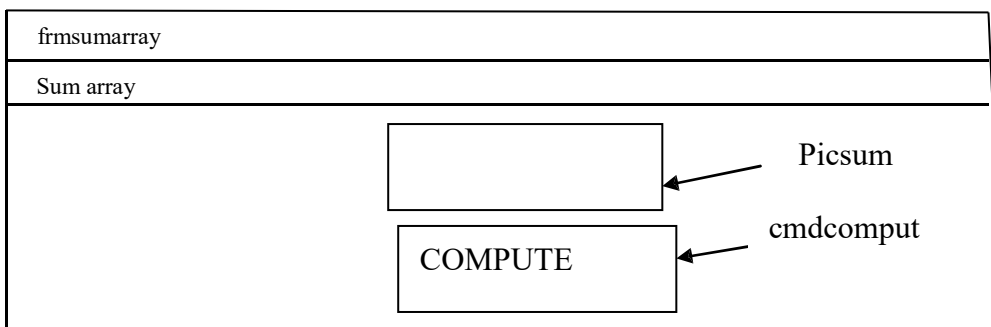
```
Next j
```

```
End sub
```

Example

A VB program to prompt user for an array of 3 integer elements, compute and display their sum

a).UI



b).Code

```

Private sub cmdcompute _click()
Dim num (1 to 3) as integer
Dim i as integer, sum as integer
Sum = 0
For i = 1 to 3
Num (i) = InputBox ("enter array value,"Array")
sum = sum + num(i)
Next i
Picsum.Print "sum of values is ", sum
End sub

```

Exercise

- Rewrite example 1. On arrays so that elements are displayed in reverse in which they are inputted.
2. Rewrite example 2 so that the average of the three inputted array values is also computed and displayed.

```

Dim A (4) as interger
Private sub cimddisplay – click( )
Dim I as integer, j as integer for I = 0 to 4
A(i) = inputbox ("enter array value", "arrays")
Next i
For J = 4 to 0
Picdisplay. Print A(j)
Next j
End sub

```

Code

```

Dim num (1 to 3) as interger private sub cmdcalculate – clicki) dim I as integer, sum
as integer
Dim average as integer
Sum= 0
For I = 1 to 3
Num (i) = input Box ("enter array value", arrays")
Sum = sum + num(i)
Next i
Average = sum/3
Picsum print "sum of value is ", sum
Picsum, print "average of value is " average
End sub.

```

TWO DIMENSIONAL ARRAYS

A two dimensional array is a data structure in which elements are arranged in rows and columns. Two subscripts are used to identify an item e.g score(2,4)

Means 3rd row 5th column this is because the array was declared as from row 0 as follows:

Dim score (0 to 2, 0 to 4) as integer **OR**

Dim score (2, 4) as integer

	0	1	2	3	4
0					
1					
2					
3					
4					Score (2,4)

Example

A program to accept an array of 4 marks from 5 different students, calculate and display total marks for each student alongside the marks on the form in the format shown below:

Math	Eng	Kisw	Science	Total
-	-	-	-	-
-	-	-	-	-
-	-	-	-	-
-	-	-	-	-
-	-	-	-	-

a).UI

frmmarks
Total marks
<div>cmdcompute</div> <div>COMPUTE</div>

b).Code

```
Private sub cmdcompute_click ( )
```

```
    Dim studmarks (1 to 5, 1 to 4) as integer
```

```
    Dim student as integer, mark as integer, col As integer
```

```
    Dim sum (1 to 5) as integer
```

```

Col = 1
Me.print "math", Eng", Kisw", "Science", "Total"
For student = 1 to 5
sum (student) = 0
For mark = 1 to 4
studmarks (student, mark) = inputbox ("key in student marks", "marks" )
sum (student) = sum(student) + studmarks(student, mark)
Me.print tab(col);studmarks(student, mark); 'print information starting from column 1
col = col + 6
If col > 20 Then
Me.print tab(col);sum(student) 'print sum of marks for student and begin newline
col = 1 'reset column to start printing next information from column 1
End if
Next mark
Next student
End sub

```

Exercise

1. Write a vb program to assign the value 65 to all the locations of a 4 x 4 array and display them in a table as shown

65	65	65	65
65	65	65	65
65	65	65	65
65	65	65	65

2. Rewrite our previous example on two dimensional array so that the average mark for each student is also computed and displayed alongside other values as shown.

Math	Eng	Kisw	Science	Total	Average
-	-	-	-	-	-
-	-	-	-	-	-
-	-	-	-	-	-
-	-	-	-	-	-
-	-	-	-	-	-

SORTING AND SEARCHING

SORTING

Is an algorithm for ordering an array.

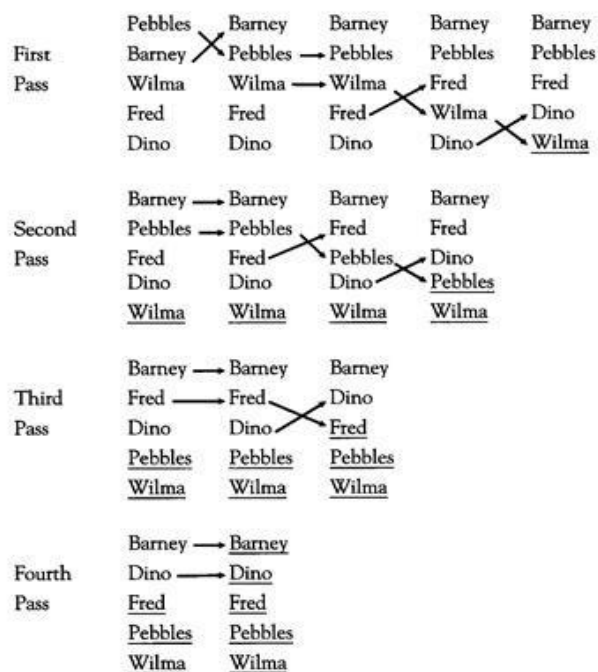
a).bubble sort

Is an algorithm that compares adjacent elements and swaps those that are out of order. This process is repeated enough times the list will be ordered. The steps for each pass through the list are as follows:

- 1.compare the 1st and 2nd item if they are out of order swap them
- 2.compare the 2nd and the 3rd items if they are out of order swap them

3.repeat this process for all the remaining pairs, the final comparison and the possible swap are between the 2nd last and the last element

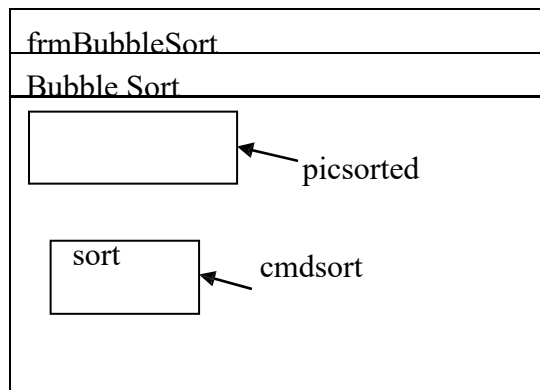
Illustration of how the elements list of names are sorted in ascending order using bubble sort(Illustration Page 193-VBA beginner how to...)



Example

A program to sort a list of 5 numbers assigned to an array in ascending order using bubble sort.

a).UI



b).Code

‘General declaration

```
Dim num (1 to 5) as integer
```

```
Private sub cmdsort_click ( )
```

```
Dim pass as integer, compare as integer, temp as integer
```

```
dim i as integer,
```

```
for pass = 1 to 5
```

```
for compar = 1 to 4
```

```
If num(compar) > num(compar+1) Then
```

```
temp = num(compar)
```

```
num(compar) = num(compar+1)
```

```
num (compar+1) = temp
```

```
End if
```

```
Next compar
```

```
Next pass
```

```
Picsorted.print "sorted elements are : "
```

```
For i = 1 to 5
```

```
Picsorted. print num(i)
```

```
Next i
```

```
End sub
```

```
Private sub form_load( )
```

```
Num (1) = 60
```

```
Num (2) = 3
```

```
Num (3) = 90
```

```
Num (4) = 7
```

```
Num (5) = 14
```

```
End sub
```

Exercise

Modify so that the above program prompts the user to enter the values to be sorted instead of being assigned to array on form_load.

SHELL SORT

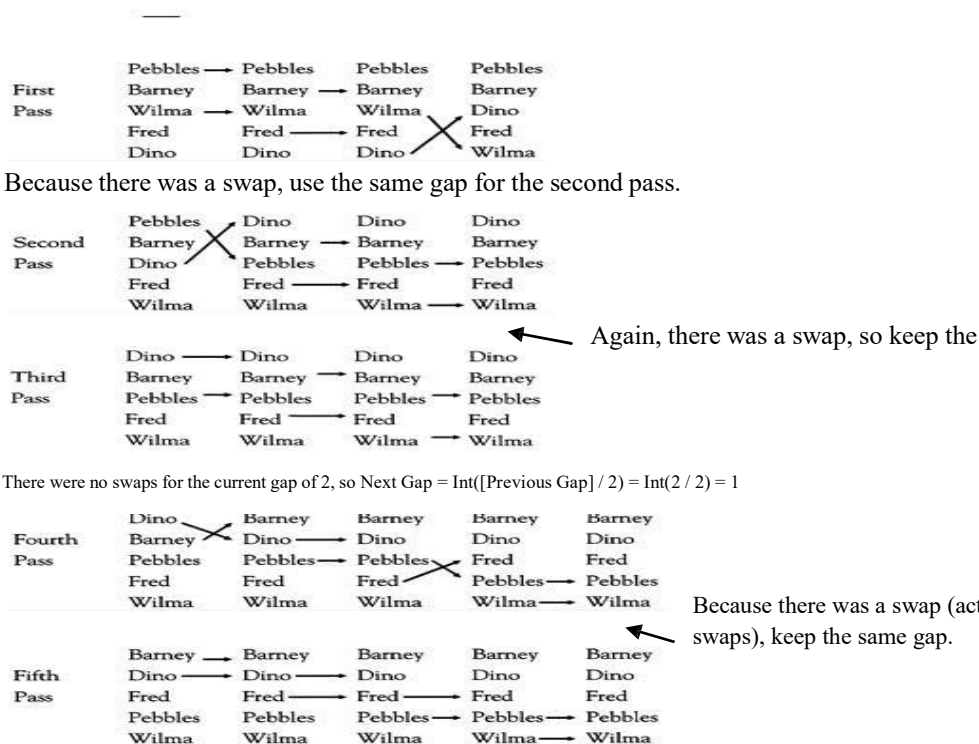
The bubble sort is easy to understand and program. However it is too slow for long lists. The shell sort is much more efficient in such cases. It compares distant items first and works its way down to nearby items. The interval separating the compared items is called the gap. The gap begins at one-half ($\frac{1}{2}$) the length of the list and it is successfully halved until eventually each item is compared with its neighbor as in the bubble sort. The algorithm for a list of n items is as follows:-

1. Begin with a gap of $g = \text{int}(n/2)$
2. Compare items 1 and $1 + g$, 2 and $2 + g$, ----- $n-g$ and n , swap any pairs that are out of order
3. Repeat step two until no swaps are made for gap g .
4. Halve the value of g
5. Repeat step 2,3 and 4 until the value of g is zero .

The shell sort is illustrated below. The elements are sorted in ascending order, crossing arrows indicate that a swap occurred.

Elements to be sorted are names: pebbles barney William, Fred, Dino

Initial gap = $\text{int}(\text{num of items}/2) = \text{int}(5/2) = 2$




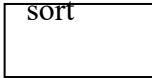
Because there were no swaps for the current gap, then Next Gap = $\text{Int}([\text{Previous Gap}] / 2) = \text{Int}(1 / 2) = 0$ and the Shell sort is complete.

Notice that the Shell sort required 14 comparisons to sort the list whereas the bubble sort required only 10 comparisons for the same list. This illustrates the fact that for very short lists, the bubble sort is preferable; however, for lists of 30 items or more, the Shell sort will consistently outperform the bubble sort.

Example

Program that sorts a list of five integers in ascending order using shell sort

a).UI

frmShellSort	
Shell Sort	
	← picsorted
	← cmdsort

b).code

‘General declaration

```
Dim num (1 to 5) as integer
```

```
const n as integer = 5
```

```
Private sub form _load()
```

‘Read the numbers into array

```
num (1) = 60
```

```
num (2) = 90
```

```
num(3) = 40
```

```
num(4) = 30
```

```
num(5) = 80
```

```
End sub
```

```
Private sub cmdsort – click ( )
```

‘sort and display elements

```
call sortdata()
```

```
call showData()
```

```
End sub
```

```
Private sub sortData( )
```

```
Dim gap as integer, Doneflag as boolean
```

```
Dim index as integer, temp as integer
```

```
Gap = int (n/2)
```

```
Do while gap >= 1
```

```
Do
```

```
Doneflag = true
```

```
For index = 1 to n-gap
```



```

If num(index) > num (index + gap) then
Temp = num (index)
Num (index) = num(index + gap)
Num (index + gap) = temp
Done flag = false
End if
Next index
Loop until (doneflag = true)
Gap = int(gap/2)
Loop
End sub

```

```

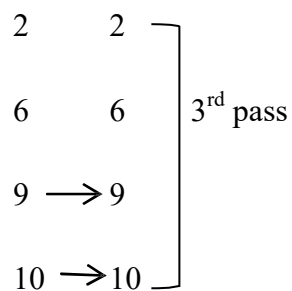
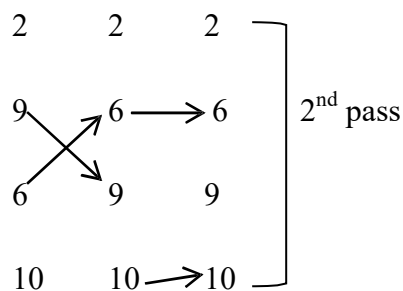
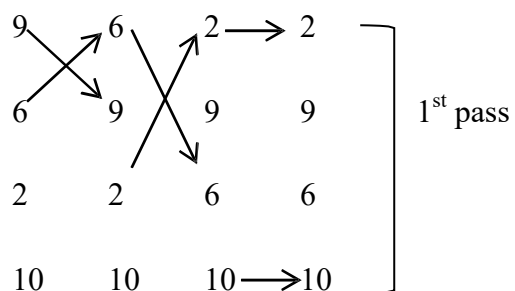
Private sub showdata( )
Dim i as integer
For i = 1 to 5
Picsorted.Print num(i)
Next i
End sub

```

Selection sort

In selection sort the item in the 1st position is compared with the other elements. If the elements being compared are not in order they are swapped.

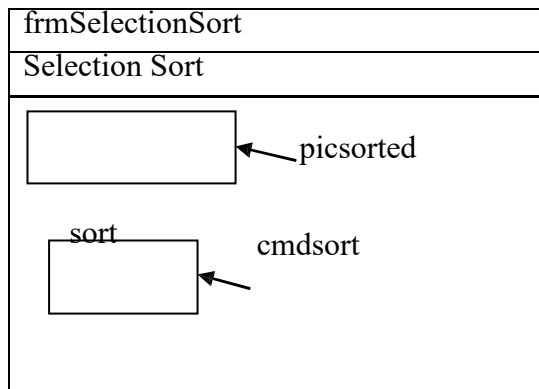
Illustration on how the elements 9, 6, 2, 10 are sorted in ascending order



Example

A program to sort a list of five names in ascending order using selection technique.

a).UI



‘General declaration

```
Dim name (1 to 5) as string
Private sub cmdsort_click()
Dim passnum as integer, compar as integer
For passnum = 1 to 4
For compar = passnum + 1 to 5
If name (passnum) > name (compar) Then
Temp = name(passnum)
Name (passnum) = name(compare)
Name (compar) = temp
End if
Next compar
Next passnum

For i = 1 to 5
Picsorted.print name (i)
Next i
End sub

Private sub form_load()
Name (1)="mike"
Name(2)= "Zachary"
Name(3)= "Abel"
Name (4)= "James"
Name(5)= "William"
```

End sub

SEARCHING TECHNIQUES

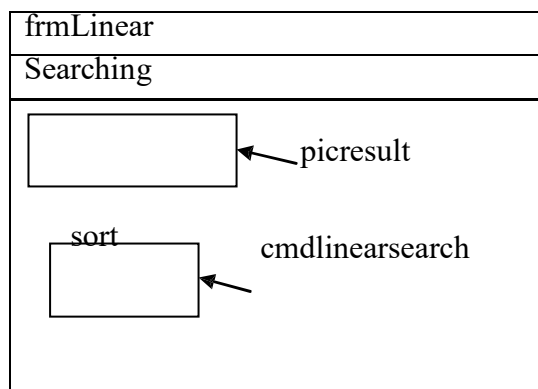
1.Linear search/ sequential search

Compares the **key** (element being searched) successively with the elements in the search list. If a match is found the program reports that the element is in the list(i.e found) otherwise the program reports that the element is not in the list(i.e not found).

Example

A program to search for an element from an array

a).UI



b).Code

```
Dim list(1 to 5) as integer
Private sub cmdlinearsearch_Click()
Dim key as integer, i as integer
Key = Inputbox ("key in element to search ", "search")
'Initialize first location to search from
i= 1
While (list(i) <> key) AND (i <> 6)
i = i +1 'check the next location
Wend
If (list(i) = key) Then
Picresult.print "element found"
Else
Picresult.print "element not found"
End if
End sub

Private sub form_load()
List(1)= 7
List(2) = 2
List (3)= 4
List (4)= 10
```

List (5) =3

End sub

2. Binary Search

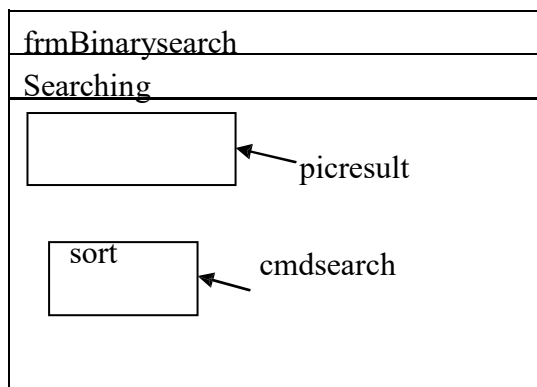
Works on a sorted list,

- i. Divides an array by half by finding the middle element in the array i.e $\text{middle element} = \text{first} + \text{last}/2$
- ii. It then compares the middle element with the key.
 - a. If the middle element is greater than the key then the last location is computed by $\text{last} = \text{middle} - 1$ and search proceeds to the first half of array.
 - b. If the middle element is less than the key then the new first location is computed by $\text{first} = \text{middle} + 1$ and search proceeds to the second half of the array.
 - c. If the middle element is equal to the key the search is over and the program reports that the element is in the list otherwise program displays that the element was not found.

Example

A program to search for an element from an array

a).UI



b).code

```
dim list(1 to 5) As integer private
sub cmdsearch_(Click) dim key as
integer, flag as boolean
key = inputbox ("enter value to search","search")
dim first as integer, middle as integer
dim last as integer
flag = false
first = 1
last = 5
Do while (first <= last) and (flag = false)
middle = int((first + last)/2)
```

```

Select case list(middle)
case is = key
flag = true
case is > key
last = middle-1
case is < key
first = middle + 1
End select
Loop
If (flag=true)Then
picresult.print "found"
else
picresult.print "not found"
end if
End sub

```

```

private sub form_load()
list (1)= 10
list(2)=20
list(3) =30
list(4)=40
list(5)=60
End sub

```

FILES

SEQUENTIAL FILES

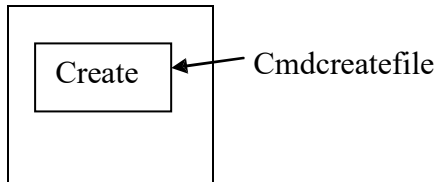
Creating sequential files

1. Choose a file name. A file name contains up to 255 characters consisting of letters, digits, and a few other assorted characters but including spaces and periods.
2. Choose a number from 1 through 511 to be the reference number of the file, while the file is in use it will be identified by this number.
3. Execute the statement open "filespec" FOR OUTPUT AS #n, where n is the reference number. This process is referred to as opening a file for output. It allows data to be output from the computer and recorded in the specified files.
4. Place data into the file with the write#n statements e.g. if a is a string then the statement write #n, a writes the string surrounded by quotation marks into the file.
If c is a number then the statement write #n, c writes the number c without any leading or trailing spaces in the file number.
5. After the data has been recorded in the file, execute close # n whenever n is the reference number. This statement breaks the communication link with the file and dissociates the number n from the file.

Example

The following program illustrates the different statements of the write statement. Notice the absence of the leading & trailing spaces for numbers and the presence of quotation marks surrounding strings

a).UI



b).Code

```
Private sub cmdcreatefile_click()  
Dim name1 as string, name2 as string  
Dim num as integer  
Num = 80  
Name1 = "Eniac"  
Name2 = "Mauldily"  
Open "C: \pioneer.txt " for output as # 1  
Write # 1, "Eniac", 1946, num  
Write # 1,1946, name1, name2  
Write #1, 14*139, "J.P" & name1, name2,"joe"  
Close #1  
End sub
```

N/B : If an existing sequential file is opened for file mode output. The computer will erase the existing data and create a new empty file.

ADDING ITEMS TO A SEQUENTIAL FILE

Execute the statement;

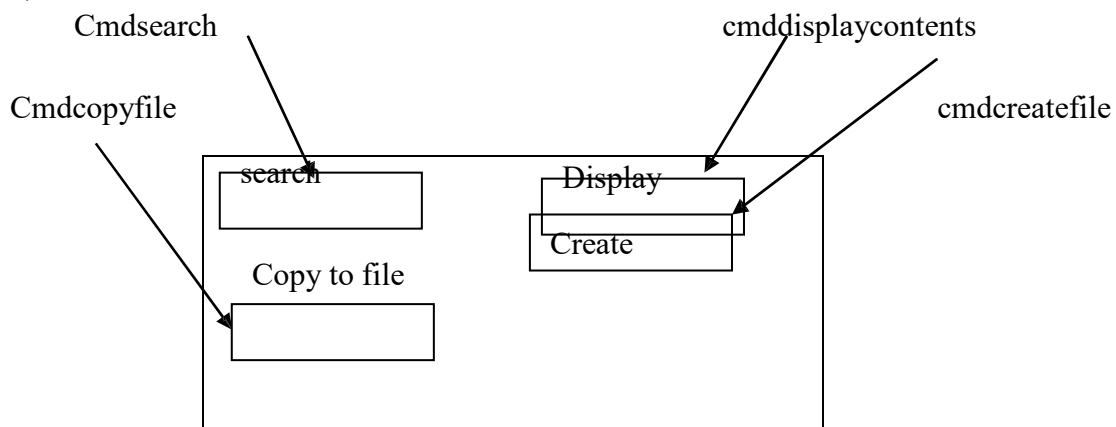
Open "filespec" for append as #n

The append option for opening the file is intended to adding data to an existing file.

Example

Code to create a file and add data to an existing file using append

a).UI



b).Code

```
Private sub cmdcreatefile_Click()  
Dim intmsg as integer  
Dim num as integer , i as integer  
Open "C : \numberlist.txt" for append as #1  
Intmsg = msgbox ("numberlist.txt opened successfully")  
For num = 1 to 6  
i = Inputbox ("enter number to store in file ")  
Write # 1,i Next num  
Intmsg=msgbox("numberlist.txt closed")  
Close # 1  
End sub
```

Example**Displaying contents of a sequential file (program)****Notes:**

Open "number.txt" for input is #1
Input mode - opens an existing file for a read operation

```
Private sub cmddisplaycontents_Click()  
Dim num as integer  
Open "c: \ numberlist.txt" for input as #1  
Do while not EOF(1)  
Input # 1, num  
Me.print, num  
Loop  
Close# 1  
Msgbox "contents diplayed"  
End sub
```

Copying contents of numberlist file to numberlistcopyfile

```
private sub cmdcopyfile_click()  
dim num as integer  
open "C : \ numberlist.txt "for input as # 1  
open "C : \ numberlistcopy.txt" for append as #2  
Do until EOF(1)  
input#1, num  
print#2,num  
Loop  
close # 1  
close # 2  
msgbox" contents copied"  
End sub
```

Searching for file contents

```
private sub cmdsearch_click()  
dim num2search as integer, num as integer  
open "C ; \numberlist.txt" for input as #1  
Do while (num<>num2search) and NOT EOF(1)  
INPUT#1,num  
Loop  
If num2search=num Then  
Me.print,num2search  
Msgbox"contents displayed"  
Else  
Msgbox"no such record"  
End if  
Close # 1  
End sub
```

RANDOM ACCESS FILES

A random-access file is like an array of records stored on a disk. The records are numbered 1, 2, 3, and so on, and can be referred to by their numbers.

Therefore, a random-access file resembles a box of index cards, each having a numbered tab. Any card can be selected from the box without first reading every index card preceding it; similarly, any record of a random-access file can be read without having to read every record preceding it.

One statement suffices to open a random-access file for all purposes: creating, appending, writing, and reading. Suppose a record type has been defined with a Type block and a record variable, called recVar, has been declared with a Dim statement. Then after the statement

Open "filespec" For Random As #n

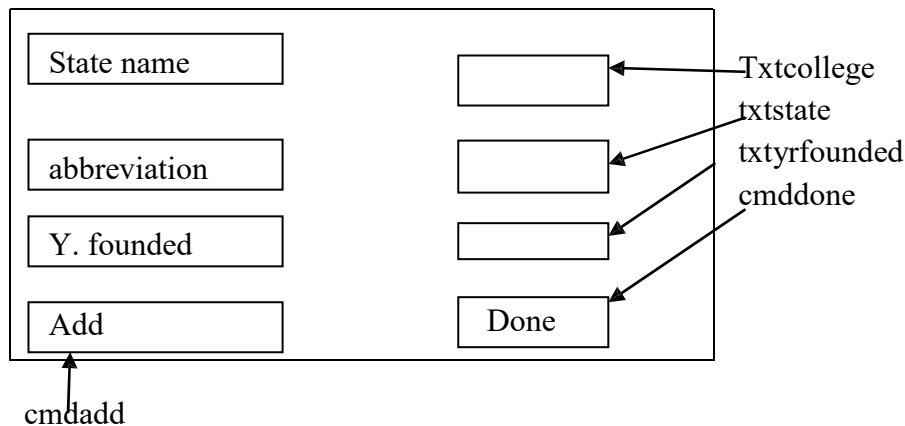
is executed, records may be written, read, added, and changed. The file is referred to by the number n. Each record will have as many characters as allotted to each value of recVar. Suppose appropriate Type, Dim, and Open statements have been executed. The two-step procedure for entering a record into the file is as follows.

1. Assign a value to each field of a record variable.
2. Place the data into record r of file #n with the statement

Put #n, r, recVar where recVar is the record variable from Step 1.

Example

The following program creates and writes records to the random access file known as colleges.txt



‘module.BAS

```
Public type collegeData
nam as string *30
state as string*20
yrfounded as integer
end type
```

‘general declaration

```
Dim recordnum as integer
Private sub cmdadd_click()
dim college as collegeData
college.nam=txtcollege.text
college.state = txtstate.text
college.yrfounded = val(txtYrfounded.text)
recordnum = recordnum+1
put #1 ,recordnum,college
txtcollege.text =""
txt y founded.text = ""
txtcollege.setfocus
End sub
```

```
private sub cmddone_click()
close#1
end sub
```

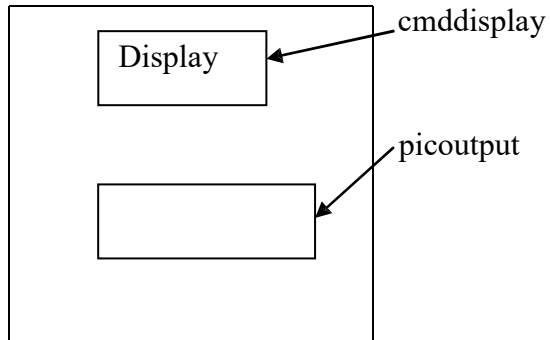
```

private sub form_load()
Dim college collegeData
Open"C : \ colleges" for random as #1
Recordnum = 0
End sub

```

Example

Displaying the contents of a random file



```

Private sub cmddisplay_click()
Dim recordnum as integer
Dim college as collegedata
open "C : \ colleges.txt" for random as # 1
picoutput .cls
picoutput.print "colleges"; tab(30); "state";tab(45); " year founded"
for recordnum=1 to 5
get #1, recordnum,college
picoutput.print college.nam;tab(30); college.state; tab(45); college.yrfounded
next recordnum
close #1
end sub

```

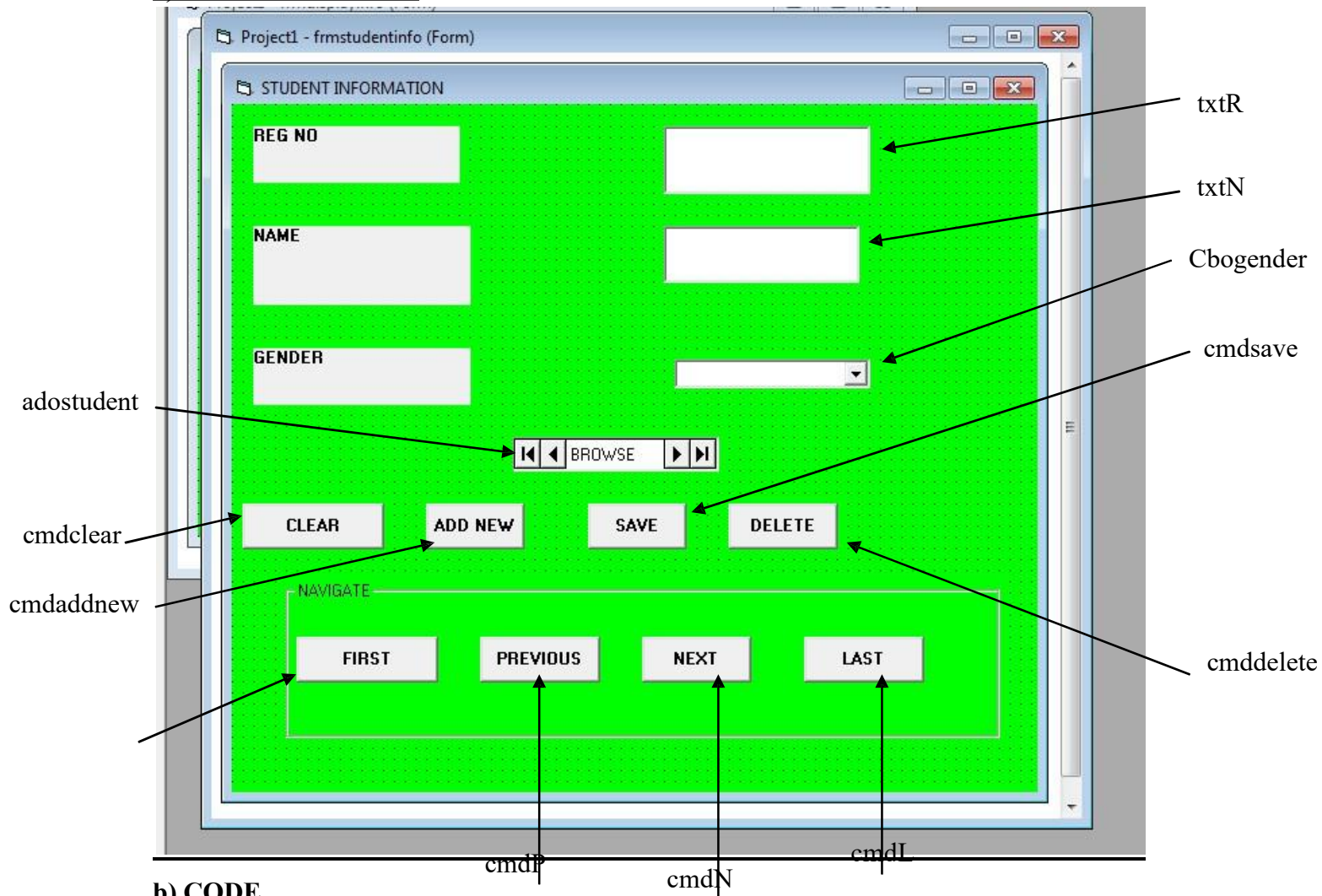
APPLICATION AND DATABASES

-Procedure for linking to database and generation of reports in:

- Vb tutor Revised.pdf-pg 59-82-Read on Lesson 21: Creating VB database applications using ADO control
- Visual basic AVU.pdf-pg 344-360-Read on reports

SAMPLE PROJECT EXAMPLE AND CODE

a).USER INTERFACE



b).CODE

```
Private Sub Form_Load()  
    cbogender.AddItem "MALE"  
    cbogender.AddItem "FEMALE"  
End Sub
```

```
Private Sub cmdclear_Click()  
    txtR.Text = ""  
    txtN.Text = ""  
    cbogender.Text = ""  
End Sub
```

```
Private Sub cmdaddnew_Click()
```

```
Adostudent.Recordset.AddNew
End Sub
```

```
Private Sub cmdsave_Click()
Adostudent.Recordset.Fields("Regno") = Val(txtR.Text)
Adostudent.Recordset.Fields("Name") = txtN.Text
Adostudent.Recordset.Fields("Gender") = cbogender.Text
Adostudent.Recordset.Update
MsgBox "Record saved"
End Sub
```

```
Private Sub cmddelete_Click()
Adostudent.Recordset.Delete
End Sub
```

```
Private Sub cmdF_Click()
Adostudent.Recordset.MoveFirst
End Sub
```

```
Private Sub cmdL_Click()
Adostudent.Recordset.MoveLast
End Sub
```

```
Private Sub cmdN_Click()
Adostudent.Recordset.MoveNext
If Adostudent.Recordset.BOF Then
Adostudent.Recordset.MoveFirst
End If
End Sub
```

```
Private Sub cmdP_Click()
Adostudent.Recordset.MovePrevious
If Adostudent.Recordset.BOF Then
Adostudent.Recordset.MoveLast
End If
End Sub
```

Code to select and display details of a record from a database on a list box and or/ using a Combo box

‘Code to display intended information on combo box and list box

```
Private Sub Form_Load()
List1.AddItem ("NAME" & " " & "GENDER")
List1.AddItem ("-----" & " " & "-----")
While Not Adogetstud.Recordset.EOF
cboreg.AddItem (Adogetstud.Recordset!regno)
List1.AddItem (Adogetstud.Recordset!Name & " " & Adogetstud.Recordset!Gender)
Adogetstud.Recordset.MoveNext
Wend
Adogetstud.Recordset.MoveFirst
End Sub
```

‘Code to select and display information using combo box

```
Private Sub cboreg_Click()
While Not Adogetstud.Recordset.EOF
If cboreg.Text = Adogetstud.Recordset!regno Then
txtn.Text = Adogetstud.Recordset!Name
txtg.Text = Adogetstud.Recordset!Gender
Adogetstud.Recordset.MoveFirst
Exit Sub
End If
```

```
Adogetstud.Recordset.MoveNext  
Wend  
End Sub
```

```
Private Sub txtb_KeyUp(KeyCode As Integer, Shift As Integer)  
Const feeRequired As Integer = 20000  
txtb.Text = feeRequired - Val(txtb.Text)  
End Sub
```