**VISUAL BASIC**

***Question:***

***What is Visual Basic programming language?***

**Visual Basic** is a tool used to develop Windows (Graphic User Interface - **GUI**) applications.

***Name any 4 VB events you know.***

button press, click, mouse movement, double click

**FEATURES OF VISUAL BASIC**

1. A set of objects called tools. They are used to 'draw' the application
2. Allows database integration with many applications
3. Lots of icons and pictures for use
4. Response to mouse and keyboard actions
5. Clipboard and printer access
6. Has many pre-defined functions
7. Supports sequential and random file access
8. Has debugger and error-handling facilities
9. Powerful database access tools
10. ActiveX support
11. Can support internet

***Question:***

***What is a variable?***

Variables refer to named memory location which holds data.

***State the rules followed in naming a variable.***

The following are the rules when naming the variables in Visual Basic:

1. It must be less than 255 characters
2. No spacing is allowed
3. It must not begin with a number
4. Period is not permitted
5. Can begin with alephabate letter
6. Special symbols such as #, ?, / are not allowed
7. Under score is allowed

***Question:***

***What are the components of a VB Application?***

**VISUAL BASIC APPLICATION**

This is made up of:

(a) **Forms** - Windows that you create for user interface

(b) **Controls** - Graphical features drawn on forms to allow user interaction

These include: **text boxes, labels, scroll bars, command buttons….**

(Forms and Controls are **objects**.)

(c) **Properties** - Every characteristic of a form or control is specified by a

property. Example properties include names, captions, size, color,

position, and contents. Visual Basic applies default properties. You can

change properties at design time or run time.

(d) **Methods** - Built-in procedure that can be invoked to impart some action to a particular object.

(e) **Event Procedures** - Code related to some object. This is the code that is

executed when a certain event occurs.

(f) **General Procedures** - Code not related to objects. This code must be

invoked by the application.

(g) **Modules** - Collection of general procedures, variable declarations, and

constant definitions used by application.

***State any ten controls available in VB.***

Command button

Pointer

Label

Check box

Combo box

Image box

Picture box

File list box

List box

Option button

File list box

Frame

Text box

Timer

**VB MODES:**

They are three namely:

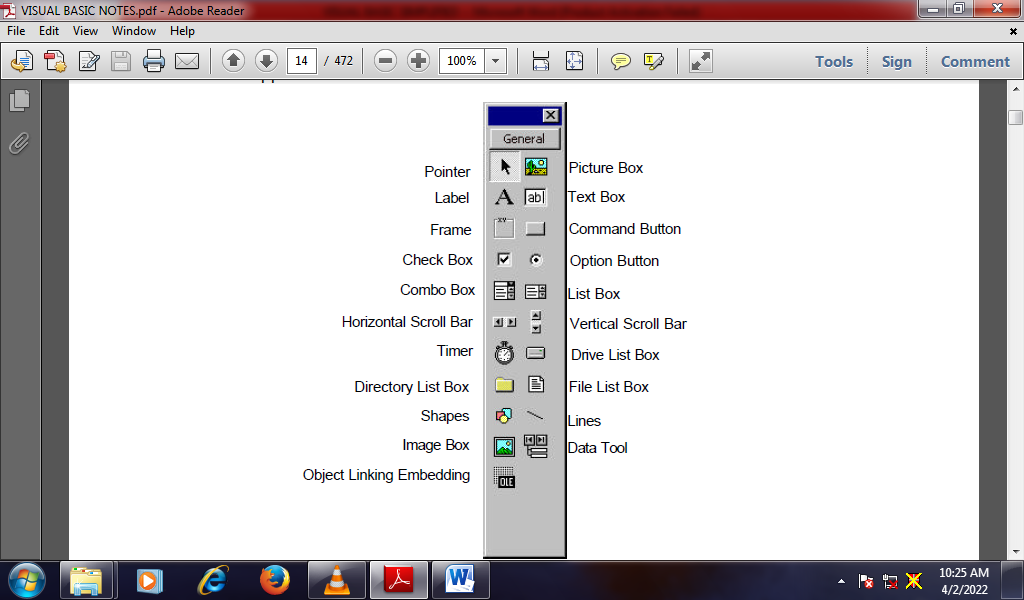
(a) Design mode - used to build application

(b) Code mode – used to attach codes to the controls.

(c) Run mode - used to run the application

However, VB may also be in **Break** mode meaning the application is halted.

The Tool Box has a set of tools used to develop applications.



**Label2.Text = Val(TextBox1.Text) + Val(TextBox2.Text)**

**The above code displays the sum of the value stored in TextBox1 and TextBox2 in a label named Label2.**

**(c) List Box**

The function of the list box is to present a list of items where the user can click and select the items from the list.

**ListBox1.Items.Add("ORANGE")**

***Question:***

***What is the syntax for adding an item to a ListBox?***

**ListBox1.Items.Add("ORANGE")**

To remove Item “ORANGE” from the ListBox, we use:

**ListBox1.Items.Remove("ORANGE")**

**ComboBox1.Items.Add("ITB")**

***Question:***

***What will the codes below do?***

1. ***ComboBox1.Items.Remove("ITB")***

It will remove ITB from ComboBox1

1. ***MyCombo.Items.Remove(MyCombo.SelectedItem)***

It will remove selecteditem from MyCombo

1. **ComboBox1.Items.Clear()**

It will clear items in ComboBox1

***Question:***

***Name any seven controls found on the Tool Box.***

Command

Pointer

Label

Check box

Combo box

Image box

Picture box

File list box

List box

Option button

File list box

Frame

Text box

Timer

***Describe the three most important windows of a VB project.***

**Project window.**

Is a sub window of the main window which holds the frame window.

**Frame Window.**

Is where you draw your application.

**Properties window**

This is used to set properties of the controls.

**The Tool Box**

The Tool Box has a set of tools used to develop applications.

**StartTime = Now**

**lblStart.Caption = Format(StartTime, "hh:mm:ss")**

**displays start time in a format of hours, minutes, seconds**

**SAMPLE CODES:**

**Private Sub cmd1\_Click()**

**Dim a, b, sum As Integer**

**a = Val(Text1.Text)**

**b = Val(Text1.Text)**

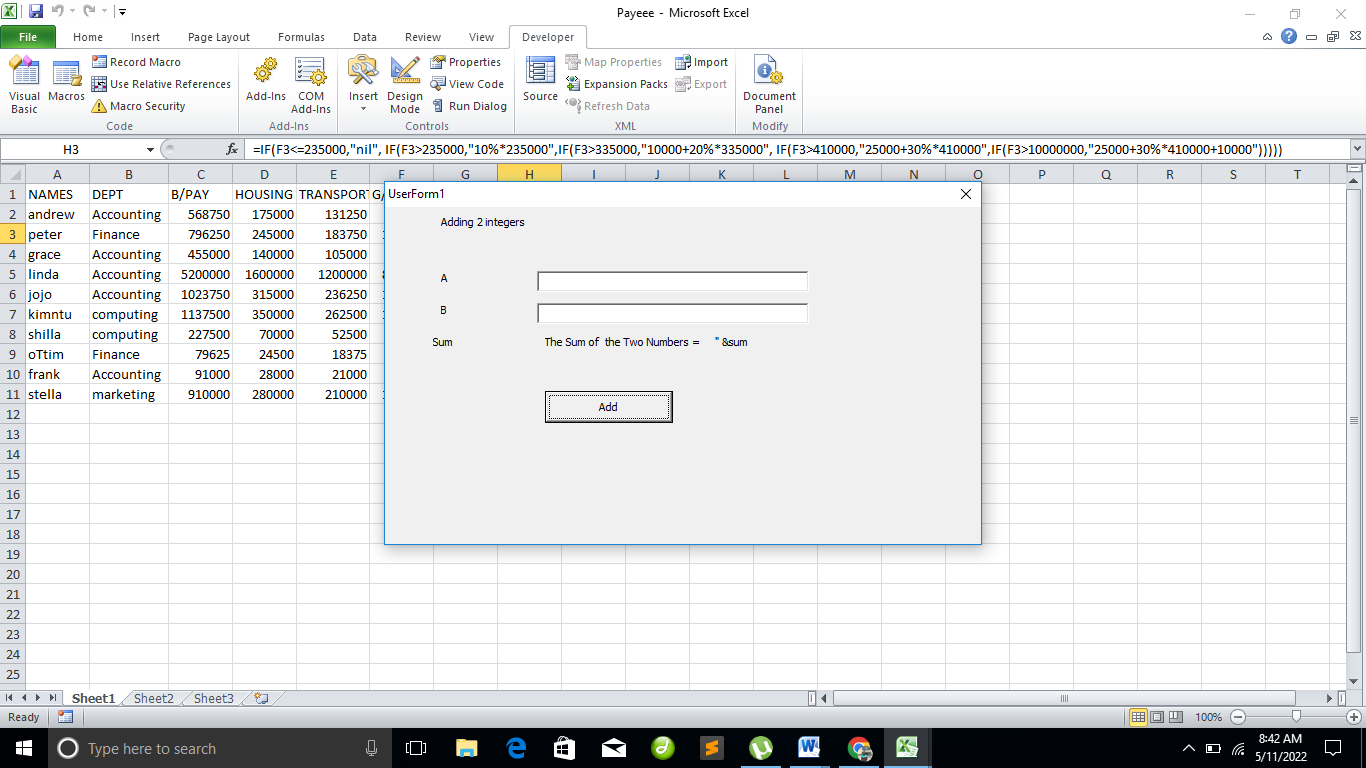
**sum = a + b**

**label3 = ”The Sum of the Two Numbers = “ &sum**

**End Sub**

***Question:***

*Develop an interface whose codes are displayed above.*



***What is the purpose of the above program?***

Calculate the Sum of the Two Numbers

***What is the purpose of each of the five lines in the program?***

**Dim a, b, sum As Integer**

Declare a, b, and sum as integer

**a = Val(Text1.Text)**

captures the value of a in Text1.Text

**b = Val(Text1.Text)**

overrides the value of a in Text1.Text

**sum = a + b**

adds a and b, in variable sum

**label3 = ”The Sum of the Two Numbers = “ &sum**

Displays the sum of a and b

**Example:**

**Dim grade As String**

**Private Sub Compute\_Click ()**

**grade = txtgrade.Text**

**Select Case grade**

**Case “A”**

**Label1.Text = “Distinction”**

**Case “B”**

**Label1.Text = “Credit”**

**Case “C**

**Label1.Text = “Pass”**

**Case “Else”**

**Label1.Text = “Fail”**

**End Select**

**End Sub.**

***Question***

***What would this program be used for?***

Used to grade

***Name any three possible controls used in this program.***

Command button

Label

Text box

Qtn. Private Sub Btn1\_Click(ByVal sender As System.Object, byVal e As System.EventArgs) Handles Btn1.Click

Dim mk As Single

mk = Txt1.Text

Select Case mk

Case 0 To 49

Lbl1.Text = “E”

Case 50 To 59

Lbl1.Text = “E”

Case 0 To 49

Lbl1.Text = “D”

Case 60 To 69

Lbl1.Text = “C”

Case 70 To 79

Lbl1.Text = “B”

Case 80 To 100

Lbl1.Text = “A”

Case Else

Lbl1.Text = “Error, Please enter a mark”

End Sub.

Note: **ByVal** is short or **By Value** instead of **By Reference**.

**Sender** is an object of **System.Object** class

**e** is an object within **System.EventArgs**.

Passing arguments can be by reference or by value. In this case we passing arguments by value.

***Question***

***What would be the output of the above program?***

***Grade marks interms of A,B,C,….***

***Study the codes and explain the program.***

Dim Mark As Integer

This line declares Mark as integer

Dim Grade as String

This line Grade Mark as string

Mark = Val(TextBox1.Text)

Captures the value of mark in textBox1.text

If Mark>=80 Then

Grade="A"

This code will grade marks greater or equal to 80 as A

ElseIf Mark>=60 And Mark<80 Then

Grade="B"

Grade as B if mark is greater or equal to 60 and less than 80

ElseIf Mark>=50 And Mark<60 Then

Grade="C"

Grade as C if mark is greater or equal to 50 and less than 60

Else

Grade="D"

Marks below 50 are graded as D

End If

MsgBox("You grade is " & Grade)

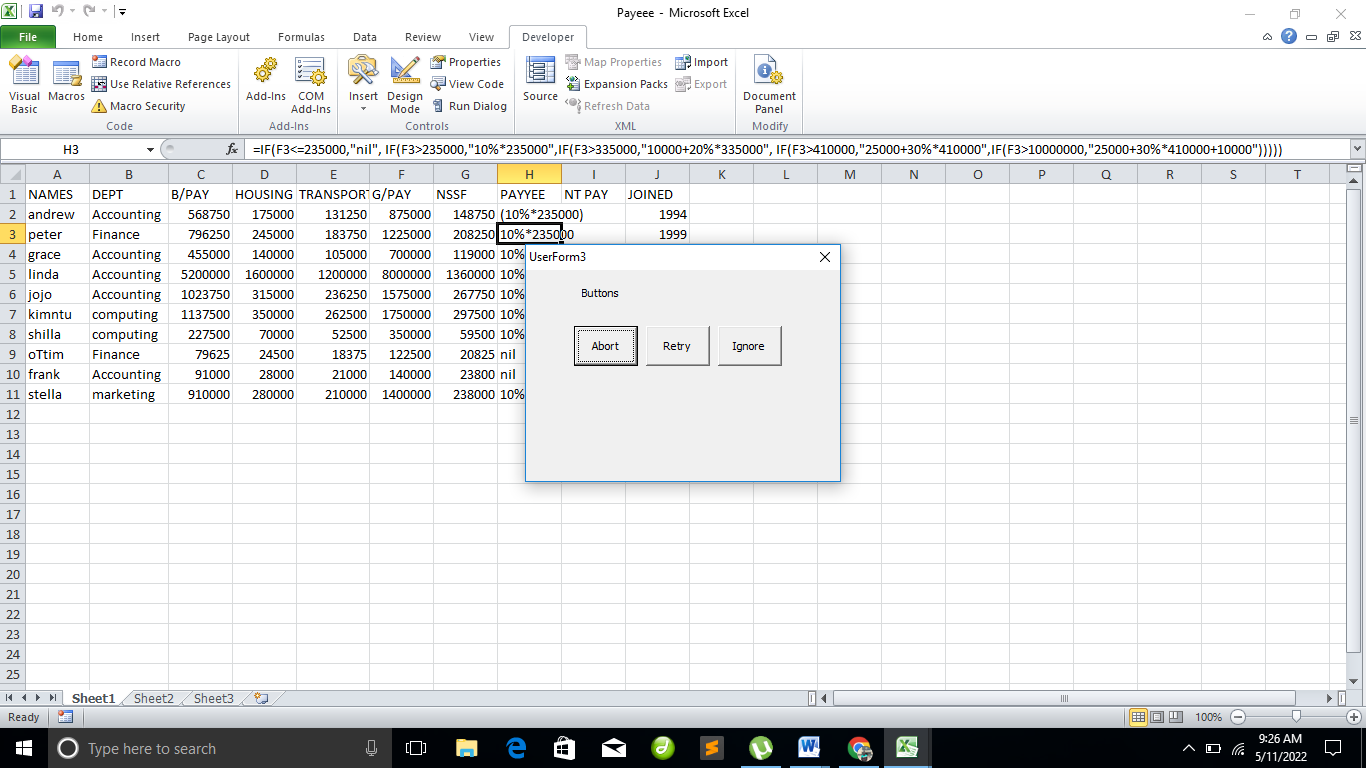
Dispplays the grade.

End Sub

***Question***

*What buttons would be displayed by each of the above.*

**vbAbortRetryIgnore**



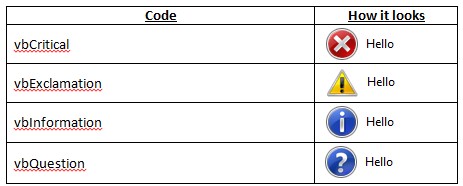
**ICONS**

**vbCritical**

**vbInformation**

**vbExclamation**

**vbQuestion**

****

**Response = MsgBox("Incorrect password", vbRetryCancel + vbCritical, "Access Denied")**

**Dim counter, sum As Integer**

**sum = 1000**

**For counter = 100 To 5 Step -5**

**sum - = counter**

**ListBox1.Items.Add(sum)**

**Next**

**Question:**

**Design a form and write code to find the summation of even numbers (from 0 to 100).**

**Solution:**

**Private Sub form\_load() Form1.show**

**Dim a As Integer**

**Total As Integer**

**For a = 0 To 100 step 2**

**Total= Total + a**

**Next a**

**Print "Total= "; Total**

**End Sub**

**REVISION QUESTIONS**

**PART ONE**

1. **(a) Explain what is meant by the term Graphical User Interface**

**(b) State three (3) benefits of a good interface design**

* **Increases productivity/efficiency**
* **Reduces errors**
* **Reduces costs for both training and fixing problems etc.**
* **Increases employee satisfaction and therefore acceptance**
* **Improves customer service**

1. **(a) In today’s world of computer programming, give three (3) reasons why**

**Visual Basic is preferred among the high level languages.**

* **Simple**
* **Popular**
* **Has many tools**
* **Provides internet tools**
* **Many inbuilt methods (functions)**
* **Easy to integrate to database**
* **Has powerful debuggers**
* **Event driven making it easy to control**

1. **(a) Explain the difference between a label and a textbox control as used in the design of interfaces**

**Text in Label is static, meaning contents cannot be changed at run time while the one for Textbox can be modified**

**(b) Name any two (2) properties that apply to the above controls**

* **Name**
* **Text**
* **Font**
* **TextAlign property.**

1. **Write down the syntax of the *If….Then….Else* statement and explain how it works**

**If*ConditionToCheck* Then**

***Statement1***

**Else**

***Statement2***

**End If**

1. Explain the meaning of each of the following lines in the program below.

Show what will be printed after compiling and running the given code

1. *'Demonstrate the Do While loop This is a comment*
2. *Dim intCount As Integer = 0 Declaring a variable intCount and assigning it value 0*
3. *Do While intCount < 10 Test condition if intCount is less than 10*
4. *lstOutput.Items.Add("Hello") Display Hello ten times*
5. *intCount += 1 Increment intCount by 1*
6. *Loop*

**PART TWO**

**Question 1**

1. Describe any **THREE** advantages of using graphical user interfaces.
2. Symbols recognized faster than text.
3. Faster learning. Research has also found that graphical, pictorial representation aids learning, and symbols can also be easily learned.
4. Faster use and problem solving. Visual or spatial representation of information has been found to be easier to retain and manipulate and leads to faster use and more successful problem solving. Symbols have also been found to be effective in conveying simple instructions.
5. Fewer errors. Reversibility of actions reduces error rates because it is always possible to undo the last step. Error messages are less frequently needed.
6. Increased feeling of control. The user initiates actions and feels in control. This increases user confidence and hastens system mastery.
7. Immediate feedback. The results of actions furthering user goals can be seen immediately.
8. Predictable system responses. Predictable system responses also speed learning.
9. Easily reversible actions. The user has more control. This ability to reverse unwanted actions also increases user confidence and hastens system mastery.
10. Less anxiety concerning use. Hesitant or new users feel less anxiety when using the system because it is so easily comprehended, is easy to control, and has predictable responses and reversible actions.
11. May consume less space. Icons may take up less space than the equivalent in words. More information can often be packed in a given area of the screen.
12. Replaces national languages. Language-based systems are seldom universally applicable. Language translations frequently cause problems in a text-based system. Icons possess much more universality than text and are much more easily comprehended worldwide.
13. Easily augmented with text displays. Where graphical design limitations exist, it can easily be augmented with text displays. The reverse is not true.
14. Low typing requirements. Pointing and selection controls, such as the mouse or trackball, eliminate the need for typing skills.
15. Smooth transition from command language system. Moving from a command language to a direct-manipulation system has been found to be easy. The reverse is not true.

(b) Describe any **THREE** advantages of using graphical user interfaces.

1. **Greater design complexity.** The elements and techniques available to the graphical screen designer far out number those that were at the disposal of the text-based screen designer.
2. **Learning still necessary.** The first time one encounters many graphical systems, what to do is not immediately obvious. The meanings of many words and icons may not be known. It is not often possible to guess their meanings, especially the more arbitrary symbols. How to use a pointing device may also have to be learned.
3. **Lack of experimentally-derived design guidelines.** The graphical interface is still burdened today by a lack of widely available experimentally-derived design guidelines. Early on, more developer interest existed in solving technical rather than usability issues, so few studies to aid in making design decisions were performed.
4. **Inconsistencies in technique and terminology.** Many differences in technique, terminology, and look and feel exist among various graphical system providers, and even among successive versions of the same system. These inconsistencies occur because of copyright and legal implications, product differentiation considerations, and our expanding knowledge about the interface. The result is that learning, and relearning, for both designers and users are much more difficult than it should be.
5. **Not always familiar.** Symbolic representations may not be as familiar as words or numbers. We have been exposed to words and numbers for a long time.
6. **Few tested icons exist.** Icons must appear in different sizes, weights, and styles. Icons must be researched, designed, tested, and then introduced into the marketplace.
7. **Inefficient for touch typists.** For an experienced touch typist, the keyboard is a very fast and powerful device. Moving a mouse or some other pointing mechanism may be slower.
8. **Not always the preferred style of interaction.** Not all users prefer a pure iconic interface. A study comparing commands illustrated by icons, icons with text, or text-only, found that users preferred alternatives with textual captions.
9. **Increased chances of clutter and confusion.** A graphical system does not guarantee elimination of clutter on a screen. Instead, the chance for clutter is increased, thereby increasing the possibility of confusion. The possibility that clutter may exist is evidenced by the fact that many people, when working with a window, expand it to fill the entire display screen.
10. **The futz and fiddle factor.** With the proliferation of computer games, computer usage can be wasteful of time. Experts have said that the most used program in Microsoft Windows is Solitaire!

Futzing and fiddling does have some benefits, however. It is a tool for learning how to use a mouse, for example, and it is a vehicle for exploring the system and becoming familiar with its capabilities. It is of value when done in moderation.

1. **May consume more screen space.**
2. **Hardware limitations.** Good design also requires hardware of adequate power, processing speed, screen resolution, and graphic capability. Insufficiencies in these areas can prevent a graphic system’s full potential from being realized
3. **Explain any four general principles of user interface design**

***Aesthetically Pleasing***

Provide visual appeal by following these presentation and graphic design principles: Create groupings; align screen elements and groups and use color and graphics effectively and simply.

***Compatibility***

Provide compatibility with the following: the user; the task and the job; the product.

*User compatibility.* Design must be appropriate and compatible with the needs of the user or client. Effective design starts with understanding the user’s needs and adopting the user’s point of view.

***Task and job compatibility.***

The organization of a system should match the tasks a person must do to perform the job.

***Product compatibility.***

The intended user of a new system is often the user of other systems or earlier versions of the new system. Habits, expectations, and a level of knowledge have been established and will be brought to bear when learning the new system. If these habits, expectations, and knowledge cannot be applied to the new system, confusion results and learning requirements are greatly increased.

***Consistency***

The system should display an appropriate level of consistency. For example, commands and menus should have the same format, etc.

***Control***

Control is feeling in charge, feeling that the system is responding to your actions. The user must control the interaction.

* Actions should result from explicit user requests.
* Actions should be performed quickly.
* Actions should be capable of interruption or termination.
* The user should never be interrupted for errors.

***Familiarity***

The interface should be based on user-oriented terms and concepts rather than computer concepts. For example, an office system should use concepts such as letters, documents, folders etc. rather than directories, file identifiers, etc.

***User diversity (Flexibility)***

A system must be sensitive to the differing needs of its users, enabling a level and type of performance based upon:

* Each user’s knowledge and skills.
* Each user’s experience.
* Each user’s personal preference.
* The physical conditions at that moment.

***Forgiveness***

The system should provide some resilience to user errors and allow the user to recover from errors. This might include undo facility, confirmation of destructive actions, deletes etc.

***Predictability (Minimal surprise)***

If a command operates in a known way, the user should be able to predict the operation of comparable commands

***Responsiveness***

* The system must rapidly respond to the user’s requests.
* Provide immediate acknowledgment for all user actions:
* Visual.
* Textual.
* Auditory.

A user request must be responded to quickly. Knowledge of results, or feedback, is a necessary learning ingredient.

***Simplicity***

Some of the ways to make an interface simple include:

* + - Provide defaults.
    - Provide uniformity and consistency.

***Transparency***

Permit the user to focus on the task or job, without concern for the mechanics of the interface. Workings and reminders of workings inside the computer should be invisible to the user.

***Trade-Offs***

Final design will be based on a series of trade-offs balancing often-conflicting design principles. People’s requirements always take precedence over technical requirements.

**Question 2**

1. Explain the difference between a variable and a constant

* Variables are areas allocated by the computer memory to hold data
* Constants are different from variables in the sense that their values do not change during the running of the program

1. Differentiate between the following declarations;
2. Dim password As String and

* Variable-length string

1. Dim password As String\*10

* Fixed-length string

1. Outline at least **four (4)** rules followed in naming valid variables in VB programming

* It must be less than 255 characters
* No spacing is allowed
* It must not  begin with a number
* Can begin with aleph abate letter
* Period is not permitted
* Special symbols such as #, ?, / are not allowed
* Under score is allowed

1. **With the help of illustration, differentiate between the following controls as used in the design of interfaces**
2. **Check box and radio buttons**

Check boxes are excellent controls for displaying true/false and yes/no values. However, check boxes work independently of one another. If you have five check boxes on a form, each one can be checked or unchecked—in any combination. Radio buttons, on the other hand, are mutually exclusive to the container on which they’re placed. This means that only one radio button per container can be selected at a time. Selecting one radio button automatically deselects any other radio buttons on the same container. Radio buttons are used to offer a selection of items when the user is allowed to select only one item.

1. **List Box and Combo Box**

**Displaying a List with the List Box**

A list box is used to present a list of items to a user. You can add items to, and remove items from, the list at any time with very little Visual Basic code. In addition, you can set up a list box so that a user can select only a single item or multiple items. When a list box contains more items than it can show because of the control’s size, a scrollbar appears automatically.

**Creating Drop-Down Lists Using the Combo Box**

List boxes are great, but they have two shortcomings. First, they take up quite a bit of space. Second, users can’t enter their own values; they have to select from the items in the list. If you need to conserve space, or if you want to enable a user to enter a value that might not exist in the list, use the Combo Box control.

1. Study the flow chart below and answer the following questions

Counter =

EndValue?

statement(s)

False

True

set

counter

to StartValue

increment

counter

1. Write the syntax of the above flow chart

**For *CounterVariable = StartValue To EndValue* [*Step*]**

***statement***

**Next [*CounterVariable*]**

1. State the difference between the above loop and the Do Until Loop

**For--- Next**

* The loop repeats as long as the counter variable is not greater than an end value

**A *Do Until* loop**

* Repeats as long as its test expression is false
* Ends when its test expression becomes true

1. Using any of the loops of your choice, write a program that prints the first odd numbers between 1 and 50

*Dim num1 As Integer*

*num1=1*

*ListBox1.items.Add(“old numbers”)*

*Do While num1< =50*

*ListBox1.items.Add(num1)*

*num=num+2*

*loop*

**Question:** Develop an application foraddition, subtraction and multiplication of two numbers.

Dim num1, num2, difference, product, quotient As Single

num1 = TextBox1.Text

num2 = TextBox2.Text

sum=num1+num2

difference=num1-num2

product = num1 \* num2

LblSum.Text=sum

LblDiff.Text=difference

LblPro.Text = product

End Sub