Date __/__/____

VISUAL BASICS .NET

Visual Basic .NET (VB.NET) is an object-oriented computer programming language implemented on the .NET Framework. Although it is an evolution of classic Visual Basic language, it is not backwards compatible with VB6, any code written in the old version does not compile under VB.NET.

Everything in VB.NET is an object, including all of the primitive types (short, Integer, Long, String, Boolean, etc.) and user-defined ~~function~~ types, events, and even assemblies. All object inherits from the base class object.

VB.NET is implemented by Microsoft's .NET framework. Therefore it has full access to all the libraries in the .NET Framework. It's also possible to run VB.NET programs on Mono, the open-source alternative to .NET, not only under Windows but even Linux or Mac OSX.

Date __/__/____

* The following reason make VB.NET a widely used professional language -

- Modern, general purpose.
- object oriented.
- component oriented.
- Easy to learn
- structured language
- It produces efficient programs.
- It can be compiled on a variety of computer platforms.
- Part of .NET Framework.

* Strong Programming Features VB.NET -

VB.Net has numerous strong programming features that make it ~~ender~~ endearing to multitude of programmers world wide. Let us mention some of these ~~feaut~~ features -

- Boolean conditions
- Automatic Garbage collection

Date __/__/____

- Standard Library
- Assembly Versioning
- Properties and Events
- Delegates and Events Management
- Easy-to-use Generics
- Indexers
- conditional compilation
- Simple Multithreading

Date __/__/____

1. Introduction to VB.NET

Event Driven Programming -

Event-driven programming is a programming paradigm in which the flow of program execution is determined by events - for example a user action such as a mouse click, key press, or a message from the operating system or another program.

An event-driven application is designed to detect events as they occur, and then deal with them using an appropriate event-handling procedure.

A visual programming IDE such as VB.NET provides much of the code for detecting events automatically when a new application is created.

.NET as better Programming Platform -

.NET is a free, cross-platform, open source developer platform for buiding many different types of applications.

with .NET, You can use multiple languages, editors and libraries to build for web, mobile, desktop, games and IoT.

Date __/__/____

You can write .NET apps in C#, F# or visual Basic.

.NET Framework — .NET Framework is a software development platform developed by Microsoft for building and running windows applications. The .Net framework consists of developer tools, programming languages, and libraries to build desktop and web applications. It is also used to build websites, web services and games.

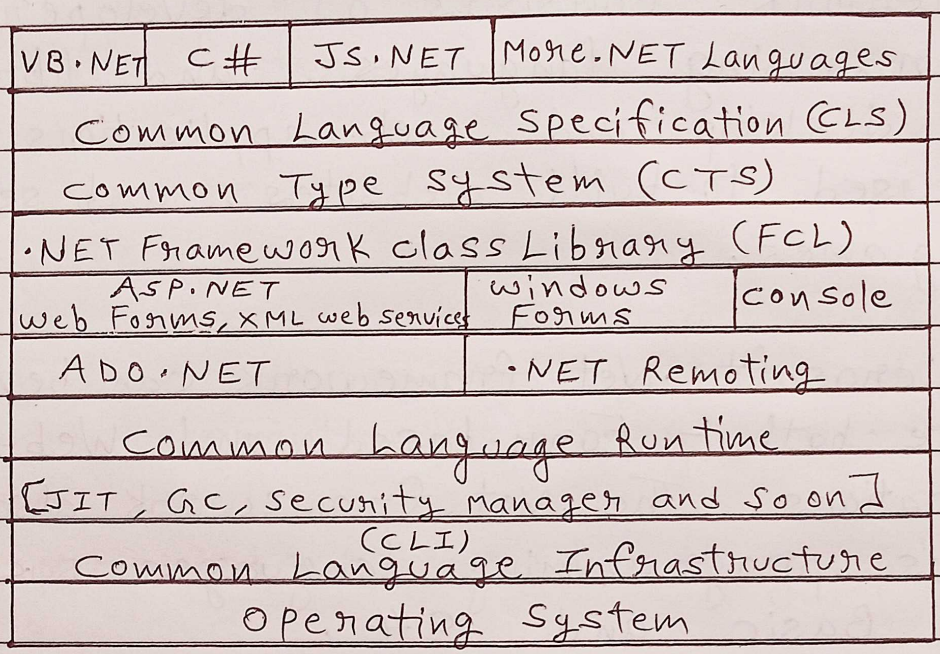
The microsoft .Net framework can be used to create both - Form-based and Web-based applications. The .Net framework also supports various programming languages such as Visual Basic and C#.

.NET Framework Architecture — .NET is tiered, modular and hierarchical. Each tier of the .Net framework is a layer of abstraction. .NET languages are the top tier and the most abstracted level. The common language run-time is the bottom tier, the least abstracted and closet to the native environment. This is important since the common language run-time works closely with the operating environment to manage .NET applications. The .NET Framework is partitioned into modules,

Date __/__/____

each with its own distinct responsibility. Finally, since higher tiers request services only from the lower tiers, .NET is hierarchical.

- The architecture layout of the .NET Framework is illustrated in figure -



- The .NET languages share :

→ The common language runtime.

→ The Framework class Library (FCL).

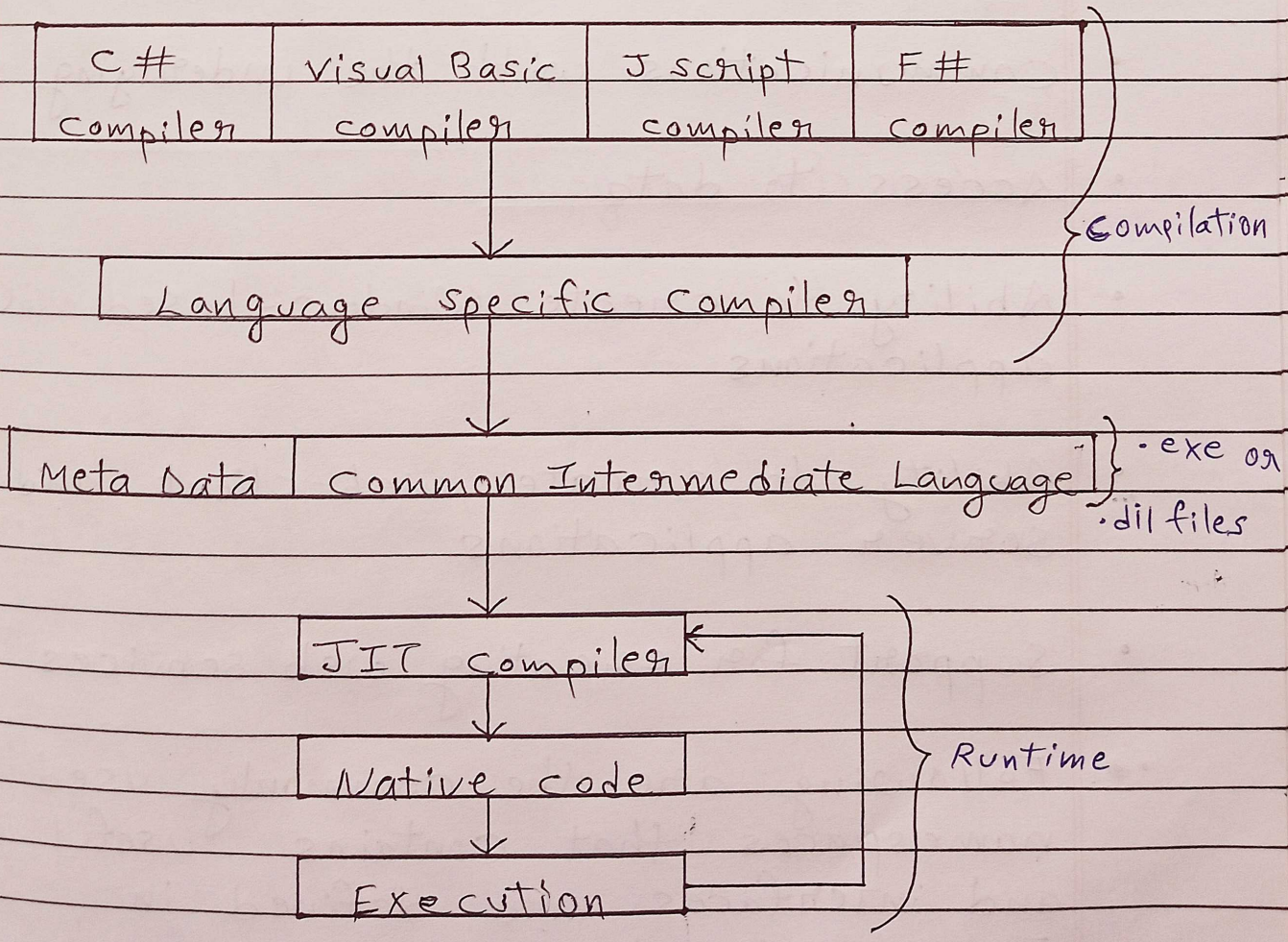
→ A common component model.

→ Common types system.

Date __/__/____

Just-In-Time compiler – JIT is a part of common Language Runtime (CLR) in .NET which is responsible for managing the execution of .NET programs regardless of any .NET programming language. A language specific compiler converts the source code to the intermediate language. This intermediate language is then converted into the machine code by the Just-In-Time compiler. This machine code is specific to the computer environment that the JIT compiler runs on.

Working of JIT compiler



Date __/__/____

.NET Framework class Library -

.NET Framework class Library is the collection of classes, namespaces, interfaces and value types that are used for .NET applications.

It contains thousand of classes that supports the following functions.

- Base and user-defined data types.
- Support for exceptions handling.
- input/output and stream operations.
- communications with the underlying system
- Access to data
- Ability to create Window-based GUI applications
- Ability to create web-client and server applications.
- Support for creating web services
- Following are the commonly used namespaces that contains useful classes and interfaces and defined in Framework class Library.

Date __/__/____

System, System.Data, System.IO, System.Diagonostics, System.Net, System.Windows.Forms, System.Web, System.Security, System.xml

Common Language Runtime (CLR) -

The common language runtime (CLR) is the programming (virtual Machine Component) that manages the execution of programs written in any language that uses the .NET Framework.

It converts code into native code which further can be executed by the CPU.

* CLR Functions -

- It converts the program into native code.
- Handles Exceptions.
- Provides type-safety
- Memory management
- Provides security
- Improved Performance
- Language independent

Page No. ____

Date __/__/____

- Platform independent
- Garbage collection
- Provides language features such as inheritance, interfaces and overloading for object-oriented programmings.

Common type system –

The common type system defines how types are declared, used and managed in the common language runtime and is also an important part of the runtime's support for cross-language integration.

The common type system performs the following function:

- Establishes a framework that helps enable cross-language integration, type-safety and high-performance code execution.
- Provides an object-oriented model that supports the complete implementation of many programming languages.
- Defines rules that languages must follow, which helps ensure that objects written in different languages can interact with

Date __/__/____

each other.

- Provides a library that contains the primitive data types (such as Boolean, Byte, char, Int32, UInt64) used in application development.

Managed code - Managed code is a code whose execution is managed by common language Runtime. It gets the managed code and compiles it into machine code. After that, the code is executed. The runtime here i.e CLR provides automatic memory management, type safety, etc.

Managed code is written in high-level languages run on top of .NET. This can be C#, F#, etc.

* C, C++ code called "unmanaged code" do not have that privilege.

Microsoft Intermediate Language (MSIL) -

A .NET programming language (C#, J#, VB.NET) does not compile into executable code; instead it compiles into an intermediate code called MSIL. As a programmer one need not worry about the syntax of MSIL - since our source code is automatically converted to MSIL. The

Date __/__/____

MSIL code is then send to the CLR (common language runtime) that converts the code to machine language which is then run on the host machine.

MSIL is similar to Java Byte code.

9.) WAP in VB.NET that year is leap year or not?

```
Module Program
    Sub Main(args As String())
        Dim y As Integer
        Console.WriteLine("Enter Year")
        y = Console.ReadLine()
        If (y Mod 4 = 0 And y Mod 100 <> 0
            Or y Mod 400 = 0)
        Then
        Console.WriteLine("Leap Year")
        Else
        Console.WriteLine("Not a leap year")
        End If
    End Sub
End Module
```

Date __/__/____

2. Elements of User Interface

Windows Forms – Visual Basic Form is a container for all the controls that make up the user interface. Every window you see in a running visual basic application is a form, thus the terms form and window describe the same entity. Visual Studio creates a default form for you when you create a Windows Forms Application.

Every form will have title bar on which the form's caption is displayed and there will be buttons to close, maximize and minimize.

- Microsoft visual studio → File → New → Project

Form 1 ← OK ← Windows Forms Applications

* Form Properties –

AcceptButton, CancelButton, Autoscale, AutoScroll, AutoScrollMinSize, AutoScrollPosition, BackColor, BorderStyle, ControlBox, Enabled, Font, Height, HelpButton, MinimizeBox, MaximizeBox, MinimumSize, Name, StartPosition, Text, Top, Left, TopMost, width

Date __/__/____

TextBoxes – Text box Controls allow entering text on a form at runtime. By default, it takes a single line of text, however, you can make it accept multiple texts and even add scroll bars to it.

Buttons – The Button control represents a standard Windows button. It is generally used to generate a click event by providing a handler for the click event.

Labels – The Label control represents a standard Windows label. It is generally used to display some informative text on the GUI which is not changed during runtime.

checkBoxes – The CheckBox Control allows the user to set true/false or yes/no type options. The user can select or deselect it. When a checkbox is selected it has the value True, and when it is cleared, it holds the value False.

Radio Buttons – The RadioButton Control is used to provide a set of mutually exclusive options. The user can select one radio button in a group. If you need to place more than one group of radio buttons in the same form, you

Date __/__/____

should place them in different container controls like a GroupBox control.

List Boxes – The ListBox represents a Windows control to display a list of items to a user. A user can select an item from the list. It allows the programmer to add items at design time by using the properties window or at the runtime.

Combo Boxes – The ComboBox Control is used to display a drop-down list of various items. It is a combination of a text box in which the user enters an item and a drop-down list from which the user selects an item.

Picture Boxes – The PictureBox Control is used for displaying images on the form. The image property of the control allows you to set an image both at design time or at runtime.

ScrollBars – The ScrollBars controls display vertical and horizontal scroll bars on the form. This is used for navigation through large amount of information. There are two types of scroll bars controls : HscrollBar for horizontal scroll bars and VscrollBar for vertical scroll bars. These are used independently from each other.

Date __/__/____

Splitters – Splitter controls are used to resize docked controls at run time. The splitter control is often used on forms with controls that have varying lengths of data to present, like Windows Explorer, whose data panes contain information of varying widths at different times.

Timer – The timer control is a looping control used to repeat any task in a given time interval. It is an important control used in client-side and server-side programming, also in windows services.

Menus – A menu is used as menu bar in the Windows form that contains a list of related commands, and it is implemented through MenuStrip Control. The menu items are created with ToolStripMenuItem objects.

Built-in-Dialogs – There are many built-in dialog boxes to be used in Windows forms for various tasks like opening and saving files, printing a page, providing choices for colors, fonts, page setup, etc, to the user of an application. These built-in dialog boxes reduce the developer's time and workload. All of these dialog box control classes inherit from the CommonDialog Class.

Page No.

Date __/__/____

Image List - The ImageList is a simple control that stores images used by other controls at runtime.

~~Trees~~ Tree Views - The TreeView control is used to display hierarchical representations of ~~data~~ items similar to the ways the files and folders are displayed in the left pane of the Windows Explorer. Each node may contain one or more child nodes.

List Views - The ListView control displays a list of items along with icons. The Item property of the ListView control allows you to add and remove items from it. The SelectedItem property contains a collection of the selected items. The MultiSelect property allows you to set select more than one item in the list view.

Toolbars - A Toolbar Control is a combination of Toolbar buttons where each button represents a function. A Toolbar button can display an image, text or a combination of both. The button click event handler is responsible for executing some code.

Date __/__/____

Status Bar – The Status Bar provides a location for the application developer to convey information to the user about what is happening in an application.

Progress Bar – The Progress Bar control is used to provide visual feedback to your users about the status of some task. It shows a bar that fills in from left to right as the operation progresses. The main properties of a progress bar are Value, Maximum and Minimum.

Q.) WAP in VB.Net to find greater no ?

```
Module Program
    Sub Main (args As String ())
        Dim x, y As Integer
        Console.WriteLine ("Enter x:")
        x = Console.ReadLine()
        Console.WriteLine ("Enter y:")
        y = Console.ReadLine ()
        If (x > y) Then
            Console.WriteLine ("x is greater")
        Else
            Console.WriteLine ("y is greater")
        End If
    End Sub
End Module
```

Date __/__/____

3. Mastering VB Language

Data Types – Data types refer to an extensive system used for declaring variables or functions of different types. The type of a variable determines how much space it occupies in storage and how the bit pattern stored is interpreted.

Boolean – Depends on implementing platform
Byte – 1 byte
Char – 2 bytes
Date – 8 bytes
Decimal – 16 bytes
Double – 8 bytes
Integer – 4 bytes
Long – 8 bytes
Object – 4 bytes (32-bit) / 8 bytes (64-bit)
SByte – 1 byte
Short – 2 bytes
Single – 4 bytes
String – Depends on implementing platform
UInteger – 4 bytes
ULong – 8 bytes
User Defined – Depends on implementing platform
Ushort – 2 bytes

Operators – An operator is a symbol that tells the compiler to perform specific mathematical or logical manipulations.

Date __/__/____

→ Following types of commonly used operators-

- Arithmetic Operators
- Comparison operators
- Logical / Bitwise Operators
- Bit shift Operators
- Assignment Operators
- Miscellaneous Operators

Conditionals – There are four conditional statements :

- If statement – executes a set of code when a condition is true.
- If...Then...Else statement – Select one of two sets of lines to execute.
- If...Then...ElseIf statement – Select one of many sets of lines to execute.
- Select Case statement – Select one of many sets of lines to execute.

Loops – A loop is used to repeat the same process multiple times until it meets the specified condition in a program. A loop is also used to reduce the program complexity, easy to understand and easy to debug.

Page No.

Date __/__/____

* Types of Loops :-

1.) Do While Loop
2.) For Next Loop
3.) For Each Loop
4.) While End Loop
5.) With End Loop

1.) Do while Loop -

```
Module loops
    Sub Main()
        'local variable definition
        Dim a As Integer = 10
        'do loop execution
        Do
            Console.WriteLine("value of a: {0}", a)
            a = a + 1
        Loop While (a < 20)
        Console.ReadLine()
    End Sub
End Module
```

⇒ Output :

```
Value of a: 10
:
:
:
:
:
Value of a: 19
```

Date __/__/____

2.) For Next Loop -

```
Module loops
    Sub Main()
        Dim a As Byte
        ' for loop execution
        For a = 10 To 20
            Console.WriteLine("value of a: {0}", a)
        Next
        Console.ReadLine()
    End Sub
End Module
```

=> output :

```
value of a: 10
:
:
value of a: 20
```

3.) For Each Loop -

```
Module loops
    Sub Main()
        Dim anArray() As Integer = {1, 3, 5, 7, 9}
        Dim arrayItem As Integer
        'displaying the values
        For Each arrayItem In anArray
            Console.WriteLine(arrayItem)
        Next
        Console.ReadLine()
    End Sub
End Module
```

Page No.

Date __/__/____

=> output:

```
1
3
5
7
9
```

a)

4.) While End Loop -

```
Module loops
    Sub Main()
        Dim a As Integer = 10
        ' While loop execution '
        While a < 20
            Console.WriteLine("value of a: {0}", a)
            a = a + 1
        End While
        Console.ReadLine()
    End Sub
End Module
```

=> output:

```
value of a: 10
:
:
:
:
value of a: 19
```

Date __/__/____

5.) With End Loop – It is not exactly a looping construct. It executes a series of statements that repeatedly refers to a single object or structure.

Date __/__/____

Procedures – A procedure is a block of Visual Basic statements (Function, Sub, Operator, Get, Set) and a matching End declaration. All executable statements in Visual Basic must be within some procedure.

4. OOP in VB.NET

class and object –

```
Module mybox
   Class Box
      Public length As Double
      Public breadth As Double
      Public height As Double
   End Class
   Sub Main()
      Dim Box1 As Box = New Box()
      Dim Box2 As Box = New Box()
      Dim volume As Double = 0.0

      Box1.height = 5.0
      Box1.length = 6.0
      Box1.breadth = 7.0

      Box2.height = 10.0
      Box2.length = 12.0
      Box2.breadth = 13.0
```

Date __/__/____

```
Volume = Box1.height * Box1.length * Box1.breadth
Console.WriteLine("Volume of Box1: {0}",
                                    Volume)

Volume = Box2.height * Box2.length * Box2.breadth
Console.WriteLine("Volume of Box2: {0}",
                                    Volume)
    Console.Readkey()
  End Sub
End Module
```

Output:
Volume of Box1 : 210
Volume of Box2 : 1560

Properties – A property is a value or characteristic held by a visual Basic object, such as caption or Fore color.
Properties can be set at design time by using the Properties window or at run time by using statements in the program code.

```
Object.Property = value
```

where,
- object is the name of the object you're customizing.
- Property is the characteristic you want to change.
- value is the new property setting.

eg: Form1.Caption = "Hello"

Page No.

Date __/__/____

Methods - A method is a procedure created as a member of a class and they cause an object to do something. Methods are used to access or manipulate the characteristics of an object or a variable.

~~For example, the MessageBox control has a method name Show, below is code.~~

~~Public Class Form1~~

```
Sub Main()

    hello()
End Sub
Private
```

```
Module vishal
Sub Main()
    hello()
End Sub
Private Sub hello()
    Console.WriteLine("hello vishal")
    Console.ReadLine()
End Sub
End Module
```

=> output:

hello vishal

Date __/__/____

Events — Events are basically a user action like key press, clicks, mouse movements, etc, or some occurence like System generated notifications. Applications need to respond to events when they occur.

Constructors — A class constructor is a special member Sub of a class that is executed whenever we create new objects of that class. A Constructor has the name New and it does not have any return type.

```
Class Line
    Private l As Double
    Public Sub New()
        Console.WriteLine("object is being created")
    End Sub
    Public Sub setLength(Byval len As Double)
        l = len
    End Sub
    Public Function getLength() As Double
        Return l
    End Function
    Shared Sub Main()
        Dim line As Line = New Line()
        line.setLength(6.0)
        Console.WriteLine("Length of line : {0}",
                          line.getLength())
        Console.Readkey()
    End Sub
End Class
```

Date __/__/____

=> output:

```
Object is being created
Length of line : 6
```

* Parameterized Constructors –

```
Class Line
    Private length As Double
    Public Sub New (ByVal len As Double)
        Console.WriteLine ("object is being created,
                            length = {0}", len)
        length = len
    End Sub
    Public Function getLength () As Double
        Return length
    End Function
    Shared Sub Main ()
        Dim line As Line = New Line (10.0)
        Console.WriteLine ("Length of ~~the~~ line
            set by constructor : {0}", line.getLength())
        line.setLength (6.0)
        Console.WriteLine ("Length of line set by
            setLength : {0}", line.getLength ())
        Console.ReadKey ()
    End Sub
End Class
```

=> output:

```
Object is being created, length = 10
Length of line set by constructor : 10
Length of line set by setLength : 6
```

Page No.

Date __/__/____

Destructor — A destructor is a special member Sub of a class that is executed whenever an object of its class goes out of scope.

A destructor has the name Finalize and it can neither return a value nor can it take any parameters.

```
Class Line
   Private length As Double
   Public Sub New()
      Console.WriteLine("object is being
                         created")
   End Sub
   Protected Overrides Sub Finalize()
      Console.WriteLine("object is being
                         deleted")
   End Sub
   Public Sub SetLength(ByVal len As Double)
      length = len
   End Sub
   Public Function getLength() As Double
      Return length
   End Function
   Shared Sub Main()
      Dim line As Line = New Line()
      line.SetLength(6.0)
      Console.WriteLine("Length of line : {0}",
                        line.getLength())
      Console.Readkey()
   End Sub
End Class
```

Page No.

Date __/__/____

⇒ output:

```
object is being created
Length of line : 6
object is being deleted
```

Method overloading - (Polymorphism)

```
Module Vishal
    Class Sample
        Public Sub Add(ByVal num1 As Integer,
                       ByVal num2 As Integer)
            Dim sum As Integer = 0
            sum = num1 + num2
            Console.WriteLine("Sum is : {0}", sum)
        End Sub
        Public Sub Add(ByVal num1 As Integer,
          ByVal num2 As Integer, ByVal num3
                                  As Integer)
            Dim Sum As Integer = 0
            sum = num1 + num2 + num3
            Console.WriteLine("Sum is : {0}", sum)
        End Sub
    End Class
    Sub Main()
        Dim obj As New Sample()
        obj.Add(10, 20)
        obj.Add(10, 20, 30)
    End Sub
End Module
```

⇒ Output:

```
Sum is : 30
Sum is : 60
```

Page No. ☐

Date __/__/____

Method overriding - (Inheritance)

```
Module vishal
    Class Sample1
        Overridable Sub Fun()
            Console.WriteLine("Sample1 called")
        End Sub
    End Class

    Class Sample2
        Inherits Sample1
        Overrides Sub Fun()
            Console.WriteLine("Sample2 called")
        End Sub
    End Class

    Sub Main()
        Dim S2 As New Sample2()
        S2.Fun()
    End Sub
End Module
```

⇒ Output :

```
Sample2 called
```

~~Interfaces~~ →

~~Module vishal~~

Date __/__/____

Interfaces –

```
Module Ravindra
    Interface Antanu
        Sub GetDetails(ByVal x As String)
    End Interface
    Class Vishal
        Implements Antanu
        Public Sub GetDetails(ByVal a As String)
                Implements Antanu.GetDetails
            Console.WriteLine("Name : {0}", a)
        End Sub
    End Class
    Class Vishalji
        Implements Antanu
        Public Sub GetDetails(ByVal a As String)
                Implements Antanu.GetDetails
            Console.WriteLine("Location : {0}", a)
        End Sub
    End Class
    Sub Main(ByVal args As String())
        Dim u As Antanu = New Vishal()
        u.GetDetails("Vishal Kumar")
        Dim u1 As Antanu = New Vishalji()
        u1.GetDetails("Ranchi")
        Console.ReadLine()
    End Sub
End Module
```

⇒ Output :

```
Name : Vishal Kumar
Location : Ranchi
```

Page No

Date __/__/____

5. Exception Handling

* An exception is a problem that arises during the execution of a program. An exception is a response to an exceptional circumstance that arises while a program is running, such as an attempt to divide by zero.

* Exceptions provide a way to transfer control from one part of a program to another. VB.Net exception handling is built upon four keywords - Try, Catch, Finally and Throw.

* In the .Net Framework, exceptions are represented by classes.

* Following are the Exception Class -

System.IndexOutOfRangeException
System.IO.IOException
System.ArrayTypeMismatchException
System.NullReferenceException
System.DivideByZeroException
System.InvalidCastException
System.OutOfMemoryException
System.StackOverflowException

Date __/__/____

Q) System.DivideByZeroException

```
Module Program
    Sub division (ByVal num1 As Integer,
                  ByVal num2 As Integer)
        Dim result As Integer
        Try
            result = num1 \ num2
        Catch e As DivideByZeroException
            Console.WriteLine ("Exception caught:
                               {0}", e)
        Finally
            Console.WriteLine ("Result : {0}", result)
        End Try
    End Sub
    Sub Main ()
        division (25, 0)
        Console.ReadKey()
    End Sub
End Module
```

=> Output:

```
Exception Caught: System.DivideByZeroException:
Attempted to divide by zero.

Result: 0
```

Date __/__/____

Throwing Objects -

You can throw an object if it is either directly or indirectly derived from the System.Exception class.

```
Module Vishal
    Sub Main()
        Try
            Throw New ApplicationException("A
                Custom exception is being thrown...")
        Catch e As Exception
            Console.WriteLine(e.Message)
        Finally
            Console.WriteLine("Now Inside the
                                Finally Block")
        End Try
        Console.Readkey()
    End Sub
End Module
```

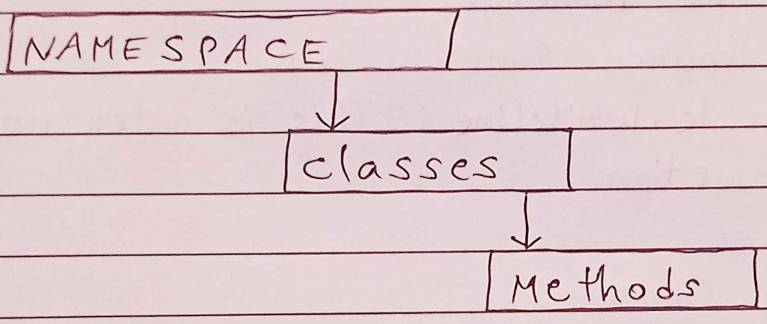
=> Output :

```
A Custom exception is being thrown...
Now Inside the Finally Block
```

Date __/__/____

6. Name Spaces

NameSpaces allow us to organize Classes so that they can be easily accessed in other applications. Name Spaces enable us to avoid any naming conflicts between classes that have the same name. We can use two classes with the same name in an application provided they belong to different namespaces. (name scope)



* Some of the namespaces and the actual files they are in for a Windows Forms Application are:

→ System > in System.dll
→ System.Data > in System.Data.dll
→ System.Deployment >
→ System.Deployment.dll
→ System.Drawing > System.Drawing.dll
→ System.Drawing > System.Drawing.dll
→ System.Windows.Forms
→ System.Windows.Forms.dll

Date __/__/____

```
* Namespace MyColor
  Public Class Color
   Sub Color()
   ' Do Something
  End Sub
  End Class
  End Namespace
```

```
# Imports System
  Namespace outer
    Public Class nameout
      Public Shared Function disp()
        Console.WriteLine("This is outer namespace")
      End Function
    End Class
    Namespace inner
      Public Class namein
        Public Shared Function disp()
          Console.WriteLine("This is inner namespace")
        End Function
      End Class
    End Namespace
  End Namespace
  Module module1
    Sub main()
      Console.clear()
      Outer.nameout.disp()
      outer.inner.namein.disp()
    End Sub
  End Module
```

Output:

```
This is outer namespace
This is inner namespace
```

Page No.

Date __/__/____

7. Database in VB.NET

Applications communicate with a database, firstly, to retrieve the data stored there and present it in a user-friendly way and secondly, to update the database by inserting, modifying and deleting data.

- Microsoft ActiveX Data Objects.Net (ADO.Net) is a model, a part of the .Net framework that is used by the .Net applications for retrieving, accessing and updating data.

- ADO.Net object model is nothing but the structured process flow through various components.
  - → Data Provider
  - → Data Set
  - → Data Reader
  - → Data Store

Connecting to a Database –

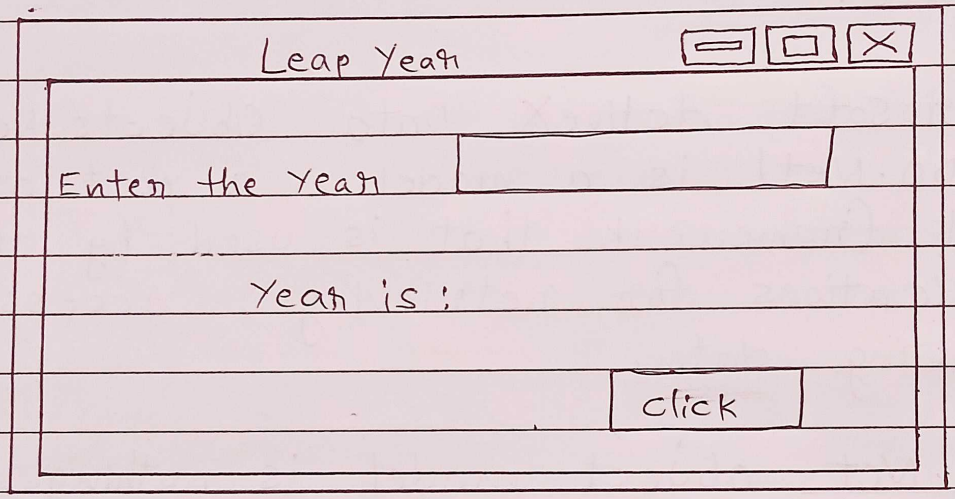
The .Net framework provides two types of Connection classes.

- SqlConnection – designed for connecting to Microsoft SQL Server.

Date __/__/____

- OleDb Connection – designed for connecting to a wide range of database, like Microsoft Access and Oracle.

9. WAP in VB.net that year is leap year or not using windows form?




```
Public Class Form1
    Private Sub Button1_Click(sender As Object,
        e As EventArgs) Handles Button1.Click
    Dim y As Integer
    y = Val(TextBox1.Text)
    If (TextBox1.Text = "") Then
        MsgBox ("Please enter value")
        Exit Sub
    End If
    If(y Mod 4 = 0 And y Mod 100 <> 0 Or
        y Mod 400 = 0) Then
        Label3.Text = " Leap Year"
    Else
        Label3.Text = " NOT a Leap year"
```

Page No.

Date __/__/____

```
        End If
    End Sub
End Class
```