Name of Experiment Installation of VS Date 21-08-2023

Experiment No. 1 Experiment Result

Objective → To show Installation Process of VS

Steps to install Visual Studio in PC :-

Step-1 → Download Visual Studio Installer from site → https://visualstudio.microsoft.com

Step-2 → Run "vs-installer.exe"

Step-3 → Choose ".NET ~~desktop~~ developement" for C# Programming as workload.

Step-4 → We can choose other indivisual components for customize installation

Step-5 → Choose installation location for VS

Step-6 → Click on 'Install' button. This might take some time as it downloads and installs selected components.

Step-7 → After installation completion, click on 'Launch' button. There will be a prompt from microsoft for you to sign up in VS and after signing-up you can use Visual Studio by agreeing to Terms & Conditions.

Teacher's Signature : _____

C# program to print 'Hello World'.

```
using System;
namespace HelloWorld
{
    class Program
    {
        static void Main (string[] args)
        {
            Console.WriteLine("Hello World");
        }
    }
}
```

Name of Experiment C# Program Date 27-08-2023

Experiment No. 2 Experiment Result

Page No. 3

Objective → To define Classes, Libraries & Functions in C# program.

System → System is the foundation namespace containing essential classes like 'Object', 'String' and 'Console'. It organizes core functionalities and prevents naming clashes, simplifying coding.

NameSpace → In C# language, namespace keyword is used to create a container that organizes related code elements and prevents naming clashes. It helps structure your code and keeps things organized and manageable.

using → using key word is used to use a namespace or library created previously or inbuilt library.

class → In C#, class keyword is used to create a class, which is a blueprint for creating objects, defining their properties or attributes and actions.

Main → Main refers to the entry point method of a program. The 'Main'

Teacher's Signature :

Name of Experiment..... Date.....

Experiment No..... Experiment Result.....

method is where program execution starts

static → 'static' in 'Main' function means the method can be called directly on the class without creating an object. Its the program's starting point.

void → void means method doesn't return a value

Console → Console is a class in System namespace which handles input & output operations in command-line environment.

WriteLine → This is a method of Console class which can be used using (.) dot operator to give output to display.

{} → Curly braces defines the scope of code or forms a block of code to be executed together.

Teacher's Signature :

1. as →

```
int a = 12;
if ( a < 18 )
{
  Console.WriteLine(" Ok");
}
```

output → Ok

2. as →

```
int a = 12;
if (a < 10)
{ Console.WriteLine(" Ok");
}
else
{ Console.WriteLine (" not Ok");
}
```

output → not Ok

3 as →

```
int a = 12;
if ( a < 10) { Console.Write(" no");
else if (a == 12) { Console.Write(" Ok");
else { Console.Write (" Maybe"); }
```

Output → Ok.

Name of Experiment: Conditional Statements Date: 21-08-2023

Experiment No: 3 Experiment Result: Page No. 5

Objective → To study Conditional Statements in C# Language.

1- if Statement → if given condition is satisfied then code gets executed.

Syntax →
```
if (condition)
{ // block of code }
```

2- if-else → If given condition is satisfied then code gets executed (within if block code) otherwise else block code gets executed.

Syntax →
```
if (condition)
{ // code1 }
else
{ // code2 }
```

3- else if → if there are more than one condition to put and also different type of code related to those conditions to be executed then "else if" is used.

Syntax →
```
if (condition1) { //code1 }
else if (condition2) { //code2 }
else (condition3) { //code3 }
```

Teacher's Signature : _____

4.) ans →

```
int j = 4;
switch (j)
{
    case 1:
        Console.Write("One");
        break;

    case 2:
        Console.Write("Two");
        break;

    case 3:
        Console.Write("Three");
        break;

    case 4:
        Console.Write("Four");
        break;

    default:
        Console.Write("None");
}
```

Output → Four

Name of Experiment..... Date.....

Experiment No..... Experiment Result..... Page No. 6

4- Switch → Switch provides us facility to choose which code to be executed given ~~case~~ ~~the~~ which choice case select.

Syntax :-

```
switch (expression)
{
    case #1 :
        // code1 ; break;
    case 2 :
        // code2 ; break;
    case 3 :
        // code 3 ; break;

    default :
        // code4; break;

}
```

In switch-case code is executed continuously when choice is made so we use break keyword to end the execution or get out the switch block.

default → default keyword is written to specify ~~the~~ to be executed code when no choice is matched.

Teacher's Signature : _____

5- ans ->

```
int time = 20;
string result = (time < 18) ? "day" : "Night";
Console.Write(result);
```

Output -> day.

Name of Experiment..... Date.....

Experiment No..... Experiment Result..... Page No. 7

5- Ternary Operator :→ It is shortform of If-else block.

Syntax →

(condition) ? code1 : code2 ;

If condition is true code1 gets executed and If false then code2 gets executed

Teacher's Signature : _____

1- ans →

```
int i = 1;
while (i < 5)
{
    Console.Write(i);
    i++;
}
```

Output → 1234

2- ans →

```
for (int i = 1; i < 5; i++)
{ Console.Write(i);
}
```

Output → 1234

3- ans →

```
string[] cars = {"V", "B", "F"};
foreach (string i in cars)
{ Console.WriteLine(i);
}
```

Output →

V
B
F

Name of Experiment Loops Date 21-08-2023

Experiment No. 4 Experiment Result Page No. 8

Objective To study various types of loops used in C# Language.

1- While Loop → code gets executed till condition is true.

Syntax → While (condition)
{
// code to be executed.
}

2- for Loop → Unlike while loop initialization and increment or decrement of variable meant to be used in condition can be done in condition section in for loop →

Syntax → for (statement1 ; statement2; statement3)
{ // code
}

3- foreach Loop → This is used to loop through elements in an array.

Syntax → foreach (type variable in arrayname)
{
// code.
}

Teacher's Signature :

```
Using System;
namespace MyPrivateAssembly
{
public class Privateclass
{
private void PrivateMethod()
{
Console.WriteLine("This is a Private Method");
}
public void PublicMethod()
{
Console.Writeline("This is a public Method");
PrivateMethod();
}}}
```

After building this class library Project :-

```
Using System;
Using MyPrivateAssembly;
namespace ConsumeProject
{
class Program
{
static void Main(string[] arr)
{
Privateclass privateInstance = new Privateclass();
PrivateInstance.PublicMethod();
}
}
}
```

Name of Experiment: Assembly & its type Date:

Experiment No: 5 Experiment Result: Page No. 9

Objective :- Assembly & its type

Assembly :- It is a collection of code files that are compiled into an executable or a dynamic link library (DDL) Depending on their location and intended use

1. Implemented as .exe or .dll.
2. Loaded into memory when needed.
3. Info about assembly can be reflected.
4. Can share assemblies b/w GAC

Assemblies can be divided into Many :-

1) Private Assembly :- It is used only by a single app & usually it is stored in that app install directory.

Some steps :-

1) Create a class Library Project.
2) Write your c# code with the class files of the project.
3) Build the project.
4) Access Control Using Modifiers.
5) Use the assembly in Another Project.
6) Assending the private assembly.

// class library ex

Teacher's Signature : _____

Ex1

```
namespace SharedAssembly
{
public class Calculator
{
public int Add (int a, int b)
{
return a+b;
}}}.
```

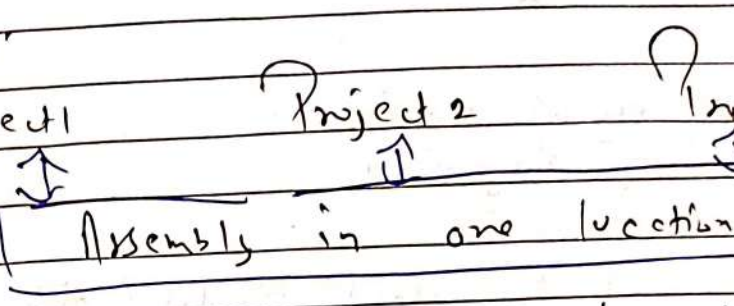
→ Generate a DLL file.

```
using System;
using SharedAssembly;
namespace AnotherProject
{
class Program
{
static void Main (string[] args)
{
Calculator calc = new Calculator();
int result = calc.Add(5,10);
Console.WriteLine("Result:" + result);
}}}
```

Name of Experiment..... Date.....

Experiment No..... Experiment Result.....

2. Shared Assembly: A Shared assembly is one that can be referenced by More than one app^n. If Mul app^ need to access an assembly.



no need to copy assembly in bin folder

Steps to Create Shared Assembly:

1. Create a class library.
2. Write code in library in cludes - class, Met
3. Build class library Project.
4. Ref the DLL in other projects.
5. Access Shared functionality.

Add a ref. to "Shared Assembly.dll" in the "Another project". you can now use the 'calculator' class from the Shared assembly in this project.

Teacher's Signature : _____

Ex1

```
Using System;
{
public delegate String Mydel (string str);
 class Program
 {
  event Mydel MyEvent;
   public Program()
 {
   this.MyEvent = new Mydel (this.WelcomeUser);
  }
public String WelcomeUser (String Username)
 {
  return "welcome to" + Username;
  }
```

Name of Experiment: Event Date:

Experiment No: 6 Experiment Result:

Page No. 11

Program 6

Objective / V.S

Introduction: Event

Event enables a class or obj. to notify other classes or obj. when something of interest occurs. The class that sends the event called Publisher.

Publisher: class that sends the event

Subscriber: class that receive the event.

An event can have Multiple Subscribe. If no event raise then no Subscribe.

In the .net class library event is based on the Event handler delegate & the EventArgs base class.

Ex:

```
Using System;
class EventPublisher {
public event EventHandler SomethingHappened;
public void DoSomething()
{
Console.WriteLine("Doing Something...");
if (SomethingHappened != null)
{
SomethingHappened(this, EventArgs.Empty);
}
}
```

Teacher's Signature :

Name of Experiment..... Date.....

Experiment No..... Experiment Result..... Page No. 13

```
class Program
{
static void main()
{
EventPublisher publisher = new EventPublisher();
EventSubscriber subscriber = new EventSubscriber();
publisher.SomethingHappened += subscriber.OnSomethingHappened;
publisher.DoSomething();
}
}
```

Teacher's Signature : _____

Name of Experiment: Window based calculator Date:

Experiment No: 7 Experiment Result:

Program

Introduction:- Window based Calculator.

```
Using System;
Using System.windows.Forms;
namespace CalculatorApp
{
public partial class Calculator form: Form
{
    double Num1;
    String operation;
    public Cal()
    {
    Initialize Componet();
    }
Private void NumberButtonClick(obj: Sender, EventArgs e)
    {
    Button button = (Button) Sender;
if (textBoxDisplay.Text == "0" || button.Text == "0")
{
    textBoxDisplay.clear();
}
    textBoxDisplay.Text += button.Text;
    }
Privite void OperationButtonClick(obj Sender, EventArgs e)
    {
    Button button = (Button) Sender;
if (!String.IsNULLOrEmpty(textBoxDisplay.Text))
    {
    firstNumber = double.Parse(textBoxDisplay.Text);
```

Teacher's Signature :

```
ext}
ext = "0";

Button(lick( obj) Sender, EventArgs e)


;
S NULL OR EMPTY ( text Box Display.Text);
)


.1 + num2;


.1 - num2;

: ;
num1 * num2)

.4 ;
```

```
operation = button.Text;
TextBoxDisplay.Text = "0";
} }

private void EqualButtonClick(obj sender, EventArgs e)
{
    double num2;
    double result;
if (!string.IsNullOrEmpty(textBoxDisplay.Text))
switch (operation)
{
    case "+":
    result = num1 + num2;
      break;
    case "-";
    result = num1 - num2;
      break;
    case "*";
     result = num1 * num2;
       break;
    case "/";
        result = num1 / num2;
          break;
if (num2 != 0)
 result = num1 / num2;
 else
{ MsgBox.Show("cannot divided by zero");
    return;
}     break;
      default:
      return;
}
```

```
textBoxDisplay.Text = result.ToString();
}}
private void clearButton_OnClick(obj sender, EventArgs e)
{
    textBoxDisplay.Text = "0";
}
private void ClearEntryButton_Click(obj sender, EventArgs e)
{
    textBoxDisplay.Text = "0";
}}
```

Teacher's Signature : _____

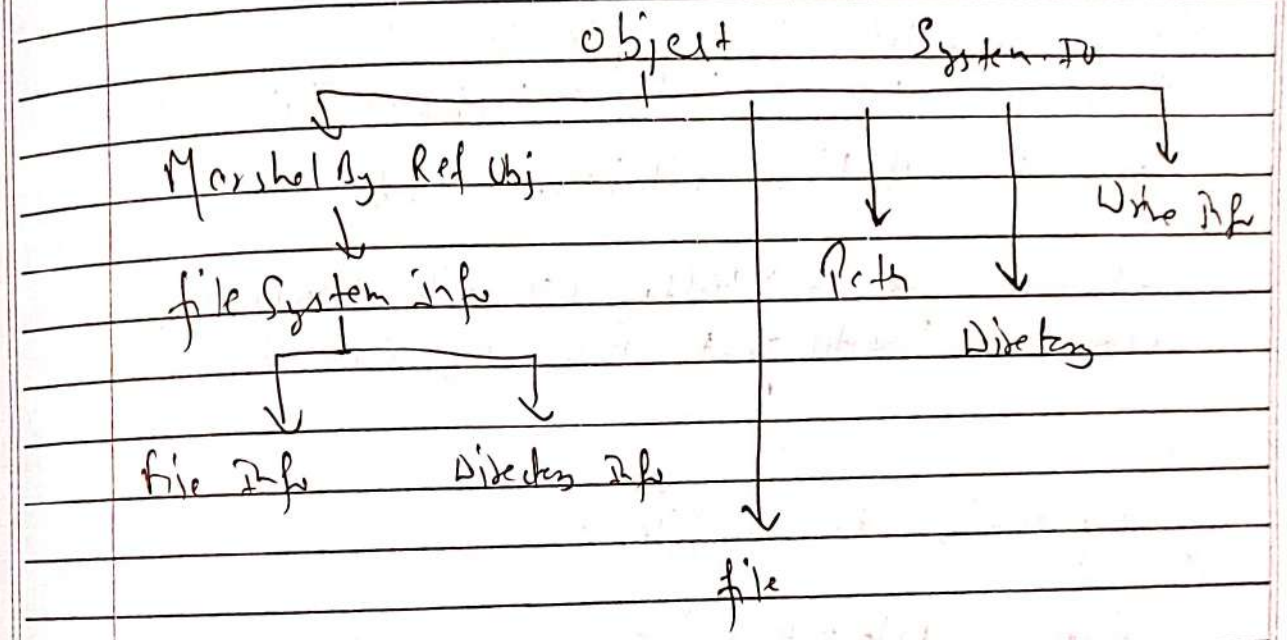
Name of Experiment: File handling Date:

Experiment No: 8 Experiment Result:

Program 8

Intro: File Handling

The term file Handling in c# refers to the various operation that we can perform on a file such as Creating a file, reading data from the file.



1) Reads from a file.

```
Using System;
Using System.IO;
Class Program
{
  Static void Main()
{
  String filePath = "example.txt";
  String[] lines = File.ReadAllLines(filePath);
foreach (String line in lines)
{
```

Teacher's Signature :

```
Console.WriteLine(line);
}
string content = File.ReadAllText(filePath);
Console.WriteLine(content);
}
}
```

2. Writing to a file:

```
{
string filePath = "example.txt";
string[] lines = { "line1", "line2", "line3" };
File.WriteAllLines(filePath, lines);

string content = "Hello, C# file Handling";
File.WriteLineAllText(filePath, content);
}
```

3. Appending to a file:

```
{
string filepath = "example.txt";
string[] newlines (filePath, newlines);

File.AppendAllLines(filePath, newline)

string additional(content = "This is additional content";
File.AppendAllText(filePath, additional(content);
}
```

Name of Experiment..... Date.....

Experiment No..... Experiment Result..... Page No. 16

4 Check is file Exists.

```
using System;
class Program
{
  static void Main()
{
  string filePath = "example.txt";
  if (file.Exists(file path))
    {Console.WriteLine("file exist");}
  }
  else
{  Console.WriteLine("file doesn't exist");
  }}
```

{ Delete a file

```
using System;
class program {
static void Main()
{
    file.Delete(file path);
Console.WriteLine("file deleted successfully");
else
  {Console.WriteLine("file does not exist");
  }}
```

Teacher's Signature : _____

Name of Experiment: Web Services Date:

Experiment No. 9 Experiment Result:

Program 9:

Intro: Web Services

Web device is known as the software program. These services use the XML to exchange the info with the other protocols.

Points:

- Web service is not depended on any particular language
- Protocol Independent
- Platform independent
- also known as Stateless Architecture
- it is Scalable

Eg: Web portal, weather Reports, Stock Quote, news headline.

Steps:

1) Create a new Asp.net Core Project

2) Define Controller:

Using Microsoft.AspNetCore.Mvc;

[Route("api/[controller]")]

Public class valuesController : ControllerBase

{

```
{
public ActionResult <string> Get()
{
return "Hello from your web service";
}
}
```

3. Run the Application - Using .NET CLI :- dotnet run.

4. Testing the Service :- Use tools like Postman, CURL, or any web browser to make a GET Request to your Service URL.

Teacher's Signature : _____

Name of Experiment Web Form Date

Experiment No. 10 Experiment Result

Prgrm- 10

Intro Web form

Web form app Can be Written in any Programming language Support the CLR.

Steps:

1) Create a new .Asp.NET Web App:
   a) Open VS b) Select "Create a new project"
   c) Choose "Asp.net App" template.
   d) Name & click "Create".
2) Design the web form.
3) Code Behind (C#)

Ex-1

```
Using System;
Using System.Web.UI;
namespace XYZ
{
    public partial class Default : Page
    {
        protected void Page_Load(obj Sender, EventArgs e)
        {
            ---
        }
        protected void SubmitButton_click(obj Sender, EventArgs e)
        {
            String UserInput = TextBoxInput.Text;
```

Teacher's Signature :

```
label Output.Text = " you entered: " + User Input;
}
}
```

Html code:

```
<!-- Default.aspx -->
<%@ Page language="C#" AutoEventWireup="true" %>
<!DOCTYPE HTML>
<HTML xmlns="https://www.w3.org/1999/xhtml">
<title> Simple web form </title>
</head>
<body>
<form id="form1" runat="server">
<div>
<asp:TextBox ID="TextBoxInput" runat="server">
</asp:TextBox>
<asp:Button ID="SubmitButton" runat="server"
Text="Submit" OnClick="SubmitButton_Click" />
</br>
<asp:Label ID="LabelOutput" runat="server">
</asp:Label>
</div>
</form>
</body>
</html>
```

Teacher's Signature : _____