# LAB MANUAL

## .NET TECHNOLOGY

MISHIL DOBARIYA
160470107013
VVPEC CE SEM-6

# Contents

# Practical-1

## Aim:

Introduction to c#:

Variables:
- Initialization
- Scope
- Constant

Predefined Data Types
- Value Types
- Reference TYpes

Flow Control
- Conditional Statements(if, switch)
- Loop(for, while, dowhile, foreach)
- Jump(goto, break, continue, return)

Eumerations

Passing Arguments

```
using System;

using System.Threading;
namespace P1
{
    class P1
    {
        static int j = 90;
        public enum TimeOfDay
        {
            Morning = 0,
            Afternoon = 1,
            Evening = 2
        }
        public static void Main(string[] args)
        {
            Console.WriteLine("First Program");


            int i;
            i = 25;
            Console.WriteLine("Scope of Variables.\n1:");
            int j;
```

```
 for (int j = 0; j < 2; j++) //removing comment from for loop will raise
 error
{
    //int j;
    //uncomment above line to error "A local variable named 'j' cannot be
    declared in this
    //scope because it would give a different meaning to 'j', which is
    already
    //used in a 'parent or current' scope to denote something else"
    Console.Write("{0} {1}\n", j, P1.j);
}
Console.WriteLine("2:");
for (int k = 0; k < 3; k++)
{
    Console.Write("{0} ", k);
}
Console.Write("\n");
Console.Write(k);

for (int k = 3; k > 0; k--)
{
    Console.Write("{0} ", k);
}

Console.WriteLine("Constants");
const int valConst = 100; // This value cannot be changed.
Console.WriteLine("{0} is constant value", valConst);
valConst = 45;

const int valConst2 = valConst + 9 /* + j*/;

Console.WriteLine("Another Constant: {0}", valConst2);

Console.WriteLine("\nPredefined Data Types\n\nValue Types and Reference
Types");

//Value Types
int vali = 2, valj = vali;
Console.WriteLine("vali is: {0} and valj is: {1}", vali, valj);
valj = 90;
Console.WriteLine("vali is: {0} and valj is: {1}", vali, valj);

//Referece Types
Vector x, y;
x = new Vector();
x.value = 3;
y = x;
Console.WriteLine("x is: {0} and y is:{1}", x.value, y.value);
y.value = 234;
Console.WriteLine("x is: {0} and y is:{1}", x.value, y.value);

y = null;
Console.Write("Value for y is: " + y.value);

Console.WriteLine("\nInteger Types");
```

```
sbyte sb = 33;
short s = 33;
int _i = 33;
long l = 33L;

//Unsigned Integers
byte b = 33;
ushort us = 33;
uint ui = 33U;
ulong ul = 33UL;
 Console.WriteLine("{0} {1} {2} {3} {4} {5} {6} {7}", sb, s, _i, l, b,
 us,  ui, ul);

//Floating point types
float f = 11.22334455F;
double d = 11.2233445566778899;
Console.Write("\nFloat and Double:\n");
Console.WriteLine("{0} and \n{1}", f, d);

//Decimal Type
decimal dec = 111.22233344455566677788899 9M;
Console.WriteLine("Decimal:\n{0}", dec);


//Boolean
Console.WriteLine("\nBoolean:");
bool valBoolean = true;
Console.WriteLine("Status: " + valBoolean);

//Character
Console.WriteLine("\nCharacter:\nSingle Quote \'");
Console.WriteLine("Double Quote \"");
Console.WriteLine("Back Slash \\");
char charA = 'A';
Console.WriteLine(charA);
charA = '\0';
Console.WriteLine("Now null: " + charA);
Console.WriteLine("\a"); //Notofication Sound
Thread.Sleep(1000);
Console.Beep(); //another notification sound



object o1 = "Hi, I am an Object";
object o2 = 34;
string strObj = o1 as string;
Console.WriteLine(strObj);
Console.WriteLine(o1.GetHashCode() + " " + o1.GetType());
Console.WriteLine(o2.GetHashCode() + " " + o2.GetType());
Console.WriteLine(o1.Equals(o2));

//string
string s1, s2;
s1 = "String 1";
s2 = s1;
```

```
Console.WriteLine("S1 is: {0} and s2 is {1}", s1, s2);
s2 = "New String 1";
Console.WriteLine("S1 is: {0} and s2 is {1}", s1, s2);
s1 = "c:\\NewFolder\\Hello\\P1.cs";
Console.WriteLine(s1);
s1 = @"c:\NewFolder\Hello\P1.cs";
Console.WriteLine(s1);
s1 = @"We can also write
like this";
Console.WriteLine(s1);

//Flow Control
//The if Statement
bool isZero;
Console.WriteLine("\nFlow Control: (if)\ni is " + i);
if (i == 0)
{
    isZero = true;
    Console.WriteLine("i is Zero");
}
else
{
    isZero = false;
    Console.WriteLine("i is Non - zero");
}

//else if
Console.WriteLine("\nType in a string:");
string input;
input = Console.ReadLine();
if (input == "")
{
    Console.WriteLine("You typed in an empty string");
}
else if (input.Length < 5)
{
    Console.WriteLine("The string had less than 5 characters");
}
else if (input.Length < 10)
{
        Console.WriteLine("The string had at least 5 but less than 10
        characters");
}
Console.WriteLine("The string was " + input);

//Switch
int integerA = 2;
Console.WriteLine("\nSwitch:");

switch (integerA)
{
    case 1:
        Console.WriteLine("integerA = 1");
        break;
    case 2:
```

```
                Console.WriteLine("integerA = 2");
                //goto case 3;
                break;
            case 3:
                Console.WriteLine("integerA = 3");
                break;
            default:
                Console.WriteLine("integerA is not 1, 2, or 3");
                break;
        }


         //Loops - to be explored
         //jump statements goto, break, continue, return - to be explored

         //Enumerations
         //An enumeration is a user-defined integer type.
         //Benefits:
         //1.As mentioned, enumerations make your code easier to maintain
         //2.Enumerations make your code clearer by allowing you to refer to integer values
         by descriptive names
          //3.Enumerations make your code easier to type, too. When you go to
         assign a value to an instance of an enumerated type,
         //the Visual Studio .NET IDE will, through IntelliSense, pop up a list
         box of acceptable values in order to save
         //you some keystrokes and to remind you of what the possible options
         are.

          WriteGreeting(TimeOfDay.Morning);
          Console.WriteLine("Argument is: {0}",args[1]);
    }

    static void WriteGreeting(TimeOfDay timeOfDay)
    {
        switch (timeOfDay)
        {
            case TimeOfDay.Morning:
                Console.WriteLine("Good morning!");
                break;
            case TimeOfDay.Afternoon:
                Console.WriteLine("Good afternoon!");
                break;
            case TimeOfDay.Evening:
                Console.WriteLine("Good evening!");
                break;
            default:
                Console.WriteLine("Hello!");
                break;
        }
    }
}
```

```
        public class Vector
        {
            public int value;
        }
}
```

## Output:

```
E:\Sem-6\VS>p1.exe
First Program
Scope of Variables.
1:
0 90
1 90
2:
0 1 2
3 2 1 Constants
100 is constant value
Another Constant: 109

Predefined Data Types

Value Types and Reference Types
vali is: 2 and valj is: 2
vali is: 2 and valj is: 90
x is: 3 and y is:3
x is: 234 and y is:234


Integer Types
33 33 33 33 33 33 33 33

Float and Double:
11.22334 and
11.2233445566779
Decimal:
111.222333444555666777888999

Boolean:
Status: True

Character:
Single Quote '
Double Quote "
Back Slash \
A
Now null:

Hi, I am an Object
-1735802816 System.String
34 System.Int32
False
S1 is: String 1 and s2 is String 1
S1 is: String 1 and s2 is New String 1
```

```
c:\NewFolder\Hello\P1.cs
c:\NewFolder\Hello\P1.cs
We can also write
            like this


Flow Control: (if)
i is 25
i is Non - zero

Type in a string:
mishil
The string had at least 5 but less than 10 characters
The string was mishil

Switch:
integerA = 2
Good morning!
```

# Practical-2

## Aim:

GTU Programs

Program 1. Write console based program in code behind language VB or C# to print following pattern.

@@@@@
@@@@
@@@
@@
@

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;

namespace p2
{
    class Pattern1
    {
        static void Main(string[] args)
        {
            for (int i = 5; i > 0; i--) {
                for (int j = i; j > 0; j--) {
                    Console.Write('@');
                }
                Console.WriteLine();
            }
            Console.ReadKey();
        }
    }
}
```

## Output:

```
E:\Sem-6\VS\p2\p2>Pattern1.exe
@@@@@
@@@@
@@@
@@
@
```

Program 2. Write console based program in code behind language VB or C# to print following pattern.

1
12
123
1234

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;

namespace p2
{
    class Pattern2
    {
        static void Main(String[] ar){
            for(int i=1;i<5;i++){
                for(int j=1;j<=i;j++){
                    Console.Write(j);
                }
                Console.WriteLine();
            }
            Console.ReadKey();
        }
    }
}
```

## Output:

```
E:\Sem-6\VS\p2\p2>Pattern2.exe
1
12
123
1234
```

Program 3. Write C# code to prompt a user to input his/her name and country name and then the output will be shown as an example below:
Hello Ram from country India

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;

namespace p2
{
    class Read
    {
        static void Main(String[] ar) {
            Console.WriteLine("Enter your name:");
            string name = Console.ReadLine();
            Console.WriteLine("Enter your City:");
            string city = Console.ReadLine();
            Console.WriteLine("Hello {0} from city {1}",name,city);
        }
    }
}
```

## Output:

```
E:\Sem-6\VS\p2\p2>Read.exe
Enter your name:
mishil
Enter your City:
rajkot
Hello mishil from city Rajkot
```

Program 4. What is inheritance? Create C# console application to define Car class and derive Maruti and Mahindra from it to demonstrate inheritance.

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;

namespace p2
{
    public class Car
    {
        public virtual void display()
        {
            Console.WriteLine("This is Car class...");
        }
    }
    public class Mahindra : Car
    {
        public override void display()
        {
            Console.WriteLine("This is Mahindra class...");
        }
    }
    public class Maruti : Car
    {
        public override void display()
        {
            Console.WriteLine("This is maruti class");
        }
    }
    class Inheritance
    {
        static void Main(String[] ar){
        Maruti m = new Maruti();
        Mahindra mm = new Mahindra();
        m.display();
        mm.display();
        }
    }
}
```

## Output:

```
E:\Sem-6\VS\p2\p2>Inheritance.exe
This is maruti class
This is Mahindra class...
```

# Practical-3

## Aim:

### Overloading

Program 1: Write a c# program to add two integers, two vectors and two metric using method overloading.

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;

namespace p2
{
    public class P3_1
    {
        public int add(int a, int b) {
            return a + b;
        }
        public static Vector add(Vector v1,Vector v2) {
            Vector v= new Vector();
            v.a = v1.a + v2.a;
            v.b = v1.b + v2.b;
            return v;
        }
        public static int[,] add(int[,] a, int[,] b) {
            int[,] s = new int[2, 2];
            for (int i = 0; i < 2; i++) {
                for (int j = 0; j < 2; j++) {
                    s[i, j] = a[i, j] + b[i, j];
                }
            }
                return s;
        }
        public static void Main(String[] ar) {
            int n,n1, n2;
            Vector v = new Vector();

            Console.WriteLine("Enter Number 1:");
            n1 = Convert.ToInt32(Console.ReadLine());
            Console.WriteLine("Enter Number 2:");
            n2 = Convert.ToInt32(Console.ReadLine());
            n = n1 + n2;
            Console.WriteLine("Addition of Number:{0}", n);

            Console.WriteLine("Enter Vector 1:");
            n1 = Convert.ToInt32(Console.ReadLine());
            n2 = Convert.ToInt32(Console.ReadLine());
            Vector v1 = new Vector(n1,n2);

            Console.WriteLine("Enter Vector 2:");
```

```
            n1 =Convert.ToInt32(Console.ReadLine());
            n2 = Convert.ToInt32(Console.ReadLine());
            Vector v2 = new Vector(n1,n2);

            v = add(v1, v2);

            Console.WriteLine("Addition of vector: x={0}, y={1}",v.a,v.b);

            int[,] a = new int[,] { { 1, 2 }, { 3, 4 } };
            int[,] b = new int[,] { { 5, 6 }, { 7, 8 } };

            int[,] c = add(a, b);
            Console.WriteLine("Addition of two matrics:");
            for (int z = 0; z < 2; z++) {
                for (int m = 0; m < 2; m++) {
                    Console.WriteLine("Addition: "+ c[z, m]);
                }
            }
                Console.ReadKey();
        }
    }
    public class Vector {
        public int a, b;
        public Vector() { }
        public Vector(int a, int b)
        {
            this.a = a;
            this.b = b;
        }
    }
}
```

## Output:

```
E:\Sem-6\VS\p2\p2>P3.1.exe
Enter Number 1:
1
Enter Number 2:
2
Addition of Number:3

Enter Vector 1:
1
2
Enter Vector 2:
3
1
Addition of vector: x=4, y=3

Addition of two metrics:
Addition: 6
Addition: 8
Addition: 10
Addition: 12
```

Program 2: Write a c# program that create student object. Overload constructor to create new instant with following details.

1. Name
2. Name , Enrollment
3. Name , Enrollment, Branch

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;

namespace p2
{
    public class Student
    {
        string name, enrollment, branch;
        public Student(string name) {
            this.name = name;
             Console.WriteLine(“First Constructor initiated..”);
        }
        public Student(string name, string enrollment) {
            this.name = name;
            this.enrollment = enrollment;
             Console.WriteLine(“Second Constructor initiated..”);
        }
        public Student(string name, string enrollment, string branch) {
            this.name = name;
            this.enrollment = enrollment;
            this.branch = branch;
             Console.WriteLine(“Third Constructor initiated..”);
        }
        public static void Main(String[] ar) {
            Student s1 = new Student("Mishil");
            Student s2 = new Student("Mishil","160470107013");
            Student s3 = new Student("Mishil","160470107013","Computer");
        }
    }
}
```

## Output:

```
E:\Sem-6\VS\p2\p2>P3.2.exe
First Constructor initiated..
Second Constructor initiated..
Third Constructor initiated..
```

# Practical-4

## Aim:

### Reflection

Create a c# program to find Methods, Properties and Constructors from class of running program.(Use Class from previous practical)

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Reflection;

namespace p2
{
    class Reflection
    {
        static void Main()
        {
            Type T = Type.GetType("p2.Customer");
            MethodInfo[] methods = T.GetMethods();
            foreach (MethodInfo method in methods)
            {
                Console.WriteLine(method.ReturnType + " " + method.Name);
            }

            PropertyInfo[] properties = T.GetProperties();

            Console.WriteLine("\nProperties");
            foreach (PropertyInfo property in properties)
            {
                Console.WriteLine(property.PropertyType + " " + property.Name);
            }

            Console.WriteLine("\nConstructors");
            ConstructorInfo[] constructors = T.GetConstructors();
            foreach (ConstructorInfo constructor in constructors)
            {
                Console.WriteLine(constructor.ToString());
            }
        }
    }
    class Customer
    {
        public int ID { get; set; }
        public string Name { get; set; }
        public Customer(int ID, string Name)
        {
            this.ID = ID;
            this.Name = Name;
        }
```

```
        public Customer()
        {
            this.ID = -1;
            this.Name = string.Empty;
        }
      public void printID()
        {
            Console.WriteLine("ID is: {0}", this.ID);
        }
        public void printName()
        {
            Console.WriteLine("Name is: {0}", this.Name);
        }
    }

}
```

## Output:

```
E:\Sem-6\VS\p2\p2>Reflection.exe
System.Int32 get_ID
System.Void set_ID
System.String get_Name
System.Void set_Name
System.Void printID
System.Void printName
System.String ToString
System.Boolean Equals
System.Int32 GetHashCode
System.Type GetType

Properties
System.Int32 ID
System.String Name

Constructors
Void .ctor(Int32, System.String)
Void .ctor()
```

# Practical-5

## Aim:

File Handling

Program 1: Write a C# program to copy data from one file to another using StreamReader and StreamWriter class.

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.IO;

namespace p2
{
    class P4_1
    {
        public static void Main(){
            string f1 = @"f1.txt";
            string f2 = @"f2.txt";
            using (StreamReader reader = new StreamReader(f1))
            using (StreamWriter writer = new StreamWriter(f2))
                writer.Write(reader.ReadToEnd());
        }
    }
}
```

## Output:

```
F1.txt: Hello World…
F2.txt: Hello World…
```

Program 2: Write a C# Program to Read Lines from a File until the End of File is Reached.

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.IO;

namespace p2
{
    public class CopyFile
    {
        public void copyFile(string f1, string f2)
        {
            using (StreamReader reader = new StreamReader(f1))
            using (StreamWriter writer = new StreamWriter(f2))
            {
                string line = null;
                while ((line = reader.ReadLine()) != null)
                    writer.WriteLine(line);
            }
        }
    }
    public class mmain{
        public static void Main(){
            CopyFile cp = new CopyFile();
            string f1 = @"E:\Sem-6\VS\p2\p2\f1.txt";
            string f2 = @"E:\Sem-6\VS\p2\p2\f2.txt";
            cp.copyFile(f1,f2);

        }
    }
}
```

## Output:

```
F1.txt:
Hello World.....
hii

how
are you
???

F2.txt:
Hello World.....
hii

how
are you
???
```

Program 3: Write a C# Program to List Files in a Directory.

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.IO;

namespace p2
{
    class ListFile
    {
        public static void Main() {
            string[] Directories = Directory.GetDirectories(@"E:\Sem-6\VS");
            foreach (string dir in Directories)
                Console.WriteLine(dir);
            string[] files = Directory.GetFiles(@"E:\Sem-6\VS");
            foreach (string file in files)
                Console.WriteLine(file);

            Console.ReadKey();

        }
    }
}
```

## Output:

```
E:\Sem-6\VS\p2\p2>P4.3.exe

E:\Sem-6\VS\P1-master
E:\Sem-6\VS\p2
E:\Sem-6\VS\Assignment.docx
E:\Sem-6\VS\C# word.txt
E:\Sem-6\VS\Doc1.docx
E:\Sem-6\VS\P1-master.zip
E:\Sem-6\VS\p1.cs
E:\Sem-6\VS\p1.exe
E:\Sem-6\VS\VS.docx
E:\Sem-6\VS\~$VS.docx
```