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

Code:

```
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=25;

            Console.WriteLine("Scope of Variables.\n1:");
            int j;
            for (j = 0; j < 2; j++)
            {
                //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);
            }//Scope of k ends here
            Console.Write("\n");

            for (int k = 3; k > 0; k--)
            {
                Console.Write("{0} ", k);
            }//scope of k ends here again

            Console.WriteLine("\nConstants");
            //As the name implies, a constant is a variable whose value cannot be
changed throughout its lifetime:

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

            //const only allow constant variables into the expression
            const int valConst2 = valConst + 9 /* + j*/;
            //remove comments from the above line to see error "The expression being
assigned to 'valConst2' must be constant"
            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);
            //If a variable is a reference, it is possible to indicate that it does
not refer to any object by setting its value to null:
            y = null;
            //Console.Write("Value for y is: " + y.value);
            //uncomment above line to see runtime exception
"System.NullReferenceException: Object reference not set to an instance of an
object."
```

```
            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(2000);
            Console.Beep(); //another notification sound


            //Predefined Reference Types
            //object:
            //We can use an object reference to bind to an object of any particular
sub-type.
            //The object type implements a number of basic, general-purpose methods,
which include Equals(), GetHashCode(), GetType(), and ToString().
            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");

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

            //Enumerations
            //An enumeration is a user-defined integer type.

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