PART – A

1. Write a Program in C# to Check whether a number is Palindrome or not.

```
using System;
namespace LAB_Programs
{
   class Palindrome
   {
      public static void Main(String[] args)
      {
         int r, n, rev, m;
         Console.WriteLine("Enter the NO:");
         n = int.Parse(Console.ReadLine());
         m = n;
         rev = 0;
         while (n != 0)
         {
            r = n % 10;
            rev = rev * 10 + r;
            n = n / 10;
         }
         if (rev == m)
         {
            Console.WriteLine("its Palindrom");
         }
         else
         {
            Console.WriteLine("Not a Palindrom");
         }
      }

   }
}
```

OUTPUT

```
Enter the NO:
070
It is Palindrom
Press any key to continue . . .

Enter the NO:
567
Not a Palindrom
Press any key to continue . . .
```

2. Write a Program in C# to demonstrate Command line arguments processing.

```
using System;
namespace LAB_Programs
{
   class CommandLine
   {
         public static void Main(String[] args)
         {
            int n,f,i,j;
            for(i=0;i<args.Length;i++)
            {
               n=int.Parse(args [i]);
               f=1;
               for(j=1;j<=n;j++)
               {
                  f=f*j;
               }
               Console .WriteLine("Factorial of "+n+ " is " +f);
            }
         }
         }
}
```

<u>OUTPUT</u>

```
Factorial of 5 is 120
Factorial of 6 is 720
Press any key to continue . . .
```

3. Write a Program in C# to find the roots of Quadratic Equation.

```
using System;
namespace LAB_Programs
{
    class PP3
    {
        public static void Main(String[] args)
        {
            double a, b, c, r1, r2, d;
            Console.WriteLine("Enter a , b and c");
            a = double.Parse(Console.ReadLine());
            b = double.Parse(Console.ReadLine());
            c = double.Parse(Console.ReadLine());
            d = b * b - 4 * a * c;
            if (d == 0)
            {
                Console.WriteLine("Equal Roots");
                r1 = -b / (2 * a);
                r2 = r1;
                Console.WriteLine("Root1=" + r1 + "\nRoot2=" + r2);
            }
            else if (d > 0)
            {
                Console.WriteLine("Distinct Roots");
                r1 = (-b + Math.Sqrt(d)) / (2 * a);
                r2 = (-b - Math.Sqrt(d)) / (2 * a);
                Console.WriteLine("Root1=" + r1 + "\nRoot2=" + r2);
            }
            else
            {
                Console.WriteLine("Imaginary Roots");
                r1 = -b / (2 * a);
                r2 = Math.Sqrt(Math.Abs(d)) / (2 * a);
                Console.WriteLine("Root1=" + r1 + "+i" + r2);
                Console.WriteLine("Root1=" + r1 + "-i"  + r2);
            }
        }
            }
}
```

OUTPUT

```
Enter a , b and c
2
6
9
Imaginary Roots
Root1=-1.5+i1.5
Root1=-1.5-i1.5
```

4. Write a Program in C# to demonstrate boxing and Unboxing.

```
using System;
namespace LAB_Programs
{
    class Box_Unbox
    {
        public static void Main(String[] args)
        {
            int m=70;
            object o=m; //boxing
            try
            {
                int p=(int)o; //Unboxing;
                Console .WriteLine (p);
                long q=(long)o;
                Console .WriteLine (q);
            }
            catch (InvalidCastException e)
            {
                                       Console .WriteLine ("Unboxing int to different data type");
            }
        }
    }
}
```

OUTPUT

```
70
Unboxing int to different data type
Press any key to continue . . .
```

5. Write a Program in C# to implement Stack operations.

```
using System;
namespace LAB_Programs
{

   class Stack
   {
      int top;
      int size;
      int[] s;
      public Stack(int n)
      {
         size = n;
         top = -1;
         s = new int[size];
      }

      public void push(int item)
      {
         if (top == size - 1)
         {
            Console.WriteLine("Stack Overflow");
                  return;
         }
         top = top + 1;
         s[top] = item;
      }

      public void pop()
      {
         if (top == -1)
         {
            Console.WriteLine("Stack Underflow");
            return;
        }
        Console.WriteLine("Poped element is " + s[top]);
        top = top - 1;
    }

    public void display()
    {
        if(top ==-1)
        {
            Console .WriteLine ("Stack is Empty");
            return ;
        }
        Console .WriteLine ("Contents of Stack are");
        for(int i=top;i>=0;i–)
        {
            Console .WriteLine (s[i]);
        }
    }
}

class Stack_Operation
{
    public static void Main(String []args)
    {
        int n,item,op;
        Console .WriteLine ("Enter the size of Stack");
        n=int.Parse (Console .ReadLine ());
        Stack stk=new Stack (n);
        do
        {
            Console .WriteLine ("1.PUSH");
            Console .WriteLine ("2.POP");
            Console .WriteLine ("3.DISPLAY");
            Console .WriteLine ("4.EXIT");
        Console .WriteLine ("Enter your Choice");
        op=int.Parse (Console .ReadLine ());
        switch (op)
        {
                    case 1:Console .WriteLine ("Enter the Element to Insert");
            item =int.Parse (Console .ReadLine());
            stk.push (item );
            break ;
          case 2:
            stk.pop();
            break;
          case 3:
            stk.display ();
            break ;
        }
      }while(op!=4);
    }
  }
}
```

OUTPUT

```
Enter the size of Stack
2
1.PUSH
2.POP
3.DISPLAY
4.EXIT
Enter your Choice
1
Enter the Element to Insert
555
1.PUSH
2.POP
3.DISPLAY
4.EXIT
```

```
Enter your Choice
1
Enter the Element to Insert
777
1.PUSH
2.POP
3.DISPLAY
4.EXIT
Enter your Choice
1
Enter the Element to Insert
222
Stack Overflow

1.PUSH
2.POP
3.DISPLAY
4.EXIT
Enter your Choice
3
Contents of Stack are
777
555

1.PUSH
2.POP
3.DISPLAY
4.EXIT
Enter your Choice
2
Poped element is 777



1.PUSH
```

```
2.POP
3.DISPLAY
4.EXIT
Enter your Choice
2
Poped element is 555

1.PUSH
2.POP
3.DISPLAY
4.EXIT
Enter your Choice
2
Stack Underflow

1.PUSH
2.POP
3.DISPLAY
4.EXIT
Enter your Choice
3
Stack is Empty

1.PUSH
2.POP
3.DISPLAY
4.EXIT
Enter your Choice
4
Press any key to continue . . .
```

6. Write a program to demonstrate Operator overloading.

```
using System;
namespace LAB_Programs
{
   class Complex
   {
      double real,imag;
      public void getdata()
      {
         Console .WriteLine ("Enter real and imaginary parts");
         real=double .Parse(Console .ReadLine ());
         imag =double .Parse (Console .ReadLine  ());
      }
      public void putdata()
      {
         Console.WriteLine("Sum of 2 Complex Number is");
         Console .WriteLine (real+"+i"+imag);
      }

      public static Complex  operator + (Complex c1,Complex c2)
      {
         Complex t = new Complex ();
         t.real =c1.real +c2.real ;
         t.imag =c1.imag +c2.imag ;
         return (t);
      }
   }

   class Operator_Over
   {
      public static void Main(String []args)
      {
         Complex c1=new Complex ();
```

```
            Complex c2=new Complex();
            Console.WriteLine("Enter the First Complex number\n");
            c1.getdata ();
            Console .WriteLine ("Enter the Second Complex number");
            c2.getdata ();
            Complex c3=c1+c2;
            c3.putdata ();
        }
    }
}
```

OUTPUT

```
Enter the First Complex number
Enter real and imaginary parts
7.5
8.2
Enter the Second Complex number
Enter real and imaginary parts
9.1
4.7
Sum of 2 Complex Number is
16.6+i12.9
Press any key to continue . . .
```

7. Write a Program in C# to find the second largest element in a single dimensional

array.

```
using System;
namespace LAB_Programs
{
   class Second_Largest
   {
      public static void Main(String[] args)
      {
         int i, j, n, t, sl;
         Console.WriteLine("Enter the Array size");
         n = int.Parse(Console.ReadLine());
         int[] a = new int[n];
         Console.WriteLine("Enter Array Elements");
         for (i = 0; i < n; i++)
         {
            a[i] = int.Parse(Console.ReadLine());
         }
         for (i = 0; i < n; i++)
         {
            for (j = i + 1; j < n; j++)
            {
               if (a[i] < a[j]) //Decending Order Sorting
               {
                  t = a[i];
                  a[i] = a[j];
                  a[j] = t;
               }
            }
         }

         sl = a[0];
         for (i = 0; i < n; i++)
         {
            if (a[i] == sl)
```

```
                continue;
            else
            {
                sl = a[i];
                break;
            }
        }
        Console.WriteLine("Second Largest is" + sl);
    }
  }
}
```

OUTPUT

```
Enter the Array size
5
Enter Array Elements
45
10
88
73
60
Second Largest is73
Press any key to continue . . .
```

8. Write a Program in C# to multiply to matrices using Rectangular arrays.

```
using System;
namespace LAB_Programs
{
    class Matrix_Multi
    {
        public static void Main(String[] args)
        {
            int m, n, p, q, i, j, k;
            Console.WriteLine("Enter size of First Matrix");
            m = int.Parse(Console.ReadLine());
            n = int.Parse(Console.ReadLine());
            Console.WriteLine("Enter size of Second Matrix");
            p = int.Parse(Console.ReadLine());
            q = int.Parse(Console.ReadLine());
            if (n != p)
            {
                                        Console.WriteLine("Matrix Multiplication not Possible");
                Environment.Exit(0);
            }

            int[,] a = new int[m, n];
            int[,] b = new int[p, q];
            int[,] c = new int[m, q];
            Console.WriteLine("Enter Elements of First Matrix\n");
            for (i = 0; i < m; i++)
            {
                for (j = 0; j < n; j++)
                {
                    a[i,j] = int.Parse(Console.ReadLine());
                }
            }
            Console.WriteLine("Enter Elements of Second Matrix");
            for (i = 0; i < p; i++)
            {
                for (j = 0; j < q; j++)
                {
                    b[i,j] = int.Parse(Console.ReadLine());
                }
            }

            for (i = 0; i < m; i++)
            {
                for (j = 0; j < q; j++)
                {
                    c[i,j] = 0;
                    for (k = 0; k < n; k++)
                    {
                        c[i,j] = c[i,j] + a[i,k] * b[k,j];
                    }
                }
            }

            Console.WriteLine("Multiplication Matrix is\n");
            for (i = 0; i < m; i++)
            {
                for (j = 0; j < q; j++)
                {
                    Console.WriteLine(c[i,j] + " ");
                }
                Console.WriteLine();
            }
        }
    }
}
```

OUTPUT

```
Enter size of First Matrix
3
2
Enter size of Second Matrix
3
2
Matrix Multiplication not Possible
Enter size of First Matrix
2
3
Enter size of Second Matrix
3
2
Enter Elements of First Matrix
1
2
3
4
5
6
Enter Elements of Second Matrix
9
8
7
6
5
4
Multiplication Matrix is
38      32
101     86
Press any key to continue . . .
```

9. Find the sum of all the elements present in a jagged array of 3 inner arrays.

```
using System;
namespace LAB_Programs
{
    class Jagged_Array
    {
        public static void Main(String[] args)
        {
            int m,n,i,j,sum=-0;
            Console .WriteLine ("Enter no of Rows");
            m=int.Parse (Console .ReadLine());
            int [][]a=new int [m][];
            for(i=0;i<m;i++)
            {
                                     Console .WriteLine ("Enter no of Elements in "+(i+1)+"row");
                n=int.Parse (Console .ReadLine());
                a[i]=new int [n];
                for(j=0;j<n;j++)
                {
                    Console .WriteLine ("Enter the Elements");
                    a[i][j]=int.Parse (Console .ReadLine());
                    sum=sum+a[i][j];
                }
            }
            Console.WriteLine("Entered Elements are");
            for(i=0;i<m;i++)
            {
                for(j=0;j<a[i].Length ;j++)
                {
                    //Console.WriteLine();
                    Console .WriteLine (a[i][j]+" ");
                    Console.WriteLine(" ");
                }
                //Console .WriteLine ("\n");
            }
            Console .WriteLine ("SUM="+sum);
        }
    }
}
```

OUTPUT

```
Enter no of Rows
2
Enter no of Elements in 1row
3
Enter the Elements
11
Enter the Elements
22
Enter the Elements
33
Enter no of Elements in 2row
2
Enter the Elements
99
Enter the Elements
88
Entered Elements are
11
22
33
99
88
SUM=253
Press any key to continue . . .
```

10. Write a program to reverse a given string using C#.

```
using System;
namespace LAB_Programs
{
    class String_Rev
    {
        public static void Main(string[] args)
        {
            String s1;
            String s2="";
            Console.WriteLine("Enter a string :\n");
            s1=Console.ReadLine();
            for(int i=s1.Length-1;i>=0;i–)
            {
                s2=s2+s1.Substring(i,1);
            }
            Console.WriteLine("The reversed string is: "+s2);
         }
            }
}
```

OUTPUT

```
Enter a string :
 dog
The reversed string is: god
Press any key to continue . . .
```

11. Using Try, Catch and Finally blocks write a program in C# to demonstrate error handling.

```
using System;
namespace LAB_Programs
{
    class Error_Handling
    {
        public static void Main(string[] args)
        {
            int[] ar = { 10, 20, 30 };
            int a, b, c, d;
            Console.WriteLine("Enter a b c");
            a = int.Parse(Console.ReadLine());
            b = int.Parse(Console.ReadLine());
            c = int.Parse(Console.ReadLine());
            try
            {
                d = a / (b - c);
                Console.WriteLine("d=" + d);
                Console.WriteLine("arr[1]="+ar[1]);
                Console.WriteLine("aa[3]="+ar[3]);
                          }
            catch (DivideByZeroException e1)
            {
                Console.WriteLine("b and c must not be equal");
            }
            catch (IndexOutOfRangeException e2)
            {
                                    Console.WriteLine("accessing array out of Index Exception");
            }
            catch
            {
                Console.WriteLine("general exception");
            }
            finally
            {
                Console.WriteLine("finally is executed");
            }
        }
            }
}
```

OUTPUT

```
1. Enter a b c
5
7
7
b and c must not be equal
finally is executed
Press any key to continue . . .
2. Enter a b c
50
10
5
d=10
arr[1]=20
accessing array out of Index Exception
finally is executed
Press any key to continue . . .
```

12. Design a simple calculator using Switch Statement in C#.

```
using System;
namespace LAB_Programs
{
   class Calci
   {
      public static void Main(String[] args)
      {
         double a,b,rpt=1;
         int choice;
         while(rpt!=5)
         {
            Console.WriteLine("Select the operation");
            Console.WriteLine("1 . Addition");
            Console.WriteLine("2 . Subtraction");
            Console.WriteLine("3 . Multiplication");
            Console.WriteLine("4 . Division");
            Console.WriteLine("5 . Exit");
            Console.WriteLine("Enter ur choice :");
            choice=int.Parse(Console.ReadLine());
            switch(choice)
            {
               case 1:
                  Console.WriteLine("Enter two numbers:");
                  a=double.Parse(Console.ReadLine());
                  b=double.Parse(Console.ReadLine());
                                    Console.WriteLine("Result of Addition : "+(a+b));
                  break;
               case 2:
                  Console.WriteLine("Enter two numbers:");
                  a=double.Parse(Console.ReadLine());
                  b=double.Parse(Console.ReadLine());
                                    Console.WriteLine("Result of Subtraction : "+(a-b));
                  break;
               case 3:
                  Console.WriteLine("Enter two numbers:");
                  a=double.Parse(Console.ReadLine());
                  b=double.Parse(Console.ReadLine());
                                    Console.WriteLine("Result of Multiplication : "+(a*b));
                  break;
               case 4:
                  Console.WriteLine("Enter two numbers:");
                  a=double.Parse(Console.ReadLine());
                  b=double.Parse(Console.ReadLine());
                  if(b==0)
                  {
                          Console.WriteLine("Division not possible");
                  }
                  else
                  {
                                Console.WriteLine("Result of Division : "+(a/b));
                  }
                  break;
               case 5:
                  rpt=5;
                  break;
               default:
                  Console.WriteLine("Invalid selection");
                  break;
            }
         }
      }
         }
}
```

OUTPUT

```
Select the operation
1 . Addition
2 . Subtraction
3 . Multiplication
4 . Division
5 . Exit
Enter ur choice :
2
Enter two numbers:
6.7
9.5
Result of Subtraction : -2.8
Select the operation
1 . Addition
2 . Subtraction
3 . Multiplication
4 . Division
5 . Exit
Enter ur choice :
1

Enter two numbers:
300
678
Result of Addition : 978
Select the operation
1 . Addition
2 . Subtraction
3 . Multiplication
4 . Division
5 . Exit
Enter ur choice :
4
```

```
Enter two numbers:
5.9
8
Result of Division : 0.7375
Select the operation
1 . Addition
2 . Subtraction
3 . Multiplication
4 . Division
5 . Exit

Enter ur choice :
3
Enter two numbers:
77
0.56
Result of Multiplication : 43.12
Select the operation
1 . Addition
2 . Subtraction
3 . Multiplication
4 . Division
5 . Exit
Enter ur choice :
5
Press any key to continue . . .
```

13. Demonstrate Use of Virtual and override key words in C# with a simple program.

```
using System;
namespace LAB_Programs
{
    class A
    {
        public virtual void disp()
        {
            Console.WriteLine("hi");
        }
    }
    class B : A
    {
        public override void disp()
        {
            Console.WriteLine("Bangalore");
             }
    }
    class Virtual_Override
    {
        public static void Main(string[] args)
        {
            A x = new A();
            x.disp();
            x = new B();
            x.disp();
        }
            }
}
```

OUTPUT

```
hi
Bangalore
Press any key to continue . . .
```

14. Implement linked lists in C# using the existing collections name space.

```
using System;
using System.Collections.Generic;
using System.Text;
namespace LAB_Programs
{
  class Linhed_List
  {
    public static void Main(String []args)
    {
             Console.WriteLine("\n**DEMONSTRATION OF LINKED LIST **\n");
      LinkedList<int> ll=new LinkedList<int>();
      LinkedListNode<int> node;
      int ch, x;
             Console.WriteLine("Initial number of elements : " + ll.Count);
      Console.WriteLine();
      do
      {
        Console.WriteLine("Linked List Operations\n");
        Console.WriteLine("----------------------");
        Console.WriteLine("1.AddFirst ");
        Console.WriteLine("\n2.AddLast ");
        Console.WriteLine("\n3.RemoveFirst ");
        Console.WriteLine("\n4.RemoveLast ");
        Console.WriteLine("\n5.RemoveSpecified ");
        Console.WriteLine("\n6.Display ");
        Console.WriteLine("\n7.Exit ");
        Console.WriteLine();
        Console.WriteLine("Enter your choice : ");
        ch = int.Parse(Console.ReadLine());
```

```
switch (ch)
{
           case 1: Console.WriteLine("Enter element to   AddFirst: ");
    x = int.Parse(Console.ReadLine());
    ll.AddFirst(x);
    Console.WriteLine();
           Console.WriteLine("No of elements:" + ll.Count);
    break;

           case 2: Console.WriteLine("Enter element to AddLast : ");
    x = int.Parse(Console.ReadLine());
    ll.AddLast(x);
    Console.WriteLine();
           Console.WriteLine("No of elements:" + ll.Count);
    break;

           case 3: Console.WriteLine("Removed First element : ");
    ll.RemoveFirst();
    Console.WriteLine();
  Console.WriteLine("No of elements:" + ll.Count);
    break;

           case 4: Console.WriteLine("Removed Last element : ");
    ll.RemoveLast();
    Console.WriteLine();
  Console.WriteLine("No of elements:" + ll.Count);
    break;

           case 5: Console.WriteLine("Enter element to Remove : ");
    x = int.Parse(Console.ReadLine());
    ll.Remove(x);
         Console.WriteLine("Element "+x+" is Removed");
  Console.WriteLine("No of elements:" + ll.Count);
    break;
```

```
                    case 6: Console.WriteLine("No of elements:" + ll.Count);
                Console.WriteLine("Elements are : ");
                                for (node = ll.First; node != null; node = node.Next)
                Console.Write(node.Value + " ");
                Console.WriteLine("\n");
                break;

            case 7: Environment.Exit(0);
                break;
            default:
                Console.WriteLine("Invalid Choice : ");
                break;
         }
      } while (ch !=7);
    }
  }
}
```

OUTPUT

```
**DEMONSTRATION OF LINKED LIST **
Initial number of elements: 0
Linked List Operations
--------------------
1.AddFirst
2.AddLast
3.RemoveFirst
4.RemoveLast
5.RemoveSpecified
6.Display
7.Exit

Enter your choice :
1
Enter element to AddFirst:
70
```

```
No of elements:1
Linked List Operations
--------------------
1.AddFirst
2.AddLast
3.RemoveFirst
4.RemoveLast
5.RemoveSpecified
6.Display
7.Exit
Enter your choice :
1

Enter element to AddFirst:
55
No of elements:2
Linked List Operations
--------------------
1.AddFirst
2.AddLast
3.RemoveFirst
4.RemoveLast
5.RemoveSpecified
6.Display
7.Exit
Enter your choice :
6
No of elements:2
Elements are :
55 70
Linked List Operations
--------------------
1.AddFirst
2.AddLast
3.RemoveFirst
```

```
4.RemoveLast
5.RemoveSpecified
6.Display
7.Exit
Enter your choice :
2
Enter element to AddLast :
300
No of elements:3
Linked List Operations
---------------------
1.AddFirst
2.AddLast
3.RemoveFirst
4.RemoveLast
5.RemoveSpecified
6.Display
7.Exit
Enter your choice :
6
No of elements:3
Elements are :
55 70 300
Linked List Operations
---------------------
1.AddFirst
2.AddLast
3.RemoveFirst
4.RemoveLast
5.RemoveSpecified
6.Display
7.Exit
Enter your choice :
5
Enter element to Remove :
```

```
55
Element 10 is Removed
No of elements:2
Linked List Operations
---------------------
1.AddFirst
2.AddLast
3.RemoveFirst
4.RemoveLast
5.RemoveSpecified
6.Display
7.Exit
Enter your choice :
6
No of elements:2
Elements are :
70 300
Linked List Operations
---------------------
1.AddFirst
2.AddLast
3.RemoveFirst
4.RemoveLast
5.RemoveSpecified
6.Display
7.Exit
Enter your choice :
7
Press any key to continue . . .
```

15. Write a program to demonstrate abstract class and abstract methods in C#.

```
using System;
namespace LAB_Programs
{
    abstract class A1
    {
        public int m, n;
        public void getdata()
        {
            Console.WriteLine("Enter m and n values");
            m = int.Parse(Console.ReadLine());
            n = int.Parse(Console.ReadLine());
        }
        public void add()
        {
            Console.WriteLine("addition of two numbers=" + (m + n));
        }
        public abstract void sub();
    }

    class B1 : A1
    {
        public override void sub()
        {
            Console.WriteLine("difference of two numbers=" + (m - n));
        }

    }
    class Abstract_Method_Class
    {
        public static void Main(string[] args)
        {
            B1 x = new B1();
            x.getdata();
            x.add();
            x.sub();
         }
    }
}
```

OUTPUT

```
Enter m and n values
77
90
addition of two numbers=167
difference of two numbers=-13
Press any key to continue . . .
```

16. Write a program in C# to build a class which implements an interface which already exists.

```
using System;
namespace LAB_Programs
{
      interface Addition
      {
         int Add();
      }

      interface Multiplication
      {
         int Multiply();
      }

      class Compute : Addition, Multiplication
      {
         int x, y;
         public Compute(int a, int b)
         {
             this.x = a;
             this.y = b;
         }

         public int Add()
         {
             return (x + y);
         }

         public int Multiply()
         {
             return (x * y);
         }
      }

      class Interface_Demo
      {
         public static void Main(string[] args)
         {
             int a, b;
             Console.WriteLine("Enter 2 Numbers : ");
             a = Convert.ToInt32(Console.ReadLine());
             b = Convert.ToInt32(Console.ReadLine());
             Compute ob1 = new Compute(a, b);
             Console.WriteLine("Addition is:" + ob1.Add());
                         Console.WriteLine("Multiplication is:" + ob1.Multiply());
         }
      }
```

```
            }
```

OUTPUT

```
Enter 2 Numbers:
35
57
Addition is: 92
Multiplication is: 1995
Press any key to continue . . .
```

17. Write a program to illustrate the use of different properties in C#.

```
using System;
namespace LAB_Programs
{
    class Student
    {
        public String name;
        public int usn;

        public int Rollno
        {
            get
            {
                return usn;
            }
            set
            {
                usn = value;
            }
        }

        public String Name
        {
            get
            {
                return name;
            }
            set
            {
                name = value;
            }
        }


        class System_Properties
        {
            public static void Main(string[] args)
            {
                Student s = new Student();
                s.Rollno = 007;
                s.Name = "mca";
                int r = s.Rollno;
                String n = s.Name;
                Console.WriteLine("Name=" + n);
                Console.WriteLine("USN=" + r);
            }
        }
    }
}
```

OUTPUT

```
Name=mca
USN=007
Press any key to continue . . .
```

18. Demonstrate arrays of interface types with a C# program.

```
using System;
namespace LAB_Programs
{
    interface geometry
    {
        void area();
    }
    class triangle : geometry
    {
```

```
    public void area()
    {
        double b, h;
        Console.WriteLine("\nAREA OF TRIANGLE");
        Console.WriteLine("Enter base");
        b = double.Parse(Console.ReadLine());
        Console.WriteLine("Enter height");
        h = double.Parse(Console.ReadLine());
        double area = 0.5 * b * h;
        Console.WriteLine("area = "+ area);
    }
}
class rectangle : geometry
{
    public void area()
    {
        double b, h;
        Console.WriteLine("\nAREA OF RECTANGLE");
        Console.WriteLine("Enter breadth");
        b = double.Parse(Console.ReadLine());
        Console.WriteLine("Enter length");
        h = double.Parse(Console.ReadLine());
        double area = b * h;
        Console.WriteLine("area =" + area);
    }
}
class circle : geometry
{
    public void area()
    {
        double r;
        Console.WriteLine("\nAREA OF CIRCLE");
        Console.WriteLine("Enter radius");
        r = double.Parse(Console.ReadLine());
        double area = 3.142 * r * r;
```

```
            Console.WriteLine("area =" + area);
        }
    }

    class Array_Interface
    {
        public static void Main(string[] args)
        {
            int i;
                        geometry[] g = { new triangle(), new rectangle(), new circle() };
            for (i = 0; i < g.Length; i++)
            {
                g[i].area();
            }

        }
    }
}
```

<u>OUTPUT</u>

```
AREA OF TRIANGLE
Enter base
5.2
Enter height
2.2
area = 5.72

AREA OF RECTANGLE
Enter breadth
4
Enter length
3
```

```
area=12

AREA OF CIRCLE
Enter radius
0.87
Area =2.3781798
Press any key to continue . . .
```