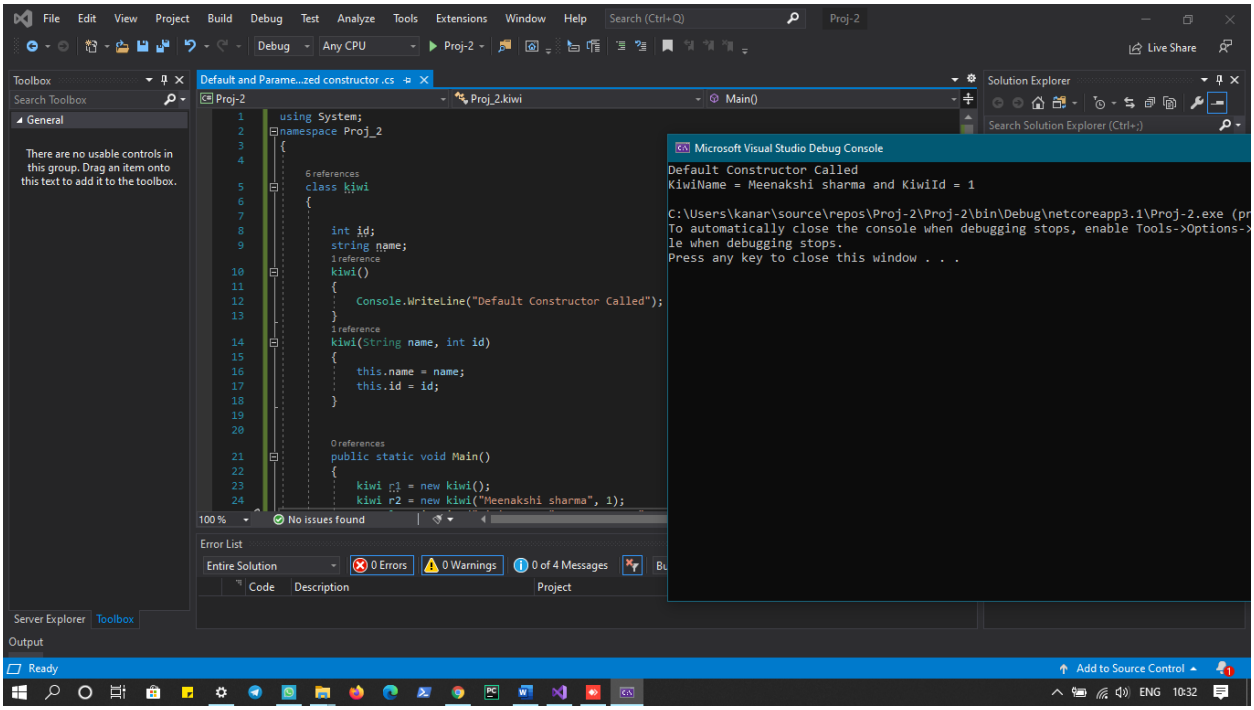
## Question 1:

Create class and try hands on all 5 constructors.

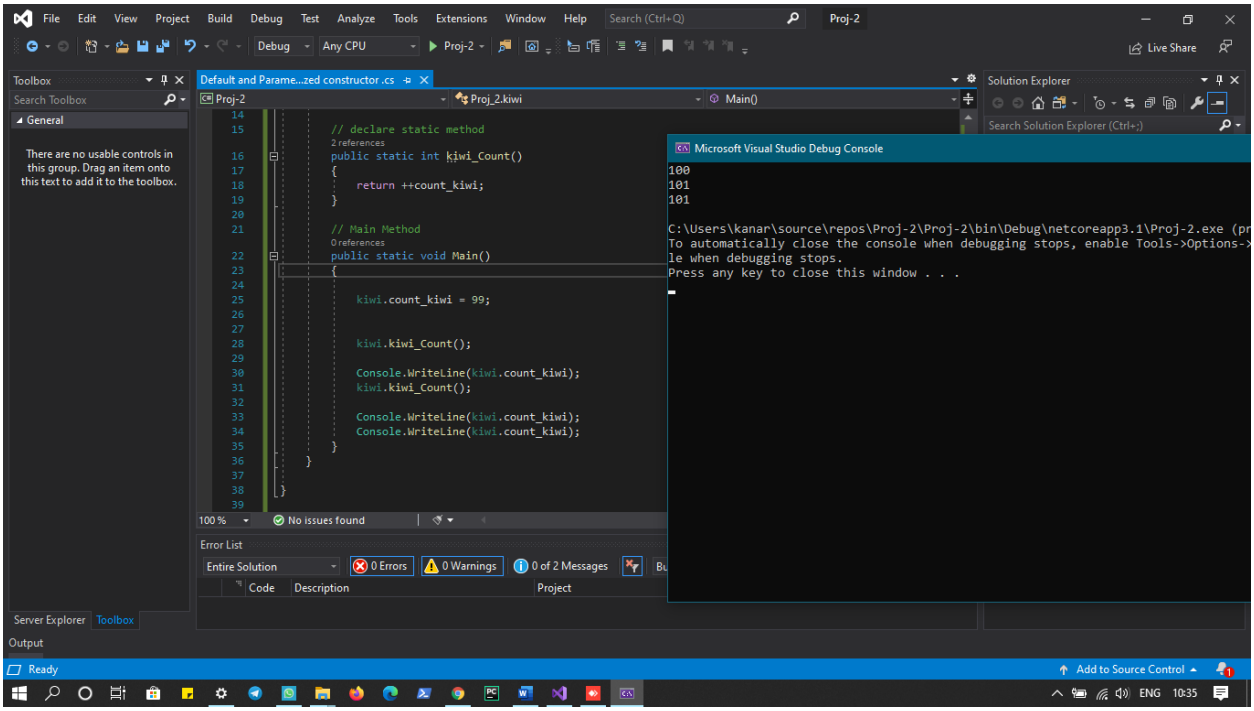



<CODE>

## For Default and Parameterized constructor

```
using System;
namespace Proj_2
{

    class kiwi
    {

        int id;
        string name;
        kiwi()
        {
            Console.WriteLine("Default Constructor Called");
        }
        kiwi(String name, int id)
        {
            this.name = name;
            this.id = id;
        }


        public static void Main()
        {
```

```
            kiwi r1 = new kiwi();
            kiwi r2 = new kiwi("Meenakshi sharma", 1);
            Console.WriteLine("KiwiName = " + r2.name + " and KiwiId = " + r2.id);

        }
    }
}
```

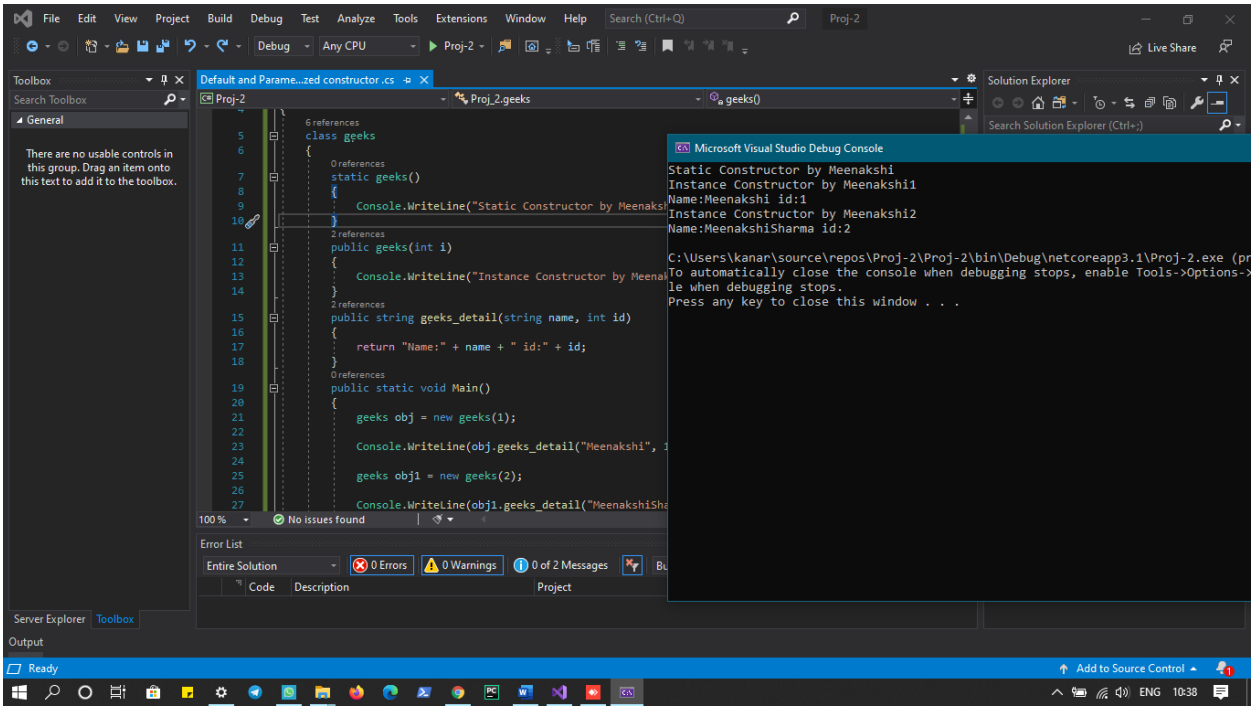



<CODE>

## For private constructor

```
using System;


namespace Proj_2
{
    public class kiwi
    {

        // declare private Constructor
        private kiwi()
        {
        }
        public static int count_kiwi;

        // declare static method
        public static int kiwi_Count()
        {
            return ++count_kiwi;
```

```
        }

        // Main Method
        public static void Main()
        {

            kiwi.count_kiwi = 99;


            kiwi.kiwi_Count();

            Console.WriteLine(kiwi.count_kiwi);
            kiwi.kiwi_Count();

            Console.WriteLine(kiwi.count_kiwi);
            Console.WriteLine(kiwi.count_kiwi);
        }
    }

}
```



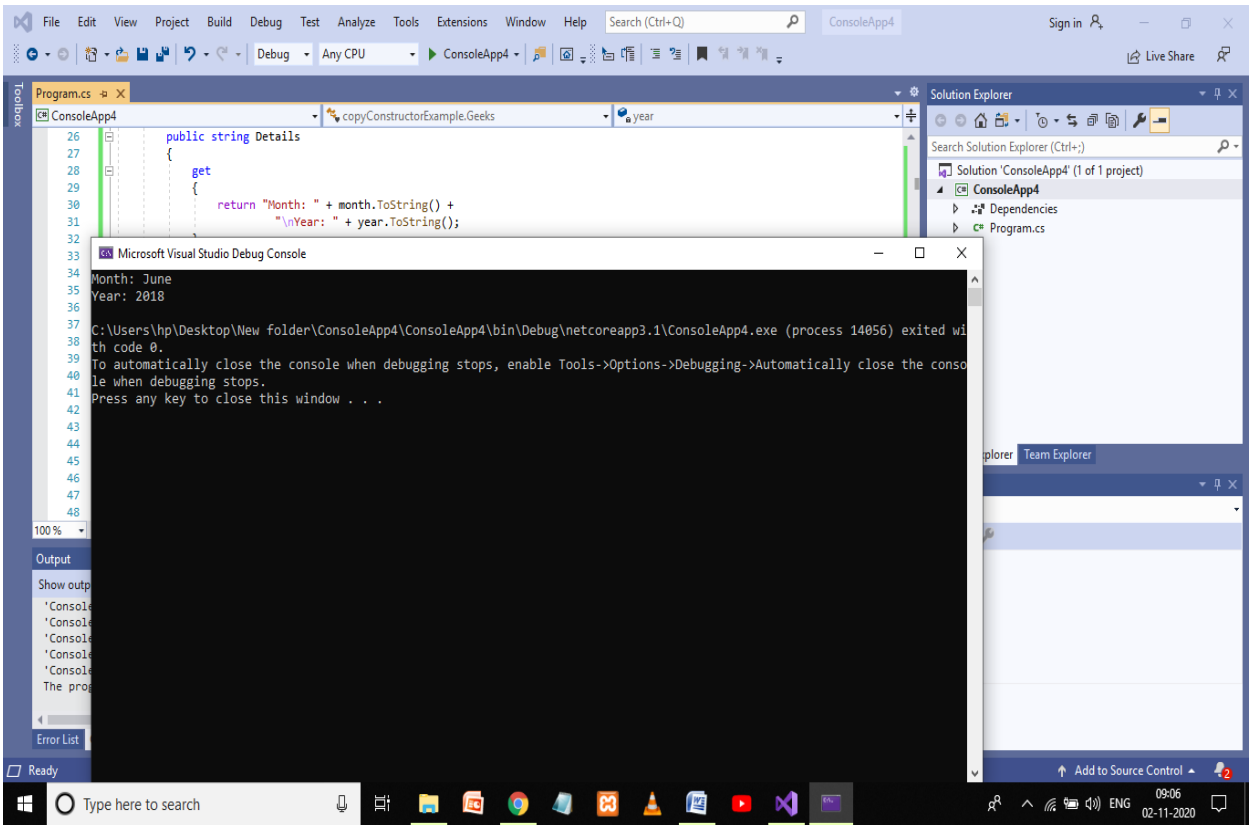

<CODE>

For static constructor

```
using System;

namespace Proj_2
```

# 4 Lab Assignment

```
{
    class geeks
    {
        static geeks()
        {
            Console.WriteLine("Static Constructor by Meenakshi");
        }
        public geeks(int i)
        {
            Console.WriteLine("Instance Constructor by Meenakshi" + i);
        }
        public string geeks_detail(string name, int id)
        {
            return "Name:" + name + " id:" + id;
        }
        public static void Main()
        {
            geeks obj = new geeks(1);

            Console.WriteLine(obj.geeks_detail("Meenakshi", 1));

            geeks obj1 = new geeks(2);

            Console.WriteLine(obj1.geeks_detail("MeenakshiSharma", 2));
        }
    }
}
```



<CODE>

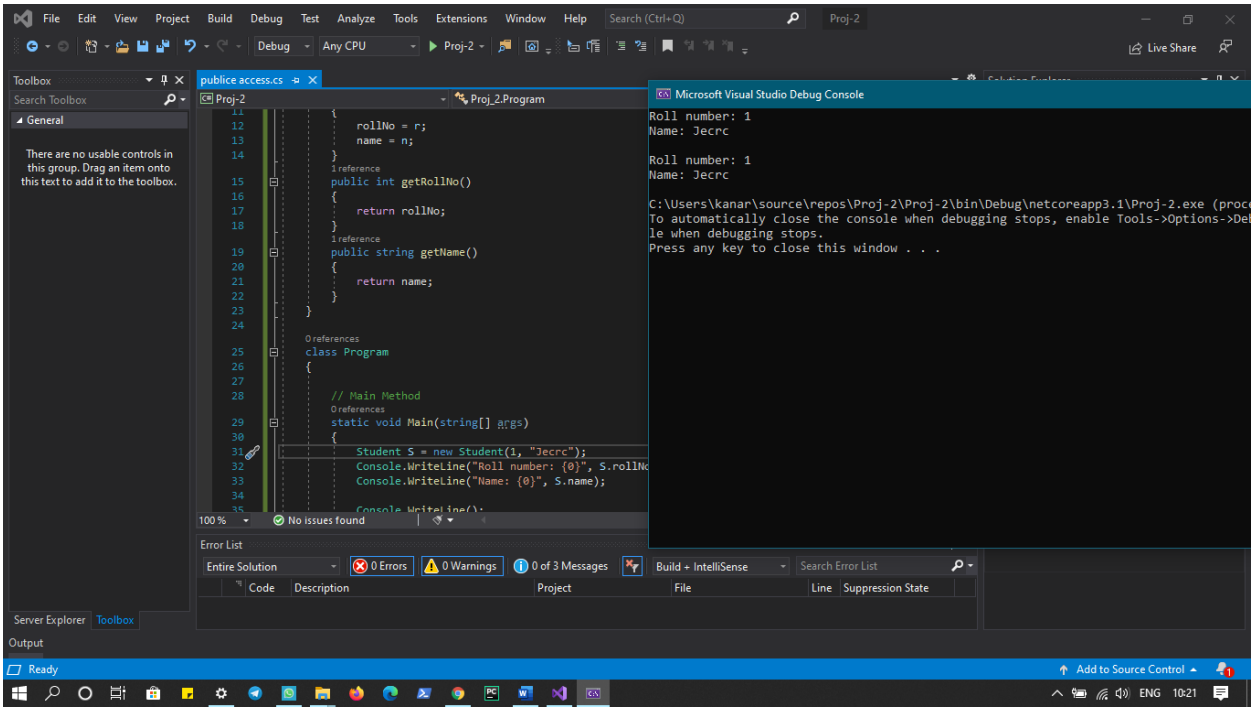
For copy constructor

```
using System;
namespace Proj_2
{

    class Geeks
    {

        private string month;
        private int year;
        public Geeks(Geeks s)
        {
            month = s.month;
            year = s.year;
        }
        public Geeks(string month, int year)
        {
            this.month = month;
            this.year = year;
        }
        public string Details
        {
            get
            {
                return "Month: " + month.ToString() +
                         "\nYear: " + year.ToString();
            }
        }
        public static void Main()
        {
            Geeks g1 = new Geeks("June", 2018);
            Geeks g2 = new Geeks(g1);

            Console.WriteLine(g2.Details);
        }
    }
}
```

Question 2:

Create class and try hands on all access modifiers.

# Lab Assignment



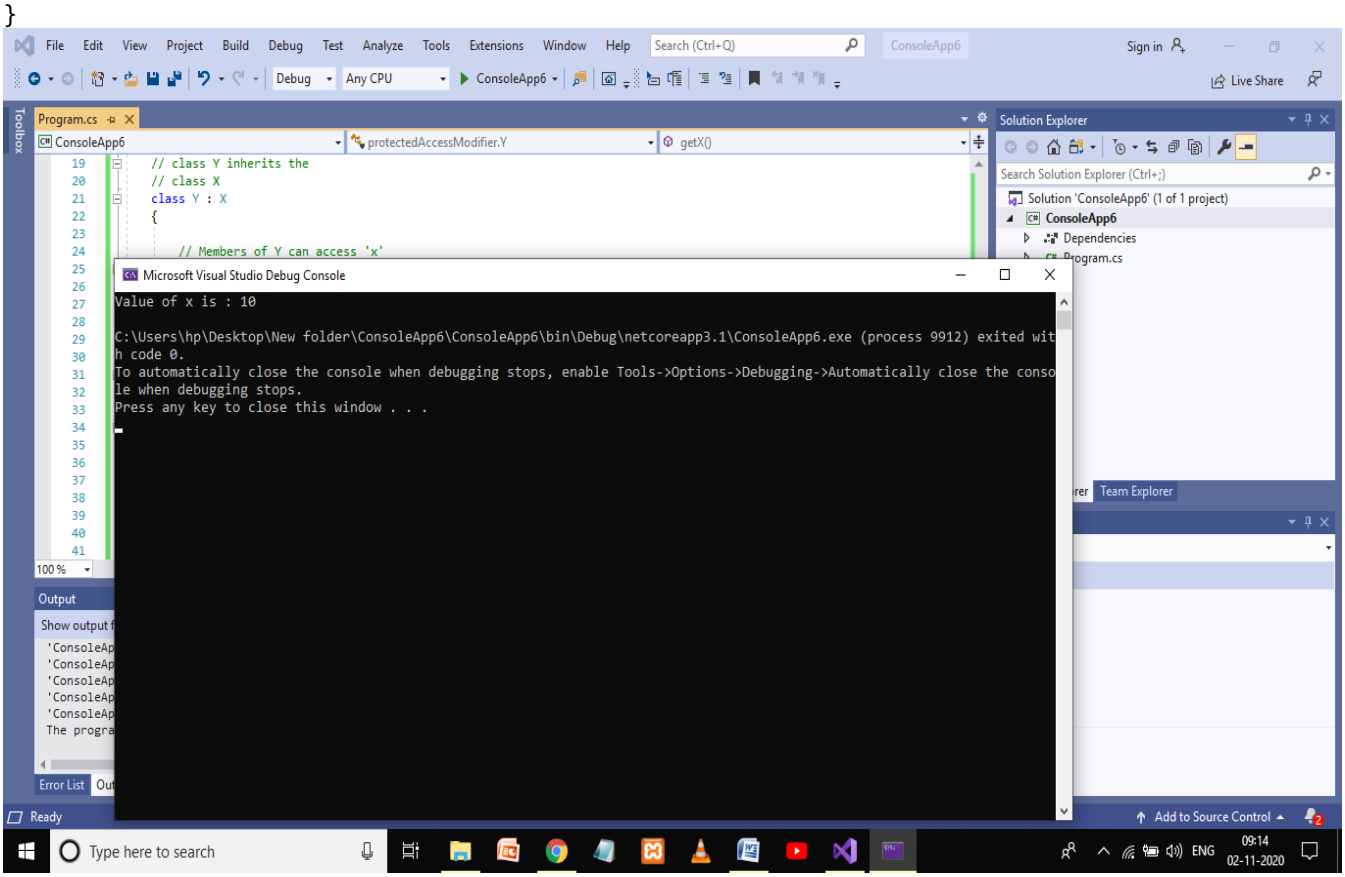

<CODE>

## For public accessibility

```
using System;

namespace Proj_2
{

    class Student
    {
        public int rollNo;
        public string name;
        public Student(int r, string n)
        {
            rollNo = r;
            name = n;
        }
        public int getRollNo()
        {
            return rollNo;
        }
        public string getName()
        {
            return name;
        }
    }

    class Program
    {
```

# Lab Assignment

```
        // Main Method
        static void Main(string[] args)
        {
            Student S = new Student(1, "Jecrc");
            Console.WriteLine("Roll number: {0}", S.rollNo);
            Console.WriteLine("Name: {0}", S.name);

            Console.WriteLine();
            Console.WriteLine("Roll number: {0}", S.getRollNo());
            Console.WriteLine("Name: {0}", S.getName());
        }
    }
}
```



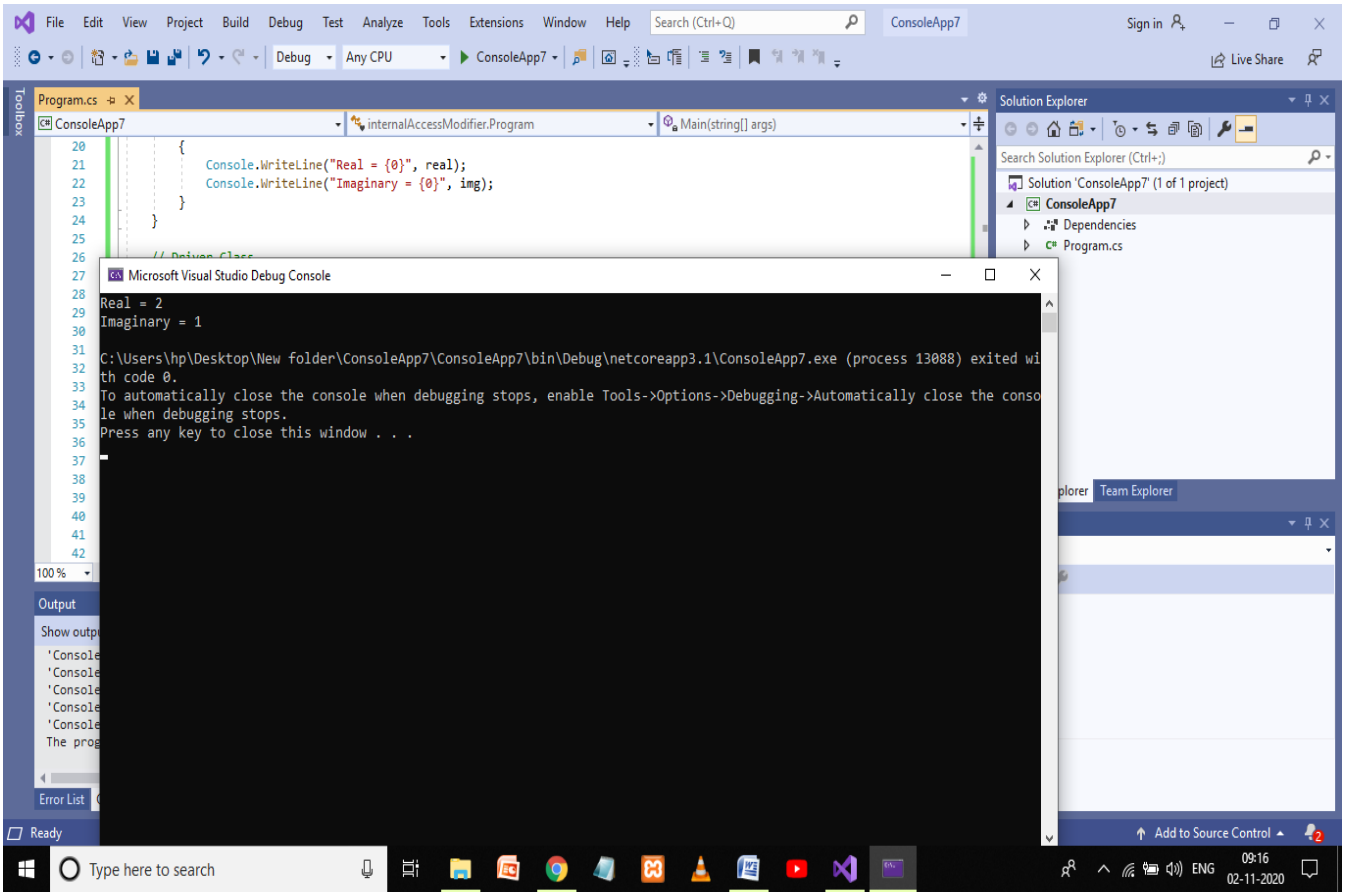

<CODE>

## For protected accessibility

```
using System;

namespace protectedAccessModifier
{

    class X
    {

        protected int x;
```

```
        public X()
        {
            x = 10;
        }
    }


    class Y : X
    {


        public int getX()
        {
            return x;
        }
    }

    class Program
    {

        static void Main(string[] args)
        {
            X obj1 = new X();
            Y obj2 = new Y();

            // Displaying the value of x
            Console.WriteLine("Value of x is : {0}", obj2.getX());
        }
    }
}
```

# Lab Assignment

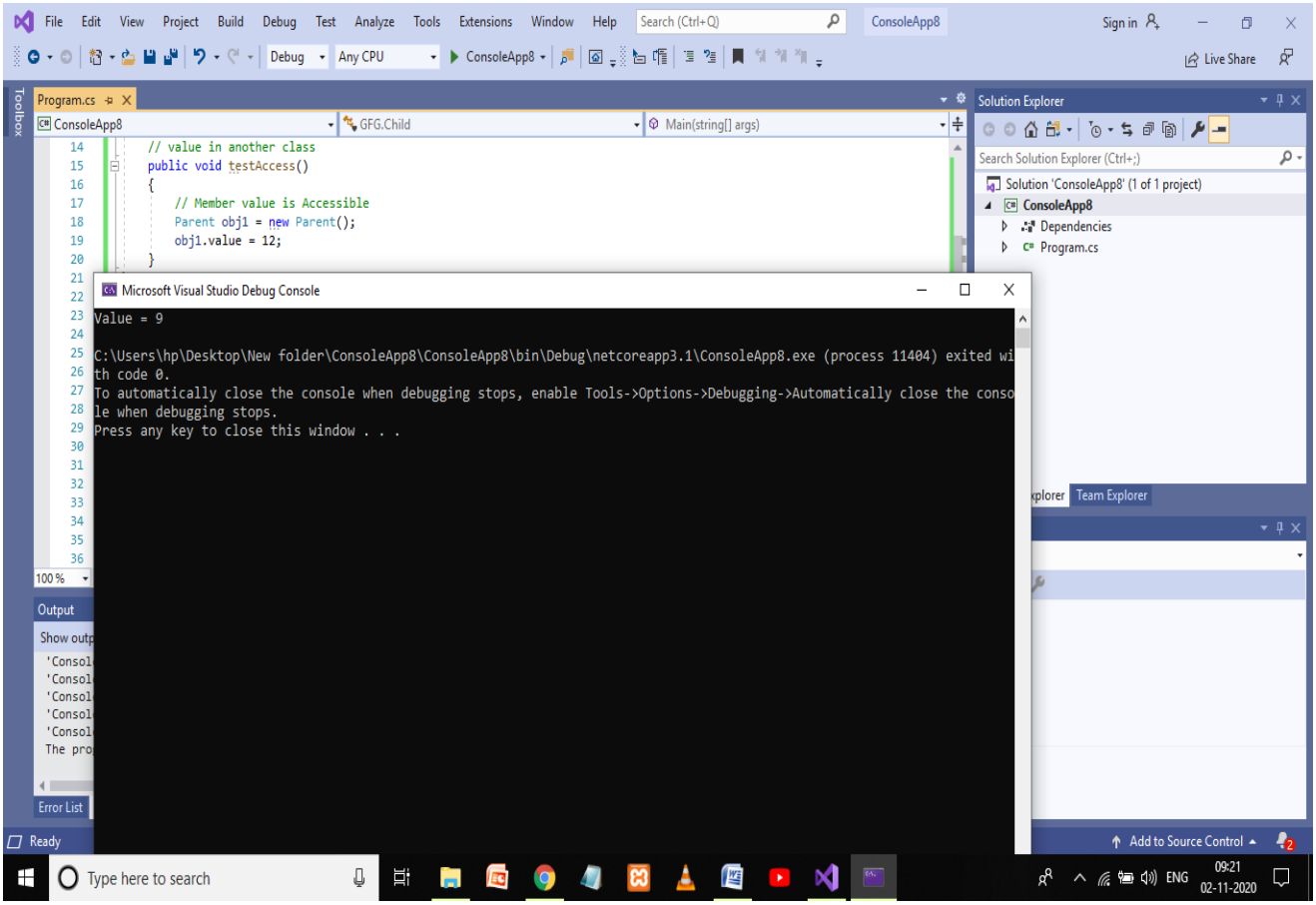



<CODE>

## For internal accessibility

```
using System;

namespace internalAccessModifier
{
    internal class Complex
    {

        int real;
        int img;

        public void setData(int r, int i)
        {
            real = r;
            img = i;
        }

        public void displayData()
        {
            Console.WriteLine("Real = {0}", real);
            Console.WriteLine("Imaginary = {0}", img);
        }
```

```
    }


    class Program
    {
        static void Main(string[] args)
        {
            Complex c = new Complex();


            c.setData(2, 1);
            c.displayData();
        }
    }
}
```




<CODE>

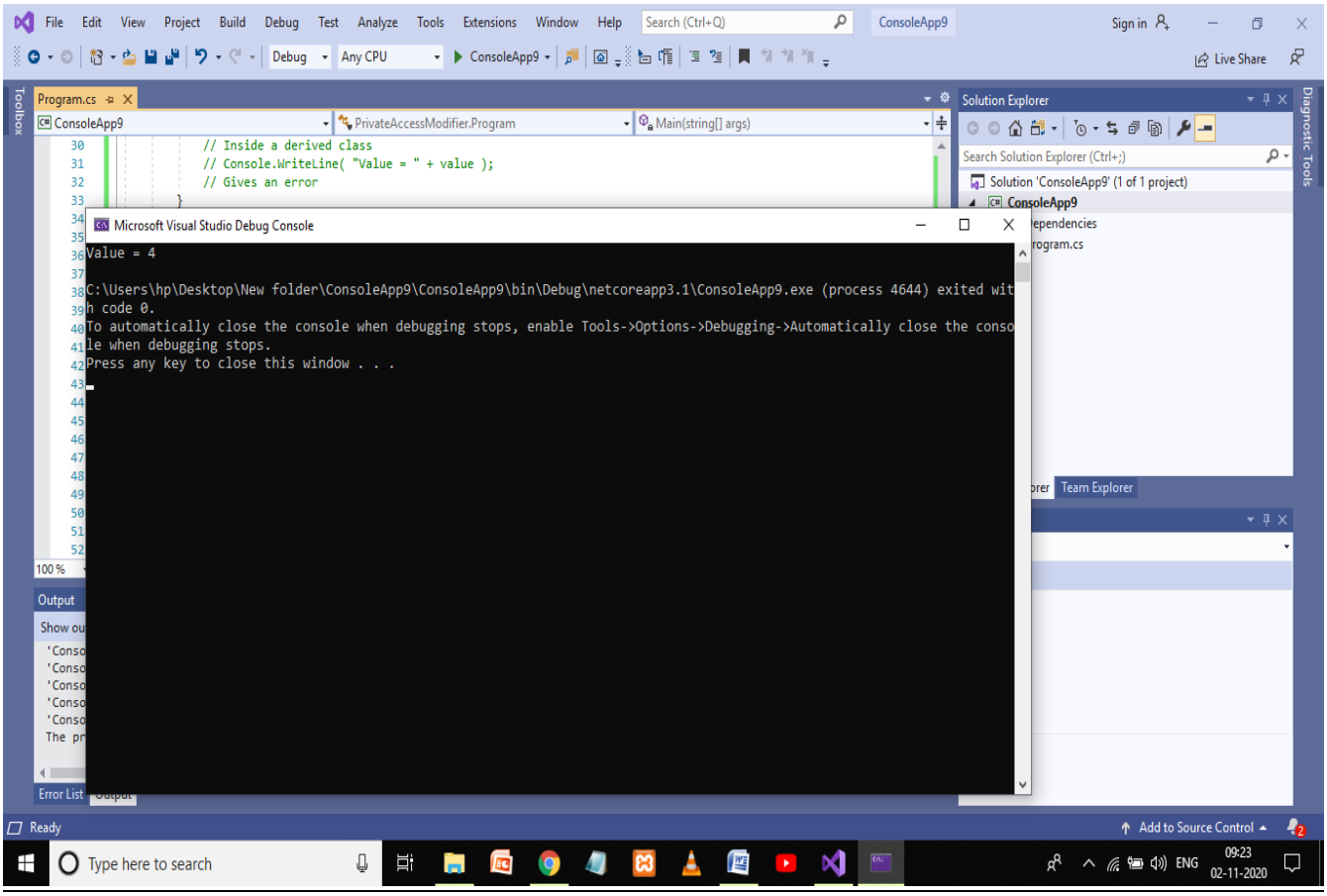
For protected internal accessibility

```
using System;

public class Parent
{
```

# Lab Assignment

```
    protected internal int value;
}

class ABC
{


    public void testAccess()
    {

        Parent obj1 = new Parent();
        obj1.value = 12;
    }
}



namespace GFG
{

    class Child : Parent
    {

        // Main Method
        public static void Main(String[] args)
        {
            Child obj3 = new Child();
            obj3.value = 9;
            Console.WriteLine("Value = " + obj3.value);
        }
    }
}
```

# Lab Assignment

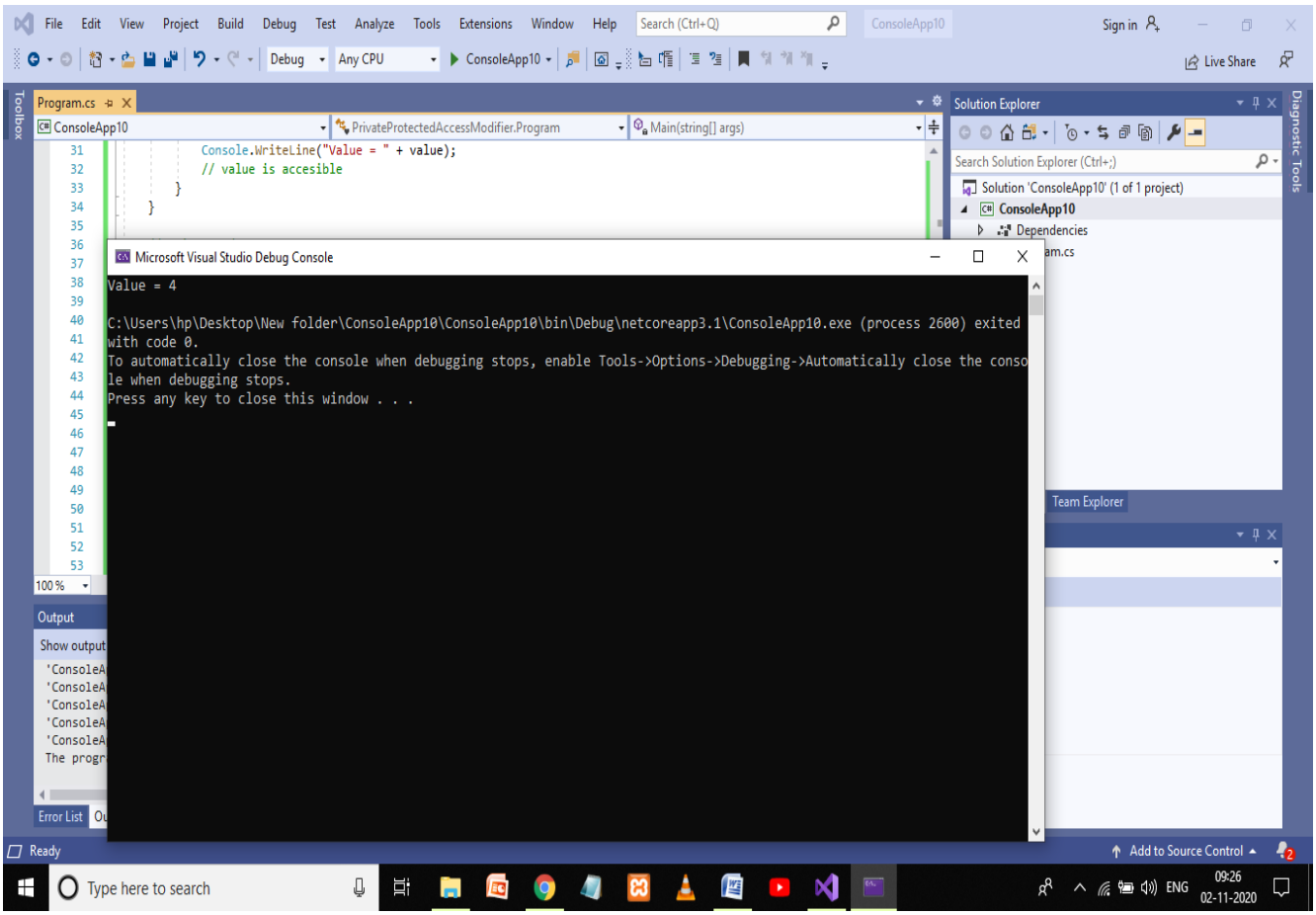



<CODE>

## For private accessibility

```
using System;

namespace PrivateAccessModifier
{

    class Parent
    {
        private int value;


        public void setValue(int v)
        {
            value = v;
        }

        public int getValue()
        {
            return value;
        }
    }
    class Child : Parent
```

```
    {

        public void showValue()
        {
        }
    }

    // Driver Class
    class Program
    {

        static void Main(string[] args)
        {
            Parent obj = new Parent();
            obj.setValue(4);
            Console.WriteLine("Value = " + obj.getValue());
        }
    }
}
```




<CODE>

For private protected accessibility

# Lab Assignment

```
using System;

namespace PrivateProtectedAccessModifier
{

    class Parent
    {
        private protected int value;
        public void setValue(int v)
        {
            value = v;
        }
        public int getValue()
        {
            return value;
        }
    }

    class Child : Parent
    {

        public void showValue()
        {

            Console.WriteLine("Value = " + value);

        }
    }

    // Driver Code
    class Program
    {

        // Main Method
        static void Main(string[] args)
        {
            Parent obj = new Parent();
            obj.setValue(4);
            Console.WriteLine("Value = " + obj.getValue());
        }
    }
}
```