ASSIGNMENT-3

PROGRAMMING WITH C#.NET

Name : Dadhaniya Malay A.

Bsc IT Java-3

Enrolment No : 220802018

Roll No : 08

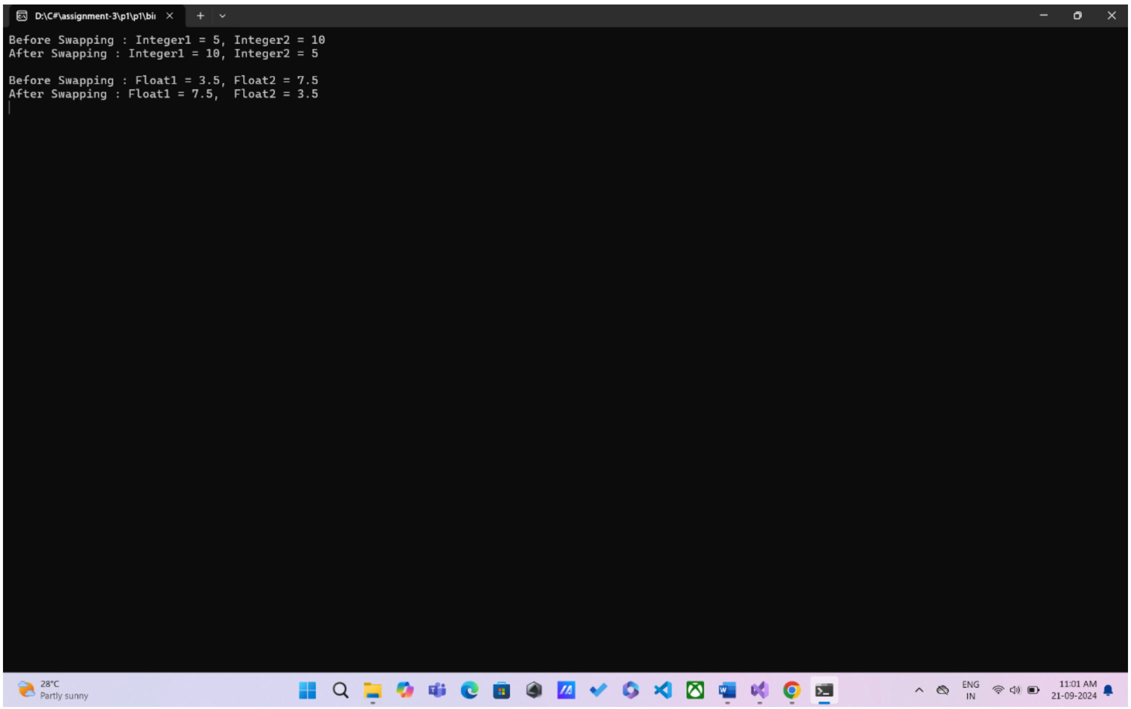
1) Write a program using function overloading to swap two integer numbers and swap two float numbers.

Program :

```
using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;


namespace ConsoleApp1

{

  class Program
  {

    public void Swap(ref int n1, ref int n2)
    {

      int temp = n1;

      n1 = n2;

      n2 = temp;

      Console.WriteLine("After Swapping : " + "Integer1 = " + n1 + ",\tInteger2 = " + n2);
    }

    public void Swap(ref float n1, ref float n2)
    {

      float temp = n1;

      n1 = n2;

      n2 = temp;

      Console.WriteLine("After Swapping : " + "Float1 = " + n1 + ",\tFloat2 = " + n2);
    }

    static void Main(string[] args)
    {

      Program program = new Program();


      int a = 5, b = 10;

      Console.WriteLine("Before Swapping : "+ "Integer1 = "+ a +  ",\tInteger2 = "+b);
```

```
            program.Swap(ref a, ref b);

            float c = 1.5f, d = 2.5f;
            Console.WriteLine("\nBefore Swapping : " + "Float1 = " + c + ",\tFloat2 = " + d);
            program.Swap(ref c, ref d);
            Console.ReadKey();
        }

    }

}
```

Output :



```
Before Swapping : Integer1 = 5, Integer2 = 10
After Swapping : Integer1 = 10, Integer2 = 5

Before Swapping : Float1 = 3.5, Float2 = 7.5
After Swapping : Float1 = 7.5,  Float2 = 3.5
```

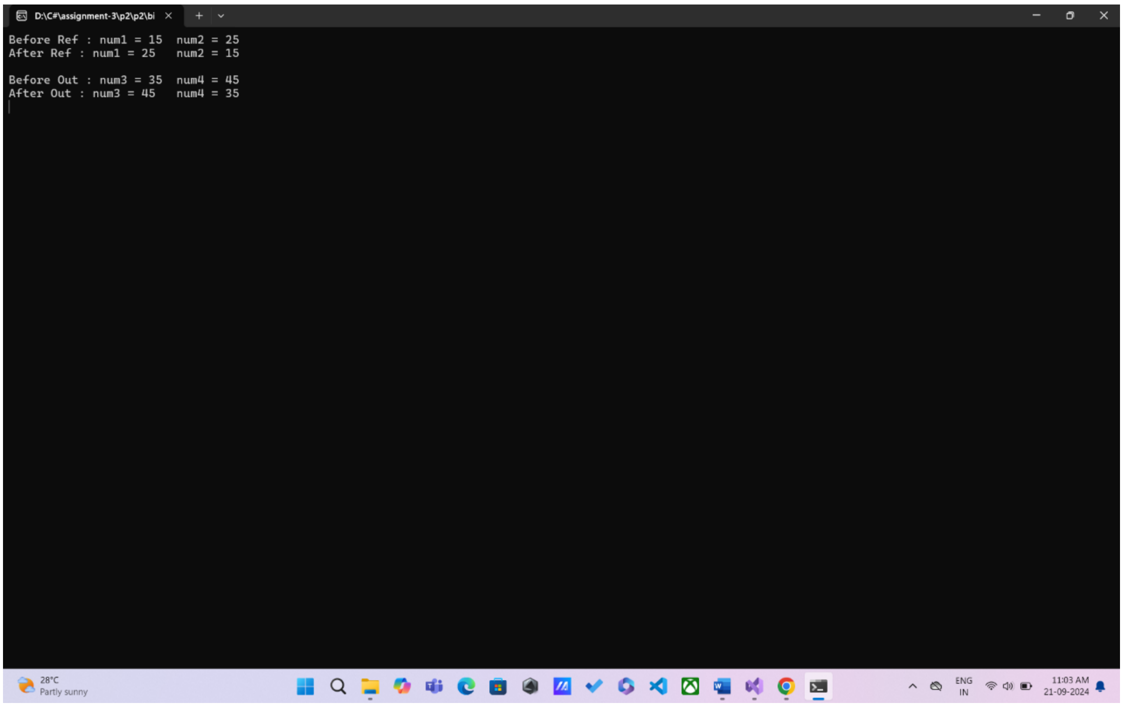
2) Write a program using ref out to swap the value of two variables.

Program :

```
using System;

using System.Collections.Generic;

using System.Linq;

 using System.Text;

using System.Threading.Tasks;


namespace ConsoleApp2

{

   class Program
   {

       static void Ref(ref int a, ref int b)
       {

           int temp = a;

           a = b;

           b = temp;

       }

       static void Out(out int a, out int b, int x, int y)
       {

             a = y;
             b = x;

       }

       static void Main(string[] args)
       {

           int num1 = 10;

           int num2 = 20;


           Console.WriteLine("Before Ref : " + "num1 = " + num1 + "\tnum2 = " + num2);


           Ref(ref num1, ref num2);


           Console.WriteLine("After Ref : " + "num1 = " + num1 + "\tnum2 = " + num2);
```

```
            int num3 = 30;
            int num4 = 40;

            Console.WriteLine("\nBefore Out : " + "num3 = " + num3 + "\tnum4 = " + num4);

            Out(out num3, out num4, num3, num4);

            Console.WriteLine("After Out : " + "num3 = " + num3 + "\tnum4 = " + num4);
            Console.ReadKey();
        }
    }
}
```

Output :



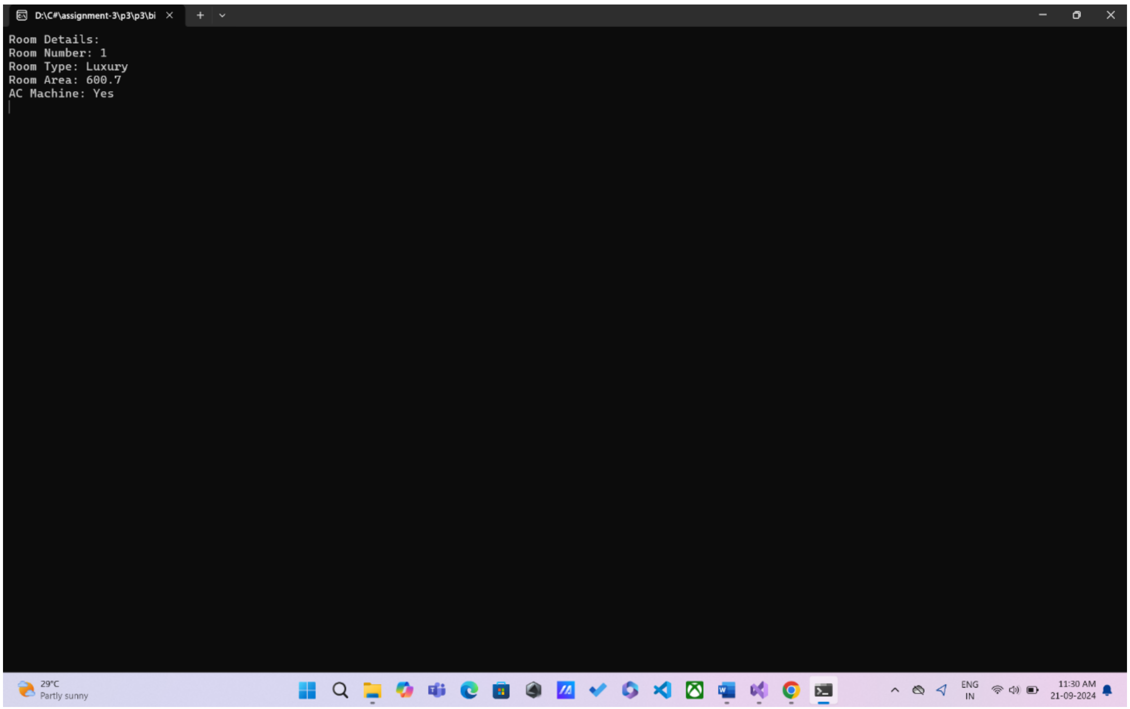
```
Before Ref : num1 = 15  num2 = 25
After Ref : num1 = 25   num2 = 15

Before Out : num3 = 35  num4 = 45
After Out : num3 = 45   num4 = 35
```

3) Write a program to create a room class, the attributes of this class is roomno, roomtype, roomarea and ACmachine. In this class the member functions are sedata() and displaydata().

Program :

```
using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;


namespace ConsoleApp3

{

  class Room
{
      private int roomno;
      private string roomtype;
      private float roomarea;
      private string ACmachine;
      public void setdata(int rno, string type, float area, string AC)       {
         roomno = rno;

        roomtype = type; roomarea = area;

         ACmachine = AC;

      }

      public void display()
      {

         Console.WriteLine("Room Details:");

         Console.WriteLine("Room Number: " + roomno);

         Console.WriteLine("Room Type: " + roomtype);

         Console.WriteLine("Room Area: " + roomarea);

         Console.WriteLine("AC Machine: " + ACmachine);

      }

  }


  class Program
  {
     static void Main(string[] args)
     {
        Room r1 = new Room();

        r1.setdata(10, "Luxury", 550.5f, "Yes");

        r1.display();

        Console.ReadKey();

     }

  }

}
```

Output :



```
Room Details:
Room Number: 1
Room Type: Luxury
Room Area: 600.7
AC Machine: Yes
```

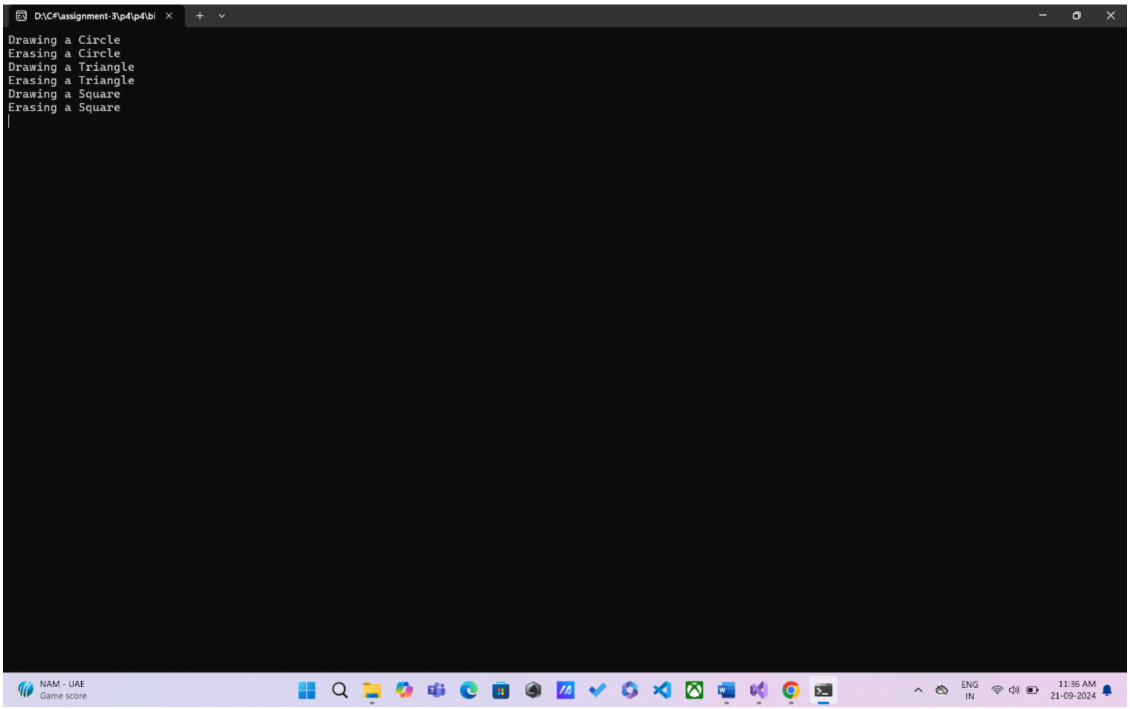
4) Write a program to create a class named shape. In this class we have three sub classes circle, triangle and shape each class two- member function named draw() and erase(). Create these using polymorphism concepts.

Program :

```
using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;


namespace ConsoleApp4

{

      class Shape
      {
               public void draw()
               {

                          Console.WriteLine("Draw The Shape");

        }

        public void erase()
        {

           Console.WriteLine("Erase The Shape");

        }
}

    class Circle : Shape

    {

        public void draw()
        {

           Console.WriteLine("Draw The Circle");

        }

        public void erase()
        {

           Console.WriteLine("Erase The Circle");
    }
}
  class Triangle : Shape
  {
    public void draw()
    {
      Console.WriteLine("Draw The Triangle");

    }
    public void erase()
    {
     Console.WriteLine("Erase The Triangle");
    }
}
  class Square : Shape

  {
    public void draw()
    {
      Console.WriteLine("Draw The Square");

    }
    public void erase()
    {
      Console.WriteLine("Erase The Square");

    }   ;
 }
 class Program
{
    static void Main(string[] args)
    {
      Shape shape;


      shape = new Circle();

     shape.draw();

     shape.erase();


      shape = new Triangle();

      shape.draw();
```

```
            shape.erase();

            shape = new Square();
            shape.draw();
            shape.erase();
            Console.ReadKey();
        }
    }
}
```

Output :



```
Drawing a Circle
Erasing a Circle
Drawing a Triangle
Erasing a Triangle
Drawing a Square
Erasing a Square
```

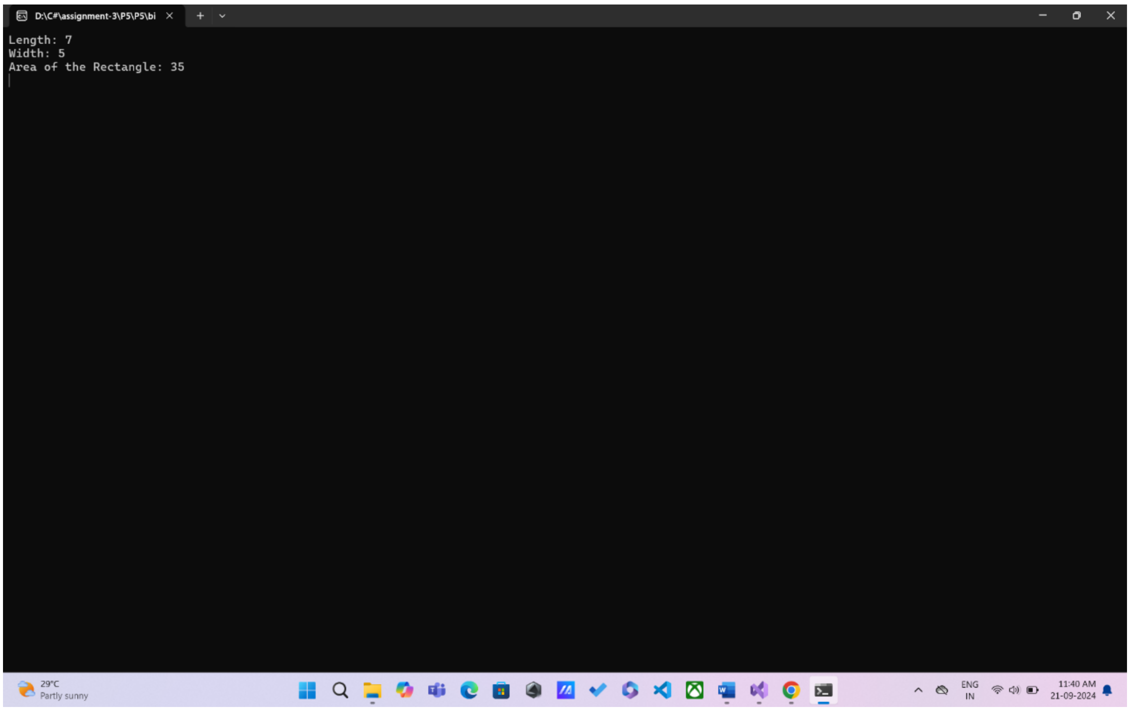
5) Create a class name rectangle and declare necessary variables (length, width, area) and method (Find_Area), create a subclass named “Tabletop” to inherit Rectangle class, create another class named Execute Rectangle and create objects for derived class and execute the method.

Program :

```
using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;


namespace ConsoleApp5

{

  class Rectangle
  {

    public double length;
    public double width;
    public double area;
    public void Find_Area()
    {

      area = length * width;

      Console.WriteLine("Area of the Rectangle: " + area);        }

  }

  class Tabletop : Rectangle

  {

    public void Display()
    {

      Console.WriteLine("Length: " + length);

      Console.WriteLine("Width: " + width);

    }
}

  class Program
  {

    static void Main(string[] args)
```

```
        {
            Tabletop table = new Tabletop();
            table.length = 5.0;

            table.width = 3.0;

            table.Display();

            table.Find_Area();

            Console.ReadKey();

        }

    }


}
```

Output :



```
Length: 7
Width: 5
Area of the Rectangle: 35
```

6) Create a class name “Person” and declare necessary variables (Roll No., Name) and methods (Public void Get_DATA(), Public virtual Display_Date()), create a subclass named “Student” to inherit Person class, create another subclass named “Details” to inherit from Student Create a class named “TestClass” and create objects for derived class and execute the methods.

Program :

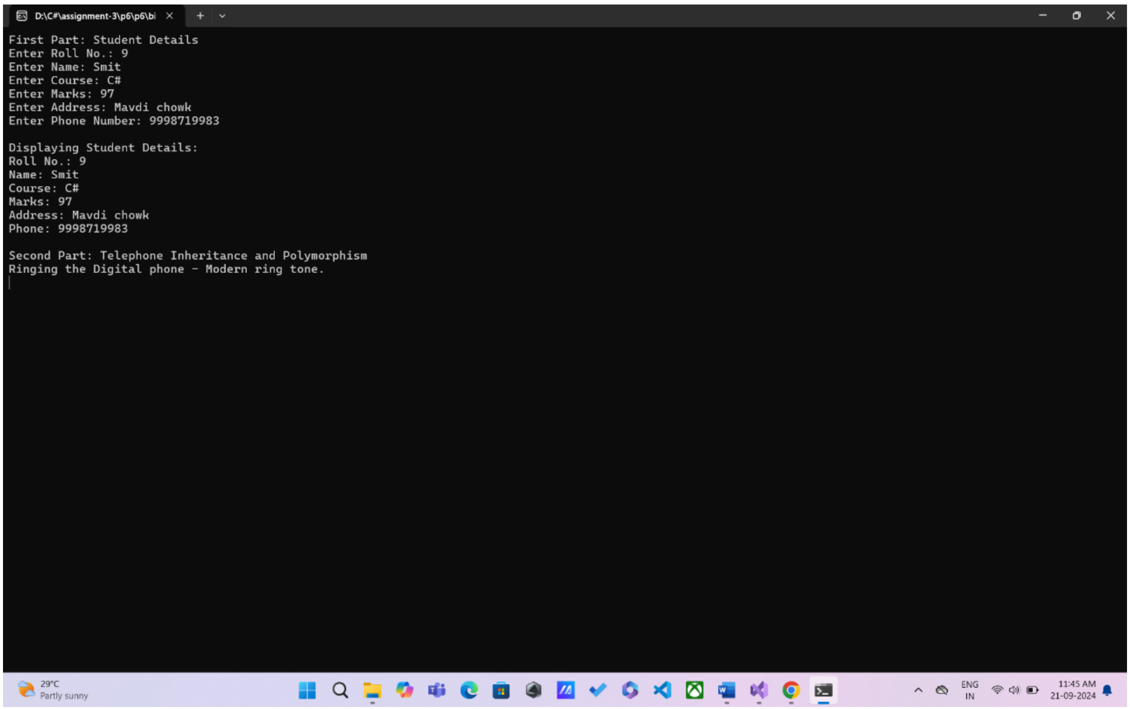
```
using System;

using System.Collections.Generic;

using System.Linq;

 using System.Text;

using System.Threading.Tasks;


namespace ConsoleApp6

{

   class Person
   {

       protected int RollNo;
       protected string Name;
       public void Get_Data()
       {

           Console.Write("Enter Roll No.: ");
           RollNo = Convert.ToInt32(Console.ReadLine());


           Console.Write("Enter Name: ");

           Name = Console.ReadLine();

       }

       public virtual void Display_Data()
       {

           Console.WriteLine($"Roll No.: {RollNo}");

           Console.WriteLine($"Name: {Name}");

       }

   }
```

```
class Student : Person

{

    protected string Course;
    protected double Marks;
    public void Get_Student_Data()
    {

        base.Get_Data();

        Console.Write("Enter Course: ");

        Course = Console.ReadLine();


        Console.Write("Enter Marks: ");

        Marks = Convert.ToDouble(Console.ReadLine());

    }

    public override void Display_Data()
    {

        base.Display_Data();

        Console.WriteLine($"Course: {Course}");

        Console.WriteLine($"Marks: {Marks}");

    }

}


class Details : Student

{

     private string Address;
     private string Phone;
     public void Get_Details()
    {

        base.Get_Student_Data();


        Console.Write("Enter Address: ");
        Address = Console.ReadLine();


        Console.Write("Enter Phone Number: ");
        Phone = Console.ReadLine();

    }

    public override void Display_Data()
```

```
        {

            base.Display_Data();

            Console.WriteLine($"Address: {Address}");

            Console.WriteLine($"Phone: {Phone}");

        }
}

    class Telephone
    {

         protected string phonetype;
         public virtual void Ring()
        {

            Console.WriteLine($"Ringing the {phonetype} phone.");

        }

    }


    class ElectronicPhone : Telephone
    {

        public ElectronicPhone()
        {

            phonetype = "Digital";

        }

        public override void Ring()
        {

            Console.WriteLine($"Ringing the {phonetype} phone - Modern ring tone.");
        }
}

    class TestClass
    {

        static void Main(string[] args)
        {

            Console.WriteLine("First Part: Student Details");
            Details details = new Details();

            details.Get_Details();

            Console.WriteLine("\nDisplaying Student Details:");

             details.Display_Data();
```

```
 Console.WriteLine("\nSecond Part: Telephone Inheritance and Polymorphism");
      ElectronicPhone digitalPhone = new ElectronicPhone();

      digitalPhone.Ring();

      Console.ReadLine();

      Console.ReadKey();

    }

  }


}
```

Output :



```
First Part: Student Details
Enter Roll No.: 9
Enter Name: Smit
Enter Course: C#
Enter Marks: 97
Enter Address: Mavdi chowk
Enter Phone Number: 9998719983

Displaying Student Details:
Roll No.: 9
Name: Smit
Course: C#
Marks: 97
Address: Mavdi chowk
Phone: 9998719983

Second Part: Telephone Inheritance and Polymorphism
Ringing the Digital phone - Modern ring tone.
```

7) Change the Telephone class to abstract, and make Ring() an abstract method. Derive two new classes from Telephone: DigitalPhone and TalkingPhone. Each derived class should set the phonetype, and override the Ring() method.

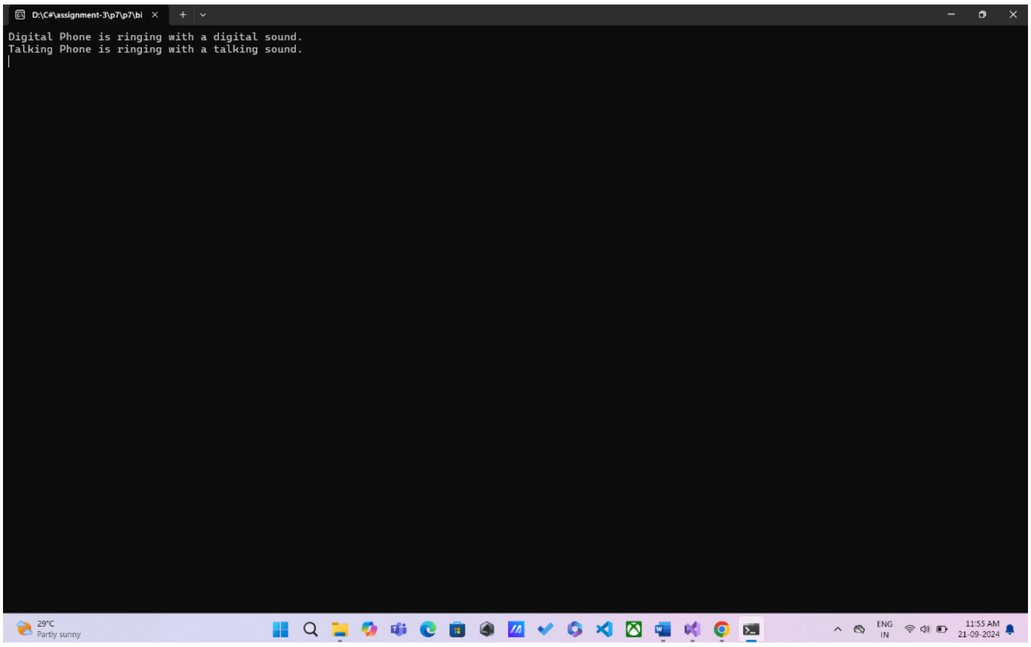
a. Create a base class, Telephone, and derive a class ElectronicPhone from it. In Telephone, create a protected string member phonetype, and a public method Ring() that outputs a text message like this: “Ringing the .” In ElectronicPhone, the constructor should set the phonetype to “Digital”. In the Run() method, call Ring() on the ElectronicPhone to test the inheritance.

b. Extend above Exercise to illustrate a polymorphic method. Have the derived class override the Ring() method to display a different message.

Program :

```
using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;


namespace ConsoleApp6

{

  abstract class Telephone
  {

      public string phonetype { get; protected set; }
      public abstract void Ring();
  }

  class DigitalPhone : Telephone
  {

      public DigitalPhone()
      {

         phonetype = "Digital Phone";
      }

      public override void Ring()
      {
            Console.WriteLine($"{phonetype} is ringing with a digital sound.");         }
    }

    class TalkingPhone : Telephone
    {

        public TalkingPhone()
        {

            phonetype = "Talking Phone";
        }

        public override void Ring()
        {

            Console.WriteLine($"{phonetype} is ringing with a talking sound.");
        }
    }
    class Program
    {

        static void Main(string[] args)
        {

            Telephone myDigitalPhone = new DigitalPhone();

            myDigitalPhone.Ring();

            Telephone myTalkingPhone = new TalkingPhone();

            myTalkingPhone.Ring();

            Console.ReadKey();

        }

    }

}
```

Output :



```
Digital Phone is ringing with a digital sound.
Talking Phone is ringing with a talking sound.
```

8) Write a program to create the class named "Employee" also create private variables named count,Emp_Id,Emp_Name,Basic_Salary,Gross_Salary using get and set properties also create the method named "Calculate_Gross_Salary() & Display_Data().", create the list of Employee class and perform the crud operation,menus are as given below a. Add Data

b. Display Data
c. Search By Id
d. Search By Name
e. Update By Id
f. Delete By Id

Program :

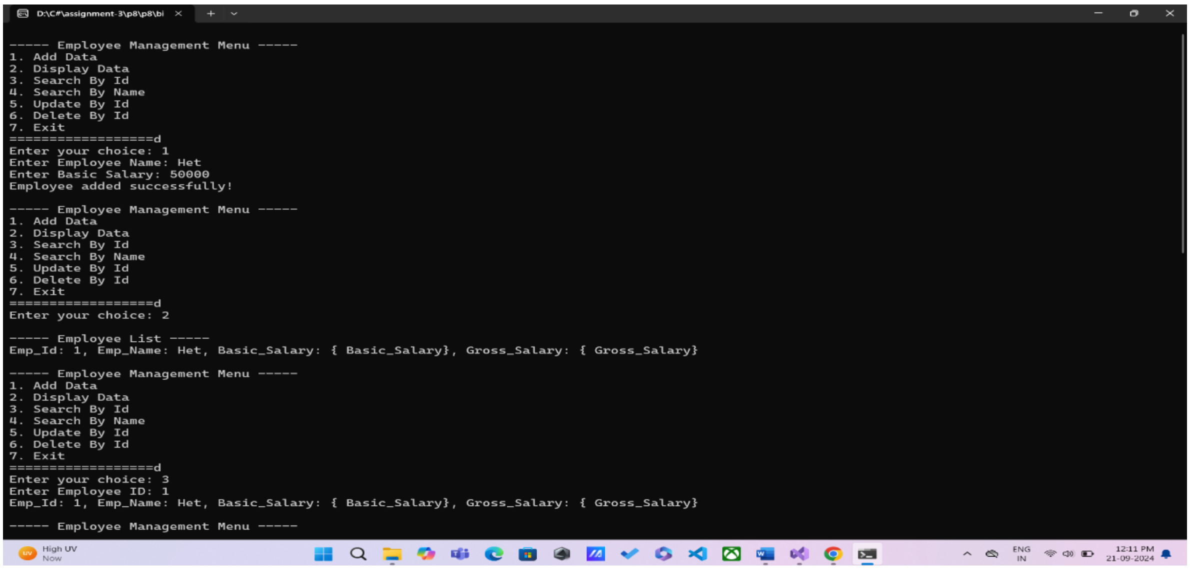
```
using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;
```

```
namespace ConsoleApp8

{

    class Employee
    {

        private static int count = 0;
        private int Emp_Id;
        private string Emp_Name;
        private double Basic_Salary;
        private double Gross_Salary;
        public int empId
        {

            get { return Emp_Id; }

            set { Emp_Id = value; }

        }

        public string empName
        {

            get { return Emp_Name; }

            set { Emp_Name = value; }

        }

        public double basicSalary
        {

            get { return Basic_Salary; }

            set { Basic_Salary = value; }

        }

        public double grossSalary
        {

            get { return Gross_Salary; }

        }

        public Employee(string name, double basicSalary)

        {

            count++;

            this.Emp_Id = count;

            this.Emp_Name = name;
    this.Basic_Salary = basicSalary;

    Calculate_Gross_Salary();

  }

  public void Calculate_Gross_Salary()

  {

    Gross_Salary = Basic_Salary + (0.3 * Basic_Salary);

  }

  public void Display_Data()

  {

    Console.WriteLine($"Emp_Id: {Emp_Id}, Emp_Name: {Emp_Name}, Basic_Salary:
     {Basic_Salary}, Gross_Salary: {Gross_Salary}");

  }

}

class Program
{

  static List<Employee> employeeList = new List<Employee>();

  static void Main(string[] args)
  {

    int choice;

    do{

      Console.WriteLine("\n----- Employee Management Menu -----");
      Console.WriteLine("1. Add Data");

      Console.WriteLine("2. Display Data");

      Console.WriteLine("3. Search By Id");

      Console.WriteLine("4. Search By Name");

      Console.WriteLine("5. Update By Id");

      Console.WriteLine("6. Delete By Id");

      Console.WriteLine("7. Exit");

     Console.WriteLine("=================d");

     Console.Write("Enter your choice: ");

     choice = Convert.ToInt32(Console.ReadLine());
```

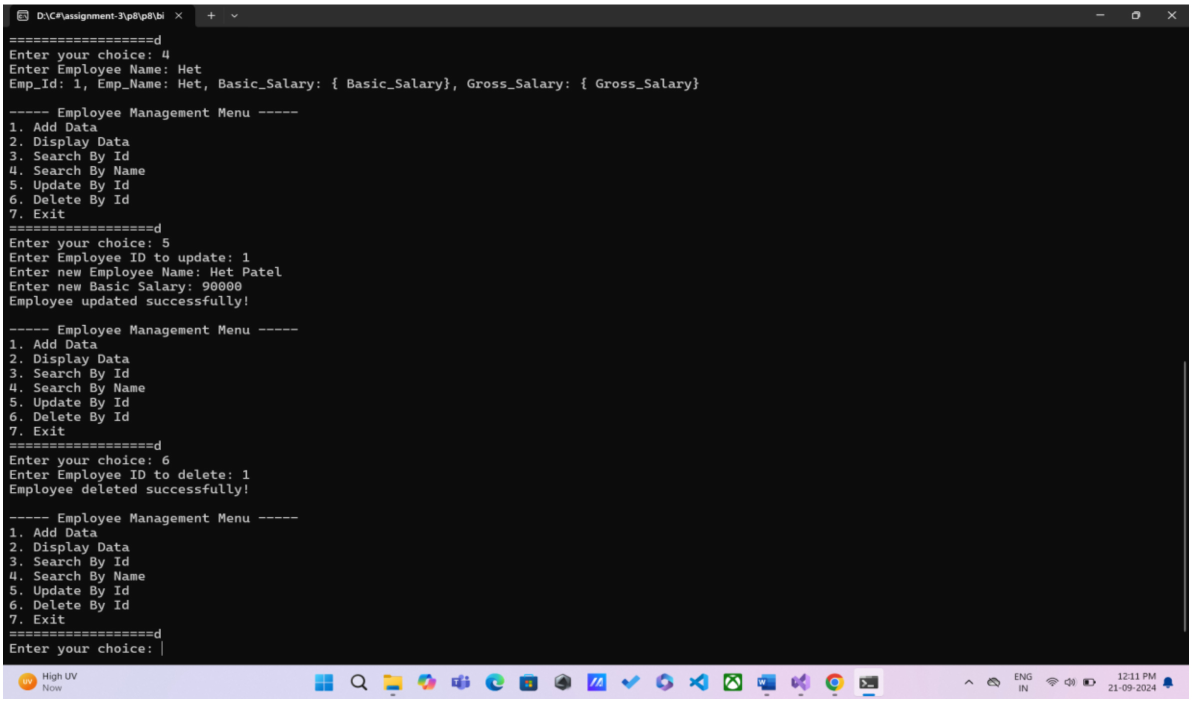
```
        switch (choice)
        {
            case 1:
                AddData();
                break;
           case 2:
                DisplayData();
                break;
           case 3:
                SearchById();
               break;
           case 4:
                SearchByName();
                break;
            case 5:
                UpdateById();
                break;
            case 6:
                DeleteById();
               break;
            case 7:
                Console.WriteLine("Exiting program...");
               break;
            default:
                Console.WriteLine("Invalid choice, please try again.");
                break;
        }

    } while (choice != 7);
}
static void AddData()
{
    Console.Write("Enter Employee Name: ");
    string name = Console.ReadLine();

    Console.Write("Enter Basic Salary: ");
    double basicSalary = Convert.ToDouble(Console.ReadLine());
    Employee emp = new Employee(name, basicSalary);
    employeeList.Add(emp);

    Console.WriteLine("Employee added successfully!");
}
static void DisplayData()
{
    if (employeeList.Count > 0)
    {
        Console.WriteLine("\n----- Employee List -----");
        foreach (var emp in employeeList)
        {
            emp.Display_Data();
        }
   }
   else
    {
        Console.WriteLine("No employee data found.");
    }

}
static void SearchById()
{
    Console.Write("Enter Employee ID: ");
    int id = Convert.ToInt32(Console.ReadLine());

    Employee emp = employeeList.FirstOrDefault(e => e.empId ==id);
    if (emp != null)
    {
        emp.Display_Data();
    }
    else
    {
        Console.WriteLine("Employee not found.");
    }
}
static void SearchByName()
{
    Console.Write("Enter Employee Name: ");
    string name = Console.ReadLine();

    Employee emp = employeeList.FirstOrDefault(e =>e.empName.Equals(name,
      StringComparison.OrdinalIgnoreCase));
    if (emp != null)
    {
        emp.Display_Data();
    }
    else
    {
        Console.WriteLine("Employee not found.");
    }
}
static void UpdateById()
{
    Console.Write("Enter Employee ID to update: ");
    int id = Convert.ToInt32(Console.ReadLine());

    Employee emp = employeeList.FirstOrDefault(e => e.empId == id);
    if (emp != null)
    {
        Console.Write("Enter new Employee Name: ");
        emp.empName = Console.ReadLine();
        Console.Write("Enter new Basic Salary: ");
        emp.basicSalary = Convert.ToDouble(Console.ReadLine());
```

```
                emp.Calculate_Gross_Salary();
                Console.WriteLine("Employee updated successfully!");
            }
            else
            {
                Console.WriteLine("Employee not found.");
            }
        }
        static void DeleteById()
        {
            Console.Write("Enter Employee ID to delete: ");
            int id = Convert.ToInt32(Console.ReadLine());

            Employee emp = employeeList.FirstOrDefault(e => e.empId == id);
            if (emp != null)
            {
                employeeList.Remove(emp);
                Console.WriteLine("Employee deleted successfully!");
            }
            else
            {
                Console.WriteLine("Employee not found.");
            }
        }
    }
}
```

Output :



```
----- Employee Management Menu -----
1. Add Data
2. Display Data
3. Search By Id
4. Search By Name
5. Update By Id
6. Delete By Id
7. Exit
=================d
Enter your choice: 1
Enter Employee Name: Het
Enter Basic Salary: 50000
Employee added successfully!

----- Employee Management Menu -----
1. Add Data
2. Display Data
3. Search By Id
4. Search By Name
5. Update By Id
6. Delete By Id
7. Exit
=================d
Enter your choice: 2

----- Employee List -----
Emp_Id: 1, Emp_Name: Het, Basic_Salary: { Basic_Salary}, Gross_Salary: { Gross_Salary}

----- Employee Management Menu -----
1. Add Data
2. Display Data
3. Search By Id
4. Search By Name
5. Update By Id
6. Delete By Id
7. Exit
=================d
Enter your choice: 3
Enter Employee ID: 1
Emp_Id: 1, Emp_Name: Het, Basic_Salary: { Basic_Salary}, Gross_Salary: { Gross_Salary}

----- Employee Management Menu -----
```



```
=================d
Enter your choice: 4
Enter Employee Name: Het
Emp_Id: 1, Emp_Name: Het, Basic_Salary: { Basic_Salary}, Gross_Salary: { Gross_Salary}

----- Employee Management Menu -----
1. Add Data
2. Display Data
3. Search By Id
4. Search By Name
5. Update By Id
6. Delete By Id
7. Exit
=================d
Enter your choice: 5
Enter Employee ID to update: 1
Enter new Employee Name: Het Patel
Enter new Basic Salary: 90000
Employee updated successfully!

----- Employee Management Menu -----
1. Add Data
2. Display Data
3. Search By Id
4. Search By Name
5. Update By Id
6. Delete By Id
7. Exit
=================d
Enter your choice: 6
Enter Employee ID to delete: 1
Employee deleted successfully!

----- Employee Management Menu -----
1. Add Data
2. Display Data
3. Search By Id
4. Search By Name
5. Update By Id
6. Delete By Id
7. Exit
=================d
Enter your choice: 
```

9) Write a program to create the class named “Product” also create the private variables named count,Cust_Id,Cust_Name,Product_Name,Price with indexer also create the method named, “Show_Products()”, create the list of Product class and perform the crud operations,menus are as given below.

a. Add Data

b. Display Data

Program :

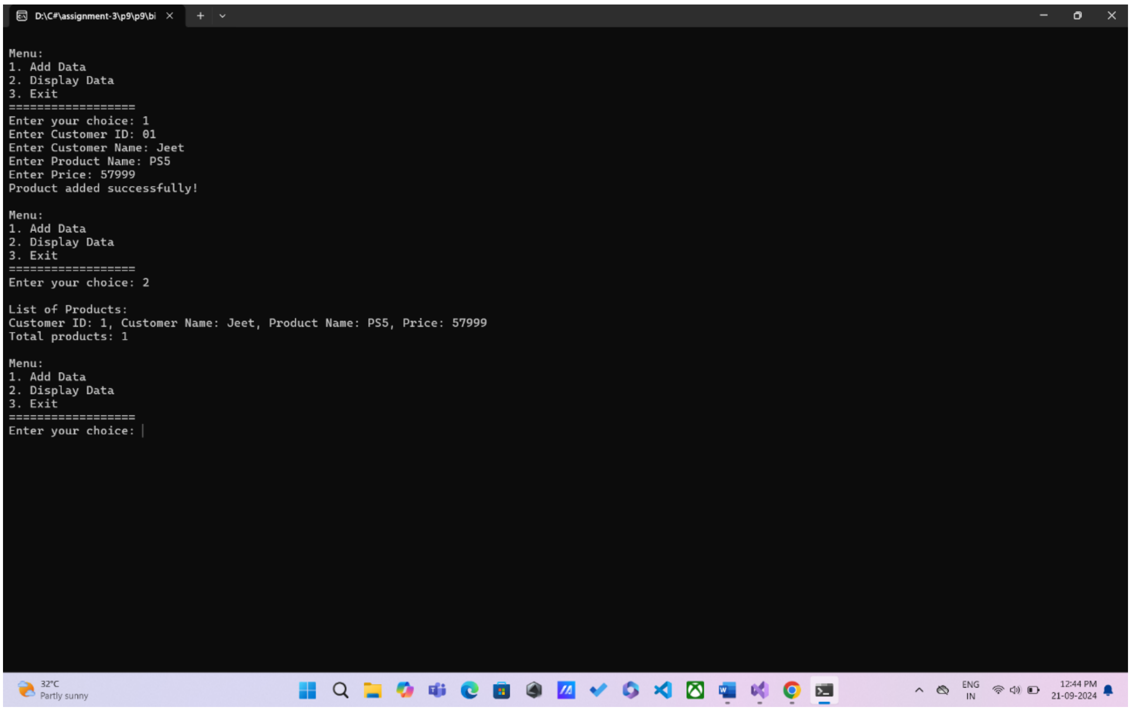
```
using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;


namespace ConsoleApp9

{

  class Product
  {

    private int Cust_Id;
    private string Cust_Name;
    private string Product_Name;
    private double Price;
    private static int count = 0;
    public Product(int custId, string custName, string productName, double price)       {
        this.Cust_Id = custId;

        this.Cust_Name = custName;

        this.Product_Name = productName;
        this.Price = price;

        count++;

    }

    public object this[int index]
    {
      get
      {
         switch (index)
```

```
               {
                  case 0:
                       return Cust_Id;

                   case 1:

                       return Cust_Name;

                  case 2:

                       return Product_Name;

                   case 3:

                       return Price;
                   default:
                       return null;
               }
}
          set
           {

               switch (index)

               {
                  case 0:
                       Cust_Id = (int)value;

                       break;

                   case 1:

                       Cust_Name = (string)value;

                       break;

                   case 2:

                       Product_Name = (string)value;

                      break;

                  case 3:

                       Price = (double)value;

                       break;
               }

           }

        }

        public void Show_Products()

        {

           Console.WriteLine($"Customer ID: {Cust_Id}, Customer Name: {Cust_Name}, Product Name:
```

```
           {Product_Name}, Price: {Price:C}");

    }

    public static void ShowCount()
    {

        Console.WriteLine($"Total products: {count}");

    }

}


class Program
{

    static List<Product> productList = new List<Product>();


    static void Main(string[] args)
    {

        int choice;

       do

        {

            Console.WriteLine("\nMenu:");

            Console.WriteLine("1. Add Data");

            Console.WriteLine("2. Display Data");

            Console.WriteLine("3. Exit");

            Console.WriteLine("=================");

           Console.Write("Enter your choice: ");

           choice = int.Parse(Console.ReadLine());


            switch (choice)

            {
                case 1:
                    AddProduct();

                    break;
                case 2:
                    DisplayProducts();

                   break;

               case 3:
                Console.WriteLine("Exiting...");
                break;
            default:
                Console.WriteLine("Invalid choice, please try again.");
                break;
        }
    } while (choice != 3);
    Console.ReadKey();
}
static void AddProduct()
{
    Console.Write("Enter Customer ID: ");
    int custId = int.Parse(Console.ReadLine());

    Console.Write("Enter Customer Name: "); ‘
    string custName = Console.ReadLine();

    Console.Write("Enter Product Name: ");
   string productName = Console.ReadLine();

    Console.Write("Enter Price: ");
    double price = double.Parse(Console.ReadLine());

    Product newProduct = new Product(custId, custName, productName, price);
    productList.Add(newProduct);

    Console.WriteLine("Product added successfully!");
}
static void DisplayProducts()
{
    if (productList.Count == 0)
    {
        Console.WriteLine("No products available.");
    }
```

```
          else
           {
              Console.WriteLine("\nList of Products:");

              foreach (Product product in productList)

              {

                 product.Show_Products();

              }

              Product.ShowCount();

           }

        }

     }


  }
```

Output :



```
Menu:
1. Add Data
2. Display Data
3. Exit
==================
Enter your choice: 1
Enter Customer ID: 01
Enter Customer Name: Jeet
Enter Product Name: PS5
Enter Price: 57999
Product added successfully!

Menu:
1. Add Data
2. Display Data
3. Exit
==================
Enter your choice: 2

List of Products:
Customer ID: 1, Customer Name: Jeet, Product Name: PS5, Price: 57999
Total products: 1

Menu:
1. Add Data
2. Display Data
3. Exit
==================
Enter your choice:
```

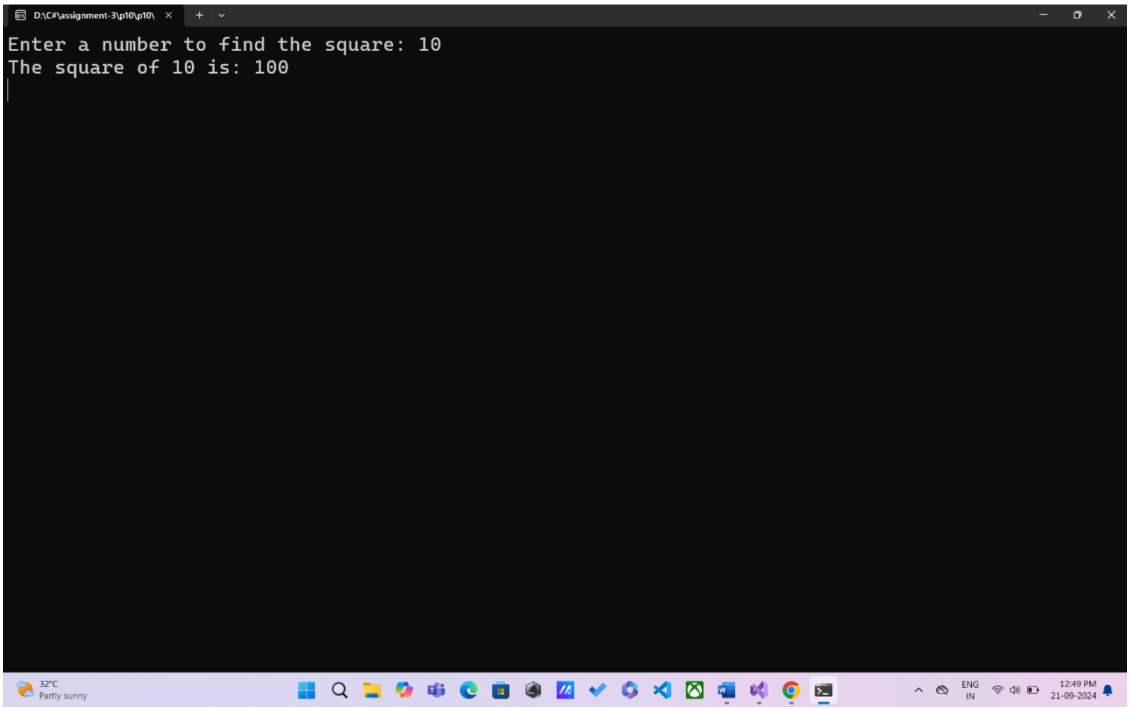
10) Write a program to create interface named test. In this interface the member function is square. Implement this interface in arithmetic class. Create one new class called ToTestInt in this class use the object of arithmetic class.

Program :

```
using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;


namespace ConsoleApp10

{

  interface Test
  {

    int Square(int num);

  }

  class Arithmetic : Test
  {

    public int Square(int num)
    {

      return num * num;

    }
}

  class ToTestInt
  {

    public void Execute()
    {

      Arithmetic arithmetic = new Arithmetic();


      Console.Write("Enter a number to find the square: ");

      int number = int.Parse(Console.ReadLine());
```

```
            int result = arithmetic.Square(number);

            Console.WriteLine($"The square of {number} is: {result}");

        }
}

    class Program
    {

        static void Main(string[] args)
        {

            ToTestInt testInt = new ToTestInt();

            testInt.Execute();

            Console.ReadKey();

        }

    }

}
```

Output :



```
Enter a number to find the square: 10
The square of 10 is: 100
```

11) Delegate program steps are as given below.

a. Create a class named as “Delegate_Demo” and declare static variable ‘num’ to retain its value for addition and multiplication operations.

b. Create a delegate named as “Number Changer” and declare globally.

C. Define methods to add and multiply numbers and name is ‘Addnum’ and ‘multnum’.

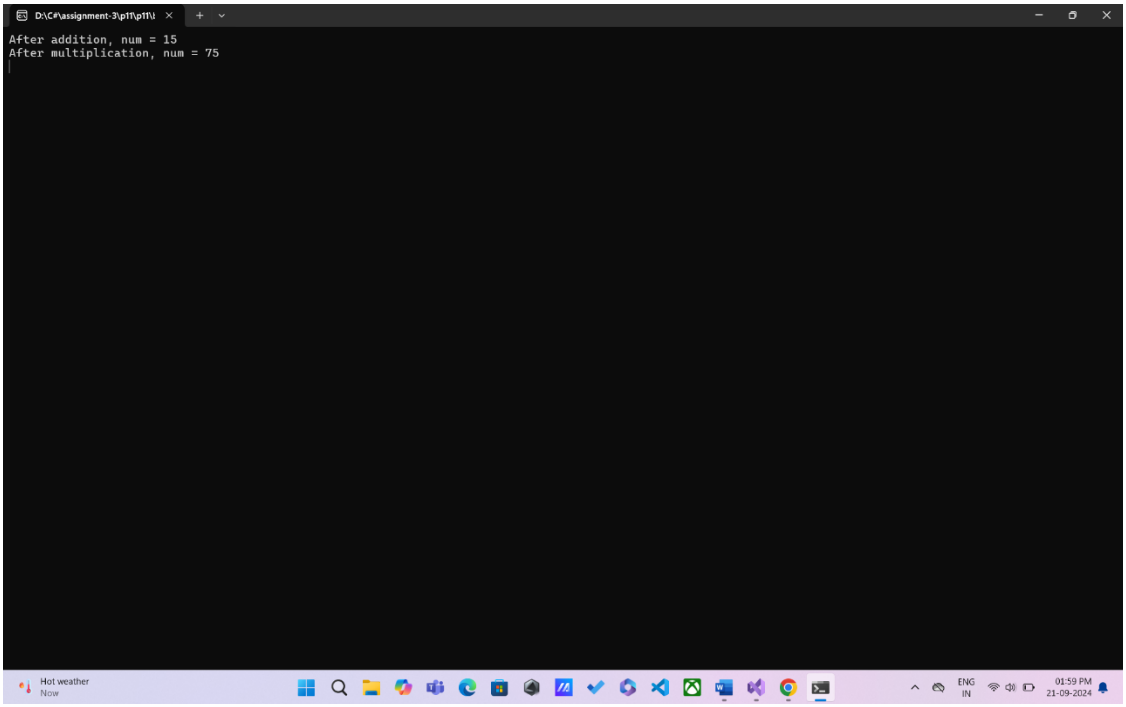
d. In main method, create two objects for delegates as nc1, nc2

e. addnum and multnum methods are passed as arguments within nc1 and nc2 Step 6: print the addition and multiplication result.

Program :

```
using System;

using System.Collections.Generic;

using System.Linq; using
System.Text; using
System.Threading.Tasks;


namespace ConsoleApp11

{

    public delegate void NumberChanger(int n);
    public class Delegate_Demo
    {

        private static int num = 10;
        public static void Addnum(int p)
        {

            num += p;

            Console.WriteLine("After addition, num = " + num);

        }


        public static void Multnum(int q)
        {

            num *= q;
            Console.WriteLine("After multiplication, num = " + num);
         }
         public static void Main(string[] args)
         {

            NumberChanger nc1 = new NumberChanger(Addnum);

            NumberChanger nc2 = new NumberChanger(Multnum);

            nc1(25);

            nc2(5);

            Console.ReadKey();

        } }

 }
```

Output :



```
After addition, num = 15
After multiplication, num = 75
```

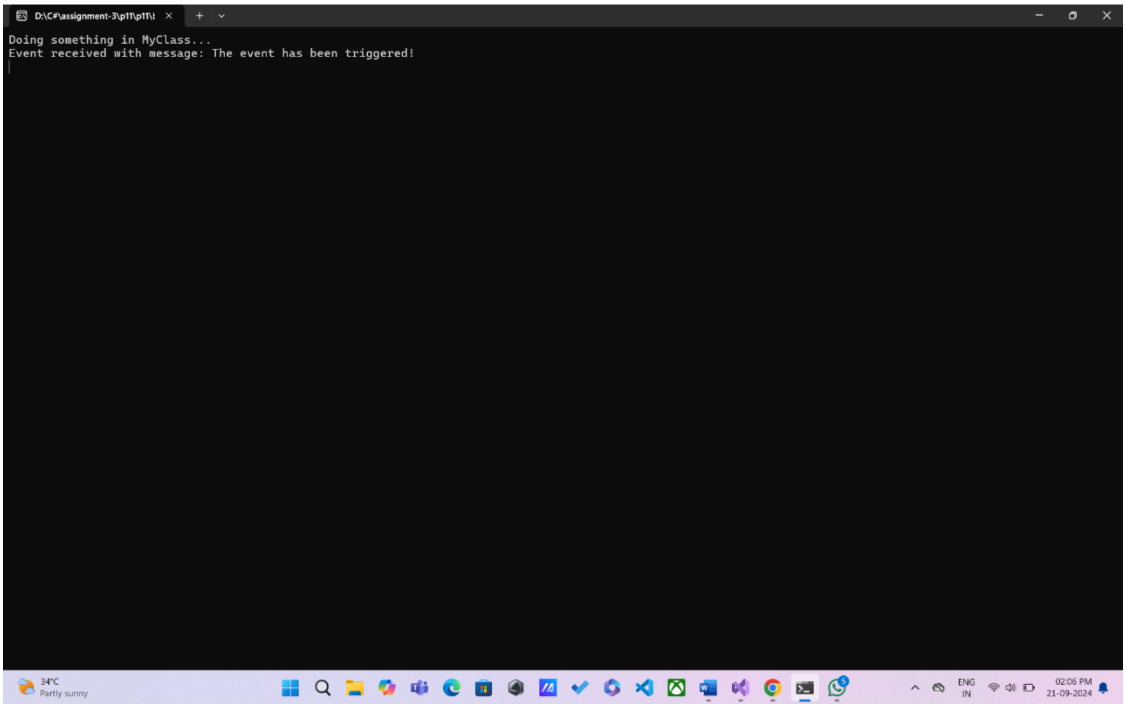
12) Write a program that creates an event MyEvent in the MyClass and subscribes to it using an instance of EventSubscriber. When DoSomething is called, the event is raised, and the event handler in EventSubscriber executes the event logic.

<u>Program :</u>

```
using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;
namespace ConsoleApp12

{

   public class MyEventArgs : EventArgs

   {

      public string Message { get; set; }
   }

   public class MyClass
   {

      public event EventHandler<MyEventArgs> MyEvent;

      public void DoSomething()

      {

         Console.WriteLine("Doing something in MyClass...");
         OnMyEvent(new MyEventArgs { Message = "The event has been triggered!" });
      }
      protected virtual void OnMyEvent(MyEventArgs e)

      {

         MyEvent?.Invoke(this, e);

      }
}

   public class EventSubscriber
   {

      public void OnMyEventReceived(object sender, MyEventArgs e)

      {

         Console.WriteLine($"Event received with message: {e.Message}");
```

```
        }
    }

      class Program
      {
          static void Main(string[] args)
          {
              MyClass myClassInstance = new MyClass();

              EventSubscriber subscriber = new EventSubscriber();

             myClassInstance.MyEvent += subscriber.OnMyEventReceived;

              myClassInstance.DoSomething();

              Console.ReadKey();

          }

      }

    }
```

Output :



```
Doing something in MyClass...
Event received with message: The event has been triggered!
```