

Name : Karan Udani

Enrollment No :220801359

Roll No : 25

Division :J3

Subject : C#.Net

Task -3

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

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

namespace T1
{
    public class swap
    {
        public void swapdata(int a, int b)
        {
            int temp = a;
            a = b;
            b = temp;
            Console.WriteLine("After Swapping Integer : {0},{1}" ,a,b);
        }
        public void swapdata(float a, float b)
        {
            float temp = a;
            a = b;
            b = temp;
            Console.WriteLine("After Swapping Float : {0} , {1}", a, b);
        }
    }


    class Program
    {
        static void Main(string[] args)
        {
            swap s1 = new swap();
            int p = 5;
            int q = 10;
```

```

        float x = 5.5f;
        float y = 10.5f;
        Console.WriteLine("Before Swapping Integer: {0},{1}", p, q);
        s1.swapdata(p, q);
        Console.WriteLine("-----");
        Console.WriteLine("Before Swapping Float : {0},{1}", x, y);
        s1.swapdata(x, y);
        Console.ReadKey();
    }
}

```

OutPut:



```

D:\D-SEM5\c#\Task-3\T1\T1b
Before Swapping Integer: 5 , 10
After Swapping Integer : 10 , 5
-----
Before Swapping Float : 5.5 , 10.5
After Swapping Float : 10.5 , 5.5

```

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

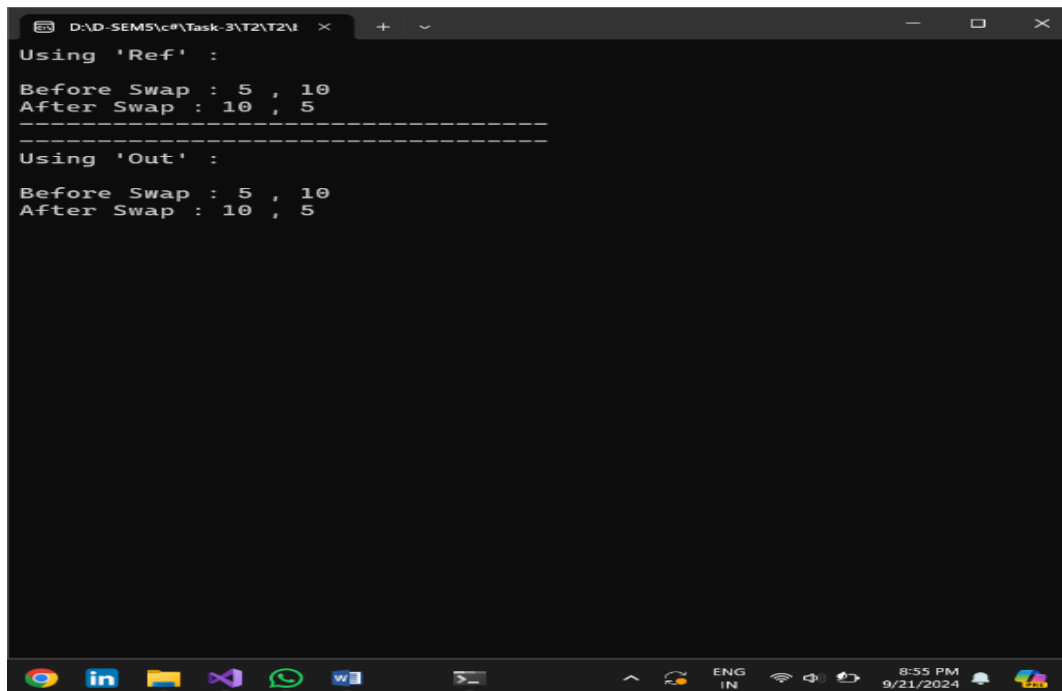
```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;

namespace T2
{
    class Program
    {
        static void swapdata(ref int a, ref int b)
        {
            int temp = a;
            a = b;
            b = temp;
        }

        static void swapdata(int c, int d, out int newc, out int newd)
        {
            newc = d;
            newd = c;
        }
        static void Main(string[] args)
        {
            int w = 5;
            int x = 10;
            int p = 5;
            int q = 10;
            int newp;
            int newq;
            Console.WriteLine("Using 'Ref' : ");
            Console.WriteLine("Before Swap : {0} , {1}", w, x);
            swapdata(ref w, ref x);
            Console.WriteLine("After Swap : {0} , {1}", w, x);
            Console.WriteLine("-----");
            Console.WriteLine("-----");
            Console.WriteLine("Using 'Out' : ");
            Console.WriteLine("Before Swap : {0} , {1}", p, q);
            swapdata(p, q, out newp, out newq);
            Console.WriteLine("After Swap : {0} , {1}", newp, newq);

            Console.ReadKey();
        }
    }
}
```

OutPut:



```
D:\D-SEM5\c#\Task-3\T2\T2\1
Using 'Ref' :
Before Swap : 5 , 10
After Swap : 10 , 5
-----
Using 'Out' :
Before Swap : 5 , 10
After Swap : 10 , 5
```

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 setdata() and displaydata().

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;
namespace T3
{
    public class room
    {
        public int roomno;
        public string roomtype;
        public string roomarea;
        public string ACmachine;
        public void setdata()
        {
            Console.Write("Enter Room No : ");
            roomno = int.Parse(Console.ReadLine());
            Console.Write("Enter Room Type : ");
            roomtype = Console.ReadLine();
            Console.Write("Enter Room Area (Square Meters) : ");
            roomarea = Console.ReadLine();
            Console.Write("Does the Room have ACmachine ? (Yes/No) : ");
            ACmachine = Console.ReadLine();
        }
    }
}
```

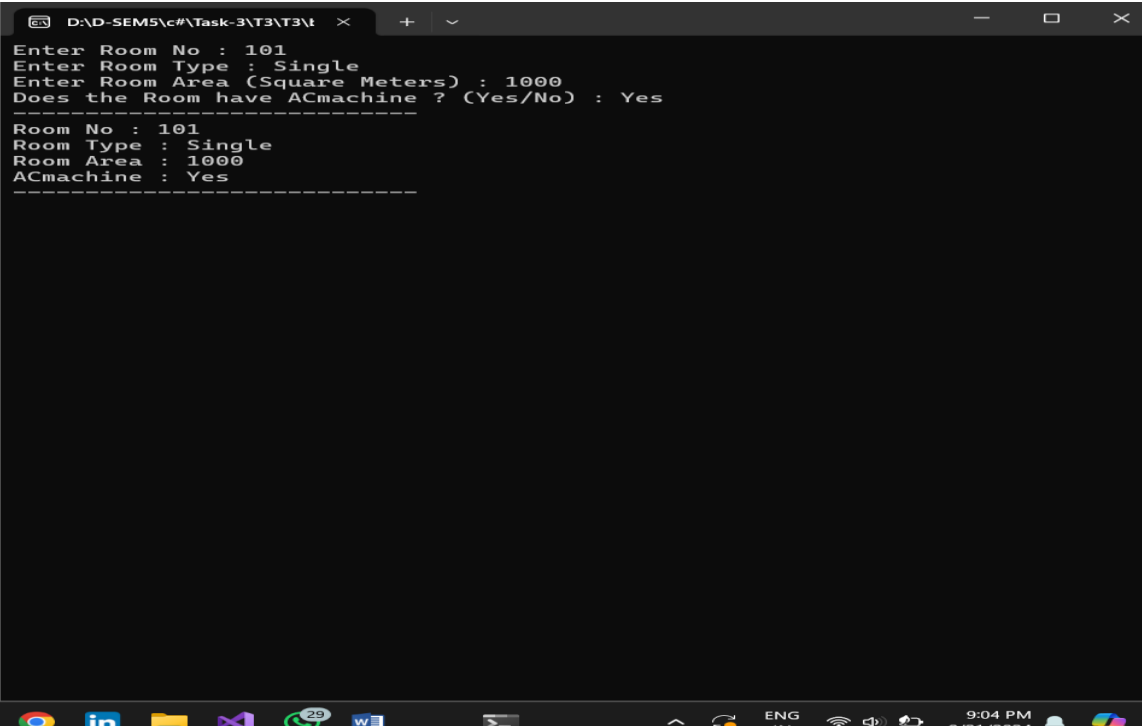
```

public void displaydata()
{
    Console.WriteLine("-----");
    Console.WriteLine("Room No : {0}", roomno);

    Console.WriteLine("Room Type : {0}", roomtype);
    Console.WriteLine("Room Area : {0}", roomarea);
    Console.WriteLine("ACmachine : {0}", ACmachine);
    Console.WriteLine("-----");
}
}
class Program
{
    static void Main(string[] args)
    {
        room r1 = new room();
        r1.setdata();
        r1.displaydata();
        Console.ReadKey();
    }
}

```

OutPut:



The screenshot shows a Windows desktop environment. At the top, a taskbar contains icons for Google Chrome, LinkedIn, File Explorer, Microsoft Teams, a green chat application with a '29' notification, Microsoft Word, and a folder icon. The main area of the screen is a dark-themed terminal window titled 'D:\D-SEM5\c#\Task-3\T3\T3\'. The terminal displays the following text:

```

Enter Room No : 101
Enter Room Type : Single
Enter Room Area (Square Meters) : 1000
Does the Room have ACmachine ? (Yes/No) : Yes
-----
Room No : 101
Room Type : Single
Room Area : 1000
ACmachine : Yes
-----

```

At the bottom of the screen, the Windows taskbar shows the system clock as 9:04 PM on 9/21/2024, along with icons for network, volume, and battery status.

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

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

namespace T4
{
    abstract class Shape
    {
        public abstract void Draw();
        public abstract void Erase();
    }
    class Circle : Shape
    {
        public override void Draw()
        {
            Console.WriteLine("\nDrawing a Circle");
        }

        public override void Erase()
        {
            Console.WriteLine("Erasing a Circle");
        }
    }
    class Triangle : Shape
    {
        public override void Draw()
        {
            Console.WriteLine("\nDrawing a Triangle");
        }

        public override void Erase()
        {
            Console.WriteLine("Erasing a Triangle");
        }
    }

    class Square : Shape
    {
        public override void Draw()
        {
            Console.WriteLine("\nDrawing a Square");
        }

        public override void Erase()
        {
            Console.WriteLine("Erasing a Square");
        }
    }
}
```

```

internal class Program
{
    static void Main(string[] args)
    {
        Console.Write("Select a shape[1:Circle,2:Triangle,3:Square] : ");
        int choice = int.Parse(Console.ReadLine());

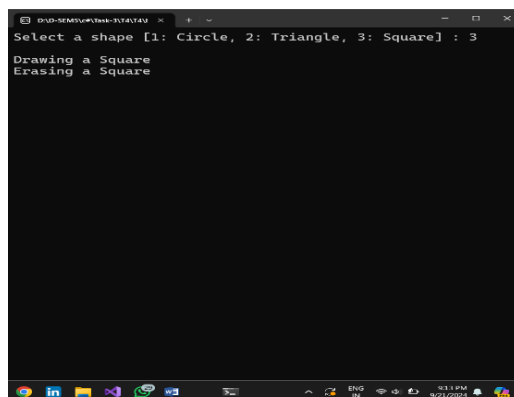
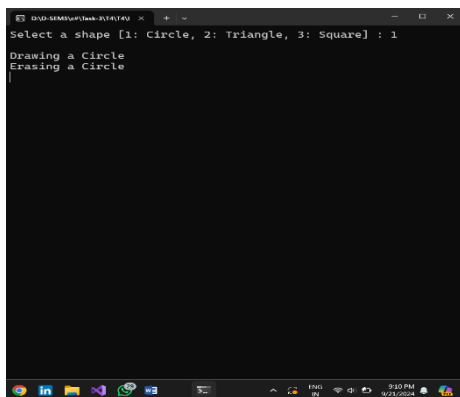
        Shape shape = null;

        switch (choice)
        {
            case 1:
                shape = new Circle();
                break;
            case 2:
                shape = new Triangle();
                break;
            case 3:
                shape = new Square();
                break;
            default:
                Console.WriteLine("Invalid choice.");
                return;
        }

        shape.Draw();
        shape.Erase();
        Console.ReadLine();
    }
}

```

OutPut:

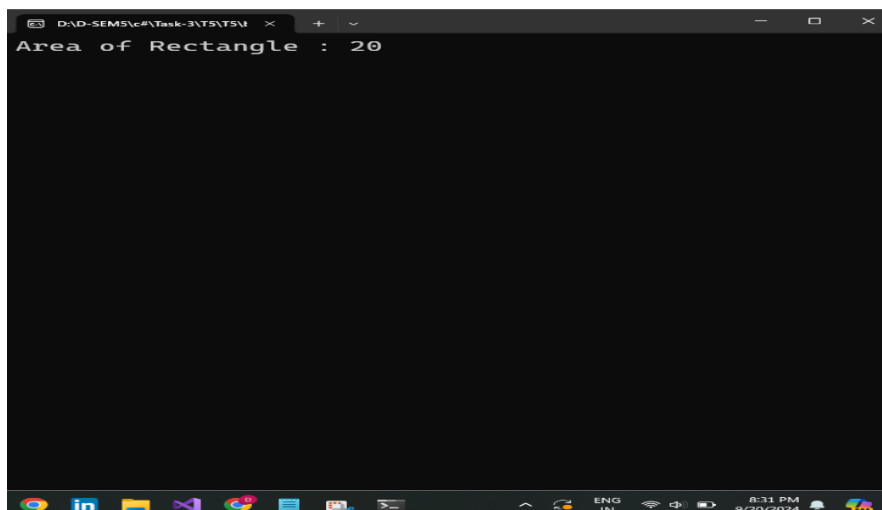


5. Create a class name Rectangle and declare necessary variables (length, width, area) and method (Find _Area), create a subclass named “Tabletop” to inherit Rectangle class, create another class named Execute Rectangle and create objects for derived class and execute the method.

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;
namespace T5
{
    public class Rectangle
    {
        public int length=5;
        public int width=4;
        public int area;
        public void Find_Area()
        {
            Console.WriteLine("Area of Rectangle : {0}",length*width);
        }
    }
    public class Tabletop : Rectangle
    {
    }

    class Program
    {
        static void Main(string[] args)
        {
            Tabletop t1 = new Tabletop();
            t1.Find_Area();
            Console.ReadKey();
        }
    }
}
```

OutPut:



6. Create a class name "Person" and declare necessary variables (Roll No., Name) and methods (Public void Get_Data (), Public virtual Display_Data()), 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.

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

namespace T6
{
    public class Person
    {
        public int Rollno;
        public string Name;
        public virtual void getdata()
        {
            Console.Write("Enter Roll No : ");
            Rollno = int.Parse(Console.ReadLine());
            Console.Write("Enter Name : ");
            Name = Console.ReadLine();
        }
        public virtual void displaydata()
        {
            Console.WriteLine("-----");
            Console.WriteLine("Roll no: {0}", Rollno);
            Console.WriteLine("Name: {0}", Name);
        }
    }
    public class Student : Person
    {
        public string course;
        public override void getdata()
        {
            base.getdata();
            Console.Write("Enter Course name: ");
            course = Console.ReadLine();
        }
        public override void displaydata()
        {
            base.displaydata();
            Console.WriteLine("Course Name: {0}", course);
        }
    }
    public class Details : Student
    {

```

```

        public string department;
        public override void getdata()
        {

            base.getdata();
            Console.Write("Enter Department name: ");
            department = Console.ReadLine();
        }
        public override void displaydata()
        {
            base.displaydata();
            Console.WriteLine("Department Name: {0}", department);
            Console.WriteLine("-----");
        }
    }

    class TestClass
    {
        static void Main(string[] args)
        {
            Details detail = new Details();
            detail.getdata();
            detail.displaydata();
            Console.ReadKey();
        }
    }
}

```

OutPut:

```

D:\D-SEM5\c#\Task-3\T6\T6.t
Enter Roll No : 25
Enter Name : Karan
Enter Course name: B.C.A
Enter Department name: CS & IT
-----
Roll no: 25
Name: Karan
Course Name: B.C.A
Department Name: CS & IT
-----

```

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.

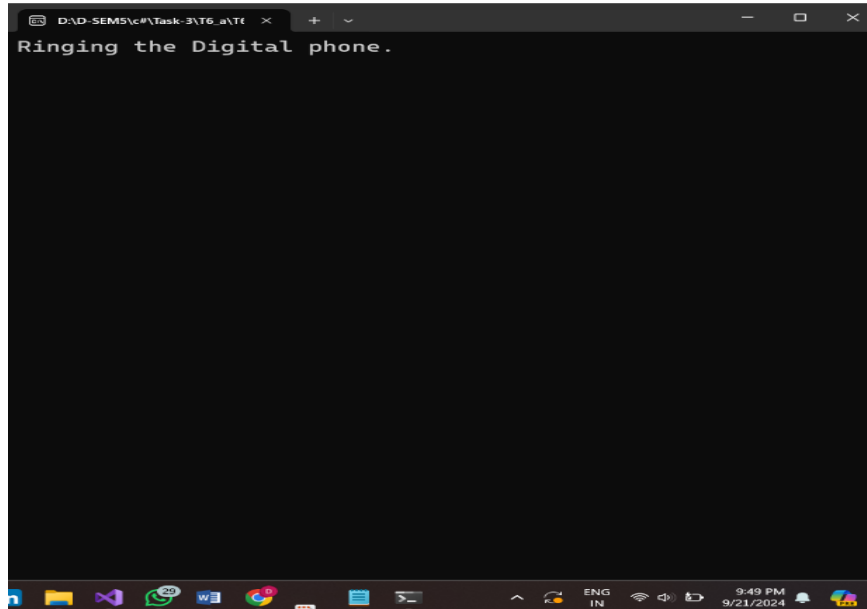
```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;

namespace T7_a
{
    class Telephone
    {
        protected string phonetype;

        public void Ring()
        {
            Console.WriteLine($"Ringing the {phonetype} phone.");
        }
    }
    class ElectronicPhone : Telephone
    {
        public ElectronicPhone()
        {
            phonetype = "Digital";
        }
        public void Run()
        {
            Ring();
        }
    }
    internal class Program
    {
        static void Main(string[] args)
        {
            ElectronicPhone ePhone = new ElectronicPhone();
            ePhone.Run();
            Console.ReadKey();
        }
    }
}
```

```
}
```

OutPut:



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

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

namespace T7_b
{
    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 with electronic sound.");
        }
        public void Run()
        {
            Ring();
        }
    }
}
```

```

    }
}
internal class Program
{
    static void Main(string[] args)
    {
        Telephone basicPhone = new Telephone();
        basicPhone.Ring();

        ElectronicPhone ePhone = new ElectronicPhone();
        ePhone.Run();

        Telephone polymorphicPhone = new ElectronicPhone();
        polymorphicPhone.Ring();

        Console.ReadKey();
    }
}
}

```

OutPut:

```

D:\C#\p6-btp6-b\bin\Debug\
Ringing the phone.
Ringing the Digital phone with electronic sound.
Ringing the Digital phone with electronic 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

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

namespace T8
{
    class Program
    {

        class Employee
        {
            private static int count = 0;
            private string Emp_Id, Emp_Name;
            private double Basic_Salary, Gross_Salary;
            public Employee(string empName, double basicSalary)
            {
                Emp_Id = Convert.ToString(++count);
                Emp_Name = empName;
                Basic_Salary = basicSalary;
                Calculate_Gross_Salary();
            }
            public string this[int index]
            {
                set
                {
                    switch (index)
                    {
                        case 0:
                            Emp_Name = value;
                            break;
                        case 1:
                            Basic_Salary = Convert.ToDouble(value);
                            break;
                    }
                }
            }
        }
    }
}
```

```

        case 2:
            Gross_Salary = Convert.ToDouble(value);
            break;
        default:
            throw new ArgumentOutOfRangeException("index");
    }
}
get
{
    switch (index)
    {
        case 0:
            return Emp_Id;
            break;
        case 1:
            return Emp_Name;
            break;
        case 2:
            return Basic_Salary.ToString("F2");
            break;
        case 3:
            return Gross_Salary.ToString("F2");
            break;
        default:
            throw new ArgumentOutOfRangeException("index");
    }
}
}
public void Calculate_Gross_Salary()
{
    Gross_Salary = Basic_Salary + (0.20 * Basic_Salary);
}
public void Display_Data()
{
    System.Console.WriteLine("-----");
    System.Console.WriteLine("Emp_Id : " + Emp_Id);
    System.Console.WriteLine("Emp_Name : " + Emp_Name);
    System.Console.WriteLine("Basic Salary : " + Basic_Salary);
    System.Console.WriteLine("Gross Salary : " + Gross_Salary);
    System.Console.WriteLine("-----");
}
public void Update_Data(string newName, double newSalary)
{
    this.Emp_Name = newName;
    this.Basic_Salary = newSalary;
    Calculate_Gross_Salary();
}

public string GetEmployeeID() => Emp_Id;
public string searchId() => Emp_Id;
public string GetEmployeeName() => Emp_Name;
}
static void Main(string[] args)
{
    int ch = 0;
    Employee emp = null;

```

```

bool exit = false;
List<Employee> employees = new List<Employee>();

while (!exit)
{
    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("-----");
    Console.Write("Enter Choice : ");
    ch = int.Parse(Console.ReadLine());
    switch (ch)
    {
        case 1:
            Console.Write("\nEnter Employee Name: ");
            string name = Console.ReadLine();

            Console.Write("Enter Basic Salary: ");
            double salary = double.Parse(Console.ReadLine());

            employees.Add(new Employee(name, salary));
            Console.Write("\n");
            Console.WriteLine("Employee added successfully.");
            Console.WriteLine("-----");
            break;
        case 2:
            if (employees.Count > 0)
            {
                foreach (Employee emps in employees)
                {
                    emps.Display_Data();
                }
            }
            else
            {
                Console.WriteLine("\nNo employees to display.");
            }
            break;
        case 3:
            Console.Write("\nEnter Employee ID : ");
            string searchId = Console.ReadLine();
            Employee empById = employees.Find(emps=>
            emps.GetEmployeeID() == searchId);

            if (empById != null)
            {
                empById.Display_Data();
            }
            else
            {
                Console.WriteLine("\nNot found....");
            }
    }
}

```



```

        break;
    case 4:
        Console.Write("\nEnter Employee Name : ");
        string searchName = Console.ReadLine().ToLower();

        Employee empByName = employees.Find(emps =>
            emps.GetEmployeeName().ToLower() == searchName);

        if (empByName != null)
        {
            empByName.Display_Data();
        }
        else
        {
            Console.WriteLine("\nNot found.");
        }

        break;
    case 5:
        Console.Write("\nEnter Employee ID to update: ");
        string updateId = Console.ReadLine();

        Employee empToUpdate = employees.FirstOrDefault(emps =>
            emps.GetEmployeeID() == updateId);

        if (empToUpdate != null)
        {
            Console.Write("Enter new Employee Name: ");
            string newName = Console.ReadLine();

            Console.Write("Enter new Basic Salary: ");
            if (double.TryParse(Console.ReadLine(), out
                double newSalary))
            {
                empToUpdate.Update_Data(newName, newSalary);
                Console.WriteLine("\nUpdated successfully.");
            }
            else
            {
                Console.WriteLine("\nInvalid...");
            }
        }
        else
        {
            Console.WriteLine("\nEmployee Not found.");
        }

        Console.WriteLine("-----");
        break;
    case 6:
        Console.Write("\nEnter Employee ID to delete: ");
        string deleteId = Console.ReadLine();
        Employee empToDelete = employees.Find(emps =>
            emps.GetEmployeeID() == deleteId);

```

```

        if (empToDelete != null)
        {
            employees.Remove(empToDelete);

            Console.WriteLine("\nEmployee deleted
            successfully.");
        }
        else
        {

            Console.WriteLine("\nEmployee with the given ID not
            found.");
        }
        Console.WriteLine("-----");
        break;
    case 7:
        ch = false;
        break;
    default:
        System.Console.WriteLine("\nInvalid Input....");

        break;
    }
}
}
}
}
}

```

OutPut:

ADD DATA :

```

D:\D-SEM5\c#\Task-3\T8\T8\
1.Add Data
2.Display Data
3.Search By Id
4.Search By Name
5.Update By Id
6.Delete By Id
7.Exit
-----
Enter Choice : 1

Enter Employee Name: Neha
Enter Basic Salary: 23000

Employee added successfully.
-----

```

DISPLAY DATA :

```
1.Add Data
2.Display Data
3.Search By Id
4.Search By Name
5.Update By Id
6.Delete By Id
7.Exit
-----
Enter Choice : 2
-----
Emp_Id : 1
Emp_Name : Neha
Basic Salary : 23000
Gross Salary : 27600
-----
Emp_Id : 2
Emp_Name : Ridham
Basic Salary : 45000
Gross Salary : 54000
-----
Emp_Id : 3
Emp_Name : Sidhdhi
Basic Salary : 23000
Gross Salary : 27600
-----
```

SEARCH BY ID :

```
1.Add Data
2.Display Data
3.Search By Id
4.Search By Name
5.Update By Id
6.Delete By Id
7.Exit
-----
Enter Choice : 3
-----
Enter Employee ID : 3
-----
Emp_Id : 3
Emp_Name : Sidhdhi
Basic Salary : 23000
Gross Salary : 27600
-----
```

SEARCH BY NAME :

```
1.Add Data
2.Display Data
3.Search By Id
4.Search By Name
5.Update By Id
6.Delete By Id
7.Exit
-----
Enter Choice : 4

Enter Employee Name : Ridham
-----
Emp_Id : 2
Emp_Name : Ridham
Basic Salary : 45000
Gross Salary : 54000
-----
```

UPDATE BY ID :

```
D:\D-SEM5\c#\Task-3\T8\T8\t  × + ▾
1.Add Data
2.Display Data
3.Search By Id
4.Search By Name
5.Update By Id
6.Delete By Id
7.Exit
-----
Enter Choice : 5

Enter Employee ID to update: 2
Enter new Employee Name: Jiya
Enter new Basic Salary: 56000

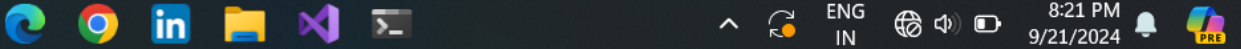
Updated successfully.
-----
1.Add Data
2.Display Data
3.Search By Id
4.Search By Name
5.Update By Id
6.Delete By Id
7.Exit
-----
Enter Choice : 2
-----
Emp_Id : 1
Emp_Name : Neha
Basic Salary : 23000
Gross Salary : 27600
-----
Emp_Id : 2
Emp_Name : Jiya
Basic Salary : 56000
Gross Salary : 67200
-----
Emp_Id : 3
Emp_Name : Sidhdhi
Basic Salary : 23000
```

DELETE BY ID :

```
1.Add Data
2.Display Data
3.Search By Id
4.Search By Name
5.Update By Id
6.Delete By Id
7.Exit
-----
Enter Choice : 6

Enter Employee ID to delete: 1

Employee deleted successfully.
-----
1.Add Data
2.Display Data
3.Search By Id
4.Search By Name
5.Update By Id
6.Delete By Id
7.Exit
-----
Enter Choice : 2
-----
Emp_Id : 2
Emp_Name : Jiya
Basic Salary : 56000
Gross Salary : 67200
-----
Emp_Id : 3
Emp_Name : Sidhdhi
Basic Salary : 23000
Gross Salary : 27600
-----
```



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

(a). Add Data

(b). Display Data

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

```
namespace T9
{
```

```

class Program
{
    class Product
    {
        private static int count = 0;
        private string cust_id, cust_name, product_name, price;

        public Product()
        {
            cust_id = Convert.ToString(++count);
        }

        public string this[int index]
        {
            set
            {
                switch (index)
                {
                    case 0:
                        cust_name = value;
                        break;
                    case 1:
                        product_name = value;
                        break;

                    case 2:
                        price = value;
                        break;
                    default:
                        throw new ArgumentOutOfRangeException("index");
                }
            }
            get
            {
                switch (index)
                {
                    case 0:
                        return cust_id;
                    case 1:
                        return cust_name;
                    case 2:
                        return product_name;
                    case 3:
                        return price;
                    default:
                        throw new ArgumentOutOfRangeException("index");
                }
            }
        }

        public void Show_products()
        {
            System.Console.WriteLine("-----");
            System.Console.WriteLine("Cust_id:" + cust_id);
            System.Console.WriteLine("Cust_name:" + cust_name);
        }
    }
}

```

```

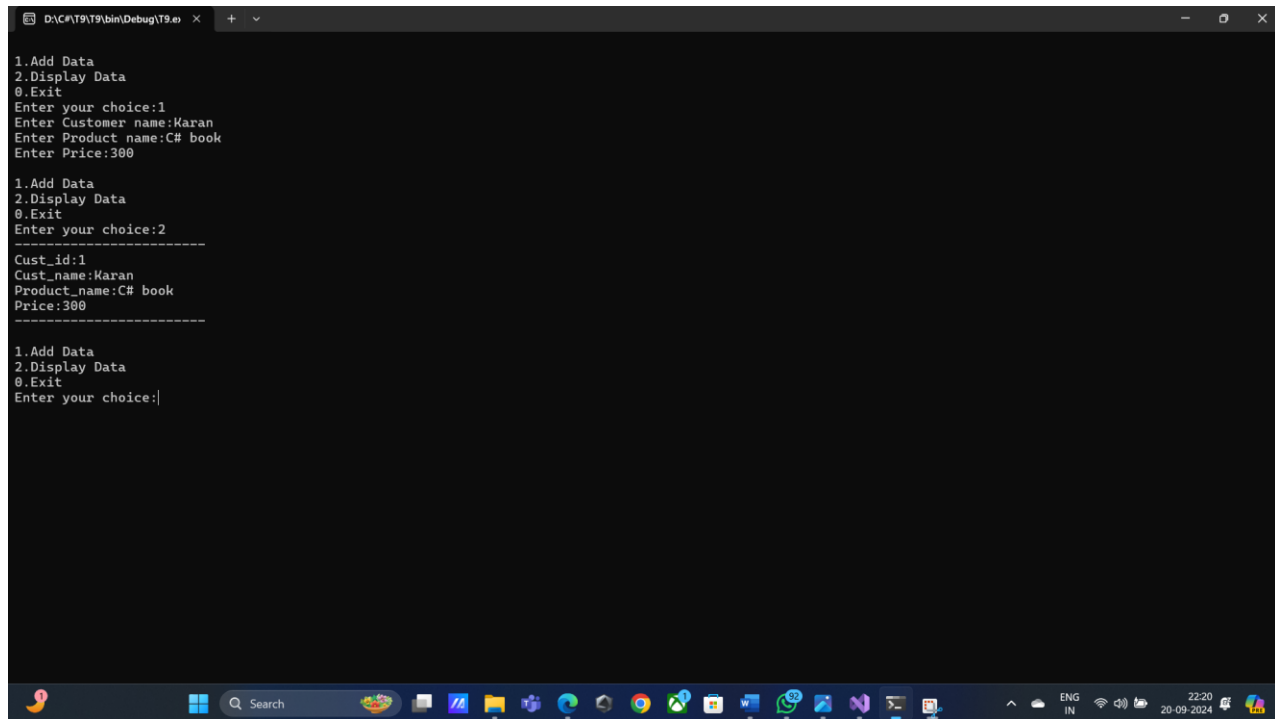
        System.Console.WriteLine("Product_name:" + product_name);
        System.Console.WriteLine("Price:" + price);
        System.Console.WriteLine("-----");
    }
}
static void Main(string[] args)
{
    int ch = 0;
    Product prd = null;
    List<Product> products = new List<Product>();

    while (true)
    {
        System.Console.WriteLine("\n1.Add Data");
        System.Console.WriteLine("2.Display Data");

        System.Console.WriteLine("0.Exit");
        System.Console.Write("Enter your choice:");
        ch = int.Parse(Console.ReadLine());
        switch (ch)
        {
            case 1:
                prd = new Product();
                System.Console.Write("Enter Customer name:");
                prd[0] = Console.ReadLine();
                System.Console.Write("Enter Product name:");
                prd[1] = Console.ReadLine();
                System.Console.Write("Enter Price:");
                prd[2] = Console.ReadLine();
                products.Add(prd);
                break;
            case 2:
                for (int i = 0; i < products.Count; i++)
                {
                    products[i].Show_products();
                }
                break;
            case 0:
                Environment.Exit(0);
                break;
            default:
                System.Console.WriteLine("Invalid input...");
                break
                Console.ReadLine();
        }
    }
}
}

```

OutPut:



```
D:\C#\T9\bin\Debug\T9.exe
1.Add Data
2.Display Data
0.Exit
Enter your choice:1
Enter Customer name:Karan
Enter Product name:C# book
Enter Price:300

1.Add Data
2.Display Data
0.Exit
Enter your choice:2
-----
Cust_id:1
Cust_name:Karan
Product_name:C# book
Price:300
-----

1.Add Data
2.Display Data
0.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 .

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

namespace T10
{
    public interface test
    {
        int square(int num);
    }
    public class arithmetic : test
    {
        public int square (int num)
        {
            return num * num;
        }
    }
    class Program
    {
        static void Main(string[] args)
        {
```



```

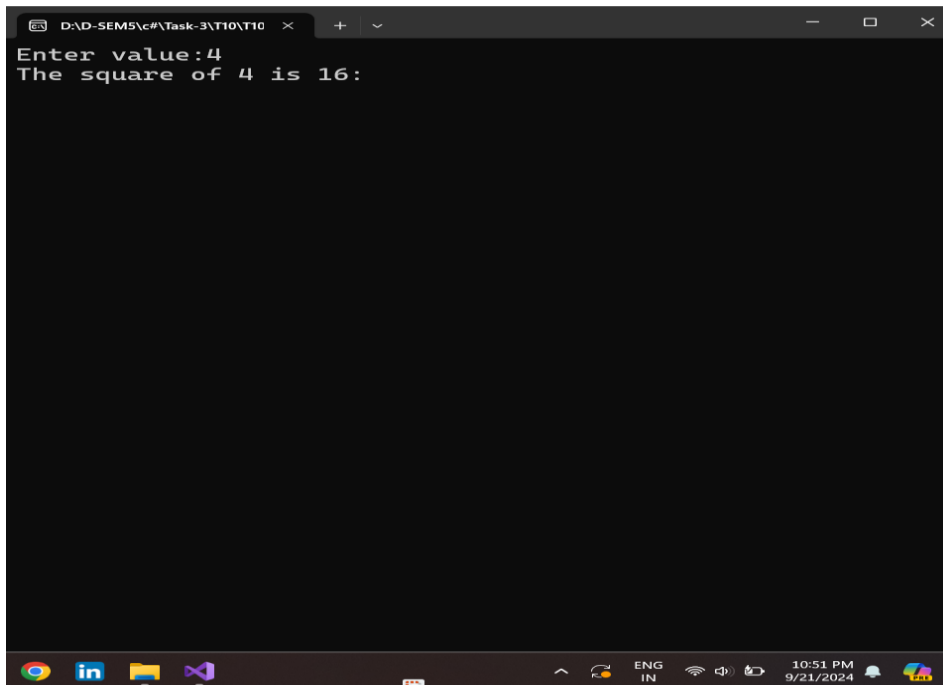
        test t1 = new arithmetic();
        Console.Write("Enter value:");
        int a = int.Parse(Console.ReadLine());

        Console.WriteLine("The square of {0} is {1}:",a,t1.square(a));
        Console.ReadKey();

    }
}

```

OutPut:



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 it as '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.

```

using System;
using System.Collections.Generic;
using System.Linq;
using System.Security.Cryptography;
using System.Text;
using System.Threading.Tasks;

namespace T11
{
    class Delegate_Demo
    {
        public delegate void Number_Changer(int num1, int num2);
        public void Addnum(int num1, int num2)
        {
            int add = num1 + num2;
            Console.WriteLine("Addition of "+num1+ "and " + num2 + ":" + add);
        }
        public void multnum(int num1, int num2)
        {
            int multi = num1 * num2;
            Console.WriteLine("Multiplication of"+num1+"and"+num2+":" + multi);
        }

        internal class Program
        {
            static void Main(string[] args)
            {
                Delegate_Demo obj = new Delegate_Demo();

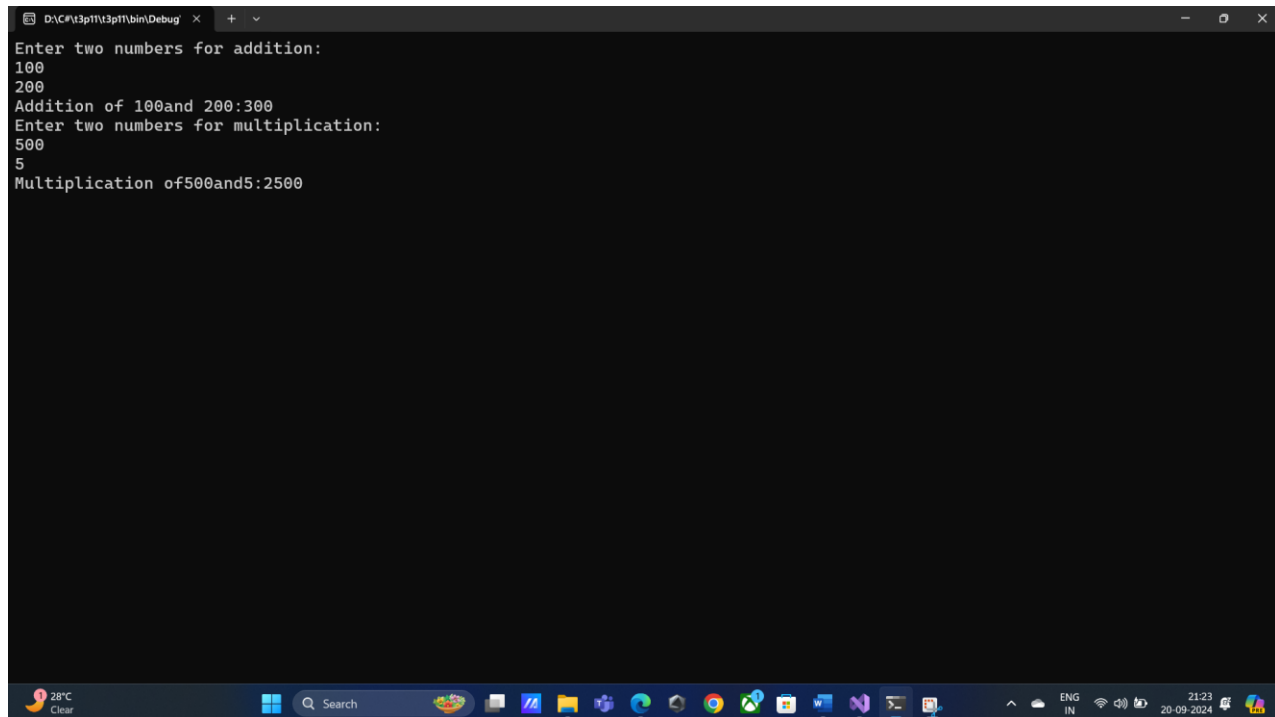
                Number_Changer nc1 = new Number_Changer(obj.Addnum);
                Number_Changer nc2 = new Number_Changer(obj.multnum);

                Console.WriteLine("Enter two numbers for addition:");
                int addNum1 = int.Parse(Console.ReadLine());
                int addNum2 = int.Parse(Console.ReadLine());
                nc1.Invoke(addNum1, addNum2);

                Console.WriteLine("Enter two numbers for multiplication:");
                int multNum1 = int.Parse(Console.ReadLine());
                int multNum2 = int.Parse(Console.ReadLine());
                nc2.Invoke(multNum1, multNum2);
                Console.ReadLine();
            }
        }
    }
}

```

OutPut:



```
D:\C#\t3p1\13p1\bin\Debug
Enter two numbers for addition:
100
200
Addition of 100and 200:300
Enter two numbers for multiplication:
500
5
Multiplication of 500and5:2500
```

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.

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

namespace T12
{
    public delegate void Myevent(object sender, EventArgs e);

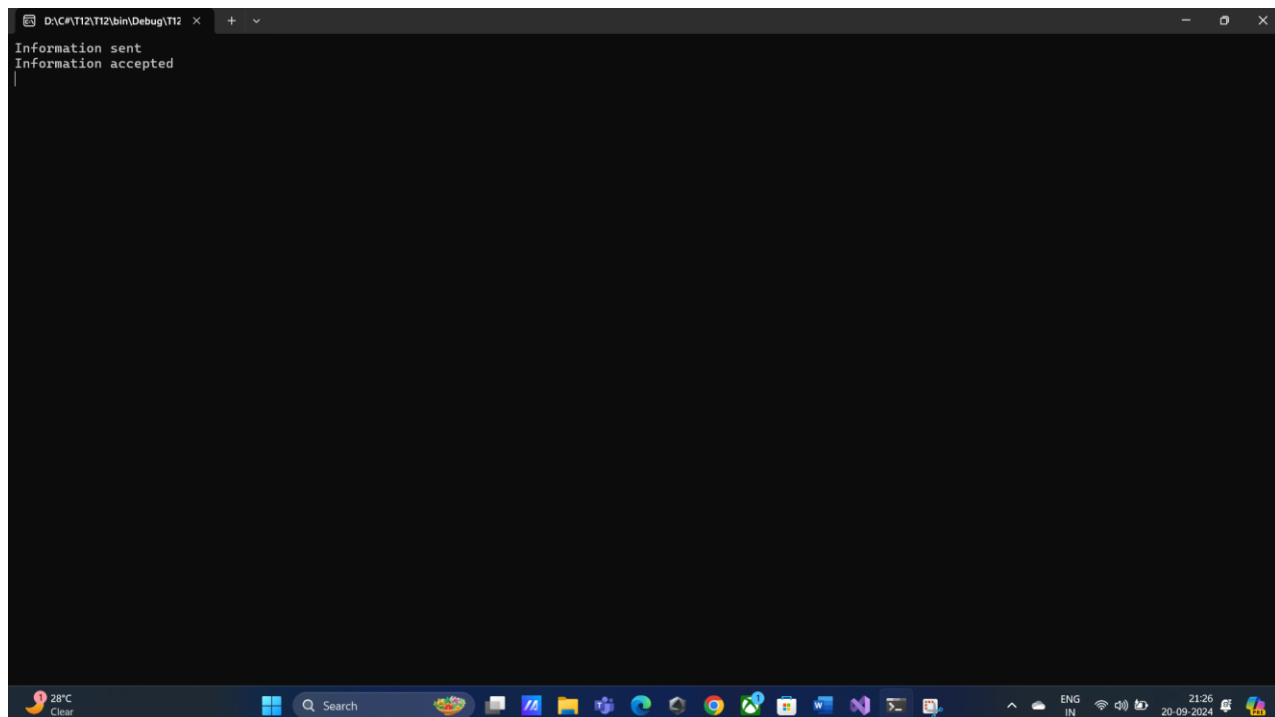
    public class MyClass
    {
        public event Myevent notify;
        public void publisher()
        {
            Console.WriteLine("Information sent");
            notify?.Invoke(this, EventArgs.Empty);
        }
    }

    public class Eventsubscriber
    {

```

```
        public void subscriber(object sender, EventArgs e)
        {
            Console.WriteLine("Information accepted");
        }
    }
}
class Program
{
    static void Main(string[] args)
    {
        Myclass m1 = new Myclass();
        Eventssubscriber e1 = new Eventssubscriber();
        m1.notify += e1.subscriber;
        m1.publisher();
    }
}
```

OutPut:



```
D:\C#\T12\T12\bin\Debug\T12
Information sent
Information accepted
```