TAAZAA TRAINING

Assignment- Day 8

Submitted By :- Gurpreet Singh

1) List Manipulation

Source Code :-

Contact.cs

```
namespace Day8.Models
{
    public class Contact
    {
        public int cId
        {
            get;
            set;
        }

        public string cName
        {
            get;
            set;
        }

        public string cLocation
        {
            get;
            set;
        }
        public long cPhNo
        {
            get;
            set;
        }
        public string eMail
        {
            get;
            set;
        }
    }
```

```
}
```

ContactList.cs

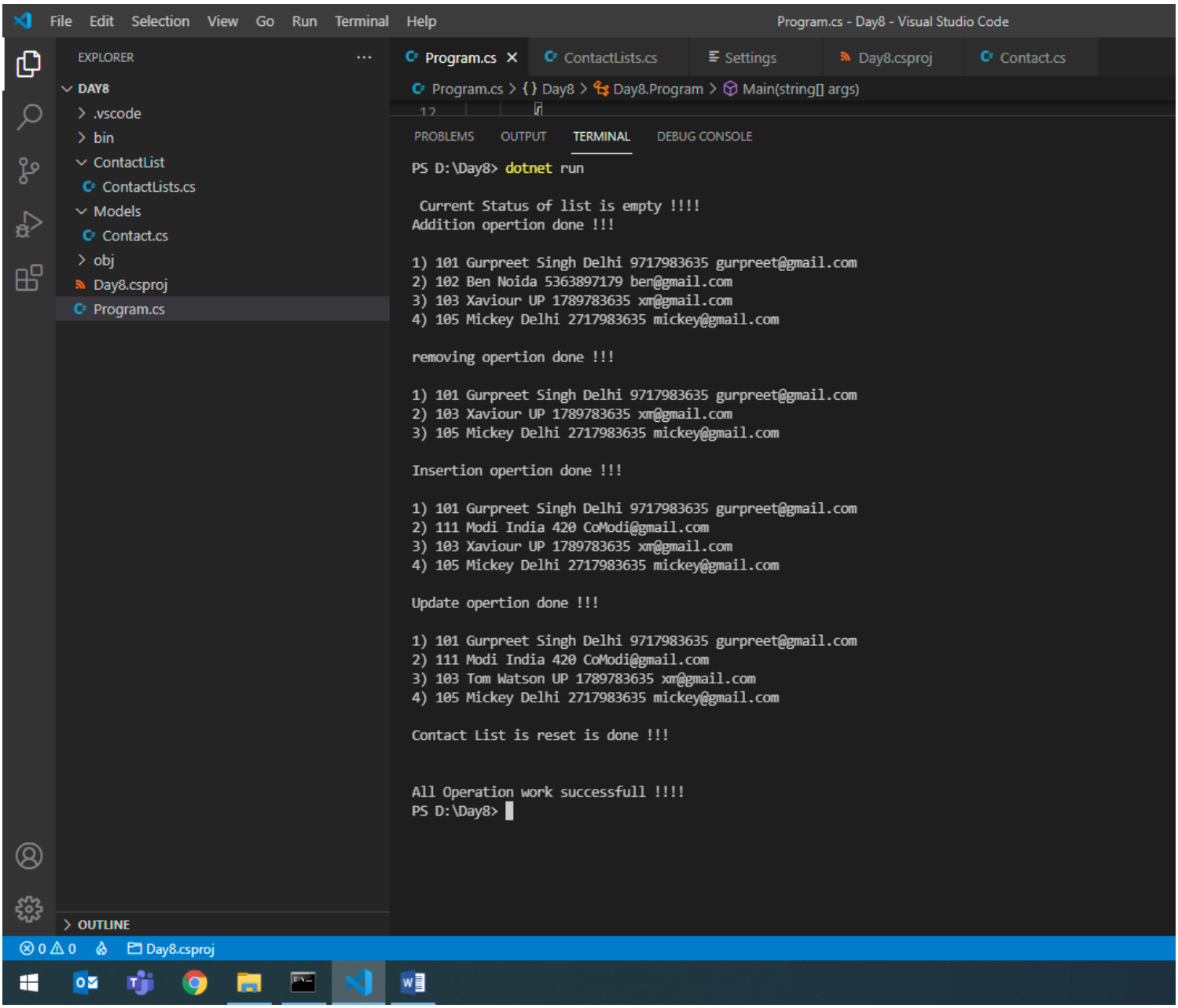
```
using Day8.Models;
using System;
using System.Collections.Generic;
namespace Day8.ContactList
{
    public class ContactLists
    {
        List<Contact> obj; // Not Yet Memory is allocated it is an instance variable
        public ContactLists() // if object of class is created then memory allocation will take p
lace from list it is constructor
        {
            obj=new List<Contact>();
        }
        public void createContact()// it is member function for list creation
        {
            obj.Add(new Contact{
                cId=101,
                cName="Gurpreet Singh",
                cPhNo=9717983635,
                eMail="gurpreet@gmail.com",
                cLocation="Delhi"

            });
            obj.Add(new Contact{
                cId=102,
                cName="Ben",
                cPhNo=5363897179,
                eMail="ben@gmail.com",
                cLocation="Noida"

            });
            obj.Add(new Contact{
                cId=103,
                cName="Xaviour",
                cPhNo=1789783635,
                eMail="xm@gmail.com",
                cLocation="UP"

            });
            obj.Add(new Contact{
                cId=105,
                cName="Mickey",
                cPhNo=2717983635,
                eMail="mickey@gmail.com",
                cLocation="Delhi"

            });
```

```
            Console.WriteLine("Addition opertion done !!! ");
        }

        public void removalContact() // to remove element
        {
            obj.RemoveAt(1);
            Console.WriteLine("removing opertion done !!! ");
        }
        public void insertContact()
        {
            List<Contact> obj2=new List<Contact>();
            obj2.Add(new Contact{
                cId=111,
                cLocation="India",
                cPhNo=420,
                cName="Modi",
                eMail="CoModi@gmail.com"
            });
            obj.InsertRange(1,obj2);
            Console.WriteLine("Insertion opertion done !!! ");
        }
        public void updateContact()
        {
            var temp=obj[2];
            temp.cName="Tom Watson";
            Console.WriteLine("Update opertion done !!! ");
        }

        public void truncateContactList()
        {
            obj.Clear();
            Console.WriteLine("Contact List is reset is done !!! ");
        }
        public void displayContactList(){
            var n=0;
            foreach(var i in obj)
            {
                n++;
                Console.WriteLine(n+") "+i.cId+" "+i.cName+" "+i.cLocation+" "+i.cPhNo+" "+i.eMai
l);
            }
        }
    }
}
```

Program.cs

```
using System;
using Day8.ContactList;
using System.Collections.Generic;
using Day8.Models;

namespace Day8
{
    class Program
    {

        static void Main(string[] args)
        {
            ContactLists obj=new ContactLists();
            //var temp=obj.createContact();
            Console.WriteLine("\n Current Status of list is empty !!!!");
            obj.createContact();
            Console.WriteLine();
            obj.displayContactList();
            Console.WriteLine();
            obj.removalContact();
            Console.WriteLine();
            obj.displayContactList();
            Console.WriteLine();
            obj.insertContact();
            Console.WriteLine();
            obj.displayContactList();
            Console.WriteLine();
            obj.updateContact();
            Console.WriteLine();
            obj.displayContactList();
            Console.WriteLine();
            obj.truncateContactList();
            Console.WriteLine();
            obj.displayContactList();
            Console.WriteLine();
            Console.WriteLine("All Operation work successfull !!!!");



        }
    }
}
```

Output :-


File Edit Selection View Go Run Terminal Help
Program.cs - Day8 - Visual Studio Code
EXPLORER
DAY8
.vscode
bin
ContactList
ContactLists.cs
Models
Contact.cs
obj
Day8.csproj
Program.cs
OUTLINE
Program.cs
ContactLists.cs
Settings
Day8.csproj
Contact.cs
Program.cs > {} Day8 > Day8.Program > Main(string[] args)
PROBLEMS
OUTPUT
TERMINAL
DEBUG CONSOLE
PS D:\Day8> dotnet run
Current Status of list is empty !!!!
Addition opertion done !!!
1) 101 Gurpreet Singh Delhi 9717983635 gurpreet@gmail.com
2) 102 Ben Noida 5363897179 ben@gmail.com
3) 103 Xaviour UP 1789783635 xm@gmail.com
4) 105 Mickey Delhi 2717983635 mickey@gmail.com
removing opertion done !!!
1) 101 Gurpreet Singh Delhi 9717983635 gurpreet@gmail.com
2) 103 Xaviour UP 1789783635 xm@gmail.com
3) 105 Mickey Delhi 2717983635 mickey@gmail.com
Insertion opertion done !!!
1) 101 Gurpreet Singh Delhi 9717983635 gurpreet@gmail.com
2) 111 Modi India 420 CoModi@gmail.com
3) 103 Xaviour UP 1789783635 xm@gmail.com
4) 105 Mickey Delhi 2717983635 mickey@gmail.com
Update opertion done !!!
1) 101 Gurpreet Singh Delhi 9717983635 gurpreet@gmail.com
2) 111 Modi India 420 CoModi@gmail.com
3) 103 Tom Watson UP 1789783635 xm@gmail.com
4) 105 Mickey Delhi 2717983635 mickey@gmail.com
Contact List is reset is done !!!
All Operation work successfull !!!!
PS D:\Day8>
Day8.csproj

2) Delegate,Anonymous method and Lamda function Program

Source Code :-

Airthmetic.cs

```
using System;
namespace Airth
{
    public class Airthmetic
    {
        public static void addition(double no1,double no2)
        {
            Console.WriteLine("Addition of "+no1 +" and "+no2+" = "+(no1+no2));
        }
        public static void subtraction(double no1,double no2)
        {
            Console.WriteLine("Subtraction of "+no1 +" and "+no2+" = "+(no1-no2));
        }
        public static void multiplication(double no1,double no2)
        {
            Console.WriteLine("Multiplication of "+no1 +" and "+no2+" = "+(no1*no2));
        }
        public static void division(double no1,double no2)
        {
            Console.WriteLine("Division of "+no1 +" and "+no2+" = "+(no1/no2));
        }

    }
}
```

TypeOfOperation.cs

```
using System;
using Airth;
delegate void option(double a,double b);

namespace AdvanceDeligate.Type
{
    public class TypeOfOperation
    {
        double x,y;
        public TypeOfOperation(double a,double b)
        {
        x=a;y=b;
        }
        public void delegateType()
        {
            var dobj=new option(Airthmetic.addition);
            Console.WriteLine("\n ___All Operations are Done using Delegate____ \n");
            dobj+=Airthmetic.subtraction;
            dobj+=Airthmetic.multiplication;
            dobj+=Airthmetic.division;
            dobj.Invoke(x,y);
        }
        public  void anonymousType()
        {
            Console.WriteLine("\n ___All Operations are Done using Anonymous Method ____ \n");
            option add=delegate(double no1,double no2)
            {
                Console.WriteLine("Addition of "+no1 +" and "+no2+" = "+(no1+no2));
            };
            option sub=delegate(double no1,double no2)
            {
                Console.WriteLine("Substraction of "+no1 +" and "+no2+" = "+(no1-no2));
            };
            option mul=delegate(double no1,double no2)
            {
                Console.WriteLine("Multiplication of "+no1 +" and "+no2+" = "+(no1*no2));
            };
            option div=delegate(double no1,double no2)
            {
                Console.WriteLine("Division of "+no1 +" and "+no2+" = "+(no1/no2));
            };
            add(x,y);
            sub(x,y);
            mul(x,y);
            div(x,y);
        }
        public void lamdaType()
        {
        Console.WriteLine("\n ___All Operations are Done using Lamda Function ____ \n");
            option add=(double no1,double no2)=>
```

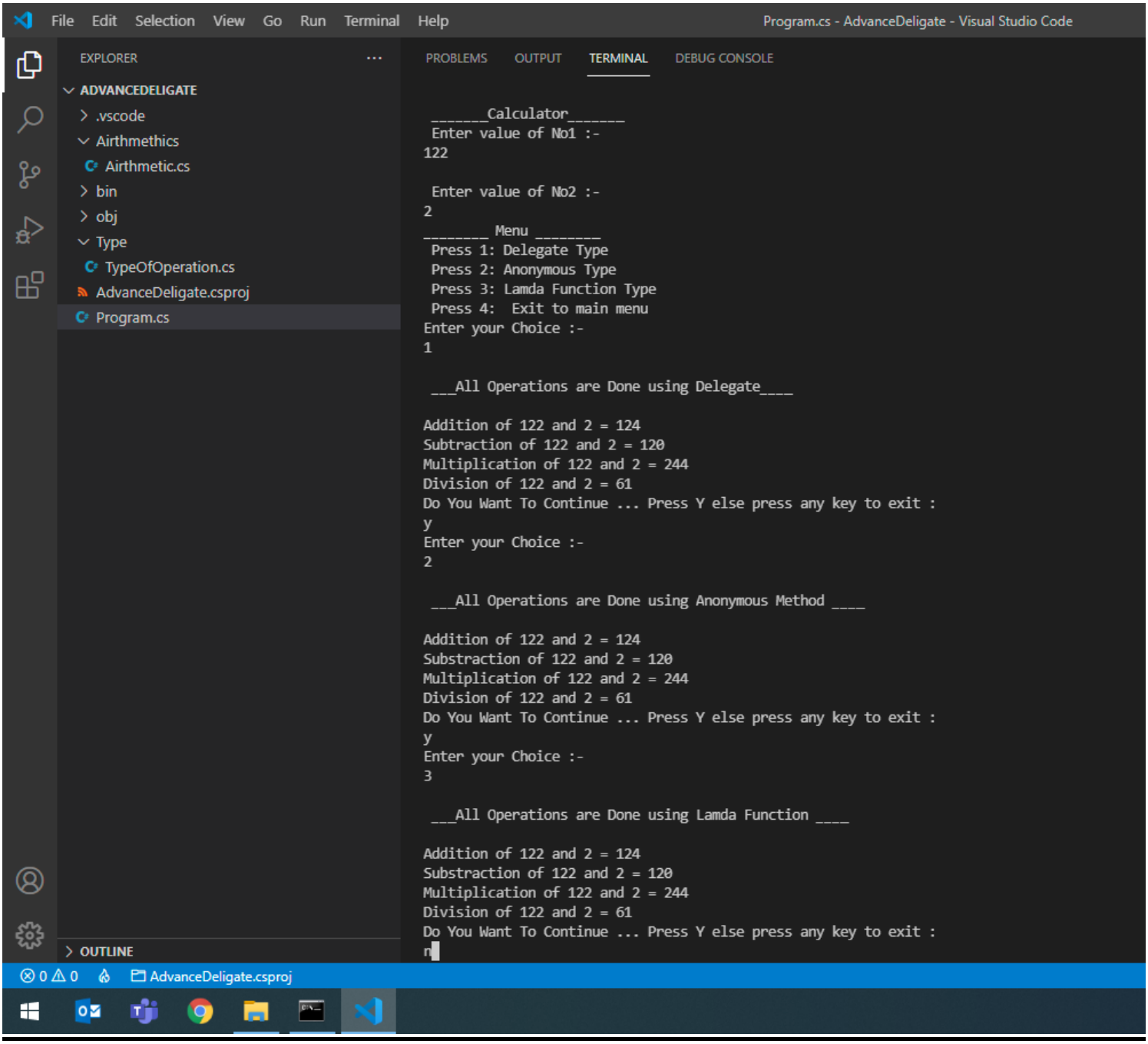
```
            {
                Console.WriteLine("Addition of "+no1 +" and "+no2+" = "+(no1+no2));
            };
            option sub=(double no1,double no2)=>
            {
                Console.WriteLine("Substraction of "+no1 +" and "+no2+" = "+(no1-no2));
            };
            option mul=(double no1,double no2)=>
            {
                Console.WriteLine("Multiplication of "+no1 +" and "+no2+" = "+(no1*no2));
            };
            option div=(double no1,double no2)=>
            {
                Console.WriteLine("Division of "+no1 +" and "+no2+" = "+(no1/no2));
            };
            add(x,y);
            sub(x,y);
            mul(x,y);
            div(x,y);
        }
    }
}
```

Program.cs

```
using System;
using AdvanceDeligate.Type;

delegate void Cal(double x,double y);

namespace AdvanceDeligate
{
    class Program
    {
        public static void Main()
        {

            TypeOfOperation obj;
            Console.WriteLine("\n _______Calculator_______ \n Enter value of No1 :- ");
            double a=Convert.ToDouble(Console.ReadLine());

            Console.WriteLine("\n Enter value of No2 :- ");
            double b=Convert.ToDouble(Console.ReadLine());

            obj=new TypeOfOperation(a,b);

            string uCh="Y";
            Console.WriteLine("________ Menu ________\n Press 1: Delegate Type \n Press 2: Anonym
ous Type \n Press 3: Lamda Function Type  \n Press 4:  Exit to main menu ");
            while(uCh=="Y" || uCh=="y")
            {
                Console.WriteLine("Enter your Choice :-");
                int ch=Convert.ToInt32(Console.ReadLine());
                switch(ch)
                {
                    case 1:
                        obj.delegateType();
                        break;
                    case 2:
                        obj.anonymousType();
                        break;
                    case 3:
                        obj.lamdaType();
                        break;

                    default:
                        Console.WriteLine("Invalid Option");
                        break;
                }
                Console.WriteLine("Do You Want To Continue ... Press Y else press any key to exit
 : ");
                uCh=Console.ReadLine();

            }
        }
```

```
    }
}
```

Output:-


File Edit Selection View Go Run Terminal Help
Program.cs - AdvanceDeligate - Visual Studio Code
EXPLORER
ADVANCEDELIGATE
.vscode
Airthmethics
Airthmetic.cs
bin
obj
Type
TypeOfOperation.cs
AdvanceDeligate.csproj
Program.cs
OUTLINE
PROBLEMS OUTPUT TERMINAL DEBUG CONSOLE
_______Calculator_______
Enter value of No1 :-
122
Enter value of No2 :-
2
_______ Menu _______
Press 1: Delegate Type
Press 2: Anonymous Type
Press 3: Lamda Function Type
Press 4: Exit to main menu
Enter your Choice :-
1
___All Operations are Done using Delegate____
Addition of 122 and 2 = 124
Subtraction of 122 and 2 = 120
Multiplication of 122 and 2 = 244
Division of 122 and 2 = 61
Do You Want To Continue ... Press Y else press any key to exit :
y
Enter your Choice :-
2
___All Operations are Done using Anonymous Method ____
Addition of 122 and 2 = 124
Substraction of 122 and 2 = 120
Multiplication of 122 and 2 = 244
Division of 122 and 2 = 61
Do You Want To Continue ... Press Y else press any key to exit :
y
Enter your Choice :-
3
___All Operations are Done using Lamda Function ____
Addition of 122 and 2 = 124
Substraction of 122 and 2 = 120
Multiplication of 122 and 2 = 244
Division of 122 and 2 = 61
Do You Want To Continue ... Press Y else press any key to exit :
n
0 0
AdvanceDeligate.csproj