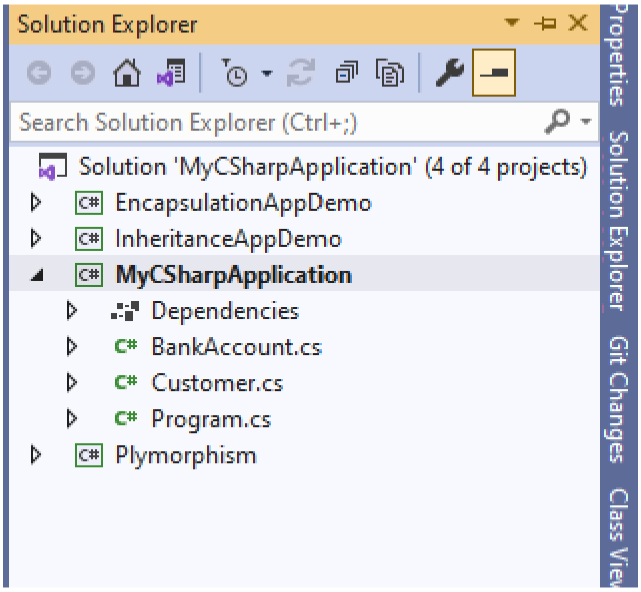
CSharp Bank Application

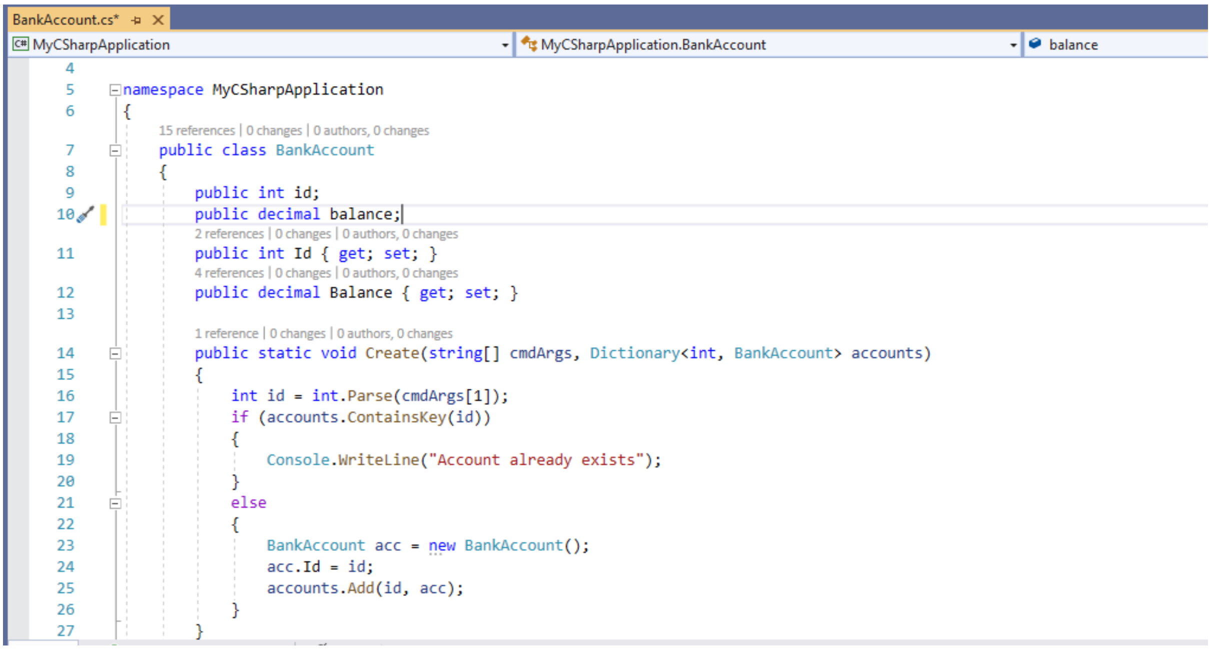
Yêu cầu: Viết lại chương trình. Phát triển tính năng để biết tài khoản(Account) là của khánh hàng nào(Customer)

Tạo chương trình với cấu trúc sau (My CSharpApplication)

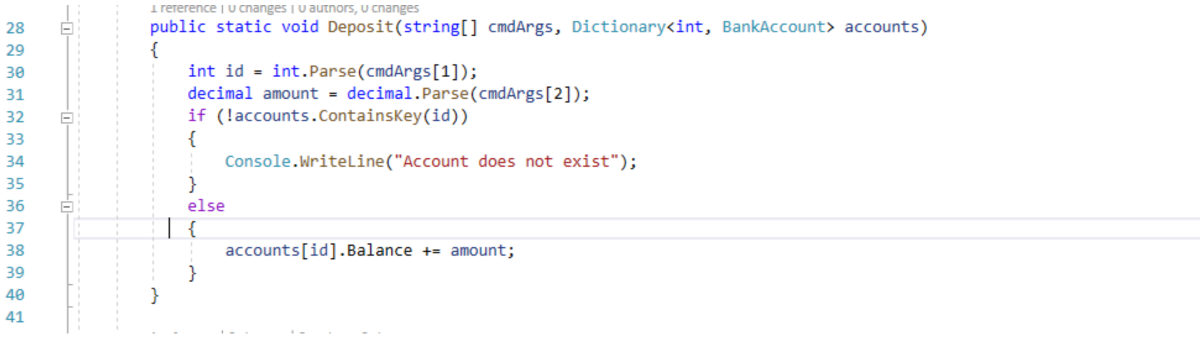


Solution 'MyCSharpApplication' (4 of 4 projects)
- EncapsulationAppDemo
- InheritanceAppDemo
- MyCSharpApplication
  - Dependencies
  - BankAccount.cs
  - Customer.cs
  - Program.cs
- Plymorphism

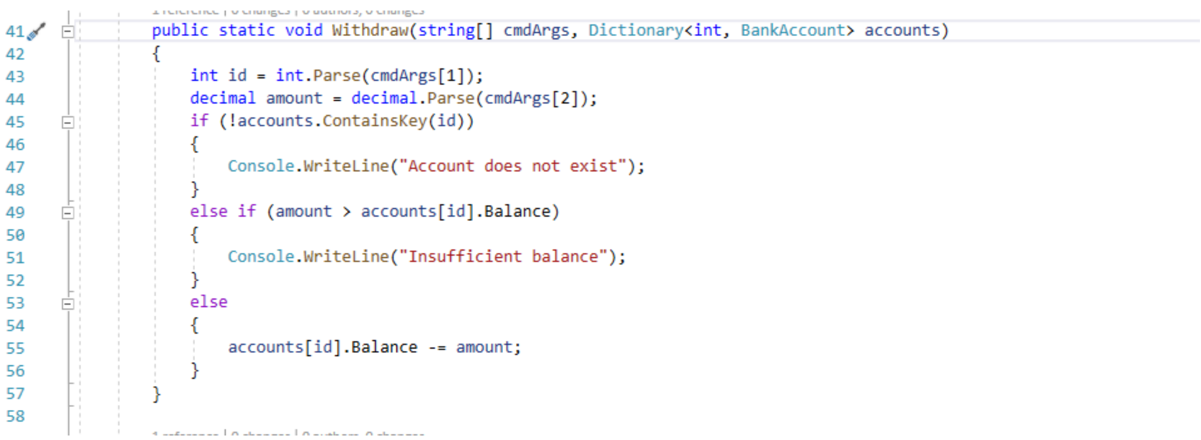
BankAccount.cs class



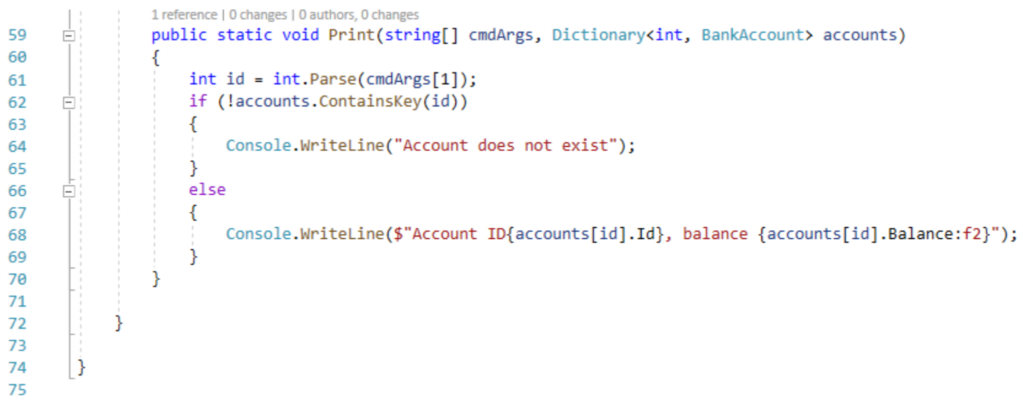
```
namespace MyCSharpApplication
{
    15 references | 0 changes | 0 authors, 0 changes
    public class BankAccount
    {
        public int id;
        public decimal balance;
        2 references | 0 changes | 0 authors, 0 changes
        public int Id { get; set; }
        4 references | 0 changes | 0 authors, 0 changes
        public decimal Balance { get; set; }

        1 reference | 0 changes | 0 authors, 0 changes
        public static void Create(string[] cmdArgs, Dictionary<int, BankAccount> accounts)
        {
            int id = int.Parse(cmdArgs[1]);
            if (accounts.ContainsKey(id))
            {
                Console.WriteLine("Account already exists");
            }
            else
            {
                BankAccount acc = new BankAccount();
                acc.Id = id;
                accounts.Add(id, acc);
            }
        }
```



```
        1 reference | 0 changes | 0 authors, 0 changes
        public static void Deposit(string[] cmdArgs, Dictionary<int, BankAccount> accounts)
        {
            int id = int.Parse(cmdArgs[1]);
            decimal amount = decimal.Parse(cmdArgs[2]);
            if (!accounts.ContainsKey(id))
            {
                Console.WriteLine("Account does not exist");
            }
            else
            {
                accounts[id].Balance += amount;
            }
        }
```

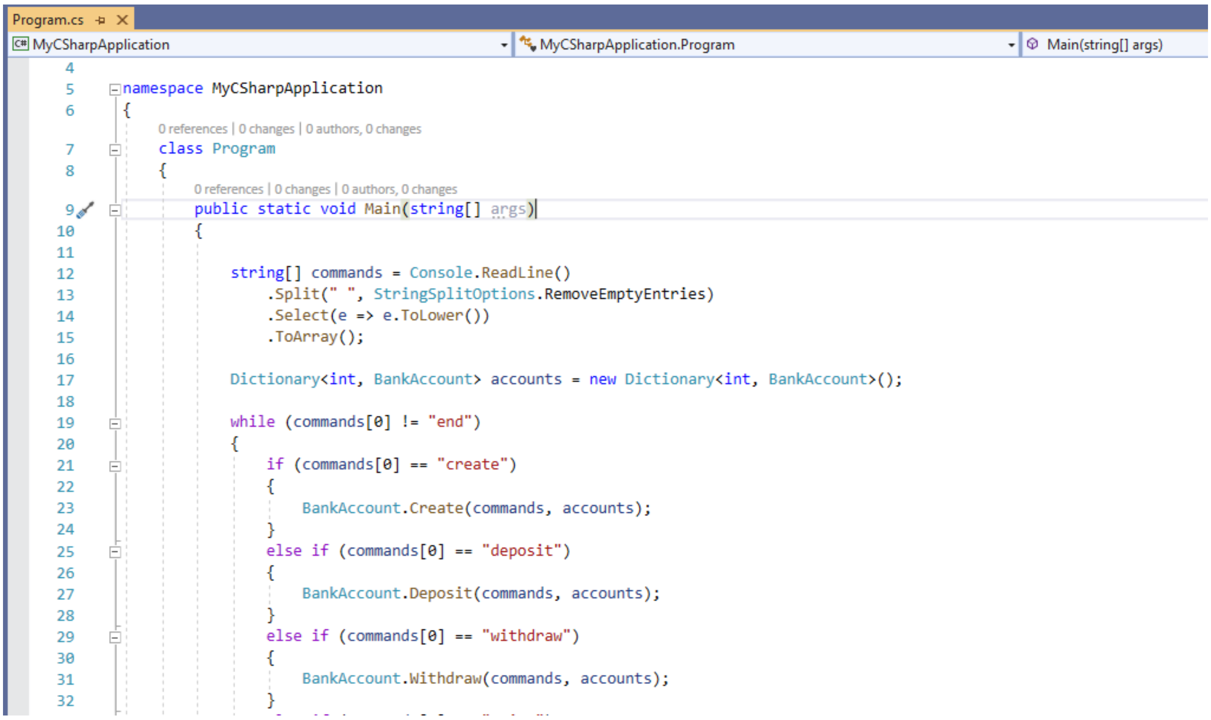


```
        1 reference | 0 changes | 0 authors, 0 changes
        public static void Withdraw(string[] cmdArgs, Dictionary<int, BankAccount> accounts)
        {
            int id = int.Parse(cmdArgs[1]);
            decimal amount = decimal.Parse(cmdArgs[2]);
            if (!accounts.ContainsKey(id))
            {
                Console.WriteLine("Account does not exist");
            }
            else if (amount > accounts[id].Balance)
            {
                Console.WriteLine("Insufficient balance");
            }
            else
            {
                accounts[id].Balance -= amount;
            }
        }
```

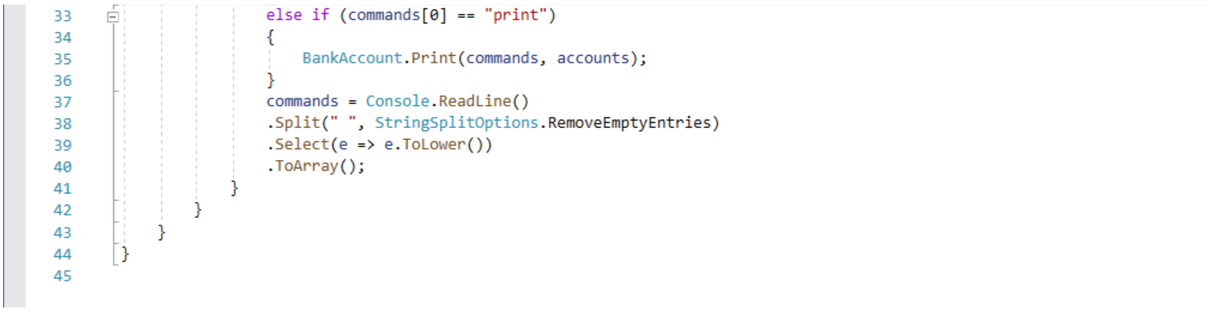


```
        1 reference | 0 changes | 0 authors, 0 changes
        public static void Print(string[] cmdArgs, Dictionary<int, BankAccount> accounts)
        {
            int id = int.Parse(cmdArgs[1]);
            if (!accounts.ContainsKey(id))
            {
                Console.WriteLine("Account does not exist");
            }
            else
            {
                Console.WriteLine($"Account ID{accounts[id].Id}, balance {accounts[id].Balance:f2}");
            }
        }

    }

}
```

Program.cs class

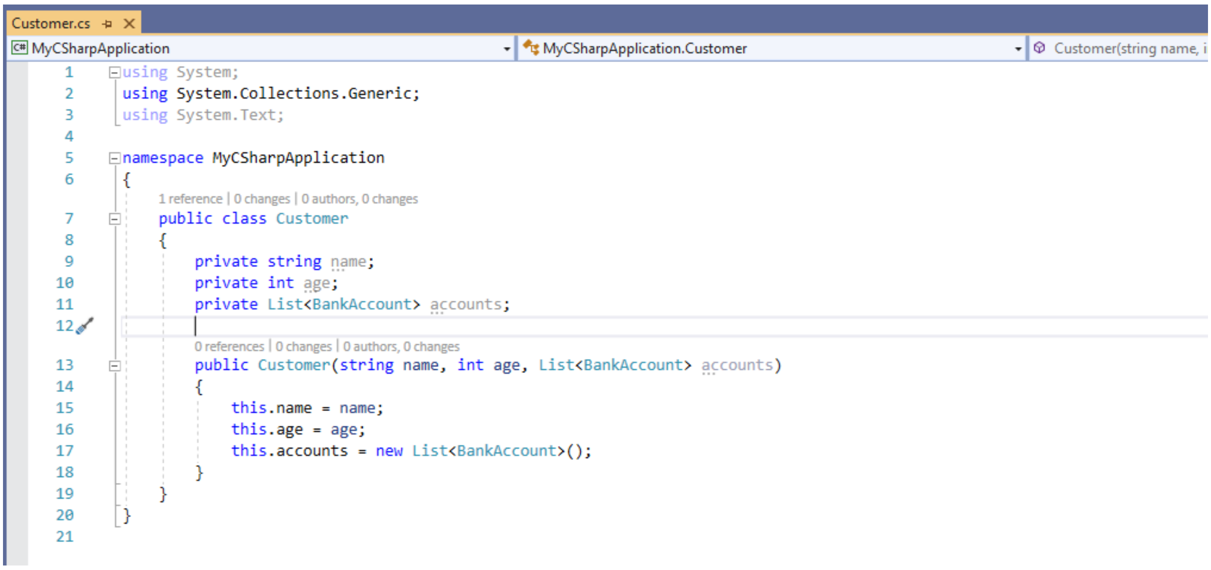


```

namespace MyCSharpApplication
{
    0 references | 0 changes | 0 authors, 0 changes
    class Program
    {
        0 references | 0 changes | 0 authors, 0 changes
        public static void Main(string[] args)
        {

            string[] commands = Console.ReadLine()
                .Split(" ", StringSplitOptions.RemoveEmptyEntries)
                .Select(e => e.ToLower())
                .ToArray();

            Dictionary<int, BankAccount> accounts = new Dictionary<int, BankAccount>();

            while (commands[0] != "end")
            {
                if (commands[0] == "create")
                {
                    BankAccount.Create(commands, accounts);
                }
                else if (commands[0] == "deposit")
                {
                    BankAccount.Deposit(commands, accounts);
                }
                else if (commands[0] == "withdraw")
                {
                    BankAccount.Withdraw(commands, accounts);
                }
```



```
                else if (commands[0] == "print")
                {
                    BankAccount.Print(commands, accounts);
                }
                commands = Console.ReadLine()
                .Split(" ", StringSplitOptions.RemoveEmptyEntries)
                .Select(e => e.ToLower())
                .ToArray();
            }
        }
    }
}

```

Customer.cs class



```
using System;
using System.Collections.Generic;
using System.Text;

namespace MyCSharpApplication
{
    1 reference | 0 changes | 0 authors, 0 changes
    public class Customer
    {
        private string name;
        private int age;
        private List<BankAccount> accounts;

        0 references | 0 changes | 0 authors, 0 changes
        public Customer(string name, int age, List<BankAccount> accounts)
        {
            this.name = name;
            this.age = age;
            this.accounts = new List<BankAccount>();
        }
    }
}

```