PROBLEM

Discount For Customer

In the bank, customers can be Normal, Priviledged, SeniorCitizen and so on. The bank also introduces an offer where privileged customers get a 30% off on the bill while senior citizens get 12% off. Lets implement the inheritance with discount yet again a better understanding.

1. Create Customer, PrivilegedCustomer & SeniorCitizenCustomer class with data members as given below.
2. Implement GenerateBillAmount Method as per the specification.

Write a program to get the customer details and display bill, discount amount based on customer type.

**[Note : Strictly adhere to the object-oriented specifications given as a part of the problem statement.
Follow the naming conventions as mentioned. Create separate classes in separate files.]**

Consider a class named **Customer** with the following private attributes

Data Type	Attributes
string	_name
string	_address
string	_mobileNumber
int	_age

The methods for **getters, setters** and **constructors** are given in the template code.

Include the following public method in **Customer** class.

Method Name	Description
public void DisplayCustomer()	This method displays the person details.

Consider a class **SeniorCitizenCustomer** which extends the class **Customer**.

Include the following public method in **SeniorCitizenCustomer** class.

Method Name	Description
double GenerateBillAmount(int amount)	This method is used to calculate and return the payment amount where the discount is 12%.

Consider a class **PrivilegeCustomer** which extends the class **Customer**.

Include the following public method in **PrivilegeCustomer** class.

Method Name	Description
double GenerateBillAmount(int amount)	This method is used to calculate and return the payment amount where the discount is 30%.

Consider a driver class named **Program** which creates an instance of the above mentioned classes and their functionalities are tested.

Read the respective customer details (Senior Citizen or Privileged) and call the corresponding GenerateBillAmount() method based on the choice as shown in the sample output.

The link to download the template code is given below
Code Template

Input and Output Format:

The bill amount double value should be display 1 decimal palces.

The total amount to be paid value should be displayed upto 2 decimal palces.

Refer sample input and output for formatting specifications.

[All text in bold corresponds to input and the rest corresponds to output.]

Sample Input and Output 1:

1)Privilege Customer
2)SeniorCitizen Customer
Enter Customer Type
1
Enter The Name
Smith
Enter The Age
25
Enter The Address
New York
Enter The Mobile Number
9576531641
Enter The Purchased Amount
5000
Bill Details
Name Smith
Mobile 9576531641
Age 25
Address New York
Your bill amount is Rs 5000.0. Your bill amount is discount under privilege customer
You have to pay Rs 3500.00

1)Privilege Customer
2)SeniorCitizen Customer
Enter Customer Type
2
Enter The Name
Jack
Enter The Age
46
Enter The Address
Chennai
Enter The Mobile Number
7894561230
Enter The Purchased Amount
500
Bill Details
Name Jack
Mobile 7894561230
Age 46
Address Chennai
Your bill amount is Rs 500.0. Your bill amount is discount under senior citizen customer
You have to pay Rs 440.00

Sample Input and Output 3:

1)Privilege Customer
2)SeniorCitizen Customer
Enter Customer Type
3
Invalid Customer Type

eniorCitizenCustomer.cs/

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

class SeniorCitizenCustomer : Customer
{
    double payment;
    public double GenerateBillAmount(int amount)
    {
        payment = amount*0.88;
        return payment;
    }
}
```

Customer.cs/

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

class Customer
{
    string _name, _address, _mobileNumber;
    int _age;

    public string MobileNumber
    {
        get { return _mobileNumber; } set { _mobileNumber = value; }
    }

    public string Address
    {
        get { return _address; }  set { _address = value; }
    }

    public string Name
    {
        get { return _name; }  set { _name = value; }
    }

    public int Age
    {
        get { return _age; }
        set { _age = value; }
    }

    public void DisplayCustomer()
    {
    Console.WriteLine("Bill Details\nName {0}\nMobile {1}\nAge {2}\nAddress {3}",_name,_mobileNumber,_age,_address);
    }
}
```

Program.cs/

```
    static void Main(string[] args)
    {
        Customer cus = new Customer();
        PrivilegeCustomer cus1 = new PrivilegeCustomer();
        SeniorCitizenCustomer cus2 = new SeniorCitizenCustomer();
        double output;
        Console.WriteLine("1)Privilege Customer\n2)SeniorCitizen Customer\nEnter Customer Type");
        int ch = int.Parse(Console.ReadLine());
        if (ch > 0 && ch < 3)
        {
            Console.WriteLine("Enter The Name");
            cus.Name = Console.ReadLine();
            Console.WriteLine("Enter The Age");
            cus.Age = int.Parse(Console.ReadLine());
            Console.WriteLine("Enter The Address");
            cus.Address = Console.ReadLine();
            Console.WriteLine("Enter The Mobile Number");
            cus.MobileNumber = Console.ReadLine();
            Console.WriteLine("Enter The Purchased Amount");
            int purc = int.Parse(Console.ReadLine());
            cus.DisplayCustomer();
            if (ch == 1)
            {
                output = cus1.GenerateBillAmount(purc);
                Console.WriteLine("Your bill amount is Rs {0:0.0}. Your bill amount is discount under privilege customer", purc);
                Console.WriteLine("You have to pay Rs {0:0.00}", output);
            }
            else
            {
                output = cus2.GenerateBillAmount(purc);
                Console.WriteLine("Your bill amount is Rs {0:0.0}. Your bill amount is discount under senior citizen customer", purc);
                Console.WriteLine("You have to pay Rs {0:0.00}", output);
            }
        }
        else
        Console.WriteLine("Invalid Customer Type");
    }
}
```

Program.cs/ PrivilegeCustomer.cs/

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

class PrivilegeCustomer : Customer
{
    double payment;
    public double GenerateBillAmount(int amount)
    {
        payment = amount*0.7;
        return payment;
    }
}
```

PROBLEM

Account Details

Write a program to read and display the various type of account details.

**[Note : Strictly adhere to the object-oriented specifications given as a part of the problem statement.
Follow the naming conventions as mentioned. Create separate classes in separate files.]**

Consider the class **Account** with the following private attributes/variables.

Data Type	Variable
string	_holderName
long	_accountNumber
string	_IFSCCode
long	_contactNumber

Include appropriate **getters** and **setters**.
Include **default** and **parameterized** constructor for the class.
Prototype for the Parameterized Constructor **Account(string _holderName,long _accountNumber,string _IFSCCode,long _contactNumber)**

Define the following method in the **Account** class.

Method	Description
public void Display()	This method displays account details in the following order _holderName,_AccountNumber,_IFSCCode,_contactNumber. Display the statement '**Your Contact Details**' inside this method.

Consider the class **SavingAccount** which inherits **Account** class with the following private attributes/variables.

Data Type	Variable
double	_interestRate

Define the following method in the **SavingAccount** class.

Method	Description
public void Display()	This method is used to call the base class Display() and in addition displays _interestRate.

Include **default** and **parameterized** constructor for the class.
Prototype for the Parameterized Constructor **SavingAccount(string _holderName, long _accountNumber, string _IFSCCode, long _contactNumber, double _interestRate)**
Use **base** Keyword to call the base class constructor.

Consider the class **CurrentAccount** which inherits **Account** class with the following private attributes/variables.

Data Type	Variable
string	_organizationName
long	_TIN

Define the following method in the **CurrentAccount** class

Method	Description
public void Display()	This method is used to call base class Display() and in addition displays _organizationName,_TIN

Include **default** and **parameterized** constructor for the class.
Prototype for the Parameterized Constructor **CurrentAccount(string _holderName, long _accountNumber, string _IFSCCode, long _contactNumber,string _organizationName, long _TIN)**
Use **base** Keyword to call the base class constructor.

Create **Program** class with **Main** method, get user details in comma seperated format in the following order **(HolderName,Account Number,IFSC code,Contact Number).**
Display the Account Details by calling **method of base class with child class object.**

The link to download the template code is given below
Code Template

Input and Output Format:

Refer sample input and output for formatting specifications.

[All text in bold corresponds to the input and the rest corresponds to output.]

Sample Input and Output 1:

Enter User Details(HolderName,Account Number,IFSC code,Contact Number)
Steffan,982714210,S160030600514,9092304676
Enter Account Type
saving
Enter Interest Rate
12.0
Your Contact Details
HolderName : Steffan
Account Number : 982714210
IFSCCode : S160030600514
ContactNumber : 9092304676
Interest Rate : 12

Sample Input and Output 2:

Enter User Details(HolderName,Account Number,IFSC code,Contact Number)
John,7889142075,S1600ABY0576,9944001700
Enter Account Type
current
Enter organization Name
pentamedia Graphics Limited
Enter TIN number
7841
Your Contact Details
HolderName : John
Account Number : 7889142075
IFSCCode : S1600ABY0576
ContactNumber : 9944001700
Organization Name : pentamedia Graphics Limited
TIN : 7841

Sample Input and Output 3:

Enter User Details(HolderName,Account Number,IFSC code,Contact Number)
Shira,987451024,SWQ78914AF,9078425168
Enter Account Type
curr
Enter valid Account Type

CurrentAccount.cs/

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

namespace Inheritance2P
{
    class CurrentAccount : Account
    {
        string _orgName;
        long _TIN;

        public CurrentAccount(string _holderName, long _accNumber, string _IFSCCode, long _contactNumber, string _orgName, long _TIN)
                : base(_holderName, _accNumber, _IFSCCode, _contactNumber)
        {
            this._orgName = _orgName;
            this._TIN = _TIN;
        }
        public override void Display()
        {
            base.Display();
            Console.WriteLine("Organization Name : " + _orgName);
            Console.WriteLine("TIN : " + _TIN);
        }
    }
}

```

CurrentAccount.cs/ SavingAccount.cs/

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

namespace Inheritance2P
{
    class SavingAccount : Account
    {
        double _interestRate;

        public SavingAccount(string _holderName, long _accNumber, string _IFSCCode, long _contactNumber, double _interestRate)
            : base(_holderName, _accNumber, _IFSCCode, _contactNumber)
        {
            this._interestRate = _interestRate;
        }
        public override void Display()
        {
            base.Display();
            Console.WriteLine("Interest Rate : " + _interestRate);
        }
    }
}

```

Program.cs/

```
using System.Threading.Tasks;

namespace Inheritance2P
{
    class Program
    {
        static void Main(string[] args)
        {
            Console.WriteLine("Enter User Details(HolderName,Account Number,IFSC code,Contact Number)");
            string details = Console.ReadLine();
            char del = Convert.ToChar(",");
            string[] str1 = details.Split(del);
            long acc = Convert.ToInt64(str1[1]);
            long contact = Convert.ToInt64(str1[3]);
            Console.WriteLine("Enter Account Type");
            string acc_type = Console.ReadLine();
            if (acc_type == "current" || acc_type == "Current")
            {
                Console.WriteLine("Enter organization Name");
                String org = Console.ReadLine();
                Console.WriteLine("Enter TIN number");
                long TIN = Convert.ToInt64(Console.ReadLine());
                CurrentAccount cacc = new CurrentAccount(str1[0], acc, str1[2], contact, org, TIN);
                cacc.Display();
            }
            else if (acc_type == "saving" || acc_type == "Saving")
            {
                Console.WriteLine("Enter Interest Rate");
                double in_rate = Convert.ToDouble(Console.ReadLine());
                SavingAccount sacc = new SavingAccount(str1[0], acc, str1[2], contact, in_rate);
                sacc.Display();
            }
            else
                Console.WriteLine("Enter valid Account Type");

        }
    }
}
```

Program.cs/ Account.cs/

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

namespace Inheritance2P
{
    class Account
    {
        protected string _holderName;
        protected long _accNumber;
        protected string _IFSCCode;
        protected long _contactNumber;

        public Account() { }
        public Account(string _holderName, long _accNumber, string _IFSCCode, long _contactNumber)
        {
            this._holderName = _holderName;
            this._accNumber = _accNumber;
            this._IFSCCode = _IFSCCode;
            this._contactNumber = _contactNumber;
        }
        public virtual void Display()
        {
            Console.WriteLine("Your Contact Details");
            Console.WriteLine("HolderName : " + _holderName);
            Console.WriteLine("Account Number : " + _accNumber);
            Console.WriteLine("IFSCCode : " + _IFSCCode);
            Console.WriteLine("ContactNumber : " + _contactNumber);
        }
    }
}
```

PROBLEM

Calculate Reward Points

ABC Bank announced a new scheme of reward points for a transaction using an ATM card. Each transaction using the normal card will be provided by 1% of the transaction amount as reward point. If a transaction is made using a premium card and it is for fuel expenses, additional 10 points will be rewarded. Help the bank to calculate the total reward points.

**[Note : Strictly adhere to the object-oriented specifications given as a part of the problem statement.
Follow the naming conventions as mentioned. Create separate classes in separate files.]**

Consider a class **VISACard** with the following method.

Method	Description
public double ComputeRewardPoints(string _type, double _amount)	This method returns the 1% of the transaction amount as reward points.

Consider a class named **HPVISACard** which extends **VISACard** class and overrides the following method.

Method	Description
public double ComputeRewardPoints(string _type, double _amount)	In this method, calculate the reward points from the base class and add 10 points if it is for fuel expense.

Hint:
Use base keyword to calculate reward points from base class.

Consider the class **Program** with **Main** method, get the transaction details as a comma separated values.
(Transaction type, amount, card type)

The card type will be either '**VISA card**' or '**HPVISA card**'. Otherwise, display '**Invalid data**'

Calculate the reward points corresponding to the card type and transaction type and print the reward points.

Input and Output Format:

Reward point double values should be displayed upto **2 decimal** palces
Refer sample input and output for formatting specifications.

[All text in bold corresponds to the input and the rest corresponds to output.]

Sample Input and Output 1:

Enter the transaction detail
Shopping,5000,VISA card
Total reward points earned in this transaction is 50.00
Do you want to continue?(Yes/No)
Yes
Enter the transaction detail
Fuel,5000,HIVISA card
Invalid data
Do you want to continue?(Yes/No)
Yes
Enter the transaction detail
Fuel,5000,HPVISA card
Total reward points earned in this transaction is 60.00
Do you want to continue?(Yes/No)
No

HPVISACard.cs/

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

namespace Inheritance3P
{
    class HPVISACard : VISACard
    {
        public new double computeRewardPoints(string _type, double _amount)
        {
            _amount = base.computeRewardPoints(_type, _amount);
            double sum = _amount + 10.00;
            Console.WriteLine("Total reward points earned in this transaction is {0:0.00}", sum);
            return _amount;
        }
    }
}

```

Program.cs/

```
{
    class Program
    {
        static void Main(string[] args)
        {
            VISACard visa = new VISACard();
            HPVISACard hpvisa = new HPVISACard();
            do
            {
                Console.WriteLine("Enter the transaction detail");
                string details = Console.ReadLine();
                char del = Convert.ToChar(",");
                string[] str = details.Split(del);
                if (str[2] == "VISA card" || str[2] == "HPVISA card")
                {
                    double amount = Convert.ToDouble(str[1]);
                    if ((str[0] == "Fuel" || str[0] == "fuel") && str[2] == "HPVISA card")
                    {
                        hpvisa.computeRewardPoints(str[0], amount);
                    }
                    else
                    {
                        amount = visa.computeRewardPoints(str[0], amount);
                        Console.WriteLine("Total reward points earned in this transaction is {0:0.00}", amount);
                    }
                }
                else
                {
                    Console.WriteLine("Invalid data");
                }
                Console.WriteLine("Do you want to continue?(Yes/No)");
                string ch = Console.ReadLine();
                if (ch == "yes" || ch == "Yes")
                    continue;
                else
                    break;
            } while (true);
        }
```

VISACard.cs/

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

namespace Inheritance3P
{
    class VISACard
    {
        public double computeRewardPoints(string _type, double _amount)
        {
            _amount = _amount * 0.01;
            return _amount;
        }
    }
}
```

PROBLEM

GST Calculation

Write a program to calculate the total amount with GST for the events. There are two types of Events Stage show and Exhibition. For Stage show GST will be 15% and for exhibition GST will be 5%.

[Note : Strictly adhere to the object-oriented specifications given as a part of the problem statement.
Follow the naming conventions as mentioned. Create separate classes in separate files.]

Consider a class names **Event** with the following protected attributes.

Data Type	Attributes
string	_name
string	_type
double	_costPerDay
int	_noOfDays

Include **default** and **parameterized** constructor for the class.
Prototype for the Parameterized Constructor **Event(string _name, string _type, double _costPerDay, int _noOfDays)**

Consider the class **Exhibition** which inherits the **Event** class with the following private attributes.

Data Type	Attributes
static int	_gst = 5
int	_noOfStalls

Include **default** and **parameterized** constructor for the class.
Prototype for the Parameterized Constructor **Exhibition(string _name, string _type, double _costPerDay, int _noOfDays, int _noOfStalls)**

Define the following method in the **Exhibition** class.

Method	Description
public double totalCost()	This method is to calculate the total amount with 5% GST.

Consider the class **StageEvent** which inherits the **Event** class with the following private attributes.

Data Type	Attributes
static int	_gst = 15
int	_noOfSeats

Include **default** and **parameterized** constructor for the class.
Prototype for the Parameterized Constructor **StageEvent(string _name, string _type, double _costPerDay, int _noOfDays, int _noOfSeats)**

Define the following method in the **StageEvent** class.

Method	Description
public double totalCost()	This method is to calculate the total amount with 15% GST.

Use base() to call and assign values in base class constructor.
Override ToString() method to display the event details and the total amount inside this ToString() method.

Create **Program** class with **Main** method.
In the Main() method, read the event details from the user and then create the object of the event according to the event type. Display the statement '**Event Details**' inside the Main() method.

The link to download the template code is given below
Code Template

Input and Output Format:

All the double values should be displayed upto **2 decimal** palces

Refer sample input and output for formatting specifications.

[All text in bold corresponds to the input and the rest corresponds to output.]

Sample Input and Output 1:

Enter event name
Sky Lantern Festival
Enter the cost per day
1500
Enter the number of days
3
Enter the type of event
1.Exhibition
2.Stage Event
2
Enter the number of seats
100
Event Details
Name:Sky Lantern Festival
Type:Stage Event
Number of seats:100
Total amount: 5175.00

Sample Input and Output 2:

Enter event name
Glastonbury
Enter the cost per day
5000
Enter the number of days
2
Enter the type of event
1.Exhibition
2.Stage Event
1
Enter the number of stalls
10
Event Details
Name:Glastonbury
Type:Exhibition
Number of stalls:10
Total amount: 10500.00

Sample Input and Output 3:

Enter event name
Glastonbury
Enter the cost per day
5000
Enter the number of days
2
Enter the type of event
1.Exhibition
2.Stage Event
3
Invalid input

Exhibition.cs/

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

namespace Inheritance4P
{
    class Exhibition : Event
    {
        static int _gst = 5;
        int _noOfStalls;
        public Exhibition(string _name, string _type, double _costPerDay, int _noOfDays, int _noOfStalls) : base(_name, _type, _costPerDay, _noOfDays)
        {
            this._noOfStalls = _noOfStalls;
        }
        public double totalCost()
        {
            double _totalCost;
            //_totalCost = _costPerDay + _gst;
            //_totalCost = (_costPerDay*_noOfDays) + (_gst/100);
            _totalCost = ((_costPerDay * _noOfDays) + ((_costPerDay * _noOfDays) * _gst / 100));
            return _totalCost;
        }
        public override string ToString()
        {
            string str;
            str = "Name:" + _name;
            str += "\nType:" + _type;
            str += "\nNumber of stalls:" + _noOfStalls;
            str += String.Format("\nTotal amount:{0:0.00}", totalCost());
            return str;
        }
    }
}

```

Exhibition.cs/ Event.cs/

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

namespace Inheritance4P
{
    class Event
    {
        protected string _name;
        protected string _type;
        protected double _costPerDay;
        protected int _noOfDays;
        protected string Name { get => _name; set => _name = value; }
        protected string Type { get => _type; set => _type = value; }
        protected double CostPerDay { get => _costPerDay; set => _costPerDay = value; }
        protected int NoOfDays { get => _noOfDays; set => _noOfDays = value; }
        public Event(string _name, string _type, double _costPerDay, int _noOfDays)
        {
            this._name = _name;
            this._type = _type;
            this._costPerDay = _costPerDay;
            this._noOfDays = _noOfDays;
        }
    }
}

```

Exhibition.cs/ | StageEvent.cs/

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

namespace Inheritance4P
{
    class StageEvent : Event
    {
        static int _gst = 15;
        int _noOfSeats;


        public StageEvent(string _name, string _type, double _costPerDay, int _noOfDays, int _noOfSeats) : base(_name, _type, _costPerDay, _noOfDays)
        {

            this._noOfSeats = _noOfSeats;

        }
        public double totalCost()
        {
            double _totalCost;
            //_totalCost = _costPerDay + _gst;
            //_totalCost = (_costPerDay*_noOfDays) + (_gst/100);
            _totalCost = ((_costPerDay * _noOfDays) + ((_costPerDay * _noOfDays) * _gst / 100));
            return _totalCost;
        }
        public override string ToString()
        {
            string str;
            str="Name:" + _name;
            str+="\nType:" + _type;
            str+="\nNumber of seats:" + _noOfSeats;
            str+=String.Format("\nTotal amount:{0:0.00}",totalCost());

            return str;
        }
```

Exhibition.cs/ | StageEvent.cs/ | Program.cs/

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



            Console.WriteLine("Enter event name");
            string _name = Console.ReadLine();
            Console.WriteLine("Enter the cost per day");
            double _costPerDay = Convert.ToDouble(Console.ReadLine());
            Console.WriteLine("Enter the number of days");
            int _noOfDays = Convert.ToInt32(Console.ReadLine());
            Console.WriteLine("Enter the type of event\n1.Exhibition\n2.Stage Event");
            int _type = Convert.ToInt32(Console.ReadLine());
            Event e;
            if (_type == 1)
            {
                Console.WriteLine("Enter the number of stalls");
                int _noOfStall = Convert.ToInt32(Console.ReadLine());
                e = new Exhibition(_name, "Exhibition", _costPerDay, _noOfDays, _noOfStall);
                Console.WriteLine("Event Details");
                Console.WriteLine(e.ToString());
            }
            else if (_type == 2)
            {
                Console.WriteLine("Enter the number of seats");
                int _noOfSeats = Convert.ToInt32(Console.ReadLine());
                e = new StageEvent(_name, "Stage Event", _costPerDay, _noOfDays, _noOfSeats);
                Console.WriteLine("Event Details");
                Console.WriteLine(e.ToString());
            }
            else
            {
                Console.WriteLine("Invalid input");
            }
            Console.ReadLine();
        }
```

Settings | Split | text | //Program.cs | Currently you have edit access to this file.