Description Code Editor

Luxe Salon

Scenario:

A hair salon Luxe is an establishment that offers professional hair styling services for men and women. That salon provides a wide range of hair services, such as professional hair styling and texturizing. The hair salon also offers hair coloring, highlights, head and scalp treatments, and formal styling. The salon provides discount offers based on membership cards like Gold, Diamond, Platinum, etc. They need software to calculate the service charge after the discount.

As their software consultant, you can help them by developing a C# application.

Functionalities:

In class **Customer**, implement the below-given properties.

Data Type	Property Name
string	CustomerName
string	MemberType
double	ServiceCharge

Remaining

2
Hrs

Assess

Type here to search

In class **Customer,** implement the below-given properties.

Data Type	Property Name
string	CustomerName
string	MemberType
double	ServiceCharge

In class **CustomerDetails,** implement the below-given methods and also **Inherit** the class **Customer**.

Method	Description
public bool ValidateMemberType()	This method is used to validate the customer membership type. If the member type is "**Gold**" or "**Diamond**" or "**Platinum**" , then return **true**. Otherwise, return **false.**
public double ServiceChargeAfterDiscount()	This method is used to calculate the service charge after the discount and return it. The calculation procedures are given below.

Formula :

Service charge after discount = Service charge - Discount amount

Member Type	Discount amount
Gold	5% of Service charge
Diamond	10% of Service charge
Platinum	15% of Service charge

Note:

Member type is **Case-sensitive**.

In **Program** class - **Main** method,

1. Get the values from the **user**.

2. Call the **ValidateMemberType** method, If it returns true, then move on to step 3, If it returns false then display **Invalid membership.**

3. Use the values in method **ServiceChargeAfterDiscount** and display the result as per the sample output.

Note:

Note:

Member type is **Case-sensitive**.

In **Program** class - **Main** method,

1. Get the values from the **user**.

2. Call the **ValidateMemberType** method, If it returns true, then move on to step 3, If it returns false then display **Invalid membership.**

3. Use the values in method **ServiceChargeAfterDiscount** and display the result as per the sample output.

Note:

- Keep the properties, methods and classes as **public.**
- Please read the method rules **clearly**.
- Do not use **Environment.Exit()** to terminate the program.
- Do not change the given code template.

Sample Input 1:

Enter the customer name

Sam

Enter the membership type

Enter the membership type

Gold

Enter the service charge

1000

Sample Output 1:

Service charge after discount is 950

Sample Input 2:

Enter the customer name

John

Enter the membership type

Bronze

Enter the service charge

1500

Description Code Editor

Mechtronic Customer Card

Scenario:

Mechtronic is one of the leading electronic shops in the city, known for its wide range of high-quality electronic products and excellent customer service. In order to reward their loyal customers and attract new ones, the management at Mechtronic has decided to provide new discount offers based on their card type. They need an application to calculate the total amount after the discount.

As their software consultant, you help them by developing a C# application.

Functionalities:

In class **Bill,** implement the below-given properties.

Data Type	Property Name
string	CustomerName
string	CardType
double	PurchaseAmount

In class **Service**, implement the below-given methods and also **Inherit** the class **Bill**.

Method	**Description**
public bool ValidateCardType()	This method is used to validate the **card type.** The card type should be "**Classic**" or "**Gold**" or "**Platinum**" or "**Signature**". If the card type is valid then return **true**. Otherwise, return **false.** **Note:** CardType is **Case-sensitive**.
public double CalculateTotalAmount()	This method is used to calculate the **total amount** after the discount based on the card type and return it. The calculation procedure is given below.

Formula :

Total amount = PurchaseAmount - (PurchaseAmount * Discount Percentage)

Formula :

Total amount = PurchaseAmount - (PurchaseAmount * Discount Percentage)

Card Type	Discount Percentage
Classic	5%
Gold	10%
Platinum	15%
Signature	20%

In **Program** class - **Main** method,

1. Get the values from the **user**.
2. Call the **ValidateCardType** method, If it returns true, then move on to step 3, If it returns false then display **Invalid card type.**
3. Use the values in method **CalculateTotalAmount** and display the result as per the sample output.

Note:

Sample Input 1:

Enter the customer name

Nisha

Enter the card type

Gold

Enter the purchased amount

35200

Sample Output 1:

Total Amount is 31680

Sample Input 2:

Enter the customer name

Robin

Enter the card type

Silver

Enter the purchased amount

Sample Output 1:

Total Amount is 31680

Sample Input 2:

Enter the customer name

Robin

Enter the card type

Silver

Enter the purchased amount

40000

Sample Output 2:

Invalid card type

Description Code Editor

Weather Report

Scenario:

The Regional Meteorological Center is responsible for collecting, analyzing and storing weather data for the entire region throughout the year. To ensure that the data is easily accessible and usable, RMC wants to create a weather analysis and manipulation application. This application will be used by meteorologists, researchers, and other staff members to easily retrieve and analyze data based on location and date.

You being their software consultant, help them by developing a C# application.

Functionalities:

In class **Weather**, implement the below-given properties.

Data Type	Properties
string	Location
string	Date
int	Temperature

string	Status

In class **Program,** implement the below-given method.

public static List<Weather> WeatherList -In the code template, it is already provided.

implement the features listed below.

Method	Description
public void AddWeatherDetails (string[ ] weatherDetails)	This method is used to add the weather details into the **WeatherList.** This method should separate the values in each string by a **comma** from the input array and assign those values to the **Weather** object, then add each object to the **WeatherList.**
public List<Weather> ViewDetailsByLocation(string location)	This method is used to find the weather details based on the **location** passed as an argument.

public List<Weather> ViewDetailsByLocation(string location)	This method is used to find the weather details based on the **location** passed as an argument. If the **location** is found in the **WeatherList**, then add those weather details to the List and return that **List**. else, return an **empty list.**
public List<Weather> ViewDetailsByDate(string date)	This method is used to find the weather details based on the **date** passed as an argument. If the **date** is found in the **WeatherList**, then add those weather details to the List and return that **List**. else, return an **empty list.**

In **Program** class, **Main** method,

1. Get the values from the **user**.
2. Call the methods accordingly and display the result.

   – If the **ViewDetailsByLocation** method returns an empty list, then display "**Location is not found**".

Sample Input and Output :

1. Add Weather Details

2. View Details By Location

3. View Details By Date

4. Exit

Enter the choice

1

Enter the number of entries

3

Miami,25/01/2023,20,Partly Cloud

Miami,26/01/2023,33,Mostly Sunny

Berth,25/01/2023,25,Sunny

1. Add Weather Details

2. View Details By Location

3. View Details By Date

2. View Details By Location

3. View Details By Date

4. Exit

Enter the choice

2

Enter the location

Miami

Location	Date	Temperature	Status
Miami	25/01/2023	20	Partly Cloud
Miami	26/01/2023	33	Mostly Sunny

1. Add Weather Details

2. View Details By Location

3. View Details By Date

4. Exit

2

Enter the location

Sydney

Location is not found

1. Add Weather Details

2. View Details By Location

3. View Details By Date

4. Exit

Enter the choice

3

Enter the date

25/01/2023

Location	Date	Temperature	Status

3. View Details By Date

4. Exit

Enter the choice

3

Enter the date

10/01/2000

Date is not found

1. Add Weather Details

2. View Details By Location

3. View Details By Date

4. Exit

Enter the choice

4

Thank you.