1 .Scenario:

Techwing Solutions is a prominent software company in the city that prioritizes workplace safety for its employees. As part of their safety measures, the company collects vaccination status information from their employees, and even incentivizes those who have not yet been vaccinated to get vaccinated by offering rewards. To effectively manage this process, the company is in need of software that can efficiently handle and organize employee vaccination details.

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

Functionalities:

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

public static Dictionary<string, string> vaccinationDetails -In the code template, it is already provided.

implement the features listed below.

Method	Description
public void AddVaccinationDetails(Dictionary<string, string> dic)	In this method pass the employee id and their vaccination status as dictionary. Using this method add the details into the **vaccinationDetails** dictionary**.**
public List<String> GetEmployeeId(string status)	This method is used to get the employees id based on the vaccination status. Then store the employees id into the **List** and return it
public string UpdateVaccinationStatus(string id, string status)	This method is used to update the vaccination status based on the employee id and return employee id with vaccination status in the given format.

	Employee id : **Emp1** Status : **Fully Vaccinated** **Emp1_Fully Vaccinated** In between add **underscore(_).**

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

1. Get the values from the **user** as per the Sample Input.

2. Call the methods accordingly and display the result.

3. In the Sample Input / Output provided, the highlighted text in bold corresponds to the input given by the user and the remaining text represents the output.

Note:

- Keep the method and class 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 / Output:

1. Add the vaccination details

2. Get employee id by status

3. Update vaccination details

4. Exit

Enter your choice

1

Enter the employee id

EMP1

Enter the status

Fully Vaccinated

1. Add the vaccination details

2. Get employee id by status

3. Update vaccination details

4. Exit

Enter your choice

1

Enter the employee id

EMP2

Enter the status

Partial Vaccinated

1. Add the vaccination details

2. Get employee id by status

3. Update vaccination details

4. Exit

Enter your choice

1

Enter the employee id

EMP3

Enter the status

Not Vaccinated

```
1. Add the vaccination details

2. Get employee id by status

3. Update vaccination details

4. Exit

Enter your choice

2

Enter the status

Fully Vaccinated


EMP1


1. Add the vaccination details

2. Get employee id by status

3. Update vaccination details

4. Exit

Enter your choice

3

Enter the employee id

EMP2

Enter the status

Fully Vaccinated


EMP2_Fully Vaccinated
```

1. Add the vaccination details

2. Get employee id by status

3. Update vaccination details

4. Exit

Enter your choice

4

Thank You

2.

Scenario:

ConTV is dedicated to being the epitome of user-centric television companies, continually surpassing expectations through groundbreaking advancements in software, hardware, and services. In their relentless pursuit of customer satisfaction, ConTV actively seeks feedback on the TV installation process. To further enhance their understanding of customer sentiments, they are in need of a sophisticated software solution capable of calculating ratings based on feedback comments.

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

Functionalities:

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

Class	**Properties**
Installation	String ExpectedDate String InstalledDate double Rating String Feedback

In class **InstallationDetails,** implement the below-given methods and also **Inherit** the class **Installation** .

Method	**Description**
public void GetCustomerFeedback()	This method is used to get the feedback from the customer based on the expected date and installed date. If the installed date is less than expected date, then assign "**VeryGood**" in the **Feedback** property. If the expected and installed dates are the same, then assign "**Good**" in the **Feedback** property. If the installed date is within three days from expected date, then assign "**Average**" in the **Feedback** property. If the installed date is more than three days from expected date, then assign "**Poor**" in the **Feedback** property.
public double CalculateRating()	This method is used to calculate the updated rating based on the feedback and return it. If the rating is not between **1** and **10** then return **0**. The calculation conditions are given below.

- If the feed back comment is **VeryGood,** then add 50% of the rating to the rating.
- If the feed back comment is **Good,** then add 20% of the rating to the rating.
- If the feed back comment is **Average,** then subtract 20% of the rating to the rating.
- If the feed back comment is **Poor,** then subtract 50% of the rating to the rating.

Note:

Feedback is **Case-Sensitive**.

The date format is **(MM/dd/yyyy).**

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

1. Get the values from the **user** as per the Sample Input

2. Call the method **GetCustomerFeedback.**

3. Call the method **CalculateRating,** if it returns **0** then display **'Invalid rating'** in Main method. Else 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 expected date

05/25/2023

Enter the installed date

05/20/2023

Enter the rating

5

Sample Output 1:

Now your rating is 7.5

Sample Input 2:

Enter the expected date

05/20/2023

Enter the installed date

05/27/2023

Enter the rating

7

Sample Output 2:

Now your rating is 3.5

Sample Input 3:

Enter the expected date

04/14/2023

Enter the installed date

04/17/2023

Enter the rating

15

Sample Output 3:

Invalid rating

3.

Scenario:

The Public Cricket Academy is a facility where young or aspiring cricketers can go to receive training and coaching in the sport. These academies typically have experienced coaches on staff who provide instruction on various aspects of the game, such as batting, bowling, fielding, and strategy. The coach of the Public Cricket Academy (PCA) wants to store the players run details and calculate the average runs secured by individual players on his team.

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

Functionalities:

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

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

implement the features listed below.

Method	**Description**
public void AddScoreDetails(int runScored)	This method is used to add the player's **runScored** passed as an argument into the **playerList.**
public double GetAverageRunsSecured()	This method is used to calculate the average run secured by a player and return it. Average runs can be calculated based on the sum of all run scores available in the **playerList** and dividing it by the number of elements in the **playerList.** If the **playerList** is empty this method should return **0.** The average run should be in two decimal places. **Hint :** Use **Math.Floor()**

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

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

2. Call the methods accordingly and display the result.

3. In the Sample Input / Output provided, the highlighted text in bold corresponds to the input given by the user and the remaining text represents the output.

Note:

- Keep the method and class 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 / Output:

```
1. Add Runs Scored

2. Average Score of the Player

3. Exit

Enter your choice

1

Enter the runs scored

150



1. Add Runs Scored

2. Average Score of the Player

3. Exit

Enter your choice
```

```
1
Enter the runs scored
50

1. Add Runs Scored
2. Average Score of the Player
3. Exit
Enter your choice
1
Enter the runs scored
50

1. Add player Details
2. Average Score for a Player
3. Exit
Enter your choice
2
Average runs secured: 83.33

1. Add player Details
2. Average Score for a Player
3. Exit
Enter your choice
3
```

Thank You

4.

Scenario:

Food Hub is the world's largest food service provider. They provide a fun and safe environment where our customers can enjoy good food made with quality ingredients at affordable prices. They serve happiness to our customers through delicious, quality meals and an extraordinary restaurant experience while working toward the greater good for our employees, community, and environment. They need software to manage their hotel bills.

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

Functionalities:

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

Data Type	**Properties**
string	BillNo
double	BillAmount
string	Status

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

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

implement the features listed below.

Method	**Description**
public List<HotelBill> GetBillDetails(string billNo)	This method is used to get the bill details based on the bill number. If the **billNo** is available in the **billList ,** it should return that item as **List**.

	If the **billNo** is not available in **billList ,** then return an empty **List.** If this method returns an **empty List**, then print **"Invalid bill number**" in **Main()** method.
public List<HotelBill> UpdateBillStatus(string billNo,string status)	This method is used to update the bill **status** based on the **billNo** . If the **billNo** is available in the **billList ,** it should return that updated item as **List**. If the **billNo** is not available in **billList ,** then return an empty **List.** If this method returns an **empty List**, then print **"Invalid bill number**" in **Main()** method.
public List<HotelBill> SortBillByStatus()	This method is used to display all the bill details available in the **billList** in descending order by bill status. The result should be **List**.

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

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

2. Call the methods accordingly and display the result.

3. In the Sample Input / Output provided, the highlighted text in bold corresponds to the input given by the user and the remaining text represents the output.

Note:

- Keep the method and class 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 / Output:

1. Get bill details

2. Update Bill Status

3. Sort Bill By Status

4. Exit

Enter your choice

1

Enter bill number

H01

BillNo	BillAmount	Status
H01	450	UnPaid

1. Get bill details

2. Update Bill Status

3. Sort Bill By Status

4. Exit

Enter your choice

2

Enter bill number

H01

Enter bill status

Paid

BillNo	BillAmount	Status
H01	450	Paid

1. Get bill details

2. Update Bill Status

3. Sort Bill By Status

4. Exit

Enter your choice

2

Enter bill number

H08

Enter bill status

Paid

Invalid bill number

1. Get bill details

2. Update Bill Status

3. Sort Bill By Status

4. Exit

Enter your choice

3

BillNo	BillAmount	Status
H04	2000	UnPaid
H01	450	Paid
H02	50	Paid
H03	500	Paid

1. Get bill details

2. Update Bill Status

3. Sort Bill By Status

4. Exit

Enter your choice

4

Thank You

5.

Scenario:

Water Wonders is a private water supplier company that specializes in providing drinking water to customers. They operate through a network of pipelines or delivery trucks, sourcing water from wells or other surface water sources. Prior to distribution, the company ensures that the water is treated to meet health and safety standards. However, they encountered challenges when it came to manually calculating water prices, prompting the need for software to streamline their billing processes.

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

Functionalities:

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

Datatype	**Property Name**
string	UsageType
double	Volume
double	PricePerGallon

In class **BillUtility,** implement the below-given methods and also **Inherit** the class **WaterBill**.

Method	**Description**
public bool ValidateUsageType()	This method is used to validate the water usage type. If the usage type is **Residential** or **Commercial** or **Industrial**, then return **true**. Otherwise return **false.**

public double WaterPriceCalculation()	This method is used to calculate the water price with tax amount and return it. The calculation procedures are given below.

Formula :

Water Price Calculation = Price + Tax amount

Price = Volume of water * Price per gallon

Usage Type	**Tax amount**
Residential	10% of the price
Commercial	20% of the price
Industrial	30% of the price

Note: Usage type is **Case-sensitive**.

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

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

2. Call the **ValidateUsageType** method, If it returns true then get the **Volume** , and **Price Per Gallon** value from the **user** and move on to step 3, If it returns false then display **Invalid usage type.**

3. Use the values in method **WaterPriceCalculation.**

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 usage type

Industrial

Enter the volume of water

1000

Enter price per gallon

30

Sample Output 1:

The total price is 39000

Sample Input 2:

Enter the usage type

Home

Sample Output 2:

Invalid usage type

Scenario:

Lisa, an aspiring entrepreneur, envisions establishing her own home appliance company. She recently acquired a small room located in the bustling city center to kickstart her venture. To initiate her business operations, Lisa decided to begin with selling washers and dryers. Unfortunately, due to the limited visibility of her newly opened shop, the sales of washers and dryers have been sluggish.

In an effort to enhance her shop's popularity and propel her business to new heights, Lisa devises an ingenious strategy. She decides to introduce a bulk offer for the washers and dryers, offering them at a discounted price. To facilitate this enticing deal, Lisa recognizes the need for software that can accurately calculate the discounted amount for each unit.

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

Functionalities:

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

Class	Properties
Machine	string Type double CubicFootCapacity double NoOfRevolution double NoOfSeconds double Price

In class **MachineDetails,** implement the below-given methods and also **Inherit** class **Machine** .

Class	Method	Description
MachineDetails	public double FindTheSpinSpeed()	This method is used to calculate the spin speed of the washing machine and return it. **Formula :** Spin Speed (RPM) = (60 x (Number of revolutions)) / (Number of seconds)
MachineDetails	public double CalculatePrice()	This method is used to calculate the price of the washing machine after the

		discount and return it. Refer to the below procedures to calculate price after the discount.

Formula :

Price after discount = Price - Discount

Type	Cubic Foot Capacity	Discount
Front Load	Between 2.0 and 5.5 Above 5.5 Less than 2.0	20% of the price 40% of the price No discount
Top Load	Between 2.0 and 5.5 Above 5.5 Less than 2.0	30% of the price 50% of the price No discount

Note : Type is **case sensitive.**

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

1. Get the values from the **user** as per the Sample Input.

2. Call the methods **FindTheSpinSpeed** and **CalculatePrice**

3. 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 load type

Front Load

Enter cubic foot capacity

2.5

Enter number of revolution

400

Enter number of seconds

20

Enter the price

15000

Sample Output 1:

Spin speed is 1200 RPM

The price after discount 12000

Sample Input 2:

Enter the load type

Top Load

Enter cubic foot capacity

1.8

Enter number of revolution

300

Enter number of seconds

30

Enter the price

10000

Sample Output 2:

Spin speed is 600 RPM

The price after discount 10000

Scenario:

Food Hub is the world's largest food service provider. They provide a fun and safe environment where our customers can enjoy good food made with quality ingredients at affordable prices. They serve happiness to our customers through delicious, quality meals and an extraordinary restaurant experience while working toward the greater good for our employees, community, and environment. They need software to manage their hotel bills.

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

Functionalities:

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

Data Type	**Properties**
string	BillNo
double	BillAmount
string	Status

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

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

implement the features listed below.

Method	**Description**
public List<HotelBill> GetBillDetails(string billNo)	This method is used to get the bill details based on the bill number. If the **billNo** is available in the **billList ,** it should return that item as **List**. If the **billNo** is not available in **billList ,** then return an empty **List.** If this method returns an **empty List**, then print **"Invalid bill number**" in **Main()** method.
public List<HotelBill> UpdateBillStatus(string billNo,string status)	This method is used to update the bill **status** based on the **billNo** . If the **billNo** is available in the **billList ,** it should return that updated item as **List**. If the **billNo** is not available in **billList ,** then return an empty **List.** If this method returns an **empty List**, then print **"Invalid bill number**" in **Main()** method.
public List<HotelBill> SortBillByStatus()	This method is used to display all the bill details available in the **billList** in descending order by bill status. The result should be **List**.

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

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

2. Call the methods accordingly and display the result.

3. In the Sample Input / Output provided, the highlighted text in bold corresponds to the input given by the user and the remaining text represents the output.

Note:

- Keep the method and class 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 / Output:

1. Get bill details

2. Update Bill Status

3. Sort Bill By Status

4. Exit

Enter your choice

1

Enter bill number

H01

BillNo	BillAmount	Status
H01	450	UnPaid

1. Get bill details

2. Update Bill Status

3. Sort Bill By Status

4. Exit

Enter your choice

2

Enter bill number

H01

Enter bill status

Paid

BillNo	BillAmount	Status
H01	450	Paid

1. Get bill details

2. Update Bill Status

3. Sort Bill By Status

4. Exit

Enter your choice

2

Enter bill number

H08

Enter bill status

Paid

Invalid bill number

1. Get bill details

2. Update Bill Status

3. Sort Bill By Status

4. Exit

Enter your choice

3

BillNo	BillAmount	Status
H04	2000	UnPaid
H01	450	Paid
H02	50	Paid
H03	500	Paid

1. Get bill details

2. Update Bill Status

3. Sort Bill By Status

4. Exit

Enter your choice

4

Thank You

Scenario:

Water Wonders is a private water supplier company that specializes in providing drinking water to customers. They operate through a network of pipelines or delivery trucks, sourcing water from wells or other surface water sources. Prior to distribution, the company ensures that the water is treated to meet health and safety standards. However, they encountered challenges when it came to manually calculating water prices, prompting the need for software to streamline their billing processes.

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

Functionalities:

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

Datatype	Property Name
string	UsageType
double	Volume
double	PricePerGallon

In class **BillUtility,** implement the below-given methods and also **Inherit** the class **WaterBill**.

Method	Description
public bool ValidateUsageType()	This method is used to validate the water usage type. If the usage type is **Residential** or **Commercial** or **Industrial**, then return **true**. Otherwise return **false.**
public double WaterPriceCalculation()	This method is used to calculate the water price with tax amount and return it. The calculation procedures are given below.

Formula :

Water Price Calculation = Price + Tax amount

Price = Volume of water * Price per gallon

Usage Type	Tax amount
Residential	10% of the price
Commercial	20% of the price
Industrial	30% of the price

Note: Usage type is **Case-sensitive**.

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

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

2. Call the **ValidateUsageType** method, If it returns true then get the **Volume** , and **Price Per Gallon** value from the **user** and move on to step 3, If it returns false then display **Invalid usage type.**

3. Use the values in method **WaterPriceCalculation.**

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 usage type

Industrial

Enter the volume of water

1000

Enter price per gallon

30

Sample Output 1:

The total price is 39000

Sample Input 2:

Enter the usage type

Home

Sample Output 2:

Invalid usage type

Scenario:

Lisa, an aspiring entrepreneur, envisions establishing her own home appliance company. She recently acquired a small room located in the bustling city center to kickstart her venture. To initiate her business operations, Lisa decided to begin with selling washers and dryers. Unfortunately, due to the limited visibility of her newly opened shop, the sales of washers and dryers have been sluggish.

In an effort to enhance her shop's popularity and propel her business to new heights, Lisa devises an ingenious strategy. She decides to introduce a bulk offer for the washers and dryers, offering them at a discounted price. To facilitate this enticing deal, Lisa recognizes the need for software that can accurately calculate the discounted amount for each unit.

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

Functionalities:

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

Class	Properties
Machine	string Type double CubicFootCapacity double NoOfRevolution double NoOfSeconds double Price

In class **MachineDetails,** implement the below-given methods and also **Inherit** class **Machine** .

Class	Method	Description
MachineDetails	public double FindTheSpinSpeed()	This method is used to calculate the spin speed

		of the washing machine and return it. **Formula :** Spin Speed (RPM) = (60 x (Number of revolutions)) / (Number of seconds)
MachineDetails	public double CalculatePrice()	This method is used to calculate the price of the washing machine after the discount and return it. Refer to the below procedures to calculate price after the discount.

Formula :

Price after discount = Price - Discount

Type	**Cubic Foot Capacity**	**Discount**
Front Load	Between 2.0 and 5.5	20% of the price

	Above 5.5 Less than 2.0	40% of the price No discount
Top Load	Between 2.0 and 5.5 Above 5.5 Less than 2.0	30% of the price 50% of the price No discount

Note : Type is **case sensitive.**

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

1. Get the values from the **user** as per the Sample Input.

2. Call the methods **FindTheSpinSpeed** and **CalculatePrice**

3. 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 load type

Front Load

Enter cubic foot capacity

2.5

Enter number of revolution

400

Enter number of seconds

20

Enter the price

15000

Sample Output 1:

Spin speed is 1200 RPM

The price after discount 12000

Sample Input 2:

Enter the load type

Top Load

Enter cubic foot capacity

1.8

Enter number of revolution

300

Enter number of seconds

30

Enter the price

Scenario:

Crazy Frank Cakes is a small bakery that specializes in custom cakes for weddings, birthdays, and other special occasions. The owner, Maria, has been running the business for the past 5 years, and while it has been successful, she feels like it has reached a plateau. She wants to take the business to the next level, but she is not sure how to do it. One day, Maria decides to hire a consultant to develop a marketing plan and create a website to promote the new services.
As their software consultant, you need to help her by developing a C# application

Functionalities:

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

Class	Properties
Cake	string CakeType string Flavour int Quantity int PricePerKg double TotalPrice

In class **CakeUtility,** implement the below-given methods and also **Inherit** class **Cake**.

Method	Description
public bool ValidateCakeType()	This method is used to validate the cake type. If the cake type is **Butter** or **Sponge** or **Chiffon**, then return **true**. Otherwise return **false**, and Also display **Invalid cake type** in Main method.
public string[ ] CalculatePrice( )	This method is used to calculate cost of the cake after the discount based on the below given formula. Then store the details such as **CakeType**, **Flavour**, **Quantity**, **PricePerKg** and **TotalPrice** in **string array** and return it.

Formula :

Total Price = (Quantity * Price Per Kg) - Discount

Flavour	**Discount Percentage**
Vanilla	5% of Total Price
Chocolate	10% of Total Price
Others	0% of Total Price

Note:

CakeType and **Flavour** is **Case-sensitive**.

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

1. Get the **CakeType**, **Flavour**, **Quantity**, and **PricePerKg** value from the **user**.

2. Call the **ValidateCakeType** method and if it returns true then move on to step 3, otherwise display **Invalid cake type** in the Main method

3. Use the values in method **CalculatePrice** 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 cake type

Butter

Enter the flavour

Chocolate

Enter the quantity

5

Enter the price per kg

200

Sample Output 1:

Cake Type : Butter

Cake Flavour : Chocolate

Quantity : 5

Price Per Kg : 200

Total Price : 900

Sample Input 2:

Enter the cake type

Chiffon

Enter the flavour

Strawberry

Enter the quantity

10

Enter the price per kg

150

Sample Output 2:

Cake Type : Chiffon

Cake Flavour : Strawberry

Quantity : 10

Price Per Kg : 150

Total Price : 1500

Sample Input 3:

Enter the cake type

Lemon

Enter the flavour

Chocolate

Enter the quantity

1

Enter the price per kg

180

Sample Output 3:

Invalid cake type

Scenario:

Retire Hub is a leading entertainment club in the city. In their last board meeting, board members decided to collect the membership fee for improving and maintaining the club in top-class condition. As an initiative, the club outlined the development of software that would calculate the membership fee.

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

Functionalities:

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

Data Type	Property Name
string	MemberId
string	MemberName
string	MemberType

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

Method	Description
public bool ValidateMemberId()	This method is used to validate the member id. The member id should be the member type, followed by **4 digits**. The member type is "**Gold**" or "**Premium**" Example : **Premium3456** If the above conditions are satisfied, then return **true**. Otherwise return **false.**
public double CalculateMembershipFees()	This method is used to calculate the membership fees and return the **fees**. If the member type is "**Gold**" , then return the membership fees of **50000**. If the member type is "**Premium**" , then return the membership fees of **75000**.

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

1. Get the **MemberId** value from the **user**.

2. Call the **ValidateMemberId** method, If it returns true, then get the **MemberName** and **MemberType** values from the user and move on to step 3, If it returns false then display **Invalid member id.**

3. Use the values in method **CalculateMembershipFees** 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 member id

Gold1123

Enter member name

Nisha

Enter member type

Gold

Sample Output 1:

Membership Fees is 50000

Sample Input 2:

Enter member id

Silver1123

Scenario:

Hamels is an experienced traveler and an avid mountaineer. He has decided that his next adventure will be climbing a mountain and he wants to choose one that is particularly tall. He has researched several mountains and narrowed down his options to a few that he is interested in climbing.

However, he wants to make sure that he chooses the tallest one out of the options he has. He wants software that provides information on the height of different mountains around the world.

You being his software consultant help him by developing a C# application.

Functionalities:

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

public static Dictionary<string, int> MountainDetails -In the code template, it is already provided.

implement the features listed below.

Method	**Description**
public void AddMountainDetails (string[ ] mountain)	This method is used to add the mountain details into the **MountainDetails** dictionary. This method should separate the values in each string by a **colon(:)** from the input array and store them in a **Dictionary**.
public int FindMountainHeight (String mountainName)	This method is used to find the height of the mountain based on the **mountainName** passed as an argument. If the mountain name is found in the Dictionary, then return that mountain **height**. else, return **-1** and print "**No mountains are available**" in the Main method.
public List <String> FindTheHighestMountains()	This method is used to find the highest height mountain from the **MountainDetails** dictionary, then store the

	mountain name as a **list** and return it. **Note:** If the highest height has more than one mountain, then consider that all are the highest height mountain s, and they need to be added to the list.

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

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

2. Call the methods accordingly and display the result.

3. In the Sample Input / Output provided, the highlighted text in bold corresponds to the input given by the user and the remaining text represents the output.

Note:

- Keep the method and class 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 and Output :

1. Add Mountain Details

2. View Mountain Height

3. View Mountains With Highest Height

4. Exit

Enter the choice

1

Enter the number of entries

5

Mount Elbrus:7970

Aconcagua:7827

Manaslu:4510

Mount Vinson:7970

K2:6120

1. Add Mountain Details

2. View Mountain Height

3. View Mountains With Highest Height

4. Exit

Enter the choice

2

Enter the mountain name needs to be searched

K34

No mountains are available

1. Add Mountain Details

2. View Mountain Height

3. View Mountains With Highest Height

4. Exit

Enter the choice

```
2

Enter the mountain name needs to be searched

Aconcagua

Height is : 7827



1. Add Mountain Details

2. View Mountain Height

3. View Mountains With Highest Height

4. Exit

Enter the choice

3

Mountain names with the highest height are:

Mount Elbrus

Mount Vinson



1. Add Mountain Details

2. View Mountain Height

3. View Mountains With Highest Height

4. Exit

Enter the choice

4

Thank you.
```

Scenario:

Treasure Mine, a jewellery store, is planning to develop a software system to keep track of their stock. The store specialises in unique and high-end jewellery pieces, and they want to ensure that they can easily keep track of their inventory levels and sales. The store wants to be able to quickly and easily see how many of each item they have in stock, so they can reorder items that are running low.

You being their software consultant help them with the process by developing a C# application.

Functionalities:

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

public static Dictionary<String, int> ProductDetails -In the code template, it is already provided.

implement the features listed below.

Method	**Description**
public void AddProductDetails (string itemName, int quantity)	This method is used to add the product details such as **itemName** and **quantity** passed as an argument into the **ProductDetails** dictionary.
public List<string> CheckReorderLevel(int reorderLevel)	This method is used to find the **items name** that have less stock than the **reorderLevel** passed as an argument. If the stock is available less than the reorder level, then store the items name as **list** and return it. else, return an **empty list** and print "**No need for reorder**" in Main method.

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

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

2. Call the methods accordingly and display the result.

3. In the Sample Input / Output provided, the highlighted text in bold corresponds to the input given by the user and the remaining text represents the output.

Note:

- Keep the method and class 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 and Output 1:

Enter the number of products

3

Enter the item name

Diamond set

Enter the stock quantity

100

Enter the item name

Gold chain

Enter the stock quantity

2000

Enter the item name

Platinum set

Enter the stock quantity

197

Enter re-order level

500

List of Products - reorder level below 500

Diamond set

Platinum set

Sample Input and Output 2:

Enter the number of products

2

Enter the item name

Diamond set

Enter the stock quantity

1500

Enter the item name

Gold chain

Enter the stock quantity

2000

Enter re-order level

500

No need for reorder

Scenario:

George owns one of the leading detergent stores in the city with two outlets. He plans to automate the process in his store as the sales increase. He wants the system to store detergent details and help him generate bills.

Detergent name, packet capacity (in grams), and cost per pack are the expected details to be stored in the system. And whenever a customer buys a detergent packet, the bill needs to be calculated for the given quantity (in kg).

You being their software consultant help them with the process by developing a C# application.

Functionalities:

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

Data Type	Property Name
string	BillId
string	Name
int	GramsInPack
double	CostPerPack

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

Method	Description
public bool ValidateBillId( )	This method is used to validate the **bill id**. The bill id should consists of **5** characters, the first **two** characters should be **digits**, followed by **colon (:)** and the last **two** characters should be **alphabets** in upper case. If the above conditions are satisfied, then return **true**. Otherwise, return **false.**
public double CalculateTotalCost(float quantity)	This method is used to calculate the total cost and return the **cost**. Refer to the below formula to calculate the total cost.

Formula :

Total Cost = (CostPerPack * (quantity * 1000) / GramsInPack)

1. Get the **BillId** value from the **user**.

2. Call the **ValidateBillId** method, If it returns true, then get the **Name**, **GramsInPack**, **CostPerPack** and **quantity** values from the user and move on to step 3, If it returns false then display **Invalid bill id.**

3. Use the values in method **CalculateTotalCost** 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 bill id

12:AB

Enter detergent name

Arial

Enter pack capacity in grams

250

Enter cost per pack

80

Enter Quantity to purchase in kgs

1

Sample Output 1:

Detergent cost is 320

Sample Input 2:

Enter bill id

123-ABC

Sample Output 2:

Invalid bill id

Scenario:

Alia Cabs has emerged as a rapidly expanding startup, riding the wave of technological advancements in the Internet and consumer space. This has presented them with an exciting market opportunity, enabling them to incorporate cutting-edge technology into their operations. With a diverse fleet comprising hatchbacks, sedans, and SUVs, Alia Cabs aims to cater to the varying needs of their customers. To streamline their services, they are seeking innovative software that can accurately calculate fares based on the specific vehicle type selected by passengers.
As their software consultant, you can help them by developing a C# application

Functionalities:

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

Class	Properties
Cab	string BookingID string CabType double Distance double Fare

In class **CabDetails,** implement the below-given methods and also **Inherit** class **Cab** .

Class	Method	Description
CabDetails	public bool ValidateBookingID()	This method is used to validate the booking id. In this method extract the booking id by character **(@).** The id should have **AC** before the character **@** and three numbers after the character **@.** **For Example : AC@123** The above conditions are passed then return **true**, Otherwise return **false**
CabDetails	public Cab CalculateFareAmount()	This method is used to calculate the fare of the cab bookings based on the below formula. Store the result in the property **Fare** Then store the details such as **BookingID**, **CabType**, **Distance** and **Fare** in **Cab** object and return it. Refer to the table below to calculate fare.

Formula :

Fare = Distance * Price per km

Cab Type	Price per km
Hatchback	10
Sedan	20
SUV	30

Note : Cab type is **case sensitive.**

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

1. Get the values from the **user** as per the Sample Input

2. Call the method **ValidateBookingID**, if it returns **true** then move on to next step, else display **"Invalid booking id"** in Main method

3. Call the method **CalculateFareAmount** 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 booking id

AC@234

Enter the cab type

SUV

Enter the distance in km

10

Sample Output 1:

Booking Id : AC@234

Cab Type : SUV

Distance : 10

Fare : 300

Sample Input 2:

Enter the booking id

DC@123

Sample Output 2:

Invalid booking id

Sample Input 3:

Enter the booking id

AC@123456

Sample Output 3:

Invalid booking id

Scenario:

As the sun shines brightly on this picturesque spring day, Maria, the proprietor of The Plant Haven, a thriving plant store in the heart of the town, busies herself with curating a diverse array of botanical offerings for her valued customers. From colorful annuals and hardy perennials to fragrant herbs and nutritious vegetables, she has something for everyone. To sweeten the deal, she even offers generous discounts to her patrons. As the day progresses, Maria is delighted by the steady stream of customers who come through her doors. Recognizing the need to streamline her operations, she decides to invest in software that can accurately calculate the cost of her plants after discounts have been applied.
You being her software consultant, help her by developing a C# application

Functionalities:

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

Datatype	Property Name
string	PlantName
int	NoOfSapling
string	Category
int	PricePerSapling

In class **PlantUtility,** implement the below-given methods and also **Inherit** class **Plant**.

Method	**Description**
public Plant ExtractDetails(string plantDetails)	This method is used to extract plant details from a **colon(:) - separated** input **string** and assign the values to **Plant** class **object** and return that**.** In this method, pass the plant details as an argument based on the below-given string format. **<PlantName:NoOfSapling:Category:PricePerSapling>**
public double CalculateCost()	This method is used to calculate the cost of the plant after the discount and return it as a **double** datatype. **Total Amount** = **NoOfSapling * PricePerSapling** Refer to the below procedure to calculate cost after the discount.

Formula :

Cost after the discount = Total Amount - Discount

Total Amount	**Discount**
> 500 to <=1000	10 % of Total Amount
Above 1000	20 % of Total Amount
<=500	No discount

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

1. Get the input details from the **user**.

2. Call the **ExtractDetails** method by using the input values.

3. Call the **CalculateCost** method and finally display all the details as per the output given.

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 plant details

Sunflower:10:flowering:70

Sample Output 1:

Plant name is : Sunflower

No of sapling is : 10

Category : flowering

Price per sapling : 70

Total cost is : 630

Sample Input 2:

Enter the plant details

Lavender:5:herb:30

Sample Output 2:

Plant name is : Lavender

No of sapling is : 5

Category : herb

Price per sapling : 30

Total cost is : 150

Scenario:

Win Cinemas is an inclusive theater that caters to everyone in our community. They take pride in enriching all their community through exceptional and daring theatrical performances, as well as engaging educational experiences. With their commitment to excellence, they guarantee a world-class theatrical journey enhanced by the mesmerizing 4K Dolby Atmos technology. They need a software to maintain movie screening details.

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

Functionalities:

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

Data Type	Property Name
string	MovieName
string	ScreenedDate
string	RemovedDate
double	Price

In class **Program,**

public static Dictionary<int,Movie> screeningDetails -In the code template, it is already provided.

Implement the below-given method.

Class	Method	Description
Program	public Dictionary<string, double> MovieScreenedMoreNumberOfDays()	This method is used to find the movie which is screened more number of days using the screened and removed date from **screeningDetails** dictionary. Then store the **movie name** and **price** in **Dictionary** and return it.
Program	public Dictionary<string, double> MovieWithScreenedDays()	This method is used to find all the movie as well as total number of screened days. Then store the **movie name** and that movie's **total number of screened days** as **Dictionary** and return it.

Note : The date format is **(MM/dd/yyyy).**

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

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

2. Call the methods accordingly and display the result.

3. In the Sample Input / Output provided, the highlighted text in bold corresponds to the input given by the user and the remaining text represents the 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.

Assume these below-given data as example,

If you need you can try this in your code also.

Sample Input / Output:

```
1. Movie screening more number of days

2. Movie with their screening days

3. Exit

Enter your choice

1

Avatar 150


1. Movie screening more number of days

2. Movie with their screening days

3. Exit

Enter your choice

2

Eternals  35

Iron Man  31

Avatar     103

LightYear  20

Black Panther  24


1. Movie screening more number of days

2. Movie with their screening days

3. Exit

Enter your choice
```

3

Thank You

Scenario:

There are various types of mobile apps and web apps that greatly assist customers of a well-known software company in the United States, which provides products to its customers. Recently, the company received an order from a client for a tax amount calculator. The employee provides their ID and salary, and the company wishes to display their tax amount based on the salary.

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

Functionalities:

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

Datatype	**Property Name**
string	EmployeeId
double	Salary

In class **EmployeeUtility,** implement the below-given methods and also **Inherit** the class **Employee** .

Method	**Description**
public bool ValidateEmployeeId()	This method is used to validate the employee id. Condition : 1. The employee id length should be 4. 2. It should be in the format of a capital letter followed by 3 digits **For Example :** E123

	If the above-given conditions are passed, then return **true**. Otherwise return **false.**
public double CalculateTaxAmount()	This method is used to calculate the tax amount of the employee based on their salary and return it. If the employee's salary is up to 20,000, then the tax amount is **0**. If the employee's salary is between 20,001 and 50,000, the first 20,000 is tax-free, after that, the tax rate is **10%**. If an employee's salary is between 50,001 and 100000, the first 20,000 is tax-free, after that, the tax rate is **10%**, and after 50,000, the tax rate is **20%**. If an employee's salary is above 100000, the first 20,000 is tax-free, after that, the tax rate is **10%,** and after 50,000, the tax rate is **20%,** and after 100000, the tax rate is **30%**.

For Example :

1. **Employee Salary = 45000.**

The first 20000 is tax free, and the remaining amount of 25000 is taxed at 10%.

So, **Tax Amount** = (Salary - 20000) * 0.10

2. **Employee Salary = 65000.**

The first 20,000 is tax-free, the next 30,000 is taxed at 10%, and the remaining 15,000 is taxed at 20%.

So, **Tax Amount** = (20000 * 0) + (30000 * 0.10) + ((Salary - 50000) * 0.20)

3. **Employee Salary = 155000.**

The first 20,000 is tax-free, the next 30,000 is taxed at 10%, the next 50,000 is taxed at 20% and the remaining 55,000 is taxed at 30%.

So, **Tax Amount** = (20000 * 0) + (30000 * 0.10) + (50000 * 0.20) + ((Salary - 100000) * 0.30)

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

1. Get the values from the **user** as per the Sample Input.

2. Call the **ValidateEmployeeId** method, If it returns true then get the **Salary** value from the **user** and move on to step 3, If it returns false then display **Invalid employee id.**

3. Use the values in method **CalculateTaxAmount** 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 employee id

H456

Enter the salary

15000

Sample Output 1:

Tax amount to pay is 0

Sample Input 2:

Enter the employee id

A234

Enter the salary

200000

Sample Output 2:

Tax amount to pay is 43000

Sample Input 3:

Enter the employee id

A123456

Sample Output 3:

```
Invalid employee id
```

Sample Input 4:

```
Enter the employee id
```

e123

Sample Output 4:

```
Invalid employee id
```

Scenario:

Get Holiday is a well-known tourist agency and they usually book hotel rooms for their customers based on the hotel rating. Now, manually locating the best hotel is extremely difficult. So, they need an application that can use web scraping techniques to gather information on various hotels and their ratings, and then allow the user to filter and sort through the results to find the best options for their customers.

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

Functionalities:

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

public static Dictionary<String, float> hotelDetails -In the code template, it is already provided.

implement the features listed below.

Method	**Description**
public Dictionary<String, float> SearchHotel(String hotelName)	This method is used to find the hotel details by hotel name. If the hotel name is available in the **hotelDetails,** this method should return the hotel name and rating as a **Dictionary**.

	If the hotel name is not available in the **hotelDetails ,** then it should return an empty **Dictionary**. If this method returns an empty **Dictionary,** then print **"Hotel Not Found"** in Main method.
public Dictionary<String, float> UpdateHotelRating(string hotelName, float rating)	This method is used to update the rating of the hotel by hotel name. If the hotel name is available in the **hotelDetails ,** it should return the hotel name and updated rating as a **Dictionary**. If the hotel name is not available in the **hotelDetails ,** then it should return an empty **Dictionary**. If this method returns an empty **Dictionary,** then print **"Hotel Not Found"** in Main method.
public Dictionary<String, float> SortByHotelName()	This method is used to display all the hotels available in the **hotelDetails** in ascending order by hotel name. The result should be **Dictionary**

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

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

2. Call the methods accordingly and display the result.

3. In the Sample Input / Output provided, the highlighted text in bold corresponds to the input given by the user and the remaining text represents the output.

Note:

- Keep the method and class 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 / Output:

```
1. Search by hotel name

2. Update hotel rating
```

```
3. Sort hotels by name

4. Exit

Enter your choice

1

Enter the hotel name

The Hay Adams

The Hay Adams 3


1. Search by hotel name

2. Update hotel rating

3. Sort hotels by name

4. Exit

Enter your choice

2

Enter the hotel name

Montage Kapalua Bay

Enter the rating

4

Montage Kapalua Bay  4


1. Search by hotel name

2. Update hotel rating

3. Sort hotels by name

4. Exit
```

```
Enter your choice
2
Enter the hotel name
Line Bay
Enter the rating
5
Hotel Not Found

1. Search by hotel name
2. Update hotel rating
3. Sort hotels by name
4. Exit
Enter your choice
3
Jungle Resort   4.5
Mandarin Oriental   5
Montage Kapalua Bay   4
The Greenwich Hotel    5
The Hay Adams    3

1. Search by hotel name
2. Update hotel rating
3. Sort hotels by name
4. Exit
```

Enter your choice

4

Thank You