1. Fresh Farm is one of the best producers of fresh farm products in the city. Their products include dairy products, vegetables, and fruits. They want an estimate of the maximum product which is sold from the farm each day. Develop a java application for the same using Streams.

**Component Specification: Carton (POJO Class)**

|  |  |  |
| --- | --- | --- |
| **Type (Class)** | **Attributes** | **Methods** |
| **Carton** | String productName  int quantity  double productCost | Getters and Setters are given in the code skeleton. |

**Component Specification: CartonUtility**

|  |  |  |  |
| --- | --- | --- | --- |
| **Type (Class)** | **Attributes** | **Methods** | **Responsibilities** |
| **CartonUtility** | List<Carton> cartonList |  | Provide the setter and getter for the class attribute. |
| **CartonUtility** |  | public Stream<Carton> **convertToStream**() | Convert the list of Carton objects to the stream of Carton and return it. |
| **CartonUtility** |  | public Carton **findMax**(Stream<Carton> stream1) | Find the maximum quantity of product sold for that day and return that object. |

**Note:**The class and methods should be declared as public and all the attributes should be declared as private.

**Note:**

[-](https://capgemini.tekstac.com/mod/vpl/view.php?id=6844&userid=5209)**The number of cartons should be a valid natural number, if fails print as "Invalid".**

-      **The number of quantities should be a valid natural number, if fails print as "Quantity number should be a valid number".**

-       **Input records are entered in the format: productName/ quantity/ productCost.**

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

-       Ensure to follow the object-oriented specifications provided in the question description.

-       Ensure to provide the names for classes, attributes, and methods as specified in the question description.

-       Adhere to the code template, if provided.

**Sample Input/Output 1:**

Enter the number of cartons

**3**

Enter carton details

**egg/40/300**

**brinjal/50/600**

**turnip/60/900**

turnip had the highest quantity with 60 nos

**Sample Input/Output 2:**

Enter the number of cartons

**2**

Enter carton details

**brinjal/0/0**

Quantity number should be a valid number

**Sample Input/Output 3:**

Enter the number of cartons

**0**

Invalid

2. A famous fruit stall in the marketplace approaches you to create an application in which their customers can estimate the total bill amount for the fruits in the basket. As a Java developer, create a Java application to add fruits to the basket and calculate the bill amount.

**Component Specification: FruitBasket (POJO Class)**

|  |  |  |
| --- | --- | --- |
| **Type (Class)** | **Attributes** | **Methods** |
| **FruitBasket** | String fruitName  int weightInKgs  int pricePerKg | Getters and Setters as well as three arguments constructor are given in the code skeleton. |

**Component Specification:**FruitBasketUtility

|  |  |  |
| --- | --- | --- |
| **Type (Class)** | **Methods** | **Responsibilities** |
| **FruitBasketUtility** | List<FruitBasket> fruitBasketList | Getters and Setters are given in the code skeleton. |
| **FruitBasketUtility** | public void **addToBasket**(FruitBasket fbObj) | This method of the FruitBasketUtility class adds the FruitBasket object into the fruitBasketList. |
| **FruitBasketUtility** | public int **calculateBill**(Stream<FruitBasket> fruitBasketStream) | This method calculates the total bill amount from the Stream of FruitBasket objects. Each object will have a fruit detail. On multiplying the weightInKgs and pricePerKg of each FruitBasket object, the individual bill amount for that particular fruit can be calculated. This method should return the total bill amount by adding each individual bill amount of fruits present in the Stream. |

**Note:**The class and methods should be declared as public.

In the UserInterface class for retrieving the list of FruitBasket objects from the FruitBasketUtility class and converting the List of objects into a Stream of objects. Then pass the Stream of FruitBasket objects to the calculateBill method in the FruitBasketUtility class and display the total bill amount.

**User Interface:**

* Display the options to choose as "Select an option: 1. Add Fruit to Basket 2. Calculate Bill 3. Exit" for all iterations.
* Any valid option can be chosen as 1 or 2 or 3. Otherwise, display a message as "Invalid option. Please try again." and continue to display the options.
* For option 1: get the fruitName, weightInKgs and pricePerKg as inputs to process the functional requirements and continue to display the options.
* For option 2: retrieve the list as mentioned in the functional requirements. If the retrieved list is empty, then display "**Your basket is empty. Please add fruits**." and continue to display the options. Otherwise, display the total bill amount as "**The estimated bill amount is Rs <total bill amount>"** and continue to display the options.
* For option 3: display a message "**Thank you for using the application**" and terminate the program.

**Note:**

* In the Sample Inputs/ Outputs provided, the highlighted text in bold corresponds to the input given by the user and the rest of the text represents the output.
* Few of the User Interface requirements will be provided in the code template itself. Adhere to the code template. Enclose your code in the respective required blocks alone.
* Do not edit or delete the codes provided in the code template.
* Adhere to the Sample Inputs/ Outputs.

**Do not use System.exit(0) for terminating the program.**

**Sample Input/ Output 1:**

Select an option:

1.Add Fruit to Basket

2.Calculate Bill

3.Exit

**2**

Your basket is empty. Please add fruits.

Select an option:

1.Add Fruit to Basket

2.Calculate Bill

3.Exit

**1**

Enter the fruit name

**Mango**

Enter weight in Kgs

**5**

Enter price per Kg

**30**

Select an option:

1.Add Fruit to Basket

2.Calculate Bill

3.Exit

**2**

The estimated bill amount is Rs 150

Select an option:

1.Add Fruit to Basket

2.Calculate Bill

3.Exit

**1**

Enter the fruit name

**Apple**

Enter weight in Kgs

**2**

Enter price per Kg

**80**

Select an option:

1.Add Fruit to Basket

2.Calculate Bill

3.Exit

**1**

Enter the fruit name

**Kiwi**

Enter weight in Kgs

**3**

Enter price per Kg

**45**

Select an option:

1.Add Fruit to Basket

2.Calculate Bill

3.Exit

**2**

The estimated bill amount is Rs 445

Select an option:

1.Add Fruit to Basket

2.Calculate Bill

3.Exit

**3**

Thank you for using the application.

**Sample Input/ Output 2:**

Select an option:

1.Add Fruit to Basket

2.Calculate Bill

3.Exit

**4**

Invalid option. Please try again.

Select an option:

1.Add Fruit to Basket

2.Calculate Bill

3.Exit

**2**

Your basket is empty. Please add fruits.

Select an option:

1.Add Fruit to Basket

2.Calculate Bill

3.Exit

**3**

Thank you for using the application.