Overview

In this task, you will have to simulate a shopping cart for GroceryItems. GroceryItems like Sugar, Rice, Fruits etc. are purchased in kgs and grams. So, **GroceryItem** will have a **name** & **pricePerKg**. We will save the quantity first in a **String** as “6kg 250g” and then **extract the quantity in float** as 6.25 .
As a user may not purchase every item, saving **quantity** & calculated **price** in GroceryItem class is **overhead**. Hence, we will extend GroceryItem class as **GroceryCartItem** with quantity & calculated price additional parameters.
At last, we will create a class **Cart** which will have **ArrayList<GroceryCartItem>** and **totalAmount**.

Defining Class

- Define a `class GroceryItem` with the following data members :
  - name (String)
  - pricePerKg (int)

Defining Constructors

- Define a **parameterized constructor** passing name & pricePerKg to it

---

Inheritance

- Define another `class GroceryCartItem` extending `GroceryItem class` with following additional data members :
  - price (int)
  - quantity (float)
- Define a parameterized Constructor for `class GroceryCartItem` by passing name & pricePerKg to it. Invoke `super(name, pricePerKg)` in it.
- Override **toString()** for `class GroceryCartItem` and return a string in the following format :

  “`Sugar (₹35 X 3.0kg) = ₹105.0`”

  i.e. “`<name> (₹<PricePerKg> X <quantity>kg) = ₹<price>`”

  *Tip : You can use either **StringBuilder** or **String.format()***

Static Methods

- Define a static method -

  `float extractQuantity(String quantityStr)`

in class GroceryCartItem to extract quantity as float from string.
Example : “5kg 500g” -> 5.5, “0kg 250g” -> 0.25

Steps :
- Learn about the split() method of String class.
- split() will help you get String[] as
  “5kg 500g” -> [“5kg”, “500g”]
- Learn about the replace() method of String class.
- replace() will help you get rid of kg & g in string as
  [“5kg”, “500g”] -> [“5”, “500”]
- Learn about the static method Integer.parseInt(). It will help you get int from string as “5” -> 5 & “500” -> 500
- Rest you know how to form float 5.5 from ints 5 & 500

- Define a static method -

  `GroceryCartItem createNew(GroceryItem item, String quantityStr)`

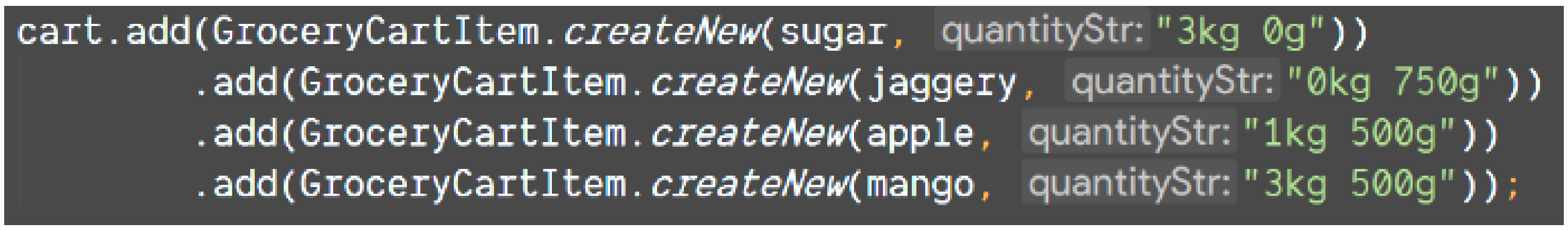
  in class GroceryCartItem as follows :
  - Create a new object of class GroceryCartItem as cartItem, call its constructor and pass (item.name, item.pricePerKg) to it
  - Set cartItem.quantity by invoking extractQuantity() method
  - Calculate & set cartItem.price
  - Return the object

Cart Class

- Define a class Cart with the following data members :
  - cartItems (ArrayList<GroceryCartItem>)
  - totalAmount (float)
- Define an empty constructor and in it :
  - Instantiate cartItems as new ArrayList<>()
  - Set totalAmount to 0
- Define a method

  `Cart add(GroceryCartItem item)`

  and add the item to list cartItems. Also update the totalAmount.
  Return this at the end so that we can chain the methods as



```
cart.add(GroceryCartItem.createNew(sugar, quantityStr: "3kg 0g"))
        .add(GroceryCartItem.createNew(jaggery, quantityStr: "0kg 750g"))
        .add(GroceryCartItem.createNew(apple, quantityStr: "1kg 500g"))
        .add(GroceryCartItem.createNew(mango, quantityStr: "3kg 500g"));
```

- Override toString() method to return string in the following format

```
Cart{
    groceryCartItems = [
        Sugar (₹35 X 3.0kg) = ₹105.0,
        Jaggery (₹40 X 0.75kg) = ₹30.0,
        Apple (₹100 X 1.5kg) = ₹150.0,
        Mango (₹60 X 3.5kg) = ₹210.0
    ],
    totalAmount = ₹495.0
}
```

*Tip : Use can use **StringBuilder**, '\n' & '\t' for this purpose*

Driver Code

- In the **Main class**,
    - Create 4 groceryItems of your choice
    - Create a new object of **class Cart** as **cart**
    - Add all the 4 items to the cart by using GroceryCartItem.create() and cart.add() method chaining. You may enter a quantity of your choice.
    - Print the cart

Future

Drawbacks of current simulation which will be eliminated in **next task :**

- A User can add the same item multiple times to the cart. It should be replaced by previous quantity. We will learn about HashMap and then solve this problem.
- QuantityStr like "5kg" and "250g" are not supported but only like "1kg 250g" is. So, we will see how we can allow such quantities also. If you are curious and have time, think of solving it now itself.
- Items can't be removed from cart

Submission

- Create a new package inside previous project's src folder named "Task2" and place all of the files for this task in it
- Push the code to GitHub
- Fill this form