CIS 41A – Lab 5: OOP, Inheritance, Regular Expression

Write a program that lets customers at a cafeteria purchase food and drink

**Overview**

The customer is presented with a menu of several items of hot food, packaged food, and drinks, along with their prices.

The customer chooses the food / drink, and then a receipt of all items, prices, and total price is printed for the customer.

**Program Design Overview**

The program has several classes:

* A UI class to interact with the user: print a menu, read in the user choices, and prints the receipt.
* An Inventory class that reads in all foods and drinks from an input file and store all items in containers. Each item is an object of the sale item inheritance hierarchy.
* A sale item inheritance hierarchy of your choice, to support these types of sale items:
  + *Packaged food*: food that are prepackaged (such as a bag of chips or a boxed salad)  
    with attributes: name, id number, price  
    Example: Granola Bar, 10, 2.50
  + *Hot food*: food that are prepared and heated (like pizza)  
    with attributes: name, id number, price, tax  
    Example: Pizza, 20, 9.00  
    - The tax rate should be a constant and set to 9.13% (rate for SC county)
  + *Drink*: bottled or canned drinks  
    with attributes: name, id number, price, size, tax, CRV  
    Example: Soda, 30, 1.50, S  
    - The CRV should be a constant and set to 0.05 for a small drink (S above) or 0.10 for a large drink (L). The CRV will be added to the price when the customer buys the drink  
    - The tax rate should be a constant and set to 9.13% (rate for SC county). The tax applies to the CRV also.

**The sale item inheritance hierarchy**

Design:

* All shared attributes and methods should be pushed up into the superclass. Try not to duplicate a method or attribute at the subclass level, when this attribute can appear one time in the superclass.
* A subclass is a specialization of its superclass, with specialized attributes / methods.  
  Conversely, if a subclass has no specialized attributes / methods, then there’s no need to create this subclass.

Requirements:

* The \_\_repr\_\_ method should return:
* For the packaged food: a string in the format name(id number):price
* For the hot food: a string in the format name(id number):price - heated
* For the drink: a string in the format name(id number):price
* Price calculation: each object is responsible for calculating its final price, whether with or without tax, with or without CRV

**The Inventory class**

* The input file is *items.csv*. Each line of the file is for one food or drink item, in the format:  
   id number,name,price,optional size
  + Example: 10,Granola Bar,3.50
  + For the foods there is no size field. For the drinks the size field is either ‘S’ or ‘L’ (for small or large)
  + A packaged food id number is in the 10-19 range, the hot food id number is 20-29, the drink id number is 30-39
* The \_\_init\_\_ method:
* reads each line of the input file
* creates and initializes an appropriate item object, based on the id number of the item. Don’t hard code the first 2 items to be one type, the next 2 items to be the next type, etc.
* stores the object in container(s) of your choice
* The \_\_repr\_\_ method returns the string with the number of each type of food or drink.
* Any other method to support the UI object

**The UI class Version 1**

* The \_\_init\_\_ method that:
* creates an Inventory object and prints the object
* or end the program if the input file can’t be opened (note that the UI object should not open files)
* A method to print the menu of food and drink for the user:
* Show a header for each food / drink type
* Print each item to see that the \_\_repr\_\_ of each item is in the correct format.  
  For each item that has tax and CRV, also print the final price of the item
* Sample output

2 packaged foods, 2 hot foods, 2 drinks # print of Inventory object

Packaged food # header

Granola Bar(10):3.5

Fruit(11):2.0

Salad(12):6.0

Hot food # header

Wings(20):9.5 - heated

10.37 # final price with tax

Pizza(21):7.0 - heated

7.64

Drink # header

Soda(30):1.5

1.69 # final price with CRV and tax

Juice(31):2.0

2.29

Water(32):1.0

1.15

END of Part 1 (11pts, due Mon 11:30, 11/14)

Make an appointment to meet with me (on Zoom is okay) to:

* Explain your sale item inheritance hierarchy
* Show your code and run it to get the menu as shown above

BEGIN of Part 2 (11pts, due Wed 11:30am, 11/16)

**UI class**

Add code to the UI class V1 to create the final UI class

* Modify the method to print the menu so that instead of printing the \_\_repr\_\_ string of the food/drink object:   
  Granola Bar(10):3.5

The code uses regular expression to extract the name and price from the string and print them in a more user friendly way: 1 Granola Bar $ 3.50  
where the 1 is the menu choice for the item.

The entire menu should be printed as below:  
 Packaged food

1 Granola Bar $ 3.50

2 Fruit $ 2.00

3 Salad $ 6.00

Hot food

4 Wings $ 9.50

5 Pizza $ 7.00

Drinks

6 Soda $ 1.50

7 Juice $ 2.00

8 Water $ 1.00

Note that the name and price in the menu must be extracted from the \_\_repr\_\_ string of the food/drink object. Don’t use a getter method to access the name and price from the object.

* Add a method for the user to order the food/drink by choosing one or more numeric choices from the menu.  
  The numeric choices are entered on one line of input, and there can be spaces or words or punctuations in between the numbers. Hint: regular expressions can help here.
* Add code to use the input numbers to print each food/drink item and its final price (including optional tax and CRV) in column format, with dollar signs where appropriate, and then print the total of all the items.  
  If the user input is not within range, print all the invalid numbers after printing the total (See sample output)
* Add code to ask the user whether to continue, and let the user buy more food/drink if the choice is ‘y’.

**Documentation**

* Put your name at the top of each source file.
* Docstring for each public method.

**Sample output** User input in blue

3 packaged foods, 2 hot foods, 3 drinks # from printing Inventory object

Packaged food

1 Granola Bar $ 3.50

2 Fruit $ 2.00

3 Salad $ 6.00

Hot food

4 Wings $ 9.50

5 Pizza $ 7.00

Drinks

6 Soda $ 1.50

7 Juice $ 2.00

8 Water $ 1.00

Enter your choice numbers: 2,5 and ummm... 8

2. Fruit $ 2.00

5. Pizza $ 7.64

8. Water $ 1.15

Total: $10.79 # all valid choices

Buy more? y/n: Y

Packaged food

1 Granola Bar $ 3.50

2 Fruit $ 2.00

3 Salad $ 6.00

Hot food

4 Wings $ 9.50

5 Pizza $ 7.00

Drinks

6 Soda $ 1.50

7 Juice $ 2.00

8 Water $ 1.00

Enter your choice numbers: let me have 6 and 12

6. Soda $ 1.69

Total: $ 1.69

Invalid choice: 12 # 1 invalid choice

Buy more? y/n: y

Packaged food

1 Granola Bar $ 3.50

2 Fruit $ 2.00

3 Salad $ 6.00

Hot food

4 Wings $ 9.50

5 Pizza $ 7.00

Drinks

6 Soda $ 1.50

7 Juice $ 2.00

8 Water $ 1.00

Enter your choice numbers on the line: -1 0 haha jk. 6

6. Soda $ 1.69

Total: $ 1.69

Invalid choice: -1, 0 # multiple invalid choices

Buy more? y/n: y

Packaged food

1 Granola Bar $ 3.50

2 Fruit $ 2.00

3 Salad $ 6.00

Hot food

4 Wings $ 9.50

5 Pizza $ 7.00

Drinks

6 Soda $ 1.50

7 Juice $ 2.00

8 Water $ 1.00

Enter your choice numbers on the line: one two three # no choice selected

Buy more? y/n: n