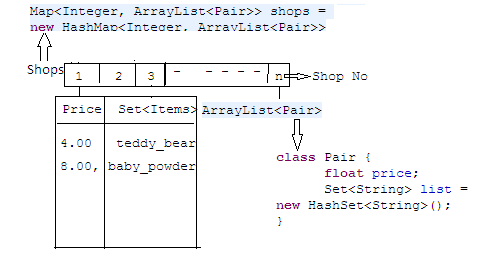
**Programming Language: JAVA**

**JARS USED:** opencsv-2.3.jar (for open csv file ).

**Data Structure Used:** HashMap, ArrayList, HashSet,.

***File: Challenge.java***

PROGRAM ARCHITECTURE



**Steps of program Execution:**

1. Make Pair class that includes two attribute.
2. Price of Item
3. List of Item (Item may be single or ComboPack)

***class*** *Pair {*

***float*** *price;*

*Set<String> list =* ***new*** *HashSet<String>();*

*}*

2. Initialize one HashMap that Include all shops including their Items with price.

*Map<Integer, ArrayList<Pair>> shops =* ***new*** *HashMap<Integer, ArrayList<Pair>>();*

That map the shop id with List of Items.

3. Initialize CSV Reader that read data from CSV File.

*CSVReader reader =* ***new*** *CSVReader(****new*** *FileReader("data.csv"), ',', '"',0);*

4. Iterate all the lines of data.csv file and store this data in map *shops*.

*while ((nextLine = reader.readNext()) != null){}*

5. *Shops* map *shopID* to *ArrayList of Pair* . *Pair* contains thePrice of Item and List of Item (Item may be single or ComboPack).

6. Take Input of List of products from user using *Scanner* and store these products in *products*

*List<String> products = new ArrayList<String>(Arrays.asList(input));*

7. Iterate every *shop* of *shops* HashMap and check required items is present or not.

*for (Map.Entry<Integer, ArrayList<Pair>> shop : shops.entrySet()) {};*

For that initialize temporary SET *tset* that is set of required products.

*Set<String> tset = new HashSet<String>(products.subList(2, products.size()))*

8. On every *Pair* of shop iterate through every ItemList of that *Pair,* and take *set difference* of *ProductList to Pair,list.*

*tset.removeAll(pl.list)*

9. If we got all required products on shop then our *tset*(Temporary set of required product) become empty.

Then we compare the *tres(Temporary result)* to previous shop’s price and store the minimum in *res variable.*

***if*** *(tres < res && tset.size() == 0) {*

*res = tres;*

*shopid = shop.getKey();*

*flag =* ***true****;*

*}*

10. Finally we print the result.

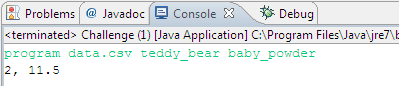
*i****f*** *(flag)*

*System.out.println(shopid + ", " + res);*

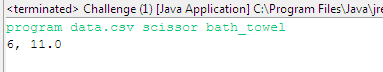
***else***

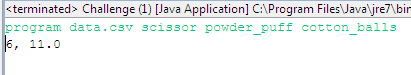
*System.out.println("none");*

Snapshots of output:









**Code Improvement:**

***File: Challenge.java***

1. This program is working fine for given set of CSV data. But It will fail if CSV data will change.

For example if we have repeated items is available on same shop like individual or in combo pack.

i.e. newData.csv

1, 4.00, teddy\_bear

1, 8.00, baby\_powder

2, 5.00, teddy\_bear

2, 6.50, baby\_powder

3, 4.00, pampers\_diapers

3, 8.00, johnson\_wipes

4 , 5.00 , johnson\_wipes

4, 2.50, cotton\_buds

5, 4.00, bath\_towel

5, 8.00, scissor

6, 3.00, bath\_towel

6, 3.00, cotton\_balls

6, 5.00, bath\_towel, cotton\_balls, powder\_puff

Input: *program data.csv bath\_towel cotton\_balls*

In this code it will return 6. 00

But actual output should be 5.00, because we can buy combo pack with min price.

Improvement in code.

For this we can create on new class *IndivisualPair.*

***public*** *IndivisualPair(****float*** *price, String item) {*

***this****.price = price;*

***this****.item = item;*

*}*

1. This class contains the actual individual price of single item and effective price of each items of combo pack and store all the *IndivisualPair* in itemList
2. effective price of items in combo: (Total price of combo)/(No of required product in combo).

*ArrayList<IndivisualPair> itemList = new ArrayList<IndivisualPair>();*

1. For above Example itemList will be :

*[ [3.00, bath\_towel], [3.00, cotton\_balls], [2.50, bath\_towel], [2.50, cotton\_balls] ]*

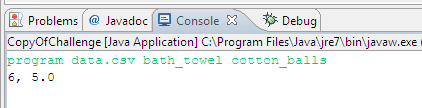
Then we sort the itemList according to effective price of item using Comparator.

1. Crate temporary set of required product List ProductList.
2. Then we iterate a loop on itemList and add the effective price in result.

After that repeat the steps of previous program.

Snapshots:

Output:



***Output for previous program.***

