Hands On Exercises - Regular Expressions

Assignment 1

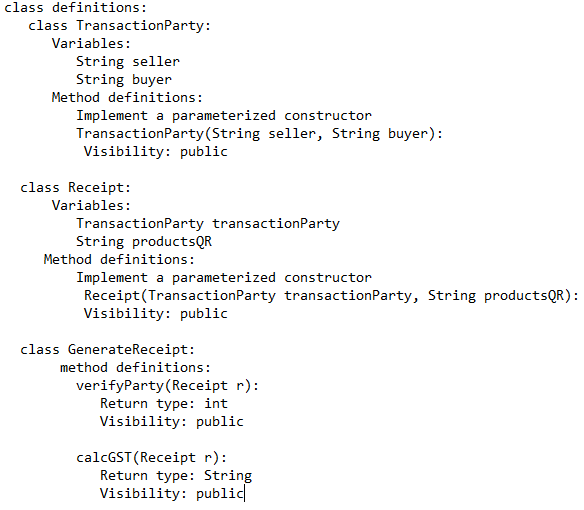
##### **Validating Users**

 Coding

##### Description

Your task here is to implement a****Java**** code based on the following specifications. Note that your code should match the specifications in a precise manner.  Consider****default visibility**** of classes, data fields and methods unless mentioned otherwise.

## ****Specifications****



****Tasks****

Class ****TransactionParty****

****-****define the ****String**** variable ****seller****

****-****define the ****String**** variable ****buyer****

-implement a parameterized ****constructor****

****class Receipt****

-declare the below instance variables and implement a parameterized constructor to initialize them.

1. ****TransactionParty**** ****transactionParty****

2. ****String**** ****productsQR****

* a string of the format: "****<Rate>,<Quantity>@<Rate>,<Quantity>@<Rate>,<Quantity>****"
* ****e.g:**** "250,10@100,3@50,7"
* only 3 products' ****Rate**** and ****Quantity**** present in the string

To access a variable in ****TransactionParty**** class through ****Receipt**** object we use:

****e.g:**** To access "****seller****" name from the ****Receipt**** object ****r**** we use:



class ****GenerateReceipt****

-Implement the below methods for this class:

****1.int verifyParty(Receipt r):****

In this method, you have to use regex to check if the names -****buyer****and ****seller****of ****TransactionParty**** object available in ****Receipt r****, are valid or not.Validate the names as per the below condition:

****Names:****

* should start and end with alphabets (uppercase/lowercase)
* can have white-spaces in between alphabets
* can have a single quote symbol OR a hyphen symbol in between alphabets
* ****e.g: Daniel D'Cruz, Giselle Dawn-Wright**** and ****Giselle Dawn****are valid names.

****Return:****

* 2 if the both names are valid
* 1 if only one name is valid
* 0 if both names are invalid.

****2. String calcGST(Receipt r):****

In this method, you have to use the ****productsQR**** variable of the ****Receipt**** ****r**** object to extract the ****Quantity**** and ****Rate**** of 3 products.

* The variable ****productsQR****will have a string value of the format ****<Rate>,<Quantity>@<Rate>,<Quantity>@<Rate>,<Quantity>****
* ****Rate**** is the price rate of the product
* ****Quantity**** is the number of units bought
* ****e.g****: ****productsQR**** = "****250,10@100,3@50,7****" has the ****Rate**** as 250, 100 and 50 while ****Quantity**** as 10, 3 and 7 respectively.

Use the ****GST\_Rate**** as ****12%.****

Calculate the value of GST using the formula:



The data type of GST should be ****int**** only. (Do NOT round-off the result)

Return the value of ****GST**** as a string value, using the ****toString()**** method.

****NOTE****

* You can make suitable function calls and use ****RUN CODE**** button to check your ****main()**** method output.

##### Execution time limit

10 seconds

import java.util.\*;

import java.lang.\*;

import java.util.regex.\*;

class TransactionParty {

}

class Receipt{

}

class GenerateReceipt{

}

class Source{

public static void main(String[] args){

}

}

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*