CS 111, Programming Fundamentals II
Homework 3: Credit Card Transactions

Computer Science

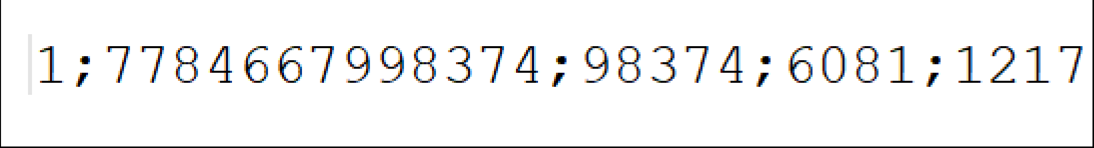
For this programming assignment you will write code to process credit card transactions. You are provided a custom exception class, and a text file containing credit card transaction information. The goal of this homework is to gain experience with throwing custom exceptions in Java. A template (***ProcessTransactions.java***) for the solution is also provided. Download the following files from canvas and explore them.

ProcessTransactions.java
CreditCardException.java
transactions1.txt

The file ***ProcessTransactions.java*** is the one you will need to finish, in order to successfully complete this assignment. Have a look at the methods, do not change the return types or the parameters lists for the methods.

The file ***CreditCardException.java*** contains three custom exceptions. The class ***CreditCardException*** inherits from the class ***Exception***. There are also two subclasses of ***CreditCardException***. The first one is ***CreditLimitExceededException***, which is thrown when the credit limit exceeds the purchase amount. The ***FradulentTransactionException*** also extends ***CreditCardException*** and is thrown when a transaction is flagged as being fraudulent.

The file ***transactions1.txt*** contains some randomly generated credit card transactions. The file contains 1000 transactions, one per line. Each transaction contains the following information: **transaction ID**, **account number**, **transaction zip code, credit limit**, and **transaction amount**. The **account number** contains the zip code of the credit card, it is the last 5 digits of the account number. Have a look at **Figure 1** for a better understanding about the file structure.



```
1;7784667998374;98374;6081;1217
```

Figure 1 : A single transaction from the file

Transaction ID = 1
Account Number = 7784667998374
Transaction Zip Code = 98374
Credit Limit = 6081
Transaction Amount = 1217

Add code to the file ***ProcessTransactions.java*** to process the transactions from the file. You need to process one transaction at a time. For the transaction to be successfully processed, the credit cards zip code has to match the transaction zip code, and the transaction amount has to be less than the credit limit. However, if the transaction amount only exceeds the credit limit by 20%, then the transaction should be approved. If the transaction amount exceeds the credit limit by more than 20% of the original limit, then the ***CreditLimitExceededException*** exception should be thrown.

Steps for each method are listed below. The entire solution can be written in less than 60 lines of codes.

I. main()

The main method needs to have a try-catch. In the try clause prompt the user to enter a file name. Use the Scanner to open the file. Use a while loop to read lines from the file and pass them to the method **processTransaction**. The while loop should run as long as there are more lines in the file. The while loop should read a single line from the file, and call the method **processTransaction**.

The catch statement should catch the exception named ***FileNotFoundException***. Add a print statement to the code block of the catch. Print out the returning value from the method ***getMessage***, which can be called on the reference variable of the exception.

II. processTransaction()

This method should also have a try- catch. Inside of the method process the transaction by calling the methods ***checkZipCodes*** and ***checkTransactionAmount***. Call both of those methods in that exact order inside of the try clause. There should be two catch statements in this method. One should catch the ***FradulentTransactionException*** exception and the other one should catch the exception named ***CreditLimitExceededException***. If neither exception is thrown, then print out that the transaction was processed successfully.

If an exception of type ***FradulentTransactionException*** is thrown, print out that the transaction has been declined and call the method ***getMessage*** to print out the reason why it was declined. Repeat the same process for the other exception.

III. checkTransactionAmount()

In this method throw an exception of type ***CreditLimitExceededException*** if the purchase amount exceeds the credit limit by more than 20% of the original limit. For example if the credit limit is 100, and the purchase amount is 110, then the transaction should be approved (100*120% = 120). If the credit limit is 100 and the purchase amount is 121, then the transaction should be cancelled.

Have an if statement to make this check, if the transaction amount exceeds the limit by more than 20% of the original limit, then use the keyword **throw**, to throw a new exception of type ***CreditLimitExceededException.***

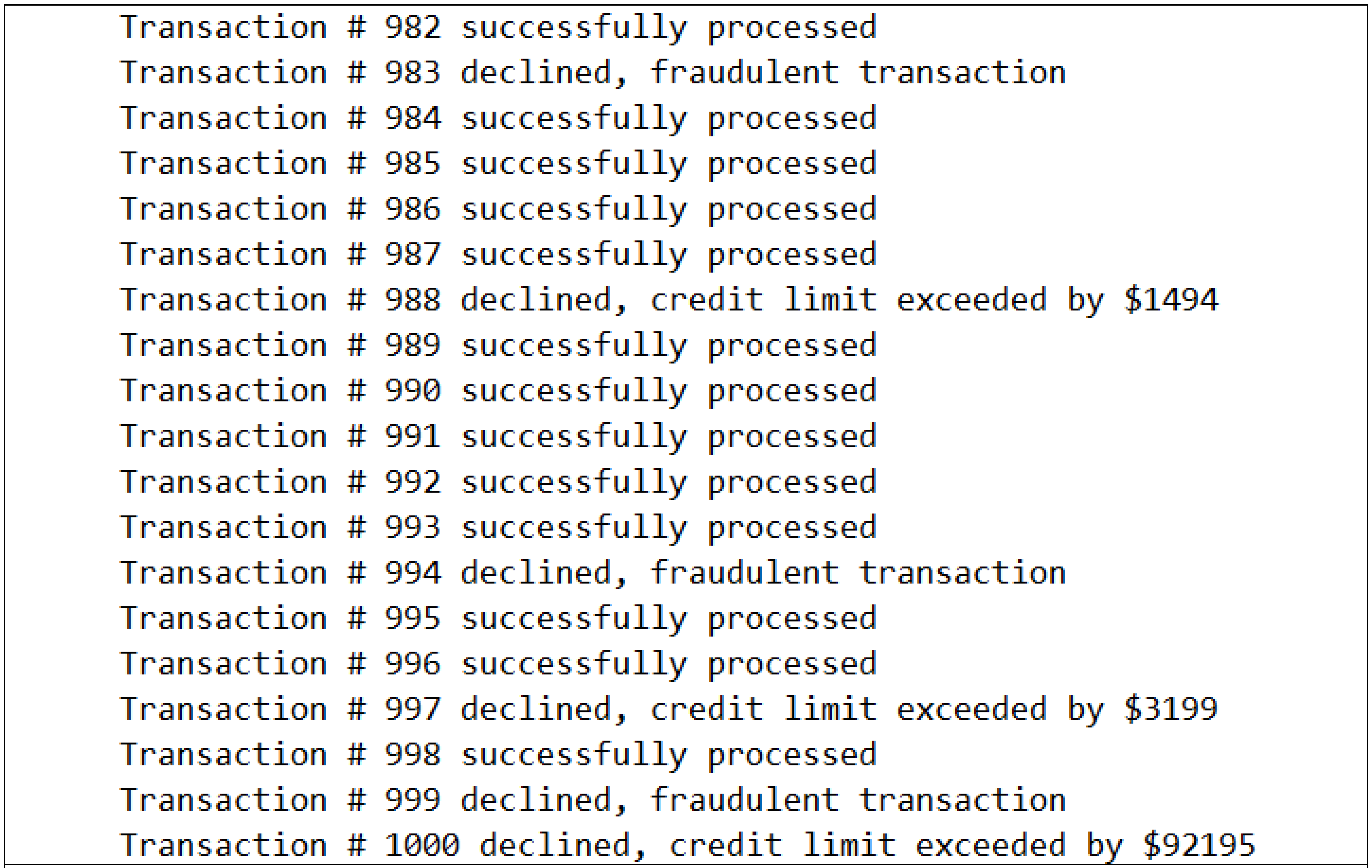
IV. checkZipCodes()

In this method throw an exception of type ***FradulentTransactionException*** if the zip code of the credit card account is different than the zip code of the transaction. The zip code of the account is the last 5 digits of the account number. Have a look at Figure 1, if you are not sure where the zip code of the credit card is located in the transaction.

If the zip codes do no match, throw an exception of type ***FradulentTransactionException.***

V. Sample Output

Sample output can be seen below in **Figure 2**. It is important that your output has the same format as the sample output. Place your ***System.out.print*** and ***System.out.println*** statements in such way that your output matches the sample output. This will help you better understand of what happens when an exception is throw, how the code proceeds after an exception has been generated.



```
Transaction # 982 successfully processed
Transaction # 983 declined, fraudulent transaction
Transaction # 984 successfully processed
Transaction # 985 successfully processed
Transaction # 986 successfully processed
Transaction # 987 successfully processed
Transaction # 988 declined, credit limit exceeded by $1494
Transaction # 989 successfully processed
Transaction # 990 successfully processed
Transaction # 991 successfully processed
Transaction # 992 successfully processed
Transaction # 993 successfully processed
Transaction # 994 declined, fraudulent transaction
Transaction # 995 successfully processed
Transaction # 996 successfully processed
Transaction # 997 declined, credit limit exceeded by $3199
Transaction # 998 successfully processed
Transaction # 999 declined, fraudulent transaction
Transaction # 1000 declined, credit limit exceeded by $92195
```

Figure 2 – Sample output

VI. Upload your work to Canvas

Make sure that you upload the following files to the Homework 3 assignment in your Canvas account:

ProcessTransaction.java
Screenshot image containing an output of the program

Using the ***Snipping Tool*** take a screenshot of the output and save it as an image.

There will be additional files (the .class files, that you've generated when compiling your code) in your homework 3 folder, but don't upload them to Canvas. Those files are not graded; they are just the byte code files that are used by the Java Virtual Machine.

V. Rubric

File / Lab	**Points**
Method *checkZipCodes* is implemented correctly and throws correct exception	10
Method *checkTransactionAmount* is implemented correctly and throws correct exception	10
Method *processTransaction* is implemented correctly and catches two specified exceptions	20
main method catches the *FileNotFoundException*, and uses while loop correctly	10
Program compiles and processes the file as expected producing the desired output	15
Code is commented and nicely formatted and a screenshot is provided	5
Total	70