Java, GUI and IDEs

InLab-9

Submission Guidelines for assignment
----

1. In the file readme.txt in the team-name directory, which contains the contribution of each team member, roll number and references (cite where you get code/code snippets from).

2. Rename the directory team-name to actual team name instead of
E.g. Coders

3. Compress the directory to <team_name>.tar.gz
e.g. coders.tar.gz

4. Submit one assignment per team. Please.

Problem 1 - q1.java

Write a program named q1.java, that takes command-line arguments, command-line arguments will be numbers. The program should display the number of arguments, their sum, and their product.
Output Format :- all three numbers,comma separated in a single line)

Ex:

```
java q1 10 4 5
```

The output should be:

```
3, 19, 200
```

Problem 2 - q2.java

Write a program that takes N as input from the user (using console, not argument). Create an array of size N and fill the array with random integers. Eventually, display their mean and standard deviation. *Hint: If you can ensure the random numbers being filled are in a range, then assessing whether your mean and standard deviation output is correct, would not be much of an issue. Once you are done assessing the correctness, try and remove that range.*

Output:
Mean
Standard deviation

Problem 3 - q3.java

Write a program that takes a string and two-position number from console and prints the substring from between those two positions. Make sure the input scheme is strictly followed as below:

```
java q3
```

Input: Hello how are you, 2, 6
Output: llo h

Input format--
<string><comma><space><number><comma><space><number>

Problem 4 - q4.java

Create a program which displays current time, it should keep updating it every 1 second. You have to use the Runnable interface to implement it (Thread).

Problem 5 - q5.java

Create a program which performs the following operation:-

Read a plain text file (sample.txt) in the string (file name provided as an argument)
Tokenize the string to remove stop words={and, the, is, in, at, of, his, her, him} and punctuation symbols.
Store the remaining words in ArrayList.
Process the ArrayList so that repeated words are removed, and display the frequency of each word.
(We would appreciate if you can utilize Collection Framework for processing ArrayList and getting the frequency. This would be graded manually. Do this one as you wish.)

OUTPUT FORMAT: Sort(increasing order) the output words list based on their frequencies.
Word \t freq

** the words mentioned are case sensitive (ex: the and The are different)

==
BONUS Question - C++ (not C; be careful; use maps/vectors/classes;
(no doubts entertained) **create a file name cq1.cpp**
==

The main program given below is meant to keep track of a fare table. You can add entries into the table, or you can query for the fare between any two cities, or you can ask for all the cities reachable from a certain city. **You are to write the required functions**. Read the given main program below and write those functions:

```
int main(){
  faretabletype faretable; // create faretabletype using typedef

  while(true){
    char command; cin >> command;
    if(command == 'x') break;
    if(command == 'a'){
      string origin, destination; cin >> origin >> destination;
      double fare; cin >> fare;
      addfare(faretable, origin, destination, fare);
    }
    else if(command == 'g'){
      string origin, destination; cin >> origin >> destination;
      double fare;
      bool found = getfare(faretable,origin, destination,fare);
      if(found) cout << fare <<endl;
      else cout << "Not found."<<endl;
    }
    else if(command == 'c'){
      string origin; cin >> origin;
      cout << connections(faretable,origin) << endl;
    }
    else cout <<"Illegal command."<<endl;
  }
}
```

Note that it is probably better to write the faretable as a class, and implement the operations as member functions. However, given that we are autograding, we would not be able to tell whether you are implementing the internals using a map -- you could do it entirely without maps!

IMPORTANT: Think about which arguments must be called by reference.

VERY IMPORTANT: before indexing into a map, you must first check if the index is defined (before writing A[x] you should check A.count(x) unless you know A[x] is defined). You have to do this for both the indices. Note that C++ allows you to write A[x] even if A[x] has not been defined and it creates A[x]. In fact, this behaviour is required: if you write fare[x][y] = z; C++ will create fare[x], first initializing it to an empty map from strings to double, and then set they index for that map to z.

Sample input

```
a mumbai pune 500
a mumbai nashik 600
a pune nashik 700
g mumbai nashik
g nashik pune
c mumbai
x
```

Output

```
600
Not found.
nashik pune
```