Computing and Algorithms I

Project 3

CS-101 **Summer, 2016**

This is a 50 point project. You will be reading data from 2 files and writing results to another file. You will be using loops for this project, but you will not be writing other classes to create your own object type, nor will you be using arrays in this project.

In project 3 you are reading lines of input from a file specified by command line argument 0 (args[0]). Each line contains sales data and an identification number for a sales person. There is another input file which contains the identification number and name of each sales person, one person's information per line of the file. This file name will be specified by command line argument 1 (args[1]). All your results will be written to a file whose name is specified by command line argument 2 (args[2]).

The format of a line of the sales data file is as follows:

- identification number – this is actually a String consisting of digits and alphabetic characters.
- one or more int values (representing a sales amount in pennies)

You will be using the pumping method of input for this file. A line containing the word quit is the sentinel for the file. The letters in the word quit may be in any form of capitalization, furthermore, there may be any other characters in that line and the word quit may be a part of any other sequence of characters in the line. For example, if the following is a line in the file:

It is no fun camping with mOsQuItoEs in Brazil!

The above line contains the word quit as part of mOsQuItoEs, thus it is not processed, it marks the end of input. The file containing the names of the sales people will have lines with the following format

- identification number – this is actually a String consisting of digits and alphabetic characters.
- name of the sales person (note this may have any number of tokens such as first, middle, and last names). The name will start with the first non-white space character after the identification number, and end with the last non-white space character on the line.

Note that there is no sentinel value for the end of the file containing names.

For each salesperson (line of the sales input file) you will be writing the name, each sale amount and the total amount of sales for that salesperson in the format described later. After processing each salesperson's data, you will write the name and sales amount of the salesperson with the highest sales and the name and sales amount of the salesperson with the lowest sales.

You will need to use constructors and methods for objects from the java API. **You will also write at least one method other than main in the solution of this problem**. Note that the money amounts in the input file are integers. Floating point values suffer from inaccuracy due to approximation of values, thus **you will not use any floating point literals, constants, or variables** in your code. The Data Table and Algorithm sections from project 2 are included here for your convenience and are virtually unchanged. I have added a statement that each of these design features is to be created for each method in your code.

Data Table

A data table is a table that has two columns. The left column will be labeled "Variable or Constant" and the right column will be labeled "Purpose". In each row you will list a variable or a constant used in your program in the left column, and in the right column you will explain its use. In your comments you will have a section whose first line will have the words "Data Table" and the remaining lines of the section will be the data table itself. Every method has its own data table.

Algorithm

Every method has an algorithm. An algorithm is a step by step solution of a problem which can be translated on a line by line basis into a computer program. We will write algorithms in pseudocode. Pseudocode has no absolute syntax rules, so the technical aspects of coding are ignored (such as blocks are contained within braces, ({}), and statements end in semicolons). For the purpose of this program, each line of pseudocode will correspond to one statement inside the main method. You will begin the algorithm section with a line consisting of the word "Algorithm". The next non blank line will consist of the name of the method and its arguments, in particular for the main method, this line will be "main(args)". The rest of the algorithm will consist of lines of pseudocode corresponding 1 to 1 with the executable statements in the algorithm. In particular, a declaration statement such as "int num" is not executable and will not appear in the pseudocode. An initialization statement such as "int num = 1" is executable and will have pseudocode something like

```
"num <-- 1"
```

.

Input

Use a Scanner object to read in a single line of text at a time from the sales input file. The name of the input file will be supplied to your code as command line argument 0 (args[0] in main). You will be using the pumping method of input. To refresh your memory, this means there will be a read before the while loop (priming the pump); the processing of the line occupies the beginning portion of the while loop; and at the end of the while loop there is another read (pumping). A line containing the word quit as described earlier is the sentinel for the file. The file can have any number of lines and characters after the sentinel, but these will be ignored by your program.

For each line of sales input, your program will need to read through the file containing names to retrieve the name of the sales person. (The identification number will be the same in both files). This means you will have to open the file of names for each line of sales data. Note that the appropriate method to retrieve the name is nextLine(). There is one problem with using the retrieved String: it will contain white space before the first letter of the name, and may contain white space after the last letter in the name. **You will use a String method to remove leading and trailing white space.**

For each line of text your code processes from the sales data file, create a Scanner object for that line (which is a String object). The first token is an identification number, all the remaining

tokens are int values. There will be one or more int tokens on the line following the identification number. Read each of these as an int.

Processing

Each sales data line is input about one salesperson. Read and process that person's information keeping track of which person has the highest sales and which person has the lowest sales. At the end of processing the sales input file, report the salesperson with the highest sales and the salesperson with the lowest sales. The int values represent sales amounts in pennies. **You will handle all numeric values using the int data type only.** When you print the amounts, the formatting will be the same style we used for doubles. By using the division and remainder operators, you can get the dollar and cents portion of the values separately, and can format them separately (using DecimalFormat for example). Note that you are not writing the identification number, you are writing the name. The second input file will give you the name of the sales person.

The file of names is allowed to have more identification numbers and names than the sales file. However, each identification number in the sales file must also be in the name file. If this is not the case, your program will print an error message and quit upon finding a number in the sales file not in the name file. (The method System.exit(2) will stop your program and report exit number 2. Any number other than 0 as an argument to exit is considered to be the number for an error exit).

Output

Your output will be written to a file. The name of the file will be supplied to your code as command line argument 2 (args[2] in main). Your output will begin with a title (something concerning a salesperson's report). Following this will be your name and course number (CS101).

For each line of the file you will print a label indicating that the next information is the name of the salesperson followed by the name. You will print a label for the sales amounts, each sales amount will be printed on a separate line and will be tabbed in from the person's name. Following this, the labelled total sales for the sales person will be printed. For example:

```
Salesperson:  Soupy Sales
     Sales Amounts:
        $6,123.45
        $67.23
     Total Sales: $6,190.68
```

The report for each salesperson will be followed by a blank line before the report of the next salesperson begins. After the reports of the individual salespeople, the name and total sales of the salesperson with the highest total sales will be printed with appropriate labels. After this (separated by a blank line) will be the name and total sales of the salesperson with the lowest sales. Label these outputs appropriately.

Style

Use white space (indentation, blank lines) to show the program structure. Use a meaningful class name, it should describe the purpose of the class. (In particular, Project3 is not a good class name,

it says nothing about sales reports). Use meaningful variable names. Follow conventions in naming identifiers. Use constants for any constant value used in your program.

Deliverables

On Monday August 22, at the beginning of class, you will turn in a document printed from a printer containing the Java source code (of course with all comments as described above). Your Java code will be printed in portrait, not landscape, mode. None of your lines in the file are allowed to extend past the right edge of the paper. In particular, this means do not have statements wrapping around to the next line (which often happens automatically). If you have multiple pages of code, they will be stapled into one package before the beginning of class.

You will submit your code (just the .java file) as a zip file using Blackboard before class begins. The submission will be in the project page for this project. Be sure to zip the code, do not use other forms of compression such as rar. The reason for zipping the code is that blackboard, for some reason and at various times, changes variable names in code. Projects not turned in via Blackboard and/or not turned in as a document will receive no credit.

Stapled to your code will be two input files (sales file and name file) and the output file your program produced by running on the input from these input files. Create your own files for testing and for this submission. Have your name in the sales data file following your sentinel line.

Grading

The program itself will be graded on 35 points. If the program does not compile, it is worth 0. If the program compiles but does not solve the problem, the score will be at most 5 points, depending on how close the solution is to being correct. If the program works correctly, that is worth 30 points, with the other five points reserved for the clarity of output. (Use white space and labels on output to make it readable).

The data tables are worth 5 points. Full credit will be given to complete and well formatted tables.

The algorithms are worth 5 points. Full credit will be given to complete and well written pseudocode.

For a program that compiles and runs, the style score will be from 0 (hard to read) to 5 (follows all conventions and style guides) points.

The total maximum score for the project is thus 50 points.

A program which does not compile will be worth 0 points. A program which does not run correctly will be worth at most 5 points.

A program which is late (not complete and ready to hand in and demonstrate at the beginning of class on Monday) will have a late penalty of 10 points if completed and demonstrated by Tuesday August 23 by the beginning of class.

If you find an error in this problem description (and are the first to report it) you may earn up to 5 bonus points (depending on the severity of the error. A maximum of 1 point if I typed a word incorrectly).

If you finish by Thursday August 18, see me to demonstrate that your program works correctly. I will have a final test file available for "official" testing. A correct program turned in and demonstrated early will also earn 5 points bonus.