|  |
| --- |
| **CS1101 Lab 13 – 2nd Comprehensive Lab** |

**Released on: Monday October, 30**  **Due on: Wednesday November, 15**

**OBJECTIVE OF THIS LAB:** You will apply, in a practical scenario, all topics covered so far in class, with an emphasis on **file input/output**, **repetition** (loops and recursion), and **arrays**.

|  |
| --- |
| You are building a company that aims to offer personal financial assistance to its clients. The financial assistance comes in the form of helping the clients track their spending by category.  To this end, you decide to build a software assistant whose usage you will sell to your clients. Here is how using this software financial assistant is expected to unfold:  1/ Get user’s basic info: (get info and create a file)   * First and last names: the first initial and last name will be used to create the user’s expenses file.   For instance, if the user’s name if Jane Davis, her expenses file should be called “JDavidExpenses.txt”.  2/ Get information about the client’s expenses: (get info and write in a file)   * Ask the user for his/her expenses. * For each expense, the user is expected to provide:   + The expense category (you are given an array of these categories), and   + The expense amount   While the user still wants to enter spending info, keep getting this information. reading it and write it into a file.   * For each piece of information entered by the user:   + If the information is incorrect (e.g., unknown category or amount that is negative or not a number), give a second chance to the user to enter it correctly, and then stop acquiring information from the client.   + If the information is correct, write it in the user’s expenses file: one line per expense. Each line should read: an expense category followed by a number (which is a dollar amount)   3/ Once the user is done entering expenses, read the file and create an array: (read file, create 1 array)   * Create and fill one array (2D) that contains for each type of expense (in the same order as the array of expenses that is given to you):   + The minimum amount spent in this category,   + The maximum amount spent in this category, and   + The total amount spent in this category (each of these amounts should be of type double).   4/ You can then allow the user to query his/her spending by category: (access arrays)   * Ask the user for the category of interest. * Provide the minimum, maximum, and total amounts spent in this category. * However, if the chosen category was not valid, allow the user two more tries and then shut down the system.   5/ Recursively compute the total amount spent by the user: (recursively access array)   * Recursively access the 2D array of expenses and a start index * To compute the total amount of expenses (which is the sum of all totals)   *Note: the partial code of this method is given to you. You just have to complete it.* |

In order to implement this financial software assistant, you have to use the java file provided to you: lab13.java. You should not modify the main method. You have to implement each of the following methods:

1/ Get basic info: Method name is GetUserInfo:

* Takes nothing as input
* Return a string: the name of the file to create

2/ Get expenses info and write file: Method name is WriteExpenses:

* Takes a string: the name of the file to write in, which is the result of the method GetUserInfo and a 1D array of the type of expenses
* Returns nothing: the file that is written in this method will still exist as a result of executing this method, even after closing the method

3/ Create an array of min, max, and total expenses per category: Method name is ExpensesStats:

* Takes:
  + A string: the name of the file to read to acquire the info that goes in the array
  + A 1D array: the array of expense categories
* Returns a 2D array of expenses per category with, for each category, the min, max, and total amounts spent in this category
* *Note: the resulting array should be of size n x 3, where n is the size of the array that contains the categories of expenses*

4/ Allow the user to query an expense category: Method name CategoryQuery:

* Takes:
  + A 1D array: the array of expense categories
  + A 2D array: the array of stats on the user’s spending in each category (resulting from method ExpensesStats)
* Returns nothing: it only prints out the queried information, which is the min, max, and total amount of the queried category
* *Note: The user is asked which category is queried within method CategoryQuery. The user is also given two more chances to enter a correct category if he/she enters an incorrect one.*

5/ Compute the total spending: Method name TotalSpending:

* Takes a 2D array: the 2D array of expenses per category
* Returns a double value: the total amount spent over all categories
* *Note: This method is expected to be implemented recursively.*

**WHAT DO YOU HAVE TO TURN IN AND HOW?**

You will turn in all your work as a zipped file: YourLastNameYourFirstNameLab13.zip on Piazza, under the folder lab13. This zipped file will include:

**Algorithm/Pseudocode**

Create a word document named “yourLastNameYourFirstNameLab13.docx”. In this document, you will write the pseudocode of each of the above methods (1 through 5).

**Java file lab13.java.**

You will implement the pseudocode created in the Word document in Java in lab13.java provided to you. Your implementation should match your pseudocode. It is OK if after starting the implementation, you refine your pseudocode.

You should only code where prompted. For instance, you are not allowed to modify the main method.

**GRADING CRITERIA**

30% - Pseudocode (word document):

* 10% - Correct method signatures
* 20% - Correct pseudocode bodies of each of the methods (1 through 5)

60% Code (Java file):

* 30% - Methods’ bodies are a true translation of what is the word file and do what they are expected to
* 5% - The java file compiles and executes properly
* 5% - Appropriate commenting
* 10% - Correct indentation
* 10% - Demo

10% Submission according to specifications

Late submission rules:

You will lose 15% for every 24 hours of lateness, for up to 72 hours. For example, if you submit 36 hours later your maximum percentage is 70%.

**EXAMPLE OF USE:**

To help you in your work, please see below, an example of use of the software financial assistant:

Please enter your first name:

Martine

Please enter your last name:

Ceberio

Your expenses will be stored in a file named: MCeberioExpenses.txt

Please enter a new expense information as prompted.

Please enter an expense category:

Vacation

Please enter an amount for this expense:

25

Do you have more expenses to enter?

yes

Please enter a new expense information as prompted.

Please enter an expense category:

Utilities

Please enter an amount for this expense:

47

Do you have more expenses to enter?

yes

Please enter a new expense information as prompted.

Please enter an expense category:

Vacation

Please enter an amount for this expense:

13

Do you have more expenses to enter?

yes

Please enter a new expense information as prompted.

Please enter an expense category:

EatingOut

Please enter an amount for this expense:

26

Do you have more expenses to enter?

no

Lodging-- Min = 0.0, Max = 0.0, Total = 0.0

Utilities-- Min = 47.0, Max = 47.0, Total = 47.0

Groceries-- Min = 0.0, Max = 0.0, Total = 0.0

EatingOut-- Min = 26.0, Max = 26.0, Total = 26.0

Vacation-- Min = 13.0, Max = 25.0, Total = 38.0

Which category do you want to check?

Vacation

The total expenses in category Vacation is: 38.0.

The total amount of expenses is: 111.0

The resulting file MceberioExpenses.txt contains the following:

