Java Practice Exam: Expense Management System

You must submit the completed program as per Coursera.

You CANNOT:

- Work with someone else on this exam.
- Copy from someone else's exam.
- Use StackOverflow or any search engine, such as Google.com, for any reason (other than to go to the course exam).
- Use an internet search for keywords in the exam. For example, do not Google "loading a .csv file in Java".
- Discuss the exam with other students until cleared by the instructor to do so. Other students have not yet taken the exam, and you CANNOT provide academically dishonest assistance to them.

You CAN:

- Reference any material from the course or recitation that can be accessed in **Coursera**. This includes videos, readings, slides, code samples, homework assignments, quizzes, and practice coding exercises.
- Reference any online Java documentation.

Use of Piazza/Office Hours as it Relates to the Exam

- You CAN ask any question on Piazza or during office hours that directly relates to the exam logistics and/or technical errors with Codio or Coursera. For example, you can ask about a glitch in Codio unrelated to your code.
- You CANNOT ask a question specifically related to code or "how to do something". For example, you cannot ask "How should I go about writing the code for this method?". You also cannot post code and ask "Can you help me understand why my code isn't working?"

The Assignment

This Java exam will involve implementing an *expense management system*. You will be provided

with the skeleton of the program and you will have to implement the methods and write the unit tests. The design of the program has been set up for you.

In this system, expense files can be loaded, parsed, and managed in the context of the system. Expenses (represented by a single Expense class and combining into the plural Expenses class) are loaded and stored in a management system (represented by an *ExpenseManagement* class). An *ExpenseFileReader* class will take care of initially loading the expense files and doing some basic clean-up with the assistance of an *ExpenseFileParser* class.

Overall, the program will load an expense file (expenses.txt) with *ExpenseFileReader* and *ExpenseFileParser*, create *Expense* objects stored in an *Expenses* collection, then store them in the *ExpenseManagement* system. The required methods are already called in the *main* method in each section of the exam.

Exam Instructions

To help guide you through implementing the program, **the exam has been split into 4 sections**. For each section, there will be 2 tasks. The first task will be to complete the required methods in one java file and the second task will be to complete the unit testing for those methods in another java file. You will be provided with all of the necessary java files to complete the tasks in each section.

You are not required to complete the sections in a specific order. This is merely set up as a guide.

Exam Overview

For each section on the exam, the first task is to complete the required methods in the given class file. Then compile and run your code.

- Javadocs have already been provided
- You can create any number of helper methods, with appropriate names and javadocs
- Name your variables appropriately
- Add brief comments to all non-trivial lines of code
- For convenience, all of the required methods in the given class file are already called in its respective *main* method (follow along to learn the sequence of the methods). (Note, you are not required to run the *main* method in the given class file.)

The second task is to test your code by running (and passing) the provided test cases in the given test class file. (**Do not modify any of the provided test cases!)** Then **write additional test cases for each test method as noted** and make sure they pass as expected. (Note, you do not

have to write unit tests for any helper methods you write.)

The final structure of the project in the Codio Filetree should look like this:

Note, you can download the entire "submit" directory, which contains the starter code, as a .zip file. You can then import into a project within Eclipse.

Exam Outline

- **Section 1**
  - Task 1: `ExpenseFileReader.java`
    - Complete the `loadExpenses` method in the `ExpenseFileReader` class. Compile and run your code.
      - *Note: You can use the "Run Your Program - ExpenseFileReader" option in Codio to run the main method in this class.*
  - Task 2: `ExpenseFileReaderTest.java`
    - Test your code by running (and passing) the provided test cases in the `ExpenseFileReaderTest` class. Write additional test cases as noted in the `testLoadExpenses` method and make sure they pass as expected. At the end, the `testLoadExpenses` method should **have a total of 2 distinct test cases.**
      - *Note: You can use the "Run JUnit Tests" option in Codio to run all tests.*

- **Section 2**
  - Task 1: `ExpenseFileParser.java`
    - Complete the `parseExpenses` method in the `ExpenseFileParser` class. Compile and run your code.
      - *Note: You can use the "Run Your Program - ExpenseFileParser" option in Codio to run the main method in this class.*
  - Task 2: `ExpenseFileParserTest.java`
    - Test your code by running (and passing) the provided test cases in the `ExpenseFileParserTest` class. Write additional test cases as noted in the `testParseExpenses` method and make sure they pass as expected. At the end, the `testParseExpenses` method should **have a total of 2 distinct test cases.**
      - *Note: You can use the "Run JUnit Tests" option in Codio to run all tests.*

- **Section 3**
  - Task 1: `Expense.java` and `Expenses.java`
    - Complete the `equals` method in the Expense class, then the `getMonthlyExpenses`, `addExpenses`, and `addExpense` methods in the `Expenses` class. Compile and run your code.
      - *Note: You can use the "Run Your Program - Expenses" option in Codio to run the main method in this class.*
  - Task 2: `ExpenseTest.java` and `ExpensesTest.java`
    - Test your code by running (and passing) the provided test cases in the `ExpenseTest` class. Then write additional test cases as noted in the `testEquals` method and make sure they pass as expected. At the end, the `testEquals` test method should **have a total of 2 distinct new test cases.**
      - *Note: You can use the "Run JUnit Tests" option in Codio to run all tests.*
    - Test your code by running (and passing) the provided test cases in the `ExpensesTest` class. Then write additional test cases as noted in the `testAddExpense` and `testAddExpenses` methods and make sure they pass as expected. At the end, each test method should **have a total of 5 distinct new test cases.**
      - *Note: You can use the "Run JUnit Tests" option in Codio to run all tests.*

- **Section 4**
  - Task 1: `ExpenseManagement.java`

- Complete the `getMonthlyExpenses`, `getMonthlyExpenses`, `getTotalMonthlyExpenses`, and `getMostExpensiveMonth` methods in the `ExpenseManagement` class. Compile and run your code.
  - *Note: You can use the "Run Your Program - ExpenseManagement" option in Codio to run the main method in this class.*

- Task 2: `ExpenseManagementTest.java`
  - Test your code by running (and passing) the provided test cases in the `ExpenseManagementTest` class. Then write additional test cases as noted in the `testGetMonthlyExpensesIntListOfExpense`, `testGetMonthlyExpensesStringListOfExpense`, `testGetTotalMonthlyExpenses`, and `testGetMostExpensiveMonth` methods and make sure they pass as expected. At the end, each test method should **have a total of 3 distinct new test cases.**
    - *Note: You can use the "Run JUnit Tests" option in Codio to run all tests.*

Evaluation

1. Did you implement the individual methods correctly? – *30 points*
   a. Can you get the monthly expense data successfully?
   b. Does your program successfully load and parse the expenses.txt files and store it properly?
   c. Does it correctly extract the expense information
2. Did you write good unit tests? – *20 points*
   a. Did you make sure each test method has the required number of test cases?
   b. Does your program pass all of your own tests?