COMP 1030

Lab #7

Arrays

Introduction

During this assignment you will build two java classes. The first class will contain the required state and behaviour for the object, **but NO main method**. The second class will contain simply the main method to give the JRE an entry point into the program, a line to instantiate a new object based upon the first class and a few lines to exercise the functionality of the first class.

When writing your code, keep these guidelines in mind:

- Start each class with the proper javadoc comment header. The first line of that comment should be the purpose of the class.
- Provide a comment for **each** speciality method.
- Follow the layout for your class as illustrated below:

Javadoc comment header
Import statements (if required)
Class declaration
    State (instance variables/data)
    Constructor(s) (if required)
    Behaviour(s) (method(s))
Close class declaration

- Use whitespace and indentations to make your code more readable and easy to debug.
- Be sure to clearly understand your work – do not simply copy code from someone else.

Instructions: Use any IDE to complete this assignment other than a simple notepad or BlueJ.

- Create a class that acts as a blueprint for a Bank Transaction Record.
  - This class should have the following state:
    - boolean canadianFunds
    - double exchangeRate
    - int transactionNumber
    - long transactionReferenceNumber
  - This class should have the following behaviour:
    - A constructor that takes four arguments.
    - Appropriate setters and getters for all state except static variables

- Create a second class called BankTransactionRecordTestHarness to test the first class:
  - Instantiate 500 Bank Transaction Records.
  - Each record should contain random data for all state(use the java.util.Random class to accomplish this)
  - Store the reference for each object in an array.
  - Use a public **class** variable to keep track of the number of objects instantiated.

- Print out the state of each object along with the object number on separate lines followed by a blank line followed by a line of dashes (----------).
- Ask the user if they wish to complete phase two of the test. If they say no end the program. If they say yes, set the state of each object to the same default value using the appropriate setters (you choose the value).
- Print out the state of each object along with the object number on separate lines followed by a blank line followed by a line of dashes (----------).

Things to consider for success:

- Follow the layout for your class as illustrated above.
- Write a section at a time and compile after each section so you do not have a volume of compiler errors to deal with.
- Comment as you go.
- Use indents and whitespace appropriately to make your code readable.
- Be sure that you actually understand the code you have written, simply copying someone else's code will not aid in your understanding of the java language.
- Stay focused and work diligently, collaborate with others if you are stuck.