CSc 110 Assignment 7
More Classes and Objects, with Sorting

The Bank (with lots of Bank Accounts)

Due: Before 11:50 pm on Thursday, April 5, ~~2017~~ 2018.

Learning Outcomes: When you have completed this assignment, you should understand:

- How to use a class to instantiate objects
- How to use instance methods
- How to create and use an array of objects
- How to sort and search for data in an array
- How to output to files

Programming Problem Description:

This assignment requires the writing of a program that simulates the operations of a bank, including the tracking of many bank accounts for many different clients. An object will be created for each account, then transactions will be applied to the accounts and the accounts will be output in various orders to both the screen and a file.

Examine & add comments to the Bank Account class:

The following code defines a very simple class called **`BankAccount`**. Observe that the class is similar but not the same as the one that was used in Assignment 6. It contains 3 *data items* (or *attributes*) and a collection of *instance methods* (or *behaviours*) for changing the values of that data.

```
import java.util.*;

public class BankAccount {
  private int accountNumber;
  private String ownerName;
  private double balance;

  public BankAccount() {
    accountNumber    = 0;
    ownerName = "";
    balance = 0.0;
  }

  public BankAccount(int number, String name, double initialDeposit) {
    accountNumber    = number;
    ownerName = name;
    balance = initialDeposit;
  }
```

```
  public int getAccountNumber() {
    return accountNumber;
  }

  public void setAccountNumber(int number) {
    accountNumber = number;
  }

  public String getOwnerName() {
    return ownerName;
  }

  public void setOwnerName(String name) {
    ownerName = name;
  }

  public double getBalance() {
    return balance;
  }

  public void setBalance(double newAmount) {
    balance = newAmount;
  }

  public void deposit(double amount) {
    balance += amount;
  }

  public void withdrawl(double amount) {
    balance -= amount;
  }

  public String toString() {
    return accountNumber + " " + ownerName + " " + balance;
  }
}
```

Use this code, including its methods where-ever possible: there is no need to change it.

The Bank: A program that uses the BankAccount class:

In a different Java class, called `TheBank.java` create a Bank program with the following specifications:

- The Bank's client file information is loaded from a file (named `AccountBackUp.dat`) that contains at most 1000 lines. Each line of the file contains a string representing the customer name, a unique integer identification number for the account and the current account balance in the account. (An array of `BankAccounts` should be created to hold the client file information. Use a constructor from the `BankAccount` class to instantiate each of the accounts.)
- Next, the Bank's monthly transaction file (named `TransactionsJanuary.dat`) containing a month of transaction information is loaded from a file. (There is no limit on the number of lines in the file.) Each line contains a customer name and unique identification number, the type of transaction and the amount of the transaction. The transaction type could be one of (deposit, withdrawal, interest, charge). The amount of a deposit must be added to the `BankAccount`'s balance; the amount of a withdrawal or (service) charge must be subtracted from the balance; and

interest requires adding 'amount' % interest to the `BankAccount`. Before applying each transaction, the name and identification number must be checked to see that they correspond to a single account; an error log must be added to an *error log file* (called error.dat) if they do not. Use member methods of the `BankAccount` class to alter the attributes of the appropriate `BankAccount` object.

- Output the entire updated Bank client file information (including customer name, identification number and account balance) into a file, called `TransactionsFebruary.dat`, ensuring that it is sorted in order of ascending identification number. Observe that the instance method `toString()` can be used to make a single String containing all attributes of a `BankAccount` object.
- Output, to the screen, the Bank client file information (just the customer name and account balance) to the screen, ensuring that it is sorted in order of descending account balance amount. This output should be output in groups of 20 lines, then the output stops waits for the user to hit the 'Enter' key.

HAND IN: Submit your `TheBank.java` file using the 'Assignments' link of the course web page. Your final submission must occur 11:50 pm on Thursday, April 5, 2018.

To assist you during this last week of the term:

- TheBank.java program has been started for you and is provided.
- A sample sorting program is provided. This sample program sorts integers, whereas your program needs to sort `BankAccounts`. You can use the sort method provided in your `TheBank.java` file, as long as you give credit, and then alter it appropriately to suit your work.

More to Discover

- Searching and sorting algorithms have been studied by Computer Scientists for nearly as long as the discipline has existed. The challenge is to ensure an efficient solution. https://www.toptal.com/developers/sorting-algorithms/ provides demo's of many algorithms!)
- In addition to arrays, there are other techniques for storing large amounts of data. (i.e., Linked Lists https://www.studytonight.com/data-structures/introduction-to-linked-list)
- These are topics that are considered in CSc 115, why not give it a try?