C212 Lab 10

Intro to Software Systems

Instructions:

Please complete the following exercises. For each, review the requirements given below and include all stated specifications. Please compress your project, which contains all of your files, into a .zip and submit it through Canvas. The grading scheme will also be provided in Canvas, which you may review as you work.

Lab: Recursion and Java Collections Framework

Part 1: Recursive v/s Iterative (50 pts)

In this part of the Lab, you will be asked to create four different methods that determine whether different inputs are palindromes. This will test your knowledge of recursive versus iterative solutions to different problems, and you will be expected to write your methods according to the way that it is asked in the prompt, recursive or iterative.

The first method that you will make is one that iteratively checks whether an int is a palindrome. As a note, you cannot convert this int to String for processing.
Boolean isIntPalindromeIterative(int num)

The second method will do the same thing, checking iteratively, but with a String input.
Boolean isStrPalindromeIterative(String str)

Then, you will create two more methods that check whether an int or String is a palindrome using a recursive method.
Boolean isIntPalindromeRecursive(int num)
Boolean isStrPalindromeRecursive(String str)

Note: Any String with only one letter or any int with only one number is by default a palindrome.

Examples:
isIntPalindromeIterative(121) -> true
isIntPalindromeRecursive(121) -> true

isIntPalindromeIterative(122) -> false
isIntPalindromeRecursive(122) -> false

isStrPalindromeIterative("racecar") -> true
isStrPalindromeRecursive("racecar") -> true

isStrPalindromeIterative("racecars") -> false
isStrPalindromeRecursive("racecars") -> false

Two separate versions of each palindrome method are expected, *isIntPalindrome…* and *isStrPalindrome…* But you can name them differently.

Part 2: Bank Accounts (50 pts)

You will be developing a program that takes a text file as input containing all the transactions of the people having their bank accounts with the bank. The bank wants you to write a program that takes a file containing these transactions as input and returns the list of people with their individual current bank account balances. You will be achieving this with the help of **HashMap.**

Note: Each line in the file represents the transaction entry where the 1st value is the name of the individual, 2nd value is the amount and 3rd value indicates either the amount was deposited or withdrawn (d - deposit and w - withdraw).
Eg: John,200,d
    Mary,50,w
You can assume that everyone in this file starts with an initial balance of 0 and that there will always be a deposit before a withdraw so that their balances to not go negative. Additionally, if there are people with the same names in the file, you can assume that they are the same person making additional transactions.

1.Create a class BankAccount

The constructor for this class has the following signature BankAccount*(String filename)* where filename is the name of the file that will be processed.

- The constructor should open the appropriate file
- Add each transaction entry in the file in the **ArrayList<String>**

Eg: transactionList = {"John,200,d","Mary,50,w"}

2. Now create a method currentBalances that takes the above created ArrayList as an input and returns a **HashMap<String, Integer>** containing current bank balances of each individual where key will be the individual's name and value will be their current balance.
Eg: allBalances = {key:"John" value:"200", key:"Mary" value:"150"}
You must use an **Iterator** to iterate over the ArrayList.
Hint: You might want to have a look at several HashMap methods like containsKey(), put(), get()

3. Now the bank comes up with a scheme where they decide to give $100 to each individual whose current balance is greater than or equal to $1500. You need to write a method updateBalances that will use an **Iterator** to iterate over the allBalances HashMap and update the balances accordingly and return the updated HashMap.

4. Write a main method that will display the balances of all individuals before and after the scheme update by iterating over the HashMap.