Data Structures and Java, Assignment 5

Collections

Contents

[Assignment objectives 3](#_Toc413849188)

[Requirements 3](#_Toc413849189)

[Overview 3](#_Toc413849190)

[Part 1: Phase 1 Requirements 3](#_Toc413849191)

[Part 2: Phase 2 requirements 4](#_Toc413849192)

[Part 3: Final submission 4](#_Toc413849193)

[Sample Output 4](#_Toc413849194)

# Assignment objectives

* Java Collections Framework
* Sorting, Comparable, Comparators

# Requirements

Create a project called A5FirstLastName.

## Overview

In this assignment, you will write a class called Palindrome.java which will accept a line from the user (console) and determine whether the line entered is a palindrome or not. You should complete the assignment using appropriate methods from both the Stack class and the LinkedList implementation of the Queue interface.

## Part 1: Phase 1 Requirements

1. Allow the user to enter strings to check if they are a palindrome or not. You should have a loop that will read a line, perform step 2 and wait for another user entry. The loop should quit only when the user enters a blank line. Write the menu in the main method.
2. For each string entered:
   1. Call a method named isPalindrome(), defined as follows:

public static boolean isPalindrome(String input)

which checks to see if the line is a palindrome (i.e. reads the same in reverse as well when you ignore case and non-letters/alphabets).

**Note that our implementation is non-standard for palindromes** (i.e. we only care about letters/alphabetic characters). You can find a method in the Character class to test for only letters.

You will use instances of Stack and Queue to perform the logic (**do not use built in methods for reversing**). Big hint: Consider using both Stacks and Queues together to perform the palindrome-checking logic.

The following describes the logic:

As you read each character that is a letter, place it on both a Queue and a Stack. Recall that the first is FIFO while the latter is LIFO. This allows you to then, after having read all characters, retrieve elements from the Stack and Queue and check for equality of each pair. If any pair of letters is not equal, we know the string cannot be a palindrome, at which point you can return a boolean with the value, false. If no mismatches occur, return a boolean with the value, true.

* 1. In the main method, obtain the return value of the method isPalindrome() and print the result to the user.
  2. If the user’s entry is a palindrome, do the following additional steps:
     1. Print the reverse of the original line (as entered by the user) by calling a method:

public static String reverseString (String s1)

Once again, use a Stack to perform this reversal (do not use built in methods for reversing). As you read each character from the passed line (parameter), place it in a Stack. Once done, retrieve each element, and create a new reversed string, which may be done using StringBuilder.

* + 1. Print a list of the original palindrome strings (as entered by the user) and their length , sorted by characters (string content). Recall that the String class already has the compareTo() method defined for you where sorting is done using content by default (no need to write an additional class).

## Part 2: Phase 2 requirements

If input is a palindrome, additionally print a list of the original palindrome strings and their length. This list should be reverse sorted by length first and (if equal length) then reverse sort by content (characters). Hint: You will need to write one additional class to do this.

## Part 3: Final submission

Turn in a single zip file for the project called A5FirstLastName.zip.

# Sample Output

Your expression (or return to end): Redivider

That is a palindrome.

The reversed string is: redivideR

String length

Redivider 9

*Phase 2 requirement output (note: same as original at this point):*

String length

Redivider 9

\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Your expression (or return to end): Re-divider

That is a palindrome.

The reversed string is: redivid-eR

String length

Re-divider 10

Redivider 9

*Phase 2 requirement output (note: same as original at this point):*

String length

Re-divider 10

Redivider 9

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Your expression (or return to end): Madam I'm Adam

That is a palindrome.

The reversed string is: madA m'I madaM

String length

Madam I'm Adam 14

Re-divider 10

Redivider 9

*Phase 2 requirement output (note: same as original at this point):*

String length

Madam I'm Adam 14

Re-divider 10

Redivider 9

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Your expression (or return to end): Racecar

That is a palindrome.

The reversed string is: racecaR

String length

Madam I'm Adam 14

Racecar 7

Re-divider 10

Redivider 9

*Phase 2 requirement output (note: different from original output):*

String length

Madam I'm Adam 14

Re-divider 10

Redivider 9

Racecar 7

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Your expression (or return to end): Race123car

That is a palindrome.

The reversed string is: rac321ecaR

String length

Madam I'm Adam 14

Race123car 10

Racecar 7

Re-divider 10

Redivider 9

*Phase 2 requirement output (note: different from original output):*

String length

Madam I'm Adam 14

Re-divider 10

Race123car 10

Redivider 9

Racecar 7

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Your expression (or return to end):

Bye!