XJTLU | Computing

CPT108: Data Structures and Algorithms

Semester 2, 2023-24

Laboratory 2: Recursion

Palindrome

A palindrome is a number, word or phrase (after removing all spaces and punctuations), or a sequence of symbols that reads the same backwards and forwards, such as "madam", "refer", "level", "bird rib", "Never odd or even", "Step on no pets!", 191, 91019, etc.

Your task: Recursive vs Iterative Palindrome Check

There are *two* tasks in this lab exercise.

Given a string:

1. Write a *recursive* function that checks if the given string is a palindrome; and
2. Write an *iterative* (i.e., non-recursive) function that checks if the given string is a palindrome.

There are two files in the downloaded package. You can implement your functions in `Palindrome.java` (inside the source folder), and run the test in `PalindromeTest.java` (inside the test folder).

You can filter out spaces and punctuations by examinating each character in a string one-by-one. However, you may also use other tools, such as *regular expression*[a,b] (in string), to do so.

[a] A comprehensive tutorial on how to use regular expression with `String` is available here: `https://ioflood.com/blog/java-replaceall-method-a-comprehensive-guide`

[b] Javadoc of `Pattern` class: `https://docs.oracle.com/en/java/javase/21/docs/api/java.base/java/util/regex/Pattern.html`

Before start implementing your code, you should start analysing the problem by identifying:

- Input: What is given?
- Output: What is required?
- Constraints: Under what conditions?
- Abstraction: What information are essential?

ALL must be presented in some form of data that can be processed by computer.

For the *recursive step*, you should think about:

- What is the stopping case?
- What is the recursion step?
- How to ensure that the recursion step will eventually terminate?
- Are there any variables / methods that need to be added?
- What needs to be initiated on each computation?
- What needs to be initiated on each recursive call?

Marking Scheme

Be prepared to explain to the teaching assistants (TAs) all the algorithm that you use, and what needs to be verified and done within the recursive function(s).

Your programming style (how clearly and how well you speak a programming language, such as Java, CPP, and Python) is what will be graded. Correct functioning of your program is *necessary* but ***not*** *sufficient*! Documentation is also a critically important part of your software engineering. Your use of Javadoc will also be graded.

Your grade consists of two parts: (i) *program correctness*, worth 60%, and (ii) *style & design*, worth 30%. Points are deducted for the error types as shown (but the minimum you can get for each part is 0%, so you cannot get negative point scores on either program correctness or style & design).

#	Description	Points	Notes
Program Correctness (Maximum: 60%, Minimum: 0%)			
1	Test results for recursive function	30	
2	Test results for iterative function	30	
☹	Program prints irrelevant information	-10	
Style & Design (Maximum: 40%, Minimum: 0%)			
3	Base point	40	
☹	Inconsistent naming of functions and variables	-10	
☹	Incorrect usage of constant variable	-10	
☹	Improper private data encapsulation	-10	
☹	Poor readability of function and variable names	-10	
☹	No comment has been written	-20	
☹	Inadequate function level documentation	-10	
☹	Inadequate algorithm level (i.e., within function) documentation	-15	
☹	Inadequate variable level documentation (i.e., meaning of the variable)	-10	
	Total	100	

You should *submit* your code to Learning Mall.

Final Remark

You must write your final version of the program on your own. If you worked in study groups, you must also acknowledge your collaborators in the write-up for each problem, whether or not they are classmates. Other cases will be dealt with as plagiarism. Re-read the policy on the course information sheet, and note the University's tougher policy regarding cheating.