UNIVERSITEIT VAN PRETORIA
UNIVERSITY OF PRETORIA
YUNIBESITHI YA PRETORIA

Department of Computer Science

COS212: Practical 2

Release: Thursday 23 February 2017
Deadline: Friday 24 February 2017, 18:00

Instructions

Complete the tasks below. Certain classes have been provided for you in the *files* subfolder of the practical download. You have been given a main file which will test some code functionality, but it is by no means intended to provide extensive test coverage. You are encouraged to edit this file and test your code more thoroughly. Each main file will have the expected output given in comments for you to compare to. Remember to test "corner" cases. Upload **only** the given source files, as well as Recursive.java, with your changes in a zip archive before the deadline. Please comment your name **and** student number in at the top of each file.

For this practical you will manipulate a singly-linked list with the use of recursive functions. We provide you with a basic Node class that contains a double value and a link to the next node. The value `null` is used to indicate the end of the list.

Since this practical is focused on recursion, you will be penalised for any loop structures in your code.

Task 1: Add scalar [5]

The first task is a simple case of working recursively through a linked list and adding an increasing scalar to each element. You need to implement a static method in the class `Recursive` called `addScalar`. This function must recursively call itself for each node that needs to be visited. For the first node, it must simply add the scalar provided by Main to the node's value. However, for the second node the scalar must be increased by 1. And then by another 1 for the third node. So each time it visits a node, the scalar is increased.

Task 2: Calculate sum [5]

For this task you must calculate the sum of all the values inside the linked list. You need to implement a static method in the class `Recursive` called `sum`. This function must recursively call itself for each node that it needs to visit, and add up a total during the process.

Task 3: Merge some values [10]

For this task you must recursively merge elements in a list. Particularly, any value that is followed by a larger value should be merged with that larger value. For example, the sequence $2, 1, 2, 3$ should result in $2, 3, 3$. The order in which you perform the removals does matter, so for the sequence $2, 3, 4$ the result should be $5, 4$. Once two elements have been merged, they are no longer considered for future mergers, so for the sequence $1, 2, 4$ the result should be $3, 4$.

Task 4: Merging with the end [10]

This task builds on task 3. Each value will still be compared to the next value. However, instead of merging a value with the next value, they should be merged last value in the list. So for the sequence $2, 1, 2, 3$ the result should be $2, 4, 2$.

Submission

Submit your source files on the CS Website. Place all the files in a zip archive named as uXXXXXXXX-task#.zip where XXXXXXXX is your student number and # is the task number. You have 24 hours to finish this practical, regardless of what practical session you attend. Upload your archive to the relevant task slot on the CS website. Submit your work before the deadline. No late submissions will be accepted.