UNIVERSITEIT VAN PRETORIA
UNIVERSITY OF PRETORIA
YUNIBESITHI YA PRETORIA

Department of Computer Science

COS212: Practical 9

Release: Monday 21 June 2021, 18:00
Deadline: Friday 25 June 2021, 18:00

PLAGIARISM POLICY

UNIVERSITY OF PRETORIA

The Department of Computer Science considers plagiarism as a serious offence. Disciplinary action will be taken against students who commit plagiarism. Plagiarism includes copying someone else's work without consent, copying a friend's work (even with consent) and copying material (such as text or program code) from the Internet. Copying will not be tolerated in this course. For a formal definition of plagiarism, the student is referred to `http://www.library.up.ac.za/plagiarism/index.htm` (from the main page of the University of Pretoria site, follow the *Library* quick link, and then choose the *Plagiarism* option under the *Services* menu). If you have any form of question regarding this, please ask one of the lecturers, to avoid any misunderstanding. Also note that the OOP principle of code re-use does not mean that you should copy and adapt code to suit your solution.

Objectives

The aim of this practical is to learn how to implement and use efficient sorting algorithms.

Instructions

Complete the tasks below. Certain classes have been provided for you alongside this specification in the *Student files* folder. You have also 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. Remember to test boundary cases.

Sorting [25]

Sorting functions are very useful utilities for many programming tasks. You have been given a partially implemented `Sort` class to use. Your task is to implement the following methods in the `Sort` class according to the given specification.

Task 1: Implement Heap Sort

You are required to implement the following heap sort function for generic data type arrays:

```
void heapsort(T[] data, boolean debug)
```

This heap sort function should be implemented iteratively as described on page 517 of the textbook. The array `data[]` contains a list of unsorted objects, that needs to be sorted in place. The parameter `debug` enables output for marking purposes. The `heapsort()` function should make use of the helper function `movedown()` to construct a heap out of all the elements of `data[]` except the last. The parameter `debug` from the `heapsort()` function needs to be passed on to this helper function.

Task 2: Implement Merge Sort

You are required to implement the following merge sort function for generic data type arrays:

```
void mergesort(T[] data, int first, int last, boolean debug)
```

This merge sort function should be implemented recursively as described on page 527 of the textbook. The array `data[]` contains a list of unsorted objects, that needs to be sorted in place. The parameters `first` and `last` indicate the beginning and end of the portion of the `data[]` array that needs to be sorted at a specific invocation. The parameter `debug` enables output for marking purposes. The `mergesort()` function should make use of the helper function merge() to put the two sorted halves back together. The parameter debug from the mergesort() function needs to be passed on to this helper function.

You may add additional helper functions and additional fields if you desire. You may not modify any of the given method signatures. Do not add any additional imports or classes.

Submission

You need to submit your source files on the Fitch Fork website (https://ff.cs.up.ac.za/). All methods need to be implemented (or at least stubbed) before submission. Place your **Sort.java** in a zip archive named uXXXXXXXX.zip where XXXXXXXX is your student number. There is no need to include any other files in your submission. You have 4 days to complete this practical, regardless of which practical session you attend. You have 5 submissions and your best mark will be your final mark. Upload your archive to the *Practical 9* slot on the Fitch Fork website. Submit your work before the deadline. **No late submissions will be accepted!**