Assignment 2
Algorithms and Data Structures 1 (1DL210)
2014

Jari Stenman Yunyun Zhu Tuan-Phong Ngo

September 26, 2014

1 Instructions

- Unless you have asked for and received special permission, solutions must be prepared in pairs. If you got special permission for the first assignment, you also have special permission for this assignment.
- Read the following section and do the assignment.
- Make sure that your programs run on the Unix system. Any (sub)solutions that don't run on the Unix system **will** receive 0 points. In particular, you should verify the output of your implementation against the output of the provided sorting algorithm BUBBLESORT.
- If you are not using the python skeleton code, make sure that your program behaves similarly. In this case, you probably need to implement yourself the BUBBLESORT algorithm.
- When you have completed the assignment, you should have five files: four sorting programs and one pdf document explaining the differences. Archive these files together with `rangen.py` by executing the following: `tar -cf firstname1_lastname1_firstname2_lastnamse2.tar file1 file2 file3 file4 file5 rangen.py`
- Submit the resulting `.tar` file to Studentportalen. Only **one** of you needs to do this.

The last day to hand in is **October 12**.

2 Sorting Algorithms

For this assignment, you are going to implement four different sorting algorithms based on their textbook descriptions. Each algorithm is to be implemented as its own program.

- The program should read a file called `nums.txt`, which can be assumed to contain integers separated by newlines, sort them using the algorithm in question, and then output a file `nums_sorted.txt` of the same form but with all numbers sorted.
- Additionally, your program should verify that your result matches the output of the BUBBLESORT algorithm.

Note that the arrays in the textbook are indexed from 1, whereas in most programming languages they are indexed from 0.

2.1 Skeleton Code

We provide some skeleton code in Python, which is the language we recommend for the assignments in this course. You are of course free to implement the algorithms in the language of your choice as long as the programs meet the requirements. The skeleton code consists of the following:

- `rangen.py` takes an integer argument n and outputs a file `nums.txt` containing n rows with random numbers in the range $\{0, \ldots, n\}$. Example of usage: `python rangen.py 100`
- `sort.py` contains an implementation of the BUBBLESORT algorithm along with the function `test` that verifies the output of your implementation matches the output of the provided sorting algorithm BUBBLESORT. Example of usage: `python sort.py`

2.2 Insertion Sort (2p)

Implement INSERTION-SORT from the second lecture.

2.3 Merge Sort (2p)

Implement the MERGE function from the third lecture.

2.4 Quicksort (2p)

Start by implementing PARTITION. Then use it to implement QUICKSORT from the fourth lecture.

2.5 Heapsort (3p)

Implement the functions MAX-HEAPIFY, BUILD-MAX-HEAP, and HEAPSORT from the fifth lecture.

2.6 Comparison (1p)

What are the differences between the four algorithms? For each algorithm, mention a situation where it has an advantage over the other three.