UNIVERSIDAD DE COSTA RICA

ECCI Escuela de Ciencias de la Computación e Informática

CI-0116 Analysis of Algorithms and Data Structures

Homework 1

1 Goal

The goal of the homework is to implement the sorting algorithms studied in the first third of the course and verify that theoretical differences in efficiency correspond to reality.

2 Algorithms

The algorithms to be implemented are the following: (i) selection sort [8 pts.], (ii) insertion sort [8 pts.], and (iii) merge sort [8 pts.] for **part I**, and (iv) heapsort [8 pts.], (v) quicksort [8 pts.], and (vi) radix sort with base $2^{\lfloor \lg n \rfloor}$digits [15 pts.] for **part II**.

3 Comparison

The steps to compare the algorithms are the following:

1. The algorithms must be compared using arrays of randomly chosen integers of sizes 50.000, 100.000, 150.000, and 200.000. Execute every algorithm at least three times on each of the arrays.[1] Report these times and their mean in a table [2.5 pts. each algorithm].

2. Plot mean execution times versus array size for each of the algorithms and analyze the resulting curve to identify if the shape is as expected (i.e., if the curves of the $\Theta(n^2)$ algorithms have a parabolic shape and the curves of the $\Theta(n)$ and $\Theta(n \log n)$ algorithms are almost straight).[2] (Do not forget to indicate the units used in the time axis, i.e., s, ms, etc.) [2.5 pts. each algorithm].

[1] If the largest execution time is at least 1.5 times the smallest, the operating system was probably doing something else while running your task. If this happens ignore that execution and run the algorithm again. Repeat as necessary until the largest execution time is no more than 1.5 times the smallest.

[2] Recall that $n \log n < n^{1+\epsilon}$ for $\epsilon > 0$ y n sufficiently large n.

3. Plot the curves again all together in a new set of axis. Show the time in a logarithmic scale to counteract the large execution time differences between the quadratic and (almost) linear algorithms. Analyze if the relation between the curves is as expected [2.5 pts. each algorithm].

4 Deliverables

The homework has two parts. The first one is a preliminary report that analyzes the selection sort, insertion sort, and merge sort algorithms. The second part analyzes the heapsort, quicksort, radix sort (with base $2^{\lfloor \lg n \rfloor}$digits), and the former three sorting algorithms. You must submit the file *Ordenador.h* published with this description with the implemented methods. (Implement them in that file, not in a *cpp* file.)

The code must be written in the C++ programming language, using the provided templates. The methods' headers must not be modified since the teaching assistants will use a script to run them, and any change in the methods interface will make the script to fail. (However, you can add customized private methods and call them in the body of the required methods.) To avoid compilation errors, the language must be used in its standard form. In particular, the code must compile if `g++` and the standard C++ libraries are used. *If the code does not compile you will receive a grade of zero. If a particular algorithm it does not correctly sort the arrays, you will receive a grade of zero for that algorithm.* To determine if a sorting algorithm is correct, we will run the following test:

```
1 for(int i=1; i<n; i++)
2     if( A[i] < A[i-1] )
3         cout << "Failed!";
```

5 Submission

This homework must be submitted before its due date through Mediacion Virtual. The student is responsible for checking the integrity of the submitted homework, specially if it is in a zipped file.[3] If you submitted multiple copies, only the last one will be considered. If the homework is submitted after the due date, the following police (specified in the syllabus) will be applied: *the grade received by a student who submitted a homework late will not be higher than the lower grade received by a student who submitted the homework on time.*

If you are having problems to submit the homework and the due date is approaching, send it to the assistant(s) with copy to the instructor (`arturo.camacho@ucr.ac.cr`). Please write the following in the subject: *Tarea de CI-0116.* In case we receive multiple submissions by email, we will consider only the first one.[4] The *late submission* policy mentioned in the previous paragraph applies to email submissions as well.

[3] Download the submitted file and unzip it to verify the integrity of its contents.

[4] The purpose is to discourage multiple email submissions.