Datenstrukturen und Algorithmen CSE - SS22

Exercise 2

Handout: 28.02.2022 08:00

Due: 10.03.2022 23:59

Improving Sorting

Open Task

Improving Sorting

Given is the sorting algorithm that you have seen in class (a variant of selection sort) as function `my_sort` in file `my_sorting.cpp`.

Implement different sorting algorithms `your_sort1`, `your_sort2` and `your_sort3` in file `your_sorting.cpp` such that it provides an increased efficiency, compared to `my_sort`. You can copy-paste from `my_sort` and start improving. When you run the program, running times of your algorithms are displayed and visually compared against the original version of this algorithm.

Four scenarios are compared: random, almost_sorted, ascending and descending.

Try to come up with improvements of the algorithm by yourself. This exercise is for you, to prepare for next week's lecture. We will accept all submissions that sorts correctly in reasonable time.

- Do it yourself!
- Do not look up any online sources (yet)!
- Do not use algorithms from the standard library!
- You do not necessarily have to implement three algorithms, but are encouraged to try several approaches

Input / Output

The input is the vector length n, optionally followed by a number m of vector lengths to compare, optionally followed by a number r of repetitions of the sorting algorithm.

If m is ommitted or set to 1, only n is used for the vector length and no graphical output is generated. Otherwise m vectors with multiples of length n/m are generated.

r determines the number of repetitions a vector is sorted. If omitted, $r = 1$.

Example input:

```
Enter n [m [r]]
100 1 10
```

Example output (not improved algorithm)

```
n = 100 descending
repetitions: 10
             time            swaps       comparisons
mine           67            49500           49500
yours          66            49500           49500
speedup   1.01515
n = 100 random
repetitions: 10
             time            swaps       comparisons
mine          190            21070           49500
yours         164            21070           49500
speedup   1.15854
n = 100 almost_sorted
repetitions: 10
             time            swaps       comparisons
mine           88             7100           49500
yours          87             7100           49500
speedup   1.01149
n = 100 ascending
repetitions: 10
             time            swaps       comparisons
mine           78                0           49500
yours         101                0           49500
speedup  0.772277
```