



Exercise 12

for **Introduction to Computer Science I** - WS 02/03

(January, 31 - February, 7 2003)

Task 12.1 Complexity

Can you think of an algorithm for each of the following complexity classes that has the respective running time?

- $O(n^3)$
- $O(\log n)$
- $O(n \log n)$

Don't "cheat" by using the respective function for the loop condition. For instance, the following solution is not allowed for $O(n^2)$:

```
for (i=1; i < n*n; i++ ) {  
    ...  
}
```

Task 12.2 Complexity Classes

Determine for the following functions the respective complexity class, i.e. the smallest one that contains the function, in the sense of the O-notation:

- $f_1(n) = \frac{n}{2}(\log n - 1 + 5n) \frac{n+1}{3}$
- $f_2(n) = 3 \frac{n+n^2}{5n} + \frac{5n}{3}$
- $f_3(n) = n^7 + 2001 \cdot n^6 + 7230 \cdot 10^{20}$

Task 12.3 Determination of Complexity

Examine the following algorithm, that sorts an array of elements of type `int` in an ascending order:

```
void sort(int[] table) {
    int i = 0;
    int n = table.length;

    while (i < n - 1) {
        if (table[i] > table[i + 1]) {
            // swap elements
            int tmp = table[i];
            table[i] = table[i + 1];
            table[i + 1] = tmp;
            // reset loop
            i = 0;
        } else
            i++;
    }
}
```

- a. How many steps does the algorithm need in the best case?
- b. Determine the number of steps for an array of length 2 and 3 for the worst case.
- c. Can you significantly increase the algorithm's efficiency by just applying a *small* change to the code?