

Tutorial 9

Exercise 1: Prove in detail that $f \in O(g)$ and $g \in \Theta(f)$ holds for the following functions:

$$f(n) = 5n^3 + 2n^2 - 9n + 13 \qquad g(n) = n^3$$

Deduce from this that also $g \in O(f)$, $g \in \Omega(f)$, $f \in \Theta(g)$, $g \in \Theta(f)$, $g \notin o(f)$, and $g \notin \omega(f)$.

Exercise 2: Order the following functions according to their asymptotic growth, i.e., order them into a sequence g_1, g_2, \dots, g_{15} , where $g_1 \in O(g_2)$, $g_2 \in O(g_3)$, \dots , $g_{14} \in O(g_{15})$. Describe also, for which pairs of functions g_i and g_{i+1} in this sequence $g_i \in \Theta(g_{i+1})$ a prokeré $g_i \in o(g_{i+1})$ holds.

n^2	2^n	n	$\log_2 n$	n^n
$n!$	$\log_2(n^2)$	$(\log_2 n)^2$	n^3	\sqrt{n}
2^{2^n}	10^n	n^{1000}	$\sqrt[3]{n}$	$n \log_2 n$

Exercise 3: Determine for the following triples of functions f_1, f_2, f_3 , which relations of the form $f_i \in O(f_j)$, $f_i \in \Omega(f_j)$, $f_i \in \Theta(f_j)$, $f_i \in o(f_j)$, and $f_i \in \omega(f_j)$ hold and which do not.

- a) $f_1(n) = 3n^2 + 5n - 1$, $f_2(n) = 2n^3 - 15n - 183$, $f_3(n) = (n + 1)(n - 1)$
- b) $f_1(n) = 4n^2 + n^2 \log_2 n$, $f_2(n) = \log_2^5 n$, $f_3(n) = 17n + 3$
- c) $f_1(n) = n \sqrt[5]{n}$, $f_2(n) = n$, $f_3(n) = \sqrt{n}$
- d) $f_1(n) = 2^n$, $f_2(n) = n^{1024}$, $f_3(n) = n!$
- e) $f_1(n) = 2^n$, $f_2(n) = n^n$, $f_3(n) = n!$
- f) $f_1(n) = 2^n$, $f_2(n) = n^n$, $f_3(n) = n^{\log_2 n}$
- g) $f_1(n) = 10^n$, $f_2(n) = 2^n$, $f_3(n) = 2^{2^n}$
- h) $f_1(n) = \log_{10}(n^2)$, $f_2(n) = \log_2 n$, $f_3(n) = \log_2(n^2)$
- i) $f_1(n) = n + \sqrt{n} \cdot \log_2 n$, $f_2(n) = n \cdot \log_2 n$, $f_3(n) = \sqrt{n} \cdot \log_2^2 n$
- j) $f_1(n) = 2^n$, $f_2(n) = 2^{\sqrt{n}}$, $f_3(n) = n!$
- k) $f_1(n) = n/2048$, $f_2(n) = \sqrt{n} \cdot 3n$, $f_3(n) = n + n \cdot \log_2 n$
- l) $f_1(n) = (\log_2 n)^n$, $f_2(n) = n^n$, $f_3(n) = 10^{\sqrt{n}}$

Exercise 4: Determine as precisely as possible the time and space complexity of Algorithm 1. You can assume that value n represents the number of elements in array A , and that this array is indexed from zero.

Exercise 5: Determine as precisely as possible the time and space complexity of Algorithm 2 (recall this algorithm from the previous tutorial).

(You can assume that value n represents the number of elements in array A , that this array is indexed from zero, and that x is a value of searched element.)

Algorithm 1: Selection sort

```
SELECTION-SORT (A, n):  
  i := n - 1  
  while i > 0 do  
    k := 0  
    for j := 1 to i do  
      if A[k] < A[j] then  
        k := j  
    x := A[k]; A[k] := A[i]; A[i] := x  
    i := i - 1
```

Algorithm 2: Binary search

```
BSEARCH (x, A, n):  
  l := 0  
  r := n  
  while l < r do  
    k :=  $\lfloor (\ell + r) / 2 \rfloor$   
    if A[k] < x then  
      l := k + 1  
    else  
      r := k  
  if l < n and A[l] = x then  
    return l  
  return NOTFOUND
```

Exercise 6: By pseudocode describe an arbitrary algorithm for solving the following problem, and estimate its time and space complexity as accurately as possible. (What is an appropriate measure of the size of an input in this problem?)

INPUT: Matrices A, B with integer elements.

OUTPUT: Matrix $A \cdot B$.

Remark: You don't have to deal with input and output in your algorithm.

Do not assume that matrices A and B must be square matrices. However, you can assume that sizes of both matrices are such that it is possible to multiply the matrices, i.e., that matrix A is of size $m \times n$ a matrix B is of size $n \times p$, where m , n , and p are some natural numbers.

Exercise 7: Design an algorithm solving the following problem:

INPUT: A number n and a sequence of numbers a_1, a_2, \dots, a_n , where for each $i = 1, 2, \dots, n$ is $a_i \in \{1, 2, \dots, n\}$.

QUESTION: Does the sequence a_1, a_2, \dots, a_n contain every $x \in \{1, 2, \dots, n\}$ exactly once?

Analyze the time complexity of your algorithm. If it is greater than $O(n)$, try to design an algorithm with the time complexity $O(n)$.