FTP_Alg_Week 1: Exercises

jungkyu.canci@hslu.ch

Exercise with the asterisk * are considered non easy ones.

Exercise 1 *A sorting method with "Big-Oh" complexity $O(n \log n)$ spends exactly 1 millisecond to sort $1'000$ data items. Assuming that time $T(n)$ of sorting n items is directly proportional to $n \log n$, that is, $T(n) = c \cdot n \log n$, derive a formula for $T(n)$, given the time $T(N)$ for sorting N items, and estimate how long this method will sort $1'000'000$ items.*

Exercise 2 *A quadratic algorithm with processing time $T(n) = cn^2$ spends $T(N)$ seconds for processing N data items. How much time will be spent for processing $n = 5'000$ data items, assuming that $N = 100$ and $T(N) = 1\text{ms}$?*

Exercise 3 *An algorithm with time complexity $O(f(n))$ and processing time $T(n) = c \cdot f(n)$, where $f(n)$ is a known function of n, spends 10 seconds to process 1'000 data items. How much time will be spent to process 100'000 data items if $f(n) = n$ and $f(n) = n^3$?*

Exercise 4 *Assume that each of the expressions below gives the processing time $T(n)$ spent by an algorithm for solving a problem of size n. Select the dominant term(s) having the steepest increase in n and specify the lowest Big-Oh complexity of each algorithm.*

Expression	Dominant term(s)	$O(\ldots)$
$5 + 0.001n^3 + 0.025n$		
$500n + 100n^{1.5} + 50n \log_{10} n$		
$0.3n + 5n^{1.5} + 2.5 \cdot n^{1.75}$		
$n^2 \log_2 n + n(\log_2 n)^2$		
$n \log_3 n + n \log_2 n$		
$3 \log_8 n + \log_2 \log_2 \log_2 n$		
$100n + 0.01n^2$		
$0.01n + 100n^2$		
$2n + n^{0.5} + 0.5n^{1.25}$		
$0.01n \log_2 n + n(\log_2 n)^2$		
$100n \log_3 n + n^3 + 100n$		
$0.003 \log_4 n + \log_2 \log_2 n$		

Exercise 5 *The statements below show some features of "Big-Oh" notation for the functions $f \equiv f(n)$ and $g \equiv g(n)$. Determine whether each statement is TRUE or FALSE and correct the formula in the latter case.*

Statement	Is it TRUE or FALSE?	If it is FALSE then write the correct formula
Rule of sums: $O(f+g) = O(f) + O(g)$		
Rule of products: $O(f \cdot g) = O(f) \cdot O(g)$		
Transitivity: if $g = O(f)$ and $h = O(f)$ then $g = O(h)$		
$5n + 8n^2 + 100n^3 = O(n^4)$		
$5n + 8n^2 + 100n^3 = O(n^2 \log n)$		

Exercise 6 *Work out the computational complexity of the following piece of code:*

```
Algorithm
  for i = n to 1 do
    for j = 1 to n do
      for k = 0 to n do
        ... // constant number of operations
        k = k + 2
      end for
      j = j * 2
    end for
    i = i / 2
  end for
```

Exercise 7 (*) *Work out the computational complexity of the following piece of code:*

```
  sum=0
  for i = 1 to n do
    for j = n to 1 do
      for k = j to n do
        sum=sum + (i+j*k)
        k = k + 2
      end for
      j = j / 2
    end for
    i = i * 2
  end for
```

Exercise 8 (*) *Running time $T(n)$ of processing n data items with a given algorithm is described by the recurrence: $T(n) = k \cdot T\left(\frac{n}{k}\right) + c \cdot n; \quad T(1) = 0.$*

Derive a closed form formula for $T(n)$ in terms of c, n, and k. What is the computational complexity of this algorithm in a "Big-Oh" sense? [Hint: To have the well-defined recurrence, assume that $n = k^m$ with the integer $m = \log_k n$ and k.]