Weekly Assignment 1:
Big-O

September 2025

Some facts (no exercises)

Some tips and tricks (most of these are not needed for this particular exercise sheet, but might come in handy at some point during the course). Summations:

$$1 + 2 + \cdots + n = \sum_{i=0}^{n} i = \frac{n(n+1)}{2} \qquad 1 + r + \cdots + r^n = \sum_{i=0}^{n} r^i = \frac{r^{n+1} - 1}{r - 1}$$

If these are not familiar to you, try to prove them yourself (for example with induction), practicing such proofs is always useful ;-). About logarithms and exponential:

$$2^0 = 1 \qquad 2^1 = 2 \qquad 2^a 2^b = 2^{a+b} \qquad 2^{ab} = (2^a)^b = (2^b)^a$$

$$\log_2(1) = 0 \qquad \log_2(2) = 1 \qquad \log_2(ab) = \log_2(a) + \log_2(b)$$

If the logarithm rules are unfamiliar to you, try proving them from the exponentiation rules and the identity:

$$\log_2(2^x) = x \qquad \text{for all } x \in \mathbb{R}$$

1. Suppose we have a computer which can perform 1 million ($= 10^6$) operations per second. The seven formulas below denote the running time of some algorithms (measured in number of operations) depending on the number of elements n we feed to the algorithm. Determine for each algorithm how many elements can be processed in 1 minute.
   (a) $n \log n$
   (b) $n\sqrt{n}$
   (c) 4^n
   (d) $n^3\sqrt{n}$
   (e) n^{24}
   (f) $n!$
   (g) $n \;\ln\; n$

2. Given a function f. Is it always true that $f(n) \in \mathcal{O}(f(n+1))$? If so, give a proof. If not, give a counterexample.

3. Recall that in order to prove $f \in \mathcal{O}(g)$ one has to choose $c \in \mathbb{R}$ with $c > 0$ and $n_0 \in \mathbb{N}$ and then prove that $f(n) \leq cg(n)$ for all $n \geq n_0$.
   (a) Prove $2n \in \mathcal{O}(n)$.
   (b) Prove that $3n + 5 \in \mathcal{O}(n)$.

4. Consider the following algorithm.

Algorithm 1 Search

```
function SEARCH(int v[], int n)
    int i = 0
    bool foo = false
    while i < n ∧ ¬foo do
        if i ≥ 5 ∧ v[i] mod 2 == 0 then
            foo = true
        end if
        i = i + 1
    end while
    return i
end function
```

(a) What is the worst case scenario? How many operations does it take in this case (your answer should use $\mathcal{O}$ notation and depend on n)?

(b) What is the best case scenario? How many operations does it take in this case?

5. In mathematics, the factorial of a non-negative integer n, denoted by $n!$, is the product of all positive integers less than or equal to n such as

$$\begin{cases} n! = n \times (n-1)! \\ 0! = 1 \end{cases}$$

Write a recursive function returning the factorial of n and comment on its asymptotic time complexity.