Exercise Session 02

Solve the following exercises. The exercises that are more involved are marked with a star.

Exercise 1.

CLRS-3 3.1–1 Let $f(n)$ and $g(n)$ be asymptotically nonnegative functions. Using the basic definition of Θ-notation, prove that $\max(f(n), g(n)) = \Theta(f(n) + g(n))$.

Answer:
We need to show that there exist witnesses c_1 and c_2 such that:

$c_1 \cdot (f(n) + g(n)) \leq max(f(n), g(n)) \leq c_2 \cdot (f(n) + g(n))$

By choosing $c_1 < 1$ and $c_2 > 1$ we will ensure that our witnesses will be a tight upper an lower bound for our functions such that: $\Theta(f(n) + g(n))$

CLRS-3 3.1–4 Is $2^{n+1} = \Theta(2^n)$? Is $2^{2n} = O(2^n)$?

a) We ignore smaller terms and can therefor conclude that the first statement is correct
b)

$$\begin{aligned} 2^{2n} &\leq 2^n \\ 2^n \cdot 2^n &\leq 2^n \\ (2 \cdot 2)^n &\leq 2^n \\ 4^n &\leq 2^n \end{aligned}$$

This is clearly a contradiction if $n > 0$, and $2^{2n} = O(2^n)$ is therefor not true. The correct upper bound is instead $O(4^n)$

Exercise 2.

Consider the algorithm SumUpTo that takes as input a natural number $n \in \mathbb{N}$.

```
SumUpTo(n)
  s = 0
  for i = 1 to n
        s = s + i
  return s
```

Use the technique of loop invariants to prove that, given $n \in \mathbb{N}$, SumUpTo terminates and returns $\frac{n(n+1)}{2}$.

Initialisation Before the the first iteration of the loop $s = 0$ and $i = 1$ we want to show that the base case holds true where we substitute n with $i - 1$:

$$\begin{aligned} s &= \frac{n(n+1)}{2} \\ &= \frac{(i-1)(i-1+1)}{2} \\ &= \frac{(i-1)i}{2} \end{aligned}$$

Therefor the initial loop invariant holds true for $s = \frac{(i-1)i}{2}$. Showing this holds true for $s = 0$ and $i = 1$

$$s = \frac{(1-1) \cdot 1}{2} = 0$$

The base case therefore holds true

Maintenance

$$
\begin{aligned}
s + i &= \frac{(i-1)i}{2} + i \\
&= \frac{i^2 - i}{2} + i \\
&= \frac{i^2 - i + 2i}{2} \\
&= \frac{i^2 + i}{2} \\
&= \frac{i(i+1)}{2}
\end{aligned}
$$

Termination

The for loop terminates when the condition $i > n$ is false. In other words when $i = n + 1$

Exercise 3.

By getting rid of the asymptotically insignificant parts on the expressions, give a simplified asymptotic tight bounds (big-theta notation) for the following functions in n. Here, $k \geq 1$, $e > 0$ and $c > 1$ are constants.

(a) $0.001n^2 + 70000n \implies \Theta(n^2)$

(b) $2^n + n^{10000} \implies \Theta(2^n)$

(c) $n^k + c^n \implies \Theta(c^n)$

(d) $\log^k n + n^e \implies \Theta(n^e)$

(e) $2^n + 2^{n/2} \implies \Theta(2^n)$

(f) $n^{\log c} + c^{\log n}$ (hint: look at some properties of the logarithm at CLRS-3 p. 56 or CLRS-4 p. 66)

★ **Exercise 4.**

Consider the following algorithm that takes an array $A[1\,..\,n]$ and rearrange its elements in nondecreasing order.

```
SORT(A)
for i = 1 to A.length
    for j = i + 1 to A.length
        if A[i] > A[j]
            key = A[i]
            A[i] = A[j]
            A[j] = key
```

(a) Try SORT(A) on the the instance $A = [4, 2, 8, 7, 1]$. Explain in your words how the algorithms works in general;

(b) Prove that SORT solves the sorting problem (hint: determine suitable invariants for both loops);

(c) Determine the asymptotic worst-case running time using the Θ notation.

★ **Exercise 5.**

Let $p(n) = \sum_{i=0}^{d} a_i n^i$, where $a_d > 0$, be a degree-d polynomial in n, and let k be a constant. Use the definitions of the asymptotic notations to prove the following properties

(a) if $k \geq d$, then $p(n) = O(n^k)$;

(b) if $k \leq d$, then $p(n) = \Omega(n^k)$;

(c) if $k = d$, then $p(n) = \Theta(n^k)$;