Reminder: \( \log n \) denotes the binary logarithm, i.e., \( \log n = \log_2 n \).

1 **O-notation: Part I**

Give formal proofs of the following statements using the definition of Big-O from the lecture.

1. \( 10 \in O(1) \).

   **Solution.** We need to show that there are positive constants \( c, n_0 \) such that \( 10 \leq c \cdot 1 \), for every \( n \geq n_0 \). Observe that this expression does not depend on \( n \) at all. Therefore any positive value for \( n_0 \) would work, e.g., \( n_0 = 1 \) (or \( n_0 = 23 \) or any other value). We chose \( c = 10 \) which implies that \( 10 \leq c \cdot 1 \) is satisfied. This proves that \( 10 \in O(1) \). ✓

2. \( 5n \in O(n) \).

   **Solution.** Again, we need to find positive constants \( c, n_0 \) such that \( 5n \leq c \cdot n \) holds, for every \( n \geq n_0 \). We chose \( c = 5 \) and obtain \( 5n \leq 5n \). This is always true and we can therefore pick any positive \( n_0 \), e.g., \( n_0 = 1 \). ✓

3. \( n^2 + 10n \in O(\frac{1}{10}n^3) \).

   **Solution.** We need to find positive constants \( c, n_0 \) such that \( n^2 + 10n \leq c \cdot \frac{1}{10}n^3 \) holds for every \( n \geq n_0 \). We first pick \( n_0 = 1 \). Then, under the assumption that \( n \geq n_0 = 1 \), we have \( n^2 + 10n \leq n^3 + 10n^3 = 11n^3 \). If we satisfy the inequality \( 11n^3 \leq c \cdot \frac{1}{10}n^3 \), which is equivalent to \( 11 \leq c/10 \), then we also satisfy the initial inequality \( n^2 + 10n \leq c \cdot \frac{1}{10}n^3 \). This is easy to do: We simply select \( c = 110 \).

   We have shown that for \( c = 110 \) and \( n_0 = 1 \) the inequality \( n^2 + 10n \leq c \cdot \frac{1}{10}n^3 \) holds for every \( n \geq n_0 \), which implies the result. ✓

4. \( \sum_{i=1}^{n} i \in O(4n^2) \).

   **Solution.** First, observe that \( \sum_{i=1}^{n} i = n(n+1)/2 = \frac{n^2}{2} + \frac{n}{2} \). We need to find positive constants \( c, n_0 \) such that \( \frac{n^2}{2} + \frac{n}{2} \leq c \cdot 4n^2 \), for every \( n \geq n_0 \). We pick \( n_0 = 1 \). Since \( n \leq n^2 \), for every \( n \geq n_0 = 1 \), we will satisfy the inequality \( \frac{n^2}{2} + \frac{n}{2} \leq c \cdot 4n^2 \), which is equivalent to \( 1 \leq 4c \). We can hence pick \( c = 1 \) and we are done. ✓
2 Racetrack Principle

1. Use the racetrack principle to prove the following statement:

\[ n \leq e^n \] holds for every \( n \geq 1 \).

**Solution.** First, we verify that \( n \leq e^n \) holds for \( n = n_0 = 1 \). This is true, since \( 1 \leq e \) holds. Next, we verify that \((n)'^2 \leq (e^n)'^2\) holds for every \( n \geq n_0 \). We have \((n)' = 1\) and \((e^n)' = e^n\). We thus need to show that \( 1 \leq e^n \) holds for every \( n \geq 1 \). Taking the natural logarithm on both sides, we obtain \( 0 \leq n \), which is true for every \( n \geq n_0 = 1 \). Hence, \( n \leq e^n \) holds for every \( n \geq 1 \). \( \checkmark \)

2. Use the racetrack principle and determine a value \( n_0 \) such that

\[ \frac{2}{\log n} \leq \frac{1}{\log \log n} \] holds for every \( n \geq n_0 \). (Difficult!)

**Hint:** Transform the inequality and eliminate the log-function from one side of the inequality before applying the racetrack principle.

Recall that \((\log n)' = \frac{1}{n \ln(2)}\). The inequality \( \ln(2) \geq \frac{1}{2} \) may also be useful.

**Solution.** We use the provided “Hint” and transform the given inequality as follows:

\[
\begin{align*}
\frac{2}{\log n} &\leq \frac{1}{\log \log n} \\
2 \log \log n &\leq \log n \\
2^{\log \log n} &\leq 2 \log n \\
(\log n)^2 &\leq n.
\end{align*}
\]

We now pick \( n_0 = 16 \). Then, \((\log 16)^2 \leq 16\) holds. Next, observe that \((\log n)^2 = \frac{2 \ln(n)}{(\ln(2))^2 n}\) and \((n)' = 1\). Using the racetrack principle, it is enough to show that \( \frac{2 \ln(n)}{(\ln(2))^2 n} \leq 1 \), for every \( n \geq n_0 = 16 \). This is equivalent to showing that \( 2 \ln(n) \leq \ln(2)^2 n \) (for every \( n \geq 16 \)).

We now apply the racetrack principle again: To this end, we first verify that \( 2 \ln(n) \leq \ln(2)^2 n \) holds for \( n = n_0 = 16 \): We indeed have \( 2 \ln(16) = 2 \ln(2^4) = 8 \ln(2) \leq \ln(2)^2 \cdot 16 \) (which holds since \( \ln(2) \geq 1/2 \)). Next, observe that \((2 \ln(n))' = \frac{2}{n} \) and \((\ln(2)^2 n)' = \ln(2)^2\).

It thus remains to argue that \( \frac{2}{n} \leq \ln(2)^2 \) for every \( n \geq 16 \). The previous inequality is equivalent to \( \frac{2}{\ln(2)^2} \leq \frac{2}{(\ln(2))^2} \leq 8 \leq n \), which holds for every \( n \geq 16 \).

Hence, \( \frac{2}{\log n} \leq \frac{1}{\log \log n} \) holds for every \( n \geq 16 \). \( \checkmark \)

3 O-notation: Part II

Give formal proofs of the following statements using the definition of Big-O from the lecture.

1. \( f \in O(h_1), g \in O(h_2) \) then \( f + g \in O(h_1 + h_2) \).
Solution. Since $f \in O(h_1)$, we know that there are constants $c_1, n_1$ such that $f(n) \leq c_1 \cdot h_1(n)$, for every $n \geq n_1$. Similarly, since $g \in O(h_2)$, we know that there are constants $c_2, n_2$ such that $g(n) \leq c_2 \cdot h_2(n)$, for every $n \geq n_2$. We now need to show that there are constants $C, N$ such that $f(n) + g(n) \leq C \cdot (h_1(n) + h_2(n))$ holds for every $n \geq N$. Let $N = \max\{n_1, n_2\}$ and let $C = \max\{c_1, c_2\}$. Then, for every $n \geq N$ we have:

$$f(n) + g(n) \leq c_1 h_1(n) + c_2 h_2(n) \leq C(h_1(n) + h_2(n)),$$

which proves the result.

2. $f \in O(h_1), g \in O(h_2)$ then $f \cdot g \in O(h_1 \cdot h_2)$.

Solution. Similar as in the previous exercise, we know that there are constants $c_1, c_2, n_1, n_2$ such that $f(n) \leq c_1 \cdot h_1(n)$, for every $n \geq n_1$, and $g(n) \leq c_2 \cdot h_2(n)$, for every $n \geq n_2$. Then:

$$f(n) \cdot g(n) \leq c_1 \cdot h_1(n) \cdot c_2 \cdot h_2(n) = c_1 c_2 \cdot h_1(n) h_2(n)$$

for every $n \geq \max\{n_1, n_2\}$. We thus select $C = c_1 \cdot c_2$ and $N = \max\{n_1, n_2\}$ and obtain $f(n)g(n) \leq C(h_1(n)h_2(n))$, for every $n \geq N$.

3. $2^n \in O(n!)$

Solution. To prove this statement, we will show that $2^n \leq C \cdot n!$ holds for $C = 2$ and every $n \geq 2$. To this end, observe that $2^n \leq 2n!$ is equivalent to $2^{n-1} \leq n!$. Observe that

$$2^{n-1} = \underbrace{2 \cdot 2 \cdot \cdots \cdot 2}_{(n-1) \text{ times}},$$

and

$$n! = \underbrace{2 \cdot 3 \cdot \cdots \cdot n}_{(n-1) \text{ factors, each larger equal to } 2}.$$

Trading off the factors of the two expressions, we see that $2^{n-1} \leq n!$, which proves the result.

4 Fast Peak Finding

Consider the following variant of FAST-Peak-Finding where the “$\geq$” sign in the condition in instruction 4 is replaced by a “$<$” sign:

1. if $A$ is of length 1 then return 0
2. if $A$ is of length 2 then compare $A[0]$ and $A[1]$ and return position of larger element
3. if $A[[n/2]]$ is a peak then return $[n/2]$
4. Otherwise, if $A[[n/2]-1] < A[[n/2]]$ then return FAST-Peak-Finding($A[0, [n/2] - 1]$)
5. else return $[n/2] + 1 +$ FAST-Peak-Finding($A[[n/2] + 1, n - 1]$)

1. Give an example instance of length 8 on which this algorithm fails.
Solution. Consider the instance $A[i] = i$, for every $0 \leq i \leq 7$. Then the algorithm recurses on the subarray $A[0\ldots2]$ in line 4. Observe however that none of the elements in $A[0\ldots2]$ constitute a peak in array $A$. ✓

2. Consider now the correct version of Fast-Peak-Finding given in the lecture. Suppose that we replaced the $\lfloor n/2 \rfloor$ by $\lfloor n/10 \rfloor$ throughout the algorithm. Would the algorithm still work? Would it be more or less efficient?

Solution. The algorithm would still work, however, it would be slightly less efficient. The runtime would still be $O(\log n)$, however, the hidden constant in the big-$O$ notation is larger. ✓