Algorithms and Data Structures 2023/24
Week 4 tutorial sheet

Below are a list of *suggested* exercises, some from the back of either lecture 1, 2 or 3). Ask your tutor to cover the questions most important to you. You should see the tutorial as a resource to get answers to any questions, so don't feel compelled to stick to the sheet.

1. Consider the recursive algorithm REC-MAX (below) for finding the maximum element in an array of integers.

   Give a recurrence for the worst-case running time of REC-MAX, justify its constants with reference to the algorithm, and solve the recurrence using the Master Theorem.

   Algorithm REC-MAX(A, i, j)

   ```
   if i < j then
       m ← ⌊(i+j)/2⌋
       ℓ ← REC-MAX(A, i, m)
       r ← REC-MAX(A, m+1, j)
       if ℓ ≥ r then
           return ℓ
       else
           return r
   else
       return A[i]
   ```

2. Consider the following recurrence:

$$T(n) = \begin{cases} 1 & \text{if } n = 1, \\ 4T(\lfloor n/2 \rfloor) + n^2 & \text{if } n > 1. \end{cases}$$

   Prove $T(n) \in \Omega(n^2 \cdot \lg(n))$ by *first principles* (not using Master theorem) in 3 steps:

   (a) Prove by induction $T(\widehat{n}) = \widehat{n}^2(1 + \lg(\widehat{n}))$ if $\widehat{n} = 2^p$ for $p \in \mathbb{N}^0$ (power-of-2 case).

   (b) Prove by induction that $T(j) \leq T(k)$ for all $j < k$.
   Similar to "Step 2"for MERGESORT in Lecture slides 2,3.

   (c) For the lower bound, consider the largest power-of-2 2^q satisfying $2^q \leq n$. Show that we have $2^q > n/2$. Hence use (b) and (a) to show $T(n) \in \Omega(n^2 \cdot \lg(n))$.
   This is similar to "Step 3" for MERGESORT *in Lecture slides 2,3*. However, because we want to prove Ω, we choose the closest power-of-2 lower than n.

3. Use Strassen's algorithm to compute the matrix product

$$\begin{pmatrix} 1 & 3 \\ 5 & 7 \end{pmatrix} \begin{pmatrix} 8 & 4 \\ 6 & 2 \end{pmatrix}.$$

   This is Exercise 28.2-1 of [CLRS].

4. Describe an algorithm for efficiently multiplying a $(p \times q)$ matrix with a $(q \times r)$ matrix, where p, q, r are arbitrary positive integers. The running time should be $\Theta(n^{\lg(7)})$, where $n = \max\{p, q, r\}$.