Program Design & Data Structures (Course 1DL201)
Uppsala University Autumn 2020/Spring 2021
Homework Assignment 2: Analysis of Algorithms

Prepared by Johannes Borgström

Lab:	Thursday 3 December
Submission Deadline:	**18:00 on Thursday 7 January**

Goal

The goal of this assignment is to promote and assess your understanding of the key concepts of algorithm analysis: growth of functions, converting code to recurrences, and solving recurrences (i.e., finding closed forms).

Please submit your answers to the following questions in a report in PDF format. We strongly prefer computer written solutions over handwritten ones. You can generate nice maths using LaTeX.

Do not just provide equations as answers. *Explain* how you obtained your answers.

1 Real World Algorithm

Consider the following algorithm, which is executed by people. The objective of the algorithm is to sort a group of people by age. The people all stand in a line side-by-side, facing the same direction, in numbered positions with position 1 having nobody on their left. The algorithm consists of iterating the following steps:

Step (i) People in even-numbered positions check with the person on their right to see who is younger. If the person on the right is younger, they swap.

Step (ii) People in odd-numbered positions check with the person on their right to see who is younger. If the person on the right is younger, they swap.

Step (iii) If nobody switched places in Step (i) or (ii), the algorithm stops.

1. How many iterations of the algorithm are required to guarantee that n people will be sorted in the worst case?

2. Discuss briefly how this human computation model compares to the computations performed by a (single core) computer.

Answers may be presented as the precise number of steps or using big-O notation. The most important thing is to show your reasoning.

2 Growth of Functions

Recall the definition of $\Theta(g(n))$: for non-negative functions f and g,

> $f(n) = \Theta(g(n))$ if and only if there exists $n_0 \geq 0$ and c_1, $c_2 > 0$ such that for all $n > n_0$ the following holds: $c_1 \cdot g(n) \leq f(n) \leq c_2 \cdot g(n)$.

Find the precise bound—a simple function $g(n)$ such that $f(n) = \Theta(g(n))$—for the function

$$f(n) = 14 + 23n + 11n^2 \cos(n) + 27n^3.$$

Prove that your answer is correct by finding some n_0, c_1, and c_2 satisfying the definition above.

3 Recurrences

Compute the first 10 values of the following recurrence (starting from $n = 0$):

$$\begin{aligned} f(0) &= 1 \\ f(n) &= 2f(n-1) + 3n + 7, \qquad n > 0 \end{aligned}$$

As usual, show your reasoning.

4 From Code to Recurrence

For the following programs, give a recurrence indicating the run-time based on the input. Indicate whether your recurrence is defined in terms of the size of the input, the value of the input, or something else. You do **not** need to give the closed form of the recurrence.

1. The following function appends one list to the end of another. What is the recurrence for $T_{\texttt{append}}$ that describes the run-time cost of `append xs ys`?

```
append :: [a] -> [a] -> [a]
append []     ys = ys
append (x:xs) ys = x : append xs ys
```

2. Let `p` be an arbitrary (but fixed) integer in the following function. What is the recurrence for $T_{\texttt{split}}(n)$ that describes the run-time cost of `split p xs`?

```
split :: Int -> [Int] -> ([Int], [Int])
split _ [] = ([], [])
split p (x:xs) =
  let
    (lows, highs) = split p xs
  in
   if x < p
     then (x:lows, highs)
     else (lows, x:highs)
```

5 Solving Recurrences

For the following questions you do **not** need to prove your answer (e.g., using induction). You just need to apply the specified method to produce a reasonable guess for the closed form.

1. Use the substitution method to obtain a closed form for the following recurrence:

$$\begin{aligned} f(0) &= 15 \\ f(n) &= f(n-1) + n + 4, \qquad n > 0 \end{aligned}$$

   Hint: $1 + 2 + 3 + \cdots + n = \frac{n(n+1)}{2}$ — you do **not** need to prove this fact.

2. Use the expansion method to obtain a closed form for the following recurrence:

$$\begin{aligned} g(0) &= 2 \\ g(n) &= 2g(n-1) + 5, \qquad n > 0 \end{aligned}$$

   Hint: keep track of 2s and 5s.

Grading

Your answers are graded on a U/3–5 scale for correctness of results and explicitness of reasoning. Each of the five questions will be graded separately.

If your answer for *any* of the questions shows little or no evidence of engagement in the question, you get a U grade for the *entire* homework assignment. Thus, you need to get a grade of 3 (or better) on *all* questions to pass the assignment.

If your answer shows some relevant engagement in the question, e.g., choice of a partially correct strategy or some partially correct reasoning with trial and error, you get (at least) a 3 for your answer.

If your answer is mostly correct and your reasoning is comprehensible, with very few major omissions or errors, you get a 4 for your answer.

If your answer is correct and your reasoning is explicit, with at most minor omissions or oversights, you get a 5 for your answer.

Final Grade

You pass the assignment if (and only if) your grade is at least 3 on *all* questions. In this case, your final score for the assignment is the sum of the five per-question grades (on a scale of 15–25).

Resubmission

For students who did not pass the assignment, there will be resubmission opportunities in Period 3 and also in August.

Modalities

- The assignment will be conducted in groups of 2 (or possibly 3) students. Groups can be selected in Studium. *If you cannot find your partner until Wednesday 2 December, please contact Johannes to assign you a new partner—if possible.*
- Answers must be submitted via Studium. Only one report per group shall be submitted. Ensure that **both** group members' names appear on the report.

By submitting a solution you are certifying that it is solely the work of your group, except where explicitly attributed otherwise.

Good luck!