Data Structures and Algorithms **Spring-Summer 2013/14**

Assignment 2

Mouna Kacimi, Werner Nutt, Camilo Thorne

Asymptotic Complexity and Substring Matching

Instructions: Your assignment should represent your own effort. However, you are not expected to work alone. It is fine to discuss the exercises and try to find solutions together, but each student shall write down and submit his/her solutions separately. If you worked together with other students, please list their names.
You can write up your answers by hand (provided your handwriting is legible) or use a word processing system like Latex or Word.
For a programming task, your solution must contain (i) an explanation of your solution to the problem, (ii) the Java code, in a form that we can run it, (iii) instructions how to run it. Also put the source code into your solution document. For all programming tasks, it is not allowed to use any external libraries ("import") if not stated otherwise.
Please, include name and email address in your submission.

1. Comparison According to Asymptotic Complexity

Order the following functions according to their asymptotic complexity, from the function having the smaller asymptotic complexity to the function having the larger one (i.e., such that $f_1 = O(f_2)$; $f_2 = O(f_3)$; ...):

- $50 \cdot \log_2 n$
- $5 \cdot n + n^2 + 1$
- $\log_{10}^2 n$
- $2^n + 5$

- $3^{\sqrt{n}}$
- $5^3 \cdot n$
- $3 \cdot \log_{10} n$
- $(n+1)!$
- $4^{\log_2 n}$
- $\sqrt{n}$.

(10 Points)

2. Asymptotic Equalities

Prove or disprove the following statements:

a) $2n^n + 2^{n+1} = \Theta(2n^n + 2^n)$

b) $(n+a)^b = \Omega(n^b)$ for all $a, b > 0$

c) $8n + n \cdot \log_2 n = O(n)$

(5 Points)

3. Asymptotic Puzzles

For each one of the following statements, write two functions f and g, where $f(n), g(n) \geq 0$ for all $n \geq 0$, that satisfy the given condition.

a) $f(n) = O(g^2(n))$

b) $f(n) = \Omega(f^2(n) + g(n))$

c) $f(n) = \Theta(g(n^2))$

(5 Points)

4. Substring Matching

Write a substring matching algorithm that satisfies the following specification:

> Given two character strings, string A of length n and string B of length m, the algorithm returns 0 if B is *not* a substring of A, and returns the start position s of B in A if B is a substring of A.

For example, if $A =$ "`assignment`" and $B =$ "`sign`", then the algorithm should return the number 3. Develop your algorithm from first principles, by treating strings as arrays of characters, that is, use only the methods `length()` and `charAt()` to access characters in a string.
Proceed as follows:

a) Write down the idea of your algorithm.

b) Develop the algorithm in pseudocode. Make sure that your algorithm is valid for all special cases of input data.

c) Implement the algorithm in Java.

(10 Points)

Submission: Until Mon, 24 March 2014, 8:30 am, to

`dsa-submissions AT inf DOT unibz DOT it.`