Midterm Exam Data Structures and Algorithms 2015-2016

Friday September 25, 2015, 15.30-17.00

7 exercises

Answers may be given in English or in Dutch.

Exercise 1. *(5+5+5 points)*
This exercise is concerned with $\mathcal{O}$.

(a) Consider the pseudo-code for the procedure partition for quick-sort. It works on a sub-array $A[p \ldots r]$ of an array A of length n. Explain why its worst-case running time is in $\mathcal{O}(n)$.

```
Algorithm partition(A, p, r):
    x := A[r]
    i := p − 1
    for j = p to r − 1 do
        if A[j] ≤ x then
            i := i + 1
            exchange A[i] with A[j]
    exchange A[i + 1] with A[r]
    return i + 1
```

(b) What is the worst-case time complexity in terms of $\mathcal{O}$ of the following loop? Motivate your answer.

```
Algorithm LoopA(n):
    a := 0
    for i := 1 to 2n do
        for j := 1 to i do
            a := a + i
```

(c) Give an example of an algorithm that has linear best-case time complexity, and quadratic worst-case time complexity. Motivate your answer.

Exercise 2. (*5+5 points*)
This exercise is concerned with sorting.

(a) What is the worst-case time complexity in terms of $\mathcal{O}$ of selection sort, bubble sort, merge sort, quick sort, counting sort. You do not need to motivate your answer.

(b) What is the lower bound for the worst-case time complexity of comparison-based sorting? You do not need to motivate your answer.

Exercise 3. (*5+6 points*)
This exercise is concerned with insertion sort; its pseudo-code is given:

```
Algorithm insertionSort(A, n):
  for j := 2 to n do
    key := A[j]
    i := j − 1
    while i ≥ 1 and A[i] > key do
      A[i + 1] := A[i]
      i := i − 1
    A[i + 1] := key
```

(a) Apply insertion sort to the array $[3, 4, 1, 5, 2]$.

Give (at least) the intermediate result after every iteration of the for-loop.

(b) Give pseudo-code for a recursive version of insertion sort, that sorts the first $n - 1$ elements recursively, and inserts the last element at the right position.

Exercise 4. (*5+5+5 points*)

(a) Give the merge-sort tree for sorting the input array $[3, 6, 1, 7, 5, 8, 2, 4]$.

(b) Solve the following recurrence equation for merge sort
(you may assume n to be a power of 2):

$$T(n) = \begin{cases} 1 & \text{als } n = 1 \\ 2T(\frac{n}{2}) + n & \text{als } n > 1 \end{cases}$$

(c) Explain the divide-and-conquer programming paradigm.

Exercise 5. (*6+7 points*)
This exercise is concerned with singly linked lists. In a node v we have operations $v.next$ and $v.element$ with the suggested meaning. For a list L we have operations $L.first$ and $L.last$ with the suggested meaning. Do not assume predefined operations on lists.

(a) Implement a queue with the operations enqueue and dequeue as a singly linked list.

(b) Give a $\Theta(n)$-time non-recursive procedure that reverses a singly-linked list of n elements, only using a constant amount of additional storage.

Exercise 6. (*5+5 points*)
This exercise is concerned with hashing.

(a) What is the worst-case time complexity of searching for hashing where collisions are solved using chaining? Briefly explain your answer.

(b) We consider a hash table of length 11 (an array with indices $0 \ldots 10$), and the hash function $h(k) = k \mod 11$. Add the following numbers in this order to the initially empty hash table:

1 13 2 24 10 12

solving collisions by open addressing with linear probing.

Exercise 7. (*6+5+5 points*)
This exercise is concerned with binary search trees and AVL-trees.

(a) Give pseudo-code for a recursive procedure for searching a key in a binary search tree. The inputs for the procedure are a pointer x to a node, and a key k. The output is a pointer to a node with key k if such node exists, and nil otherwise.

(b) What is the worst-case time complexity of search in a binary search tree? And in an AVL tree? Briefly motivate your answers.

(c) Construct an AVL-tree by inserting the numbers

3 2 1 5 4 6

one by one, starting from the empty tree. After each insertion, rebalance the tree if needed. Give your answer in pictures (with comments if needed).

The mark for the midterm is (the total number of points plus 10) divided by 10.