ÉCOLE POLYTECHNIQUE
FÉDÉRALE DE LAUSANNE

Old Exam Questions, Algorithms 2013-2014

This document contains a sample of exam questions. In particular, the problems have previously appeared on a (midterm) exam. The points of the exercises reflect their importance to the final score of the exams, where the total amount of points of an exam is 100. One should note however, that these exams were during 3 hours whereas the mid-term will be during 2 hours. For some exercises, I have also written comments on how related the questions are to this year's version of the course.
Good luck with your preparation!

1 (5 pts) Consider the natural implementation of the three sorting algorithms (Insertion-Sort, Merge-Sort, Heap-Sort) that we studied in class. Which one of these implementations is *not* "in-place"?

2 (5 pts) Simplify and arrange the following functions in increasing order according to asymptotic growth.

$$3^N, \sqrt{4^N}, \log^2 N, \sqrt{N}, N^2, \log N, 20N$$

3 (10 pts) Let $f(n)$ and $g(n)$ be the functions defined for positive integers as follows:

Function $f(n)$**:**
```
ans ← 0
for i = 1, 2, ..., n − 1 do
    for j = 1, 2, ..., n − i do
        ans ← ans + 1
    end for
end for
return ans
```

Function $g(n)$**:**
```
1: if n = 1 then
2:     return 1
3: else
4:     return g(⌊n/2⌋) + g(⌊n/2⌋)
5: end if
```

3a (6 pts) What is, in Θ notation, the running time of these algorithms, given that addition runs in time $\Theta(1)$?

3b (4 pts) What is, in Θ notation, the running time of algorithm $g(n)$ if we at line 4 replace $g(\lfloor n/2 \rfloor) + g(\lfloor n/2 \rfloor)$ by $2g(\lfloor n/2 \rfloor)$?

(Assume that multiplication runs in time $\Theta(1)$.)

4 (10 pts) Let $f(n)$ be the function given below in pseudocode:

```
Call: f(n)
1: a ← 0
2: b ← ln(n)
3: for i = 1, . . . , n do
4:     a ← a + b
5: end for
6: for j = 1, . . . , n do
7:     for k = 1, . . . , j do
8:         for ℓ = j + 1, . . . , j + n do
9:             a ← a + b
10:        end for
11:    end for
12: end for
13: return a
```

4a Find a closed-form formula for $f(n)$.

4b Find s and t such that

$$f(n) = \theta\big(n^s \cdot \ln(n)^t\big).$$

4c What is, in θ notation, the running time of this algorithm, given that line 2 runs in time $\theta(1)$?

5 (10 pts)

5a (4 pts) Give a formal specification of the following problem: given an array of integers, and another integer x, determine whether there are two elements in the array that sum up to x. *Note: What formal means is slightly unclear from the perspective of this year's course.*

5b (6 pts) Design an algorithm that solves the problem of the previous part in $O(n \log n)$ steps, where n is the length of the array, and a step is either an addition or a comparison of integers.

6 (10 pts) Suppose that, given a list of distinct integers and a positive integer i, we wish to find the i largest integers on the list. Consider the following two approaches to solve the problem:

1. Use the MERGESORT algorithm to sort the list in increasing order, and pick the last i items from the resulting sequence.
2. Create a heap in a bottom-up fashion, and then obtain the i largest elements by calling the DELETEMAX operation i times.

What is the running time of each solution? Which approach is more favorable for finding the 10 largest items on a list of a billion integers?

7 (15 pts) Suppose that you are given a sorted sequence of n *distinct* integers $\{a_1, \ldots, a_n\}$. Give an algorithm to determine whether there exists an index i such that $a_i = i$, outputting one i if such an i exists, and which uses $O(\log(n))$ steps. For example, in $\{-10, -3, 3, 5, 7\}$, $a_3 = 3$, whereas $\{2, 3, 4, 5, 6, 7\}$ has no such i.

8 (15 pts) Given an $O(n \log(k))$-algorithm that merges k sorted list of integers with a total of n elements into one sorted list. *(Hint: use a heap of size k)*

9 (20 pts) **Analysis of d-ary heaps:** a *d-ary heap* is similar to a binary heap with one exception. The non-leaf nodes have d children instead of 2 children.

9a How would you represent a d-ary heap in an array?

9b What is the height of a d-ary heap of n elements in terms of n and d?

9c Let `EXTRACT-MAX` be an algorithm that returns the maximum element from a d-ary heap and removes it while maintaining the heap property. Give an efficient implementation of `EXTRACT-MAX` for a d-ary heap. Analyze its running time in terms of d and n.

9d Let `INSERT` be an algorithm that inserts an element in a d-ary heap. Give an efficient implementation of `INSERT` for a d-ary heap. Analyze its running time in terms of d and n.

10 (20 pts) A *contiguous subsequence* of a sequence S is a subsequence consisting of consecutive elements of S. For example, we might have

$$S = (2, -5, 10, 4, -12, 5, 0, 1),$$

for which $(10, 4, -12)$ is a contiguous subsequence but $(2, 4, 5, 1)$ and $(-12, 4)$ are not.

Using dynamic programming, design a linear time algorithm that, given a sequence $S = (s_1, \ldots, s_n)$ of integers, finds a contiguous subsequence of S with maximum sum. That is, a contiguous subsequence $(s_i, s_{i+1}, ..., s_j)$ of S for which the summation of all entries $(s_i + s_{i+1} + \cdots + s_j)$ is maximum.

Hint: for each $j \in \{1, ..., n\}$, consider the subproblem of finding the optimal contiguous subsequence within the first j elements of S.

11 (20 pts) Let $x_1 \ldots x_m$ and $y_1 \ldots y_n$ be two strings. For $0 \leq i \leq m$ and $0 \leq j \leq n$, let $c[i,j]$ be the length of the longest common subsequence of $x_1 \ldots x_i$ (or the empty string if $i = 0$) and $y_1 \ldots y_j$ (or the empty string if $j = 0$). In other words, $c[i,j]$ is defined to be the maximum k such that there exist indices $1 \leq i_1 < i_2 < \cdots < i_k \leq i$ and $1 \leq j_1 < j_2 < \cdots < j_k \leq j$ satisfying $x_{i_\ell} = y_{j_\ell}$ for all $\ell = 1, \ldots, k$.

11a (10 pts) Complete the recurrence relation for $c[i,j]$ that can be used for dynamic programming:

$$c[i,j] = \begin{cases} \underline{\hspace{4cm}} & \text{if } i = 0 \text{ or } j = 0 \\ \underline{\hspace{4cm}} & \text{if } i,j > 0 \text{ and } x_i = y_j \\ \underline{\hspace{4cm}} & \text{if } i,j > 0 \text{ and } x_i \neq y_i \end{cases}$$

11b (10 pts) Use the recurrence relation to return the length of the longest common subsequence of the two strings BABDBA and DACBCBA by filling in the table of $c[i,j]$ values below (in a bottom-up dynamic programming fashion).

	j	0	1	2	3	4	5	6	7
i		y_j	D	A	C	B	C	B	A
0	x_i								
1	B								
2	A								
3	B								
4	D								
5	B								
6	A								

12 (20 pts) A max-min algorithm finds both the largest and the smallest elements in an array of n integers. Design and analyze a divide-and-conquer max-min algorithm that uses $\lceil 3n/2 \rceil - 2$ comparisons for any integer n. (You will receive 10 points if you can show this for the case when n is a power of 2, and an additional 10 points if you can prove it for general n.)

Hint: If $T(n)$ denotes the number of comparisons of your algorithm, try to find a recursion relating $T(n+m)$ to $T(n)$ and $T(m)$. Then study first the case where n is a power of 2.

13 (20 pts, *Note: this one is quite tricky and was a bonus problem*) Consider a binary heap containing n numbers where the root stores the largest number. Let $k < n$ be a positive integer, and x be another integer. Design an algorithm that determines whether the kth largest element of the heap is greater than x or not. The algorithm should take $O(k)$ time and may use $O(k)$ additional storage.

Hint: don't try to find *the kth largest element.*