Lecture

We will start on computational geometry. We will see two algorithms for computing the convex hull of a set of points. You should read CLRS section 33.0, 33.1, and 33.3.

Expected values

a) Let X be a random variable which assumes the values 2, 5 and 8 with probabilities 1/3, 1/2 and 1/6 respectively, i.e., $P[X = 2] = 1/3$ etc. What is the expected value $\mathbb{E}X$ of X.

b) Let X be a random variable which assumes the values $2^n, ..., n = 0, 1, 2, ...$ with probabilities $2^{-(n+1)}$ respectively, i.e., $P[X = 2^n] = 1/2^{(n+1)}$ etc. What is the expected value $\mathbb{E}X$ of X.

Randomized print Consider the following program:

$i \leftarrow 0$;
while ($rand$(1, 10)$\neq$ $rand$(1, 10)) **do**
$i \leftarrow i + 1$;
end while
print(i);

What is the probability that the value 3 is printed? What is the expected value of i that is printed?

Boxes of beer You are given n boxes $B_1, \ldots, B_n$. Exactly k boxes contain a bottle of beer ($k \leq n$) the rest is empty. From the outside one cannot see whether a box is empty or not. The aim is to find a box with a beer it. The following deterministic algorithm is suggested: Open the boxes $B_1, B_2, \ldots$ in this order. The algorithm stops when a beer is found. We count opening a box as one computational step.

a) What is the best-case running time of the deterministic algorithm.

b) What is the worst-case running time of the deterministic algorithm.

c) Design a randomized algorithms for this problem.

d) What is the expected running time of your randomized algorithm.

e) What is the worst-case running time of your randomized algorithm.

Nuts and bolts. (G. J. E. Rawlins) You have a mixed pile of N nuts and N bolts and need to quickly find the corresponding pairs of nuts and bolts. Each nut matches exactly one bolt, and each bolt matches exactly one nut. By fitting a nut and bolt together, you can see which is bigger. But it is not possible to directly compare two nuts or two bolts. Given an efficient method for solving the problem.[1]
Hint: customize quicksort to the problem. Side note: only a very complicated deterministic $O(NlogN)$ algorithm is known for this problem.

---

[1] This exercise is from http://algs4.cs.princeton.edu/23quicksort/

Quicksort Solve CLRS 7.4-5.

Implementation of quicksort [CJ] Implement a program that given a sequence of integers sorts these using quicksort, and outputs the sorted sequence.

Mandatory assignment [old exam exericse] Given is a sequence $x_1, x_2, \ldots, x_n$ of n integers. The sequence has a *majority element* t, if a number t occurs strictly more than $n/2$ times in the sequence. For example the sequence $1, 2, 3, 1, 2, 2, 2$ has a majority element, namely 2, whereas the sequence $2, 2, 1, 2, 3, 3$ has no majority element.

Here is a randomized algorithm for this problem: Randomly pick an number x_i from the sequence. Then check whether it occurs more than $n/2$ times in the sequence. If so it is a majority element. Otherwise the algorithm answers that there is no majority element.

```
i ← rand(1, n);
t ← x_i;
k ← 0;
for j = 1, ..., n do
|   if (x_j = t) then
|   |   k ← k + 1
|   end
end
if (k > n/2) then
|   return Majority element is t.
else
|   return No majority element.
end
```

The procedure $rand(1, n)$ returns a random number $i \in \{1, 2, \ldots, n\}$ according to the uniform distribution, that is, every number has probability $1/n$ to be returned.

a) Determine the running time of the algorithm.

b) Can the algorithm return a majority number, if the sequence does not have one? Justify your answer.

c) Can the algorithm claim that there is no majority number, even though the sequence does have one? Justify your answer.

d) Determine the probability for incorrect answers.
