6331 - Algorithms, Spring 2014, CSE, OSU
Homework 3
Instructor: Anastasios Sidiropoulos

Problem 1. In Section 12.4 of the book it is shown that the expected height of a randomly built binary search tree on n distinct keys is $O(\log n)$. That is, if we start with an empty tree, and we insert n distinct elements in a *random* order, then the expected height of the tree is $O(\log n)$. Here, by "random order" we mean an order chosen uniformly at random from the set of all possible orderings of the n keys. Also note that since the ordering is random, the height of the resulting tree is a random variable. In other words, the above result states that the expectation of this random variable is $O(\log n)$.

Use the above fact to construct a randomized algorithm for sorting n elements, with expected running time $O(n \cdot \log n)$.

Hint: You don't need to use the proof of the above statement about the expected height from Section 12.4. It is enough to assume that the statement is true.

Problem 2. Give an algorithm with running time $O(n^2)$ to find the longest monotonically increasing subsequence of a sequence of n numbers.

Problem 3. Alice and Bob play the following game. They start with a pile of n sticks. Alice plays first, and takes k sticks from the pile, for some integer $1 \leq k \leq \lfloor n/2 \rfloor$. Bob plays next, and he also picks at least one, and at most half of the remaining sticks. The two players keep alternating until there is only one stick left. At that point, the player who has to play next is the loser.

Here is an example:

- Initially, there are $n = 20$ sticks.
- Alice picks 10 sticks, so there are 10 left.
- Bob picks 1 stick, so there are 9 left.
- Alice picks 4 sticks, so there are 5 left.
- Bob picks 2 sticks, so there are 3 left.
- Alice picks 1 stick, so there are 2 left.
- Bob picks 1 stick, so there is 1 left, and Alice loses.

Describe an algorithm that given n, outputs an integer k, such that the best strategy for Alice is to pick k sticks. That is, if it is always possible for Alice to win when she starts with n sticks on the pile, then it is never possible for Bob to win when he starts with $n - k$ sticks on the pile.

Your algorithm should have running time polynomial in n.

Hint: Using dynamic programming, compute the optimal value for k for increasing values of n.