Design and Analysis of Algorithms
Class Test 1

August 30, 2012

Time: 1 hr 30 min | Marks: 30

Instructions:

1. Solve as many questions as you can.
2. Try to optimize your algorithm as much as possible.
3. Whenever you are asked to design an algorithm, write a correctness proof and analyze its time complexity.

Questions:

1. Compare each of the following pairs of functions and state whether they are related by O, o, Ω, ω, or Θ relation. **6**
   (a) $\sqrt(n), (\log n)^2$
   (b) $n^{\frac{3}{2}}, n^{\frac{1}{2}}$
   (c) $2^{\sqrt{n}}, n^2$
   (d) $2^n, 4^n$
   (e) $2^n, n!$
   (f) $2^n, n^{\log\log n}$

2. Solve the following recurrences by the method indicated:
   (a) $T(n) = 4T(\frac{n}{2}) + n$ by the master method **2**
   (b) $T(n) = T(\frac{n}{2}) + \Theta(1)$ by substitution method **2**
   (c) $T(n) = 2T(\frac{n}{2}) + n\log n$ by recursion tree. Can the master method be applied here? If yes, also find the answer by the master method. If no, why? **3**

3. To cut a wooden board, a sawmill charges proportional to the length of the board. The cost of cutting a single board into many smaller boards will thus depend on the order of the cuts.

   As an example, lets say cutting a 10m board into two pieces costs \$10. Then to cut a 10m long board at marked positions 3m and 5m costs \$10+\$7=\$17 if it is first cut at position 3m and then at 5m. On the other hand, if it is cut at 5m position first, and then at 3m, it would cost \$10+\$5=\$15.

   As input, you are given a board of length n with k marks on it. You need to give an algorithm that, given an input length n and a set of k desired cut points along the board, will produce a cutting order with minimal cost in $O(k^c)$ time, for some constant c.

   Does a greedy strategy work here? If yes, give a greedy algorithm and prove its optimality. Otherwise give a dynamic programming solution. **6**

4. Consider the following pseudocode for finding the ith smallest element from an array A of n elements: Initial call is made with Select(A,i,1,n).

```
Select(A, i, x, y) //Finds ith smallest of A[x, ..., y].
Divide the elements of A into groups of 5 each.
m = y − x + 1 //Number of elements in the portion of A being considered.
B = array of medians of all the groups
r = Select(B, ⌈m/10⌉, 1, ⌊m/5⌋)
Partition A around r. Let q be the correct position of r in A.
if q = i then
  return r.
else
  if q > i then
    return Select(A,i,x,x + q)
  else
    return Select(A,i − q,x + q + 1,y)
  end if
end if
```

(a) Show how to implement Select in-place i.e. without using the auxillary array B. **5**

(b) What is the running time of Select if elements are divided into groups of 3 instead of groups of 5? **2**

5. Given a weighted directed graph with no negative weight cycles (but possibly negative weight edges), let m_{uv} be the minimum number of edges which appear on a shortest path from u to v. Let $m = max_{u,v} m_{uv}$. Modify the Bellman-Ford algorithm to run in time $O(mE)$ (assuming m is not known to you). **4**