OLD FINAL EXAM
SOLUTIONS

1. (20) Describe the algorithm TOPDOLLAR(PRICE, N). The input is an array of nonnegative real numbers PRICE[1⋯N] where PRICE[I] represents the price for a rod of length I. Your output should be VALUE, which should be the optimal total price among all cuttings of the rod. You may, and should, use an auxiliary array. You may write in pseudocode or actual code but in either case you must give clear comments describing what you are doing and why it works. Analyze (just the answer will not suffice!), in Θ-land, the time for your algorithm.

Solution: This is the RODCUTTER algorithm, as discussed in the text and in class.

2. (5) For n large which is faster: Θ(lg³ n) or Θ(√n) algorithm?

Solution: Θ(lg³ n). Any power of the log grows slower than any power of n so that the time is less so it is faster.


(a) How long will COUNTING-SORT take?

Solution: O(N²). The main part is creating the array of length N².

(b) How long will RADIX-SORT take using base N?

Solution: N applications of COUNTING-SORT on N, each one takes O(N), so total O(N²).

(c) Give an algorithm (no proof required!) that does better than both of the above, and does so in worst case.

Solution: Two possibilities are MERGESORT and HEAPSORT but not QUICKSORT which does badly in worst case.

4. (5) In a max Heap what is the property of the value at the root? (That is, how does it compare to the values of the other nodes.)

Solution: It is the biggest value.

5. (20) Let G be a connected graph on vertex set V. Let a nonnegative weight w(e) be assigned to every edge e. Let S ⊂ V with S ≠ ∅ and S ≠ V.

(a) Prove (yes, prove!) that the Minimal Spanning Tree necessarily contains that edge e = {x, y} with x ∈ S and y ∉ S which has
minimal weight. A good picture won’t hurt. Assume no two edges have the same weight. (Warning: Saying that an algorithm tells you to pick this edge $e$ is not a proof!)

**Solution:** Take the MST $T$ not containing $e$. There is a path from $x$ to $y$ in the tree and at some point an edges must cross from some $v \in S$ to some $w \notin S$. Replace $\{v, w\}$ by $\{x, y\}$ to get a tree with smaller weight.

(b) In what algorithm is the above result used.

**Solution:** Prim’s Algorithm.

6. (10) Consider the recursion $T(2n) = 3T(n) + n + 1$ with initial value $T(1) = 5$.

(a) What is $T(4)$?

**Solution:** With $n = 1$, $T(2) = 3 \cdot 5 + 1 + 1 = 17$ so with $n = 2$, $T(4) = 3 \cdot 17 + 2 + 1 = 54$.

(b) What is $T(n)$ in $\Theta$-land.

**Solution:** This is low overhead case and $T(n) = n^\alpha$ with $\alpha = \log_2 3$.

7. (20) If multiplication mod $p$ takes 1 second then show, using Island-Hopping, how to calculate $x^{1023}$ quickly. How many seconds did it take? (Hint: What is special about 1023?)

**Solution:** $1023 = 2^{10} - 1$, in binary it is all ones. Square $x$ nine times getting $x^{2^i}$ for $1 \leq i \leq 9$. Multiply these and $x$ together getting $x^{1023}$. The last takes nine seconds so a total of eighteen seconds.

8. (20) Find the Huffman code on letters $a, b, c, d, e, f$ with frequencies $.13, .16, .09, .45, .05, .12$ respectively. Give pictures and words clearly indicating the intermediate steps in finding the code.

**Solution:** Assignment.

9. (5) State the Chinese Remainder Theorem.

**Solution:** When $m, n$ are relatively prime and $a, b$ arbitrary there is a solution to the system $x \equiv a \pmod{m}$ and $x \equiv b \pmod{n}$.

10. (15) We call a positive integer $n$ **SQUAREFREE** if there is no integer $d > 1$ with $d^2$ dividing $n$. We call $n$ **NOTSQUAREFREE** if $n$ is not **SQUAREFREE**.

(a) Argue **NOTSQUAREFREE** is in NP.

**Solution:** Oracle gives $d$. Verifier calculates $d^2$ and divides it into $n$. 


(b) (This is harder!) Argue SQUAREFREE is in NP.
Solution: Oracle gives the prime factorization of $n$. Verifier checks that the multiplication comes out to $n$ and, via AKS, that all the “primes” really are primes, and that all of the primes were distinct and to the first power.

11. (5) What was the breakthrough of Agrawal, Kayal and Saxena?
Solution: Polynomial Test for Primality.

12. (20) In Depth-First Search each vertex $v$ gets a discovery time $d[v]$ and a finishing time $f[v]$. Let $G$ be a graph and $v, w$ two distinct vertices for which $w \in \text{Adj}[v]$.

Solution: During DFS-VISIT[$v$], $w$ will be found so that DFS-VISIT[$w$] will be a subroutine of DFS-VISIT[$v$] and so will end first.

(b) Now further assume that $G$ is a DAG (a Directed Acyclic Graph.) Assume $d[w] < d[v]$. Argue that $f[w] < f[v]$. 
Solution: As $w \in \text{Adj}[v]$ and $G$ is a DAG there is no path from $w$ to $v$. Thus DFS-VISIT[$w$] does not find $v$ and so is completed before $v$ is discovered, hence before it is finished.

13. (5) What is the difference, if any, between a $\Theta(\lg n)$ algorithm and a $\Theta(\log_{10} n)$ algorithm? (Short reason please.)
Solution: None. $\lg n$ and $\log_{10} n$ differ by a constant factor and so are the same in $\Theta$-land.