Questions for PhD Qualifying Exam in Algorithms

Spring 2012

DIRECTIONS: Please make sure to write very clear and easy to read answers.

PART I: REQUIRED:

A. Give a detailed and correct formal definition for the sets:

1. O

2. Omega

3. Theta

4. P

5. NP

6. NPC
B. Explain briefly the importance of efficiency in your PhD research. Does the research involve solving an optimization problem? Do you need your code to work in real time? Can the problem that you deal with have very large inputs? What are the bottlenecks that make efficiency hard to achieve? Do you need to design new algorithms?
PART III (EASIER QUESTIONS): PICK ANY AND ONLY THREE

1. The quicksort routine implemented in many libraries is frequently not a pure implementation of quicksort. For problems of size 6 or less, the sorting implementation may use insertion sort rather than further recursive calls to quicksort. Explain why this is done -- what is the reasoning behind this implementation decision?

2. Why is best-case running time not a good measure of the efficiency of an algorithm?

3. Is $2^{n+1} \in O(2^n)$? Is $2^{2^n} \in O(2^n)$? Explain your answers.
4. Use a recursion tree to determine a good asymptotic upper bound on the recurrence
   \[ T(n) = 2T(n-1) + 1. \] Use the substitution method to verify your answer.

5. Assume open addressing. Write pseudo code for inserting a record with key k into a
   hash table.
PART II (HARDER QUESTIONS): PICK ANY AND ONLY THREE

1. You are hired to arrange meetings between investors and startup companies. Each investor is interested in meeting with only one startup company, and the investors have different areas of interest. For example Investor A might be interested in companies that work in biotech or e-commerce, while Investor B is interested in e-commerce, social network startups, and coffee shops. Each startup company is involved in only one area.

What would your approach be to find an arrangement that maximizes the number of meetings between a large set of investors (each with different areas of interest), and startup companies (each with only one area)? Can you guarantee an optimal solution? What would the Big-O time complexity of your approach be?
2. You run a ferry that moves cars from Long Island to Connecticut. The ferry has two lanes into which you can load cars. Cars have different (integer) lengths; they line up in single file, and then you can load them onto the ferry in order, directing each car to either the left or right lane.

For example, suppose your ferry is of length 10, and the cars are of length 4, 3, 6, and 7. If you load the cars in as left/right/left/right, you can place all the cars on the ferry. A different order, starting with two lefts, or with two rights, would leave the last car off the ferry.

a. If you know the order and length of cars in advance, and the length of the ferry, describe an efficient method to place the maximum number of cars onto the ferry. You might want to consider using dynamic programming for your solution.

b. Is your algorithm polynomial in the size of the input to the problem?
3. You are given a connected weighted graph $G = (V, E, W)$. Given the Mystery algorithm below:

```
Mystery(V, E, W)
    Sort E in non-increasing order of the weights
    A=E
    for each edge e taken in non-increasing order by weight
        if A-{e} is a connected graph
            A=A-{e}
    Return A
```

a. Does G have a minimum spanning tree? Explain

b. Do the edges in A contain every vertex in V? Explain

c. Do the edges in A form a minimum spanning tree? Prove that it does or show that it does not.

d. Describe an efficient implementation of Mystery and analyze its computation time. (Do not write code).
4. Prof. A and Prof. B found a treasure of \( x \) diamonds and \( y \) opals. All the diamonds are identical and each is worth \( D \). Similarly all opals are identical and each is worth \( O \).
   a. Describe an algorithm that the professors can use to divide the treasure as evenly as possible.
   b. What is the time complexity of your algorithm?

5. Assume that we have a polynomial time algorithm to determine whether a given graph has a Hamiltonian cycle. Describe a polynomial algorithm for generating a Hamiltonian cycle.