

15-251: Great Theoretical Ideas In Computer Science

Homework 11 (due Thursday, November 24)

Directions: Write up carefully argued solutions to the following problems. The first task is to be complete and correct. The more subtle task is to keep it simple and succinct. Your solution should be clear enough that it should explain to someone who does not already understand the answer why it works. You may use any results proven in lecture without proof. Anything else must be argued rigorously. Unless otherwise specified, all answers are expected to be in closed form.

0. Warmup (0 points)

1. Given two positive integers x and y , give an algorithm that takes $O(\log y)$ multiplications to calculate x^y .
2. Show that $O(n^{\log_a b}) = O(b^{\log_a n})$.
3. Prove or disprove the following:
 - (a) $f(n) \in O(a(n))$ and $g(n) \in O(b(n))$ implies that $f(n) + g(n) \in O(a(n) + b(n))$.
 - (b) $j^n \in \Theta(k^n)$ for all values $j, k \geq 2$.

1. Asymptotic Growth (20 points)

Order each list of functions below according to increasing asymptotic growth. Your answers should look something like: $n^{1/2} < n = 5n < n^2$, where here $a_n < b_n$ means $a_n = O(b_n)$ and $a_n = b_n$ means $a_n = \Theta(b_n)$. Provide a brief argument justifying each successive step in the ordering.

- (a) (10 points) Fast growing functions: $(\log^* n)^{\log n}$, $n^{\log^* n}$, $n!$, $2^{n\sqrt{\log n}}$.
- (b) (10 points) Slow growing functions: $2^{\log^* n}$, $2^{\sqrt{\log n}}$, $\log \log \sqrt{n}$, n , \sqrt{n} .

2. A Not So Happy Happyville (15 points)

Happyville, a quiet town in a quaint land, far far away, consists of a total of n villagers. Recently, fewer than half of the villagers became tired of the blissful nature of the town, and decided to wreak havoc upon the city. Your goal, as the intrepid investigator, is to find at least one villager in Happyville who has not become corrupt.

It turns out that each of the residents of Happyville has the unique ability to tell whether any other resident of the city is corrupt. In fact, for any two different people, call them Alice and Bob, you may ask Alice whether or not she believes Bob is corrupt. If Alice is indeed happy, she will correctly tell you whether Bob is happy or corrupt. However, if Alice is corrupt herself, she will give any answer that she so wishes.

If each test consists of asking one villager whether or not he/she thinks another villager is happy, design an algorithm that uses at most n tests to find a single happy person.

3. Main Rules (15 points)

Suppose that you have a long list of database entries (over 100,000,000,000), with duplicates allowed. You'd like to determine whether or not an entry appears more than half of the times in the list. Call such an element the *main element*. Since running an algorithm on a large database requires significant system resources, you'd like a solution that takes relatively few resources and cycles.

Suppose that you have an iterator $I(x)$ which, when called, will give you the next database entry in the database x . Given a maximum of 5 variables, each which can store a single database entry, an integer, or a double, create a simple procedure that returns the main element of the database if one exists, and that returns any arbitrary element if a main element does not exist. Your solution should require only one pass through the database.

Hint: You do not actually need all 5 variables to solve this problem.

4. Interview Questions (20 points)

When software companies interview CMU students for full-time and internship positions, they commonly ask technical questions on algorithms. Typically, these questions test a student's ability to formulate a solution to a typical problem found in software development and to analyze the runtime of said solution. The following problems have been asked by various companies (Google, VMWare, Microsoft, etc.) at some point in time.

- (a) (8 points) As shown in the last few lecture slides of "Grade School Revisited," preprocessing can help save time when running future operations. Suppose that you are given an n -by- n matrix of integers and want to perform queries to compute the sum of all of the values from (x_0, y_0) to (x_1, y_1) . In other words, you wish to be able to quickly find the sum over the entries in any subrectangle. Show how you can preprocess the matrix with $O(n^2)$ operations, so that any query can be performed with $O(1)$ operations.
- (b) (12 points) Given n points on a coordinate axis (with each point consisting of an x and y value), give an algorithm that will determine the line that goes through the greatest number of points. Give a $\Theta(\cdot)$ bound on the runtime of your solution, and explain why your algorithm satisfies that runtime.

Note that it is not difficult to find an $O(n^3)$ solution to this problem, but a faster solution exists. If you choose to use data structures taught in 15-111, you may state and use runtimes of lookups and inserts into those data structures without proof in your explanation. Finally, recall that the line that goes between two points can be written as $y_1 - y_0 = b(x_1 - x_0)$, where b is the slope.

5. Egg Drop Loop (20 points)

(Another interview question...) An agricultural firm has developed a new type of egg, whose shell is much stronger than the shell of a normal egg. You have a building with n floors, and your task is to determine the maximum floor number $1 \leq k \leq n$ out of which you can throw an egg without it breaking. However, since a lot of money has been put into this research, the firm has only given you two of the new eggs to experiment with.

- (a) (3 points) If you have only one egg, what is the best algorithm?
- (b) (10 points) For two eggs, give a lower bound on the number of trials required by an optimal algorithm.
- (c) (7 points) Using part (b) above, give an algorithm that requires at most twice as many trials as the optimal algorithm in the worst case.

Note: Proving that an algorithm requires at least m operations can often be quite difficult. Be careful when writing your solution to part (b). As an example, you may wish to look at Recitation 10, Problem 4.

6. Fruit for Thought (10 points)

A large pile of two thousand fruits is made up of a thousand fruits weighing one pound each, and another thousand fruits weighing 0.99 pounds each. Your goal is to pick out two piles of fruit with an equal number of fruit in each of them, but different total weights. Your balance will tell you the weight of the fruit in the left pan minus the weight of the fruit in the right pan. What is the minimal number of weighings needed to do this?