Final Examination

- This exam ends at 3:15 P.M. You must solve exactly 8 of 10 problems. You have 175 minutes to earn 200 points.
- This exam is closed book. You may use one $8\frac{1}{2}'' \times 11''$ crib sheet. When the exam is over, staple your crib sheet to the back of the exam.
- Write your solutions in the space provided. If you need more space, write on the back of the sheet containing the problem. Do not put part of the answer to one problem on the back of the sheet for another problem.
- Do not spend too much time on any problem. Read them all through first and attack them in the order that allows you to make the most progress.
- Show your work, as partial credit will be given. You will be graded not only on the correctness of your answer, but also on the clarity with which you express it. Be neat.

Circle 8 problems in this table.
We will only grade those problems.

Problem	Points	Grade
1	25	
2	25	
3	25	
4	25	
5	25	
6	25	
7	25	
8	25	
9	25	
10	25	
Total	200	

Name: ______________________________

Problem 1.

Give asymptotically tight (Θ) bounds for $T(n)$ in each of the following recurrences. Justify your solutions by naming the particular case of the master theorem, by iterating the recurrence, or by using the substitution method. Make sure you prove both upper and lower bounds.

In each case, you should assume that $T(a) = O(1)$ for sufficiently small a.

[8 points] (a) $T(n) = 3T(n/2) + n^2$.

[8 points] (b) $T(n) = 8T(n/2) + n^3$.

[9 points] (c) $T(n) = T(3n/4) + T(n/2) + n^2$.

Problem 2.

When doing arithmetic on large numbers, it is often necessary to do the calculations in smaller parts. We consider multiplying two n-bit numbers, X and Y. Assume that $X = A \cdot 2^{n/2} + B$ and $Y = C \cdot 2^{n/2} + D$, where A, B, C, and D are $(n/2)$-bit numbers.

[12 points] (a) We can calculate XY using the fact, $XY = AC2^n + (AD + BC)2^{n/2} + BD$. Assume that adding/subtracting two k-bit numbers can be done in $O(k)$ time. Give a recurrence for $T(n)$, the time required to multiply two n-bit numbers in this fashion, and then solve the recurrence.

[12 points] (b) We can also compute $XY = AC2^n + [(A-B)(D-C) + AC + BD]2^{n/2} + BD$, reusing the calculations for AC and BD. Give a recurrence for $T(n)$, the time to multiply two n-bit numbers using this method, and solve this recurrence.

[1 point] (c) Which method is better, asymptotically?

Problem 3.

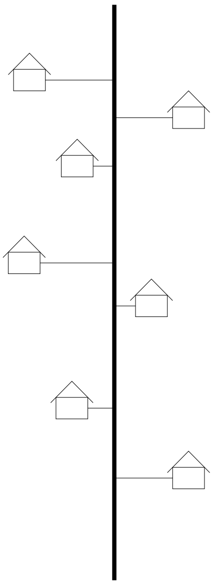
[8 points] (a) Suppose that we have an array of n data records to sort and that the key of each record has the value 0 or 1. Give a simple, linear-time algorithm for sorting the n data records in place. Use no storage of more than constant size in addition to the storage provided by the array.

[8 points] (b) Can your sort from part (a) be used to radix sort n records with b-bit keys in $O(bn)$ time and using at most a constant amount of additional storage? Explain how or why not.

[9 points] (c) Suppose that the n records have keys in the range from 1 to k. Show how to modify counting sort so that the records can be sorted in place in $O(n + k)$ time. You may use $O(k)$ storage outside the input array. (*Hint:* How would you do it for $k = 3$?)

[25 points] **Problem 4.**

Due to the apparent difficulty of keeping power to all the houses up and down the West Coast of the country, PG&E has asked you to solve the following problem. Their plan is to build one new "reliable" main power line which runs **due North-South** from the Mexican border, up to the Canadian border. Then, for every house, they will create a secondary line which connects that house to the main power line along the shortest path (either toward the East or West). The cost of this plan depends on the the sum of these secondary lines which connect the houses to the main power line. They are free to pick the East/West placement of the main line, so long as it lies due North/South.



They have asked you to pick the optimal placement of the main North-South line, which minimizes this sum of secondary distances. Given a list of n houses, along with the (x, y) coordinates for each house, explain an efficient method to find the optimal placement of the main line. Analyze the running time of your solution.
[Hint: There is a quicker solution than you may think.]

Problem 5.

[9 points] (a) A binary tree is such that it produces,
$A, L, G, O, R, I, T, H, M, S$ with an inorder walk, while it produces,
$L, R, O, G, A, H, T, S, M, I$ with a postorder walk.
(note: this is not necessarily a search tree)
What does a preorder visit of this tree produce?

[8 points] (b) Insert the letters R, E, D, B, L, A, C, K into a 2-3-4 tree, in the order given. Draw the final tree.

[8 points] (c) Now draw the final tree above as a Red-Black tree. Make sure you label the color of the nodes in your diagram.

Problem 6.

Assume that we have a fair 6-sided die. We are interested in the expected number of times we will need to roll the die to see all six numbers appear.

Let X_i be the random variable signifying the number of rolls we need to finish our game, assuming that there are still i numbers which we have not yet seen. Therefore, $X_0 = 0$ by definition, and we are eventually interested in $\mathrm{E}[X_6]$.

[15 points] (a) Give a probabilistic recurrence for $\mathrm{E}[X_i]$, which will allow us to calculate the expected values for various i.

[10 points] (b) What is $\mathrm{E}[X_6]$?

Problem 7.

[12 points] (a) We are building a hash table with 11 slots, using open addressing with linear probing. Draw the hash table that results when inserting the sequence, $\langle 19, 6, 16, 8, 5, 13, 1 \rangle$, while using the hash function $h(x) = 3x + 4 \pmod{11}$.

[13 points] (b) Draw the skip lists which results from inserting $\langle 19, 6, 16, 8, 5, 13, 1 \rangle$. When inserting an element, assume that *Heads* results in raising the level of the element, and that our coin outputs: $(H, T, H, T, T, T, H, T, H, H, H, T, H, H, T)$.

[25 points] **Problem 8.**

Well, it turns out that having one main power line up-and-down the entire West Coast wasn't such a good idea (see Problem 4.) It seems the power went out because some section of the wire was broken by a tree branch, and now PG&E has to send out workers to check every inch of the wires which connect the n houses to the main line.

There are n houses whose respective distances from the main line are $d_1 d_2, \ldots, d_n$. The time to check all of the lines would be $\sum_{i=1}^{n} d_i$, however PG&E has k workers which it can send out separately to check different sets of houses. Now they will have to wait for all k workers to report back, and so the wait time is equal to the maximum time assigned to one of the k workers. The question is how to break up the houses into k groups to minimize the time.

As an example, if the houses have costs $\langle 8, 12, 3, 7, 2, 10 \rangle$, and $k = 3$, then the optimal breakup is $\langle 8 \rangle$, $\langle 12, 3 \rangle$, and $\langle 7, 2, 10 \rangle$. This solution has cost 19. (Notice, you are not allowed to rearrange the order of the houses.)

Your task is to use dynamic programming to design an algorithm which minimizes the cost. You do not need to re-explain the principal of dynamic programming. We are looking for three things:

- A clear definition for what the subproblems are
- A recursive formula which your dynamic program will use to solve the problem based on the subproblems
- An anaylsis of how long your dynamic program will take overall, using your recursive formula. You should analyze the running time of your algorithm in terms of both n and k.

You should give some basic explanation/justification for all parts of your solution.

[25 points] **Problem 9.**

Your department of the company has had quite a lot of internal bickering, and so a decision was made to break the department into two separate ones. Rather than divide people based on their interests, they have decided to try to fill each of the new departments with compatible people. You are in charge of deciding who ends up in which department. There are a total of n employees, each of which has given you a list of other people in the department who they refuse to work with.

Give an algorithm which decides whether or not it is possible to create two compatible departments, and if so, labels each person in either the *Red* department or the *Blue* department, so that no one will have to work with someone on their list.

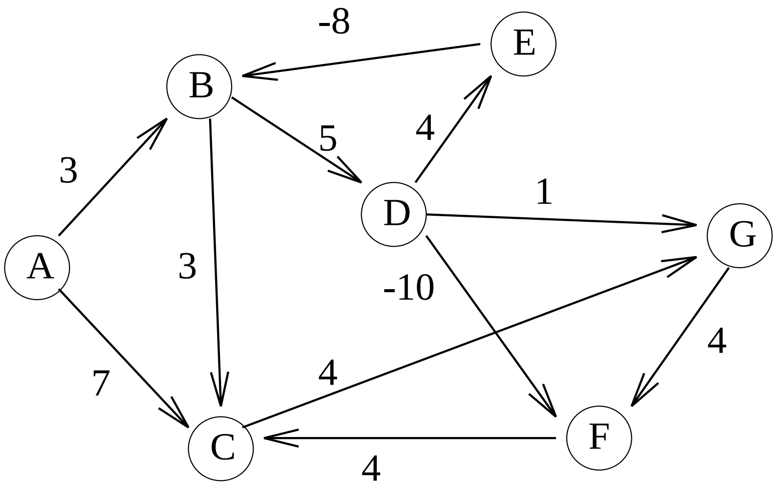
Analyze your algorithm in terms of both n, the number of people, as well as L, the total combined length of everyone's refusal list.

Problem 10.

[9 points] (a) Prove that if all of the edge weights are unique for some graph G, then there exists a unique minimum spanning tree.

[8 points] (b) Give a counterexample to the claim that a graph with unique edge weights has a unique shortest path tree.

[8 points] (c) Give the shortest path distances from A to each node of the following graph:



A	B	C	D	E	F	G