CSC 373 Midterm, Winter 2010

Name: _____

Student Id: _____

Please put your name on each page of this midterm.

Advice: The midterm has three questions, together worth 50 points. We expect you to spend as many minutes on a problem as it is worth points. You should read through the whole midterm, and start with the problem you find easiest. You will get more credit for doing one problem well then for doing two poorly. If you cannot answer a question, you will receive 10% of the marks for that question if you leave it blank. Good luck.

Name:

Problem 1 (20 points): Under new regulations all cars must pass an emissions test before any oil change. Your auto-repair shop has an unlimited supply of mechanics who can change the oil, and a single machine that can check the emissions. For every car i, you are given e_i, the amount of time it needs for the emissions test and o_i, the amount of time it takes to change the oil. All of the cars are brought in the morning, and you need to come up with a schedule for the emissions tests that will allow you to close the shop as early as possible in the evening. Assume that all cars will pass the emissions test, and as many oil changes as needed can take place simultaneously. For each of the strategies below either prove that they are optimal, or show a counter-example. You will get 10 points for the correct proof, and 5 points for each correct counter-example. The proof can use the exchange argument.

a) Greedily schedule cars in ascending order (smallest to largest) of emission test time.

No, consider (emission, oil) pairs (5, 1), (10, 3). In this order you are done after 13 (5+10+3) minutes. By switching the order you are done in (10+5+1)=11 minutes.

Name:

b) Greedily schedule cars in descending order (largest to smallest) of oil change time

This is optimal. Consider an optimal schedule and the one created by the algorithm. If the optimal schedule has a pair that is out of order with regard to oil change time (a smaller oil time before a larger), we can exchange the two without increasing the overall finish time: the larger oil change would be started earlier, so it will be finished earlier. The smaller oil job will be started at the exact same time as the larger one was in the previous ordering, so it will be finished earlier, and hence will not lead to a later overall closing time.

We can use this argument to prove the optimality by induction.

c) Greedily schedule cars in descending order of total time (emission test and oil change).

No, consider (emission, oil) pairs (50, 1), (10, 3). In this order you are done after 63 (50+10+3) minutes. By switching the order you are done in (10+50+1)=11 minutes.

Name:

Problem 2 (20 points): You are given a rectangular matrix, A[NxN]. Every row and every column of the matrix are in ascending order (so, for example, the smallest element in the matrix will be at A[1,1], and the largest at A[N,N]. You are asked to determine if a certain element x is present in the matrix. You decide to use a divide and conquer approach: you will compare the center element of the array (A[N/2,N/2]) to x, and depending on the result stop considering some fraction of the matrix.

(a, 10 points) Write down the pseudo-code for this algorithm (Hint: you will need multiple recursive calls).

```
Find(A,x) {
   if (A[n/2][n/2] == x)  return true;
   if (n == 1) return false;
   y = Find(A[1…n/2, n/2…n],x); // note that bounds are inclusive
   z = Find(A[n/2…n, 1…n/2],x);
   if (A[n/2][n/2] > x) return y || z || Find(A[1…n/2, 1…n/2],x);
   if (A[n/2][n/2] > x) return y || z || Find(A[n/2…n, n/2…n],x);
}
```

(b, 7 points) Write the recurrence describing the running time of your algorithm, and solve it to demonstrate the overall running time. You may find it useful that $\log_4(3) \approx 0.79$. Similarly, it may be helpful to introduce the variable $k=N^2$, the area of the matrix, though your final solution should be in terms of N.

let $k = N^2$ be the area of the matrix under consideration.
The recurrence relationship is

$$T(k) = 3T(k/4) + O(1)$$

Which, by Master theorem, is $O(k^{\log_4(3)}) = O(k^{0.79}) = O(N^{1.58})$

Name:

(c, 3 points) Your friend suggests an alternative algorithm: just run binary search for x in each row of the matrix, as those are sorted. What is the running time of your friend's algorithm, and is it faster or slower than the one you developed above?

Your friend's is faster, as log n is faster than n^k for all (positive) k.

Problem 3 (10 points): Consider the Huffman Code for a set of symbols $f_1, f_2, \ldots, f_k$. What is the longest that an encoding for any symbol can be? Construct an example of frequencies of the symbols that will lead to some character having an encoding of this length. (For full credit these should be for an arbitrary k, though partial credit will be given for an example with fixed k>5).

The length is k-1. This will happen if the "tree" is basically a line:

(a (b (c (d (… (j k)))…)).

An example of such a set of frequencies is:

$f_1 = 2^{-1}$

$f_2 = 2^{-2}$

…

$f_{k-1} = 2^{-(k-1)}$

$f_k = 2^{-(k-1)}$