COMPSCI 320SC 2009 Midterm Test

Attempt *all* questions. (Use of calculators is NOT permitted.)

Put the answers in the space below the questions. Write clearly and *show all your work*!

Marks for each question are shown below and just before each answer area.

This 50 minute test is worth 10% of your final grade for the course.

Question #:	1	2	3	Total
Possible marks:	8	10	12	30
Awarded marks:				

University ID: _____

Student Name: _____

Student Signature: _____

Time Finished: _____

1. You have been selected to appear on a game show. The puzzle the presenter gives you is as follows: he writes down a big number of n decimal digits (not starting with zero) and you are asked to erase $k < n$ digits. Let m be the remaining digits. These digits, when juxtaposed together, forms an integer. The amount of money you win is $\lfloor\sqrt{m}\rfloor$ dollars!

   For example, if the presenter writes down these $n = 7$ digits '1231237' and you are told to delete $k = 3$ of them, then the you could deposit $\lfloor\sqrt{3237}\rfloor = 56$ dollars (but no more) in the bank.

   Of course, there are several different amounts one can win depending on which k digits one deletes. You are greedy and you want to win as much money as possible.

   (a) Show that the greedy algorithm of deleting the k smallest digits (leftmost first, if ties occur) from the presenter's string of n digits doesn't always give the largest take-home prize.

   (***5 marks***)

   (b) Describe a greedy algorithm that is guaranteed to yield the largest prize. Briefly justify your answer. (***3 marks***)

2. Consider the following divide-and-conquer "algorithm":

```
function printer(int n)
    for i = 1 to n do
        for j = 1 to i do
            printline "hello world"
    if n > 0 then
        for k = 1 to 4 do
            printer(⌊n/2⌋)
```

Let $T(n)$ denote the number of lines of output generated by a call of printer(n).

(a) Provide a recurrence equation for $T(n)$. (***5 marks***)

(b) Solve the recurrence asymptotically for general n.

You may want to make use of the following 'master recurrence theorem':
Assume $t(n) = a \cdot t(n/b) + \Theta(n^c)$ is the total time for a divide-and-conquer algorithm then:

$$t(n) \in \begin{cases} \Theta(n^c) & \text{if } a < b^c \\ \Theta(n^c \log n) & \text{if } a = b^c \\ \Theta(n^{\log_b a}) & \text{if } a > b^c \end{cases}$$

(***5 marks***)

3. Consider the following recurrence

$$p(m,n) = \begin{cases} 0 & \text{if } m < n \\ 1 & \text{if } m = n \\ p(m-n,n) & \text{if } m > n \end{cases}$$

(a) Complete the following table of values for $p(m,n)$. The first three rows (for $m = 1, 2, 3$) have already been filled in for you.

$p(m,n) =$

m\n	**1**	**2**	**3**	**4**	**5**	**6**
1	1	0	0	0	0	0
2	1	1	0	0	0	0
3	1	0	1	0	0	0
4						
5						
6						

(***3 marks***)

(b) Let $Q(m) = \sum_{1<i<m} p(m,i)$. What are the values of $Q(4)$, $Q(5)$, and $Q(6)$? (***2 marks***)

(c) If you used dynamic programming to compute the value of $Q(m)$, would your algorithm run in $O(m^2)$ time? Would your dynamic program run in $O(m)$ time? Briefly explain your reasoning. You should *not* assume that $p(m,n)$ has been precomputed; instead, the input to your algorithm is a positive integer m, and its output is the value of $Q(m)$. You *should* assume that elementary arithmetic operations (i.e. adding two integers) can be done in constant time. Partial credit will be awarded if your answer uses the word "memoize", in a way that makes it clear that you know what this word means.

(***5 marks***)

(d) Can you see a way to compute the value of $Q(100)$ without computing the $p()$ array? If so, what is the value of $Q(100)$ and how did you compute it?

(***2 marks***)