AA

Comprehensive Exam: Algorithms and Concrete Mathematics Autumn 2000

This is a one hour closed-book exam and the point total for all questions is 60.
All of the intended answers may be written within the space provided. If necessary, you may use the back of the preceding page for additional scratch work. If you to use the back side of a page to write part of your answer, be sure to mark your answer clearly.
The following is a statement of the Stanford University Honor Code:

A. The Honor Code is an undertaking of the students, individually and collectively:

(1) that they will not give or receive aid in examinations; that they will not give or receive unpermitted aid in class work, in the preparation of reports, or in any other work that is to be used by the instructor as basis of grading;

(2) that they will do their shar part in seeing to it that others as well as themselv the Honor Code.

B. The faculty on its pa onor of its students by refraining from proctoring exam and unreasonable precautions to prevent the forms of d aculty will also avoid, as far as practicable, academic p to violate the Honor Code.

C. While the faculty alone set academic requirements, the students and faculty wi establish optimal conditions for honorable academic work.

By writing my "magic number" below, I certify that I acknowledge and accept the Honor Code.

____
(Number)

Prob	# 1	# 2	# 3	# 4	# 5	Total
Score						
Max	10	20	10	10	10	60

1. *(10 points)* *Big-Oh notation, Running times*
For each of the following functions, circle the best upper bound from among the choices given. You only need to circle one answer in each case. For example, if you say that a function is $O(n)$, you do not need to also say that it is also $O(n^2)$, $O(n \log n)$, and so on.

   (a) $f(n) = \sqrt{n}$
   Choose: $O(n)$ $O(n^2)$ $O(n^3)$ $O(\log n)$ $O(n \log n)$ $O(2^n)$ none

   (b) $f(n) = (\sqrt{n}+1)(\sqrt{n}-1)\sqrt{n}$
   Choose: $O(n)$ $O(n^2)$ $O(n^3)$ $O(\log n)$ $O(n \log n)$ $O(2^n)$ none

   (c) $f(n) = 422n^3 + 5n^2 \log n + 121231234$
   Choose: $O(n)$ $O(n^2)$ $O(n^3)$ $O(\log n)$ $O(n \log n)$ $O(2^n)$ none

   (d) $f(n) = n^n$
   Choose: $O(n)$ $O(n^2)$ $O(n^3)$ $O(\log n)$ $O(n \log n)$ $O(2^n)$ none

For each of the following programs (or fragments thereof) give a good upper bound on the running time of the algorithm using "big-Oh" notation, as a function of the value of n.

   (e) *(2 points)*

```
for $i$ from 1 to $n$ do
    for $j$ from $i$ to $n$ do
        ( something that is $O(\log n)$ )
```

   (f) *(2 points)*

```
/* print $n$ in binary if $n > 1$; mod runs in constant time */
bprint($n$)
    if $n > 0$ then
        bprint($\lfloor n/2 \rfloor$);
        print($n$ mod 2);
```

2. (*20 points*) *Recurrences.*
Solve each of these three recurrences. You may give an exact solution, or give a good upper-bound using big-oh notation.

a) If $f(n) = f(n/2) + \log n$, with $f(1) = 0$, then find a closed form expression for $f(n)$.

b) If $g(n) = 2g(n/2) + \sqrt{(n)}$, with $g(0) = 0$ and $g(1) = 1$, then find a closed form expression for $g(n)$.

c) If $h(n) = h(n/2) + h(n/4)$, with $h(0) = h(1) = 1$, then find a closed-form expression for $h(n)$.

3. (*10 points*) *Matching.*

One day, upon returning from the laundromat, I realize that, as usual, I've lost a few socks. In fact, no two of my socks are an exact match! Luckily, though, some of them are close and I have a similarity scoring function: any two socks i and j have an associated score $0 \leq s_{ij} \leq 1$ which is large if they are very similar and small if they are very different. Naturally, $s_{ij} = s_{ji}$ for all i and j. I have an even number of socks and decide to try to put my socks together two-by-two so as to maximize the sum of scores.

I decide to use the following greedy algorithm:

```
Repeat until all socks have mates:
    • Pick a random unmated sock $i$.
    • Among the other unmated socks, find the sock $j$ so that $s_{ij}$ is maximum
      (i.e., no other unmated sock $k$ has $s_{ik} > s_{ij}$).
```

Show that this method may produce a poor matching. More specifically, the *score* for a matching is the sum of the similarty scores for the pairs I pick. Show that for any $0 < \delta < 1$, there is a set of socks, and similarity scoring function as described above, so that if I pick socks in a particular order I will get a score less than δ times the score for the best overall matching of this set of socks.

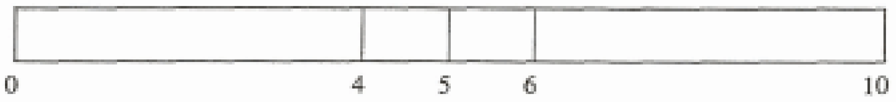
4. (*10 points*) *Amortized Analysis.*
A bank offers a combined savings/checking account with options to deposit \$1000 to the savings account, deposit \$1000 to checking, withdraw an integer multiple of \$1000 from the savings account, withdraw an integer multiple of \$1000 from checking, or transfer an integer multiple of \$1000 from the savings to the checking account. Each of these operations has a certain overhead cost to the bank, but actually storing the money costs them nothing. (In the event that you attempt to withdraw more money from an account than you actually have in that account, or to transfer more money from savings than you have in savings, the operations fails and costs nothing to the bank.) Below is a table of the operations and the overhead costs to the bank for each of them (assume α and β are constants):

Option	Actual Cost to the Bank
Deposit \$1000 to Savings	α
Deposit \$1000 to Checking	α
Withdraw $k \times$ \$1000 from savings	$\alpha + k\beta$
Withdraw $k \times$ \$1000 from checking	$\alpha + k\beta$
Transfer $k \times$ \$1000 from savings to checking	$\alpha + k\beta$

(a) Say you open an account today with a balance of \$0. Show that the bank can offset the overhead costs from your first n operations by simply charging an $O(1)$ service fee every time you deposit to savings or checking. That is, give an amount the bank could charge so that, regardless of your first n operations, the overhead would not exceed the total service charge, and show why this fee would offset the overhead costs.

(b) Now say the bank adds a sixth option: "transfer an integer multiple of \$1000 from *checking* to *savings*," and assume that the bank decides it will still only charge a service fee for deposits, and that this fee will still be constant. Show that, no matter what the service fee is, you could open a new account (also with an initial balance of \$0) and perform a series of operations so that the bank would lose money.

(c) *(10 points)* *Dynamic Programming.*
Assume you have an l-foot log (i.e., a tree trunk, not a logarithm) that has a few marks spraypainted on it, indicating that you need to saw it at these places. For example, the log may look like the one below:



where the numbers indicate distance from the left end of the log. Assume further that the cost of making a particular cut is the length of the section in which you make the cut. For example, in the diagram, if we cut first at 4, then at 5, then at 6, the cost is $10 + 6 + 5$.

Give an efficient, dynamic-programming algorithm for deciding the cheapest cut order. More precisely, assume you are given a list $L = (l_1, l_2, ..., l_k)$ of cuts, where the ith cut l_i is given as the distance between the left end of the log and the place at which that cut is to be made. (The above log would have the list $L = (4, 5, 6)$.) Your algorithm should take such a list and output an ordered list L' that gives the cuts in the cheapest order.

Also, give a big-Oh time bound for your algorithm.