

CS 573: Graduate Algorithms, Fall 2008

Homework 0

Due in class at 12:30pm, Wednesday, September 3, 2008

Name:	
Net ID:	Alias:

I understand the course policies.

-
- Each student must submit their own solutions for this homework. For all future homeworks, groups of up to three students may submit a single, common solution.
 - Neatly print your full name, your NetID, and an alias of your choice in the boxes above, and staple this page to the front of your homework solutions. We will list homework and exam grades on the course web site by alias.

Federal privacy law and university policy forbid us from publishing your grades, even anonymously, without your explicit written permission. **By providing an alias, you grant us permission to list your grades on the course web site. If you do not provide an alias, your grades will not be listed.** For privacy reasons, your alias should not resemble your name, your NetID, your university ID number, or (God forbid) your Social Security number.

- Please carefully read the course policies linked from the course web site. If you have *any* questions, please ask during lecture or office hours, or post your question to the course newsgroup. Once you understand the policies, please check the box at the top of this page. In particular:
 - You may use any source at your disposal—paper, electronic, or human—but you **must** write your solutions in your own words, and you **must** cite every source that you use.
 - Unless explicitly stated otherwise, **every** homework problem requires a proof.
 - Answering “I don’t know” to any homework or exam problem is worth 25% partial credit.
 - Algorithms or proofs containing phrases like “and so on” or “repeat this for all n ”, instead of an explicit loop, recursion, or induction, will receive 0 points.
 - This homework tests your familiarity with prerequisite material—big-Oh notation, elementary algorithms and data structures, recurrences, discrete probability, graphs, and most importantly, induction—to help you identify gaps in your background knowledge. **You are responsible for filling those gaps.** The early chapters of any algorithms textbook should be sufficient review, but you may also want consult your favorite discrete mathematics and data structures textbooks. If you need help, please ask in office hours and/or on the course newsgroup.
-

1. (a) [5 pts] Solve the following recurrences. State tight asymptotic bounds for each function in the form $\Theta(f(n))$ for some recognizable function $f(n)$. Assume reasonable but nontrivial base cases. If your solution requires a particular base case, say so.

$$A(n) = 4A(n/8) + \sqrt{n}$$

$$B(n) = B(n/3) + 2B(n/4) + B(n/6) + n$$

$$C(n) = 6C(n-1) - 9C(n-2)$$

$$D(n) = \max_{n/3 < k < 2n/3} (D(k) + D(n-k) + n)$$

$$E(n) = (E(\sqrt{n}))^2 \cdot n$$

- (b) [5 pts] Sort the functions in the box from asymptotically smallest to asymptotically largest, indicating ties if there are any. **Do not turn in proofs**—just a sorted list of 16 functions—but you should do them anyway, just for practice. We use the notation $\lg n = \log_2 n$.

n	$\lg n$	\sqrt{n}	3^n
$\sqrt{\lg n}$	$\lg \sqrt{n}$	$3^{\sqrt{n}}$	$\sqrt{3^n}$
$3^{\lg n}$	$\lg(3^n)$	$3^{\lg \sqrt{n}}$	$3^{\sqrt{\lg n}}$
$\sqrt{3^{\lg n}}$	$\lg(3^{\sqrt{n}})$	$\lg \sqrt{3^n}$	$\sqrt{\lg(3^n)}$

2. Describe and analyze a data structure that stores set of n records, each with a numerical *key* and a numerical *priority*, such that the following operation can be performed quickly:

- $\text{RANGETOP}(a, z)$: return the highest-priority record whose key is between a and z .

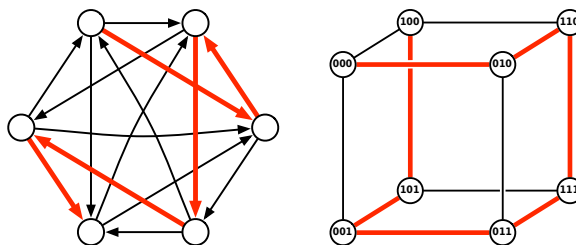
For example, if the (key, priority) pairs are

$$(3, 1), (4, 9), (9, 2), (6, 3), (5, 8), (7, 5), (1, 10), (0, 7),$$

then $\text{RANGETOP}(2, 8)$ would return the record with key 4 and priority 9 (the second in the list).

Analyze both the size of your data structure and the running time of your RANGETOP algorithm. For full credit, your space and time bounds must both be as small as possible. You may assume that no two records have equal keys or equal priorities, and that no record has a or z as its key. [Hint: How would you compute the number of keys between a and z ? How would you solve the problem if you knew that a is always $-\infty$?]

3. A *Hamiltonian path* in G is a path that visits every vertex of G exactly once. In this problem, you are asked to prove that two classes of graphs always contain a Hamiltonian path.
- (a) [5 pts] A *tournament* is a directed graph with exactly one edge between each pair of vertices. (Think of the nodes in a round-robin tournament, where edges represent games, and each edge points from the loser to the winner.) Prove that every tournament contains a *directed* Hamiltonian path.
- (b) [5 pts] Let d be an arbitrary non-negative integer. The d -dimensional *hypercube* is the graph defined as follows. There are 2^d vertices, each labeled with a different string of d bits. Two vertices are joined by an edge if and only if their labels differ in exactly one bit. Prove that the d -dimensional hypercube contains a Hamiltonian path.



Hamiltonian paths in a 6-node tournament and a 3-dimensional hypercube.

4. Penn and Teller agree to play the following game. Penn shuffles a standard deck¹ of playing cards so that every permutation is equally likely. Then Teller draws cards from the deck, one at a time without replacement, until he draws the three of clubs ($3\clubsuit$), at which point the remaining undrawn cards instantly burst into flames.

The first time Teller draws a card from the deck, he gives it to Penn. From then on, until the game ends, whenever Teller draws a card whose value is smaller than the last card he gave to Penn, he gives the new card to Penn.² To make the rules unambiguous, they agree beforehand that $A = 1$, $J = 11$, $Q = 12$, and $K = 13$.

- (a) What is the expected number of cards that Teller draws?
- (b) What is the expected *maximum* value among the cards Teller gives to Penn?
- (c) What is the expected *minimum* value among the cards Teller gives to Penn?
- (d) What is the expected number of cards that Teller gives to Penn?

Full credit will be given only for *exact* answers (with correct proofs, of course). [Hint: Let $13 = n$.]

¹In a standard deck of playing cards, each card has a *value* in the set $\{A, 2, 3, 4, 5, 6, 7, 8, 9, 10, J, Q, K\}$ and a *suit* in the set $\{\spadesuit, \heartsuit, \clubsuit, \diamondsuit\}$; each of the 52 possible suit-value pairs appears in the deck exactly once. Actually, to make the game more interesting, Penn and Teller normally use razor-sharp ninja throwing cards.

²Specifically, he hurls them from the opposite side of the stage directly into the back of Penn's right hand. Ouch!

5. (a) The **Fibonacci numbers** F_n are defined by the recurrence $F_n = F_{n-1} + F_{n-2}$, with base cases $F_0 = 0$ and $F_1 = 1$. Here are the first several Fibonacci numbers:

F_0	F_1	F_2	F_3	F_4	F_5	F_6	F_7	F_8	F_9	F_{10}
0	1	1	2	3	5	8	13	21	34	55

Prove that any non-negative integer can be written as the sum of distinct, non-consecutive Fibonacci numbers. That is, if the Fibonacci number F_i appears in the sum, it appears exactly once, and its neighbors F_{i-1} and F_{i+1} do not appear at all. For example:

$$17 = F_7 + F_4 + F_2, \quad 42 = F_9 + F_6, \quad 54 = F_9 + F_7 + F_5 + F_3 + F_1.$$

- (b) The Fibonacci sequence can be extended backward to negative indices by rearranging the defining recurrence: $F_n = F_{n+2} - F_{n+1}$. Here are the first several negative-index Fibonacci numbers:

F_{-10}	F_{-9}	F_{-8}	F_{-7}	F_{-6}	F_{-5}	F_{-4}	F_{-3}	F_{-2}	F_{-1}
-55	34	-21	13	-8	5	-3	2	-1	1

Prove that $F_{-n} = -F_n$ if and only if n is even.

- (c) Prove that *any* integer—positive, negative, or zero—can be written as the sum of distinct, non-consecutive Fibonacci numbers *with negative indices*. For example:

$$17 = F_{-7} + F_{-5} + F_{-2}, \quad -42 = F_{-10} + F_{-7}, \quad 54 = F_{-9} + F_{-7} + F_{-5} + F_{-3} + F_{-1}.$$

[Hint: Zero is both non-negative and even. Don't use weak induction!]