CS 573: Graduate Algorithms, Fall 2010
Homework 0

Due Wednesday, September 1, 2010 **in class**

- This homework tests your familiarity with prerequisite material (http://www.cs.uiuc.edu/class/fa10/cs573/stuff-you-already-know.html) to help you identify gaps in your background knowledge. **You are responsible for filling those gaps.** Fr most topics, 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.
- Each student must submit individual solutions for these homework problems. For all future homeworks, groups of up to three students may submit (or present) a single group solution for each problem.
- 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. In particular:
  - Submit five separately stapled solutions, one for each numbered problem, with your name and NetID clearly printed on each page. Please do *not* staple everything together.
  - 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. In particular, each solution should include a list of *everyone* you worked with to solve that problem.
  - Unless explicitly stated otherwise, ***every*** homework problem requires a proof.
  - Answering "I don't know" to any homework or exam problem (except for extra credit problems) is worth 25% partial credit.
  - Algorithms or proofs containing phrases like "and so on" or "repeat this process for all n" instead of an explicit loop, recursion, or induction, will receive 0 points.

---

1. (•) ***Write the sentence "I understand the course policies."***
   Solutions that omit this sentence will not be graded.

   (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 none are given. ***Do not submit proofs***—just a list of five functions—but you should do them anyway, just for practice.

   - $A(n) = 4A(n-1) + 1$
   - $B(n) = B(n-3) + n^2$
   - $C(n) = 2\,C(n/2) + 3\,C(n/3) + n^2$
   - $D(n) = 2\,D(n/3) + \sqrt{n}$
   - $E(n) = \begin{cases} n & \text{if } n \le 3, \\ \dfrac{E(n-1)E(n-2)}{E(n-3)} & \text{otherwise} \end{cases}$ *[Hint: This is easier than it looks!]*

   (b) **[5 pts]** Sort the following functions from asymptotically smallest to asymptotically largest, indicating ties if there are any. ***Do not submit proofs***—just a sorted list of 16 functions—but you should do them anyway, just for practice.

   Write $f(n) \ll g(n)$ to indicate that $f(n) = o(g(n))$, and write $f(n) \equiv g(n)$ to mean $f(n) = \Theta(g(n))$. We use the notation $\lg n = \log_2 n$.

n	$\lg n$	$\sqrt{n}$	7^n
$\sqrt{\lg n}$	$\lg \sqrt{n}$	$7^{\sqrt{n}}$	$\sqrt{7^n}$
$7^{\lg n}$	$\lg(7^n)$	$7^{\lg \sqrt{n}}$	$7\sqrt{\lg n}$
$\sqrt{7^{\lg n}}$	$\lg(7^{\sqrt{n}})$	$\lg \sqrt{7^n}$	$\sqrt{\lg(7^n)}$

2. Professore Giorgio della Giungla has a 23-node binary tree, in which every node is labeled with a unique letter of the Roman alphabet, which is just like the modern English alphabet, but without the letters **J**, **U**, and **W**. Inorder and postorder traversals of the tree visit the nodes in the following order:

   - Inorder: **S V Z A T P R D B X O L F E H I Q M N G Y K C**
   - Postorder: **A Z P T X B D L E F O H R I V N M K C Y G Q S**

   (a) List the nodes in Prof. della Giungla's tree in the order visited by a *preorder* traversal.

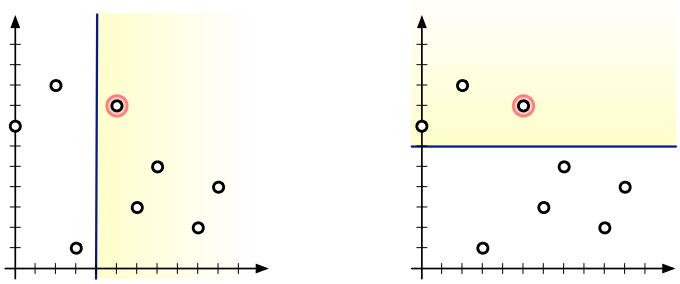
   (b) Draw Prof. della Giungla's tree.

3. The original version of this problem asked to support the mirror-image operations LOWESTTORIGHT and LEFTMOSTABOVE, which are *much* harder to support with a single data structure that stores each point at most once. We will accept $O(n)$-space data structures for either version of the problem for full credit.

   Describe a data structure that stores a set S of n points in the plane, each represented by a pair (x, y) of coordinates, and supports the following queries.

   - **HIGHESTTORIGHT(ℓ):** Return the highest point in S whose x-coordinate is greater than or equal to ℓ. If every point in S has x-coordinate less than ℓ, return NONE.
   - **RIGHTMOSTABOVE(ℓ):** Return the rightmost point in S whose y-coordinate is greater than or equal to ℓ. If every point in S has y-coordinate less than ℓ, return NONE.

   For example, if $S = \{(3,1), (1,9), (9,2), (6,3), (5,8), (7,5), (10,4), (0,7)\}$, then both HIGHESTTORIGHT(4) and RIGHTMOSTABOVE(6) should return the point $(5,8)$, and HIGHESTTORIGHT(15) should return NONE.



HIGHESTTORIGHT(4) = RIGHTMOSTABOVE(6) = (5,8)

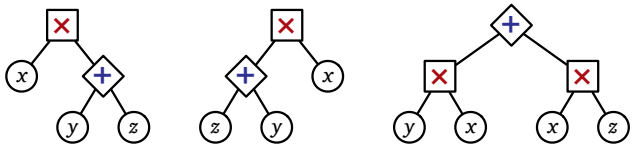
   Analyze both the size of your data structure and the running times of your query algorithms. For full credit, your data structure should use $O(n)$ space, and each query algorithm should run in $O(\log n)$ time. ***For 5 extra credit points, describe a data structure that stores each point at most once.*** You may assume that no two points in S have equal x-coordinates or equal y-coordinates.

   [Hint: Modify one of the standard data structures listed at http://www.cs.uiuc.edu/class/fa10/cs573/stuff-you-already-know.html, but just describe your changes; don't regurgitate the details of the standard data structure.]

4. An ***arithmetic expression tree*** is a binary tree where every leaf is labeled with a variable, every internal node is labeled with an arithmetic operation, and every internal node has exactly two children. For this problem, assume that the only allowed operations are $+$ and $\times$. Different leaves may or may not represent different variables.

   Every arithmetic expression tree represents a function, transforming input values for the leaf variables into an output value for the root, by following two simple rules: (1) The value of any $+$-node is the sum of the values of its children. (2) The value of any $\times$-node is the product of the values of its children.

   Two arithmetic expression trees are ***equivalent*** if they represent the same function; that is, the same input values for the leaf variables always leads to the same output value at both roots.



Three equivalent expression trees. Only the third tree is in normal form.

An arithmetic expression tree is in ***normal form*** if the parent of every +-node (if any) is another +-node.

Prove that for any arithmetic expression tree, there is an equivalent arithmetic expression tree in normal form. *[Hint: Be careful. This is trickier than it looks.]*

5. Recall that a standard (Anglo-American) deck of 52 playing cards contains 13 cards in each of four suits: spades (♠), hearts (♥), diamonds (♦), and clubs (♣). Within each suit, the 13 cards have distinct *ranks*: 2, 3, 4, 5, 6, 7, 8, 9, 10, jack (J), queen (Q), king (K), and ace (A). The ranks are ordered $2 < 3 < \cdots < 9 < 10 < J < Q < K < A$; thus, for example, the jack of spades has higher rank thank the eight of diamonds.

   Professor Jay is about to perform a public demonstration with two decks of cards, one with red backs ('the red deck') and one with blue backs ('the blue deck'). Both decks lie face-down on a table in front of Professor Jay, shuffled uniformly and independently. Thus, in each deck, every permutation of the 52 cards is equally likely.

   To begin the demonstration, Professor Jay turns over the top card from each deck. Then, while he has not yet turned over a three of clubs (3♣), the good Professor hurls the two cards he just turned over into the thick, pachydermatous outer melon layer of a nearby watermelon (that most prodigious of household fruits) and then turns over the next card from the top of each deck. Thus, if 3♣ is the last card in both decks, the demonstration ends with 102 cards embedded in the watermelon.

   (a) What is the *exact* expected number of cards that Professor Jay hurls into the watermelon?

   (b) For each of the statements below, give the *exact* probability that the statement is true of the ***first*** pair of cards Professor Jay turns over.

      i. Both cards are threes.
      ii. One card is a three, and the other card is a club.
      iii. If (at least) one card is a heart, then (at least) one card is a diamond.
      iv. The card from the red deck has higher rank than the card from the blue deck.

   (c) For each of the statements below, give the *exact* probability that the statement is true of the ***last*** pair of cards Professor Jay turns over.

      i. Both cards are threes.
      ii. One card is a three, and the other card is a club.
      iii. If (at least) one card is a heart, then (at least) one card is a diamond.
      iv. The card from the red deck has higher rank than the card from the blue deck.

   Express each of your answers as rational numbers in simplest form, like 123/4567. ***Do not submit proofs***—just a list of rational numbers—but you should do them anyway, just for practice.