CS 473: Undergraduate Algorithms, Fall 2012
Homework 0

Due Tuesday, September 4, 2012 at noon

Quiz 0 (on the course Moodle page)
is also due Tuesday, September 4, 2012 at noon.

- Homework 0 and Quiz 0 test your familiarity with prerequisite material—big-Oh notation, elementary algorithms and data structures, recurrences, graphs, and most importantly, induction—to help you identify gaps in your background knowledge. **You are responsible for filling those gaps.** The course web page has pointers to several excellent online resources for prerequisite material. If you need help, please ask in headbanging, on Piazza, in office hours, or by email.

- 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 on the course web site. If you have *any* questions, please ask in lecture, in headbanging, on Piazza, in office hours, or by email. In particular:
  - Submit separately stapled solutions, one for each numbered problem, with your name and NetID clearly printed on each page, in the corresponding drop boxes outside 1404 Siebel.
  - You may use any source at your disposal—paper, electronic, human, or other—but you ***must*** write your solutions in your own words, and you ***must*** cite every source that you use (except for official course materials). Please see the academic integrity policy for more details.
  - No late homework will be accepted for any reason. However, we may *forgive* quizzes or homeworks in extenuating circumstances; ask Jeff for details.
  - Answering "I don't know" to any (non-extra-credit) problem or subproblem, on any homework or exam, 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 a score of 0.
  - Unless explicitly stated otherwise, ***every*** homework problem requires a proof.

---

1. **[CS 173]** The ***Lucas numbers*** L_n are defined recursively as follows:

$$L_n = \begin{cases} 2 & \text{if } n = 0 \\ 1 & \text{if } n = 1 \\ L_{n-2} + L_{n-1} & \text{otherwise} \end{cases}$$

You may recognize this as the Fibonacci recurrence, but with a different base case ($L_0 = 2$ instead of $F_0 = 0$). Similarly, the ***anti-Lucas numbers*** Γ_n are defined recursively as follows:

$$\Gamma_n = \begin{cases} 1 & \text{if } n = 0 \\ 2 & \text{if } n = 1 \\ \Gamma_{n-2} - \Gamma_{n-1} & \text{otherwise} \end{cases}$$

Here are the first several Lucas and anti-Lucas numbers:

n	0	1	2	3	4	5	6	7	8	9	10	11	12
L_n	2	1	3	4	7	11	18	29	47	76	123	199	322
Γ_n	1	2	−1	3	−4	7	−11	18	−29	47	−76	123	−199

(a) Prove that $\Gamma_n = (-1)^{n-1} L_{n-1}$ for every positive integer n.

(b) Prove that any non-negative integer can be written as the sum of distinct *non-consecutive* Lucas numbers; that is, if L_i appears in the sum, then L_{i-1} and L_{i+1} cannot. For example:

$$\begin{aligned} 4 &= 4 &&= L_3 \\ 8 &= 7+1 &&= L_4 + L_1 \\ 15 &= 11+4 &&= L_5 + L_3 \\ 16 &= 11+4+1 &&= L_5 + L_3 + L_1 \\ 23 &= 18+4+1 &&= L_6 + L_3 + L_1 \\ 42 &= 29+11+2 &&= L_7 + L_5 + L_0 \end{aligned}$$

2. **[CS 173 + CS 373]** Consider the language over the alphabet $\{\spadesuit, \heartsuit, \diamondsuit, \clubsuit\}$ generated by the following context-free grammar:

$$S \to \heartsuit \mid \spadesuit S \mid S \clubsuit \mid S \diamondsuit S$$

Prove that every string in this language has the following properties:

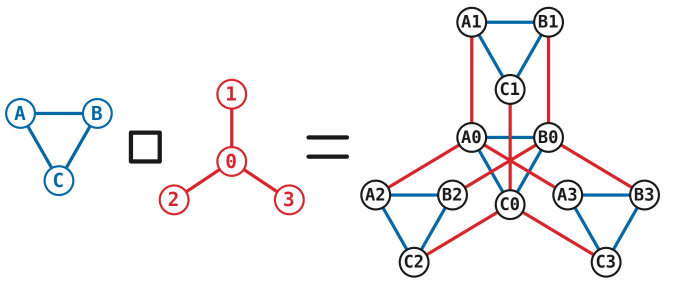
(a) The number of ♥s is exactly one more than the number of ♦s.

(b) There is a ♦ between any two ♥s.

3. **[CS 173 + mathematical maturity]** Given two undirected graphs $G = (V, E)$ and $G' = (V', E')$, we define a new graph $\boldsymbol{G \,\square\, G'}$, called the ***box product*** of G and G', as follows:

   - The vertices of $G \,\square\, G'$ are all pairs (v, v') where $v \in V$ and $v' \in V'$.
   - Two vertices (v, v') and (w, w') are connected by an edge in $G \,\square\, G'$ if and only if either $\big(v = w$ and $(v', w') \in E'\big)$ or $\big((v, w) \in E$ and $v' = w'\big)$.

   Intuitively, every pair of edges $e \in E$ and $e' \in E'$ define a "box" of four edges in $G \,\square\, G'$. For example, if G is a path of length n, then $G \,\square\, G$ is an $n \times n$ grid. Another example is shown below.



The box product of two graphs.

   (a) Let I denote the unique connected graph with two vertices. Give a concise *English* description of the following graphs. You do ***not*** have to prove that your answers are correct.

      i. What is $I \,\square\, I$?

      ii. What is $I \,\square\, I \,\square\, I$?

      iii. What is $I \,\square\, I \,\square\, I \,\square\, I$?

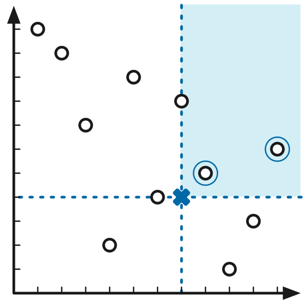
   (b) Recall that a ***Hamiltonian path*** in a graph G is a path in G that visits every vertex of G exactly once. Prove that for any graphs G and G' that both contain Hamiltonian paths, the box product $G \,\square\, G'$ also contains a Hamiltonian path. *[Hint: **Don't** use induction.]*

4. **[CS 225]** Describe and analyze a data structure that stores a set S of n points in the plane, each represented by a pair of integer coordinates, and supports queries of the following form:

   SomethingAboveRight(x, y): Return an arbitrary point (a, b) in S such that $a > x$ and $b > y$. If there is no such point in S, return None.

   For example, if S is the 11-point set $\{(1,11), (2,10), (3,7), (4,2), (5,9), (6,4), (7,8), (8,5), (9,1), (10,3), (11,6)\}$, as illustrated on the next page, then

   - SomethingAboveRight$(0, 0)$ may return any point in S;
   - SomethingAboveRight$(7, 7)$ must return None;
   - SomethingAboveRight$(7, 4)$ must return either $(8, 5)$ or $(11, 6)$.



SomethingAboveRight$(7,4)$ returns either $(8,5)$ or $(11,6)$.

A complete solution must (a) describe a data structure, (b) analyze the space it uses, (c) describe a query algorithm, (d) prove that it is correct, and (e) analyze its worst-case running time. You do *not* need to describe how to build your data structure from a given set of points. Smaller and simpler data structures with faster and simpler query algorithms are worth more points. You may assume all points in S have distinct x-coordinates and distinct y-coordinates.

*5. ***[Extra credit]*** A ***Gaussian integer*** is a complex number of the form $x + yi$, where x and y are integers. Prove that any Gaussian integer can be expressed as the sum of distinct powers of the complex number $\alpha = -1 + i$. For example:

$$\begin{aligned}
4 &= 16 + (-8-8i) + 8i + (-4) &&= \alpha^8 + \alpha^7 + \alpha^6 + \alpha^4 \\
-8 &= (-8-8i) + 8i &&= \alpha^7 + \alpha^6 \\
15i &= (-16+16i) + 16 + (-2i) + (-1+i) + 1 &&= \alpha^9 + \alpha^8 + \alpha^2 + \alpha^1 + \alpha^0 \\
1+6i &= (8i) + (-2i) + 1 &&= \alpha^6 + \alpha^2 + \alpha^0 \\
2-3i &= (4-4i) + (-4) + (2+2i) + (-2i) + (-1+i) + 1 &&= \alpha^5 + \alpha^4 + \alpha^3 + \alpha^2 + \alpha^1 + \alpha^0 \\
-4+2i &= (-16+16i) + 16 + (-8-8i) + (4-4i) + (-2i) &&= \alpha^9 + \alpha^8 + \alpha^7 + \alpha^5 + \alpha^2
\end{aligned}$$

The following list of values may be helpful:

$\alpha^0 = 1$	$\alpha^4 = -4$	$\alpha^8 = 16$	$\alpha^{12} = -64$
$\alpha^1 = -1 + i$	$\alpha^5 = 4 - 4i$	$\alpha^9 = -16 + 16i$	$\alpha^{13} = 64 - 64i$
$\alpha^2 = -2i$	$\alpha^6 = 8i$	$\alpha^{10} = -32i$	$\alpha^{14} = 128i$
$\alpha^3 = 2 + 2i$	$\alpha^7 = -8 - 8i$	$\alpha^{11} = 32 + 32i$	$\alpha^{15} = -128 - 128i$

[Hint: How do you write $-2 - i$?]