CS 473G: Graduate Algorithms, Spring 2007
Homework 0

Due **in class** at 11:00am, Tuesday, January 30, 2007

Name:	
Net ID:	Alias:

☐ I understand the Course Policies.

- Neatly print your full name, your NetID, and an alias of your choice in the boxes above, and staple this page to your solution to problem 1. We will list homework and exam grades on the course web site by alias. **By providing an alias, you agree to let us list your grades; 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 use the same alias for every homework and exam.

- Read the Course Policies on the course web site, and then check the box above. Among other things, this page describes what we expect in your homework solutions, as well as policies on grading standards, regrading, extra credit, and plagiarism. In particular:

  - Submit each numbered problem separately, on its own piece(s) of paper. If you need more than one page for a problem, staple just *those* pages together, but keep different problems separate. **Do not staple your entire homework together.**
  - You may use *any* source at your disposal—paper, electronic, or human—but you *must* write your answers in your own words, and you *must* cite every source that you use.
  - Algorithms or proofs containing phrases like "and so on" or "repeat this for all n", instead of an explicit loop, recursion, or induction, are worth zero points.
  - Answering "I don't know" to any homework or exam problem is worth 25% partial credit.

  If you have *any* questions, please ask during lecture or office hours, or post your question to the course newsgroup.

- 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 knowledge. **You are responsible for filling those gaps on your own.** The early chapters of Kleinberg and Tardos (or any algorithms textbook) should be sufficient review, but you may also want consult your favorite discrete mathematics and data structures textbooks.

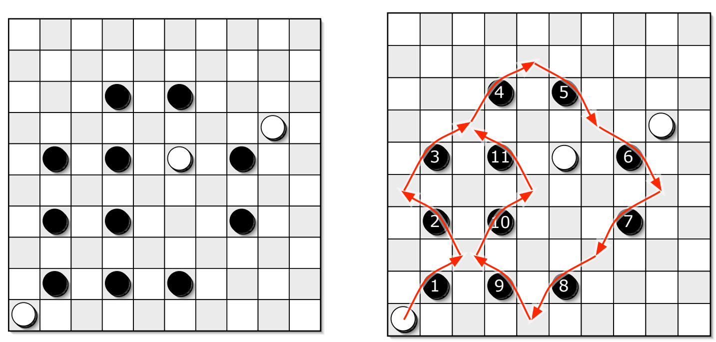
- Every homework will have five problems, each worth 10 points. Stars indicate more challenging problems. Many homeworks will also include an extra-credit problem.

---

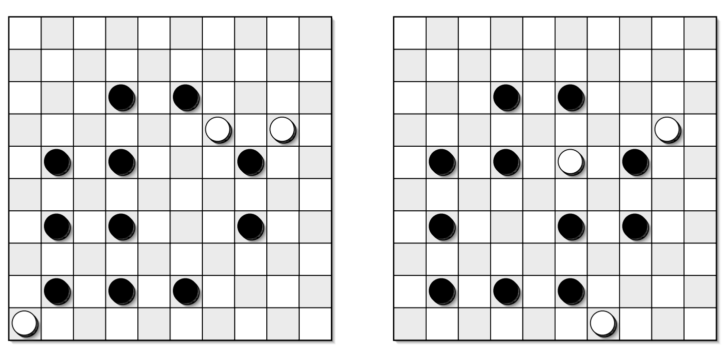
*1. Draughts/checkers is a game played on an $m \times m$ grid of squares, alternately colored light and dark. (The game is usually played on an 8×8 or 10×10 board, but the rules easily generalize to any board size.) Each dark square is occupied by at most one game piece (usually called a *checker* in the U.S.), which is either black or white; light squares are always empty. One player ("White") moves the white pieces; the other ("Black") moves the black pieces.

Consider the following simple version of the game, essentially American checkers or British draughts, but where every piece is a king.[1] Pieces can be moved in any of the four diagonal directions, either one or two steps at a time. On each turn, a player either *moves* one of her pieces one step diagonally into an empty square, or makes a series of *jumps* with one of her checkers. In a single jump, a piece moves to an empty square two steps away in any diagonal direction, but only if the intermediate square is occupied by a piece of the opposite color; this enemy piece is *captured* and immediately removed from the board. Multiple jumps are allowed in a single turn as long as they are made by the same piece. A player wins if her opponent has no pieces left on the board.

Describe an algorithm[2] that correctly determines whether White can capture every black piece, thereby winning the game, *in a single turn*. The input consists of the width of the board (m), a list of positions of white pieces, and a list of positions of black pieces. For full credit, your algorithm should run in $O(n)$ time, where n is the total number of pieces, but any algorithm that runs in time polynomial in n and m is worth significant partial credit.



White wins in one turn.



White cannot win in one turn from either of these positions.

*[Hint: The greedy strategy—make arbitrary jumps until you get stuck—does **not** always find a winning sequence of jumps even when one exists.]*

[1]Most variants of draughts have 'flying kings', which behave *very* differently than what's described here.
[2]Since you've read the Course Policies, you know what this phrase means.

2. (a) Prove that any positive integer can be written as the sum of distinct powers of 2. *[Hint: "Write the number in binary" is **not** a proof; it just restates the problem.]* For example:

$$\begin{aligned} 16+1 = \quad & 17 & = 2^4 + 2^0 \\ 16+4+2+1 = \quad & 23 & = 2^4+2^2+2^1+2^0 \\ 32+8+1 = \quad & 42 & = 2^5+2^3+2^1 \end{aligned}$$

(b) Prove that *any* integer (positive, negative, or zero) can be written as the sum of distinct powers of -2. For example:

$$\begin{aligned} -32+16-2+1 = \quad & -17 & = (-2)^5+(-2)^4+(-2)^1+(-2)^0 \\ 64-32-8-2+1 = \quad & 23 & = (-2)^6+(-2)^5+(-2)^3+(-2)^1+(-2)^0 \\ 64-32+16-8+4-2 = \quad & 42 & = (-2)^6+(-2)^5+(-2)^4+(-2)^3+(-2)^2+(-2)^1 \end{aligned}$$

3. Whenever groups of pigeons gather, they instinctively establish a *pecking order*. For any pair of pigeons, one pigeon always pecks the other, driving it away from food or potential mates. The same pair of pigeons always chooses the same pecking order, even after years of separation, no matter what other pigeons are around. Surprisingly, the overall pecking order can contain cycles—for example, pigeon A pecks pigeon B, which pecks pigeon C, which pecks pigeon A.

   Prove that any finite set of pigeons can be arranged in a row from left to right so that every pigeon pecks the pigeon immediately to its left.

4. On their long journey from Denmark to England, Rosencrantz and Guildenstern amuse themselves by playing the following game with a fair coin. First Rosencrantz flips the coin over and over until it comes up tails. Then Guildenstern flips the coin over and over until he gets as many heads in a row as Rosencrantz got on his turn. Here are three typical games:

   Rosencrantz: $\underline{\mathsf{H\ H}}$ T
   Guildenstern: H T $\underline{\mathsf{H\ H}}$

   Rosencrantz: T
   Guildenstern: (no flips)

   Rosencrantz: $\underline{\mathsf{H\ H\ H}}$ T
   Guildenstern: T H H T H H T H T T $\underline{\mathsf{H\ H\ H}}$

   (a) What is the expected number of flips in one of Rosencrantz's turns?

   (b) Suppose Rosencrantz flips k heads in a row on his turn. What is the expected number of flips in Guildenstern's next turn?

   (c) What is the expected total number of flips (by both Rosencrantz and Guildenstern) in a single game?

Prove that your answers are correct. If you have to appeal to "intuition" or "common sense", your answer is almost certainly wrong! You must give *exact* answers for full credit, but a correct asymptotic bound for part (b) is worth significant credit.

5. (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.

$$A(n) = 3A(n/9) + \sqrt{n}$$

$$B(n) = 4B(n-1) - 4B(n-2)$$

$$C(n) = \frac{\pi\, C(n-1)}{\sqrt{2}\, C(n-2)} \qquad \textit{[Hint: This is easy!]}$$

$$D(n) = \max_{n/4<k<3n/4} \big(D(k) + D(n-k) + n\big)$$

$$E(n) = 2E(n/2) + 4E(n/3) + 2E(n/6) + n^2$$

Do *not* turn in proofs—just a list of five functions—but you should do them anyway, just for practice. *[Hint: On the course web page, you can find a handout describing several techniques for solving recurrences.]*

(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.

To simplify your answer, write $f(n) \ll g(n)$ to indicate that $f(n) = o(g(n))$, and write $f(n) \equiv g(n)$ to mean $f(n) = \Theta(g(n))$. For example, the functions $n^2, n, \binom{n}{2}, n^3$ could be sorted either as $n \ll n^2 \equiv \binom{n}{2} \ll n^3$ or as $n \ll \binom{n}{2} \equiv n^2 \ll n^3$.

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

Recall that $\lg n = \log_2 n$.