Announcements

• Mid-term exam is scheduled on Thurs, Mar 11\textsuperscript{th}. In class, closed-book.
• Review \textbf{quizzes, assignments, and slides}.
• Assignment solutions are posted in SPARK.
• If you have a schedule conflict, you must request make-up exam \textbf{ahead of time}. No request is accepted during or after the exam.
Announcements

• Big-O notation
• Binary search
• Simple sorting algorithms
• Stacks and Queues
• Linked list insertion, deletion, search
• Recursion
CS 187: Programming with Data Structures (Spring 2010)

Lecture 13: Recursion 2

Rui Wang
The Towers of Hanoi

- A puzzle consisting of a number of disks placed on **three columns (towers)**: A, B, and C.
The Towers of Hanoi

• The disks are of different sizes.
• The disks have holes to fit on each column.
• **Objective**: transfer all disks from A to C.
• **Rules:**
  1. Only one disk can be moved at a time;
  2. No disk can be placed on a disk smaller than it. (in other words, the disks on the same column must be sorted at all times)
The Towers of Hanoi

• The disks are of different sizes.
• The disks have holes to fit on each column.
• **Objective**: transfer all disks from A to C.
• **Rules**:
  1. Only one disk can be moved at a time;
  2. No disk can be placed on a disk smaller than it. (in other words, the disks on the same column must be sorted at all times)
• **The way disks get moved is similar to a stack.**
The Towers of Hanoi

• How do we move the disks to achieve the goal?
• Our job is to write a program to solve it.
• We also want to know in general, how many steps it takes to move N disks.
The Towers of Hanoi

• Let’s play with it to get some intuition:
  – N = 1
  – N = 2
  – N = 3
  – N = 4


• Workshop applet
The Towers of Hanoi

• Let’s call the initial pyramid-shaped arrangements of disks on column A a **tree**.
• This has nothing to do with the ‘tree’ data structure, which we will learn later.
The Towers of Hanoi

- Let’s call the initial pyramid-shaped arrangements of disks on column A a tree.
- This has nothing to do with the ‘tree’ data structure, which we will learn later.
- We call a smaller set of the disks a subtree.
- It turns out that the intermediate steps in the solution involves moving a subtree.
The Towers of Hanoi

• Idea:
• Assume, for now, that you have a (magical) way of moving a subtree from A to B via C;
• Then you move A to C;
• Finally move the subtree from B to C via A.
• This is similar to the 2-disk case, except the top disk is now a subtree.
• This can be done recursively.
The Towers of Hanoi

• Suppose you want to move all disks from a source tower $S$ to a destination tower $D$, via an intermediate tower $I$.

• Initially
  – Source tower is A
  – Destination tower is C
  – Intermediate tower is B
The Towers of Hanoi

1. Move the subtree consisting of the top n-1 disks from S to I;
2. Move the remaining (largest) disk from S to D;
3. Move the subtree from I to D.
Recursion

• So how exactly do I move a subtree!?
• Don’t worry, this is where the \textit{recursion} comes in handy.

• This is because moving a subtree is no different from moving the original tree, except you have 1 less disk! (solve a smaller problem).

• What’s the base case?
```java
class TowersApp {
    static int nDisks = 3;

    public static void main(String[] args) {
        doTowers(nDisks, 'A', 'B', 'C');
    }

    public static void doTowers(int topN, char from, char inter, char to) {
        if (topN == 1)
            System.out.println("Disk 1 from " + from + " to " + to);
        else
            { doTowers(topN - 1, from, to, inter); // from-->inter
              System.out.println("Disk "+ topN + " from " + from + " to " + to);
              doTowers(topN - 1, inter, from, to); // inter-->to
            }
    }
}
```
The Tower of Hanoi

• How many steps does the solution take to solve a N-disk problem?
  • N=1: 1 step
  • N=2: 3 steps
  • N=3: 7 steps
  • What’s your guess?
The Tower of Hanoi

• How many steps does the solution take to solve a N-disk problem?

• Use recursive formula
  \[ T(N) = T(N-1) + 1 + T(N-1) = 2T(N-1) + 1 \]

• So how to solve this?
The Tower of Hanoi

• How many steps does the solution take to solve a N-disk problem?
• Use recursive formula
  \[ T(N) = T(N-1) + 1 + T(N-1) = 2T(N-1) + 1 \]
• So how to solve this?
The Tower of Hanoi

• How many steps does the solution take to solve a N-disk problem?
• Use recursive formula
  \[ T(N) = T(N-1) + 1 + T(N-1) = 2 \times T(N-1) + 1 \]
• So how to solve this?

\[
T(N) = 2^{N-1} + 2^{N-2} + 2^{N-3} + \ldots + 1 = 2^N - 1
\]

• So it takes an exponential number of steps!