CS 152
Computer Programming Fundamentals
Towers of Hanoi

Brooke Chenoweth

University of New Mexico

Spring 2024

Towers of Hanoi

- Puzzle invented by the French mathematician Èdouard Lucas in 1883.
- Stack of discs start on one of 3 rods. The objective of the puzzle is to move the entire stack to another rod, obeying the following simple rules:
  - Only one disk can be moved at a time.
  - Each move consists of taking the upper disk from one of the stacks and placing it on top of another stack i.e. a disk can only be moved if it is the uppermost disk on a stack.
  - No disk may be placed on top of a smaller disk.
- `http://www.mathsisfun.com/games/towerofhanoi.html`

Questions

- How many moves does it take to solve the puzzle for different numbers of disks?
- Which specific disks do you need to move to solve the puzzle?
- Can you write a program to generate the list of moves?

Solving the puzzle

- To move a single disc from the source rod to the destination rod, just move the disc.
- To move a stack of discs from the source to the destination:
  1. Move all but one disc from the source to the helper rod.
  2. Move the single disc from source to destination.
  3. Move the discs from the helper to the destination.

Code – `move`

We'll just print out each move using the following method. One could imagine replacing this method with something that operates a robot arm if you wanted to solve the problem using physical discs.

```
  public static void move(char from, char to) {
    System.out.println(from + " -> " + to);
  }
```

Code – starting off

To solve the problem for n discs, we want to move from rod A to rod C using rod B as the helper.

```
  public static void hanoi(int n) {
    hanoi(n, ’A’, ’C’, ’B’);
  }
```

Code – recursive method

```
public static void hanoi(int n, char from,
                         char to, char help) {
  if(n == 1) {
    move(from, to);
  } else {
    hanoi(n-1, from, help, to);
    move(from, to);
    hanoi(n-1, help, to, from);
  }
}
```

Explore the code

- Try running the code for different problem sizes and see what happens.
- Add output to the beginning and end of the recursive method to see which arguments are passed to the method.
- Try running the code in the debugger, add breakpoints, and explore the stack trace.

Challenge

- Instead of printing out the moves, I would like to put them all in an array.
- How would I do this?