Recursion

CSSE 221

Fundamentals of Software Development Honors

Rose-Hulman Institute of Technology

ROSE-HULMAN INSTITUTE OF TECHNOLOGY

Announcements

- Exam Wednesday, Oct 23 & Friday, Oct 25
  - Sample exam is posted on Moodle

- We think Markov Chains are used in the SwiftKey smartphone app.
  - http://www.swiftkey.net/

If you don't have a base case for your recursion, it can become a nightmare!

Recursion

- What is a recursive method?
  - **A method that calls itself, but on a simpler problem**
- Used for any situation where parts of a whole look like mini versions of the whole:
  - Folders within folders on computers
  - Some computer languages (Scheme)
  - Trees in general
- Cons: Takes more space (but time can be roughly equal; it depends)
- Pros: Can gives code that's very easy to understand

Recursion template

- For a method that calculates a value:

```
int foo(int n) {
    if (n <=1) {     //Base case
        return (some easy expression);
    } else {
        return (some expr. with foo(n-1);
        //not just foo(n)) so progress
    }
```

Of course, you can write void recursive methods, and ones that recurse on values other than n-1

Four Rules of Recursion

1. **Base case**
   - You need at least 1 base case that can be solved without recursing
2. **Progress**
   - You can only recurse on a simpler problem
3. **"You gotta believe"**
   - Otherwise, you'll try to solve the problem both recursively and non-recursively. This is bad.
4. **Compound interest rule**
   - Efficiency: Don't duplicate work by solving the same instance of the problem in separate recursive calls
   - Later

Demo

Let’s watch in the debugger

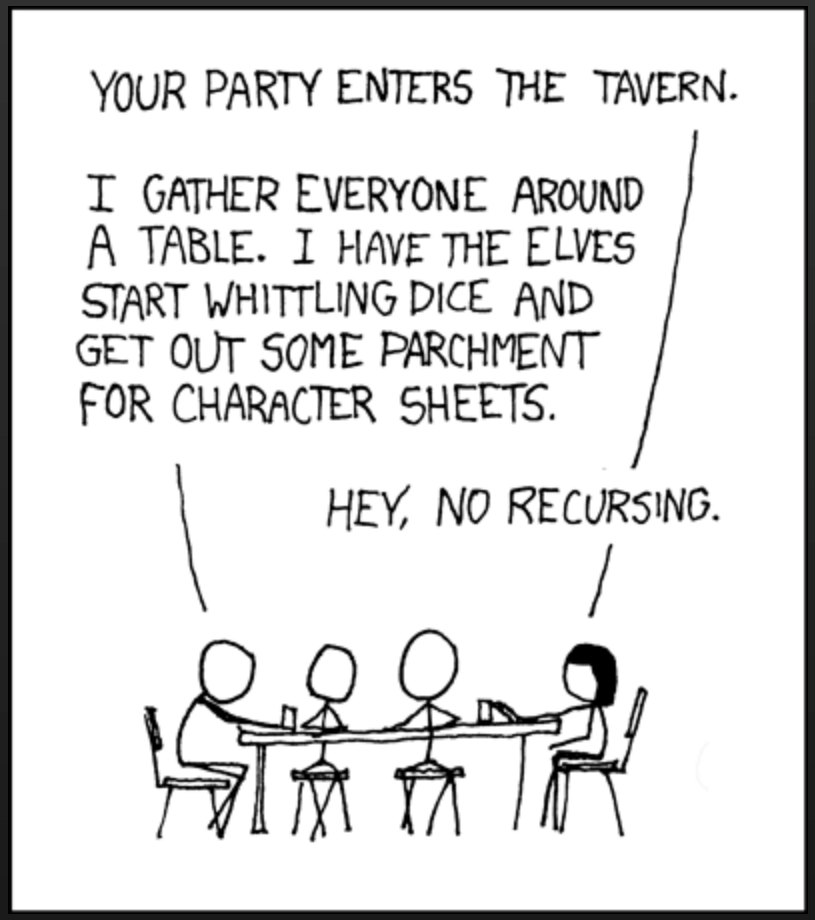
- Checkout Recursion project
- Navigate to memoization package.
- Let's look at stack trace for Fibonacci.fib()
- What if missing base case?

What else can we do recursively?

- gcd(a,b): //assumes a > b
  - if **a** is a multiple of b, return b
  - Otherwise, return gcd(b, a % b) (guaranteed to be smaller)

- myPow(x, a)
- Program this now
- Contest: Which table can write a version with the shallowest call stack?

Break


YOUR PARTY ENTERS THE TAVERN.
I GATHER EVERYONE AROUND A TABLE. I HAVE THE ELVES START WHITTLING DICE AND GET OUT SOME PARCHMENT FOR CHARACTER SHEETS.
HEY, NO RECURSING.

“Memoization”

- What if I wish to speedup the calculation of fib(n)?
- Can I do this any faster with recursion?
- What is memoization?
- How can I use memoization to speedup calculation?

Mutual Recursion

- Two or more methods that call each other repeatedly
  - For example, Hofstadter Female and Male Sequences

$$F(n) = \begin{cases} 1 & \text{if } n = 0 \\ n - M(F(n-1)) & \text{if } n > 0 \end{cases}$$

$$M(n) = \begin{cases} 0 & \text{if } n = 0 \\ n - F(M(n-1)) & \text{if } n > 0 \end{cases}$$

  - Burning Questions for you to figure out now by coding:
    - How often are the sequences different in the first 50 positions? first 500? first 5,000? first 5,000,000?
    - This is part of the homework

Two Mirrors

If you actually do this, what really happens is Douglas Hofstadter appears and talks to you for eight hours about strange loops.

A graphical exercise on recursion

- Sierpinski's Triangle…
  - http://www.shodor.org/interactivate/activities/SierpinskiTriangle/
- See starting code in the repository.
- How can you use recursion to solve this problem?
  - Discuss with a partner
- You may pair-program this if you want
- Fun extensions:
  - Add color
  - Play with non-equilateral triangles