

CSSE 220

Day 12

Recursion

Checkout *Recursion* project from SVN

Questions?

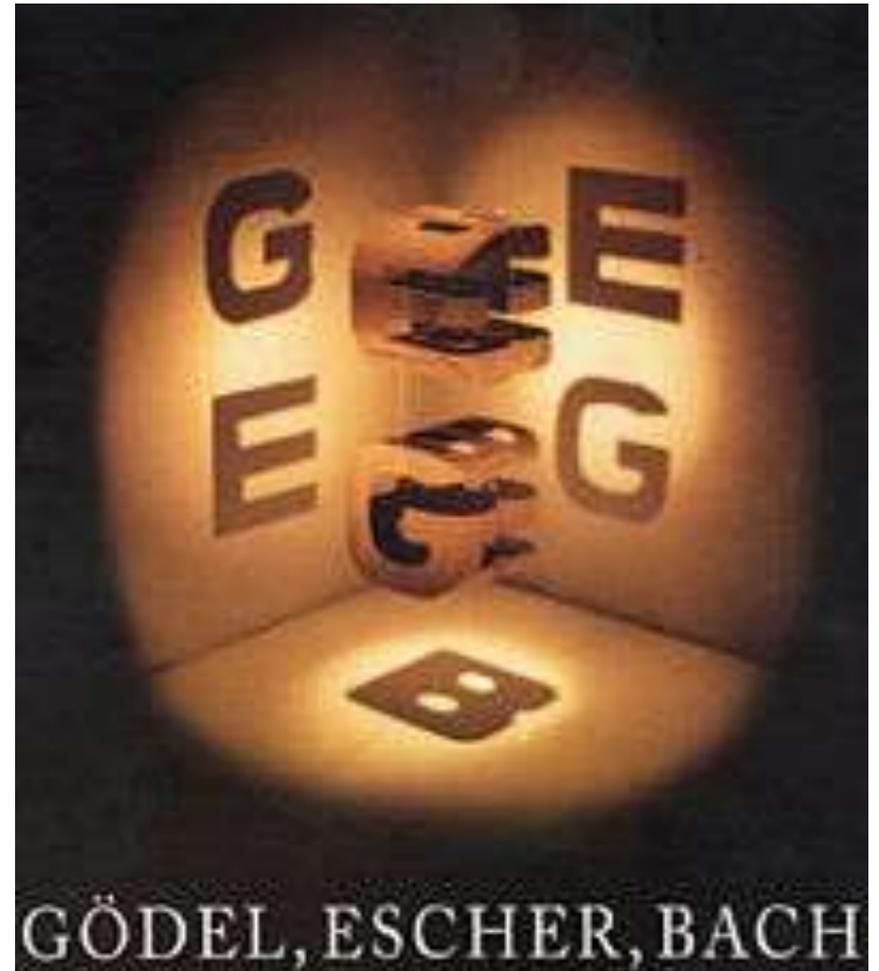
Packages

- ▶ Let us group related classes
- ▶ We've been using them:
 - **javax.swing**
 - **java.awt**
 - **java.lang**
- ▶ Can (and should) group our own code into packages
 - Eclipse makes it easy...



Gödel, Escher, Bach

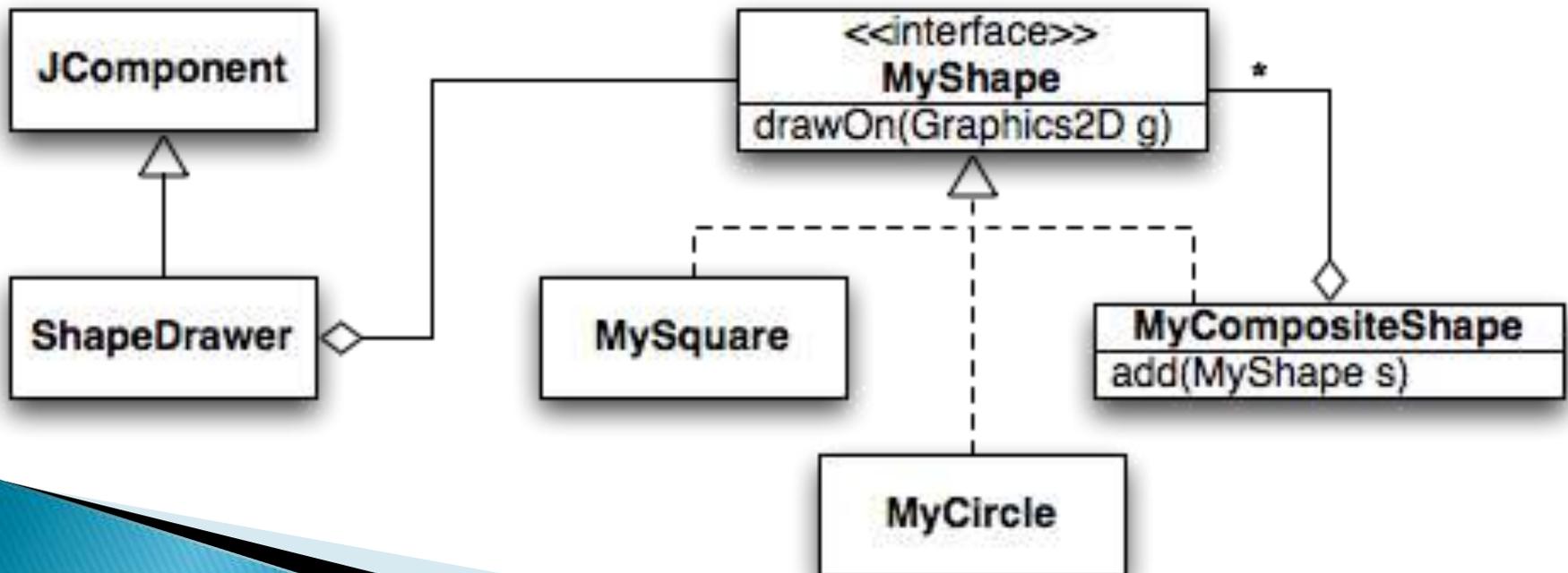
- ▶ By Douglas Hofstadter
- ▶ Argues that intelligence arises (in part) because of **our ability to think about thinking**



Recursion

- ▶ A solution technique where the same computation **occurs repeatedly** as the problem is solved

recurs



An example – Triangle Numbers

- ▶ If each red block has area 1, what is the *area* $A(n)$ of the Triangle whose *width* is n ?

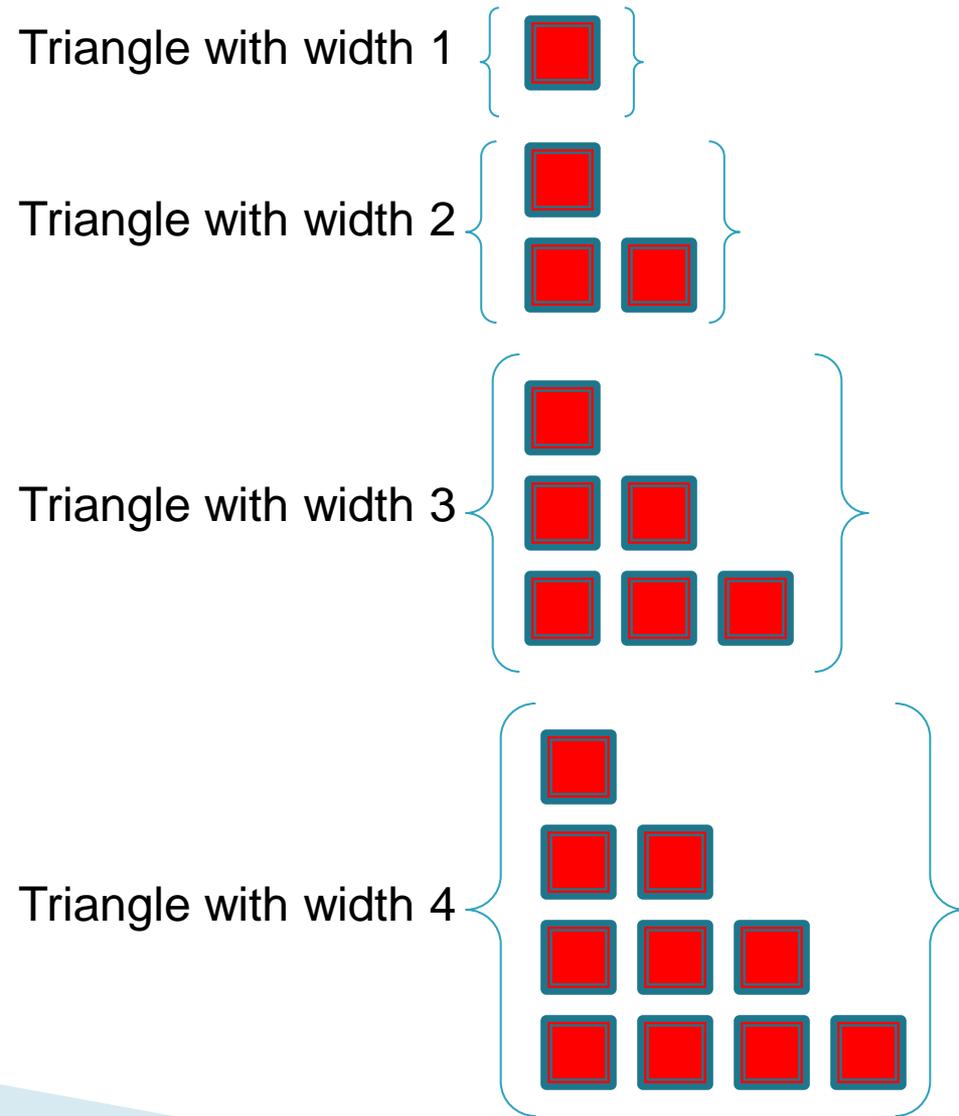
- Answer:

$$A(n) = n + A(n-1)$$

- ▶ The above holds for which n ? What is the answer for other n ?

- Answer: The recursive equation holds for $n > 1$.

For $n = 1$, the area is 1.



Frames for Tracing Recursive Code

1. Draw box when method starts

2. Fill in name and first line no.

3. Write class name (for static method) or draw reference to object (for non-static method)

method name, line number

scope box

parameters
and local variables

4. List every parameter and its argument value.

5. List every local variable declared in the method, **but no values yet**

6. Step through the method, update the line number and variable values, draw new frame for new calls

7. "Erase" the frame when the method is done.

Thanks for
David Gries for
this technique

Q1-Q2

Suggested Practice

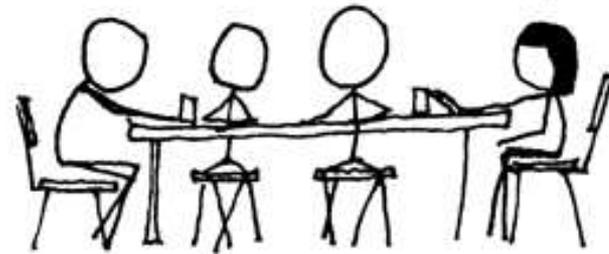
- ▶ Trace the **buildShape(MAX_DEPTH)** method call in **shapes.Main's** main method

Tabletop Roleplaying

I may have also tossed one of a pair of teleportation rings into the ocean with interesting results.

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.



Key Rules to Using Recursion

- ▶ Always have a **base case** that **doesn't recurse**
- ▶ Make sure recursive case always makes **progress**, by **solving a smaller problem**
- ▶ **You gotta believe**
 - Trust in the recursive solution
 - Just consider one step at a time

Programming Problem

- ▶ Add a recursive method to Sentence for computing whether Sentence is a palindrome

Sentence
String text
String toString() boolean isPalindrome

Recursive Helpers

- ▶ Our `isPalindrome()` makes lots of new Sentence objects
- ▶ We can make it better with a “recursive helper method”
 - ▶ Many recursive problems require a helper method

```
public boolean isPalindrome() {  
    return isPalindrome(0, this.text.length() - 1);  
}
```



Position of first letter of the remaining String to check



Position of last letter of the remaining String to check

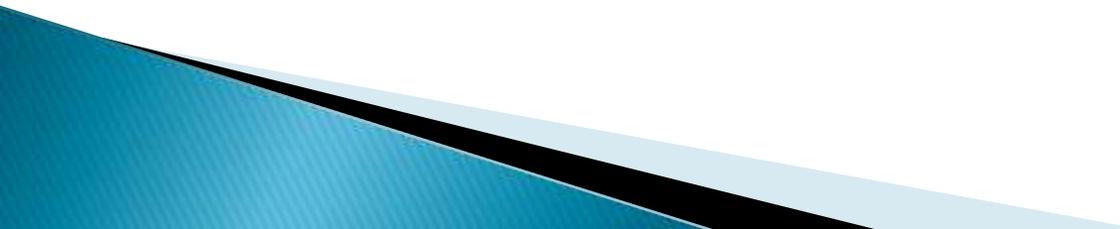
Homework part 1

- ▶ Reverse a string...recursively!
 - ▶ A recursive helper can make this really short!
- 

Another Definition of Recursion

- ▶ “If you already know what recursion is, just remember the answer. Otherwise, find someone who is standing closer to Douglas Hofstadter than you are; then ask him or her what recursion is.”

—Andrew Plotkin



Recursive Functions

- ▶ Factorial:

$$n! = \begin{cases} 1 & \text{if } n \leq 1 \\ n * (n - 1)! & \text{otherwise} \end{cases}$$

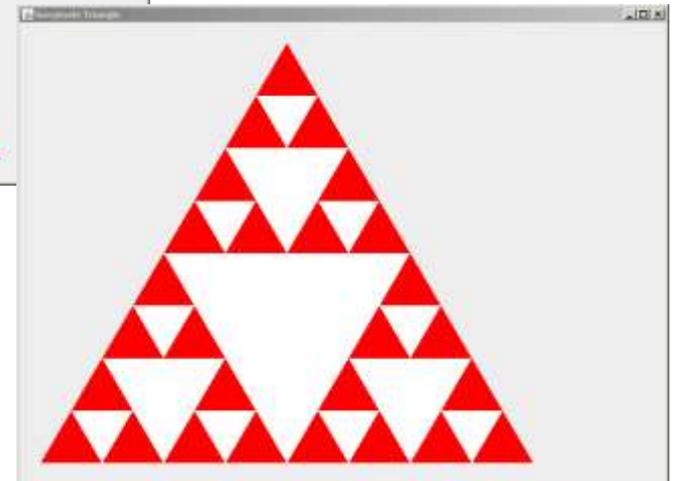
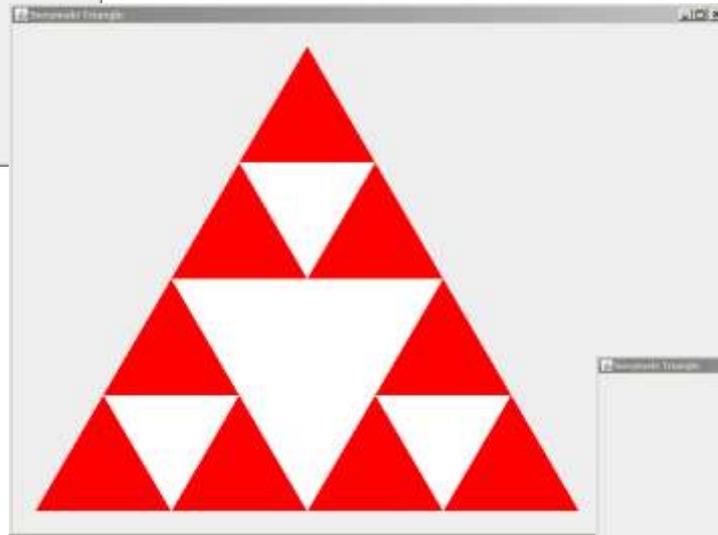
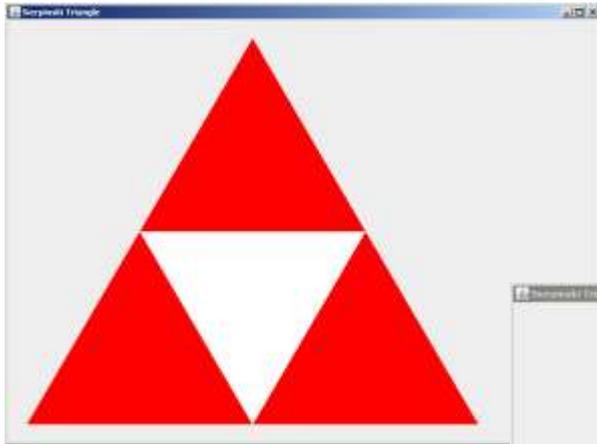
Base Case

Recursive step

- ▶ Ackermann function:

$$A(m, n) = \begin{cases} n + 1 & \text{if } m = 0 \\ A(m - 1, 1) & \text{if } m > 0 \text{ and } n = 0 \\ A(m - 1, A(m, n - 1)) & \text{otherwise} \end{cases}$$

HW, part 2: Sierpinski



Work Time

»» Homework 12: Sierpinski