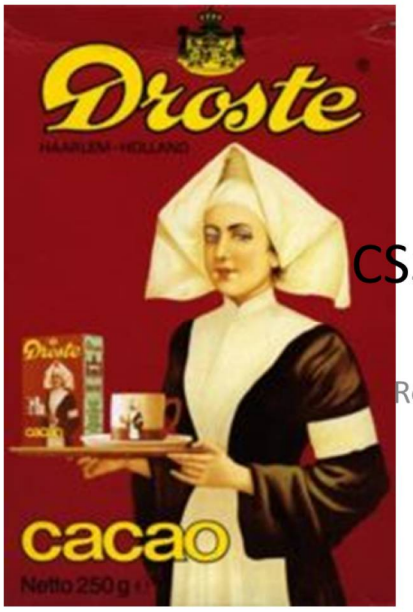


CSSE 220

Recursion

Checkout *Recursion* project from SVN

Bring hardcopy of code from RecursionSolution:
Sentence.java
Triangle.java
FactorialCalculator.java

It would be helpful to add line numbers to your printout for Triangle.java and FactorialCalculator.java

Recursion

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

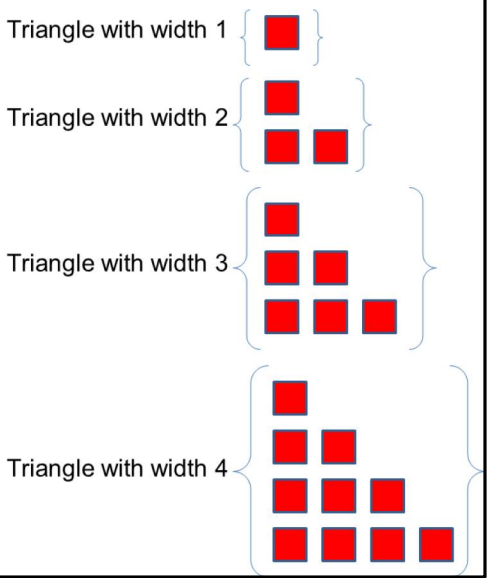
- Examples:
  - Sierpinski Triangle
  - Towers of Hanoi:
    http://www.mathsisfun.com/games/towerofhanoi.html
    or search for Towers of Hanoi

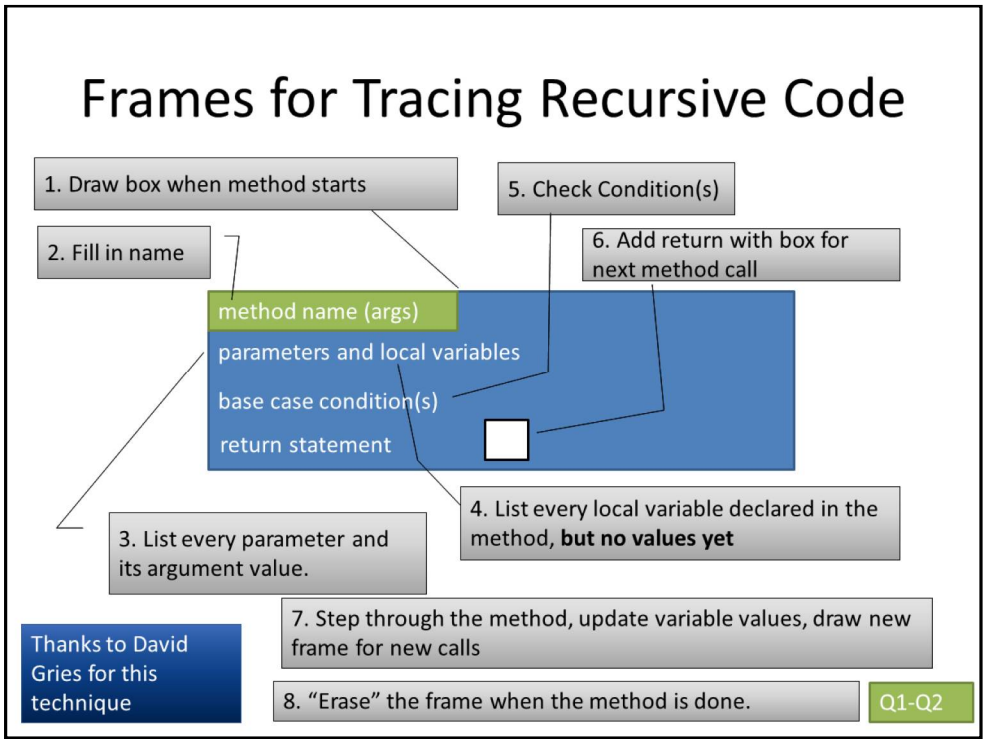
I used this slide instead of the Shapes one.

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 = 0, the area is 0.





Have them look at Triangle in Eclipse (in examples package) while you project this and draw on board.

Buffalo says I personally had a lot of trouble with scope boxes so I just left it out.

When looking at the code, discuss "base case", "recursive case", and "trusting the recursion".

Trace the Triangle Numbers example from text on board and quiz, be sure to show "return" arrows from callee to caller, labeled with return value.

Set them lose on tracing recursive factorial on quiz, comparing answers.

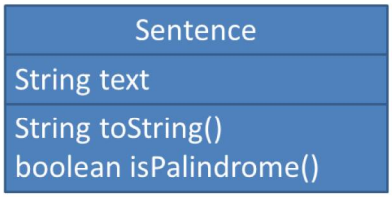
Show one or more of these examples in the debugger. (factorial?)

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



Live code the solution.

Buffalo says I prefer simple palindrome

- What are the simplest cases? These are the base cases.
-If we aren't in a simplest case, how can we make progress to a smaller problem?

-Do a solution that does not include a helper method.

Practice Practice Practice

- Head to http://codingbat.com/java/Recursion-1 and solve 5 problems. I personally like bunnyEars, bunnyEars2, count7, fibonacci, and noX
- Get help from me if you get stuck
- Then take a look at the recursion homework

bunnyEars:

```
public int bunnyEars(int bunnies) {
  if (bunnies == 0) {
    return 0;
  }
  return 2 + bunnyEars(bunnies-1);
}
```

bunnyEars2:

```
public int bunnyEars2(int bunnies) {
  if (bunnies==0) {
    return 0;
  }
  if (bunnies%2==0){
    return 3 + bunnyEars2(bunnies-1);
```

```
  }
  return 2 + bunnyEars2(bunnies-1);
}
```

Count7

```
public int count7(int n) {
  if (n==0) {
    return 0;
  }
  if (n%10 == 7) {
    return 1 + count7(n/10);
  }
  return count7(n/10);
}
```

Fibonacci

```
public int fibonacci(int n) {
  if (n==0) {
    return 0;
  }
  if (n==1 || n==2) {
    return 1;
  }
  return fibonacci(n-1) + fibonacci(n-2);
}
```

noX

```
public String noX(String str) {
  if (str.length() == 0) {
    return "";
```

```
  }
  char c = str.charAt(0);
  if (c=='x') {
    return noX(str.substring(1));
  }
  return c + noX(str.substring(1));
}
```