Methods & Recursion

Oct 30

More method writing

Chalkboard

Lucas numbers and the Golden mean

- Program to get a single integer, N, from user (or command line)
- Method to calculate and return the first N Lucas numbers
  - $$L_n := \begin{cases} 2 & \text{if } n = 0; \\ 1 & \text{if } n = 1; \\ L_{n-1} + L_{n-2} & \text{if } n > 1. \end{cases}$$
- method signature should be
  `public static int[] lucasNumbers(int count)`
- the int[] returned should have length equal to count.
- In main, print L[n]/L[n-1] (as a double) for n=1 .. n=(N-1).

Overloading

why write only one!

- Can write several method with the same name
  - must have
    - same return type
    - same modifiers
    - different parameters
- Java considers methods different if they have different signatures

- WHY would I ever want to do this???
  - Squaring

In summary

- Primitive datatypes are passed by value (copy)
- Arrays are passed by reference (alias)
  - So contents changes survive
    - if you do not change the pointer

Recursion

- Idea, write a method that calls itself! (what could possibly go wrong)
- Important, it should call itself with a slightly simpler problem

- Factorial
  - 6! = 6 * 5 * 4 * 3 * 2 * 1
  - 6! = 6 * (5 * 4 * 3 * 2 * 1)
    - the parenthesized stuff is 5!
    - so 6! = 6 * 5!

Recursive addition

What is 5+4

Problem: "I only how how to add and subtract 1”

so: 5+4

5+(4-1) + 1

5+((3-1)+1) +1

5+((2-1+1) +1 + 1

5+1+1+1+1

6+1+1+1

7+1+1

8+1

9

Stopping Recursion

The "base case"

```
public void loop2(int c) {
 int i;
 for (i=c; i >= 0; i--) {
    System.out.println(c);
    }
}
```

```
public void badRecurse(int c) {
 System.out.println("B" + c);
 badRecurse(c-1);
}

public void okRecurse(int c){
 System.out.println("OK" + c);
 if (c==0) return;
 okRecurse(c-1);
}

public void goodRecurse(int c) {
 System.out.println("G" + c);
 if (c>=0) {
      goodRecurse(c-1);
   }
}
```

Recursion Overview

- Base case(s):
  - no recursive calls are performed
  - every chain of recursive calls must reach a base case
- Recursive calls:
  - Calls to the same method in a way that progress is made towards a base case

Recursive addition

What is 5+4

Problem: I only how how to add and subtract 1”

so: 5+4

5+(4-1) + 1

5+((3-1)+1) +1

5+((2-1+1) +1 + 1

5+1+1+1+1

6+1+1+1

7+1+1

8+1

9

Write code

Chalkboards

```java
/** Print the given char the number of times given by num consecutively on
     * the same line. After the last, print a newline.
     * @param ch the char to print
     * @param num the number of times to print the char
     */
    public void rowOfChars(char ch, int num)


    /** Compute the nth power of a the given number.
      * DO NOT USE Math.pow
    */
    public int nthPower(int num, int Power)
```

```java
// Usage (e.g. in a main method)
rowOfChars('d', 17);
rowOfChars('X', 15);
System.out.println(nthPower(2,10));
System.out.println(nthPower(3,5));
System.out.println(nthPower(7,4));
```

Recursion choices

before or after

- Often with recursion you have a choice of when to act
  - on the way down
  - on the way back up
  - Consider again factorial: $$f(n) = \begin{cases} 1 & \text{if } n = 0 \\ n \cdot f(n-1) & else \end{cases}$$
  - We can implement in either way!
  - See factorial implementation on the class website
    - Problem -- on the way down usually requires more parameters to the function

Recursive helpers

- Consider a recursive function to print all of the items in an array
  public static void printArray(int[] arra);

- Problem, how do you do it?
  - You need another parameter to hold the index in the array

- Recursive helpers!
  - not really intended for anyone else to use
  - actually do the recursion, unlike the intended entry point

- What would a helper look like for printArray?

Printing Digits

Recursively

- Task: Print each of the digits of a number -- on its own line
  - Do this using recursion
    - (Yes you can do it with a while loop) -- so start there
  - Integers
    - The numbers come out in the reverse order
    - Reverse sucks .... How can I fix that!!
  - What about floating point numbers
    - use recursion for that also -- HOW???