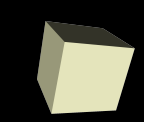
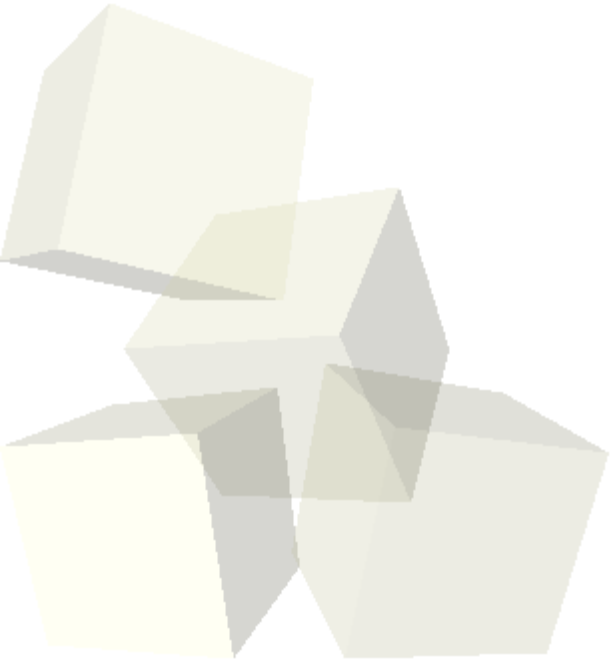


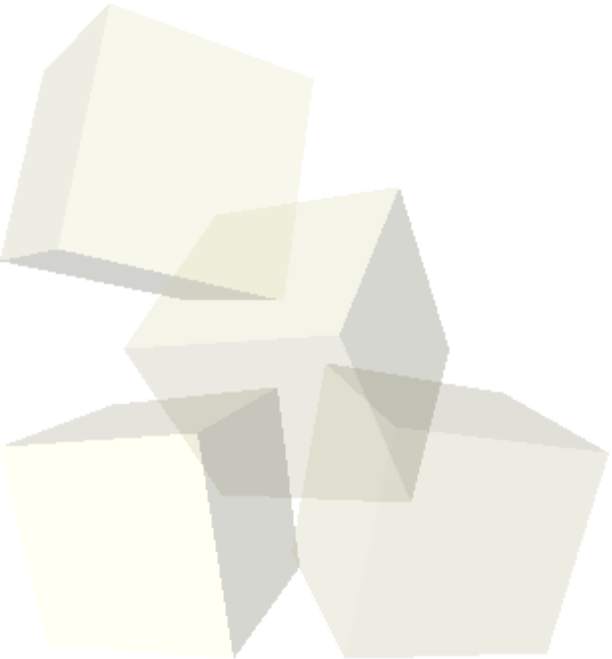
Recursion

3-30-2006



Opening Discussion

- What did we talk about last class?
- Do you have any questions about the assignment?



Recap of Coding

- Let's go look at what we got through last time with our drawing program. I have added a little bit more into it that we can see as well.

What is Recursion?

- A recursive function in mathematics is one that is defined in terms of itself. In computer science it is a function that calls itself.
- Mathematically these functions have a general relation for most input values as well as some defined “termination” values.
- The Fibonacci numbers are a nice example of a recursive function, but we can also define factorial recursively.

Recursion and the Stack

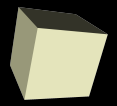
- Java and most other modern programming languages use a stack for function calls. The stack is a section of memory that sits at one end of the memory that program owns.
- Each function gets a stack frame that holds local variables and arguments to the function. Calling functions push these frames and returns pop them.

Recursion and Loops

- The simplest recursive functions are basically loops. They call themselves once and have an argument that works as the iteration variable.
- The call to itself has to be conditional. Otherwise it is like an infinite loop, but you get a stack overflow when it runs out of memory.
- Note that a recursive function is a bit more powerful than a standard loop because the stack remembers earlier values. So a recursive “loop” can not only visit all the values, it can revisit them in reverse order.
- Let's write some code to demonstrate how this all works.

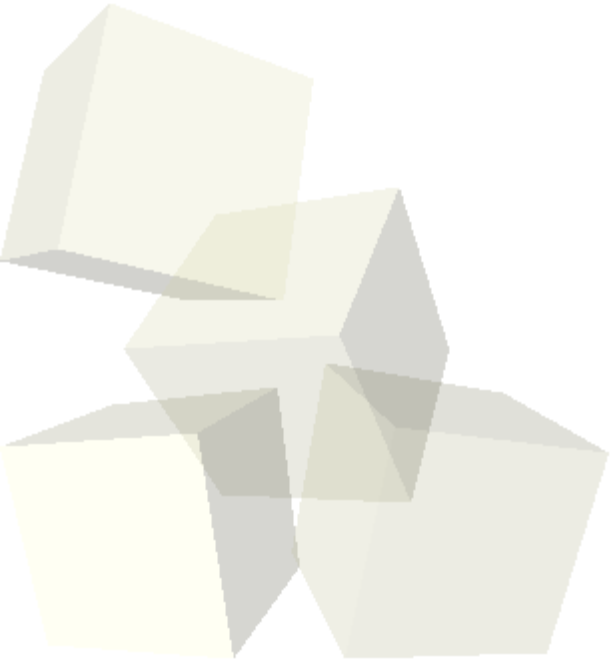
The Power of Recursion

- Recursive functions are truly powerful when they call themselves more than once. Converting these types of function to loops requires significant reworking of the logic (at best).
- The stack is what enables this because with the stack the flow of control can go off in one direction, then return to the function and go off in another direction.
- These functions have “tree” call structures.
- Any function that needs to do or test several options to get a result can be written most easily with recursion.



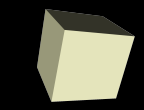
floodFill

- You are probably all familiar with the option in paint programs that allows you to “pour” in paint to fill a region of a picture. How would you do this with loops in a program?
- With recursion we just write a function that calls itself four times. Because of the memory of the stack, it can go “around corners”.



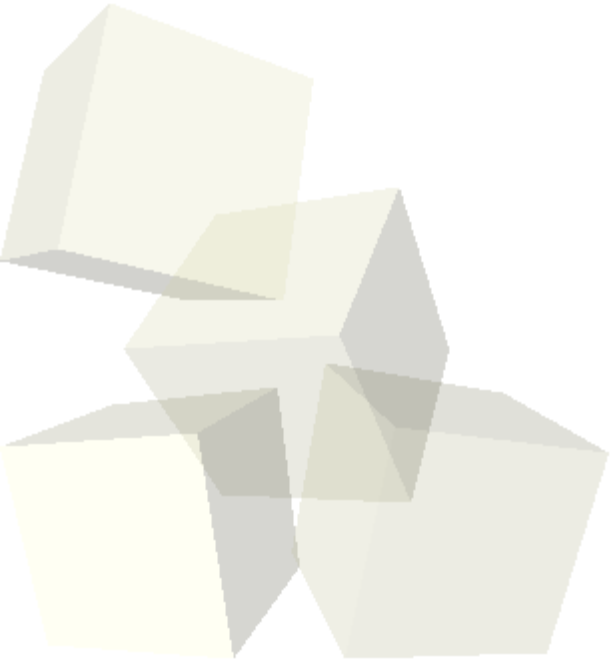
Maze and Graph Traversals

- By adding a bit of extra logic to a floodFill we can turn it into a function that searches through mazes. 2D mazes are a special example of the general category of graphs. A graph has vertices that are connected by edges.
- The key to these problems is that we want to leave behind “bread crumbs” to mark our path and pick them up as we backtrack.
- We can also can leave other markers that can help speed up our algorithms depending on what we are doing.



Drawing Program

- Let's do a bit of work on our drawing program now. Between today and next class I want us to put in a maze drawing object.



Minute Essay

- Write a method that takes an int and an int[] and returns the largest element in the array.
- Remember that you have to turn in assignment #5 today. Unfortunately, I won't be available at all this afternoon because of a department meeting followed by the UPE induction followed by the “Dress for Success” event. By the way, you should consider going to that at 7pm tonight.