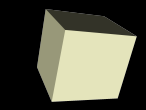
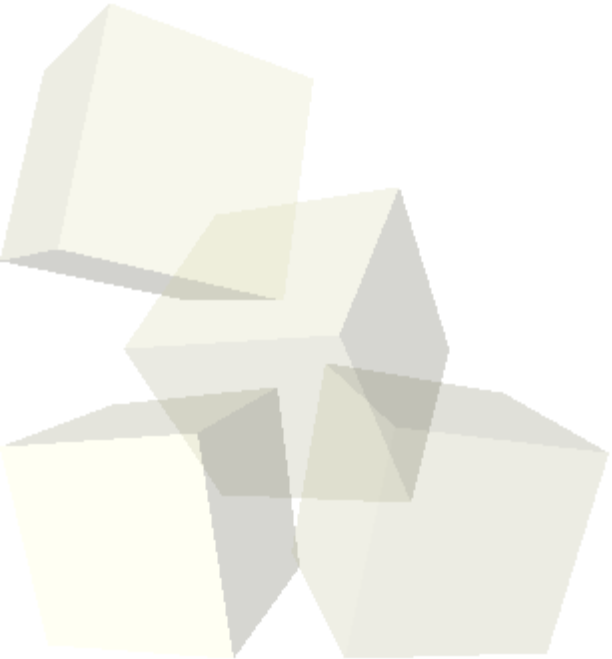


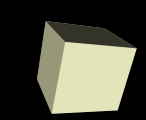
Exceptions and Refactoring

3/20/2007



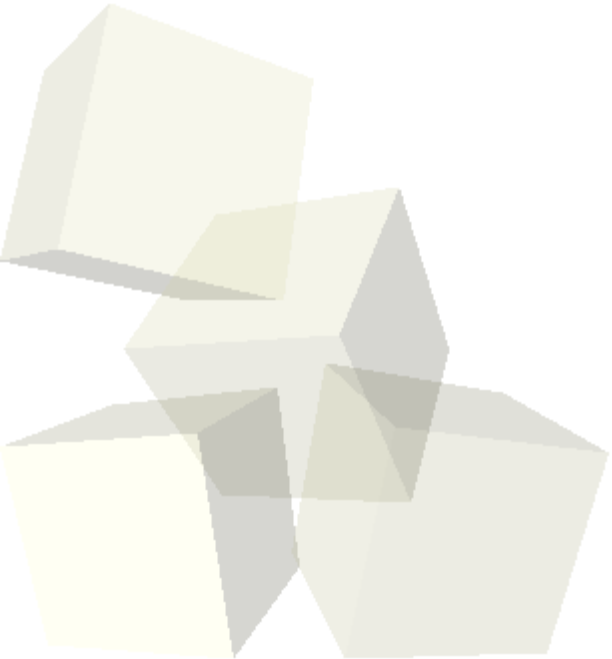
Opening Discussion

- Do you have any questions about the quiz?
- Do you have any questions about the assignment?
- Do you have any questions about the reading?
- Why do we use exceptions instead of return codes? What are the specific advantages?
- What is refactoring? What are reasons for refactoring our code?



Code

- Let's go into the code we have written previously and add some proper error handling.
- Where are some places in our existing code where we would want to throw some more informative exceptions?



Refactoring Code

- This is something that you do when you don't want to change the functionality of your code, but you want to change how it does something.
- There is an aspect of our drawing program that fits with one of the “smells” that you read about. It is related to how we add objects into our tree. I'd like to use a mechanism that doesn't have a switch statement and uses polymorphism instead.
- Part of the reason I want to do this is that I don't like having the add button. I'd rather have an add menu and putting in menu items for lots of new things will become a pain.

Recursion

- You learned about recursive functions in PAD1. A recursive function is simply a function that calls itself.
- You can use recursion to imitate loops, but we won't do that very often in C or Java. Where recursion comes in really handy is when a function needs to test more than one alternative at a time.
- This works nicely because the call stack remembers where you are in a given function so when you return back, you can take off from that point again.

Maze Solving

- One of my favorite recursive algorithms is maze solving. This is a special case of graph traversals which are common problems in CS.
- We'll use a 2D array of ints as our maze and we can even put this into our drawing program.
- A simple warm-up piece of code is flood fill like you would have in a drawing program.
- Once we have that we can see how to convert it to do things like find the shortest path through a maze or count all paths through a maze.
- We can try to make this nice and graphical as well so it fits properly into our drawing program.

Formula Parsing

- Another one of my favorite recursive algorithms is formula parsing. This allows us to have the user type in a function and our code can evaluate it.
- We do this through “divide and conquer”. We split the formula in two across the lowest precedence operator then recursively evaluate the two halves.
- We can use this to put function plotting into our program if we give it the ability to handle a variable.

Minute Essay

- Write a recursive function to find the longest path through a maze.
- The design for assignment #5 is due a week from today.
- If you have thought about doing TopCoder, this is a good time to register because they are about to have the 2007 Top Coder Open.