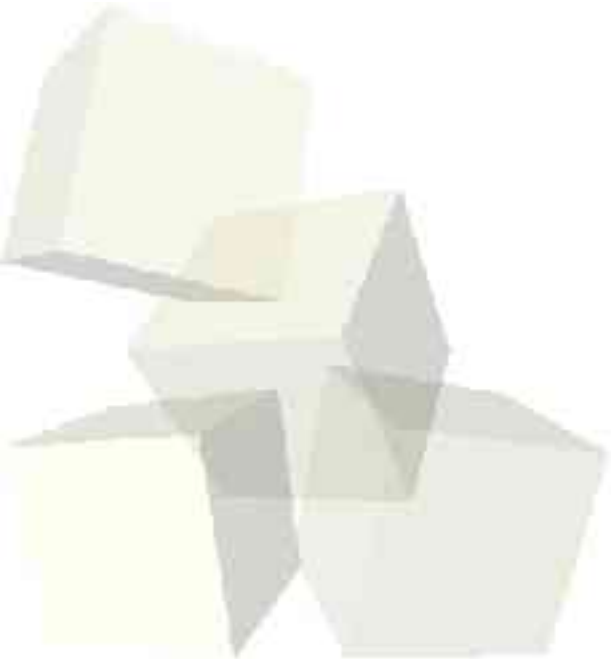
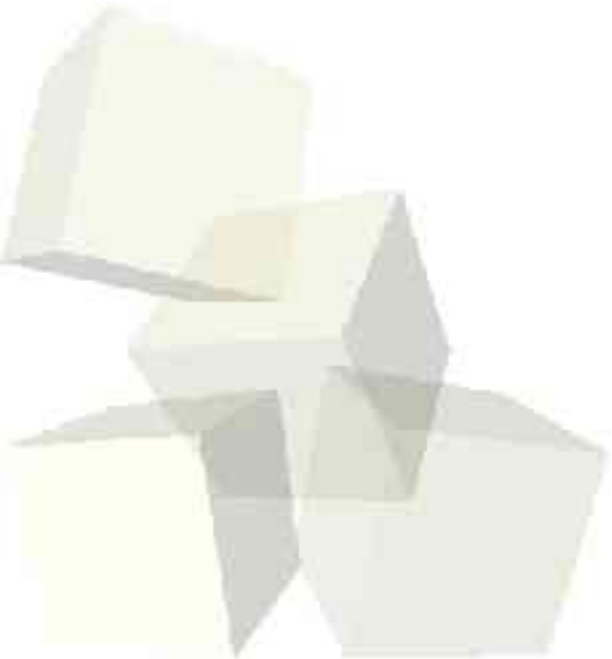
Math, Data Abstraction, and Data Driven Recursion

9-10-2004



Opening Discussion

- What did we talk about last class?
- Let's look at how you will submit your programs to me.



User Defined Math Functions

- We can write math functions as well. Last time we did factorial and saw how Scheme does exact arithmetic with big integers.
- Let's write some other quick functions as well.
  - List-ref – get the specified element of a list
  - Taylor-cos – do a Taylor approximation of cos

Data Abstraction

- Your book uses the example of rational numbers for data abstraction. The idea is that we define functions to get at data in some list structure and functions to manipulate it.
- The user never has to know the exact internal representation, just the functions to deal with.
- We can instead do this with a complex number data abstraction.

Flat Recursion

- Functions that only play with the top level of a list are sometimes called flat. Most of what we have written has been of this type. What is the exception?
- We can write some others.
  - Merge – merges two sorted lists into a single list.
  - Remove – removes all instances of an item from a list.
- Let's take a second to look a bit closer at how recursion works by tracing one of these functions.

Deep Recursion

- Functions that not only recurse over the elements of a list, but also into the elements of a list are said to use deep recursion.
- We have done one of these, deep-count. Let's write two others.
  - Remove-all – removes all occurrences of an item for a list and all its sublists.
  - Reverse-all – reverses the list and all the sublists in it to any depth.

Minute Essay

- What are your thoughts on the speed of the class so far?
- Remember that your assignment is due by tonight at midnight.