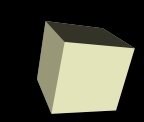
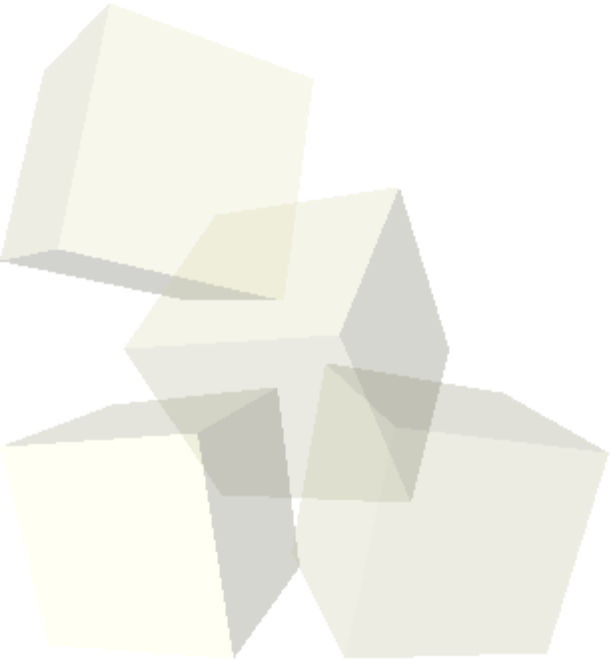


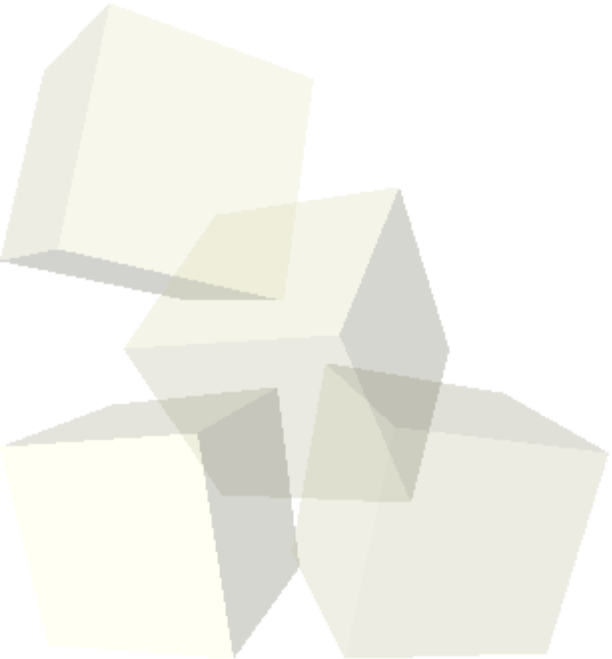
Concluding Sorting and Threads

2-9-2006



Opening Discussion

- Do you have any questions about the reading?
- Do you have any questions about the assignment?



Shell Sort

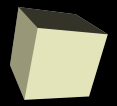
- Arrays.sort uses either quicksort or merge sort, depending on the data type. We will talk about those later when we do recursion. There are faster sorts you can write without using recursion.
- One example of that is Shell sort. Shell sort is also called the decreasing gap sort. It seems rather magical because do insertion sort multiple times with smaller gaps each time and you actually come out with a sort that is faster than a single insertion sort.

Measuring Performance

- I have said that bubble sort is bad and that insertion sort is a bit better while Shell sort is truly better. How do you know though? How can we demonstrate this?
- Since the primary operation for the sorts is comparisons, we can “instrument” our comparator so that it counts how many times it is called. Using that on a large data set will give use a solid number for the number of comparisons a particular sort has to do.

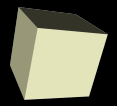
Threads

- This is a topic that has been in PAD2 since we switched to Java, but that I have moved up to earlier in the semester over time.
- The reason is simply that is is a topic that is becoming more important in computing.
- All the programming you have done so far has been in a single thread of execution. That is to say that the program goes from one line to the next doing one at a time in order. In a program with multiple threads the same thing happens, but in multiple places at once.



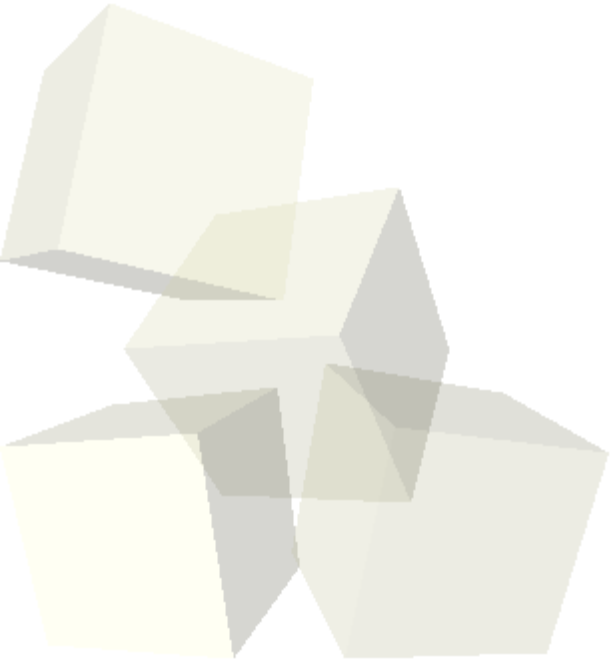
Why Threads?

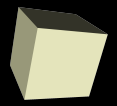
- On a machine with a single processor and a single core threads simply give the impression of two things happening at once. With the widespread arrival of dual core processors, most of the new machines have the ability to actually do two things at once, assuming programs have more than one thread.
- For at least a while, the future is about adding more cores to processors so software is going to have to change and that means programmers have to change as well.



Threads in Java

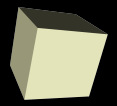
- What are some of the key concepts associated with threads in Java? How long has Java been able to handle multiple threads?
- How do you create new threads?
- What are some of the issues associated with programs that have multiple threads? How do you get around these issues?





Coding

- Let's go ahead and write some code where we do some large sorts in parallel. Since threads are likely quite new to you we will do these as a class. You should also bring up the API and look at the Thread class.
- We can see how much of a speed up we can get on various machines in the department that have different numbers of processors.

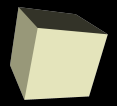


Problems

- The primary problem one runs into with multithreaded programs is that threads share memory and more than one thread can access a piece of memory at once. This isn't a problem if they are just reading, but if any thread is writing you can have bad situations.
- An extreme condition would be to consider two threads operating on an array. Worst case is both are sorting the array at the same time. You could imagine one sorting while another tries to do a binary search and the results are similarly bad.
- The simplest (and most common) example is a bank account where a race condition occurs.

Synchronization

- The way to prevent two threads from accessing the same piece of memory at the same time is to synchronize the critical pieces of the code. You can put the synchronized keyword in front of methods or make synchronized blocks.
- Each object and class in Java can have a monitor that is locked when synchronized code is being executed. Only one thread can hold the lock on the monitor at a given time. This insures that you never have two threads executing critical code on a single object at the same time.



Wait and Notify

- We can get even more control over how threads behave with the wait and notify methods.
- The wait method will stop the execution of a thread until some other thread tells it to continue execution. The notify and notifyAll methods are how threads tell other threads that they are supposed to wake up.
- All of these must be called by a thread that holds the monitor to the object they are being invoked on. Typically that means that are called from inside synchronized code.

Minute Essay

- Give an example of a place in a program where you could potentially use multiple threads to speed up execution.
- Remember that the assignment is due today. I'll be a bit late getting open lab because we have a departmental meeting at 3:30.