CS3134 #1
9/2/03
Janak J Parekh

Intro
• Website location
• Instructor and TA contact info, office hours, locations
• Textbook (why, applets)
• Course structure (HW: 6*150 + Q = 50 + F = 100 + class participation)
• Homework structure, submission, lateness
• Exams open-book; midterm on 10/16?
• Prerequisite (Java)
• Reasonable Person Principle, lecture material, sleeping
• Cheating, feedback

Poll
• School (GS, SEAS, CC)
• Level of Java knowledge
  – Who took CS1004
  – Basic applications
  – Basic applets, AWT, Swing
  – OO
    • Subclassing
    • Interfaces
    • Polymorphism
    • Inheritance
    • Visibility modifiers
  – Java Collections: Vector/ArrayList, Hashtable/HashMap, etc.
  – Intro recitation?
• C/C++ knowledge
• Midterm

Motivation
• What are the two things computers do?
  – Store information
  – Manipulate information
• Why do we need to know how it does it? There’s Java Collections, right?
  – No “one” way of doing it
  – Each way has its advantages and disadvantages
  – Raw CPU power can’t overcome inefficiency
  – Java Collections aren’t a catch-all, even though they’re a nice abstraction
• But don’t we need to know the problem beforehand?
  – Not necessarily
  – We want to develop a “toolkit” to be useable in the future
  – One fundamental concept makes it feasible...

Abstraction
• Fundamental concept in Computer Science, especially applies here
• Lafore defines it as “considered apart from detailed specifications or implementation”
• Create a layered system, building up to complex applications
• Abstract data types as fundamental building blocks of information
  – What data types does Java support?
  – Primitive vs. reference data types
• Abstract algorithms as fundamentally useful to a broad range of applications
  – Manipulation, sorts, searches
• You won’t always have to design them, but you’ll always have to use them
Example

- Employee database
  - How can we represent this information?
  - What kinds of operations would we do on such an application?
  - What problems do we encounter with a naïve implementation?
  - Can we do better?
- Can an abstract knowledge of data structures and algorithms help?

What’s out there?

- Data structures?
  - Arrays (sorted or unsorted), stacks, queues, linked lists, trees, hashtables, heaps, graphs
- Algorithms?
  - Insert
  - Search
  - Delete
  - Iterate
  - Sort
  - Recurse

Object-Oriented Programming, Java

- What is OO?
- How does OO help?
  - Improves abstraction
  - Allows code reuse
  - Access control to data: makes it more reliable – encapsulation
- Why do we use Java in a class like this?
  - OO is nice, but...
  - Java has no pointers
  - Strongly-typed
  - Garbage collection

What we’ll be doing the rest of the semester…

- Learning about these data structures
- Learning about some of the algorithms for them
- Learning which is best when
  - Elementary analysis of algorithms
  - Take the real class if you want to know the details
- Becoming better programmers!

Homework & Next Time

- No “official” homework, but...
- HW0 posted on webpage – no submission
- Get the book
- Next time: start looking at ADTs and OO design more closely, “refresher” on Java
  OO constructs