H section

• I’m trying an experimental approach to the course this semester
• Basic approach
  • Reading and absorbing “lecture” material can be done well outside of class, at your own speed
  • Long coding efforts outside of class can, at times, be very unproductive
Changes

• Less lecture in class. Read book, readings, lecture notes from Nelson’s section outside of class

• bring questions to class

• Lots of small programming exercises, many of them in class

• Lots of pair programming in class
About me

• You can call me Bill or Professor Pugh, whatever you are more comfortable with

• It may take me a while to learn your names

• Just got back from a year long sabbatical at Google in Mountain View

• just moved back into my house friday, so I apologize if a few things are disorganized at the start of class

Wednesday, September 2, 2009
Experience and Interests

• Work a lot with professional software developers

• Developed FindBugs, an open source tool downloaded 850,000+ times and used by Google, eBay, Amazon and in more than 160 countries

• Also interested in testing, concurrency, algorithms and data structures, and computer science education
Attendance

- Class in Mondays and Wednesdays, 10-11:50am, Friday 10-10:50am
- You are expected to make a good faith attempt to come to every class
- attendance will be taken, and you will loose points for more than 3 unexcused absences.
Absences

• If you know you can’t make it to a particular class due to a conflict, let us know in advance
Why

• We’re not enforcing attendance because we get paid more when you show up, or because we are mean

• Our experience, over and over again, has been that some students just don’t attend regularly, get behind, dig themselves into a hole, and have a terrible time learning what they could learn and getting the grade they could get
H1N1 flu

• H1N1 may turn all plans for the semester upside down, at least for a while

• If you think you have the flu, any flu, don’t come to class

• Might make pair programming ill-advised for a while

• We will handle any prolonged absences
# Grading

<table>
<thead>
<tr>
<th>Percentage</th>
<th>Component</th>
</tr>
</thead>
<tbody>
<tr>
<td>20%</td>
<td>Attendance</td>
</tr>
<tr>
<td>42%</td>
<td>Graded programming efforts</td>
</tr>
<tr>
<td>10%</td>
<td>Quizzes</td>
</tr>
<tr>
<td>10%</td>
<td>Midterms</td>
</tr>
<tr>
<td>18%</td>
<td>Final exam</td>
</tr>
</tbody>
</table>

Wednesday, September 2, 2009
Textbook

- Recommended

- “Objects, Abstractions, Data Structures and Design Using Java (version 5.0)”

- By Elliot Koffman and Paul Wolfgang
Textbook (cont.)

- Recommended
  - “Java Precisely (2nd Edition)”
  - By Peter Sestoft
Individual projects

• If you are interested in doing some significant programming effort of your own (e.g., an iPhone app), I may allow you to substitute that for some of the other graded programming assignments

• We will have limited time to discuss topics such as iPhone coding as an entire class

• May have time for group efforts in class
Academic honesty

- I take academic honesty very seriously
- Academic dishonesty results in a grade of XF
  - I have sent far more cases to the honor board than I want to, and the honor board usually upholds the cases I send
- Don’t think you can outsmart the instructional staff
Google code jam

• Encourage everyone to sign up for Google code jam
  • http://code.google.com/codejam/
• Have to register by Thursday, September 3rd
• Lots of programming exercises
  • may do some of them in class
Topics Preview

- Algorithms & data structures
  - Asymptotic efficiency
  - Lists, stacks, queues
  - Trees, heaps
  - Sets, maps, graphs
  - Recursion
Topics Preview

- Object-oriented software development
  - Software life cycle
  - Requirements & specifications
  - Designing objects & classes
  - Testing & code coverage
  - Programming paradigms
  - Design patterns
Topics Preview

- Programming skills
  - Java collection framework
  - Exceptions
  - Threads, synchronization
  - Java APIs
    - Graphics User Interfaces (GUI)
Questions

• How many took CMSC 131?
• How many have used Eclipse?
• How many could/would bring laptops to class?
• How many have iPhones, iPod touches, or Android phones?
Code Design exercise

- Pair up
- Paper and pencil/pen only
- Write a method that takes an array of ints as an argument, and compressing out the 0’s, moving nonzero entries towards lower indexes, filling in with 0’s at the top
- \([1, 0, 2, 0, 3]\) gets compressed to \([1, 2, 3, 0, 0]\)
Compressing out zero’s

• design the algorithm
  • pictures might be helpful
• ask questions
• think up test cases, including corner cases
• write code