COMP 401-F15
COURSE OVERVIEW

Instructor: Prasun Dewan (FB 150, help401-001@cs.unc.edu)
ENROLLMENT ISSUES

Some will drop (Normally 20% but with the course structure this time, the number may be much lower)

Please fill the enrollment request sheet if you want to be added

Check the enrolled list if in class, auto drop if you do not attend

No seats for auditing
Comp 401 – Foundations of Programming

Course Overview

This course is intended for people who have learned to program. Its goal is to teach you how to program well. The common programming strategy of beginners is to write the first solution they can think of without carefully identifying and weighing different alternatives. For all but the simplest problems, this approach of writing “quick and dirty” programs will take you to the debugging stage very quickly, but will make debugging slow. For large, complex programs, you need to identify multiple alternative solutions to the problem, choose an alternative that most directly solves the problem, and think carefully what your solution does, and how it works. The claim is that, although “quick and dirty” programming may produce a program faster, the concepts we teach will help you produce a correct program faster. Moreover, they will lead to programs that are easy to change and reuse.

We assume you have learned the following basic programming concepts: primitive types (Integers, real numbers, Booleans), variables,
Sign up on Piazza asap, as all announcements will be made there

First assignment is due in a week!
## COMP 401 vs. 110

<table>
<thead>
<tr>
<th></th>
<th>401</th>
<th>110</th>
</tr>
</thead>
<tbody>
<tr>
<td>CS Majors</td>
<td>Psychology, Biology, ...</td>
<td></td>
</tr>
<tr>
<td>Object-Oriented</td>
<td>Functional, Imperative, ...</td>
<td></td>
</tr>
<tr>
<td>Java</td>
<td>C++, Python, ...</td>
<td></td>
</tr>
</tbody>
</table>

- **Majors vs. Non Majors?**
  - Majors usually start with 401
  - But many 110 students become majors.

- **Object-oriented vs. Conventional?**
  - Both 401 and (current) 110 focus on objects.

- **Java vs. Non-Java?**
  - 110 and 401 are both in Java
  - Language is not the issue
  - Expected to use only those Java features taught in class
  - Course is not about Java
“Intermediate” vs. “introductory” programming

- Introductory may be object-oriented
- Introductory may be conventional
- Assume background in conventional programming and will teach Java syntax for it.
- Repetition for those who know object-oriented programming.
- Programs with large number of classes
- ~40
INTRODUCTORY CONVENTIONAL PROGRAMMING

- Types, variables, assignment, constants, expression
- Conditionals and loops.
- Input and output
- Arrays and/or Strings
- Procedures/Functions/Subroutines/Methods
Comp 401-001 - Assignment 1: Writing a Number Scanner

Date Assigned: Tue Aug 18, 2015
Completion Date: Fri Aug 28, 2015 (11:55 pm)
Early Submission Date: Wed Aug 26, 2015 (11:55 pm)

In this assignment, you will revise your programming skills by doing an assignment involving the use of many of the concepts that are a pre-requisite for this course, which include loops, end methods (procedures). In addition, you will learn how to use a string.

Extra credit if submitted early on a Wednesday

Normal submission date is a Friday
**Extra Credit Programming**

**Extra Credit**

Allow (a) a number to be succeeded or preceded by a variable number of blanks as in "2 45625 3000 " (b) an arbitrary number of numbers in a line. Do not terminate the program after encountering the first illegal (unexpected) character. Print the illegal character and continue scanning assuming the character had not been input.

---

**Students have varying interests and abilities**

Make up or insurance against bad grade in other assignments or exams

Better to give early without extra credit than late with

But if you are already late, might as well do extra credit to make up for late points
CONSTRAINTS

Constraints

1. Java has libraries that make the writing of this program trivial. The only library functions you should use are the Character.isDigit(), substring() and the Integer.parseInt() functions. Character.isDigit() is like Character.isUpperCase() except that it tells us whether a character is a digit rather than whether it is an uppercase letter. substring(), applicable to any string, is explained in the class material. Integer.parseInt() takes a string consisting of digits and converts into an int. For all of these cases, the ability to use the libraries is not essential to proving correctness.

Forbid use of certain Java libraries

Goal is not to teach Java and its libraries

It is to teach you how to build these libraries

Usually Java features not covered in class will be banned

Require use of certain programming techniques

Correctness is only one of the goals

Program must also be efficient and well crafted
## Layered Assignments = Project

<table>
<thead>
<tr>
<th>Course Information (8/18)</th>
<th>Bridge Scene - 1st day (long)</th>
<th>Bridge Scene - 2nd day (short)</th>
</tr>
</thead>
</table>

- Assignment 4
- Assignment 3
- Assignment 2
- Assignment 1

Assignments will build on each other to create a semester project.

See video of project to understand the full picture.

Due dates normally separated by a week (holidays, exams can cause more separation).
## Late Penalty

<table>
<thead>
<tr>
<th>Assignment 1</th>
<th>Assignment 1 Due Date</th>
</tr>
</thead>
<tbody>
<tr>
<td>Assignment 2</td>
<td>Assignment 2 Due Date</td>
</tr>
<tr>
<td>Assignment 3</td>
<td>Assignment 3 Due Date</td>
</tr>
<tr>
<td>Assignment 4</td>
<td>Assignment 4 Due Date</td>
</tr>
</tbody>
</table>

Can submit up to two class days (1 week) late with penalty

10 percent late if submitted next wed, 25% if next friday
SHIIFTING DATES

Assignment 1
Assignment 2
Assignment 3
Assignment 4

Assignment 1 Due Date
Assignment 2 Due Date
Assignment 3 Due Date
Assignment 4 Due Date

What if you get permanently behind?

Can shift assignment dates N times if last N assignments will not be done.

But you sacrifice the last N assignments, whose scores will go in fudge factor.
SKIPPING ONCE

Assignment 1 Due Date
Assignment 2 Due Date
Assignment 3 Due Date
Assignment 4 Due Date

Assignment 1
Assignment 2
Assignment 3
SKIPPING TWICE

Assignment 1
Assignment 2
Assignment 3
Assignment 4

Do submit sacrifice assignments if you catch up – will take this work into account in fudge factor
WORKING VS. ALMOST WORKING

Big difference between getting code working and almost working

Big differences in grades also

Very little partial credit if program not working

Errors will accumulate because of layered assignments

The two TAS, one UGA and I are here to help you
SCALABILITY ISSUES

This class is large

Issues?
EASY TO HIDE DIVERSIONS
Diversity in Students

Diversity, specially as you have not all taken the same prerequisite

Why bad?
Discussion is About Concrete Programs

Java Program Structure

```java
package lectures.scanning;
public class AnUpperCasePrinter {
    public static void main(String[] args) {
        if (args.length != 1) {
            System.out.println("Illegal number of arguments:" + args.length + ". Terminating program.");
            System.exit(-1);
        }
        String scannedString = args[0];
        System.out.println("Upper Case Letters:");
        int index = 0;
        while (index < scannedString.length()) {
            char nextLetter = scannedString.charAt(index);
            if (Character.isUpperCase(nextLetter))
                System.out.print(nextLetter);
            index++;
        }
        System.out.println();
    }
}
```

Pace at which you understand lecture in general and code in particular varies

Solution?
Recorded YouTube Videos?

Java Program Structure

Rewind, pause, fast-forward to match understanding pace

What do we lose?
ISSUES

- Personalized questions
- Socratic approach
- Scheduling time to watch video

- Should be few, piazza, post quiz synchronous discussion
- Recordings of live lectures rather than 15 minute lessons
- Quizzes