Syllabus – CS0007 - Fall 2013 - Jim Roberts

Description

CS 0007 is a first course in programming for students who are satisfying a course requirement for their major or minor or considering taking more couses in Computer Science. Even though it is designed for students with very little or no prior programming experience, the course is not a slow course or an easy course. The course operates on the idea of see it, do it, then talk about it, See it means that we write code and/or look at code during class to explore new topics. Do it means that you write similar code for your own practice and for lab assignments and program assignments. Talk about it happens when you have questions about your attempts. The material is presented in a stair step pattern with each step forming the foundation for the next step. Mastery of the material at each step is necessary in order to achieve success in the following steps. The class assumes that you know how to use a computer and a mouse. Beyond this, we will provide guidance, a bit of information, and many chances to explore.

This course will expose you to the fundamental constructs of many programming languages (Java, C, C++ to name a few), types, collections of data, procedural programming, object oriented programming. It will also expose you to planning, coding, debugging and testing of programs. Every assignment will require you to write a program that generates a required output.

In keeping with university tradition, much of what you learn will be on your own working alone or with others in the class. Like most college courses, we generate far more questions than we answer.

Collaboration is a good thing. You are encouraged to work together. Collaborations refers to higher level discussions about the problem at hand, strategies for solving the problem, ways to test the solution. It **does not mean** working together on a set of common code for the problem solution.

Collaboration means that you must write your own code, understand the code that you have written, and be able to modify that code to do similar tasks. Sharing code with others by any means crosses the boundary from collaboration into cheating. Be very careful about this.

If you do collaborate with other students, you must cite those students in the program comment for the assignment.

Work Load

You have to participate in the class. Just sitting in and listening, or worse not coming to class and relying on the web will not work. Likewise, missing one day in two or going away for two weeks when other stuff gets busy are very flawed strategies.

CS0007 is a three unit course. This means that you owe CS0007 nine hours of work outside of class on average every week. Some weeks will be less, some more, but in order to have a a chance of doing well, the time is required. You need to work every week. Ideally, you need to work at least an hour every day. Programming is closer to piano than physics. It requires practice on regular intervals.

Textbook

The CS Department chose Starting Out With Java by Tony Gaddis. The book is very good, but it is expensive. The class will follow the book fairly closely first nine chapters. Time does not allow us to explore much beyond that. If you cannot afford the text or are not willing to spend that much money, find a reasonable introductory Java book to use in its place. Jim will list the topics for the week on a separate web page so you can "read along" in your book. Used versions of Gaddis should be ok.

The web pages, lecture notes, and class will probably not provide sufficient information and background for doing all of the work. Some form of reference reading is important.

Programming Environment

We will NOT be using an IDE (Integrated Development Environment) such as Eclipse. You may do so if you wish but we will not talk about it or use it in class or recitation. We will use a text editor and the command line interface. The text editor is is your choice and the command line interface will be either a command prompt window on Windows or a terminal window on Macintosh. This will be explained and used during the first class and recitation meeting.

Classrooms and Time

Lecture is in 300 Old Engineering Hall on Monday and Wednesday from 3:00 pm to 4:15 pm.
Recitation is in 6110 Sennott Square on either Thrusday from 10:00 to 10:50 or Friday from 3 to 3:50.

Grading

Your course grade will be based on exams, labs, and programs.

- midterm exam - 25%
- final exam - 25%
- labs - 10%
- programs - 40%

The Grading Scale is:

- 90% or higher - A
- 80% - 89.9% - B
- 70% - 79.9% - C
- 60% - 69.9% - D
- Less than 60% - F

Weekly Reading Topics

This list of topics is for class members who are using a book other than Gaddis.

Week 12 Nov 11 Same as Weeks 10 and 11

Week 11 Nov 4 Gaddis Chap 7.9 pp 454-465

- Two dimensional arrays

Week 10 Oct 28 Gaddis Chap 6

- OOP - Object Oriented Programing

Week 9 Oct 21 Gaddis Chap pp: 451-452, 470-472 Chap 7.6, 7.7

- Arrays: swap, shuffle
- Arrays: insert and delete
- selection sort
- Arrays class
- ArrayList class

Week 8 Oct 15 Gaddis Chap 4 pp: 143-150; Chap 7 pp:445-448, 451-453; Chap 9 pp: 600-902

- Array of Strings
- split()
- equals( )
- compareTo()
- searching an array
- timmming an array
- growning an array

Week 7 Oct 7 Same as week 6

Week 6 Sep 30 Gaddis Chap 7.1-7.6

- Arrays
- Accessing array elements
- Arrays as arguments
- Returning arrays from methods
- Searching an array

Week 5 Gaddis Sep 23 Chap 5.1 - 5.6

- Methods
- Methods and arguments
- Local variables
- returning a value from a method

Week 4 Gaddis Sep 16 Chap 4.5 and pages 71-76 and 142-148

- The for loop
- Strings: charAt, equals, compareTo, toUpperCase, toLowerCase

---

Week 3 Gaddis Sep 9 Chap 4.1 - 4.4, 4.10 pp 441(last paragraph) to 452

- The while loop
- Scanner class (again)
- File class combined with the Scanner class
- Strategy for reading files

Week 2 Gaddis Sep 2 Chap 3.1 - 3.7

- The if statement
- The if/else statement
- Nested if statements
- Relational and logical operators
- Comparing String objects
- System.out.printf( ) method

Week 1 Gaddis Aug 26Chap 2.1 - 2.13

- Structure of a Java program
- The print() and println() methods
- Variables and literals
- Primitive data types
- Arithmetic operators
- Conversion between primitive types (casting)
- Creating constants
- The String class
- Comments
- Reading keyboard input with the Scanner class