

Note: Student work submitted as part of this course may be reviewed by Oxford College and Emory College faculty and staff for the purposes of improving instruction and enhancing Emory education.

CS 170Q - Introduction to Computer Science I Course Syllabus Spring 2012: Section 11A, Oser

Instructor: Paul Oser
Email: poser3@emory.edu
Office: Pierce 122A
Hours: (CS 170 only) 2:00 to 3:00 PM on M,Tu,Th,F and “Open-door” policy;
(all Math and CS courses) 3:00 – 6:00 PM on M,T,W,Th

Lecture: MWF 11:45 – 12:35 AM in Pierce 206
Lab: W 2:00 – 3:00 PM in Pierce 206

Web: http://mathcenter.oxford.emory.edu/wiki2/index.php/CS_170

Textbook: Java Concepts, 6th Edition by Cay Horstmann (reference only)
Other Materials: A 1GB (or bigger) USB Flash Drive is required.

Overview: This course is an introduction to computer science for the student who expects to make serious use of the computer in course work or research. Topics include: fundamental computing concepts, general programming principles, and the Java programming language. Emphasis will be on algorithm development with examples highlighting topics in data structures.

Goals for Student Learning: Students at the conclusion of this course should be able to...

- Effectively use primitive data types and pre-made objects in the Java language
- Effectively use program-flow-control concepts (i.e., "for"-loops, "while"-loops, "if"-statements, etc...)
- Effectively use arrays and strings for storing and manipulating a large amount of data
- Build classes and objects of their own design
- Effectively use subclasses and interfaces to facilitate Object-oriented design
- Begin to become familiar with event-driven programming

Prerequisites:

There are no prerequisites although some familiarity with email and web browsers will be helpful. Knowledge of high school algebra and basic problem solving skills are assumed. This course is the first of a two semester sequence for computer science majors and is followed by CS 171.

The “Ways of Inquiry” at Oxford:

“Ways of Inquiry” courses are designed to introduce students to the specific ways knowledge is pursued in each discipline through active engagement in the discipline's methods of analysis. INQ courses start with questions, are student-centered and often collaborative, and they place increasing responsibility on students for their own learning. Students not only experience each discipline's distinctiveness but also move beyond its boundaries to understand connections with other disciplines and fields.

The “Ways of Inquiry” Used in this Course:

Writing a computer program is an act of inquiry. There is no "recipe" that can be given to students so charged. On the contrary (and in a quite literal sense) – with every program they write, students "create their own recipe" to accomplish some given task.

Students will be given many opportunities (all of the lab activities, and many more opportunities during class) to write programs. They will have a goal (a task their program must perform); they will have the tools they need (the language specification and API); and how they get to that goal is up to them. Students will attack these tasks in a variety of ways, some elegant, some brutish, but all will have to pull up their sleeves and get down in the trenches of figuring out how this new thing can be done with what they know.

The instructor’s job in this course is to demonstrate the requirements and capabilities of the Java language, give students some good guiding principles, and then largely get out of their way – letting them discover how to use this language to do what they want to do.

The instructor will also play a supporting role: 1) helping students see how certain "fundamental" questions can guide their efforts in accomplishing the goals given to them; 2) driving students away from inefficient solutions; and 3) revealing to students (through questioning) cases they haven't considered, that might cause their program to behave in a manner contrary to what they intended – and alerting them to the need to debug when necessary.

Students will have the opportunity to pair up with other students as they engage in many of their programming tasks – and the tasks themselves will often connect in some way to interesting problems or subfields of mathematics, cryptography, and/or other disciplines.

Lab Work / Assignments:

There is one hour of scheduled lab time per week (W 2-3:00 pm in Pierce 206) but students should ***plan on spending considerably more time than that in front of a computer this semester*** (either in the lab or working from home). In the weekly lab, focus will be given to reinforcing concepts learned in lecture by writing programs. Students should make every effort to attend all of the labs, as programming is a skill best learned by “doing”. The Java software in Pierce 206 should also be available to you on the specialty computers in the Library and in the Kaleidoscope Lab, should you not have access to a computer of your own.

Assignments will involve designing, coding, testing and debugging programs based on a written assignment specification. These programs will involve a conceptual understanding of language features as well as requiring skill with various software tools. With programming it is important to “work smarter, not harder.” Brute-force approaches often lead to long, tedious, unsuccessful hours of work. The right approach can help you write correct, easy-to-understand and efficient code with minimal effort. Programs must be completed individually or as otherwise directed, although you are welcome to discuss general principles and concepts about the assignments with other students (and the instructor).

Exams:

There will be two closed-book midterm exams and a final that will test your conceptual understanding of the material and will require some attention to programming details. Doing well on the exams strongly correlates to reading and understanding the notes online and other reference material provided! Questions will often require writing or analyzing short bits of code.

Grading:

Lab Work / Assignments:	35%	(weighted average)
Midterms:	30%	
Final Exam:	25%	
Active Class Participation:	10%	

Late Policy:

Students are expected to be present for all scheduled tests. Any conflicts should be brought to the instructor's attention as soon as possible. If a legitimate reason exists for missing a test – as determined by the instructor – then the test must be taken prior to the regularly scheduled date. In the unusual circumstance where taking the test early is not possible, ***students should be aware that any make-up tests given will be designed to be more difficult to offset the additional time given for study.*** Students must provide written documentation in advance of any special accommodations required for testing. This includes additional time or other needs. The final exam cannot be rescheduled.

In general, late programming assignments will not be accepted; however, each student is granted two penalty-free extensions. These extensions must be requested before the project is due, and the length of the extension will be determined by the instructor at the time of the request, based upon the individual assignment being considered. The above policies will be waived only in an "emergency" situation with appropriate documentation and at the instructor's discretion.

Honor Code Policy:

All class work is governed by the Oxford College Honor Code. No collaboration is allowed on programming assignments unless you are asked to do a group assignment, and then collaboration should be limited to one's group members. Talking to other students about general Java features used in the programs is OK but ***students should not look at anyone else's code or allow anyone else to look at their code prior to submission.*** (Students should be aware that it is actually fairly easy to detect inappropriate collaboration or copying by running programs that analyze submitted programs.)

Every program project submitted must have the following comment included at the top of the file containing the respective "main()" method.

```
/*  
THIS CODE WAS WRITTEN IN ACCORDANCE WITH THE HONOR CODE POLICY FOR THIS  
COURSE. I HAVE NOT BEEN INVOLVED IN ANY INAPPROPRIATE COLLABORATION WITH  
RESPECT TO THE DEVELOPMENT OF THIS PROJECT. _Your_Name_Here_  
*/
```