Syllabus for CSCI 235 Programming I: Problem Solving Spring 2016

Cary G. Gray
Office: Meyer 159, x5875
Home: *omitted online*
E-mail: *omitted online*
Office hours: MF 1:30-2:30 p.m.
WTh 1:30-3:30 p.m.
and by appointment (esp. Thursday mornings)
TA: Johnny Edman *omitted online*

Class meetings

MWF 11:30 a.m.–12:35 p.m., Meyer 131
Lab: T 1:15–3:05 p.m., Meyer 154
Final exam: 8:00–10:00 a.m on Thursday, May 5

On-line resources

Additional (and updated) course information will be available at the class page at

http://cs.wheaton.edu/~cgray/csci235/

Textbook

Savitch, *Absolute Java*, fifth edition, Addison Wesley, 2012. (Fourth edition is also fine.)

Note that this textbook—a reference, actually—is also used for CSCI 245.

Description

This course is the first half of a two-semester introduction to programming, emphasizing the process of solving computational problems and expressing those solutions in an object-oriented language. Here is the official catalog description from this year's catalog:
CSCI 235 Programming I: Problem Solving
A first course in computer programming for beginners. Structured and object-oriented programming in Java or a similar programming language. Types, control structures, methods, and recursion; objects, classes, interfaces, encapsulation and polymorphism; exceptions, library classes, file I/O, linked lists, and graphical user interfaces.

Objectives

By the end of this class, you should be able to:

1. Design basic algorithms and organize their description:
   - organize data in terms of types and variables;
   - arrange expressions and statements in their logical order;
   - use the control structures of branching, iteration, and recursion; and
   - encapsulate functionality in methods.
2. Express algorithms in the Java programming language:
   - correctly use the features of the Java language to implement the concepts above; and
   - use the standard Java terminology in describing the parts and process of a computer program.
3. Practice the basics of object-oriented design:
   - design new types using classes and interfaces;
   - encapsulate data and functionality;
   - design the relationships among types.

Assignments

To help you achieve those objectives (and to determine how well you achieve them), there will be four kinds of assignments or graded work in this class.

Class Preparation It is vital that you come to class prepared. To help with that, you should expect there to be a *preparation assignment* for most class meetings. Answer the questions marked as *prep'* and hand in a (paper) copy before class begins. Bring a second copy of your answers to use during class, because it will provide the starting point for what we do together.

Prep assignments will be graded pass-fail: what is required on the prep questions is a *good-faith* effort to answer every question, based on the reading. If you do not (yet) fully understand the material, it is sufficient to explain what you think is the case and what you don't understand.

Labs Labs provide an opportunity to practice problem-solving and to develop your understanding of the Java language. Most of the Thursday lab sessions plus a few class meetings (as indicated on the schedule) will be spent working with a partner on a single computer in a practice known as *pair programming*. The *driver* controls the mouse and keyboard, entering and editing the program, while the *navigator* watches, catches mistakes, and thinks about how to test what is currently being programmed and about to approach the next task. You will periodically swap roles.

During labs, you will normally be using lab accounts (logged in for you) instead of your personal accounts. You will work with a different partner in each meeting. Because we have to form teams, it is especially important that you arrive on time and that you provide me advance notice when you will be absent.

You must turn in your progress on the lab no later than the scheduled ending time, but you will be permitted to resubmit additional work later.

Projects Projects are where you put the ideas from this class into practice, solving problems of increasing size as the semester progresses.

Exams In-class examinations, including the final, will evaluate your mastery of the course material. See the class calendar for the mid-term exam dates. Note that material in this class is naturally cumulative; so you can expect the same for exams.

Grading

Your grade in the course will reflect three factors.

Preparation Given the number of class meetings that are neither exam nor review days, you should expect 40–49 prep assignments during the semester. An upper bound on your course grade will be determined by the fraction of the prep assignments that you complete satisfactorily:

For a grade of at least	minimum prep assignments
A-	90%
B-	75%
C-	60%
D	50%

"Satisfactory" completion requires that you

- make a good-faith effort to answer every prep question, based on the reading;
- turn in a (paper) copy of those answers before class begins; and
- be present and participate as we work in class.

If you must miss class, any arrangement to drop that day from consideration must be negotiated in advance; allowance for illness or emergency will be made after the fact if you've provided appropriate notice.

Graded work Grades for labs, projects, and exams will be averaged with the following weights:

component	*weight*
mid-term exams (total)	4
final exam	3
labs	1
projects	8

Note that to get a particular grade for the course, you must meet the minimum for prep assignments in addition to achieving that average on the graded work.

Disruptive behavior Finally, your course grade may be lowered up to one letter for disruptive behavior, including recurrent tardiness.

Policies

Attendance You are responsible for what happens in class, whether you are present or not. If you are sick or must miss because of other school responsibilities, let me know in advance. If there is an emergency, let me know as soon as practical. (You will do well to think of this as practice for keeping a job. Note that you can reach me by email, and my office phone takes messages at all hours.) Being in class means being on time: arriving late is highly disruptive. If the door is locked when you arrive, do not disturb the class.

On-time arrival is doubly important for labs, because you will need to connect with a (new) partner each time.

This class meets immediately following chapel. You should not expect to be able to do anything (such as stop at CPO or print something in the lab) between chapel and class.

Labs and exams will happen as scheduled. Make-up or other alternative arrangements must be made in advance (or as required for emergencies, if notification is provided promptly).

Your presence in class calls for your full attention. That implies no use of networked devices to be virtually somewhere else. *If* you need to use a computer of some sort for access to your textbook or to take notes, you should do so with the networking turned off, and you are implicitly promising that you will not use it for any other purpose during class. Talk with me beforehand if you think you need to do this.

Academic integrity. Because you are encouraged to work together—especially during labs—and provide each other assistance, you do risk inadvertent plagiarism. Be cautious, especially when you ask for or provide assistance. Make sure that you don't let someone else do your work for you, and make sure that you don't do someone else's work.

Because projects are your individual work, what you must not share is code. You may not program together or watch each other programming either to give or receive help. You may, on the other hand, discuss the problem in abstract, work through examples, or even draw diagrams or sketch small bits of pseudocode. Sharing test cases is another good form of collaboration.

If you are stuck trying to understand a compiler error or a program misbehavior, you may ask each other for help. The rule is that you should not show each other *working* code, and you should not ask someone else to *fix* your code. When you give assistance, you should avoid giving the correction; instead, help the other student to better understand the problem or to find helpful material in the book.

There is a lot of code and other information available in books and on the Internet—and some of it is even correct. Looking up information is acceptable, but you should be careful to give credit to sources you use. As when you are working with each other, copying *code* is not permitted, nor is asking for solutions. If you inadvertently find a resource that is too helpful, be honest and acknowledge it.

I expect you to conduct yourself honestly in this course. When you submit work, you assert that it is your own. If you use an outside source or receive assistance, acknowledge it. Deliberate misrepresentation will result in no credit for the assignment; a second offense will result in failing the course. All offenses will be reported and are subject to college disciplinary action as well.

Note that handing in a preparation assignment is a claim that you are present for class. It would be dishonest to represent yourself as present when you are not.

Late projects Note that projects are due at 5:00 p.m. on the indicated date. You are granted two free late days during the semester; that translates to being two days late with one assignment or one day late with each of two. Don't waste that slack, because other late work will not be accepted.

Special circumstances and needs Wheaton College is committed to providing reasonable accommodations for students with disabilities. Any student with a documented disability needing academic adjustments is requested to contact the Academic and Disability Services Office as early in the semester as possible. Call 630.752.5941 or email jennifer.nicodem@wheaton.edu for further information.

Gender-neutral language For academic discourse, spoken and written, the faculty expects students to use gender inclusive language for human beings.

Acknowledgement CSCI 235 and 245 are both based on work by Dr. VanDrunen. Significant portions of this syllabus and many other class materials are his work or derived from his work.

Schedule

Here is an initial class schedule; additions and changes will show up online.

Date	Reading		Projects
Jan 11		Introduction to computing, algorithms	
12		*Lab 0: Intro to the lab*	
13		More algorithms	
15	1.1, 1.3	**Programming fundamentals** Java programs; strings	
18		*MLK Holiday*	
19		*Lab 1: First Java programs*	
20	1.2, 1.4, 2.1	Types, variables, expressions, statements	start Project 1 (due Feb 1)
22	2.2, 3.1, 3.2	The Boolean type; more strings	
25	3.3, 3.4	More control structures	
26		*Lab 2: More strings*	start Project 2 (due Feb 8)
27	6.1, 6.2 6.3	Arrays; switch statement	
29		*Lab 3: Arrays*	
Feb 1	6.4, 5.1	More arrays, multidimensional arrays; intro to methods Mercurial Library methods	Project 1 due
2		*Faculty Development Workshop*	
3	5.1	Methods	
5		*Lab 4: First methods*	
8		More with methods; sorting	Project 2 due start Project 3 (due Feb 19)
9		*Lab 5: Methods*	
10	11.1–3	Recursion	
12		*Lab 6: Introducing recursion*	
15		*Presidents Day*	
16		*Lab 7: Recursive methods*	
17		More recursion; classes as modules	start Project 4 (due Feb 29)
19		Review	Project 3 due
22	4.1	**Object-Oriented Fundamentals** Classes as types	
23		*Lab 8: First classes*	
24		**Exam**	
26	4.2, 4.3–4, 5.1–4	Classes, methods, encapsulation	start Project 5 (due Mar 4)
29	8.1, 13.1	Interfaces, subtype polymorphism	Project 4 due
Mar 1		*Lab 9: Subtyping*	
2		Interaction of objects	
4		More interaction examples	Project 5 due

Date	Reading		Projects
Mar 7–11		*Spring break*	
14		Simulation	start Project 6 (due Mar 28)
15		*Lab 10: Simulation*	
16	15.1	**Linked structures** linked lists	
18		*Lab 11: Linked lists*	
21	15.4	More lists, trees	
22		*Lab 12: Trees*	
23		More trees and lists	start Project 7 (due Apr 6)
25		*Good Friday*	
28	14.1, 16.1–2 5e/pp. 277–279 4e/pp. 271–273	**Applied Topics** collections Collections API More collections; iterators	Project 6 due
29		*Lab 13: Using collections*	
30	16.3	for-each; file I/O File I/O classes File I/O overview	
Apr 1		*Lab 14: More collections*	
4	17.1–2, 17.4–5 GUI classes	GUI, events	start Project 8 (due Apr 20)
5		*Lab 15: GUI event handling*	
6		Review	Project 7 due
8		**Exam**	
11	17.3	More GUI, layouts	start Project 9 (due Apr 27)
12		*Lab 16: More GUI*	
13	graphics 18.3–4	More GUI; graphics	
15		*Lab 17: Graphics*	
18		File I/O, linked structures, and exceptions	
19		*Lab 18: File I/O*	
20		Handling exceptions	Project 8 due
22		I/O and classes	
25		Review Example	
26		*Lab 19: Putting it all together*	
27		Review	Project 9 due
29		Review	
8:00–10:00am, Thurs 5 May		**Final exam**	