CSE 174 — Fall 2013

Fundamentals of Programming and Problem Solving

An introduction to computer programming in a contemporary language. Algorithm development, refinement, and problem solving approaches. Data types and control structures. Program debugging and testing. Interactive and file input/output. Single and multi-dimensional arrays. Simple searching algorithms. Introduction to classes, objects, and object-oriented programming.

Instructor:	Norm Krumpe, norm.krumpe@miamioh.edu
Phone:	529-0351
Office:	205L Benton Hall
Course website:	http://my.csi.miamioh.edu (agenda, handouts, homework submission, gradebook)
Office Hours:	Tuesdays: 1:00 – 3:00 p.m. Wednesdays: 8:30 – 11:30 a.m. Can't make these hours because of a conflict? I will work with you to arrange a time that fits my schedule and yours.
Open Lab:	16 Benton Hall
Text:	*Learning Processing*, Shiffman
Clicker:	TurningPoint's "ResponseCard NXT Clicker" or the older "ResponseCard XR clicker", (*or* ResponseWare license for iPhone/iPod Touch/Blackberry/Laptop/Smartphone...the app is free, but the license that will make it work in class is not free)
Software (free):	*Java SE Development Kit (JDK 7)* Mac users probably have this already Windows users can download for free from http://java.sun.com *Processing*, free from http://processing.org but only works AFTER you install JDK 7 *Dr. Java*, free from http://drjava.org but only works AFTER you install JDK 7

Learning Outcomes: By the end of this course, you should be able to...

1. Write programs in a contemporary programming language
2. Solve programming problems in a clean and robust fashion using a procedural approach
3. Use a contemporary programming environment
4. Trace and explain the flow of programs that include multiple data types, conditionals, loops, console and file I/O, and multiple methods
5. Identify, describe, and eliminate errors in programs
6. Read, understand, and communicate technical information
7. Write code and documentation that emphasizes readability and uses specified style guidelines
8. Understand and apply mathematical concepts and models
9. Communicate a main idea in a logical way with supporting evidence
10. Identify, analyze, and compare multiple solutions to problems

Important dates

Fri, Sep 13	Last day to drop this course (no grade assigned)
Thu, Sep 19	**Exam 1**
Thu, Oct 17	**Exam 2**
Mon, Oct 28	Last day to withdraw from this course (course grade will be "W")
Thu, Nov 14	**Exam 3**
Thu, Nov 27	Thanksgiving – No class
Fri, Dec 6	Last day of classes in Oxford
Thu, Dec 12	**Final exam 8:00 a.m. – 10:00 a.m.**

Labs and other in-class activities

- Roughly half of our meeting time each week will be devoted to labs and other hands-on activities.
- Lab assignments are to be completed during in-class time...they are not homework.
- Lowest lab assignment grade will be dropped.
- No make-ups for missed labs.

Exams

- There will be three mid-term exams, and one final exam.
- All exams are cumulative, closed-book.
- You may have one sheet of notes for each exam. This sheet must be handwritten (and it must be hand-written by *you*), and turned in at the end of your exam.
- **No make-ups for missed exams**. If you are absent for an exam, your grade for that exam will be zero.

Your grade will be determined as follows:

4 Exams	65%
Homework	25%
Labs/Participation/In-Class work	10%
Total	100%

If grade is at least...	but less than...	your grade will be...
92	100	A
90	92	A-
88	90	B+
82	88	B
80	82	B-
78	80	C+
72	78	C
70	72	C-
68	70	D+
62	68	D
60	62	D-
0	60	F

Topics

Basic Language Elements

- Types, operators, variables, constants
- Strings
- Type conversion

Input/Output

- Interactive input/output
- Sequential file input/output
- **Mouse and keyboard interaction**

Control Structures

- Sequence
- Selection (branching)
- Iteration (looping)

Testing and Debugging

- Compile-time vs. run-time errors
- Testing and debugging strategies
- Test case selection
- Use of a debugging tool

Using Standard Classes

- String, Random, ArrayList, Math, Scanner
- Instantiating objects and invoking their methods
- Using static and non-static data and methods

Methods

- Designing and implementing methods
- Method decomposition
- Variable lifetime and scope
- Parameter passing and return values

Problem solving

- Functional decomposition
- Designing methods

Arrays

- Single and two-dimensional arrays
- Arrays of objects

Algorithms

- Searching arrays: linear and binary
- Standard array processing algorithms
- Non-recursive sorting techniques
- Searching and sorting using library routines

Graphics and Animation

- **2D and 3D coordinate systems**
- **Vector-based vs. bitmap-based graphics**
- **Color representation**
- **Images processing**
- **Animation techniques**

Course policies:

E-mail and Announcements

Regularly check the course website (http://my.csi.miamioh.edu) and your *Miami* e-mail for announcements.

Attendance

- Course grade decreases by 4% for each absence beyond 2 (this is in addition to any deductions due to missed in-class labs/activities). This means, for example, that if your final grade in the course were a 95%, but you missed 4 days of class, your final grade would be reduced to 87% (95% - 4% - 4%).
- More than five minutes late = half absence,
- I reserve the right to drop you from the course if you miss more than 4 days
- In case of an absence:
  - inform me beforehand, if possible
  - submit on time any work that is due
  - understand that there are no make-ups for missed labs, quizzes, exams or other in-class work

Regarding the idea of "excused" absences, the following excerpt is from Miami's Student Handbook:

> "There are no University-recognized excused absences except for religious observances that require absence from a class session and other required class activities. Students must give written notification to their instructor within the first two weeks of class of the religious event that prohibits class attendance and the date that will be missed, if officially known."

Textbook (part of your participation grade):

Keep up with the reading. Work through the activities in each chapter. Answer clicker questions in class that are based on the reading. Answer online quiz questions. You generally do not need to bring your textbook to class, but there are times when you may want it with you as a programming reference.

Taking notes

I use in-class electronic presentations to give you basic information. These are not a substitute for taking notes.

- Take notes during lectures, because...
- ...lab activities will often depend on you to use what you wrote in your notes.
- "Good notes" does not mean "Write everything". Be selective.
- Focus on writing sample code, diagrams, "notes to self".

Participation

Being in the room when someone is talking does not count as participation. Here is how to participate:

- Keeping up with the reading
- Asking questions in class
- Volunteering answers to questions in class (clicker and non-clicker questions, in-class activities)
- Contributing to online forum discussions (asking good questions, helping provide answers, not posting code)
- **Helping other students during labs**

Clickers (part of your participation grade):

- We will start using clickers on the second day of class. Register your clicker by going to the course website, and using the link in the Blog entry.
- Your responses are part of your participation grade.
- Alternative to clickers: A ResponseWare license, which lets you use your iPhone or other smartphone.
- Have a friend with a clicker? Sharing is fine, as long as the friend is not in the same class.
- I assume that everyone will have a couple bad clicker days, and so I compensate for that in your participation grade.

Homework

- **Roughly 10-12 assignments, typically (but not always) programming assignments**
- **Can not be turned in late.**
- Start early...start early...start early...start early...start early.
- WRITE YOUR CODE ALONE...learn to help one another without sharing any code.

Computer Science and Software Engineering
Academic Integrity Expectations for
Individual and Group Problem Solving Assignments

The Department of Computer Science and Software Engineering is committed to maintaining strict standards of academic integrity. The department expects each student to understand and comply with the University's Policy on Academic Integrity: http://www.miamioh.edu/integrity and the undergraduate student handbook and graduate student handbook. **Students may direct questions regarding academic integrity expectations to their instructor or to the department chair.** All work submitted must be **original** for that class. Submitting the same project for two different classes is grounds for charging a student with academic misconduct unless prior written permission is received from **both** instructors.

"Problem Solving Assignments" are assignments that involve **programming, math, proofs, derivations, and puzzles**.
The purpose of a problem solving assignment is for you to develop the skills necessary to solve similar problems in the future. To learn to solve problems you must solve the problems and write your solutions independently.

It is worth reiterating that the important aspect of the assignment is that you actually create the solution from start to finish; simply copying a solution and then **understanding it after the fact is not a substitute** for actually developing the solution.

The notion of academic integrity can be confusing in courses with substantial problem solving because certain forms of collaboration and investigation are permitted, but you are still required to complete your assignment independently. The following scenarios are meant to help distinguish between acceptable and unacceptable levels of collaboration and research, but are not all-inclusive:

ACCEPTABLE:

- Consulting solutions from the current course textbook, but not from other published sources.
- Seeking help on how to use the programming environment such as the editor, the compiler, or other tools.
- Seeking help on how to fix a program syntax error or how a certain language feature works.
- Discussing strategies with a fellow student on how to approach a particular problem. This discussion should not include significant sections of completed work or source code (including printouts, email, viewing on a monitor). Discussions should begin with a clean sheet of paper and end with conceptual drawings and/or pseudo-code.

UNACCEPTABLE:

- Looking at another solution including those written by current students, past students, or outside sources such as code or solutions found on the Web, or in publications other than the current class textbook.
- Using another solution as a starting point and then modifying the code or text as your own work.
- Providing a copy of your solution or a portion of your solution, in any form (electronic, hard copy, allowing another student to view your code on a monitor), to another student.
- Giving or receiving code fragments to fix a problem in a program.

If you are stuck on a problem and you are tempted to search for a solution on the Web or to look at another student's solution **STOP** and email or ask your instructor for help.