Important Information

Class Website: Canvas (available through CIS)

Lectures: Mondays and Wednesdays 8:05-9:25a in L105 WEB

Labs (attendance required): Fridays 8:35-9:25a, 9:40-10:30a, 10:45-11:35a, and 11:50a-12:40p in L124 WEB

Instructor: D. Erin Parker, 3144 MEB, parker@cs.utah.edu


Lab supplies: ARDX v1.3 Experimentation Kit for Arduino (Uno) Detailed requirements on class website, order ASAP

Test Dates: Mark your calendar – tests may not be missed!
Test 1: Monday, September 25 (during class)
Test 2: Monday, November 6 (during class)
Final exam: Friday, December 15 8-10a

Final course grade: Programming assignments and Arduino projects 45%, Written homework and labs 15%, Test 1 and Test 2 25%, Final exam 15%, Failing test/exam average → failing course grade

Getting help: The class website has details on how to see TAs and the instructor outside of class, as well as how to post questions to the class forum and email to the course staff.

Course Description

Catalog description. CS 1410: Introduction to Object-Oriented Programming is the second course required for students intending to major in computer science or computer engineering. The course introduces the engineering and mathematical skills required to effectively program computers, as well as the range of issues confronted by computer scientists. A major theme of the course is the role of procedural and data abstraction in decomposing programs into manageable pieces. The course also introduces the concepts central to object-oriented programming (OOP).
Students will complete extensive programming exercises that involve the application of elementary software engineering techniques.

**Better description.** This course is a different approach to CS 1410. CS 1410-030 (or 040) covers the same concepts as other CS 1410 sections, but the emphasis is on programming in C/C++ for embedded systems. Embedded systems are simply computers that are part of a larger device, such as the computers found in TV sets, traffic light controllers, microwaves, portable music players, modems, printers, automobiles, cooperative robots, or even the Mars rover.

In this course, students explore the basics of programming in C++ and quickly move to more advanced C++ topics and object-oriented programming. Students use their C++ knowledge to build small hardware devices that are controlled by an Arduino prototyping board. The capstone of this course is an Arduino programming project during the last several weeks of the semester.

In this course, students also explore the roles of abstraction, algorithms, and problem solving in the design of larger software projects. Lastly, students learn about complexity analysis and software engineering practices that will aid them as they move on to more advanced studies in computer science.

*Fair warning:* The pacing in this class is brisk. Students should be aware that not all of the topics they need to know will be covered during lectures. Students should spend a considerable amount of time reading, studying, and experimenting outside of lecture.

Students taking this course should have completed (or currently be taking) a calculus course. CS 1030 (formerly 1400) is the prerequisite for CS 1410. Students who enter this class without taking CS 1030 must be aware that they are responsible for independently learning any material they missed.

Students who complete this course are ready for further study in computer science or computer engineering (CS 2420).

**Course Materials**

**Website.** The class website is a Canvas course available through CIS. *It is always under development,* with updates to the class schedule, lecture notes, laboratory exercises, assignment specifications, and more, occurring regularly. *It is critical that students become familiar with the class website right away and plan to visit it several times a week, at a minimum.*

**Textbook.** Regular reading assignments are made from *C++ for Everyone* (Horstmann, ISBN: 978-0-470-92713-7). This textbook is an introductory approach to C++ and perfect for beginners. Advanced students may prefer a different C++ reference. *Note:* If you do not buy this book, it is your responsibility to locate a different C/C++ reference.

**Pencil/pen and paper.** Students should bring a pencil/pen and paper to every lecture and lab session. In-class problem solving, in particular writing source code by hand, is used regularly to prepare students for tests. Laptops, tablets, phones, and other devices are not permitted during tests; likewise, they may not be used for in-class problem solving exercises.

**Lecture notes.** The instructor will often make use of typed notes, sample source code, and other materials during lecture. These items will be posted on the class website following the lecture;
however, such posted items may not represent completely the material covered in class. Students
who must miss class are strongly encouraged to check with a classmate.

**Laboratory practice.** Lab sections will meet on Fridays at the times listed in L124 WEB.
Students must attend the lab section for which they are registered, unless prior arrangements are
made with the instructor. New students should create a CADE lab account at https://webhandin.
eng.utah.edu/cade.

**Laboratory equipment.** To complete the hardware-based lab exercises and assignments, each
student must have an Arduino lab kit. CS 1410 uses the ARDX v1.3 Experimentation Kit for
Arduino (Uno R3). Instructions for purchasing this kit are at the end of this document, as well
as posted on the class website. All students must have a kit for use in the second lab session on
September 1. **Order your kit right away!**

**Student Evaluation**

**Programming assignments and Arduino projects.** The instructions for each assignment
and its due date are posted on the class website at least one week before it must be submitted.
Assignments are submitted online via the submission tool located beneath the instructions for
each assignment. As strange as it may seem, some students have trouble submitting assignments
electronically. It is the student’s responsibility to ensure the successful and timely submission of
each programming assignment — **start early and follow the instructions carefully.** Corrupted or
missing files will not be grounds for extensions. Double-check your submissions, and save a digital
copy of all of your work. **No assignments will be accepted late, except in the case of a documented
medical emergency.**

**Written homework and lab.** Problems and questions are given each week to test knowledge
not related to programming or to prepare students for the weeks lab activity. Written homework
must be completed before lab — print it, complete it, and bring it to lab to serve as your lab sheet.
Each week, the score earned for the written homework and the score earned for the lab exercise are
recorded as a combined score. **No written homework is accepted late, nor a lab session made up,
except in the case of a documented medical emergency.**

**Tests and final exam.** Tests will be given during class meetings on September 25 and November
6. The final exam is cumulative and will take place December 15 8-10a. **No test or the final exam
may be taken at a different time for any reason other than a documented medical emergency.**

**Final course grade.** **If the average score for Test 1, Test 2, and Final exam is 65% or lower,
the final course grade will be no higher than a D+.** Otherwise, the final course grade will be based
on programming assignments and Arduino projects (45%), written homework and labs (15%), Test
1 and Test 2 (25%), and Final exam (15%).

**Regrades.** Students who wish to appeal a score on an assignment, a written homework, or a
test must do so **within one week of receiving the score** and use the **Regrade Request Form**
posted on the class website.

**Drop scores.** Students may miss a deadline or a lab session for a reason that is not granted an
exception. Therefore, to allow for such an occurrence, the lowest score earned on a programming
assignment and the lowest score on a written homework / lab are dropped from the record of each
student at the end of the semester. Students should plan to use the “drop scores” judiciously — there is only one for a programming assignment and one for a written homework / lab. **No test or final exam scores are dropped.**

**Letter grades.** The following table is used to associate numerical scores with the corresponding letter grade.

<table>
<thead>
<tr>
<th>A</th>
<th>87 ≤ X &lt; 90</th>
<th>B+</th>
<th>77 ≤ X &lt; 80</th>
<th>C+</th>
<th>67 ≤ X &lt; 70</th>
<th>D+</th>
</tr>
</thead>
<tbody>
<tr>
<td>A-</td>
<td>90 ≤ X &lt; 93</td>
<td>B</td>
<td>83 ≤ X &lt; 87</td>
<td>C</td>
<td>63 ≤ X &lt; 67</td>
<td>D</td>
</tr>
<tr>
<td></td>
<td>80 ≤ X &lt; 83</td>
<td>B-</td>
<td>70 ≤ X &lt; 73</td>
<td>C-</td>
<td>60 ≤ X &lt; 63</td>
<td>D-</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>X &lt; 60</td>
<td></td>
<td></td>
<td>E</td>
</tr>
</tbody>
</table>

**How to Approach This Class**

Beginning students are often surprised by the amount of human effort that has to go into designing, writing, and testing a program. Complaints from students about the amount of time required by introductory computer programming courses are universal. You should expect to spend three hours outside of class for every hour that you spend in class. In other words, you should expect to spend twelve hours per week reading, studying, and developing programs. Many of you will spend more than that. Please keep this in mind when setting up your schedule for the semester!

In addition, take note of the following suggestions:

- Work near other students. Students who work at home alone are at a huge disadvantage on the assignments. Working in the lab allows you to ask other students and TAs for help understanding the material. Be careful, though, to respect the rules regarding plagiarism (given at the end of this document).

- Use Google or some other search engine to find information you do not have. Again, respect the rules regarding plagiarism and do not copy solutions from the web.

- Ask questions. If you find yourself stuck on a problem for more than thirty minutes, ask the teaching staff for help instead of spending hours searching for a solution. (Searching for hours for an answer cannot be credited as hard work — it is wasted time.)

- Give yourself time to think about the material. Plan on working on the assignments a little each day, and ask questions when you get stuck. This is an efficient way to reduce time spent and maximize learning. Do not plan on solving assignments all at once — it takes approximately three times as long for most students to “push through” the work in a single session.

**Getting Help**

To get help understanding course material, students may see the Teaching Assistants during **TA Consulting Hours**, see the instructor during **Office Hours**, post a question to the **Discussion Forum**, or send email to the course staff at teach-cs1410-030@list.eng.utah.edu. See the **How to Get Help** link on the class website for details.
Behavior in the Classroom

All students are expected to maintain professional behavior, according to the University of Utah Student Code at www.regulations.utah.edu/academics/guides/students/studentRights.html. Students should read the Code carefully and know that they are responsible for the content. According to Faculty Rules and Regulations, it is the faculty responsibility to enforce responsible classroom behaviors, beginning with verbal warnings and progressing to dismissal from class and a failing grade. Students have the right to appeal such action to the Student Behavior Committee. Students are expected to engage with the instructor and classmates during class meetings. Students are permitted to use a laptop or mobile device to take notes and/or as clickers. Use of a laptop or mobile device for any other purpose is not permitted, and students who do so will be asked to leave the classroom.