COURSE OVERVIEW

This course (combined with CS151) covers the entry level CS course requirements for the CS and ECE degree programs at Oregon State University. Alternatively students may take the CS161/162 sequence to meet this requirement.

ECE/CS 151 focuses on programming from the point of view of an embedded programmer working on low resource systems. This approach builds from the ‘bottom up’ rather than working from the ‘top down’ where large code bases can be used to do advanced functions quickly. The initial CS151 course covers the basics of C and its use on a light weight embedded platform. CS152 continues expanding experience in C and adds an object oriented language element to introduce higher level concepts. The motivation for the object orientated programming is based on usability rather than learning object orientated concepts for their own sake.

COURSE LEARNING OUTCOMES

At the completion of the courses, students will be able to perform the following tasks:

1. Write C code for several moderately complex embedded projects
2. Interface and control an embedded controller from a PC using custom written code on both ends
3. Show several methods for debugging code on embedded and PC targets
4. Have made a simple GUI front end

ABET OUTCOMES

1. **Design** and **implement** programs that require
   a. multiple classes, structures
   b. hierarchies of classes that use inheritance and polymorphism
   c. an understanding of abstraction, modularity, separation of concerns, and exception handling
2. **Construct** and **use** basic linear structures (arrays, stacks, queues, and various linked lists) in programs, and be able to **describe** instances appropriate for their use
3. **Write** an object-oriented program that efficiently communicates to another device, using multiple classes, methods, and objects
4. **Develop** test-data sets and testing plans for programming projects
5. **Produce** recursive algorithms, and **choose** appropriately between iterative and recursive algorithms.
6. **Using** embedded control, actuate motors, servos, displays, and/or other traditionally embedded devices.
7. **Classify** moderately complicated algorithms in these complexity classes: \(O(1), O(\log n), O(n), O(n \log n),\) and \(O(n^2)\)
Academic Dishonesty

At Oregon State University academic dishonesty is defined by the Oregon Administrative Rules 576-015-0020.1.a-c as: 
An intentional act of deception in which a student seeks to claim credit for the work or effort of another person or uses unauthorized materials or fabricated information in any academic work.

Academic dishonesty includes:

• **CHEATING** - use or attempted use of unauthorized materials, information or study aids or an act of deceit by which a student attempts to misrepresent mastery of academic effort or information. This includes unauthorized copying or collaboration on a test or assignment or using prohibited materials and texts.

• **FABRICATION** - falsification or invention of any information (including falsifying research, inventing or exaggerating data and listing incorrect or fictitious references.

• **ASSISTING** - helping another commit an act of academic dishonesty. This includes paying or bribing someone to acquire a test or assignment, changing someone's grades or academic records, or taking a test/doing an assignment for someone else (or allowing someone to do these things for you). It is a violation of Oregon state law to create and offer to sell part or all of an education assignment to another person (ORS 165.114).

• **TAMPERING** - altering or interfering with evaluation instruments and documents.

• **PLAGIARISM** - representing the word or ideas of another person as one's own OR presenting someone else's words, ideas, artistry or data as one's own. This includes copying another person's work (including unpublished material) without appropriate referencing, presenting someone else's opinions and theories as one's own, or working jointly on a project, then submitting it as one's own.

PLEASE NOTE: We will be checking code that is supposed to be done individually using a program called MOSS that automates the process of finding direct code copying. We also reference previous courses submissions. You MUST write your own code.

IEEE Code of Ethics

As a community of Electrical and Computer Engineers, we have a duty to present ourselves and our profession to each other and the public in the best light possible. The IEEE has a code of Ethics that should always be considered. It reads:

“We, the members of the IEEE, in recognition of the importance of our technologies in affecting the quality of life throughout the world, and in accepting a personal obligation to our profession, its members and the communities we serve, do hereby commit ourselves to the highest ethical and professional conduct and agree:

1. to accept responsibility in making decisions consistent with the safety, health and welfare of the public, and to disclose promptly factors that might endanger the public or the environment;
2. to avoid real or perceived conflicts of interest whenever possible, and to disclose them to affected parties when they do exist;
3. to be honest and realistic in stating claims or estimates based on available data;
4. to reject bribery in all its forms;
5. to improve the understanding of technology, its appropriate application, and potential consequences;
6. to maintain and improve our technical competence and to undertake technological tasks for others only if qualified by training or experience, or after full disclosure of pertinent limitations;
7. to seek, accept, and offer honest criticism of technical work, to acknowledge and correct errors, and to credit properly the contributions of others;
8. to treat fairly all persons regardless of such factors as race, religion, gender, disability, age, or national origin;
9. to avoid injuring others, their property, reputation, or employment by false or malicious action;
10. to assist colleagues and co-workers in their professional development and to support them in following this code of ethics. “
Late Work Policy
All late work will receive no credit. Only pre-discussed exceptions will be accepted.

Grading Break Down:
Labs: 40%
Assignments: 20%
Midterms: 30%
Final Project: 10%

Submitting Assignments:
Some assignments are to be submitted electronically. These submissions should be done via TEACH. Additionally students should place a copy of their files on the ENGR server when they submit to TEACH. This is to ensure in the event of a ‘computer glitch’ that there are two copies of the code on the server with proper time stamps. Some assignments will be reviewed automatically. For these assignments, the first submission will be graded and your grade sent to you via email. If you are not satisfied with the score, you may improve your code and resubmit so long as it is before the assignment deadline. The second submission will be for full credit. Each following submission will lose 10% of its score.

Lab Submissions:
Most labs consist of a Prelab, Lab, and study question sections. When included, prelabs sections must be typed and submitted to your TA within 5 minutes of the start of your lab. Study questions are due the week following the lab within the same 5 minutes of the next lab. All labs should be demonstrated to your TA at the beginning of lab the following week. You may only show your lab to your TA. When you demo your code, you should have a printed copy of the code on hand to give to your TA. Your TA will quickly verify the handed in code matches the code compiled and demonstrated.

Course Schedule:
The course schedule and details of each lecture is available on the ‘Google™ calendar.’ While no changes are expected, please review it periodically to double check.