CS 165 – Accelerated Introduction to Computer Science

Catalog Description: Overview of the fundamental concepts of Computer Science. Introduction to problem solving, algorithm development, data types, and basic data structures. Introduction to analysis of algorithms and principles of software engineering. System development and computer programming using imperative / object-oriented paradigms. (Note: Students may take either the CS 161/162 sequence or CS 165, but cannot receive credit for both.)

Credits: 8   Terms Offered: All

Prerequisites: Enforced: (MTH 112 or MPAL >= 75)
Unenforced: CS Double Degree students: BA/BS and prior programming coursework or practicing coding.

Courses that require this as a prerequisite: CS 261, CS 271, CS 290, CS 352

Structure:
Ecampus: Term totals: This course combines approximately 240 hours of instruction, online activities, and assignments for 8 credits (45 hours of online instruction, 15 hours of online participation, 5 hours of online quizzes, 60 hours of offline reading/study, 40 hours of offline homework/lab assignments, 70 hours of offline programming projects, and 5 hours of proctored exams).

Instructors: TBA

Course Content:
- Problem-solving methods
- Modularization, design principles
- Primitive and structured data types
- Control structures (procedures/parameters, decision, iteration, recursion)
- Abstract linear data structures
- Object-oriented design/implementation
- Error/exception handling
- Algorithm analysis
- Software testing
- Software quality factors

Learning Resources: One or more of the following:
- Building Java Programs (2nd edition), Reges/Stepp
- Big Java (4th edition), Horstmann
- Absolute C++ (5th edition), Savitch
- Programming and Problem Solving with C++ (5th edition), Dale/Weems
- Additional online resources.
Measurable Student Learning Outcomes:

At the completion of the course, students will be able to…

1. **Translate** natural language expressions into appropriate arithmetic, relational, and logic expressions (Level 3; ABET Outcomes: i)

2. **Design** and **implement** programs that require
   a. multiple classes, structures, and control mechanisms
   b. hierarchies of classes that use inheritance and polymorphism
   c. an understanding of abstraction, modularity, separation of concerns, and exception handling
   (Level 4; ABET Outcomes: b, C, I, k)

3. **Construct** and **use** basic linear data structures in programs, and be able to **describe** instances appropriate for their use. (Level 4; ABET Outcomes: A, B, C, I, K)

4. **Produce** recursive algorithms, and **choose** appropriately between iterative and recursive algorithms. (Level 3; ABET Outcomes: A, j)

5. **Develop** test-data sets and testing plans for programming projects (Level 4; ABET Outcomes: A, c, I, K, L)

6. **Classify** moderately complicated algorithms in these complexity classes:
   O(1), O(log n), O(n), O(n log n), and O(n^2) (Level 3; ABET Outcomes: A, J)

7. **Describe** the relationship between the software engineering design principles and software quality (Level 1; ABET Outcomes: I, k)

Evaluation of Student Learning:

- programming projects (approx. 35%)
- other homework (approx. 10%)
- quizzes (approx. 10%)
- midterm exams (approx. 25%)
- final exam (approx. 20%)

Students with Disabilities:

Accommodations for students with disabilities are determined and approved by Disability Access Services (DAS). If you, as a student, believe you are eligible for accommodations but have not obtained approval please contact DAS immediately at 541-737-4098 or at [http://ds.oregonstate.edu](http://ds.oregonstate.edu). DAS notifies students and faculty members of approved academic accommodations and coordinates implementation of those accommodations. While not required, students and faculty members are encouraged to discuss details of the implementation of individual accommodations.

Link to Statement of Expectations for Student Conduct, i.e., cheating policies

Created: Spring 2012
Revised: Winter 2018