CS 007B: COMPUTER SCIENCE II

Originator
mflora

Justification / Rationale
Updating dates

Effective Term
Spring 2022

Credit Status
Credit - Degree Applicable

Subject
CS - Computer Science

Course Number
007B

Full Course Title
Computer Science II

Short Title
COMPUTER SCIENCE II

Discipline

Disciplines List
Computer Science

Modality
Face-to-Face
100% Online
Hybrid

Catalog Description
This second course in computer science introduces more advanced topics in programming. Students will use modularity to develop solutions for larger-scale programming problems. Recursion, file processing, and object-oriented programming are implemented. This course will be taught using the C++ programming language.

Schedule Description
Develop solutions for larger-scale programming problems using recursion, file processing, and object-oriented programming. This course uses C++. Prerequisite: CS-007A and MATH-012

Lecture Units
2

Lecture Semester Hours
36

Lab Units
1

Lab Semester Hours
54

In-class Hours
90

Out-of-class Hours
72

Total Course Units
3
Total Semester Hours
162

Prerequisite Course(s)
CS 007A & MATH 012

Required Text and Other Instructional Materials

Resource Type
Book

Formatting Style
MLA

Author
Gaddis, Tony

Title
Starting Out with C++ from Control Structures to Objects

Edition
9

Publisher
Pearson

Year
2018

College Level
Yes

ISBN #
9780134498379

Resource Type
Book
Open Educational Resource
No

Formatting Style
MLA

Author
Vahid, Frank

Title
Programming in C++

Publisher
zyBooks

Year
2021

College Level
Yes

Class Size Maximum
28

Entrance Skills
Demonstrate proficiency with using programming development environment.

Requisite Course Objectives
CS 007A-Demonstrate effective use of a program development environment by entering/editing and executing programs.

Entrance Skills
Demonstrate proficiency with using fundamental types, variable assignments, and arithmetic expressions.

Requisite Course Objectives
CS 007A-Use the basic syntax and semantics of C++ to declare and define variables of specific types, and to form expressions, and assign these expressions to other variables of compatible types using simple Input/Output with conditional and iterative control structures.

Entrance Skills
Demonstrate proficiency with the use, design, and implementation of user-defined functions.

Requisite Course Objectives
CS 007A-Design functions using structured decomposition/modularization and pseudo-code with the "check/expect" model of development that checks that all input expressions evaluate to the value of the expected output expression.

Entrance Skills
Demonstrate proficiency with the design and implementation of for and while loops.

Requisite Course Objectives
CS 007A-Design and implement various types of loops appropriate to a given problem.

Course Content

I. Programming Fundamentals (PF) PF3. Fundamental data structures

Topics

1. Primitive types
2. Arrays
3. Records
4. Strings and string processing
5. Data representation in memory
6. Static, stack, and heap allocation
7. Runtime storage management
8. Pointers and references
9. Linked structures
10. Strategies for choosing the right data structure
11. Introduce basic concepts related to implementation strategies for stacks, queues, and hash tables
12. Introduce basic concepts related to implementation strategies for trees

PF4. Recursion

Topics

1. The concept of recursion
2. Recursive mathematical functions
3. Simple recursive procedures
4. Divide-and-conquer strategies

5. Recursive backtracking
6. Implementation of recursion

II. Programming Languages (PL) PL4. Declarations and types

Topics

1. The conception of types as a set of values together with a set of operations
2. Declaration models (binding, visibility, scope, and lifetime)
3. Overview of type-checking
4. Garbage collection

PL5. Abstraction Mechanisms

Topics

1. Procedures, functions, and iterators as abstraction mechanisms
2. Parameterization mechanisms (reference vs. value)
3. Activation records and storage management
4. Type parameters and parameterized types - templates or generics
5. Modules in programming languages

PL6. Object-oriented programming

Topics

1. Object-oriented design
2. Encapsulation and information-hiding
3. Separation of behavior and implementation
4. Classes and subclasses
5. Inheritance (overriding, dynamic dispatch)
6. Polymorphism (subtype polymorphism vs. inheritance)
7. Class hierarchies
8. Collection classes and iteration protocols 9. Internal representations of objects and method tables

III. Software Engineering (SE) SE1. Software design

Topics

1. Fundamental design concepts and principles
2. Design strategy

Lab Content

1. Complete programming assignments incorporating design elements and code.
2. Consult with the teacher and classmates in small groups to tackle special programming tasks that arise as part of (1), above.

Course Objectives

	Objectives
Objective 1	Write programs that use each of the following data structures: arrays, strings, and structs.
Objective 2	Implement, test, and debug simple recursive functions and procedures
Objective 3	Evaluate tradeoffs in lifetime management (reference counting vs. garbage collection)
Objective 4	Explain how abstraction mechanisms support the creation of reusable software components
Objective 5	Design, implement, test, and debug simple programs in an object-oriented programming language
Objective 6	Compare and contrast object-oriented analysis and design with structured analysis and design

Student Learning Outcomes

	Upon satisfactory completion of this course, students will be able to:
Outcome 1	Create programs which use standard C++ language features, including functions, arrays, arrays of arrays, pointers, pointer arithmetic, dynamic memory allocation, structured data (structs).
Outcome 2	Design and implement an abstract data type using a class with member variables, member functions, constructors, and a destructor.
Outcome 3	Design and implement modular programs, created in appropriate .h and .ccp files, that use multiple classes with inheritance relationships, friend functions, friend classes, and operator overloading, using modern C++ language features and STL vectors and iterators where appropriate.

Methods of Instruction

Method	Please provide a description or examples of how each instructional method will be used in this course.
Laboratory	Students will practice developing the design principles introduced in lecture by writing programs that solve problems of varying difficulty. Typically, students may be assigned to work either individually or in small groups to address the problem of writing code to accept input in a specific format and analyze that input produce a desired output.
Lecture	Programming design practices and principles are introduced in concept and by example.
Collaborative/Team	Take turns role-playing as designer/tester/developer in solving programming challenges to produce software meeting prescribed input/output specification.

Methods of Evaluation

Method	Please provide a description or examples of how each evaluation method will be used in this course.	Type of Assignment
Mid-term and final evaluations	There will be a midterm and a final exam, generally in written format, but this may be combined with some computer work. (4 hrs)	In Class Only
Tests/Quizzes/Examinations	There will be a sequence of short quizzes to gauge student understanding of new concepts. (3 hrs)	In and Out of Class
Group activity participation/observation	Three or more major projects encompassing at least two weeks for development of complex solutions to complex tasks. Typical problems involve an assignment such as implementing an object-oriented using appropriate classes and objects as described in project specification.	In and Out of Class
Laboratory projects	These will require students to solve problems from the their lab manuals while using object-oriented programming concepts introduced in lecture. (4 hrs/wk)	In Class Only

Assignments

Other In-class Assignments

1. Participate in discussion.
2. Develop original programs to solve given problems.

Other Out-of-class Assignments

1. Read the text. (2 hrs/wk)
2. Write descriptions of programs in pseudocode. (0.5 hrs/wk)
3. Finish unfinished lab work. (2 hrs/wk)
4. Take quizzes. (0.5 hrs/wk)

Grade Methods

Letter Grade Only

Distance Education Checklist

Include the percentage of online and on-campus instruction you anticipate.

Online %

100

On-campus %

0

What will you be doing in the face-to-face sections of your course that necessitates a hybrid delivery vs a fully online delivery?

Although the course can be offered entirely online, it may also be offered hybrid to take advantage of collaboration activities that are more suited to in-person interaction.
Examinations can be given in a controlled location.

Lab Courses

How will the lab component of your course be differentiated from the lecture component of the course?

Lab assignments involve more interaction. For example, they may require students collaborate with a classmate, utilize a tutoring resource, or interview someone who is not part of the course.

From the COR list, what activities are specified as lab, and how will those be monitored by the instructor?

Lab activities are discussions and assignments that involve solving problems or exploring concepts with other students, with people not part of the course, or under the guidance of the professor or instructional support assistant. Discussions and other assignments that are completed in Canvas are monitored and evaluated by the professor. Assignments that do not take place in Canvas are evaluated by the professor based on write-ups (which may include summaries and feedback from the participants). Anonymous and non-anonymous feedback opportunities will be available to students to allow the professor further monitor effectiveness and appropriateness of activities that take place somewhere other than on the course LMS.

How will you assess the online delivery of lab activities?

Reports and other forms of write-ups will be submitted on the course LMS for evaluation and feedback.

Instructional Materials and Resources

If you use any other technologies in addition to the college LMS, what other technologies will you use and how are you ensuring student data security?

Depending on the textbook used, the professor may choose to use Pearson MyLab and Mastering, zyBooks, WebAssign, Replit, or GitHub. All of these are considered to be safe for use in education for both faculty and students. All can also be integrated with the college LMS (Canvas), which decreases the amount of times students will need to sign-in-and-out of accounts and open them up to data breaches.

If used, explain how specific materials and resources outside the LMS will be used to enhance student learning.

Professors who choose to use Pearson MyLab and Mastering, zyBooks, WebAssign, Replit, or GitHub do so in order to assign pre-written or instructor-created problems that are more complicated than those that can be created in Canvas while still receiving instantaneous feedback.

Effective Student/Faculty Contact

Which of the following methods of regular, timely, and effective student/faculty contact will be used in this course?

Within Course Management System:

Chat room/instant messaging
Discussion forums with substantive instructor participation
Online quizzes and examinations
Private messages
Regular virtual office hours
Timely feedback and return of student work as specified in the syllabus
Weekly announcements

External to Course Management System:

Direct e-mail
Posted audio/video (including YouTube, 3cmediasolutions, etc.)
Synchronous audio/video
Telephone contact/voicemail

For hybrid courses:

Scheduled Face-to-Face group or individual meetings

Briefly discuss how the selected strategies above will be used to maintain Regular Effective Contact in the course.

Faculty will regularly contact students individually and as a group through Canvas messages and/or COD email. Students will also receive regular announcements with information about the course, COD as a whole, or other relevant information.
In discussions and through other lab assignments, students will communicate with each other and their professor regularly and frequently.

If interacting with students outside the LMS, explain how additional interactions with students outside the LMS will enhance student learning.
Students may prefer to contact their professor via email or on the phone, which allows for an improved experience for those who communicate better in those contexts. The professor may direct students to access free supplemental resources as well.

Other Information

Comparable Transfer Course Information

University System
UC
Campus
UC San Diego

Course Number
CSE 8B
Course Title
Introduction to Programming and Computational Problem-Solving II

Catalog Year
2021

University System
UC
Campus
UC Los Angeles

Course Number
COM SCI 32
Course Title
Introduction to Computer Science II

Catalog Year
2021

University System
UC
Campus
UC Riverside

Course Number
CS 10B
Course Title
C++ Programming II

Catalog Year
2021

University System
CSU
Campus
CSU Fullerton

Course Number
CPSC 121

Course Title
Object-Oriented Programming

Catalog Year
2021

MIS Course Data

CIP Code
11.0701 - Computer Science.

TOP Code
070600 - Computer Science (transfer)

SAM Code
E - Non-Occupational

Basic Skills Status
Not Basic Skills

Prior College Level
Not applicable

Cooperative Work Experience
Not a Coop Course

Course Classification Status
Credit Course

Approved Special Class
Not special class

Noncredit Category
Not Applicable, Credit Course

Funding Agency Category
Not Applicable

Program Status
Program Applicable

Transfer Status
Transferable to both UC and CSU

General Education Status
Y = Not applicable

Support Course Status
N = Course is not a support course

C-ID
COMP 132

Allow Audit
No

Repeatability
No

Materials Fee
No

Additional Fees?
No

Approvals

Curriculum Committee Approval Date
11/18/2021

Academic Senate Approval Date
12/9/2021

Board of Trustees Approval Date
01/21/2022

Chancellor's Office Approval Date
11/18/2017

Course Control Number
CCC000587414

Programs referencing this course
Engineering AS Degree (http://catalog.collegeofthedesert.eduundefined/?key=24)
Liberal Arts: Business and Technology AA Degree (http://catalog.collegeofthedesert.eduundefined/?key=27)
Liberal Arts: Math and Science AA Degree (http://catalog.collegeofthedesert.eduundefined/?key=29)
Computer Science AS-T Degree (http://catalog.collegeofthedesert.eduundefined/?key=35)