Master Course Syllabus
School of Engineering and Computer Science
Washington State University Vancouver
CS 121
Program Design and Development
4 Semester Hours
(3 lecture hours, 3 laboratory hours)

Catalog Description

Formulation of problems and top-down design of programs in a modern structured language for their solution on a digital computer.

Prerequisite Courses

- Math 108, 171, 172, 182, 201, 202, 206 or 220, each with a C or better,
- A minimum ALEKS math placement score of 78%

Prerequisite Topics

- Algebra and linear equations
- Mathematical notation and formalism
- Polynomial, trigonometric, logarithmic and exponential functions

WSU Course Outcomes

Students taking this course will:

1. Analyze and explain the behavior of simple computer programs involving fundamental programming constructs (WSU Goal 2)
2. Design, code, test and debug programs that use the following fundamental programming constructs: basic computation, simple I/O, conditional and iterative control structures, composite data types and function definition and invocation. (WSU Goal 7)
3. Apply techniques of structured decomposition to break a problem into multiple program components. (WSU Goal 2)

Required Textbooks

- H. M. Deitel and T. J. Deitel, C., *How to Program*, Prentice Hall.

or

- Mark Guzdial, *Introduction to Computing and Programming in Python*, Pearson Prentice Hall.

or

- Bradley Miller and David Ranum, *Problem Solving with Algorithms and Data Structures using Python*, Franklin, Beedle & Associates.

or

- Jennifer Cambell, Paul Gries, Jason Montojo and Greg Wilson, *Practical Programming: An Introduction to Computer Science Using Python*, Pragmatic Bookshelf.

Reference Material

None

Major Topics Covered in the Course

1. Data storage, data types, and variables
2. Operators, expressions and statements
3. Control structures and control flow design
4. Concepts of modularity, encapsulation, information hiding
5. Problem partitioning and decomposition; top-down design
6. Functions, returned values and parameter lists
7. Recursion and recursive algorithms
8. Arrays and indexing
9. Searching and sorting
10. Code tracing and debugging
11. Simple graphical programming

Projects

All programming projects and assignments are to be derived and developed by students individually. In this course, students do not perform assignments as members of teams.
Programming projects will be assigned approximately weekly with the expectation that laboratory time will be used to enable students to individually engage with the instructor or lab assistant to understand and work through programming issues.

Programming Project Area	Weeks
(Small Assignments, See Below)	

Design, Implementation and Analysis

This course requires the student to craft 10-12 small, correctly functioning computer programs. The requirements for each program will necessitate that the student comprehend and apply mathematical knowledge to design and implement programs. These programs range from 10-20 lines of code at the beginning of the semester, to a final assignment whose design entails ~200 lines of program code. Examinations require students to correctly analyze and predict the functioning of small computer programs given by the instructor.

CS2013

This course provides coverage of CS2013 knowledge areas. Values listed are minimum course hours dedicated to the topic, percentages indicate the fraction of CS2013 knowledge area topics covered (acceptable values are: <25%, 25-75%, >75%, or 100%).

Area	Tier 1	Tier 2	Elective
AL/Fundamental Data Structures and Algorithms	4 (<25%)		
PL/Object-Oriented Programming	2 (<25%)		
PL/Functional Programming	2 (<25%)		
PL/Event-Driven Programming		2 (<25%)	

Area	Tier 1	Tier 2	Elective
SDF/Algorithms and Design	8 (25-75%)		
SDF/Fundamental Programming Concepts	10 (>75%)		
SDF/Fundamental Data Structures	5 (<25%)		
SDF/Development Methods	5 (<25%)		

Course Coordinator:	Ben McCamish
Last Updated:	January 24, 2022
Syllabus Version Number:	2.2