Master Course Syllabus
School of Engineering and Computer Science
Washington State University Vancouver
CS 261
C and Assembly Language Programming
3 Semester Hours
(3 lecture hours)

Catalog Description

C language concepts, professional practices and C programming, module linkage; assembly language concepts and programming.

Prerequisite Courses

- CS 122 with a C or better
- Credit not granted for both CS 251 and CS 261

Prerequisite Topics

- Proficiency using an imperative programming language
- Knowledge of computer instruction set architecture
- Experience with computer data representation
- Knowledge of computer memory and I/O architecture

Measured Course Outcomes

Students taking this course will:

1. Make effective use of a debugger or analysis tool to find errors in a program (Contributes to performance criterion 2-c.)
2. Design, code, analyze, or debug programs in the C programming languages, utilizing arrays, pointers, or dynamic memory (Contributes to performance criterion 6-c.)

Covered Course Outcomes

Students taking this course will also:

1. Design, code and debug an assembly language program which implements the function calling conventions of a high or mid-level language, such as C. (Relevant to performance criterion 6-d.)

Recommended Textbooks

- *Assembly Language Step-by-step: Programming with Linux*; Jeff Duntemann Wiley and Sons.

Reference Material

- C Programming; K. N. King.
- Programming in C; Stephen G. Kochan.

Major Topics Covered in the Course

1. Compilation and Module Linkage concepts: a) using the compiler and build management; b) library linkage; c) debugging
2. C syntax and semantics: a) data storage, types, and variables; b) operators, expressions and statements; c) control structures and control flow design; d) control flow testing and test documentation; e) functions, returned values and parameter lists; f) arrays, character arrays and strings
3. The C standard library: a) standard library character and string functions; b) file input/output
4. Assembly Language: a) registers and operations; b) memory addressing and access; c) subprograms and stacks; d) communicating with C programs
5. Dynamic memory management and self-referential data structures

Projects

Programming Project Area	Weeks
(Small Assignments, See Below)	

Design, Implementation, and Analysis

This course requires the student to implement 8 - 10 correctly functioning computer programs. Some programs will require the student to design a solution which satisfies a problem statement, while others will provide a design the student must implement. Programs address, minimally, developing a dynamic, self-referential data structure with C, calling assembly language functions from C, and comprehensive use of assembly language.

This course also requires students to analyze existing programs to locate bugs or determine the purpose of code.

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
AR/Machine Level Representation of Data		1 (25-75%)	
AR/Assembly Level Machine Organization		7 (25-75%)	
AR/Memory System Organization and Architecture		2 (<25%)	
PL/Advanced Programming Constructs			1 (<25%)
SDF/Fundamental Programming Concepts	3 (>75%)		
SDF/Fundamental Data structures	3 (25-75%)		
SDF/Development Methods	3 (25-75%)		
SE/Tools and Environments		3 (25-75%)	

Course Coordinator:	Paul Bonamy
Last Updated:	January 24, 2022
Syllabus Version Number:	2.2