Lecture Contact Hours: 48-54, Homework Hours: 96-108, Total Student Learning Hours: 144-162
Laboratory Contact Hours: 48-54, Homework Hours: 0, Total Student Learning Hours: 48-54

CUYAMACA COLLEGE
COURSE OUTLINE OF RECORD

COMPUTER SCIENCE 165 – ASSEMBLY LANGUAGE AND MACHINE ARCHITECTURE

3 hours lecture, 3 hours laboratory, 4 units

Catalog Description
This introductory course covers organization and behavior of real computer systems at the assembly-language level. Topics covered include number theory, registers, memory, CPU, linkers, debuggers, basic language syntax and high-level language/operating system interface. This course is intended for persons with a prior background in any other programming language and will emphasize those applications not easily performed using higher-level languages.

Prerequisite
"C" grade or higher or "Pass" in CS 181 or CS 182 or equivalent, or experience programming in C/C++ or Java

Entrance Skills
Without the following skills, competencies and/or knowledge, students entering this course will be highly unlikely to succeed:
1) Identify common tools, functions, and utilities available in modern programming languages.
2) Define and utilize structured programming principles.
3) Utilize the three basic logic control structures of sequence, decision and repetition.
4) Understand arrays.
5) Trace the flow of execution and values of all variables through a simple program.

Course Content
1) System architecture and overall operation.
2) Numbering systems: decimal, binary, and hexadecimal.
3) I/O units, operation and data representation.
4) Coding systems used.
5) CPU architecture.
6) I/O bus structure.
7) Registers, instructions and data flow.
8) Instruction formats, machine and assembler.
9) Internal data storage representation.
10) Basic instructions.
11) Loaders and linkages.
12) Running the assembler.
13) Segments: stack and stack frame, code, data.
14) High level language interface (such as C).
15) Public ad external data in both assembler and high level language.
16) Public procedures and functions in both assembler and high level language.
17) Parameter passing between assembler and high level language in both directions.
18) Macro commands and macro development both assembler and high level language.
19) Advance I/O.

20) Timing considerations.
21) More advanced instructions for specialized functions.
22) The relationships between high-level languages and assembly level programming.

Course Objectives
Students will be able to:
1) Analyze from a software and user's point of view the various architectural components of the specific small computer systems used in this class.
2) Design programs for the computer using major machine and assembly language commands.
3) Create simple modifications or extensions to existing operating system structures to provide additional system capabilities.
4) Choose specialized commands as needed to provide further capabilities.
5) Practice and demonstrate ability to apply knowledge of assembly language programming and machine architecture.
6) Examine the relationships between high-level languages and assembly level programming.

Method of Evaluation
A grading system will be established by the instructor and implemented uniformly. Grades will be based on demonstrated proficiency in subject matter determined by multiple measurements for evaluation, one of which must be essay exams, skills demonstration or, where appropriate, the symbol system.
1) Quizzes and exams that measure students' ability to use programming terminology and explain technical concepts related to computer organization.
2) Practical exams requiring students to trace programming code to predict outcome. For example, a 10-20 line section of code is provided and students must be able to identify how all of the values in all variables are altered by that code and predict the final output as specified in the code.
3) Projects requiring students to write complete application programs demonstrating knowledge of an assembly language.

Special Materials Required of Student
Flash drive

Minimum Instructional Facilities
Computer lab with networked computers and software to create an assembly language programming environment

Method of Instruction
1) Lecture and demonstration
2) Hands-on practice
3) Lab problems

Out-of-Class Assignments
1) Design, code and debug multiple assembly language programs
2) Analyze instructor-assigned pre-coded programs, post analysis comments on the class discussion board.

Texts and References
1) Required (representative example): Plantz, Robert G., Introduction to Computer Organization with x86-64 Assembly Language and GNU/Linux (2017, Downloadable at http://bob.cs.sonoma.edu).
2) Supplemental: None

Student Learning Outcomes

Upon successful completion of this course, students will be able to:

1) Design, write, and test computer assembly language programs.
2) Extend other programming languages, such as C++, using assembly language procedures and data.