

Virginia Western Community College
EGR 125
Introduction to Computer Programming for Engineers

Prerequisites

MTH 162 or MTH 167 or equivalent or departmental approval

Corequisites

EGR 121

Course Description

Introduces problem solving and implementation of computer software solutions using a high-level programming language in a structured environment. Includes concepts and practice of algorithm design, language syntax, control structures, arrays, and introduction to object-oriented programming. Covers engineering applications, such as mathematical modeling, file input and output, and basic numerical methods. The assignments in this course require mathematical problem-solving skills, algebraic modeling, and functions, and use of variables.

Semester Credits: 4 Lecture Hours: 4 Lab/Clinical/Internship Hours: 0

Required Materials**Textbooks:**

Textbook: C++ for Engineers and Scientists, 4th Edition, Gary J. Bronson, ISBN-13: 978-1-133-18784-4.

Other Required Materials:

Microsoft Visual Studio (free download)

General Course Purpose

EGR 125 is an introductory course in computer programming for engineers. It serves as the standard sequence of minimal programming content for all engineering majors and the first course in a programming sequence for electrical and electrical computer engineers. It is similar to the first course in computer science (CSC 221) with additional engineering programming applications included. Students may not earn credit for both. This course introduces computer software-based problem solving and implementation of solutions in a high-level programming language.

See: <https://courses.vccs.edu/courses/EGR125-IntroductiontoComputerProgrammingforEngineers/detail>

Course Outcomes

At the completion of this course, the student should be able to utilize the following:

- Basic concepts of computer systems
 - Differentiate computer components by functionality.
 - Define basics of computer storage devices.
 - Illustrate the computer structure.
 - Define Binary and Hexadecimal numeration systems.
 - Define types of software.
 - Explain the use of computers, and the social impact they have.
 - Discuss secure programming
 - Evaluate the ethical aspects of programming
- Processing Code
 - Editors, compilers and/or interpreters; distinguishing source code, object code, and executables.
 - Reading and understanding compilation error messages.
 - Executing programs.
 - Analyzing and resolving run-time errors.
- Problem analysis and algorithmic modeling
 - List and apply the steps involved in problem solving through algorithmic modeling.
 - Describe activities related to program modeling and design including algorithm development.
 - Solve problems using techniques such as pseudocode, flowcharts, and model development.
 - Verify algorithms and identify errors.
 - Distinguish between procedural techniques and object-oriented techniques
 - Write programs using good programming practices.
- Use of data
 - Compare and contrast data types
 - Describe the use of variables.
 - Build expressions using variables, literal data, and operators, correctly using rules of operator precedence.
- Decision structures
 - Describe how conditional selection operations are used to alter the sequential execution of a program.
 - Describe how relational and Boolean operators are used to form logical expressions that evaluate to true or false.
 - Identify techniques to evaluate selection statements for logic errors.
 - Develop programs using sequential and selection operations.
- Repetition structures
 - Describe how repetition structures are used to alter the sequential execution of a program.
 - Choose appropriate repetition structures based on the type of application.
 - Identify techniques to evaluate repetition statements for logic errors.
 - Develop programs using repetition structures.
- Programming with Procedures
 - Apply modularization to manage complexity of programming.

- Describe the roles of parameters in a procedure definition.
 - Illustrate parameter passing when invoking procedures.
 - Solve problems using procedures.
- Collections
 - Arrays and lists as appropriate to the programming language.
 - Defines nature and purpose.
 - Use collections as parameters and returned values in procedures.
 - Develop applications using collections
- Memory Allocation
 - Pointers and structures (C++), lists and dictionaries (Python), utilization of stack and heap (Java) as appropriate to the programming language.
- Classes and Introduction to Libraries
 - Describe information hiding and encapsulation.
 - Describe the concept of class and object of a class.
 - Use language classes from the standard library to develop programs.
- Engineering Applications
 - File input and output.
 - Introduction to mathematical modeling
 - Numerical methods, including root finding and numerical integration

Topical Description

- Basic concepts of computer systems
- Processing Code
- Problem analysis and algorithmic modeling
- Use of data
- Decision structures
- Repetition structures
- Programming with Procedures
- Collections
- Memory Allocation
- Classes and Introduction to Libraries
- Engineering Applications

Notes to Instructors

- All instructors teaching this course in any given semester will use the same textbooks.
- This course must contain a project for all students.
- Weekly quizzes, one or two in-semester tests and a final exam are recommended to be included in the course.
- The content of this course will be updated every few years in collaboration with engineering faculty from across the VCCS.