

COMPUTER SCIENCE II: PROGRAMMING

Computer Science II: Programming explores and builds skills in programming and a basic understanding of the fundamentals of procedural program development using structured, modular concepts. Coursework emphasizes logical program design involving user-defined functions and standard structure elements. Discussions will include the role of data types, variables, structures, addressable memory locations, arrays and pointers and data file access methods. An emphasis on logical program design using a modular approach, which involves task oriented program functions. The required prerequisite is Computer Science I.

- DOE Code: 5236
- Recommended Grade Level: 11, 12
- Required Prerequisite: Computer Science I
- Credits: 2 semester course, 2 semesters required, 1-3 credits per semester, 6 credits maximum
- Counts as a Directed Elective or Elective for all diplomas

Dual Credit

This course provides the opportunity for dual credit for students who meet postsecondary requirements for earning dual credit and successfully complete the dual credit requirements of this course.

Application of Content and Multiple Hour Offerings

Intensive laboratory applications are a component of this course and may be either school based or work based or a combination of the two. Work-based learning experiences should be in a closely related industry setting. Instructors shall have a standards-based training plan for students participating in work-based learning experiences. When a course is offered for multiple hours per semester, the amount of laboratory application or work-based learning needs to be increased proportionally.

Career and Technical Student Organizations (CTSOs)

Career and Technical Student Organizations are considered a powerful instructional tool when integrated into Career and Technical Education programs. They enhance the knowledge and skills students learn in a course by allowing a student to participate in a unique program of career and leadership development. Students should be encouraged to participate in Business Professional of America, DECA, or Future Business Leaders of America, the CTSOs for this area.

Content Standards

Domain – Task Analysis

Core Standard 1 Students evaluate the tasks a computer program is to perform.

Standards

CS2P-1.1 Differentiate between the different tasks a computer program should perform

CS2P-1.2 Formulate solutions to the different tasks a computer program should perform

CS2P-1.3 Interpret prior solutions; In case prior code and design could be reused

Domain – Problem Analysis

Core Standard 2 Students analyze a problem and develop an advanced solution by creating a computer program.

Standards

- CS2P-2.1 Recognize and explain how to use a computer program to solve a problem
- CS2P-2.2 Construct interactive computer programs that accept various forms of input and produce various forms of output, as a solution to an advanced computer programming problem
- CS2P-2.3 Use print charts, file layouts, program narratives, hierarchy charts, and system flowcharts, which accurately depict the problem assigned and describe the solution
- CS2P-2.4 Justify what programming methodology to use—object oriented or procedural
- CS2P-2.5 Appraise the program schematics and usage; document the program and describe its use
- CS2P-2.6 Recognize and explain the standard program flowchart symbols and use them correctly within the context of the basic control structures of sequence, selection and looping

Domain – Software Tools

Core Standard 3 Students apply and adapt software tools to develop an advanced computer program.

Standards

- CS2P-3.1 Construct an advanced program that processes information
- CS2P-3.2 Identify programming languages as procedural and object oriented
- CS2P-3.3 Recognize and explain class in object oriented programming
- CS2P-3.4 Recognize and explain object in object oriented programming
- CS2P-3.5 Recognize and explain method in object oriented programming
- CS2P-3.6 Recognize and explain instance variable in object oriented programming
- CS2P-3.7 Recognize and explain polymorphism in object oriented programming
- CS2P-3.8 Recognize and explain inheritance in object oriented programming
- CS2P-3.9 Recognize and explain overwriting methods in OOP
- CS2P-3.10 Recognize and explain encapsulation in computer programming
- CS2P-3.11 Apply and adapt at an advanced level fundamental programming concepts, including data types, control structures, methods, and arrays
- CS2P-3.12 Develop advanced programs using reusable modules (modularization)
- CS2P-3.13 Use advanced debugging techniques to correct and validate the computer program
- CS2P-3.14 Construct the program in a high-level programming language based on a created design
- CS2P-3.15 Determine how to integrate a computer program with a web browser
- CS2P-3.16 Determine how to use a common code/ GUI library
- CS2P-3.17 Identify controls (push buttons, entry fields, etc.), their properties, methods, and when to use each control

Domain – Algorithms

Core Standard 4 Students design a solution to the problem using algorithms.

Standards

- CS2P-4.1 Develop advanced algorithms to solve a computer programming problem(s)
- CS2P-4.2 Apply and adapt math operators in a computer program
- CS2P-4.3 Prescribe the use of algorithms to provide a solution to a programming problem

- CS2P-4.4 Use pseudo code to describe a solution to an advanced programming problem
- CS2P-4.5 Create a program flowchart and ANSI standard flowcharting symbols to define a solution to an advanced programming problem
- CS2P-4.6 Explain how the algorithm can be used to solve a problem

Domain – Program Development

Core Standard 5 Students create an advanced functional computer program.

Standards

- CS2P-5.1 Define the process of programming. For example: STAIR, Statement, Tools, Algorithm, Implement, and Refine
- CS2P-5.2 Create an advanced computer program that corresponds to an algorithm or proposed solution
- CS2P-5.3 Demonstrate programming structures
- CS2P-5.4 Appraise the use of data variables and constants
- CS2P-5.5 Appraise the use of local and global scope
- CS2P-5.6 Appraise the use of conditionals (IF statements)
- CS2P-5.7 Appraise the use of loops (while statements, for statements)
- CS2P-5.8 Use single and multidimensional Arrays
- CS2P-5.9 Create programmer defined functions and methods to break down the program logic and support reuse
- CS2P-5.10 Define the graphical user interface
- CS2P-5.11 Identify the parts of the programming platform
- CS2P-5.12 Identify different types of errors and handle them programmatically
- CS2P-5.13 Use the order of operations when using calculations
- CS2P-5.14 Construct an advanced computer program using proper condition and loop techniques
- CS2P-5.15 Use correct naming conventions in variable declarations, function declarations, class declarations, and other

Domain – Program Verification and Debugging

Core Standard 6 Students prove that an advanced computer program solution works by using verification and debugging techniques.

Standards

- CS2P-6.1 Predict and explain output
- CS2P-6.2 Identify cause/effect for input/output
- CS2P-6.3 Perform input validation
- CS2P-6.4 Scrutinize peers code for errors
- CS2P-6.5 Show the use of proper internal documentation and coding comments

Domain – Documentation

Core Standard 7 Students connect the associated task with the code by providing documentation.

Standards

- CS2P-7.1 Describe the function of an advanced computer program
- CS2P-7.2 Identify the purposes of an advanced computer program

- CS2P-7.3 Explain concepts related to an advanced computer program
- CS2P-7.4 Evaluate how to use an advanced computer program
- CS2P-7.5 Identify cause/effect by explaining input and output related to an advanced computer program
- CS2P-7.6 Interpret input/output of an advanced computer program