PARKLAND COLLEGE

Course Information Form (CIF)

Course Code: CSC 123 (IAI CS 911)

Course Title: Computer Science I (C/C++)

Department: Business/Computer Science and Technologies
Effective Date: Summer 2026
PCS Code: 1.1 - Baccalaureate/Transfer
CIP Code: 11.0202
Repeatability: 0

Credit Hours

Catalog Notation: 3-2-4
Credit Hour Distribution:

Lecture: 3	Lab: 2	Clinical: 0	**Total: 4**

General Course Information

Catalog Description

Introductory topics in computer science, intended for Computer Science and related majors. Emphasis on algorithms, program structure, data types, decision statements, strings, looping, functions, files, arrays, classes, objects, and documentation.

General Course Objectives

To become proficient in the implementation of algorithms in the C++ language to solve problems, and to understand and use basic computer science topics such as sorting, searching, looping, arrays, classes, and simple algorithms.

Minimum Placement Levels

English	Reading	Math
None	None	Placement out of MAT 098

Prerequisites

Credit in CIS 122 or approval of department chair

Methods of Evaluation

2-3 examinations, 5-10 quizzes, about 15 lab assignments, approximately 4 programming projects, and a final exam.

Instructional Materials and Additional Supplies

Problem Solving w/ C++ (w/cd), current edition, by Savitch, Addison-Wesley.

Course Content

General Learning Outcomes (GLOs)

- Reasoning and Inquiry: Students will demonstrate the ability to solve problems using deductive reasoning and logic, quantitative reasoning, or the scientific method.
- Technology: Students will demonstrate the ability to evaluate, select, and appropriately use current and emerging tools.

Course Segments and Student Learning Outcomes

Course Segment	Learning Outcomes	Lecture Hours	Lab Hours	Clinical Hours
Programming Environment	1. Demonstrate how to enter, compile/run a simple program, and identify the major components of programs. 2. Explain appropriate debugging and testing techniques for the language and environment used.	2	2	0
Data Types; Input and Output	1. Identify intrinsic data types and declare variables. 2. Plan and develop code that will input numeric and string values into variables, write arithmetic expressions to process data, and output the results with formatting. 3. Use both integer and floating point mathematics. 4. Identify constants, variables, and operators. 5. Apply naming conventions to achieve good programming style.	2	2	0
Selection	1. Identify the algorithmic need for selection, choose an appropriate selection structure, and successfully plan and develop code with a selection structure that implements the algorithm. 2. Apply the comparison operators. 3. Identify the need for logical operators and successfully code them in a selection structure.	5	3	0
Repetition	1. Identify the algorithmic need for repetition; choose an appropriate repetition structure, including nested loops if appropriate; and successfully code a repetition structure that implements the algorithm. 2. Plan and develop code with integer variables that count and control the repetition. 3. Plan and develop code with sentinel variables that end the repetition.	5	4	0
Functions and Top Down Design	1. Plan and develop function definitions within a program; identify the need for parameters and be able to code them; and show the limits of the scope of variables, including global variables and local variables. 2. Plan and develop code for necessary features of object-oriented languages, including pass-by-reference, pass-by-value, and return values. 3. Use libraries of functions to solve common problems like random number generation. 4. Dramatize the use of the call stack. 5. Describe when to use return types void and bool. 6. Distinguish between a prototype declaration, the call to a function, and the definition of a function; and explain the need and syntax for each. 7. Demonstrate appropriate program design and documentation methods using good programming style. 8. Plan and develop a C++ program with appropriate top down design and documentation methods using good programming style.	6	3	0

Course Segment	Learning Outcomes	Lecture Hours	Lab Hours	Clinical Hours
ASCII Text Files	1. Plan and develop code to store formatted data to an ASCII text file, and read formatted data from an ASCII text file into program variables. 2. Plan and develop code to store formatted data to an ASCII text file, and read formatted data from an ASCII text file into array structures with proper error checking.	2	2	0
Arrays	1. Plan and develop code to store data into arrays or equivalent structures, including multidimensional arrays or equivalent structures, and walk traverse those arrays or equivalent structures to access the data. 2. Construct code that implements a bubble sort. 3. Plan and develop code to store data into null terminated character arrays and walk those arrays to access the data. 4. Construct code that implements an insertion sort. 5. Construct code that implements a linear search. 6. Construct code that implements a binary search. 7. Construct code that implements a sort and search on a one-dimensional array or equivalent structures.	6	4	0
Strings and Vectors	1. Construct code that inputs string values into variables, writes string manipulation expressions to process data, and outputs the results with formatting. 2. Plan and implement a program to store data into vectors and walk those vectors to access the data. 3. Construct the code for the string class as a vector of characters, implementing its most important methods. 4. Construct the code for the vector class as an array, implementing its most important methods.	5	4	0
Memory and Addressing	1. Explain the fundamental memory concepts of stack and heap. 2. Explain and use pointers, references, and dereferencing. 3. Plan and develop code that dynamically allocates memory using malloc() and free(). 4. Plan and develop code that dynamically allocates objects using new and delete. 5. Calculate hexadecimal numbers from binary numbers and binary numbers from hexadecimal numbers. 6. Calculate decimal numbers from binary numbers and binary numbers from decimal numbers with values up to and including 255.	4	2	0

Course Segment	Learning Outcomes	Lecture Hours	Lab Hours	Clinical Hours
Object Oriented Programming	1. Plan and develop method definitions within a class, including the constructor methods; identify, plan, and develop the need for parameters and be able to code them; and show the limits of the scope of variables, including instance variables, class variables, and local variables. 2. Plan and develop appropriate bottom up program design and documentation methods using good programming style. 3. Describe the bottom up design process, employ a bottom up design on a problem, and then implement that design and include appropriate debugging and testing in the programs. 4. Plan and develop the C++ code necessary for features of object-oriented languages, including overloaded methods, intrinsic data, return values, and comments. 5. Use data models with functions in object-oriented design. 6. Apply object-oriented principles to a problem. 7. Describe the differences between an object's interface and implementation, and properly code public and private data members. 8. Apply the idea of information hiding with the use of accessor and mutator methods.	8	4	0

Total Contact Hours

Lecture Hours	Lab Hours	Clinical Hours
45	30	0