



# UTICA COMMUNITY SCHOOLS

---

## Course Title: Computer Science C++

### Course Content Expectations:

This course is intended to give students a solid foundation in object-oriented design and programming concepts using ANSI C++. Students will learn algorithm development, problem solving, and computer science concepts. The material is presented via a depth-first linear progression that guides students through procedural orientation, object-orientation, and data structures. **Students should know and be able to apply the following concepts to various situations.**

### **Introduction to C++ Programming**

- The vocabulary of C++, case sensitivity, reserved words, library identifiers, syntax, and identifiers.
- Data types and output including type **int**, type **double**, type **char**, strings, output formatting integers, formatting real numbers, and formatting strings.
- Graphics including pixels, points, coordinates systems, graphics operations, and setting the graphic mode.

### **Arithmetic, Variables, Input, Constants, and Library of Functions**

- Arithmetic in C++ including basic operations for integers, order of operations for integers, using modulus and division, representation of integers, basic operations for real numbers, and representation of real numbers.
- Use variables including memory locations, assignment statements, expressions, compound assignment, and output.
- Input including input statements, interactive input, reading numeric data, character sets, and reading character data.
- String variables including string input with **>>**, string input with **getline**, memory for string variables, the empty string, string concatenation, and compound assignment with strings.
- Constants including formatting constants and the library of constants.
- Library functions including their use, member functions and random numbers
- Type compatibility and type conversion including type cast.
- Graphics including using variables to specify coordinates, relative coordinates, geometric shapes, changing foreground and background colors, clearing the screen, setting the line width and style and determining the screen size.
- Structured programming including modularity.

- User defined functions including function declarations in a program, implementing functions, function headings, data and executable statements in a function implementation, using stubs, using drivers, functions without returned values or parameters, and functions with strings.
- Parameters including value parameters, reference parameters, and constant reference parameters.
- Subprograms including functions as subprograms, cohesive subprograms, functional abstraction, and information hiding.
- Identifiers including Global and Local Identifiers, side effects and parameters, and names of identifiers.
- Programmer-Defined Libraries.

### **Selection Statements**

- Boolean expressions including defining Boolean constants and a Boolean data type, saving a new type definition in a library file, relational operators and simple Boolean expressions, comparing strings, = vs. ==, Boolean functions, and logical operators and compound Boolean expressions.
- **If** statements including compound statements and **if** statements with functions.
- **If...else** statements
- **Nested** and **extended if** statements
- **Switch** statements
- Assertions
- Graphics

### **Repetition statements**

- The **for loop** including increment and decrement operators, accumulators, **for loops** that count down, and loops that count by factors other than one.
- **While loops** including compound conditions.
- **Do...while loops** including compound conditions.
- **Nested loops**
- **Repetition loops**

### **Files**

- Streams and stream processing including standard input and output streams, file streams, output file streams, using string variables to name files, and loops with output streams.
- Using functions with files including files and strings and buffered strings.
- Character input and output with **put** and **get**.

### **Arrays**

- Arrays including declaration, assignment statements, arithmetic, reading and writing.
- Using arrays including loops for input and output, loops for assigning, and processing with loops.

- Array parameters and functions
- Sorting and searching an array.
- Two-dimensional arrays including reading, writing and manipulation.

### **Building structured data: Structs and Classes**

- The struct data type including using structs with functions.
- Classes
- Object orientated programming
- A rational number class
- Derived classes and inheritance

### **Safe Arrays**

- Vectors including vector classes operations for vectors, constructing vectors, and indexing vectors.
- Strings including user requirements, operations, Declaration, construction, substrings, assignment, concatenation, compound assignment, input and output, and string comparison
- Ordered collections including operations, declaration, construction, indexing, and adding and removing elements.
- Sorted collections including user requirements, specifying the operations for sorted collections, declaring the sorted collections class, implementing the sorted collection class, and contained and derived classes.
- Matrices including specifying the operations for matrices, declaring the matrix class, and implementing the matrix class.

### **Linked Lists**

- Linked list including the characteristics and logical structure of a linked list, linked list operations, creating a linked list, detecting an empty linked list, moving to the next node, detecting the end of the list, moving to the next node, adding data to the list, removing data from the list, and using a linked list to solve problems.
- Defining the linked list ADT as a C++ Class including defining a pointer to a node, declaring the get\_node function, creating a linked list, inserting data into a linked list, and the first and next operations.
- Pointers and the management of computer memory including the addresses and values of simple variables, array variables, and pointer variables.

### **Recursion and Efficient Searching and Sorting**

- Recursion
- Binary Search
- Quick Sort
- Linked lists and recursion