

---

Student Name

---

High School or Vocational Center

**COMPETENCY RECORD FOR ARTICULATION**

**Computer Science Program  
Baker College of Muskegon**

Please check below each skill the student has mastered as described, with 80 percent accuracy, or with an A or B grade. The skills needed for articulation of each course are listed.

**CS217A C++ PROGRAMMING**

<b>Task</b>	<b>Satisfactory</b>	<b>Unsatisfactory</b>
Review basic trends in computer programming as they relate to the C++ language and object-oriented programming including the following:		
a. Object-oriented vs. procedural and structural programming		
b. Reserved words		
c. ANSI/ISO standards		
d. C++ program processing		
e. Compiler		
Review and reinforce the steps of analyzing a problem and to set up strategy prior to writing C++ code including the following:		
a. Developing algorithms		
b. Differentiating among sequential, selective (decision), and repetitious control structures		
c. Designing input/output charts using pseudo code		
Identify the stages of development of a C++ program including the following:		
a. Developing source code		
b. Selecting an appropriate name, data type, and initial value for a memory location		
c. Explaining how data is stored in memory		
d. Developing object code		
e. Developing executable code		
Demonstrate the ability to use a development environment to perform the following tasks:		
a. Understanding and discussing the components of a C++ program		
b. Saving, closing and opening a C++ solution		

<b>Task</b>	<b>Satisfactory</b>	<b>Unsatisfactory</b>
c. Building, executing, and printing a C++ program		
d. Locating an error in a C++ program		
Demonstrate the ability to create and invoke functions, pass information to functions, understand variables, use the MATH::POW() method, write function prototypes and format numeric output in C++ including the following tasks:		
a. Creating and invoking a function that returns a value		
b. Passing information, by value, to a function		
c. Understanding the scope and lifetime of a variable		
d. Raising a number to a power using the MATH::POW() method		
e. Writing a function prototype		
f. Formatting the numeric output in a C++ program		
Demonstrate the ability to create and implement various types of functions in C++ including the following tasks:		
a. Creating and invoking a function that does not return a value		
b. Passing information, by reference, to a function		
c. Passing a String variable, by value and by reference, to a function		
Demonstrate the ability to develop and chart algorithms, use the if and if/else forms, use comparison and logical operators in C++ including the following tasks:		
a. Using the selection structure in a program including flowcharts and coding		
b. Using comparison operators (relational operators) and conditions		
c. Using logical (Boolean) operators		
d. Using switch forms of if/else		
e. Compare strings		
Demonstrate the ability to understand and use nested selection structures and switch forms of if/else including the following tasks:		
a. Using a nested selection structure in pseudo code and in a flowchart		
b. Coding a nested selection structure in C++		
c. Coding the switch form of the selection structure in C++		
Demonstrate the ability to create and implement repetition structures in C++ including the following tasks:		
a. Using a repetition structure in pseudo code and in a flowchart		
b. Coding a pretest loop using the C++ while statement		
c. Initializing and updating counters and accumulators		

Task	Satisfactory	Unsatisfactory
d. Coding a pretest loop using the C++ for statement		
Demonstrate the ability to use posttest loops and nested repetition structures in C++ including the following tasks:		
a. Showing the posttest repetition structure in pseudo code and in a flowchart		
b. Coding a posttest loop using the C++ do statement		
c. Understanding and using nested repetition structures		
Demonstrate the ability to create and manipulate sequential access files in C++ including the following tasks:		
a. Opening and closing a sequential access data file		
b. Writing and reading information to and from a sequential access file		
Demonstrate the ability to create and manipulate user-defined simple data type including the following tasks:		
a. Declaring, initializing, Enumeration type		
b. Namespaces		
c. String type		
d. Typedef statement		
Demonstrate the ability to work as a team member on a programming assignment that includes the analysis, design, development, documentation, debugging, and presentation of a successful C++ program using sequential, selective, and repetition structures learned in class.		

Teacher Signature \_\_\_\_\_ Date \_\_\_\_\_