

Unit 202 Create Software Components using 'C++' Level 2 (Core)**Rationale**

The aim of this unit is to enable candidates to understand the principles required to create software using the 'C++' programming language. Candidates will develop the skills required to create and test software components to solve a given problem.

There are 7 outcomes to this unit. The candidate will be able to:

1. specify the basic tools required to create, compile and execute a program
2. perform output to the screen
3. construct and execute a 'C++' program using input, output and different data types
4. use pre-defined functions
5. use operators for arithmetic and logical purposes
6. use suitable control structures and functions within a program
7. produce programs, which are tested and presented to a specified/agreed standard.

Guided learning hours

The recommended guided learning hours for this unit are 90 hours.

Connections with other awards**NVQ links**

Outcome	This award contributes to the knowledge and understanding of the following elements of NVQ(s)
1,2,3,4,5,6	<i>C&G 4300 - Developing IT Programs Level 2</i> 216.1 - Assist the creation of software
7	216.2 - Assemble and test software components

Key Skills links

Communication	C3.2
Application of Number	N1.1
Working with others	None
Information technology	None
Improving own learning	LP3.1, LP3.2, LP3.3
Problem solving	PS3.1, PS3.2, PS3.3

Assessment

Assessment will be by means of a **set assignment** covering practical activities, and a **multiple choice test** covering underpinning knowledge.

Outcome 1: Specify the basic tools required to create, compile and execute a program.

	Candidate's signature	Date
<p>Practical activities The candidate will be able to:</p> <ol style="list-style-type: none"> select the tools required to create source code create a simple program using a 'C++' text editor save and retrieve source code to/from disk use the standard 'C++' input/output library <p style="text-align: center;">#include <iostream.h></p> <ol style="list-style-type: none"> use meaningful names when naming programs. 		
<p>Underpinning knowledge The candidate will be able to:</p> <ol style="list-style-type: none"> describe the meaning of a stand-alone program identify the basic structure of a 'C++' program identify that 'C++' is a superset of 'C' explain the need for the #include <iostream.h> directive in a 'C++' program describe the pre-processor stage explain why source code is compiled describe why object code is linked with library files in producing executable code state the file types that are created when program code is fully compiled describe the difference between source code, object code and executable code identify which file type can be edited identify the processes required in saving and retrieving source code from disk. 		

Outcome 2: Perform output to the screen.

	Candidate's signature	Date
<p>Practical activities The candidate will be able to:</p> <ol style="list-style-type: none"> 1. use the correct output syntax to produce screen messages 2. compile source code 3. interpret and resolve compilation error messages 4. edit source code 5. use stream manipulators in a program. 		
<p>Underpinning knowledge The candidate will be able to:</p> <ol style="list-style-type: none"> 1. describe the meaning of the following syntax: - <ul style="list-style-type: none"> • #include • #define • int main() • return • void main() • cout<< • cin>> • ; (semi-colon) • /* */ • // 2. identify the need for indentation in code to aid readability 3. state the purpose of compiling source code 4. describe the purpose of the following standard escape sequences: - <p style="text-align: center;">endl \n \r \t \\ \a \' \'</p> 5. explain the difference between compilation errors and run-time errors 6. state that some compilers produce object code that can only be run on a run-time system. 		

Outcome 3: Construct and execute a 'C++' program using input, output and different data types.

	Candidate's signature	Date
<p>Practical activities The candidate will be able to:</p> <ol style="list-style-type: none">1. create a program which defines data types<ul style="list-style-type: none">• char• int• float• char[]2. use cin>> to read from the keyboard3. use meaningful variable names4. use #define to create a symbolic name or constant in a program5. use const to declare constants of types: -<ul style="list-style-type: none">• int• float• char6. create and use one-dimensional arrays of types: -<ul style="list-style-type: none">• char• int• float.		

Underpinning knowledge

The candidate will be able to:

1. identify the correct data types to be used in a program
2. state that variable names in 'C++' are case sensitive
3. explain the reason for using uppercase for symbolic constant names and lowercase for variables and function names
4. explain the difference between a character variable, a character array and a character string
5. state the purpose of the null terminator in relation to a string
6. identify the problems in using `cin>>` to input strings (whitespace)
7. explain the difference between strings and characters when using the symbols “ “ or ‘ ‘
8. explain the difference between a constant and a variable
9. identify the declaration construct an array
10. identify that data types must be compatible with the data being assigned.

Outcome 4: Use pre-defined functions.

	Candidate's signature	Date
<p>Practical activities The candidate will be able to:</p> <ol style="list-style-type: none"> use pre-defined functions in a program similar to: - <ul style="list-style-type: none"> getch() getche() cin.getline() gets() clrscr() clreol() tolower() toupper() convert strings using: - <ul style="list-style-type: none"> atoi() atof() use strcpy() to assign a string input strings using: - <ul style="list-style-type: none"> cin.getline() gets(). 		
<p>Underpinning knowledge The candidate will be able to:</p> <ol style="list-style-type: none"> identify the purpose of pre-defined functions state the meaning of 'exceeding array bounds' when dealing with strings state the purpose of the strcpy() function explain the difference between echoed and non-echoed character input (e.g. getch() and getche()) identify the appropriate pre-defined function for converting a string to a numeric value identify the resulting data type when using atoi() and atof(). 		

Outcome 5: Use operators for arithmetic and logical purposes.

	Candidate's signature	Date
<p>Practical activities The candidate will be able to:</p> <ol style="list-style-type: none"> 1. use the assignment operator = (equals) in a program 2. use the ++ and - - operators in prefix and postfix mode 3. use the arithmetic operators: - i.e. * / - + % 4. use a conditional statement within a program which includes if and else 5. use a switch statement in a program 6. create simple and compound statements 7. use relational operators in a program 8. use logical operators in a program in decision making processes: - i.e. !(not) && (and) (or) 9. use constants to represent TRUE and FALSE 10. use the ASCII code as part of validation. 		

Underpinning knowledge

The candidate will be able to:

1. identify the precedence of arithmetic operators
2. state the difference between the = and == symbols
3. identify the role of conditional statements within a program
4. identify the correct operators to use when making a conditional statement
5. state the use of a switch statement
6. describe the ++ and - - operators purpose in prefix and postfix modes
7. describe compound statements (e.g. the **nesting** of **if** statements)
8. describe the purpose of the symbols

{ } () [] < >

9. describe the order of precedence for arithmetic operators including the use of parenthesis
10. describe the actions of the relational operators: -

< <= == != > >=

Outcome 6: Use suitable control structures and functions within a program.

	Candidate's signature	Date
<p>Practical activities The candidate will be able to:</p> <ol style="list-style-type: none"> 1. use control structures within a program utilising the loops: - <ul style="list-style-type: none"> • while • do...while • for 2. use control structures as part of a validation process 3. create functions with/without parameters 4. use a function to return a value 5. demonstrate the difference between local and global variables. 		
<p>Underpinning knowledge The candidate will be able to:</p> <ol style="list-style-type: none"> 1. identify the purpose and format of the loops: - <ul style="list-style-type: none"> • while, • do...while • for 2. describe how control structures can assist in the validation of user input 3. state that functions create a modular solution to a program and even main() is a function 4. explain why function prototypes have to be declared in a program 5. explain the difference between global and local variables 6. explain the meaning of scope in relation to local and global variables 7. state the difference between passing parameters to a function: - <ul style="list-style-type: none"> • by value (copy) • by reference. 		

Outcome 7: Produce programs, which are tested and presented to a specified/agreed standard.

	Candidate's signature	Date
<p>Practical activities The candidate will be able to:</p> <ol style="list-style-type: none"> 1. write programs using case sensitivity to improve program readability 2. write syntax that is consistently indented 3. add comments to a program using <code>/* */</code> and/or <code>//</code> 4. print a program listing 5. select an appropriate method to test for the expected outcome of a program 6. compare expected output from test data to the actual output of the run-time program 7. provide evidence that the program complies with the specification. 		
<p>Underpinning knowledge The candidate will be able to:</p> <ol style="list-style-type: none"> 1. explain the need to add meaningful comments to a program to aid understanding of a program 2. explain the difference between <code>/* */</code> and <code>//</code> 3. state the benefits of printing a hard copy (program listing) of source code 4. identify that testing for expected output can assist in determining whether or not the program is working correctly and conforms to specification. 		