COURSE OUTLINE: CSD102 - PROGRAMMING C++

Prepared: Sasha Coleman
Approved: Corey Meunier, Chair, Technology and Skilled Trades

Course Code: Title	CSD102: PROGRAMMING USING C++
Program Number: Name	2090: COMPUTER PROGRAMMER
Department:	COMPUTER STUDIES
Semesters/Terms:	21W
Course Description:	The primary focus of this programming course is to develop the student`s logical problem-solving skills. At the same time, the student will learn the constructs inherent in all programming languages. To understand the program development process, the following concepts will be discussed: structured programming techniques, pseudocode, algorithm development, syntax, data types/variables, debugging, documentation, conditions, looping, user-defined functions, arrays, pointers, structures, file handling and an introduction to OOP using classes. Problem-solving skills are developed through programming assignments of increasing complexity.
Total Credits:	5
Hours/Week:	5
Total Hours:	5
Prerequisites:	CSD105
Corequisites:	There are no co-requisites for this course.
This course is a pre-requisite for:	CSD206, CSD207, CSD210, CSD211
Vocational Learning Outcomes (VLO's) addressed in this course: **Please refer to program web page for a complete listing of program outcomes where applicable.**	**2090 - COMPUTER PROGRAMMER** VLO 1 Identify, analyze, develop, implement, verify and document the requirements for a computing environment. VLO 5 Communicate and collaborate with team members and stakeholders to ensure effective working relationships. VLO 6 Select and apply strategies for personal and professional development to enhance work performance. VLO 9 Support the analysis and definition of software system specifications based on functional and non-functional requirements. VLO 10 Cntribute to the development, documentation, implementation, maintenance and testing of software systems by using industry standard software development methodologies based on defined specifications and existing technologies/frameworks. VLO 11 Apply one or more programming paradigms such as, object-oriented, structured or functional programming, and design principles, as well as documented requirements, to the software development process.
Essential Employability	EES 1 Communicate clearly, concisely and correctly in the written, spoken, and visual form

In response to public health requirements pertaining to the COVID19 pandemic, course delivery and assessment traditionally delivered in-class, may occur remotely either in whole or in part in the 2020-2021 academic year.

Skills (EES) addressed in this course:

that fulfills the purpose and meets the needs of the audience.

EES 2 Respond to written, spoken, or visual messages in a manner that ensures effective communication.

EES 3 Execute mathematical operations accurately.

EES 4 Apply a systematic approach to solve problems.

EES 5 Use a variety of thinking skills to anticipate and solve problems.

EES 6 Locate, select, organize, and document information using appropriate technology and information systems.

EES 7 Analyze, evaluate, and apply relevant information from a variety of sources.

EES 8 Show respect for the diverse opinions, values, belief systems, and contributions of others.

EES 9 Interact with others in groups or teams that contribute to effective working relationships and the achievement of goals.

EES 10 Manage the use of time and other resources to complete projects.

EES 11 Take responsibility for ones own actions, decisions, and consequences.

Course Evaluation:

Passing Grade: 50%, D

A minimum program GPA of 2.0 or higher where program specific standards exist is required for graduation.

Other Course Evaluation & Assessment Requirements:

Grade
Definition Grade Point Equivalent
A+ 90 - 100% 4.00
A 80 - 89% 4.00
B 70 - 79% 3.00
C 60 - 69% 2.00
D 50 - 59% 1.00
F(Fail) below 50% 0.00

CR (Credit)
Credit for diploma requirements has been awarded.
S Satisfactory achievement in field/clinical placement or non-graded subject area.
U Unsatisfactory achievement in field/clinical placement or non-graded subject area.
X A temporary grade limited to situations with extenuating circumstances giving a student additional time to complete the requirements for a course.
NR Grade not reported to Registrar`s office.
W Student has withdrawn from the course without academic penalty.

OTHER EVALUATION CONSIDERATIONS

1. In order to pass this course the student must obtain an overall test/quiz average of 50% or better, as well as, an overall assignment average of 50% or better. A student who is not present to write a particular test/quiz, and does not notify the professor beforehand of their intended absence, may be subject to a zero grade on that test/quiz.

2. There will be no supplemental or make-up quizzes/tests in this course unless there are extenuating circumstances.

In response to public health requirements pertaining to the COVID19 pandemic, course delivery and assessment traditionally delivered in-class, may occur remotely either in whole or in part in the 2020-2021 academic year.

3. Assignments must be submitted by the due date according to the specifications of the professor. Late assignments will normally be given a mark of zero. Late assignments will only be marked at the discretion of the professor in cases where there were extenuating circumstances.

4. Any assignment/projects submissions, deemed to be copied, will result in a zero grade being assigned to all students involved in that particular incident.

5. It is the responsibility of the student to ask the professor to clarify any assignment requirements.

6. The professor reserves the right to modify the assessment process to meet any changing needs of the class.

Attendance:

Sault College is committed to student success. There is a direct correlation between academic performance and class attendance, therefore, for the benefit of all its constituents, all students are encouraged to attend all of their scheduled learning and evaluation sessions. This implies arriving on time and remaining for the duration of the scheduled session. It is the departmental policy that once the classroom door has been closed, the learning process has begun. Late arrivers may not be granted admission to the room.

Absences due to medical or other unavoidable circumstances should be discussed with the professor, otherwise a penalty may be assessed. The penalty depends on course hours and will be applied as follows:
Course Hours Deduction
5 hrs/week (75 hrs) 1.0% /hr
4 hrs/week (60 hrs) 1.5% /hr
3 hrs/week (45 hrs) 2.0% /hr
2 hrs/week (30 hrs) 3.0% /hr

Final penalties will be reviewed and assessed at the discretion of the professor.

Books and Required Resources:

Starting Out with C++: From Control Structures through Objects by Tony Gaddis
Publisher: Pearson Edition: 9th
ISBN: 978-0134498379, 0134498372

http://www.cplusplus.com/doc/tutorial/introduction/ by Cplusplus Site

Course Outcomes and Learning Objectives:

Course Outcome 1	**Learning Objectives for Course Outcome 1**
1. Discuss and apply the concepts involved in the development of a program to solve problems using the computer and write simple C/C++ programs applying the concepts of input/output, arithmetic, and assignment. References at cplusplus.com:	Elements of the performance: 1.1 demonstrate an understanding of the Microsoft Visual C++ environment 1.2 explain the main components of a C/C++ program 1.3 name and distinguish C/C++ basic data types 1.4 explain and properly use the naming conventions for C/C++ identifiers 1.5 differentiate between character, string, and numeric constants 1.6 differentiate between character and numeric variables

Basics of C++: Structure of a program Variables. Data Types. Constants Operators Basic Input/Output This learning outcome will comprise 10% of the course.	1.7 declare and initialize variables correctly 1.8 use assignment operators (=, +=, -=, *=, /=) to assign values/expressions to variables 1.9 use increment/decrement operators (++, --) to increase/decrease values by 1 1.10 use arithmetic operators and apply their precedence (+, -, *, /, %) 1.11 evaluate integer and mixed-mode arithmetic correctly 1.12 use various C++ math library functions to perform arithmetic calculations 1.13 explain automatic promotion and apply typecasting to define data types 1.14 describe the purpose of a compiler/interpreter 1.15 describe the process of transforming a source program to an executable module 1.16 differentiate between syntax and logic errors 1.17 apply the cin object to perform input of data 1.18 apply the cout object to perform output of data 1.19 apply the cin.getline( ) function to accept string values that include a space(s) 1.20 apply the setw( ), setprecision( ), and setf( ) manipulators to format 1.21 explain and apply the #include directive 1.22 explain the purpose of include files for the cin and cout objects 1.23 write algorithms to solve problems using pseudocode 1.24 write, test, and debug programs using the concepts above
Course Outcome 2	**Learning Objectives for Course Outcome 2**
2. Develop algorithms and write C/C++ programs to solve problems involving the standard computer operations of decisions/conditions and selection. References at cplusplus.com: Control Structures: Control Structures (conditions) This learning outcome will comprise 10% of the course.	Elements of the performance: 2.1 describe and use the relational operators (==, !=, <, <=, >, >=) 2.2 describe the use of the logical operators (&&, ``) and use them to write both simple and complex expressions 2.3 describe the operation of the following C/C++ decision-making structures and use them in C/C++ programs: i. if...else ii. nested ifs iii. if...else if...else iv. the switch statement 2.4 write algorithms to solve problems containing decision-making structures, and describe them using pseudocode 2.5 write, test, and debug programs containing decision structures
Course Outcome 3	**Learning Objectives for Course Outcome 3**
3. Develop algorithms and	Elements of the performance:

write C/C++ programs to solve problems involving the standard computer operations of looping and repetition, and, debug program logic errors using the C++ Debugger. References at cplusplus.com: Control Structures: Control Structures (loops) This learning outcome will comprise 10% of the course.	3.1 discuss the concept of repetition/looping in computer programs 3.2 describe the operation of the following C/C++ repetition structures and use them in C/C++ programs: i. while ii. do...while iii. for iv. nested loops 3.3 use break, continue, and exit to terminate the iteration of a loop 3.4 write algorithms to solve problems containing repetition structures, and describe them using pseudocode 3.5 describe and correct an infinite loop problem 3.6 execute code one line at a time using the Step Debugger 3.7 use the following stepping options: Go, Step Into, Step Over, Step Out, Watch, and Run to Cursor 3.8 define, as well as, insert and remove break 3.9 write, test, and debug programs containing repetition structures
Course Outcome 4	**Learning Objectives for Course Outcome 4**
4. Discuss and create user-written, independently-compiled functions. References at cplusplus.com: Control Structures: Functions (I) Functions (II) Pointers This learning outcome will comprise 20% of the course.	Elements of the performance: 4.1 distinguish between local and global variables 4.2 discuss and apply the concepts of `passing` arguments to called functions by value 4.3 discuss and apply the concept of `returning` values to calling functions 4.4 write, test, and debug programs containing functions 4.5 discuss and apply the concept of pointers and pointer arithmetic 4.6 discuss and apply the concept of pointers in C/C++ 4.7 define and apply the concepts of the following terms: scope calling vs called functions Function prototypes local vs global variables pass by value return statement class pass by reference overloaded functions auto vs static variables arguments/parameters • develop modularized, structured programs by creating user-written functions • discuss and apply the concepts of `passing` arguments to called functions by value • discuss and apply the concept of `returning` values to calling functions • discuss and apply the concepts of `passing` arguments to called functions by reference • develop modularized, structured programs by creating user-written functions

In response to public health requirements pertaining to the COVID19 pandemic, course delivery and assessment traditionally delivered in-class, may occur remotely either in whole or in part in the 2020-2021 academic year.

Course Outcome 5	**Learning Objectives for Course Outcome 5**
5. Develop algorithms and write C++ programs to solve problems involving tables/arrays. References at cplusplus.com: Compound Data Types: Arrays This learning outcome will comprise approximately 15% of the course.	Elements of the performance: 5.1 define and apply the concepts of the following terms: one-dimensional array index value subscript two-dimensional array null character 5.2 discuss the purpose and concepts relating to one- and two-dimensional arrays 5.3 declare and initialize both numeric and character arrays 5.4 apply the concept of pointers to arrays 5.5 access and process array elements 5.6 pass arrays between functions 5.7 write, test, and debug programs containing arrays
Course Outcome 6	**Learning Objectives for Course Outcome 6**
6. Discuss and apply the concepts of character sequences/arrays and string manipulation with reference to C/C++ library functions. References at cplusplus.com: Compound Data Types: Character Sequences This learning outcome will comprise approximately 10% of the course.	Elements of the performance: 6.1 understand and utilize the C++ string class and its associated functions to declare string variables and manipulate string values 6.2 discuss and apply character-based functions such as: cin.get( ) tolower( ) toupper( ) isalpha( ) isdigit( ) isalnum( ) islower( ) isupper( ) 6.3 discuss and apply string functions such as: str.append( ) str.compare( ) str.length( ) str.copy( ) 6.4 write, test, and debug programs containing character and string functions
Course Outcome 7	**Learning Objectives for Course Outcome 7**
7. Develop algorithms to solve problems involving the use of data structures. References at cplusplus.com: Compound Data Types: Data Structures This learning outcome will comprise approximately 10% of the course.	Elements of the performance: 7.1 define and apply the concepts of the following terms: structure member record internal pointer 7.2 discuss the concept of structures in C/C++ 7.3 declare and initialize a structure 7.4 access and process structure members 7.5 apply the use of arrays of structures 7.6 apply methods of passing and returning structures to and from functions 7.7 write, test, and debug programs containing structures
Course Outcome 8	**Learning Objectives for Course Outcome 8**
8. Develop algorithms to solve problems involving the	Elements of the performance:

In response to public health requirements pertaining to the COVID19 pandemic, course delivery and assessment traditionally delivered in-class, may occur remotely either in whole or in part in the 2020-2021 academic year.

use of file manipulation. References at cplusplus.com: C++ Standard Library: Input/Output with fi... This learning outcome will comprise approximately 5% of the course.	8.1 define and apply the concepts of the following terms: file open read close write append 8.2 Create a disk file 8.3 write data to, and, read data from a disk file 8.4 perform disk I/O with records 8.5 create, and manipulate sequential and random access files 8.6 write, test, and debug programs containing files
Course Outcome 9	**Learning Objectives for Course Outcome 9**
9. Introduce the concept of object-oriented programming using classes and objects by comparing with structures. References at cplusplus.com: Classes: Classes I Classes II This learning outcome will comprise approximately 10% of the course.	Elements of the Performance: 9.1 Identify the most important features of Object-oriented programming languages. 9.2 Assess the strengths and weaknesses of OOP and procedural programming. 9.3 Define classes and implement class members and member functions. 9.4 Compare classes to structures. 9.5 Explain the relationship between class and object declarations. 9.6 Develop and manipulate an array of classes. 9.7 Use classes as parameters in function calls. 9.8 Declare and define constructors and destructors for classes. 9.9 Implement operator overloading. 9.10 Use pointers to point to a class object 9.11 Explain the use of inheritance in C++ programs. 9.12 Derive new classes from base/parent classes. 9.13 Write and debug programs utilizing the components above.

Evaluation Process and Grading System:

Evaluation Type	Evaluation Weight
Labs	30%
Lecture Assignments	9%
Quizes	10%
Test-1	17%
Test-2	17%
Test-3	17%

Date: July 22, 2020

Addendum: Please refer to the course outline addendum on the Learning Management System for further information.

In response to public health requirements pertaining to the COVID19 pandemic, course delivery and assessment traditionally delivered in-class, may occur remotely either in whole or in part in the 2020-2021 academic year.