

**SAULT COLLEGE OF APPLIED ARTS AND TECHNOLOGY**

**SAULT STE. MARIE, ONTARIO**



**COURSE OUTLINE**

**COURSE TITLE:** INTRODUCTION TO PROGRAMMING

**CODE NO. :** CSD102 **SEMESTER:** 2

**PROGRAM:** ALL INFORMATION TECHNOLOGY PROGRAMS

**AUTHOR:** Dennis Ochoski

**DATE:** Jan. 2010 **PREVIOUS OUTLINE DATED:** Jan. 2009

**APPROVED:** "B. Punch"

---

	<b>CHAIR</b>	<b>DATE</b>
<b>TOTAL CREDITS:</b>	5	
<b>PREREQUISITE(S):</b>	<u>None</u>	
<b>HOURS/WEEK:</b>	<u>5</u>	

**Copyright ©2010 The Sault College of Applied Arts & Technology**  
*Reproduction of this document by any means, in whole or in part, without prior written permission of Sault College of Applied Arts & Technology is prohibited.*  
*For additional information, please contact Brian Punch, Chair,*  
*Natural Environment/Outdoor Studies, & Technology Programs*  
*(705) 759-2554, Ext. 2681*

**I. COURSE DESCRIPTION:**

This course is intended to provide a firm foundation of computer programming skills needed in the information technology area. The C/C++ programming language is used to develop the student's skills in problem solving, computer programming,, and software utilization.

**II. TOPICS TO BE COVERED:**

1. Introduction to computer programming concepts.
2. C/C++ program structures and format.
3. Decisions/Conditions in C/C++.
4. Repetition/Looping in C/C++.
5. Modularization using User-Defined Functions.
6. Advanced Concepts with User-Defined Functions.
7. The Debugger.
8. Arrays/Tables.
9. Pointers.
10. Advanced Concepts with Characters and Strings.
11. Data Structures.
12. Files (if time permits).

### III. LEARNING OUTCOMES AND ELEMENTS OF THE PERFORMANCE:

Upon successful completion of this course the student will demonstrate the ability to:

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:**

**Basics of C++:** [Structure of a program](#)  
[Variables. Data Types.](#)  
[Constants](#)  
[Operators](#)  
[Basic Input/Output](#)

This learning outcome will comprise **15%** of the course.

#### ***Elements of the performance:***

- define the concept of a "computer program/software"
- differentiate between a high level language, compiler and machine language
- describe the top-down process of developing a logical solution to a problem and use pseudocode to plan a series of detailed steps leading to a solution
- understand the "golden rule" for writing computer programs
- demonstrate an understanding of the Microsoft Visual C++ environment
- explain the main components of a C/C++ program
- name and distinguish C/C++ basic data types
- explain and properly use the naming conventions for C/C++ identifiers
- differentiate between character, string, and numeric constants
- differentiate between character and numeric variables
- declare and initialize variables correctly
- explain computer memory concepts and how they relate to processing data
- use assignment operators (=, +=, -=, \*=, /=) for assigning values/expression results to variables
- use increment/decrement operators (++ , --) to increase/decrease values by 1
- use arithmetic operators and apply their precedence (+, -, \*, /, %)
- evaluate integer and mixed-mode arithmetic correctly
- use various C++ math library functions to perform arithmetic calculations
- explain automatic promotion and apply typecasting to define data types
- describe the purpose of a compiler/interpreter

- describe the process of transforming a source program to an executable module

***Elements of the performance(cont'd):***

- differentiate between syntax and logic errors
  - apply the *cin* object to perform input of data
  - apply the *cout* object to perform output of data
  - apply the *getline( )* function to accept string values that include a space(s)
  - apply the *setw( )*, *setprecision( )*, and *setf( )* manipulators to format output on the screen
  - explain and apply the *#include* directive
  - explain the purpose of "include" files for the *cin* and *cout* objects
  - write algorithms to solve problems using pseudocode
  - write, test, and debug programs using the concepts above
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 **15%** of the course.

***Elements of the performance:***

- describe and use the relational operators (*==*, *!=*, *<*, *<=*, *>*, *>=*, *!|*)
- describe the use of the logical operators (*&&*, *||*) and use them to write both simple and complex expressions
- 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
- write algorithms to solve problems containing decision-making structures, and describe them using pseudocode
- write, test, and debug programs containing decision structures

3. Develop algorithms and 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 **15%** of the course.

***Elements of the performance:***

- discuss the concept of repetition/looping in computer programs
- describe the operation of the following C/C++ repetition structures and use them in C/C++ programs:

- while*
- do...while*
- for*
- nested loops

- use *break*, *continue*, and *exit* to terminate the iteration of a loop
- write algorithms to solve problems containing repetition structures, and describe them using pseudocode
- describe and correct an "infinite loop" problem
- execute code one line at a time using the Step Debugger
- use the following stepping options: **Go**, **Step Into**, **Step Over**, **Step Out**, **Watch**, and **Run to Cursor**
- define, as well as, insert and remove break
- write, test, and debug programs containing repetition structures

4. Discuss and create user-written, independently-compiled functions.

**References at cplusplus.com:**

**Control Structures:** [Functions \(I\)](#)  
[Functions \(II\)](#)

This learning outcome will comprise **15%** of the course.

***Elements of the performance:***

- understand the role and operation of functions in C/C++ and other languages
- distinguish between the *calling* and the *called* functions

***Elements of the performance(cont'd):***

- understand the concept of *scope*
- distinguish between *local* and *global* variables
- discuss and apply the concepts of 'passing' arguments to called functions by value
- discuss and apply the concept of 'returning' values to calling functions
- write, test, and debug programs containing functions
- discuss and apply the concept of pointers and pointer arithmetic
- discuss and apply the concept of pointers in C/C++
- 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
5. Develop algorithms and write C++ programs to solve problems involving tables/arrays.

**References at [cplusplus.com](http://cplusplus.com):**

**Compound Data Types:** [Arrays](#)

This learning outcome will comprise approximately **15%** of the course.

***Elements of the performance:***

- define and apply the concepts of the following terms:

one-dimensional array    index value            subscript  
two-dimensional array    null character

- discuss the purpose and concepts relating to one- and two-dimensional arrays
- declare and initialize both numeric and character arrays
- apply the concept of pointers to arrays

***Elements of the performance(cont'd):***

- access and process array elements
- pass arrays between functions
- write, test, and debug programs containing arrays

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 **5%** of the course.

***Elements of the performance:***

- understand and utilize the C++ string class and its associated functions to declare string variables and manipulate string values
- discuss and apply character-based functions such as:

cin.get( )            tolower( )            toupper( )            isalpha( )  
isdigit( )            isalnum( )            islower( )            isupper( )

***Elements of the performance(cont'd):***

- discuss and apply string functions such as:

strcat( )    strcmp( )    strlen( )    strcpy( )  
atoi( )    atof( )    atol( )    itoa( )

- write, test, and debug programs containing character and string functions

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:***

- define and apply the concepts of the following terms:

structure          member          record          internal pointer

- discuss the concept of structures in C/C++
- declare and initialise a structure
- access and process structure members
- apply the use of arrays of structures
- apply methods of passing and returning structures to and from functions
- write, test, and debug programs containing structures

8. Develop algorithms to solve problems involving the use of file manipulation.  
(if time permits)

**References at cplusplus.com:**

**C++ Standard Library:** [Input/Output with fi...](#)

This learning outcome will comprise approximately **5%** of the course.

***Elements of the performance:***

- define and apply the concepts of the following terms:

file          open          read          close          write          append



INTRODUCTION TO PROGRAMMING  
COURSE NAME

CSD102  
COURSE CODE

- create a disk file
- write data to, and, read data from a disk file
- perform disk I/O with records
- create, and manipulate sequential and random access files
- write, test, and debug programs containing files

**IV. REQUIRED RESOURCES/TEXTS/MATERIALS**

Internet Link: <http://www.cplusplus.com/doc/tutorial/introduction/>

Visual C++ 2008 Express Edition Software:  
<http://www.microsoft.com/express/vc/#webInstall>

**V. EVALUATION PROCESS/GRADING SYSTEM:**

<b>Evaluation Methods</b>	<b>Weight</b>
Quizzes	20%
Tests	50%
Assignments	<u>30%</u>
	100%

The following semester grades will be assigned to students in postsecondary courses:

<b><u>Grade</u></b>	<b><u>Definition</u></b>	<b><u>Grade Point Equivalent</u></b>
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	

INTRODUCTION TO PROGRAMMING  
COURSE NAME

CSD102  
COURSE CODE

	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.

## VI. 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.
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.

## VII. SPECIAL NOTES:

INTRODUCTION TO PROGRAMMING  
COURSE NAME

CSD102  
COURSE CODE

Course Outline Amendments:

The professor reserves the right to change the information contained in this course outline depending on the needs of the learner and the availability of resources.

Retention of Course Outlines:

It is the responsibility of the student to retain all course outlines for possible future use in acquiring advanced standing at other postsecondary institutions.

Prior Learning Assessment:

Students who wish to apply for advance credit transfer (advanced standing) should obtain an Application for Advance Credit from the program coordinator (or the course coordinator regarding a general education transfer request) or academic assistant. Students will be required to provide an unofficial transcript and course outline related to the course in question. Please refer to the Student Academic Calendar of Events for the deadline date by which application must be made for advance standing.

Credit for prior learning will also be given upon successful completion of a challenge exam or portfolio.

Substitute course information is available in the Registrar's office.

Disability Services:

If you are a student with a disability (e.g. physical limitations, visual impairments, hearing impairments, or learning disabilities), you are encouraged to discuss required accommodations with your professor and/or the Disability Services office. Visit Room E1101 or call Extension 2703 so that support services can be arranged for you.

Communication:

The College considers **WebCT/LMS** as the primary channel of communication for each course. Regularly checking this software platform is critical as it will keep you directly connected with faculty and current course information. Success in this course may be directly related to your willingness to take advantage of the **Learning Management System** communication tool.

The professor reserves the right to use other tools and/or techniques that may be more applicable. These other tools and/or techniques for effective communication will be discussed, identified and presented throughout the delivery of the course content.

INTRODUCTION TO PROGRAMMING  
COURSE NAME

CSD102  
COURSE CODE

Plagiarism:

Students should refer to the definition of “academic dishonesty” in *Student Code of Conduct*. A professor/instructor may assign a sanction as defined below, or make recommendations to the Academic Chair for disposition of the matter. The professor/instructor may (i) issue a verbal reprimand, (ii) make an assignment of a lower grade with explanation, (iii) require additional academic assignments and issue a lower grade upon completion to the maximum grade “C”, (iv) make an automatic assignment of a failing grade, (v) recommend to the Chair dismissal from the course with the assignment of a failing grade. In order to protect students from inadvertent plagiarism, to protect the copyright of the material referenced, and to credit the author of the material, it is the policy of the department to employ a documentation format for referencing source material.

Student Portal:

The Sault College portal allows you to view all your student information in one place. **mysaultcollege** gives you personalized access to online resources seven days a week from your home or school computer. Single log-in access allows you to see your personal and financial information, timetable, grades, records of achievement, unofficial transcript, and outstanding obligations. Announcements, news, the academic calendar of events, class cancellations, your learning management system (LMS), and much more are also accessible through the student portal. Go to <https://my.saultcollege.ca>.

Electronic Devices in the Classroom:

Students who wish to use electronic devices in the classroom will seek permission of the faculty member before proceeding to record instruction. With the exception of issues related to accommodations of disability, the decision to approve or refuse the request is the responsibility of the faculty member. Recorded classroom instruction will be used only for personal use and will not be used for any other purpose. Recorded classroom instruction will be destroyed at the end of the course. To ensure this, the student is required to return all copies of recorded material to the faculty member by the last day of class in the semester. Where the use of an electronic device has been approved, the student agrees that materials recorded are for his/her use only, are not for distribution, and are the sole property of the College.

INTRODUCTION TO PROGRAMMING  
COURSE NAME

CSD102  
COURSE CODE

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:

<b>Course Hours</b>	<b>Deduction</b>
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

Absentee reports will be discussed with each student. Final penalties will be reviewed and assessed at the discretion of the professor.