SAULT COLLEGE OF APPLIED ARTS AND TECHNOLOGY

SAULT STE. MARIE, ONTARIO

Sault College

COURSE OUTLINE

COURSE TITLE: **INTRODUCTION TO PROGRAMMING**

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

PROGRAM: **ALL COMPUTER STUDIES PROGRAMS**

AUTHOR: **Dennis Ochoski**

DATE: **Jan. 2009**   **PREVIOUS OUTLINE DATED:**   **Jan. 2008**

APPROVED:

"B. Punch"

_____   _____
                      CHAIR   **DATE**

TOTAL CREDITS: 5

PREREQUISITE(S):   **None**

HOURS/WEEK: **5**

INTRODUCTION TO PROGRAMMING   CSD102
_____   _____
    COURSE NAME   COURSE CODE

I. COURSE DESCRIPTION:

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

INTRODUCTION TO PROGRAMMING   CSD102
_____   _____
     COURSE NAME   COURSE CODE

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.
   (chapters 1 and 2, and, lecture notes)

   This learning outcome will comprise **10%** 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
   - 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
   - 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

INTRODUCTION TO PROGRAMMING   CSD102
_____   _____
COURSE NAME   COURSE CODE

Elements of the performance(cont'd):

- 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 (and, to a lesser extent, flowcharts)
- 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. (chapter 3)

    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 flowcharts (and, to a lesser extent, pseudocode)
    - write, test, and debug programs containing decision structures

INTRODUCTION TO PROGRAMMING   CSD102
_____   _____
   COURSE NAME   COURSE CODE

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.
     (chapter 3 and Appendix I)

     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:

                              i.   *while*
                              ii. *do...while*
                              iii. *for*
                              iv. 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 flowcharts (and, to a lesser extent, 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.
     (chapters 4 and 5)

     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
     • understand the concept of *scope*
     • distinguish between *local* and *global* variables

INTRODUCTION TO PROGRAMMING   CSD102
_____   _____
COURSE NAME   COURSE CODE

Elements of the performance(cont'd):

- 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.   (Johnson: chapter 6)

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

INTRODUCTION TO PROGRAMMING   CSD102
_____   _____
 COURSE NAME   COURSE CODE

6. Discuss and apply the concepts of character and string manipulation with reference to
    C/C++ library functions.   (Johnson: chapter 6 and Appendix G)

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

    Elements of the performance:

    • discuss and apply character-based functions such as:

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

    • discuss and apply string functions such as:

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

    • understand and utilize the C++ string class and its associated functions to declare
      string variables and manipulate string values
    • write, test, and debug programs containing character and string functions

7. Develop algorithms to solve problems involving the use of data structures.
    (Johnson: chapter 7)

    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

INTRODUCTION TO PROGRAMMING   CSD102
_____   _____
COURSE NAME   COURSE CODE

8. Develop algorithms to solve problems involving the use of file manipulation.
    (Johnson: Appendix F)

    This learning outcome will comprise approximately **10%** of the course.

    Elements of the performance:

    •   define and apply the concepts of the following terms:

            file   open   read   close   write   append

    •   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

        Text: C++ Programming Today, 2nd Edition
                by Barbara Johnston
                ISBN: 1-13-615099-3

INTRODUCTION TO PROGRAMMING   CSD102
_____   _____
COURSE NAME   COURSE CODE

V. EVALUATION PROCESS/GRADING SYSTEM:

The following is a breakdown of the quiz and assignment distribution as
well as the grade definitions for the course:

Outcome	Assignments	Quizzes	Total
1	6%	5%	11%
2	8%	16%	24%
3	8%	16%	24%
4	6%	10%	16%
5	4%	10%	14%
6, 7 & 8	_4%	_7%	_11%
	36%	64%	100%

Grade	Definition	Grade Point Equivalent
A+	90 -- 100%	4.00
A	80 – 89.9%	4.00
B	70 – 79.9%	3.00
C	60 – 69.9%	2.00
D	50 – 59.9%	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.

INTRODUCTION TO PROGRAMMING   CSD102
_____   _____
        COURSE NAME   COURSE CODE

VI.   SPECIAL NOTES:

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.

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.

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 **LearningManagement 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.

Plagiarism:
Students should refer to the definition of "academic dishonesty" in *Student Rights and Responsibilities*. Students who engage in "academic dishonesty" will receive an automatic failure for that submission and/or such other penalty, up to and including expulsion from the course/program, as may be decided by the professor/dean. 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.

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.

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

INTRODUCTION TO PROGRAMMING   CSD102

_____   _____
COURSE NAME   COURSE CODE

Other Pertinent Information

1.   In order to pass this course the student must obtain an overall 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 quiz, and does not
       notify the professor beforehand of their intended absence, may be subject to
       a zero grade on that 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.   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.

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