SAULT COLLEGE OF APPLIED ARTS AND TECHNOLOGY

SAULT STE. MARIE, ONTARIO

Sault College

COURSE OUTLINE

COURSE TITLE: **COMPUTER PROGRAMMING I**

CODE NO. : **CSD100** **SEMESTER: 1**

PROGRAM: **ALL COMPUTER STUDIES PROGRAMS**

AUTHOR: **Dennis Ochoski**

DATE: **Aug, 2003** **PREVIOUS OUTLINE DATED:** **Aug, 2002**

APPROVED:

_____ _____

DEAN **DATE**

TOTAL CREDITS: **5**

PREREQUISITE(S): **NONE**

HOURS/WEEK: **5**

COMPUTER PROGRAMMING I	CSD100
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. It is the first of two courses that use the C/C++ programming language to develop the student's computer programming and problem solving skills.

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

COMPUTER PROGRAMMING I **CSD100**

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 software to solve problems using the computer. (chapter 1, Appendix C, 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 prewritten software and custom-designed software
   - differentiate between high level languages and machine language
   - describe the top-down process of developing a logical solution to a problem
   - understand the "golden rule" for writing computer programs
   - write algorithms and describe them using flowcharts (and, to a lesser extent pseudocode)

2. Write a simple C/C++ program applying the concepts of input/output, arithmetic, and assignment. (chapters 2 and 3, Appendices H and K)

   This learning outcome will comprise **16%** of the course.

   Elements of the performance:

   - demonstrate a basic understanding of the Microsoft Visual C++ IDE
   - explain the main components of a C/C++ program
   - name and distinguish C/C++'s 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

COMPUTER PROGRAMMING I **CSD100**

COURSE NAME **COURSE CODE**

Elements of the performance(cont'd):

- 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
- explain and apply the #include directive
- explain the purpose of "include" files for the *cin* and *cout* objects
- write, test, and debug programs using the concepts above

3. Develop algorithms and write C/C++ programs to solve problems involving the standard computer operations of decisions/conditions and selection. (chapter 4)

   This learning outcome will comprise **26%** of the course.

   Elements of the performance:

   - describe the use of the relational operators (==, !=, <, <=, >, >=, !|) and use them to write both simple and complex expressions
   - describe the use of the logical operators (&&, ||) and use them to write both simple and complex expressions

Elements of the performance(cont'd):

- describe the operation of the following C/C++ decision-making structures and use them in C/C++ programs:

    a. *if...else*
    b. nested *ifs*
    c. *if...else if...else*
    d. 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

4. Develop algorithms and write C/C++ programs to solve problems involving the standard computer operations of looping and repetition.
(chapter 5)

    This learning outcome will comprise **27%** 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:

        a. *while*
        b. *do...while*
        c. *for*
        d. 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
    - write, test, and debug programs containing repetition structures

COMPUTER PROGRAMMING I **CSD100**

COURSE NAME **COURSE CODE**

6. Discuss and create elementary user-written functions. (chapter 6)

   This learning outcome will comprise **21%** 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
   - 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
   - write, test, and debug programs containing functions

COMPUTER PROGRAMMING I **CSD100**

COURSE NAME **COURSE CODE**

IV. EVALUATION METHODS:

The mark for this course will be arrived at as follows:

Quizzes:		**Assignments:**		**Total:**	
outcome #1	5%	outcome #1	5%	outcome #1	10%
outcome #2	10%	outcome #2	6%	outcome #2	16%
outcome #3	20%	outcome #3	6%	outcome #3	26%
outcome #4	20%	outcome #4	7%	outcome #4	27%
outcome #5	15%	outcome #5	6%	outcome #5	21%
	70%		30%		100%

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

Grade	Definition	Grade Point Equivalent
A+	90 - 100%	4.00
A	80 - 89%	3.75
B	70 - 79%	3.00
C	60 - 69%	2.00
F (Fail)	59% and below	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.

COMPUTER PROGRAMMING I **CSD100**

COURSE NAME **COURSE CODE**

ELIGIBILITY FOR XGRADES/UPGRADING OF INCOMPLETES

When a student's course work is incomplete or final grade is below 60%, there is the possibility of upgrading to a pass when a student meets all of the following criteria:

1. The student's attendance has been satisfactory.
2. An overall average of at least 55% has been achieved.
3. The student has not had a failing grade in all of the theory tests taken.
4. The student has made reasonable efforts to participate in class and complete assignments.

Note: **The opportunity for an X grade is usually reserved for those with extenuating circumstances.** The nature of the upgrading requirements will be determined by the professor and may involve one or more of the following: completion of existing labs and assignments, completion of additional assignments, re-testing on individual parts of the course or a comprehensive test on the entire course.

ASSIGNMENTS

Required format for lab assignments will be detailed by the professor before labs are assigned.

ATTENDANCE:

Absenteeism will affect a student's ability to succeed in this course. Absences due to medical or other unavoidable circumstances should be discussed with the professor. There will be an attendance factor included in the lab evaluation.

COMPUTER PROGRAMMING I **CSD100**

COURSE NAME **COURSE CODE**

V. SPECIAL NOTES

1. In order to pass this course the student must obtain an overall quiz average of **60%** or better, as well as, an overall assignment average of **60%** 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. Assignments must be submitted by the due date according to the specifications of the professor. Late assignments will normally be subject to a penalty of 10% per day late. Late assignments will only be marked at the discretion of the professor in cases where there are extenuating circumstances.

3. Any assignment submissions deemed to be copied will result in a **zero** grade being assigned to **all** students involved in that particular incident.

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

5. If you are a student with special needs (e.g. physical limitations, visual impairments, hearing impairments, or learning disabilities), you are encouraged to discuss required accommodations with your professor and/or the Special Needs office. Visit Room E1204 or call Extension 493 so that support services can be arranged for you.

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

7. 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.

VI. PRIOR LEARNING ASSESSMENT:

Students who wish to apply for advanced credit in the course should consult the professor. Credit for prior learning will be given upon successful completion of a challenge exam or portfolio.

COMPUTER PROGRAMMING I **CSD100**

COURSE NAME **COURSE CODE**

VII. DIRECT CREDIT TRANSFERS:

Students who wish to apply for direct credit transfer (advanced standing) should obtain a direct credit transfer form from the Dean's secretary. Students will be required to provide a transcript and course outline related to the course in question.

VIII. REQUIRED RESOURCES/TEXTS/MATERIALS

Text: Brief Version of Starting Out With C++, 4th edition
by Tony Gaddis
ISBN: 1-57676-121-5

Diskettes: minimum of 3, 3 1/2"