SAULT COLLEGE OF APPLIED ARTS & TECHNOLOGY
SAULT STE. MARIE, ONTARIO

COURSE OUTLINE

Course Title: **COMPUTER PROGRAMMING 2**

Code No.: **CSD101** Semester: **WINTER 2000**

Program: **CPA/CET/CNT/CSST**

Instructor: **DENNIS OCHOSKI**

Date: **JANUARY 2000** Previously Dated: **JANUARY 1999**

Approved: _____ _____
Dean Date

COMPUTER PROGRAMMING 2 — CSD101

COURSE NAME — COURSE CODE

TOTAL CREDITS: 4

PREREQUISITE(S): CSD100

I. COURSE DESCRIPTION: This course is intended to extend the foundation of computer programming skills needed in the computer studies area. It is the second course in the C/C++ programming language, and further develops the student's problem-solving, computer programming, and software utilization skills.

II. TOPICS TO BE COVERED:

1. Advanced data-manipulation operators and library functions.
2. User-defined functions.
3. Arrays/Tables.
4. Pointers and strings.
5. Data structures.
6. Files.

COMPUTER PROGRAMMING 2 CSD101

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 of additional C/C++ operators and library functions used to manipulate character, string, and numeric data.
   (unit 9: pgs. 185-191, unit 10: pgs. 196-206, unit 15)

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

   Elements of the performance:

   - define and apply the concepts of the following terms:

conditional operators (? :)		increment/decrement operators (++, --)
TRUE	bitwise OR	bit shifting
FALSE	bitwise XOR	bitwise complement
bit manipulation	bitwise AND	

   - apply conditional operators to relational tests
   - apply increment/decrement operators to C expressions
   - discuss the concept of truth tables
   - apply bitwise operators
   - discuss and apply additional standard library functions found in the <math.h>, <string.h>, and <ctype.h> libraries of Turbo C++, and how to determine the libraries that are available and which library a particular function is located

   - discuss and apply character-based I/O functions such as:

get( )	getche( )	put( )	putchar( )	tolower( )
getch( )	getchar( )	putch( )		toupper( )

   - discuss and apply character-testing functions such as:

isalpha( )	isalnum( )	islower( )
isdigit( )		isupper( )

Elements of the performance(cont'd):

- discuss and apply string functions such as:

  strcat( ) strcmp( ) strlen( ) strcpy( )

- discuss and apply math functions such as:

  ceil( ) pow( ) rand( )
  floor( ) sqrt( ) srand( )

- write, test, and debug programs using the above operators and functions

2. Discuss and create user-written, independently-compiled functions. (unit 16)

   This learning outcome will comprise approximately **35%** of the course.

   Elements of the performance:

   - define and apply the concepts of the following terms:

scope	calling vs called functions	pointers
local vs global variables	pass by value	address operator
class	pass by reference	
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

   Elements of the performance(cont'd):

   - discuss and apply the concept of ‘returning’ values to calling functions
   - discuss and apply the concepts of ‘passing’ arguments to called functions by reference
   - write, test, and debug programs containing functions

3. Develop algorithms and write C programs to solve problems involving tables/arrays.

(unit 17)

This learning outcome will comprise approximately **10%** 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
- pass arrays between C functions
- write, test, and debug programs containing arrays

4. Develop algorithms to solve problems involving the use of pointers, with specific application string manipulation. (unit 18)

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

   Elements of the performance:

   - discuss and apply the concept of pointers and pointer arithmetic
   - apply the concept of pointers to arrays
   - discuss and apply the concept of strings and pointers in C/C++
   - discuss and apply the use of the following string functions: strcpy, strcat, strcmp
   - write, test, and debug programs using pointers and strings

5. Develop algorithms to solve problems involving the use of data structures.
   (units 19 and 21)

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

Elements of the performance:

- define and apply the concepts of the following terms:

structure	record	append
member	open	internal pointer
record	close	

- discuss the concept of structures in C/C++
- apply the use of arrays of structures
- discuss and apply methods of passing and returning structures to and from functions
- write, test, and debug programs containing structures

6. Develop algorithms to solve problems involving the use of file manipulation. (units 19 and 21)

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

Elements of the performance:

- create a disk file
- write data to, and, read data from a disk file
- perform disk I/O with records
- discuss and apply the use of the following functions: stdin, stdout, and stderr
- understand, create, and manipulate sequential files
- write, test, and debug programs containing files

COMPUTER PROGRAMMING 2 CSD101

COURSE NAME COURSE CODE

IV. EVALUATION METHODS:

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

Quizzes:	
outcome #1	10%
outcome #2	25%
outcomes #3 & #4	15%
outcomes #5 & #6	15%
Assignments:	
outcome #1	5%
outcome #2	10%
outcomes #3 & #4	10%
outcomes #5 & #6	10%
Total	100%

The grading scheme used will be as follows:

A+	90 - 100%	Outstanding achievement
A	80 - 89%	Excellent achievement
B	70 - 79%	Average achievement
C	60 - 69%	Satisfactory achievement
R	Repeat	
X	Incomplete	A temporary grade limited to special circumstances that have prevented the student from completing objectives by the end of the semester. An X grade reverts to an R grade if not upgraded within a specified time.

COMPUTER PROGRAMMING 2 CSD101

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%.

2. Assignments must be submitted by the due date according to the specifications of the instructor. Late assignments will be penalized at a 10% reduction per day, up to 3 days late, after which the assignment will be given a mark of zero. Assignments submitted more than 3 days late may be marked at the discretion of the instructor in cases where there are extenuating circumstances.

3. The instructor reserves the right to modify the assessment process to meet any changing needs of the class. Consultation with the class will be done prior to any changes.

4. Their will be **no** re-write of any quiz unless the instructor feels there are extenuating circumstances.

5. Students with special needs (eg. physical limitations, visual impairments, hearing impairments, learning disabilities) are encouraged to discuss required accommodations confidentially with the instructor.

6. Your instructor reserves the right to modify the course content as he/she deems necessary to meet the needs of students.

VI. PRIOR LEARNING ASSESSMENT:

Students who wish to apply for advanced credit in the course should consult the instructor.

VII. REQUIRED STUDENT RESOURCES

Text: Turbo C++ Programming in 12 Easy Lessons
by Greg Perry

Diskettes: minimum of 3, 3 1/2"