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

COURSE OUTLINE

COURSE TITLE:	INTRODUCTION TO COBOL PROGRAMMING		
CODE NO.:	CSD208	**SEMESTER:**	FALL 98
PROGRAM:	COMPUTER PROGRAMMER/PROGRAMMER ANALYST		
AUTHOR:	DENNIS OCHOSKI		
DATE:	JUNE 1998	**PREVIOUSLY DATED:**	JUNE 1997

APPROVED: [illegible]
DEAN

980527
DATE

INTRODUCTION TO COBOL PROGRAMMING	CSD208
COURSE NAME	COURSE CODE

TOTAL CREDITS: 5

PREREQUISITE(S): CSD101

I. COURSE DESCRIPTION: This course is designed to further develop the student's programming skills using a 3rd generation language, COBOL. The course will focus on the program development process, re-emphasizing the use of structured programming techniques, and the solution to traditional business programs. It will extend the concepts taught in CSD100/101 to include such topics as data editing, file handling, sorting, table handling, subprograms, and screen management.

II. TOPICS TO BE COVERED:

1. COBOL Program Structure.
2. Input/Output Operations.
3. Decisions/Conditions.
4. Repetition/Looping.
5. Data Editing.
6. Table Handling.
7. Sorting.
8. Control Break Reporting.
9. File Handling.
10. Screen Management.
11. Subprograms.
12. Year 2000 Dilemma.

III. LEARNING OUTCOMES AND ELEMENTS OF THE PERFORMANCE:

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

1. Apply the basic programming concepts learned in CSD100/101, in order to write programs incorporating structure, arithmetic, assignment, input/output, conditions, and looping. (Grauer: chapters 1, 2, 3, 4, 5, 6, 7, and 9)

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

   Elements of the performance:

   - define the terms: field, record, and file
   - identify the four divisions of a COBOL program
   - state the six COBOL language elements
   - state the rules for creating a programmer-defined name; distinguish between valid and invalid names
   - state the difference between numeric and nonnumeric literals
   - state the rules associated with the COBOL coding sheet, and enter a program appropriately
   - distinguish between compilation and execution; describe the function of a link program
   - compile, link, and execute a COBOL program
   - find and correct errors in compilation or execution
   - describe how a hierarchy chart is developed; discuss three criteria for evaluating a completed hierarchy chart
   - explain the one entry point/one exit point philosophy of structured programming
   - differentiate between structured programming and structured design; distinguish between a functionally oriented technique and one that is procedurally oriented
   - describe what is meant by top down design and implementation
   - describe the COBOL notation and determine the proper syntax for any statement
   - complete the Identification and Environment Divisions of a COBOL program
   - understand and implement sequential file processing concepts
   - code a record description
   - code a Working-Storage Section to define various print lines
   - explain the use of an assumed decimal point
   - write the OPEN, CLOSE, READ, and WRITE statements necessary for sequential file processing
   - describe the purpose of the priming READ, and place it correctly in the Procedure Division
   - discuss the purpose of the MOVE statement as it applies to numeric and alphanumeric fields

Elements of the performance(cont'd):

- describe the PERFORM statement; show how it is used to process a file until all of its records have been read
- describe the IF statement and how it is used with and without an ELSE clause; explain the significance of the END-IF scope terminator
- use the EVALUATE statement to implement a case construct
- state the hierarchy of operations for a COMPUTE statement; describe the individual arithmetic statements, ADD, SUBTRACT, MULTIPLY, and DIVIDE
- decribe the ROUNDED and SIZE ERROR options as they apply to any of the arithmetic statements
- explain the relationship between a Procedure Division and its associated hierarchy chart
- use the DISPLAY statement as a debugging tool
- explain how an interactive debugger can be used to find and correct errors
- describe the use of file status codes in correcting data management errors
- explain what is meant by a structured walkthrough
- list the complete set of COBOL editing characters
- differentiate between a numeric field and a numeric-edited field; predict the results when a numeric field is moved to a numeric-edited field
- underdstand the difference between an implied decimal point and an actual decimal point; state the role of each in editing
- describe the rules for signed numbers and the editing characters +, -, CR, and DB
- describe the rationale for coding standards that go beyond the syntactical requirements of COBOL
- differentiate between the DO WHILE and DO UNTIL structures; describe how each is implemented in conjunction with a PERFORM statement
- explain the relationship between a Procedure Division and its associated hierarchy chart
- define an in-line perform and a false-condition branch; explain how the combination of these features eliminates the need for a priming read statement
- differentiate between a paragraph and a section
- code the READ INTO and WRITE FROM statements in the Procedure Division
- use the INITIALIZE statement
- perform basic string processing operations through the use of the INSPECT, STRING, and UNSTRING statements
- define a duplicate data name and use qualification to eliminate ambiguity; describe the use of the MOVE CORRRESPONDING statement

2. Apply techniques to validate input data.
(Grauer: chapter 8)

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

Elements of the performance:

- describe the importance of data validation and its implementation in a stand-alone edit program
- define the following validity tests; numeric test, alphabetic test, consistency check, sequence check, completeness check, date check, and subscript check
- describe the various types of conditions in an IF statement
- define a nested IF; indicate guidelines for proper indentation in coding such statements
- describe the advantage of the END-IF scope terminator; show how it eliminates the need for the NEXT SENTENCE clause
- obtain the date (calendar and Julian) and time of execution; implement date checking in a program to ensure that the day and month are consistent

3. Apply array processing techniques to manipulate data in one dimensional tables.
(Grauer: chapters 11 and 12)

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

Elements of the performance:

- define a table and describe its use in programming
- use the OCCURS (at either the group or elementary level) to implement a table in COBOL
- use the PERFORM VARYING statement to process a table
- distinguish between fixed and variable length records; use the OCCURS DEPENDING ON clause to implement a variable length table
- state the purpose of the USAGE clause
- differentiate between a subscript and an index
- define a table lookup and describe why it is used
- distinguish between a numeric, alphabetic, and alphanumeric code; describe several attributes of a good coding system
- distinguish between a sequential table lookup, a binary table lookup, and direct access to table entries

Elements of the performance(cont'd):

- distinguish between a table that is hard coded versus one that is input loaded
- state the purpose of the VALUE, OCCURS, and REDEFINES clauses as they pertain to table definition and initialization
- define a range-step table
- code SEARCH and SEARCH ALL statements to implement table lookups

4. Apply techniques used for sorting data records before processing, and, efficient printing of group reports and control totals.
(Grauer: chapters 14 and 15)

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

Elements of the performance:

- distinguish between an internal sort, a utility sort, and the COBOL SORT statement
- differentiate between an ascending and a descending sort; between major, intermediate, and minor sort keys; and between primary, secondary, and tertiary keys
- define collating sequence; discuss the most signifcant differences between EBCDIC and ASCII and how the collating sequence affects fields with an embedded sign
- explain the syntax of the COBOL SORT statement, and the supporting RELEASE, RETURN, and SD statements
- explain the use of INPUT PROCEDURE to sort on a calculated field, and/or to selectively pass records to the sort work file
- distinguish between a merge and a sort
- define control break; distinguish between a single-level and a multi-level control break
- explain the relationship between sorting and control breaks
- design a hierarchy chart and pseudocode to implement any number of control breaks; evaluate it for completeness, functionality, and span of control
- use a general purpose algorithm to write a COBOL program for any number of control breaks
- write a COBOL program for one- and two-dimensional control breaks
- distinguish between rolling and running totals

5. Apply techniques to process sequential and indexed sequential files
(Grauer: chapters 17 and 18)

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

Elements of the performance:

- describe the file maintenance operation; distinguish between the old master, transaction, and new master files
- describe the three transaction types associated with file maintenance
- differentiate between sequential and nonsequential file maintenance
- describe at least three types of errors that can be detected in a standard edit program; list two errors that cannot be detected in such a program
- discuss the balance line algorithm
- define top-down testing; explain how a program may be tested before it is completely coded
- describe how an index file enables both sequential and/or nonsequential retrieval of individual records
- discuss the clauses in the SELECT statement for an indexed file; indicate which clauses are optional and which are required
- define file status bytes; state how they may be used to verify the success of an I/O operation
- differentiate between the READ statements for sequential and nonsequential access of an indexed file
- differentiate between the WRITE, REWRITE, and DELETE statements as they apply to the file maintenance of an indexed file
- describe the syntax of the START statement and give a reason for its use
- distinguish between the primary and alternate keys of an indexed file

6. Apply simple on-line programming techniques to process data in an on-line environment.
(Grauer: chapter 10)

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

Elements of the performance:

- discuss the concept of screen I-O versus the file-oriented approach
- describe the ACCEPT and DISPLAY statements
- describe the SCREEN SECTION and indicate why its use may be preferrable to individual ACCEPT and DISPLAY statements

Elements of the performance(cont'd):

- differentiate between the background and foreground colours; implement a colour scheme using ACCEPT and DISPLAY statements and/or Screen Section
- describe how interactive data validation is implemented in a screen I-O program; contrast this technique to the batch-oriented procedure

7. Apply techniques to execute called programs as subroutines(subprograms).
   (Grauer: chapter 16)

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

   Elements of the performance:

   - define a subprogram and describe its implementation in COBOL
   - distinguish between a called and a calling program; describe the use of a hierarchy chart to show the relationship of programs within a system
   - state the purpose of the COPY statement; indicate where it may be used within a program and how it can be used to pass a parameter list
   - distinguish between the BY CONTENT and BY REFERENCE clauses as they relate to subprograms
   - explain the function of the INITIAL phrase in the PROGRAM-ID paragraph
   - describe the purpose of the linkage-editor

8. Understand how the Year 2000 problem will affect businesses, and, formulate a plan of action to correct this problem.
   (Grauer: chapter 19)

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

   Elements of the performance:

   - describe the implications of the Year 2000 problem
   - state the causes of the problem
   - identify the types of routines that may cause the problem
   - discuss several types of date arithmetic
   - use COBOL intrinsic calendar functions to do date conversions

IV. EVALUATION METHODS:

Tests:

Test #1 (10%):	outcome #1	10%
Test #2 (15%):	outcome #1	15%
Test #3 (14%):	outcome #2	7%
:	outcome #3	7%
Test #4 (17%):	outcome #4	10%
:	outcome #5	7%
Test #5 (13%):	outcome #6	7%
:	outcome #7	3%
:	outcome #8	3%
		69%

Assignments:

Asgn #1 (4%):	outcome #1	4%
Asgn #2 (6%):	outcome #1	6%
Asgn #3 (6%):	outcome #2	3%
:	outcome #3	3%
Asgn #4 (8%):	outcome #4	5%
:	outcome #5	3%
Asgn #5 (7%):	outcome #6	3%
:	outcome #7	2%
:	outcome #8	2%
		31%
	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 the objectives by the end of the semester. An X grade reverts to an R grade if not upgraded within a specified time period.

VI. SPECIAL NOTES

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

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

3. The instructor reserves the right to modify the assessment process.

4. The method of upgrading an incomplete grade is at the discretion of the instructor, and may consist of such things as make-up work, rewriting tests, and comprehensive examinations.

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 as he/she deems necessary to meet the needs of students.

VII. PRIOR LEARNING ASSESSMENT:

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

VIII. REQUIRED STUDENT RESOURCES

Texts: COBOL: From Micro to Mainframe - Preparing for the New Millennium, 3rd edition,
by Robert T. Grauer, Carol Vazquez Villar, Arthur R. Buss
Prentice Hall Publishing