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

COURSE OUTLINE

Course Title: ADVANCED DESIGN AND IMPLEMENTATION

Code No.: CSD314 **Semester:** WINTER 1999

Program: PROGRAMMER ANALYST (2091)

Instructor: DENNIS OCHOSKI

Date: JANUARY 1999 **Previously Dated:** NEW

Approved: K. DeRusario
Dean

Jan. 14/99
Date

ADVANCED DESIGN AND IMPLEMENTATION — COURSE NAME

CSD314 — COURSE CODE

TOTAL CREDITS: 5

PREREQUISITE(S): CSD202, CSD208

I. COURSE DESCRIPTION: Software and hardware trends change so rapidly that by the time the third year student is ready to graduate they may miss out on skills that the industry demands immediately for employment. Therefore, this course will focus on capturing popular and current industry trend development areas to ensure the marketability of the Computer Analyst graduate. Areas of study may focus on WW applications, advanced OOP program development, or anything else that the corporate computing industry has embraced.

In the this particular semester the student will refine their COBOL programming skills. Although COBOL is a 3rd generation language and it is not widely used for new development today, many existing systems throughout the world have COBOL incorporated in them. Another major focus for this semester will be the "Year 2000 Problem" and its effect on current computerized systems. Many of these systems in jeopardy were written in COBOL, therefore the emphasis with respect to the Y2K problem will be COBOL related.

II. TOPICS TO BE COVERED:

1. Review of COBOL concepts discussed in CSD208.
2. Editing.
3. Sorting.
4. Control Break Reporting.
5. ISAM File Processing.
6. Screen Management.
7. Table/Array Processing.
8. Subprograms.
9. Date Representation in COBOL.
10. Date Calculations.
11. The Year 2000 Problem.
12. Solutions to the Year 2000 problem.

ADVANCED DESIGN AND IMPLEMENTATION | CSD314

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. Review the basic programming concepts learned in CSD208, in order to write programs incorporating structure, arithmetic, assignment, input/output, conditions, and looping. (Wessler: chapters 1, 2, 3, 4, 5, and 6)

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

   Elements of the performance:

   - identify the four divisions of a COBOL program
   - state the difference between numeric and nonnumeric literals
   - state the rules associated with the COBOL coding sheet, and enter a program appropriately
   - 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
   - write 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
   - describe the PERFORM statement; used it to transfer control to other parts of a program
   - 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
   - use the COMPUTE statement and describe the individual arithmetic statements: ADD, SUBTRACT, MULTIPLY, and DIVIDE
   - describe the ROUNDED and SIZE ERROR options as they apply to 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
   - list the 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

Elements of the performance(cont'd):

- understand the difference between an implied decimal point and an actual decimal point
- describe the rules for signed numbers and the editing characters +, -, CR, and DB
- differentiate between the 'do while' and 'do until' structures; describe how each is implemented in conjunction with a PERFORM statement
- 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
- define a duplicate data name and use qualification to eliminate ambiguity; describe the use of the MOVE CORRRESPONDING statement
- 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

2. Apply techniques used for sorting data records , and, efficient printing of group reports and control totals.
(Wessler: chapter 15)

This learning outcome will comprise approximately **20%** 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 and minor sort keys
- define collating sequence; discuss the most significant differences between EBCDIC and ASCII
- explain the syntax of the COBOL SORT statement, and the supporting RELEASE, RETURN, and SD statements
- explain the use of INPUT PROCEDURE to selectively pass records to the sort work file, and, an OUTPUT PROCEDURE to process record after they are sorted
- 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
- 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

3. Apply techniques to process indexed sequential files
(Wessler: chapter 7)

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

Elements of the performance:

- differentiate between sequential and nonsequential file maintenance
- 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

4. Apply simple on-line programming techniques to process data in an on-line environment.
(Wessler: chapter 25)

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 preferable to individual ACCEPT and DISPLAY statements
- 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

5. Apply array processing techniques to manipulate data in one dimensional tables. (Wessler: chapter 11)

   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
   - 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

6. Apply techniques to execute called programs as subroutines(subprograms). (Wessler: chapter 14)

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

Elements of the performance(cont'd):

- 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

7. Understand how the Year 2000 problem will affect business and industry, and, formulate a plan of action to correct this problem.
(Wessler: chapters 13, 30, 31, 32)

This learning outcome will comprise approximately **30%** of the course.

Elements of the performance:

- discuss the history of the modern calendar used in computer dating
- understand and define the correct rules for determining a leap year
- understand and describe date formats and calendars used in computer dating including:

the Julian calendar	the Gregorian calendar	Lilian dates
Julian period dating	Gregorian format dates	ANSI/ISO dates
Julian format dates		

- use computer dating to calculate the day of the week and to add days to calculate a new date
- understand the year 2000 problem and its causes, and, describe its implications
- use COBOL intrinsic functions to help solve year 2000 problems
- understand solutions currently being used in the industry for the year 2000 problem

ADVANCED DESIGN AND IMPLEMENTATION | CSD314

COURSE NAME | COURSE CODE

IV. EVALUATION METHODS:

Quizzes:

Quiz #1 (5%)	:outcome #1	5%
Quiz #2 (10%)	:outcome #2	10%
Quiz #3 (10%)	:outcome #3	5%
	:outcome #4	5%
Quiz #4 (10%)	:outcome #5	5%
	:outcome #6	5%
Quiz #5 (10%)	:outcome #7	10%
		45%

Assignments:

Asgn #1 (5%)	:outcome #1	5%
Asgn #2 (10%)	:outcome #2	10%
Asgn #3 (10%)	:outcome #3	5%
	:outcome #4	5%
Asgn #4 (10%)	:outcome #5	5%
	:outcome #6	5%
Asgn #5 (10%)	:outcome #7	10%
Asgn #6 (10%)	:outcome #7	10%
		55%

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.

ADVANCED DESIGN AND IMPLEMENTATION | CSD314

COURSE NAME | COURSE CODE

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 Unleashed: The Comprhensive Solution,
by Jon Wessler, et al.
Prentice Hall Publishing