LSC Use Only
Number: _____
Submission Date: _____
Action-Date: _____

RECEIVED FEB - 8 2001 LIBERAL STUDIES

UWUCC USE Only 01-126
Number: ~~00-551~~
Submission Date: _____
Action-Date: App 10/9/01
Senate App 12/4/01

CURRICULUM PROPOSAL COVER SHEET
University-Wide Undergraduate Curriculum Committee

I. CONTACT

Contact Person: William W. Oblitey Phone: 7-4491

Department Computer Science

II. PROPOSAL TYPE (Check All Appropriate Lines)

_____ **COURSE** APPL COMPUTER PROGMG
Suggested 20 character title

___ New Course* _____
Course Number and Full Title

_X_ Course Revision COSC 220 Applied Computer Programming
Course Number and Full Title

RECEIVED OCT - 3 2001 LIBERAL STU

___ Liberal Studies Approval+ for new or existing course _____
Course Number and Full Title

___ Course Deletion _____
Course Number and Full Title

___ Number and/or Title Change _____
Old Number and/or Full Old Title

_____
New Number and/or Full New Title

___ Course or Catalog Description Change _____
Course Number and Full Title

Rev.

_____ **PROGRAM:** _____ Major _____ Minor _____ Track

RECEIVED NOV 19 2001 LIBERAL STUDIES

___ New Program* _____
Program Name

___ Program Revision* _____
Program Name

___ Program Deletion* _____
Program Name

___ Title Change _____
Old Program Name

_____
New Program Name

III. Approvals (signatures and date)

Ty Watts
Department Curriculum Committee

G. L. Bretthauer
Department Chair

02/08/01
College Curriculum Committee

William D. Eddy
College Dean

+Director of Liberal Studies (where applicable)

*Provost (where applicable)

Part II. Description of Curriculum Change

1. New Syllabus of Record

See Attachment A the new syllabus of record.

2. Summary of the proposed revision.

The new course is a combination of the salient topics in the old COSC 220 and COSC 315 courses. The number of credit hours has been increased from 3sh to 4sh.

3. Justification for the revision

Due to the increase of topics and subject areas in the computing discipline, we have found it necessary to combine most of the contents of COSC 220 and COSC 315 to enable us address the important topics that students need without extending the total number of credits required. COSC 315 will thus become inactive. If it should be reactivated, it will have to be revised.

4. Old Syllabus of Record

See Attachment B the old syllabus of record.

5. Letters of Support

N/A

ATTACHMENT A

Indiana University of Pennsylvania November 9, 2001
Computer Science Department

Course Revision

COSC 220 - Applied Computer Programming

I. Catalog Description

COSC 220	Applied Computer Programming	4 credits 0 lab hour 4 lecture hours 4c-0l-4sh

Prerequisites: COSC 110 or equivalent.

Structured programming principles and techniques, as implemented through the ANSI COBOL language; program design using top-down techniques; program and project documentation; introduction to sequential and random file algorithms and integrated file systems.

II. Course Objectives

Upon successful completion of this course, the student should be able to:

A. Develop data definitions and record structures from user problem specifications.

B. Distinguish the environmental differences between various computing platforms as they relate to the execution of COBOL programs.

C. Compile, link, and execute COBOL programs using available options.

D. Recognize and use the correct COBOL syntax.

E. Recognize how COBOL employs structured programming techniques to solve programming oriented problems.

F. Differentiate between how COBOL handles the different file maintenance techniques.

G. Design user-friendly screen displays for use with interactive COBOL programs.

III. Detailed Course Outline

1. Introduction to Structured Programming. (3 hours)
An insight into the process of designing and writing computer programs; the four phases of program development; the use of pseudocode and flowcharting in program development; use of structure charts in program design; modularization and module-numbering conventions. A brief look at COBOL from the historical perspective of programming languages and the tenacity of the COBOL language.

2. The COBOL Language Overview (3 hours)
COBOL format and terminology. A look at the Divisions of the COBOL language and an in-depth study of the contents of the first two Divisions. An exhaustive study of the COBOL metalanguage.

3. COBOL Data Types (4 hours)
The basic components of the COBOL DATA Division; COBOL data descriptions; Descriptions of input and output records. Fixed-length versus variable-length records. COBOL data storage types and internal data representation. A discussion of the most significant differences between ASCII and EBCDIC collating sequences.

4. The Procedure Division and Common Data Manipulation Verbs (3 hours)
Development of a full COBOL program; processing of loops in COBOL. An explanation for the use of the priming read in COBOL. A look at some COBOL verbs: OPEN, READ, MOVE, WRITE, CLOSE. Simple conditional operations using the IF statement. A look at simple arithmetic operations in COBOL.

5. Data Storage and Editing Standards (3 hours)
A more exhaustive look at the categories of PICTURE clauses. A study of the complete set of COBOL editing characters. Distinguishing between numeric and numeric-edited fields. Explain the difference between implied and actual decimal points. Learn the rules for signed numbers and editing characters. Predict the effect of data movement between fields.

6. Conditional Operations and Data Validation (3 hours)
Study of the various types of condition tests: relation, class, sign, and condition-name. A look at various validity tests in COBOL: numeric test, alphabetic test, consistency check, sequence check, completeness check, date check, and subscript check. Examination of compound conditions in COBOL and a description of the advantages of the END-IF and other scope terminators.

7. COBOL String Processing and Reference Modification (4 hours)
Differentiation between the DO WHILE and the DO UNTIL structures in COBOL. A look at some COBOL string processing statements like the INITIALIZE, INSPECT, STRING, UNSTRING, ACCEPT, and DISPLAY

statements. A study of the basic reference modification principles.

8. Table Processing (4 hours)
Table definition and description of table use in COBOL programming. Distinguishing between fixed and variable length records. Differentiating between a subscript and an index. Distinguishing between sequential table lookup, binary table lookup, and direct access to table entries.

9. Multilevel Table Processing (3 hours)
A conceptual view of multidimensional tables. Implementation of a multidimensional table. Searching a multidimensional table.

10. File Sorting and Merging (4 hours)
Distinguishing between an internal sort, a utility sort, and the COBOL SORT statement. Differentiating between ascending and descending sorts; between major, intermediate, and minor keys. Defining a collating sequence. Explanation of the use of INPUT PROCEDURE and OUTPUT PROCEDURE routines in the SORT statement. Distinguishing between a sort and a merge.

11. Report Concepts and Control Break Programming (6 hours)
An examination of the general common areas of a good report design. A look at the general guidelines for report design. Defining control break; distinguishing between a single control break and multilevel control break programs. Distinguishing between running and rolling totals. Study of the single-level and two-level control break algorithms. Generalization to three-level, etc. algorithms.

12. COBOL Subprograms (4 hours)
Definition and implementation of subprograms in COBOL. Distinguishing between a called program and a calling program. Description of the purpose of the linkage-editor. The use of BY CONTENT and BY REFERENCE clauses as they relate to subprograms. Explanation of the meaning of unresolved external reference. Compilation, linkage and running of COBOL subprograms.

13. Sequential File Maintenance (4 hours)
Description of the file maintenance operation; distinguishing between old master, transaction, and new master files. Explanation of the three transaction types associated with file maintenance. Explaining how HIGH-VALUES and LOW-VALUES are used with the sequential update process. The basic sequential update algorithm.

14. Indexed File Processing (4 hours)
Explanation of how an indexed file enables both sequential and non-sequential retrieval of individual records. Introduction to indexed and virtual sequential file implementations. Distinguishing between indexed and virtual sequential file organizations. Study of Indexed file syntax in the ENVIRONMENT and DATA Divisions. Indexed file verbs in the PROCEDURE Division. Programming specifications for interactive file maintenance.

15. Interactive Program Design in COBOL (4 hours)
Discussion of the concept of interactive screen I/O versus the batch I/O approach. Description of the SCREEN Section and a discussion of how its use may be preferable to the individual ACCEPT and DISPLAY statements. Application of ergonomics in interactive program design. Explanation of the beneficial features of data validation using interactive programming as opposed to batch-oriented programming

IV. Evaluation Methods

20% Homework assignments, Class work, and Quizzes. These will be based on material discussed in class.

40% Examinations. Two in-class exams (each of which counts 29% of the total examination points) and a final exam (which counts 42% of the total examination points). The examinations consist of short-answer, explanation, analysis, and what-if questions.

40% Programming Projects. About six to seven projects of varying complexity each based on topics discussed so far in the semester.

Grading Scale: The standard grading scale will be used.
90%+=A; 80-89%=B; 70-79%=C; 60-69%=D; below 60%=F.

Suggested Projects:

Project #1: (duration: One week)
Compile/Link/Execute: Project comprises of a simple program furnished by the professor preferably with introduced errors. Students are informed of the errors and they employ the Language Sensitive Editor to eliminate the errors. Students then compile, link and execute the program for submission. Project should stress the use of the AT END scheme for detecting the end of file in program execution.

Project #2: (duration: One week)
A simple COBOL program. Project comprises of simple data manipulation, preferably just moving input fields to output fields. Professor may introduce simple arithmetic with the project. Students code the IDENTIFICATION, ENVIRONMENT, and DATA DIVISIONS; the PROCEDURE DIVISION will be furnished by the professor.

Project #3: (duration: Two weeks)
A simple COBOL report program with calculations. Project comprises of a COBOL program that does arithmetic, preferable with some decision structures involved. Professor supplies the input file and the program specifications. The specification should include the requirement for a formatted output. Students will have to code the entire program.

Project #4: (duration: Two weeks)
A table processing COBOL program. Project should require students to write a COBOL program that does table processing (preferably multi-level table processing using PERFORM VARYING in several dimensions). Professor supplies the program specifications and students code the program.

Project #5: (duration: Two weeks)
A COBOL control break program. Project would require students to code a multi-level control break program (preferable a two-level control break). The input file supplied to students need not be sorted and the project specification should demand that students sort the input file before its use in the program (preferable coding an INPUT PROCEDURE with the program).

Project #6: (duration: Two weeks)
A Sequential File Update program. Project would require students to update a master file with transactions either from a file or provided for interactive update. Preferably the specifications would require that student programs be able to handle multiple transactions belonging to particular records. Project may involve merging files to constitute the transactions or perhaps sorting the transaction file or both the master and transaction files.

Project #7: (duration: Two weeks)
A Nonsequential File Update program. Project would require students to develop an interactive program that would randomly update a master file with transactions.

V. Required Textbook(s), Supplementary Books and Readings

Grauer, Robert T. and Villar, C. V., *COBOL: From Micro to Mainframe.* Fourth Edition, Prentice-Hall Inc. Englewood Cliffs, New Jersey, April, 2000.

Keogh, J. E., and Keogh, J., *COBOL Programmer's Notebook.* Prentice-Hall Inc. Englewood Cliffs, New Jersey, 1998.

VI. Special Resource Requirements

None.

VII. Bibliography

1. Abel, Peter, COBOL Programming: A Structured Approach, Prentice-Hall Inc. Englewood Cliffs, New Jersey 1988.

2. Baroundi, Carol, Mastering COBOL, Sybex Inc., San Francisco, California, 1999.

3. Brown, Gary, D. Advanced Cobol for Structured and Object-Oriented Programming, John-Wiley & Sons, New York, NY, 1998.

4. Collopy, David M., Introduction to Cobol : A Guide to Modular Structured Programming, Prentice-Hall Inc. Englewood Cliffs, New Jersey 1999.

5. Doke, E. Reed & Hardgrave, Bill B., An Introduction to Object COBOL, John Wiley & Sons, New York, NY, 1998.

6. Gleason, Gary M. and Horn, L. W., Comprehensive Structured COBOL, 3rd Edition, Course Technology, Cambridge, MA 1997.

7. Grauer, Robert T. and Villar, C. V., COBOL: From Micro to Mainframe, Prentice-Hall Inc. Englewood Cliffs, New Jersey 1994.

8. Keogh, James E., & Keogh, Jim, COBOL Programmer's Notebook, Prentice Hall Inc. Englewood Cliffs, New Jersey 1998.

9. Levey, Robert, Reengineering COBOL with Objects: Step-by-Step to Sustainable Legacy Systems, McGraw Hill, Berkeley, California, 1996.

10. Noll, Paul, Structured COBOL Methods, Mike Murack & Associates, 1997.

11. Shelly, Gary B., Structured COBOL Programming: with Microfocus for Windows 1.1, Course Technology, Cambridge, MA 1996.

12. Welburn, Tyler. and Price, Wilson, Structured COBOL: Fundamentals and Style, McGraw-Hill, Inc. San Francisco, California 1995.

13. Wessler, Jon, COBOL Unleased, Sams Publishing, Indianapolis, IN, 1998.

14. Yarmish, Rina et. al., Structured COBOL: A Direct Approach, Prentice-Hall Inc., Englewood Cliffs, New Jersey 1993.