RECEIVED
OCT 20 1995
LIBERAL STUDIES

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

UWUCC USE Only
Number: 95-42
Submission Date: ______
Action-Date: App 12/12/95
senate App 2/6/96

CURRICULUM PROPOSAL COVER SHEET
University-Wide Undergraduate Curriculum Committee

I. CONTACT

Contact Person James L. Wolfe Phone 6104

Department Computer Science

II. PROPOSAL TYPE (Check All Appropriate Lines)

X COURSE PRB SLVG & STRC PRGM
Suggested 20 character title

___ New Course* ______
Course Number and Full Title

X Course Revision CO 110 Problem Solving and Structured Programming
Course Number and Full Title

___ 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

X Course or Catalog Description Change CO 110 Problem Solving and Structured Programming
Course Number and Full Title

___ PROGRAM: ___ Major ___ Minor ___ Track

___ 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)

[illegible]
Department Curriculum Committee

Gary L Buterbaugh
Department Chair

[illegible]
College Curriculum Committee

John D. Eck 10/20/95
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.

2. Summary of the proposed revision

   The principal revision is to change the programming language used in the course from FORTRAN to C++. The techniques and concepts taught, as well as the overall approach for the course, remains the same; only the language changes. There have been adjustments in the syllabus terminology to reflect the new language and adjustments in language processing to reflect the fact that C++ is to be taught on microcomputers rather than on the mainframe.

3. Justification for the revision

   FORTRAN is a programming language that is not being used as widely as in years past. It is used only in specialized situations - heavy mathematical applications and some parallel computations. As FORTRAN's use has been fading, the use of C (and C++, a superset of C), has been on the up-swing. C and C++ have become a worldwide choice for program development of nearly all types, even mathematical applications.

   Object-oriented programming has for several years been widely touted as a very important programming and design approach. To keep up with the changing field of computer science, we must find methods of including the object-oriented approach in our degree programs. The programming language C++ was developed for use in object-oriented programming.

   By changing the language used in CO 110, we can catch up with a major programming trend in the computer field and, at the same time, lay the ground work for incorporating the object-oriented approach into our curriculum. This approach will actually be taught in CO 310.

4. Old Syllabus of Record

   See Attachment B.

5. Letters of Support

   See Attachment C

Part II. Description of Curriculum Change

1. New Syllabus of Record

See Attachment A.

2. Summary of the proposed revision

The principal revision is to change the programming language used in the course from FORTRAN to C++. The techniques and concepts taught, as well as the overall approach for the course, remains the same; only the language changes. There have been adjustments in the syllabus terminology to reflect the new language and adjustments in language processing to reflect the fact that C++ is to be taught on microcomputers rather than on the mainframe.

3. Justification for the revision

FORTRAN is a programming language that is not being used as widely as in years past. It is used only in specialized situations - heavy mathematical applications and some parallel computations. As FORTRAN's use has been fading, the use of C (and C++, a superset of C), has been on the up-swing. C and C++ have become a worldwide choice for program development of nearly all types, even mathematical applications.

Object-oriented programming has for several years been widely touted as a very important programming and design approach. To keep up with the changing field of computer science, we must find methods of including the object-oriented approach in our degree programs. The programming language C++ was developed for use in object-oriented programming.

By changing the language used in CO 110, we can catch up with a major programming trend in the computer field and, at the same time, lay the ground work for incorporating the object-oriented approach into our curriculum. This approach will actually be taught in CO 310.

4. Old Syllabus of Record

See Attachment B.

5. Letters of Support

See Attachment C

I. Catalog Description

CO 110 Problem Solving and Structured Programming 3c-01-3sh

(For science, mathematics, and computer science majors, and for others who have a sufficiently quantitative orientation.) Basic structure of modern digital computers; problem analysis and computer solution using flowcharting and the C++ language. Exemption or credit by examination possible.

II. Course Objectives

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

A. Use an integrated programming environment.

B. Develop algorithms from user problem statements.

C. Express the solutions to computer oriented problems in flowcharts and/or pseudocode.

D. Give commands to compile, link, and run their own programs, including using common options.

E. Proficiently transform designs of problem solutions into C++ programming language.

F. Apply debugging and testing techniques to locate errors and determine the effectiveness of a program.

G. Recognize and use the correct C++ programming language syntax.

H. Apply structured programming techniques including design approaches, mnemonic naming, use of documentation, and avoidance of excessive branching.

I. Use these programming elements: variable declaration, use of data types and simple data structures (arrays and records), decision structures, loop structures, input and output for terminals and files, output form, and subordinate functions.

III. Course Outline

A. Introduction 2 hrs

1. History of computers
2. Components of a computer
3. Programming languages
4. Compilation vs interpretation

B. The Programming Environment 4 hrs

1. Editing
2. Compiling
3. Linking
4. Options
5. Debugging
6. Redirection of input and output

C. Algorithm development using flowcharts/psuedocode 4 hrs

1. Software engineering method
2. Classic problems - maximum, minimum, sum, average

D. Basic input and output 1 hr

E. Data types 4 hrs

1. Constants
2. Variables
3. Expressions
4. Library files and functions

F. Simple Data Structures 7 hrs

1. One dimensional arrays
2. Strings as arrays
3. Multi-dimensional arrays
4. Records
5. Classic problems - searching, sorting

G. Use of decision structures 3 hrs

1. Single alternative
2. Double alternative
3. Multiple alternative
4. Nested structures

H. Loops 3 hrs

1. While loop
2. Do-while loop
3. For loop
4. Counting loop
5. Priming read loop

I. Programming language form 2 hrs

1. Syntax
2. Structured code
3. Documentation
4. Case sensitivity

J. Advanced formatted I/O including file access 4 hrs

1. Formatted input and output
2. File I/O
3. Sequential file processing

K. Testing and Debugging Techniques 1 hr

L. Functions 3 hrs

1. Argument passing
2. Returning results
3. Recursion
4. Testing a program system

M. Introduction to the Object Oriented Approach 1 hr

IV. Evaluation Methods

The final grade for the course is determined as follows:

50-60% Examinations. Three mid-term exams and the final - each consisting primarily of multiple choice, true-false, and short answer questions.

30-35%	Programming assignments. There are approximately six programming assignments worth varying numbers of points that collectively count this portion of grade. Suggested tasks for the assignments include: linear search, sorting, 2-D array processing, interactive programming, simulation, sequential file processing, modularization.
10-15%	Class participation and quizzes. This may be based on written questions, verbal discussions, computer lab sessions, or other form of interaction.

Suggested Grading Scale:	90-100%	A
	80-89%	B
	70-79%	C
	60-69%	D
	0-59%	F

V. Required textbook, supplemental books and readings

Adams, Joel, Leestma, Sanford, and Nyhoff, Larry. C++: An Introduction to Computing. Prentice Hall. 1995.

VI. Special resource requirements

None.

VII. Bibliography

Staugaard, Andrew C., Jr. Structuring Techniques: An Introduction Using Turbo C++.

Perry, Greg. Turbo C++ Programming 101. Sams Publishing. 1993.

Friedman, Frank, and Koffman, Elliot. Problem Solving, Abstraction, and Design Using C++. Addison-Wesley. 1994.

Schildt, Herbert. C++ from the group up. McGraw-Hill. 1994.

Savitch, Walter. Problem Solving with C++. Addison-Wesley. 1996.

I. Catalog Description

CO 110 Problem Solving and Structured Programming 3c-01-3sh

(For science, mathematics, and computer science majors, and for others who have a sufficiently quantitative orientation.) Basic structure of modern digital computers; batch processing vs. interactive time-shared online computing; problem analysis and computer solution using flowcharting and the FORTRAN language. Exemption or credit by examination possible.

II. Course Objectives

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

A. Use the computing environment at IUP, including appropriate command language and editor(s).

B. Develop algorithms from user problem statements.

C. Express the solutions to computer oriented problems in flowcharts and/or pseudocode.

D. Give commands to compile, link, and run their own programs, including using common options.

E. Proficiently transform designs of problem solutions into programming language.

F. Apply debugging and testing techniques to locate errors and determine the effectiveness of a program.

G. Recognize and use the correct programming language syntax.

H. Apply structured programming techniques including design approaches, use of documentation, and avoidance of excessive branching.

I. Use these programming elements: variable declaration, use of data types (including scalar and arrays), decision structures, loop structures, input and output for terminals and files, output form, and subprograms.

III. Course Outline

A.	Introduction	2 hrs

1. History of computers
2. Components of a computer
3. Programming languages
4. Compilation vs interpretation

B.	The Computing Environment	6 hrs

1. Logging in, facilities
2. Editing
3. Command Language
4. Compilation and linking, with options
5. Command file and batch printouts
6. Introduction to the debugger

C.	Algorithm development using flowcharts/psuedocode	4 hrs

1. Software engineering method
2. Classic problems - maximum, minimum, sum, average

D.	Basic input and output	1 hr
E.	Data types	3 hrs

1. Constants
2. Variables
3. Expressions
4. Library functions

F.	Arrays	6 hrs

1. One dimensional
2. Array I/O
3. Multi-dimensional
4. Classic problems - searching, sorting

G.	Use of decision structures	3 hrs

1. Single alternative
2. Double alternative

3. Multiple alternative
4. Nested structures

H.	Loops	3 hrs

1. While loop
2. Repeat-until loop
3. Counting loop
4. Priming read loop

I.	Programming language form	2 hrs

1. Syntax
2. Structured code
3. Documentation

J.	Advanced formatted I/O including file access	5 hrs

1. Formatted input and output
2. File I/O
3. Sequential file processing

K.	Testing and Debugging	1 hr
L.	Subprograms	3 hrs

1. Functions
2. Subroutines
3. Argument passing
4. Testing a program system

IV. Evaluation Methods

The final grade for the course is determined as follows:

50-60%	Examinations. Three mid-term exams and the final consisting primarily of multiple choice, true-false, and short answer questions. (450 points)
30-35%	Programming assignments. There are approximately six programming assignments worth varying numbers of points that collectively count this portion of grade. Suggested tasks for the

assignments include: linear search, sorting, 2-D array processing, interactive programming, simulation, sequential file processing, modularization. (180 points)

10-15% Class participation and quizzes. This may be based on written questions, verbal discussions, computer lab sessions, or other form of interaction. (80 points)

V. Required textbook, supplemental books and readings

Koffman, Elliot. B., and Friedman, Frank L., FORTRAN with Engineering Applications, Fifth Edition, Addison-Wesley Publishing Co., 1993.

VI. Special resource requirements

None.

VII. Bibliography

Barnard, D.T. and D.B. Skillicorn, Effective FORTRAN 77 for Engineers and Scientists, Wm C. Brown Publisher, 1992.

Bezner, H.C., FORTRAN 77, Prentice-Hall, 1989.

Ellis, T.M.R, et. al, FORTRAN 90 Programming, Addison-Wesley, 1994.

Etter, D.M., Structured FORTRAN 77 for Engineers and Scientists, Third Edition, Benjamin Cummings, 1990.

Herbst, R.T., Software Design Using FORTRAN 77, Prentice-Hall, 1990.

Ortega, J.M., An Introduction to FORTRAN 90 for Scientific Computing, Saunders College Publishing, 1994.

Starkey, J.D. and R.J. Ross, Fundamental Programming with FORTRAN 77, West Publishing, 1987.

Worland, P.B., Modern FORTRAN 77 for Scientists and Engineers, HBJ, 1989.

To: James Wolfe, Chairman
Computer Science Department Curriculum Committee

From: Gerald Buriok, Chairman
Mathematics Department [illegible]

Date: September 11, 1995

Subject: Computer Science Curriculum Changes

In your proposal of March 29, 1995, you outlined five changes the Computer Science Department wishes to implement in your curriculum. Since students majoring in our Mathematics and Applied Mathematics programs are required to take CO 110 Problem Solving and Structured Programming, and students in Applied Mathematics are required to take CO 250 Introduction to Numerical Methods, changes in these courses will directly affect our students. Other changes in your proposal may affect students in the Mathematics Department who are seeking a minor in Computer Science.

The faculty of the Mathematics Department support the five changes outlined in your proposal, and in particular, the changes you are recommending for CO 110 and CO 250. Using the computer programming language C++ in CO 110 while maintaining FORTRAN as the language of CO 250 means our students will be exposed to two programming languages instead of one, as is currently the case. This would appear to be an advantage to them. Likewise, since C++ will be used on personal computers while FORTRAN will still be taught on the VAX system, our students will have broader exposure to programming on different platforms. One potential drawback to this, however, is that students who have completed CO 220 prior to enrolling in CO 250 will have VAX experience, while those without CO 220 (such as Applied Mathematics majors) who have completed only CO 110 will have no prior VAX experience. It is my understanding that Computer Science faculty who teach CO 250 will address this concern.

```
From:    GROVE::WHITSON          7-SEP-1995 15:10:45.50
To:      JIM_WOLFE
CC:      GLBUTER
Subj:    Dual level courses in Computer s
```

Jim:
The following is a memo that I am also sending as a hard copy to you:

Date: Thursday, September 7, 1995

To: Jim Wolfe,
Chair Computer Science Dept. Curriculum Committee

From: Dennis Whitson,
Chair, Physics Department

Re: Use of the C language in CO 110 and CO 310

This issue was brought up at a Physics Department meeting on September 5, 1995. The Physics Faculty were in favor of switching to the use of the C language in CO 110 if the non-object approach was used. It was felt that the Physics Student needs to be introduced to the general subject of programming in a way that emphasizes algorithm development and fosters capabilities comparable to the current course which uses FORTRAN. You should leave object oriented programming to CO 310, though introducing it in the last few weeks of CO 110 would not be objectionable.

The use of FORTRAN in CO 250, while using C in CO 110 would inevietably degrade CO 250 to some degree, but I think that the experience of having two computer languages would be of benefit to our students. This issue was not taken up at the above meeting, but I don't believe that there would be any significant objection to using either FORTRAN or C in CO 250.

The other issues in your projected curriculum changes are not of any real

Date: Thursday, September 7, 1995

To: Jim Wolfe,
Chair Computer Science Dept. Curriculum Committee

From: Dennis Whitson,
Chair, Physics Department

Re: Use of the C language in CO 110 and CO 310

This issue was brought up at a Physics Department meeting on September 5, 1995. The Physics Faculty were in favor of switching to the use of the C language in CO 110 if the non-object approach was used. It was felt that the Physics Student needs to be introduced to the general subject of programming in a way that emphasizes algorithm development and fosters capabilities comparable to the current course which uses FORTRAN. You should leave object oriented programming to CO 310, though introducing it in the last few weeks of CO 110 would not be objectionable.

The use of FORTRAN in CO 250, while using C in CO 110 would inevitably degrade CO 250 to some degree, but I think that the experience of having two computer languages would be of benefit to our students. This issue was not taken up at the above meeting, but I don't believe that there would be any significant objection to using either FORTRAN or C in CO 250.

The other issues in your projected curriculum changes are not of any real concern to the Physics Department.