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

UWUCC USE Only
Number: 01-12c ~~00-55c~~
Submission Date: _____
Action-Date: App 4/10/01
Senate App 12/4/01

CURRICULUM PROPOSAL COVER SHEET
University-Wide Undergraduate Curriculum Committee

I. CONTACT

Contact Person Sanwar Ali Phone 357-7994

Department Computer Science

II. PROPOSAL TYPE (Check All Appropriate Lines)

_____ **COURSE** Data Struct & Algor
Suggested 20 character title

___ New Course* _____
Course Number and Full Title

_X_ Course Revision **COSC 310 Data Structure**
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

_X_ Number and/or Title Change **COSC 310 Data Structures**
Old Number and/or Full Old Title

COSC 310 Data Structures and Algorithms
New Number and/or Full New Title

Rev.

___ Course or Catalog Description Change _____
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)

Tim Watts
Department Curriculum Committee

G. L. Buttenbaum
Department Chair

02/23/01
College Curriculum Committee

John 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 principal revision of course description and syllabus from COSC 310 Data Structures to COSC 310 *Data Structures and Algorithms* are contained in Attachment A, the new syllabus of record.

3. Justification of the revision

Since the department does not offer a separate course on Algorithm Analysis and Design, it is now necessary to revise and change the course contents of COSC 310 and rename it as *Data Structures and Algorithms*. Concepts of algorithms are essential for our students, who want to be software developers or programmers. Many colleges and universities, such as Penn Sate, University of Pittsburgh, James Madison University, VA, University of Illinois at Springfield, and University of Texas at Arlington are offering this course with proposed new name and course syllabus.

In the past, unlike in other universities, our department did not have Object-Oriented C++ course. Without the knowledge of Objected-Oriented Programming (OOP) approach students had very difficult time grasping the concepts and writing projects in COSC 310 Data Structures course. This forced our department to design another new course COSC 210 Object-Oriented and GUI Programming. Now students are required first to take COSC 110 Programming and Problem Solving in C++, then COSC 215 Object-Oriented and GUI Programming, and finally COSC 310 Data Structures and Algorithms. **[COSC 110 → COSC 210 → COSC 310.]** Hence the pre-requisite of COSC 310 must be changed to COSC 210.

4. Old Syllabus of Record

See Attachment B the old syllabus of record.

5. Letters of Support

N/A

NEW COSC 310 Syllabus of Record

Attachment A
3 class hours
0 lab hours
3 semester hours
3c-0l-3sh

I. Catalog Description

COSC 310 Data Structures and Algorithms

Prerequisite: COSC 210

Fundamental concepts of data design and implementation, data abstraction, data structures, arrays, linked-lists, stacks, queues, recursion, trees, graphs, and hashing. The course will also cover sorting algorithms, divide and conquer techniques, greedy methods, and analysis of algorithms. The object-oriented paradigm will be employed in this course using an object-oriented language.

II. Course Objectives:

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

a. Use abstract data structures, including stacks, queues, trees, graphs, hash tables, and linked-lists, that are appropriate for a given algorithm.
b. Implement abstract data structures, objects and algorithms using an object-oriented approach and determine the effect of the implementation on the performance of the algorithm.
c. Analyze the time and space efficiency of several significant classes of algorithms, sorting
d. Use the object-oriented approach to program design and recognize its capabilities relative to the procedural paradigm.

III. Detailed Course Outline:

1. Linear Data Structures (12 hours)
   a. Abstract Data Type
   b. array storage mapping functions
   c. linked lists - singly and doubly linked lists
   d. dynamic storage allocation – pointers
   e. stacks and queues – array-based and pointer-based implementations
   f. circular lists and multi-lists

2. Recursion/Sorting/Algorithms Analysis (12 hours)
   a. recursion as a programming technique
   b. divide and conquer technique
   c. advanced sorting techniques- Mergesort and Quicksort
   d. algorithm efficiency analysis (big-O notation)
   e. Hashing- Hash functions, resolving collisions, efficiency of Hashing

3. Trees (9 hours)
   a. binary trees
   b. ADT binary tree, array and pointer implementations of binary trees
   c. Binary tree traversals- preorder, postorder, and inorder traversals
   d. special applications of binary trees (binary search trees, heaps), parse tree
   e. other trees (AVL, b-trees, 2-3-trees)
   f. heap sort
   g. tree algorithm efficiency analysis

4. Graphs (6 hours)
   a. undirected and directed graphs, adjacency matrix and adjacency list
   b. graph traversals- depth-first and breadth-first search
   c. greedy method- single-source shortest path and all-pairs shortest paths
   d. Applications of graphs- spanning trees, minimum spanning trees (Prim's and Kruskal's algorithms)
   e. graph algorithm efficiency analysis

5. Two Class Tests (3 hours)

Total hours in semester = **(42 hours)**

IV. Evaluation Method:

Evaluation: The final grade of the course will be determined as follows:

Two Class Tests	30-40%
Final Comprehensive Exam	20-30%
Projects, Assignments & Participation	40-50%

Grading Scale: The grading scale will be:
90-100% = A, 80-89% = B, 70-79% = C, 60-69% = D, and < 60% = F.

Attendance policy: The attendance policy will conform to the University wide attendance criteria.

Sample Projects: (All projects will use an Object-Oriented Approach.)

Project #1 Create a class definition of a linked-list using array-based or pointer-based implementation and then insert, retrieve, and delete items from it.

Project #2 Convert an infix arithmetic expression into a postfix form and evaluate it using an array implementation of a stack.

Project #3 Develop a program for preorder, postorder, and inorder traversals of a binary search tree.

Project #4 Develop a program for graph traversal using either depth-first or breadth-first search.

Project #5 Create a program to find minimum spanning tree (MST) either using Kruskal's or Prim's algorithm.

Project #6 Create a program for either mergesort or quicksort.

V. Required Textbook, Supplemental Books and Readings:

Sahni, Sartaj *Data Structures, Algorithms, and Applications in C++*, McGraw-Hill, 1998, ISBN #0-07-109219-6

VII. Bibliography:

Baase, Sara and Gelder, Allen V. *Computer Algorithms* (3rd Ed.), Addison-Wesley, 2000, ISBN #0-201-61244-5

Budd, Timothy *Data Structures in C++*, Addison-Wesley, 1998, ISBN #0-201-31659-5

Carrano, Helman, and Veroff *Data Abstraction and Problem Solving with C++* (2nd Ed.), Addison-Wesley, 1998, ISBN #0-201-87402-4

Dale, Nell C++ Plus Data Structures, Jones and Bartlett Publishers, 1999, ISBN #0-7637-0621-3

Ford, William and Topp, William Data Structures with C++, Prentice Hall, 1996, ISBN #0-02-420971-6

Heileman, Gregory L. *Data Structures, Algorithms, and Object-Oriented Programming*, McGraw-Hill, 1996, ISBN #0-07-027893-8

Langsam Y., Augenstein M. J., and Tenenbaum A.M. *Data Structures using C and C++* (2nd Ed.), Prentice Hall, 1996, ISBN #0-13-036997-7

Preiss, Bruno R. Data Structures and Algorithms, John Wiley & Sons, 1999, ISBN #0-471-24134-2

Weiss, Mark A. *Data Structures & Algorithm Analysis in C++* (2nd Ed.), Addison-Wesley, 1999, ISBN #0-201-36122-1

Weiss, Mark A. *Data Structures and Problem Solving Using C++* (2nd Ed.), Addison-Wesley, 2000, ISBN #0-201-61250-X

OLD COSC 310 Syllabus of Record Attachment B

I. Catalog Description

COSC 310 Data Structures 3c-0l-3sh

Pre-requisite: COSC110
Basic concepts of data; storage systems and structures; lists, arrays, strings, hashing techniques, searching and sorting techniques; data structures in programming languages; string processing. Programming in an object-oriented language.

II. Course Objectives

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

1. Analyze the time and space efficiency of several significant classes of algorithms.
2. Use abstract data structures, including stacks, queues, trees, graphs, hash tables, and linked lists, that are appropriate for a given algorithm.
3. Implement abstract data structures, objects and algorithms using an object-oriented approach and determine the effect of the implementation on the performance of the algorithm.
4. Use the object-oriented approach to program design and recognize its capabilities relative to the procedural paradigm.

III. Course Outline

1. Introduction to abstract data types and objects (6 hours)
   a. abstract data types
   b. encapsulation
   c. objects, classes, and inheritance
   d. polymorphism
   e. public and private attributes
   f. passing arguments
2. Linear Data Structures (12 hours)
   a. strings
   b. array storage mapping functions
   c. dynamic storage allocation - pointers
   d. linked lists - singly and doubly linked lists
   e. stacks and queues - as objects

f. circular lists and multi-lists

g. implementations of linear objects using several actual data structures

3. Sorting (6 hours)

a. elementary sorting techniques (exchange, selection, and insertion)

b. recursion as a programming technique

c. advanced sorting techniques (merge and quick)

d. algorithm efficiency analysis (big O notation)

4. Hierarchical Data Structures (9 hours)

a. general trees

b. binary trees

c. array and pointer implementations of binary trees

d. preorder, postorder, inorder traversals

e. special applications of binary trees (search trees, heaps)

f. heap sort

g. other trees (AVL, b-trees, 2-3-trees)

5. Graphs (3 hours)

6. Hash tables (3 hours)

IV. Evaluation Methods

		Suggested Grading Scale:	
Exam 1	100 points		
Exam 2	100 points	90-100%	A
Final	100 points	80-89%	B
Projects	300 points	70-79%	C
Quizzes	~50 points	60-69%	D
Homework	~50 points	0-59%	F

V. Text

Frank M. Carrano, Data Structures and Problem Solving with C++, Walls and Mirrors, Benjamin Cummings Publishing Co. (1995)

VI. Special Resource Requirements

None - the needed resources are already in place.

VII. Bibliography

Joseph Bergin, Data Abstraction, McGraw-Hill, 1994.

William Ford and William Topp, Data Structures with C++, Prentice Hall, 1995.

James F. Korsh and Leonard J. Garrett, Data Structures, Algorithms, and Programming Style using C, PWS-Kent, 1988.

Johnsonbaugh, Richard, and Kalin, Martin. Object-Oriented Programming in C++. Prentice Hall. 1995.

Sengupta, Saumyendra, and Korobkin, Carl. C++: Object-Oriented Data Structures. Springer-Verlag. 1994.