**William Paterson University of New Jersey**
**Department of Computer Science**
**College of Science and Health**
**Course Outline**

1. TITLE OF COURSE AND COURSE NUMBER: Data Structures, CS342, Credits:4, (Major core course)

2. DESCRIPTION OF THE COURSE: Concepts, implementations, and applications of lists, stacks, queues, trees, graphs, sorting and searching algorithms, hashing, memory management and advanced data structures using object-oriented programming.

3. COURSE PREREQUISITES: CS240 and CS260 with grades of C- or better

4. COURSE OBJECTIVES:

   a. To learn the classic data structures that are found in most computer algorithms.
   b. To develop analytical techniques with which to evaluate data structures.
   c. To apply object-oriented and software design practices.
   d. To become familiar with the pragmatic of implementation of data structures and the related algorithms.

5. STUDENT LEARNING OUTCOMES:
   Upon completion of this course, students will be able to:
   - Explain and apply the fundamental data structures used in programming in terms of their definitions, examples, trade-offs, optimal application of them in different programming situations, their implementation, and other aspects.
   - Program, debug, and maintain source-code for data structures and incorporate them into larger programming frame-works. Foremost, students will be able to efficiently implement fundamental operations associated with each data structure, such as initialization, insertion, deletion, traversal, search, and maintenance (e.g. balancing).
   - Critically analyze programs to determine where and which particular data-structures would be most beneficial. This capacity to apply data-structures methodologies is necessary in several directions of further Computer Science work such as Data Communications, Operating Systems, and Computer Graphics.
   - Have a reinforced level of experience in Object-Oriented-Programming which they first studied in CS240. Students should be capable of representing data structures using classes and templates as well as with structs/typedef.
   - Appreciate the inter-related issues of complexity and program design which will be further developed in CS372. Students should demonstrate the ability to analyze algorithms, determine complexity classes, and make critical decisions based on that.
   - Through classroom participation in problem-solving sessions and discussions, homework, papers, and other assignments, this course also reinforces the following student learning outcomes in accordance with the university:

a) Effectively express themselves in written and oral form. Measure: exams, homework and projects.
b) Demonstrate the ability to think critically. Measure: exams, homework and projects.
c) Locate and use information. Measure: projects.
d) Demonstrate the ability to integrate knowledge and ideas in a coherent and meaningful manner. Measure: exams, surveys, and projects.

6. SUGGESTED TOPICAL OUTLINE OF THE COURSE CONTENT:
    1. Review
        a. Object-oriented programming
        b. Abstract data type (ADT), focusing on classes and templates
        c. Complexity of algorithms
        d. Recursion
        e. Storage allocation for structured objects
        f. Strings, Pointers, and File access
    2. Lists
        a. Simple lists: Array versus linked list
        b. Ordered lists
        c. Self-organized lists
        d. Double ended lists
        e. Array versus linked list implementations
    3. Stacks and queues
        a. Definition of stacks and queues
        b. Implementations of stacks and queues
        c. Stack Application: Expression evaluation and recursion
        d. Application: Examples of queuing simulations for systems/networks
    4. Trees
        a. Binary trees and implementations
        b. Binary search trees and expression trees
        c. Binary tree traversals
        d. Applications: operator precedence, expression evaluation/parsing, and Huffman coding
        e. Balancing binary trees
        f. General trees and forests
        g. B-trees and other models for fast access or balancing
        h. Applications
    5. Searching and Sorting
        a. Linear, binary, and interpolation search
        b. Binary search trees, tries, and other models
        c. Quadratic Sorts: selection sort, insertion sort, bubble/exchange sort
        d. N*log(N) Sorts: merge sort, quick sort, bucket and radix sorts
        e. Applications
    6. Graphs and digraphs
        a. Representations: Adjacency matrix, array of edges, adjacency list, multi-linked records, weighted models

b. Traversal with backtracking/tagging and search: concepts of depth-first search, breadth-first search, and others
c. Concepts of graph operations and algorithms: connectivity testing, shortest path, coloring, Hamiltonian/Eulerian, minimal spanning tree, maximal flow
d. Applications

7. Hash Tables concepts
   a. Collision resolution (the three methods)
   b. Asymptotic analysis of hash table operations
   c. Hash functions
   d. Applications
   e. Other hashing models (such as external and dynamic hashing)
8. Priority queues concepts
   a. Heaps (array and linked tree representations)
   b. Heap sort
   c. Skew heaps
   d. Applications
9. Sets
   a. Set operations
   b. Bit vector sets
   c. Building sets from hash tables

7. GUIDELINES/SUGGESTIONS FOR TEACHING METHODS AND STUDENT LEARNING ACTIVITIES:

   a. Classroom lectures, discussions, and problem-solving session.
   b. Homework and pre-exam reviews.

8. GUIDELINES/SUGGESTIONS FOR METHODS OF STUDENT ASSESSMENT (OUTCOMES):
   a. Periodic examinations, quizzes, and final examination.
   b. Programming projects
   c. Homework assignments.

9. SUGGESTED READINGS, TEXTS, OBJECTS OF STUDY:

   ADTs, Data Structures, and Problem Solving with C++, 2nd edition, Larry R. Nyhoff, Prentice Hall/Pearson, 2005

10. BIBLIOGRAPHY OF SUPPORTIVE TEXTS AND OTHER MATERIALS:

Data Abstraction & Problem Solving with C++, 5th edition, F. M. Carrano, Addison-Wesley, 2007

C++: Classes and Data Structures, J. Childs, Prentice Hall/Pearson, 2007

C++ Plus Data Structures, 4th edition, N. Dale, Jones & Bartlett, 2006

Data Structures and Algorithms in C++, 3rd edition, A. Drozdek, Course Technology, 2004

Data Structures with C++ Using STL, : 2nd edition, W. Ford and W. Topp, Prentice Hall, 2002

Fundamentals of Data Structures in C++, E. Horowitz, S. Sahni, and D. Mehta, Silicon Press, 2006

Data Structures and other Objects Using C++, 3rd edition, M. Main and W. Savitch, Addison-Wesley, 2004

Data Structures Using C++, 2nd edition, D.S.Malik, Course Technology/Cengage, 2009

Handbook of Data Structures and Applications, D. P. Mehta and S. Sahni, Chapman & Hall/CRC., 2004 (reference)

An Introduction to Data Structures and Algorithms, J.A. Storer and John C. Cherniavsky, Birkhäuser Boston, 2001 (reference)

Data Structures and Algorithm Analysis in C++, 3rd edition, M. A. Weiss, Addison Wesley/Pearson, 2006

Objects, Abstraction, Data Structures and Design: Using C++, E. Koffman and P. Wolfgang, Wiley, 2005

11. PREPARER'S NAME AND DATE: N/A

12. ORIGINAL DEPARTMENTAL APPROVAL DATE: Spring 1996

13. REVISERS' NAME AND DATE: Dr. John Najarian

14. DEPARTMENTAL REVISION APPROVAL DATE: Fall 2009