**CMPS 223 Data Structures and Algorithms**

**Catalog Description**

**CMPS 223 Data Structures and Algorithms (5)**

Builds on the foundation provided by CMPS 221 to introduce the fundamental concepts of data structures and algorithms that proceed from them within the framework of object-oriented programming technology. Topics include: recursion, fundamental data structures (including lists, stacks, queues, hash tables, trees and graphs) and basics of algorithmic analysis. Necessary components of object-oriented programming method will be introduced. Each week lecture meets for 200 minutes and lab meets for 150 minutes. Prerequisite: CMPS 221 with C- or higher.

**Prerequisites by Topic**

Fundamentals of any programming language including selective and repetitive constructors, concepts of subprograms (functions), and user-defined type, structure, or record.

**Units and Contact Time**

5 quarter units. 4 units lecture (200 minutes), 1 unit lab (150 minutes).

**Type**

Required for CS

**Required Textbook**

ADTs, Data Structures, and Problem Solving with C++, Second Edition, Larry Nyhoff. ISBN 0-13-14909-3

**Recommended Textbook and Other Supplemental Materials**

None

**Coordinator(s)**

Huaqing Wang

**Student Learning Outcomes**

ACM/IEEE Body of Knowledge Topics:

·        [AL1] *Object-oriented design*: problems are analyzed via Object-Based Design paradigms

·        [AL2] *Basic algorithm design*: Searching, Sorting algorithms are studied in depth

·        [AL1,2] *Algorithms and problem-solving*: Classic techniques for algorithm design; problem-solving in the object-oriented paradigm; application of algorithm design techniques to a medium-sized project

·        [AL3]*Basic algorithmic analysis*: Asymptotic analysis of upper and average complexity bounds; identifying differences among best, average,

and worst case behaviors; big "O" notation; standard complexity classes; empirical measurements of performance; time and space tradeoffs in

algorithms

·        [PF4] *Recursion*: The concept of recursion; recursive mathematical functions; simple recursive procedures; divide-and-conquer strategies; recursive backtracking; implementation of recursion; recursion on trees and graphs

·        [AL3]*Fundamental computing algorithms*: Searching algorithms, Sorting algorithms in contiguous array, link and binary tree search and tree sorting.

·        [PF3]*Fundamental data structures*: Pointers and references; linked list structures; implementation strategies for stacks, queues, and hash table

implementation strategies for trees.

**ABET Outcome Coverage**

a. An ability to apply knowledge of computing and mathematics appropriate to the discipline.

3b. An ability to analyze a problem, and identify and deﬁne the computing requirements and

speciﬁcations appropriate to its solution.

3c. An ability to design, implement and evaluate a computer-based system, process, component,

or program to meet desired needs.

3j. An ability to apply mathematical foundations, algorithmic principles, and computer science

theory in the modeling and design of computer-based systems in a way that demonstrates

comprehension of the tradeoﬀs involved in design choices.

**Lecture Topics and Rough Schedule**

Chapter 1 - 5 Brief Review of some contents.

Chapter 6 Lists: ADT, Static and Dynamic Array-based, linked list.

Chapter 7 Stacks: Array and linked implementation

Chapter 8 Queues

Chapter 9 ADT Implementations: Templates and Standard Containers

Chapter 10 ADT Implementation: Recursion, Algorithm Analysis and Standard Algorithms

Chapter 11 More Linking up with Linked Lists (variants of linked structures)

Chapter 12 Searching: Binary Trees and Hash Tables

Chapter 13 Sorting Chapter 14 OOP and ADTs

Chapter 15 Trees, Chapter 16 Graphs and Digraphs

Week 1: Review and enhance on user/programmer-defined types, abstract data type (ADT), pointers and dynamic

memory allocation and de-allocation, C-structures, classes in object-oriented programming languages,

constructors and destructor. Basic concepts are introduced in CMPS 221 and some are covered in CMPS

222.

Week 2 Chapter 6 Lists: ADT, Static and Dynamic Array-based, linked list. Students may not take CMPS 222

and transferred students my use different language than C++. List implementations will be done in C-

structures, in C++ classes. The implementations will use static and dynamic array-based, singly and  
doubly linked structures.

Week 3 Chapter 7 Stacks: Array and linked implementation, Implement stacks in arrays and linked structures.

Usage of stacks in recursive calls.

Week 4 Chapter 8 Queues – Queue implementation in Arrays and dynamic structures. Variations of singly-

linked queue. FIFO queue and sorted queues.

Week 5 Chapter 9 ADT Implementations: Templates and Standard Containers, Chapter 10 ADT Implementation:

Recursion, Algorithm Analysis and Standard Algorithms, Chapter 14 OOP and ADTs, Chapter 9 is

discussed and is extended to cover generic types, using class templates and usages.

Inheritances and polymorphism will be used to re-design and re-implement the lists, stacks and queues.

Week 6 Chapter 6, Chapter 10 ADT Implementation: Recursion, Algorithm Analysis and Standard Algorithms

Recursions, Algorithm analysis, time and storage complexities, Big-O notation will be introduced and

used in problem analysis.

Week 7 Chapter 12 Searching: Binary Trees and Hash Tables, Analysis of linear and binary search,

implementation and analysis of hash table will be introduced in this week.

Week 8 Chapter 13 Sorting, Chapter 14 OOP and ADTs. Various sorting algorithms such as insertion sort ,

quick sort, heap sort and priority-queue will be

Introduced, implemented and analyzed. The generic (template) functions will be introduced and used to

Implement those sorting algorithms.

Week 9 Chapter 15 Trees. The concepts and implementations of Binary trees, binary search tresses AVL trees

will be introduced, and their insertions, deletions and searching time complexities will be analyzed in

this week.

Week 10 Chapter 16 Graphs and Digraphs. Concepts and implementation methods of graphs will be introduced.

Different searching and traversal algorithms will be introduced and analyzed also.

**Design Content Description**

Not applicable to this course.

**Prepared By**

Huaqing Wang on [date]

**Approval**

Approved by CEE/CS Department on [date]   
Effective [term]