Bergen Community College  
Computer Science Department  
Course Syllabus

Instructor: ___________________________  Phone: __________________

Email: ___________________________  Office hours_________________________

Course Title: CIS-277 Data Structures and Algorithms

Prerequisites: CIS-265 C++ Programming II

Credits/Hours: 3 Credits  3 Lecture / 1 Lab

Gen’l Ed. Course: No

Course Description:

Data Structures and Algorithms is a study of the representation and implementation of abstract data types and related algorithms that are used in computer science. Topics considered include lists, strings, stacks, queues, trees, graphs, networks, file structures, recursive functions, sorting techniques, hashing, and the analysis of algorithms.

Student Learning Outcomes: Upon completion of the course, the student will:

1. Understand the concept of an abstract data type.
2. Be able to select the appropriate data structure and design the corresponding operations to implement an abstract data type.
3. Know the fundamental order of magnitude growth rates and how they are used to measure the run-time efficiency of an algorithm.
4. Know the principles of pointers, dynamic memory management, and be able to apply these concepts in constructing dynamic data structures.
5. Know the fundamental properties of stacks and queues and be able to implement them using dynamic linked-lists.
6. Will be able to incorporate recursive techniques in the representation and implementation of an abstract data type.
7. Know the fundamental properties of binary trees, binary search trees, and general trees and be able to implement them using dynamic data structures.
8. Be able to represent graphs and networks using adjacency matrices and adjacency lists, and implement them using the appropriate data structure.
9. Know the major sorting algorithms and be able to identify the advantages and disadvantages of each.
10. Understand the use of hashing techniques in the storage and retrieval of data.

Student Learning Outcomes Assessment Measurement:
Each of the above listed student learning outcomes will be assessed by: (1) written assignments and/or quizzes; (2) written examinations and a comprehensive file exam.
Course Grade: see the grading policy for the course.


Course Content:

1. Abstract Data Types
   - Abstract data types (ADTs) in program design
   - Data structure selection criteria
   - ADT: vector

2. Analysis of Algorithms
   - Worst and average case behavior
   - Order of magnitude analysis of algorithms
   - Expected run-time of an algorithm

3. ADT: Stack
   - Terminology and fundamental operations
   - Infix, prefix, and postfix notation
   - Algorithm to convert infix form to postfix form
   - Parsing and evaluating arithmetic expressions
   - Array implementation of a stack

4. ADTs: Queue, Priority Queue, and Dequeue
   - Terminology and fundamental operations
   - Array implementation of a queue
   - Circular implementation of a queue
   - Priority queue: analysis and implementation

5. Data Structure: Dynamic Linked-List
   - Pointers and dynamic memory management
   - Linked-list terminology
   - Declaration form for dynamic linked-lists
   - Dynamic linked-list implementation of stacks and queues

6. ADT: List
   - Dynamic linked-list implementation of lists
   - Load list and dump list algorithms
   - Insertion and deletion of list elements
   - Other list processing algorithms
   - Array implementation versus dynamic linked-list implementation

7. Recursion
   - Recursive functions and algorithms
   - Iterative versus recursive implementations
8. **ADTs: Binary Tree, Binary Search Tree, and General Trees**
   - Tree terminology
   - Linked implementation of trees
   - Inserting and deleting elements in a binary search tree
   - Binary search tree performance analysis
   - Tree processing algorithms
   - General Trees

9. **ADTs: Graph and Network**
   - Graph terminology and graph traversals
   - Adjacency matrix representation
   - Adjacency list representation
   - Fundamental algorithms for graphs and networks

10. **Searching Techniques**
    - ADT: Dictionary
    - Hashing techniques and collision resolution
    - M-way search trees

11. **Sorting Techniques**
    - Criteria for selecting a sorting algorithm
    - Quick sort
    - ADT: heap and heap sort algorithm
    - Radix sort algorithm
    - Shell sort algorithm
    - External sorting – merge sort algorithm

Rev: 8/12