HOSTOS COMMUNITY COLLEGE
DEPARTMENT OF MATHEMATICS

Data Structures

Course Number:	CSC 300
Course Title:	Data Structures
Credit Hours:	3.0
Equated Hours:	3.0
Class Hours:	3.0

Pre Requisite: CSC 275 or CSC 215 Modern Programming & CSC 205 Discrete Mathematical Structures & MAT 220
Pre/Co-requisite: ENG 93/ESL 91/ ESL 93 or equivalent

Course Description: Abstract characterizations of data structures, such as arrays, stacks, queues, trees, and graphs, will be studied along with algorithms that make use of such structures, including algorithms for sorting, searching, and memory management. Implementation issues will be considered, and students will write programs that embody these structures and algorithms

RECOMMENDED TEXTBOOKS: C++ How to Program (10th Edition) By Paul Deitel & Harvey Deitel ISBN-13: 978-0-13-444823-7

Data Structures and Algorithms in C++ (4th edition) by Adam Drozdek ISBN-13: 978-1-133-60842-4 ISBN-10: 1-133-60842-6

Objectives: To provide experience to students in using these skills:
1. Analysis of algorithms,
2. Class design, in C++, based on performance requirements,
3. Understanding dynamic structures and their use in resource management, and
4. Correctly applying the fundamental searching and sorting algorithms.

Grade is based upon Programming Projects and Final Exam:
Students will complete 8-10 programming projects taken from the list of topics in the course outline. Specific topics to be covered in these projects may include: union-find algorithms; basic iterable data types (stack, queues, and bags); sorting algorithms (quicksort, mergesort, heapsort) and applications; priority queues; binary search trees; red-black trees; hash tables; and symbol-table applications.

Student Learning Objectives

1) Student will demonstrate fluency with formulating, and analyzing algorithms
2) Student will demonstrate proficiency with abstract data types
3) Student will demonstrate proficiency with sequential, sorted, iterated list structures
4) Students will demonstrate proficiency programming with stacks, queues and recursive structures
5) Students will demonstrate proficiency representing and applying search trees within programs
6) Students will demonstrate fluency with elementary concept of programming with Python

or other modern programming language

7) Students will demonstrate proficiency with strings, variables and arrays within modern programming language

Course Outline (by Module)

I: **Abstraction and Analysis** (½ week)

1.2 Functional Abstraction

1.3 Algorithm analysis

II: **Data Abstraction** (1 week)

2.2 Abstract Data Types

2.3 ADTS and Objects

2.4 An Examples ADT: Datasets

2.5 An Example ADT: Rational

III: **Container Classes** (1 week)

3.2 Lists

3.3 A Sequential Collection: A Deck of Cards

3.4 A Sorted Collection: Hand

IV: **Linked Structures and Iterations** (1½ weeks)

4.3 A linked Implementation of Lists

4.4 Linked Implementation of a List ADT

4.5 Iterators

4.7 Lists vs. Arrays

V: **Stacks and Queues** (1 week)

5.2 Stacks

5.3 Queues

5.4 Queue Implementation

5.5 An Examples Application: Queueing Simulations

VI: **Recursion** (1 week)

6.2 Recursive Definitions

6.3 Simple Recursive Examples

6.4 Analyzing Recursion

6.5 Sorting

6.6 A “Hard” Problem: The Tower of Hanoi

VII: **Trees** (1½ weeks)

7.2 Tree Terminology

7.3 An Example Application: Expression Trees

7.4 Tree Representations

7.5 An Application: A Binary Search Tree

VIII: **C++** (2 weeks)

8.2 C++ History and Background

8.3 Comment, Blocks o f Code, Identifiers, and Keywords

8.4 Data Types and variable declarations

8.5 Include Statements, Namespaces, and Input/Output

8.6 Compiling

8.7 Expressions and Operator Precedence

8.8 Decision Statements

8.9 Type Conversion

8.10 Looping Statements

8.11 Arrays

8.12 Function Details

8.13 Header Files and Inline Functions

8.14 Assert Statements and Testing

8.15 The Scope and Lifetime of Variables

IX: **C++ Classes** (½ week)

9.1 Basic Syntax and Semantics

9.2 Strings

9.3 File Input and Output

9.4 Operator Overloading

9.5 Class Variables and Methods

X: **C++ Dynamic Memory** (1 week)

10.2 C++ Pointer
10.3 Dynamic Arrays
10.4 Dynamic Memory Classes
10.5 Dynamic Memory Errors

XI: **C++ Linked Structures** (1 week)

11.2 A C++ Linked Structure Class
11.3 A C++ Linked List
11.4 C++ Linked Dynamic Memory Errors

XII: **C++ Templates** (½ week)

12.2 Template Functions
12.3 Template Classes

XIII: **Heaps, Balanced Trees, and Hash Tables** (1 week)

13.2 Priority Queues and Heaps
13.5 Hash Tables

XIV: **Algorithm Techniques** (½ week)

14.2 Divide and Conquer
14.3 Greedy Algorithm