CS 3350: DATA STRUCTURES

Semester Hours: 3.0
Contact Hours: 3

Coordinator: Venu Dasigi

Text: Data Abstraction and Problem Solving With C++, 6/E

Author(s): FRANK CARRANO & TIMOTHY HENRY

Year: 2013

SPECIFIC COURSE INFORMATION

Catalog Description:
Abstract data types including stacks, queues, lists, trees and graphs. Introduction to analysis of algorithms. Recursive searching and sorting algorithms. Adaptation and use of generic data structures and types. Functional concepts. Prerequisite: MATH 2220 or MATH 3220 and Grade of C or better in CS 2020.

Course type: REQUIRED

SPECIFIC COURSE GOALS

- I can solve computational problems using recursion.
- I can implement and apply stacks, queues, trees, and other custom data structures.
- I can create generic functions and classes.
- I understand algorithmic complexity (e.g. Big “O” notation).
- I can understand the relationship between data structures and algorithms.
- I can understand the design tradeoffs (e.g., code complexity and performance) in data structures and algorithms.

COMPUTER SCIENCE STUDENT OUTCOMES ADDRESSED BY THIS COURSE

- CS 1 Analyze a complex computing problem and to apply principles of computing and other relevant disciplines to identify solutions
- CS 2 Design, implement, and evaluate a computing-based solution to meet a given set of computing requirements in the context of the program’s discipline
- CS 3 Communicate effectively in a variety of professional contexts
- CS 6 Apply computer science theory and software development fundamentals to produce computing-based solutions

SOFTWARE ENGINEERING STUDENT OUTCOMES ADDRESSED BY THIS COURSE

- SE 1 An ability to identify, formulate, and solve complex engineering problems by applying principles of engineering, science, and mathematics
- SE 3 An ability to communicate effectively with a range of audiences

LIST OF TOPICS COVERED

- Review of CS2020 concepts with benchmark quiz (1 week = 7%)
  - Advanced programming concepts
  - Function Templates
  - Arrays (1D and 2D)
  - Linked Lists
  - Recursion (direct/linear)
- More recursion (emphasis on tail) (.5 week = 4%)
- Stacks & Queues (priority queue) (1 week = 7%)
- Introduction to sets and graphs (1 week = 7%)
- Intro to Analysis of Algorithms (1 week = 7%)
  - Calculating running time using primitive operations
  - Rates of Growth: O(n), Time Complexity, Space Complexity
- Trees (2.5 weeks = 17%)
  - Binary Trees, Binary Tree Manipulation
  - Binary Search Trees
  - Heaps and Priority Queues, Balanced Trees
- Hash Tables (1.5 weeks = 11%)
  - Hash Functions (load factor)
- Open Hashing, Closed Hashing
- Collision resolution (linear probing, chaining)
- Sorting algorithms (2 weeks = 14%)
  - Insertion Sort, Selection Sort, Merge Sort, Quicksort, Heapsort
- Advanced programming concepts (.5 week = 4%)
  - Smart pointers
  - Exception handling (throw/catch)
- Adaptation and use of generic data structures and types (1.5 weeks = 11%)
  - Class templates
  - Iterators
  - Vectors
  - Auto type inference
- Functional concepts (1.5 weeks = 11%)
  - Immutable data structures (map/filter/reduce)
  - Lambda expressions, anonymous functions, function parameters

**COMPUTER SECURITY TOPICS**

Faculty who recently offered CS 3350 have discussed and identified a list of topics related to computer security in this course. Below is a list for instructors to incorporate. (*) indicates topics that are mandatory.

<table>
<thead>
<tr>
<th>Security Topic</th>
<th>Description</th>
<th>Textbook Reference(^1)</th>
<th>Estimated Class Hours</th>
</tr>
</thead>
<tbody>
<tr>
<td>*Initialization before use</td>
<td>Review how C++ default initialization occurs – what is/isn’t initialized prior to use, why is that important?</td>
<td>Interlude 2</td>
<td>1</td>
</tr>
<tr>
<td>*Principle of least privilege</td>
<td>Making class data members private and allowing only certain ‘holes in the wall’ to access what the user needs and nothing more. Controlling what sub-types can access via private vs. protected in inherited classes.</td>
<td>Chapter 1 Interlude 1 Interlude 2</td>
<td>2</td>
</tr>
<tr>
<td>*Hash functions</td>
<td>Discussion of what a hash function is, several simple examples. The use of hash functions to implement a dictionary.</td>
<td>Chapter 18</td>
<td>&lt;1</td>
</tr>
</tbody>
</table>

\(^1\)Data Abstraction and Problem Solving With C++ by Carrano & Henry, 7th Edition.