CSC/CPE 103 Fundamentals of Computer Science III

1. CSC/CPE 103 Fundamentals of Computer Science III

2. credit units 4  contact hours 6

3. Course Coordinator: Hasmik Gharibyan

4. Textbook:(and/or other required material) Data Structures and Algorithm Analysis in Java by Mark Allen Weiss, third edition, Addison-Wesley, 2012

5. a. Course Description: Introduction to data structures and analysis of algorithms. Abstract data types. Specification and implementation of advanced data structures. Theoretical and empirical analysis and proofs of properties of recursive and iterative algorithms. Software performance evaluation and testing techniques. 3 lectures, 1 laboratory. Crosslisted as CPE/CSC 103.

   b. Prerequisite: CPE/CSC 102 with a grade of C- or better or CPE/CSC 108 with a grade of C- or better, CSC 141 with a grade of C- or better, and MATH 141 with a grade of C- or better, or consent of instructor.

   c. Required/Elective/Selective Elective for CPE, CSC, EE, SE

<table>
<thead>
<tr>
<th>Required</th>
<th>CSC</th>
<th>CPE</th>
<th>SE</th>
</tr>
</thead>
<tbody>
<tr>
<td>Elective</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Selective Elective</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

6. a. Course Goals/Outcomes
   The student will be able to:
   
   - Describe and define the Abstract Data Types (ADT) covered, including stacks, queues, priority queues, binary search trees, AVL trees, hash tables, and graphs.
   - Understand, explain, and track by hand alternate (iterative, as well as recursive) implementations of methods associated with each ADT.
   - Analyze for performance all algorithms associated with ADTs, reason about correctness and effectiveness of procedures, compare and contrast array and linked implementations of certain ADTs.
   - Implement and test each ADT implementation as a generic program.
   - Use ADTs in software applications.

   b. How Student Outcomes addressed
   (“B” = Basic level, “I” = Intermediate level, “A” = Advanced level)
7. **Major Topics Covered: (number of lecture hours each)**

- Collections, data structures, Abstract Data Types (ADT) (1)
- Introduction to algorithm analysis; working out the abstract running time of iterative, as well as recursive algorithms (2)
- Analysis of List ADT operations for alternate implementations (1)
- Advanced sorting algorithms: merge sort, quick sort; implementation and analysis (3)
- Iterators; supplying iterators for alternate implementations of List ADT (1)
- Stack ADT; alternate implementations and analysis (1)
- Queue ADT; alternate implementations and analysis (1)
- Priority Queue ADT; heap implementation and analysis (1)
- Binary Search Tree ADT; implementation and analysis (3)
- AVL Tree ADT; implementation and analysis (2)
- Binary tree traversal strategies: pre-order, in-order, post-order, and level order; supplying iterators for binary tree traversals (2)
- Hash Table ADT; alternate implementations and analysis (3)
- Graph ADT; alternate implementations and analysis (1)
- Graph algorithms (e.g. topological sort, shortest path, etc.); implementation and analysis (3)