Category	Outcome	Importance	Proficiency	BigO Expectations	Dissent
Abstract Analysis	Classify and rank common algorithms and data structure operations by asymptotic time complexity.	Essential	Competent		
Abstract Analysis	Understand the different cases for which complexity can be analyzed (best, worst, average) including basic amortized analysis (insertEnd in an array based list).	Essential	Competent		EOU only covers BigO
Abstract Analysis	Algebraically manipulate standard notations for complexity. (Recognize that n(3n + 2) => 3n^2 + 2n => O(n^2)).	Essential	Competent		
Abstract Analysis	Identify the asymptotic complexity of new algorithms involving standard operations on array, linked list, binary search tree, and hash table. Select appropriate tools based on time complexity	Essential	Proficient		
Abstract Analysis	Correctly identify the resources/operations to be counted in complexity analysis. e. g. Given a list of 10,000 numbers, but only 500 unique values, correctly reason about the efficency of building a set.	Essential	Competent		
Abstract Analysis	Describe the situations where asymptotic time complexity fails to capture important differences in algorithms. e.g. Small problem sizes and large constant factors (especially for algorithms in the same BigO category)	Essential	Competent		
Abstract Analysis	Discuss considerations other than time complexity that might be used to select an algorithm: space complexity, programming time, maintainability, etc...	Essential	Exposure		EOU saves for Algorithms
Searching & Sorting	Implement linear and binary searches	Essential	Proficient		
Searching & Sorting	Describe different characteristics for sorting algorithms including stability, in-place, adaptivity, partial sorting. Select appropriate algorithms based on these criteria	Essential	Proficient		
Searching & Sorting	Implement quadratic sorts - selection, insertion.	Essential	Proficient	Proficient	
Searching & Sorting	Implement mergesort and quicksort.	Essential	Proficient	Proficient	Not covered at EOU, UO
Searching & Sorting	Describe heapsort (Essential) and implement it (Recommended)	Essential	Exposure		Not covered at EOU
Searching & Sorting	Describe the logic of hybrid sorting algorithms (introsort, Timsort)	Recommended	Exposure		Not covered at EOU
Searching & Sorting	Describe non-comparison based sorting algorithms (bucket sort/radix sort) and identify situations in which they are appropriate	Recommended	Exposure		Not covered at EOU
General Data Structures	Recognize the difference between an Abstract Data Type and an implementation of that ADT.	Essential	Proficient		
General Data Structures	Describe expected operations for the following Abstract Data Types: list, sorted list, stack, queue, set, table.	Essential	Proficient		
General Data Structures	Describe expected operations for the following Abstract Data Types: deque.	Recommended	Proficient		
General Data Structures	Reason about different implementations of an abstract data type. Identify the contraints of implementations and recognize implicit vs explicit structure (representing a heap with an array; maintaining a sorted list).	Essential	Competence		
General Data Structures	Implement static or dynamically sized containers. Reason about complexities related to resizing a container and possible optimizations for a static container.	Essential	Competence	Competence	
General Data Structures	Build nested data structures using arrays and lists. (e.g. array of lists that might be used to implement a hash table)	Essential	Competence	Exposure	Not covered at EOU
General Data Structures	Implement shallow or deep copies of data structures and choose the appropriate type of copy for a particular job.	Essential	Proficient		
General Data Structures	Use features of a language to create a generic data structure. (i.e. a LinkedList that can hold any data type, not just a IntegerLinkedList)	Essential	Competence		
General Data Structures	Recognize iterator concepts and how to use them to write algorithms that interact with data structures.	Recommended	Exposure		Not covered at EOU
Lists	Implement a singly linked or doubly linked list including circular lists. For those structures implement fundamental algorithms like insert, remove, traverse, delete, and merge.	Essential	Proficient	Proficient	
Lists	Implement an array based list with fundamental algorithms like insert, remove, traverse, delete	Essential	Proficient	Proficient	
Stacks & Queues	Implement both stacks and queues using linked lists and arrays.	Essential	Proficient	Proficient	
Stacks & Queues	Use stacks and queues to implement algorithms. (e.g. basic parsing task using a stack)	Essential	Proficient	Proficient	
BST	Implement a node-based BST and fundamental algorithms like insert, contains, delete.	Essential	Proficient	Proficient	Not covered at UO
BST	Use appropriate terminology to describe BSTs and their nodes (height, depth, completness, subtree, etc...)	Essential	Proficient		Not covered at UO
BST	Implement pre/in/post order traversals and pick the appropriate strategy for a given task.	Essential	Proficient	Proficient	Not covered at UO
Heaps	Implement a binary heap with fundamental operations like add, get min/max, delete.	Essential	Competence	Competence	
Hashing & Hash Tables	Implement hash tables with fundamental operations like insert, remove, delete.	Essential	Proficient	Proficient	
Hashing & Hash Tables	Recognize what makes a good (or perfect) hash function. Construct a hash function for different data types.	Essential	Competence		
Hashing & Hash Tables	Use either open addressing or chaining to resolve colisions in a hash table.	Essential	Proficient	Proficient	
Self Balancing Trees	Describe how a self balancing tree will handle inserting or removing a value. Be familiar with logic behind AVL and/or RedBlack trees.	Essential	Competence	Competence	Not covered at UO
Self Balancing Trees	Describe how a B tree is organized and how values would be inserted or removed.	Essential	Competence		Not covered at EOU, UO
Graphs	Use appropriate terminology to describe graphs.	Essential	Competence	Not Expected	EOU covers in Algorithms.
Graphs	Identify different representations of a graph (adjacency list/matrix) and translate between them.	Essential	Proficient		EOU covers in Algorithms. WOU in Algorithms or math courses
Graphs	Identify traversal order for breadth first or or depth first searches on a graph.	Essential	Proficient		EOU covers in Algorithms. WOU in Algorithms or math courses
Graphs	Perform by hand basic graph based algorithms (e.g. shortest path, minimal spanning tree)	Recommended	Exposure	Not Expected	EOU covers in Algorithms. WOU in Algorithms or math courses