Welcome

CSCI 345
Data Structures and Algorithms
Wheaton College
Thomas VanDrunen
Spring 2022

1. The correctness of an algorithm can be verified formally using loop invariants and other proof techniques and empirically using unit tests.
2. The efficiency of an algorithm can be measured formally using algorithmic analysis, big-oh categories, etc, and empirically by running experiments.
3. Abstract data types, especially list, stack, queue, set, bag, and map, are specified by how they are used; data structures, such as arrays, linked lists, binary trees, and hash tables, are implementation strategies, each with trade-offs.
4. Searching in an unordered data structure such as a map can be done in logarithmic time using a balanced binary search tree .
5. Searching in an unordered data structure can be done in constant time using a hash table.
6. Problems with overlapping subproblems and optimal substructure can be solved efficiently using dynamic programming.

1. The correctness of an algorithm can be verified formally using loop invariants and other proof techniques and empirically using unit tests.

2. The efficiency of an algorithm can be measured formally using algorithmic analysis, big-oh categories, etc, and empirically by running experiments.

	formally		**empirically**
Correctness, verified	by invariants and correctness proofs	*and*	by unit tests
Efficiency, measured	by big-oh categories and related notation	*and*	by experiments

3. Abstract data types, especially list, stack, queue, set, bag, and map, are specified by how they are used; data structures, such as arrays, linked lists, binary trees, and hash tables, are implementation strategies, each with trade-offs.

ADTs	**Data structures**
List	Array
Set	Linked list and other linked structures
Map	Binary search tree
Stack	Hash table
Queue	
Bag	

The quest for the more efficient map

4. Searching in an unordered data structure such as a map can be done in logarithmic time using a balanced binary search tree .

5. Searching in an unordered data structure can be done in constant time using a hash table.

6. Problems with overlapping subproblems and optimal substructure can be solved efficiently using dynamic programming.

Other smaller topics: Sorting algorithms, graph algorithms, string algorithms, regular expressions, . . .

For next time:

Read Section 1.1

Start reading Section 1.2 (long section to be read by next Wednesday)

Do the pretest