Test 1 will be Tuesday, June 9 at 12:30 PM in class. It will be closed-book and notes, except for one 8.5" x 11" sheet of paper containing any notes that you want. (Yes, you can use both the front and back of this piece of paper.) The test will cover the following topics (and maybe more).

Chapter 11. Searching, Sorting, and Complexity Analysis
Machine dependent measures of performance: program running time and instruction count
Machine independent measures of performance: big-oh notation (definition), orders of complexity
Complexity analysis of an algorithm to determine its big-oh notation in the best, worst, and average cases
Analysis of searches and simple sort algorithms
Recursive divide-and-conquer vs. dynamic programming to improve the complexity of an algorithm, e.g., Fibonacci.
General concept of program profiling

Chapter 12. Tools for Design, Documentation, and Testing
Software design process with UML: class diagrams, use-case diagrams, and collaboration diagrams
Documentation at the Module, class, method, and function level in Python, pydoc
Preconditions and Postconditions, enforcement with by raising exceptions
Testing Approaches: haphazard, black-box, and white-box testing
When to test: unit, integration, acceptance, and regression testing
General concept of "proofs of program correctness"
pyunit testing in Python

Chapter 13. Collections, Arrays, and Linked Structures
General idea of collections and operations on collections, Abstract Data Types (ADTs) idea
Implementing Collections with arrays and tradeoffs
Implementing Collections with as linked structures and tradeoffs

Chapter 14. Linear Collections: Stacks
General concept of a stack: LILO, top and bottom
Stack Operations: pop, push., peek, len, isEmpty, str
Stack Implementations: ArrayStack and LinkedStack including complexity of operations
Stack Applications: evaluating arithmetic expressions, backtracking, run-time stack

Chapter 15. Linear Collections: Queues
General concept of a queue: FIFO, front and rear
Queue Operations: enqueue, dequeue, peek, len, isEmpty, str
Queue Implementations: LinkedQueue and ArrayQueue (circular, array implementation) including complexity of operations
Queue Applications: Simulations, round-robin scheduling
General concept of Priority Queue operations
Priority queue implementations: Comparable class, LinkedPriorityQueue implementation

Sections: 18.9 - 18.11 on Heaps their usage to implement priority queues
General concept of a storing a complete-binary tree in an array
Implementing a heap: heap-order property
Using a heap to implement a priority queue: HeapPriorityQueue class