COMP152 - SP22 - Standards

Topic/Skill	Standard
Program Analysis	
Big-O	Understand and interpret the worst-case, Big-0 characterization of a function or program running time
	Understand the limitations of Big-O analysis
	Make comparisons between programs based on the Big-O characterization of their running time
	Know the basic functions used by the text for Big-O characterization and their relative order
	Know the formal definition of Big-O it's implications for runtime analysis
	Be able to use Big-O to characterize the running time of a function or algorithm
Experimental	Know how to do basic profiling experiments by timing code on a variety of inputs
	Understand the limitations of experimental studies.
General	Compare and contrasts the role and value of Big-O analysis and experimental analysis when considering the performance of a program, function, or algorithm.
Programming	
Conditionals	Have some proficiency with the use of branching if..elif..else conditional statements. Know when to use them and how to write them.
	Understand and make use of boolean operators not, and, or when expressing conditionals.
	Be able to design and implement predicate functions for use in conditionals
Loops and Iteration	Be able to traverse through lists and arrays by value and by index. Know when to use the two approaches and how to write the corresponding loops.
	Understand and recognize when by-index traversal is necessary
	Know when to use iteration and accumulation to solve problems and know how to program them with loops.
Functions	Know the difference between functions and operations that return values and those that modify or mutate objects. Know how to use both.
	Have proficiency with the design, implementation, and testing of functions that return values, functions that modify one or more arguments, and functions that produce I/O effects.
Recursion	Understand how a recursive function works and how to analyze its running time.
	Know how to implement a recursive function with extra attention on identifying non-recursive base cases versus the recursive cases
Repetition Generally	Compare and contrast loop-based repetition versus recursive functions.
Abstract Data Types	Understand and explain the difference between mutable and immutable data types

Topic/Skill	**Standard**
	Understand how to interpret and produce an abstract, behavioral specification of a data type through constructors, accessors, mutators, and other key functionality.
	Understand the use of classes and Object-Oriented Programming to implement ADTs while hiding the underlying implementation details.
	Be able to use Big-O to characterize the runtime cost of using ADTs
	Know when and how to use the stack ADT.
	Know when and how to use the queue ADT
	Know when and how to use the positional list ADT
	Know when and how to use the heap and priority queue ADT
	Know when and how to use the dictionary ADT
	Know how to use while loops when working with mutation-heavy ADTs like stacks and queues.
	Be able to analyze a programming problem in order to determine which ADTs most appropriatly address the needs of that problem.
	Be familiar with binary trees, their associated vocabulary, and the analysis of their structure, especially in the case of heaps.
Program Development	
Design	Understand the use of top-down and bottom-up design, their differences, and when and how to use them when building a program
	Know how to create and use a function stub
	Be able to read and write program documentation. Know what makes for good documentation.
	Be proficient with the design and use of helper/auxiliary functions to manage program complexity
	Understand the use of versioning and iterative refinement in program design.
Debugging and Testing	Know how to write and use unit tests
	Know how to use and interpret the results of program Debuggers
	Know how to use print statements for the purpose of analyzing and debugging code
	Know how to work with Interactive Consoles, when available, for the purpose of analyzing and debugging code.
	Understand the uses and limits of testing when addressing the correctness of a program
Program UI	Know how to write and design a program with a basic command-line interface
	Know how to read and write to files and in particular the use of structured file formats like Comma separated values (CSV)
	Know how to carry out basic interactive I/O sequences as part of a program's UI.
Python	

Topic/Skill	Standard
	Know how to use ranges including backwards counting ranges and ranges with step sizes greater than 1
	Know when and how to use sequence slicing
	Know when and how to use list and dictionary comprehensions