Computer Science Introductory Course Topics

GEN_ENG 151 (note, all topics in the context of **one** language; represents AP CS A content at a 5 level)

- Basics of programming in **one** language (usually Java)
    - This class will assume you are **already** used to designing, developing, analyzing, and documenting code *in some object-oriented programming language.*
- Variables, Types, Objects
- Boolean expressions
- Arrays, ArrayLists, and Nested Arrays
- Iteration
- Abstraction + Basic Class design and implementation (constructors, instance variables, encapsulation, self/this/etc.; methods)
- Subclasses/superclasses (inheritance; polymorphism)
- Basic documentation and testing
- Basic recursion
- Introductory ethical / social thinking around CS

CS 180
- Basics of programming
- Types and type signatures
- Unit testing, test-driven development
- Recursion and special cases of recursion (iterative, generative, tree recursion)
- Debugging and reasoning about program failure
- Basic compound data (lists and record types)
- Procedures as data
- Functional programming (lambda expressions, etc.)
- Functional list and stream processing (mapping, filtering, folding)
- Imperative programming (sequencing, assignment, object mutation)
- Packaging code with data (e.g. methods in classes)
- Run-time type dispatch

CS 208
- Asymptotic performance analysis (worst case, average case, amortized complexity)
- Basic sequence structures (arrays, linked lists, dynamic arrays)
- ADTs: stacks, queues, dictionaries, etc.
- Sorting algorithms
- Hash tables and hash
- Tree and graph representations
- Tree walks and graph search
- Search trees, including self-balancing trees
- Shortest path algorithms, minimum spanning trees

- Priority queues, binary heaps
- Disjoint sets (union-find), including path compression
- Data design
- Relational model

CS 211
- Semantics of C and C++
- Basics of OOP - classes, inheritance, visibility, etc.
- parameter passing mechnism
- von Neumann machine model (memory, addresses, etc.)
- Pointer semantics, pointer arithmetic, etc.
- Basics of the Unix tool chain (g++/gcc, make, gdb, etc)
- Reasoning about and debugging in languages that aren't type- or memory-safe, i.e. memory leaks, uninitialized pointers, buffer overflow, etc.

- C++: variables, functions, standard I/O, file I/O, parameter passing mechanism, vector, array, classes and objects, inheritance, iterator, foreach, template, using search and sort, using built-in map
- C: standard I/O, struct, union, array, function, parameter passing mechanism, strings
- Computer memory: stack and heap, static and dynamic memory allocation
- Pointer arithmetic, pointer semantics
- Basics of the Unix tool chain (g++/gcc, make, gdb, etc)
- Unit testing
- Debugging non-trivial programs
- Reasoning about and debugging in languages that aren't type- or memory-safe, i.e. memory leaks, uninitialized pointers, buffer overflow, etc.