**Python**

**Level 0**

- ~~Hello World~~

- ~~Loop through a collection~~

- ~~Make a conditional decision~~

- ~~Read keyboard input~~

- ~~Brute force search an unsorted collection~~

- ~~Find the maximum key~~

- ~~Find the minimum key~~

- Binary search a sorted collection

- Simple Recursion

- Towers of Hanoi

- FizzBuzz

**Level 1**

- Binary addition

- Data Structures

- Implement a linked list

- Implement a stack

- Implement a queue

- Implement a heap

- Conway's game of life

**Level 2**

- Sorting

- Insertion Sort

- Bubble Sort

- Selection Sort

- Counting Sort

- Heap Sort

- Merge Sort

- Quick Sort

- Binary Insertion Sort

**Level 3**

- Data structures

- Implement a binary search tree

- Implement a red-black tree

- Implement an order statistic tree

- Dynamic programming

- Fastest way through assembly line

- Matrix chain multiplication

- Longest common subsequence

- Optimal Binary Search Tree

- Greedy algorithms

- Activity Selection Problem

- Huffman Code Tree

**Level 4**

- Data structures

- Implement a B-Tree

- Implement a Binomial Heap

- Implement a Fibonacci Heap

- Graph Algorithms

- Breadth-First Search

- Depth-First Search

- Topological Sort using DFS

- Minimum Spanning Tree using Kruskal's algorithm

- Minimum Spanning Tree using Prim's algorithm

**Level 5**

**Level 6**

**Level 7**

**Level 8**

**Web**

- Serve a static web page

- Serve a dynamic web page based on URL parameters

- Serve a dynamic web page based on POST request body parameters

- Accept a SOAP XML request and return a dynamic response

- Accept a REST JSON request and return a dynamic response

- Return a response based on data in a database

**Testing**

- Execute a unit test