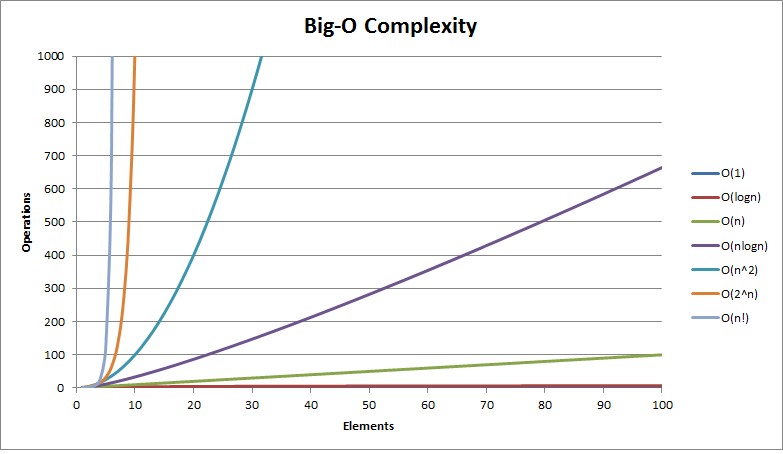
**Continuation of DS & A with a heavier emphasis on common algorithms (as well as Big O Analysis, type of data implementation, pros/cons, alternatives)**

* Exploration into Data Structures and Algorithms
  + Multiple "micro-projects" to demonstrate uses for different data structures (aka “ADTs”)
    - [**Lists**](https://en.wikipedia.org/wiki/List_(abstract_data_type))
      * Linked Lists
      * Arrays
    - **Sets**
    - **Stacks/Queues**
    - **Trees**/*Graphs* (not like stats graphs, but like [networks](https://sites.google.com/site/fusiontablestalks/stories/images/network-nodes.png))
    - *Hash Tables* (aka ”Dictionaries” in Python)
    - *Heaps* (Max Heaps and Min Heaps)
  + Deliverable: One “Assignment” per week to focus on data structures
    - Investigate different “implementations” of data structures
    - Explain some possible uses.
    - Draw a diagram showing common “methods” of the Data Structure.
    - Choose an implementation and write it in some language.
      * Have some tests to show your methods work.

**More complex understanding/use of Computational Complexity concepts in terms of algorithm design and popular implementation of data types**

* Computational Complexity ("Big O" Analysis)
  + A project demonstrating differences in the "computational complexity"/speed of different algorithms using a student-highlighted data structure. (Perhaps focusing on trees, since they're really cool, useful, and flexible.)
  + “Computational Complexity” = “How much work does this algorithm need to do?”
    - How do algorithms “scale” as the amounts of things change?
      * Constant, Logarithmic, Linear, Quadratic, Polynomial, Exponential, etc.



* + Deliverable: One “Assignment” per week to focus on data structures
    - Identify the complexities of the implementations you made.
      * Casey can review things if you want.
    - Explore how you can improve them -- and what pros/cons that causes.

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Week 1** | **Week 2** | **Week 3** | **Week 4** | **Week 5** | **Week 6** |
| Term 2 prep.  CompSci review  Sets  Graph traversals/Dijkstra’s algorithm | Sorting algorithms  Insertion,  Exchange/ Bubble,  Selection,  Bucket Merging | Spring Break | Continued review of traversing/sorting algorithms  Researched Max Heaps and Min Heaps | Heuristic Algorithms  Implementation of Dijkstra's Algorithm | Minimum Spanning Trees  Continuation of graph algorithms |
| **Weekly Assignment 1** | **Weekly Assignment 2** | **Weekly Assignment 3** | **Weekly Assignment 4** | **Weekly Assignment 5** | **Weekly Assignment 6** |
| N/A | *Assignment1T2Sorting.py* | N/A | *Assignment2T2Heaps.py* | *GREEDY\_SOLUTION\_(not\_optimal)\_Assignment3T2ShortestPath.py* | *Assignment3T2ShortestPath.py* |

**Week 1:**

**Day 1:**

▶ Prepared/refined curriculum for Term 2

▶ Worked on uploading, organizing materials from Term 1 on Google Drive/ GitHub/ laptop for future reference

▶ Started research and notes on sets

**Day 2:**

▶ Reviewed [Com-Sci](http://interactivepython.org/runestone/static/pythonds/Introduction/toctree.html) (e.g. definition of Comp-Sci, abstraction, programming, algorithms, primitive/abstract data types, data structures (implementations), object oriented programming paradigm, Input/Output system)

**Day 3:**

▶ Studied graph traversals/graph search algorithms

▶ Began Term 2’s focus on algorithms for graph/tree traversal:

*Dijkstra's Algorithm*, BFS, DFS, GBF A\*

**Week 2:**

**Day 1:**

▶ Started SORTING ALGORITHMS assignment *(Assignment1T2Sorting.py)*

▶ Studied/took notes on sorting algo. materials supplied in assignment

**Day 2:**

▶ Continued notes, studying, assignment

▶ Summarized different sorting algorithms (In GitHub “Term 2” Projects location)

**Week 3:**

**Spring Break**

**Week 4:**

**Day 1:**

▶ Finished working on sorting algorithm assignment

▶ Studied notes and web resources on breadth-first and depth-first graph searches

▶ Began research on min-heaps and max-heaps, as well as the searching/sorting algorithms used to traverse data stored as heaps *(Assignment2T2Heaps.py)*

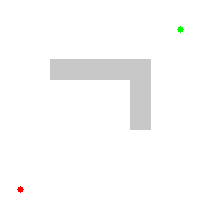
**Day 2:**

▶ Took notes on binary heaps using Interactive Python textbook

▶ Implementation of max heap

**Week 5:**

**Day 1:**

[**https://upload.wikimedia.org/wikipedia/commons/2/23/Dijkstras\_progress\_animation.gif**](https://upload.wikimedia.org/wikipedia/commons/2/23/Dijkstras_progress_animation.gif)

▶ Deeper exploration on the STEPS of Dijkstra's shortest path algorithm

▶ Studying the use of heuristic functions in computer algorithms

▶ Researching the conditions of a problem which would *warrant* the use of a heuristic (Uses Definition, Trade-Offs, Pitfalls)

**Day 2:**

▶ Programmed an interpretation of Dijkstra’s Algorithm to find the shortest possible path between a start node and and end node using data types: lists, tuples, and sets *(Assignment3T2ShortestPath.py)*

**Take home homework:**

▶ Getting Dijkstra’s Algorithm to pass the small tests successfully

**Week 6:**

**Day 1:**

▶ Continued to work on Assignment #3 Shortest Path

▶ Began review of previously learned topics from notes (and summarization) to prepare for the final 6 week cohesive project

▶ Researched Minimum Spanning Trees and the Kruskal Algorithm

**Day 2:**

▶ Continue notes on Min Spanning Trees (going through Kruskal’s Algorithm with test data input)

**Day 3:**

▶ Complete Dijkstra, solve issues in code

▶ Took notes on Prim’s Spanning Tree Algorithm

**Take home homework:**

▶ Brainstorm/plan out Term 3 Project

**\* Color when any work has been uploaded/resubmitted/etc. to GitHub**

***\* Style when a quiz or test has been taken***

**\* Color when there is no school scheduled for that day**