Keylon: "Let's break down the 18-week semester into (3) 6-week terms. So, what 3 topics would you like to explore? Once you decide, look at the curriculum you'll use. Yes, we do have a Lynda.com account if you'd like to use them. After curriculum is decided, what do you want to accomplish after each of the three units? This would be the assessment piece that is required on the IS form."

Proposed Outline for Independent Study

- Exploration into Data Structures and Algorithms
    - Multiple "micro-projects" to demonstrate uses for different data structures (aka "ADTs")[1][2]
        - **Lists**
            - Linked Lists
            - Arrays
        - **Sets**
        - **Stacks/Queues**
        - **Trees**/*Graphs* (not like stats graphs, but like networks)
        - *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.[3]

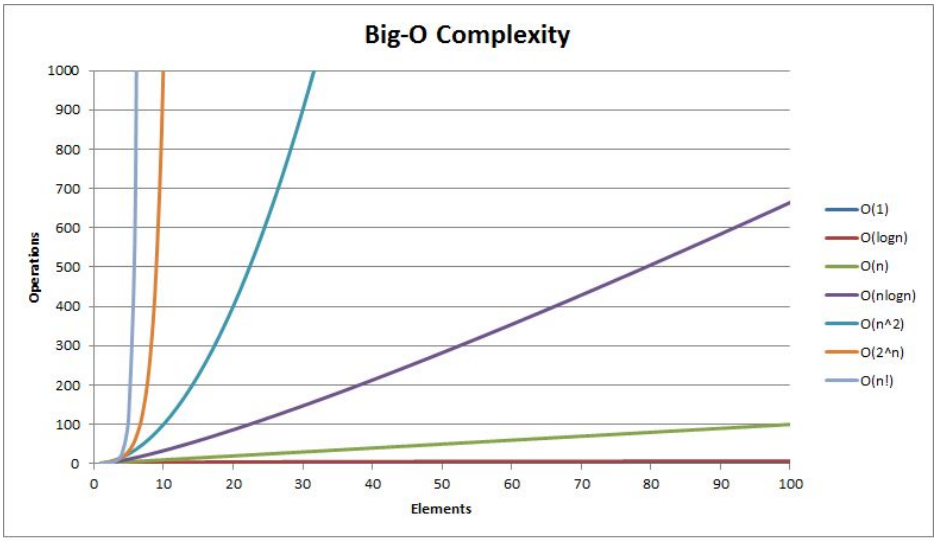
```
class List
    def __init__(self):
        my_thing = []
    def add(self, thing):
        self.my_stuff += [thing]
```

- Computational Complexity ("Big O" Analysis)[4]
    - 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?

---

[1] http://www.algolist.net/
[2] Best Resource Ever: https://interactivepython.org/runestone/static/pythonds/index.html
[3] http://docs.python-guide.org/en/latest/writing/tests/
[4] https://interactivepython.org/runestone/static/pythonds/index.html (Chapter 2)
https://hackr.io/tutorials/learn-data-structures-algorithms

- 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.
    - Put your analysis in a weekly “summary” or something? Whatever.

Imagine you had a list of N (5, 100, 10000, etc.) things. Now suppose you wanted to add ONE MORE thing to that list -- how much “work” do you have to do?

```
N = 10000
while (N>0):
   print N
   N -= 1
print “you done, bruh.”
```

- Synthesis Project
  - A synthesis project demonstrating good "data structure" and algorithm use, designed by the student. (Eg: a game, a complex program of some sort, etc.)
  - Deliverable: One “big” project to be turned in at the end of the semester.
    - Based on a topic of interest to the student.
    - Includes at least 4 different data structures with examples of “good use”
    - Brief explanation (in terms of “Big O”) of your data structure choices.
    - Put on GitHub
  - Possible ideas:
    - Dungeon-crawling game of your design
    - Visualize some data or something
    - Write a script to do something cool.

- Etc.