Presentation - 10-15 Minutes of Free Format Awesomness

Due: TBD

- Below is a large list of potential `data structures` and `algorithms` as topics that you might choose for your presentation.
- This list ranges from well-known topics to some more obscure or advanced ones.
- **I need to clear any topic that you might choose**, but especially if its something basic, in which you will need to get creative to present one of those with permission.
- My approval is needed so we get a good dispersal of algorithmic topics to keep the talks informative and not redundant.

Topics

```
topics:
  Basic:
    - Sorting Algorithms:
        - QuickSort
        - MergeSort
        - HeapSort
        - Counting Sort
        - Radix Sort
        - Bucket Sort
    - Searching Algorithms:
        - Binary Search
        - Interpolation Search
        - Linear Search
    - Hash Tables:
        - Chaining
        - Open Addressing
        - Cuckoo Hashing
        - Bloom Filters
        - Count-Min Sketch
    - Graph Algorithms:
        - BFS / DFS
        - Dijkstra's Algorithm
        - Bellman-Ford Algorithm
        - Floyd-Warshall Algorithm
        - A* Search
        - Topological Sort
    - Minimum Spanning Tree (MST) Algorithms:
        - Kruskal's Algorithm (using Union-Find)
        - Prim's Algorithm
        - Boruvka's Algorithm
    - Balanced Trees:
        - AVL Trees
        - Red-Black Trees
        - B-Trees / B+ Trees
        - Splay Trees
        - Treaps
```

```
  Intermediate:
    - Disjoint Set / Union-Find Structures (with Path Compression & Union
by Rank)
    - Heaps:
        - Binary Heap
        - Fibonacci Heap
        - Pairing Heap
    - Data Structures for Range Queries:
        - Segment Trees
        - Binary Indexed Trees (Fenwick Trees)
        - Interval Trees
    - String Data Structures:
        - Tries (Prefix Trees)
        - Suffix Trees
        - Suffix Arrays
        - Suffix Automata
        - Radix Trees
    - Dynamic Programming:
        - Knapsack Problems
        - Longest Common Subsequence
        - Edit Distance Algorithms
        - Matrix Chain Multiplication
    - Graph Flow Algorithms:
        - Ford-Fulkerson / Edmonds-Karp
        - Dinic's Algorithm
        - Push-Relabel Algorithm
  Advanced/Obscure:
    - Advanced Trees and Structures:
        - Weight-Balanced Trees
        - Skip Lists
        - Persistent Data Structures (e.g., Persistent Segment Trees)
        - Cache-Oblivious Data Structures
        - Link/Cut Trees
        - Euler Tour Trees
        - Skip Graphs
        - X-fast and Y-fast Tries
        - Wavelet Trees
    - Computational Geometry:
        - Convex Hull Algorithms (Graham Scan, Jarvis March, Quickhull)
        - Delaunay Triangulation
        - Voronoi Diagrams
    - Randomized and Online Algorithms:
        - Randomized QuickSort
        - Monte Carlo and Las Vegas Algorithms
        - Online Algorithms (Ski Rental Problem, Paging Algorithms)
        - Competitive Analysis
    - Streaming Algorithms:
        - Reservoir Sampling
        - HyperLogLog
    - Approximation Algorithms:
        - Greedy Algorithms for NP-hard Problems (e.g., Set Cover)
        - Polynomial-Time Approximation Schemes (PTAS)
    - Specialized Topics:
        - Graph Coloring and Partitioning Algorithms
```

```
        - Concurrency Data Structures (Lock-free, Wait-free)
        - Self-organizing Data Structures (e.g., Move-to-Front Heuristic)
        - Data Structure Lower Bounds and Decision Trees
        - Advanced String Matching (e.g., Knuth-Morris-Pratt, Boyer-Moore,
Rabin-Karp)
    - Random Topics:
        - Compression algorithms (huffman, LZW)
        - crypotographic hashes (one way hashes or public key encryption)
        - Evolutionary (old name Genetic Algorithms)
```

Presentation Topic Considerations

- **Interdisciplinary Topics**:
  - I don't care where you get a topic (unconventional, Interdisciplinary, or wacky smurf).
  - Your topic must be somewhat challenging and interesting for you and us (the class).
- **Hybrid/Modern Topics**: For those who want to delve into recent developments, topics like machine learning optimization algorithms (gradient descent variations), network flow in modern settings, or even aspects of algorithmic game theory might be appealing.

🎤 Student Presentation Guidelines

🎯 Purpose

This short presentation is an opportunity for you to dive deep into an algorithm or data structure and explain it clearly and confidently to an audience of your peers. The focus is on **communication, conceptual understanding, and clarity**—not flashy slides or polished visuals.

🧱 Structure (Suggested Outline)

You have 10–15 minutes, so use your time wisely:

1. **Intro (1–2 mins)**
   - Introduce your topic: What is the algorithm or data structure?
   - Why is it interesting? Where might it show up in the real world?
2. **Core Mechanics (4–6 mins)**
   - Walk through how it works at a high level.
   - Include a simple example to illustrate key steps or structure.
   - Highlight edge cases or design tradeoffs (e.g., when it works well, when it falls apart).
3. **Use Cases / Impact (2–3 mins)**
   - Real-world applications or theoretical significance.
   - Compare it briefly to alternatives if relevant.
4. **Wrap-Up / Q&A (1–2 mins)**
   - Quick summary of key points.
   - Open the floor for questions or discussion.

5. **Time Constraints**
    - If your topic is interestung and sparks discussion, I won't hold you to any time limit. In fact, some of my favorite teaching moments have been during student presentations where the (nearly) entire class was totally engrossed and asking questions.

📋 Topic Approval

- **Pick a topic** you're interested in or curious about (I will help you choose one if you're unsure).
- You **must get your topic approved** to avoid duplication and ensure appropriate scope.
- Bonus points for picking something a little unusual (e.g., Bloom filters, suffix trees, red-black trees, Aho-Corasick). Classic topics (binary search, BFS/DFS, heaps) are fine too—just make it your own and have some twist to make it interesting.

🎨 Presentation Style

- Digital materials (PowerPoint, slides, animations) are **optional**, unless you don't know your topic. Well, just know your topic.
- **Use the whiteboard** if that works better for drawing or explaining.
- **Use the doccam** I discovered this during this semester as I was lead by the nose by a particular group of three that acted like "awww shucks ... we think the doccam would be better because your writing on the whiteboard when breaking down Fibonacci has left the entire class scarred, probably for life ....".
- However, the key is: **don't rely on your slides to do the talking for you.** You should understand your topic so that the whiteboard or whatever tool is just in addition.
- In all seriousness ... I was a little cocky and had not done any multiple recursive call functions in a couple of years. You may not know this, and it may not be apparent, but I'm pretty good at what I do (mostly) especially for never having slides or notes.
- If I would have practiced for 10 minutes before class, I would have been golden.
- Your practice time will be longer, done in front of roomates, friends, in the car, and bathroom mirror. If you don't speak outloud when you prepare, your're not really preparing. The more you prepare, the more enjoyable it is.

💡 Tips for Success

- Practice your talk once or twice aloud—it'll boost your timing and confidence.
- Anticipate common questions or confusion points ("Why not just use a hash table?").
- Use real-world analogies or fun examples where it makes sense.
- If you hit a wall, ask for help—we're here to help you prep. Or turn it back to the class if you get stuck.

Presentation Readme

- An important part of doing a presentation is generating the materials from which you gleen your information.

- I don't mind the use of any LLVM (Chat GPT) since you need to know your topic when speaking about it.
- I encourage the use of an LLVM to help you create a readme file for your assignment that is full of facts, organized, and written in markdown.

Exam questions and a Handout

- Provide the following questions dealing with your chosen topic:
    - 3 multiple choice questions
    - 2 fill in the blank questions
- Creating a handout would be nice for the students. You don't have to print it, just create it and make it available within your repo. I can show you how Chat GPT can do this for you.

Goal: Show us that you understand a cool algorithm or data structure well enough to explain it. You don't need to be flashy—just clear, accurate, and interesting.

Final note: Teaching something to others is the final phase of learning in which it really resonates in your gray matter. So get up there and strut your algorithmic stuff!

✅ Deliverables

- Create a folder called P01 in your assignments folder.
    - Place a README in your P01 folder that will be the basis for your presentation material.
    - Make you cite any sources. If you give credit, you aren't plagiarizing.
    - If you do make presentation materials or handouts, place these here as well.
- **80%**

Category	Weight	Description
Conceptual Understanding	40%	Do you understand how and why the algorithm works?
Communication Clarity	30%	Can you explain the ideas clearly to your peers without jargon overload?
Organization	20%	Is your talk well-paced, with logical structure and transitions?
Engagement	10%	Did you involve the audience or make the topic feel interesting and lively?

- **20%**

Category	Description
Materials	Readme, handouts, slides, etc. All uploaded items.