Project 3: Report

I. Administrative

Team Name: Team Rex

Team Members: Gage Zahn, Marcus Lugrand, Benjamin Weiss

GitHub URL: <https://github.com/marcuslugrand/Ante-Up-by-Team-Rex>

Video:

II. Refined Proposal

1. Problem: What problem are we trying to solve?
   1. Provide a tool that allows for Poker hand Research.
2. Motivation: Why is this a problem?
   1. When playing a game of poker, it is often advantageous to know what hands are possible and how good each hand is.
3. Features implemented
   1. A working GUI menu to select from:
      1. Find all hands of quality X
      2. Find the quality of hand X
   2. Correctly executes menu options
   3. Displays a history of run times
   4. Displays card images of hands
   5. Runs on two different data structures
4. Description of data
   1. <https://archive.ics.uci.edu/ml/datasets/Poker+Hand>
   2. 1) S1 "Suit of card #1"
   3. 2) C1 "Rank of card #1"
   4. 3) S2 "Suit of card #2"
   5. 4) C2 "Rank of card #2"
   6. 5) S3 "Suit of card #3"
   7. 6) C3 "Rank of card #3"
   8. 7) S4 "Suit of card #4"
   9. 8) C4 "Rank of card #4"
   10. 9) S5 "Suit of card #5"
   11. 10) C5 "Rank of card 5"
   12. 11) CLASS "Poker Hand"
5. Tools/Languages/APIs/Libraries used
   1. C++, SFML
6. Algorithms implemented
   1. None
7. Additional Data Structures/Algorithms used
   1. Hash Map and Hash Tree
8. Distribution of Responsibility and Roles: Who did what?
   1. Ben: Hash Map
   2. Gage: SFML, Interaction and Display, Testing and Debugging
   3. Marcus: Tree Map

III. Analysis

1. Change Log
   1. Data Structures changed from Hash Map and Hash Set to Hash Map and Tree Map
   2. Subtle changes to interface
      1. Added Run Button
      2. Removed Data Structure Slider
   3. Feature Reduction
2. Complexity Analysis
   1. Map:
   2. unordered\_map:
      1. Function: Constructor
         1. O(1) in all cases.
      2. Function: insert
         1. O(1) average and best case, O(n) worst case if a rehash is triggered.
      3. Function: Search by Hand
         1. O(1) best and average case, O(n) worst case since open addressing is used.
      4. Function: Search by Quality
         1. O(1) all cases.
      5. Function: Access
         1. O(1) best and average case, O(n) worst case.

IV. Reflection

We ran into many issues with the maps. Getting the proper functionality to work and work efficiently enough to load in a reasonable time was often a challenge. There were times where our program took hours to load and open.

I would have liked to pick a project that had more real-world value. Unfortunately, it was difficult to find and obtain a data set that was both the required size, and practical.

Gage: I had never really understood how GitHub works before this project. I now feel like I understand enough to start keeping my projects on GitHub. I also learned how to store pairs in a hash map, and how to accept text input via SFML.

V. References

1. <https://archive.ics.uci.edu/ml/datasets/Poker+Hand>
2. <https://www.geeksforgeeks.org/how-to-create-an-unordered_map-of-pairs-in-c/>
3. <http://www.milefoot.com/math/discrete/counting/cardfreq.htm>