**Project 1**

**Implementation:**

This project was implemented into a single file containing all the necessary classes. Classes consisting of APriori, PCY and BitVector. The Apriori class is self identifying as the class which takes the support and data value and executes the apriori algorithm on them. The PCY class is also self-identifying as taking the support and data size and executing the selected PCY algorithm. Within the PCY class there are 3 types of PCY algorithms: 1. Normal PCY, 2. Multistage PCY, and 3. MultiHash PCY. The BitVector class is used by the MultiHash PCY algorithm.

The execution is quite straight forward, execute the main file(the only file) and then choose which algorithm you would like to run. The program will execute in order of increase support and increasing data size within each support. For example: 1% support and 1% data all the way to 10% support and 100% data for that particular algorithm.

**Computer Specifications:**

2020 MacBook Pro 16’

Processor: Intel Core i7-9750H 6-Core 2.6GHZ

Memory: 16GB DDR4 2667MHZ

Operating System: MacOS Big Sur 11.0.1

**Run-Time Results:**

**Conclusion:**

As we can see, A Priori consistently outperforms all variations of PCY. However, as we increase support and data size, we can see that Multistage PCY begins to match PCY and with increased values may ever outperform standard PCY. A very obvious observation from the data is the performance of MultiHash PCY. We can see that every other algorithm outperforms MultiHash PCY by a long shot, with very little change it it’s efficiency.