Running

Aprirori: *javac Apriori.java*
java Apriori

AprioriImproved: *javac AprioriImproved.java*
java AprioriImproved

FPGrowth: *javac FPGrowth.java*
java FPGrowth

It will ask you for file name and gives you suggestions for data.txt or test.txt see below. It will also ask for min support

Book (Any reference of the book from this documentation will refer to this)
Data Mining Concepts and Techniques: Jiawei Han, Micheline Kamber and Jian Pei

Data used for Testing (http://archive.ics.uci.edu/ml/datasets/Adult)
*UCI database **adult.data** (referred to as data.txt in this project) and **adult.test** (referred to as test.txt in this project)*

Project Structure
This project has 5 important .java folders and 2 data .txt files

1. **Apriori.java**
   a. Contain the implementation of Apriori algorithm based on **Chapter 6.2.1**
   b. See below for how it works
2. **AprioriImproved.java**
   a. Improvement based on some recommendations by graduate students in University of Rochester Data Mining Lab and the **Book Chapter 6.2.3**
   b. This improvement uses Hashtable and will be extrained more below.
3. **FPGrowth.java**
   a. Contain the implementation of Frequent Pattern Growth from the book
   b. More information below
4. **FPTree**
   a. Used by FPGrowth.java
5. **TableNode**
   a. Used by FPGrowth.java to generate table (referred to as header table)

<u>Apriori Algorithm</u>

I used almost exactly the same direction as in the book. The book provide step by step guide to this. For more internet based explanation I found that (https://www.hackerearth.com/blog/machine-learning/beginners-tutorial-apriori-algorithm-data-mining-r-implementation/) was really helpful.

I won't go into details with this because I follow almost the exact processes using ArrayList, String token and set to accomplish it

Apriori Improved Algorithm

I use the Hash Table and transaction scan reduction. HashTable is used to store the id and the count so you won't have to recalculate every time (recalculate the frequent). We also need to remove any set that doesn't pass the threshold.

This algorithm is explained beautifully by this slides (http://infolab.stanford.edu/~ullman/mining/pdf/assoc-rules2.pdf) by stanford.edu. It's referred to as PCY improvement (Park-Chen-Yu)

FP-Growth Algorithm

Like Apriori the book is a lovely tool for this but I really enjoyed reading this slides (http://www.cis.hut.fi/Opinnot/T-61.6020/2008/fptree.pdf) by a professor from csi. It's really an interesting read. In his explanation the concept of why we use FP Tree is easily explained.

Other Features

I made a lot of use of lambda functions for Java. Mostly in sorting but keep an eye on them.

Runtime Analysis:

For support of 0.15

Apriori 229.236ms
AprioriImproved 1.6ms
FPGrowth 0.774ms