CSCE 378H
Project 2 - Part 1

Michael Hollman
Cassey Lottman
Scott Johnson
Darren Johnson

Data Representation
We first defined an object model in C# to represent baskets and items. These map pretty closely to the provided basket.h and item.h files we provided, but use native C#. While we were executing the A-Priori algorithm, we represented the items we had found and their respective counts as a dictionary with keys of tuples containing the item ids (a single item ID on the first pass, a tuple of two items on the second pass, and a tuple of three items on the third pass). The value of the dictionary was the number of times that set had appeared in the processed baskets.

We used tuples because we could use them as sets. Tuples, as opposed to some other data structure like lists, allow for value equality, as opposed to object equality. This made comparisons of sets of item IDs very quick and easy. Using a dictionary allowed us to map each set to its count. Finally, LINQ extension methods for IEnumerable made it very easy to filter the dictionary by sets whose count was greater than our threshold of 3.

Timing Results
Timing our program yielded the following results:

File read time: 424 milliseconds
Data analysis runtime: 110 milliseconds
Total Execution time: 534 milliseconds

Complexity Analysis
The worst case complexity of our algorithm is O(b*i^3) where b is the number of baskets and i is the number of items. This case could result if all the baskets each have all the possible items. However, average case complexity is much smaller because most baskets contain a small subset of the total number of items, and by the time we get to the third iteration, we have narrowed down the items we need to check to only items that appear in more than 3 baskets. This complexity is in regards to the entire algorithm, but does not exhaustively include the (generally negligible) complexity discrepancies introduced by using some of C#'s helper methods, notably IEnumerable extension methods.