**Midterm Project Report**

*Implementation & Code Usage*

The main concept of the Apriori Algorithm is to create associations. In order to create associations, I had to figure out what items were most frequent when given the list of transactions. Once the items that are most frequent were found, depending on the user’s support parameter, the support would have to be calculated for each item. After calculating the support value for each item, we can eliminate the items that do not meet the user-defined support parameter. Next, I can then generate set combinations and calculate the support for those item-sets by cross-referencing them to the list of transactions and examine if these item-sets existing in the transactions list.

We would repeat this process and increase the number of items in the set by one until I no longer meet the user-defined support parameter. Once I have the item-sets that meet the support then I would have to generate the permutations and calculate the confidence with each item set.

When having the support and confidence of each item-set we can then filter the item-sets that meet the both user-defined parameters of support and confidence and acquire our final associations.

As for code usage, it was in my best interest to use the notions of sets, counters, data frames, and lambdas, loops, and well-defined functions to carry out specific tasks to achieve the Apriori Algorithm.

*Screenshots*

Here are what the csv files (This program takes in two separate csv files: Item Names & Transactions)



Figure 1: Item Names CSV file



Figure 2: Transactions CSV file

Below are screenshots of the code from python file:

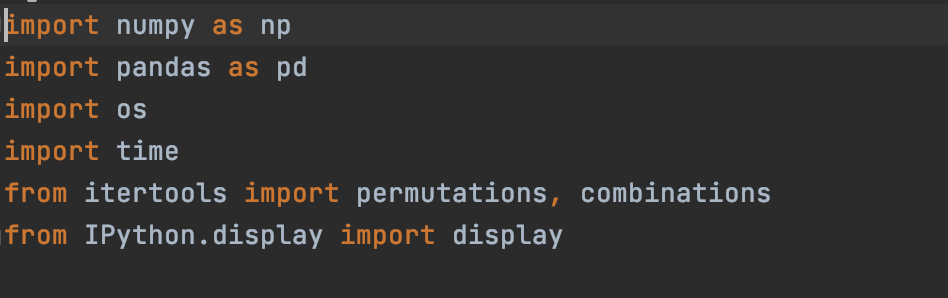


Figure 3: Imports to run the py file

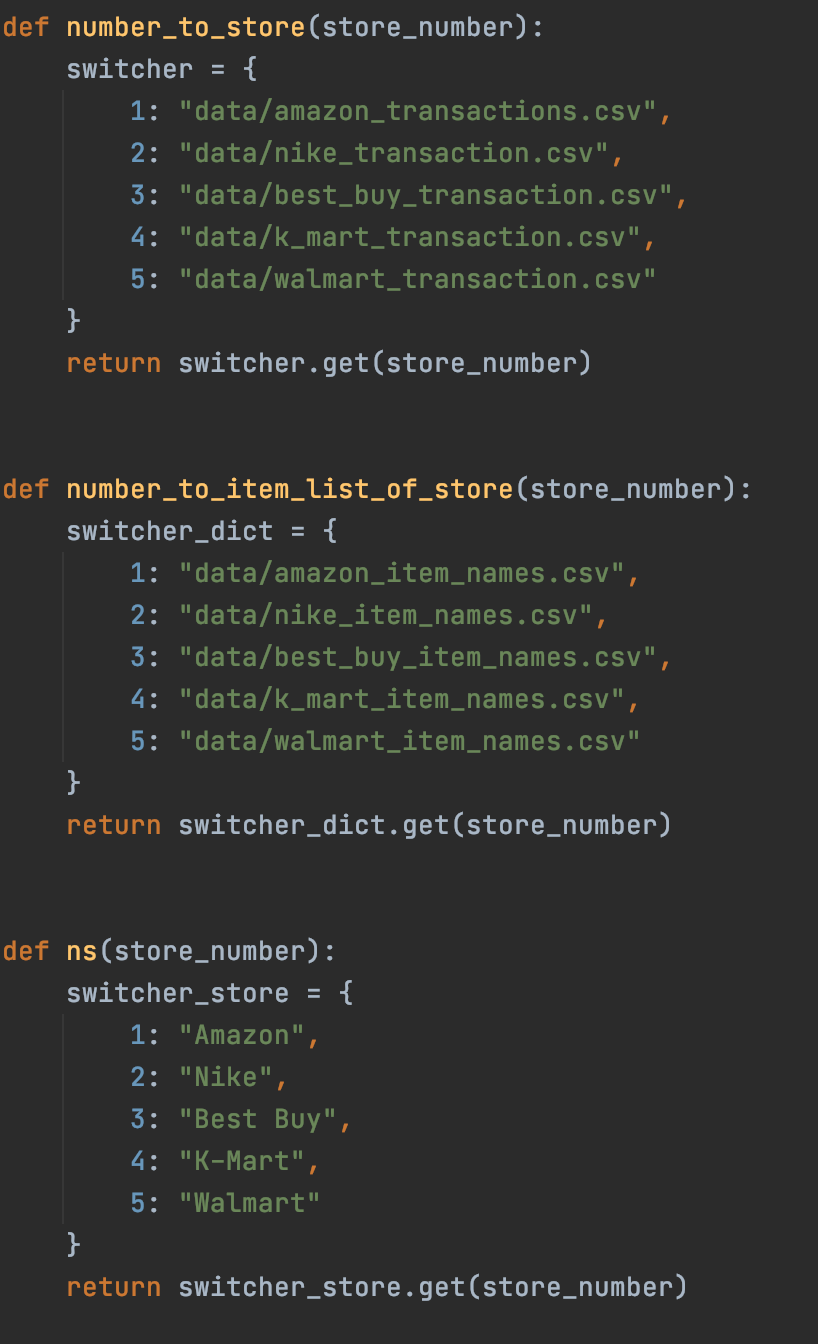


Figure 4:These are my dictionaries to choose which store to get based in Key-Value Pairs

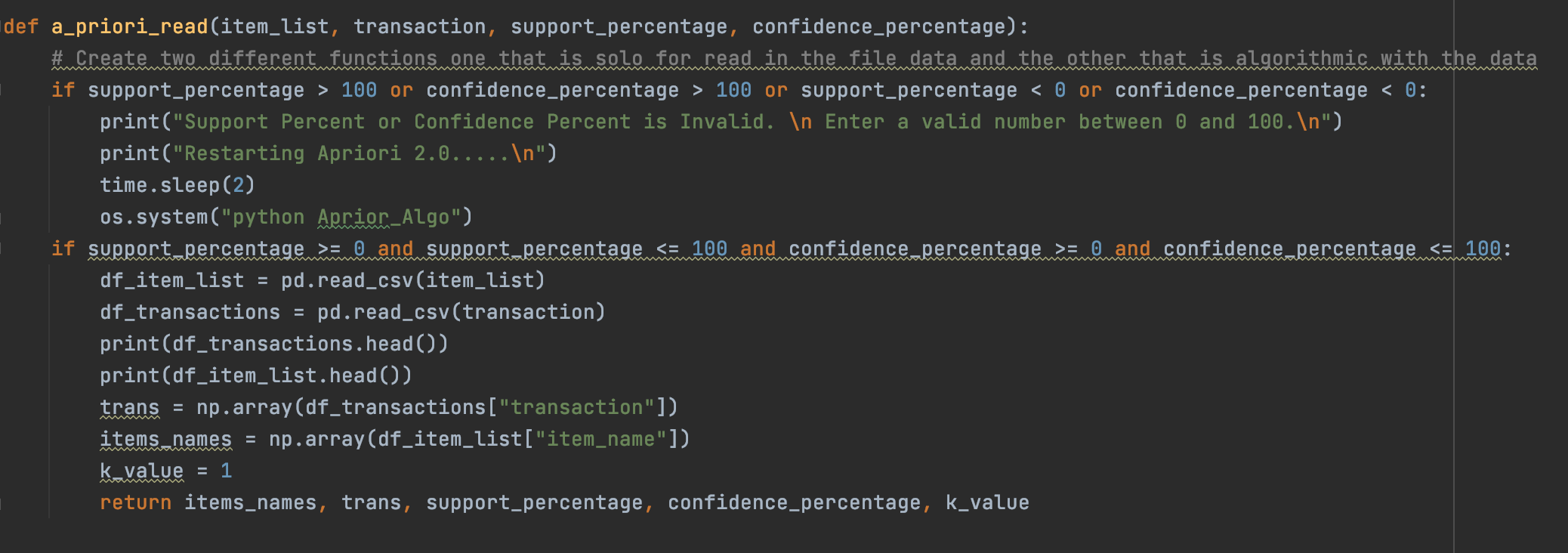


Figure 5:We first have to read in the csv files and make sure that the inputs received from the user are valid



Figure 6:The first go around of the Apriori Algorithm we find the items that are most frequent when K=1. This is so that we can find the most frequent items given the transactions

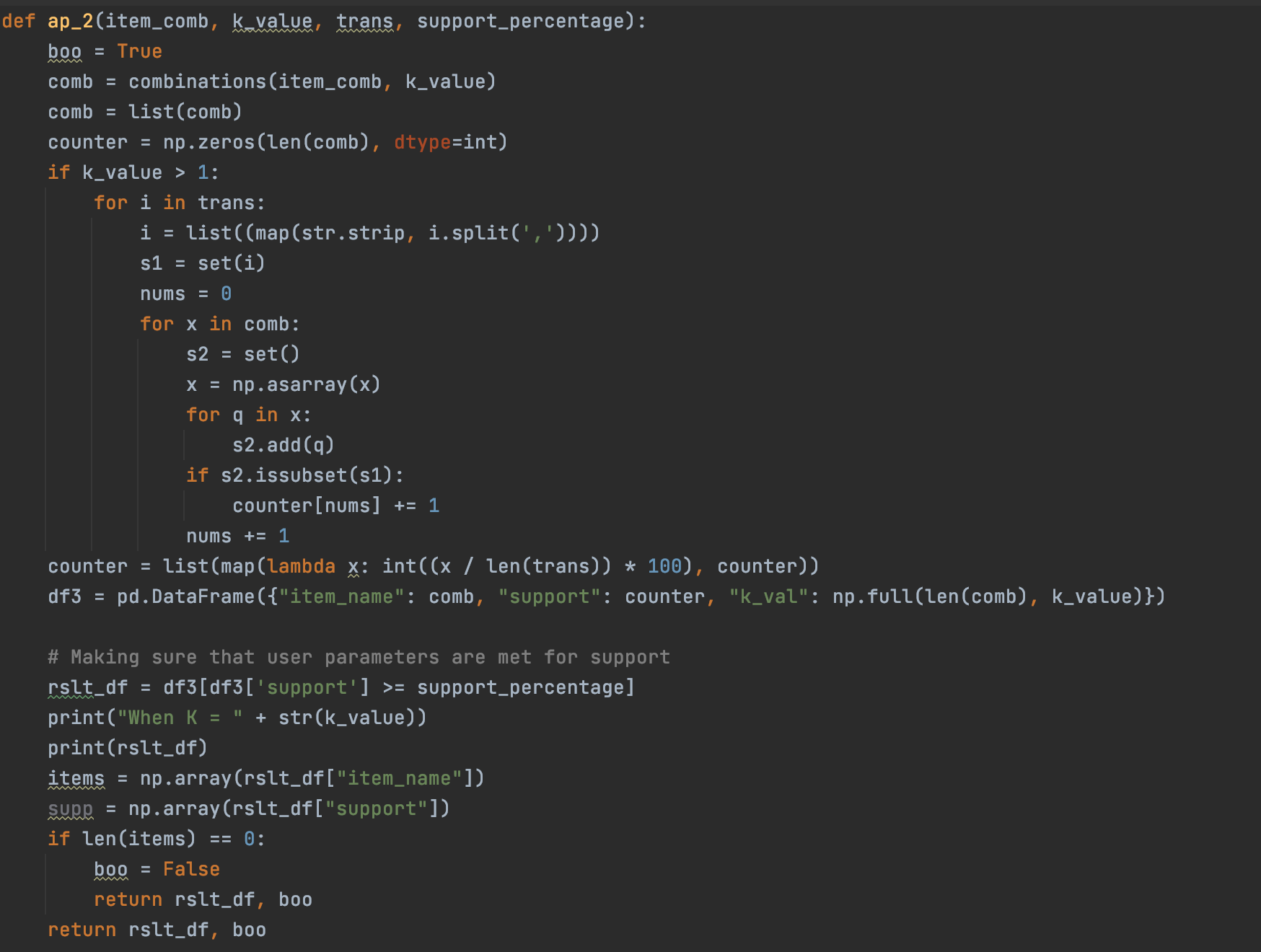


Figure 7: Then we use this function below to find item sets that are most frequent when K > 1

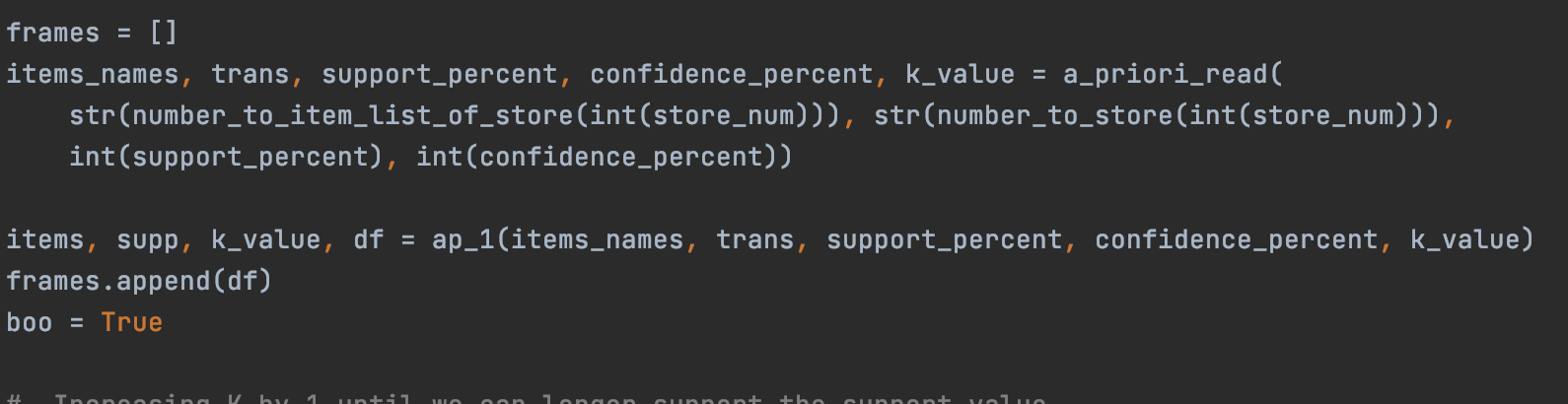


Figure 8: Calls of functions and variable saving

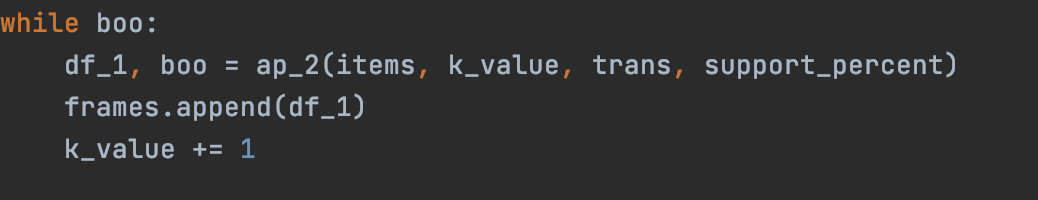


Figure 9:Increasing K by 1 until we can longer support the support value

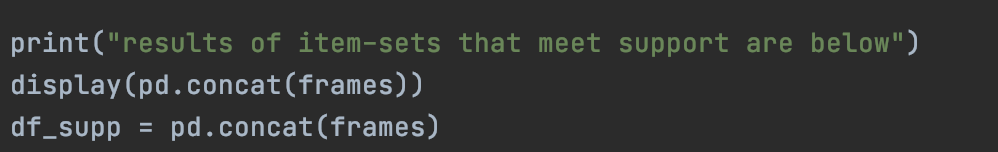


Figure 10: Combine the dataframes we have from when we increase K

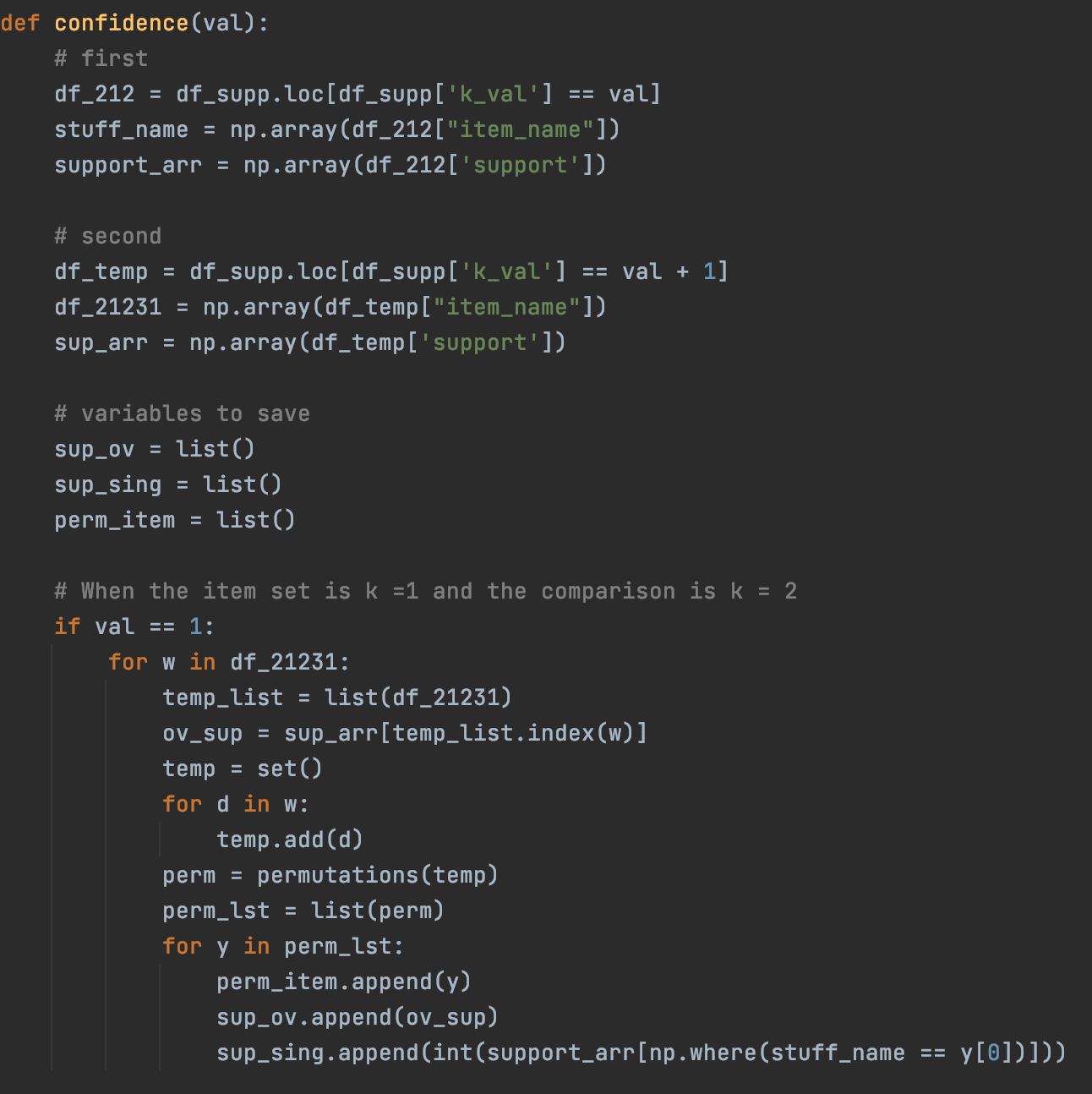


Figure 11.1: This is the FUNCTION that generates the Associations (Permutations) and calculating the Confidence of the item sets



Figure 10.2: This is still part of the confidence function. When the item set is k > 1 and the comparison is k += k + 1



Figure 12: Final output (final associations) of the Apriori Algorithm

Below are screenshots to show that the program runs in the Terminal.

Text

Description automatically generated

Figure 13: Follow the prompts when asked

The final output should be the following:

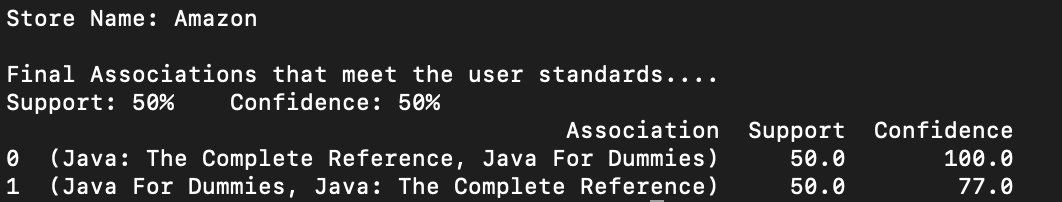


Figure 14: These are the following Associations given the user parameters

*Other*

The source code (.py file) and data sets (.csv files) will be attached to the zip file.

*Link to Git Repository*

<https://github.com/MichaelWoo-git/Apriori_Algorithm>