DATA MINING CS 634
MID-TERM PROJECT

APRIORI ALGORITHM
&
BRUTE FORCE APPROACH

By:

Divya Guduru

31488969

dg499@njit.edu

Problem

MIDTERM PROJECT 1

Create 30 items usually seen in Amazon, K-mart, or any other supermarkets (e.g. diapers, clothes, etc.).

i. Create a database of 20 transactions each containing some of these items. The information can be stored in a file, or a DBMS (e.g. ORACLE).

ii. Repeat (1) by creating 4 additional, different databases each containing 20 transactions. Using Apriori, generate and print out all the association rules and the input transactions for each of the 5 transaction databases you created (support and confidence should be user-determined parameter values, so the output should show different support and confidence values).

MIDTERM PROJECT 2

iii. Implement the brute force method and compare the brute force method with the Apriori algorithm on each of the 5 transaction databases you created. Present computation (CPU or clock) time to demonstrate that the Apriori algorithm is faster than the brute force method on each of the 5 transaction databases. The brute force method and Apriori algorithm should output the same association rules on each database.

- **Programming language used: JAVA**
- **OS Name- Microsoft Windows 10 Home**

AprioriAlgorithm Code:

```
package mining.data;

import java.io.*;
import java.util.*;

public class AprioriAlgorithm {
  List<String> items = new ArrayList<>(
      Arrays.asList("bread", "milk", "yogurt", "eggs", "cereal", "chips",
"chocolates", "cookies",
          "butter", "cheese", "almonds", "cashews", "walnuts", "pistachios",
"oats", "broccoli",
          "carrots", "spinach", "sweetcorn", "greenpeas", "tomato",
"orange", "apple", "banana",
          "strawberry", "blueberry", "blackberry", "avacado", "onion",
"potato"));

  List<String> totalItems = new ArrayList<String>();
  String splitter = " ";
  String fileName;
  double support;
  double confidence;
  String inputData[];
  Hashtable<String, Integer> supportportData = new Hashtable<String,
Integer>();

  private void candidateGeneration(int transactionCount) {
    String temp1, temp2;
    StringTokenizer token1, token2;
    ArrayList<String> tempItems = new ArrayList<String>();

    if (transactionCount == 1)
      for (int item = 1; item <= items.size(); item++)
        tempItems.add("" + item);
    else if (transactionCount == 2)
      for (int i = 0; i < totalItems.size(); i++) {
        token1 = new StringTokenizer(totalItems.get(i));
        temp1 = token1.nextToken();
        for (int j = i + 1; j < totalItems.size(); j++) {
          token2 = new StringTokenizer(totalItems.get(j));
          temp2 = token2.nextToken();
          tempItems.add(temp1 + " " + temp2);
        }
      }
    else
      for (int i = 0; i < totalItems.size(); i++) {
        for (int j = i + 1; j < totalItems.size(); j++) {
          temp1 = "";
          temp2 = "";
          token1 = new StringTokenizer(totalItems.get(i));
          token2 = new StringTokenizer(totalItems.get(j));
          for (int s = 0; s < transactionCount - 2; s++) {
            temp1 = temp1 + " " + token1.nextToken();
            temp2 = temp2 + " " + token2.nextToken();
          }
```

```java
          if (temp2.compareToIgnoreCase(temp1) == 0)
            tempItems.add((temp1 + " " + token1.nextToken() + " " +
token2.nextToken()).trim());
        }
      }
    totalItems.clear();
    totalItems = new ArrayList<String>(tempItems);
  }

  private void frequencyItemGenerator(int a) {
    StringTokenizer tokenIt, tokenFil;
    String line = null;
    boolean fl;
    boolean transactions[] = new boolean[items.size()];
    Vector<String> frequentItems = new Vector<String>();
    BufferedReader br = null;
    int item_counter[] = new int[totalItems.size()];
    int lines_count = 0;
    try {
      br = new BufferedReader(new FileReader(fileName));
      while ((line = br.readLine()) != null) {
        tokenFil = new StringTokenizer(line, splitter);
        for (int j = 0; j < items.size(); j++) {
          transactions[j] =
(tokenFil.nextToken().compareToIgnoreCase(inputData[j]) == 0);
        }
        for (int i = 0; i < totalItems.size(); i++) {
          fl = false;
          tokenIt = new StringTokenizer(totalItems.get(i));
          while (tokenIt.hasMoreTokens()) {
            fl = (transactions[Integer.valueOf(tokenIt.nextToken()) - 1]);
            if (!fl)
              break;
          }
          if (fl)
            item_counter[i]++;
        }
        lines_count++;
      }
      for (int i = 0; i < totalItems.size(); i++) {
        if ((item_counter[i] / (double) lines_count) >= support) {
          frequentItems.add(totalItems.get(i));
          supportportData.put(totalItems.get(i), item_counter[i]);
        }
      }
    } catch (IOException e) {
      e.printStackTrace();
      System.out.println(e);
    }
    totalItems.clear();
    totalItems = new ArrayList<String>(frequentItems);
  }

  public void initialize() {
    inputData = new String[items.size()];
    for (int i = 0; i < inputData.length; i++)
      inputData[i] = "1";
    totalItems = new ArrayList<String>();

    int freq_count = 0;
    do {
      String itemsData = null;
      String[] input_split_data = null;

      freq_count++;
      candidateGeneration(freq_count);
      frequencyItemGenerator(freq_count);

      if (totalItems.size() != 0) {
        System.out.println("Frequent " + freq_count + " set of items are:
");
        System.out.println("=======================");
        for (int i = 0; i < totalItems.size(); i++) {
          itemsData = totalItems.get(i);
          input_split_data = itemsData.split(splitter);
          int tot_confidence = supportportData.get(itemsData);
          int ind_confidence = 0;

          for (int j = 0; j < input_split_data.length; j++) {
            if (j == 0) {
              ind_confidence = supportportData.get(input_split_data[j]);
              if ((ind_confidence / tot_confidence) >= confidence) {
                System.out.print(" " +
items.get(Integer.parseInt(input_split_data[j]) - 1));
                if (freq_count != 1)
                  System.out.print(" ->");
              }
            } else {
              if ((ind_confidence / tot_confidence) >= confidence) {
                System.out.print(" " +
items.get(Integer.parseInt(input_split_data[j]) - 1));
              }
              if (j == input_split_data.length - 1)
                System.out.print(" - confidence: " + (double) ind_confidence
/ 4);
            }
          }
          System.out.println();
        }
        System.out.println();
      }
    } while (totalItems.size() > 1);
  }

  public static void main(String[] args) {
    long startTime = System.currentTimeMillis();
    AprioriAlgorithm algorithm = new AprioriAlgorithm();
    Scanner scn = new Scanner(System.in);
    try {
      System.out.print("Please provide file name: ");
      algorithm.fileName = scn.next();
      System.out.print("provide support value: ");
      algorithm.support = scn.nextDouble();
      if (algorithm.support <= 0.0 || algorithm.support >= 1.0) {
        System.out.println("support value should lie between 0 and 1: " +
algorithm.support);
        System.exit(0);
      }
      System.out.print("provide confidence value: ");
      algorithm.confidence = scn.nextDouble();
      if (algorithm.confidence <= 0.0 || algorithm.confidence >= 1.0) {
        System.out.println("confidence value should lie between 0 and 1: " +
algorithm.confidence);
        System.exit(0);
      }
      System.out.println();
      algorithm.initialize();
      long endTime = System.currentTimeMillis();
      long totalTime = endTime - startTime;
      System.out.println("Total time taken in seconds: " + (totalTime /
1000));
    } catch (Exception e) {
      e.printStackTrace();
      System.out.println("Please provide valid input.");
    }
  }
}
```

Explanation:

- As stated in the problem, used 30 different items that are found in 5 different stores.
- The Items include bread, milk, yogurt, eggs, cereal, chips, chocolates, cookies, butter, cheese, almonds, cashews, walnuts, pistachios, oats, broccoli, carrots, spinach, sweetcorn, greenpeas, tomato, orange, apple, banana, strawberry, blueberry, blackberry, avacado, onion, potato.
- Considered 5 stores BJ's, Costco, Kmart, Samsclub and Walmart.
- 0's and 1's was used to represent the items i.e., if an item was purchased it is represented as 1 else it was represented as 0.
- 20 transactions in the above-mentioned pattern were used for each store.
- Sample Transaction: Let us assume the customer purchases milk, yogurt, cereal, chocolates, cheese, cashews, walnuts, oats, carrots, greenpeas, orange, apple, strawberry, blackberry, potato. Then, the pattern will be represented as below:
  - 0 1 1 0 1 0 1 0 0 1 0 1 1 0 1 0 1 0 0 1 0 1 1 0 1 0 1 0 0 1
- Used the above details both for Apriori Algorithm and Bruteforce method.

Execution:

- On execution of the code, we need to provide below details:
    - File name i.e., store name (any of the 5 stores)
    - Support and Confidence values
- Note: Support and Confidence should lie between 0 and 1
- Upon providing the above details, association rules will be generated, and the time taken to perform this computation
- Screen shots of input files and execution times provided after bruteforce approach algorithm explanation.
- Both algorithms are compared with respect to time taken to compute the frequent itemsets.

BruteForce Approach:

```
 /* Main.java */
package mining.data;
import java.io.BufferedReader;
import java.nio.file.Files;
import java.nio.file.Paths;
import java.util.ArrayList;
import java.util.Arrays;
import java.util.HashSet;
import java.util.List;
import java.util.Scanner;
import java.util.Set;

public class Main {

  static List<String> items = new ArrayList<>(
      Arrays.asList("bread", "milk", "yogurt", "eggs", "cereal", "chips",
"chocolates", "cookies",
          "butter", "cheese", "almonds", "cashews", "walnuts", "pistachios",
"oats", "broccoli",
          "carrots", "spinach", "sweetcorn", "greenpeas", "tomato",
"orange", "apple", "banana",
          "strawberry", "blueberry", "blackberry", "avacado", "onion",
"potato"));

  public static void main(String[] args) {
    long startTime = System.currentTimeMillis();
    FrequentItemsetGenerator generator = new FrequentItemsetGenerator();
    List<Set<String>> itemsetList = new ArrayList<>();
    @SuppressWarnings("resource")
    Scanner scanner = new Scanner(System.in);
    try {
      System.out.print("Please provide file name: ");
      String fileName = scanner.next();
      System.out.print("please provide support value: ");
      Double support = scanner.nextDouble();
      if (support <= 0.0 || support >= 1.0) {
        System.out.println("support value should lie between 0 and 1: " +
support);
        System.exit(0);
      }
      System.out.print("provide confidence value: ");
      Double confidence = scanner.nextDouble();
      if (confidence <= 0.0 || confidence >= 1.0) {
        System.out.println("confidence value should lie between 0 and 1: " +
confidence);
        System.exit(0);
      }
      try (BufferedReader bufferedReader =
Files.newBufferedReader(Paths.get(fileName))) {
        String line;
        while ((line = bufferedReader.readLine()) != null) {
          line = line.toLowerCase();
          String[] values = line.split(" ");
          Set<String> transaction = new HashSet<String>();
          for (int i = 0; i < values.length; i++) {
            if (values[i].equals("1"))
              transaction.add(items.get(i));
          }
          itemsetList.add(transaction);
        }
        bufferedReader.close();
      } catch (java.io.IOException e) {
        e.printStackTrace();
      }
      FrequentItemsetDto dataSet = generator.generate(itemsetList, support,
items);
      int i = 1;

      System.out.println("computed top level associations:");
      for (Set<String> itemset : dataSet.getFrequentItemsetList()) {
        if (itemset.size() == dataSet.getK() - 1)
          System.out.printf("%2d: %10s \n", i++, itemset);
      }

      long endTime = System.currentTimeMillis();
      long totalTime = endTime - startTime;
      System.out.println("Total time taken in seconds with brute force
approach: " + (totalTime / 1000));
    } catch (Exception e) {
      e.printStackTrace();
      System.out.println("Please provide valid input.");
    }
  }
}

/* FrequentItemsetGenerator.java */
package mining.data;
import java.util.ArrayList;
import java.util.Collections;
import java.util.Comparator;
import java.util.HashMap;
import java.util.HashSet;
import java.util.List;
import java.util.Map;
import java.util.Objects;
import java.util.Set;

public class FrequentItemsetGenerator {

  public FrequentItemsetDto generate(List<Set<String>> transactionList,
double minimumSupport,
      List<String> itemAttributes) {
    Objects.requireNonNull(transactionList, "The itemset list is empty.");
    checkSupport(minimumSupport);

    if (transactionList.isEmpty()) {
      return null;
    }

    Map<Set<String>, Integer> supportCountMap = new HashMap<>();

    List<Set<String>> frequentItemList =
        findFreqItems(transactionList, supportCountMap, minimumSupport,
itemAttributes);

    Map<Integer, List<Set<String>>> map = new HashMap<>();
    map.put(1, frequentItemList);

    int k = 1;

    do {
      ++k;
      List<Set<String>> itemsSet = new ArrayList<>();
      getAllItemSets(itemAttributes, itemAttributes.size(), k, itemsSet);
      System.out.println("generated possible " + k + "-itemsets " +
itemsSet.size());

      List<Set<String>> candidateList = itemsSet;

      for (Set<String> transaction : transactionList) {
        List<Set<String>> candidateList2 = subset(candidateList,
transaction);

        for (Set<String> itemset : candidateList2) {
          supportCountMap.put(itemset, supportCountMap.getOrDefault(itemset,
0) + 1);
        }
      }



      map.put(k,
          getNextItemsets(candidateList, supportCountMap, minimumSupport,
transactionList.size()));

      System.out.println("frequent Item sets List from possible " + k + "-
item sets of "
          + itemsSet.size() + " are:");
      map.get(k).stream().forEach(s -> System.out.println(s));
      System.out.println();

    } while (!map.get(k).isEmpty());

    return new FrequentItemsetDto(extractFrequentItemsets(map),
supportCountMap, minimumSupport,
        transactionList.size(), k);
  }

  private List<Set<String>> extractFrequentItemsets(Map<Integer,
List<Set<String>>> map) {
    List<Set<String>> ret = new ArrayList<>();

    // ret.addAll(map.get(k));
    for (List<Set<String>> itemsetList : map.values()) {
      ret.addAll(itemsetList);
    }

    return ret;
  }

  private List<Set<String>> getNextItemsets(List<Set<String>> candidateList,
      Map<Set<String>, Integer> supportCountMap, double minimumSupport, int
transactions) {
    List<Set<String>> ret = new ArrayList<>(candidateList.size());

    for (Set<String> itemset : candidateList) {
      if (supportCountMap.containsKey(itemset)) {
        int supportCount = supportCountMap.get(itemset);
        double support = 1.0 * supportCount / transactions;

        if (support >= minimumSupport) {
          ret.add(itemset);
        }
      }
    }

    return ret;
  }

  private List<Set<String>> subset(List<Set<String>> candidateList,
Set<String> transaction) {
    List<Set<String>> ret = new ArrayList<>(candidateList.size());

    for (Set<String> candidate : candidateList) {
      if (transaction.containsAll(candidate)) {
        ret.add(candidate);
      }
    }

    return ret;
  }

  @SuppressWarnings({"unused"})
  private List<Set<String>> generateCandidates(List<Set<String>>
itemsetList) {
    List<List<String>> list = new ArrayList<>(itemsetList.size());
```

```java
    for (Set<String> itemset : itemsetList) {
      List<String> l = new ArrayList<>(itemset);
      Collections.<String>sort(l, ITEM_COMPARATOR);
      list.add(l);
    }

    int listSize = list.size();

    List<Set<String>> ret = new ArrayList<>(listSize);

    for (int i = 0; i < listSize; ++i) {
      for (int j = i + 1; j < listSize; ++j) {
        Set<String> candidate = mergeItemSets(list.get(i), list.get(j));

        if (candidate != null) {
          ret.add(candidate);
        }
      }
    }

    return ret;
  }



  private static final Comparator<Object> ITEM_COMPARATOR = new
Comparator<Object>() {

    @SuppressWarnings("unchecked")
    @Override
    public int compare(Object o1, Object o2) {
      return ((Comparable<Object>) o1).compareTo(o2);
    }

  };

  private List<Set<String>> findFreqItems(List<Set<String>> itemsetList,
      Map<Set<String>, Integer> supportCountMap, double minimumSupport,
List<String> itemAttributes) {
    Map<String, Integer> map = new HashMap<>();

    List<Set<String>> itemsSet = new ArrayList<>();
    getAllItemSets(itemAttributes, itemAttributes.size(), 1, itemsSet);
    System.out.println("generated possible " + 1 + "-itemsets " +
itemsSet.size());

    // Count the support counts of each item.
    for (Set<String> itemset : itemsetList) {
      for (String item : itemset) {
        Set<String> tmp = new HashSet<>(1);
        tmp.add(item);

        if (supportCountMap.containsKey(tmp)) {
          supportCountMap.put(tmp, supportCountMap.get(tmp) + 1);
        } else {
          supportCountMap.put(tmp, 1);
        }

        map.put(item, map.getOrDefault(item, 0) + 1);
      }
    }

    List<Set<String>> frequentItemsetList = new ArrayList<>();

    for (Map.Entry<String, Integer> entry : map.entrySet()) {
      if (1.0 * entry.getValue() / itemsetList.size() >= minimumSupport) {
        Set<String> itemset = new HashSet<>(1);
        itemset.add(entry.getKey());
        frequentItemsetList.add(itemset);
      }
    }

    System.out.println(
        "frequent Item sets List from possible " + 1 + "-item sets of " +
itemsSet.size() + "are:");
    frequentItemsetList.stream().forEach(s -> System.out.println(s));
    System.out.println();
    return frequentItemsetList;
  }

  private void checkSupport(double support) {
    if (Double.isNaN(support)) {
      throw new IllegalArgumentException("The input support is NaN.");
    }

    if (support > 1.0) {
      throw new IllegalArgumentException(
          "The input support is too large: " + support + ", " + "should be
at most 1.0");
    }

    if (support < 0.0) {
      throw new IllegalArgumentException(
          "The input support is too small: " + support + ", " + "should be
at least 0.0");
    }
  }

  private void combinationUtil(List<String> arr, int n, int r, int index,
String data[], int i,
      List<Set<String>> itemsSet) {
    if (index == r) {

      Set<String> items = new HashSet<>();
      for (int j = 0; j < r; j++) {
        items.add(data[j]);
      }
      itemsSet.add(items);
      return;
    }

    if (i >= n)
      return;
    data[index] = arr.get(i);
    combinationUtil(arr, n, r, index + 1, data, i + 1, itemsSet);

    combinationUtil(arr, n, r, index, data, i + 1, itemsSet);
  }

  private void getAllItemSets(List<String> arr, int n, int r,
List<Set<String>> itemsSet) {
    String data[] = new String[r];
    combinationUtil(arr, n, r, 0, data, 0, itemsSet);
  }

  private Set<String> mergeItemSets(List<String> itemset1, List<String>
itemset2) {
    int length = itemset1.size();

    for (int i = 0; i < length - 1; ++i) {
      if (!itemset1.get(i).equals(itemset2.get(i))) {
        return null;
      }
    }

    if (itemset1.get(length - 1).equals(itemset2.get(length - 1))) {
      return null;
    }

    Set<String> ret = new HashSet<>(length + 1);

    for (int i = 0; i < length - 1; ++i) {
      ret.add(itemset1.get(i));
    }

    ret.add(itemset1.get(length - 1));
    ret.add(itemset2.get(length - 1));
    return ret;
  }
}

 /* FrequentItemsetDto.java */
package mining.data;
import java.util.List;
import java.util.Map;
import java.util.Set;


public class FrequentItemsetDto {
  private final double minimumSupport;
  private final int numberOfTransactions;
  private int k;
  private final List<Set<String>> freqItemsetList;
  private final Map<Set<String>, Integer> supportCountMap;

  FrequentItemsetDto(List<Set<String>> frequentItemsetList,
      Map<Set<String>, Integer> supportCountMap, double minSupport, int
transactionNumber, int k) {
    this.freqItemsetList = frequentItemsetList;
    this.supportCountMap = supportCountMap;
    this.minimumSupport = minSupport;
    this.numberOfTransactions = transactionNumber;
    this.k = k;
  }

  public List<Set<String>> getFrequentItemsetList() {
    return freqItemsetList;
  }

  public Map<Set<String>, Integer> getSupportCountMap() {
    return supportCountMap;
  }

  public double getMinimumSupport() {
    return minimumSupport;
  }

  public int getTransactionNumber() {
    return numberOfTransactions;
  }

  public double getSupport(Set<String> itemset) {
    return 1.0 * supportCountMap.get(itemset) / numberOfTransactions;
  }

  public int getK() {
    return k;
  }

  public void setK(int k) {
    this.k = k;
  }
}
```

Explanation:

- As stated in the problem, used 30 different items that are found in 5 different stores.
- The Items include bread, milk, yogurt, eggs, cereal, chips, chocolates, cookies, butter, cheese, almonds, cashews, walnuts, pistachios, oats, broccoli, carrots, spinach, sweetcorn, greenpeas, tomato, orange, apple, banana, strawberry, blueberry, blackberry, avacado, onion, potato.
- Considered 5 stores BJ’s, Costco, Kmart, Samsclub and Walmart.
- 0’s and 1’s was used to represent the items i.e., if an item was purchased it is represented as 1 else it was represented as 0.
- 20 transactions in the above-mentioned pattern were used for each store.

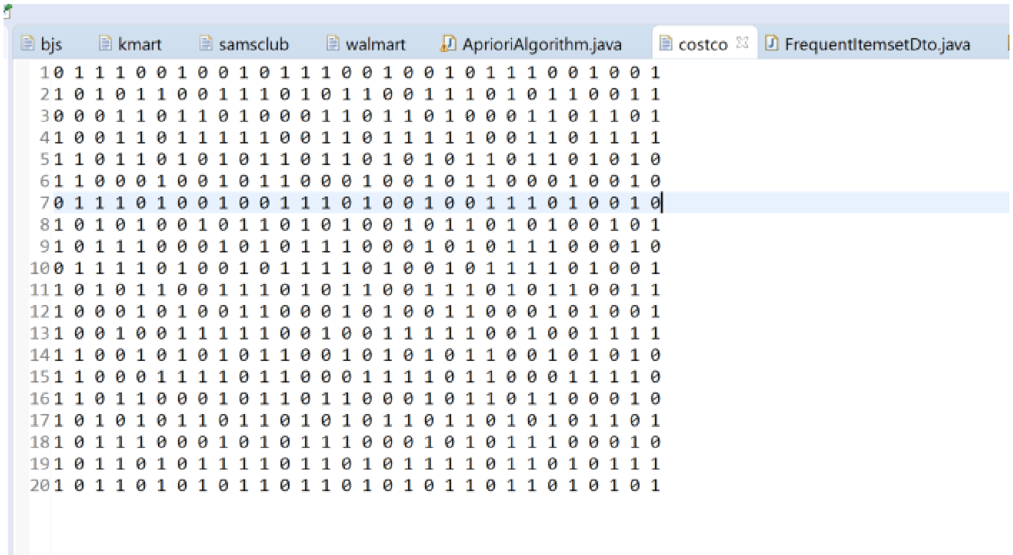
- Sample Transaction: Let us assume the customer purchases milk, yogurt, cereal, chocolates, cheese, cashews, walnuts, oats, carrots, greenpeas, orange, apple, strawberry, blackberry, potato. Then, the pattern will be represented as below:
    - 0 1 1 0 1 0 1 0 0 1 0 1 1 0 1 0 1 0 0 1 0 1 1 0 1 0 1 0 0 1
- The brute force method computes the frequent item sets works as follows. And enumerates through all generated possible 1-itemsets and 2-itemsets, possible 3-itemsets and soon, untill the next frequent itemsets possible from the generated sets and then terminates the candidate generation process.
- Used the above details both for Brute force method.

Execution:

- On execution of the code, we need to provide below details:
    - File name i.e., store name (any of the 5 stores)
    - Support and Confidence values
- Note: Support and Confidence should lie between 0 and 1
- Upon providing the above details, association rules will be generated, and the time taken to perform this computation followed program termnation.
- To run this program use any ide and run as java application and provide the inputs as described above.

Transaction Files:

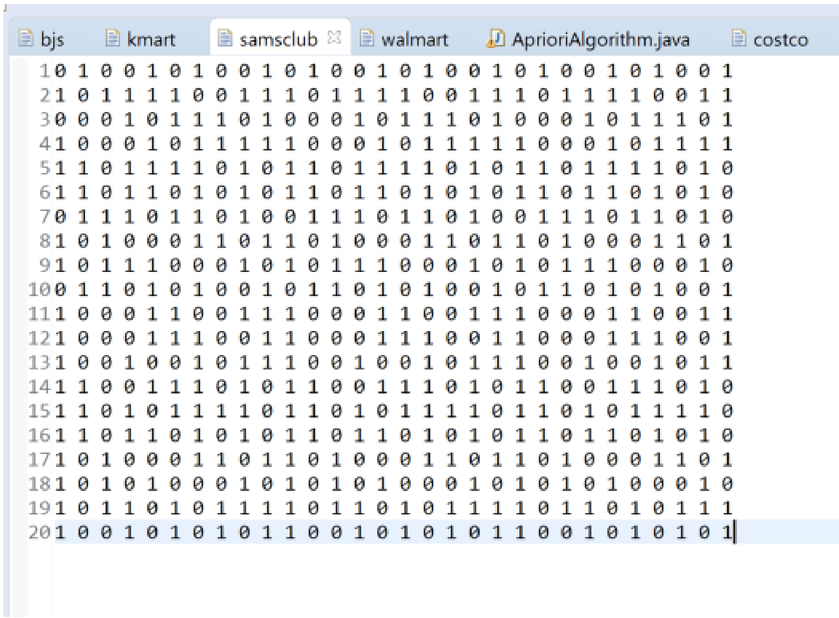
costco:



```
0 1 1 1 0 0 1 0 0 1 0 1 1 1 0 0 1 0 0 1 0 1 1 1 0 0 1 0 0 1
1 0 1 0 1 1 0 0 1 1 1 0 1 0 1 1 0 0 1 1 1 0 1 0 1 1 0 0 1 1
0 0 0 1 1 0 1 1 0 1 0 0 0 1 1 0 1 1 0 1 0 0 0 1 1 0 1 1 0 1
1 0 0 1 1 0 1 1 1 1 1 0 0 1 1 0 1 1 1 1 1 0 0 1 1 0 1 1 1 1
1 1 0 1 1 0 1 0 1 0 1 1 0 1 1 0 1 0 1 0 1 1 0 1 1 0 1 0 1 0
1 1 0 0 0 1 0 0 1 0 1 1 0 0 0 1 0 0 1 0 1 1 0 0 0 1 0 0 1 0
0 1 1 1 0 1 0 0 1 0 0 1 1 1 0 1 0 0 1 0 0 1 1 1 0 1 0 0 1 0
1 0 1 0 1 0 0 1 0 1 1 0 1 0 1 0 0 1 0 1 1 0 1 0 1 0 0 1 0 1
1 0 1 1 1 0 0 0 1 0 1 0 1 1 1 0 0 0 1 0 1 0 1 1 1 0 0 0 1 0
0 1 1 1 1 0 1 0 0 1 0 1 1 1 1 0 1 0 0 1 0 1 1 1 1 0 1 0 0 1
1 0 1 0 1 1 0 0 1 1 1 0 1 0 1 1 0 0 1 1 1 0 1 0 1 1 0 0 1 1
1 0 0 0 1 0 1 0 0 1 1 0 0 0 1 0 1 0 0 1 1 0 0 0 1 0 1 0 0 1
1 0 0 1 0 0 1 1 1 1 1 0 0 1 0 0 1 1 1 1 1 0 0 1 0 0 1 1 1 1
1 1 0 0 1 0 1 0 1 0 1 1 0 0 1 0 1 0 1 0 1 1 0 0 1 0 1 0 1 0
1 1 0 0 0 1 1 1 1 0 1 1 0 0 0 1 1 1 1 0 1 1 0 0 0 1 1 1 1 0
1 1 0 1 1 0 0 0 1 0 1 1 0 1 1 0 0 0 1 0 1 1 0 1 1 0 0 0 1 0
1 0 1 0 1 0 1 1 0 1 1 0 1 0 1 0 1 1 0 1 1 0 1 0 1 0 1 1 0 1
1 0 1 1 1 0 0 0 1 0 1 0 1 1 1 0 0 0 1 0 1 0 1 1 1 0 0 0 1 0
1 0 1 1 0 1 0 1 1 1 1 0 1 1 0 1 0 1 1 1 1 0 1 1 0 1 0 1 1 1
1 0 1 1 0 1 0 1 0 1 1 0 1 1 0 1 0 1 0 1 1 0 1 1 0 1 0 1 0 1
```

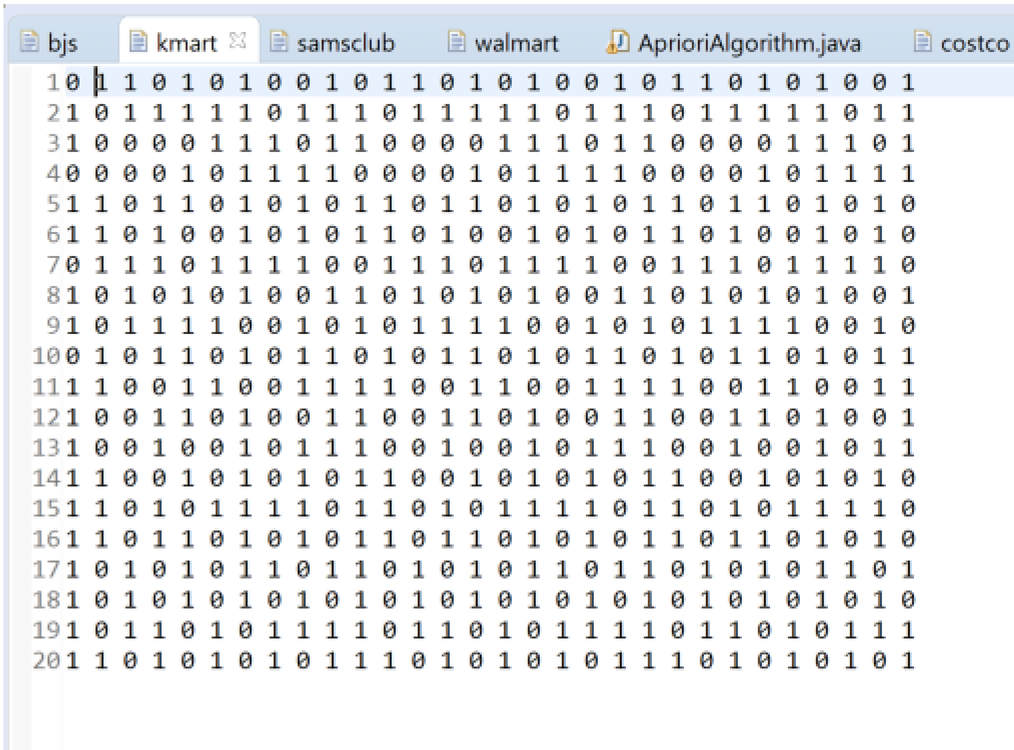
```
0 1 1 1 0 0 1 0 0 1 0 1 1 1 0 0 1 0 0 1 0 1 1 1 0 0 1 0 0 1
1 0 1 0 1 1 0 0 1 1 1 0 1 0 1 1 0 0 1 1 1 0 1 0 1 1 0 0 1 1
0 0 0 1 1 0 1 1 0 1 0 0 0 1 1 0 1 1 0 1 0 0 0 1 1 0 1 1 0 1
1 0 0 1 1 0 1 1 1 1 1 0 0 1 1 0 1 1 1 1 1 0 0 1 1 0 1 1 1 1
1 1 0 1 1 0 1 0 1 0 1 1 0 1 1 0 1 0 1 0 1 1 0 1 1 0 1 0 1 0
1 1 0 0 0 1 0 0 1 0 1 1 0 0 0 1 0 0 1 0 1 1 0 0 0 1 0 0 1 0
0 1 1 1 0 1 0 0 1 0 0 1 1 1 0 1 0 0 1 0 0 1 1 1 0 1 0 0 1 0
1 0 1 0 1 0 0 1 0 1 1 0 1 0 1 0 0 1 0 1 1 0 1 0 1 0 0 1 0 1
1 0 1 1 1 0 0 0 1 0 1 0 1 1 1 0 0 0 1 0 1 0 1 1 1 0 0 0 1 0
0 1 1 1 1 0 1 0 0 1 0 1 1 1 1 0 1 0 0 1 0 1 1 1 1 0 1 0 0 1
1 0 1 0 1 1 0 0 1 1 1 0 1 0 1 1 0 0 1 1 1 0 1 0 1 1 0 0 1 1
1 0 0 0 1 0 1 0 0 1 1 0 0 0 1 0 1 0 0 1 1 0 0 0 1 0 1 0 0 1
1 0 0 1 0 0 1 1 1 1 1 0 0 1 0 0 1 1 1 1 1 0 0 1 0 0 1 1 1 1
1 1 0 0 1 0 1 0 1 0 1 1 0 0 1 0 1 0 1 0 1 1 0 0 1 0 1 0 1 0
1 1 0 0 0 1 1 1 1 0 1 1 0 0 0 1 1 1 1 0 1 1 0 0 0 1 1 1 1 0
1 1 0 1 1 0 0 0 1 0 1 1 0 1 1 0 0 0 1 0 1 1 0 1 1 0 0 0 1 0
1 0 1 0 1 0 1 1 0 1 1 0 1 0 1 0 1 1 0 1 1 0 1 0 1 0 1 1 0 1
1 0 1 1 1 0 0 0 1 0 1 0 1 1 1 0 0 0 1 0 1 0 1 1 1 0 0 0 1 0
1 0 1 1 0 1 0 1 1 1 1 0 1 1 0 1 0 1 1 1 1 0 1 1 0 1 0 1 1 1
1 0 1 1 0 1 0 1 0 1 1 0 1 1 0 1 0 1 0 1 1 0 1 1 0 1 0 1 0 1
```

samsclub



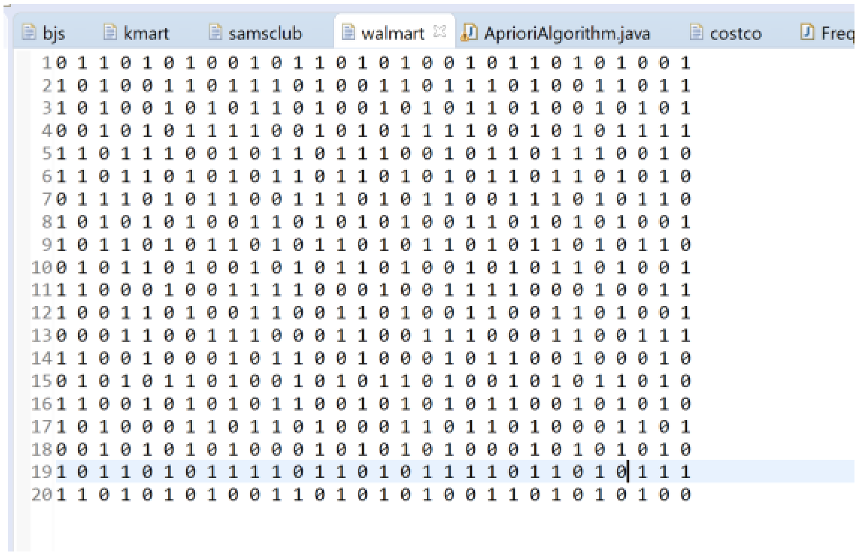
```
0 1 0 0 1 0 1 0 0 1 0 1 0 0 1 0 1 0 0 1 0 1 0 0 1 0 1 0 0 1
1 0 1 1 1 1 0 0 1 1 1 0 1 1 1 1 0 0 1 1 1 0 1 1 1 1 0 0 1 1
0 0 0 1 0 1 1 1 0 1 0 0 0 1 0 1 1 1 0 1 0 0 0 1 0 1 1 1 0 1
1 0 0 0 1 0 1 1 1 1 1 0 0 0 1 0 1 1 1 1 1 0 0 0 1 0 1 1 1 1
1 1 0 1 1 1 1 0 1 0 1 1 0 1 1 1 1 0 1 0 1 1 0 1 1 1 1 0 1 0
1 1 0 1 1 0 1 0 1 0 1 1 0 1 1 0 1 0 1 0 1 1 0 1 1 0 1 0 1 0
0 1 1 1 0 1 1 0 1 0 0 1 1 1 0 1 1 0 1 0 0 1 1 1 0 1 1 0 1 0
1 0 1 0 0 0 1 1 0 1 1 0 1 0 0 0 1 1 0 1 1 0 1 0 0 0 1 1 0 1
1 0 1 1 1 0 0 0 1 0 1 0 1 1 1 0 0 0 1 0 1 0 1 1 1 0 0 0 1 0
0 1 1 0 1 0 1 0 0 1 0 1 1 0 1 0 1 0 0 1 0 1 1 0 1 0 1 0 0 1
1 0 0 0 1 1 0 0 1 1 1 0 0 0 1 1 0 0 1 1 1 0 0 0 1 1 0 0 1 1
1 0 0 0 1 1 1 0 0 1 1 0 0 0 1 1 1 0 0 1 1 0 0 0 1 1 1 0 0 1
1 0 0 1 0 0 1 0 1 1 1 0 0 1 0 0 1 0 1 1 1 0 0 1 0 0 1 0 1 1
1 1 0 0 1 1 1 0 1 0 1 1 0 0 1 1 1 0 1 0 1 1 0 0 1 1 1 0 1 0
1 1 0 1 0 1 1 1 1 0 1 1 0 1 0 1 1 1 1 0 1 1 0 1 0 1 1 1 1 0
1 1 0 1 1 0 1 0 1 0 1 1 0 1 1 0 1 0 1 0 1 1 0 1 1 0 1 0 1 0
1 0 1 0 0 0 1 1 0 1 1 0 1 0 0 0 1 1 0 1 1 0 1 0 0 0 1 1 0 1
1 0 1 0 1 0 0 0 1 0 1 0 1 0 1 0 0 0 1 0 1 0 1 0 1 0 0 0 1 0
1 0 1 1 0 1 0 1 1 1 1 0 1 1 0 1 0 1 1 1 1 0 1 1 0 1 0 1 1 1
1 0 0 1 0 1 0 1 0 1 1 0 0 1 0 1 0 1 0 1 1 0 0 1 0 1 0 1 0 1
```

kmart



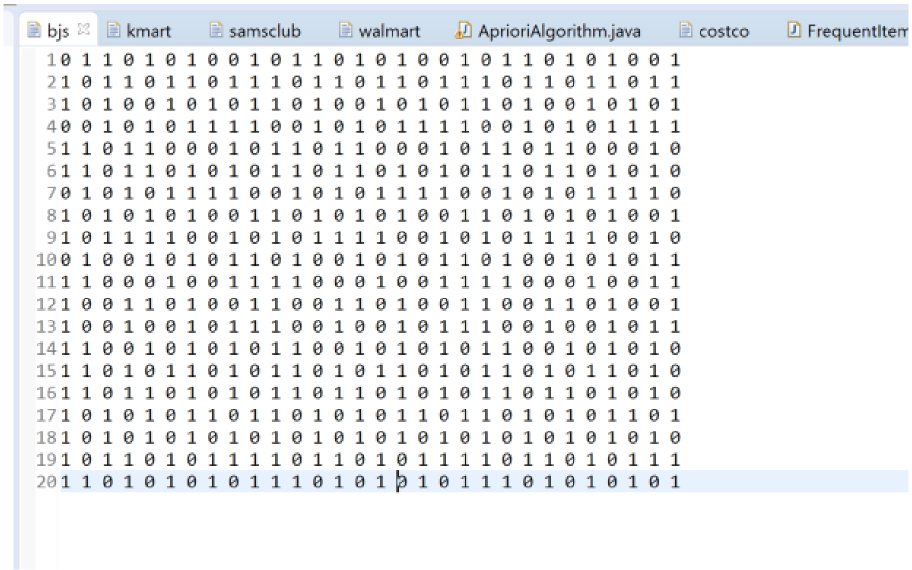
```
0 1 1 0 1 0 1 0 0 1 0 1 1 0 1 0 1 0 0 1 0 1 1 0 1 0 1 0 0 1
1 0 1 1 1 1 1 0 1 1 1 0 1 1 1 1 1 0 1 1 1 0 1 1 1 1 1 0 1 1
1 0 0 0 0 1 1 1 0 1 1 0 0 0 0 1 1 1 0 1 1 0 0 0 0 1 1 1 0 1
0 0 0 0 1 0 1 1 1 1 0 0 0 0 1 0 1 1 1 1 0 0 0 0 1 0 1 1 1 1
1 1 0 1 1 0 1 0 1 0 1 1 0 1 1 0 1 0 1 0 1 1 0 1 1 0 1 0 1 0
1 1 0 1 0 0 1 0 1 0 1 1 0 1 0 0 1 0 1 0 1 1 0 1 0 0 1 0 1 0
0 1 1 1 0 1 1 1 1 0 0 1 1 1 0 1 1 1 1 0 0 1 1 1 0 1 1 1 1 0
1 0 1 0 1 0 1 0 0 1 1 0 1 0 1 0 1 0 0 1 1 0 1 0 1 0 1 0 0 1
1 0 1 1 1 1 0 0 1 0 1 0 1 1 1 1 0 0 1 0 1 0 1 1 1 1 0 0 1 0
0 1 0 1 1 0 1 0 1 1 0 1 0 1 1 0 1 0 1 1 0 1 0 1 1 0 1 0 1 1
1 1 0 0 1 1 0 0 1 1 1 1 0 0 1 1 0 0 1 1 1 1 0 0 1 1 0 0 1 1
1 0 0 1 1 0 1 0 0 1 1 0 0 1 1 0 1 0 0 1 1 0 0 1 1 0 1 0 0 1
1 0 0 1 0 0 1 0 1 1 1 0 0 1 0 0 1 0 1 1 1 0 0 1 0 0 1 0 1 1
1 1 0 0 1 0 1 0 1 0 1 1 0 0 1 0 1 0 1 0 1 1 0 0 1 0 1 0 1 0
1 1 0 1 0 1 1 1 1 0 1 1 0 1 0 1 1 1 1 0 1 1 0 1 0 1 1 1 1 0
1 1 0 1 1 0 1 0 1 0 1 1 0 1 1 0 1 0 1 0 1 1 0 1 1 0 1 0 1 0
1 0 1 0 1 0 1 1 0 1 1 0 1 0 1 0 1 1 0 1 1 0 1 0 1 0 1 1 0 1
1 0 1 0 1 0 1 0 1 0 1 0 1 0 1 0 1 0 1 0 1 0 1 0 1 0 1 0 1 0
1 0 1 1 0 1 0 1 1 1 1 0 1 1 0 1 0 1 1 1 1 0 1 1 0 1 0 1 1 1
1 1 0 1 0 1 0 1 0 1 1 1 0 1 0 1 0 1 0 1 1 1 0 1 0 1 0 1 0 1
```

walmart



```
0 1 1 0 1 0 1 0 0 1 0 1 1 0 1 0 1 0 0 1 0 1 1 0 1 0 1 0 0 1
1 0 1 0 0 1 1 0 1 1 1 0 1 0 0 1 1 0 1 1 1 0 1 0 0 1 1 0 1 1
1 0 1 0 0 1 0 1 0 1 1 0 1 0 0 1 0 1 0 1 1 0 1 0 0 1 0 1 0 1
0 0 1 0 1 0 1 1 1 1 0 0 1 0 1 0 1 1 1 1 0 0 1 0 1 0 1 1 1 1
1 1 0 1 1 1 0 0 1 0 1 1 0 1 1 1 0 0 1 0 1 1 0 1 1 1 0 0 1 0
1 1 0 1 1 0 1 0 1 0 1 1 0 1 1 0 1 0 1 0 1 1 0 1 1 0 1 0 1 0
0 1 1 1 0 1 0 1 1 0 0 1 1 1 0 1 0 1 1 0 0 1 1 1 0 1 0 1 1 0
1 0 1 0 1 0 1 0 0 1 1 0 1 0 1 0 1 0 0 1 1 0 1 0 1 0 1 0 0 1
1 0 1 1 0 1 0 1 1 0 1 0 1 1 0 1 0 1 1 0 1 0 1 1 0 1 0 1 1 0
0 1 0 1 1 0 1 0 0 1 0 1 0 1 1 0 1 0 0 1 0 1 0 1 1 0 1 0 0 1
1 1 0 0 0 1 0 0 1 1 1 1 0 0 0 1 0 0 1 1 1 1 0 0 0 1 0 0 1 1
1 0 0 1 1 0 1 0 0 1 1 0 0 1 1 0 1 0 0 1 1 0 0 1 1 0 1 0 0 1
0 0 0 1 1 0 0 1 1 1 0 0 0 1 1 0 0 1 1 1 0 0 0 1 1 0 0 1 1 1
1 1 0 0 1 0 0 0 1 0 1 1 0 0 1 0 0 0 1 0 1 1 0 0 1 0 0 0 1 0
0 1 0 1 0 1 1 0 1 0 0 1 0 1 0 1 1 0 1 0 0 1 0 1 0 1 1 0 1 0
1 1 0 0 1 0 1 0 1 0 1 1 0 0 1 0 1 0 1 0 1 1 0 0 1 0 1 0 1 0
1 0 1 0 0 0 1 1 0 1 1 0 1 0 0 0 1 1 0 1 1 0 1 0 0 0 1 1 0 1
0 0 1 0 1 0 1 0 1 0 0 0 1 0 1 0 1 0 1 0 0 0 1 0 1 0 1 0 1 0
1 0 1 1 0 1 0 1 1 1 1 0 1 1 0 1 0 1 1 1 1 0 1 1 0 1 0 1 1 1
1 1 0 1 0 1 0 1 0 0 1 1 0 1 0 1 0 1 0 0 1 1 0 1 0 1 0 1 0 0
```

bjs



```
0 1 1 0 1 0 1 0 0 1 0 1 1 0 1 0 1 0 0 1 0 1 1 0 1 0 1 0 0 1
1 0 1 1 0 1 1 0 1 1 1 0 1 1 0 1 1 0 1 1 1 0 1 1 0 1 1 0 1 1
1 0 1 0 0 1 0 1 0 1 1 0 1 0 0 1 0 1 0 1 1 0 1 0 0 1 0 1 0 1
0 0 1 0 1 0 1 1 1 1 0 0 1 0 1 0 1 1 1 1 0 0 1 0 1 0 1 1 1 1
1 1 0 1 1 0 0 0 1 0 1 1 0 1 1 0 0 0 1 0 1 1 0 1 1 0 0 0 1 0
1 1 0 1 1 0 1 0 1 0 1 1 0 1 1 0 1 0 1 0 1 1 0 1 1 0 1 0 1 0
0 1 0 1 0 1 1 1 1 0 0 1 0 1 0 1 1 1 1 0 0 1 0 1 0 1 1 1 1 0
1 0 1 0 1 0 1 0 0 1 1 0 1 0 1 0 1 0 0 1 1 0 1 0 1 0 1 0 0 1
1 0 1 1 1 1 0 0 1 0 1 0 1 1 1 1 0 0 1 0 1 0 1 1 1 1 0 0 1 0
0 1 0 0 1 0 1 0 1 1 0 1 0 0 1 0 1 0 1 1 0 1 0 0 1 0 1 0 1 1
1 1 0 0 0 1 0 0 1 1 1 1 0 0 0 1 0 0 1 1 1 1 0 0 0 1 0 0 1 1
1 0 0 1 1 0 1 0 0 1 1 0 0 1 1 0 1 0 0 1 1 0 0 1 1 0 1 0 0 1
1 0 0 1 0 0 1 0 1 1 1 0 0 1 0 0 1 0 1 1 1 0 0 1 0 0 1 0 1 1
1 1 0 0 1 0 1 0 1 0 1 1 0 0 1 0 1 0 1 0 1 1 0 0 1 0 1 0 1 0
1 1 0 1 0 1 1 0 1 0 1 1 0 1 0 1 1 0 1 0 1 1 0 1 0 1 1 0 1 0
1 1 0 1 1 0 1 0 1 0 1 1 0 1 1 0 1 0 1 0 1 1 0 1 1 0 1 0 1 0
1 0 1 0 1 0 1 1 0 1 1 0 1 0 1 0 1 1 0 1 1 0 1 0 1 0 1 1 0 1
1 0 1 0 1 0 1 0 1 0 1 0 1 0 1 0 1 0 1 0 1 0 1 0 1 0 1 0 1 0
1 0 1 1 0 1 0 1 1 1 1 0 1 1 0 1 0 1 1 1 1 0 1 1 0 1 0 1 1 1
1 1 0 1 0 1 0 1 0 1 1 1 0 1 0 1 0 1 0 1 1 1 0 1 0 1 0 1 0 1
```

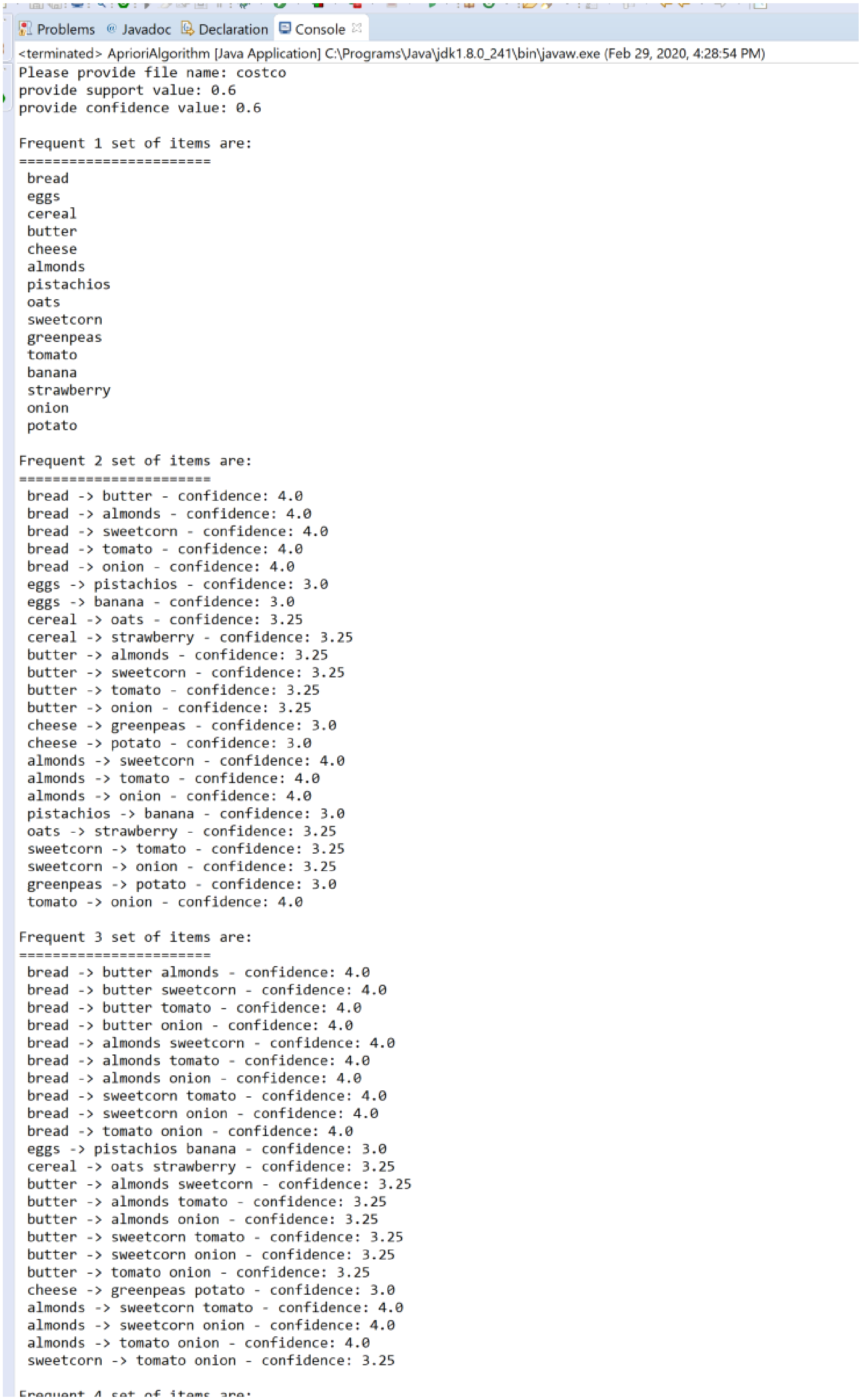
OUTPUT SCREENSHOTS:

Costco (Support value= 0.6, Confidence= 0.6)

Time taken for Apriori algorithm is 7 seconds.

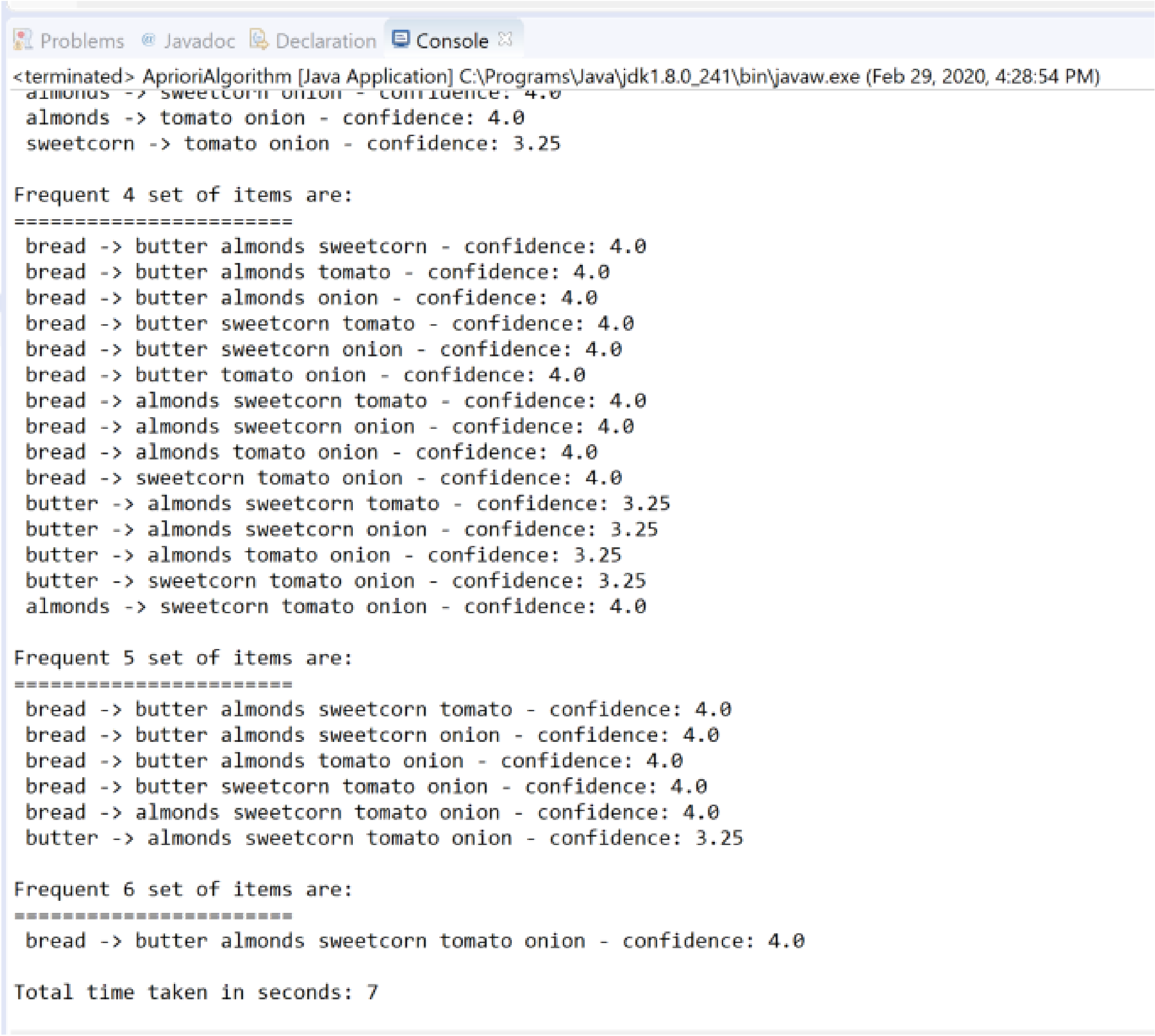
Time taken for brute force is 14 seconds

Apriori algorithm:



<terminated> AprioriAlgorithm [Java Application] C:\Programs\Java\jdk1.8.0_241\bin\javaw.exe (Feb 29, 2020, 4:28:54 PM)

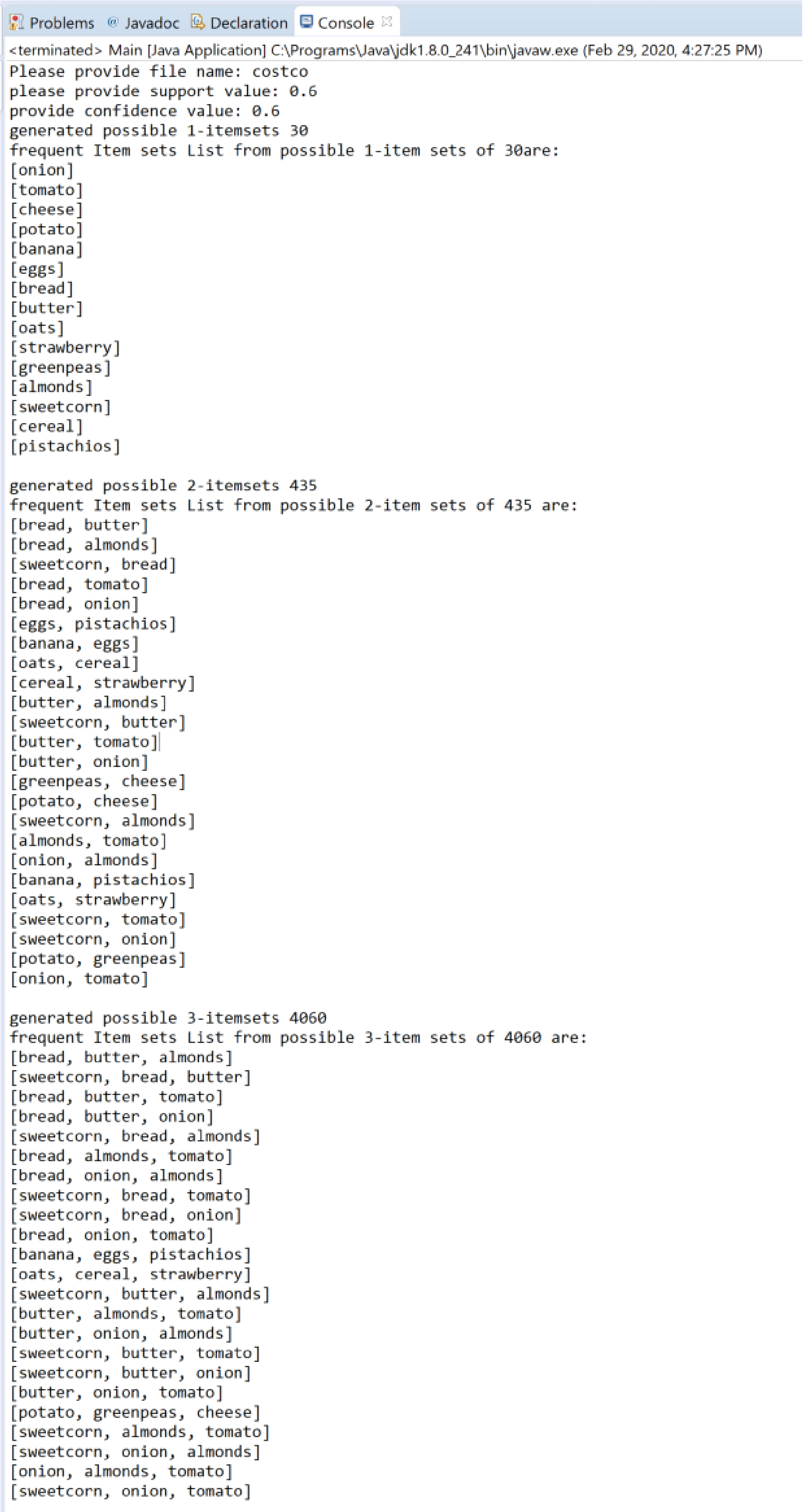
```
Please provide file name: costco
provide support value: 0.6
provide confidence value: 0.6

Frequent 1 set of items are:
=======================
 bread
 eggs
 cereal
 butter
 cheese
 almonds
 pistachios
 oats
 sweetcorn
 greenpeas
 tomato
 banana
 strawberry
 onion
 potato

Frequent 2 set of items are:
=======================
 bread -> butter - confidence: 4.0
 bread -> almonds - confidence: 4.0
 bread -> sweetcorn - confidence: 4.0
 bread -> tomato - confidence: 4.0
 bread -> onion - confidence: 4.0
 eggs -> pistachios - confidence: 3.0
 eggs -> banana - confidence: 3.0
 cereal -> oats - confidence: 3.25
 cereal -> strawberry - confidence: 3.25
 butter -> almonds - confidence: 3.25
 butter -> sweetcorn - confidence: 3.25
 butter -> tomato - confidence: 3.25
 butter -> onion - confidence: 3.25
 cheese -> greenpeas - confidence: 3.0
 cheese -> potato - confidence: 3.0
 almonds -> sweetcorn - confidence: 4.0
 almonds -> tomato - confidence: 4.0
 almonds -> onion - confidence: 4.0
 pistachios -> banana - confidence: 3.0
 oats -> strawberry - confidence: 3.25
 sweetcorn -> tomato - confidence: 3.25
 sweetcorn -> onion - confidence: 3.25
 greenpeas -> potato - confidence: 3.0
 tomato -> onion - confidence: 4.0

Frequent 3 set of items are:
=======================
 bread -> butter almonds - confidence: 4.0
 bread -> butter sweetcorn - confidence: 4.0
 bread -> butter tomato - confidence: 4.0
 bread -> butter onion - confidence: 4.0
 bread -> almonds sweetcorn - confidence: 4.0
 bread -> almonds tomato - confidence: 4.0
 bread -> almonds onion - confidence: 4.0
 bread -> sweetcorn tomato - confidence: 4.0
 bread -> sweetcorn onion - confidence: 4.0
 bread -> tomato onion - confidence: 4.0
 eggs -> pistachios banana - confidence: 3.0
 cereal -> oats strawberry - confidence: 3.25
 butter -> almonds sweetcorn - confidence: 3.25
 butter -> almonds tomato - confidence: 3.25
 butter -> almonds onion - confidence: 3.25
 butter -> sweetcorn tomato - confidence: 3.25
 butter -> sweetcorn onion - confidence: 3.25
 butter -> tomato onion - confidence: 3.25
 cheese -> greenpeas potato - confidence: 3.0
 almonds -> sweetcorn tomato - confidence: 4.0
 almonds -> sweetcorn onion - confidence: 4.0
 almonds -> tomato onion - confidence: 4.0
 sweetcorn -> tomato onion - confidence: 3.25

Frequent 4 set of items are:
```



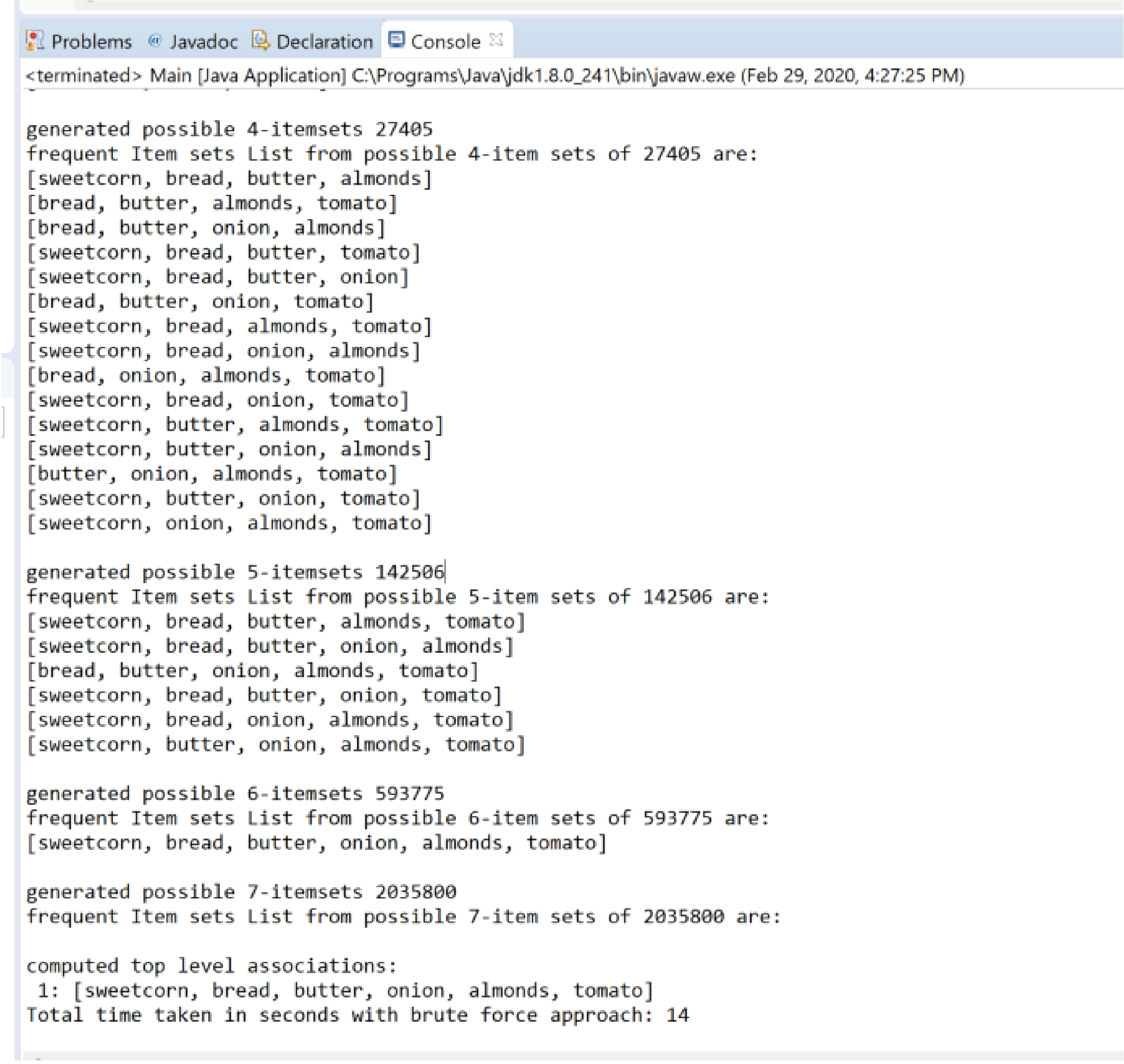
<terminated> AprioriAlgorithm [Java Application] C:\Programs\Java\jdk1.8.0_241\bin\javaw.exe (Feb 29, 2020, 4:28:54 PM)

```
 almonds -> sweetcorn onion - confidence: 4.0
 almonds -> tomato onion - confidence: 4.0
 sweetcorn -> tomato onion - confidence: 3.25

Frequent 4 set of items are:
=======================
 bread -> butter almonds sweetcorn - confidence: 4.0
 bread -> butter almonds tomato - confidence: 4.0
 bread -> butter almonds onion - confidence: 4.0
 bread -> butter sweetcorn tomato - confidence: 4.0
 bread -> butter sweetcorn onion - confidence: 4.0
 bread -> butter tomato onion - confidence: 4.0
 bread -> almonds sweetcorn tomato - confidence: 4.0
 bread -> almonds sweetcorn onion - confidence: 4.0
 bread -> almonds tomato onion - confidence: 4.0
 bread -> sweetcorn tomato onion - confidence: 4.0
 butter -> almonds sweetcorn tomato - confidence: 3.25
 butter -> almonds sweetcorn onion - confidence: 3.25
 butter -> almonds tomato onion - confidence: 3.25
 butter -> sweetcorn tomato onion - confidence: 3.25
 almonds -> sweetcorn tomato onion - confidence: 4.0

Frequent 5 set of items are:
=======================
 bread -> butter almonds sweetcorn tomato - confidence: 4.0
 bread -> butter almonds sweetcorn onion - confidence: 4.0
 bread -> butter almonds tomato onion - confidence: 4.0
 bread -> butter sweetcorn tomato onion - confidence: 4.0
 bread -> almonds sweetcorn tomato onion - confidence: 4.0
 butter -> almonds sweetcorn tomato onion - confidence: 3.25

Frequent 6 set of items are:
=======================
 bread -> butter almonds sweetcorn tomato onion - confidence: 4.0

Total time taken in seconds: 7
```

Brute force approach:



```
<terminated> Main [Java Application] C:\Programs\Java\jdk1.8.0_241\bin\javaw.exe (Feb 29, 2020, 4:27:25 PM)
Please provide file name: costco
please provide support value: 0.6
provide confidence value: 0.6
generated possible 1-itemsets 30
frequent Item sets List from possible 1-item sets of 30are:
[onion]
[tomato]
[cheese]
[potato]
[banana]
[eggs]
[bread]
[butter]
[oats]
[strawberry]
[greenpeas]
[almonds]
[sweetcorn]
[cereal]
[pistachios]

generated possible 2-itemsets 435
frequent Item sets List from possible 2-item sets of 435 are:
[bread, butter]
[bread, almonds]
[sweetcorn, bread]
[bread, tomato]
[bread, onion]
[eggs, pistachios]
[banana, eggs]
[oats, cereal]
[cereal, strawberry]
[butter, almonds]
[sweetcorn, butter]
[butter, tomato]
[butter, onion]
[greenpeas, cheese]
[potato, cheese]
[sweetcorn, almonds]
[almonds, tomato]
[onion, almonds]
[banana, pistachios]
[oats, strawberry]
[sweetcorn, tomato]
[sweetcorn, onion]
[potato, greenpeas]
[onion, tomato]

generated possible 3-itemsets 4060
frequent Item sets List from possible 3-item sets of 4060 are:
[bread, butter, almonds]
[sweetcorn, bread, butter]
[bread, butter, tomato]
[bread, butter, onion]
[sweetcorn, bread, almonds]
[bread, almonds, tomato]
[bread, onion, almonds]
[sweetcorn, bread, tomato]
[sweetcorn, bread, onion]
[bread, onion, tomato]
[banana, eggs, pistachios]
[oats, cereal, strawberry]
[sweetcorn, butter, almonds]
[butter, almonds, tomato]
[butter, onion, almonds]
[sweetcorn, butter, tomato]
[sweetcorn, butter, onion]
[butter, onion, tomato]
[potato, greenpeas, cheese]
[sweetcorn, almonds, tomato]
[sweetcorn, onion, almonds]
[onion, almonds, tomato]
[sweetcorn, onion, tomato]
```



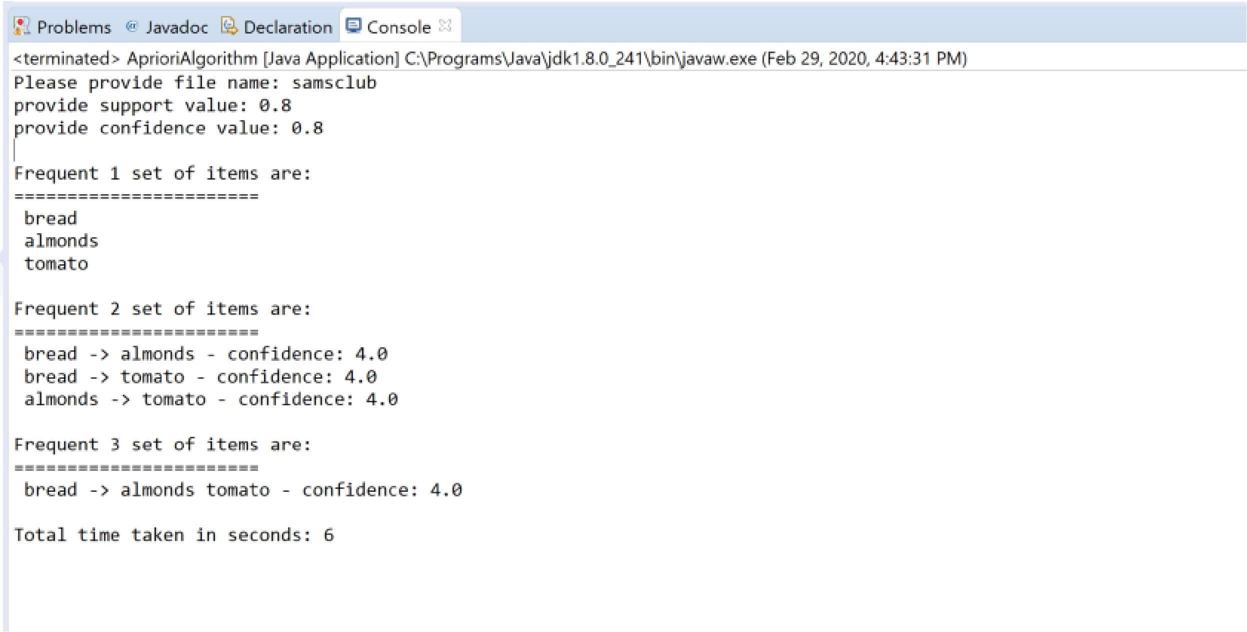
```
<terminated> Main [Java Application] C:\Programs\Java\jdk1.8.0_241\bin\javaw.exe (Feb 29, 2020, 4:27:25 PM)

generated possible 4-itemsets 27405
frequent Item sets List from possible 4-item sets of 27405 are:
[sweetcorn, bread, butter, almonds]
[bread, butter, almonds, tomato]
[bread, butter, onion, almonds]
[sweetcorn, bread, butter, tomato]
[sweetcorn, bread, butter, onion]
[bread, butter, onion, tomato]
[sweetcorn, bread, almonds, tomato]
[sweetcorn, bread, onion, almonds]
[bread, onion, almonds, tomato]
[sweetcorn, bread, onion, tomato]
[sweetcorn, butter, almonds, tomato]
[sweetcorn, butter, onion, almonds]
[butter, onion, almonds, tomato]
[sweetcorn, butter, onion, tomato]
[sweetcorn, onion, almonds, tomato]

generated possible 5-itemsets 142506
frequent Item sets List from possible 5-item sets of 142506 are:
[sweetcorn, bread, butter, almonds, tomato]
[sweetcorn, bread, butter, onion, almonds]
[bread, butter, onion, almonds, tomato]
[sweetcorn, bread, butter, onion, tomato]
[sweetcorn, bread, onion, almonds, tomato]
[sweetcorn, butter, onion, almonds, tomato]

generated possible 6-itemsets 593775
frequent Item sets List from possible 6-item sets of 593775 are:
[sweetcorn, bread, butter, onion, almonds, tomato]

generated possible 7-itemsets 2035800
frequent Item sets List from possible 7-item sets of 2035800 are:

computed top level associations:
 1: [sweetcorn, bread, butter, onion, almonds, tomato]
Total time taken in seconds with brute force approach: 14
```

SamsClub (Support Value= 0.8, Confidence value= 0.8)

Time taken for Apriori algorithm is 6 seconds.

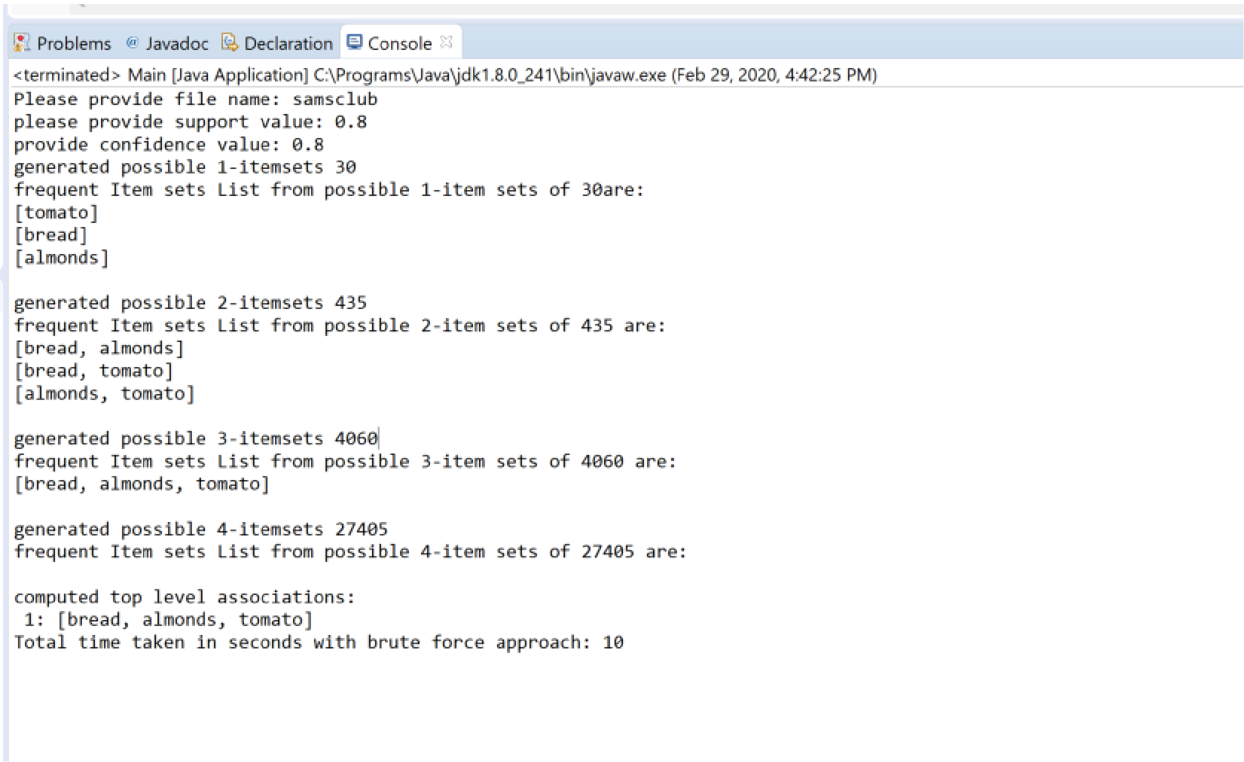
Time taken for brute force is 10 seconds.

Apriori algorithm:



```
<terminated> AprioriAlgorithm [Java Application] C:\Programs\Java\jdk1.8.0_241\bin\javaw.exe (Feb 29, 2020, 4:43:31 PM)
Please provide file name: samsclub
provide support value: 0.8
provide confidence value: 0.8

Frequent 1 set of items are:
=======================
 bread
 almonds
 tomato

Frequent 2 set of items are:
=======================
 bread -> almonds - confidence: 4.0
 bread -> tomato - confidence: 4.0
 almonds -> tomato - confidence: 4.0

Frequent 3 set of items are:
=======================
 bread -> almonds tomato - confidence: 4.0

Total time taken in seconds: 6
```

Brute force approach:



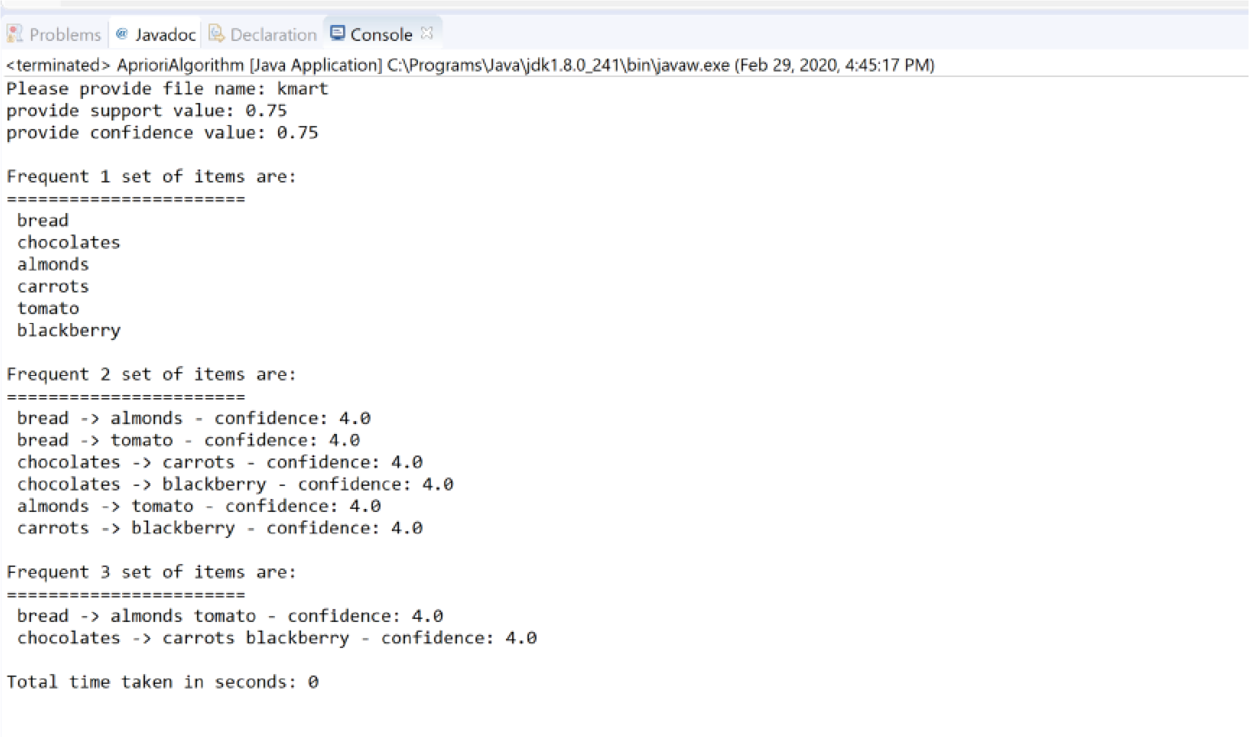
```
<terminated> Main [Java Application] C:\Programs\Java\jdk1.8.0_241\bin\javaw.exe (Feb 29, 2020, 4:42:25 PM)
Please provide file name: samsclub
please provide support value: 0.8
provide confidence value: 0.8
generated possible 1-itemsets 30
frequent Item sets List from possible 1-item sets of 30are:
[tomato]
[bread]
[almonds]

generated possible 2-itemsets 435
frequent Item sets List from possible 2-item sets of 435 are:
[bread, almonds]
[bread, tomato]
[almonds, tomato]

generated possible 3-itemsets 4060
frequent Item sets List from possible 3-item sets of 4060 are:
[bread, almonds, tomato]

generated possible 4-itemsets 27405
frequent Item sets List from possible 4-item sets of 27405 are:

computed top level associations:
 1: [bread, almonds, tomato]
Total time taken in seconds with brute force approach: 10
```

Kmart (Support value= 0.75, Confidence value= 0.75)

Time taken for Apriori algorithm is 0 seconds.

Time taken for brute force is 6 seconds.

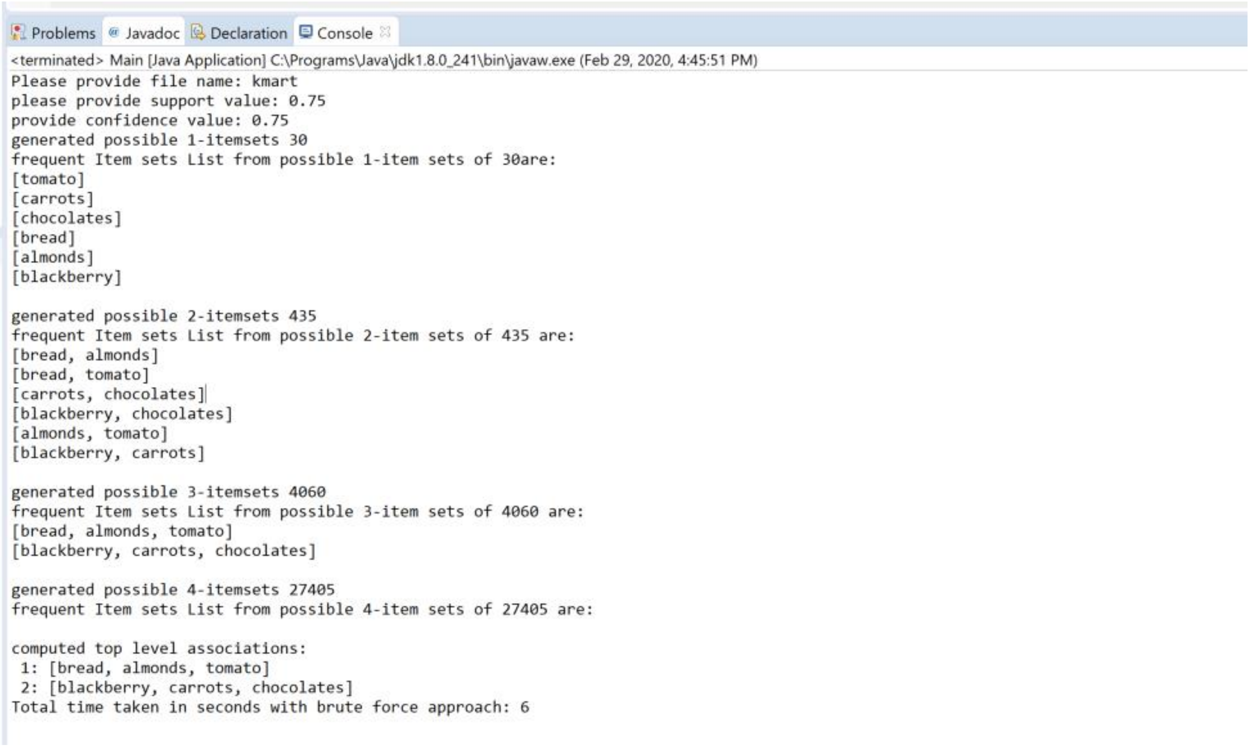
Apriori algorithm:



<terminated> AprioriAlgorithm [Java Application] C:\Programs\Java\jdk1.8.0_241\bin\javaw.exe (Feb 29, 2020, 4:45:17 PM)

```
Please provide file name: kmart
provide support value: 0.75
provide confidence value: 0.75

Frequent 1 set of items are:
=======================
 bread
 chocolates
 almonds
 carrots
 tomato
 blackberry

Frequent 2 set of items are:
=======================
 bread -> almonds - confidence: 4.0
 bread -> tomato - confidence: 4.0
 chocolates -> carrots - confidence: 4.0
 chocolates -> blackberry - confidence: 4.0
 almonds -> tomato - confidence: 4.0
 carrots -> blackberry - confidence: 4.0

Frequent 3 set of items are:
=======================
 bread -> almonds tomato - confidence: 4.0
 chocolates -> carrots blackberry - confidence: 4.0

Total time taken in seconds: 0
```

Brute force approach:



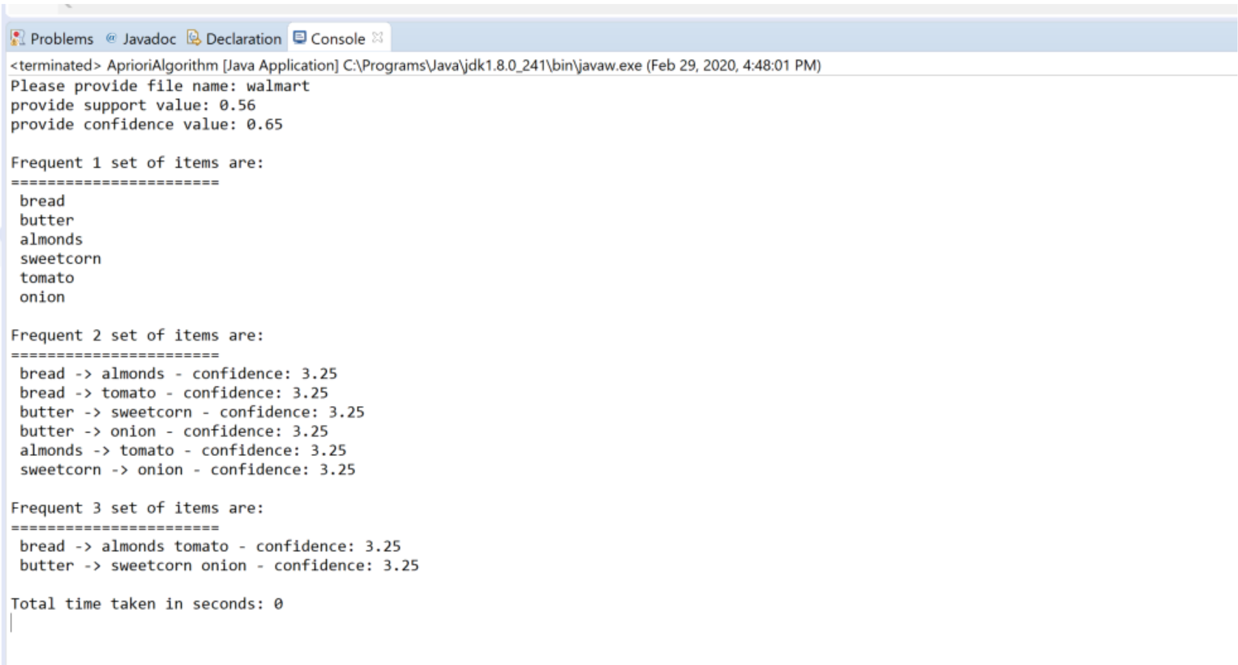
```
<terminated> Main [Java Application] C:\Programs\Java\jdk1.8.0_241\bin\javaw.exe (Feb 29, 2020, 4:45:51 PM)
Please provide file name: kmart
please provide support value: 0.75
provide confidence value: 0.75
generated possible 1-itemsets 30
frequent Item sets List from possible 1-item sets of 30are:
[tomato]
[carrots]
[chocolates]
[bread]
[almonds]
[blackberry]

generated possible 2-itemsets 435
frequent Item sets List from possible 2-item sets of 435 are:
[bread, almonds]
[bread, tomato]
[carrots, chocolates]
[blackberry, chocolates]
[almonds, tomato]
[blackberry, carrots]

generated possible 3-itemsets 4060
frequent Item sets List from possible 3-item sets of 4060 are:
[bread, almonds, tomato]
[blackberry, carrots, chocolates]

generated possible 4-itemsets 27405
frequent Item sets List from possible 4-item sets of 27405 are:

computed top level associations:
 1: [bread, almonds, tomato]
 2: [blackberry, carrots, chocolates]
Total time taken in seconds with brute force approach: 6
```

Walmart (Support value= 0.56, Confidence value= 0.65)

Time taken for Apriori algorithm is 0 seconds.

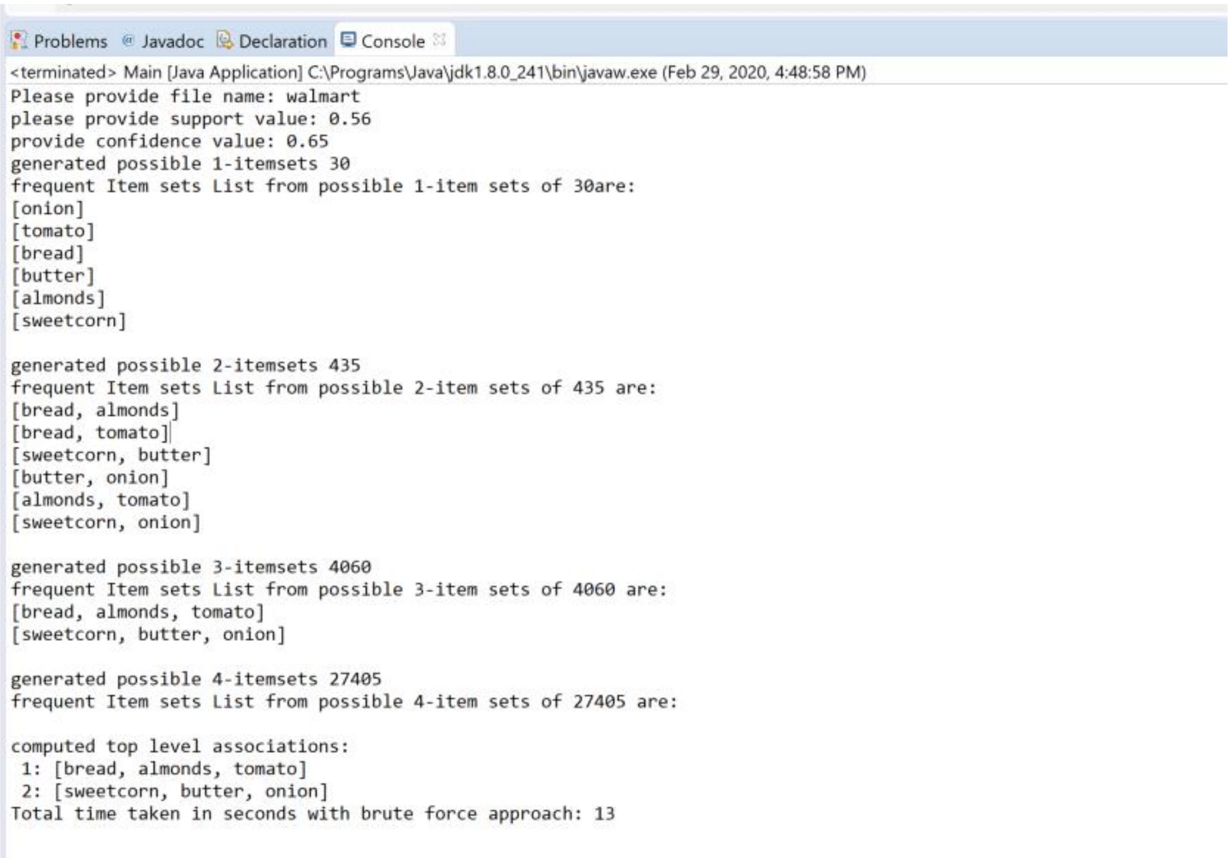
Time taken for brute force is 13 seconds.

Apriori algorithm:



```
<terminated> AprioriAlgorithm [Java Application] C:\Programs\Java\jdk1.8.0_241\bin\javaw.exe (Feb 29, 2020, 4:48:01 PM)
Please provide file name: walmart
provide support value: 0.56
provide confidence value: 0.65

Frequent 1 set of items are:
=======================
 bread
 butter
 almonds
 sweetcorn
 tomato
 onion

Frequent 2 set of items are:
=======================
 bread -> almonds - confidence: 3.25
 bread -> tomato - confidence: 3.25
 butter -> sweetcorn - confidence: 3.25
 butter -> onion - confidence: 3.25
 almonds -> tomato - confidence: 3.25
 sweetcorn -> onion - confidence: 3.25

Frequent 3 set of items are:
=======================
 bread -> almonds tomato - confidence: 3.25
 butter -> sweetcorn onion - confidence: 3.25

Total time taken in seconds: 0
```

Brute force approach:



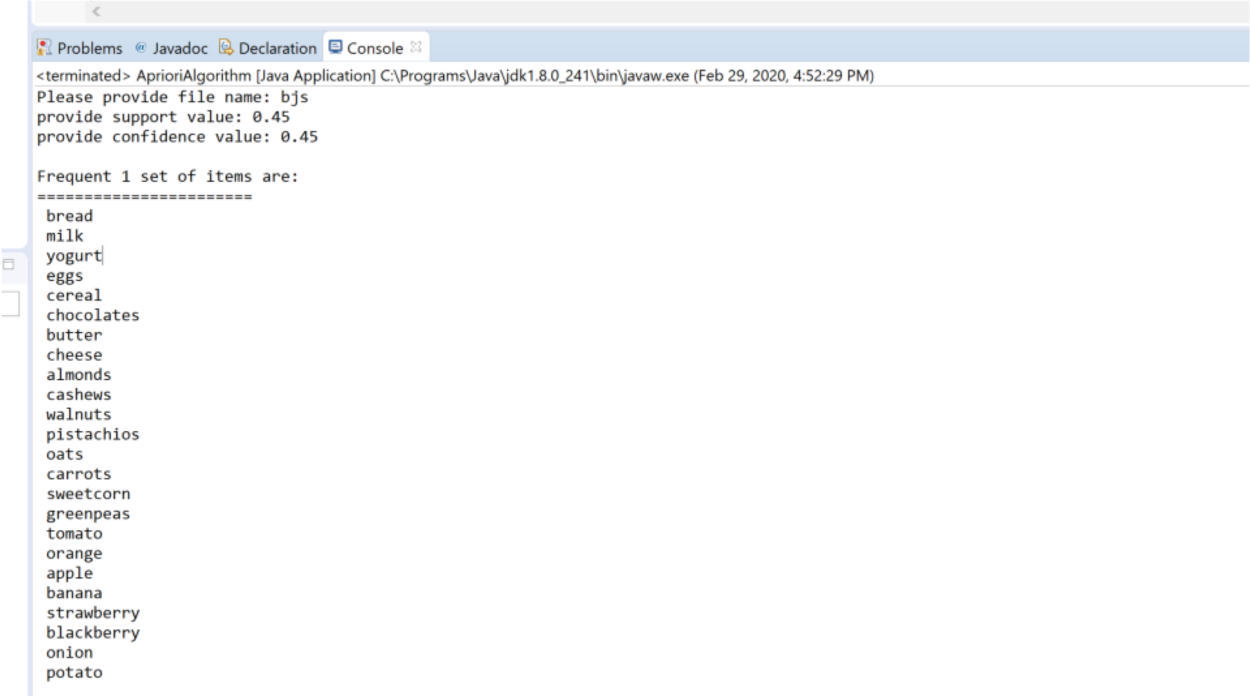
```
<terminated> Main [Java Application] C:\Programs\Java\jdk1.8.0_241\bin\javaw.exe (Feb 29, 2020, 4:48:58 PM)
Please provide file name: walmart
please provide support value: 0.56
provide confidence value: 0.65
generated possible 1-itemsets 30
frequent Item sets List from possible 1-item sets of 30are:
[onion]
[tomato]
[bread]
[butter]
[almonds]
[sweetcorn]

generated possible 2-itemsets 435
frequent Item sets List from possible 2-item sets of 435 are:
[bread, almonds]
[bread, tomato]
[sweetcorn, butter]
[butter, onion]
[almonds, tomato]
[sweetcorn, onion]

generated possible 3-itemsets 4060
frequent Item sets List from possible 3-item sets of 4060 are:
[bread, almonds, tomato]
[sweetcorn, butter, onion]

generated possible 4-itemsets 27405
frequent Item sets List from possible 4-item sets of 27405 are:

computed top level associations:
 1: [bread, almonds, tomato]
 2: [sweetcorn, butter, onion]
Total time taken in seconds with brute force approach: 13
```

Bjs.txt (Support value= 0.45, Confidence value= 0.45)

Time taken for Apriori algorithm is 0 seconds.

Time taken for brute force is 14 seconds.

Apriori algorithm:



```
<terminated> AprioriAlgorithm [Java Application] C:\Programs\Java\jdk1.8.0_241\bin\javaw.exe (Feb 29, 2020, 4:52:29 PM)
Please provide file name: bjs
provide support value: 0.45
provide confidence value: 0.45

Frequent 1 set of items are:
=======================
 bread
 milk
 yogurt
 eggs
 cereal
 chocolates
 butter
 cheese
 almonds
 cashews
 walnuts
 pistachios
 oats
 carrots
 sweetcorn
 greenpeas
 tomato
 orange
 apple
 banana
 strawberry
 blackberry
 onion
 potato
```

```
Please provide file name: bjs
provide support value: 0.45
provide confidence value: 0.45

Frequent 1 set of items are:
=======================
 bread
 milk
 yogurt
 eggs
 cereal
 chocolates
 butter
 cheese
 almonds
 cashews
 walnuts
 pistachios
 oats
 carrots
 sweetcorn
 greenpeas
 tomato
 orange
 apple
 banana
 strawberry
 blackberry
 onion
 potato

Frequent 2 set of items are:
======================
 bread -> eggs - confidence: 4.0
 bread -> cereal - confidence: 4.0
 bread -> chocolates - confidence: 4.0
 bread -> butter - confidence: 4.0
 bread -> cheese - confidence: 4.0
 bread -> almonds - confidence: 4.0
 bread -> pistachios - confidence: 4.0
 bread -> oats - confidence: 4.0
 bread -> carrots - confidence: 4.0
 bread -> sweetcorn - confidence: 4.0
 bread -> greenpeas - confidence: 4.0
 bread -> tomato - confidence: 4.0
 bread -> banana - confidence: 4.0
 bread -> strawberry - confidence: 4.0
 bread -> blackberry - confidence: 4.0
 bread -> onion - confidence: 4.0
 bread -> potato - confidence: 4.0
 milk -> cashews - confidence: 2.5
 milk -> orange - confidence: 2.5
 yogurt -> walnuts - confidence: 2.25
 yogurt -> apple - confidence: 2.25
 eggs -> butter - confidence: 2.75
 eggs -> almonds - confidence: 2.75
 eggs -> pistachios - confidence: 2.75
 eggs -> sweetcorn - confidence: 2.75
 eggs -> tomato - confidence: 2.75
 eggs -> banana - confidence: 2.75
 eggs -> onion - confidence: 2.75
 cereal -> chocolates - confidence: 3.0
 cereal -> almonds - confidence: 3.0
 cereal -> oats - confidence: 3.0
 cereal -> carrots - confidence: 3.0
 cereal -> tomato - confidence: 3.0
 cereal -> strawberry - confidence: 3.0
 cereal -> blackberry - confidence: 3.0
 chocolates -> butter - confidence: 3.5
 chocolates -> almonds - confidence: 3.5
 chocolates -> oats - confidence: 3.5
 chocolates -> carrots - confidence: 3.5
 chocolates -> sweetcorn - confidence: 3.5
 chocolates -> tomato - confidence: 3.5
 chocolates -> strawberry - confidence: 3.5
 chocolates -> blackberry - confidence: 3.5
 chocolates -> onion - confidence: 3.5
 butter -> almonds - confidence: 3.5
 butter -> pistachios - confidence: 3.5
 butter -> carrots - confidence: 3.5
 butter -> sweetcorn - confidence: 3.5
 butter -> tomato - confidence: 3.5
 butter -> banana - confidence: 3.5
 butter -> blackberry - confidence: 3.5
 butter -> onion - confidence: 3.5
 cheese -> almonds - confidence: 3.0
 cheese -> greenpeas - confidence: 3.0
 cheese -> tomato - confidence: 3.0
 cheese -> potato - confidence: 3.0
 almonds -> pistachios - confidence: 4.0
 almonds -> oats - confidence: 4.0
 almonds -> carrots - confidence: 4.0
 almonds -> sweetcorn - confidence: 4.0
 almonds -> greenpeas - confidence: 4.0
 almonds -> tomato - confidence: 4.0
```

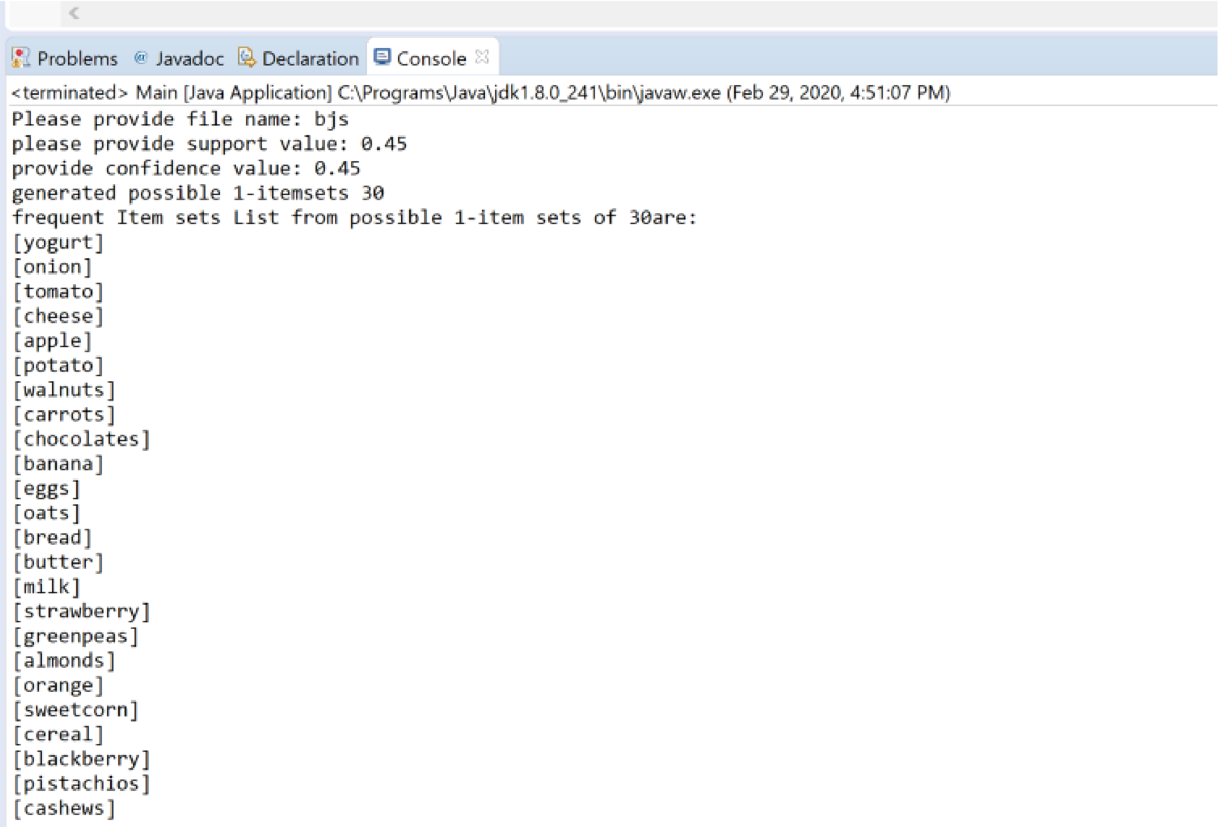
```
 almonds -> banana - confidence: 4.0
 almonds -> strawberry - confidence: 4.0
 almonds -> blackberry - confidence: 4.0
 almonds -> onion - confidence: 4.0
 almonds -> potato - confidence: 4.0
 cashews -> orange - confidence: 2.5
 walnuts -> apple - confidence: 2.25
 pistachios -> sweetcorn - confidence: 2.75
 pistachios -> tomato - confidence: 2.75
 pistachios -> banana - confidence: 2.75
 pistachios -> onion - confidence: 2.75
 oats -> carrots - confidence: 3.0
 oats -> tomato - confidence: 3.0
 oats -> strawberry - confidence: 3.0
 oats -> blackberry - confidence: 3.0
 carrots -> sweetcorn - confidence: 3.5
 carrots -> tomato - confidence: 3.5
 carrots -> strawberry - confidence: 3.5
 carrots -> blackberry - confidence: 3.5
 carrots -> onion - confidence: 3.5
 sweetcorn -> tomato - confidence: 3.5
 sweetcorn -> banana - confidence: 3.5
 sweetcorn -> blackberry - confidence: 3.5
 sweetcorn -> onion - confidence: 3.5
 greenpeas -> tomato - confidence: 3.0
 greenpeas -> potato - confidence: 3.0
 tomato -> banana - confidence: 4.0
 tomato -> strawberry - confidence: 4.0
 tomato -> blackberry - confidence: 4.0
 tomato -> onion - confidence: 4.0
 tomato -> potato - confidence: 4.0
 banana -> onion - confidence: 2.75
 strawberry -> blackberry - confidence: 3.0
 blackberry -> onion - confidence: 3.5

Frequent 3 set of items are:
======================
 bread -> eggs almonds - confidence: 4.0
 bread -> eggs pistachios - confidence: 4.0
 bread -> eggs tomato - confidence: 4.0
 bread -> eggs banana - confidence: 4.0
 bread -> cereal almonds - confidence: 4.0
 bread -> cereal oats - confidence: 4.0
 bread -> cereal tomato - confidence: 4.0
 bread -> cereal strawberry - confidence: 4.0
 bread -> chocolates almonds - confidence: 4.0
 bread -> chocolates carrots - confidence: 4.0
 bread -> chocolates tomato - confidence: 4.0
 bread -> chocolates blackberry - confidence: 4.0
 bread -> butter almonds - confidence: 4.0
 bread -> butter sweetcorn - confidence: 4.0
 bread -> butter tomato - confidence: 4.0
 bread -> butter onion - confidence: 4.0
 bread -> cheese almonds - confidence: 4.0
 bread -> cheese greenpeas - confidence: 4.0
 bread -> cheese tomato - confidence: 4.0
 bread -> cheese potato - confidence: 4.0
 bread -> almonds pistachios - confidence: 4.0
 bread -> almonds oats - confidence: 4.0
 bread -> almonds carrots - confidence: 4.0
 bread -> almonds sweetcorn - confidence: 4.0
 bread -> almonds greenpeas - confidence: 4.0
 bread -> almonds tomato - confidence: 4.0
 bread -> almonds banana - confidence: 4.0
 bread -> almonds strawberry - confidence: 4.0
 bread -> almonds blackberry - confidence: 4.0
 bread -> almonds onion - confidence: 4.0
 bread -> almonds potato - confidence: 4.0
 bread -> pistachios tomato - confidence: 4.0
 bread -> pistachios banana - confidence: 4.0
 bread -> oats tomato - confidence: 4.0
 bread -> oats strawberry - confidence: 4.0
 bread -> carrots tomato - confidence: 4.0
 bread -> carrots blackberry - confidence: 4.0
 bread -> sweetcorn tomato - confidence: 4.0
 bread -> sweetcorn onion - confidence: 4.0
 bread -> greenpeas tomato - confidence: 4.0
 bread -> greenpeas potato - confidence: 4.0
 bread -> tomato banana - confidence: 4.0
 bread -> tomato strawberry - confidence: 4.0
 bread -> tomato blackberry - confidence: 4.0
 bread -> tomato onion - confidence: 4.0
 bread -> tomato potato - confidence: 4.0
 milk -> cashews orange - confidence: 2.5
 yogurt -> walnuts apple - confidence: 2.25
 eggs -> butter pistachios - confidence: 2.75
 eggs -> butter sweetcorn - confidence: 2.75
 eggs -> butter banana - confidence: 2.75
 eggs -> butter onion - confidence: 2.75
 eggs -> almonds pistachios - confidence: 2.75
 eggs -> almonds tomato - confidence: 2.75
 eggs -> almonds banana - confidence: 2.75
 eggs -> pistachios sweetcorn - confidence: 2.75
 eggs -> pistachios tomato - confidence: 2.75
 eggs -> pistachios banana - confidence: 2.75
 eggs -> pistachios onion - confidence: 2.75
 eggs -> sweetcorn banana - confidence: 2.75
 eggs -> sweetcorn onion - confidence: 2.75
 eggs -> tomato banana - confidence: 2.75
 eggs -> banana onion - confidence: 2.75
 cereal -> chocolates oats - confidence: 3.0
 cereal -> chocolates carrots - confidence: 3.0
 cereal -> chocolates strawberry - confidence: 3.0
 cereal -> chocolates blackberry - confidence: 3.0
 cereal -> almonds oats - confidence: 3.0
 cereal -> almonds tomato - confidence: 3.0
 cereal -> almonds strawberry - confidence: 3.0
 cereal -> oats carrots - confidence: 3.0
 cereal -> oats tomato - confidence: 3.0
 cereal -> oats strawberry - confidence: 3.0
 cereal -> oats blackberry - confidence: 3.0
 cereal -> carrots strawberry - confidence: 3.0
 cereal -> carrots blackberry - confidence: 3.0
 cereal -> tomato strawberry - confidence: 3.0
 cereal -> strawberry blackberry - confidence: 3.0
 chocolates -> butter carrots - confidence: 3.5
 chocolates -> butter sweetcorn - confidence: 3.5
 chocolates -> butter blackberry - confidence: 3.5
 chocolates -> butter onion - confidence: 3.5
 chocolates -> almonds carrots - confidence: 3.5
 chocolates -> almonds tomato - confidence: 3.5
 chocolates -> almonds blackberry - confidence: 3.5
 chocolates -> oats carrots - confidence: 3.5
 chocolates -> oats strawberry - confidence: 3.5
 chocolates -> oats blackberry - confidence: 3.5
 chocolates -> carrots sweetcorn - confidence: 3.5
 chocolates -> carrots tomato - confidence: 3.5
 chocolates -> carrots strawberry - confidence: 3.5
 chocolates -> carrots blackberry - confidence: 3.5
 chocolates -> carrots onion - confidence: 3.5
 chocolates -> sweetcorn blackberry - confidence: 3.5
 chocolates -> sweetcorn onion - confidence: 3.5
 chocolates -> tomato blackberry - confidence: 3.5
 chocolates -> strawberry blackberry - confidence: 3.5
 chocolates -> blackberry onion - confidence: 3.5
 butter -> almonds sweetcorn - confidence: 3.5
 butter -> almonds tomato - confidence: 3.5
 butter -> almonds onion - confidence: 3.5
 butter -> pistachios sweetcorn - confidence: 3.5
 butter -> pistachios banana - confidence: 3.5
 butter -> pistachios onion - confidence: 3.5
 butter -> carrots sweetcorn - confidence: 3.5
 butter -> carrots blackberry - confidence: 3.5
 butter -> carrots onion - confidence: 3.5
 butter -> sweetcorn tomato - confidence: 3.5
 butter -> sweetcorn banana - confidence: 3.5
 butter -> sweetcorn blackberry - confidence: 3.5
 butter -> sweetcorn onion - confidence: 3.5
 butter -> tomato onion - confidence: 3.5
 butter -> banana onion - confidence: 3.5
 butter -> blackberry onion - confidence: 3.5
 cheese -> almonds greenpeas - confidence: 3.0
 cheese -> almonds tomato - confidence: 3.0
 cheese -> almonds potato - confidence: 3.0
 cheese -> greenpeas tomato - confidence: 3.0
 cheese -> greenpeas potato - confidence: 3.0
 cheese -> tomato potato - confidence: 3.0
 almonds -> pistachios tomato - confidence: 4.0
 almonds -> pistachios banana - confidence: 4.0
 almonds -> oats tomato - confidence: 4.0
```

```
 almonds -> oats strawberry - confidence: 4.0
 almonds -> carrots tomato - confidence: 4.0
 almonds -> carrots blackberry - confidence: 4.0
 almonds -> sweetcorn tomato - confidence: 4.0
 almonds -> sweetcorn onion - confidence: 4.0
 almonds -> greenpeas tomato - confidence: 4.0
 almonds -> greenpeas potato - confidence: 4.0
 almonds -> tomato banana - confidence: 4.0
 almonds -> tomato strawberry - confidence: 4.0
 almonds -> tomato blackberry - confidence: 4.0
 almonds -> tomato onion - confidence: 4.0
 almonds -> tomato potato - confidence: 4.0
 pistachios -> sweetcorn banana - confidence: 2.75
 pistachios -> sweetcorn onion - confidence: 2.75
 pistachios -> tomato banana - confidence: 2.75
 pistachios -> banana onion - confidence: 2.75
 oats -> carrots strawberry - confidence: 3.0
 oats -> carrots blackberry - confidence: 3.0
 oats -> tomato strawberry - confidence: 3.0
 oats -> strawberry blackberry - confidence: 3.0
 carrots -> sweetcorn blackberry - confidence: 3.5
 carrots -> sweetcorn onion - confidence: 3.5
 carrots -> tomato blackberry - confidence: 3.5
 carrots -> strawberry blackberry - confidence: 3.5
 carrots -> blackberry onion - confidence: 3.5
 sweetcorn -> tomato onion - confidence: 3.5
 sweetcorn -> banana onion - confidence: 3.5
 sweetcorn -> blackberry onion - confidence: 3.5
 greenpeas -> tomato potato - confidence: 3.0

Frequent 4 set of items are:
======================
 bread -> eggs almonds pistachios - confidence: 4.0
 bread -> eggs almonds tomato - confidence: 4.0
 bread -> eggs almonds banana - confidence: 4.0
 bread -> eggs pistachios tomato - confidence: 4.0
 bread -> eggs pistachios banana - confidence: 4.0
 bread -> eggs tomato banana - confidence: 4.0
 bread -> cereal almonds oats - confidence: 4.0
 bread -> cereal almonds tomato - confidence: 4.0
 bread -> cereal almonds strawberry - confidence: 4.0
 bread -> cereal oats tomato - confidence: 4.0
 bread -> cereal oats strawberry - confidence: 4.0
 bread -> cereal tomato strawberry - confidence: 4.0
 bread -> chocolates almonds carrots - confidence: 4.0
 bread -> chocolates almonds tomato - confidence: 4.0
 bread -> chocolates almonds blackberry - confidence: 4.0
 bread -> chocolates carrots tomato - confidence: 4.0
 bread -> chocolates carrots blackberry - confidence: 4.0
 bread -> chocolates tomato blackberry - confidence: 4.0
 bread -> butter almonds sweetcorn - confidence: 4.0
 bread -> butter almonds tomato - confidence: 4.0
 bread -> butter almonds onion - confidence: 4.0
 bread -> butter sweetcorn tomato - confidence: 4.0
 bread -> butter sweetcorn onion - confidence: 4.0
 bread -> butter tomato onion - confidence: 4.0
 bread -> cheese almonds greenpeas - confidence: 4.0
 bread -> cheese almonds tomato - confidence: 4.0
 bread -> cheese almonds potato - confidence: 4.0
 bread -> cheese greenpeas tomato - confidence: 4.0
 bread -> cheese greenpeas potato - confidence: 4.0
 bread -> cheese tomato potato - confidence: 4.0
 bread -> almonds pistachios tomato - confidence: 4.0
 bread -> almonds pistachios banana - confidence: 4.0
 bread -> almonds oats tomato - confidence: 4.0
 bread -> almonds oats strawberry - confidence: 4.0
 bread -> almonds carrots tomato - confidence: 4.0
 bread -> almonds carrots blackberry - confidence: 4.0
 bread -> almonds sweetcorn tomato - confidence: 4.0
 bread -> almonds sweetcorn onion - confidence: 4.0
 bread -> almonds greenpeas tomato - confidence: 4.0
 bread -> almonds greenpeas potato - confidence: 4.0
 bread -> almonds tomato banana - confidence: 4.0
 bread -> almonds tomato strawberry - confidence: 4.0
 bread -> almonds tomato blackberry - confidence: 4.0
 bread -> almonds tomato onion - confidence: 4.0
 bread -> almonds tomato potato - confidence: 4.0
 bread -> pistachios tomato banana - confidence: 4.0
 bread -> oats tomato strawberry - confidence: 4.0
 bread -> carrots tomato blackberry - confidence: 4.0
 bread -> sweetcorn tomato onion - confidence: 4.0
 bread -> greenpeas tomato potato - confidence: 4.0
 eggs -> butter pistachios sweetcorn - confidence: 2.75
 eggs -> butter pistachios banana - confidence: 2.75
 eggs -> butter pistachios onion - confidence: 2.75
 eggs -> butter sweetcorn banana - confidence: 2.75
 eggs -> butter sweetcorn onion - confidence: 2.75
 eggs -> butter banana onion - confidence: 2.75
 eggs -> almonds pistachios tomato - confidence: 2.75
 eggs -> almonds pistachios banana - confidence: 2.75
 eggs -> almonds tomato banana - confidence: 2.75
 eggs -> pistachios sweetcorn banana - confidence: 2.75
 eggs -> pistachios sweetcorn onion - confidence: 2.75
 eggs -> pistachios tomato banana - confidence: 2.75
 eggs -> pistachios banana onion - confidence: 2.75
 eggs -> sweetcorn banana onion - confidence: 2.75
 cereal -> chocolates oats carrots - confidence: 3.0
 cereal -> chocolates oats strawberry - confidence: 3.0
 cereal -> chocolates oats blackberry - confidence: 3.0
 cereal -> chocolates carrots strawberry - confidence: 3.0
 cereal -> chocolates carrots blackberry - confidence: 3.0
 cereal -> chocolates strawberry blackberry - confidence: 3.0
 cereal -> almonds oats tomato - confidence: 3.0
 cereal -> almonds oats strawberry - confidence: 3.0
 cereal -> almonds tomato strawberry - confidence: 3.0
 cereal -> oats carrots strawberry - confidence: 3.0
 cereal -> oats carrots blackberry - confidence: 3.0
 cereal -> oats tomato strawberry - confidence: 3.0
 cereal -> oats strawberry blackberry - confidence: 3.0
 cereal -> carrots strawberry blackberry - confidence: 3.0
 chocolates -> butter carrots sweetcorn - confidence: 3.5
 chocolates -> butter carrots blackberry - confidence: 3.5
 chocolates -> butter carrots onion - confidence: 3.5
 chocolates -> butter sweetcorn blackberry - confidence: 3.5
 chocolates -> butter sweetcorn onion - confidence: 3.5
 chocolates -> butter blackberry onion - confidence: 3.5
 chocolates -> almonds carrots tomato - confidence: 3.5
 chocolates -> almonds carrots blackberry - confidence: 3.5
 chocolates -> almonds tomato blackberry - confidence: 3.5
 chocolates -> oats carrots strawberry - confidence: 3.5
```

```
 chocolates -> oats carrots blackberry - confidence: 3.5
 chocolates -> oats strawberry blackberry - confidence: 3.5
 chocolates -> carrots sweetcorn blackberry - confidence: 3.5
 chocolates -> carrots sweetcorn onion - confidence: 3.5
 chocolates -> carrots tomato blackberry - confidence: 3.5
 chocolates -> carrots strawberry blackberry - confidence: 3.5
 chocolates -> carrots blackberry onion - confidence: 3.5
 chocolates -> sweetcorn blackberry onion - confidence: 3.5
 butter -> almonds sweetcorn tomato - confidence: 3.5
 butter -> almonds sweetcorn onion - confidence: 3.5
 butter -> almonds tomato onion - confidence: 3.5
 butter -> pistachios sweetcorn banana - confidence: 3.5
 butter -> pistachios sweetcorn onion - confidence: 3.5
 butter -> pistachios banana onion - confidence: 3.5
 butter -> carrots sweetcorn blackberry - confidence: 3.5
 butter -> carrots sweetcorn onion - confidence: 3.5
 butter -> carrots blackberry onion - confidence: 3.5
 butter -> sweetcorn tomato onion - confidence: 3.5
 butter -> sweetcorn banana onion - confidence: 3.5
 butter -> sweetcorn blackberry onion - confidence: 3.5
 cheese -> almonds greenpeas tomato - confidence: 3.0
 cheese -> almonds greenpeas potato - confidence: 3.0
 cheese -> almonds tomato potato - confidence: 3.0
 cheese -> greenpeas tomato potato - confidence: 3.0
 almonds -> pistachios tomato banana - confidence: 4.0
 almonds -> oats tomato strawberry - confidence: 4.0
 almonds -> carrots tomato blackberry - confidence: 4.0
 almonds -> sweetcorn tomato onion - confidence: 4.0
 almonds -> greenpeas tomato potato - confidence: 4.0
 pistachios -> sweetcorn banana onion - confidence: 2.75
 oats -> carrots strawberry blackberry - confidence: 3.0
 carrots -> sweetcorn blackberry onion - confidence: 3.5

Frequent 5 set of items are:
======================
 bread -> eggs almonds pistachios tomato - confidence: 4.0
 bread -> eggs almonds pistachios banana - confidence: 4.0
 bread -> eggs almonds tomato banana - confidence: 4.0
 bread -> eggs pistachios tomato banana - confidence: 4.0
 bread -> cereal almonds oats tomato - confidence: 4.0
 bread -> cereal almonds oats strawberry - confidence: 4.0
 bread -> cereal almonds tomato strawberry - confidence: 4.0
 bread -> cereal oats tomato strawberry - confidence: 4.0
 bread -> chocolates almonds carrots tomato - confidence: 4.0
 bread -> chocolates almonds carrots blackberry - confidence: 4.0
 bread -> chocolates almonds tomato blackberry - confidence: 4.0
 bread -> chocolates carrots tomato blackberry - confidence: 4.0
 bread -> butter almonds sweetcorn tomato - confidence: 4.0
 bread -> butter almonds sweetcorn onion - confidence: 4.0
 bread -> butter almonds tomato onion - confidence: 4.0
 bread -> butter sweetcorn tomato onion - confidence: 4.0
 bread -> cheese almonds greenpeas tomato - confidence: 4.0
 bread -> cheese almonds greenpeas potato - confidence: 4.0
 bread -> cheese almonds tomato potato - confidence: 4.0
 bread -> cheese greenpeas tomato potato - confidence: 4.0
 bread -> almonds pistachios tomato banana - confidence: 4.0
 bread -> almonds oats tomato strawberry - confidence: 4.0
 bread -> almonds carrots tomato blackberry - confidence: 4.0
 bread -> almonds sweetcorn tomato onion - confidence: 4.0
 bread -> almonds greenpeas tomato potato - confidence: 4.0
 eggs -> butter pistachios sweetcorn banana - confidence: 2.75
 eggs -> butter pistachios sweetcorn onion - confidence: 2.75
 eggs -> butter pistachios banana onion - confidence: 2.75
 eggs -> butter sweetcorn banana onion - confidence: 2.75
 eggs -> almonds pistachios tomato banana - confidence: 2.75
 eggs -> pistachios sweetcorn banana onion - confidence: 2.75
 cereal -> chocolates oats carrots strawberry - confidence: 3.0
 cereal -> chocolates oats carrots blackberry - confidence: 3.0
 cereal -> chocolates oats strawberry blackberry - confidence: 3.0
 cereal -> chocolates carrots strawberry blackberry - confidence: 3.0
 cereal -> almonds oats tomato strawberry - confidence: 3.0
 cereal -> oats carrots strawberry blackberry - confidence: 3.0
 chocolates -> butter carrots sweetcorn blackberry - confidence: 3.5
 chocolates -> butter carrots sweetcorn onion - confidence: 3.5
 chocolates -> butter carrots blackberry onion - confidence: 3.5
 chocolates -> butter sweetcorn blackberry onion - confidence: 3.5
 chocolates -> almonds carrots tomato blackberry - confidence: 3.5
 chocolates -> oats carrots strawberry blackberry - confidence: 3.5
 chocolates -> carrots sweetcorn blackberry onion - confidence: 3.5
 butter -> almonds sweetcorn tomato onion - confidence: 3.5
```

```
 butter -> pistachios sweetcorn banana onion - confidence: 3.5
 butter -> carrots sweetcorn blackberry onion - confidence: 3.5
 cheese -> almonds greenpeas tomato potato - confidence: 3.0

Frequent 6 set of items are:
=======================
 bread -> eggs almonds pistachios tomato banana - confidence: 4.0
 bread -> cereal almonds oats tomato strawberry - confidence: 4.0
 bread -> chocolates almonds carrots tomato blackberry - confidence: 4.0
 bread -> butter almonds sweetcorn tomato onion - confidence: 4.0
 bread -> cheese almonds greenpeas tomato potato - confidence: 4.0
 eggs -> butter pistachios sweetcorn banana onion - confidence: 2.75
 cereal -> chocolates oats carrots strawberry blackberry - confidence: 3.0
 chocolates -> butter carrots sweetcorn blackberry onion - confidence: 3.5

Total time taken in seconds: 0
```

Brute force approach:



```
<terminated> Main [Java Application] C:\Programs\Java\jdk1.8.0_241\bin\javaw.exe (Feb 29, 2020, 4:51:07 PM)
Please provide file name: bjs
please provide support value: 0.45
provide confidence value: 0.45
generated possible 1-itemsets 30
frequent Item sets List from possible 1-item sets of 30are:
[yogurt]
[onion]
[tomato]
[cheese]
[apple]
[potato]
[walnuts]
[carrots]
[chocolates]
[banana]
[eggs]
[oats]
[bread]
[butter]
[milk]
[strawberry]
[greenpeas]
[almonds]
[orange]
[sweetcorn]
[cereal]
[blackberry]
[pistachios]
[cashews]
```

```
Please provide file name: bjs
please provide support value: 0.45
provide confidence value: 0.45
generated possible 1-itemsets 30
frequent Item sets List from possible 1-item sets of 30are:
[yogurt]
[onion]
[tomato]
[cheese]
[apple]
[potato]
[walnuts]
[carrots]
[chocolates]
[banana]
[eggs]
[oats]
[bread]
[butter]
[milk]
[strawberry]
[greenpeas]
[almonds]
[orange]
[sweetcorn]
[cereal]
[blackberry]
[pistachios]
[cashews]

generated possible 2-itemsets 435
frequent Item sets List from possible 2-item sets of 435 are:
[eggs, bread]
[bread, cereal]
[bread, chocolates]
[bread, butter]
[bread, cheese]
[bread, almonds]
[bread, pistachios]
[bread, oats]
```

```
[bread, carrots]
[sweetcorn, bread]
[bread, greenpeas]
[bread, tomato]
[banana, bread]
[bread, strawberry]
[bread, blackberry]
[bread, onion]
[bread, potato]
[milk, cashews]
[orange, milk]
[yogurt, walnuts]
[yogurt, apple]
[eggs, butter]
[eggs, almonds]
[eggs, pistachios]
[eggs, sweetcorn]
[eggs, tomato]
[banana, eggs]
[eggs, onion]
[cereal, chocolates]
[cereal, almonds]
[oats, cereal]
[cereal, carrots]
[cereal, tomato]
[cereal, strawberry]
[cereal, blackberry]
[butter, chocolates]
[almonds, chocolates]
[oats, chocolates]
[carrots, chocolates]
[sweetcorn, chocolates]
[tomato, chocolates]
[strawberry, chocolates]
[blackberry, chocolates]
[onion, chocolates]
[butter, almonds]
[butter, pistachios]
[butter, carrots]
[sweetcorn, butter]
```

```
[butter, tomato]
[banana, butter]
[butter, blackberry]
[butter, onion]
[almonds, cheese]
[greenpeas, cheese]
[tomato, cheese]
[potato, cheese]
[pistachios, almonds]
[oats, almonds]
[carrots, almonds]
[sweetcorn, almonds]
[greenpeas, almonds]
[almonds, tomato]
[banana, almonds]
[strawberry, almonds]
[blackberry, almonds]
[onion, almonds]
[potato, almonds]
[orange, cashews]
[apple, walnuts]
[sweetcorn, pistachios]
[pistachios, tomato]
[banana, pistachios]
[pistachios, onion]
[oats, carrots]
[oats, tomato]
[oats, strawberry]
[oats, blackberry]
[sweetcorn, carrots]
[carrots, tomato]
[strawberry, carrots]
[blackberry, carrots]
[onion, carrots]
[sweetcorn, tomato]
[banana, sweetcorn]
[sweetcorn, blackberry]
[sweetcorn, onion]
[greenpeas, tomato]
[potato, greenpeas]
```

```
[banana, tomato]
[strawberry, tomato]
[blackberry, tomato]
[onion, tomato]
[potato, tomato]
[banana, onion]
[blackberry, strawberry]
[blackberry, onion]

generated possible 3-itemsets 4060
frequent Item sets List from possible 3-item sets of 4060 are:
[eggs, bread, almonds]
[eggs, bread, pistachios]
[eggs, bread, tomato]
[banana, eggs, bread]
[bread, cereal, almonds]
[bread, oats, cereal]
[bread, cereal, tomato]
[bread, cereal, strawberry]
[bread, almonds, chocolates]
[bread, carrots, chocolates]
[bread, tomato, chocolates]
[bread, blackberry, chocolates]
[bread, butter, almonds]
[sweetcorn, bread, butter]
[bread, butter, tomato]
[bread, butter, onion]
[bread, almonds, cheese]
[bread, greenpeas, cheese]
[bread, tomato, cheese]
[bread, potato, cheese]
[bread, pistachios, almonds]
[bread, oats, almonds]
[bread, carrots, almonds]
[sweetcorn, bread, almonds]
[bread, greenpeas, almonds]
[bread, almonds, tomato]
[banana, bread, almonds]
[bread, strawberry, almonds]
[bread, blackberry, almonds]
```

```
[bread, onion, almonds]
[bread, potato, almonds]
[bread, pistachios, tomato]
[banana, bread, pistachios]
[bread, oats, tomato]
[bread, oats, strawberry]
[bread, carrots, tomato]
[bread, blackberry, carrots]
[sweetcorn, bread, tomato]
[sweetcorn, bread, onion]
[bread, greenpeas, tomato]
[bread, potato, greenpeas]
[banana, bread, tomato]
[bread, strawberry, tomato]
[bread, blackberry, tomato]
[bread, onion, tomato]
[bread, potato, tomato]
[orange, milk, cashews]
[yogurt, apple, walnuts]
[eggs, butter, pistachios]
[eggs, sweetcorn, butter]
[banana, eggs, butter]
[eggs, butter, onion]
[eggs, pistachios, almonds]
[eggs, almonds, tomato]
[banana, eggs, almonds]
[eggs, sweetcorn, pistachios]
[eggs, pistachios, tomato]
[banana, eggs, pistachios]
[eggs, pistachios, onion]
[banana, eggs, sweetcorn]
[eggs, sweetcorn, onion]
[banana, eggs, tomato]
[banana, eggs, onion]
[oats, cereal, chocolates]
[cereal, carrots, chocolates]
[cereal, strawberry, chocolates]
[cereal, blackberry, chocolates]
[oats, cereal, almonds]
[cereal, almonds, tomato]
```

```
[cereal, strawberry, almonds]
[oats, cereal, carrots]
[oats, cereal, tomato]
[oats, cereal, strawberry]
[oats, cereal, blackberry]
[cereal, strawberry, carrots]
[cereal, blackberry, carrots]
[cereal, strawberry, tomato]
[cereal, blackberry, strawberry]
[butter, carrots, chocolates]
[sweetcorn, butter, chocolates]
[butter, blackberry, chocolates]
[butter, onion, chocolates]
[carrots, almonds, chocolates]
[almonds, tomato, chocolates]
[blackberry, almonds, chocolates]
[oats, carrots, chocolates]
[oats, strawberry, chocolates]
[oats, blackberry, chocolates]
[sweetcorn, carrots, chocolates]
[carrots, tomato, chocolates]
[strawberry, carrots, chocolates]
[blackberry, carrots, chocolates]
[onion, carrots, chocolates]
[sweetcorn, blackberry, chocolates]
[sweetcorn, onion, chocolates]
[blackberry, tomato, chocolates]
[blackberry, strawberry, chocolates]
[blackberry, onion, chocolates]
[sweetcorn, butter, almonds]
[butter, almonds, tomato]
[butter, onion, almonds]
[sweetcorn, butter, pistachios]
[banana, butter, pistachios]
[butter, pistachios, onion]
[sweetcorn, butter, carrots]
[butter, blackberry, carrots]
[butter, onion, carrots]
[sweetcorn, butter, tomato]
[banana, sweetcorn, butter]
```

```
[sweetcorn, butter, blackberry]
[sweetcorn, butter, onion]
[butter, onion, tomato]
[banana, butter, onion]
[butter, blackberry, onion]
[greenpeas, almonds, cheese]
[almonds, tomato, cheese]
[potato, almonds, cheese]
[greenpeas, tomato, cheese]
[potato, greenpeas, cheese]
[potato, tomato, cheese]
[pistachios, almonds, tomato]
[banana, pistachios, almonds]
[oats, almonds, tomato]
[oats, strawberry, almonds]
[carrots, almonds, tomato]
[blackberry, carrots, almonds]
[sweetcorn, almonds, tomato]
[sweetcorn, onion, almonds]
[greenpeas, almonds, tomato]
[potato, greenpeas, almonds]
[banana, almonds, tomato]
[strawberry, almonds, tomato]
[blackberry, almonds, tomato]
[onion, almonds, tomato]
[potato, almonds, tomato]
[banana, sweetcorn, pistachios]
[sweetcorn, pistachios, onion]
[banana, pistachios, tomato]
[banana, pistachios, onion]
[oats, strawberry, carrots]
[oats, blackberry, carrots]
[oats, strawberry, tomato]
[oats, blackberry, strawberry]
[sweetcorn, blackberry, carrots]
[sweetcorn, onion, carrots]
[blackberry, carrots, tomato]
[blackberry, strawberry, carrots]
[blackberry, onion, carrots]
[sweetcorn, onion, tomato]
```

```
[banana, sweetcorn, onion]
[sweetcorn, blackberry, onion]
[potato, greenpeas, tomato]

generated possible 4-itemsets 27405
frequent Item sets List from possible 4-item sets of 27405 are:
[eggs, bread, pistachios, almonds]
[eggs, bread, almonds, tomato]
[banana, eggs, bread, almonds]
[eggs, bread, pistachios, tomato]
[banana, eggs, bread, pistachios]
[banana, eggs, bread, tomato]
[bread, oats, cereal, almonds]
[bread, cereal, almonds, tomato]
[bread, cereal, strawberry, almonds]
[bread, oats, cereal, tomato]
[bread, oats, cereal, strawberry]
[bread, cereal, strawberry, tomato]
[bread, carrots, almonds, chocolates]
[bread, almonds, tomato, chocolates]
[bread, blackberry, almonds, chocolates]
[bread, carrots, tomato, chocolates]
[bread, blackberry, carrots, chocolates]
[bread, blackberry, tomato, chocolates]
[sweetcorn, bread, butter, almonds]
[bread, butter, almonds, tomato]
[bread, butter, onion, almonds]
[sweetcorn, bread, butter, tomato]
[sweetcorn, bread, butter, onion]
[bread, butter, onion, tomato]
[bread, greenpeas, almonds, cheese]
[bread, almonds, tomato, cheese]
[bread, potato, almonds, cheese]
[bread, greenpeas, tomato, cheese]
[bread, potato, greenpeas, cheese]
[bread, potato, tomato, cheese]
[bread, pistachios, almonds, tomato]
[banana, bread, pistachios, almonds]
[bread, oats, almonds, tomato]
[bread, oats, strawberry, almonds]
```

```
[bread, carrots, almonds, tomato]
[bread, blackberry, carrots, almonds]
[sweetcorn, bread, almonds, tomato]
[sweetcorn, bread, onion, almonds]
[bread, greenpeas, almonds, tomato]
[bread, potato, greenpeas, almonds]
[banana, bread, almonds, tomato]
[bread, strawberry, almonds, tomato]
[bread, blackberry, almonds, tomato]
[bread, onion, almonds, tomato]
[bread, potato, almonds, tomato]
[banana, bread, pistachios, tomato]
[bread, oats, strawberry, tomato]
[bread, blackberry, carrots, tomato]
[sweetcorn, bread, onion, tomato]
[bread, potato, greenpeas, tomato]
[eggs, sweetcorn, butter, pistachios]
[banana, eggs, butter, pistachios]
[eggs, butter, pistachios, onion]
[banana, eggs, sweetcorn, butter]
[eggs, sweetcorn, butter, onion]
[banana, eggs, butter, onion]
[eggs, pistachios, almonds, tomato]
[banana, eggs, pistachios, almonds]
[banana, eggs, almonds, tomato]
[banana, eggs, sweetcorn, pistachios]
[eggs, sweetcorn, pistachios, onion]
[banana, eggs, pistachios, tomato]
[banana, eggs, pistachios, onion]
[banana, eggs, sweetcorn, onion]
[oats, cereal, carrots, chocolates]
[oats, cereal, strawberry, chocolates]
[oats, cereal, blackberry, chocolates]
[cereal, strawberry, carrots, chocolates]
[cereal, blackberry, carrots, chocolates]
[cereal, blackberry, strawberry, chocolates]
[oats, cereal, almonds, tomato]
[oats, cereal, strawberry, almonds]
[cereal, strawberry, almonds, tomato]
[oats, cereal, strawberry, carrots]
```

```
[oats, cereal, blackberry, carrots]
[oats, cereal, strawberry, tomato]
[oats, cereal, blackberry, strawberry]
[cereal, blackberry, strawberry, carrots]
[sweetcorn, butter, carrots, chocolates]
[butter, blackberry, carrots, chocolates]
[butter, onion, carrots, chocolates]
[sweetcorn, butter, blackberry, chocolates]
[sweetcorn, butter, onion, chocolates]
[butter, blackberry, onion, chocolates]
[carrots, almonds, tomato, chocolates]
[blackberry, carrots, almonds, chocolates]
[blackberry, almonds, tomato, chocolates]
[oats, strawberry, carrots, chocolates]
[oats, blackberry, carrots, chocolates]
[oats, blackberry, strawberry, chocolates]
[sweetcorn, blackberry, carrots, chocolates]
[sweetcorn, onion, carrots, chocolates]
[blackberry, carrots, tomato, chocolates]
[blackberry, strawberry, carrots, chocolates]
[blackberry, onion, carrots, chocolates]
[sweetcorn, blackberry, onion, chocolates]
[sweetcorn, butter, almonds, tomato]
[sweetcorn, butter, onion, almonds]
[butter, onion, almonds, tomato]
[banana, sweetcorn, butter, pistachios]
[sweetcorn, butter, pistachios, onion]
[banana, butter, pistachios, onion]
[sweetcorn, butter, blackberry, carrots]
[sweetcorn, butter, onion, carrots]
[butter, blackberry, onion, carrots]
[sweetcorn, butter, onion, tomato]
[banana, sweetcorn, butter, onion]
[sweetcorn, butter, blackberry, onion]
[greenpeas, almonds, tomato, cheese]
[potato, greenpeas, almonds, cheese]
[potato, almonds, tomato, cheese]
[potato, greenpeas, tomato, cheese]
[banana, pistachios, almonds, tomato]
[oats, strawberry, almonds, tomato]
```

```
[blackberry, carrots, almonds, tomato]
[sweetcorn, onion, almonds, tomato]
[potato, greenpeas, almonds, tomato]
[banana, sweetcorn, pistachios, onion]
[oats, blackberry, strawberry, carrots]
[sweetcorn, blackberry, onion, carrots]

generated possible 5-itemsets 142506
frequent Item sets List from possible 5-item sets of 142506 are:
[eggs, bread, pistachios, almonds, tomato]
[banana, eggs, bread, pistachios, almonds]
[banana, eggs, bread, almonds, tomato]
[banana, eggs, bread, pistachios, tomato]
[bread, oats, cereal, almonds, tomato]
[bread, oats, cereal, strawberry, almonds]
[bread, cereal, strawberry, almonds, tomato]
[bread, oats, cereal, strawberry, tomato]
[bread, carrots, almonds, tomato, chocolates]
[bread, blackberry, carrots, almonds, chocolates]
[bread, blackberry, almonds, tomato, chocolates]
[bread, blackberry, carrots, tomato, chocolates]
[sweetcorn, bread, butter, almonds, tomato]
[sweetcorn, bread, butter, onion, almonds]
[bread, butter, onion, almonds, tomato]
[sweetcorn, bread, butter, onion, tomato]
[bread, greenpeas, almonds, tomato, cheese]
[bread, potato, greenpeas, almonds, cheese]
[bread, potato, almonds, tomato, cheese]
[bread, potato, greenpeas, tomato, cheese]
[banana, bread, pistachios, almonds, tomato]
[bread, oats, strawberry, almonds, tomato]
[bread, blackberry, carrots, almonds, tomato]
[sweetcorn, bread, onion, almonds, tomato]
[bread, potato, greenpeas, almonds, tomato]
[banana, eggs, sweetcorn, butter, pistachios]
[eggs, sweetcorn, butter, pistachios, onion]
[banana, eggs, butter, pistachios, onion]
[banana, eggs, sweetcorn, butter, onion]
[banana, eggs, pistachios, almonds, tomato]
[banana, eggs, sweetcorn, pistachios, onion]
```

```
[oats, cereal, strawberry, carrots, chocolates]
[oats, cereal, blackberry, carrots, chocolates]
[oats, cereal, blackberry, strawberry, chocolates]
[cereal, blackberry, strawberry, carrots, chocolates]
[oats, cereal, strawberry, almonds, tomato]
[oats, cereal, blackberry, strawberry, carrots]
[sweetcorn, butter, blackberry, carrots, chocolates]
[sweetcorn, butter, onion, carrots, chocolates]
[butter, blackberry, onion, carrots, chocolates]
[sweetcorn, butter, blackberry, onion, chocolates]
[blackberry, carrots, almonds, tomato, chocolates]
[oats, blackberry, strawberry, carrots, chocolates]
[sweetcorn, blackberry, onion, carrots, chocolates]
[sweetcorn, butter, onion, almonds, tomato]
[banana, sweetcorn, butter, pistachios, onion]
[sweetcorn, butter, blackberry, onion, carrots]
[potato, greenpeas, almonds, tomato, cheese]

generated possible 6-itemsets 593775
frequent Item sets List from possible 6-item sets of 593775 are:
[banana, eggs, bread, pistachios, almonds, tomato]
[bread, oats, cereal, strawberry, almonds, tomato]
[bread, blackberry, carrots, almonds, tomato, chocolates]
[sweetcorn, bread, butter, onion, almonds, tomato]
[bread, potato, greenpeas, almonds, tomato, cheese]
[banana, eggs, sweetcorn, butter, pistachios, onion]
[oats, cereal, blackberry, strawberry, carrots, chocolates]
[sweetcorn, butter, blackberry, onion, carrots, chocolates]

generated possible 7-itemsets 2035800
frequent Item sets List from possible 7-item sets of 2035800 are:

computed top level associations:
 1: [banana, eggs, bread, pistachios, almonds, tomato]
 2: [bread, oats, cereal, strawberry, almonds, tomato]
 3: [bread, blackberry, carrots, almonds, tomato, chocolates]
 4: [sweetcorn, bread, butter, onion, almonds, tomato]
 5: [bread, potato, greenpeas, almonds, tomato, cheese]
 6: [banana, eggs, sweetcorn, butter, pistachios, onion]
 7: [oats, cereal, blackberry, strawberry, carrots, chocolates]
 8: [sweetcorn, butter, blackberry, onion, carrots, chocolates]
Total time taken in seconds with brute force approach: 14
```

Conclusion:

Upon running and comparing both the approaches, we can practically conclude that Apriori algorithm runs much faster than Bruteforce Approach.