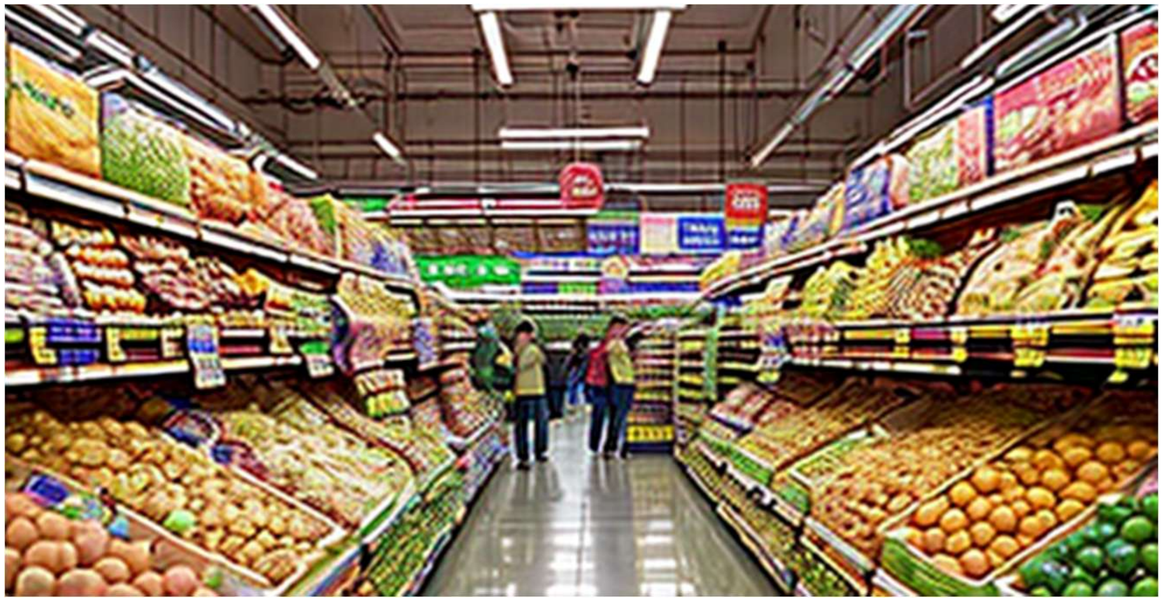
Lesson 2

Association Rule Mining
(Market Basket Analysis)

github.com/NhatDucHoang/IS_CS_466

https://github.com/NhatDucHoang/IS_CS_466/blob/main/Presentation%202_File_0.py

https://github.com/NhatDucHoang/IS_CS_466/blob/main/Presentation%202_File_1.py



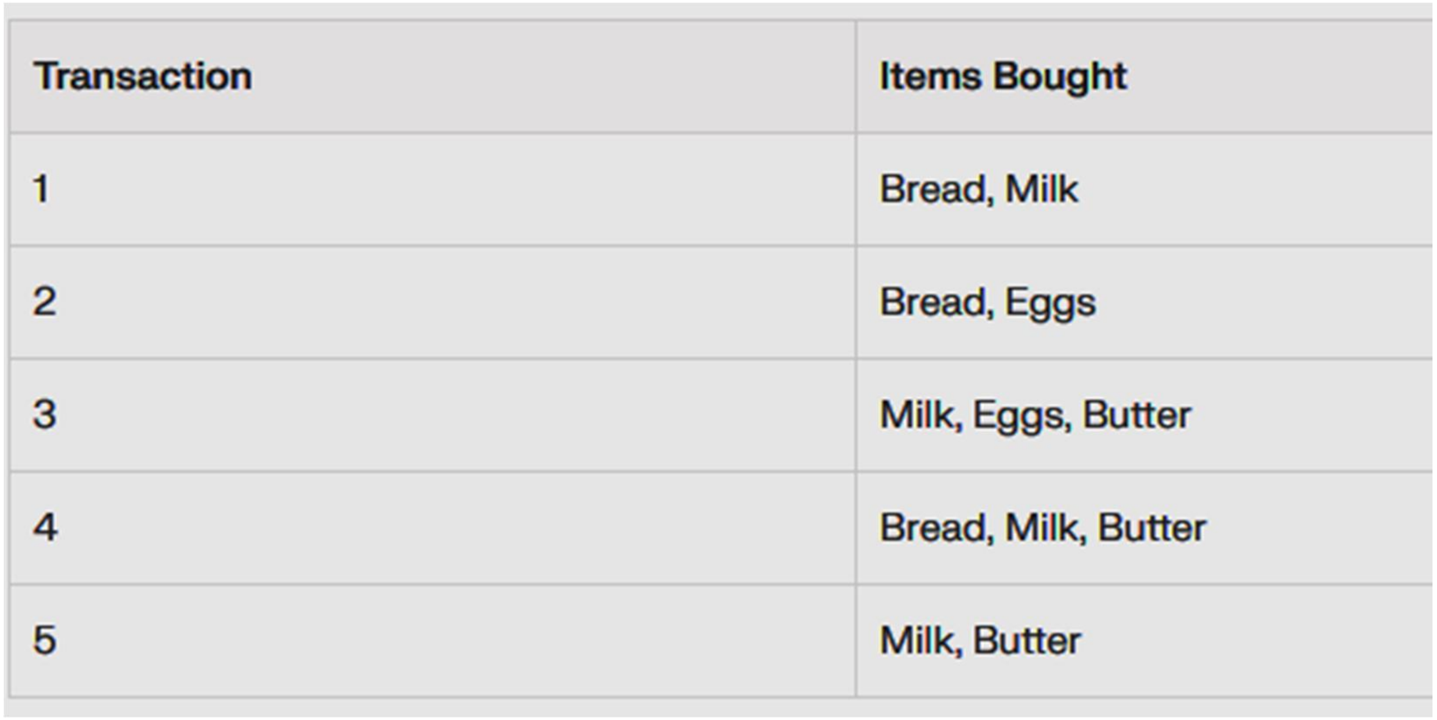
Association rule mining is a fundamental data mining technique designed to uncover meaningful if-then relationships.

Market basket analysis is primarily used for:

- Uncover relationships between products that are frequently purchased together.
- Helping businesses better understand customer purchasing behavior.
- Optimize decisions in areas such as inventory management, marketing, sales strategies, and store layout.

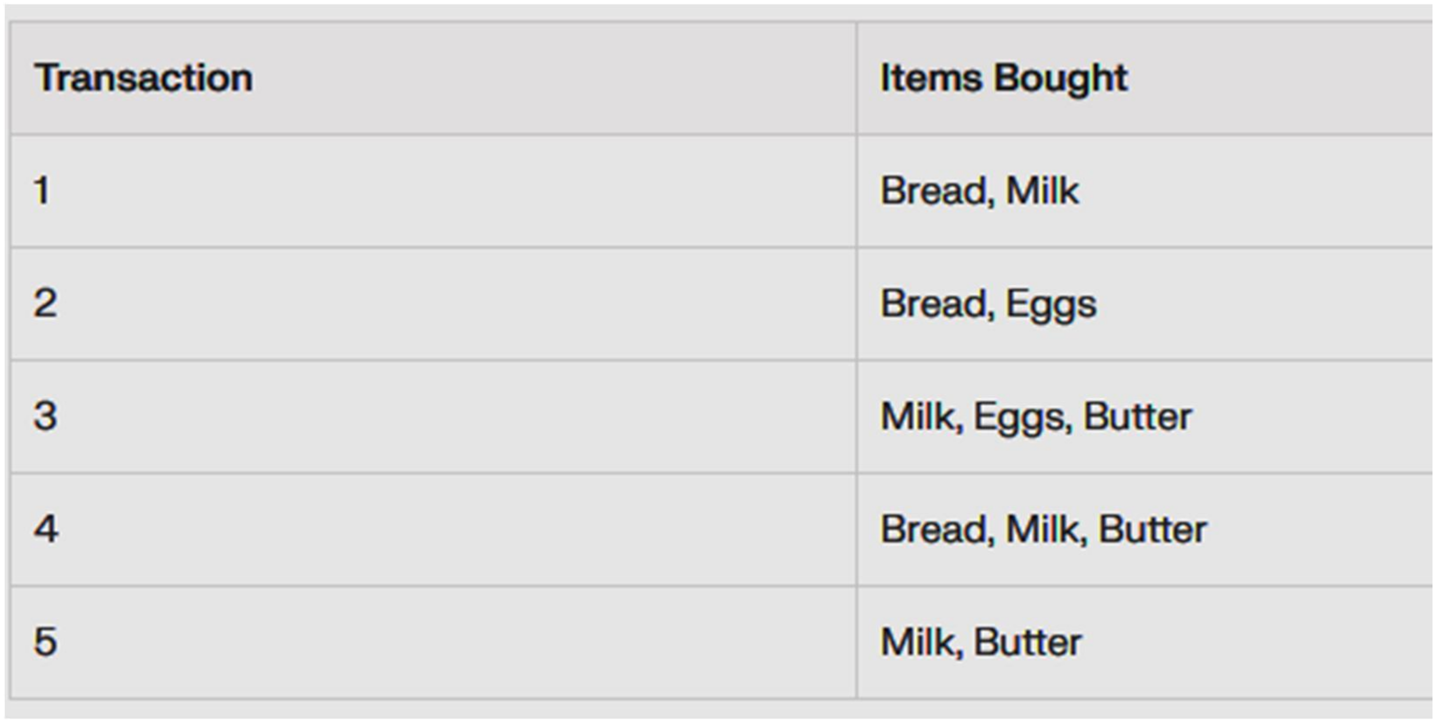
Consider a market basket analysis using association rule mining with a dataset containing 4 goods:

- Bread
- Milk
- Eggs
- Butter



Transaction	Items Bought
1	Bread, Milk
2	Bread, Eggs
3	Milk, Eggs, Butter
4	Bread, Milk, Butter
5	Milk, Butter

‘Milk → Butter’ is an association rule.
Let compute how strong the ‘Milk → Butter’ rule is ?



Transaction	Items Bought
1	Bread, Milk
2	Bread, Eggs
3	Milk, Eggs, Butter
4	Bread, Milk, Butter
5	Milk, Butter

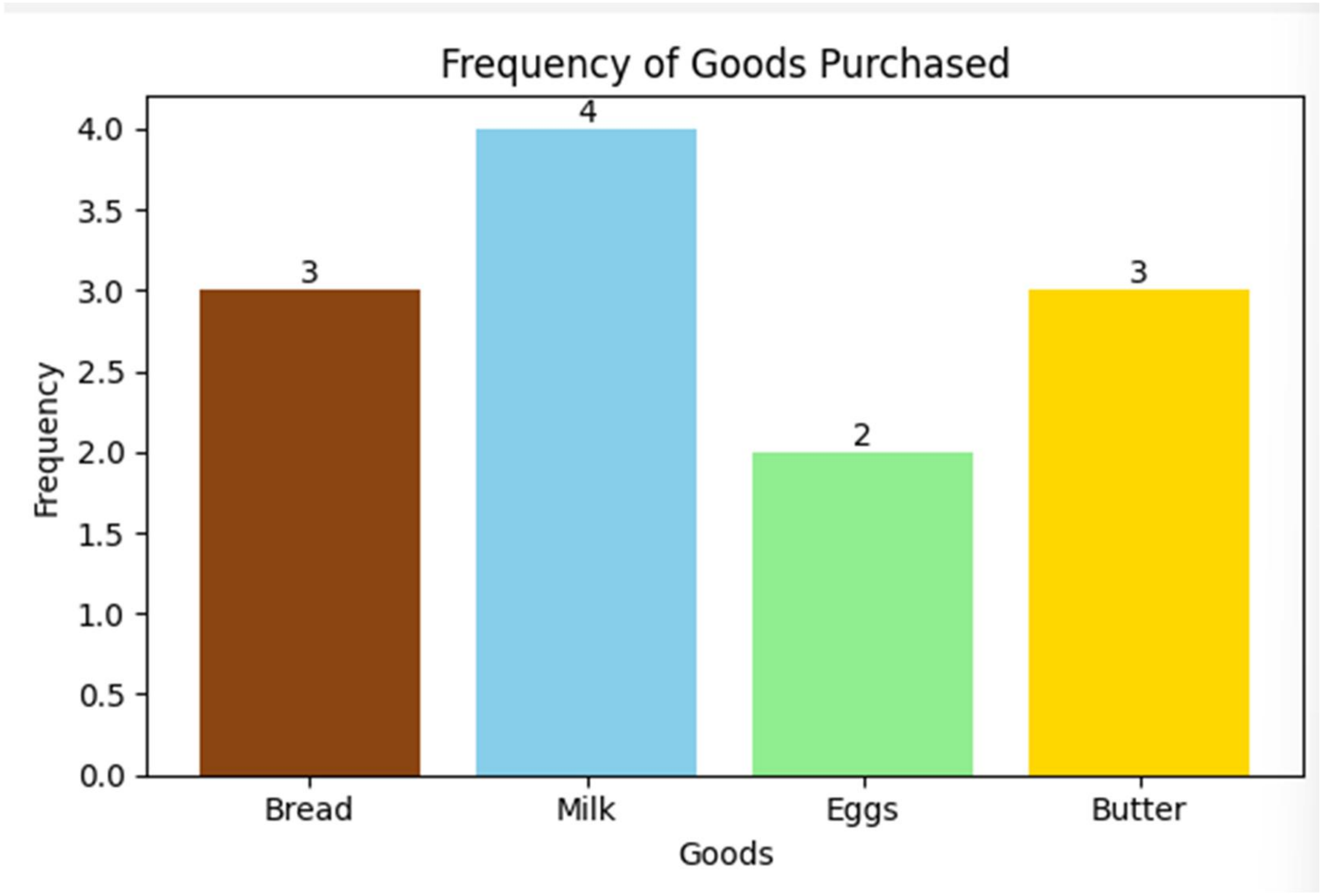
If the ‘Milk → Butter’ rule is proven to be strong, a store owner can arrange milk and butter side by side or in close proximity to maximize visibility and convenience for shoppers.

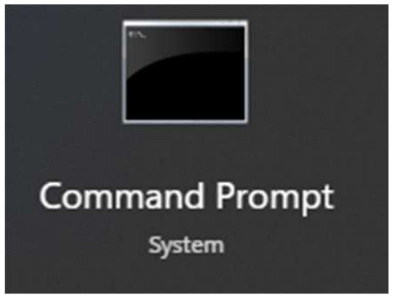
Step 1: Identify Frequent Itemsets

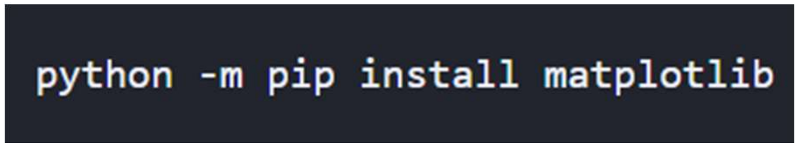
Let’s find itemsets that appear together frequently:

Single items:

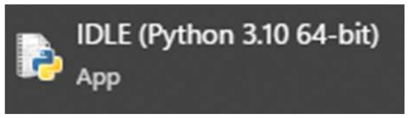
Bread (3), Milk (4), Eggs (2), Butter (3)

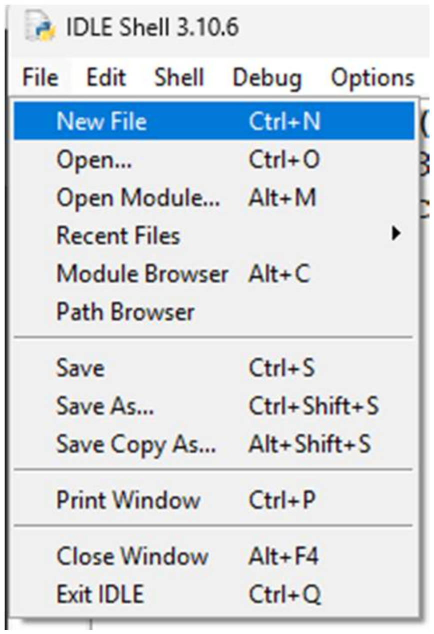


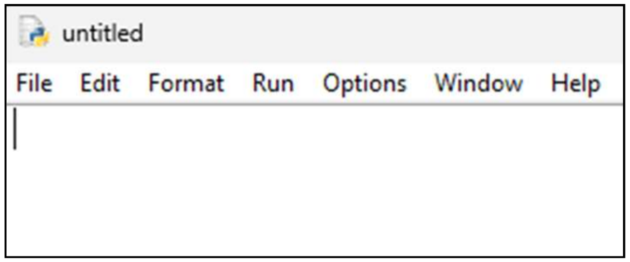


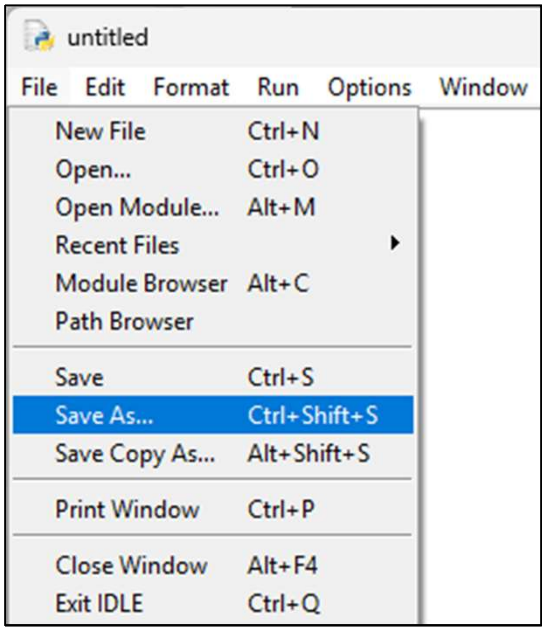


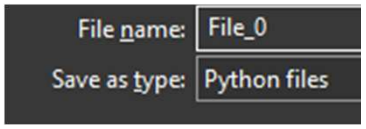
```
python -m pip install matplotlib
```



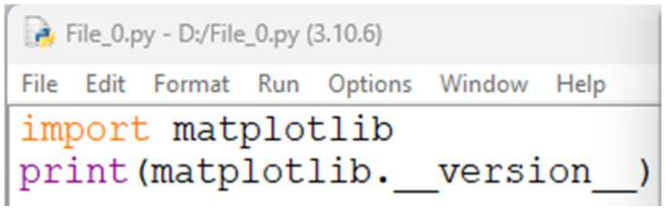




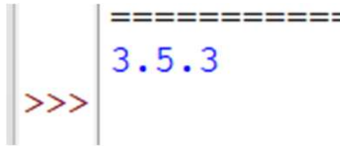




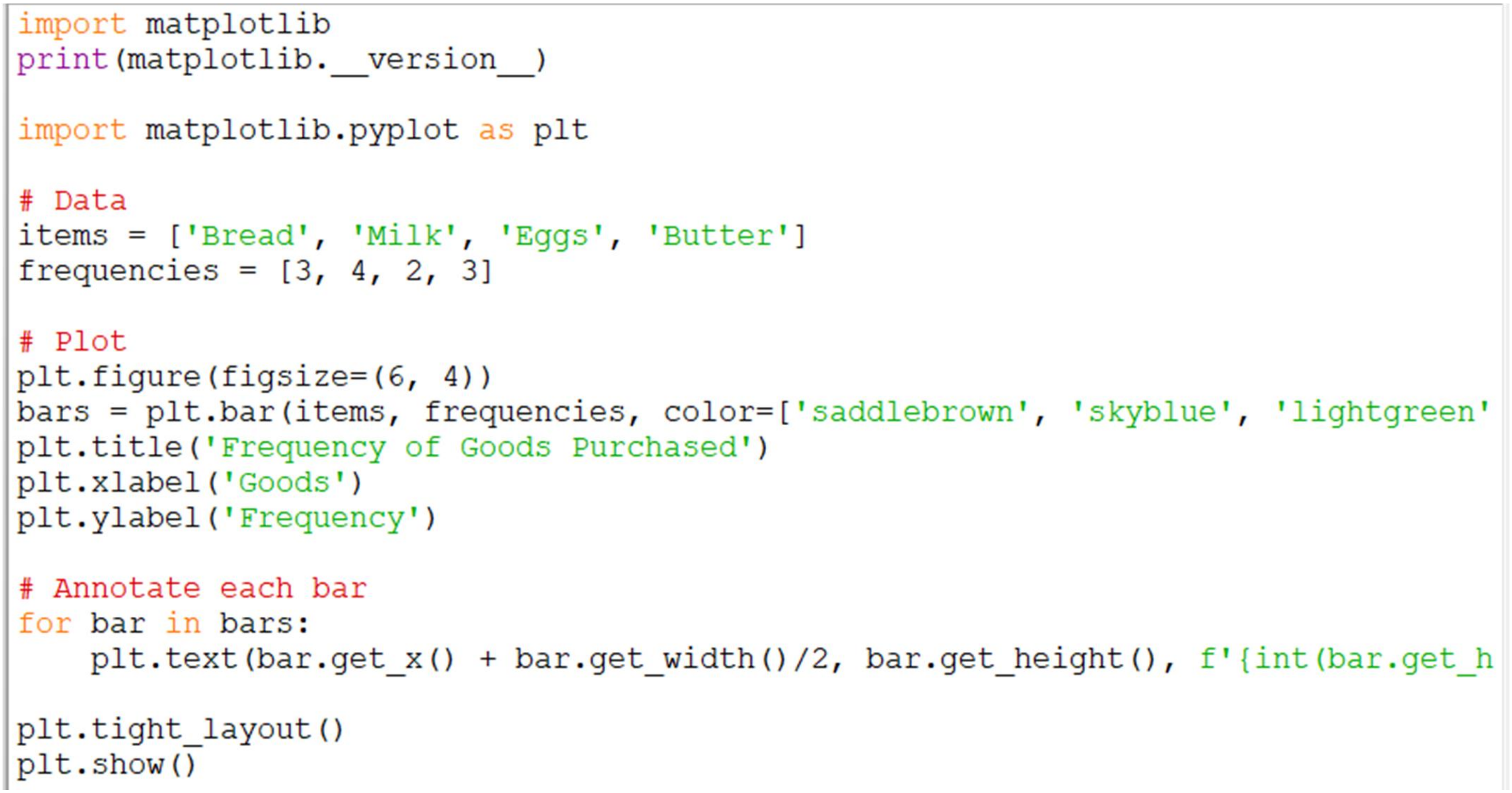
File name: File_0
Save as type: Python files


File_0.py - D:/File_0.py (3.10.6)
File Edit Format Run Options Window Help

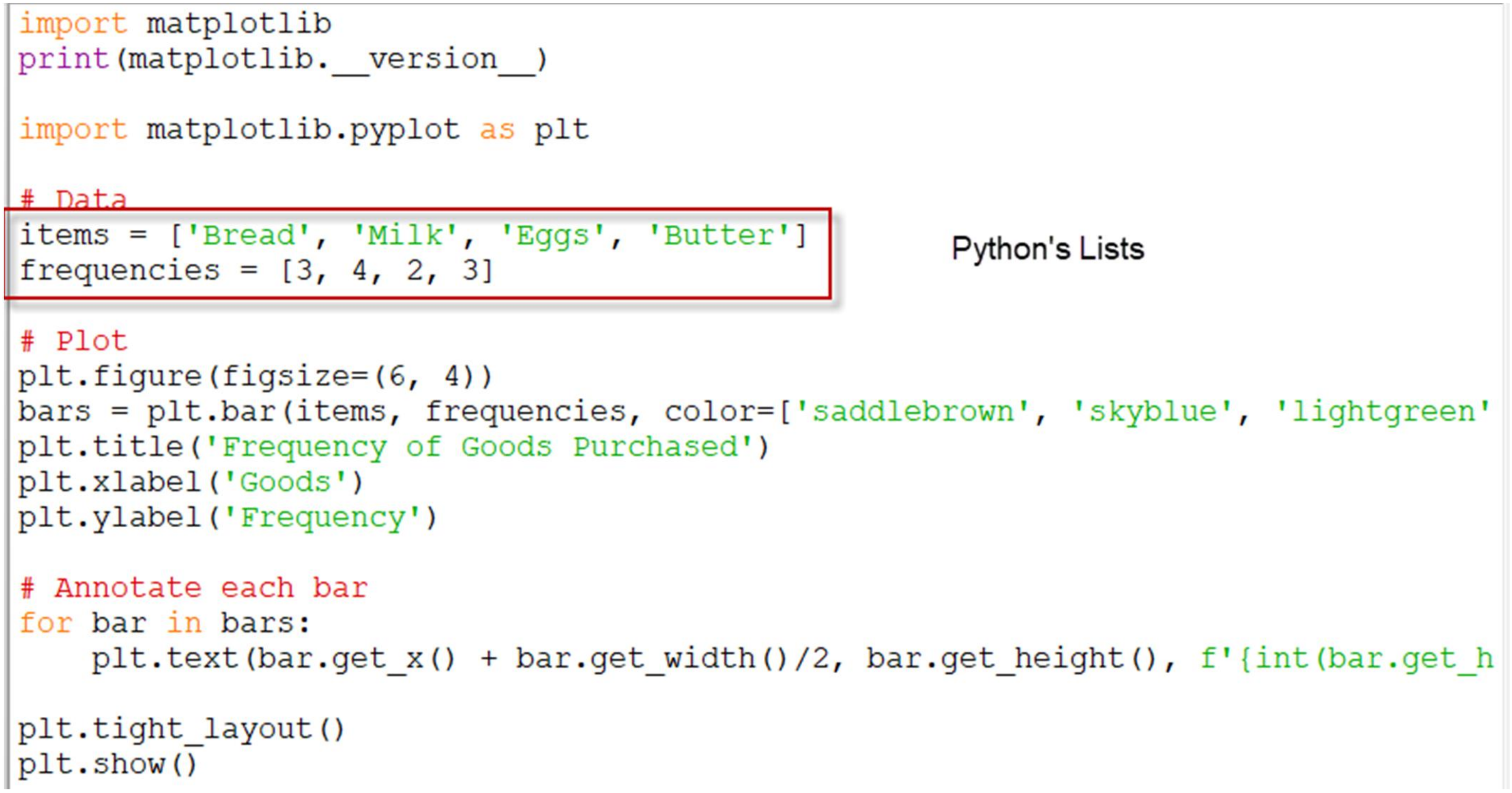
```python
import matplotlib
print(matplotlib.__version__)
```



```
===========
3.5.3
>>>
```



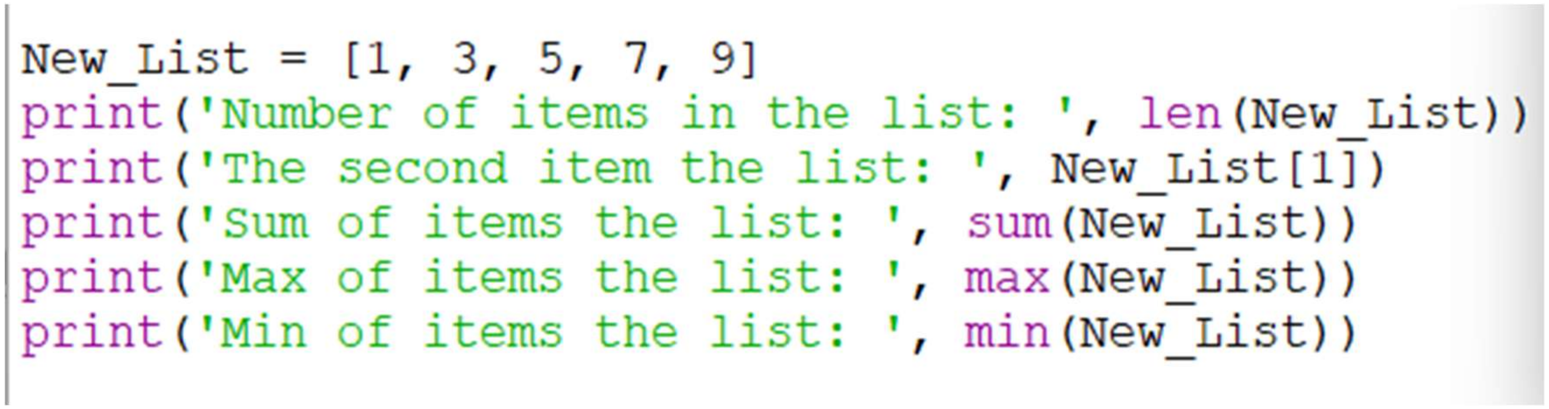
```
import matplotlib
print(matplotlib.__version__)

import matplotlib.pyplot as plt

# Data
items = ['Bread', 'Milk', 'Eggs', 'Butter']
frequencies = [3, 4, 2, 3]

# Plot
plt.figure(figsize=(6, 4))
bars = plt.bar(items, frequencies, color=['saddlebrown', 'skyblue', 'lightgreen'
plt.title('Frequency of Goods Purchased')
plt.xlabel('Goods')
plt.ylabel('Frequency')

# Annotate each bar
for bar in bars:
    plt.text(bar.get_x() + bar.get_width()/2, bar.get_height(), f'{int(bar.get_h

plt.tight_layout()
plt.show()
```



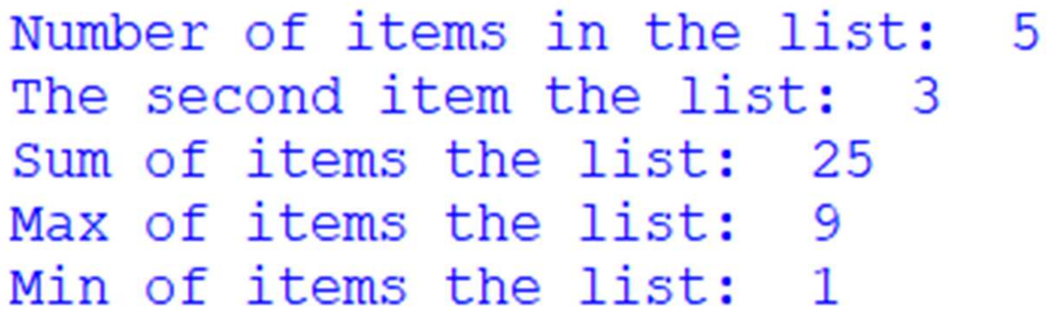
```
import matplotlib
print(matplotlib.__version__)

import matplotlib.pyplot as plt

# Data
items = ['Bread', 'Milk', 'Eggs', 'Butter']
frequencies = [3, 4, 2, 3]

# Plot
plt.figure(figsize=(6, 4))
bars = plt.bar(items, frequencies, color=['saddlebrown', 'skyblue', 'lightgreen'
plt.title('Frequency of Goods Purchased')
plt.xlabel('Goods')
plt.ylabel('Frequency')

# Annotate each bar
for bar in bars:
    plt.text(bar.get_x() + bar.get_width()/2, bar.get_height(), f'{int(bar.get_h

plt.tight_layout()
plt.show()
```

Python's Lists

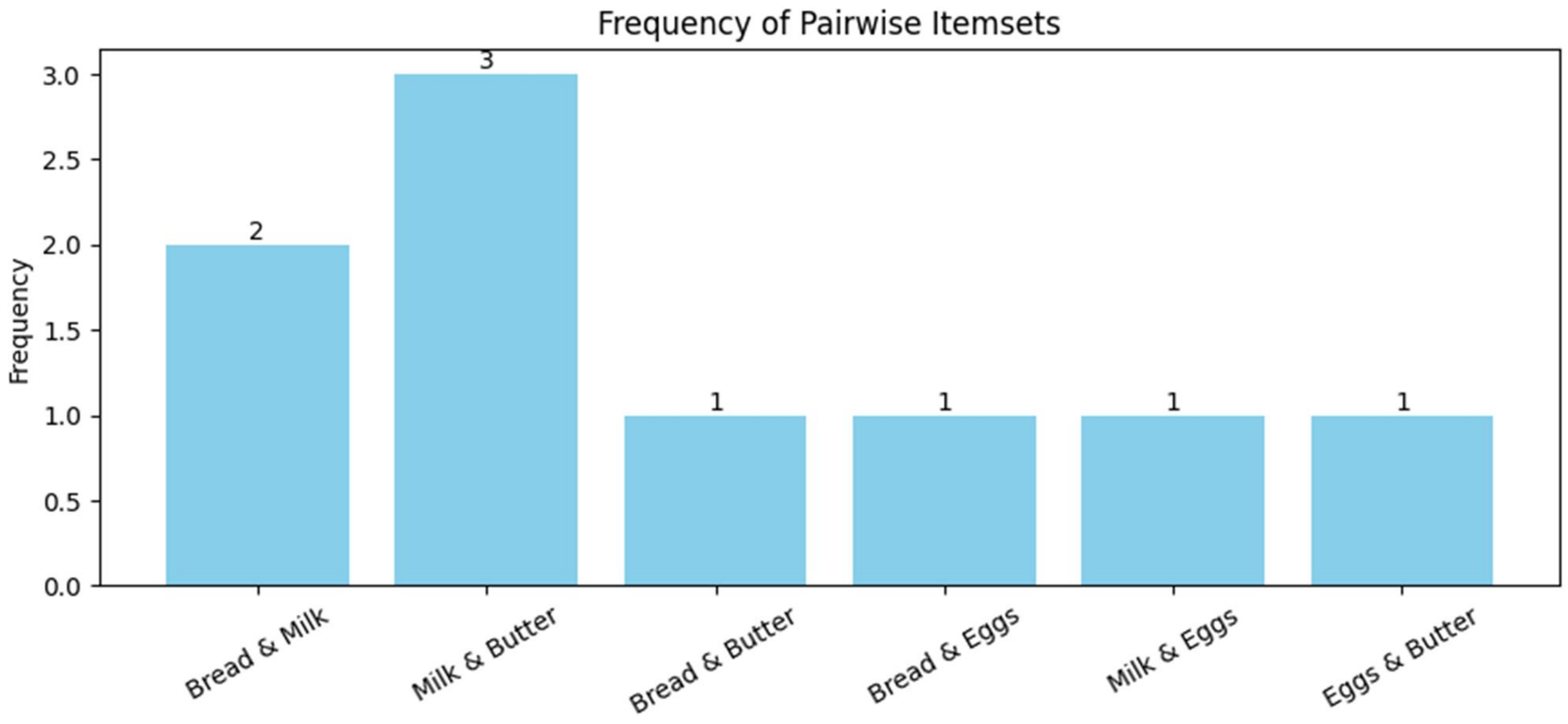
Python’s List



```
New_List = [1, 3, 5, 7, 9]
print('Number of items in the list: ', len(New_List))
print('The second item the list: ', New_List[1])
print('Sum of items the list: ', sum(New_List))
print('Max of items the list: ', max(New_List))
print('Min of items the list: ', min(New_List))
```



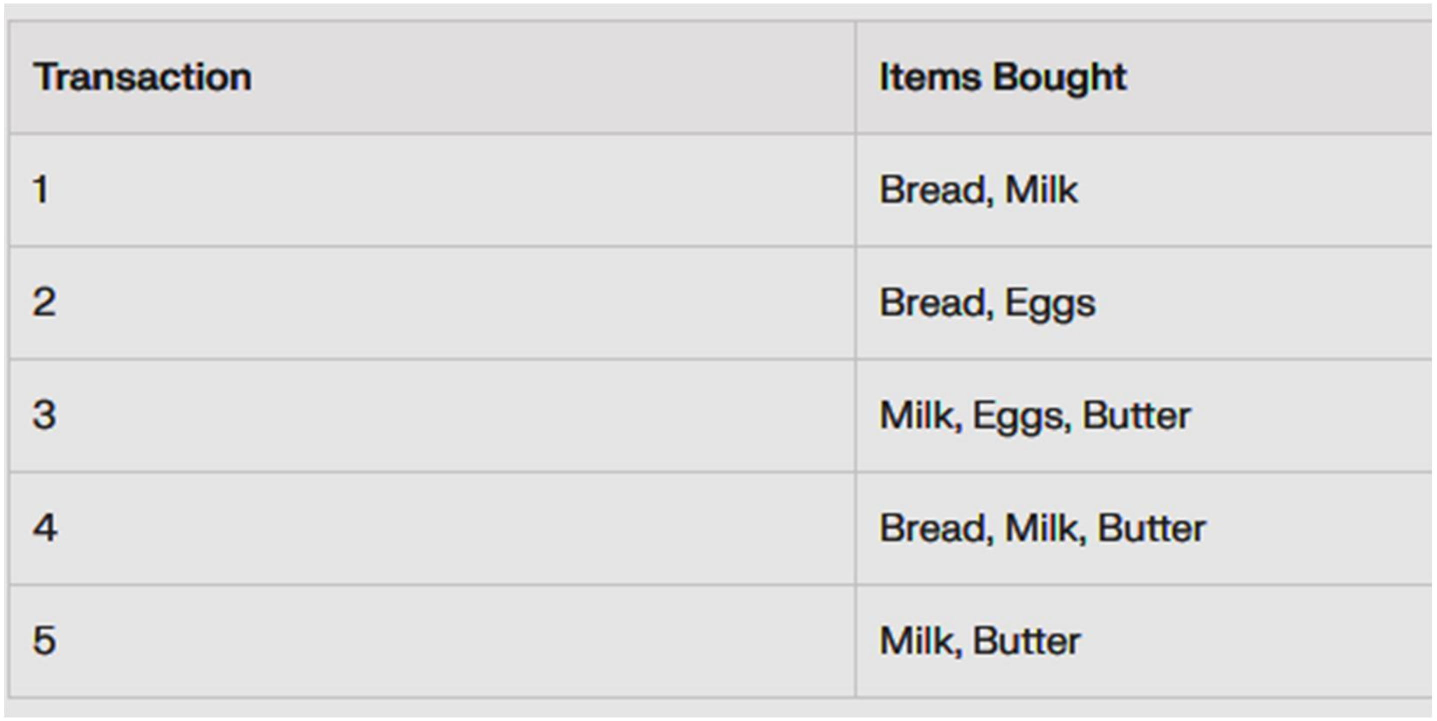
```
Number of items in the list:  5
The second item the list:  3
Sum of items the list:  25
Max of items the list:  9
Min of items the list:  1
```


Frequency of Pairwise Itemsets
Frequency
3.0
2.5
2.0
1.5
1.0
0.5
0.0
2
3
1
1
1
1
Bread & Milk
Milk & Butter
Bread & Butter
Bread & Eggs
Milk & Eggs
Eggs & Butter

Step 2: Calculate Support

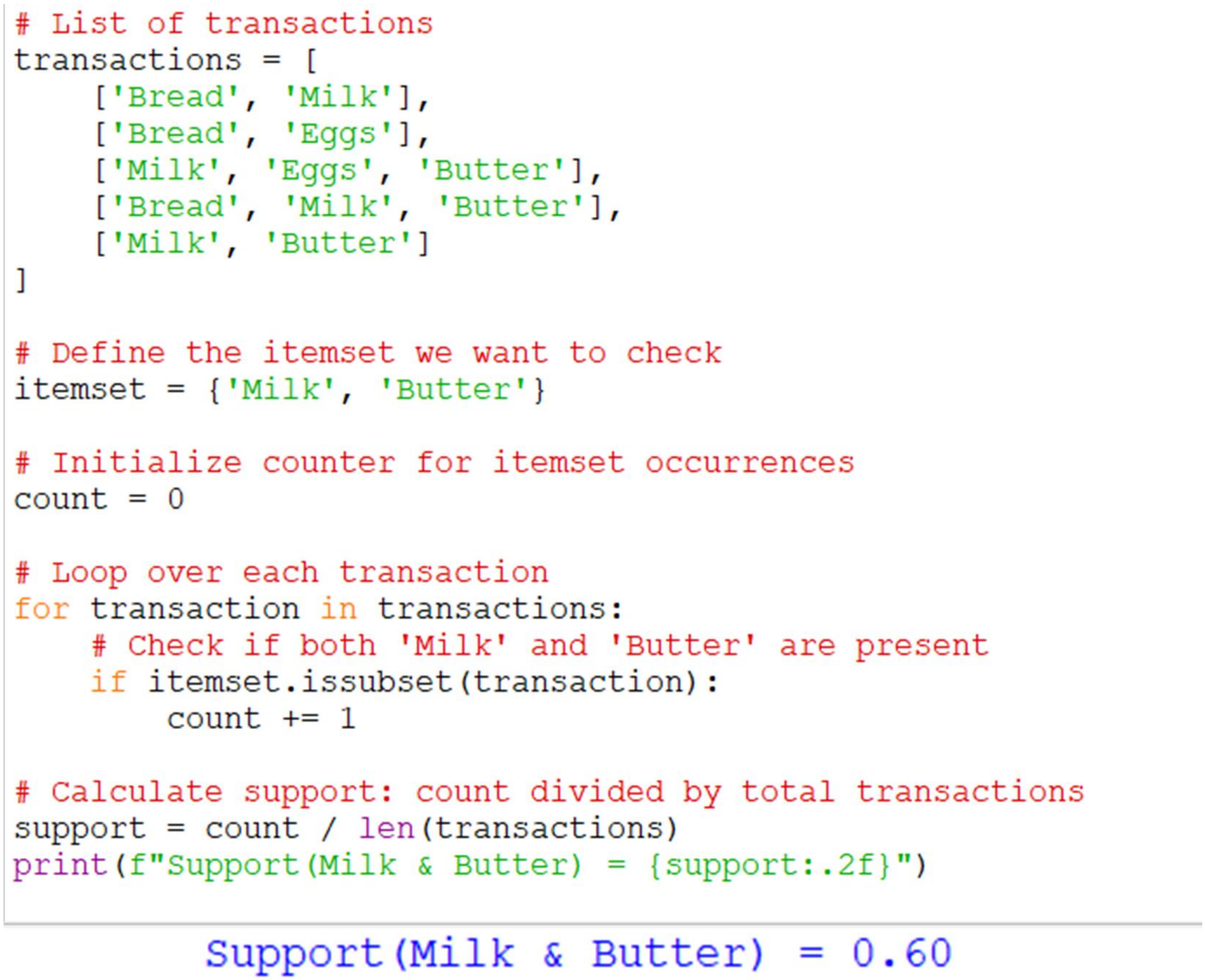
Support = frequency of itemset / total transactions

Support(Milk & Butter) = 3/5 = 60%



Transaction	Items Bought
1	Bread, Milk
2	Bread, Eggs
3	Milk, Eggs, Butter
4	Bread, Milk, Butter
5	Milk, Butter

We can use Python’s set and a for loop to compute the support of interest.



```python
# List of transactions
transactions = [
    ['Bread', 'Milk'],
    ['Bread', 'Eggs'],
    ['Milk', 'Eggs', 'Butter'],
    ['Bread', 'Milk', 'Butter'],
    ['Milk', 'Butter']
]

# Define the itemset we want to check
itemset = {'Milk', 'Butter'}

# Initialize counter for itemset occurrences
count = 0

# Loop over each transaction
for transaction in transactions:
    # Check if both 'Milk' and 'Butter' are present
    if itemset.issubset(transaction):
        count += 1

# Calculate support: count divided by total transactions
support = count / len(transactions)
print(f"Support(Milk & Butter) = {support:.2f}")
```

```
Support(Milk & Butter) = 0.60
```

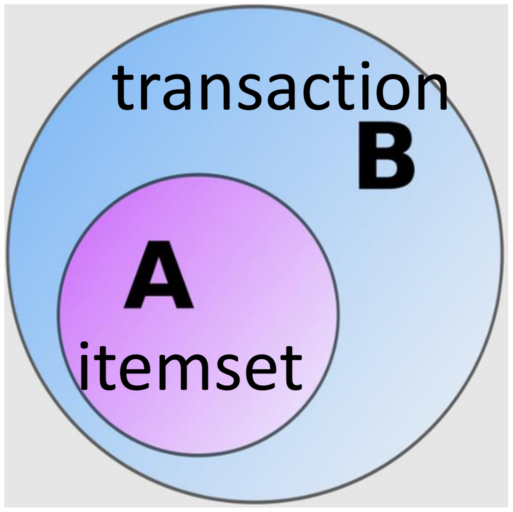
```python
# Define the itemset we want to check
itemset = {'Milk', 'Butter'}
```

A Python set is a built-in data type that represents an unordered collection of unique, immutable elements.

Sets are defined using curly braces {} with comma-separated values.

Subset checking is very fast.

```python
if itemset.issubset(transaction):
    count += 1
```



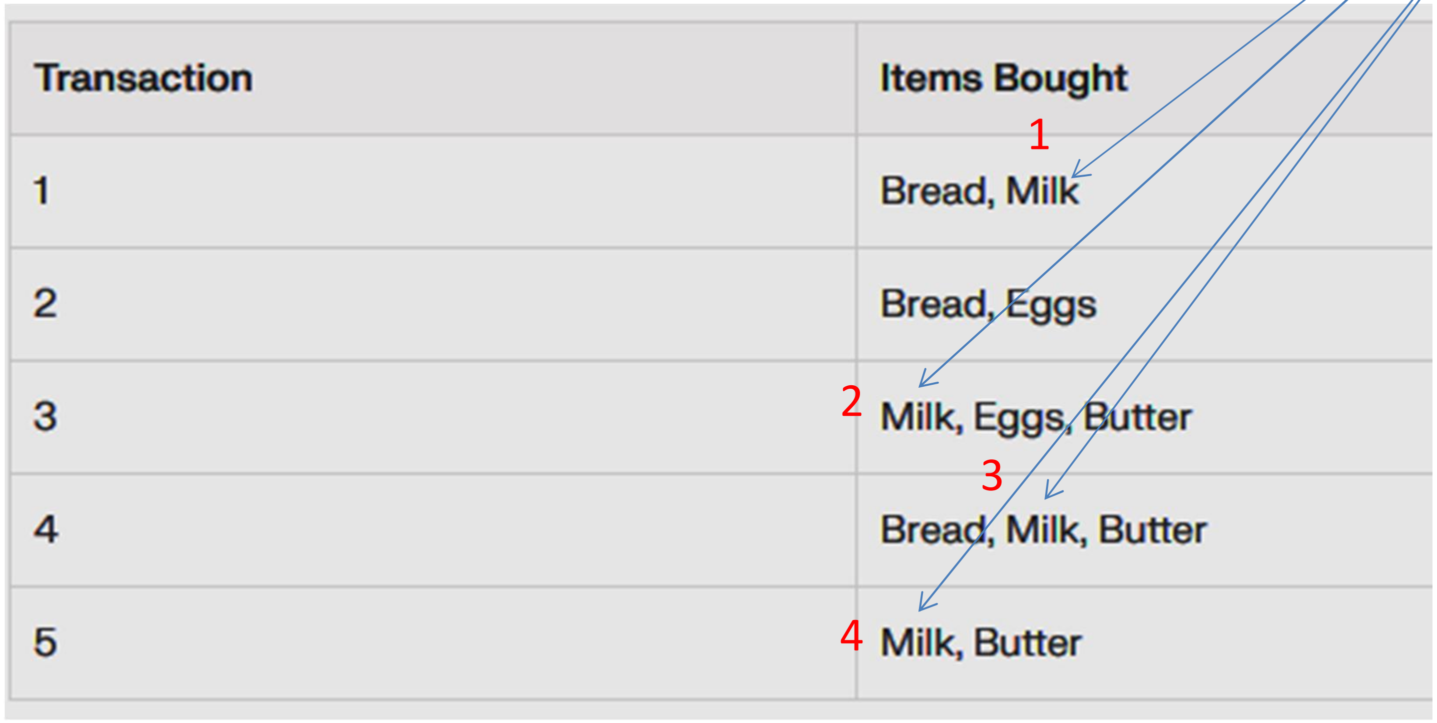
Step 3: Calculate Confidence

Confidence(A → B) = Support(A & B) / Support(A)

Milk → Butter:

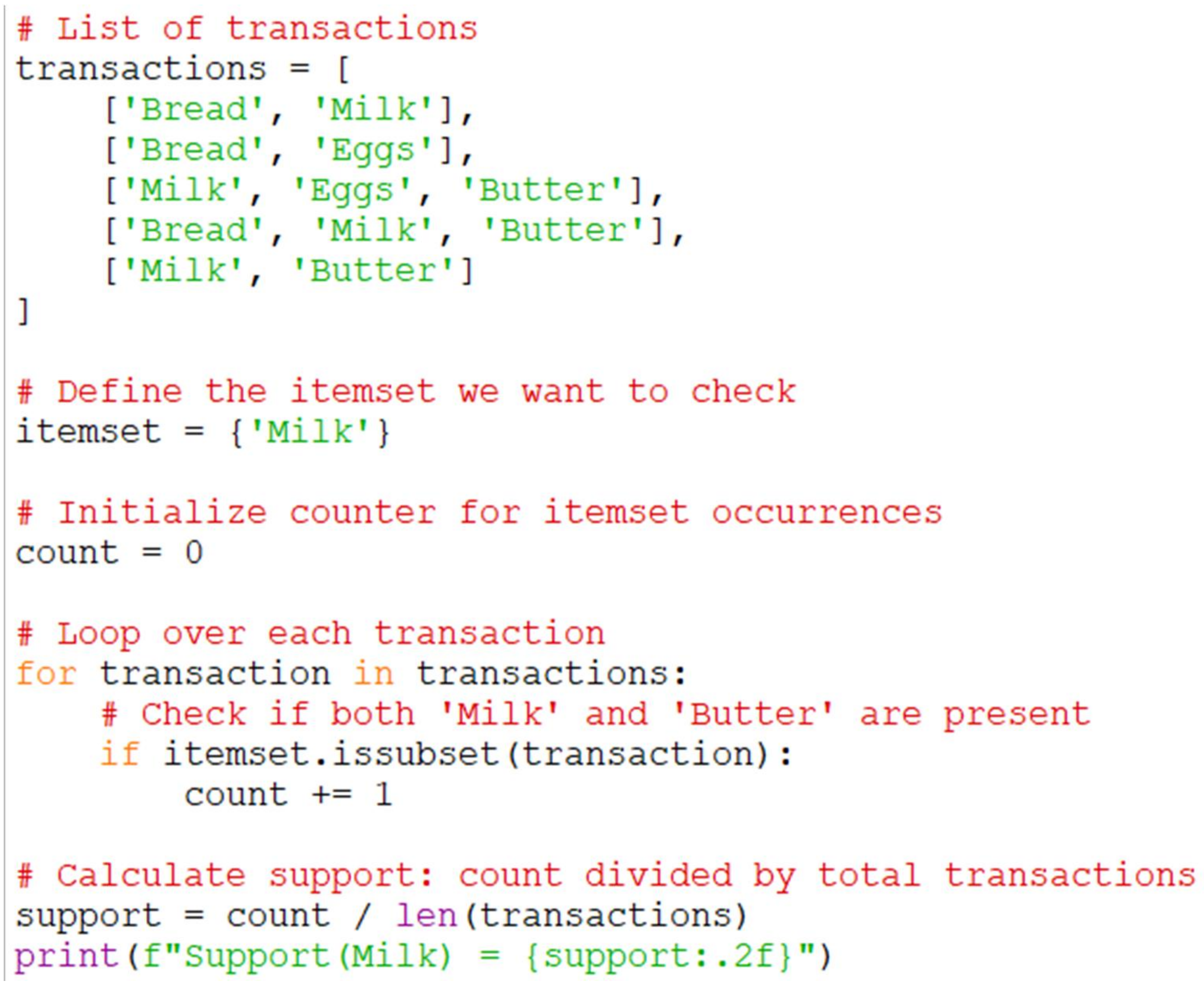
Confidence = Support(Milk & Butter) / Support(Milk)

```
Support(Milk & Butter) = 0.60
```

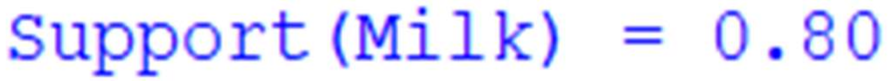


Transaction	Items Bought
1	Bread, Milk
2	Bread, Eggs
3	Milk, Eggs, Butter
4	Bread, Milk, Butter
5	Milk, Butter

Support(Milk) =
4/5 = 0.8



```python
# List of transactions
transactions = [
    ['Bread', 'Milk'],
    ['Bread', 'Eggs'],
    ['Milk', 'Eggs', 'Butter'],
    ['Bread', 'Milk', 'Butter'],
    ['Milk', 'Butter']
]

# Define the itemset we want to check
itemset = {'Milk'}

# Initialize counter for itemset occurrences
count = 0

# Loop over each transaction
for transaction in transactions:
    # Check if both 'Milk' and 'Butter' are present
    if itemset.issubset(transaction):
        count += 1

# Calculate support: count divided by total transactions
support = count / len(transactions)
print(f"Support(Milk) = {support:.2f}")
```



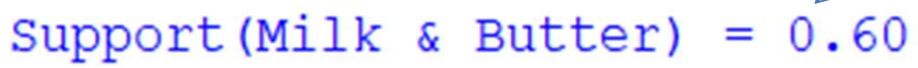
```
Support(Milk) = 0.80
```

Step 3: Calculate Confidence

Confidence(A → B) = Support(A & B) / Support(A)

Milk → Butter:

Confidence = Support(Milk & Butter) / Support(Milk)



```
Support(Milk & Butter) = 0.60
```

Support(Milk) = 4/5 = 0.8

Confidence (Milk → Butter) = 0.6/0.8 = 0.75

Rule: If Milk is bought, Butter is likely also bought.

- Support of milk & butter: 60%
- Confidence of milk & butter : 75%

To determine the usefulness of a rule, support and confidence must be considered:

- High support, high confidence: rule is both relevant and reliable.
- High confidence, low support: rule is reliable but affects few transactions (may not be valuable).
- High support, low confidence: rule is common but not predictive.

High support, high confidence

In a large supermarket, almost every time a customer buys coffee, they also put sugar in their basket.

High support: Coffee and sugar are bought together by a large proportion of shoppers.

High confidence: If someone buys coffee, we can be almost certain they will also buy sugar.

The “**if-then**” relationship (If coffee is bought → sugar is bought) is very dependable.

High confidence, low support

In a food store, whenever a customer buys caviar, they also tend to buy champagne.

High confidence: Almost *every* time caviar is purchased, champagne is also in the basket.

Low support: Only a small fraction of customers actually buy caviar — it’s an expensive product.

So the rule applies to very few transactions overall, meaning its reach is limited.

High support, low confidence

In a supermarket, milk is bought by lots of customers every. Tuna, on the other hand, is also sold frequently, but not every milk buyer chooses tuna.

In fact, most milk buyers skip tuna for other items.

High support: Both milk and tuna show up often in overall sales. They are individually popular

Low confidence: Only a small subset of milk buyers also buy tuna (milk is often for children, so hygiene concerns are considered).

The rule “If Milk is bought, Butter is likely also bought” is indeed a high support and high confidence case.

Let use the threshold of 0.6 for support and confidence.

We write a program to yield the conclusion about the rule as follows:
If support >= 0.6 and confidence >= 0.6 then display: ‘high support - high confidence’.

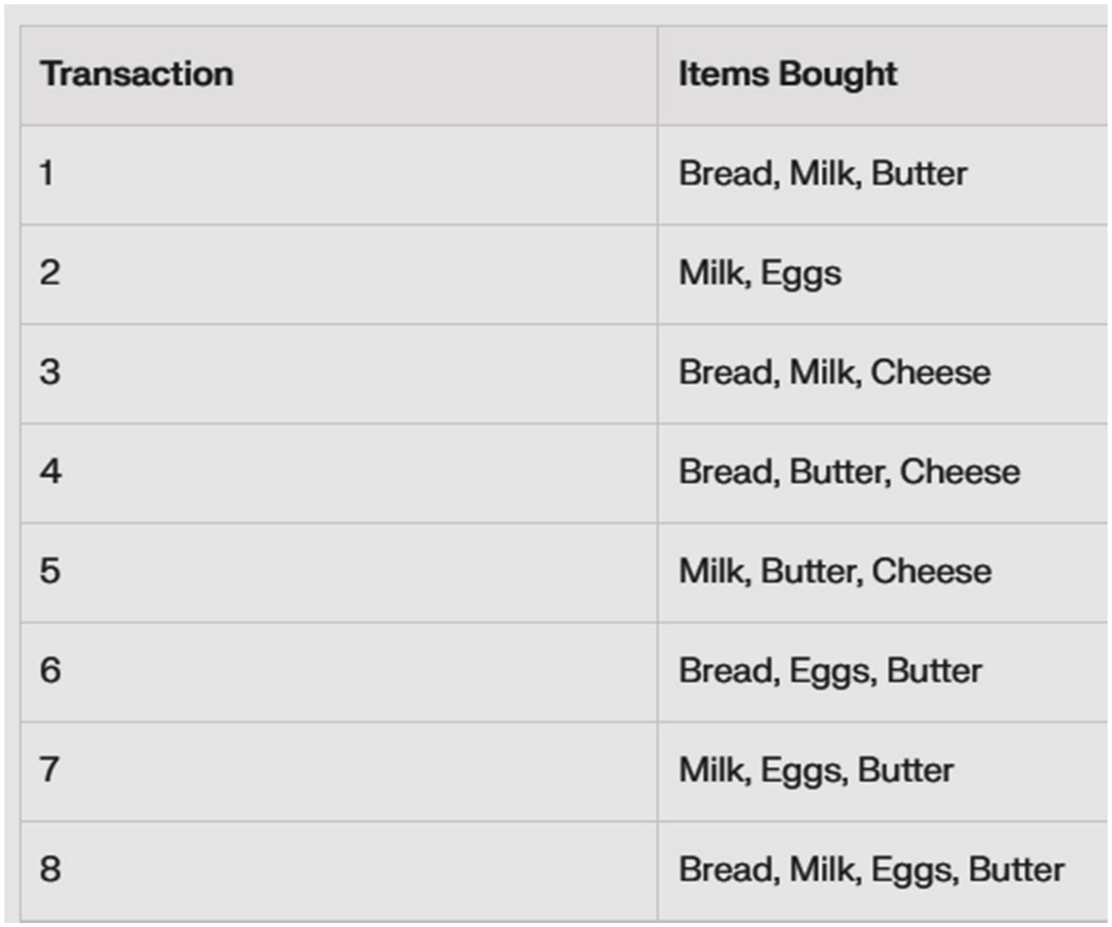
If support < 0.6 and confidence > 0.6 then display: ‘High support - low support’.

If support >= 0.6 and confidence < 0.6 then display: ‘high support - low confidence’.

If support < 0.6 and confidence < 0.6 then display: ‘low support - low confidence’.

```python
support = 0.60
confidence = 0.75

if support >= 0.6 and confidence >= 0.6:
    print("High support - high confidence"

if support < 0.6 and confidence >= 0.6:
    print("Low support - high confidence")

if support >= 0.6 and confidence < 0.6:
    print("High support - low confidence")

if support < 0.6 and confidence < 0.6:
    print("Low support - low confidence")
```

Exercise 1

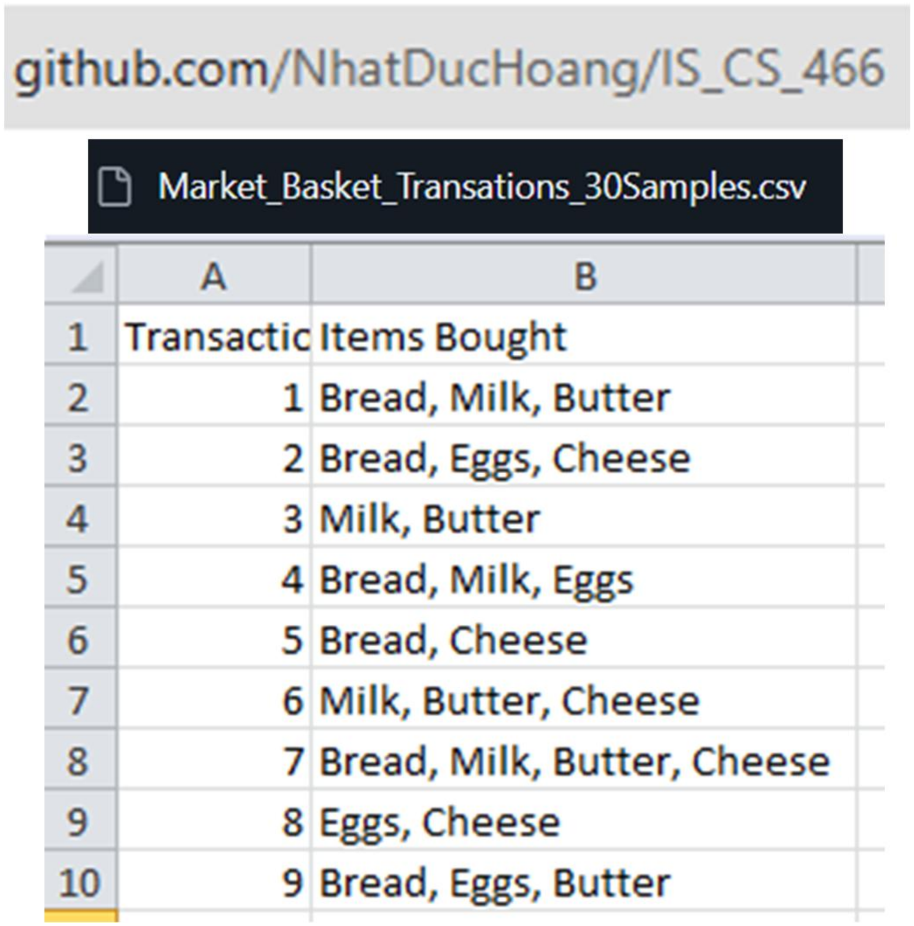


Transaction	Items Bought
1	Bread, Milk, Butter
2	Milk, Eggs
3	Bread, Milk, Cheese
4	Bread, Butter, Cheese
5	Milk, Butter, Cheese
6	Bread, Eggs, Butter
7	Milk, Eggs, Butter
8	Bread, Milk, Eggs, Butter

a. Write code to draw a bar chart to display the frequency of single items.
b. Write code to compute the support (bread & eggs) and support (bread)
c. Write code to compute the confidence of the rule: **bread -> eggs** and yield conclusion about the rule.

Confidence(A → B) = Support(A & B) / Support(A)

Exercise 2



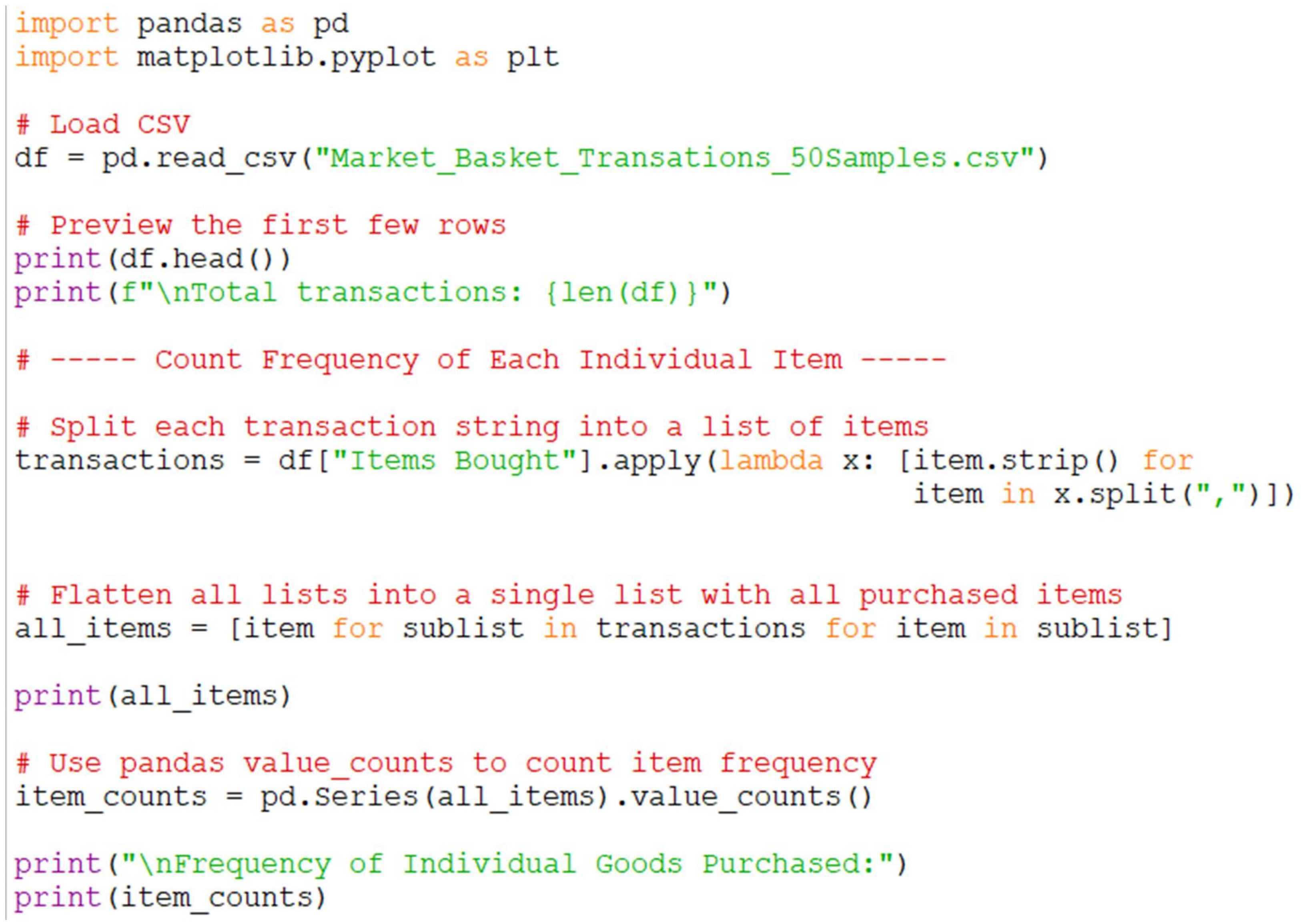
github.com/NhatDucHoang/IS_CS_466

Market_Basket_Transations_30Samples.csv

	A	B
1	Transactic	Items Bought
2	1	Bread, Milk, Butter
3	2	Bread, Eggs, Cheese
4	3	Milk, Butter
5	4	Bread, Milk, Eggs
6	5	Bread, Cheese
7	6	Milk, Butter, Cheese
8	7	Bread, Milk, Butter, Cheese
9	8	Eggs, Cheese
10	9	Bread, Eggs, Butter

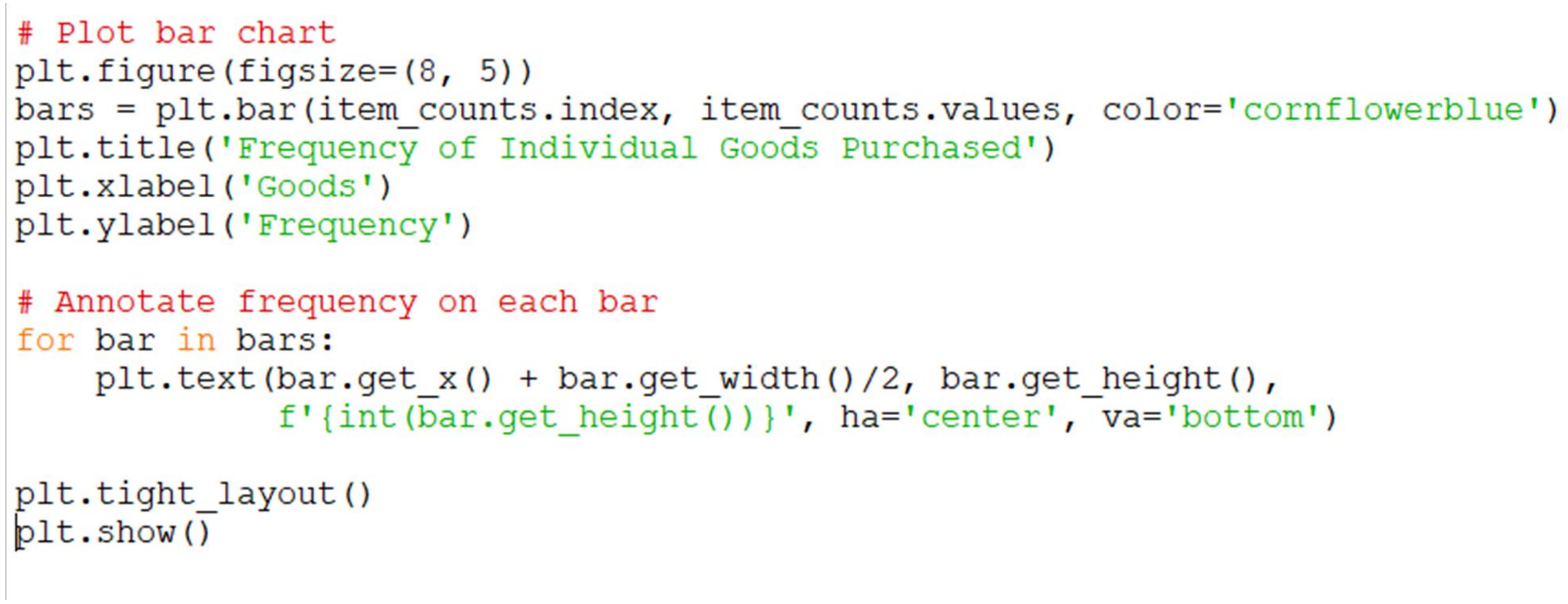
```
import pandas as pd
df = pd.read_csv("Market_Basket_Transations_30Samples.csv")
print(df.head())
print(f"\nTotal transactions: {len(df)}")

# Extract transactions as list of lists
transactions = df["Items Bought"].apply(lambda x: [item.strip() for item in x.split(",")]).tolist()

print(transactions[0])
print(transactions[1])
```

Use pandas to compute the number of individual goods purchased



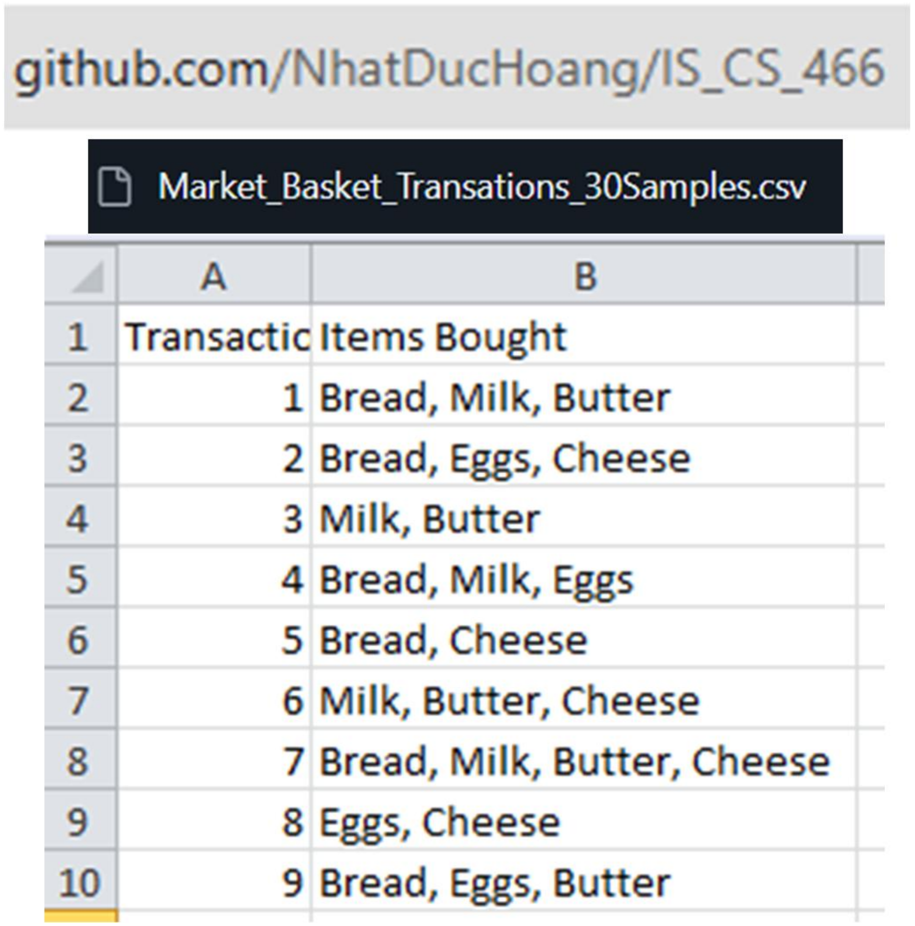
```python
import pandas as pd
import matplotlib.pyplot as plt

# Load CSV
df = pd.read_csv("Market_Basket_Transations_50Samples.csv")

# Preview the first few rows
print(df.head())
print(f"\nTotal transactions: {len(df)}")

# ----- Count Frequency of Each Individual Item -----

# Split each transaction string into a list of items
transactions = df["Items Bought"].apply(lambda x: [item.strip() for
                                                   item in x.split(",")])


# Flatten all lists into a single list with all purchased items
all_items = [item for sublist in transactions for item in sublist]

print(all_items)

# Use pandas value_counts to count item frequency
item_counts = pd.Series(all_items).value_counts()

print("\nFrequency of Individual Goods Purchased:")
print(item_counts)
```

Use pandas to compute the number of individual goods purchased



```python
# Plot bar chart
plt.figure(figsize=(8, 5))
bars = plt.bar(item_counts.index, item_counts.values, color='cornflowerblue')
plt.title('Frequency of Individual Goods Purchased')
plt.xlabel('Goods')
plt.ylabel('Frequency')

# Annotate frequency on each bar
for bar in bars:
    plt.text(bar.get_x() + bar.get_width()/2, bar.get_height(),
             f'{int(bar.get_height())}', ha='center', va='bottom')

plt.tight_layout()
plt.show()
```

https://github.com/NhatDucHoang/IS_CS_466/blob/main/Presentation%202_File_1.py



github.com/NhatDucHoang/IS_CS_466

Market_Basket_Transations_30Samples.csv

	A	B
1	Transactic	Items Bought
2	1	Bread, Milk, Butter
3	2	Bread, Eggs, Cheese
4	3	Milk, Butter
5	4	Bread, Milk, Eggs
6	5	Bread, Cheese
7	6	Milk, Butter, Cheese
8	7	Bread, Milk, Butter, Cheese
9	8	Eggs, Cheese
10	9	Bread, Eggs, Butter

a. Write code to draw a bar chart to display the frequency of single items.
b. Write code to compute the support (milk & cheese) and support (milk)
c. Write code to compute the confidence of the rule: milk **->** cheese and yield conclusion about the rule.

Computing mean and standard deviation of a sample

The variance of a sample of n measurements $y_1, y_2, \ldots, y_n$ is defined as

$$s^2 = \frac{\sum_{i=1}^{n}(y_i - \bar{y})^2}{n-1} = \frac{\sum_{i=1}^{n} y_i^2 - n\bar{y}^2}{n-1}$$

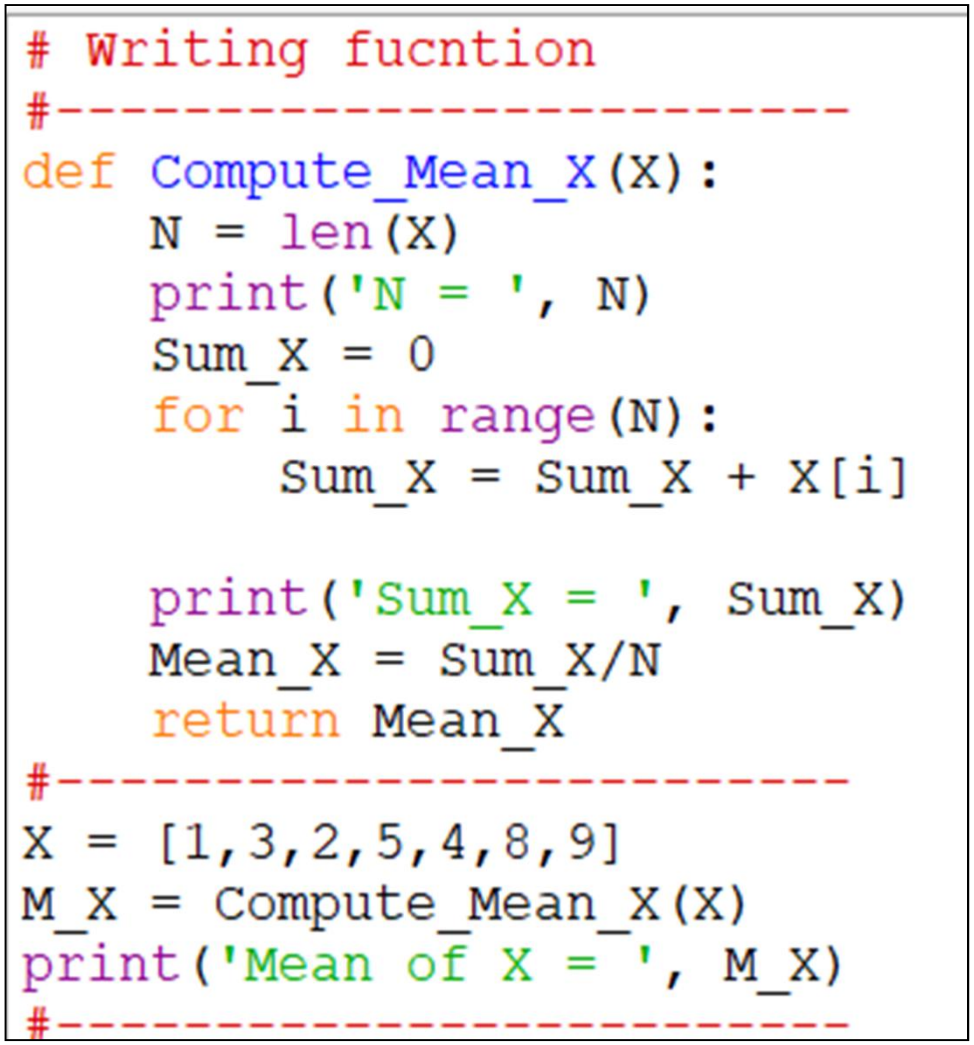
where $\bar{y}$ is the sample mean.

The standard deviation of a set of measurements is the square root of the variance.

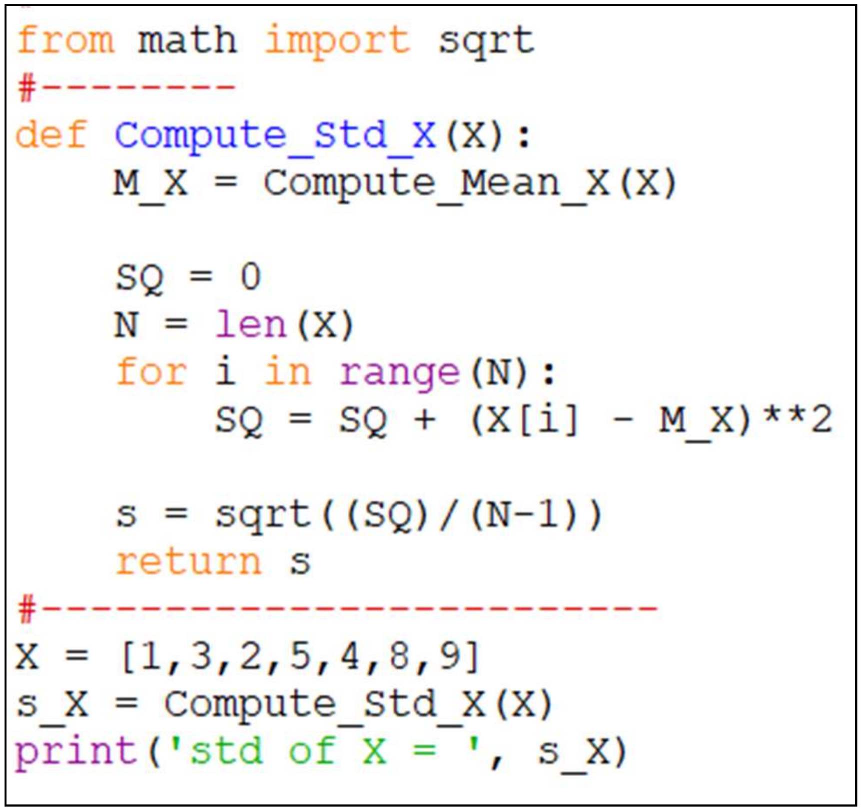
The formula for the standard deviation s is:

$$s = \sqrt{s^2}$$

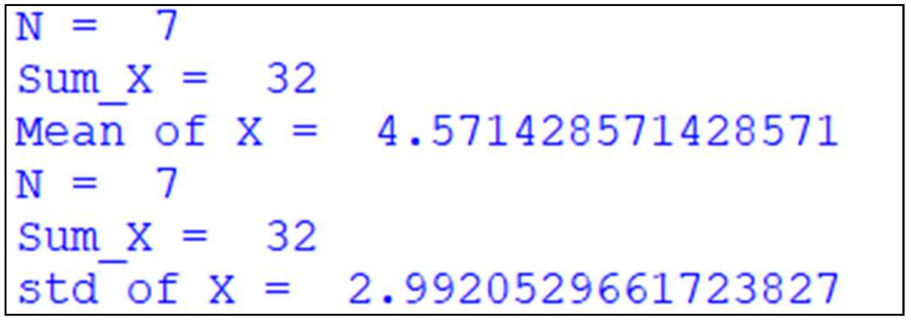
where s^2 represents the variance.



```python
# Writing fucntion
#--------------------------
def Compute_Mean_X(X):
    N = len(X)
    print('N = ', N)
    Sum_X = 0
    for i in range(N):
        Sum_X = Sum_X + X[i]

    print('Sum_X = ', Sum_X)
    Mean_X = Sum_X/N
    return Mean_X
#--------------------------
X = [1,3,2,5,4,8,9]
M_X = Compute_Mean_X(X)
print('Mean of X = ', M_X)
#--------------------------
```

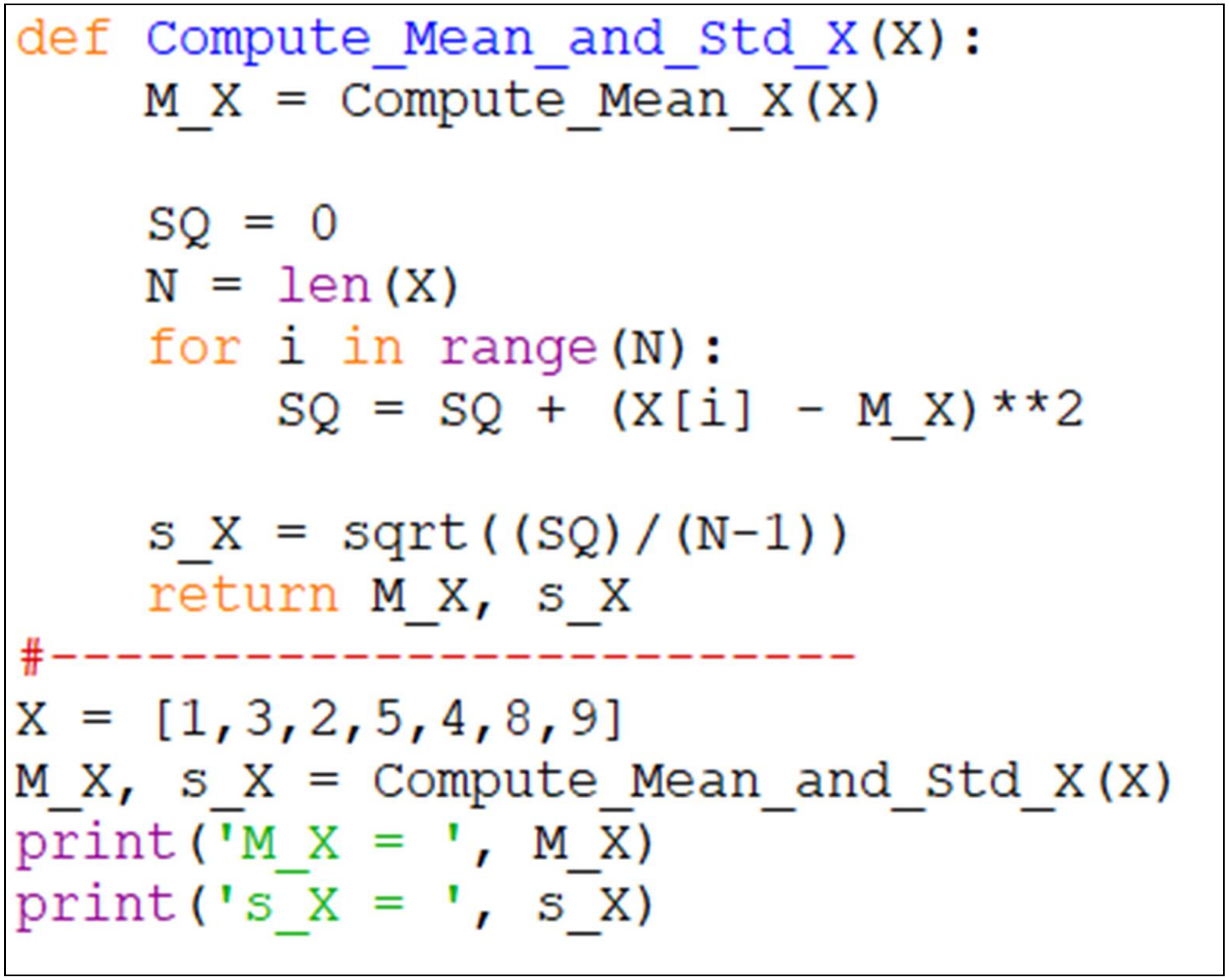


```python
from math import sqrt
#--------
def Compute_Std_X(X):
    M_X = Compute_Mean_X(X)

    SQ = 0
    N = len(X)
    for i in range(N):
        SQ = SQ + (X[i] - M_X)**2

    s = sqrt((SQ)/(N-1))
    return s
#-------------------------
X = [1,3,2,5,4,8,9]
s_X = Compute_Std_X(X)
print('std of X = ', s_X)
```



```
N =  7
Sum_X =  32
Mean of X =  4.571428571428571
N =  7
Sum_X =  32
std of X =  2.992052961723827
```

A function that return multiple outputs



```
def Compute_Mean_and_Std_X(X):
    M_X = Compute_Mean_X(X)

    SQ = 0
    N = len(X)
    for i in range(N):
        SQ = SQ + (X[i] - M_X)**2

    s_X = sqrt((SQ)/(N-1))
    return M_X, s_X
#-------------------------
X = [1,3,2,5,4,8,9]
M_X, s_X = Compute_Mean_and_Std_X(X)
print('M_X = ', M_X)
print('s_X = ', s_X)
```

Computing the confidence interval of a sample

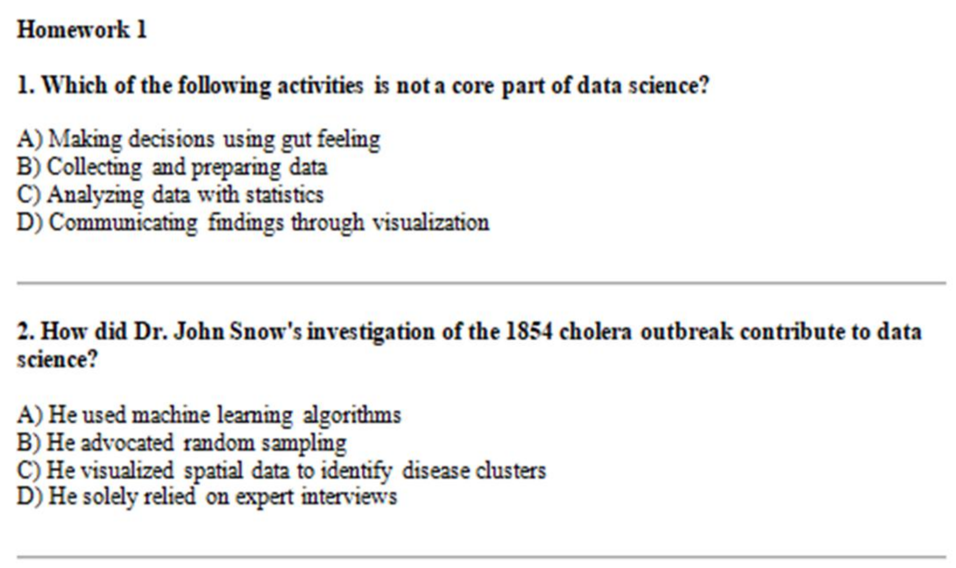
$$CI = \bar{x} \pm M \times \frac{s}{\sqrt{n}}$$

where $M = 2.26$ is the coeffcient corresponding to a 95% confidence level.

Task

Write a function code to compute the CI of a sample provided in a list.

- Input: a Python list named X
- Output: low and upper values

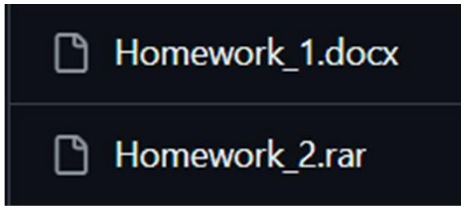

Homework 1

1. Which of the following activities is not a core part of data science?

A) Making decisions using gut feeling
B) Collecting and preparing data
C) Analyzing data with statistics
D) Communicating findings through visualization

2. How did Dr. John Snow's investigation of the 1854 cholera outbreak contribute to data science?

A) He used machine learning algorithms
B) He advocated random sampling
C) He visualized spatial data to identify disease clusters
D) He solely relied on expert interviews


Homework_1.docx
Homework_2.rar

Homework 2

Download the transaction data at:

https://github.com/NhatDucHoang/IS_CS_466/blob/main/Market_Basket_Transations_50Samples.csv

a. Write code to draw a bar chart to display the frequency of single items.

b. Write code to compute the confidence of the rule: milk -> apple and yield conclusion about the rule.

c. Should the store owner place milk and apples near each other?