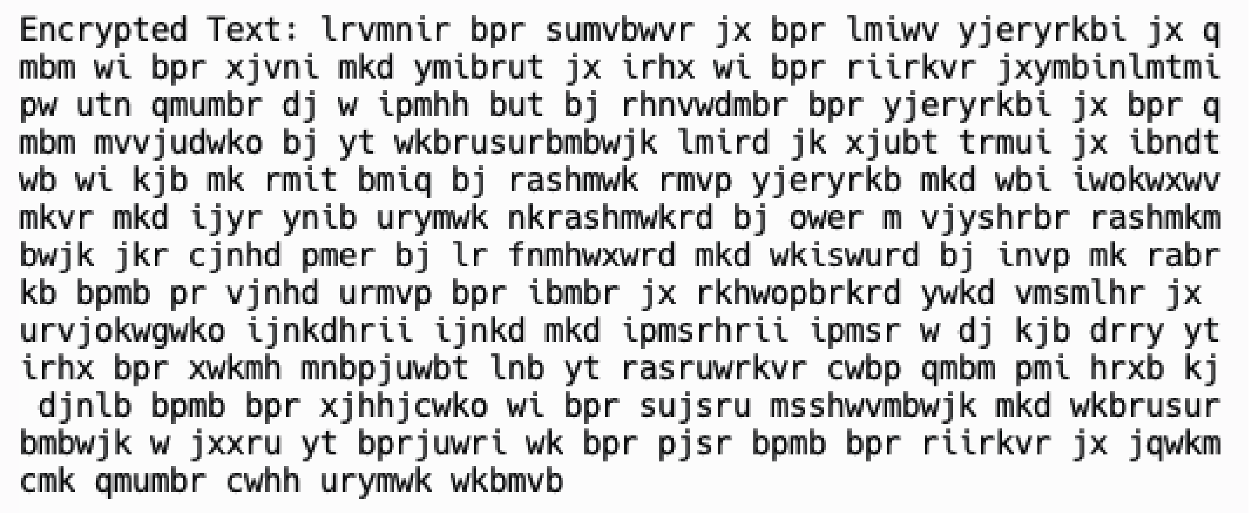
Project 1 Report

- **Programmer**: Lucas Yao
- **Class**: CSCI 360 Cryptography and Cryptanalysis
- **Professor**: Matluba Khodjaeva
- **Date**: 2/8/2024
- **Input**: N/A

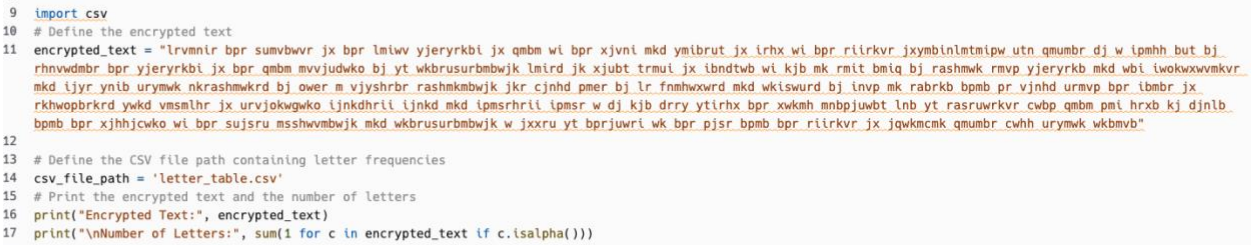
Project Description: This project aims to decrypt encrypted texts using the English frequency letter table obtained from a CSV file. The decryption process involves matching the frequencies of letters in the encrypted text with those in the CSV file to identify the corresponding letters. The Python programming language is utilized for implementation.

Procedure:

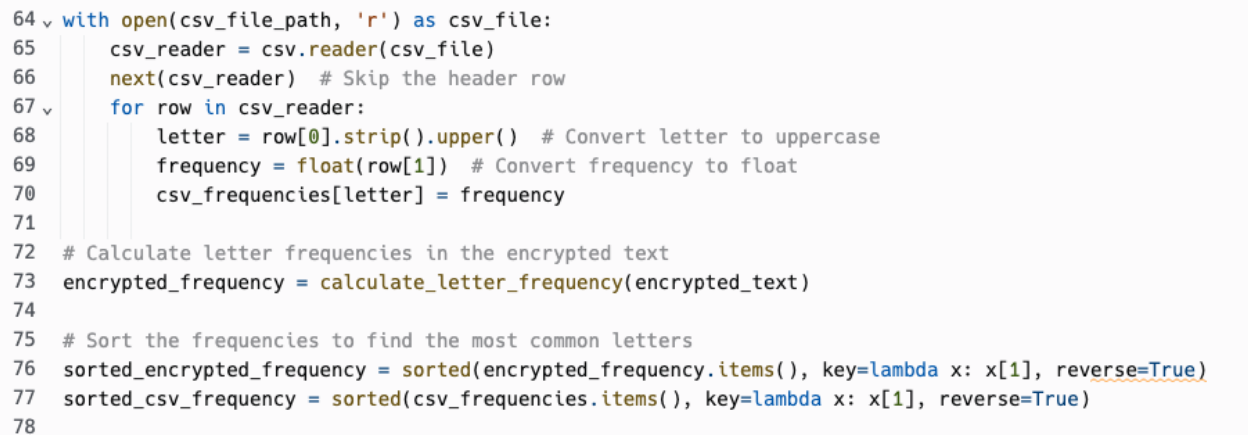
1. **Input**: The encrypted text and the path to the CSV file containing letter frequencies are provided.



```
Encrypted Text: lrvmnir bpr sumvbwvr jx bpr lmiwv yjeryrkbi jx q
mbm wi bpr xjvni mkd ymibrut jx irhx wi bpr riirkvr jxymbinlmtmi
pw utn qmumbr dj w ipmhh but bj rhnvwdmbr bpr yjeryrkbi jx bpr q
mbm mvvjudwko bj yt wkbrusurbmbwjk lmird jk xjubt trmui jx ibndt
wb wi kjb mk rmit bmiq bj rashmwk rmvp yjeryrkb mkd wbi iwokwxwv
mkvr mkd ijyr ynib urymwk nkrashmwkrd bj ower m vjyshrbr rashmkm
bwjk jkr cjnhd pmer bj lr fnmhwxwrd mkd wkiswurd bj invp mk rabr
kb bpmb pr vjnhd urmvp bpr ibmbr jx rkhwopbrkrd ywkd vmsmlhr jx
urvjokwgwko ijnkdhrii ijnkd mkd ipmsrhrii ipmsr w dj kjb drry yt
irhx bpr xwkmh mnbpjuwbt lnb yt rasruwrkvr cwbp qmbm pmi hrxb kj
 djnlb bpmb bpr xjhhjcwko wi bpr sujsru msshwvmbwjk mkd wkbrusur
bmbwjk w jxxru yt bprjuwri wk bpr pjsr bpmb bpr riirkvr jx jqwkm
cmk qmumbr cwhh urymwk wkbmvb
```

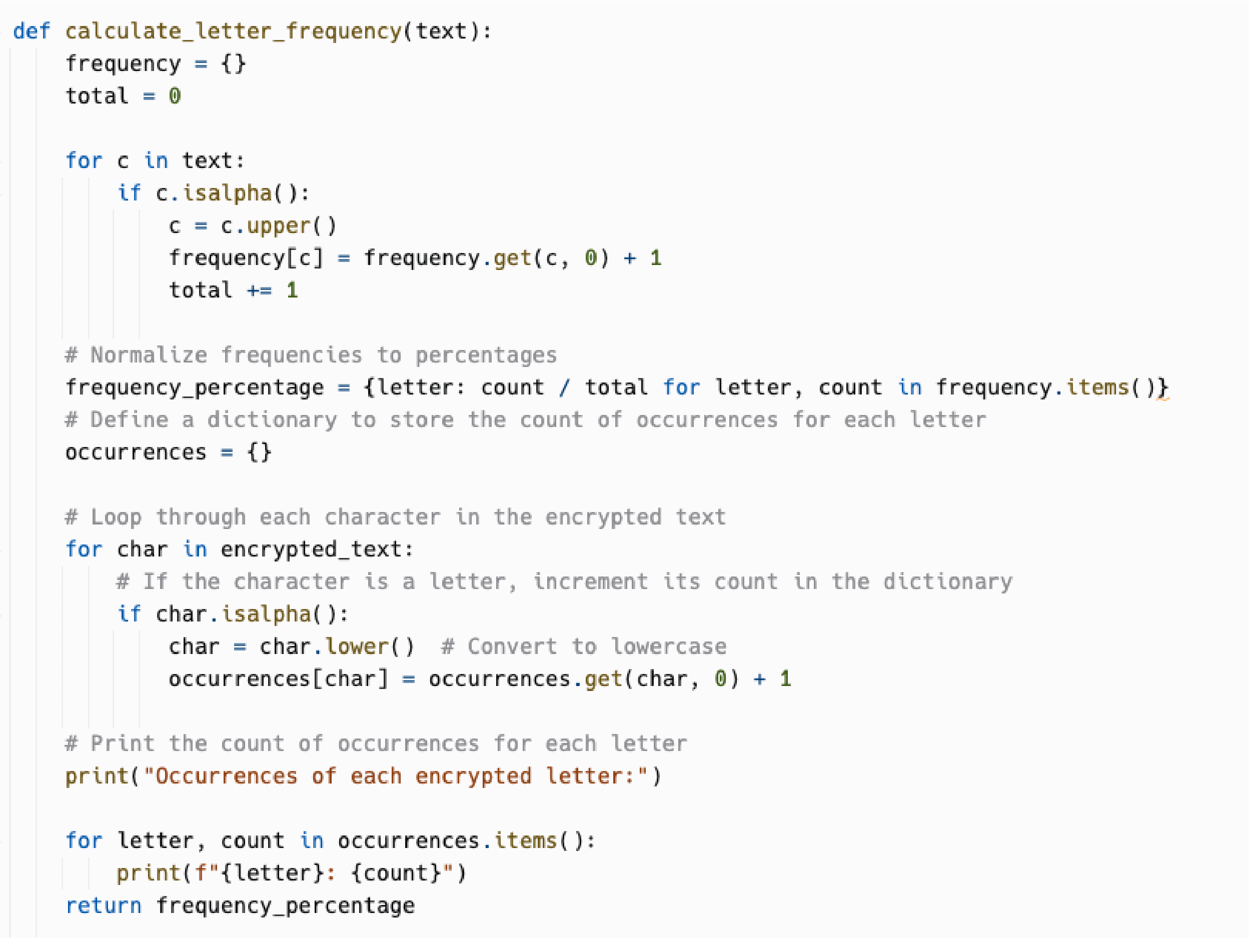


```python
import csv
# Define the encrypted text
encrypted_text = "lrvmnir bpr sumvbwvr jx bpr lmiwv yjeryrkbi jx qmbm wi bpr xjvni mkd ymibrut jx irhx wi bpr riirkvr jxymbinlmtmipw utn qmumbr dj w ipmhh but bj rhnvwdmbr bpr yjeryrkbi jx bpr qmbm mvvjudwko bj yt wkbrusurbmbwjk lmird jk xjubt trmui jx ibndtwb wi kjb mk rmit bmiq bj rashmwk rmvp yjeryrkb mkd wbi iwokwxwvmkvr mkd ijyr ynib urymwk nkrashmwkrd bj ower m vjyshrbr rashmkmbwjk jkr cjnhd pmer bj lr fnmhwxwrd mkd wkiswurd bj invp mk rabrkb bpmb pr vjnhd urmvp bpr ibmbr jx rkhwopbrkrd ywkd vmsmlhr jx urvjokwgwko ijnkdhrii ijnkd mkd ipmsrhrii ipmsr w dj kjb drry ytirhx bpr xwkmh mnbpjuwbt lnb yt rasruwrkvr cwbp qmbm pmi hrxb kj djnlb bpmb bpr xjhhjcwko wi bpr sujsru msshwvmbwjk mkd wkbrusurbmbwjk w jxxru yt bprjuwri wk bpr pjsr bpmb bpr riirkvr jx jqwkmcmk qmumbr cwhh urymwk wkbmvb"

# Define the CSV file path containing letter frequencies
csv_file_path = 'letter_table.csv'
# Print the encrypted text and the number of letters
print("Encrypted Text:", encrypted_text)
print("\nNumber of Letters:", sum(1 for c in encrypted_text if c.isalpha()))
```

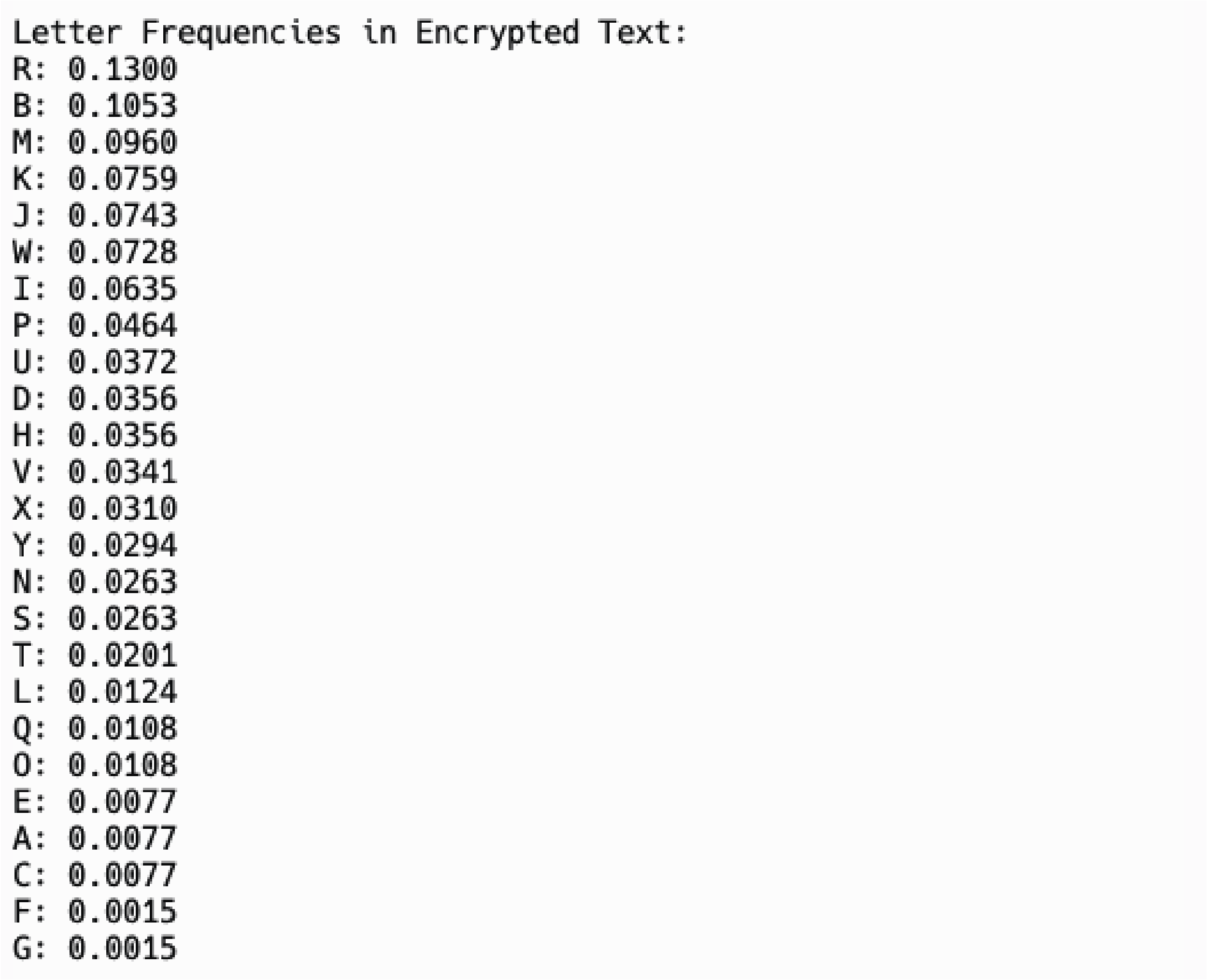


```python
with open(csv_file_path, 'r') as csv_file:
    csv_reader = csv.reader(csv_file)
    next(csv_reader)  # Skip the header row
    for row in csv_reader:
        letter = row[0].strip().upper()  # Convert letter to uppercase
        frequency = float(row[1])  # Convert frequency to float
        csv_frequencies[letter] = frequency

# Calculate letter frequencies in the encrypted text
encrypted_frequency = calculate_letter_frequency(encrypted_text)

# Sort the frequencies to find the most common letters
sorted_encrypted_frequency = sorted(encrypted_frequency.items(), key=lambda x: x[1], reverse=True)
sorted_csv_frequency = sorted(csv_frequencies.items(), key=lambda x: x[1], reverse=True)

```

2. **Analysis**: The encrypted text is analyzed to determine the frequency of occurrence for each letter.
3. **Frequency Calculation**: The frequency of each letter in the encrypted text is calculated and normalized to percentages.

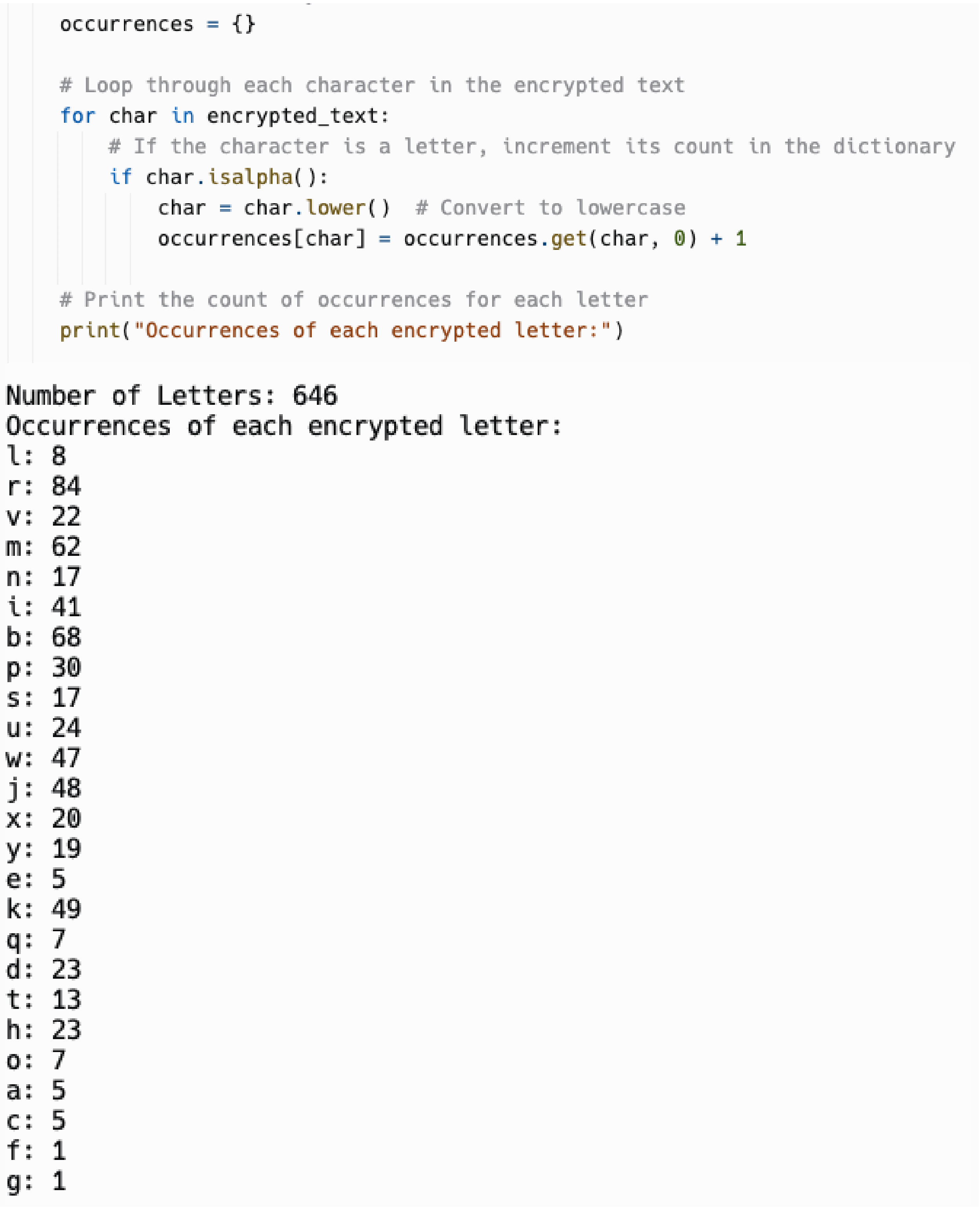


```python
def calculate_letter_frequency(text):
    frequency = {}
    total = 0

    for c in text:
        if c.isalpha():
            c = c.upper()
            frequency[c] = frequency.get(c, 0) + 1
            total += 1

    # Normalize frequencies to percentages
    frequency_percentage = {letter: count / total for letter, count in frequency.items()}
    # Define a dictionary to store the count of occurrences for each letter
    occurrences = {}

    # Loop through each character in the encrypted text
    for char in encrypted_text:
        # If the character is a letter, increment its count in the dictionary
        if char.isalpha():
            char = char.lower()  # Convert to lowercase
            occurrences[char] = occurrences.get(char, 0) + 1

    # Print the count of occurrences for each letter
    print("Occurrences of each encrypted letter:")

    for letter, count in occurrences.items():
        print(f"{letter}: {count}")
    return frequency_percentage
```



```
Letter Frequencies in Encrypted Text:
R: 0.1300
B: 0.1053
M: 0.0960
K: 0.0759
J: 0.0743
W: 0.0728
I: 0.0635
P: 0.0464
U: 0.0372
D: 0.0356
H: 0.0356
V: 0.0341
X: 0.0310
Y: 0.0294
N: 0.0263
S: 0.0263
T: 0.0201
L: 0.0124
Q: 0.0108
O: 0.0108
E: 0.0077
A: 0.0077
C: 0.0077
F: 0.0015
G: 0.0015
```

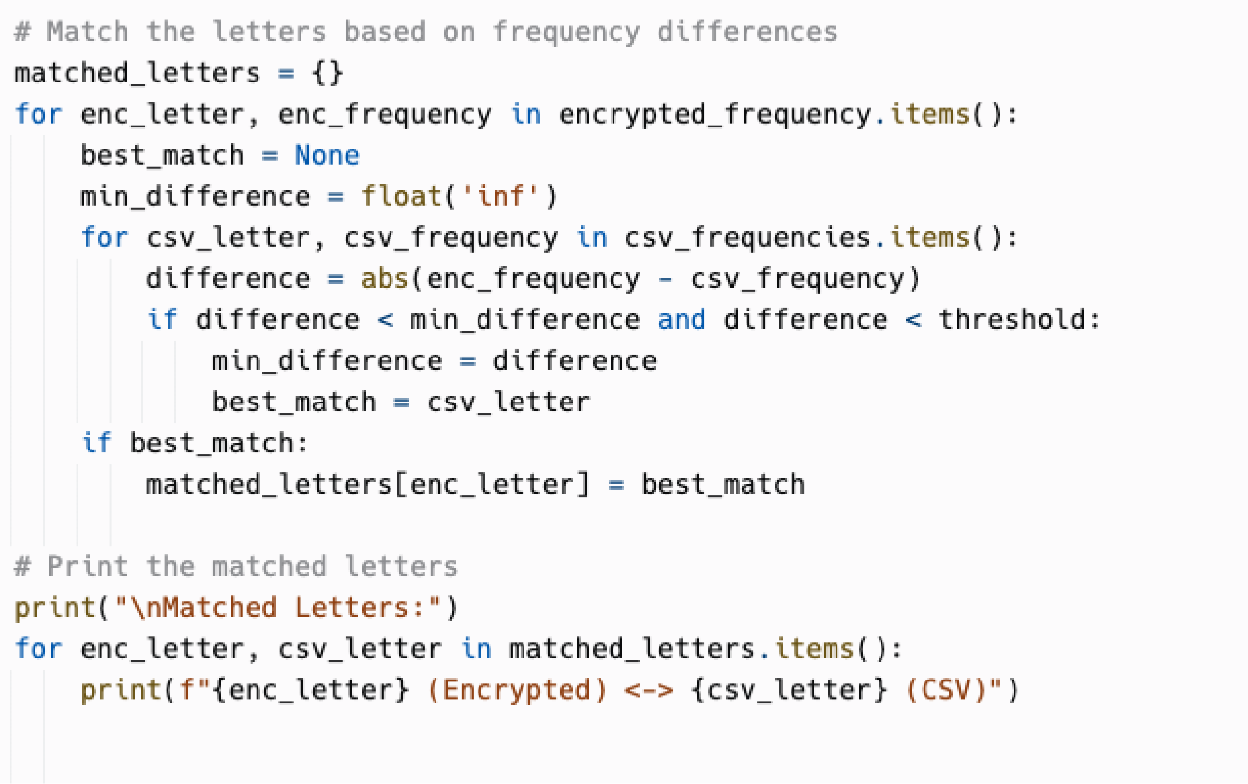
4. **Occurrences**: The count of occurrences for each encrypted letter is computed and printed.



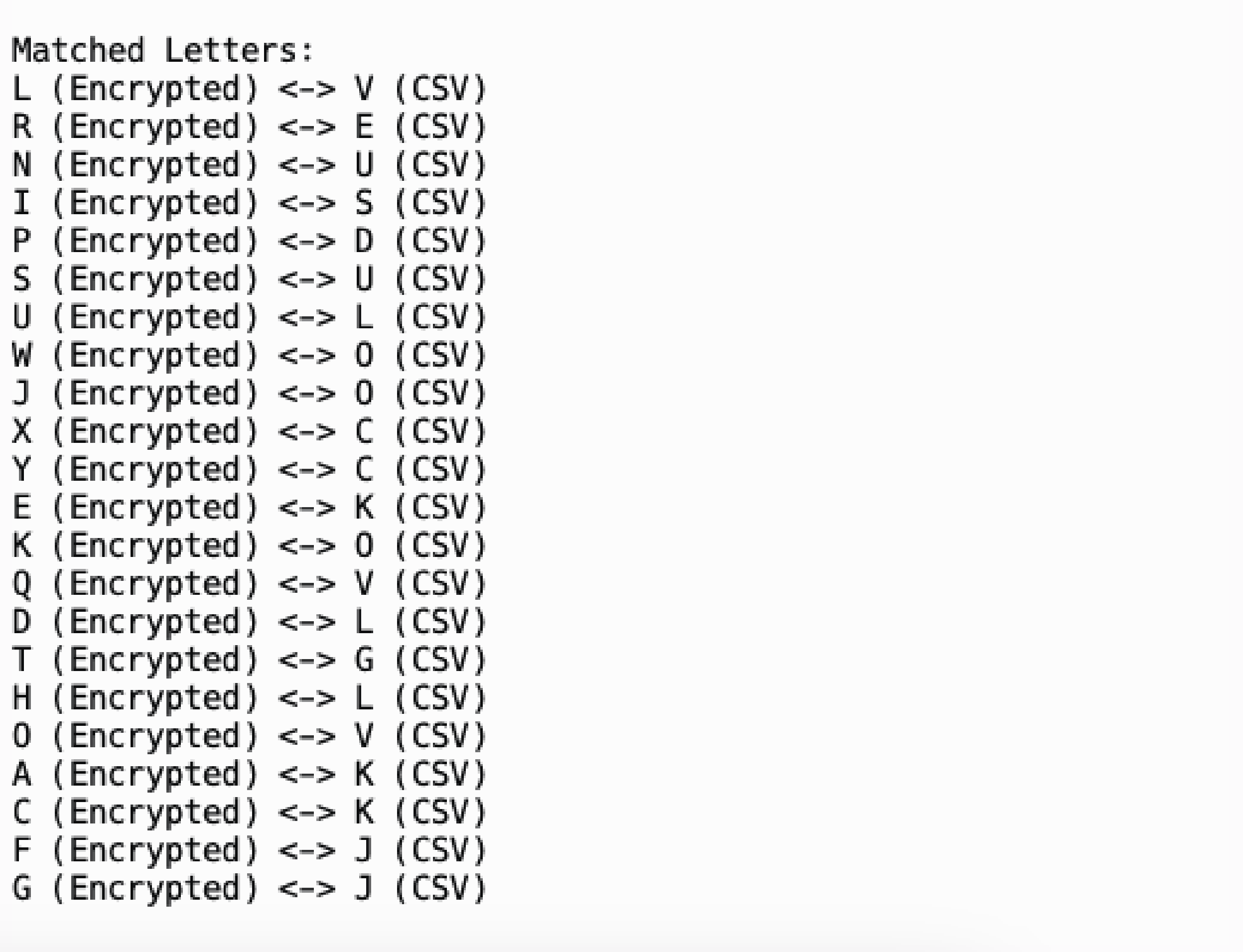
```python
    occurrences = {}

    # Loop through each character in the encrypted text
    for char in encrypted_text:
        # If the character is a letter, increment its count in the dictionary
        if char.isalpha():
            char = char.lower()  # Convert to lowercase
            occurrences[char] = occurrences.get(char, 0) + 1

    # Print the count of occurrences for each letter
    print("Occurrences of each encrypted letter:")
```

```
Number of Letters: 646
Occurrences of each encrypted letter:
l: 8
r: 84
v: 22
m: 62
n: 17
i: 41
b: 68
p: 30
s: 17
u: 24
w: 47
j: 48
x: 20
y: 19
e: 5
k: 49
q: 7
d: 23
t: 13
h: 23
o: 7
a: 5
c: 5
f: 1
g: 1
```

5. **Frequency Matching**: The frequencies of letters in the encrypted text are compared with those in the CSV file. A threshold is used to determine approximate matches.

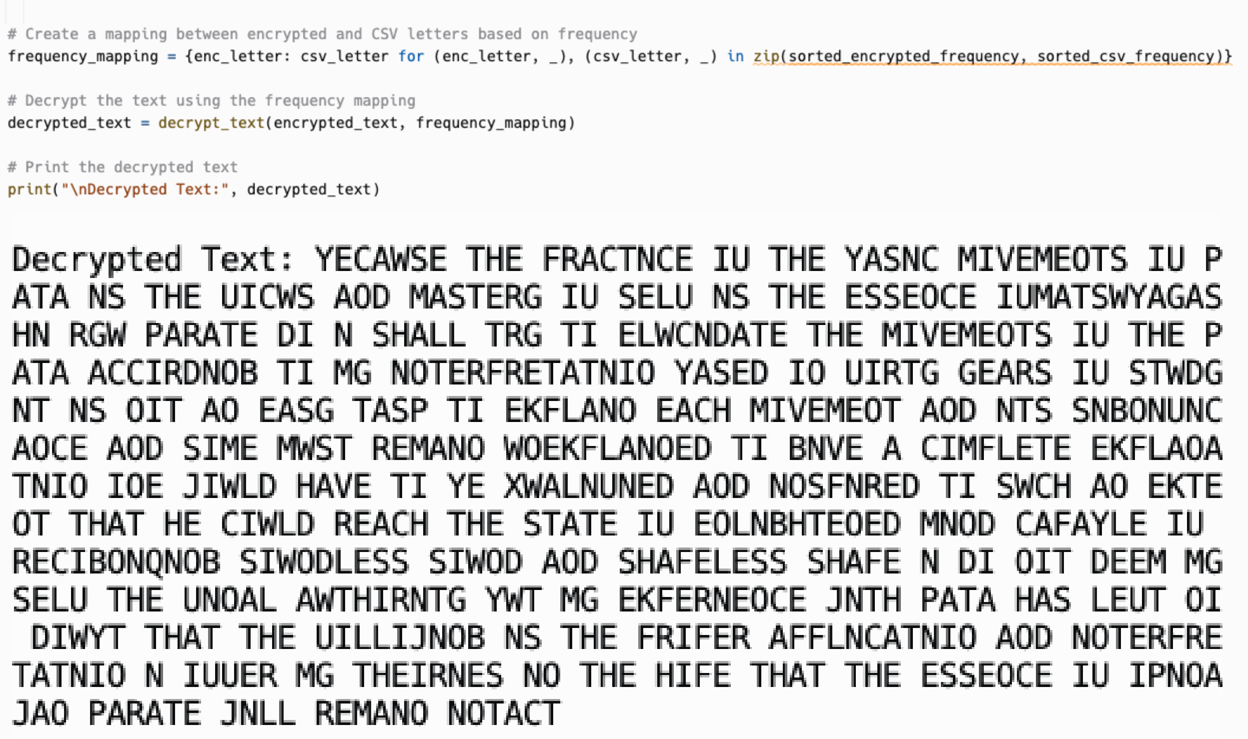


```python
# Match the letters based on frequency differences
matched_letters = {}
for enc_letter, enc_frequency in encrypted_frequency.items():
    best_match = None
    min_difference = float('inf')
    for csv_letter, csv_frequency in csv_frequencies.items():
        difference = abs(enc_frequency - csv_frequency)
        if difference < min_difference and difference < threshold:
            min_difference = difference
            best_match = csv_letter
    if best_match:
        matched_letters[enc_letter] = best_match

# Print the matched letters
print("\nMatched Letters:")
for enc_letter, csv_letter in matched_letters.items():
    print(f"{enc_letter} (Encrypted) <-> {csv_letter} (CSV)")
```



```
Matched Letters:
L (Encrypted) <-> V (CSV)
R (Encrypted) <-> E (CSV)
N (Encrypted) <-> U (CSV)
I (Encrypted) <-> S (CSV)
P (Encrypted) <-> D (CSV)
S (Encrypted) <-> U (CSV)
U (Encrypted) <-> L (CSV)
W (Encrypted) <-> O (CSV)
J (Encrypted) <-> O (CSV)
X (Encrypted) <-> C (CSV)
Y (Encrypted) <-> C (CSV)
E (Encrypted) <-> K (CSV)
K (Encrypted) <-> O (CSV)
Q (Encrypted) <-> V (CSV)
D (Encrypted) <-> L (CSV)
T (Encrypted) <-> G (CSV)
H (Encrypted) <-> L (CSV)
O (Encrypted) <-> V (CSV)
A (Encrypted) <-> K (CSV)
C (Encrypted) <-> K (CSV)
F (Encrypted) <-> J (CSV)
G (Encrypted) <-> J (CSV)
```

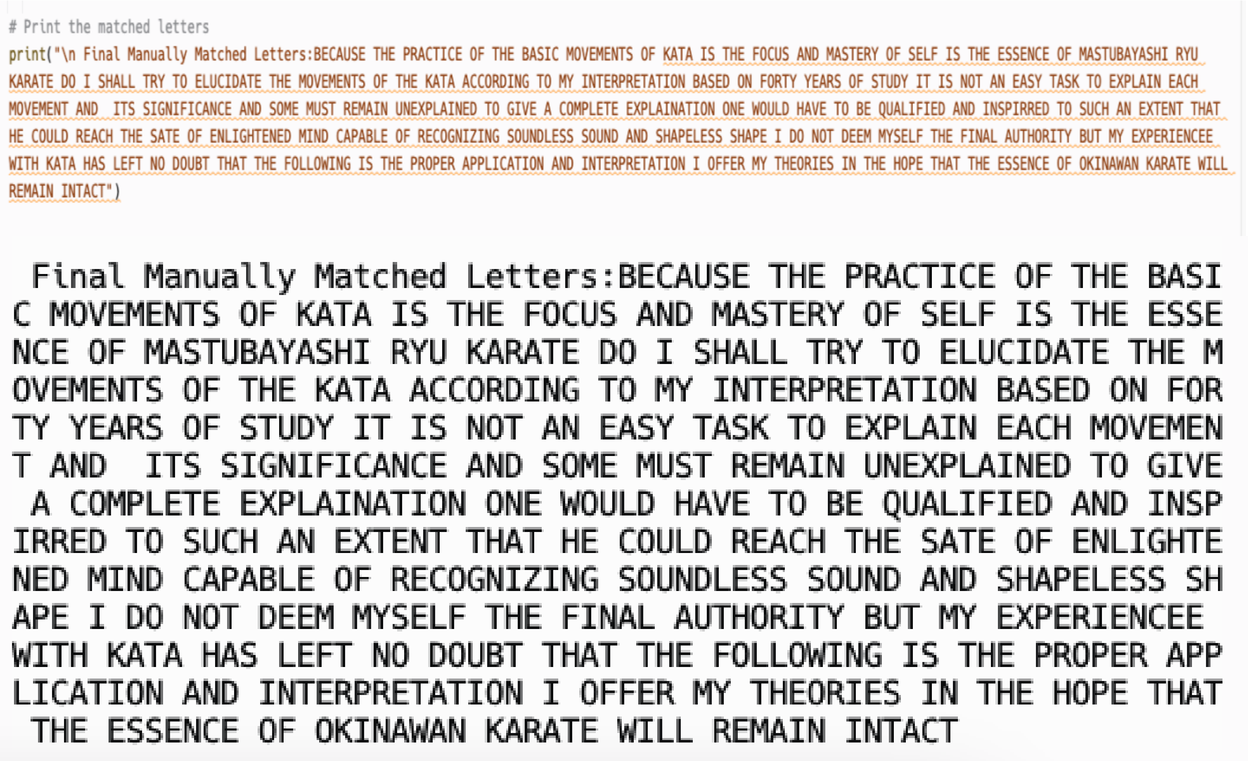
6. **Decryption**: The letters from the encrypted text are decrypted using the frequency mapping obtained from the matching process.



```
# Create a mapping between encrypted and CSV letters based on frequency
frequency_mapping = {enc_letter: csv_letter for (enc_letter, _), (csv_letter, _) in zip(sorted_encrypted_frequency, sorted_csv_frequency)}

# Decrypt the text using the frequency mapping
decrypted_text = decrypt_text(encrypted_text, frequency_mapping)

# Print the decrypted text
print("\nDecrypted Text:", decrypted_text)
```

```
Decrypted Text: YECAWSE THE FRACTNCE IU THE YASNC MIVEMEOTS IU P
ATA NS THE UICWS AOD MASTERG IU SELU NS THE ESSEOCE IUMATSWYAGAS
HN RGW PARATE DI N SHALL TRG TI ELWCNDATE THE MIVEMEOTS IU THE P
ATA ACCIRDNOB TI MG NOTERFRETATNIO YASED IO UIRTG GEARS IU STWDG
NT NS OIT AO EASG TASP TI EKFLANO EACH MIVEMEOT AOD NTS SNBONUNC
AOCE AOD SIME MWST REMANO WOEKFLANOED TI BNVE A CIMFLETE EKFLAOA
TNIO IOE JIWLD HAVE TI YE XWALNUNED AOD NOSFNRED TI SWCH AO EKTE
OT THAT HE CIWLD REACH THE STATE IU EOLNBHTEOED MNOD CAFAYLE IU
RECIBONQNOB SIWODLESS SIWOD AOD SHAFELESS SHAFE N DI OIT DEEM MG
SELU THE UNOAL AWTHIRNTG YWT MG EKFERNEOCE JNTH PATA HAS LEUT OI
 DIWYT THAT THE UILLIJNOB NS THE FRIFER AFFLNCATNIO AOD NOTERFRE
TATNIO N IUUER MG THEIRNES NO THE HIFE THAT THE ESSEOCE IU IPNOA
JAO PARATE JNLL REMANO NOTACT
```

7. **Output**: The decrypted text is printed along with the matched letters between the encrypted text and the CSV file.



```
# Print the matched letters
print("\n Final Manually Matched Letters:BECAUSE THE PRACTICE OF THE BASIC MOVEMENTS OF KATA IS THE FOCUS AND MASTERY OF SELF IS THE ESSENCE OF MASTUBAYASHI RYU
KARATE DO I SHALL TRY TO ELUCIDATE THE MOVEMENTS OF THE KATA ACCORDING TO MY INTERPRETATION BASED ON FORTY YEARS OF STUDY IT IS NOT AN EASY TASK TO EXPLAIN EACH
MOVEMENT AND  ITS SIGNIFICANCE AND SOME MUST REMAIN UNEXPLAINED TO GIVE A COMPLETE EXPLAINATION ONE WOULD HAVE TO BE QUALIFIED AND INSPIRRED TO SUCH AN EXTENT THAT
HE COULD REACH THE SATE OF ENLIGHTENED MIND CAPABLE OF RECOGNIZING SOUNDLESS SOUND AND SHAPELESS SHAPE I DO NOT DEEM MYSELF THE FINAL AUTHORITY BUT MY EXPERIENCEE
WITH KATA HAS LEFT NO DOUBT THAT THE FOLLOWING IS THE PROPER APPLICATION AND INTERPRETATION I OFFER MY THEORIES IN THE HOPE THAT THE ESSENCE OF OKINAWAN KARATE WILL
REMAIN INTACT")
```

```
 Final Manually Matched Letters:BECAUSE THE PRACTICE OF THE BASI
C MOVEMENTS OF KATA IS THE FOCUS AND MASTERY OF SELF IS THE ESSE
NCE OF MASTUBAYASHI RYU KARATE DO I SHALL TRY TO ELUCIDATE THE M
OVEMENTS OF THE KATA ACCORDING TO MY INTERPRETATION BASED ON FOR
TY YEARS OF STUDY IT IS NOT AN EASY TASK TO EXPLAIN EACH MOVEMEN
T AND  ITS SIGNIFICANCE AND SOME MUST REMAIN UNEXPLAINED TO GIVE
 A COMPLETE EXPLAINATION ONE WOULD HAVE TO BE QUALIFIED AND INSP
IRRED TO SUCH AN EXTENT THAT HE COULD REACH THE SATE OF ENLIGHTE
NED MIND CAPABLE OF RECOGNIZING SOUNDLESS SOUND AND SHAPELESS SH
APE I DO NOT DEEM MYSELF THE FINAL AUTHORITY BUT MY EXPERIENCEE
WITH KATA HAS LEFT NO DOUBT THAT THE FOLLOWING IS THE PROPER APP
LICATION AND INTERPRETATION I OFFER MY THEORIES IN THE HOPE THAT
 THE ESSENCE OF OKINAWAN KARATE WILL REMAIN INTACT
```

Results:

- The encrypted text is successfully decrypted using the English frequency letter table.

- The occurrences of each encrypted letter are determined, providing insight into the distribution of letters in the text.
- Matched letters between the encrypted text and the CSV file are identified based on frequency similarities.
- The final decrypted text is presented, revealing the original message hidden in the encrypted text.

Conclusion: Through the utilization of frequency analysis and matching techniques, the encrypted text has been deciphered, demonstrating the effectiveness of cryptographic analysis methods. The project showcases the application of frequency-based decryption algorithms in cryptanalysis and highlights the importance of frequency tables in deciphering encrypted messages.