Aim:

Decipher
“KUHPVIBQKVOSHWHXBPOFUXHRPVLLDDWVOSKWPREDDVVIDWQRBHBGLLBBPKQUNRVOHQEIRLWOKKRDD” using a combination of columnar transposition and simple shift substitution.

Solution:

After decryption, the plain text that I got, with the help of key of length 7, is: “BE HAPPY FOR THE MOMENT THIS MOMENT IS YOUR LIFE BY KHAY YAMOH AND ALSO THIS CLASS IS REALLY”, which might be incomplete but among other decrypted messages, this made the most sense to me. I used python for coding language and VS Code for IDE.

Method:

Step 1:

Imported necessary libraries-

os provides a way of interacting with the operating system, including changing the current working directory, using os I changed the directory to the one where the dictionary was situated.

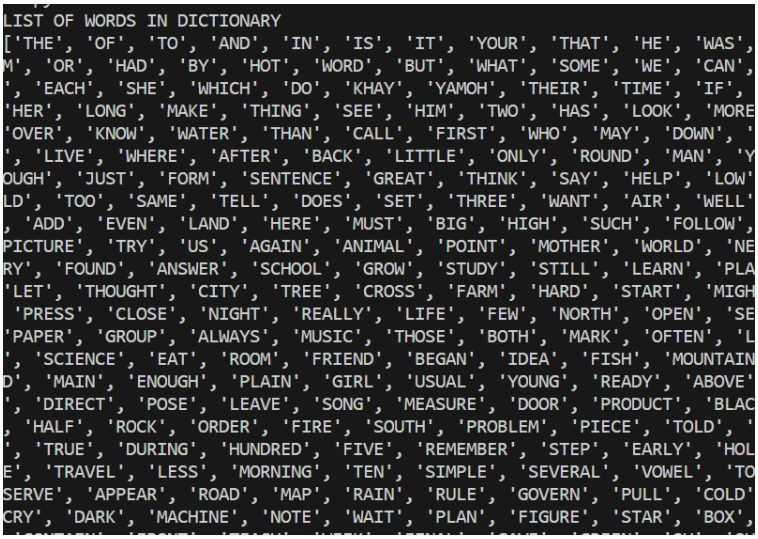
defaultdict to create a dictionary-like object with a default factory function for handling missing keys. Since it was given in the problem statement that the deciphered messages contains of only alphabets, I defined a variable that consisted of every alphabets in chronological order in upper case.

Step 2:

Defining necessary functions-

read_dictionary(list_of_words) [1]

- to retrieve all the words in the dictionary where list of words is an empty in which the read words are stored after converting them into uppercase and eliminating the white spaces in between.

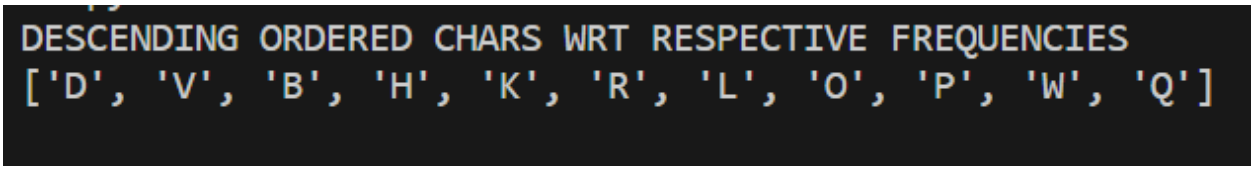


most_frequent_char(cipher_text) [2]

- to find and store all the characters n cipher_text with frequency > 3.

odered_freq_chars(freq) [2]

- to sort the characters stored from the function above in a descending order with regards to their frequency.



no_of_shifts(ordered_chars) [3]

- to find the potential number of shifts taking into consideration most common English characters [4]

```
POTENTIAL NUMBER OF SHIFTS FOR SIMPLE SHIFT SUBSTITUTION
{1, 2, 3}
```

simple_shift_decipher(shift, cipher_text) [5]

- to perform simple shift decryption with respects to the list of potential shifts obtained from above function.

```
POTENTIAL STRINGS AFTER SIMPLE SHIFT DECRYPTION
{'HREMSFYNHSLPETEUYMLCRUEOMSIIAATSLPHTMOBAASSFATNOYEYDIIYYMHNRKOSLENBFOITLHHOAA', 'ISFNTGZOITMQFUFVZNMDSVFPNTJJBBUTMQIUNPCBBTTGBUOPZFZEJJZZNIOSLPTMFOCGPJUMIIPBB',
 'JTGOUHAPJUNRGVGWAONETWGQOUKKCCVUNRJVOQDCCUUHCVPQAGAFKKAAOJPTMQUNGPDHQKVNJJQCC'}
```

permute(s) [6]

- to generate potential keys for columnar transposition by permuting characters in ALPHABETS with respect to key length ranging from 1 to 10.

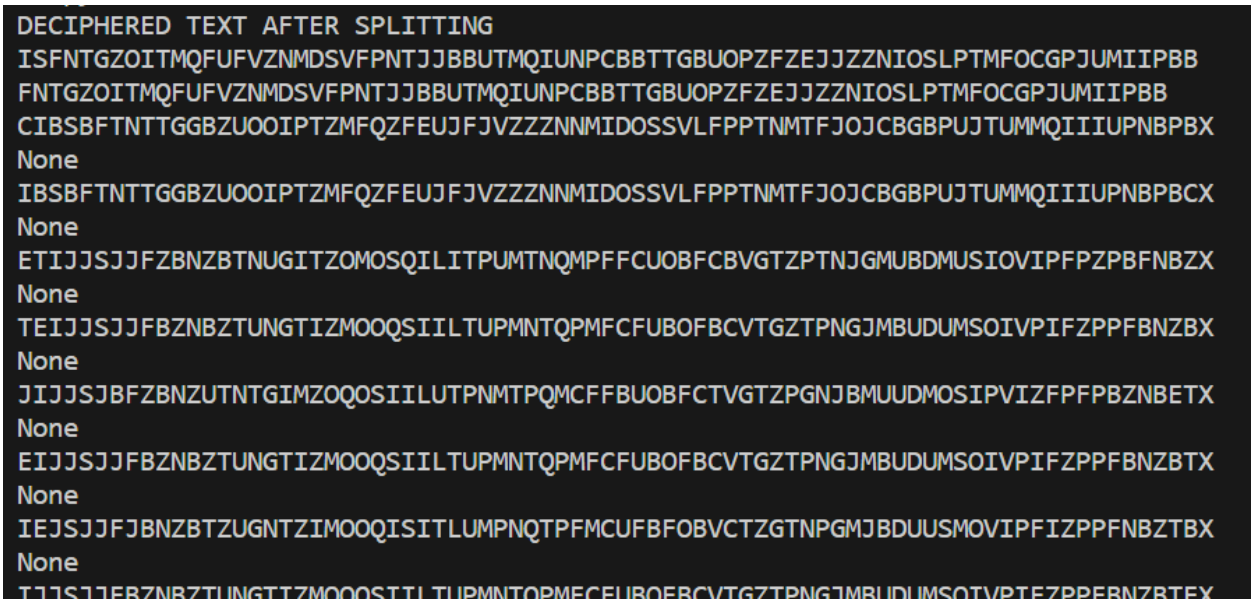
```
POTENTIAL KEYS FOR COLUMNAR TRANSPOSITION CIPHER
['A', 'BA', 'AB', 'CBA', 'BCA', 'BAC', 'CAB', 'ACB', 'ABC', 'DCBA', 'CDBA',
, 'CDAB', 'CADB', 'CABD', 'DACB', 'ADCB', 'ACDB', 'ACBD', 'DABC', 'ADBC', '
'CDBEA', 'CDBAE', 'ECBDA', 'CEBDA', 'CBEDA', 'CBDEA', 'CBDAE', 'ECBAD', 'CE
'BEDCA', 'BDECA', 'BDCEA', 'BDCAE', 'EBCDA', 'BECDA', 'BCEDA', 'BCDEA', 'BC
'DBACE', 'EBDAC', 'BEDAC', 'BDEAC', 'BDAEC', 'BDACE', 'EBADC', 'BEADC', 'BA
'DCEAB', 'DCAEB', 'DCABE', 'ECDAB', 'CEDAB', 'CDEAB', 'CDAEB', 'CDABE', 'EC
'EDACB', 'DEACB', 'DAECB', 'DACEB', 'DACBE', 'EADCB', 'AEDCB', 'ADECB', 'AD
'ACBED', 'ACBDE', 'EDABC', 'DEABC', 'DAEBC', 'DABEC', 'DABCE', 'EADBC', 'AE
'AEBCD', 'ABECD', 'ABCED', 'ABCDE', 'FEDCBA', 'EFDCBA', 'EDFCBA', 'EDCFBA',
A', 'DFCEBA', 'DCFEBA', 'DCEFBA', 'DCEBFA', 'DCEBAF', 'FDCBEA', 'DFCBEA', '
, 'DCBAEF', 'FECDBA', 'EFCDBA', 'ECFDBA', 'ECDFBA', 'ECDBFA', 'ECDBAF', 'FC
'CDEFBA', 'CDEBFA', 'CDEBAF', 'FCDBEA', 'CFDBEA', 'CDFBEA', 'CDBFEA', 'CDBE
```

columnar_transposition_decipher(cipher_text, key) [7]

- to perform columnar transposition decryption using every possible permutation of keys gained from above function.

split_string(new_string, list_of_words) [8]

- to split the string to make a new_string containing sensible words decided by comparing the words from the dictionary.



Step 3:

Calling all the functions in respective orders and execution

After calling functions, the execution takes place by going over every possible output of decrypted message combining simple shift substitution and columnar transposition and printed the one that after splitting made the most sense or most of the words from decrypted text were present in the dictionary. The plain text obtained is “BE HAPPY FOR THE MOMENT THIS MOMENT IS YOUR LIFE BY KHAY YAMOH AND ALSO THIS CLASS IS REALLY”.

```
FINAL DECIPHERED TEXT
BE HAPPY FOR THE MOMENT THIS MOMENT IS YOUR LIFE BY KHAY YAMOH AND ALSO THIS CLASS IS REALLY
```

References:

[1] https://www.w3schools.com/python/python_file_open.asp

[2] https://stackoverflow.com/questions/4131123/finding-the-most-frequent-character-in-a-string

[3] https://stackoverflow.com/questions/53380551/python-caesar-cipher-program-with-no-key

[4] https://pi.math.cornell.edu/~mec/2003-2004/cryptography/subs/frequencies.html

[5]https://www.tutorialspoint.com/cryptography_with_python/cryptography_with_python_caesar_cipher.htm

[6] https://stackoverflow.com/questions/33312532/generate-all-permutations-of-a-string-in-python-without-using-itertools

[7] https://www.geeksforgeeks.org/columnar-transposition-cipher/

[8] https://stackoverflow.com/questions/32877531/splitting-strings-in-python-without-split