# Лаб.2 - Frequency-Based Decryption

**Єлін Євген РІ-41мн**

<https://github.com/YelinYevgen/lab-2.git>

**Код Python:**

from collections import Counter

# English letter frequency ranking for reference, with 'E' as the most common.

ENGLISH\_COMMON\_LETTER = 'E'

def find\_shift(encoded\_text):

# Filter only alphabetic characters, and make them uppercase to standardize

filtered\_text = [char.upper() for char in encoded\_text if char.isalpha()]

# Count the frequency of each letter in the filtered text

frequency = Counter(filtered\_text)

# Find the most common letter in the encoded message

most\_common\_letter, \_ = frequency.most\_common(1)[0]

# Calculate the shift needed to align the most common letter with 'E'

shift = (ord(most\_common\_letter) - ord(ENGLISH\_COMMON\_LETTER)) % 26

return shift

def caesar\_decrypt(text, shift):

decrypted\_text = []

for char in text:

if char.isalpha():

# Determine if character is uppercase or lowercase

ascii\_offset = ord('A') if char.isupper() else ord('a')

# Apply reverse Caesar shift to decode

new\_char = chr((ord(char) - ascii\_offset - shift) % 26 + ascii\_offset)

decrypted\_text.append(new\_char)

else:

# Append non-alphabetic characters without changes

decrypted\_text.append(char)

return ''.join(decrypted\_text)

# Main decoding function that manages input and output

def decode\_message(encoded\_text):

shift = find\_shift(encoded\_text)

return caesar\_decrypt(encoded\_text, shift)

# Input handling and output

encoded\_text = input().strip() # Removes unnecessary leading and trailing spaces from the input

decoded\_text = decode\_message(encoded\_text)

print(decoded\_text) # Outputs exactly the decoded message without additional spaces