import numpy as np

import tkinter as tk

from tkinter import messagebox

def letter\_to\_number(letter):

return ord(letter) - ord('A')

def number\_to\_letter(number):

return chr(number % 26 + ord('A'))

def matrix\_mod\_inv(matrix, modulus):

"""Calculate the modular inverse of a 2x2 matrix"""

det = int(np.round(np.linalg.det(matrix)))

det\_inv = pow(det % modulus, -1, modulus)

adjugate = np.array([[matrix[1, 1], -matrix[0, 1]],

[-matrix[1, 0], matrix[0, 0]]])

return (adjugate \* det\_inv) % modulus

def decrypt\_hill\_cipher():

# Known plaintext and ciphertext

plaintext = "MEET"

ciphertext = "URRG"

# Convert plaintext and ciphertext to number matrices

P = np.array([[letter\_to\_number(plaintext[0]), letter\_to\_number(plaintext[2])],

[letter\_to\_number(plaintext[1]), letter\_to\_number(plaintext[3])]])

C = np.array([[letter\_to\_number(ciphertext[0]), letter\_to\_number(ciphertext[2])],

[letter\_to\_number(ciphertext[1]), letter\_to\_number(ciphertext[3])]])

# Calculate the key matrix

try:

P\_inv = matrix\_mod\_inv(P, 26)

K = (C @ P\_inv) % 26

# Verify the key by encrypting the plaintext

encrypted = encrypt(plaintext, K)

# Display results in messagebox

if encrypted == ciphertext:

messagebox.showinfo("Success", f"Recovered Key Matrix:\n{K}\n\n"

f"The recovered key correctly encrypts the plaintext to the known ciphertext.")

else:

messagebox.showerror("Error", "The recovered key does not produce the expected ciphertext.")

except Exception as e:

messagebox.showerror("Error", f"An error occurred: {e}")

def encrypt(plaintext, key):

result = ""

for i in range(0, len(plaintext), 2):

chunk = np.array([letter\_to\_number(plaintext[i]), letter\_to\_number(plaintext[i+1])])

encrypted = (key @ chunk) % 26

result += number\_to\_letter(encrypted[0]) + number\_to\_letter(encrypted[1])

return result

# Set up the GUI

root = tk.Tk()

root.title("Hill Cipher Breaker")

welcome\_label = tk.Label(root, text="Hill Cipher Breaker")

welcome\_label.pack(pady=10)

instructions = tk.Label(root, text="Click 'Analyze' to find the key matrix.")

instructions.pack(pady=5)

analyze\_button = tk.Button(root, text="Analyze", command=decrypt\_hill\_cipher)

analyze\_button.pack(pady=20)

root.mainloop()