Nama : Muhammad Rafif Ramadhansyah

Program Studi : Teknik Informatika - S1

NIM : 4611422008

Mata Kuliah : Kriptografi

Dosen Pengampu : Dr. Alamsyah, S.Si., M.Kom.

# 1. Source Code Program

import tkinter as tk  
from tkinter import filedialog, messagebox  
import numpy as np  
  
class CipherGUI:  
 def \_\_init\_\_(self, master):  
 self.master = master  
 master.title("Program Cipher Desktop")  
 master.geometry("600x400")  
  
 self.cipher\_types = ["Vigenere Cipher", "Playfair Cipher", "Hill Cipher"]  
 self.current\_cipher = tk.StringVar()  
 self.current\_cipher.set(self.cipher\_types[0])  
  
 self.create\_widgets()  
  
 def create\_widgets(self):  
 tk.Label(self.master, text="Pilih Jenis Cipher:").grid(row=0, column=0, sticky="w", padx=10, pady=5)  
 tk.OptionMenu(self.master, self.current\_cipher, \*self.cipher\_types, command=self.on\_cipher\_change).grid(row=0, column=1, sticky="w", padx=10, pady=5)  
  
 tk.Label(self.master, text="Masukkan Plaintext:").grid(row=1, column=0, sticky="w", padx=10, pady=5)  
 self.input\_text = tk.Text(self.master, height=5, width=50)  
 self.input\_text.grid(row=1, column=1, padx=10, pady=5)  
  
 self.key\_label = tk.Label(self.master, text="Key (minimal 12 karakter):")  
 self.key\_label.grid(row=2, column=0, sticky="w", padx=10, pady=5)  
 self.key\_entry = tk.Entry(self.master, width=50)  
 self.key\_entry.grid(row=2, column=1, padx=10, pady=5)  
  
 self.matrix\_frame = tk.Frame(self.master)  
 self.matrix\_frame.grid(row=3, column=1, padx=10, pady=5)  
 self.matrix\_entries = []  
 for i in range(2):  
 row\_entries = []  
 for j in range(2):  
 entry = tk.Entry(self.matrix\_frame, width=5)  
 entry.grid(row=i, column=j, padx=2, pady=2)  
 row\_entries.append(entry)  
 self.matrix\_entries.append(row\_entries)  
 self.matrix\_frame.grid\_remove()  
  
 tk.Button(self.master, text="Unggah File", command=self.upload\_file).grid(row=4, column=0, sticky="w", padx=10, pady=5)  
 tk.Button(self.master, text="Enkripsi", command=self.encrypt).grid(row=4, column=1, sticky="w", padx=10, pady=5)  
 tk.Button(self.master, text="Dekripsi", command=self.decrypt).grid(row=4, column=1, sticky="e", padx=10, pady=5)  
  
 tk.Label(self.master, text="Hasil:").grid(row=5, column=0, sticky="w", padx=10, pady=5)  
 self.output\_text = tk.Text(self.master, height=5, width=50)  
 self.output\_text.grid(row=5, column=1, padx=10, pady=5)  
  
 def on\_cipher\_change(self, \*args):  
 if self.current\_cipher.get() == "Hill Cipher":  
 self.key\_label.config(text="Matrix Key (2x2):")  
 self.key\_entry.grid\_remove()  
 self.matrix\_frame.grid()  
 else:  
 self.key\_label.config(text="Key (minimal 12 karakter):")  
 self.key\_entry.grid()  
 self.matrix\_frame.grid\_remove()  
  
 def upload\_file(self):  
 file\_path = filedialog.askopenfilename(filetypes=[("Text files", "\*.txt")])  
 if file\_path:  
 with open(file\_path, 'r') as file:  
 content = file.read()  
 self.input\_text.delete(1.0, tk.END)  
 self.input\_text.insert(tk.END, content)  
  
 def get\_hill\_matrix(self):  
 return [[int(self.matrix\_entries[i][j].get()) for j in range(2)] for i in range(2)]  
  
 def encrypt(self):  
 plaintext = self.input\_text.get(1.0, tk.END).strip()  
 cipher\_type = self.current\_cipher.get()  
   
 if cipher\_type == "Hill Cipher":  
 key = self.get\_hill\_matrix()  
 else:  
 key = self.key\_entry.get()  
 if len(key) < 12:  
 messagebox.showerror("Error", "Key setidaknya harus memiliki panjang 12 karakter.")  
 return  
  
 if cipher\_type == "Vigenere Cipher":  
 ciphertext = self.vigenere\_cipher(plaintext, key, mode='encrypt')  
 elif cipher\_type == "Playfair Cipher":  
 ciphertext = self.playfair\_cipher(plaintext, key, mode='encrypt')  
 elif cipher\_type == "Hill Cipher":  
 ciphertext = self.hill\_cipher(plaintext, key, mode='encrypt')  
  
 self.output\_text.delete(1.0, tk.END)  
 self.output\_text.insert(tk.END, ciphertext)  
  
 def decrypt(self):  
 ciphertext = self.input\_text.get(1.0, tk.END).strip()  
 cipher\_type = self.current\_cipher.get()  
   
 if cipher\_type == "Hill Cipher":  
 key = self.get\_hill\_matrix()  
 else:  
 key = self.key\_entry.get()  
 if len(key) < 12:  
 messagebox.showerror("Error", "Key setidaknya harus memiliki panjang 12 karakter.")  
 return  
  
 if cipher\_type == "Vigenere Cipher":  
 plaintext = self.vigenere\_cipher(ciphertext, key, mode='decrypt')  
 elif cipher\_type == "Playfair Cipher":  
 plaintext = self.playfair\_cipher(ciphertext, key, mode='decrypt')  
 elif cipher\_type == "Hill Cipher":  
 plaintext = self.hill\_cipher(ciphertext, key, mode='decrypt')  
  
 self.output\_text.delete(1.0, tk.END)  
 self.output\_text.insert(tk.END, plaintext)  
  
 def vigenere\_cipher(self, text, key, mode='encrypt'):  
 result = []  
 key\_length = len(key)  
 text = ''.join(filter(str.isalpha, text.upper()))  
 key = ''.join(filter(str.isalpha, key.upper()))  
  
 for i, char in enumerate(text):  
 key\_char = key[i % key\_length]  
 if mode == 'encrypt':  
 result.append(chr((ord(char) + ord(key\_char) - 2 \* ord('A')) % 26 + ord('A')))  
 else:  
 result.append(chr((ord(char) - ord(key\_char) + 26) % 26 + ord('A')))  
  
 return ''.join(result)  
  
 def playfair\_cipher(self, text, key, mode='encrypt'):  
 def create\_matrix(key):  
 alphabet = 'ABCDEFGHIKLMNOPQRSTUVWXYZ'  
 key = ''.join(dict.fromkeys(key.upper().replace('J', 'I') + alphabet))  
 return [list(key[i:i+5]) for i in range(0, 25, 5)]  
  
 def find\_position(matrix, char):  
 for i, row in enumerate(matrix):  
 if char in row:  
 return i, row.index(char)  
 return None  
  
 def prepare\_text(text):  
 text = ''.join(filter(str.isalpha, text.upper().replace('J', 'I')))  
 prepared = []  
 i = 0  
 while i < len(text):  
 if i == len(text) - 1 or text[i] == text[i+1]:  
 prepared.append(text[i] + 'X')  
 i += 1  
 else:  
 prepared.append(text[i:i+2])  
 i += 2  
 return prepared  
  
 matrix = create\_matrix(key)  
 text\_pairs = prepare\_text(text)  
 result = []  
  
 for pair in text\_pairs:  
 char1, char2 = pair  
 row1, col1 = find\_position(matrix, char1)  
 row2, col2 = find\_position(matrix, char2)  
  
 if row1 == row2:  
 if mode == 'encrypt':  
 result.append(matrix[row1][(col1+1)%5] + matrix[row2][(col2+1)%5])  
 else:  
 result.append(matrix[row1][(col1-1)%5] + matrix[row2][(col2-1)%5])  
 elif col1 == col2:  
 if mode == 'encrypt':  
 result.append(matrix[(row1+1)%5][col1] + matrix[(row2+1)%5][col2])  
 else:  
 result.append(matrix[(row1-1)%5][col1] + matrix[(row2-1)%5][col2])  
 else:  
 result.append(matrix[row1][col2] + matrix[row2][col1])  
  
 return ' '.join(result)  
  
 def hill\_cipher(self, text, key, mode='encrypt'):  
 def matrix\_mod\_inv(matrix, modulus):  
 det = int(np.round(np.linalg.det(matrix)))  
 det\_inv = pow(det % modulus, -1, modulus)  
 adjoint = np.array([[matrix[1,1], -matrix[0,1]],  
 [-matrix[1,0], matrix[0,0]]])  
 return (det\_inv \* adjoint % modulus).astype(int)  
  
 key\_matrix = np.array(key)  
 text = ''.join(filter(str.isalpha, text.upper()))  
 if len(text) % 2 != 0:  
 text += 'X'  
  
 result = []  
 for i in range(0, len(text), 2):  
 pair = np.array([ord(text[i]) - 65, ord(text[i+1]) - 65])  
 if mode == 'encrypt':  
 encrypted = np.dot(key\_matrix, pair) % 26  
 else:  
 inv\_key = matrix\_mod\_inv(key\_matrix, 26)  
 encrypted = np.dot(inv\_key, pair) % 26  
 result.extend([chr(int(c) + 65) for c in encrypted])  
  
 return ''.join(result)  
  
root = tk.Tk()  
gui = CipherGUI(root)  
root.mainloop()

# 2. Deskripsi Singkat Program

Program ini merupakan sebuah aplikasi GUI desktop yang dibangun menggunakan Python dan Tkinter. Program ini memungkinkan pengguna untuk mengenkripsi dan mendekripsi teks menggunakan tiga jenis cipher: Vigenere Cipher, Playfair Cipher, dan Hill Cipher. Pengguna dapat memasukkan plaintext, memilih jenis cipher, dan mendapatkan ciphertext yang sesuai.

# 3. Tampilan Antarmuka Program

GUI Metode Vigenere Cipher

A screenshot of a computer

Description automatically generated

GUI Metode Playfair Cipher

A screenshot of a computer

Description automatically generated

GUI Metode Hill Cipher

A screenshot of a computer

Description automatically generated

# 4. Link GitHub

GitHub Repository: https://github.com/raaapiiip/program-cipher-kriptografi.git