

TUGAS 1

KEAMANAN KOMPUTER/KRIPTOGRAFI



DISUSUN OLEH :

NAMA : SITTI ROHANI

NIM : 222061

KELAS : 5TKKO-G

PRODI : TEKNIK INFORMATIKA

UNIVERSITAS DIPA MAKASSAR

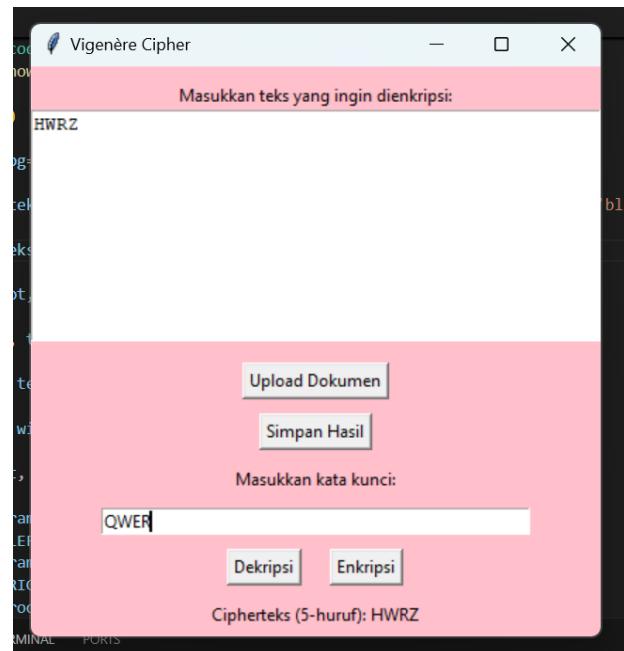
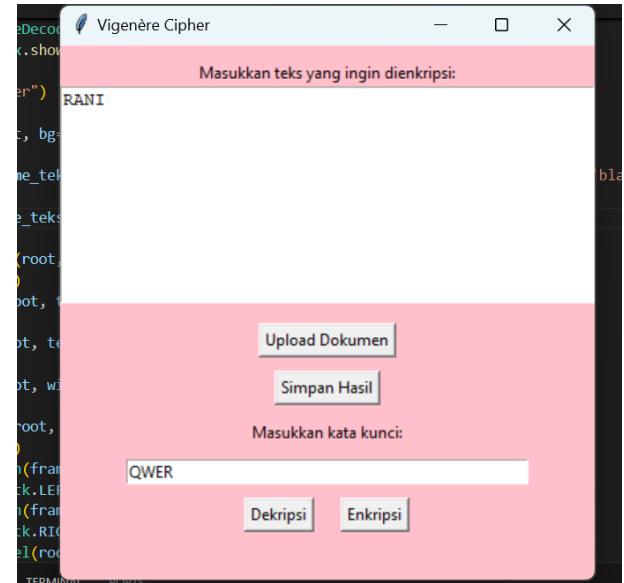
2024/2025

## LAPORAN

No.	Spesifikasi	Berhasil (✓)	Kurang Berhasil (✗)	Keterangan
1	Vigenere Cipher	✓		
2	Extended Vigenere Cipher	✓		
3	Playfair Cipher	✓		
4	Enigma Cipher	✓		
5	One-Time pad	✓		

## 1. VIGENÈRE CIPHER

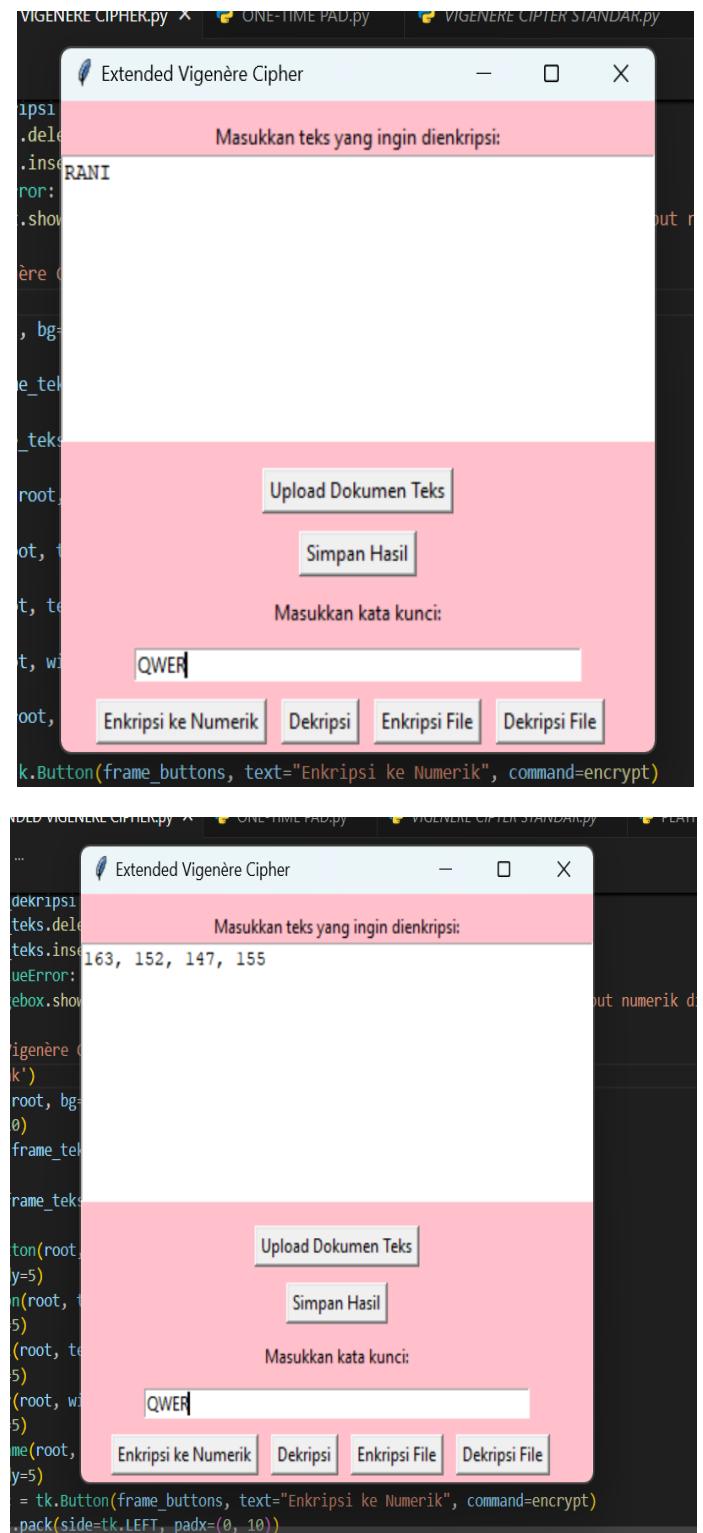
```
1 import tkinter as tk
2 from tkinter import messagebox, filedialog
3 import os
4
5 def vigenere_encrypt(teks_asli, kata_kunci):
6     teks_enkripsi = ""
7     kata_kunci_diulang = (kata_kunci * (len(teks_asli) // len(kata_kunci) + 1))[:len(teks_asli)]
8     for t_char, k_char in zip(teks_asli, kata_kunci_diulang):
9         if t_char.isalpha():
10             shift = (ord(t_char.upper()) - ord('A')) + ord(k_char.upper()) - ord('A') % 26
11             huruf_enkripsi = chr(shift + ord('A'))
12             teks_enkripsi += huruf_enkripsi
13         else:
14             teks_enkripsi += t_char
15     return teks_enkripsi
16 def vigenere_decrypt(teks_enkripsi, kata_kunci):
17     teks_dekripsi = ""
18     kata_kunci_diulang = (kata_kunci * (len(teks_enkripsi) // len(kata_kunci) + 1))[:len(teks_enkripsi)]
19     for e_char, k_char in zip(teks_enkripsi, kata_kunci_diulang):
20         if e_char.isalpha():
21             shift = (ord(e_char.upper()) - ord('A') - (ord(k_char.upper()) - ord('A'))) % 26
22             huruf_dekripsi = chr(shift + ord('A'))
23             teks_dekripsi += huruf_dekripsi
24         else:
25             teks_dekripsi += e_char
26     return teks_dekripsi
27 def encrypt():
28     teks_asli = entry_teks.get("1.0", tk.END).strip()
29     kata_kunci = entry_kunci.get()
30     if not teks_asli or not kata_kunci:
31         messagebox.showwarning("Input Error", "Mohon masukkan teks dan kata kunci.")
32         return
33     hasil_enkripsi = vigenere_encrypt(teks_asli, kata_kunci)
34     entry_teks.delete("1.0", tk.END)
35     entry_teks.insert(tk.END, hasil_enkripsi)
36     label_cipherteeks.config(text=f"Cipherteeks (5-huruf): {hasil_enkripsi}")
37 def decrypt():
38     teks_enkripsi = entry_teks.get("1.0", tk.END).strip()
39     kata_kunci = entry_kunci.get()
40     if not teks_enkripsi or not kata_kunci:
41         messagebox.showwarning("Input Error", "Mohon masukkan teks enkripsi dan kata kunci.")
42         return
43     hasil_dekripsi = vigenere_decrypt(teks_enkripsi, kata_kunci)
44     entry_teks.delete("1.0", tk.END)
45     entry_teks.insert(tk.END, hasil_dekripsi)
46 def save_file():
47     file_path = filedialog.asksaveasfilename(defaultextension=".txt", filetypes=[("Text files", ".txt"), ("All files", "*.*")])
48     if file_path:
49         with open(file_path, 'w', encoding="utf-8") as file:
50             teks = entry_teks.get("1.0", tk.END)
51             file.write(teks)
52
53 def upload_file():
54     file_path = filedialog.askopenfilename(filetypes=[("All files", "*.*")])
55     if file_path:
56         with open(file_path, 'rb') as file:
57             content = file.read()
58             try:
59                 teks = content.decode('utf-8')
60                 entry_teks.delete("1.0", tk.END)
61                 entry_teks.insert(tk.END, teks)
62             except UnicodeDecodeError:
63                 messagebox.showwarning("File Error", "File tidak dapat dibaca sebagai teks.")
64
65 root = tk.TK()
66 root.title("Vigenère Cipher")
67 root.configure(bg='pink')
68 frame_teks = tk.Frame(root, bg='pink')
69 frame_teks.pack(pady=10)
70 label_teks = tk.Label(frame_teks, text="Masukkan teks yang ingin dienkripsi:", bg='pink', fg='black')
71 label_teks.pack()
72 entry_teks = tk.Entry(frame_teks, width=50, height=10)
73 entry_teks.pack()
74 button_upload = tk.Button(root, text="Upload Dokumen", command=upload_file)
75 button_upload.pack(pady=5)
76 button_save = tk.Button(root, text="Simpan Hasil", command=save_file)
77 button_save.pack(pady=5)
78 label_kunci = tk.Label(root, text="Masukkan kata kunci:", bg='pink', fg='black')
79 label_kunci.pack(pady=5)
80 entry_kunci = tk.Entry(root, width=50)
81 entry_kunci.pack(pady=5)
82 frame_buttons = tk.Frame(root, bg='pink')
83 frame_buttons.pack(pady=5)
84 button_decrypt = tk.Button(frame_buttons, text="Dekripsi", command=decrypt)
85 button_decrypt.pack(side=tk.LEFT, padx=(0, 10))
86 button_encrypt = tk.Button(frame_buttons, text="Enkripsi", command=encrypt)
87 button_encrypt.pack(side=tk.RIGHT, padx=(10, 0))
88 label_cipherteeks = tk.Label(root, text="Cipherteeks (5-huruf):", bg='pink', fg='black')
89 label_cipherteeks.pack(pady=5)
90 root.mainloop()
```



## 2. EXTENDED VIGENÈRE CIPHER

```

1 import tkinter as tk
2 from tkinter import messagebox, filedialog
3 def extended_vigenere_encrypt(teks_asli, kata_kunci):
4     teks_enkripsi = []
5     kata_kunci_diulang = (kata_kunci + (len(teks_asli) // len(kata_kunci) + 1))[:len(teks_asli)]
6     for t_char, k_char in zip(teks_asli, kata_kunci_diulang):
7         shift = (ord(t_char) + ord(k_char)) % 256
8         teks_enkripsi.append(chr(shift))
9     return ''.join(teks_enkripsi)
10 def extended_vigenere_decrypt(teks_enkripsi, kata_kunci):
11     teks_dekripsi = []
12     kata_kunci_diulang = (kata_kunci + (len(teks_enkripsi) // len(kata_kunci) + 1))[:len(teks_enkripsi)]
13     for e_char, k_char in zip(teks_enkripsi, kata_kunci_diulang):
14         shift = (e_char - ord(k_char)) % 256
15         teks_dekripsi.append(chr(shift))
16     return ''.join(teks_dekripsi)
17 def encrypt():
18     teks_asli = entry_teks.get("1.0", tk.END).strip()
19     kata_kunci = entry_kunci.get()
20     if not teks_asli or not kata_kunci:
21         messagebox.showwarning("Input Error", "Mohon masukkan teks dan kata kunci.")
22     else:
23         hasil_enkripsi = extended_vigenere_encrypt(teks_asli, kata_kunci)
24         entry_teks.delete("1.0", tk.END)
25         entry_teks.insert(tk.END, ' '.join(map(str, hasil_enkripsi)))
26 def decrypt():
27     teks_enkripsi = entry_teks.get("1.0", tk.END).strip()
28     kata_kunci = entry_kunci.get()
29     if not teks_enkripsi or not kata_kunci:
30         messagebox.showwarning("Input Error", "Mohon masukkan teks enkripsi dan kata kunci.")
31     else:
32         try:
33             teks_enkripsi_list = list(map(int, teks_enkripsi.split(' ')))
34             hasil_dekripsi = extended_vigenere_decrypt(teks_enkripsi_list, kata_kunci)
35             entry_teks.delete("1.0", tk.END)
36             entry_teks.insert(tk.END, ''.join(hasil_dekripsi))
37         except ValueError:
38             messagebox.showerror("Input Error", "Format enkripsi tidak valid. Pastikan input numerik dipisahkan dengan koma.")
39     def save_file():
40         file_path = filedialog.asksaveasfilename(defaultextension=".txt", filetypes=[("Text files", "*.txt")])
41         if file_path:
42             with open(file_path, 'w', encoding='utf-8') as file:
43                 file.write(teks_asli)
44     def upload_file():
45         file_path = filedialog.askopenfilename(filetypes=[("All files", "*.*")])
46         if file_path:
47             with open(file_path, 'r') as file:
48                 teks_asli = file.read()
49                 entry_teks.delete("1.0", tk.END)
50                 entry_teks.insert(tk.END, teks_asli.decode(errors='ignore'))
51     def encrypt_file():
52         file_path = filedialog.askopenfilename(filetypes=[("All files", "*.*")])
53         if file_path:
54             kata_kunci = entry_kunci.get()
55             if not kata_kunci:
56                 messagebox.showwarning("Input Error", "Mohon masukkan kata kunci.")
57             else:
58                 with open(file_path, 'rb') as file:
59                     data = file.read()
60                     hasil_enkripsi = extended_vigenere_encrypt(data.decode(errors='ignore'), kata_kunci)
61                     entry_teks.delete("1.0", tk.END)
62                     entry_teks.insert(tk.END, ' '.join(map(str, hasil_enkripsi)))
63     def decrypt_file():
64         file_path = filedialog.askopenfilename(filetypes=[("Text files", "*.txt")])
65         if file_path:
66             kata_kunci = entry_kunci.get()
67             if not kata_kunci:
68                 messagebox.showwarning("Input Error", "Mohon masukkan kata kunci.")
69             else:
70                 with open(file_path, 'r', encoding='utf-8') as file:
71                     teks_enkripsi = file.read().strip()
72                     entry_teks.delete("1.0", tk.END)
73                     teks_enkripsi_list = list(map(int, teks_enkripsi.split(' ')))
74                     hasil_dekripsi = extended_vigenere_decrypt(teks_enkripsi_list, kata_kunci)
75                     entry_teks.delete("1.0", tk.END)
76                     entry_teks.insert(tk.END, ''.join(hasil_dekripsi))
77     except ValueError:
78         messagebox.showerror("Input Error", "Format enkripsi tidak valid. Pastikan input numerik dipisahkan dengan koma.")
79 root = tk.Tk()
80 root.title("Extended Vigenère Cipher")
81 root.configure(bg='pink')
82 frame_teks = tk.Frame(root, bg='pink')
83 frame_teks.pack(pady=10)
84 label_teks = tk.Label(frame_teks, text="Masukkan teks yang ingin dienkripsi:", bg='pink')
85 label_teks.pack()
86 entry_teks = tk.Text(frame_teks, width=50, height=10)
87 entry_teks.pack()
88 button_upload = tk.Button(root, text="Upload Dokumen Teks", command=upload_file)
89 button_upload.pack(pady=5)
90 button_save = tk.Button(root, text="Simpan Hasil", command=save_file)
91 button_save.pack(pady=5)
92 label_kunci = tk.Label(root, text="Masukkan kata kunci:", bg='pink')
93 label_kunci.pack(pady=5)
94 entry_kunci = tk.Entry(root, width=50)
95 entry_kunci.pack(pady=5)
96 frame_buttons = tk.Frame(root, bg='pink')
97 frame_buttons.pack(pady=5)
98 button_encrypt = tk.Button(frame_buttons, text="Enkripsi ke Numerik", command=encrypt)
99 button_encrypt.pack(side=tk.LEFT, padx=(0, 10))
100 button_decrypt = tk.Button(frame_buttons, text="Dekripsi", command=decrypt)
101 button_decrypt.pack(side=tk.LEFT, padx=(0, 10))
102 button_encrypt_file = tk.Button(frame_buttons, text="Enkripsi File", command=encrypt_file)
103 button_encrypt_file.pack(side=tk.LEFT, padx=(0, 10))
104 button_decrypt_file = tk.Button(frame_buttons, text="Dekripsi File", command=decrypt_file)
105 button_decrypt_file.pack(side=tk.LEFT, padx=(0, 10))
106 root.mainloop()
107 
```

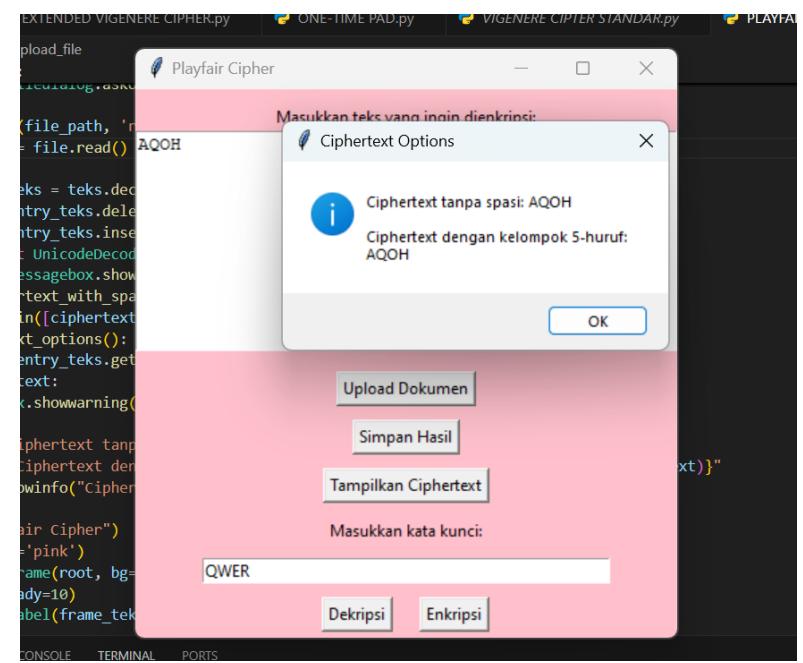
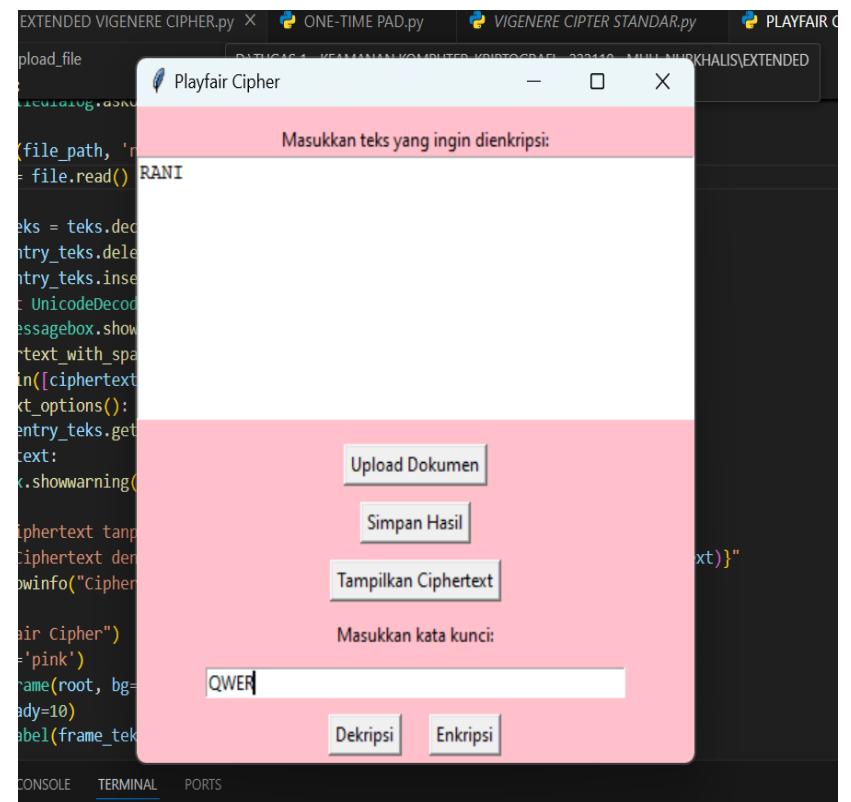


### 3. Playfair Cipher

```

1 import tkinter as tk
2 from tkinter import messagebox, filedialog
3 def create_playfair_table(kunci):
4     kunci = kunci.upper().replace('J', 'I')
5     seen = set()
6     table = []
7     for char in kunci:
8         if char not in seen and char.isalpha():
9             seen.add(char)
10            table.append(char)
11    for char in 'ABCDEFGHIKLMOPQRSTUVWXYZ':
12        if char not in seen:
13            seen.add(char)
14            table.append(char)
15    return [table[i:i+5] for i in range(0, 25, 5)]
16 def find_position(char, table):
17    for i, row in enumerate(table):
18        if char in row:
19            return i, row.index(char)
20    return None
21 def playfair_encrypt(teks_asil, kunci):
22    table = create_playfair_table(kunci)
23    teks_asil = teks_asil.upper().replace('J', 'I')
24    pairs = []
25    i = 0
26    while i < len(teks_asil):
27        char1 = teks_asil[i]
28        if i + 1 < len(teks_asil):
29            char2 = teks_asil[i + 1]
30        else:
31            char2 = 'X'
32        i += 1
33        pairs.append((char1, char2))
34    enkripsi = ''
35    for char1, char2 in pairs:
36        row1, col1 = find_position(char1, table)
37        row2, col2 = find_position(char2, table)
38        if row1 == row2:
39            enkripsi += table[row1][(col1 + 1) % 5]
40            enkripsi += table[row1][(col2 + 1) % 5]
41        elif col1 == col2:
42            enkripsi += table[(row1 + 1) % 5][col1]
43            enkripsi += table[(row2 + 1) % 5][col2]
44        else:
45            enkripsi += table[row1][col2]
46            enkripsi += table[row2][col1]
47    return enkripsi
48 def playfair_decrypt(teks_enkripsi, kunci):
49    table = create_playfair_table(kunci)
50    dekripsi = ''
51    pairs = []
52    i = 0
53    while i < len(teks_enkripsi):
54        pairs.append((teks_enkripsi[i], teks_enkripsi[i + 1]))
55        i += 2
56    for char1, char2 in pairs:
57        row1, col1 = find_position(char1, table)
58        row2, col2 = find_position(char2, table)
59        if row1 == row2:
60            dekripsi += table[row1][(col1 - 1) % 5]
61            dekripsi += table[row1][(col2 - 1) % 5]
62        elif col1 == col2:
63            dekripsi += table[(row1 - 1) % 5][col1]
64            dekripsi += table[(row2 - 1) % 5][col2]
65        else:
66            dekripsi += table[row1][col2]
67            dekripsi += table[row2][col1]
68    return dekripsi
69 def encrypt():
70    teks_asil = entry_teks.get("1.0", tk.END).strip()
71    kata_kunci = entry_junci.get()
72    if not teks_asil or not kata_kunci:
73        messagebox.showwarning("Input Error", "Mohon masukkan teks dan kata kunci.")
74    else:
75        hasil_enkripsi = playfair.encrypt(teks_asil, kata_kunci)
76        entry_teks.delete("1.0", tk.END)
77        entry_teks.insert(tk.END, hasil_enkripsi)
78    def save_file():
79        file_path = filedialog.asksaveasfilename(defaultextension=".txt", filetypes=[("Text Files", ".txt")])
80        if file_path:
81            with open(file_path, 'w', encoding='utf-8') as file:
82                file.write(hasil_enkripsi)
83    def upload_file():
84        file_path = filedialog.askopenfilename(filetypes=[("All files", "*.*")])
85        if file_path:
86            with open(file_path, 'r') as file:
87                teks = file.read()
88            try:
89                teks = teks.decode('utf-8')
90                entry_teks.delete("1.0", tk.END)
91                entry_teks.insert(tk.END, teks)
92            except UnicodeDecodeError:
93                messagebox.showwarning("File Error", "File tidak dapat dibaca sebagai teks.")
94    frame_teks.pack(pady=10)
95    label_teks = tk.Label(frame_teks, text="Masukkan teks yang ingin dienkripsi:", bg='pink')
96    label_teks.pack(pady=5)
97    entry_teks = tk.Text(frame_teks, width=50, height=10)
98    entry_teks.pack()
99    button_upload = tk.Button(root, text="Upload Dokumen", command=upload_file)
100   button_upload.pack(side=tk.LEFT, padx=5)
101   button_save = tk.Button(root, text="Simpan Hasil", command=save_file)
102   button_save.pack(side=tk.RIGHT, padx=5)
103   button_show_ciphertext = tk.Button(root, text="Tampilkan Ciphertext", command=show_ciphertext_options)
104   button_show_ciphertext.pack(pady=5)
105   label_kunci = tk.Label(root, text="Masukkan kata kunci:", bg='pink')
106   label_kunci.pack(pady=5)
107   entry_kunci = tk.Entry(root, width=30)
108   entry_kunci.pack(pady=5)
109   frame_buttons = tk.Frame(root, bg='pink')
110   frame_buttons.pack(pady=5)
111   frame_buttons.pack(pady=5)
112   button_decrypt = tk.Button(frame_buttons, text="Dekripsi", command=decrypt)
113   button_decrypt.pack(side=tk.LEFT, padx=10)
114   button_encrypt = tk.Button(frame_buttons, text="Enkripsi", command=encrypt)
115   button_encrypt.pack(side=tk.RIGHT, padx=10)
116   root.mainloop()

```

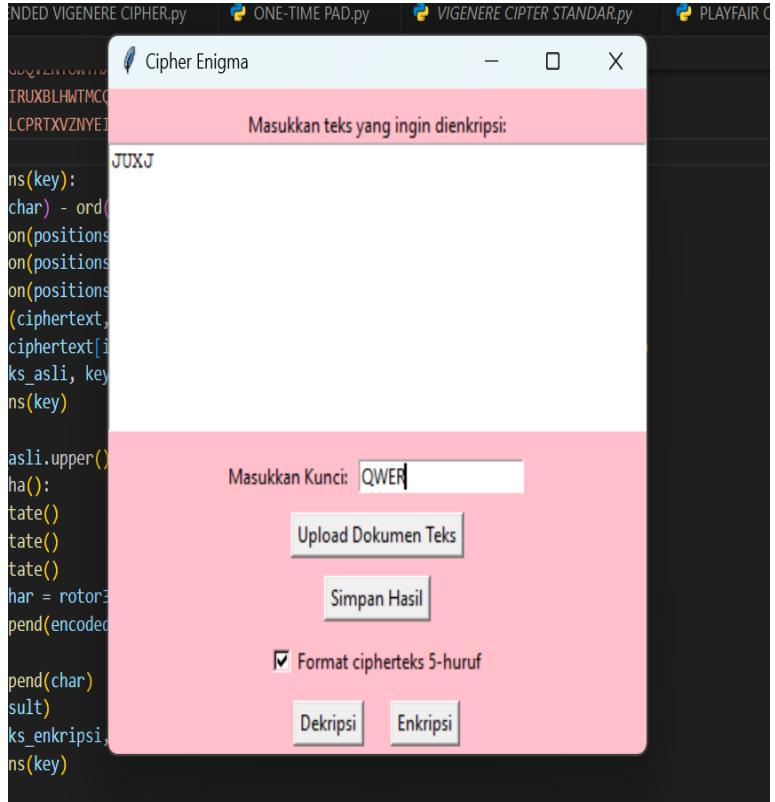
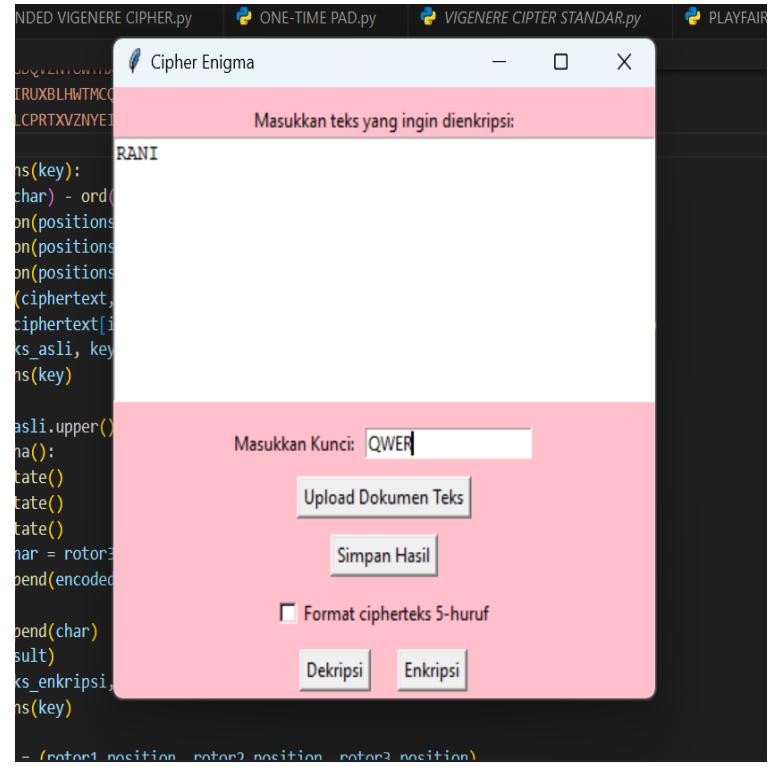


## 4. Enigma Cipher

```

1 import tkinter as tk
2 from tkinter import messagebox, filedialog
3 class Rotor:
4     def __init__(self, mapping):
5         self.mapping = mapping
6         self.position = 0
7     def set_position(self, position):
8         self.position = position % len(self.mapping)
9     def rotate(self):
10        self.position = (self.position + 1) % len(self.mapping)
11    def encode(self, char):
12        index = (ord(char) - ord('A')) + self.position % 26
13        return self.mapping[index]
14    def decode(self, char):
15        index = self.mapping.index(char)
16        return chr((index - self.position) % 26 + ord('A'))
17 rotor1 = Rotor("EKMFLGDQVZNTOWYXSPUABCRH")
18 rotor2 = Rotor("AJDKSIRUXBLHWTMCQGZNFYVPD")
19 rotor3 = Rotor("PHQCGURXVNYFLBKWJZDTSWHAE")
20 original_text = ""
21 def set_rotor_positions(key):
22     positions = [ord(char) - ord('A') for char in key.upper()]
23     rotor1.set_position(positions[0])
24     rotor2.set_position(positions[1] if len(positions) > 1 else positions[0])
25     rotor3.set_position(positions[2] if len(positions) > 2 else positions[0])
26 def format_ciphertext(ciphertext, group_size=5):
27     return " ".join(ciphertext[i:i+group_size] for i in range(0, len(ciphertext), group_size))
28 def enigma_encrypt(teks_asli, key):
29     set_rotor_positions(key)
30     result = []
31     for char in teks_asli.upper():
32         if char.isalpha():
33             rotor1.rotate()
34             rotor2.rotate()
35             rotor3.rotate()
36             encoded_char = rotor3.encode(rotor2.encode(rotor1.encode(char)))
37             result.append(encoded_char)
38         else:
39             result.append(char)
40     return ''.join(result)
41 def enigma_decrypt(teks_enkripsi, key):
42     set_rotor_positions(key)
43     result = []
44     initial_positions = (rotor1.position, rotor2.position, rotor3.position)
45     for char in teks_enkripsi.upper():
46         if char.isalpha():
47             decoded_char = rotor1.decode(rotor2.decode(rotor3.decode(char)))
48             result.append(decoded_char)
49             rotor1.rotate()
50             rotor2.rotate()
51             rotor3.rotate()
52         else:
53             result.append(char)
54     rotor1.position, rotor2.position, rotor3.position = initial_positions
55     result.append(result)
56     def encrypt():
57         global original_text
58         teks_asli = entry_teks.get("1.0", tk.END).strip()
59         key = entry_key.get().strip()
60         if not teks_asli:
61             messagebox.showwarning("Input Error", "Mohon masukkan teks.")
62             return
63         if not key:
64             messagebox.showwarning("Input Error", "Mohon masukkan kunci.")
65             return
66         original_text = teks_asli
67         hasil_enkripsi = enigma_encrypt(teks_asli, key)
68         if var_format.get() == 1:
69             hasil_enkripsi = format_ciphertext(hasil_enkripsi)
70         entry_teks.delete("1.0", tk.END)
71         entry_teks.insert(tk.END, hasil_enkripsi)
72     def decrypt():
73         teks_enkripsi = entry_teks.get("1.0", tk.END).strip()
74         key = entry_key.get().strip()
75         if not teks_enkripsi:
76             messagebox.showwarning("Input Error", "Mohon masukkan teks enkripsi.")
77             return
78         if not key:
79             messagebox.showwarning("Input Error", "Mohon masukkan kunci.")
80             return
81         hasil_dekripsi = enigma_decrypt(teks_enkripsi, key)
82         entry_teks.delete("1.0", tk.END)
83         entry_teks.insert(tk.END, original_text)
84     def save_file():
85         file_path = filedialog.asksaveasfilename(defaultextension=".txt", filetypes=[("Text files", "*.txt")])
86         if file_path:
87             with open(file_path, 'w', encoding='utf-8') as file:
88                 teks = entry_teks.get("1.0", tk.END)
89                 file.write(teks)
90     def upload_file():
91         file_path = filedialog.askopenfilename(filetypes=[("Text files", "*.txt")])
92         if file_path:
93             with open(file_path, 'r', encoding='utf-8') as file:
94                 teks = file.read()
95                 entry_teks.delete("1.0", tk.END)
96                 entry_teks.insert(tk.END, teks)
97         root = tk.Tk()
98         root.title("Cipher Enigma")
99         root.configure(bg='pink')
100        frame_teks = tk.Frame(root, bg='pink')
101        frame_teks.pack(pady=10)
102        label_teks = tk.Label(frame_teks, text="Masukkan teks yang ingin dienkripsi:", bg='pink')
103        label_teks.pack()
104        entry_teks = tk.Text(frame_teks, width=50, height=10)
105        entry_teks.pack()
106        frame_key = tk.Frame(root, bg='pink')
107        frame_key.pack(pady=5)
108        label_key = tk.Label(frame_key, text="Masukkan Kunci:", bg='pink')
109        label_key.pack(side=tk.LEFT)
110        entry_key = tk.Entry(frame_key)
111        entry_key.pack(side=tk.LEFT, padx=5)
112        button_upload = tk.Button(root, text="Upload Dokumen Teks", command=upload_file)
113        button_upload.pack(pady=5)
114        button_save = tk.Button(root, text="Simpan Hasil", command=save_file)
115        button_save.pack(pady=5)
116        var_format = tk.StringVar()
117        check_format = tk.Checkbutton(root, text="Format ciphertext 5-huruf", variable=var_format, bg='pink')
118        check_format.pack(pady=5)
119        frame_buttons = tk.Frame(root, bg='pink')
120        frame_buttons.pack(pady=5)
121        button_decrypt = tk.Button(frame_buttons, text="Dekripsi", command=decrypt)
122        button_decrypt.pack(side=tk.LEFT, padx=(0, 10))
123        button_encrypt = tk.Button(frame_buttons, text="Enkripsi", command=enrypt)
124        button_encrypt.pack(side=tk.RIGHT, padx=(10, 0))
125        root.mainloop()
126

```



## 5. One-Time pad

```

1 import tkinter as tk
2 from tkinter import messagebox, filedialog
3 import random
4 import string
5 def generate_random_key(length):
6     return ''.join(random.choice(string.ascii_uppercase) for _ in range(length))
7 def one_time_pad_encrypt(plain_text, key):
8     encrypted_text = []
9     for pt_char, key_char in zip(plain_text.upper(), key):
10         if pt_char.isalpha():
11             encrypted_char = chr((ord(pt_char) - ord('A') + ord(key_char) - ord('A')) % 26 + ord('A'))
12             encrypted_text.append(encrypted_char)
13         else:
14             encrypted_text.append(pt_char)
15     return ''.join(encrypted_text)
16 def one_time_pad_decrypt(encrypted_text, key):
17     decrypted_text = []
18     for enc_char, key_char in zip(encrypted_text.upper(), key):
19         if enc_char.isalpha():
20             decrypted_char = chr((ord(enc_char) - ord('A') - (ord(key_char) - ord('A'))) % 26 + ord('A'))
21             decrypted_text.append(decrypted_char)
22         else:
23             decrypted_text.append(enc_char)
24     return ''.join(decrypted_text)
25 def encrypt():
26     teks_asli = entry_teks.get("1.0", tk.END).strip()
27     if not teks_asli:
28         messagebox.showwarning("Input Error", "Mohon masukkan teks.")
29         return
30     key = entry_key.get().strip()
31     if len(key) < len(teks_asli):
32         messagebox.showwarning("Key Error", "Kunci harus sepanjang atau lebih panjang dari teks.")
33         return
34     hasil_enkripsi = one_time_pad_encrypt(teks_asli, key)
35     entry_teks.delete("1.0", tk.END)
36     entry_teks.insert(tk.END, f"(hasil_enkripsi)")
37 def decrypt():
38     teks_enkripsi = entry_teks.get("1.0", tk.END).strip()
39     if not teks_enkripsi:
40         messagebox.showwarning("Input Error", "Mohon masukkan teks enkripsi.")
41         return
42     key = entry_key.get().strip()
43     if len(key) < len(teks_enkripsi):
44         messagebox.showwarning("Key Error", "Kunci harus sepanjang atau lebih panjang dari ciphertext.")
45         return
46     hasil_dekripsi = one_time_pad_decrypt(teks_enkripsi, key)
47     entry_teks.delete("1.0", tk.END)
48     entry_teks.insert(tk.END, f"(hasil_dekripsi)")
49 def save_file():
50     file_path = filedialog.asksaveasfilename(defaultextension=".txt", filetypes=[("Text files", "*.txt")])
51     if file_path:
52         with open(file_path, 'w', encoding='utf-8') as file:
53             teks = entry_teks.get("1.0", tk.END)
54             file.write(teks)
55 def upload_file():
56     file_path = filedialog.askopenfilename(filetypes=[("All files", "*.*")])
57     if file_path:
58         with open(file_path, 'r') as file:
59             teks = file.read()
60             entry_teks.delete("1.0", tk.END)
61             entry_teks.insert(tk.END, teks.decode('utf-8', errors='ignore'))
62 def upload_key_file():
63     key_file_path = filedialog.askopenfilename(filetypes=[("Text files", ".txt")])
64     if key_file_path:
65         with open(key_file_path, 'r', encoding='utf-8') as file:
66             key = file.read()
67             entry_key.delete(0, tk.END)
68             entry_key.insert(0, key)
69 root = tk.Tk()
70 root.title("One-time Pad Cipher")
71 root.configure(bg="pink")
72 frame_teks = tk.Frame(root, bg="pink")
73 frame_teks.pack(pady=10)
74 label_teks = tk.Label(frame_teks, text="Masukkan teks yang ingin dienkripsi:", bg="pink")
75 label_teks.pack()
76 entry_teks = tk.Text(frame_teks, width=50, height=10)
77 entry_teks.pack()
78 frame_key = tk.Frame(root, bg="pink")
79 frame_key.pack(pady=10)
80 label_key = tk.Label(frame_key, text="Masukkan kunci:", bg="pink")
81 label_key.pack()
82 entry_key = tk.Entry(frame_key, width=50)
83 entry_key.pack()
84 button_upload = tk.Button(root, text="Upload Dokumen", command=upload_file)
85 button_upload.pack(pady=5)
86 button_save = tk.Button(root, text="Simpan Hasil", command=save_file)
87 button_save.pack(pady=5)
88 button_upload_key = tk.Button(root, text="Upload Kunci dari File", command=upload_key_file)
89 button_upload_key.pack(pady=5)
90 frame_buttons = tk.Frame(root, bg="pink")
91 frame_buttons.pack(pady=5)
92 button_decrypt = tk.Button(frame_buttons, text="Dekripsi", command=decrypt)
93 button_decrypt.pack(side=tk.LEFT, padx=(0, 10))
94 button_encrypt = tk.Button(frame_buttons, text="Enkripsi", command=encrypt)
95 button_encrypt.pack(side=tk.RIGHT, padx=(10, 0))
96 root.mainloop()
97

```

