**Tugas 1**

**TUGAS KEAMANAN  
 KOMPUTER/KRIPTOGRAFI**

****

**DISUSUN OLEH:**

Nama : Andi Amalia Ramadani   
NIM : 212084  
Kelas : 5-TKKO-I  
Dosen : Suci Rahma Dani Rachman, S.Kom., MT.

**PROGRAM STUDI TEKNIK INFORMATIKA  
UNIVERSITAS DIPA MAKASSAR  
TAHUN 2024**

Buatlah sebuah program dalam Bahasa C/C++/Java/Python/Ruby/Golang dan lainnya (pilih salah satu) dengan antarmuka (GUI) yang mengimplementasikan:

a) Vigenere Cipher standard (26 huruf alfabet)

b) Extended Vigenere Cipher (256 karakter ASCII)

c) Playfair Cipher (26 huruf alfabet)

d) Enigma Cipher dengan 3-rotor (26 huruf alfabet)

e) Bonus: One-time pad (26 huruf alfabet)

**Dalam laporan ini saya menggunakan Bahasa Pemograman Python**

**Laporan**

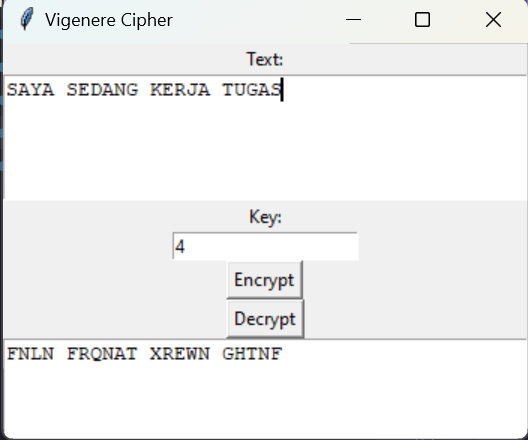
|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| No | Spesifikasi | Berhasil (√) | Kurang Berhasil (√) | Keterangan |
| 1. | Vigenere Cipher | √ |  | Berhasil melakukan enkripsi dan deskripsi |
| 2. | Extended Vigenere Cipher | √ |  | Berhasil melakukan enkripsi dan deskripsi |
| 3. | Playfair Cipher | √ |  | Berhasil melakukan enkripsi dan deskripsi. Enkripsi dan dekripsi sesuai aturan, menghasilkan teks yang mendekati plaintext asli. |
| 4. | Enigma Cipher | √ |  | Berhasil melakukan enkripsi dan deskripsi |
| 5. | One-Time pad | √ |  | Berhasil melakukan enkripsi dan deskripsi, tetapi perlu mengklik beberapa kali tombol deskripsi untuk mencari kata yang sesuai |

**Lampiran (Print Screen)**

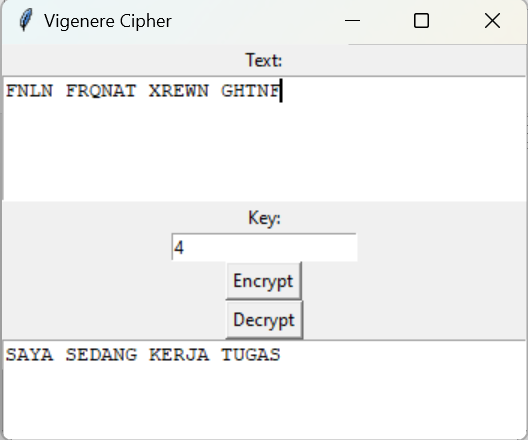
1. Vigenere Cipher
2. from tkinter import \*
3. def vigenere\_cipher(text, key, mode):
4. # Convert text and key to uppercase
5. text = text.upper()
6. key = key.upper()
7. # Generate key of the same length as text
8. key = key \* (len(text) // len(key) + 1)
9. key = key[:len(text)]
10. # Encrypt or decrypt text using Vigenere Cipher
11. result = ""
12. for i in range(len(text)):
13. if text[i].isalpha():
14. if mode == "encrypt":
15. result += chr((ord(text[i]) + ord(key[i])) % 26 + ord('A'))
16. elif mode == "decrypt":
17. result += chr((ord(text[i]) - ord(key[i])) % 26 + ord('A'))
18. else:
19. result += text[i]
20. return result
21. def encrypt():
22. # Get input from user
23. text = text\_input.get("1.0", END).strip()
24. key = key\_input.get().strip()
25. # Encrypt text using Vigenere Cipher
26. result = vigenere\_cipher(text, key, "encrypt")
27. # Display result
28. result\_output.delete("1.0", END)
29. result\_output.insert(END, result)
30. def decrypt():
31. # Get input from user
32. text = text\_input.get("1.0", END).strip()
33. key = key\_input.get().strip()
34. # Decrypt text using Vigenere Cipher
35. result = vigenere\_cipher(text, key, "decrypt")
36. # Display result
37. result\_output.delete("1.0", END)
38. result\_output.insert(END, result)
39. # Create GUI
40. root = Tk()
41. root.title("Vigenere Cipher")
42. # Create input widgets
43. text\_label = Label(root, text="Text:")
44. text\_input = Text(root, height=5, width=50)
45. key\_label = Label(root, text="Key:")
46. key\_input = Entry(root)
47. encrypt\_button = Button(root, text="Encrypt", command=encrypt)
48. decrypt\_button = Button(root, text="Decrypt", command=decrypt)
49. # Create output widget
50. result\_output = Text(root, height=5, width=50)
51. # Pack widgets
52. text\_label.pack()
53. text\_input.pack()
54. key\_label.pack()
55. key\_input.pack()
56. encrypt\_button.pack()
57. decrypt\_button.pack()
58. result\_output.pack()
59. # Start GUI
60. root.mainloop()

**Tampilan Vigenere Cipher**

1. Enkripsi

****

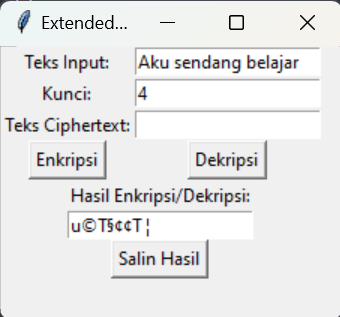
1. Deskripsi



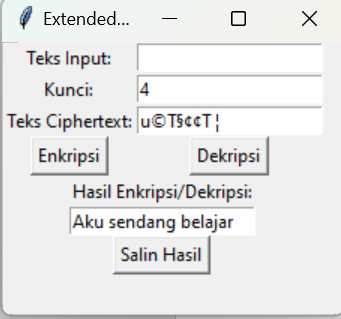
1. Extended Vigenere Cipher
2. import tkinter as tk
3. import pyperclip
4. def extended\_vigenere\_cipher(input\_text, key):
5. key\_length = len(key)
6. cipher\_text = ""
8. for i in range(len(input\_text)):
9. char = input\_text[i]
10. key\_char = key[i % key\_length]
12. cipher\_char = chr((ord(char) + ord(key\_char)) % 256)
13. cipher\_text += cipher\_char
15. return cipher\_text
16. def encrypt\_text():
17. input\_text = entry\_input\_text.get()
18. key = entry\_key.get()
19. cipher\_text = extended\_vigenere\_cipher(input\_text, key)
20. entry\_result.delete(0, tk.END)
21. entry\_result.insert(0, cipher\_text)
22. def decrypt\_text():
23. cipher\_text = entry\_cipher\_text.get()
24. key = entry\_key.get()
25. decrypted\_text = ""
27. for i in range(len(cipher\_text)):
28. char = cipher\_text[i]
29. key\_char = key[i % len(key)]
31. decrypted\_char = chr((ord(char) - ord(key\_char)) % 256)
32. decrypted\_text += decrypted\_char
34. entry\_result.delete(0, tk.END)
35. entry\_result.insert(0, decrypted\_text)
36. def copy\_result():
37. result\_text = entry\_result.get()
38. pyperclip.copy(result\_text)
39. # Membuat jendela aplikasi
40. root = tk.Tk()
41. root.title("Extended Vigenere Cipher")
42. # Membuat label
43. label\_input\_text = tk.Label(root, text="Teks Input:")
44. label\_key = tk.Label(root, text="Kunci:")
45. label\_cipher\_text = tk.Label(root, text="Teks Ciphertext:")
46. label\_result = tk.Label(root, text="Hasil Enkripsi/Dekripsi:")
47. # Membuat entry (input) dan tombol
48. entry\_input\_text = tk.Entry(root)
49. entry\_key = tk.Entry(root)
50. entry\_cipher\_text = tk.Entry(root)
51. entry\_result = tk.Entry(root)
52. encrypt\_button = tk.Button(root, text="Enkripsi", command=encrypt\_text)
53. decrypt\_button = tk.Button(root, text="Dekripsi", command=decrypt\_text)
54. copy\_button = tk.Button(root, text="Salin Hasil", command=copy\_result)
55. # Menempatkan komponen-komponen ke dalam grid
56. label\_input\_text.grid(row=0, column=0)
57. label\_key.grid(row=1, column=0)
58. label\_cipher\_text.grid(row=2, column=0)
59. label\_result.grid(row=4, column=0, columnspan=2)
60. entry\_input\_text.grid(row=0, column=1)
61. entry\_key.grid(row=1, column=1)
62. entry\_cipher\_text.grid(row=2, column=1)
63. entry\_result.grid(row=5, column=0, columnspan=2)
64. encrypt\_button.grid(row=3, column=0)
65. decrypt\_button.grid(row=3, column=1)
66. copy\_button.grid(row=6, column=0, columnspan=2)
67. root.mainloop()

**Tampilan Extended Vigenere Cipher**

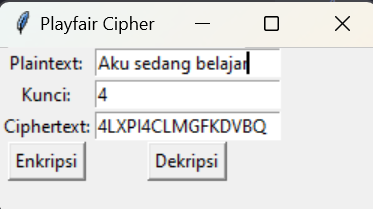
1. Enkripsi



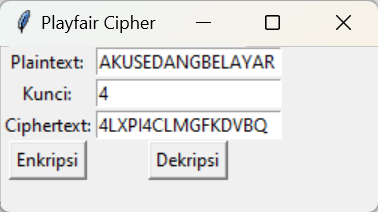
1. Deskripsi



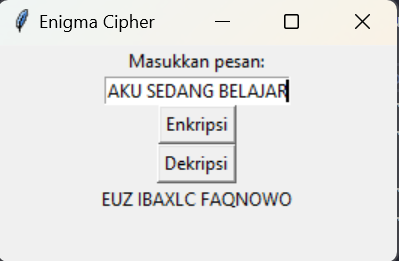
1. Playfair Cipher
2. import tkinter as tk
3. def prepare\_key(key):
4. key = key.replace('J', 'I')
5. key = ''.join(dict.fromkeys(key))
6. key = key.replace(' ', '').upper()
8. # Membuat matriks kunci
9. key\_matrix = [[0] \* 5 for \_ in range(5)]
10. alphabet = 'ABCDEFGHIKLMNOPQRSTUVWXYZ'
12. key = key + alphabet
14. for i in range(5):
15. for j in range(5):
16. key\_matrix[i][j] = key[i \* 5 + j]
18. return key\_matrix
19. def playfair\_encrypt(plain\_text, key\_matrix):
20. plain\_text = plain\_text.replace('J', 'I')
21. plain\_text = plain\_text.replace(' ', '').upper()
23. encrypted\_text = ''
24. i = 0
25. while i < len(plain\_text):
26. char1 = plain\_text[i]
27. i += 1
29. if i == len(plain\_text):
30. char2 = 'X'
31. else:
32. char2 = plain\_text[i]
33. if char2 == char1:
34. char2 = 'X'
35. i -= 1
37. row1, col1 = find\_char\_position(key\_matrix, char1)
38. row2, col2 = find\_char\_position(key\_matrix, char2)
40. if row1 == row2:
41. encrypted\_text += key\_matrix[row1][(col1 + 1) % 5]
42. encrypted\_text += key\_matrix[row2][(col2 + 1) % 5]
43. elif col1 == col2:
44. encrypted\_text += key\_matrix[(row1 + 1) % 5][col1]
45. encrypted\_text += key\_matrix[(row2 + 1) % 5][col2]
46. else:
47. encrypted\_text += key\_matrix[row1][col2]
48. encrypted\_text += key\_matrix[row2][col1]
50. i += 1
52. return encrypted\_text
53. def playfair\_decrypt(encrypted\_text, key\_matrix):
54. decrypted\_text = ''
55. i = 0
56. while i < len(encrypted\_text):
57. char1 = encrypted\_text[i]
58. i += 1
60. if i == len(encrypted\_text):
61. char2 = 'X'
62. else:
63. char2 = encrypted\_text[i]
65. row1, col1 = find\_char\_position(key\_matrix, char1)
66. row2, col2 = find\_char\_position(key\_matrix, char2)
68. if row1 == row2:
69. decrypted\_text += key\_matrix[row1][(col1 - 1) % 5]
70. decrypted\_text += key\_matrix[row2][(col2 - 1) % 5]
71. elif col1 == col2:
72. decrypted\_text += key\_matrix[(row1 - 1) % 5][col1]
73. decrypted\_text += key\_matrix[(row2 - 1) % 5][col2]
74. else:
75. decrypted\_text += key\_matrix[row1][col2]
76. decrypted\_text += key\_matrix[row2][col1]
78. i += 1
80. return decrypted\_text
81. def find\_char\_position(key\_matrix, char):
82. for i in range(5):
83. for j in range(5):
84. if key\_matrix[i][j] == char:
85. return i, j
86. return -1, -1
87. def encrypt\_text():
88. plain\_text = entry\_plain\_text.get()
89. key = entry\_key.get()
90. key\_matrix = prepare\_key(key)
91. encrypted\_text = playfair\_encrypt(plain\_text, key\_matrix)
92. entry\_encrypted\_text.delete(0, tk.END)
93. entry\_encrypted\_text.insert(0, encrypted\_text)
94. def decrypt\_text():
95. encrypted\_text = entry\_encrypted\_text.get()
96. key = entry\_key.get()
97. key\_matrix = prepare\_key(key)
98. decrypted\_text = playfair\_decrypt(encrypted\_text, key\_matrix)
99. entry\_plain\_text.delete(0, tk.END)
100. entry\_plain\_text.insert(0, decrypted\_text)
101. # Membuat jendela aplikasi
102. root = tk.Tk()
103. root.title("Playfair Cipher")
104. # Membuat label
105. label\_plain\_text = tk.Label(root, text="Plaintext:")
106. label\_key = tk.Label(root, text="Kunci:")
107. label\_encrypted\_text = tk.Label(root, text="Ciphertext:")
108. # Membuat entry (input) dan tombol
109. entry\_plain\_text = tk.Entry(root)
110. entry\_key = tk.Entry(root)
111. entry\_encrypted\_text = tk.Entry(root)
112. encrypt\_button = tk.Button(root, text="Enkripsi", command=encrypt\_text)
113. decrypt\_button = tk.Button(root, text="Dekripsi", command=decrypt\_text)
114. # Menempatkan komponen-komponen ke dalam grid
115. label\_plain\_text.grid(row=0, column=0)
116. label\_key.grid(row=1, column=0)
117. label\_encrypted\_text.grid(row=2, column=0)
118. entry\_plain\_text.grid(row=0, column=1)
119. entry\_key.grid(row=1, column=1)
120. entry\_encrypted\_text.grid(row=2, column=1)
121. encrypt\_button.grid(row=3, column=0)
122. decrypt\_button.grid(row=3, column=1)
123. root.mainloop()
124. Enkripsi



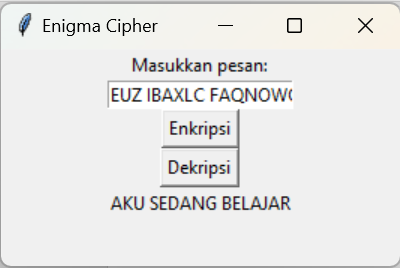
1. Deskripsi



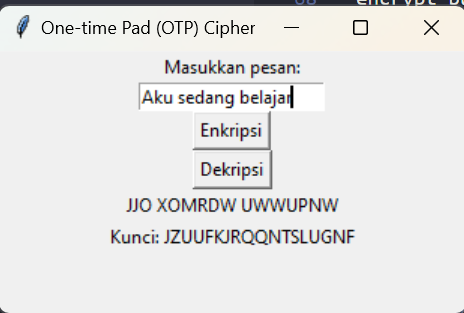
1. Enigma Cipher
2. import tkinter as tk
3. # Inisialisasi rotor dan konfigurasi awal
4. rotor1 = "EKMFLGDQVZNTOWYHXUSPAIBRCJ"
5. rotor2 = "AJDKSIRUXBLHWTMCQGZNPYFVOE"
6. rotor3 = "BDFHJLCPRTXVZNYEIWGAKMUSQO"
7. reflector = "YRUHQSLDPXNGOKMIEBFZCWVJAT"
8. rotor\_positions = [0, 0, 0]
9. # Fungsi untuk mengenkripsi pesan
10. def encrypt\_message():
11. plaintext = input\_text.get().upper()
12. encrypted\_text = ""
14. for char in plaintext:
15. if char.isalpha():
16. encrypted\_char = encrypt\_char(char)
17. encrypted\_text += encrypted\_char
18. else:
19. encrypted\_text += char
21. output\_text.set(encrypted\_text)
22. # Fungsi untuk mendekripsi pesan
23. def decrypt\_message():
24. ciphertext = input\_text.get().upper()
25. decrypted\_text = ""
27. for char in ciphertext:
28. if char.isalpha():
29. decrypted\_char = decrypt\_char(char)
30. decrypted\_text += decrypted\_char
31. else:
32. decrypted\_text += char
34. output\_text.set(decrypted\_text)
35. # Fungsi untuk mengenkripsi satu karakter
36. def encrypt\_char(char):
37. global rotor\_positions  # Menggunakan rotor\_positions global
38. rotor1\_pos, rotor2\_pos, rotor3\_pos = rotor\_positions
39. char\_index = ord(char) - ord('A')
41. # Enkripsi karakter melalui rotor 1, rotor 2, rotor 3, dan reflector
42. char = rotor1[(char\_index + rotor1\_pos) % 26]
43. char = rotor2[(ord(char) - ord('A') + rotor2\_pos) % 26]
44. char = rotor3[(ord(char) - ord('A') + rotor3\_pos) % 26]
45. char = reflector[ord(char) - ord('A')]
47. # Kembali melalui rotor 3, 2, dan 1
48. char = rotor3.index(char)
49. char = chr((char - rotor3\_pos + 26) % 26 + ord('A'))
50. char = rotor2.index(char)
51. char = chr((char - rotor2\_pos + 26) % 26 + ord('A'))
52. char = rotor1.index(char)
53. char = chr((char - rotor1\_pos + 26) % 26 + ord('A'))
54. # Putar rotor
55. rotor\_positions[0] = (rotor\_positions[0] + 1) % 26  # Rotate rotor 1
56. if rotor\_positions[0] == 0:  # Rotate rotor 2 if rotor 1 completes a cycle
57. rotor\_positions[1] = (rotor\_positions[1] + 1) % 26
58. if rotor\_positions[1] == 0:  # Rotate rotor 3 if rotor 2 completes a cycle
59. rotor\_positions[2] = (rotor\_positions[2] + 1) % 26
61. return char
62. # Fungsi untuk mendekripsi satu karakter
63. def decrypt\_char(char):
64. global rotor\_positions  # Menggunakan rotor\_positions global
65. rotor1\_pos, rotor2\_pos, rotor3\_pos = rotor\_positions
66. char\_index = ord(char) - ord('A')
67. # Mendekripsi karakter melalui rotor 1, rotor 2, rotor 3, dan reflector
68. char = rotor1[(char\_index + rotor1\_pos) % 26]
69. char = rotor2[(ord(char) - ord('A') + rotor2\_pos) % 26]
70. char = rotor3[(ord(char) - ord('A') + rotor3\_pos) % 26]
71. char = reflector[ord(char) - ord('A')]
72. # Kembali melalui rotor 3, 2, dan 1
73. char = rotor3.index(char)
74. char = chr((char - rotor3\_pos + 26) % 26 + ord('A'))
75. char = rotor2.index(char)
76. char = chr((char - rotor2\_pos + 26) % 26 + ord('A'))
77. char = rotor1.index(char)
78. char = chr((char - rotor1\_pos + 26) % 26 + ord('A'))
79. # Putar rotor
80. rotor\_positions[0] = (rotor\_positions[0] + 1) % 26  # Rotate rotor 1
81. if rotor\_positions[0] == 0:  # Rotate rotor 2 if rotor 1 completes a cycle
82. rotor\_positions[1] = (rotor\_positions[1] + 1) % 26
83. if rotor\_positions[1] == 0:  # Rotate rotor 3 if rotor 2 completes a cycle
84. rotor\_positions[2] = (rotor\_positions[2] + 1) % 26
86. return char
87. # Membuat jendela tkinter
88. window = tk.Tk()
89. window.title("Enigma Cipher")
90. window.geometry("400x200")
91. # Label dan input teks
92. tk.Label(window, text="Masukkan pesan:").pack()
93. input\_text = tk.StringVar()
94. input\_entry = tk.Entry(window, textvariable=input\_text)
95. input\_entry.pack()
96. # Tombol untuk mengenkripsi dan mendekripsi
97. encrypt\_button = tk.Button(window, text="Enkripsi", command=encrypt\_message)
98. encrypt\_button.pack()
99. decrypt\_button = tk.Button(window, text="Dekripsi", command=decrypt\_message)
100. decrypt\_button.pack()
101. # Hasil enkripsi
102. output\_text = tk.StringVar()
103. output\_label = tk.Label(window, textvariable=output\_text)
104. output\_label.pack()
105. # Memulai jendela GUI
106. window.mainloop()
107. Enkripsi



1. Deskripsi



1. One-Time pad
2. import random
3. import tkinter as tk
4. # Fungsi untuk mengenkripsi pesan dengan OTP
5. def encrypt\_message():
6. plaintext = input\_text.get().upper()
7. key = generate\_random\_key(len(plaintext))
9. encrypted\_text = ""
10. for i in range(len(plaintext)):
11. char = plaintext[i]
12. if char.isalpha():
13. encrypted\_char = encrypt\_char(char, key[i])
14. encrypted\_text += encrypted\_char
15. else:
16. encrypted\_text += char
18. output\_text.set(encrypted\_text)
19. key\_text.set("Kunci: " + "".join(key))
20. # Fungsi untuk mendekripsi pesan dengan OTP
21. def decrypt\_message():
22. encrypted\_text = input\_text.get().upper()
23. key = key\_text.get()[7:]  # Mengambil kunci dari label (menghilangkan "Kunci: ")
25. decrypted\_text = ""
26. for i in range(len(encrypted\_text)):
27. char = encrypted\_text[i]
28. if char.isalpha():
29. decrypted\_char = decrypt\_char(char, key[i])
30. decrypted\_text += decrypted\_char
31. else:
32. decrypted\_text += char
34. output\_text.set(decrypted\_text)
35. # Fungsi untuk mengenkripsi satu karakter dengan OTP
36. def encrypt\_char(char, key\_char):
37. char\_index = ord(char) - ord('A')
38. key\_index = ord(key\_char) - ord('A')
39. encrypted\_index = (char\_index + key\_index) % 26
40. return chr(encrypted\_index + ord('A'))
41. # Fungsi untuk mendekripsi satu karakter dengan OTP
42. def decrypt\_char(char, key\_char):
43. char\_index = ord(char) - ord('A')
44. key\_index = ord(key\_char) - ord('A')
45. decrypted\_index = (char\_index - key\_index) % 26
46. return chr(decrypted\_index + ord('A'))
47. # Fungsi untuk menghasilkan kunci acak dengan panjang tertentu
48. def generate\_random\_key(length):
49. return [random.choice('ABCDEFGHIJKLMNOPQRSTUVWXYZ') for \_ in range(length)]
50. # Membuat jendela tkinter
51. window = tk.Tk()
52. window.title("One-time Pad (OTP) Cipher")
53. window.geometry("400x250")
54. # Label dan input teks
55. tk.Label(window, text="Masukkan pesan:").pack()
56. input\_text = tk.StringVar()
57. input\_entry = tk.Entry(window, textvariable=input\_text)
58. input\_entry.pack()
59. # Tombol untuk mengenkripsi
60. encrypt\_button = tk.Button(window, text="Enkripsi", command=encrypt\_message)
61. encrypt\_button.pack()
62. # Tombol untuk mendekripsi
63. decrypt\_button = tk.Button(window, text="Dekripsi", command=decrypt\_message)
64. decrypt\_button.pack()
65. # Hasil enkripsi atau dekripsi
66. output\_text = tk.StringVar()
67. output\_label = tk.Label(window, textvariable=output\_text)
68. output\_label.pack()
69. # Label untuk kunci
70. key\_text = tk.StringVar()
71. key\_label = tk.Label(window, textvariable=key\_text)
72. key\_label.pack()
73. # Memulai jendela GUI
74. window.mainloop()
75. Enkripsi



1. Deskripsi

