Завдання 1

Побудувати функцію відмови і ДСА для стрічки abbaabbaa



def failure\_function(pattern): table = [0] \* len(pattern)

for i in range(1, len(pattern)): j = table[i-1]

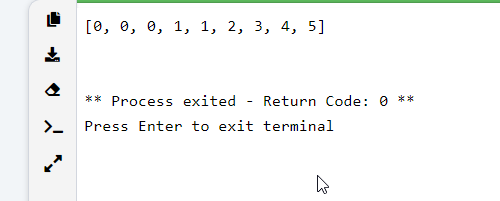
while j > 0 and pattern[i] != pattern[j]: j = table[j-1]

if pattern[i] == pattern[j]: table[i] = j + 1

return table

# Приклад використання pattern = "abbaabbaa"

failure\_table = failure\_function(pattern) print(failure\_table)



Завдання 2

Застосувати алгоритм КМП для перевірки входження ключового слова аbа, як підстрічки в abaabаbааa



def kmp(text, pattern):

table = failure\_function(pattern) i = 0

j = 0

while i < len(text):

if text[i] == pattern[j]: i += 1

j += 1

else:

if j > 0:

j = table[j-1] else:

i += 1

if j == len(pattern): return True

return False

def failure\_function(pattern): table = [0] \* len(pattern)

for i in range(1, len(pattern)): j = table[i-1]

while j > 0 and pattern[i] != pattern[j]: j = table[j-1]

if pattern[i] == pattern[j]: table[i] = j + 1

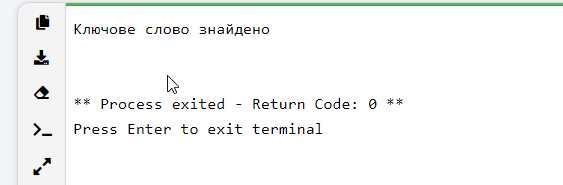
return table

# Приклад використання text = "abaabаbааa" pattern = "aba"

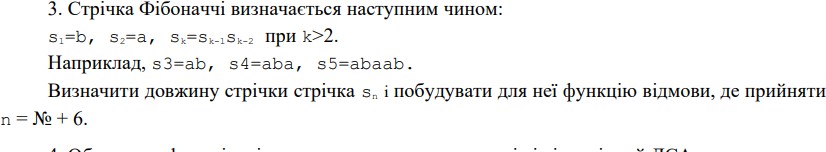
result = kmp(text, pattern) if result:

print("Ключове слово знайдено") else:

print("Ключове слово не знайдено")



Завдання 3



def fibonacci\_string(n): if n == 1:

return 'b' elif n == 2: return 'a'

else:

s1 = 'b'

s2 = 'a'

for i in range(3, n + 1): s = s2 + s1

s1, s2 = s2, s return s

def failure\_function(s): m = len(s)

f = [0] \* m

for i in range(1, m): j = f[i - 1]

while j > 0 and s[i] != s[j]: j = f[j - 1]

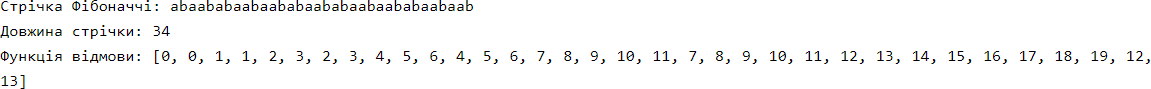
if s[i] == s[j]: j += 1

f[i] = j return f

#Приклад використання n = 9

fibonacci\_str = fibonacci\_string(n) print("Стрічка Фібоначчі:", fibonacci\_str) print("Довжина стрічки:", len(fibonacci\_str))

print("Функція відмови:", failure\_function(fibonacci\_str))



Завдання 4

Обчислити функцію відмови множини ключових слів і відповідний ДСА: {all, fall, fatal, llama}



def compute\_failure\_function(keywords): m = len(keywords)

f = [0] \* m

for i in range(1, m): j = f[i - 1]

while j > 0 and keywords[i] != keywords[j]: j = f[j - 1]

if keywords[i] == keywords[j]: j += 1

f[i] = j return f

def construct\_DFA(keywords): m = len(keywords)

alphabet = set(''.join(keywords))

transition\_table = [{char: 0 for char in alphabet} for \_ in range(m)] f = compute\_failure\_function(keywords)

for i in range(m):

for char in alphabet:

if i > 0 and char != keywords[i]:

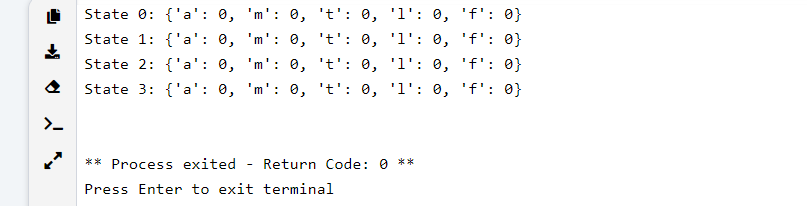
transition\_table[i][char] = transition\_table[f[i - 1]][char] else:

transition\_table[i][char] = i + (char == keywords[i]) return transition\_table

keywords = ["all", "fall", "fatal", "llama"] failure\_function = compute\_failure\_function(keywords) print("Функція відмови:", failure\_function)

DFA = construct\_DFA(keywords) print("Детермінований скінчений автомат:") for i, row in enumerate(DFA):

print(f"State {i}: {row}")



Завдання 5

Для стрічки “Прізвище Ім’я По батькові” (Дмитрик Валерій Павлович) знайти позицій усіх ключових слів утворених істинними префіксами довжиною 3 і істинними суфіксами довжиною 4.

def find\_keywords\_positions(full\_string, prefix\_length=3, suffix\_length=4): positions = []

words = full\_string.split() for word in words:

for i in range(len(word) - suffix\_length - prefix\_length + 1): prefix = word[i:i+prefix\_length]

suffix = word[i+prefix\_length+len(word)-suffix\_length:] positions.append((prefix, suffix, i))

return positions

# Вхідна стрічка

full\_name = "Дмитрик Валерій Павлович"

# Знаходження позицій ключових слів keyword\_positions = find\_keywords\_positions(full\_name) print("Позиції ключових слів:")

for keyword in keyword\_positions:

print("Префікс: {}, Суфікс: {}, Позиція: {}".format(keyword[0], keyword[1], keyword[2]))

