

Завдання 1

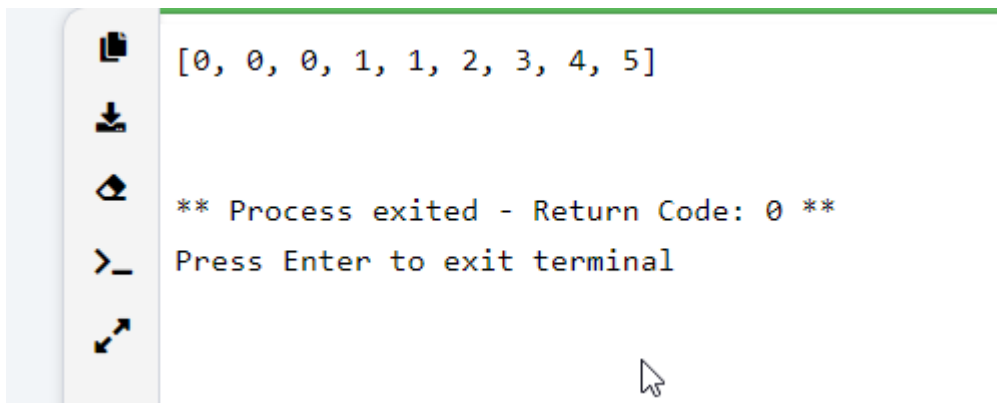
Побудувати функцію відмови і ДСА для стрічки abbaabbaa

3	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)
```



```
[0, 0, 0, 1, 1, 2, 3, 4, 5]  
  
** Process exited - Return Code: 0 **  
Press Enter to exit terminal
```

Завдання 2

Застосувати алгоритм КМП для перевірки входження ключового слова aba, як підстрічки в abaababaaa

3	abaababaaa
---	------------

```
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 = "abaababaaaa"
```

```
pattern = "aba"
```

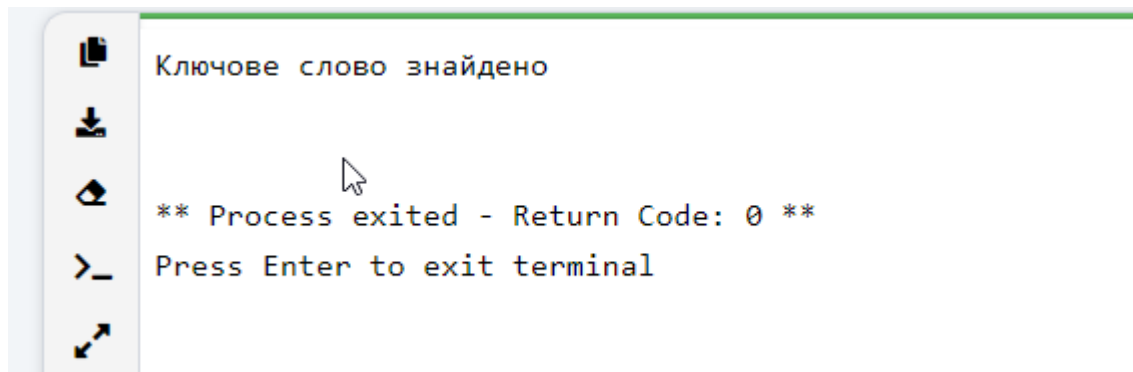
```
result = kmp(text, pattern)
```

```
if result:
```

```
    print("Ключове слово знайдено")
```

```
else:
```

```
    print("Ключове слово не знайдено")
```



Завдання 3

3. Стрічка Фібоначчі визначається наступним чином:

$s_1=b$, $s_2=a$, $s_k=s_{k-1}s_{k-2}$ при $k>2$.

Наприклад, $s_3=ab$, $s_4=aba$, $s_5=abaab$.

Визначити довжину стрічки s_n і побудувати для неї функцію відмови, де прийняти $n = \mathbb{N} + 6$.

```
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))

Стрічка Фібоначчі: abaababaabaabababababababababab

Довжина стрічки: 34

Функція відмови: [0, 0, 1, 1, 2, 3, 2, 3, 4, 5, 6, 4, 5, 6, 7, 8, 9, 10, 11, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16, 17, 18, 19, 12, 13]

Завдання 4

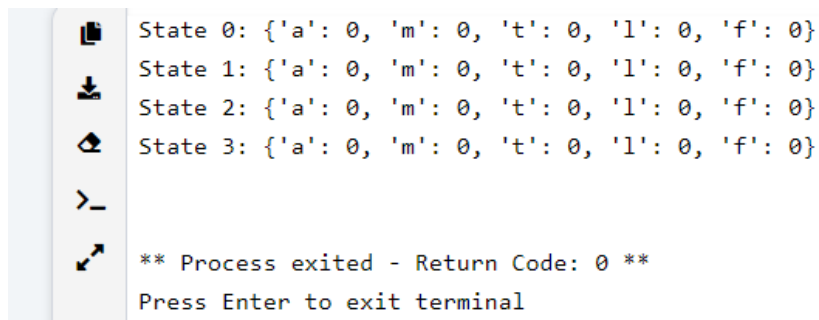
Обчислити функцію відмови множини ключових слів і відповідний ДСА: {all, fall, fatal, llama}

3	{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}")
```

A terminal window with a light blue sidebar containing icons for file operations (copy, paste, save, etc.). The terminal output shows four states of a DFA, each with a dictionary of transitions for characters 'a', 'm', 't', 'l', and 'f'. All transition values are 0. The terminal ends with a message indicating the process exited successfully.

```
State 0: {'a': 0, 'm': 0, 't': 0, 'l': 0, 'f': 0}  
State 1: {'a': 0, 'm': 0, 't': 0, 'l': 0, 'f': 0}  
State 2: {'a': 0, 'm': 0, 't': 0, 'l': 0, 'f': 0}  
State 3: {'a': 0, 'm': 0, 't': 0, 'l': 0, 'f': 0}  
  
** Process exited - Return Code: 0 **  
Press Enter to exit terminal
```

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"
```

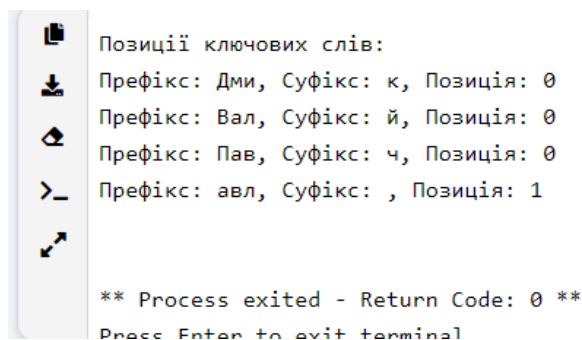
Знаходження позицій ключових слів

```
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]))
```



```
Позиції ключових слів:  
Префікс: Дми, Суфікс: к, Позиція: 0  
Префікс: Вал, Суфікс: й, Позиція: 0  
Префікс: Пав, Суфікс: ч, Позиція: 0  
Префікс: авл, Суфікс: , Позиція: 1  
  
** Process exited - Return Code: 0 **  
Press Enter to exit terminal
```