МИНОБРНАУКИ РОССИИ
САНКТ-ПЕТЕРБУРГСКИЙ ГОСУДАРСТВЕННЫЙ
ЭЛЕКТРОТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ
«ЛЭТИ» ИМ. В.И. УЛЬЯНОВА (ЛЕНИНА)
Кафедра МОЭВМ

ОТЧЕТ

по лабораторной работе №4

по дисциплине «Алгоритмы и структуры данных»

Тема: «Поиск образца в тексте: алгоритм Рабина-Карпа».

Студент гр. 1304 _____ Байков Е.С.

Преподаватель _____ Иванов Д.В.

Санкт-Петербург

2022

Цель работы

Изучить способы вычисления хэша от строк. Применить полученные знания в реализации алгоритма Рабина-Карпа.

Задание

Напишите программу, которая ищет все вхождения строки Pattern в строку Text, используя алгоритм Карпа-Рабина.

На вход программе подается подстрока Pattern и текст Text. Необходимо вывести индексы вхождений строки Pattern в строку Text в возрастающем порядке, используя индексацию с нуля.

Примечание: в работе запрещено использовать библиотечные реализации алгоритмов и структур.

Выполнение работы

В процессе реализации написано две функции: *hash_string(string)*, которая возвращает хэш строки, и *rabin_karp_algorithm(text, pattern)*, которая осуществляет алгоритм Рабина-Карпа, находя количество вхождений подстроки и выводя индексы начиная с которого подстрока находится в тексте. В функции *rabin_karp_algorithm* находятся хэши подстроки внутри текста и подстроки, которую надо найти в тексте. Кодирование происходит с помощью полиномиального кольцевого хэша и для целочисленной арифметики используется кольцо вычетов по модулю Q. Также для избегания вычислений хэша каждой подстроки была использована формула:

$$hash_{ell} = (hash_{hel} - h * x^2) * x + l$$

хэширование подстрок длины 3 в слове *hello*

Затем внутри цикла сравниваются хэши текущей подстроки из текста и искомой подстроки, если они совпадают, то проверяется совпадают ли сами строки. В случае совпадения к переменной count прибавляется 1 и вызывается функция *print* печатающая индекс подстроки в строке. Также написаны тесты (см. Приложение А).

Тестирование

Результаты тестирования представлены в табл. 1.

Таблица 1 – Результаты тестирования.

№ п/п	Входные данные	Выходные данные	Комментарии
1.	KEK AAABBBCCCKEKEKEKEKeaasasdKEKasfsdxvcxcasjdfnnKEKKEKEK	9 11 13 15 25 45 48 50	
2.	a afsdjjxcbjasdfhjbvcxlsdfjkcvnsdfjcxvbjfdadsjfkaaaa	0 10 40 46 47 48 49	
3.	aba ababaaaba	0 2 6	

Выводы

В ходе выполнения работы освоены способы вычисления хэша от строк, а также написаны программа реализующая алгоритм Рабина-Карпа и тесты с помощью *pytest*.

ПРИЛОЖЕНИЕ А
ИСХОДНЫЙ КОД ПРОГРАММЫ

Название файла: main.py

```
Q = 257
X = 17

def hash_string(string: str) -> int:
    global X, Q
    l = len(string)
    result = 0
    for i in range(l):
        result = (X * result + ord(string[i])) % Q
    return result

def rabin_karp_algorithm(text: str, pattern: str) -> int:
    global Q, X
    text_len = len(text)
    pattern_len = len(pattern)

    multiplier = 1
    for i in range(1, pattern_len):
        multiplier = (multiplier * X) % Q

    pattern_hash = hash_string(pattern)
    substring_hash = hash_string(text[:pattern_len])

    count = 0
    for i in range(text_len - pattern_len + 1):
        if pattern_hash == substring_hash and text[i : i+pattern_len]
== pattern:
            count += 1
            print(i, end=' ')

        if i < text_len - pattern_len:
            substring_hash   =   ((substring_hash   -   ord(text[i])   *
multiplier) * X + ord(text[i + pattern_len])) % Q

            if substring_hash < 0:
                substring_hash += Q

    return count

def rabin_karp_algorithm_array(text: str, pattern: str, answer: list) -
> None:
    global Q, X
    text_len = len(text)
    pattern_len = len(pattern)

    multiplier = 1
    for i in range(1, pattern_len):
        multiplier = (multiplier * X) % Q

    pattern_hash = hash_string(pattern)
```

```
    substring_hash = hash_string(text[:pattern_len])

    for i in range(text_len - pattern_len + 1):
        if pattern_hash == substring_hash and text[i : i+pattern_len]
== pattern:
            answer.append(i)

        if i < text_len - pattern_len:
            substring_hash   =   ((substring_hash   -   ord(text[i])   *
multiplier) * X + ord(text[i + pattern_len])) % Q

            if substring_hash < 0:
                substring_hash += Q
```

Название файла: tests.py

```
from main import rabin_karp_algorithm_array

def test():
    answer = []
    rabin_karp_algorithm_array('ababaaaba', 'aba', answer)
    assert answer == [0, 2, 6]
    answer.clear()

rabin_karp_algorithm_array('AAABBBCCCKEKEKEKEKeaasasdKEKasfsdxvcxcasjd
fnnKEKKEKEK', 'KEK', answer)
    assert answer == [9, 11, 13, 15, 25, 45, 48, 50]
    answer.clear()

rabin_karp_algorithm_array('afsdjjxcbjasdfhjbvcxlsdfjkcvnsdfjcxvbjfdad
sjfkaaaa', 'a', answer)
    assert answer == [0, 10, 40, 46, 47, 48, 49]
```