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

ОТЧЕТ

по лабораторной работе №4

по дисциплине «Алгоритмы и структуры данных»

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

Студент гр. 1303 _____ Кузнецов Н.А.

Преподаватель _____ Иванов Д.В.

Санкт-Петербург
2022

Цель работы.

Изучить алгоритм Рабина-Карпа и написать с помощью него поиск образца в тексте.

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.

Ограничения

$1 \leq |Pattern| \leq |Text| \leq 5 \cdot 10^5$. Суммарная длина всех вхождений образца в текста не превосходит 10^8. Обе строки содержат только буквы латинского алфавита.

Выполнение работы

В функции func_hash(str, p, degree) написана хэш-функция. Данная функция принимает подстроку, число p = 1000000007 и массив степенейчисла x = 263, строит полином и возвращает хэш-значение.

В функции solve(pattern, text) выполняется обработка строки и поиск подстроки. В самом начале происходит вычисление основных значений для обработки строки: вычисляется массив степеней числа x = 263 для того, чтобы не пересчитывать его снова, хэш-значение у последней подстроки в строке и просчитывается хэш-значение для искомой подстроки. Далее для всех подстрок вычисляется хэш-значение, причём оно выражается через следующий элемент, поэтому строка проходится с конца. В конце все полученные хэш-значения сравниваются с искомым, а также происходит проверка на идентичность найденной строки и искомой, так как возможны коллизии.

В функции main() происходит считывание строк и вывод массива индексов.

Тестирование.

Результаты тестирования представлены в табл. 1.

Таблица 1 – Результаты тестирования

№ п/п	Входные данные	Выходные данные	Комментарии
1	a aaaaaaaaaa	0 1 2 3 4 5 6 7 8 9	Верно.
2	a gopiopitopjky		Верно.
3	aba abababababababacaba	0 2 4 6 8 10 12 16	Верно.

Выводы.

В ходе данной лабораторной работы был изучен алгоритм РабинаКарпа и написан с помощью него поиск образца в тексте.

ПРИЛОЖЕНИЕ А
ИСХОДНЫЙ КОД ПРОГРАММЫ

Название файла: main.py

```
from src.modules.solve import solve

def main():
    pattern = input()
    text = input()
    answer = solve(pattern, text)
    print(*answer)
```

Название файла: hash.py

```
def func_hash(str, p, degree):
    result = 0
    for i, element in enumerate(str):
        result = (result + ord(element) * degree[i]) % p
    return result
```

Название файла: solve.py

```
from src.modules.hash import func_hash
def solve(pattern, text):
    answer = []
    p = 1000000007
    x = 263


    len_pattern = len(pattern)
    len_text = len(text)
    hash_arr = [0] * (len_text - len_pattern + 1)
    degree = [1] * (len_pattern + 1)
    for i in range(len_pattern):
        degree[i+1] = (degree[i] * x) % p
    hash_arr[-1] = func_hash(text[len_text - len_pattern:], p, degree)
    for i in range(len_text - len_pattern - 1, -1, -1):
        hash_arr[i] = (ord(text[i]) + (x * hash_arr[i+1]) % p -
(ord(text[i+len_pattern]) * degree[len_pattern-1]) * x % p) % p
    hash_pattern = func_hash(pattern, p, degree)
    for i, element in enumerate(hash_arr):
        if element == hash_pattern and pattern == text[i:
i+len_pattern]:
            answer.append(i)
    return answer
```

ПРИЛОЖЕНИЕ Б
ФАЙЛ ТЕСТИРОВАНИЯ ПРОГРАММЫ

Название файла: test.py

```
from modules.solve import solve

def test1():
assert solve('a', 'aaaaaaaaaaa') == [0, 1, 2, 3, 4, 5, 6, 7, 8, 9,
10]

def test2():
assert solve('a', 'yuopjnmoiujkliuh') == []

def test3():
assert solve('aba', 'abababababababacaba') == [0, 2, 4, 6, 8, 10,
12, 16]

def test4():
assert solve('a1', 'a11aa11aa11a') == [0, 4, 8]

def test5():
assert solve('a_', 'aaaabbaa') == []
```