**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. 1384 |  | Усачева Д.В. |
| Преподаватель |  | Иванов Д.В. |
|  |  |  |

Санкт-Петербург 2022

**Цель работы.**

Научиться работе с хэш-функцией, с её помощью решить алгоритмическую задачу.

**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 ≤ |Pattern| ≤ |Text| ≤ 5 · 105.

Суммарная длина всех вхождений образца в тексте не превосходит 108. Обе строки содержат только буквы латинского алфавита.

Подсказки**:**

1. Будьте осторожны с операцией взятия подстроки — она может оказаться дорогой по времени и по памяти.

2. Храните степени x \*\* p в списке - тогда вам не придется вычислять их каждый раз заново.

*Первой строкой добавьте #python или #c++, чтобы проверяющая система знала, каким языком вы пользуетесь.*

**Выполнение работы.**

Для решения задачи был применён алгоритм Рабина-Карпа. Суть данного алгоритма заключается в хеширование строк и сравнении полученных значений. Хэш-значение вычисляется по формуле .

Алгоритм начинается с хеширования искомой строки и подстроки длины искомой строки из исследуемой строки. Программа заходит в цикл, который проходит по всем подстрокам исследуемой строки, смещаясь каждый виток на один символ. Если хэши совпадают, в ответ записывается индекс начала подстроки в тексте. Далее берётся подстрока, сдвинутая на один символ относительно старой подстроки в исследуемой строке. Считается хэш такой подстроки и цикл повторяется.

Тестирование.

Чтобы удостовериться в правильности работы программы, она была протестирована на следующих случаях:

1. В тексте не встречается требуемой подстроки
2. Подстрока находится в начале текста
3. Подстрока находится на конце текста
4. Подстрока находится на конце текста
5. Подстрока является одной буквой
6. Подстрока длиннее текста

**Выводы.**

В ходе выполнения лабораторной работы был изучен алгоритм Рабина-Карпа, были применены навыки работы с хэш-функцией. Была успешно решена

поставленная задача

**ПРИЛОЖЕНИЕ 1**

**ИСХОДНЫЙ КОД ПРОГРАММЫ**

Название файла: main.py

#python

result = []

x = 241

p = 1000000007

def get\_hash(string):

st = 0

for j in range(len(string)):

cur = string[j]

res = result[j]

st += ord(cur) \* res

return st

def RabinKarp(pattern, text):

answer = []

txt\_len = len(text)

pat\_len = len(pattern)

if txt\_len > 0 and pat\_len > 0:

for j in range(0, len(pattern)):

result.append(x \*\* j % p)

hashPattern = get\_hash(pattern)

for j in range(0, txt\_len - pat\_len + 1):

hashText = get\_hash(text[j:j + pat\_len])

if hashText == hashPattern:

answer.append(j)

return answer

if \_\_name\_\_ == '\_\_main\_\_':

pattern = input()

text = input()

print(' '.join(list(map(str, RabinKarp(pattern, text)))))

Название файла: tests.py

from main import \*

def test\_no\_entry():

assert RabinKarp('ex', 'testnoentry') == []

def test\_prefix():

assert RabinKarp('test', 'testprefix') == [0]

def test\_suffix():

assert RabinKarp('suffix', 'testsuffix') == [4]

def test\_center():

assert RabinKarp('AA', 'testAAcenter') == [4]

def test\_letters():

assert RabinKarp('a', 'aaaaAa') == [0,1,2,3,5]

def test\_pattern\_longer\_than\_text():

assert RabinKarp('heHEhehehehhe', 'he') == []