**Київський національний університет імені Тараса Шевченка**

**Факультет комп’ютерних наук та кібернетики**

**Спеціальність програмна інженерія**

***Алгоритми та складність***

**Завдання № 8**

**Звіт**

**Виконав:**

Студент групи К-28

Саллям Салех

**Київ-2019**

*Умова завдання:*

Реалізуйте алгоритми пошуку зразка в текстовому рядку: наївний, Хорспула, Боєра-Мура, КПМ та Рабіна-Карпа і порівняйте їх ефективність. Виконайте пошук зразків різної довжини: випадкового бінарного зразка у випадковому бінарному тексті та випадкового слова у природному тексті на цій мові.

*Опис коду програми:*

*Ініціалізація:*

1. Задається рядок для пошуку;
2. Задається шуканий підрядок;

*Алгоритми:*

* *Наївний пошук.*

1.Порівнюємо перші символи підстроки і строки з індексом І;

2.Якщо вони співпали, перевіряємо наступний символ і тд.;

3.Якщо вони відмінні, то починаємо превірку спочатку збільшивши індекс на один;

4.Алгоритм завершується якщо всі символи підстроки співпали або якщо ми дійшли до (N-М+1) елементу строки.

* *Алгоритм Хорспула.*

1.Будуємо таблицю зміщень для шуканого [шаблону](https://uk.wikipedia.org/wiki/%D0%A8%D0%B0%D0%B1%D0%BB%D0%BE%D0%BD);

2.Поєднуємо початок рядка і шаблона, далі перевіряємо чи починається з останнього символу шаблону;

3.Зразок зміщуємо щодо рядка на величину, отриману з таблиці зміщень, якщо останній символ шаблону і відповідний йому при накладенні символ рядка не збігаються. Далі знову перевіряємо з останнього символу;

4.Проводимо порівняння передостаннього символу шаблону, якщо ж символи збігаються;

(Якщо всі символи шаблону збіглися з накладеними символами рядка, значить, підрядок знайден, і пошук закінчено.)

5.Зсуваємо шаблон на один символ вправо, якщо символ шаблону не збігається з відповідним символом рядка і перевіряємо знову чи починається з останнього символу.

* *Алгоритм Боєра-Мура.*

1.Спочатку будуємо таблицю зсувів для шуканого рядка;

2.Далі порівнюємо початок рядка і зразка з останнього символу зразка;

3.Зрушуємо зразок щодо рядка вправо на величину, отриману з таблиці зміщень, якщо останній символ зразка та відповідний йому символ рядка не збігаються і знову проводимо порівняння, починаючи з останнього символу зразка;

4.Проводиться порівняння передостаннього символу зразка, при збігу символів;

(Якщо всі символи зразка збіглися з відповідними символами рядка, то потрібний підрядок знайдено.)

5.Зсуваємо зразок на один символ вправо, якщо символ зразка не збігається з відповідним символом рядка і знову починаємо перевірку з останнього символу.

*Таблиця зсувів будується за такими принципами:*

1. зрушення шуканого зразка повинен бути мінімальним, таким, щоб не пропустити входження зразка в рядку;
2. якщо даний символ рядка зустрічається у зразку, то він зсувається таким чином, щоб символ рядка збігся з найбільш правим входженням цього символу в зразку;
3. якщо шукана стрічка не містить цього символу, то зразок зсувається на довжину шуканого рядка.

* *Алгоритм КПМ.*

1.Беремо значення префікс-функції pi [i] по черзі від i = 1 до n-1 (значення pi [0]=0).

2.Для підрахунку поточного значення pi [i] ми заводимо змінну j, що позначає довжину поточного розглянутого зразка. Спочатку j = pi [i-1].  
 3.Тестуємо зразок довжини j: порівнюємо символи s [j] і s [i]. Якщо вони збігаються - то вважаємо pi [i] = j+1 і переходимо до наступного індексу i+1.

4.Якщо ж символи відрізняються, то зменшуємо довжину j, вважаючи її рівною pi[j-1], і повторюємо цей крок алгоритму з початку.

5.Якщо ми дійшли до довжини j = 0 і так і не знайшли співпадіння, то зупиняємо процес перебору зразків. Вважаємо pi [i] = 0 і переходимо до наступного індексу i+1.

* *Основний алгоритм:*

1.Утворюємо рядок s + # + t, де # - це роздільник, який не повинен ніде більш зустрічатися та порахуємо для цього рядка префікс-функцію.

2.Розглянемо її значення, крім перших n + 1 (які відносяться до рядка s і раздільника). За визначенням, значення pi [i] показує найбільшу довжину підрядка, що закінчується в позиції i та співпадає з префіксом.

3. pi [i] - фактично довжина найбільшого блоку збігу з рядком s і закінчується в позиції i. Більше, ніж n, ця довжина бути не може - за рахунок роздільник.

4. Якщо pi [i] = n, то в позиції i закінчується шукане входження рядка.  
 5. Таким чином, якщо в якийсь позиції i виявилося \ pi [i] = n, то в позиції i - (n + 1) - n + 1 = i - 2 n рядка t починається чергове входження рядка s в рядок t.

* *Алгоритм Рабіна-Карпа*.

1.Створимо хеш шаблону, який ми шукаємо, і перевіримо, чи відповідає змінний хеш тексту шаблону, чи ні.

2.Якщо не відповідає, то шаблон не існує в тексті.

3.Якщо він відповідає, шаблон може бути присутнім в тексті.

*Аналіз алгоритму:*

* Наївний пошукO((N-М)\*M)
* Алгоритм Хорспула O(N+M)
* Алгоритм Боєра-Мура O(N+M)
* Алгоритм КПМ O(N+M)
* Алгоритм Рабіна-Карпа O(N+M)

*Реалізація алгоритму:*

Реалізовано на С++