Київський національний університет імені Тараса Шевченка

Факультет комп`ютерних наук та кібернетики

Кафедра інтелектуальних програмних систем

Алгоритми та складність

Завдання № 2

“ Розробіть алгоритм, який за лінійний час визначав би, чи є текстовий рядок Т циклічним зсувом іншого рядка Т\* (наприклад, abc та cab).”

Варіант № 2

Виконав студент 2-го курсу

Групи ІПС-22

Каширець Роман Віталійович

Київ - 2024

**Завдання**

Розробіть алгоритм, який за лінійний час визначав би, чи є текстовий рядок Т циклічним зсувом іншого рядка.

**Теорія**

 **Циклічний зсув** — це варіант перетворення рядка, у якому символи зсуваються циклічно, тобто частина рядка переноситься з кінця на початок.

* Наприклад, рядок "abc" має циклічні зсуви: "abc", "bca", "cab".

 Для перевірки циклічного зсуву існує ключове спостереження:  
Якщо рядок T конкатенувати із самим собою (T + T), то всі можливі циклічні зсуви T будуть підрядками цього нового рядка.

* Наприклад, якщо T = "abc", то T + T = "abcabc". Підрядки "abc", "bca", "cab" є всіма можливими зсувами.

### Алгоритм:

1. **Перевірка довжини**:
   * Якщо довжина рядків TTT і T∗T^\*T∗ не співпадає, то T∗T^\*T∗ не може бути циклічним зсувом TTT. Повертаємо false.
2. **Створення подвоєного рядка**:
   * Об’єднуємо рядок TTT із самим собою: doubledT=T+T\text{doubledT} = T + TdoubledT=T+T.
   * Усі можливі циклічні зсуви TTT містяться у doubledT\text{doubledT}doubledT як підрядки довжиною ∣T∣|T|∣T∣.
3. **Реалізація алгоритму КМП**:
   * **Побудова таблиці префіксів (lps)**:
     + Створюємо таблицю довжини ∣T∗∣|T^\*|∣T∗∣, де кожна позиція містить довжину найбільшого префікса, який є також суфіксом.
   * **Пошук рядка T∗T^\*T∗ у doubledT\text{doubledT}doubledT**:
     + Використовуємо таблицю lps, щоб оптимізувати зсув індексів при порівнянні символів T∗T^\*T∗ з doubledT\text{doubledT}doubledT.
     + Якщо знаходиться збіг довжиною ∣T∗∣|T^\*|∣T∗∣, повертаємо true.
4. **Результат**:
   * Якщо збіг знайдено, повертаємо true, інакше — false.

**Складність**

* Побудова таблиці префіксів: O(m)O(m)O(m), де mmm — довжина T\*.
* Пошук: O(n)O(n)O(n), де n=2×довжина Tn = 2 \times \text{довжина } Tn=2×довжина T.
* Загальна складність: O(n+m)≈O(n)O(n + m) \approx O(n)O(n+m)≈O(n).

**Мова реалізації алгоритму: С++**

**Модулі програми**

int main()

Містить в собі інтерфейс для взаємодії з користувачем і вся програма починається з даеної функції.

bool isCyclicShift(const string& T, const string& T\_star)

**Призначення:**

* Перевіряє, чи є рядок T\_star циклічним зсувом рядка T.

**Вхідні дані:**

* T — оригінальний рядок.
* T\_star — рядок, який перевіряється на циклічний зсув.

**Вихідні дані:**

* Повертає true, якщо T\_star є циклічним зсувом T, і false інакше.

**Логіка:**

1. Якщо довжини рядків T і T\_star не співпадають, повертає false.
2. Створює подвоєний рядок doubledT = T + T.
3. Використовує функцію KMP для перевірки, чи є T\_star підрядком doubledT.

bool KMP(const string& text, const string& pattern)

Призначення:

* Реалізує алгоритм КМП для пошуку підрядка pattern в рядку text.
* Оптимізує пошук, використовуючи таблицю префіксів lps.

Вхідні дані:

* text — рядок, у якому шукаємо підрядок.
* pattern — підрядок, який потрібно знайти.

Вихідні дані:

* Повертає true, якщо pattern є підрядком text, і false інакше.

Логіка:

1. Використовує таблицю lps, створену функцією buildKMPTable.
2. Порівнює символи pattern з символами text.
3. У разі збігу рухається далі. У разі невідповідності використовує lps для ефективного зсуву індексів.

vector<int> buildKMPTable(const string& pattern)

Призначення:

* Створює таблицю префіксів lps (Longest Prefix that is also Suffix) для рядка pattern.
* Ця таблиця використовується в алгоритмі Кнута-Морріса-Пратта (КМП), щоб оптимізувати пошук підрядків.

Вхідні дані:

* pattern — рядок, для якого будується таблиця префіксів.

Вихідні дані:

* Вектор lps довжини m, де m = pattern.length(). Кожен елемент lps[i] містить довжину найбільшого префікса рядка pattern[0..i], який є також суфіксом.

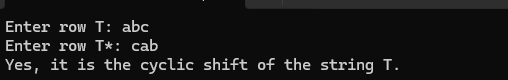
Логіка:

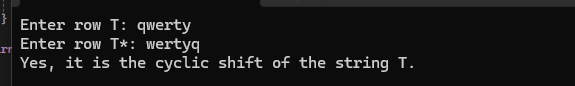
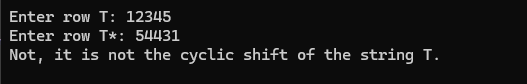
* Проходить по всьому рядку pattern.
* Якщо символи збігаються, збільшується довжина поточного префікса.
* Якщо символи не збігаються, використовується попереднє значення lps для зсуву.

**Інтерфейс користувача**

Інтерфейс складається з двох вводів користувачем даних “Enter row T” і “Enter row T\*”. І виведення інформації чи даний рядок є циклічним зсувом “Yes, it is the cyclic shift of the string T.” або не є циклічним зсувом “Not, it is not the cyclic shift of the string T.”.

**Тестові приклади**

**Висновки**

В даній роботі я створив програму яка перевіряє чи є рядок T\* циклічним зсувом рядка T. Створивши алгоритм лінійної складності О(n+m).  
Також дізнався, що таке циклічний зсув. Провів тестові випадки програми.  
Росписав все детально, щоб глядам було краще зрозуміло.  
Дякую за увагу!!!

Використані джерела:

Лекції

<https://studfile.net/preview/7818135/page:5/>

https://uk.wikipedia.org/wiki/%D0%9E%D0%B1%D1%87%D0%B8%D1%81%D0%BB%D1%8E%D0%B2%D0%B0%D0%BB%D1%8C%D0%BD%D0%B0\_%D1%81%D0%BA%D0%BB%D0%B0%D0%B4%D0%BD%D1%96%D1%81%D1%82%D1%8C