**Київський національний університет імені Тараса Шевченка**

**Факультет комп’ютерних наук та кібернетики**

**Кафедра інформаційних систем**

**Алгоритми та складність**

**Лабораторний проект № 6**

**Циклічний зсув текстового рядка**

**Звіт**

**Виконав:**

студент групи К-28

Швець Олександр Вікторович

**Київ-2018**

**Циклічний зсув текстового рядка**

1. **Умова завдання.**

Розробити алгоритм, який за лінійний час визначав би, чи є текстовий рядок Т циклічним зсувом текстового рядка Т\* (наприклад abc та cab)

1. **Опис і аналіз алгоритму.**

За основу береться алгоритм Кнута-Морріса-Пратта (КМП), який за лінійний час збуде знаходити підрядок Т\* в рядку Т. Алгоритм КМП має кращий показник продуктивності у найгіршому випадку. КМП витрачає небагато часу (за порядком розміру W[], O(n)) на попереднє обчислення таблиці, і потім використовує таблицю для швидкого пошуку рядка за час O(k).

З іншого боку, алгоритм КМП використовує інформацію про попередні порівняння. Коли KMП зустрічає незбіг на 1000-ному елементі (i = 999), тобто S[m+999] ≠ W[999], КМП знатиме, що 999 позицій вже перевірено. КМП використовує ці знання у попередньо обчисленій таблиці і додаткових змінних. Коли KMП знаходить незбіг, з таблиці префікс-функції визначається, наскільки збільшиться змінна m.

Часова асимптотична складність алгоритму становить O(N+M), де N — довжина рядка Т\*, M — довжина рядка Т.

1. **Реалізація алгоритму.**

Реалізовано на С++

1. **Інтерфейс програми.**

Дані (тести) прописані в файлі срр, а результат виводиться в консоль.

1. **Використані структури даних.**

Рядки і динамічний масив.

1. **Тестовий приклад.**

string s1 = "abbaba";

string s2 = "abaabb";

abaabb is a cyclic shift abbaba

1. **Основні модулі програми.**

//створює таблицю для префікс-функції

oid compute\_prefix(string & P,vector<int> &prefix)

//реалізація алгоритму КМП, яка перевіряє, чи є Т підрядком Р

bool kmp(string & T,string & P)

1. **Використані джерела.**

1.https://uk.wikipedia.org/wiki/%D0%90%D0%BB%D0%B3%D0%BE%D1%80%D0%B8%D1%82%D0%BC\_%D0%9A%D0%BD%D1%83%D1%82%D0%B0\_%E2%80%94\_%D0%9C%D0%BE%D1%80%D1%80%D1%96%D1%81%D0%B0\_%E2%80%94\_%D0%9F%D1%80%D0%B0%D1%82%D1%82%D0%B0