Міністерство освіти і науки України

Кременчуцький національний університет   
імені Михайла Остроградського

Навчально-науковий інститут електричної інженерії   
та інформаційних технологій

Кафедра автоматизації та інформаційних систем

НаВчальна дисципліна  
«**аЛГОРИТМИ І СТРУКТУРИ ДАННИХ**»

Звіт

З ПРАКТИЧНОЇ роботи №7

Виконав

студент групи КН-24-1

Озівський В.В.

Перевірив

доцент кафедри КІЕ

Сидоренко В. М.

Кременчук 2025

|  |  |
| --- | --- |
| Тема: | Алгоритми на рядках |
| Мета: | набути практичних навичок застосування базових алгоритмів на рядках та оцінювання їх асимптотичної складності. |

Хід роботи

Постановка завдання

Виконати індивідуальне завдання. Завдання полягає у розв’язанні задачі, яку потрібно вибрати зі списку.

Розв’язання задачі

6. Маємо дві короткі послідовності символів: «XYBAC» і «ABXC». Знайти найдовшу спільну підпослідовності символів, використовуючи алгоритм Хаббарда.

1. Алгоритм динамічного програмування

def longest\_common\_subsequence(s1, s2):  
 m = len(s1)  
 n = len(s2)  
# Створення таблиці для зберігання проміжних результатів  
 # dp[i][j] буде містити довжину найбільшої спільної підпослідовності для s1[:i] і s2[:j]  
dp = [[0] \* (n + 1) for \_ in range(m + 1)]  
  
# Заповнення таблиці знизу вгору  
for i in range(1, m + 1):  
for j in range(1, n + 1):  
if s1[i - 1] == s2[j - 1]:  
 dp[i][j] = dp[i - 1][j - 1] + 1  
else:  
 dp[i][j] = max(dp[i - 1][j], dp[i][j - 1])  
  
# Відновлення найбільшої спільної підпослідовності  
lcs = []  
 i, j = m, n  
while i >0 and j >0:  
if s1[i - 1] == s2[j - 1]:  
 lcs.append(s1[i - 1])  
 i -= 1  
j -= 1  
elif dp[i - 1][j] > dp[i][j - 1]:  
 i -= 1  
else:  
 j -= 1  
  
# Перевернення lcs, оскільки ми додавали символи з кінця  
lcs.reverse()  
return lcs

1. Тестування коду

s1 = "XYBAC"  
s2 = "ABXC"  
print("Найбільша спільна підпослідовність:",  
''.join(longest\_common\_subsequence(s1, s2)))

1. Результат роботи

Найбільша спільна підпослідовність: AC

Відповіді на контрольні питання

1. У чому полягає задача знаходження найдовшої спільної підпослідовності (LCS)?

Задача знаходження найдовшої спільної підпослідовності (LCS) полягає у пошуку найдовшої послідовності символів, яка є підпослідовністю (не обов'язково неперервною) двох або більше послідовностей.

1. Які головні методи можна використовувати для знаходження найдовшої спільної підпослідовності?

* Алгоритм динамічного програмування (класичний підхід)
* Алгоритм Хіршберга (оптимізований за пам'яттю варіант ДП)
* Алгоритми на основі суфіксних дерев
* Евристичні підходи для наближеного розв'язку

1. Як працює алгоритм динамічного програмування для знаходження LCS?

* Будує таблицю розміром (m+1)×(n+1), де m і n - довжини послідовностей
* Заповнює її поступово за формулою: якщо символи співпадають, LCS збільшується на 1, інакше береться максимум з LCS сусідніх підзадач
* Після заповнення таблиці, найдовша підпослідовність відновлюється зворотним проходом

1. Як працює алгоритм Хаббарда для знаходження LCS?
2. Які переваги та недоліки алгоритмів динамічного програмування та Хаббарда для знаходження LCS?

* Динамічне програмування: просте в реалізації, але потребує O(m×n) пам'яті
* Алгоритм Хіршберга: ефективніший за пам'яттю O(min(m,n)), але складніший у реалізації та може бути повільнішим через рекурсивні виклики

1. Які існують практичні застосування для задачі знаходження найдовшої спільної підпослідовності

* Порівняння файлів і виявлення змін (інструменти diff)
* Біоінформатика (вирівнювання ДНК і білкових послідовностей)
* Системи контролю версій
* Виявлення плагіату
* Корекція помилок у текстах
* Стиснення даних
* Розпізнавання природної мови