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 та методи обчислень

ЗВІТ

З ПРАКТИЧНОЇ РОБОТИ №3

Виконала:

студентка групи КН-23-1

Морозова Ю.О.

Кременчук 2024

**Тема.** Алгоритми сортування та їх складність. Порівняння алгоритмів сортування

**Мета:** опанувати основні алгоритми сортування та навчитись методам

аналізу їх асимптотичної складності.

**Хід роботи**

**Варіант 1**

Вивчити самостійно і записати (будь-яким способом) алгоритм бульбашкового сортування. Оцінити асимптотику алгоритму сортування методом бульбашки в найгіршому і в найкращому випадку. Порівняти за цими показниками бульбашковий алгоритм з алгоритмом сортування вставлянням. Чому на практиці бульбашковий алгоритм виявляється менш ефективним у порівнянні з сортуванням методом зливанням?

def bubble\_sort(arr):

n = len(arr)

for i in range(n):

last\_i = n - i - 1

for j in range(last\_i):

if arr[j] > arr[j + 1]:

arr[j], arr[j + 1] = arr[j + 1], arr[j]

O(n^2), Ω(1)

Hа практиці бульбашковий алгоритм виявляється менш ефективним у порівнянні з алгоритмом сортування вставкою через свою квадратичну складність та більшу кількість операцій

1. **Оцінити асимптотичну складність алгоритму сортування зливанням, скориставшись основною теоремою рекурсії.**

O(nlogn)

1. **Вивчити і записати (будь-яким способом) самостійно алгоритм швидкого сортування. Оцінити асимптотичну складність алгоритму швидкого сортування, скориставшись основною теоремою рекурсії**

def quick\_sort(arr):

if len(arr) <= 1:

return arr

else:

a = arr[0]

less\_than\_a = [x for x in arr[1:] if x <= a

greater\_than\_a = [x for x in arr[1:] if x > a]

return quick\_sort(less\_than\_a) + [pivot] + quick\_sort(greater\_than\_a)

O(nlogn)

**Контрольні питання**

1. Що таке асимптотична складність алгоритму сортування і чому вона важлива для порівняння алгоритмів?

Оцінка кількості операцій або часу, який потрібен алгоритму для сортування масиву даних.

Важлива для порівняння ефективності різних алгоритмів сортування полягає в тому, що вона дозволяє нам оцінити ефективність алгоритмів у великих масивах даних.

1. . Які алгоритми сортування мають квадратичну складність у найгіршому

випадку? Поясніть, чому це може бути проблемою для великих обсягів даних.

Сортування вибором

Сортування вставками

Сортування бульбашкою

У найгіршому випадку ці алгоритми мають квадратичну складність O(n^2), де n - кількість елементів для сортування. Це може стати проблемою для великих обсягів даних через те, що час виконання цих алгоритмів зростає квадратично зі збільшенням кількості елементів.

1. В чому полягає перевага сортування злиттям над сортуванням вставками для великих наборів даних?

Сортування вставкою має краще асимптотичну складність O(nlogn)у злиттям, коли у сортуванні вставкою O(n2)

1. Які алгоритми сортування використовуються для сортування списків у стандартних бібліотеках мов програмування, таких як Python, Java або C++?

Швидке сортування

1. Яка різниця між алгоритмами сортування злиттям і швидким сортуванням? У яких випадках краще використовувати кожен з цих алгоритмів?

Сортування злиттям має кращу асимптотичну складність у найгіршому випадку, але злиття споживає більше пам’яті через рекурсивний підхід. Злиття краще використовувати коли працюють з великими масивами та не мають обмеження пам’яті

1. Які фактори слід враховувати при виборі алгоритму сортування для конкретної задачі?

Розмір даних, пам’ять, властивості даних, рекурсія