МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ

ВІДОКРЕМЛЕНИЙ СТРУКТУРНИЙ ПІДРОЗДІЛ  
«ТЕХНІЧНИЙ ФАХОВИЙ КОЛЕДЖ

ЛУЦЬКОГО НАЦІОНАЛЬНОГО ТЕХНІЧНОГО УНІВЕРСИТЕТУ»

Випускова циклова комісія Компʼютерних систем та інформаційних технологій

**Звіт**

**з практичної роботи №4**

з дисципліни “Алгоритми та структури даних”

Виконав: студент 2 курсу, групи 22-ІСТ

Синиця Назар

Викладач: Володимир Остапюк

Здано:

Луцьк – 2024

**Зміст**

**Теорія та дослідження:**

1. Опишіть принцип роботи алгоритму сортування вставкою. Як відбувається вибір елемента та його вставка на правильну позицію?
2. Опишіть принцип роботи алгоритму швидкого сортування. Як вибирається опорний елемент? Як відбувається розбиття масиву на підмасиви?
3. Яка часова складність алгоритму сортування вставкою в найгіршому, середньому та найкращому випадках? В яких випадках цей алгоритм є ефективним?
4. Яка часова складність алгоритму швидкого сортування в найгіршому, середньому та найкращому випадках? Від чого залежить ефективність цього алгоритму?
5. Порівняйте алгоритми сортування вставкою та швидкого сортування. В яких ситуаціях доцільно використовувати кожен з них?

**Практичні завдання (Python):**

1. Реалізуйте функцію insertion\_sort(array\_to\_sort), яка сортує список array\_to\_sort за допомогою алгоритму сортування вставкою.
2. Реалізуйте функцію quick\_sort(array\_to\_sort), яка сортує список array\_to\_sort за допомогою алгоритму швидкого сортування.
3. Напишіть програму, яка зчитує список чисел з клавіатури та сортує його за допомогою обох алгоритмів. Виведіть відсортований список та кількість операцій порівняння та обміну, виконаних кожним алгоритмом.
4. Порівняйте ефективність алгоритмів сортування вставкою та швидкого сортування на списках різної довжини та з різним ступенем впорядкованості. Виміряйте час виконання(або кількість операцій) кожного алгоритму та [\*]побудуйте графік залежності часу(або кількості операцій) виконання від розміру та впорядкованості списку.

**Висновок**

**Теорія та дослідження:**

1. Це принцип роботи алгоритму сортування вставкою. Як відбувається вибір елемента та його вставка на правильну позицію?

**Алгоритм сортування вставкою** працює шляхом поступового "вставлення" елементів на їх правильні позиції в частково відсортованому списку. Він починає з другого елемента і порівнює його з попереднім елементом, якщо потрібно — переставляє їх місцями. Потім переходить до наступного елемента і продовжує цей процес до кінця списку, кожного разу вставляючи поточний елемент на правильне місце серед попередніх. Цей процес повторюється, поки весь список не буде відсортований.

Вибір елемента для вставки здійснюється по черзі, і на кожному кроці перевіряються всі попередні елементи, поки не знаходиться правильна позиція для поточного елемента.

1. Опишіть принцип роботи алгоритму швидкого сортування. Як вибирається опорний елемент? Як відбувається розбиття масиву на підмасиви?

**Алгоритм швидкого сортування:** Алгоритм швидкого сортування працює за принципом поділу масиву на менші підмасиви на основі вибору опорного елемента. Спочатку вибирається опорний елемент, і всі елементи, менші за нього, переміщуються вліво, а більші — вправо. Це розбиття масиву називається "розділенням". Після цього алгоритм рекурсивно застосовується до кожного підмасиву. Цей процес повторюється до тих пір, поки підмасиви не стануть дуже малими.

Опорний елемент може бути вибраний різними способами, найпоширенішими є:

* перший елемент,
* останній елемент,
* випадковий елемент,
* медіана трьох елементів (перший, останній і середній).

1. Яка часова складність алгоритму сортування вставкою в найгіршому, середньому та найкращому випадках? В яких випадках цей алгоритм є ефективним?

* **Найгірший випадок:** O(n2), коли список відсортований у зворотному порядку.
* **Середній випадок:** O(n2), якщо елементи розподілені випадковим чином.
* **Найкращий випадок:** O(n), якщо список вже відсортований.

**Алгоритм сортування вставкою** є ефективним на малих або майже відсортованих масивах, оскільки він працює швидше за інші алгоритми при малих розмірах даних.

1. Яка часова складність алгоритму швидкого сортування в найгіршому, середньому та найкращому випадках? Від чого залежить ефективність цього алгоритму?

* **Найгірший випадок:** **O(n2),** коли опорний елемент вибрано неправильно, і масив не розділяється рівномірно (наприклад, для вже відсортованих масивів).
* **Середній випадок:** **O(n log n),** коли масив розділяється на приблизно рівні частини.
* **Найкращий випадок:** **O(n log n),** при хорошому виборі опорних елементів.

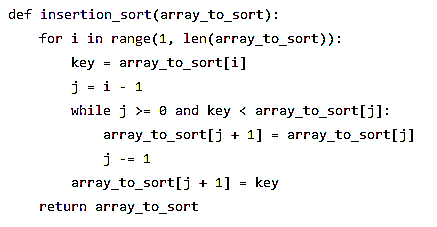
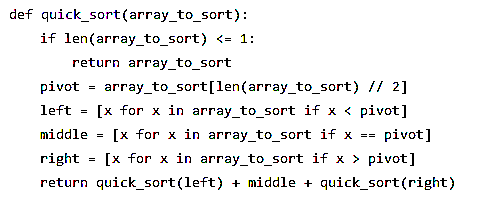
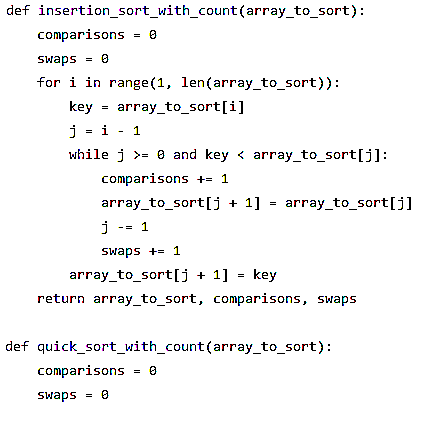
Ефективність алгоритму залежить від вибору опорного елемента, і чим рівномірніше масив розподіляється на підмасиви, тим краща ефективність.

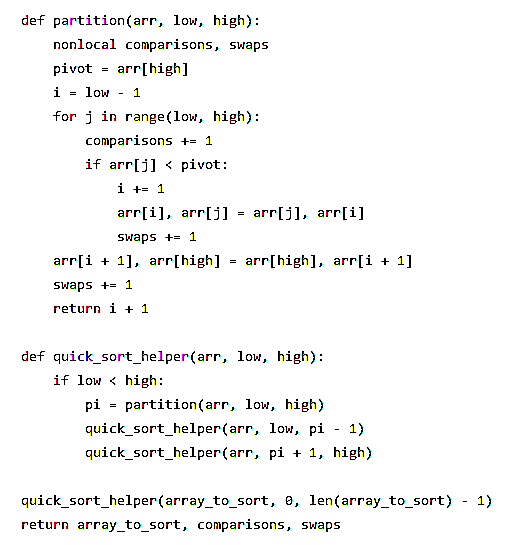
1. Порівняйте алгоритми сортування вставкою та швидкого сортування. В яких ситуаціях доцільно використовувати кожен з них?

**Алгоритм сортування** вставкою підходить для малих або майже відсортованих масивів, коли швидкість важлива для невеликих наборів даних.

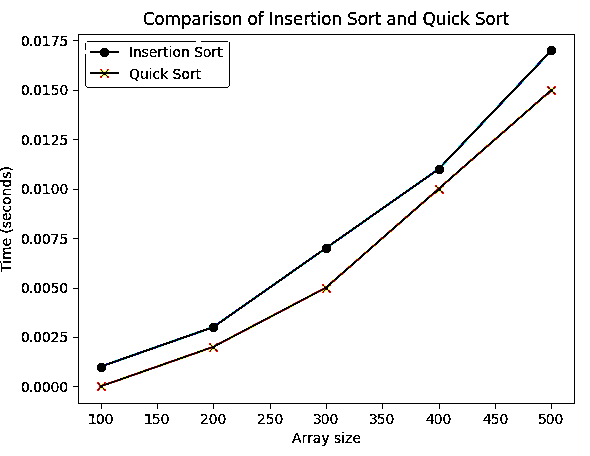
**Алгоритм швидкого сортування** є ефективнішим на більших масивах, особливо коли потрібно обробляти великі кількості елементів, де його середня складність **O(n log n)** є значно кращою, ніж квадратична складність сортування вставкою.

**Практичні завдання (Python):**

1.  Реалізуйте функцію insertion\_sort(array\_to\_sort), яка сортує список array\_to\_sort за допомогою алгоритму сортування вставкою.
2.  Реалізуйте функцію quick\_sort(array\_to\_sort), яка сортує список array\_to\_sort за допомогою алгоритму швидкого сортування.
3.  Напишіть програму, яка зчитує список чисел з клавіатури та сортує його за допомогою обох алгоритмів. Виведіть відсортований список та кількість операцій порівняння та обміну, виконаних кожним алгоритмом.



1. Порівняйте ефективність алгоритмів сортування вставкою та швидкого сортування на списках різної довжини та з різним ступенем впорядкованості. Виміряйте час виконання(або кількість операцій) кожного алгоритму та [\*]побудуйте графік залежності часу(або кількості операцій) виконання від розміру та впорядкованості списку.



**Висновок**

Алгоритм сортування вставкою є дуже простим і ефективним для маленьких або майже відсортованих масивів, але його часова складність в найгіршому випадку O(n²) робить його менш ефективним для великих масивів.

Швидке сортування є значно ефективнішим для великих масивів, оскільки його середня та найкраща складність становить O(n log n), однак ефективність залежить від вибору опорного елемента.

Практичні вимірювання та порівняння цих алгоритмів на різних розмірах та впорядкованостях масивів допомагають зрозуміти, коли кожен алгоритм є доцільним для використання.