### ****Практична робота №3****

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

**сортування**

### ****1. Алгоритм бульбашкового сортування:****

Алгоритм передбачає послідовне порівняння сусідніх елементів масиву та їх обмін, якщо вони розташовані в неправильному порядку.

#### Псевдокод алгоритму:

function bubbleSort(arr):

n = length(arr)

for i from 0 to n-1:

for j from 0 to n-i-2:

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

swap(arr[j], arr[j+1])

#### Основна ідея:

* Кожна ітерація "спливає" найбільший елемент у кінець списку, як бульбашка.
* Повторюється до тих пір, поки масив не буде відсортований.

### ****2. Асимптотична складність алгоритму:****

#### У найгіршому випадку:

* Найгірший випадок: масив відсортований у зворотному порядку.
* Для кожного з n елементів виконується n−i−1n-i-1 порівнянь, тобто: T(n)=∑i=1n−1i=n(n−1)2≈O(n2)T(n) = \sum\_{i=1}^{n-1} i = \frac{n(n-1)}{2} \approx O(n^2)

#### У найкращому випадку:

* Найкращий випадок: масив вже відсортований.
* Для оптимізованої версії алгоритму, якщо після першої ітерації немає жодного обміну, виконання зупиняється: O(n)O(n)

#### У середньому випадку:

* У середньому випадку алгоритм також виконує приблизно O(n2)O(n^2).

### ****3. Порівняння з алгоритмом сортування вставлянням:****

#### Алгоритм сортування вставлянням:

Сортування вставлянням формує відсортовану частину масиву, вставляючи кожен новий елемент у правильну позицію.

#### Складність алгоритму вставляння:

* Найгірший випадок: O(n2)O(n^2), якщо масив відсортований у зворотному порядку.
* Найкращий випадок: O(n)O(n), якщо масив вже відсортований (менше зсувів елементів).
* У середньому випадку: O(n2)O(n^2).

#### Чому вставляння швидше

* Алгоритм сортування вставлянням має кращу продуктивність для майже відсортованих даних завдяки меншій кількості обмінів.

### ****4. Чому бульбашковий алгоритм менш ефективний, ніж сортування злиттям?****

#### Сортування злиттям

* Рекурсивний алгоритм, що розділяє масив на частини, сортує їх та зливає.
* Складність: O(nlog⁡n)O(n \log n) у всіх випадках.

#### Причини меншої ефективності бульбашкового сортування

1. **Більше порівнянь і обмінів**: Бульбашкове сортування виконує забагато порівнянь навіть для майже відсортованих даних.
2. **Неоптимальна структура**: Алгоритм не використовує розподіл задач, як у сортуванні злиттям, де масив ділиться на менші частини.
3. **Складність у реальному використанні**: Для великих масивів O(n2)O(n^2) є значно менш ефективним, ніж O(nlog⁡n)O(n \log n).

### ****Висновки****

1. **Бульбашкове сортування** добре підходить лише для навчання або дуже малих масивів.
2. **Сортування вставлянням** виграє у бульбашкового в практичних випадках, але теж поступається сортуванню злиттям.
3. **Сортування злиттям** є ефективнішим для великих даних через логарифмічний порядок складності.