**МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ**

**НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ «ЛЬВІВСЬКА ПОЛІТЕХНІКА»**

Інститут **ІКНІ**

Кафедра **ПЗ**

**ЗВІТ**

До лабораторної роботи №7

На тему: «Порівняння методів сортування»

З дисципліни: «Алгоритми та структури даних»

**Лектор** : доцент каф.ПЗ

Коротєєва Т.О.

**Виконала:** ст.гр.ПЗ-23

Кохман О.В.

**Прийняв:** асистент каф.ПЗ

Франко А.В.

«\_\_\_\_»\_\_\_\_\_\_\_\_\_\_2022 р.

\_ \_ \_ \_ \_ \_ \_ \_ .

Львів – 2022

**Тема:** Порівняння методів сортування.

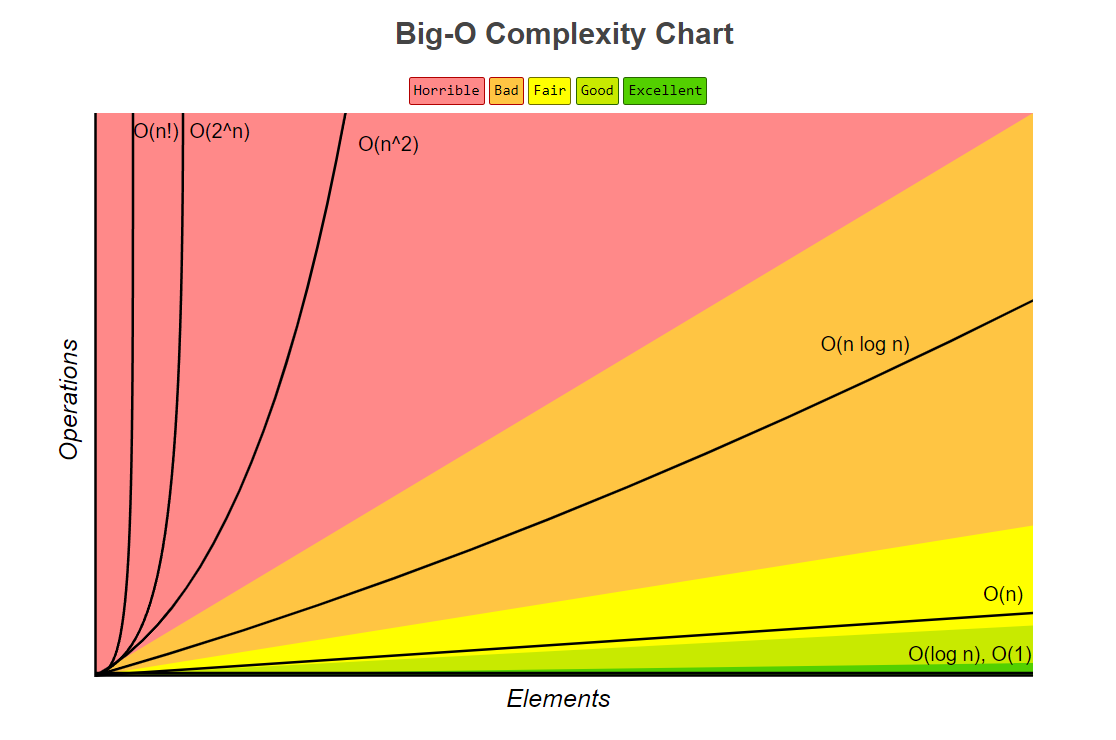
**Мета:**  Порівняти вивчені раніше алгоритми сортування. Побудувати таблицю і графік швидкодії таких алгоритмів сортування. Зробити висновки щодо застосовності цих алгоритмів.

**Теоретичні відомості**

**Алгори́тм** — набір інструкцій, які описують порядок дій виконавця, щоб досягти результату розв’язання задачі за скінченну кількість дій; система правил виконання дискретного процесу, яка досягає поставленої мети за скінченний час. Для візуалізації алгоритмів часто використовують блок-схеми.Для комп’ютерних програм алгоритм є списком деталізованих інструкцій, що реалізують процес обчислення, який, починаючи з початкового стану, відбувається через послідовність логічних станів, яка завершується кінцевим станом. Перехід з попереднього до наступного стану не обов'язково детермінований — деякі алгоритми можуть містити елементи випадковості.

Поняття алгоритму належить до підвалин математики. Обчислювальні процеси алгоритмічного характеру (як-то арифметичні дії над цілими числами, знаходження НСД двох чисел тощо) відомі людству з глибокої давнини. Проте, чітке поняття алгоритму сформувалося лише на початку ХХ століття.

**Складність обчислювальних процесів** — це поняття теорії складності обчислень, оцінка ресурсів (зазвичай часу) необхідних для виконання алгоритму.



**Індивідуальне завдання**

1. Відвідати лекцію, вислухати та зрозуміти пояснення лектора. Прочитати та зрозуміти методичні вказівки, рекомендовані джерела та будь-які інші матеріали, що можуть допомогти при виконанні лабораторної роботи. Відвідати лабораторне заняття, вислухати та зрозуміти рекомендації викладача.

2. Скомпілювати всі шість раніше написаних програм

3. Запустити на виконання кожну з написаних раніше програм щонайменше сім разів, отримати таким чином значення часу сортування масивів щонайменше семи різних розмірів кожним з шести вивчених методів. В якості набору значень розмірів масивів використати таку послідовність чисел:

1) 1024;

2) 4096;

3) 16384;

4) 65536;

5) 262144;

6) 1048576;

7) 4194304 (в разі якщо сортування відбувається довше, ніж 5 хвилин — переривати роботу програми та вважати час сортування нескінченно великим).

Кожний масив наповнити даними за допомогою функції рандомізації та записати у файл.

4. Оформити звіт про виконання лабораторної роботи.

7) за часовими результатами сортування найбільшого за розміром масиву за допомогою графічних засобів середовища програмування намалювати діаграму порівнянь (стовпчикову або секторну) ;

5. Захистити звіт про виконання лабораторної роботи. Процедура захисту передбачає перевірку оформлення звіту та відповіді на будь-яку кількість будь-яких запитань викладача, що так чи інакше стосуються теми лабораторної роботи.

**Протокол роботи**

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **sort/el.** | **1024** | **4096** | **16384** | **65536** | **262144** | **1048576** | **4194304** |
| **bubble** | 0.0040 | 0.0490 | 0.8960 | 16.6280 | 263.7610 | **∞** | **∞** |
| **selection** | 0.0010 | 0.0090 | 0.1690 | 2.3870 | 40.2910 | ∞ | ∞ |
| **shell** | 0.0010 | 0.0010 | 0.0060 | 0.0170 | 0.1190 | 0.5480 | 2.5050 |
| **quick** | 0.0000 | 0.0010 | 0.0030 | 0.0280 | 0.3870 | 5.5040 | 86.0790 |
| **merge** | 0.0010 | 0.0040 | 0.0160 | 0.0480 | 0.2130 | 0.7940 | 2.8200 |
| **counting** | 0.0020 | 0.0060 | 0.0130 | 0.0410 | 0.1480 | 0.5400 | 2.6880 |

Табл. 1 Значення часу сортування різних алгоритмів при різній кількості елементів.

Рис. 1 Графік часу сортування різних алгоритмів при різній кількості елементів**.**

Рис. 1 Стовпчаста діаграма часу сортування різних алгоритмів при різній кількості елементів**.**

**Висновок**

На даній лабораторній роботі я порівняла час виконання усіх алгоритмів сортування у таблиці, написаних у минулих лабораторних роботах. Отже, алгоритм сортування бульбашкою, який має швидкодію у найкращому випадку O(n), а у найгіршому – O(n^2), не доцільно використовувати при великій кількості даних, адже він тоді має низьку ефективність. Алгоритм варто застосовувати або у добре відсортованих заздалегідь вибірках або ж при малій кількості елементів, при порівнянні за допомогою рандомної вибірки працює найгірше. Алгоритм сортування вибором, який має швидкодію O(n^2) в усіх випадках, є таким ж ж простим як і алгоритм сортування бульбашкою, але вимагає набагато менше переміщень, тому продуктивність алгоритму є значно вищою, але також не доцільно використовувати його при великій кількості елементів. Алгоритм сортування Шелла, який має швидкодію O(n\*log n) в кращому випадку і O(n\* log n^2) в гіршому випадку, є покращеним варіантом алгоритму вставкою і час залежить від вибірки. При рандомній вибірці працює найкраще з усіх алгоритмів. Алгоритм швидкого сортування, який має швидкодію у кращому випадку O(n\*log n), у гіршому – O(n^2), вважається швидшим алгоритмом , аніж якщо порівнювати з іншими алгоритмами з такою ж швидкодією. Час роботи алгоритму залежить від збалансованості розбиття. Якщо розбиття недобре збалансоване, то алгоритм вважається не кращим алгоритму сортування вставками, при рандомній вибірці працює краще, аніж сортування бульбашкою і сортування вибором, але працює гірше, аніж сортування Шелла, сортування злиттям та сортування підрахунком. Алгоритм сортування злиттям, який має швидкодію O(n\*log n) в усіх випадках, є другим найкращим алгоритмом при рандомній вибірці. Також алгоритм потребує додаткової пам’яті. Алгоритм сортування підрахунком, який має швидкодію в усіх випадках O(n+k), застосовується, коли є мала кількість ключів, тому недоцільно використовувати його з іншими типами , окрім цілих, потребує k додаткової пам’яті. В порівнянні з іншими алгоритмами, використаними в даній лабораторній роботі, є приблизно таким же ж ефективним, як і алгоритм сортування Шелла і алгоритм сортування злиттям. Також, в цій лабораторній роботі я побудувала графік часу виконання алгоритмів з різною кількістю елементів.