МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
НАЦІОНАЛЬНИЙ УНІВЕРСТИТЕТ «ЛЬВІВСЬКА ПОЛІТЕХНІКА»

Інститут комп'ютерних наук та інформаційних технологій
Кафедра програмного забезпечення

ЗВІТ

до лабораторної роботи №7

на тему: «Порівняння методів сортування»

з дисципліни: «Алгоритми і структури даних»

Лектор:
доц. кафедри ПЗ
Коротєєва Т. О.

Виконав:
ст. гр. ПЗ-22
Чаус О. М.

Прийняв:
асист. кафедри ПЗ
Франко А. В.

« ____ » _____ 2022 р.

∑= _____ .....:

Львів-2022

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

Теоретичні відомості

Алгоритм, (латинізоване «Algorithmi», від імені узбецького математика IX століття аль-Хорезмі) — система правил виконання обчислювального процесу, що обов'язково приводить до розв'язання певного класу задач після скінченного числа операцій. При написанні комп'ютерних програм алгоритм описує логічну послідовність операцій. Поняття алгоритму належить до первісних понять математики. Обчислювальні процеси алгоритмічного характеру (арифметичні дії над цілими числами, знаходження найбільшого спільного дільника двох чисел тощо) відомі людству з глибокої давнини. Проте в явному вигляді поняття алгоритму сформувалося лише на початку XX століття.

Термін сортування (англійською «Sorting») означає розділення елементів за певними ознаками (сортами) і не дуже точно описує поставлену задачу. Більш точним була б назва впорядкування (англійською «Ordering»), але через перевантаженість слова «порядок» (англійською «Order») різними значеннями, в цьому контексті ним не скористалися.

Алгоритм сортування — це алгоритм, що розв'язує задачу сортування, тобто здійснює впорядкування лінійного списку (масиву) елементів.

На практиці елементи, що впорядковуються, рідко бувають просто числами. Набагато частіше, кожен такий елемент є записом (англійською «Record»). В кожному записі є ключ (англійською «Key»), по якому власне і здійснюється впорядкування, в той же час є й інша супутня інформація. Алгоритм сортування на практиці має бути реалізований так, щоб разом з ключами переміщати і супутню інформацію. Якщо кожен запис містить супутню інформацію великого об'єму, то з метою звести до мінімуму переписування великих об'ємів інформації, впорядкування відбувається не у самому масиві елементів, а в масиві вказівників на елементи.

Сам метод сортування не залежить від того, чи впорядковуються тільки числа, чи також і супутня інформація, тому при описі алгоритмів для простоти припускають, що елементи є числами.

Для алгоритму сортування (як і для будь-якого іншого сучасного алгоритму) основними характеристиками є обчислювальна та ємнісна складність. Крім цих двох характеристик, сортування поділяють на стабільні та нестабільні, з використанням додаткової інформації про елементи, чи без використання.

Стабільним (англійською «Stable») називається такий алгоритм сортування, що не змінює порядок елементів з однаковим ключем.

Для значної кількості алгоритмів середній і найгірший час впорядкування масиву з n елементів є O(n2), це пов’язано з тим, що в них передбачені перестановки елементів, що стоять поряд (різниця між індексами елементів не перевищує деякого заданого числа). Такі алгоритми зазвичай є стабільними, хоча і не ефективними для великих масивів.

Інший клас алгоритмів здійснює впорядкування за час O(n·log(n)). В цих алгоритмах використовується можливість обміну елементів, що знаходяться на будь-якій відстані один від одного.
Ще один клас алгоритмів використовує в своїй роботі деяку додаткову інформацію про елементи, що впорятковуються (наприклад, те що вони є різними числами в деякому діапазоні). Завдяки цьому, вони працюють за час O(n).

Індивідуальне завдання

. 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. Звіт повинен бути надрукований з однієї сторони аркушів формату А4 шрифтом 12 кеглю з одинарним інтерліньяжем та скріплений за допомогою степлера. Правильно оформлений звіт обов’язково повинен містити такі складові частини:
1) титульний лист, на якому вказуються:
1) назва міністерства, навчального закладу та структурного підрозділу, в котрому було виконано лабораторну роботу;
2) номер та назва лабораторної роботи, яку було виконано;
3) групу, ім’я та прізвище студента, який виконав лабораторну роботу;
4) посаду, ім’я та прізвище викладача, який прийняв лабораторну роботу;
5) місто та рік в яких було виконано лабораторну роботу;
2) тему роботи;

3) мету роботи;
4) короткі теоретичні відомості стосовно функції фіксації часу виконання;
5) таблицю значень часу сортування для різних методів та різних розмірів масивів (рядки — методи сортування, стовпці — розміри масивів, комірки — значення часу сортування);
6) графік часу сортування з шістьма підписаними кривими різної товщини та стилю для шести різних методів сортування (вісь абсцис — розмір масиву, вісь ординат — час сортування);
7) за часовими результатами сортування найбільшого за розміром масиву за допомогою графічних засобів середовища програмування намалювати діаграму порівнянь (стовпчикову або секторну) ;
8) розгорнуті висновки обсягом не менше 250 слів (результати порівняння різних методів сортування, зауваження щодо їх характеристик, рекомендації щодо доцільності їх застосування).

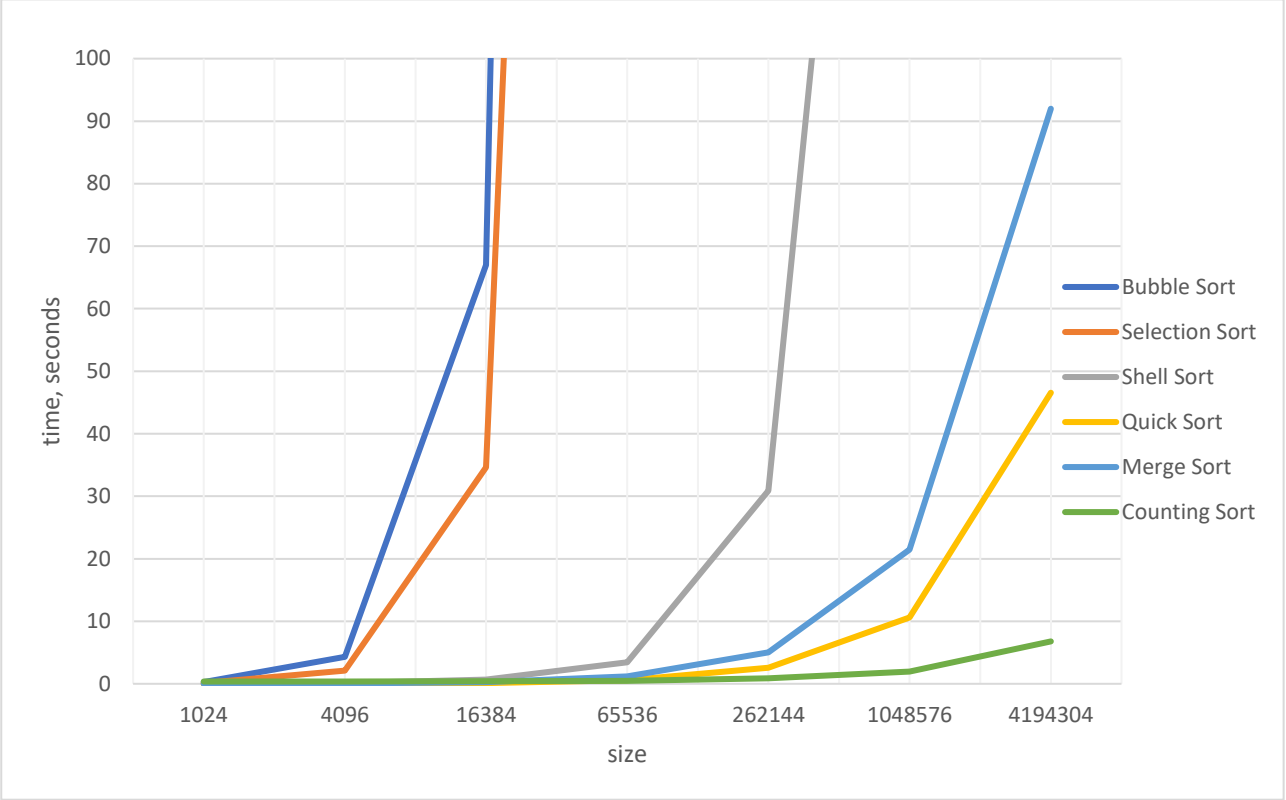
5. Захистити звіт про виконання лабораторної роботи. Процедура захисту передбачає перевірку оформлення звіту та відповіді на будь-яку кількість будь-яких запитань викладача, що так чи інакше стосуються теми лабораторної роботи.

Хід роботи

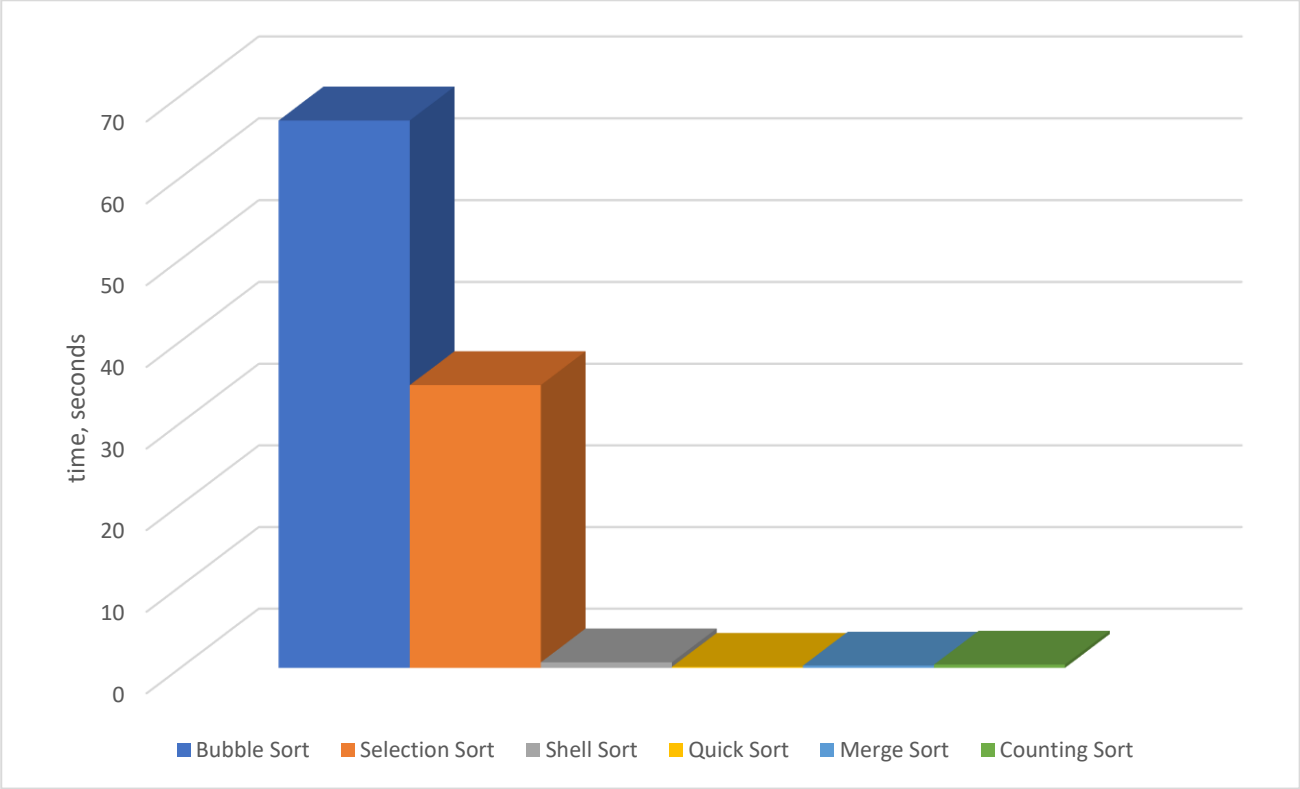
Таблиця часу сортування для всіх методів сортування:

size / sort	1024	4096	16384	65536	262144	1048576	4194304
Bubble Sort	0.280 s.	4.290 s.	66.990 s.				
Selection Sort	0.130 s.	2.120 s.	34.650 s.				
Shell Sort	0.012 s.	0.094 s.	0.680 s.	3.440 s.	30.890 s.	255.350 s.	
Quick Sort	0.006 s.	0.029 s.	0.127 s.	0.590 s.	2.570 s.	10.610 s.	46.560 s.
Merge Sort	0.014 s.	0.062 s.	0.280 s.	1.190 s.	5.019 s.	21.450 s.	91.990 s.
Counting Sort	0.370 s.	0.380 s.	0.400 s.	0.470 s.	0.860 s.	1.950 s.	6.780 s.

Графік часу сортування:



Діаграма часу сортування:



Висновок:

під час виконання лабораторної роботи дослідив швидкодію усіх вивчених раніше алгоритмів сортування та визначив переваги та недоліки кожного алгоритму.

Bubble Sort:

Алгоритм сортування бульбашкою є одним з найгірших алгоритмів. Враховуючи, що його швидкодія дорівнює $O(n^2)$, неможливо придумати ситуації, де цей алгоритм був би найдоцільнішим вибором. Проте, варто відзначити його простоту реалізації, що дозволяє навіть найнедосвідченішим програмістам його зрозуміти та реалізувати.

Selection Sort:

Хоча швидкодія алгоритму сортування вибором дорівнює $O(n^2)$, все ж він більш ефективний, ніж алгоритм бульбашки. Однак, порівняно з іншими алгоритмами, все ж він залишається одним з найгірших.

Shell Sort:

Швидкодія алгоритму Шелла залежить від вибору проміжку між елементами та внутрішнього алгоритму сортування. В середньому ця швидкодія рівна $O(n^{1.5})$. Алгоритм Шелла є швидшим за алгоритми бульбашки та вибірки, тому його доцільніше використовувати.

Quick Sort:

Алгоритм швидкого сортування є одним з найшвидших алгоритмів з швидкодією $O(n \log n)$, та, відповідно одним з найпопулярніших. Серед усіх вивчених поступається лише алгоритму сортування підрахунком.

Merge Sort:

Алгоритм сортування злиттям, разом з алгоритмом швидкого сортування, є алгоритмами Divide-and-Conquer. Хоча його швидкодія близька до алгоритму швидкого сортування, він також використовує $O(n)$ пам'яті, тому його недоцільно використовути на системах з надто обмеженою пам'яттю.

Counting Sort:

Алгоритм сортування підрахунком є надзвичайно швидким, однак на ситуації, де можна використати цей алгоритм, накладаються декілька обмежень. Алгоритм використовує додатковий масив для підрахунку кількості появи всіх значень у масиві і його швидкодія дорівнює $O(n + k)$, де n – розмір вхідного масиву, k – розмір масиву кількості появи кожного зі значень. Тому, пам'ять, яка буде використана може дуже сильно варіюватися, і залежить це від діапазону значень у масиві. Через це найкраще використовувати цей алгоритм з цілочисельними даними та невеликим діапазоном значень. Також, варто брати до уваги використовувану пам'ять на системах, де пам'ять обмежена.