ОДЕСЬКИЙ НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ ІМЕНІ І.І.МЕЧНИКОВА Факультет математики, фізики та інформаційних технологій

**Звіт з лабораторної роботи № 2**

з дисципліни: “Алгоритми та структури даних” на тему:

“Аналіз алгоритмів сортування”

Виконав: студент 2 курсу

денної форми навчання

спеціальності 123 «Комп'ютерна інженерія»

Мовчан Ілля Олександрович

**Лабораторна робота № 2**

**Тема:** Діагностика роботи алгоритмів сортування масивів.

**Мета**: Реалізувати сортування у вигляді статичного методу, що приймає масив та змінює його, сортуючи за неспаданням. Також виміряти результати швидкості виконання кожного з алгоритмів.

**Програмне забезпечення:** Windows 10, Visual Studio 2022

**Постановка задачі**

1. Реалізувати сортування за допомогою статичних методів.
2. Порівняти їх швидкість та побудувати таблицю залежності часу виконання від n для декількох реалізацій (опціонально побудувати відповідні графіки).
3. Скласти звіт, куди включити ці дані.

**Завдання**

1. [Стандартне сортування (Array.Sort())](#1epateoo5dqm)
2. [“Бульбашкове” сортування](#kix.lj77p0bkep2j)
3. [Сортування з вибором](#26x7qymwtn4)
4. [Сортування з використанням linq](#pwwq5cwtjygj)

**Посилання на код**

<https://github.com/Ilmovchan/algorithm/tree/115c65250bab5fd116fd73bb612b14da9145669b/lab%20%232>

**Стандартне сортування**

Найпростіший алгоритм сортування - це використання методу Array.Sort(). Я згенерував масив із випадкових int значень і змінював кількість елементів відстежуючи швидкість виконання методу.

Отже розглянемо як я його використав:

public static int[] DefaultSort(int[] originalArr)

{

int[] defaultSortArr = new int[originalArr.Length];

Array.Copy(originalArr, defaultSortArr, originalArr.Length);

Array.Sort(defaultSortArr, (a, b) => b.CompareTo(a));

return defaultSortArr;

}

| Кількість елементів | Час |
| --- | --- |
| 50 | 6651 |
| 100 | 10046 |
| 1000 | 119334 |
| 10000 | 1574311 |

**“Бульбашкове”** **сортування**

Бульбашкове сортування - алгоритм сортування, який з кожною ітерацією порівнює 2 сусідні елементи та якщо наступний елемент менше за попередній здійснює своп. Таким чином кожної ітерації найбільший елемент випливає у кінець масива, сортуючи його.

Розглянемо цей алгоритм:

public static int[] BubbleSort(int[] originalArr)

{

int[] arr = originalArr.ToArray();

for (int j = 0; j < arr.Length - 1; j++)

{

bool swapped = false;

for (int i = 0; i < arr.Length - j - 1; i++)

{

if (arr[i] < arr[i + 1])

{

int tmp = arr[i];

arr[i] = arr[i + 1];

arr[i + 1] = tmp;

swapped = true;

}

}

if (!swapped) return arr;

}

return arr;

}

| Кількість елементів | Час |
| --- | --- |
| 50 | 18011 |
| 100 | 44863 |
| 1000 | 5118786 |
| 10000 | 451807617 |

**Сортування з вибором**

Сортування з вибором - алгоритм сортування, кожну ітерацію знаходить найбільше значення у масиві та здійснює обмін його із останнім елементом масиву. Таким чином виконується сортування.

Розглянемо цей алгоритм:

public static int[] SelectionSort(int[] originalArr)

{

int[] arr = originalArr.ToArray();

for (int j = 0; j < arr.Length; j++)

{

int minIndex = j;

for (int i = j; i < arr.Length; i++)

{

if (arr[minIndex] < arr[i])

{

minIndex = i;

}

}

int tmp = arr[j];

arr[j] = arr[minIndex];

arr[minIndex] = tmp;

}

return arr;

}

| Кількість елементів | Час |
| --- | --- |
| 50 | 7281 |
| 100 | 23203 |
| 1000 | 1759524 |
| 10000 | 181996682 |

**Linq-сортування**

Сортування з використанням linq - алгоритм сортування, який використовує бібліотеку linq та її метод OrderByDescending(). Це дуже не оптимізований варіант сортування, тому він був використаний для демонстрації різниці швидкості сортування.

Розглянемо цей алгоритм:

public static int[] LinqSort(int[] originalArr)

{

int[] linqSortArr = new int[originalArr.Length];

Array.Copy(originalArr, linqSortArr, originalArr.Length);

IEnumerable<int> sortedNumbersDescending = linqSortArr.OrderByDescending(number => number);

return sortedNumbersDescending.ToArray();

}

| Кількість елементів | Час |
| --- | --- |
| 50 | 61890 |
| 100 | 73491 |
| 1000 | 250690 |
| 10000 | 3023272 |

**Висновок**

Я провів дослідження швидкості деяких методів сортування масивів і виявив декілька закономірностей. Можна зробити висновок що у великих масивах є сенс використовувати **стандартне сортування** та **сортування з вибором**, оскільки вони виявилися найбільш швидкими.

Але зі збільшенням кількості елементів у масиві ефективнішим виявляється **linq-сортування**, а **стандартне сортування** залишається найбільш ефективним, коли **сортування з вибором** виявляється дуже неефективним. Бульбашкове сортування є середнім, по швидкості, варіантом у всіх випадках.