

Національний технічний університет України «Київський  
політехнічний інститут імені Ігоря Сікорського»  
Факультет інформатики та обчислювальної техніки  
Кафедра інформаційних систем та технологій

## **Лабораторна робота №6**

з дисципліни «Основи програмування»

**Тема: Алгоритми сортування**

Виконали:  
Студенти групи ІА-23  
Мозоль Владислав,  
Курач Владислав,  
Лядський Дмитро  
Дата здачі: 11.11.2022

Перевірив:  
Колеснік В. М.

## Хід роботи:

1. Ознайомитись з алгоритмами сортування:

1.1. Сортування обміном (сортування бульбашкою, Bubble sort)

[https://uk.wikipedia.org/wiki/Сортування\\_бульбашкою](https://uk.wikipedia.org/wiki/Сортування_бульбашкою)

[https://en.wikipedia.org/wiki/Bubble\\_sort](https://en.wikipedia.org/wiki/Bubble_sort) 1.2.

Сортування вибором (Selection sort)

[https://uk.wikipedia.org/wiki/Сортування\\_вибором](https://uk.wikipedia.org/wiki/Сортування_вибором)

[https://en.wikipedia.org/wiki/Selection\\_sort](https://en.wikipedia.org/wiki/Selection_sort)

1.3. Сортування включенням (сортування вставкою, Insertion sort)

[https://uk.wikipedia.org/wiki/Сортування\\_включенням](https://uk.wikipedia.org/wiki/Сортування_включенням)

[https://en.wikipedia.org/wiki/Insertion\\_sort](https://en.wikipedia.org/wiki/Insertion_sort)

2. Реалізувати два методи сортування відповідно до свого варіанту з таблиці 1.

3. Відповісти на контрольні питання.

```
1  import java.util.Arrays;
2
3  public class task1 {
4      2 usages
5      static void selection_sort(double[] arr) {
6          int n = arr.length;
7          for (int i = 0; i < n-1; i++)
8          {
9              int min = i;
10             for (int j = i+1; j < n; j++)
11                 if (arr[j] < arr[min])
12                     min = j;
13             double result = arr[min];
14             arr[min] = arr[i];
15             arr[i] = result;
16         }
17     }
18     public static void main(String[] args) {
19
20         double[] arr = {64,8.05,0.05,0.009,25,12,22,11};
21         System.out.println("Non-sorted: " + Arrays.toString(arr));
22         selection_sort(arr);
23         System.out.println("Sorted: " + Arrays.toString(arr));
24         double[] another = {0.07,205,100,1000000000,0.000000008};
25         System.out.println("Non-sorted: " + Arrays.toString(another));
26         selection_sort(another);
27         System.out.println("Sorted: " + Arrays.toString(another));
28     }
29 }
```

```
F:\For programming\Java\bin\java.exe -javaagent:F:\For prog
Non-sorted: [64.0, 8.05, 0.05, 0.009, 25.0, 12.0, 22.0, 11.0]
Sorted: [0.009, 0.05, 8.05, 11.0, 12.0, 22.0, 25.0, 64.0]
Non-sorted: [0.07, 205.0, 100.0, 1.0E9, 8.0E-9]
Sorted: [8.0E-9, 0.07, 100.0, 205.0, 1.0E9]

Process finished with exit code 0
```

```

1  import java.util.Arrays;
2  public class task2{
    2 usages
3  @ static void insertion_sort(double[] arr)
4  {
5      int n = arr.length;
6      for (int k = 1; k < n; ++k) {
7          double key = arr[k];
8          int i = k - 1;
9          while (i >= 0 && arr[i] > key) {
10             arr[i + 1] = arr[i];
11             i = i - 1;
12         }
13         arr[i + 1] = key;
14     }
15 }
16
17 public static void main(String[] args) {
18     double[] arr = { 12, 11, 13, 5, 6, 5.01, 6.05};
19     System.out.println("Non-sorted: " + Arrays.toString(arr));
20     insertion_sort(arr);
21     System.out.println("Sorted: " + Arrays.toString(arr));
22     double[] another = { 10.6, 11, 1.89, 105.98, 0.85, 0.008};
23     System.out.println("Non-sorted: " + Arrays.toString(another));
24     insertion_sort(another);
25     System.out.println("Sorted: " + Arrays.toString(another));
26
27 }
28 }
29

```

```

F:\For programming\Java\bin\java.exe" "-javaagent:F:\Fo
Non-sorted: [12.0, 11.0, 13.0, 5.0, 6.0, 5.01, 6.05]
Sorted: [5.0, 5.01, 6.0, 6.05, 11.0, 12.0, 13.0]
Non-sorted: [10.6, 11.0, 1.89, 105.98, 0.85, 0.008]
Sorted: [0.008, 0.85, 1.89, 10.6, 11.0, 105.98]

Process finished with exit code 0

```

1. Порівняйте та назвіть переваги та недоліки таких методів сортування: сортування бульбашкою, сортування вибором, сортування вставкою.

Сортування бульбашкою: Хоча алгоритм є одним із найпростіших алгоритмів сортування, його ефективність є досить низькою, і він погано підходить для сортування великих списків. Більшість інших алгоритмів з такою ж швидкістю  $O(n^2)$  є ефективнішими за алгоритм сортування бульбашками, наприклад, сортування включенням.

Сортування вибором: Має ефективність  $n^2$ , що робить його неефективним при сортування великих масивів, і в цілому, менш ефективним за подібний алгоритм сортування включенням. Сортування вибором вирізняється більшою простотою, ніж сортування включенням, і в деяких випадках, вищою продуктивністю.

Сортування вставкою. Переваги: простота у реалізації, ефективний (зазвичай) на маленьких масивах, ефективний при сортуванні масивів, дані в яких вже непогано відсортовані: продуктивність рівна  $O(n + d)$ , де  $d$  — кількість інверсій, на практиці ефективніший за більшість інших квадратичних алгоритмів ( $O(n^2)$ ), як то сортування вибором та сортування бульбашкою: його швидкість рівна  $n^2/4$ , і в найкращому випадку є лінійною, є стабільним алгоритмом.

Недоліки: на великих масивах є значно менш ефективним за такі алгоритми, як швидке сортування, пірамідальне сортування та сортування злиттям.

2. Оцініть кожен з методів відповідно до наступних критеріїв:

- час роботи;

- потреби у додатковій пам'яті; - стабільність.

За якими ще критеріями ви можете порівняти ці алгоритми?

Сортування бульбашкою: сортування рядка за алгоритмом сортування бульбашками у п'ять разів повільніше за сортування того ж рядка за алгоритмом сортування включенням і на 40% повільніше за сортування вибором. Вимагає щонайменше удвічі більше операцій, ніж сортування включенням, удвічі більше кешу пам'яті. Стабільний метод сортування.

Сортування вибором: не стабільний, в деяких випадках продуктивніший ніж сортування вставкою, є потреба додатковій пам'яті

Сортування вставкою: стабільний алгоритм, швидкість рівна  $n^2/4$ , і в найкращому випадку є лінійною, є потреба в додатковій пам'яті, швидший ніж методи сортування бульбашкою та вибором .

3. Який випадок є найкращім або найгіршим для роботи цих алгоритмів? (Наприклад: частково відсортований масив або масив, відсортований у зворотному порядку). Який з методів забезпечить у цих випадках найкращій/найгірший результат?

Для сортування бульбашкою: найкращий випадок - частково відсортований масив, найгірший - масив, відсортований у зворотному порядку Для сортування вибором: не має різниці

Для сортування вставкою: найкращий випадок – частково відсортований масив , найгірший – масив, відсортований у зворотному порядку.

Якщо масив відсортований у зворотному порядку, то найгірший результат забезпечить сортування вставкою, найкращий – сортування бульбашкою

Якщо масив частково впорядкований, то найкращий результат забезпечить сортування вставкою, найгірший – сортування бульбашкою

4. Що означають позначення  $O(1)$ ,  $O(n)$ ,  $O(n^2)$ ?

$O(1)$  - Сталий час роботи не залежно від розміру задачі

$O(n)$  - Лінійне зростання — подвоєння розміру задачі подвоїть і необхідний час

$O(n^2)$  - Квадратичне зростання — подвоєння розміру задачі вчетверо збільшує необхідний час

Висновок: під час виконання цієї лабораторної роботи ми ознайомилися з алгоритмами сортування, зокрема з сортуванням обміном (бульбашкою), сортуванням вставкою, сортуванням вибором, а також дізналися як використовувати їх на практиці.