

## Лабораторна робота № 4

### АЛГОРИТМИ СОРТУВАННЯ

*Мета – розроблення програми для реалізації та аналізу алгоритмів сортування та порівняння їх часової складності.*

#### Теоретичні відомості

##### *Допоміжні конструкції:*

- для генерування випадкових чисел використовується вбудований клас **Random**:

- оголошення класу:

```
Random <назва змінної> = new Random( )
```

- генерування числа з інтервалу:

```
<назва змінної> . Next (<нижня межа>, <верхня межа> )
```

- для визначення часу виконання алгоритму сортування:

```
System.Diagnostics.Stopwatch sw = System.Diagnostics.Stopwatch.StartNew();
```

```
long elapsed;
```

```
// код алгоритму сортування
```

```
elapsed = sw.ElapsedTicks;
```

```
Console.WriteLine("Час виконання алгоритму: {0}", elapsed);
```

*Для скидання часу: **sw.Restart()**;*

# Алгоритми сортування (впорядкування) елементів одновимірного масиву

I. Представимо роботу алгоритму впорядкування за зростанням елементів масиву **методом обміну (метод "бульбашки")** наступним прикладом, що записаний на псевдокодi (узагальнена мова, що відображає порядок команд для реалізації алгоритму, проте не є достатнім кодом програми):

```
for (i=0; i<n; i++)  
    for (j=n-2; j>=i; j--)  
        if (arr[j] > arr[j+1])  
            swap(arr[j], arr[j+1]) // процедура обміну елементів  
                                    // (необхідно реалізувати самостійно!!!)
```

Таблиця 1.

**Приклад покрокової реалізації алгоритму сортування методом обміну**

Ітерація	Опис	Значення елементів масиву				
	Елементи масиву	arr[0]	arr[1]	arr[2]	arr[3]	arr[4]
	Початкові значення	10	5	6	1	3
1	Порівнюються arr[3] і arr[4]	10	5	6	1	3
	Порівнюються arr[2] і arr[3]	10	5	1	6	3
	Порівнюються arr[1] і arr[2]	10	1	5	6	3
	Порівнюються arr[0] і arr[1]	1	10	5	6	3
2	Порівнюються arr[3] і arr[4]	1	10	5	3	6
	Порівнюються arr[2] і arr[3]	1	10	3	5	6
	Порівнюються arr[1] і arr[2]	1	3	10	5	6
3	Порівнюються arr[3] і arr[4]	1	3	10	5	6
	Порівнюються arr[2] і arr[3]	1	3	5	10	6
4	Порівнюються arr[3] і arr[4]	1	3	5	6	10

II. Приклад алгоритму впорядкування за зростанням елементів масиву **методом вибору**, що записаний на псевдокодi

```
for (i = 0; i < n-1; i++)  
{  
    min = a[i];  
    index = i;  
    for (j = i+1; j < n; j++)  
    {  
        if (a[j] < min)  
        {  
            min = a[j];  
            index = j;  
        }  
    }  
}
```

```

    }
}
swap(a[i], a[index]);           // процедура обміну елементів
                                // (необхідно реалізувати самостійно!!!)
}

```

**III.** Приклад алгоритму впорядкування за зростанням елементів масиву **методом вставок**, що записаний на псевдокодi

```

for (i = 1; i < n; i++)
{
    key = a[i];
    j = i-1;
    while ((j >= 0) && (a[j]>key))
    {
        a[j+1] = a[j];
        j--;
    }
    a[j+1] = key;
}

```

**IV.** Приклад алгоритму впорядкування за зростанням елементів масиву **методом швидкого сортування**, що записаний на псевдокодi

```

void QuickSorting (int[] arr, int first, int last)
{
    p = arr[(last - first) / 2 + first];
    temp;
    i = first, j = last;
    while (i <= j)
    {
        while (arr[i] < p && i <= last) ++i;
        while (arr[j] > p && j >= first) --j;
        if (i <= j)
        {
            temp = arr[i]; arr[i] = arr[j]; arr[j] = temp;
            i++; j--;
        }
    }
    if (j > first) рекурсивний виклик (arr, first, j);
    if (i < last) рекурсивний виклик (arr, i, last);
}

```

### ЗАГАЛЬНА ПОСТАНОВКА ЗАВДАННЯ

Створити класи для реалізації методів сортування *МЕТОД\_1* та *МЕТОД\_2*, з можливістю порівняння їх швидкодії на числових вибірках випадкових чисел різної розмірності:

■ для вибірки розмірністю  $N$  елементів ( $N = 50 \dots 500$ ) – згенерувати масив чисел в межах від 100 до 1000; згенерований масив вивести на екран; після реалізації методів сортування обидві відсортовані вибірки також вивести на екран; вивести на екран швидкість виконання кожного методу сортування в мілісекундах;

■ для вибірки розмірністю  $M$  елементів ( $M = 1000 \dots 10000$ ) – згенерувати масив чисел в межах від 100 до 1000; вивести на екран швидкість виконання кожного методу сортування в мілісекундах.

Користувач вводить розмірності  $N$  та  $M$  з клавіатури.

\*\*\*\*\*

Розробити два класи: основний клас (з функцією *Main()*) для роботи з одновимірним масивом та клас з методами для реалізації сортування.

*В класі сортування розробити методи:*

- сортування елементів одновимірного масиву методом *МЕТОД\_1* (ЗГІДНО *ВАРІАНТУ*);
- сортування елементів одновимірного масиву методом *МЕТОД\_2* (ЗГІДНО *ВАРІАНТУ*);


*В основному класі розробити методи:*

- генерування одновимірного масиву цілих чисел з інтервалу від 100 до 1000;
- виведення елементів одновимірного масиву на екран.

*В методі Main():*

- 1) оголосити та створити 2 цілочисельних масиви (заданої розмірності);
- 2) за допомогою методу згенерувати значення масивів;
- 3) оголосити та створити об'єкт класу *сортування*;
- 4) викликати відповідні методи з класу сортування (масиви передаються як параметри);
- 5) для вибірки розмірності  $N$  відсортовані елементи та час виконання алгоритмів сортування (окремо для обох методів) вивести на екран;
- 6) для вибірки розмірності  $M$  вивести на екран тільки час виконання алгоритмів сортування (окремо для обох методів).

## QuickSort vs RadixSort



```
180 129 125 194 196 125 104 183 142 182
181 146 154 140 188 123 184 185 161 144
121 188 107 137 148 122 153 100 110 163
113 194 166 159 171 141 122 180 148 152
122 101 194 117 101 144 188 141 106 124

ШВИДКЕ СОРТУВАННЯ:
100 101 101 101 104 106 107 110 113 117
121 122 122 122 123 124 125 129 137 140
141 141 142 144 144 146 148 148 152 153
154 159 161 163 166 171 180 180 181 182
183 184 185 188 188 188 194 194 194 196
Час виконання алгоритму швидкого сортування в мілісекундах: 39736

ПОРІЗРЯДНЕ СОРТУВАННЯ:
100 101 101 101 104 106 107 110 113 117
121 122 122 122 123 124 125 129 137 140
141 141 142 144 144 146 148 148 152 153
154 159 161 163 166 171 180 180 181 182
183 184 185 188 188 188 194 194 194 196
Час виконання алгоритму порізрядного сортування в мілісекундах: 17393
```

ШВИДКЕ СОРТУВАННЯ:  
Час виконання алгоритму швидкого сортування в мілісекундах: 12713

ПОРОЗРЯДНЕ СОРТУВАННЯ:  
Час виконання алгоритму порозрядного сортування в мілісекундах: 155001

# ІНДИВІДУАЛЬНІ ЗАВДАННЯ

## Варіант 1

Сортування за зростанням значень елементів

- 1) Метод сортування вибором
- 2) Метод швидкого сортування

## Варіант 2

Сортування за спаданням значень елементів

- 1) Метод сортування бульбашкою
- 2) Метод порозрядного сортування

## Варіант 3

Сортування за зростанням значень елементів

- 1) Метод сортування підрахунком
- 2) Метод сортування злиттям

## Варіант 4

Сортування за спаданням значень елементів

- 1) Метод сортування вибором
- 2) Метод порозрядного сортування

## Варіант 5

Сортування за зростанням значень елементів

- 1) Метод порозрядного сортування
- 2) Метод сортування злиттям

## Варіант 6

Сортування за спаданням значень елементів

- 1) Метод сортування бульбашкою (обміном)
- 2) Метод швидкого сортування

## Варіант 7

Сортування за зростанням значень елементів

- 1) Метод сортування включенням
- 2) Метод порозрядного сортування

## Варіант 8

Сортування за спаданням значень елементів

- 1) Метод сортування вибором
- 2) Метод сортування злиттям

## Варіант 9

Сортування за зростанням значень елементів

- 1) Метод сортування злиттям
- 2) Метод порозрядного сортування

## Варіант 10

Сортування за спаданням значень елементів

- 1) Метод пірамідального сортування
- 2) Метод сортування Шелла

### **Варіант 11**

Сортування за зростанням значень елементів

- 1) Метод сортування вибором
- 2) Метод швидкого сортування

### **Варіант 12**

Сортування за зростанням значень елементів

- 1) Метод швидкого сортування
- 2) Метод сортування включенням

### **Варіант 13**

Сортування за спаданням значень елементів

- 1) Метод сортування підрахунком
- 2) Метод порозрядного сортування

### **Варіант 14**

Сортування за спаданням значень елементів

- 1) Метод сортування вибором
- 2) Метод порозрядного сортування

### **Варіант 15**

Сортування за зростанням значень елементів

- 1) Метод порозрядного сортування
- 2) Метод сортування злиттям

### **Варіант 16**

Сортування за спаданням значень елементів

- 1) Метод швидкого сортування
- 2) Метод порозрядного сортування

### **Варіант 17**

Сортування за зростанням значень елементів

- 1) Метод порозрядного сортування
- 2) Метод сортування включенням