

Міністерство освіти і науки України
Національний технічний університет України «Київський політехнічний
інститут імені Ігоря Сікорського»
Факультет інформатики та обчислювальної техніки

Кафедра ІІІ

Звіт

з лабораторної роботи № 1 з дисципліни
«Алгоритми та структури даних 2. Структури даних»

„Проектування і аналіз алгоритмів внутрішнього сортування”

Виконав(ла)

ІІІ-13 Дем'янчук Олександр Петрович
(шифр, прізвище, ім'я, по батькові)

Перевірив

Сопов Олексій Олександрович
(прізвище, ім'я, по батькові)

Київ 2022

ЗМІСТ

1	МЕТА ЛАБОРАТОРНОЇ РОБОТИ	3
2	ЗАВДАННЯ	4
3	ВИКОНАННЯ.....	5
3.1	АНАЛІЗ АЛГОРИТМУ НА ВІДПОВІДНІСТЬ ВЛАСТИВОСТЯМ	5
3.2	ПСЕВДОКОД АЛГОРИТМУ	5
3.3	АНАЛІЗ ЧАСОВОЇ СКЛАДНОСТІ.....	5
3.4	ПРОГРАМНА РЕАЛІЗАЦІЯ АЛГОРИТМУ	6
3.4.1	<i>Вихідний код</i>	<i>6</i>
3.4.2	<i>Приклад роботи</i>	<i>7</i>
3.5	ТЕСТУВАННЯ АЛГОРИТМУ	11
3.5.1	<i>Часові характеристики оцінювання.....</i>	<i>11</i>
3.5.2	<i>Графіки залежності часових характеристик оцінювання від розмірності масиву</i>	<i>15</i>
	ВИСНОВОК	17
	КРИТЕРІЇ ОЦІНЮВАННЯ	18

1 МЕТА ЛАБОРАТОРНОЇ РОБОТИ

Мета роботи – вивчити основні методи аналізу обчислювальної складності алгоритмів внутрішнього сортування і оцінити поріг їх ефективності.

2 ЗАВДАННЯ

Виконати аналіз алгоритму внутрішнього сортування на відповідність наступним властивостям (таблиця 2.1):

- стійкість;
- «природність» поведінки (Adaptability);
- базуються на порівняннях;
- необхідність додаткової пам'яті (об'єму);
- необхідність в знаннях про структуру даних.

Записати алгоритм внутрішнього сортування за допомогою псевдокоду (чи іншого способу по вибору).

Провести аналіз часової складності в гіршому, кращому і середньому випадках та записати часову складність в асимптотичних оцінках.

Виконати програмну реалізацію алгоритму на будь-якій мові програмування з фіксацією часових характеристик оцінювання (кількість порівнянь, кількість перестановок, глибина рекурсивного поглиблення та інше в залежності від алгоритму).

Провести ряд випробувань алгоритму на масивах різної розмірності (10, 100, 1000, 5000, 10000, 20000, 50000 елементів) і різних наборів вхідних даних (впорядкований масив, зворотно упорядкований масив, масив випадкових чисел) і побудувати графіки залежності часових характеристик оцінювання від розмірності масиву, нанести на графік асимптотичну оцінку гіршого і кращого випадків для порівняння.

Зробити порівняльний аналіз двох алгоритмів.

Зробити узагальнений висновок з лабораторної роботи.

Таблиця 2.1 – Варіанти алгоритмів

№	Алгоритм сортування
1	Сортування бульбашкою
2	Сортування гребінцем («розчіскою»)

3 ВИКОНАННЯ

3.1 Аналіз алгоритму на відповідність властивостям

Аналіз алгоритму **сортування бульбашкою** на відповідність властивостям наведено в таблиці 3.1.

Таблиця 3.1 – Аналіз алгоритму на відповідність властивостям

Властивість	Сортування бульбашкою	Сортування гребінцем
Стійкість	+	+
«Природність» поведінки (Adaptability)	-	-
Базуються на порівняннях	+	+
Необхідність в додатковій пам'яті (об'єм)	-	-
Необхідність в знаннях про структури даних	+	+

3.2 Псевдокод алгоритму

Сортування бульбашкою:

```
Swapped := false
Повторити для i від 0 до size - 1 з кроком 1
    Повторити для j від 0 до size - i - 1 з кроком 1
        Якщо arr[j] > arr[j + 1] то
            Swapped := true
            temp := arr[j];
            arr[j] := arr[j + 1];
            arr[j + 1] := temp;

        Інакше якщо Swapped := false то
            Вихід з циклу
    Все якщо
    Все повторити
Кінець сортування бульбашкою
```

Сортування гребінцем:

```
step := size - 1;
factor := 1.3;
Поки step >= 1 то
    Повторити для i від 0 до i + step < size з кроком 1
        Якщо arr[i] > arr[i+step] то
            temp := arr[i];
            arr[i] := arr[i + step];
            arr[i + step] := temp;
        Все якщо
    Все повторити
    step /= factor;
Все поки
```

Кінець сортування гребінцем

3.3 Аналіз часової складності

Сортування бульбашкою

Сортування бульбашкою засноване на двох циклах, перший має $(n-1)$ ітерацій, вкладений у нього – $(n-1-i)$, тому сумарно їх $(n-1)^2 - (n-1)i$, тому часова складність оцінюється в $O(n^2)$.

В найкращому випадку складність буде $O(n)$, в середньому випадку - $O(n^2)$, в найгіршому - $O(n^2)$.

Сортування гребінцем

Сортування гребінцем засноване на двох циклах: зовнішній – перевіряє чи крок є не меншим за одиницю, та внутрішній, кількість ітерацій в якому залежить від даного кроку. Тому складність алгоритму можна оцінити як $O(n^2)$. В найкращому випадку складність буде $O(n \log n)$, в середньому та найгіршому - $O(n^2)$.

3.4 Програмна реалізація алгоритму

3.4.1 Вихідний код

```
void bubble(int* arr, int size, int &comp, int &swap) {
    for (int i = 0; i < size - 1; i++) {
        for (int j = 0; j < size - i - 1; j++) {
            if (arr[j] > arr[j + 1]) {
                int temp = arr[j];
                arr[j] = arr[j + 1];
                arr[j + 1] = temp;
                swap++;
            }
        }
    }
}
```

```

        }
        comp++;
    }
}

void comb(int* arr, int size, int& comp, int& swap) {
    int step = size - 1;
    double factor = 1.3;
    while (step >= 1) {
        for (int i = 0; i + step < size; i++) {
            if(arr[i] > arr[i+step]) {
                int temp = arr[i];
                arr[i] = arr[i + step];
                arr[i + step] = temp;
                swap++;
            }
            comp++;
        }
        step /= factor;
    }
}

```

3.4.2 Приклад роботи

На рисунках 3.1 і 3.2 показані приклади роботи програми сортування масивів на 100 і 1000 елементів відповідно.

Рисунок 3.1 – Сортування масиву на 100 елементів

Сортування бульбашкою:

```
Консоль отладки Microsoft Visual Studio

enter size: 100
choose sorting method (bubble - b, comb - c): b
PERFECT ARRAY IS:
0 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25 26 27 28 29 30 31 32 33 34 35 36 37 38 39 40 41 42 43 44 45 46 47 48 49 50 51 52 53 54 55 56 57 58 59 60 61 62 63 64 65 66 67 68 69 70 71 72 73 74 75 76 77 78 79 80 81 82 83 84 85 86 87 88 89 90 91 92 93 94 95 96 97 98 99
SORTED ARRAY IS:
0 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25 26 27 28 29 30 31 32 33 34 35 36 37 38 39 40 41 42 43 44 45 46 47 48 49 50 51 52 53 54 55 56 57 58 59 60 61 62 63 64 65 66 67 68 69 70 71 72 73 74 75 76 77 78 79 80 81 82 83 84 85 86 87 88 89 90 91 92 93 94 95 96 97 98 99
comparisons: 4950
swaps: 0

RANDOM ARRAY IS:
65 22 21 0 65 6 1 24 63 46 35 53 71 83 8 52 66 68 79 3 99 92 7 85 43 14 62 53 32 91 75 24 29 23 31 89 73 55 31 95 9 26 7 3 6 31 39 98 98 20 7 75 61 44 28 62 14 37 92 19 65 37 86 58 84 88 49 59 87 97 79 33 48 67 37 83 78 88 40 77 43 81 84 49 32 93 49 77 32 78 7 64 59 75 62 51 28 25 42 50 85
SORTED ARRAY IS:
0 1 2 3 4 5 6 7 7 8 9 14 14 19 20 21 22 23 24 24 25 26 28 28 29 31 31 31 32 32 32 33 35 37 37 37 39 40 42 43 43 44 46 48 4 9 49 49 50 51 52 53 53 55 58 59 59 61 62 62 62 63 64 65 65 65 66 67 68 71 73 73 75 75 75 77 77 78 78 79 79 81 83 83 84 8 4 85 85 86 87 88 88 89 91 92 92 93 95 97 98 98 99
comparisons: 4950
swaps: 2133

WORST ARRAY IS:
100 99 98 97 96 95 94 93 92 91 90 89 88 87 86 85 84 83 82 81 80 79 78 77 76 75 74 73 72 71 70 69 68 67 66 65 64 63 62 61 60 59 58 57 56 55 54 53 52 51 50 49 48 47 46 45 44 43 42 41 40 39 38 37 36 35 34 33 32 31 30 29 28 27 26 25 24 23 22 21 20 19 18 17 16 15 14 13 12 11 10 9 8 7 6 5 4 3 2 1
SORTED ARRAY IS:
1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25 26 27 28 29 30 31 32 33 34 35 36 37 38 39 40 41 42 43 44 45 46 47 48 49 50 51 52 53 54 55 56 57 58 59 60 61 62 63 64 65 66 67 68 69 70 71 72 73 74 75 76 77 78 79 80 81 82 83 84 85 86 87 88 89 90 91 92 93 94 95 96 97 98 99 100
comparisons: 4950
swaps: 4950

C:\Users\demya\source\repos\lab1asd\х64\Debug\lab1asd.exe (процесс 20368) завершил работу с кодом 0.
Нажмите любую клавишу, чтобы закрыть это окно:
```

Сортування гребінцем:

```
Консоль отладки Microsoft Visual Studio

enter size: 100
choose sorting method (bubble - b, comb - c): c
PERFECT ARRAY IS:
0 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25 26 27 28 29 30 31 32 33 34 35 36 37 38 39 40 41 42 43 44 45 46 47 48 49 50 51 52 53 54 55 56 57 58 59 60 61 62 63 64 65 66 67 68 69 70 71 72 73 74 75 76 77 78 79 80 81 82 83 84 85 86 87 88 89 90 91 92 93 94 95 96 97 98 99
SORTED ARRAY IS:
0 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25 26 27 28 29 30 31 32 33 34 35 36 37 38 39 40 41 42 43 44 45 46 47 48 49 50 51 52 53 54 55 56 57 58 59 60 61 62 63 64 65 66 67 68 69 70 71 72 73 74 75 76 77 78 79 80 81 82 83 84 85 86 87 88 89 90 91 92 93 94 95 96 97 98 99
comparisons: 1004
swaps: 0

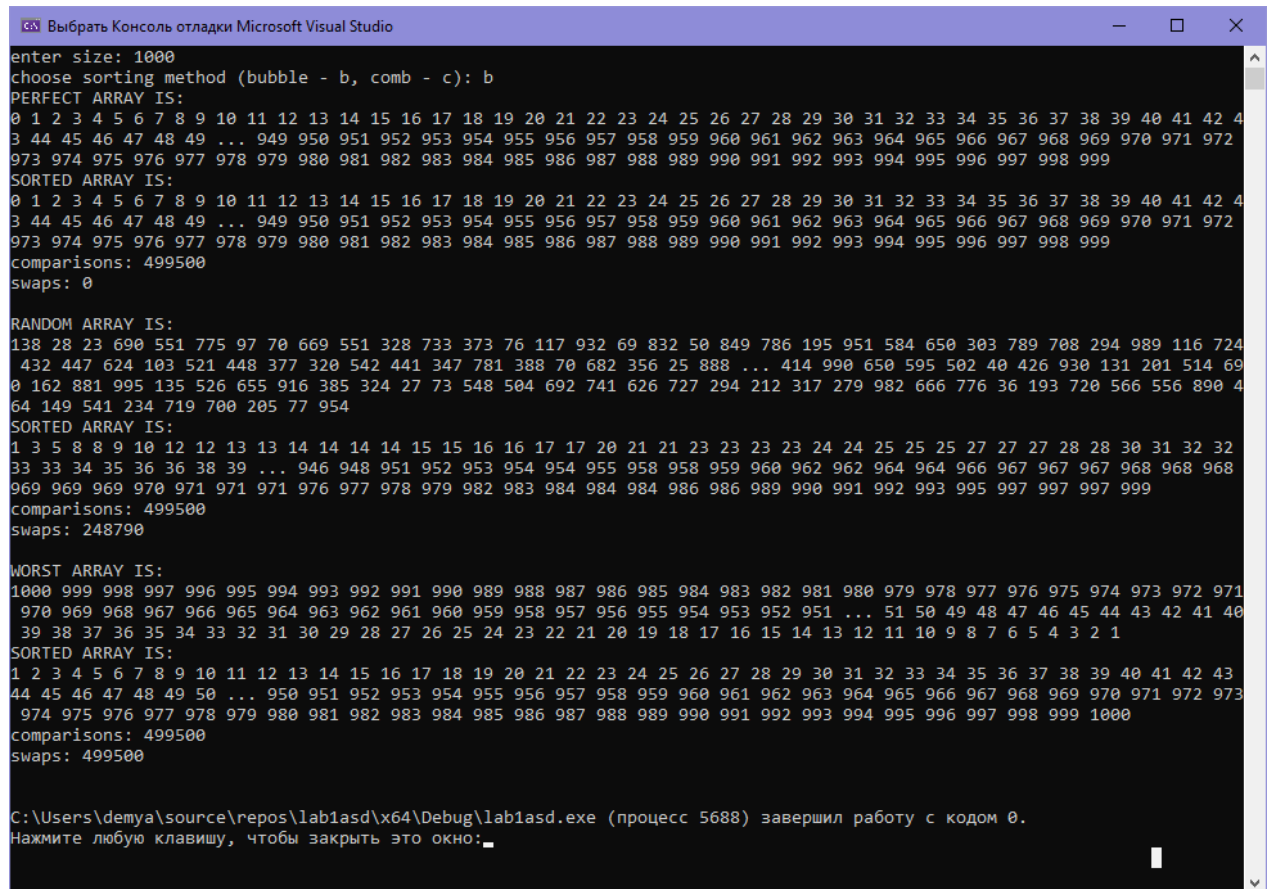
RANDOM ARRAY IS:
34 76 57 10 64 75 60 22 97 43 22 30 96 42 86 76 75 66 42 82 31 34 15 34 80 4 64 30 25 75 1 12 72 29 82 3 97 36 40 42 99 76 4 80 24 4 56 1 30 18 89 97 98 64 21 7 20 1 65 91 3 26 14 71 26 81 61 38 11 36 23 82 7 54 26 15 10 55 2 24 64 44 74 66 88 60 91 51 57 93 65 57 63 27 3 59 79 47 78 31
SORTED ARRAY IS:
1 1 1 2 3 3 3 4 4 4 7 7 10 10 11 12 14 15 18 15 20 21 22 22 23 24 24 25 26 26 26 27 29 30 30 30 31 31 34 34 34 36 36 38 40 42 42 42 43 44 47 51 54 55 56 57 57 57 59 60 60 61 63 64 64 64 64 65 65 66 66 71 72 74 75 75 75 76 76 76 78 79 80 80 81 82 82 82 86 88 89 91 93 91 96 97 97 97 98 99
comparisons: 1004
swaps: 223

WORST ARRAY IS:
100 99 98 97 96 95 94 93 92 91 90 89 88 87 86 85 84 83 82 81 80 79 78 77 76 75 74 73 72 71 70 69 68 67 66 65 64 63 62 61 60 59 58 57 56 55 54 53 52 51 50 49 48 47 46 45 44 43 42 41 40 39 38 37 36 35 34 33 32 31 30 29 28 27 26 25 24 23 22 21 20 19 18 17 16 15 14 13 12 11 10 9 8 7 6 5 4 3 2 1
SORTED ARRAY IS:
1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25 26 27 28 29 30 31 32 33 34 35 36 37 38 39 40 41 42 43 44 45 46 47 48 49 50 51 52 53 54 55 56 57 58 59 60 61 62 63 64 65 66 67 68 69 70 71 72 73 74 75 76 77 78 79 80 81 82 83 84 85 86 87 88 89 90 91 92 93 94 95 96 97 98 99 100
comparisons: 1004
swaps: 114

C:\Users\demya\source\repos\lab1asd\х64\Debug\lab1asd.exe (процесс 14320) завершил работу с кодом 0.
Нажмите любую клавишу, чтобы закрыть это окно: _
```


Рисунок 3.2 – Сортивання масиву на 1000 елементів

Сортивання бульбашкою:



```
Выбрать Консоль отладки Microsoft Visual Studio
enter size: 1000
choose sorting method (bubble - b, comb - c): b
PERFECT ARRAY IS:
0 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25 26 27 28 29 30 31 32 33 34 35 36 37 38 39 40 41 42 43 44 45 46 47 48 49 ... 949 950 951 952 953 954 955 956 957 958 959 960 961 962 963 964 965 966 967 968 969 970 971 972 973 974 975 976 977 978 979 980 981 982 983 984 985 986 987 988 989 990 991 992 993 994 995 996 997 998 999
SORTED ARRAY IS:
0 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25 26 27 28 29 30 31 32 33 34 35 36 37 38 39 40 41 42 43 44 45 46 47 48 49 ... 949 950 951 952 953 954 955 956 957 958 959 960 961 962 963 964 965 966 967 968 969 970 971 972 973 974 975 976 977 978 979 980 981 982 983 984 985 986 987 988 989 990 991 992 993 994 995 996 997 998 999
comparisons: 499500
swaps: 0

RANDOM ARRAY IS:
138 28 23 690 551 775 97 70 669 551 328 733 373 76 117 932 69 832 50 849 786 195 951 584 650 303 789 708 294 989 116 724 432 447 624 103 521 448 377 320 542 441 347 781 388 70 682 356 25 888 ... 414 990 650 595 502 40 426 930 131 201 514 69 0 162 881 995 135 526 655 916 385 324 27 73 548 504 692 741 626 727 294 212 317 279 982 666 776 36 193 720 566 556 890 4 64 149 541 234 719 700 205 77 954
SORTED ARRAY IS:
1 3 5 8 8 9 10 12 12 13 13 14 14 14 14 15 15 16 16 17 17 20 21 21 23 23 23 23 24 24 25 25 25 27 27 27 28 28 30 31 32 32 33 33 34 35 36 36 38 39 ... 946 948 951 952 953 954 954 955 958 958 959 960 962 962 964 964 966 967 967 967 968 968 968 969 969 969 970 971 971 971 976 977 978 979 982 983 984 984 984 986 986 989 990 991 992 993 995 997 997 997 999
comparisons: 499500
swaps: 248790

WORST ARRAY IS:
1000 999 998 997 996 995 994 993 992 991 990 989 988 987 986 985 984 983 982 981 980 979 978 977 976 975 974 973 972 971 970 969 968 967 966 965 964 963 962 961 960 959 958 957 956 955 954 953 952 951 ... 51 50 49 48 47 46 45 44 43 42 41 40 39 38 37 36 35 34 33 32 31 30 29 28 27 26 25 24 23 22 21 20 19 18 17 16 15 14 13 12 11 10 9 8 7 6 5 4 3 2 1
SORTED ARRAY IS:
1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25 26 27 28 29 30 31 32 33 34 35 36 37 38 39 40 41 42 43 44 45 46 47 48 49 50 ... 950 951 952 953 954 955 956 957 958 959 960 961 962 963 964 965 966 967 968 969 970 971 972 973 974 975 976 977 978 979 980 981 982 983 984 985 986 987 988 989 990 991 992 993 994 995 996 997 998 999 1000
comparisons: 499500
swaps: 499500

C:\Users\demya\source\repos\lab1asd\x64\Debug\lab1asd.exe (процесс 5688) завершил работу с кодом 0.
Нажмите любую клавишу, чтобы закрыть это окно: _
```

Сортивання гребінцем:

```
Выбрать Консоль отладки Microsoft Visual Studio
enter size: 1000
choose sorting method (bubble - b, comb - c): b
PERFECT ARRAY IS:
0 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25 26 27 28 29 30 31 32 33 34 35 36 37 38 39 40 41 42 43
3 44 45 46 47 48 49 ... 949 950 951 952 953 954 955 956 957 958 959 960 961 962 963 964 965 966 967 968 969 970 971 972
973 974 975 976 977 978 979 980 981 982 983 984 985 986 987 988 989 990 991 992 993 994 995 996 997 998 999
SORTED ARRAY IS:
0 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25 26 27 28 29 30 31 32 33 34 35 36 37 38 39 40 41 42 43
3 44 45 46 47 48 49 ... 949 950 951 952 953 954 955 956 957 958 959 960 961 962 963 964 965 966 967 968 969 970 971 972
973 974 975 976 977 978 979 980 981 982 983 984 985 986 987 988 989 990 991 992 993 994 995 996 997 998 999
comparisons: 499500
swaps: 0

RANDOM ARRAY IS:
138 28 23 690 551 775 97 70 669 551 328 733 373 76 117 932 69 832 50 849 786 195 951 584 650 303 789 708 294 989 116 724
432 447 624 103 521 448 377 320 542 441 347 781 388 70 682 356 25 888 ... 414 990 650 595 502 40 426 930 131 201 514 69
0 162 881 995 135 526 655 916 385 324 27 73 548 504 692 741 626 727 294 212 317 279 982 666 776 36 193 720 566 556 890 4
64 149 541 234 719 700 205 77 954
SORTED ARRAY IS:
1 3 5 8 9 10 12 12 13 13 14 14 14 15 15 16 16 17 17 20 21 21 23 23 23 23 24 24 25 25 25 27 27 27 28 28 30 31 32 32
33 33 34 35 36 36 38 39 ... 946 948 951 952 953 954 954 955 958 958 959 960 962 962 964 964 966 967 967 967 968 968 968
969 969 969 970 971 971 971 976 977 978 979 982 983 984 984 984 986 986 989 990 991 992 993 995 997 997 997 999
comparisons: 499500
swaps: 248790

WORST ARRAY IS:
1000 999 998 997 996 995 994 993 992 991 990 989 988 987 986 985 984 983 982 981 980 979 978 977 976 975 974 973 972 971
970 969 968 967 966 965 964 963 962 961 960 959 958 957 956 955 954 953 952 951 ... 51 50 49 48 47 46 45 44 43 42 41 40
39 38 37 36 35 34 33 32 31 30 29 28 27 26 25 24 23 22 21 20 19 18 17 16 15 14 13 12 11 10 9 8 7 6 5 4 3 2 1
SORTED ARRAY IS:
1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25 26 27 28 29 30 31 32 33 34 35 36 37 38 39 40 41 42 43
44 45 46 47 48 49 50 ... 950 951 952 953 954 955 956 957 958 959 960 961 962 963 964 965 966 967 968 969 970 971 972 973
974 975 976 977 978 979 980 981 982 983 984 985 986 987 988 989 990 991 992 993 994 995 996 997 998 999 1000
comparisons: 499500
swaps: 499500

C:\Users\demya\source\repos\lab1asd\x64\Debug\lab1asd.exe (процесс 5688) завершил работу с кодом 0.
Нажмите любую клавишу, чтобы закрыть это окно: _
```

3.5 Тестування алгоритму

3.5.1 Часові характеристики оцінювання

В таблиці 3.2 наведені характеристики оцінювання числа порівнянь і числа перестановок алгоритму сортування бульбашки для масивів різної розмірності, коли масив містить упорядковану послідовність елементів.

Таблиця 3.2 – Характеристики оцінювання алгоритму сортування бульбашки для упорядкованої послідовності елементів у масиві

Розмірність масиву	Число порівнянь	Число перестановок
10	45	0
100	4950	0
1000	499500	0
5000	12497500	0
10000	49995000	0
20000	199990000	0
50000	1249975000	0

В таблиці 3.3 наведені характеристики оцінювання числа порівнянь і числа перестановок алгоритму сортування бульбашки для масивів різної розмірності, коли масиви містять зворотно упорядковану послідовність елементів.

Таблиця 3.3 – Характеристики оцінювання алгоритму сортування бульбашки для зворотно упорядкованої послідовності елементів у масиві.

Розмірність масиву	Число порівнянь	Число перестановок
10	45	45
100	4950	4950
1000	499500	499500
5000	12497500	12497500
10000	49995000	49995000
20000	199990000	199990000
50000	1249975000	1249975000

У таблиці 3.4 наведені характеристики оцінювання числа порівнянь і числа перестановок алгоритму сортування бульбашки для масивів різної розмірності, масиви містять випадкову послідовність елементів.

Таблиця 3.4 – Характеристика оцінювання алгоритму сортування бульбашки для випадкової послідовності елементів у масиві.

Розмірність масиву	Число порівнянь	Число перестановок
10	45	21
100	4950	2410
1000	499500	246083
5000	12497500	6140949
10000	49995000	25040945
20000	199990000	100275138
50000	1249975000	624246703

СОРТУВАННЯ ГРЕБІНЦЕМ

Таблиця 3.2 – Характеристики оцінювання алгоритму сортування гребінцем для упорядкованої послідовності елементів у масиві

Розмірність масиву	Число порівнянь	Число перестановок
10	35	0
100	1004	0
1000	499500	0
5000	12497500	0
10000	49995000	0
20000	199990000	0
50000	1249975000	0

В таблиці 3.3 наведені характеристики оцінювання числа порівнянь і числа перестановок алгоритму сортування гребінцем для масивів різної розмірності, коли масиви містять зворотно упорядковану послідовність елементів.

Таблиця 3.3 – Характеристики оцінювання алгоритму сортування гребінця для зворотно упорядкованої послідовності елементів у масиві.

Розмірність масиву	Число порівнянь	Число перестановок
10	35	7
100	1004	114
1000	18727	1584
5000	123395	9574
10000	276741	20084
20000	613404	42432
50000	1683419	117078

У таблиці 3.4 наведені характеристики оцінювання числа порівнянь і числа перестановок алгоритму сортування гребінцем для масивів різної розмірності, масиви містять випадкову послідовність елементів.

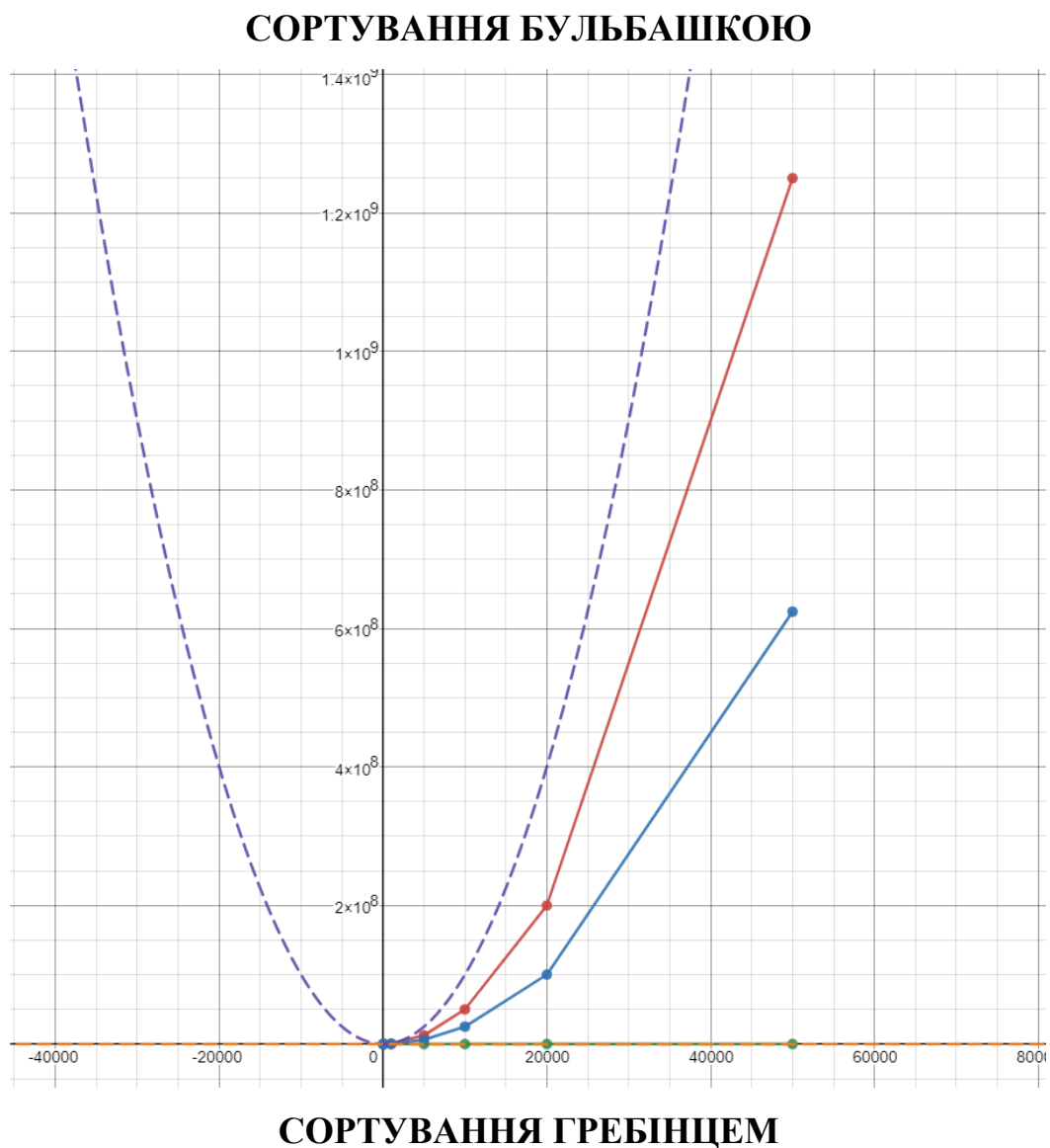
Таблиця 3.4 – Характеристика оцінювання алгоритму сортування гребінцем для випадкової послідовності елементів у масиві.

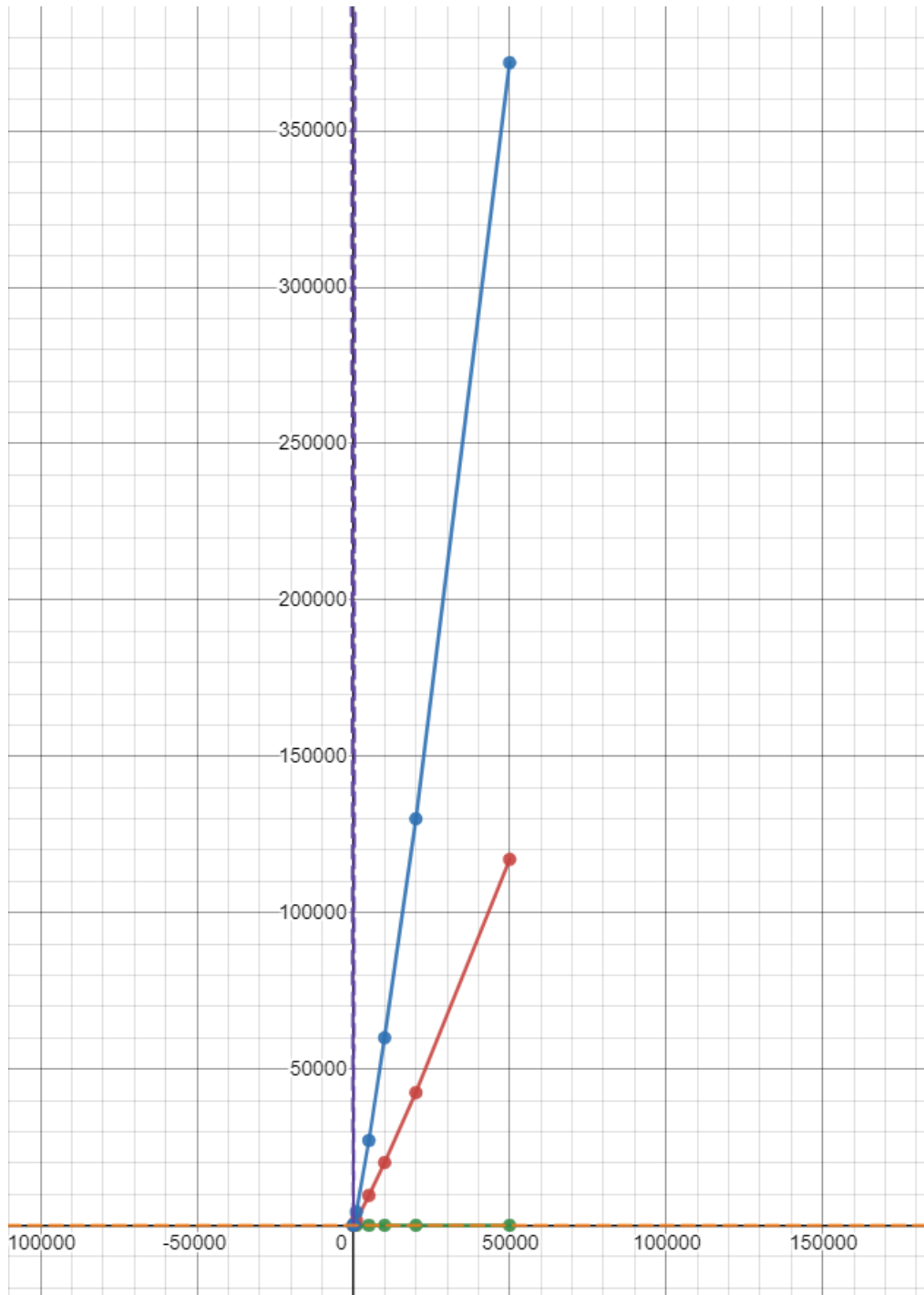
Розмірність масиву	Число порівнянь	Число перестановок
10	35	7
100	1004	218
1000	18727	4279
5000	123395	27174
10000	276741	60019
20000	613404	130043
50000	1683419	371825

3.5.2 Графіки залежності часових характеристик оцінювання від розмірності масиву

На рисунку 3.3 показані графіки залежності часових характеристик оцінювання від розмірності масиву для випадків, коли масиви містять упорядковану послідовність елементів (зелений графік), коли масиви містять зворотно упорядковану послідовність елементів (червоний графік), коли масиви містять випадкову послідовність елементів (синій графік), також показані асимптотичні оцінки гіршого (фіолетовий графік) і кращого (жовтий графік) випадків для порівняння.

Рисунок 3.3 – Графіки залежності часових характеристик оцінювання





ВИСНОВОК

При виконанні даної лабораторної роботи навчився визначати часову складність алгоритмів внутрішнього сортування та оцінювати поріг їх ефективності, застосував ці навички при аналізі сортування бульбашкою та гребінцем: порівняв характеристики алгоритмів сортування у найкращому, найгіршому та середньому випадках.

КРИТЕРІЇ ОЦІНЮВАННЯ

У випадку здачі лабораторної роботи до 21.02.2022 включно максимальний бал дорівнює – 5. Після 21.02.2022 – 28.02.2022 максимальний бал дорівнює – 2,5. Після 28.02.2022 робота не приймається

Критерії оцінювання у відсотках від максимального балу:

- аналіз алгоритму на відповідність властивостям – 10%;
- псевдокод алгоритму – 15%;
- аналіз часової складності – 25%;
- програмна реалізація алгоритму – 25%;
- тестування алгоритму – 20%;
- висновок – 5%.