Міністерство освіти і науки України

Національний технічний університет України

«Київський політехнічний інститут імені Ігоря Сікорського»

Інститут прикладного системного аналізу

Кафедра системного проектування

ЗВІТ

про виконання комп'ютерного практикуму № 3

з дисципліни «Алгоритми і структури даних»

Виконав: студент 1 курсу групи ДА-83

Цибін Максим Дмитрович

Варіант 27

Прийняв:

Київ – 2018

Зміст

ПРАКТИЧНА РОБОТА 3

1 Завдання

Реалізувати три види сортування та дослідити їх особливості шляхом порівняння теоретичних та експериментальних даних у таблиці.

Сортування:

1) **Обмін базовий(BubbleSort),**
2) **Пірамідальне(HeapSort),**
3) **Підрахунком(CountingSort)**

2 Лістинг програми

```c
#include <stdio.h>
#include <stdlib.h>
#include <time.h>
#include <string.h>

unsigned long long obmin, porivn = 0;

// наполняем масив случайными числами
void randarr(int *A, int size)
{
    srand(time(NULL));
    for(int i = 0; i < size; i++)
    {
        A[i] = rand();
    }
}

// Уже отсортированый
void allsort(int *B, int size)
{
    for(int i = 1; i < size+1; i++)
    {
        B[i-1] = i;
    }
}
```

```
// В обратном порядке
void revarr(int *C, int size)
{
    for(int i = size; i > 0; i--)
    {
        C[size-i] = i;
    }
}

//Первый метод сортировки BubbleSort
void sortmetI(int *A, int size)
    {
        int t;
        for(int i = 0; i < size;i++)
        {
            for(int k = 0; k < size-1; k++)
            {
                porivn++;
                if(A[k]>A[k+1])
                {
                    obmin++;
                    t = A[k];
                    A[k] = A[k+1];
                    A[k+1] = t;
                }
            }
        }
        for(int i = 0; i < size; i++)
        {
            printf("%d\n", A[i]);
        }
    }

//Второй метод сортировки HeapSort

void Heapify(int *A, int index,int size)
{
    int left = 2*(index+1) - 1;
    int right = 2*(index+1);
    int largest, t;
    porivn +=3;
    if(left < size && A[left] > A[index])
    {
        largest = left;
    }
    else largest = index;
```

```
    if(right < size && A[right] > A[largest])
    {
        largest = right;
    }
    if(largest != index)
    {   obmin++;
        t = A[index];
        A[index] = A[largest];
        A[largest] = t;
        Heapify(A, largest,size);

    }
}
void buildMaxHeap(int *A, int size)
{
    for(int i = size/2; i >=0;i--)
    {
        Heapify(A,i,size);
    }
}
void sortmetII(int *A, int size)
    {
        buildMaxHeap(A,size);
        int heapsize = size;
        int t;
        for(int i = size-1; i >0;i--)
        {
            obmin++;
            t = A[0];
            A[0] = A[i];
            A[i] = t;
            heapsize--;
            Heapify(A,0,heapsize);
        }



        for(int i = 0; i < size; i++)
        {
            printf("%d\n", A[i]);
        }
    }

//Третий метод сортировки CountingSort
void sortmetIII(int *A,int size, int k)
{
    int C[k];
```

```
    int B[size];
    for(int i = 0; i<=k; i++)
    {
        C[i] = 0;
    }
    for(int j = 0;j < size;j++)
    {
        C[A[j]] = C[A[j]] + 1;
    }
    for(int q = 1; q <=k; q++ )
    {
        C[q] = C[q-1]+C[q];
    }

    for(int h = 0; h < size; h++)
    {   obmin++;
        B[C[A[h]]-1] = A[h];
        C[A[h]] = C[A[h]] - 1;
    }
    for(int i = 0; i < size; i++)
        {
            printf("%d\n", B[i]);
        }

}


// Поехали
int main()
{
    int array,type,N;
    printf("Chose the type of sorting:\n 1 - Bubble sort\n 2 - Heapsort\n 3 - Counting
sort\n");
    scanf("%d",&type);
    printf("Chose array for testing:\n 1 - already sort\n 2 - array with random values\n 3
- reverse sort\n");
    scanf("%d",&array);
    printf("Enter the size of array:\n");
    scanf("%d",&N);
    int arr[N];
    switch(array)
    {
        case 1:
            allsort(arr,N);
            break;
        case 2:
            randarr(arr,N);
```

```
            break;
        case 3:
            revarr(arr,N);
            break;
        default :
            printf("Hello\n");
    }
for(int i = 0; i < N; i++)
        {
            printf("%d\n", arr[i]);
        }
    switch(type)
    {
        case 1:
            sortmetI(arr,N);
            break;
        case 2:
            sortmetII(arr,N);
            break;
        case 3:
            sortmetIII(arr,N,100000);
            break;
        default :
            printf("Hello\n");
    }
    printf("Porivnanya: %d\n", porivn);
    printf("Obmin: %d\n", obmin);

    return 0;
}
```

3 Результати експерименту у вигляді таблиці

1	BubbleSort					
	Кількість порівнянь			Кількість копіювань		
	Теор.	Есперим.	Відношення	Теор.	Есперим.	Відношення
	Найкращий випадок					
1000	1000000	999000	0,999	0	0	1
10000	100000000	99990000	0,9999	0	0	1
100000	10000000000	9999900000	0,99999	0	0	1
	Середній випадок					
1000	1000000	999000	0,999	499500	242878	0,48
10000	100000000	99990000	0,9999	49995000	25105349	0,5
100000	10000000000	9999900000	0,99999	4999950000	2456376533	0,49
	Найгірший випадок					
1000	1000000	999000	0,999	499500	499500	1
10000	100000000	99990000	0,9999	49995000	49995000	1
100000	10000000000	9999900000	0,99999	4999950000	4999950000	1

2	HeapSort					
	Кількість порівнянь			Кількість копіювань		
	Теор.	Есперим.	Відношення	Теор.	Есперим.	Відношення
	Найкращий випадок					
1000	9966	30627	3,07	9708	9708	1
10000	132880	410871	3,04	131956	131956	1
100000	1661000	5102565	3,07	1650854	1650854	1
	Середній випадок					
1000	9966	28797	2,8	9708	9098	0,93
10000	132880	387498	2,9	131956	124165	0,94
100000	16661000	4873755	2,9	1650854	1574584	0,95
	Найгірший випадок					
1000	9966	26451	2,65	8316	8316	1
10000	132880	365091	2,74	116696	116696	1
100000	16661000	4642305	2,78	1497434	1497434	1

3	CountingSort					
	Кількість порівнянь			Кількість копіювань		
	Теор.	Есперим.	Відношення	Теор.	Есперим.	Відношення
	Найкращий випадок					
1000	1000	1000	1	0	0	1
10000	10000	10000	1	0	0	1
100000	100000	100000	1	0	0	1
	Середній випадок					
1000	1000	1000	1	0	0	1
10000	10000	10000	1	0	0	1
100000	100000	100000	1	0	0	1
	Найгірший випадок					
1000	1000	1000	1	0	0	1
10000	10000	10000	1	0	0	1
100000	100000	100000	1	0	0	1

4 Висновок

В ході виконання практичної роботи мною був набут досвід розробки алгоритмів роз'язку задач. Було порівняно теоретичну оцінку для кількості операцій та експериментально пораховано кількість операцій для сортування масивів різними методами.