

ММИНОБРНАУКИ РОССИИ
САНКТ-ПЕТЕРБУРГСКИЙ ГОСУДАРСТВЕННЫЙ
ЭЛЕКТРОТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ
«ЛЭТИ» ИМ. В.И. УЛЬЯНОВА (ЛЕНИНА)
Кафедра МО ЭВМ

ОТЧЕТ
по лабораторной работе №4
по дисциплине «Алгоритмы и структуры данных»
Тема: Сортировки

Студент гр. 9384

Нистратов Д.Г.

Преподаватель

Ефремов М.А.

Санкт-Петербург

2020

Цель работы.

Разработать алгоритм пирамидальной сортировки массива.

Задание.

13. Пирамидальная сортировка.

Выполнение работы.

Была реализована функция `sortHeap`, производящая:

1. Преобразование по максимальному элементу.
2. Поочередно удаляет последний элемент, упорядочив значения.

Также была реализована вспомогательная рекурсивная функция `heapify`, производящая замену переменных в массиве.

Тестирование.

Результаты тестирования представлены в табл. 1.

Таблица 1 – Результаты тестирования

№ п/п	Входные данные	Выходные данные	Комментарии
1.	5 10 3 5 2 1	Sorted array: 1 2 3 5 10 timer : 9939 microseconds	Массив из 5 элементов прошел сортировку меньше чем за 10000 миллисекунд
2.	10 66 44 33 2 1 7 5 67 34 23	Sorted array: 1 2 5 7 23 33 34 44 66 67 timer : 12359 microseconds	Был сортирован массив из 10 элементов.
3.	5 1 2 3 4 5	Sorted array: 1 2 3 4 5 timer : 4109 microseconds	На прохождение уже отсортированного массива ушло 4109 миллисекунд.

Выводы.

В ходе выполнения лабораторной работы была изучена “Пирамидальная сортировка”, реализующая сортировку массива целых чисел, в худшем случае за $O(n \log n)$.

ПРИЛОЖЕНИЕ А

ИСХОДНЫЙ КОД ПРОГРАММЫ

Название файла: lb4_13.cpp

```
#include <iostream>
```

```
#include <chrono>
```

```
void heapify(int array[], size_t n, int i)
```

```
{
```

```
    int max = i;
```

```
    int left = 2 * i + 1;
```

```
    int right = 2 * i + 2;
```

```
    if ((left < n) && (array[left] > array[max]))
```

```
        max = left;
```

```
    if ((right < n) && (array[right] > array[max]))
```

```
        max = right;
```

```
    if (max != i)
```

```
    {
```

```
        std::swap(array[i], array[max]);
```

```
        heapify(array, n, max);
```

```
    }
```

```
}
```

```
//"Пирамидальная сортировка"
```

```
void sortHeap(int array[], size_t n)
```

```
{
```

```
    std::cout << "~~~~~" << std::endl;
```

```
    std::cout << "rearraging by max arg: " << std::endl;
```

```

    for (int i = n / 2 - 1; i >= 0; i--)
        heapify(array, n, i);

    std::cout << "array: ";
    for (int i = 0; i < n; i++)
        std::cout << array[i] << " ";
    std::cout << std::endl;
    std::cout << "~~~~~" << std::endl;

    for (int i = n - 1; i > 0; i--)
    {
        std::cout << "swap last with first arg: " << array[i]
            << " " << array[0] << std::endl;
        std::swap(array[0], array[i]);
        std::cout << "removing last arg: " << array[i] << std::endl;
        heapify(array, i, 0);
    }
    std::cout << "~~~~~" << std::endl << std::endl;
}

int main()
{
    size_t N;
    std::cin >> N;
    int array[N];
    for (int i = 0; i < N; i++)
        std::cin >> array[i];

    auto start = std::chrono::high_resolution_clock::now();
    sortHeap(array, N);

```

```

auto stop = std::chrono::high_resolution_clock::now();
auto duration = std::chrono::duration_cast<std::chrono::microseconds>(stop-
start);

std::cout << "Sorted array:" << std::endl;
for (int i = 0; i < N; i++)
    std::cout << array[i] << " ";
std::cout << std::endl;
std::cout << "timer : " << std::endl << duration.count() << " microseconds"
<< std::endl;
}

```