МИНОБРНАУКИ РОССИИ
САНКТ-ПЕТЕРБУРГСКИЙ ГОСУДАРСТВЕННЫЙ
ЭЛЕКТРОТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ
«ЛЭТИ» ИМ. В.И. УЛЬЯНОВА (ЛЕНИНА)
Кафедра МОЭВМ

ОТЧЕТ

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

Студент гр. 9381 _____ Фоминенко А.Н.

Преподаватель _____ Фирсов М.А.

Санкт-Петербург
2020

1. Цель работы.

Познакомиться с принципами сортировок в компьютерных науках

2. Задание.

Вариант 19

19. Сортировка расчёской.

3. Теоретические положения

Сортировка расчёской (англ. *comb sort*) — это довольно упрощённый алгоритм сортировки, изначально спроектированный Влодзимежом Добосевичем в 1980 г. Позднее он был переоткрыт и популяризирован в статье Стивена Лэйси и Ричарда Бокса в журнале Byte Magazine в апреле 1991 г. Сортировка расчёской улучшает сортировку пузырьком, и конкурирует с алгоритмами, подобными быстрой сортировке. Основная идея — устранить маленькие значения в конце списка, которые крайне замедляют сортировку пузырьком.

В сортировке пузырьком, когда сравниваются два элемента, промежуток (расстояние друг от друга) равен 1. Основная идея сортировки расчёской в том, что этот промежуток может быть гораздо больше, чем единица (сортировка Шелла также основана на этой идее, но она является модификацией сортировки вставками, а не сортировки пузырьком).

4. Описание работы алгоритма.

Основная идея «расчёски» в том, чтобы первоначально брать достаточно большое расстояние между сравниваемыми элементами и по мере упорядочивания массива сужать это расстояние вплоть до минимального. Таким образом, мы как бы причесываем массив. Первоначальный разрыв между сравниваемыми элементами лучше брать с учётом специальной величины, называемой фактором уменьшения, оптимальное значение которой равно примерно 1,247. Сначала расстояние между

элементами максимально, то есть равно размеру массива минус один. Затем, пройдя массив с этим шагом, необходимо поделить шаг на фактор уменьшения и пройти по списку вновь. Так продолжается до тех пор, пока разность индексов не достигнет единицы. В этом случае сравниваются соседние элементы как и в сортировке пузырьком, но такая итерация одна.

5. Пример работы программы:

Основной тест №1:

```
Console or file input? (0/1)

0

Print the size of array?

8

Print type of array (1 - int, 2 - double, 3 - char)

2

Print array

-1.9 5 -1000 8 203.5 1 2 2

Iteration 1, step = 7

Array :: -1.9 5 -1000 8 203.5 1 2 2

Iteration 2, step = 5

Array :: -1.9 5 -1000 8 203.5 1 2 2

Iteration 3, step = 4

Array :: -1.9 2 -1000 8 203.5 1 5 2

Iteration 4, step = 3

Array :: -1.9 1 -1000 2 203.5 2 5 8

Iteration 5, step = 2
```

```
Array :: -1.9 1 -1000 2 8 2 5 203.5

Iteration 6, step = 1

Array :: -1000 1 -1.9 2 5 2 8 203.5


Printing sorted array      :: -1000 -1.9 1 2 2 5 8 203.5

Printing std::sorted array :: -1000 -1.9 1 2 2 5 8 203.5

Arrays are the same
```

Дополнительное тестирование :

Номер теста	Входные данные	Результат
2	6 1 9 21 8 1 0 1	Printing sorted array :: 0 1 1 8 9 21 Printing std::sorted array:: 0 1 1 8 9 21
3	6 3 v a A n u q	Printing sorted array :: A a n q u v Printing std::sorted array :: A a n q u v
4	6 2 0 0 0 0 -0 1000000000	Printing sorted array :: 0 0 0 0 -0 1e+009 Printing std::sorted array :: 0 0 0 0 -0 1e+009
5	6 1 5 4 3 2 1 -5	Printing sorted array :: -5 1 2 3 4 5 Printing std::sorted array :: -5 1 2 3 4 5
6	1 1 -1000	Printing sorted array :: -1000 Printing std::sorted array :: -1000
7	3	Printing sorted array :: 0 0 0

	1 c a b	Printing std::sorted array :: 0 0 0

7 тест = некорректные данные

6. Выполнение программы:

1. Для ввода информации из файла необходимо ввести “1” на вопрос программы Console or file input? (0/1)
2. Для ввода информации из консоли необходимо ввести “0” на вопрос программы Console or file input? (0/1)
   после требуется ввести размер массива, тип данных и сам массив.

7. Описание функций:

```
/**
 * функция вывода вектора
 * @tparam T  тип данных
 * @param a - вектор для вывода
 */
template<typename T>
void print(vector<T> a)
================================================
**
 * функция считывания вектора
 * @tparam T тип данных
 * @param a вектор для считывания
 */
template<typename T>
```

```
void read(vector<T> &a)

==========================================================

/**

 * функция сортировки расческой

 * @tparam T тип данных

 * @param a вектор который нужно отсортировать

 */

template<typename T>

void my_sort(vector<T> &a)

==========================================================

/**

 * функция сравнения двух сортировок

 * @tparam T тип данных

 * @param a вектор для сортировки

 */

template<typename T>

void solve(vector<T> &a)
```

8. Вывод:

В ходе выполнения лабораторной работы была изучена и реализована на языке C++ сортировки расческой.

ПРИЛОЖЕНИЕ А

ИСХОДНЫЙ КОД ПРОГРАММЫ

Файл main.cpp:

```
#include <bits/stdc++.h>


#define all(v) v.begin(), v.end()


using namespace std;
/**
 * функция вывода вектора
 * @tparam T  тип данных
 * @param a - вектор для вывода
 */
template<typename T>
void print(vector<T> a) {
    for (auto &i : a)
        cout << i << " ";
    cout << '\n';
}
/**
 * функция считывания вектора
 * @tparam T тип данных
 * @param a вектор для считывания
 */
template<typename T>
void read(vector<T> &a) {
    for (auto &i : a)
```

```
        cin >> i;
}
/**
 * функция сортировки расческой
 * @tparam T тип данных
 * @param a вектор который нужно отсортировать
 */
template<typename T>
void my_sort(vector<T> &a) {
    double f = 1.2473309;
    int step = a.size() - 1;
    int iter = 1;
    while (step >= 1) {
        cout << "Iteration " << iter++ << ", step = " << step << "\nArray :: ";
        print(a);
        for (int i = 0; i + step < a.size(); i++)
            if (a[i] > a[i + step])
                swap(a[i], a[i + step]);
        step /= f;
    }
    cout << '\n';
}

/**
 * функция сравнения двух сортировок
 * @tparam T тип данных
 * @param a вектор для сортировки
 */
template<typename T>
void solve(vector<T> &a) {
    vector<T> b = a;
```

```
    my_sort(b);
    sort(all(a));
    cout << "Printing sorted array      :: ";
    print(b);
    cout << "Printing std::sorted array :: ";
    print(a);
    if (a == b) cout << "Arrays are the same\n";
    else cout << "Arrays are different\n";
}

int32_t main() {
    cout << "Console or file input? (0/1)\n";
    int type = 0;
    cin >> type;
    if (type == 1) {
        freopen("../input.txt", "r", stdin);
        freopen("../output.txt", "w", stdout);
    }
    cout << "Print the size of array?\n";
    int n;
    cin >> n;
    cout << "Print type of array (1 - int, 2 - double, 3 - char)\n";
    cin >> type;
    cout << "Print array\n";
    if (type == 1) {
        vector<int> a(n);
        read(a);
        solve(a);
    } else if (type == 2) {
        vector<double> a(n);
        read(a);
```

```
        solve(a);
    } else {
        vector<char> a(n);
        read(a);
        solve(a);
    }
    if (type == 1) {
        fclose(stdin);
        fclose(stdout);
    }
}
```