**МИНОБРНАУКИ РОССИИ**

**САНКТ-ПЕТЕРБУРГСКИЙ ГОСУДАРСТВЕННЫЙ ЭЛЕКТРОТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ «ЛЭТИ» ИМ. В.И. УЛЬЯНОВА (ЛЕНИНА)**

**Кафедра МО ЭВМ**

# ОТЧЕТ

**по лабораторной работе №1**

**по дисциплине «Программирование» Тема:**

**Обзор стандартной библиотеки**

Студент гр. 1384 _____ Усачева Д.В.

Преподаватель _____ Жангиров Т. Р.

Санкт-Петербург2022

**Цель работы.**

Целью работы является ознакомление с наиболее часто используемыми и полезными функциями стандартной библиотеки языка С, а также работа со временем.

**Задание.**

(Вариант 3)

Напишите программу, на вход которой подается массив целых чисел длины **1000**.

Программа должна совершать следующие действия:

- отсортировать массив с помощью алгоритма "сортировка пузырьком"
- посчитать время, за которое будет совершена сортировка, используя при этом **функцию стандартной библиотеки**
- отсортировать массив с помощью алгоритма "быстрая сортировка" (quick sort), используя при этом **функцию стандартной библиотеки**
- посчитать время, за которое будет совершена сортировка, используя при этом **функцию стандартной библиотеки**
- вывести отсортированный массив (элементы массива должны быть разделены пробелом)
- вывести время, за которое была совершена сортировка пузырьком
- вывести время, за которое была совершена быстрая сортировка

*Отсортированный массив, время сортировки пузырьком, время быстрой сортировки должны быть выведены с новой строки, при этом элементы массива должны быть разделены пробелами.*

**Выполнение работы.**

Основное тело программы, функция main, начинается с заполнения двух массивов bub_sorted и q_sorted с консоли. После чего измеряется прошедшее время с запуска программы и затем выполняется первая сортировка с одним массивом, после чего вновь засекается время и выполняется вторая сортировка для другого массива. После вновь засекается время. После выводят один из отсортированных массивов, а после них, путём вычитания известным нам значений время и делениям для перевода из тиков в секунды, показывается время сортировки bub_sort и qsort.

Массив bub_sorted сортируется при помощи «сортировки пузырьком». Сортировка пузырьком работает следующим образом: программа проходится по массиву, сравнивая два соседних элемента. При необходимости она меняет их местами. Дойдя до конца массива программа возвращается в начало массива и повторяет алгоритм, пока массив не будет полностью отсортирован.

Массив q_sorted сортируется быстрой сортировкой при помощи встроенной функции qsort.

Для измерения времени каждого из алгоритмов используется функция clock.

Разработанный программный код см. в приложении 2.

Результаты тестирования см. в приложении 1.

**Выводы.**

Были изучены наиболее часто используемые и полезные функциями стандартной библиотеки языка С, а также работа со временем.

# ПРИЛОЖЕНИЕ 1

# ТЕСТИРОВАНИЕ

Таблица 1 - Пример тестовых случаев(количество случаев уменьшено до 100)

| № п/п | Входные данные | Выходные данные |
|---|---|---|
| 1. | 47 -22 65 37 -100 24 85 18 -23 -14 55 -85 -39 31 62 98 -57 -73 78 -39 -10 4 65 42 49 -43 41 67 91 6 83 44 33 32 38 94 -43 45 1 66 88 -92 -79 47 -1 26 74 -21 76 88 29 -75 67 -40 84 -9 14 -71 -30 46 13 43 -2 -86 -1 15 25 -54 -76 -56 69 17 -19 68 77 -58 -88 32 97 -42 -19 60 49 43 -87 18 -45 9 -17 -17 -8 -57 -55 -30 55 75 -55 74 81 -94 1 | -100 -94 -92 -88 -87 -86 -85 -79 -76 -75 -73 -71 -58 -57 -57 -56 -55 -55 -54 -45 -43 -43 -42 -40 -39 -39 -30 -30 -23 -22 -21 -19 -19 -17 -17 -14 -10 -9 -8 -2 -1 -1 1 4 6 9 13 14 15 17 18 18 24 25 26 29 31 32 32 33 37 38 41 42 43 43 44 45 46 47 47 49 49 55 55 60 62 65 65 66 67 67 68 69 74 74 75 76 77 78 81 83 84 85 88 88 91 94 97 98<br>0.000060<br>0.000015 |

# ПРИЛОЖЕНИЕ 2
# ИСХОДНЫЙ КОД ПРОГРАММЫ

Название файла: main.c

```
#include <stdlib.h>
#include <stdio.h>
#include <stdbool.h>
#include <time.h>

void bub_sort(int *arr){
    bool flag = false;
    while(!flag){
        flag = true;
        for(int i=0; i < 1000-1; i++){
            if (arr[i]>arr[i+1]){
                flag = false;
                int a = arr[i];
                arr[i] = arr[i+1];
                arr[i+1] = a;
            }
        }
    }
}

int cmp(const void*a, const void*b){
    return *(int*)a - *(int*)b;
}

int main()
{
    int q_sorted[1000];
    int bub_sorted[1000];
    for(int i=0; i<1000; i++){
        int a;
        scanf("%d ", &a);
        q_sorted[i] = a;
        bub_sorted[i] = a;
    }

    clock_t q_start = clock();
    qsort(q_sorted, 1000, sizeof(int), cmp);
    clock_t q_end = clock();

    float q_time = (float)(q_end - q_start)/CLOCKS_PER_SEC;

    clock_t b_start = clock();
    bub_sort(bub_sorted);
    clock_t b_end = clock();

    float bub_time = (float)(b_end - b_start)/CLOCKS_PER_SEC;

    for(int i=0; i<1000; i++){
        printf("%d ", bub_sorted[i]);
    }
    printf("\n%f", bub_time);
```

```
    printf("\n%f ", q_time);

    return 0;
}
```