**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 языка Си**

Студент гр. 0382 _____ Куликов М.Д.

Преподаватель _____ Берленко Т.А.

Санкт-Петербург

2021

**Цель работы.**

Изучение стандартное библиотеки языка Си и получение опыта работы с ней.

**Задание.**

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.*

**Основные теоретические положения.**

В ходе работы была ранее неиспользованная библиотека time.h , а именно :

Функция clock() - возвращает количество тактов процессора, прошедших с момента запуска программы.

Тип данных time_t — Временной тип данных time_t способен представлять время и поддерживает арифметические операции.

Макрос CLOCKS_PER_SEC - Этот макрос заменяется на значение, представляющее число тиков в секунду, которое возвращает функция clock.

**Выполнение работы.**

В начале функции main создаются и заполняются два массива с 1000 числами, чтобы оба отсортировать разными способами.

Далее создается переменная t типа time_t для измерения скорости сортировок в тактах. В переменных time_1 и time_2 количество тактов переводится в секунды с помощью макроса CLOCKS_PER_SEC.

После использование сортировки пузырьком и функции быстрой сортировки фиксируется время их выполнения в тактах.

В конце выводятся необходимые данные в определенном порядке.

**Тестирование.**

Для удобства тестирования проведем его при массиве , состоящим из 10 чисел.

Результаты тестирования представлены в табл. 1.

Таблица 1 – Результаты тестирования

| № п/п | Входные данные | Выходные данные | Комментарии |
|---|---|---|---|
| 1. | 3 4 5 2 -10 39 0 -23 3 4 | -23 -10 0 2 3 3 4 4 5 39<br>0.133655<br>0.133656 | Корректное выполнение программы. |
| 2. | 0 0 0 0 0 0 0 0 0 0 | 0 0 0 0 0 0 0 0 0 0<br>0.127353<br>0.127354 | Корректное выполнение программы. |
| 3. | -123 -34 -982 -2345 -4534 -9828 0 10000 -100000 -500 | -100000 -9828 -4534 -2345 -982 -500 -123 -34 0 10000<br>0.144119<br>0.144121 | Корректное выполнение программы. |

**Выводы.**

В ходе выполнения работы был получен опыт работы со стандартными библиотеки языка Си и написана программа, сравнивающая скорость выполнения двух различных сортировок.

## ПРИЛОЖЕНИЕ А
## ИСХОДНЫЙ КОД ПРОГРАММЫ

Название файла: main.c

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

int comparator(const void *int1, const void *int2) {
    int *arr1 = (int *) int1;
    int *arr2 = (int *) int2;

    return (*arr1 - *arr2);
}

int main() {
    int num_amount = 10;
    int buf;
    int arr[num_amount];
    int arr2[num_amount];
    for (int i = 0; i < num_amount; i++) {
        scanf("%d", &arr[i]);
        arr2[i] = arr[i];
    }

    clock_t t;
    t = clock();
    for (int i = 0; i < num_amount; i++) {
```

```
        for (int j = num_amount - 1; j > i; j--)
            if (arr[j - 1] > arr[j]) {
                buf = arr[j - 1];
                arr[j - 1] = arr[j];
                arr[j] = buf;
            }
    }

    float time1 = ((float) t) / CLOCKS_PER_SEC;
    t = clock();
    qsort(arr2, num_amount, sizeof(int), comparator);
    float time2 = ((float) t) / CLOCKS_PER_SEC;

    for (int i = 0; i < num_amount; i++)
        printf("%d ", arr[i]);
    printf("\n%f\n", time1);

    printf("%f\n", time2);



    return 0;
}
```