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

ОТЧЕТ

по лабораторной работе №4
по дисциплине «Программирование»
Тема: Структуры и обзор stdlib

Студент гр. 0382 _____ Шангичев В. А.

Преподаватель _____ Жангиров Т. Р.

Санкт-Петербург
2021

Цель работы.

Изучить и применить на практике функции стандартных заголовочных файлов.

Задание.

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

Программа должна совершать следующие действия:

- отсортировать массив по невозрастанию модулей элементов 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 быстрая сортировка

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

Основные теоретические положения.

- Стандартный заголовочный файл `stdio.h` – в этом файле описаны функции для работы с вводом/выводом.
- Стандартный заголовочный файл `stdlib.h` - данный заголовочный файл содержит описание функций для работы с динамической памятью, генерации псевдослучайных чисел, для вычисления абсолютного значения и деления целых чисел, функции для сортировки и поиска и другие.
- Стандартный заголовочный файл `time.h` – содержит описание функций для работы со временем.
- Тип данных `time_t` – определен в заголовочном файле `time.h` и предназначен для хранения времени в секундах.

- Функция `time` – определена в заголовочном файле `time.h` и возвращает время в секундах, которое прошло с начала выполнения программы.
- Функция `qsort` – определена в заголовочном файле `stdlib.h` и предназначена для быстрой сортировки массивов. Для работы этой функции, необходимо, помимо массива, его размера и размера его типа передать указатель на “функцию-компаратор”.

Выполнение работы.

Перед написанием основного кода в первых трёх строчках осуществляется включение заголовочных файлов: `stdlib.h`, `time.h`, `stdio.h`, которые содержат функции для быстрой сортировки, мониторинга времени и ввода/вывода соответственно. Также объявляется именованная константа `LEN` для хранения размера массива.

Далее необходимо написать “функцию-компаратор”, необходимую для работы функции `qsort`. Функция должна принимать два указателя на тип данных `void`. Возвращает функция значение типа `integer`: значение, большее нуля, если первый элемент должен стоять после второго в отсортированном массиве, значение, меньшее нуля, если первый элемент должен стоять до второго и нулевое значение, если оба элемента равнозначны. В теле функции оба переданных элемента приводятся к типу `int*`, разыменовываются, и передаются на вход функции `abs`, возвращающей модуль числа. Полученные значения сохраняются в переменных `first` и `second`, после чего с помощью нескольких условий функция возвращает значения в соответствии с тем, как было описано выше.

После определения “функции-компаратора” определяется тело функции `main`. В теле функции объявляются переменные `start` и `end` для хранения времени в секундах, прошедшего с начала выполнения программы и до начала работы функции `qsort` и для хранения времени в секундах, прошедшего от начала работы программы до конца работы функции `qsort` соответственно. Вычитая значение одной переменной из значения другой можно получить время работы функции `qsort`, которое будет сохраняться в переменной `answer`.

Далее определяется массив для хранения чисел, и осуществляется считывание. После считывания в переменную `start` сохраняется время, прошедшее сначала выполнения программы, выполняется сортировка массива, и снова замеряется время, прошедшее с начала выполнения программы. Результат вычета полученных значений сохраняется в переменную `answer`, после чего осуществляется вывод массива и значения переменной `answer`.

Тестирование.

Тестирование осуществлялось с размером массива, равным 10 (Так как элементов достаточно мало, то время сортировки не достигает одной секунды).

Таблица 1 – Результаты тестирования

№ п/п	Входные данные	Выходные данные	Комментарии
1.	4 -5 7 0 -4 -4 8 9 11 20	20 11 9 8 7 -5 4 -4 -4 0 0	Программа работает верно
2.	1 2 3 4 5 6 7 8 9 10	10 9 8 7 6 5 4 3 2 1 0	Программа работает верно

Выводы.

Были изучены и применены на практике функции стандартных заголовочных файлов языка Си. Была написана программа, позволяющая отсортировать массив чисел по убыванию их модулей и вычисляющей время такой сортировки.

ПРИЛОЖЕНИЕ А

ИСХОДНЫЙ КОД ПРОГРАММЫ

Название файла: `src/main.c`

```
#include <stdio.h>
#include <stdlib.h>
#include <time.h>
#define LEN 1000

int compare(const void* a, const void* b){
    int first = abs(*(int*)a);
    int second = abs(*(int*)b);
    if (first > second){
        return -1;
    } else if (first == second){
        return 0;
    } else if (first < second) {
        return 1;
    }
}

int main() {
    time_t start;
    time_t end;
    time_t answer;
    int array[LEN];
    int i;
    for (i = 0; i < LEN; i++){
        scanf("%d", &array[i]);
    }
    start = time(NULL);
    qsort(array, LEN, sizeof(int), compare);
    end = time(NULL);
    answer = end - start;
    for (i = 0; i < LEN; i++){
        if (i != LEN-1){
            printf("%d ", array[i]);
        } else {
            printf("%d\n", array[i]);
        }
    }
    printf("%d", answer);
    return 0;
}
```