МИНОБРНАУКИ РОССИИ
САНКТ-ПЕТЕРБУРГСКИЙ ГОСУДАРСТВЕННЫЙ
ЭЛЕКТРОТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ
«ЛЭТИ» ИМ. В.И. УЛЬЯНОВА (ЛЕНИНА)
Кафедра Математического Обеспечения и Применения ЭВМ

ОТЧЕТ

по лабораторной работе №1
по дисциплине «Программирование»
Тема: Обзор стандартной библиотеки языка Си.

Студент гр. 0382 _____ Кондратов Ю.А.

Преподаватель _____ Барленко Т.А.

Санкт-Петербург
2021

Цель работы.

Целью работы является изучение возможностей стандартной библиотеки языка программирования Си.

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.

В данной работе были использованы следующие заголовочные файлы: stdio.h, stdlib.h, time.h.

Использованные функции, объявления которых содержатся в этих заголовочных файлах:

- stdio.h:
  - int scanf (const char * format, ...) - функция считывания данных с консоли;

- int printf (const char * format, ...) - функция вывода данных на консоль.

- stdlib.h:
  - void qsort (void* base, size_t num, size_t size, int (*compar)(const void*,const void*)) - стандартная функция для выполнения быстрой сортировки;
  - int abs (int n) — функция для получению абсолютного значения целого числа.
- time.h:
  - clock_t clock (void) - возвращает количество временных тактов, прошедших с начала запуска программы;
  - макрос CLOCKS_PER_SEC — этот макрос заменяется на значение, представляющее собой число тиков в секунду.

Выполнение работы.

Первым этапом написания программы является подключение всех необходимых заголовочных файлов (см. основные теоретические положения).

Далее необходимо в функции main организовать считывание массива с консоли. Размер массива записан в именованной константе ARR_SIZE.

Считывание производится в массив arr типа int размера ARR_SIZE при помощи цикла for и функции scanf:

```
int arr[ARR_SIZE];
for (int i = 0; i < ARR_SIZE; i++) {
    scanf("%d", &arr[i]);
}
```

Далее в переменную clock_t time записывается время прошедшее с момента запуска программы, после чего производится сортировка массива.

Сортировка массива arr производится при помощи функции qsort, в качестве аргументов этой функции передаются:

- массив arr;
- размер массива ARR_SIZE;
- размер одного элемента массива в байтах sizeof(int);
- функция сравнения cmp;

Функция сравнения cmp реализована так, чтобы она возвращала отрицательное значение если абсолютное значение левого элемента больше абсолютного значения правого, в противном случае функция возвращает положительное значение или 0 в случае, если абсолютные значения элементов равны:

```
int cmp(const void *a, const void *b) {
    int* num1 = (int *) a;
    int* num2 = (int *) b;
    if (abs(*num1) > abs(*num2)) return -1;
    else if (abs(*num1) < abs(*num2)) return 1;
    else return 0;
}
```

После сортировки массива в переменную time записывается разность между значением, возвращаемым функцией clock и предыдущим значением переменной time. Таким образов находится количество тиков процессора, затраченных на сортировку массива.

Вывод отсортированного массива производится при помощи цикла for и функции printf:

```
for (int i = 0; i < ARR_SIZE; i++){
    printf("%d ", arr[i]);
}
```

Для вывод времени сортировки в секундах, а не в тиках процессора, значение в переменной time делится на макрос CLOCKS_PER_SEC.

Тестирование.

Тестирование производилось при значении ARR_SIZE равном 10. Результаты тестирования представлены в табл. 1.

Таблица 1 – Результаты тестирования

№ п/п	Входные данные	Выходные данные	Комментарии
1.	1 2 4 2 3 5 3 4 5 2	5 5 4 4 3 3 2 2 2 1 0.000003	Программа работает правильно
2.	83 29 2827 2928 98 287 412 1234 54 23	2928 2827 1234 412 287 98 83 54 29 23 0.000008	Программа работает правильно
3.	0 0 0 0 0 0 0 0 0 0	0 0 0 0 0 0 0 0 0 0 0.000004	Программа работает правильно
4.	1234 4321 2345 5423 3456 6534 4567 7654 5678 8765	8765 7654 6534 5678 5423 4567 4321 3456 2345 1234 0.000003	Программа работает правильно

Выводы.

В ходе работы были изучены возможности стандартной библиотеки языка Си

Разработана программа, выполняющая считывание и сортировку массива целых чисел при помощи функции qsort, определяющая время,

затраченное на сортировку массива при помощи функции clock() и макроса CLOCKS_PER_SEC.

Программа производит вывод отсортированного массива и времени, затраченного на сортировку.

ПРИЛОЖЕНИЕ А

ИСХОДНЫЙ КОД ПРОГРАММЫ

1. Название файла: main.c

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

#define ARR_SIZE 10

int cmp(const void *, const void *);

int main() {
    clock_t time;
    int arr[ARR_SIZE];
    for (int i = 0; i < ARR_SIZE; i++) {
        scanf("%d", &arr[i]);
    }
    time = clock();
    qsort(arr, ARR_SIZE, sizeof(int), cmp);
    time = clock() - time;
    for (int i = 0; i < ARR_SIZE; i++){
        printf("%d ", arr[i]);
    }
    printf("\n%f", ((float) time)/CLOCKS_PER_SEC);
    return 0;
}

int cmp(const void *a, const void *b) {
    int* num1 = (int *) a;
    int* num2 = (int *) b;
    if (abs(*num1) > abs(*num2)) return -1;
    else if (abs(*num1) < abs(*num2)) return 1;
    else return 0;
}
```