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 СИ.

Студентка гр. 0382 _____ Михайлова О.Д.

Преподаватель _____ Берленко Т.А.

Санкт-Петербург

2021

Цель работы.

Изучить стандартную библиотеку языка СИ.

Задание.

Вариант 4.

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.

- clock(void) – функция стандартной библиотеки time.h, Возвращает количество временных тактов, прошедших с начала запуска программы;
- CLOCKS_PER_SEC – макрос, с помощью которого функция clock получает количество пройденных тактов за 1 секунду;
- qsort(void * first, size_t number, size_t size, int (* comparator)(const void *, const void *)) – функция стандартной библиотеки stdlib.h, используется для быстрой сортировки элементов массива. На вход принимается 4 аргумента: first – указатель на первый элемент сортируемого массива, number – количество элементов сортируемого массива, size – размер одного элемента массива в байтах, comparator – функция, которая сравнивает два элемента.

Выполнение работы.

Переменные:

- #define SIZE 1000 – макрос, обозначающий количество элементов в массиве;
- int arr[SIZE] – массив;
- int start – переменная, в которую записывается количество временных тактов, прошедших с начала запуска программы, до сортировки элементов массива;
- int end – переменная, в которую записывается количество временных тактов, прошедших с начала запуска программы, после сортировки элементов массива;

В функции main с помощью цикла for и функции стандартной библиотеки scanf() считываются элементы массива. Далее с помощью функции clock() в переменную start записывается количество тактов, прошедших с начала запуска программы. Затем происходит сортировка элементов массива с помощью функции qsort, для которой создана функция-компаратор compare. Она принимает на вход два аргумента const void* arg1 и const void* arg2, которые затем приводятся к типу int*, далее сравнивает эти элементы по модулю и возвращает соответствующее значение. После сортировки снова реализуется функция clock(), результат которой записывается в переменную end. Программа выводит на экран отсортированный массив и разницу между значениями переменных start и end в секундах, полученную с помощью макроса CLOCKS_PER_SEC.

Разработанный программный код см. в приложении А.

Тестирование.

Результаты тестирования представлены для массива из 10 элементов в табл. 1.

Таблица 1 – Результаты тестирования

№ п/п	Входные данные	Выходные данные	Комментарии
1.	1 -2 -4 3 5 6 -7 8 9 9	9 9 8 -7 6 5 -4 3 -2 1 0.000003	Результат верный
2.	2 1 16 2 -18 -22 15 -3 13 0	-22 -18 16 15 13 -3 2 2 1 0 0.000003	Результат верный
3.	1 345 -292 453 2 3 890 4 876 -900	-900 890 876 453 345 -292 4 3 2 1 0.000005	Результат верный
4.	1 2345 7654 12 3456 7907 34 2345 87 11	7907 7654 3456 2345 2345 87 34 12 11 1 0.000004	Результат верный

Выводы.

Были изучена стандартная библиотека языка СИ.

Разработана программа, которая выполняет считывание элементов массива и его сортировку по невозрастанию модулей элементов с помощью функции qsort, а также считает время, за которое была совершена эта сортировка. В результате работы программы на экран выводится отсортированный массив и время в секундах.

ПРИЛОЖЕНИЕ А

ИСХОДНЫЙ КОД ПРОГРАММЫ

Название файла: pr_lb_1.c

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

#define SIZE 1000

int compare(const void* arg1, const void* arg2){
    if (abs(*(int*)arg2) > abs(*(int*)arg1)){
        return 1;
    }
    else if (abs(*(int*)arg2) == abs(*(int*)arg1)){
        return 0;
    }
    else if (abs(*(int*)arg2) < abs(*(int*)arg1)) {
        return -1;
    }
}

int main(){
    int arr[SIZE];
    int i;
    for (i=0; i<SIZE; i++){
        scanf("%d", &arr[i]);
    }
    int start, end;
    start = clock();
    qsort(&arr, SIZE, sizeof(int), compare);
    end = clock();
    for (i=0; i<SIZE; i++){
        printf("%d ", arr[i]);
    }
    printf("\n%f", (double)(end - start)/CLOCKS_PER_SEC);
    return 0;
}
```