**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

|  |  |  |
| --- | --- | --- |
| Студент гр. 6303 |  | Иванов Д.В. |
| Преподаватель |  | Чайка К.В. |

Санкт-Петербург

2017

**Цель работы:** провести обзор стандартной библиотеки,научиться использовать алгоритм быстрой сортировки элементов массива, определить время работы данного алгоритма.

**Задание:**

Написать 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

**Ход работы:**

1. Подключены необходимые для работы заголовочные файлы

#include <stdio.h>

#include <time.h>

#include <stdlib.h>

1. Для удобства работы с массивами разных размеров добавлена константа LengthOfArray, отвечающая за количество элементов в массиве.

#define LenghtOfArray 1000

1. Создан массив для необходимого числа элементов, числа считаны и записаны в элементы данного массива.

int ArrayOfNumbers [LenghtOfArray];

for (int i = 0; i < LenghtOfArray; i++)

{

scanf("%d", &ArrayOfNumbers [i]);

}

1. Для реализации алгоритма быстрой сортировки и подсчета времени её работы

* Написана функция сравнения compare, а также её прототип. Поскольку необходимо упорядочить элементы массива по убыванию модулей элементов, в стандартный вид данной функции внесены изменения

int compare(const void \*, const void \*);

…

int compare(const void \* a, const void \* b)

{

return (abs(\*(int\*)b) - abs(\*(int\*)a));

}

* Для подсчета времени работы функции qsort перед её вызовом сохранено количество тактов процессора.

clock\_of\_start = clock();

* Вызвана функция быстрой сортировки qsort

qsort(ArrayOfNumbers, LenghtOfArray, sizeof(int), compare);

* После окончания работы функции qsort общее количество тактов процессора записано в переменную

clock\_of\_end = clock();

1. Написан алгоритм вывода элементов отсортированного по убыванию модулей массива чисел, а также времени работы функции

for (int i = 0; i < LenghtOfArray; i++)

{

printf("%d ", ArrayOfNumbers [i]);

}

printf("\n%f", (float)(clock\_of\_end-clock\_of\_start)/CLOCKS\_PER\_SEC);

1. Затем файл main.c с кодом написанной программы и отчёт о проведенной работе был загружен на github c помощью консоли:

* Создана новая ветка Ivanov\_2\_lr1

git checkout –b Ivanov\_2\_lr1

* Создана новая директория Ivanov\_2\_lr1 и файл main.c в ней, куда помещен код написанной программы

mkdir Ivanov\_2\_lr1

cd Ivanov\_2\_lr1

nano main.c

* Созданные файлы добавлены для загрузки

git add Ivanov\_2\_lr1/

* Добавлен комментарий

git commit –m “2 semestr. Lab 1”

* Файлы загружены на репозиторий Git

git push origin Ivanov\_2\_lr1

**Вывод:** в ходе работы получены навыки работы с функцией быстрой сортировки массивов qsort, проведен подсчёт времени работы данной функции, выполнен обзор стандартной библиотеки.