**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:  **Структуры и обзор stdlib**

|  |  |  |
| --- | --- | --- |
| Студент гр. 0384 |  | Дзаппала Д. |
| Преподаватель |  | Шевская Н.В. |

Санкт-Петербург

2021

## Цель работы.

Ознакомиться с функциями стандартной библиотеки stdlib и научиться применять их в работе.

## Задание.

Вариант 3.

## 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.*

## Выполнение работы.

Вначале программы идет выделение памяти под SIZE (SIZE = 1000) указателей на целочисленный тип данных, после чего идет заполнение кучи цифрами через цикл for и функцию ввода scanf(). Так как в работе проверяется время работы двух алгоритмов сортировки, была создана функция копирования данных из первого массива в другой, которая возвращала указатель на первый элемент нового массива.

Далее, с помощью функции clock() из библиотеки time.h, запускается отчет тактов процессора. Первой проверкой идет алгоритм сортировки Пузырьком. Чтобы узнать сколько тактов с начала сортировки прошло, в той же переменной мы используем ф-ию clock() и вычитаем эту же переменную, таким образом мы узнаем сколько тактов прошло.

Далее такой же алгоритм с проверкой алгоритма Быстрой Сортировки (Quick Sort, стандартная ф-ия qsort()).

В конце выводим отсортированный массив, время сортировки пузырьком, время быстрой сортировки. Чтобы узнать сколько времени прошло в секундах, нужно наши такты поделить на макрос CLOCKS\_PER\_SEC.

## Выводы.

Были изучены ф-ии различных стандартных библиотек.

# Приложение А Исходный код программы

Название файла: lab.c

#include<stdio.h>

#include<stdlib.h>

#include<time.h>

#define SIZE 5

int \* arrCpy(int\* a, int len){

int \* b = (int\*) malloc (len\* sizeof(int));

for (int i = 0; i < len; i++) b[i] = a[i];

return b;

}

void swap(int \*n1, int \*n2){

int temp = \*n1;

\*n1 = \*n2;

\*n2 = temp;

}

void BubbleSort(int \*arr, int len){

for (int i = 0; i < len - 1; i++){

for (int j = 0; j < len - 1 - i; j++)

if (arr[j] > arr[j+1])

swap(&arr[j], &arr[j+1]);

}

}

int comp(const void \*a, const void \*b){

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

}

int main()

{

//adding nums

int \*nums = (int\*) malloc (SIZE \* sizeof(int));

for (int i = 0; i < SIZE; i++) scanf("%d", &nums[i]);

//copying nums to another array

int \*numscpy = arrCpy(nums, SIZE);

//on clock

clock\_t timeBS = clock();

//Bubble sort

BubbleSort(nums, SIZE);

//stop time for bubble sort

timeBS = clock() - timeBS;

//on clock for QSort

clock\_t timeQS = clock();

//QSort

qsort(numscpy, SIZE, sizeof(int), comp);

//stop time

timeQS = clock() - timeQS;

for (int i = 0; i < SIZE; i++) printf("%d ", numscpy[i]);

printf("\n");

printf("%f\n%f\n", ((float)timeBS)/CLOCKS\_PER\_SEC, ((float)timeQS)/CLOCKS\_PER\_SEC);

return 0;

}