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

Студент гр. 0382 Литягин С.М.

Преподаватель Чайка К.В., Берленко Т.А.

Санкт-Петербург
2021

Цель работы.

Изучение и использование функций стандартной библиотеки Си.

Задание.

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.

Основные теоретические положения.

Функции библиотеки *stdio.h*:

- *printf()* – функция вывода на консоль;
- *scanf()* – функция ввода данных из консоли.

Циклы:

- *for(){<переменная>; <условие>; <выражение_1>}* – код в теле цикла будет выполняться до тех пор, пока объявленная в цикле переменная будет удовлетворять условию цикла, выражение_1 каким-либо способом меняет

значение этой переменной.

Операторы:

- *if(){} ... else{}* – если выполняется условия, указанное в круглых скобках, то выполняется код в фигурных скобках после if, иначе – в фигурных скобках после else (else не является обязательной частью конструкции);

Функция библиотеки time.h:

- clock() – возвращает количество тактов процессора, которое приблизительно соответствует времени работы вызывающей программы. Для преобразования этого значения в секунды нужно разделить его на значение CLOCKS_PER_SEC

Тип данных clock_t – тип данных, способный представлять временные тики и поддерживающий арифметические операции

Выполнение работы.

В начале main функции создаем два целочисленных массива arr и arr2 размером 1000. Заполняем их вводимыми числами.

Сохраняем в переменную timer_1 типа clock_t количество тактов. Производим сортировку массива пузырьком. Сохраняем в переменную timer_2 типа clock_t количество тактов.

Производим сортировку функцией быстрой сортировки. Сохраняем в переменную timer_3 типа clock_t количество тактов.

В переменные uns1 и uns2 типа float сохраняем значения сортировки пузырьком и быстрой сортировки в секундах, деля значения (timer_2 – timer_1) и (timer_3 – timer_2) на макрос CLOCKS_PER_SEC.

В конце выводим данные в заданном условием порядке.

Разработанный программный код см. в приложении А.

Тестирование.

Результаты тестирования представлены в табл. 1.
*для удобства при тестировании бралось не 1000 чисел, а 10.

Таблица 1 – Результаты тестирования

№ п/п	Входные данные	Выходные данные	Комментарии
1	1 2 3 4 5 6 8 7 10 9	1 2 3 4 5 6 7 8 9 10 0.000002 0.000002	Программа работает правильно
2	98 32 56 23 5 2 9 0 2 55	0 2 2 5 9 23 32 55 56 98 0.000002 0.000003	Программа работает правильно

Выводы.

В ходе работы были изучены основные функции стандартных библиотек Си. А также была разработана программа для сравнения времени сортировки массива методом пузырька и функцией быстрой сортировки.

Код программы см. в Приложении А.

ПРИЛОЖЕНИЕ А

ИСХОДНЫЙ КОД ПРОГРАММЫ

Название файла: main.c

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

int compare(const void * x1, const void * x2){
  return ( *(int*)x1 - *(int*)x2 );
}

int main(){
    int arr[1000];
    int arr2[1000];
    int buff;
    float uns1, uns2;
    _Bool flag;
    for(int i = 0; i < 1000; i++){
        scanf("%d", &arr[i]);
        arr2[i] = arr[i];
    }
    clock_t timer1 = clock();
    for(int i = 1000-1; i >=0; i--){
        flag = 1;
        for(int j = 0; j < i; j++){
            if(arr[j] > arr[j+1]){
                buff = arr[j];
                arr[j] = arr[j+1];
                arr[j+1] = buff;
                flag = 0;
            }
        }
        if(flag == 1){
            break;
        }
    }
    clock_t timer2 = clock();
    qsort(arr2, 1000, sizeof(int), compare);
    clock_t timer3 = clock();
    for(int i = 0; i < 1000; i++){
        printf("%d ", arr[i]);
    }
    uns1 = (float)(timer2 - timer1)/(CLOCKS_PER_SEC);
    uns2 = (float)(timer3 - timer2)/(CLOCKS_PER_SEC);
    printf("\n%f", uns1);
    printf("\n%f", uns2);
    return 0;
}
```