**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 |  | Амбарцумян А.В. |
| Преподаватель |  | Берленко Т.А. |

Санкт-Петербург

2017

**Цель работы:** провести обзор стандартной библиотеки,научиться использовать алгоритм быстрой сортировки элементов массива, алгоритм двоичного поиска, алгоритм линейного поиска и определить время работы данных алгоритмов.

**Задание:**

Написать программу, на вход которой подается массив целых чисел длины 1000, при этом число 0 либо встречается один раз, либо не встречается. Программа должна содержать следующие действия:

* отсортировать массив, используя алгоритм быстрой сортировки (из стандарной библиотеки)
* определить, присутствует ли в массиве число **0**, используя алгоритм двоичного поиска
* посчитать время, за которое совершен поиск числа **0**, используя при этом **функцию стандартной библиотеки**
* вывести строку "exists", если ноль в массиве есть и "doesn't exist" в противном случае
* вывести время, за которое был совершен двоичный поиск
* определить, присутствует ли в массиве число **0**, используя перебор всех чисел массива
* посчитать время, за которое совершен поиск числа **0**перебором, используя при этом **функцию стандартной библиотеки**
* вывести строку "exists", если **0** в массиве есть и "doesn't exist" в противном случае
* вывести время, за которое была совершен поиск перебором

**Ход работы:**

1. Подключены необходимые для работы заголовочные файлы

#include <stdio.h>

#include <stdlib.h>

#include <time.h>

1. Создан массив для необходимого числа элементов(1000), числа считаны и записаны в элементы данного массива.

int mass[1000];

int i;

for (i=0; i<1000; i++)

{

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

}

1. Массив отсортирован стандартной функцией

qsort (mass, 1000, sizeof(int),compar);

Для которой написана функция сравнения compare.

int compar(const void\* x, const void\* y)

{

return ( \*(int\*)x - \*(int\*)y );

}

1. Для реализации алгоритма двоичного поиска и подсчета времени её работы**:**

* Создано 2 переменных типа int**:** start и end
* Перед вызовом функции двоичного поиска bsearch, значению переменной start было присвоено значение функции clock() (время от начала работы программы)

start=clock();

* Вызвана функция двоичного поиска bsearch

ptr=(int\*) bsearch (&key, mass, 1000, sizeof(int), compar);

Возвращаемое значение которой передается в указатель ptr

* После окончания работы функции значению переменной end было присвоено значение функции clock()

end= clock();

1. Написан алгоритм вывода результата поиска элемента, а также времени работы функции

if(ptr==NULL)

{

printf ("doesn't exist\n");

}

else

{

printf ("exists\n");

}

printf("%d\n", (end-start));

1. Затем, для реализации алгоритма поиска нужного числа перебором**:**

* Создано переменная key типа int, хранящая элемент, который нужно найти
* Создана функция sravnenie, получающая на вход указатель на массив и на ключ и циклом do while проверяет каждый элемент, сравнивая его с ключом

int sravnenie (int\* mass, int\* key)

{

int i = 0;

do {

if (mass[i]==\*key) {

return 0;}

else {

return 1;}

} while (mass[i]!= 0);

}

* Перед вызовом функции sravnenie, значению переменной start было присвоено значение функции clock() (время от начала работы программы)

start=clock();

* Вызвана функция sravnenie для поиска числа перебором. Её значение присвоено переменной s

s=sravnenie (mass, &key);

* После окончания работы функции значению переменной end было присвоено значение функции clock()
* end= clock();

1. Написан алгоритм вывода результата поиска элемента, а также времени работы функции

if (s==0){

printf ("exist\n"); }

else {

printf ("doesn't exist\n");}

printf("%d\n", (end-start));

1. Затем файл lab1.c с кодом написанной программы и отчёт о проведенной работе был загружен на github c помощью консоли:

* Создана новая ветка Ambartsumyan\_lr\_1

git checkout –b Ambartsumyan\_lr\_1

* Создана новая директория Ambartsumyan\_lr\_1 и файл lab1.c в ней, куда помещен код написанной программы

mkdir Ambartsumyan\_lr\_1

cd Ambartsumyan\_lr\_1

nano lab1.c

* Созданные файлы добавлены для загрузки

git add Ambartsumyan\_lr\_1/

* Добавлен комментарий

git commit –m “1 Lab”

* Файлы загружены на репозиторий Git

git push origin Ambartsumyan\_lr\_1

**Вывод:** в ходе работы получены навыки работы с функцией быстрой сортировки массивов qsort, алгоритмом двоичного поиска-bsearch, алгоритмом линейного поиска и определения времени работы данных алгоритмов.