**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. 0304: |  | Шквиря Е.В. |
| Преподаватели: |  | Чайка К. В. |

Санкт-Петербург

2021

## Цель работы.

Изучить способы взаимодействия со стандартной библиотекой языка Си.

## Задание.

## Напишите программу, на вход которой подается массив целых чисел длины **1000,** при этом число **0** либо встречается один раз, либо не встречается.

Программа должна совершать следующие действия:

* отсортировать массив, используя алгоритм быстрой сортировки (см. **функции стандартной библиотеки)**
* определить, присутствует ли в массиве число **0**, используя алгоритм двоичного поиска (для реализации алгоритма двоичного поиска используйте **функцию стандартной библиотеки)**
* 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.

*Результат двоичного поиска, время двоичного поиска, результат поиска перебором и время поиска перебором должны быть выведены именно в таком порядке и разделены символом перевода строки.*

## Выполнение работы.

Порядок выполнения поставленной задачи программой:

1. Создаётся и инициализируется массив на 1000 элементов. Далее он сортируется для корректной работы бинарного поиска.
2. Создаётся переменная *start\_time* для фиксирования начального времени.
3. Результат работы бинарного поиска записывается в переменную *result1*. Если *key* нашёлся в массиве, то выводится результат «exists», иначе «doesn`t exist». Время работы фиксируется в переменной *time\_binsearch.*
4. Результат работы полного перебора записывается в переменную *result2.* Если *key* нашёлся в массиве, то выводится результат «exists», иначе «doesn`t exist». Время работы фиксируется в переменной *time\_brute.*

Используемые переменные:

SIZE — количество элементов в массиве,

array — массив чисел,

key — искомое число,

start\_time — начало отсчёта времени работы алгоритма,

result1 — результат работы бинарного поиска,

time\_binsearch — время работы бинарного поиска,

result2 — результат работы полного перебора,

time\_brute — время работы полного перебора.

Используемые функции:

comp — сравнение двух чисел.

Разработанный программный код см. в приложении А.

## Тестирование.

Результаты тестирования представлены в табл. 1.

Таблица 1 – Результаты тестирования

|  |  |  |  |
| --- | --- | --- | --- |
| № п/п | Входные данные | Выходные данные | Комментарии |
|  | 0 1 … 998 999 | exists | Верный ответ |
|  | 1 2 … 999 1000 | doesn't exist | Верный ответ |
|  | 1000 999 … 2 1 | doesn't exist | Верный ответ |
|  | 999 998 … 1 0 | exists | Верный ответ |

## Выводы.

Были изучены способы взаимодействия со стандартной библиотекой языка Си.

Разработана программа, получающая от пользователя массив чисел и ищущая в нём число 0 разными способами. Также происходит вычисление времени работы каждого из алгоритмов.

# Приложение А Исходный код программы

Название файла: lab1.c

#include <stdlib.h>  
#include <time.h>  
#include <stdio.h>  
#define SIZE 1000  
  
  
int comp(const void\* a, const void\* b)  
{  
 int A = \*((int\*)a), B = \*((int\*)b);  
 if(A > B)  
 return 1;  
 else if (A < B)  
 return -1;  
 else  
 return 0;  
}  
  
int main()  
{  
 int array[SIZE];  
 for (int i = 0; i < SIZE; ++i)  
 {  
 scanf("%d", &array[i]);  
 }  
 qsort(array,SIZE,sizeof(int),comp);  
  
 int key = 0;  
 long start\_time = clock();  
 int \*result1 = bsearch(&key,array,SIZE,sizeof(int),comp);  
 double time\_binsearch = (double)(clock() - start\_time) / CLOCKS\_PER\_SEC;  
 if(result1 != NULL)  
 printf("exists\n");  
 else  
 printf("doesn't exist\n");  
 //printf("%lf\n", time\_binsearch);  
  
 int result2 = -1;  
 start\_time = clock();  
 for (int i = 0; i < SIZE; ++i)  
 {  
 if(array[i] == key)  
 result2 = i;  
 }  
 double time\_brute = (double)(clock() - start\_time) / CLOCKS\_PER\_SEC;  
// if(result2 != -1)  
// printf("exists\n");  
// else  
// printf("doesn't exist\n");  
// printf("%lf\n", time\_brute);  
  
 return 0;  
}