Министерство образования Республики Беларусь

Учреждения образования

«Брестский государственный технический университет»

Кафедра ИИТ

Лабораторная работа №3

По дисциплине: “Алгоритмы и структуры данных”

**Тема: “ Исследование и оценка алгоритмов поиска”**

Выполнил:

Студент 1 курса

Группа ПО-7

Лобан К.Ю.

Проверила:

Глущенко Т.А.

Брест 2021

Лабораторная работа №3

**Цель:** разработка программ, реализующих различные алгоритмы поиска, и оценка их временной и пространственной сложности.

**Ход работы**

**Вариант 3**

Разработать алгоритм и программу дихотомического поиска. В качестве исходных данных использовать массив целых чисел, который вводится с клавиатуры. Аргумент поиска – число.

**Дихотомический поиск** *(BinarySearch)*

1. **Псевдокод:**

Algorithm (BinarySearch)

1: function BinarySearch(*arrA*[]*, n, chislo*)

2: *left ←* 0

3: *right ← n*

4: while *left ≤ right* do

*5: mid ←* (*left* + *right*) */* 2

6: if *chislo > arrA*[*mid*] then

7: *left r ← mid* + 1

8: else if *chislo < arrA*[*mid*] then

9: *right ← mid −* 1

10: else

11: *return mid*

12: if *left > right*

13: *return −* 1

14: end if

15: end while

16: end function

1. **Словесное описание:**

Этот поиск будет работать, если масив будет отсартированный. Алгоритм поиска можно объяснить на словаре слов. Чтобы найти какую-то букву, мо открываем словарь по середине. Потом смотрим в какой стороне буква – в правой половине или в левой. Далее пролученную половину делим ещё на половину и повторяем алгоритм дальше. Как нашли элемент, который был задан, то индекс - элемента наш ответ.

1. **Текст программы**

#include <iostream>

#include <ctime>

#include <stdlib.h>

#include <chrono>

using namespace std;

int Funkzia(int\* arrA, int N, int chislo)

{

int left = 0;

int right = N;

int mid = 0;

while (left <= right)

{

mid = (left + right) / 2;

if (chislo > arrA[mid])

{

left = mid + 1;

}

else if (chislo < arrA[mid])

{

right = mid - 1;

}

else

{

return mid;

}

if (left > right)

{

return -1;

}

}

}

int main()

{

setlocale(LC\_ALL, "rus");

int N;

cout << "Введите размерность массива: ";

cin >> N;

int\* arrA = new int[N];

cout << "Массив arrA1: " << endl;

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

{

cin >> arrA[i];

}

int temp = 0;

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

{

for (int j = i + 1; j < N; j++)

{

if (arrA[i] > arrA[j])

{

int temp = arrA[j];

arrA[j] = arrA[i];

arrA[i] = temp;

}

}

}

cout << endl;

cout << "Массив arrA2: ";

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

{

cout << arrA[i] << " ";

}

cout << endl;

int chislo;

int index = 0;

cout << "Введите число, которое нужно найти: ";

cin >> chislo;

chrono::steady\_clock::time\_point begin = chrono::steady\_clock::now();

index = Funkzia(arrA, N, chislo);

chrono::steady\_clock::time\_point end = chrono::steady\_clock::now();

if (index >= 0)

{

cout << "\nЧисло " << chislo << " находится в arrA2[" << index << "]\n" << endl;

}

else

{

cout << "\nЧисло " << chislo << " отсутствует в массиве\n" << endl;

}

\_\_int64 secondsDuratin = chrono::duration\_cast<chrono::seconds>(end - begin).count();

\_\_int64 microsecondsDuratin = chrono::duration\_cast<chrono::microseconds>(end - begin).count();

\_\_int64 nanosecondsDuratin = chrono::duration\_cast<chrono::nanoseconds> (end - begin).count();

cout << "Длительность алгоритма в секундах = " << secondsDuratin << endl;

cout << "Длительность алгоритма в микросекундах = " << microsecondsDuratin << endl;

cout << "Длительность алгоритма в наносекундах = " << nanosecondsDuratin << endl;

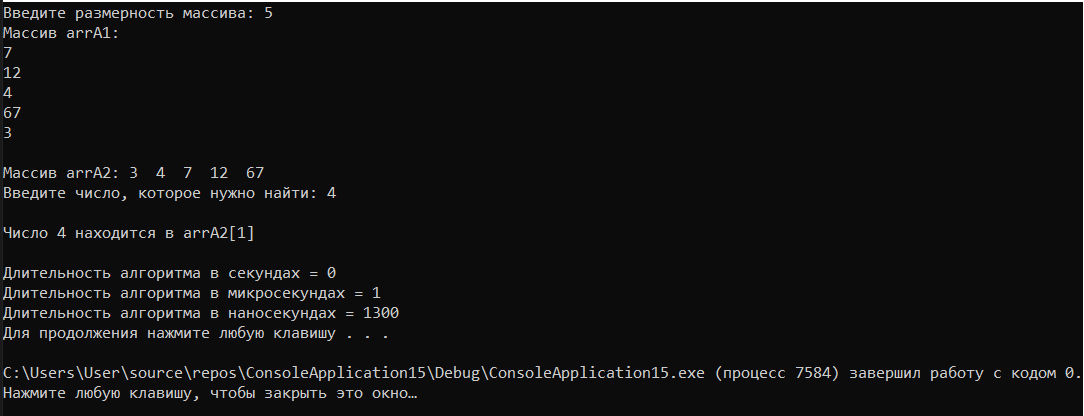
delete[] arrA;

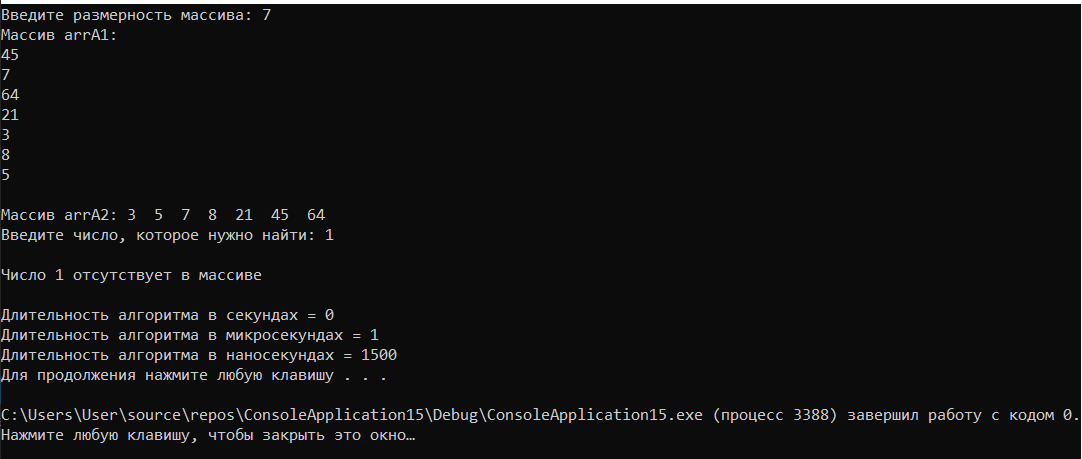
system("pause");

return 0;

}

**Результат программы:**





Экспериментальная оценка времени выполнения поиска в виде графика

**Вывод:** в ходе лабораторной работы разработал программу, реализующую дихотомический алгоритм поиска, а также оценку его временной и пространственной сложности.