Отчет по лабораторной работе 1.

1. ***Задача 1. Линейный поиск с барьером***

Реализовать линейный поиск с барьером. Выполнить анализ временной сложности.

**Входные данные**

Одномерный массив, заполненный случайными числами.

Число *A*

**Выходные данные**

*True*, если число *А* содержится в массиве, *False* – в противном случае.

using System;

namespace zadanie\_1

{

class Program

{

static int size = 100;

static void Main(string[] args)

{

Random rand = new Random();

int []arr =new int [size];

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

{

arr[i] = rand.Next(100);

Console.WriteLine("arr[" + i + "]= " + arr[i]);

}

Console.WriteLine("\nВведите число для поиска: ");

int value = int.Parse(Console.ReadLine());

if (find(arr, value))

{

Console.WriteLine("\nTrue");

}

else

{

Console.WriteLine("\nFalse");

}

}

static bool find(int []arr, int value)

{

bool findd = false;

if (size != 0)

{

int last = arr[size - 1];

arr[size - 1] = value;

int i=0;

while(arr[i] != value)

{

i++;

}

arr[size - 1] = last;

if (i != (size - 1) || value == last)

{

findd = true;

}

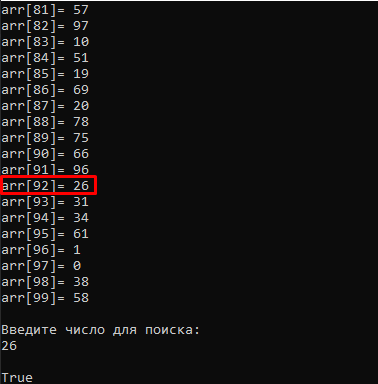
}

return findd;

}

}

}



1. ***Задача 2. Метод половинного деления уточнения корня уравнения f(x)=0 на промежутке [a; b]***

**Входные данные**

Дано уравнение f(x)=0, промежуток [a; b] и допустимая погрешность для нахождения корня e>0.

**Выходные данные**

x0 – корень уравнения

k – количество итераций

Провести вычислительный эксперимент по влиянию выбора условия выхода из цикла на число итераций и точность получаемого решения. Результаты эксперимента привести в виде таблицы

F(x)=x+5; e= 0.00000001;

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| [a; b] | |a(k)-b(k)|≤e | | | |a(k) |≤e или |b(k)|≤e или |s(k) |≤e | | | |a(k)-b(k)|≤1e-12 | | |
| k | x0 | Δ | k | x0 | Δ | k | x0 | Δ |
| [-10;10] | 31 | -5 | -0,375 | 1 | -5 | 0,25 | 13 | -5 | 3,6е-12 |
| [-100;100] | 35 | -4,999999998835847 | 0,26485 | 1 | -4,999999998835847 | 3,115 | 12 | -4,999999999999716 | 2,5е-12 |

using System;

namespace zadanie\_2

{

class Program

{

static double F(double x)

{

return x + 5;

}

static void Main()

{

const double element = 0;

double E = 0.00000001;

double a = -10;

double b = 10;

double x;

double k = 0;

do

{

k++;

x = F(((a + b) / 2));

if (x > element)

{

b = (a + b) / 2;

}

else

{

a = (a + b) / 2;

}

} while (!(Math.Abs(F(a)-F(b)) <= E));

if (E - Math.Abs(F(a)) > E - Math.Abs(F(b)))

{

Console.WriteLine(a);

}

else

{

Console.WriteLine( b);

}

Console.WriteLine("k= "+k);

x=0;

k = 0;

do

{

k++;

x = F(((a + b) / 2));

if (x > element)

{

b = (a + b) / 2;

}

else

{

a = (a + b) / 2;

}

} while (!(Math.Abs(F(a)) <= E || Math.Abs(F(b)) <= E));

if (E - Math.Abs(F(a)) > E - Math.Abs(F(b)))

{

Console.WriteLine(a);

}

else

{

Console.WriteLine(b);

}

Console.WriteLine("k= " + k);

x = 0;

k = 0;

do

{

k++;

x = F(((a + b) / 2));

if (x > element)

{

b = (a + b) / 2;

}

else

{

a = (a + b) / 2;

}

} while (!(Math.Abs(F(a) - F(b)) <= 1e-12));

if (E - Math.Abs(F(a)) > E - Math.Abs(F(b)))

{

Console.WriteLine(a);

}

else

{

Console.WriteLine(b);

}

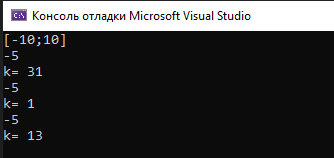
Console.WriteLine("k= " + k);

Console.ReadKey();

}

}

}



1. ***Задача 3. Бинарный поиск в частично упорядоченном массиве***

Дан частично упорядоченный массив, который был получен из упорядоченного по не убыванию элементов массива путем перестановки двух его частей. Например, массив *M*[9, 10, 13, 2, 2, 5, 7, 8] был получен из массива *Х*[2, 2, 5, 7, 8, 9, 10, 13]. Реализовать алгоритм поиска элемента *А* в массиве *M*, имеющий оценку временной сложности О(2log2*n*).

**Входные данные**

Частично упорядоченный массив *M*

Число *А*

**Выходные данные**

*True*, если число *А* содержится в массиве *М*, *False* – в противном случае.

using System;

namespace zadanie\_3

{

class Program

{

static void Main(string[] args)

{

if (bs(3))

{

Console.WriteLine("True");

}

else

{

Console.WriteLine("False");

}

}

static bool bs(int number)

{

bool find = false;

int[] array= { 13, 24, 45, 1, 2, 3, 5, 7, 9 };

int a = 0, b = array.Length;

int max = array[0];

if (number > array[array.Length - 1] && number < array[0])

return false;

if (number == array[0])

return false;

do

{

var mid = Math.Floor((double)(a + b) / 2);

if (array[(int)mid] > max)

{

a =(int) mid;

max = array[(int)mid];

}

else b = (int)mid;

} while (a != b);

int indexOfMax = a;

if (number == array[indexOfMax])

find = true;

if (number>array[0])

{

a = 0;

b = indexOfMax;

}

else

{

a = indexOfMax;

b = array.Length;

}

int pos = -1;

do

{

var mid = Math.Ceiling((double)(a + b) / 2);

if (array[(int)mid] < number)

{

a = (int)mid;

}

if (array[(int)mid] > number)

{

b = (int)mid - 1;

}

if (array[(int)mid] == number) pos = (int)mid;

} while (a != b && pos == -1);

if (pos != -1)

{

find = true ;

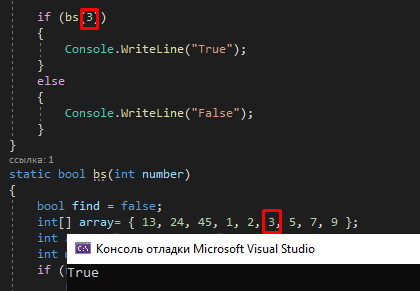
}

return find;

}

}

}



1. ***Задача 4. Поиск в двумерном массиве***

Дан двумерный массив *А* размерности *n*\**m*, каждая строка и каждый столбец которого упорядочены по не убыванию, и число *Х*. Определить, содержится ли число *Х* в массиве *А*

1. за время O(*n*2), используя последовательный просмотр всех элементов;
2. за время O(min(*n*, *m*)\*log2(max(*n*, *m*))), используя бинарный поиск;
3. за время O(*n*+*m*), используя линейный поиск.

**Входные данные**

*n* *m* – размерность массива

Массив *А*

Число *Х*

**Выходные данные**

*True*, если число *Х* содержится в массиве *А*, *False* – в противном случае.

using System;

namespace zadanie\_4

{

class Program

{

static void Main(string[] args)

{

if (bs(6))

{

Console.WriteLine("True");

}

else

{

Console.WriteLine("False");

}

if (linearsearch(6))

{

Console.WriteLine("True");

}

else

{

Console.WriteLine("False");

}

}

static bool bs(int element)

{

bool find = false;

int[,] array = {

{ 1, 3, 6 },

{ 2, 4, 13 },

{ 3, 5, 14},

{ 4, 6, 15},

{ 5, 9, 21}

};

if (array.GetLength(0) > array.Length)

{

int[] tmpArray = new int[array.Length];

int k = 0;

for (int i = 0; i < array.GetLength(0); i++)

{

for (int j = 0; j < array.GetLength(1); j++)

{

tmpArray[k] = array[i, j];

k++;

}

}

for (int i = 0; i < array.Length; i++)

{

var a = 0;

var b = array.GetLength(0) - 1;

do

{

int mid = (int)Math.Floor((double)(a + b) / 2);

if (tmpArray[mid] > element)

{

b = mid;

}

else a = mid;

if (tmpArray[a] == element)

find = true;// Console.WriteLine("i= " + i + "a=" + a); // return i, a ;

else

if (tmpArray[b] == element) find = true;// Console.WriteLine("i= " + i + "b=" + b);

} while (a != b - 1);

}

}

else

{

int[] tmpArray = new int[array.Length];

int k = 0;

for (int i = 0; i < array.GetLength(0); i++)

{

for (int j=0; j<array.GetLength(1); j++)

{

tmpArray[k] = array[i, j];

k++;

}

}

for (int i = 0; i < array.Length; i++)

{

int a = 0;

int b = array.Length - 1;

do

{

int mid = (int)Math.Floor((double)(a + b) / 2);

if (tmpArray[mid] > element)

{

b = mid;

}

else

a = mid;

if (tmpArray[a] == element)

find = true;

else

if (tmpArray[b] == element)

find = true;

} while (a != b - 1);

}

}

return find;

}

static bool linearsearch(int element)

{

bool find = false;

int[,] array = {

{ 1, 3, 6 },

{ 2, 4, 13 },

{ 3, 5, 14},

{ 4, 6, 15},

{ 5, 9, 21}

};

int i = array.GetLength(1) - 1;

for (; i >= 0; i--)

{

if (array[array.GetLength(0) - 1, i] < element) continue;

for (int j = array.GetLength(0) - 1; j >= 0; j--)

{

if (array[j, i] == element)

{

find = true;

}

}

}

return find;

}

}

}

