МИНИСТЕРСТВО ОБРАЗОВАНИЯ И НАУКИ

РОССИЙСКОЙ ФЕДЕРАЦИИ

ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ БЮДЖЕТНОЕ ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ

ВЫСШЕГО ОБРАЗОВАНИЯ

«БЕЛГОРОДСКИЙ ГОСУДАРСТВЕННЫЙ

ТЕХНОЛОГИЧЕСКИЙ УНИВЕРСИТЕТ им. В.Г.ШУХОВА»

(БГТУ им. В.Г. Шухова)

Кафедра программного обеспечения вычислительной техники и автоматизированных систем

Дисциплина: Алгоритмы и структуры данных

Лабораторная работа № 4

**Сравнительный анализ алгоритмов поиска (Pascal/C)**

|  |  |
| --- | --- |
|  | Выполнила: ст. группы ПВ-21  Зановская А.И.  Проверил: Синюк В.Г. |

Белгород

2017

Л а б о р а т о р н а я р а б о т а № 4

**Сравнительный анализ алгоритмов поиска (Pascal/C)**

**Цель работы:** изучение алгоритмов поиска элемента в массиве и **з**акрепление навыков в проведении сравнительного анализа алгоритмов.

# З а д а н и е

1. Изучить алгоритмы поиска:

1) в неупорядоченном массиве:

- линейный;

- быстрый линейный;

2) в упорядоченном массиве:

- быстрый линейный;

- бинарный;

- блочный.

2. Разработать и программно реализовать средство для проведения экспериментов по определению временных характеристик алгоритмов поиска.

#include <stdio.h>

#include <stdlib.h>

#include <locale.h>

#include <math.h>

int n=50;

int main()

{

setlocale (LC\_ALL, "Rus");

int i,x,k,a[451], b1[100],b2[100],b3[100],b4[100],b5[100];

srand(6737);

for (i=1;i<100;i++)

{

form(a,1);

if (rand()%2)

x=a[(rand()+29)%n];

else x=rand()%(2\*n);

b1[i]=n\_lin(a,x,n);

form(a,1);

a[n]=x;

b2[i]=n\_qlin(a,x);

}

form(a,2);

for (i=1;i<100;i++)

{

if (rand()%2)

x=a[rand()%n];

else x=rand()%n;

u\_lin(a,x,k,n,&k); b3[i]=k;

u\_bin(a,x,n,&k); b4[i]=k;

u\_blok (a,x,n,&k); b5[i]=k;

}

printf ("\nНеупорядоченный массив\nЛинейный поиск\n");

m(b1);

printf ("\nБыстрый линейный поиск");

m(b2);

printf ("\nУпорядоченный поиск\nБыстрый линейный поиск\n");

m(b3);

printf ("\nБинарный поиск");

m(b4);

printf ("\nБлочный поиск");

m(b5);

}

void m (int \*b)

{

int i,s=0,max=b[1];

for (i=1;i<100;i++)

{

s+=b[i];

if (b[i]>max)

max=b[i];

}

s/=100;

printf ("\nСреднее = %d\nМаксимальное = %d",s,max);

}

int n\_lin (int \*a, int x, int n)

{

int i;

for (i=0;i<n && a[i]!=x;i++);

return i;

}

//необходимо a[n]:=х

int n\_qlin (int \*a, int x)

{

int i;

for (i=0;i<n && a[i]!=x;i++);

return i;

}

int u\_lin (int \*a, int x,int i, int n, int \*k)

{

\*k=i;

for (;i<n && a[i]<x;i++);

\*k=i-\*k;

if (i<n)

return i;

else

return 0;

}

int u\_bin (int \*a, int x, int n,int \*k)

{

int l, r,m;

l=0;

r=n-1;

m=l+(r-l)/2;

while (l<r)

{

(\*k)++;

if (a[m]==x)

return m;

if (a[m]<x)

l=m+1;

else

r=m;

m=l+(r-l)/2;

}

return -(1+l);

}

int u\_blok (int \*a, int t, int n,int \*k)

{

int i,x,j,f,p;

j=x=sqrt(n);

\*k=1;

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

{

(\*k)++;

if (a[j\*i+x-1]>t)

{

p=u\_lin (a, t, j\*i, j\*i+x-2, &f);

\*k+=f;

return p;

}

else

if (a[j\*i+x-1]==t)

return (j\*i+x-1);

}

if (a[n-1]>t)

{

p=u\_lin (a, t, j\*i, j\*i+x-2, &f);

\*k+=f;

return p;

}

return -1;

}

void form (int a[], int k)

{

int i;

switch (k)

{

case 1:

srand(1022);

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

a[i]=i+rand()%n;

break;

case 2:

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

a[i]=i;

}

}

3. Провести эксперименты по определению временных характеристик алгоритмов поиска. Результаты экспериментов представить в виде таблиц 12 и 13. Клетки таблицы 12 содержат максимальное количество операций сравнения при выполнении алгоритма поиска, а клетки таблицы 13 — среднее число операций сравнения.

Таблица 12

**Максимальное количество операций сравнения**

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Алгоритмы поиска | Количество элементов в массиве | | | | | | | | |
| 50 | 100 | 150 | 200 | 250 | 300 | 350 | 400 | 450 |
| Линейный (неупорядоченный массив) | 50 | 100 | 150 | 200 | 250 | 300 | 350 | 400 | 450 |
| Быстрый линейный (неупорядоченный массив) | 51 | 101 | 151 | 201 | 251 | 301 | 351 | 401 | 451 |
| Быстрый линейный (упорядоченный массив) | 49 | 99 | 149 | 198 | 248 | 298 | 348 | 397 | 445 |
| Бинарный (упорядоченный массив) | 6 | 7 | 8 | 8 | 8 | 9 | 9 | 9 | 9 |
| Блочный (упорядоченный массив) | 13 | 18 | 22 | 26 | 29 | 32 | 35 | 36 | 40 |

Таблица 13

**Среднее количество операций сравнения**

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Алгоритмы поиска | Количество элементов в массиве | | | | | | | | |
| 50 | 100 | 150 | 200 | 250 | 300 | 350 | 400 | 450 |
| Линейный (неупорядоченный массив) | 27 | 53 | 74 | 104 | 125 | 152 | 178 | 212 | 259 |
| Быстрый линейный (неупорядоченный массив) | 28 | 56 | 78 | 106 | 125 | 159 | 174 | 207 | 275 |
| Быстрый линейный (упорядоченный массив) | 25 | 48 | 72 | 93 | 126 | 150 | 178 | 191 | 254 |
| Бинарный (упорядоченный массив) | 4 | 5 | 6 | 6 | 7 | 7 | 7 | 7 | 7 |
| Блочный (упорядоченный массив) | 7 | 10 | 12 | 13 | 15 | 17 | 18 | 20 | 20 |

4. Построить графики зависимости количества операций сравнения от количества элементов в массиве.

**Максимальное количество операций сравнения**

**Среднее количество операций сравнения**

5-6. Определить аналитическое выражение функции зависимости количества операций сравнения от количества элементов в массиве.

Определить порядок функций временной сложности алгоритмов поиска.

|  |  |  |
| --- | --- | --- |
| Поиск: | ФВС | Порядок ФВС |
| Линейный (неупорядоченный массив) | y = x | O(n) |
| Быстрый линейный (неупорядоченный массив) | y = 1,05x0,99 | O(n) |
| Быстрый линейный (упорядоченный массив) | y = 0,97x1,01 | O(n) |
| Бинарный (упорядоченный массив) | y = 2,94x0,19 | O(n0,2) |
| Блочный (упорядоченный массив) | y = 1,74x0,5095 | O(n0,5) |

|  |  |  |
| --- | --- | --- |
| Поиск: | ФВС | Порядок ФВС |
| Линейный (неупорядоченный массив) | y = 0,43x1,03 | O(n) |
| Быстрый линейный (неупорядоченный массив) | y = 0,58x0,99 | O(n) |
| Быстрый линейный (упорядоченный массив) | y = 0,5141x1,003 | O(n) |
| Бинарный (упорядоченный массив) | y = 1,49x0,27 | O(n0,3) |
| Блочный (упорядоченный массив) | y = 1,045x0,49 | O(n0,5) |