**Лабораторна робота № 9. Алгоритми простих пошуків**

***Мета:*** одержати навички та закріпити знання при виконанні операцій пошуку.

**1 Вимоги**

**1.1 Розробник**

* Макаренко Владислав Олександрович
* Студент 1-го курсу
* Групи КІТ-120а

**1.2 Загальне завдання**

Розробити та налагодити програму, в якій реалізувати два алгоритми пошуку у відповідності до завдання. Порівняти алгоритми за часом роботи.

**1.2 Загальне завдання**

Двійковий та лінійний пошуки у масиві.

**2 Описи програм**

**Код програми**

#include <iostream>  
#include <ctime>  
  
using namespace std;  
  
  
int LineSearch(int *arr*[], int *size*, int *key*); // линейный поиск  
  
void ShowArray(int *arr*[], int *size*); // показ массива  
  
int BinSearch(int *arr*[], int *left*, int *right*, int *key*); // бинарный поиск  
  
void SortMass(int\* *m*, int *n*);  
  
int main()  
{  
 srand(time(NULL));  
 setlocale(LC\_ALL, "rus");  
 const int Size = 100;  
 int Array[Size];  
 int Key = 0; // искомое значение (ключ)  
 int FindIndex; // номер элемента массива  
  
 srand(time(NULL));  
  
 //запись случ. чисел в массив от 1 до 50  
 for (int i = 0; i < Size; i++)  
 {  
 Array[i] = 1 + rand() % 444;  
 }  
 cout << "Масив випадкових чисел: " << endl;  
 ShowArray(Array, Size);  
  
 cout << "Введіть ключ пошуку - ціле число: ";  
 cin >> Key; // ввод искомого числа  
  
 clock\_t Start1 = clock();  
 //поиск искомого числа и запись номера элемента  
 FindIndex = LineSearch(Array, Size, Key);  
  
 clock\_t End1= clock();  
  
 double TimeSpent1 = (double)(End1 - Start1)/ ((double) CLOCKS\_PER\_SEC);  
  
  
 if(FindIndex != -1){  
 //если в массиве найдено искомое число - выводим индекс элемента на экран  
 cout << "Знайдено значення в елементі: " << FindIndex << endl;  
 }  
 else{  
 //если в массиве не найдено искомое число  
 cout << "В масиві нема такого значення" << endl;  
 }  
 printf( "Час лінійного пошуку: %.10f сек\n", TimeSpent1 );  
  
  
  
 //поиск искомого числа и запись номера элемента  
  
 SortMass(Array, Size);  
 cout << "Відсортований масив випадкових чисел: " << endl;  
 ShowArray(Array, Size);  
 clock\_t Start2 = clock();  
 FindIndex = BinSearch(Array, 0, Size, Key);  
  
 clock\_t End2= clock();  
  
 double TimeSpent2 = (double)(End2 - Start2)/ CLOCKS\_PER\_SEC;  
  
 if(FindIndex != -1){  
 //если в массиве найдено искомое число - выводим индекс элемента на экран  
 cout << "Знайдено значення в елементі: " << FindIndex << endl;  
 }  
 else {  
 //если в массиве не найдено искомое число  
 cout << "В масиві нема такого значення" << endl;  
 }  
  
 printf( "Час двійкового пошуку: %.10f сек\n", TimeSpent2);  
  
  
 return 0;  
}  
  
//вывод массива на экран  
void ShowArray(int *arr*[], int *size*)  
{  
 for (int i = 0; i < *size*; i++)  
 {  
 cout.width(5);  
 cout << arr[i];  
 if ((i + 1) % 10 == 0)  
 {  
 cout << endl;  
 }  
  
 }  
 cout << endl << endl;  
}  
  
//линейный поиск  
int LineSearch(int *arr*[], int *size*, int *key*){  
 for (int i = 0; i < *size*; i++){  
 if (*arr*[i] == *key*){  
 return i;  
 }  
 }  
 return -1;  
}  
  
int BinSearch(int *arr*[], int *left*, int *right*, int *key*) {  
 {  
 int midd = 0;  
 while (1) {  
 midd = (*left* + *right*) / 2;  
  
 if (*key* < *arr*[midd]) // если искомое меньше значения в ячейке  
 *right* = midd - 1; // смещаем правую границу поиска  
 else if (*key* > *arr*[midd]) // если искомое больше значения в ячейке  
 *left* = midd + 1; // смещаем левую границу поиска  
 else // иначе (значения равны)  
 return midd; // функция возвращает индекс ячейки  
  
 if (*left* > *right*) // если границы сомкнулись  
 return -1;  
 }  
 }  
}  
  
void SortMass(int\* *m*, int *n*)  
{  
 for(int i = *n* - 1; i >= 1; i--)  
 for(int j = 0; j < i; j++)  
 {  
 if(*m*[j] > *m*[j+1])  
 {  
 int foo = *m*[j];  
 *m*[j] = *m*[j+1];  
 *m*[j+1] = foo;  
 }  
 }  
}

**Результати виконання програми**

За алгоритмом коду демонструємо роботу програми для 100 елементів (див. рис. 1).

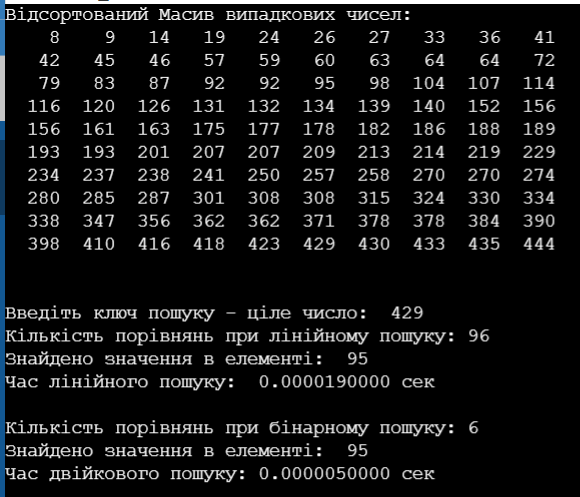


Рисунок 1 – Результати роботи першого кроку

Таблиця 1 – Результати тестування алгоритму бінарного пошуку

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Кількість елементів | 20 | 100 | 1000 | 10000 |
| Кількість порівнянь | 4 | 6 | 7 | 9 |
| Час пошуку | 0.000005 | 0.000005 | 0.000005 | 0.000009 |

Таблиця 2 – Результати тестування алгоритму лінійного пошуку

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Кількість елементів | 20 | 100 | 1000 | 10000 |
| Кількість порівнянь | 17 | 96 | 442 | 9978 |
| Час пошуку | 0.000022 | 0.000019 | 0.000022 | 0.000058 |

**Висновок:** на цій лабораторній роботі ми одержали навички та закріпили знання при виконанні операцій пошуку.