Звіт

По лабораторній роботі №4

Виконав:

Студент ФІТ 2-11

Забродський Олексій Русланович

**Завдання 1**

1. Побудувати блок-схему алгоритмів пошуку.

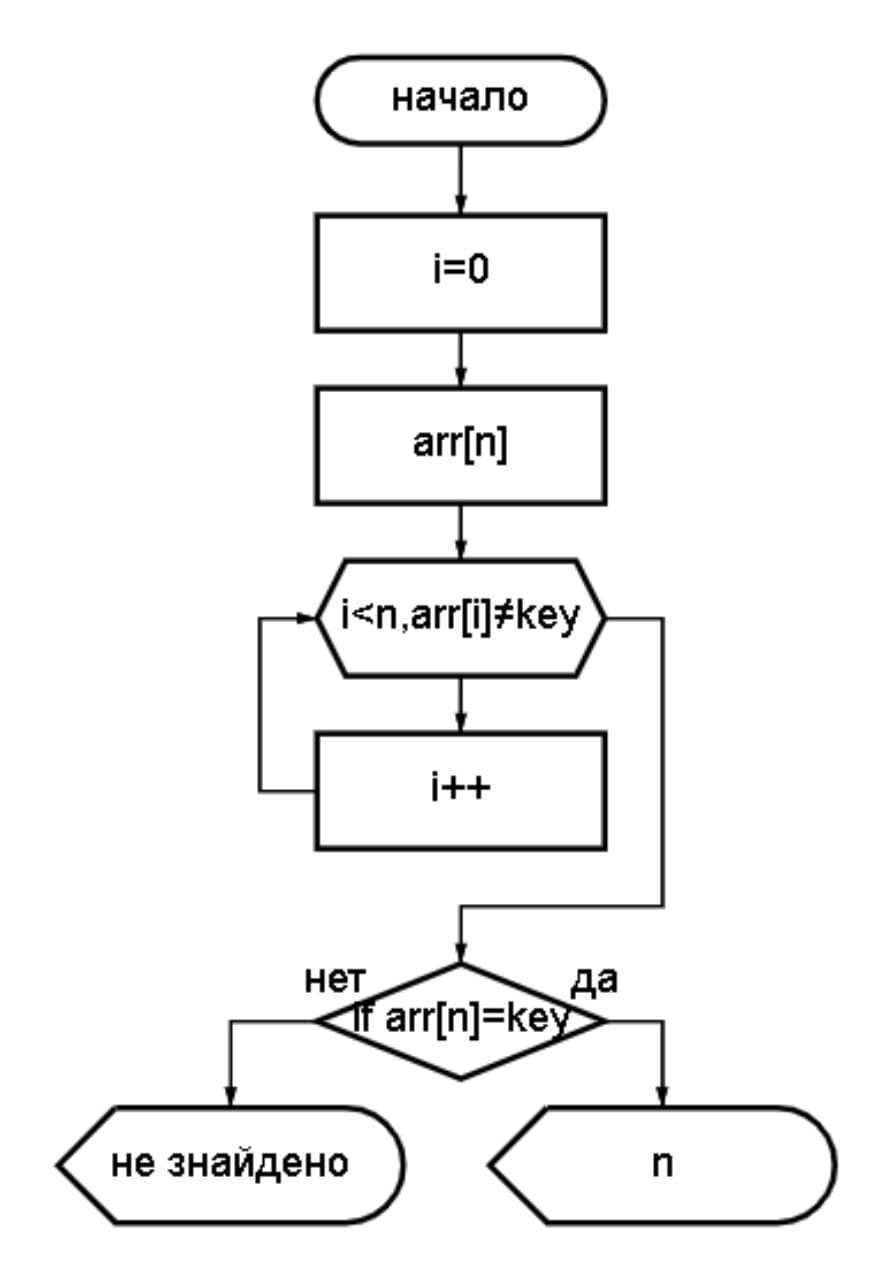
2. Виконати програмну реалізацію алгоритмів пошуку. Правильність роботи програм підтвердити скріншотами.

3. Згенерувати вихідні послідовності залежно від заданого розміру вибірки та зберегти у файлі.

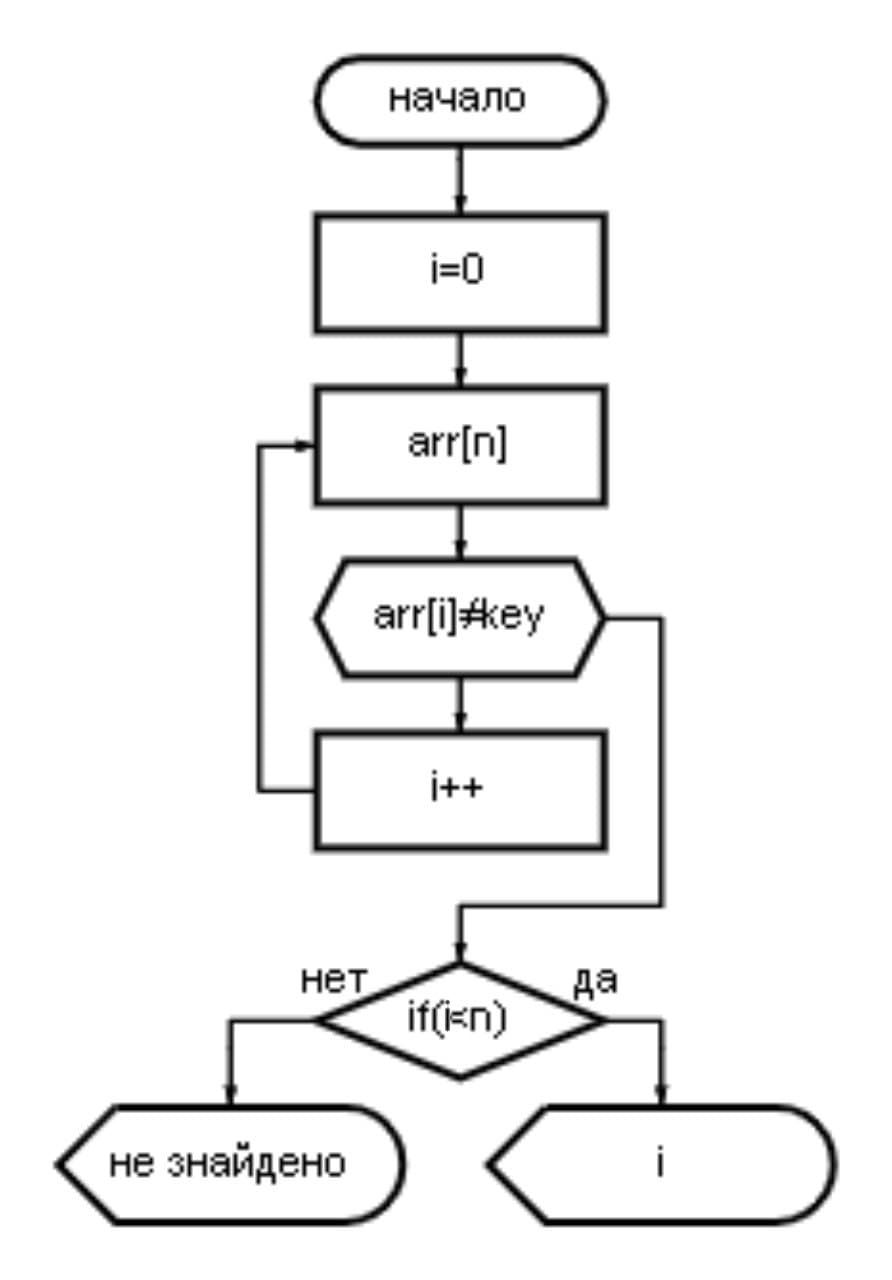
− Для генерації вихідної числової послідовності можна використовувати датчик випадкових чисел.

– Для чистоти експерименту дані мають бути з однієї згенерованої вибірки, а також повинен здійснюватися пошук одного й того самого елемента, наприклад останнього елемента вибірки чи середнього.

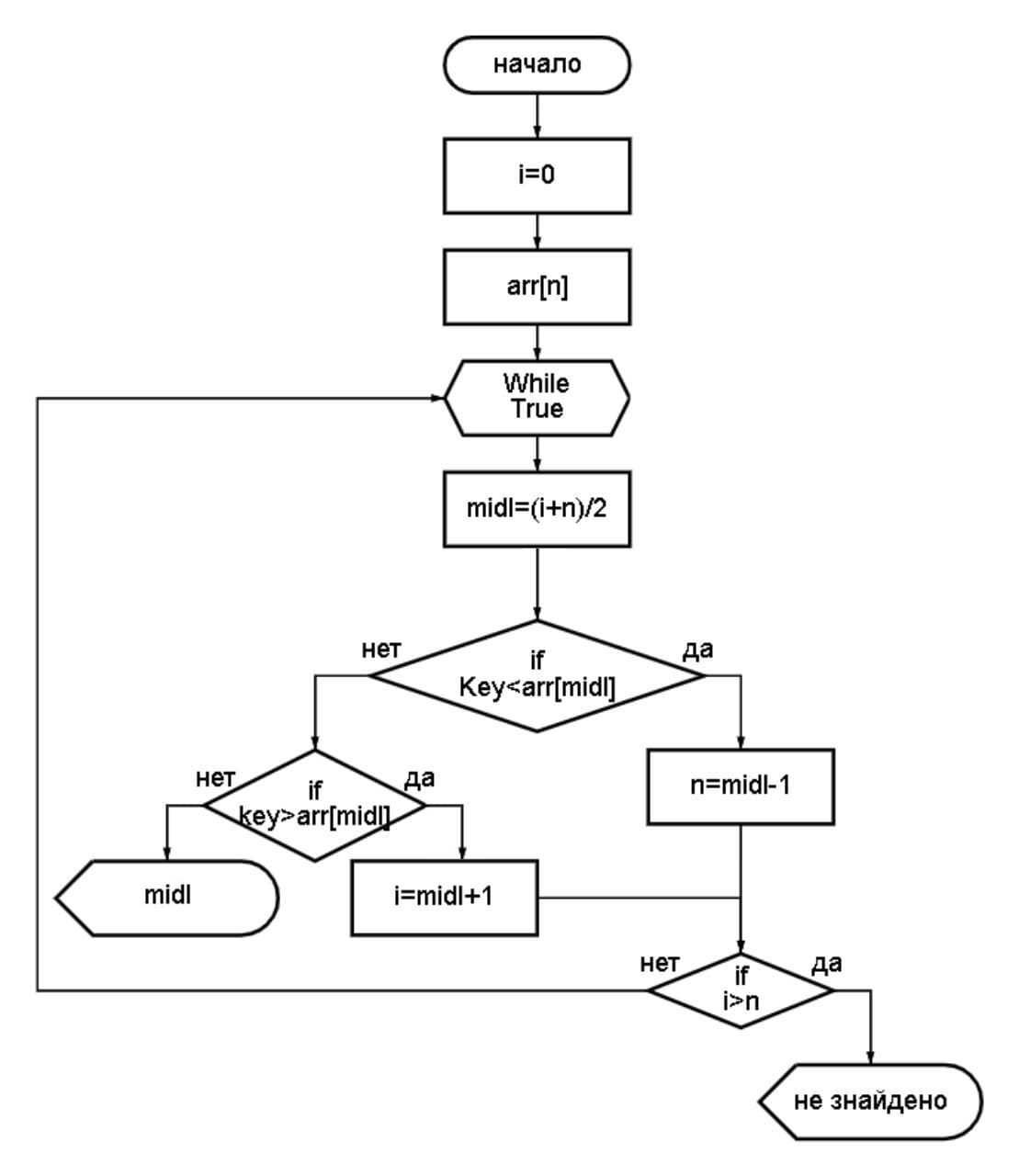
4. Виконати експерименти, під час яких визначити час роботи кожного алгоритму залежно від розміру вибірки.



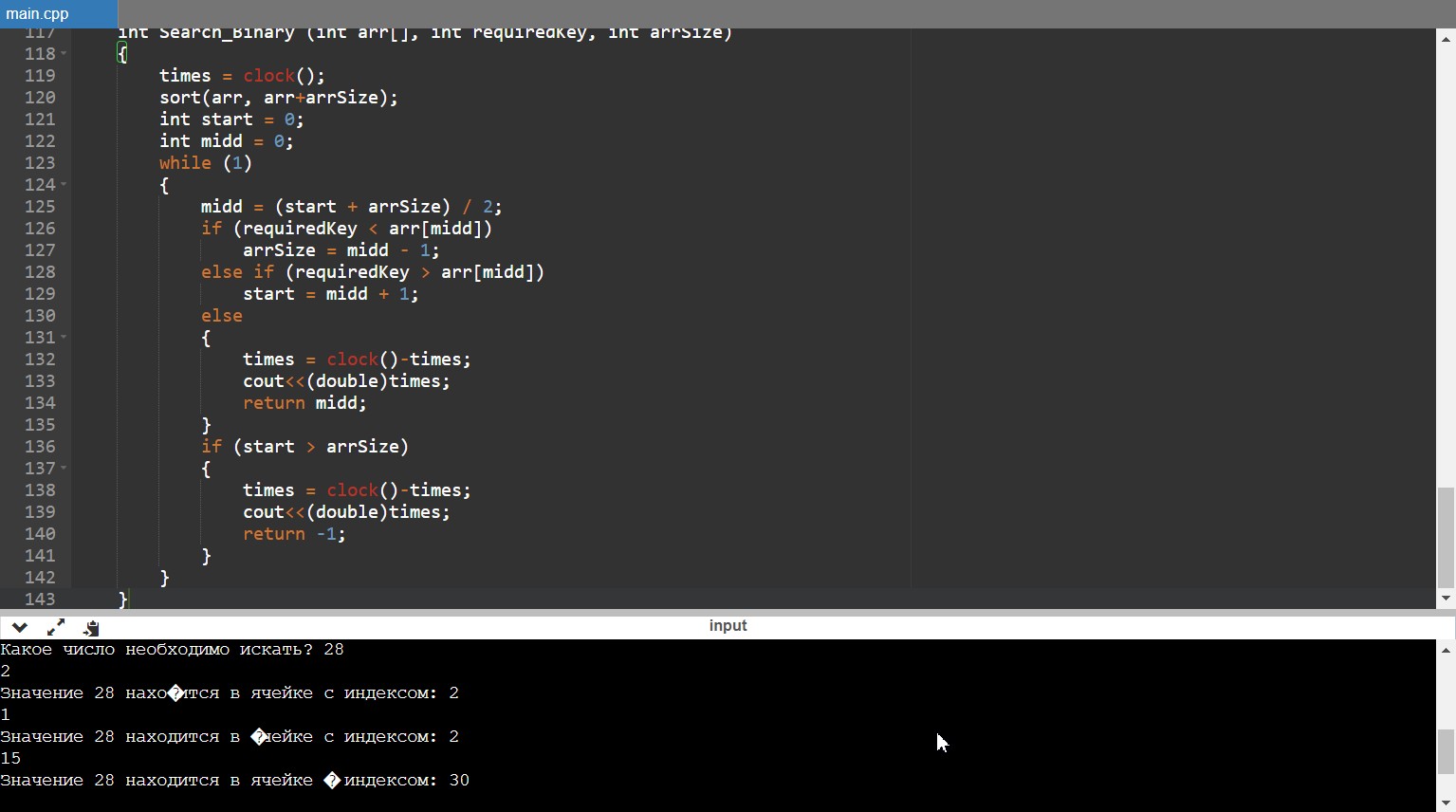
**Рис.1 Блок-схема послідовного пошуку**



**Рис.2 Блок-схема швидкого послідовного пошуку**



**Рис.3 Блок-схема бінарного пошуку**



**Лістинг програми:**

#include <iostream>

#include <iomanip>

#include <ctime>

#include <time.h>

#include <algorithm>

using namespace std;

int linSearch(int arr[], int requiredKey, int size); // линейный поиск

int linFastSearch(int arr[], int requiredKey, int size); // быстрый линейный поиск

int Search\_Binary (int arr[], int arrSize, int requiredKey); // двоичный поиск

void showArr(int arr[], int size); // показ массива

void showKey(int nElement, int requiredKey);

clock\_t times;

int main()

{

setlocale(LC\_ALL, "rus");

int arrSize;

cin >> arrSize;

int arr[arrSize];

int requiredKey = 0; // ключ

int nElement = 0; // номер элемента массива

srand(time(NULL));

//запись случайных чисел в массив от 1 до 99

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

{

arr[i] = 1 + rand() % 99;

}

showArr(arr, arrSize);

cout << "Какое число необходимо искать? ";

cin >> requiredKey;

//поиск числа

nElement = linSearch(arr, requiredKey, arrSize);

showKey(nElement, requiredKey);

nElement = linFastSearch(arr, requiredKey, arrSize);

showKey(nElement, requiredKey);

nElement = Search\_Binary(arr, requiredKey, arrSize);

showKey(nElement, requiredKey);

return 0;

}

void showKey(int nElement, int requiredKey)

{

if (nElement != -1)

{

cout << "\nЗначение " << requiredKey << " находится в ячейке с индексом: " << nElement << endl;

}

else

{

cout << "\nВ массиве нет такого значения" << endl;

}

}

//вывод массива на экран

void showArr(int arr[], int arrSize)

{

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

{

cout << setw(4) << arr[i];

if ((i + 1) % 10 == 0)

{

cout << endl;

}

}

cout << endl << endl;

}

//линейный поиск

int linSearch(int arr[], int requiredKey, int arrSize)

{

times = clock();

int w = 0;

while(w<arrSize && arr[w] != requiredKey)

w++;

if (arr[w] == requiredKey)

{

times = clock()-times;

cout<<(double)times;

return w;

}

else

{

times = clock()-times;

cout<<(float)times;

return -1;

}

}

//быстрый линейный поиск

int linFastSearch(int arr[], int requiredKey, int arrSize)

{

times = clock();

int q = 0;

while(arr[q] != requiredKey)

q++;

if (q < arrSize)

{

times = clock()-times;

cout<<(double)times;

return q;

}

else

{

times = clock()-times;

cout<<(double)times;

return -1;

}

}

// двоичный поиск

int Search\_Binary (int arr[], int requiredKey, int arrSize)

{

times = clock();

sort(arr, arr+arrSize);

int start = 0;

int midd = 0;

while (1)

{

midd = (start + arrSize) / 2;

if (requiredKey < arr[midd])

arrSize = midd - 1;

else if (requiredKey > arr[midd])

start = midd + 1;

else

{

times = clock()-times;

cout<<(double)times;

return midd;

}

if (start > arrSize)

{

times = clock()-times;

cout<<(double)times;

return -1;

}

}

}

|  |  |  |  |
| --- | --- | --- | --- |
| Обсяг вибірки,  N | Час | | |
| При послідовному пошуку | При швидкому послідовному пошуку | При бінарному пошуку |
| 100 | 1 | 1 | 12 |
| 500 | 2 | 1 | 71 |
| 1000 | 2 | 1 | 129 |
| 2500 | 2 | 1 | 327 |
| 5000 | 2 | 1 | 734 |
| 10000 | 2 | 1 | 1303 |
| 50000 | 2 | 1 | 7797 |
| 100000 | 3 | 1 | 24060 |

**Табл.1. Результати вдалого пошуку**

|  |  |  |  |
| --- | --- | --- | --- |
| Обсяг вибірки,  N | Час | | |
| При послідовному пошуку | При швидкому послідовному пошуку | При бінарному пошуку |
| 100 | 2 | 1 | 13 |
| 500 | 3 | 2 | 63 |
| 1000 | 5 | 4 | 171 |
| 2500 | 10 | 8 | 404 |
| 5000 | 18 | 15 | 766 |
| 10000 | 33 | 35 | 1648 |
| 50000 | 123 | 115 | 6916 |
| 100000 | 359 | 280 | 18725 |

**Табл.2. Результати невдалого пошуку**

**Висновок**

Виконуючи лабораторну роботу я виконав аналіз емпіричної ефективності (практичної складності) методів пошуку: послідовного, швидкого послідовного, бінарного. Ознайомився з теоретичними відомостями та матеріалами лекцій.