

Алгоритми та структури даних

**Лабораторна робота №3**

**“Дослідження елементарних алгоритмів сортування та пошуку”**

Робота : Лещинського Б.Д. группа КА-07

24.11.2020

Варіант 14

**Мета роботи** :

Ознайомитись і дослідити елементарні алгоритми сортування: “бульбашкою”, вибором, вставками; алгоритм перетасовки Фішера-Йєтса та алгоритми пошуку: лінійний пошук, бінарний пошук. Набути навичок їх реалізації мовою програмування С/C++ та порівняти ці алгоритми.

**Хід виконання роботи:**

**Задача №1**

**Умова:**

1. Базове сортування “бульбашкою”, пошук елементів у масиві

1.1 Створити масив на N = 1000 елементів та заповнити його випадковими цілими числами від -2000 до 2000

1.2 Написати функцію для сортування алгоритмом “бульбашкою”, відсортувати масив за зростанням

1.3 Написати функцію для пошуку елементу в масиві лінійним алгоритмом, яка повертає індекс знайденого елементу або -1, якщо такий елемент не знайдено

1.4 Написати функцію для пошуку елементу в масиві бінарним алгоритмом, яка повертає індекс знайденого елементу або -1, якщо такий елемент не знайдено

1.5 Порівняти швидкодію алгоритмів пошуку (пошук у масиві випадкових чисел від -2000 до 2000)

**Код програми**:

#include<iostream>

#include<cstdlib>

#include<ctime>

#include<cmath>

using namespace std;

int linearSearch(int sArr[], int size, int n){

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

if(n==sArr[i]){

return i;

}

}

return -1;

}

int binarySearch(int sArr[], int size, int n){

int mid;

int low = 0;

int high = size-1;

while(low<=high){

mid = (high + low)/2;

if(n > sArr[mid]){

low = mid + 1;

}

else{

if(n < sArr[mid]){

high = mid -1;

}

else{

return mid; // found

}

}

}

return -1; // not found

}

int main(){

srand(time(0));

const int N = 1000;

int arr[N];

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

arr[i]=rand()%4001 - 2000; //generating a random array in range [-2000;2000]

cout<<arr[i]<<'\t';

}

cout<<endl;

for(int j=0; j<(N-1);j++){ // sorting array using bubble sort

for(int i=0; i<(N-1); i++){

int swap = 0;

if(arr[i]>arr[i+1]){

swap = arr[i];

arr[i] = arr[i+1];

arr[i+1] = swap;

}

}

}

cout<<"The sorted array is: \n"; //printing out the sorted array

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

cout<<arr[i]<<'\t';

}

cout<<"\n";

int elemS;

cout<<"Element search(Notice: if no such found -1) \nEnter the element you wanna find the index of: ";

cin>>elemS;

clock\_t start1 = clock();

for(int i=0;i<100000;i++){

linearSearch(arr,N,elemS);

}

cout<<"The element`s index is: "<<linearSearch(arr,N,elemS)<<endl; // performing element`s index search using linear search algorythm

clock\_t end1 = clock();

float seconds1 = (float(end1- start1));

cout<<"Execution time of linear search: "<<seconds1<<endl;

clock\_t start2 = clock();

for(int i=0;i<100000;i++){

binarySearch(arr,N,elemS);

}

cout<<"The element`s index is: "<<binarySearch(arr,N,elemS)<<endl; // performing element`s index search using binary search algorythm

clock\_t end2 = clock();

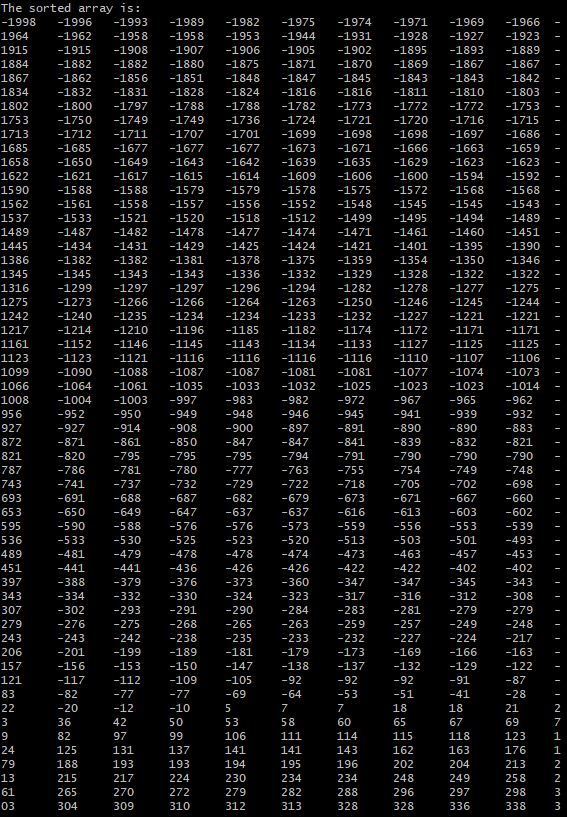
float seconds2 = (float(end2- start2));

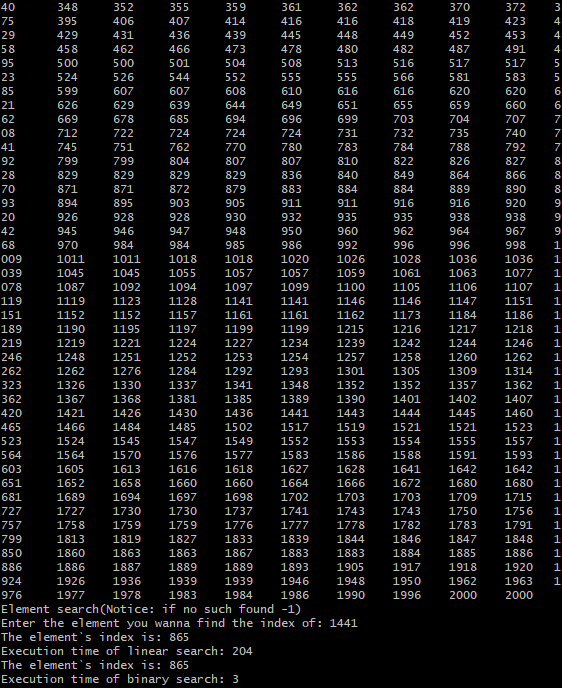
cout<<"Execution time of binary search: "<<seconds2;

return 0;

}

**Вивід програми**:





Отже, як можна побачити з виводу консолі - бінарний пошук у цьому випадку виявився у 68 разів швидше( час виконання вказується у мілісекундах).

**Задача №2**

**Умова:**

2. Сортування елементарними алгоритмами по різним критеріям

2.1 Створити масив на N = 200 елементів та заповнити його послідовними числами від 1 до N

2.2 Написати функцію для перетасовки масиву алгоритмом ФішераЙєтса, перетасувати масив

2.3 Написати функцію для сортування алгоритмом вибором / вставками, відсортувати масив за певним критерієм (див. свій варіант)

Варіант 14 Завдання 2.3. Сортування вибором за критерієм суми цифр числа у порядку зростання (напр. 61 < 53, адже 7 < 8), при рівності - спочатку менше за значенням.

**Код програми**:

#include<iostream>

#include<cstdlib>

#include<ctime>

#include<cmath>

using namespace std;

void FisherYetsAlgor(int arr[], int size){

srand(time(0));

for(int n=size-1; n>0; n--){

int swap;

int indx=rand()%(n+1);

swap=arr[indx];

arr[indx]=arr[n];

arr[n]=swap;

}

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

cout<<arr[i]<<"\t";

}

}

int equal(int x, int y){

int sumX = 0;

int sumY = 0;

while(x!=0){

sumX+=x%10;

x/=10;

}

while(y!=0){

sumY+=y%10;

y/=10;

}

return (sumX>sumY)+2\*(sumX<sumY);

}

void selectionSort(int arr[], int size){

for(int i=0; i<size-1; i++){

int inf = i;

for(int j=i+1; j<size; j++){

if(equal(arr[j],arr[inf])==2){

inf = j;

}

if(equal( arr[j],arr[inf])==0 && arr[j]<arr[inf] ){

inf = j;

}

}

int swap;

swap = arr[i];

arr[i] = arr[inf];

arr[inf] = swap;

}

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

cout<<arr[i]<<"\t";

}

}

int main(){

const int N = 200;

int arr[N];

cout<<"The initial array: \n";

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

arr[i]=1+i;

cout<<arr[i]<<"\t";

}

cout<<endl;

cout<<endl;

cout<<"The array mixed using Fisher-Yates algorithm: \n";

FisherYetsAlgor(arr,N);

cout<<endl;

cout<<endl;

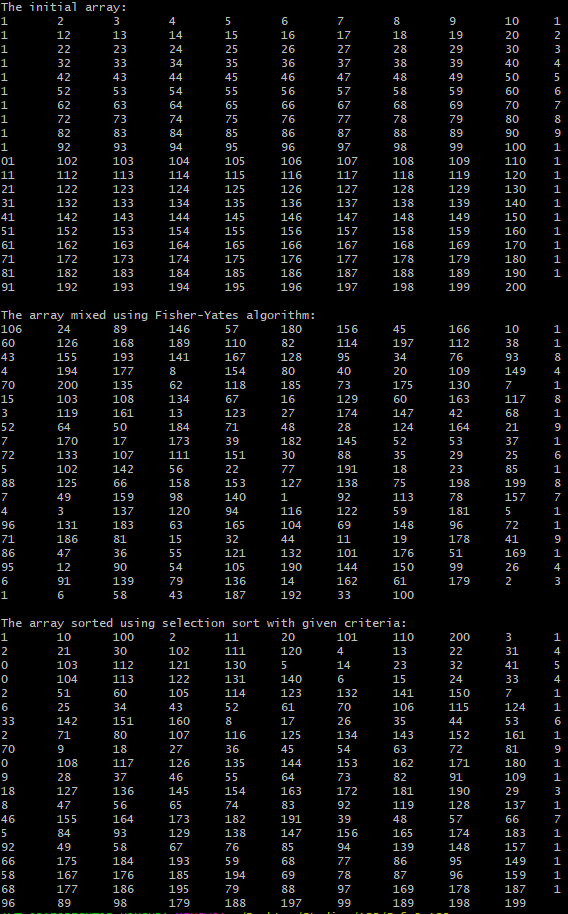
cout<<"The array sorted using selection sort with given criteria: \n";

selectionSort(arr,N);

return 0;

}

**Вивід програми**:



**Висновок:**

Я ознайомився та дослідив натупні базові алгоритми: “бульбашкою”, вибором, алгоритм перетасовки Фішера-Йєтса та алгоритми пошуку: лінійний пошук, бінарний пошук. Я набув навичок їх реалізації мовою С++ та в порівнянні встановив,що бінарний пошук набагото швидше відповідного лінійного.