**Лабораторная работа №3**

**Тема**: Алгоритмы сортировки массивов

**Цель**: Научиться реализовывать простые алгоритмы сортировки выбором, вставкой и пузырьком

Задачей сортировки является превращение исходной последовательности в последовательность, содержащая те же записи, но в порядке возрастания (или убывания) значений элементов.

**Сортировка выбором.** Этот метод основан на следующем правиле. Выбирается минимальный (максимальный) элемент последовательности и обменивается с первым элементом последовательности. Очевидно, один элемент при этом станет на свое место в отсортированной части последовательности. Далее все выше сказанное надо повторить в неотсортированные части последовательности и т.д.

FOR I = 1 TO N-1 DO

 BEGIN

 <Присвоить K индекс наименьшего элемента с a [I] ... a [N]>

 <Поменять местами a [I] и a [K]>

 END

**Сортировка вставками**. Этот метод заключается в том, что на каждом шагу берут i-й элемент последовательности и передают его в готовую отсортированную часть последовательности, вставляя его на свое место. Алгоритм сортировки включениями выглядит следующим образом:

FOR I = 2 TO N DO

    BEGIN X = a [I];

       <Вставить X на подходящее место в a [1], a [2], ..., a [I]>

    END

**СОРТИРОВКА обмен (пузырьковая).** Алгоритм обмена основан на принципе сравнения и обмена пары соседних элементов, пока не будут отсортированы все элементы.

FOR I = 2 TO N DO

FOR J = N DOWNTO I DO

IF a [J-1]> a [J] THEN

BEGIN X = a [J-1]; a [J-1] = a [J]; a [J] = X END

Задача: изучить изучены алгоритмы сортировки следующим образом:

1. Описать целочисленный массив, состоящий из N элементов (N - константа).

2. Значение элементов массива задать случайным образом. Вывести полученный массив на экран.

3. Для созданного массива реализовать различные алгоритмы сортировки ( "пузырьковая", вставками, выбору), при этом определяя количество проходов массива, сравнений и перестановок.

дополнительное задание

4. Определить время, затраченное на каждый из способов. Внимание: для определения "чистого" времени сортировки, выполните его в другой раз без использования элементов исследования (подсчета проходов и др.).

Способ определения времени работы процедуры в Delphi:

var

d1, d2: TDateTime;

d1 = Now;

<Процедура сортировки>

d2 = Now;

время выполнения = d2 - d1.

Также можно использовать счетчик GetTickCount или QueryPerformanceCounter.

5. Сравнить методы сортировки при N = 10, 100, 1000, 10000, 30000. Результатом исследования должна стать таблица (в отчете):

N Количество проходов Количество сравнений Количество перестановок Время

"Пузырьковая" вставками выбором "пузырьковая вставками выбором" пузырьковая вставками выбором пузырек вставками выбором

|  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| N | Кількість проходів | | | Кількість порівнянь | | | Кількість перестановок | | | Час | | |
| "бульбашкова" | вставками | вибором | "бульбашкова | вставками | вибором | "бульбашкова | вставками | вибором | пухирець | вставками | вибором |
| 10 |  |  |  |  |  |  |  |  |  |  |  |  |
| 100 |  |  |  |  |  |  |  |  |  |  |  |  |
| 1000 |  |  |  |  |  |  |  |  |  |  |  |  |
| 10000 |  |  |  |  |  |  |  |  |  |  |  |  |
| 30000 |  |  |  |  |  |  |  |  |  |  |  |  |

Указания: 1) чтобы в исследовании использовался тот же массив, сохраните его в дополнительном массиве, и выполняйте операцию копирования массива для подготовки к очередному сортировки;

2) с целью проверки после каждого способа выводите отсортированный массив на экран.

**Лабораторная работа №3**

**Тема**: Алгоритмы сортировки массивов

**Цель**: Научиться реализовывать простые алгоритмы сортировки выбором, вставкой и пузырьком

Задача: изучить изучены алгоритмы сортировки следующим образом:

1. Описать целочисленный массив, состоящий из N элементов (N - константа).

2. Значение элементов массива задать случайным образом. Вывести полученный массив на экран.

3. Для созданного массива реализовать различные алгоритмы сортировки ( "пузырьковая", вставками, выбору), при этом определяя количество проходов массива, сравнений и перестановок.

4. Определить время, затраченное на каждый из способов. Внимание: для определения "чистого" времени сортировки, выполните его в другой раз без использования элементов исследования (подсчета проходов и др.).

5. Сравнить методы сортировки при N = 10, 100, 1000, 10000, 30000. Результатом исследования должна стать таблица (в отчете):

// Листинг программы Prog\_Lab\_3\_1

#include <iostream>

#include <ctime>

using namespace std;

int main()

{

srand(time(0)); // установка автоматической рандомизации

// ввод кол-ва элементов в массиве

int element\_count = 0, counter; // количество элементов в массиве

cout << "Введите количество элементов в массиве : ";

cin >> element\_count;

counter = element\_count;

int item; // индекс предыдущего элемента

int temp;

int array\_for\_sort[element\_count]; // основной массив

int copy\_array[element\_count]; // массив для копирования array\_for\_sort

clock\_t time;

int col\_prohodov, col\_sravneniy, col\_perestanovok;

// заполнение массива случайными числами

for (int counter = 0; counter < element\_count; counter++)

array\_for\_sort[counter] = 1+rand()%100;

// копирование массива

for (int counter = 0; counter < element\_count; counter++)

copy\_array[counter] = array\_for\_sort[counter];

/\*

// отображение основного массива и скопированного

for (int counter = 0; counter < element\_count; counter++)

cout << counter+1<<" = "<< array\_for\_sort[counter]<< " - " << copy\_array[counter] <<endl;

\*/

/\*

// Cортировка выбором

time = clock();

for (int i = 0; i < element\_count; i++){

int min = i;

for (int j = i; j < element\_count-1; j++){

if (copy\_array[min] > copy\_array[j+1]){

min = j+1;

}

}

temp = copy\_array[i];

copy\_array[i] = copy\_array[min];

copy\_array[min] = temp;

}

time = clock()-time;

\*/

// отображение массива отсортированного методом выбора

cout << "\nСортировка выбором \n\n";

for (int counter = 0; counter < element\_count; counter++)

cout << copy\_array[counter]<< ", ";

cout << "\nВремя сортировки" << time << endl;

/\*

// Cортировка вставкою

temp=0;

// time =0;

// time = clock();

for (int i = 1; i < element\_count; i++) {

temp = copy\_array[i]; // инициализируем временную переменную текущим значением элемента массива

item = i-1; // запоминаем индекс предыдущего элемента массива

while(item >= 0 && copy\_array[item] > temp) // пока индекс не равен 0 и предыдущий элемент массива больше текущего

{

copy\_array[item + 1] = copy\_array[item]; // перестановка элементов массива

copy\_array[item] = temp;

item--;

} }

// time = clock()-time;

// отображение массива отсортированного методом вставкою

cout << "\nСортировка выбором \n\n";

for (int counter = 0; counter < element\_count; counter++)

cout << copy\_array[counter];

// cout << "Время сортировки" << time << endl;

\*/

/\*

// сортировка пузырьком

time =0;

time = clock();

for (int j=0; j<element\_count-1; j++) {

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

if(copy\_array[i]>copy\_array[i+1]){

temp=copy\_array[i+1];

copy\_array[i+1]= copy\_array[i];

copy\_array[i]= temp;}

}

counter--;

}

time = clock()-time;

// отображение массива отсортированного методом пузырька

cout << "\nСортировка пузырьком \n\n";

for (int counter = 0; counter < element\_count; counter++)

cout << copy\_array[counter];

cout << "Время сортировки" << time << endl;

\*/

return 0;

}

|  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| N | Кількість проходів | | | Кількість порівнянь | | | Кількість перестановок | | | Час | | |
| "бульбашкова" | вставками | вибором | "бульбашкова | вставками | вибором | "бульбашкова | вставками | вибором | пухирець | вставками | вибором |
| 10 | 82 | 38 | 55 | 28 | 29 | 14 | 84 | 58 | 30 | 0 | 0 | 0 |
| 100 | 7311 | 2508 | 5050 | 2262 | 2409 | 287 | 6786 | 4818 | 300 | 0 | 0 | 0 |
| 1000 | 744990 | 242566 | 500500 | 244491 | 241567 | 3983 | 733473 | 483134 | 3000 | 16 | 0 | 0 |
| 10000 | 74885807 | 24840239 | 50005000 | 24880808 | 24830240 | 40785 | 74642424 | 49660480 | 30000 | 265 | 93 | 109 |
| 30000 | 673590872 | 223273597 | 450015000 | 223575873 | 223243598 | 125030 | 670727619 | 446487196 | 90000 | 2700 | 685 | 909 |