## Лабораторная работа №10

## Методы сортировки

***Цель лабораторной работы:*** *изучение методов сортировки структур данных; совершенствование навыков процедурного программирования на языке C/С++ при решении задач обработки статических и динамических массивов.*

***Задание на программирование:*** *используя технологию процедурного программирования реализовать заданный метод сортировки и применить его для указанных фрагментов числовой матрицы в соответствии с индивидуальным заданием.*

***Порядок выполнения работы:***

1) Получить у преподавателя индивидуальное задание: метод сортировки и вид сортируемых фрагментов матрицы. Исходная матрица, содержащая 2\*n строк и 2\*n столбцов, не должна содержать элементов с одинаковыми и нулевыми значениями. Значения элементов матрицы необходимо формировать программно (с клавиатуры не вводить) с помощью формул.

2) Разработать математическую модель: описать с помощью формул и рисунков структуру массива и процесс его преобразования. У результирующей матрицы должны быть отсортированы заданные фрагменты, а значения элементов не сортируемых фрагментов должны быть обнулены.

3) Построить схему алгоритма решения задачи.

4) Составить программу на языке C/С++.

5) Входные данные на этапах тестирования и демонстрации работы преподавателю должны задаваться либо с использованием специально подобранных арифметических формул, либо вводиться с клавиатуры по запросу. **Датчики псевдослучайных чисел использовать запрещается**.

Выходные данные должны выводиться на экран с пояснениями.

6) Проверить и продемонстрировать преподавателю работу программы на полном наборе тестов, в том числе с ошибочными входными данными. Входные и выходные массивы должны выводиться в одном и том же формате.

7) Использовать стандартные потоковые объекты ввода/вывода ***cin*** и ***cout***.

8) Оформить отчет о лабораторной работе в составе: постановка задачи, математическая модель, схема алгоритма решения, текст программы, контрольные примеры.

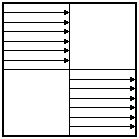
***Варианты индивидуальных заданий***

***Методы сортировки***

**1.** Сортировка по возрастанию методом выбора минимума.

***Области сортировки элементов матриц***

**1.**



***Примеры программ***

**Пример №1**

//Методы сортировки массивов данных.

//Массивы одномерные статические.

#include<iostream.h>

#include <locale.h>

//using namespace std;

const int RAZ = 10 ; //размер массива

typedef int telem ; //определение типа элементов массива

typedef telem tmas[RAZ] ; //определение типа массива

void outputmas(tmas a, int n) ; //вывод значений элементов массива

void sort\_obmen(tmas a, int n) ; //сортировка методом обмена

void sort\_obmenF(tmas a, int n) ; //сортировка методом обмена с флагом

void sort\_vibor(tmas a, int n) ; //сортировка методом выбора

void sort\_vstav(tmas a, int n) ; //сортировка методом вставки

void sort\_quick(tmas a, int t, int n) ; //метод быстрой сортировки

int main()

{tmas mas\_a = {10,9,8,7,6,5,4,3,2,1} ; //исходный массив а

tmas mas\_b = {19,17,15,13,11,9,7,5,3,1} ; //исходный массив b

tmas mas\_c = {20,18,16,14,12,10,8,6,4,2} ; //исходный массив c

tmas mas\_d = {1,10,2,9,3,8,4,7,5,6} ; //исходный массив e

tmas mas\_e = {2,9,4,10,7,1,6,5,3,8} ; //исходный массив f

int n;

setlocale(LC\_ALL,"Russian") ;

cout << " Примеры методов сортировки массивов целых чисел" << endl ;

cout << " Исходный массив а: " << endl ;

outputmas(mas\_a, RAZ) ;

cout << "\n Отсортированный массив а (метод обмена):" << endl ;

sort\_obmen(mas\_a, RAZ) ;

outputmas(mas\_a, RAZ) ;

cout << "\n Исходный массив b: " << endl ;

outputmas(mas\_b, RAZ) ;

cout<<"\n Отсортированный массив b (метод обмена с флагом):" <<endl ;

sort\_obmenF(mas\_b, RAZ) ;

outputmas(mas\_b, RAZ) ;

cout << "\n Исходный массив c: " << endl ;

outputmas(mas\_c, RAZ) ;

cout<<"\n Отсортированный массив c (метод выбора минимума):" <<endl ;

sort\_vibor(mas\_c, RAZ) ;

outputmas(mas\_c, RAZ) ;

cout << "\n Исходный массив d: " << endl ;

outputmas(mas\_d, RAZ) ;

cout << "\n Отсортированный массив d (метод вставки):" << endl ;

sort\_vstav(mas\_d, RAZ) ;

outputmas(mas\_d, RAZ) ;

cout << "\n Исходный массив e: " << endl ;

outputmas(mas\_e, RAZ) ;

cout<<"\nОтсортированный массив e (метод быстрой сортировки):"<<endl;

sort\_quick(mas\_e, 0, RAZ - 1) ;

outputmas(mas\_e, RAZ) ;

return 0 ;

}

//сортировка элементов массива по возрастанию методом обмена без флага

void sort\_obmen(tmas a, int n)

{ int i, k ;

telem z ;

for(k = n ; k > 0 ; k--)

for(i = 0 ; i < k - 1 ; i++)

if(a[i] > a[i + 1])

{z = a[i] ;

a[i] = a[i + 1] ;

a[i + 1] = z ;

}

}

//сортировка элементов массива по возрастанию методом обмена с флагом

void sort\_obmenF(tmas a, int n)

{ int i, k, flag ;

telem z ;

k = n ; //начальное количество не отсортированных элементов

do{flag = 0 ;

for(i = 0 ; i < k - 1 ; i++)

if(a[i] > a[i + 1])

{z = a[i] ;

a[i] = a[i + 1];

a[i + 1] = z ;

flag = 1 ;

}

k-- ;

}while(flag);

}

//сортировка элементов массива по возрастанию методом выбора минимума

void sort\_vibor(tmas a, int n)

{ int i, k, imin ;

telem z ;

for(i = 0 ; i < n - 1 ; i++)

{imin = i ; //поиск очередного минимума

for(k = i + 1 ; k < n ; k++)

if(a[k] < a[imin]) imin = k ;

//перестановка элементов

z = a[i] ;

a[i] = a[imin] ;

a[imin] = z ;

}

}

//сортировка элементов массива по возрастанию методом вставки

void sort\_vstav(tmas a, int n)

{ int i, j, k ;

telem z ;

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

{z = a[i] ; //выделение текущего элемента

j = 0 ;

//поиск места вставки

while(j < i && a[j] <= a[i]) j++ ;

for(k = i - 1 ; k >= j ; k--)

a[k + 1] = a[k] ; //сдвиг элементов

a[j] = z ;

}

}

//сортировка элементов массива по возрастанию методом быстрой сортировки

void sort\_quick(tmas a, int t, int n)

{ int i = t, //левая граница сортируемого фрагмента

j = n ; //правая граница сортируемого фрагмента

telem x, y ;

x = a[(t + n) / 2] ;

do{

while(a[i] < x) i++ ;

while(x < a[j]) j-- ;

if(i <= j)

{y = a[i] ;

a[i] = a[j] ;

a[j] = y ;

i++ ;

j-- ;

}

}while(i < j);

if(t < j) sort\_quick(a, t, j) ;

if(i < n) sort\_quick(a, i, n) ;

}

void outputmas(tmas a, int n)

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

cout << a[i] << ' ' ;

}

**Пример №2**

//Методы сортировки

//Массивы динамические двумерные

//Сортировка элементов всех строк двумерного массива целых чисел

//по убыванию

#include<iostream.h>

#include <locale.h>

//using namespace std;

typedef int telem ; //определение типа элементов массива

typedef telem \*tstr ; //определение типа "указатель на telem"

typedef tstr \*tmatr; //определение типа "указатель на указатель на telem"

void inputmatr(tmatr a, int str, int sto) ;

void outputmatr(tmatr a, int str, int sto) ;

void sort\_obmenF(tmatr a, int str, int sto) ;

int main()

{ int i, j ;

int n ; //число строк матрицы

int m ; //число столбцов матрицы

tmatr a; //объявляется переменная-указатель на указатель на telem

setlocale(LC\_ALL,"Russian") ;

cout << "\nВведите количество строк и столбцов матрицы: " ;

cin >> n >> m;

a = new tstr [n] ; //выделение динамической памяти под массив

//указателей на строки массива

for(i = 0; i < n; i++) //цикл выделения памяти под каждую строку

\*(a + i) = new telem [m] ;//каждому элементу массива указателей

//на строки присваивается адрес начала

//области памяти, выделяемой под строку

inputmatr(a, n, m) ;

cout << "Исходная матрица:\n" ;

outputmatr(a, n, m) ;

sort\_obmenF(a, n, m) ;

cout << "Матрица с отсортированными строками:" << endl ;

outputmatr(a, n, m) ;

cout << endl ;

//Освобождение динамической памяти

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

delete a[i] ;

delete a ;

return 0 ;

}

void inputmatr(tmatr a, int str, int sto)

{ cout << "Введите построчно через пробел элементы" << endl ;

cout << "двумерного массива размера " << str << "x" << sto << endl ;

cout << "После ввода строки нажимайте <Enter>" << endl ;

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

for(int j = 0 ; j < sto ; j++)

\*(\*(a + i) + j) = 10 \* i + j;

}

//сортировка элементов массива по возрастанию методом обмена с флагом

void sort\_obmenF(tmatr a, int str, int sto)

{ int i, j, k, flag ;

telem z ;

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

{k = sto ; //начальное количество не сортированных элементов строки

do{flag = 0 ;

for(j = 0 ; j < k - 1 ; j++)

if(\*(\*(a + i) + j) < \*(\*(a + i) + j + 1))

{z = \*(\*(a + i) + j) ;

\*(\*(a + i) + j) = \*(\*(a + i) + j + 1);

\*(\*(a + i) + j + 1) = z ;

flag = 1 ;

}

k-- ;

}while(flag);

}

}

void outputmatr(tmatr a, int str, int sto)

{ int i, j ;

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

{for(j = 0; j < sto; j++)

{cout.width(2) ; //ширина поля выводимого параметра

cout << \*(\*(a + i) + j) << ' ' ;

}

cout << '\n';

}

}