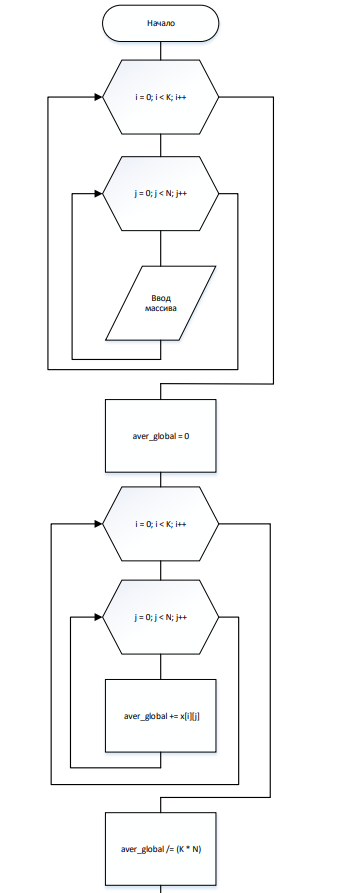
**Лабораторная работа №6. Двумерные массивы**

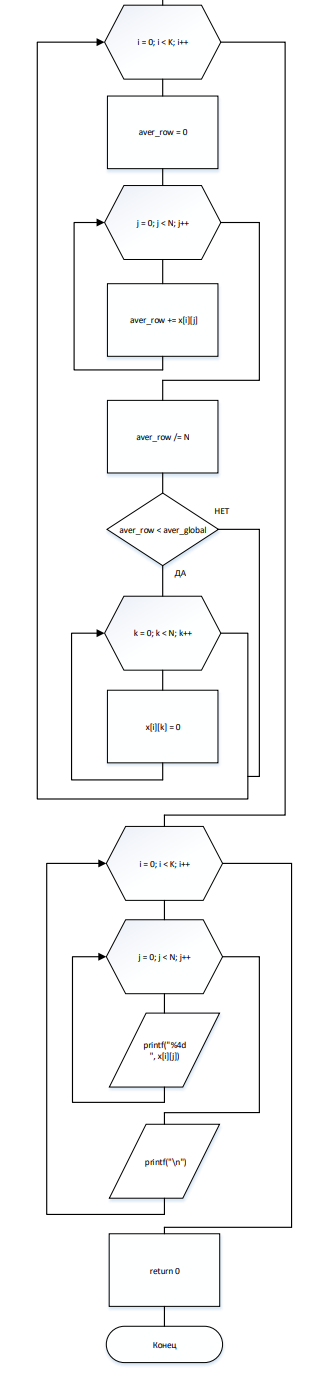
Цель работы: получение навыков работы с многомерными массивами на примере двумерных массивов.

Задача 6.12

В двумерном целочисленном массиве размера N на K обнулить строки, среднее арифметическое которых меньше среднего арифметического по всему массиву

Блок-схема





Исходный код программы

#include <stdio.h>

#define K 3 // Определение количества строк

#define N 4 // Определение количества столбцов

int main(void) {

int x[K][N]; // Объявление двумерного массива размером K на N

int aver\_global; // Переменная для хранения глобального среднего арифметического

int aver\_row; // Переменная для хранения среднего арифметического каждой строки

int i, j, k; // Объявление переменных-счетчиков

// Цикл для ввода элементов массива

for (i = 0; i < K; i++) {

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

scanf("%d", &x[i][j]); // Ввод элемента массива

}

}

// Вычисление глобального среднего арифметического значения

aver\_global = 0;

for (i = 0; i < K; i++) {

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

aver\_global += x[i][j]; // Суммирование всех элементов массива

}

}

aver\_global /= (K \* N); // Расчет среднего арифметического

// Цикл для обнуления строк, среднее арифметическое которых меньше глобального

for (i = 0; i < K; i++) {

aver\_row = 0;

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

aver\_row += x[i][j]; // Суммирование элементов текущей строки

}

aver\_row /= N; // Расчет среднего арифметического текущей строки

if (aver\_row < aver\_global) { // Проверка условия

for (k = 0; k < N; k++) {

x[i][k] = 0; // Обнуление элементов строки

}

}

}

// Вывод обновленного массива

for (i = 0; i < K; i++) {

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

printf("%4d ", x[i][j]); // Вывод элемента массива

}

printf("\n"); // Переход на новую строку после вывода строки массива

}

return 0;

}

Изображение выглядит как текст, снимок экрана, программное обеспечение, Мультимедийное программное обеспечение

Автоматически созданное описание

Доп задание

Практика

#include <stdio.h>

#define K 2 // Определение количества строк

#define N 3 // Определение количества столбцов

int main(void) {

int x[K][N]; // Объявление двумерного массива размером K на N

int i, j, k, l, temp; // Объявление переменных-счетчиков и временной переменной для обмена значениями

// Цикл для ввода элементов массива

for (i = 0; i < K; i++) {

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

scanf("%d", &x[i][j]); // Ввод элемента массива

}

}

// Сортировка элементов массива по возрастанию

// Используется четырехзвенная сортировка пузырьком для двумерного массива

for (i = 0; i < K; i++) {

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

for (k = 0; k < K; k++) {

for (l = 0; l < N; l++) {

if (x[i][j] < x[k][l]) { // Если текущий элемент меньше элемента в другой строке/столбце

temp = x[i][j]; // Сохранение текущего элемента во временной переменной

x[i][j] = x[k][l]; // Замена текущего элемента на сравниваемый

x[k][l] = temp; // Замена сравниваемого элемента на сохраненный

}

}

}

}

}

// Вывод отсортированного массива

for (i = 0; i < K; i++) {

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

printf("%4d ", x[i][j]); // Вывод элемента массива с отступом в 4 позиции

}

printf("\n"); // Переход на новую строку после вывода строки массива

}

return 0;

}

Теория

Сортировка слиянием— алгоритм сортировки, который упорядочивает списки (или другие структуры данных, доступ к элементам которых можно получать только последовательно, например — потоки) в определённом порядке.

Сначала задача разбивается на несколько подзадач меньшего размера. Затем эти задачи решаются с помощью рекурсивного вызова или непосредственно, если их размер достаточно мал. Наконец, их решения комбинируются, и получается решение исходной задачи.

Для решения задачи сортировки эти три этапа выглядят так:

1. Сортируемый массив разбивается на две части примерно одинакового размера;
2. Каждая из получившихся частей сортируется отдельно, например — тем же самым алгоритмом;
3. Два упорядоченных массива половинного размера соединяются в один.