Вариант 15 В одномерном массиве, состоящем из n вещественных элементов, вычислить:

количество элементов массива, больших С;

произведение элементов массива, расположенных после максимального по модулю элемента;

преобразовать массив таким образом, чтобы сначала располагались все отрицательные элементы, а потом — все положительные (элементы, равные 0, считать положительными).

Лабораторная работа №1. Массивы в языке Си

При решении задач с большим количеством данных одинакового типа использование переменных с различными именами, не упорядоченных по адресам памяти, затрудняет программирование. В подобных случаях в языке Си используют объекты, называемые массивами.

Математическим понятием, которое привело к появлению в языках программирования понятия «массив», являются матрица и её частные случаи: вектор-столбец или вектор-строка.

Массив - это непрерывный участок памяти, содержащий последовательность объектов одинакового типа, обозначаемый одним именем.

Массив характеризуется следующими основными понятиями:

Элемент массива (значение элемента массива) – это значение, хранящееся в определенной ячейке памяти, расположенной в пределах массива. Элементы матриц в математике принято обозначать с использованием индексов. В языке Си для этого используются квадратные скобки.

Например,

int a[10]; /\*определен массив из 10 элементов a[0], a[1], …, a[9]

float z[2][3]; /\*определен двумерный массив из двух строк и трёх столбцов

Элементы двумерного массива z можно перечислить так:

z[0][0], z[0][1], z[0][2] – элементы первой строки;

z[1][0], z[1][1], z[1][2] – элементы второй строки.

Обратите внимание, что индексы элементов массива в языке Си начинаются с 0, т.е. индексы элементов изменяются от 0 до n-1, где n – количество элементов в массиве.

Ограничений на размерность массивов, т.е. на числе индексов, в языке Си теоретически нет. Однако чаще всего используются одномерные и двумерные массивы.

Каждый элемент массива, таким образом, характеризуется тремя величинами:

· индексом элемента (порядковым номером элемента в массиве);

· значением элемента;

· адресом элемента - адресом начальной ячейки памяти, в которой расположен этот элемент.

Адресом всего массива является адрес элемента массива с индексом 0.

Имя массива – идентификатор, используемый для обращения к элементам массива.

Размер массива – количество элементов массива

Размер элемента – количество байт, занимаемых одним элементом массива.

Графически расположение массива в памяти компьютера можно представить в виде непрерывной ленты адресов.

massiv

Представленный на рисунке массив содержит q элементов с индексами от 0 до q-1. Каждый элемент занимает в памяти компьютера k байт, причем расположение элементов в памяти последовательное.

Адреса i-го элемента массива имеет значение: n+k·i.

Адрес массива представляет собой адрес начального (нулевого) элемента массива. Для обращения к элементам массива используется порядковый номер (индекс) элемента, начальное значение которого равно 0. Так, если массив содержит q элементов, то индексы элементов массива меняются в пределах от 0 до q-1.

Длина массива – количество байт, отводимое в памяти для хранения всех элементов массива.

ДлинаМассива = РазмерЭлемента \* КоличествоЭлементов

Для определения размера элемента массива может использоваться функция

int sizeof(тип);

Например,

sizeof(char) = 1;

sizeof(int) = 4;

sizeof(float) = 4;

sizeof(double) = 8;

Объявление и инициализация массивов

Для объявления массива в языке Си используется следующий синтаксис:

тип имя[размерность]={инициализация};

Инициализация массивов при объявлении не является обязательной.

Инициализация представляет собой набор начальных значений элементов массива, разделенных запятыми.

int a[10] = {1, 2, 3, 4, 5, 6, 7, 8, 9}; // массив a из 10 целых чисел

Если массив проинициализирован при объявлении, то константные начальные значения его элементов указываются через запятую в фигурных скобках. В этом случае количество элементов в квадратных скобках может быть опущено:

int a[ ] = {1, 2, 3, 4, 5, 6, 7, 8, 9};

Если количество начальных значений меньше, чем объявленная длина массива, то начальные значения получат только первые элементы массива (с меньшими значениями индекса):

int a[10] = {7, 8, 9};

В данном примере определены значения только первых трех элементов массива а: a[0]=7, a[1]=8, a[2]=9. Элементы a[3], … , a[9] не инициализированы.

Однако часто требуется задавать значения элементов массива в процессе выполнения программы. При этом используется объявление массива без инициализации. В таком случае указание количества элементов в квадратных скобках обязательно.

int a[10];

Для задания начальных значений элементов массива очень часто используется параметрический цикл:

#include <stdio.h>

int main() {

int a[5]; // объявлен массив a из 5 элементов

int i;

// Ввод элементов массива

for(i=0; i<5; i++) {

printf("a[%d] = ", i);

scanf("%d", &a[i]); // &a[i] - адрес i-го элемента массива

}

// Вывод элементов массива

for(i=0;i<5;i++) {

printf("%d ",a[i]); // пробел в формате печати обязателен

}

return 0;

}

На этапе отладки программы для того, чтобы задать значения элементам массива, часто пользуются счетчиком случайных чисел:

#include <stdio.h>

#include <stdlib.h>

#include <conio.h>

int main() {

int a[5]; // объявлен массив a из 5 элементов

int i;

// Инициализация счетчика случайных чисел

printf("Press any key\n");

srand (getch());

// Задаем значения элементов массива с помощью счетчикам случайных чисел

for(i=0; i<5; i++) {

a[i] = rand () % 100 – rand () %100; // Получим числа от -99 до +99

printf("%d ", a[i]);

}

//

ВЫПОЛНЕНО

#include <stdio.h>

#include <stdlib.h>

#include <conio.h>

#include <time.h>

*int* main()

{

*int* n;

printf("enter size of the array: ");

scanf("%d", &n);

*int* array[n];

// srand(getch());

srand(time(NULL));

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

{

array[i] = rand() % 100 - rand() % 100;

printf("%d ", array[i]);

}

*int* c;

printf("\nenter C: ");

scanf("%d", &c);

*int* cn = 0;

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

{

if (array[i] > c)

{

cn += 1;

}

}

printf("1) count of numbers larger then %d is %d", c, cn);

*int* max, max\_i;

max = array[0];

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

{

if (abs(array[i]) > abs(max))

{

max = array[i];

max\_i = i;

}

}

*float* max\_mul = 1;

for (*int* i = max\_i + 1; i < n; i++)

{

max\_mul \*= array[i];

}

printf("\n2) maximum value with abs = %d on index = %d | multiply of numbers after = %f", max, max\_i, max\_mul);

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

{

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

{

if (array[j] > array[j + 1])

{

*int* tmp = array[j];

array[j] = array[j + 1];

array[j + 1] = tmp;

}

}

}

printf("\n3) sorted array \n");

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

{

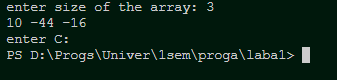
printf("%d ", array[i]);

}

return 0;

}

Результат



Разбор кода.

#include <stdio.h>

Подключает стандартную библиотеку ввода-вывода, что позволяет использовать функции, такие как printf и scanf.

#include <stdlib.h>

Подключает библиотеку для работы с библиотечными функциями, такими как rand и srand, а также для управления динамической памятью.

#include <conio.h>

Подключает библиотеку для работы с консольными вводами/выводами. Эта библиотека не стандартная и может не поддерживаться на некоторых компиляторах, например, на GCC.

#include <time.h>

Подключает библиотеку для работы с временем, позволяя использовать функции, такие как time, для генерации случайных чисел.

int main()

Начало основной функции main, с которой начинается выполнение программы.

{

int n;

Объявляет целочисленную переменную n, которая будет использоваться для хранения размера массива.

printf("enter size of the array: ");

Выводит на экран сообщение с просьбой ввести размер массива.

scanf("%d", &n);

Считывает целое число, введенное пользователем, и сохраняет его в переменной n.

int array[n];

Объявляет массив array размером n. Это позволяет создать массив переменной длины.

srand(time(NULL));

Инициализирует генератор случайных чисел на основе текущего времени, чтобы при каждом запуске программы генерировались разные случайные числа.

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

Начинает цикл for, который будет выполняться от 0 до n-1.

{

array[i] = rand() % 100 - rand() % 100;

Заполняет каждую ячейку массива array случайным значением, полученным разностью двух случайных чисел от -99 до 99.

printf("%d ", array[i]);

Выводит на экран текущее значение элемента массива.

}

Закрывает цикл for, завершив инициализацию и вывод элементов массива.

int c;

Объявляет целочисленную переменную c, которая будет использоваться для получения значения от пользователя.

printf("\nenter C: ");

Выводит сообщение с просьбой ввести значение C.

scanf("%d", &c);

Считывает введенное пользователем значение и сохраняет его в переменной c.

int cn = 0;

Объявляет целочисленную переменную cn и инициализирует ее нулем. Эта переменная будет использоваться для подсчета количества элементов массива, больших c.

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

Начинает цикл for, который будет выполняться от 0 до n-1.

{

if (array[i] > c)

Проверяет, больше ли текущий элемент массива array[i] значения c.

{

cn += 1;

Если условие верно, увеличивает счетчик cn на 1.

}

}

Закрывает условие if и цикл for.

printf("1) count of numbers larger than %d is %d", c, cn);

Выводит на экран количество элементов массива, больших c.

int max, max\_i;

Объявляет целочисленные переменные max и max\_i, которые будут использоваться для хранения максимального значения массива и его индекса соответственно.

max = array[0];

Инициализирует max первым элементом массива array.

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

Начинает цикл for, который будет выполняться от 0 до n-1.

{

if (abs(array[i]) > abs(max))

Проверяет, является ли абсолютное значение текущего элемента массива array[i] больше абсолютного значения max.

{

max = array[i];

max\_i = i;

Если условие верно, обновляет max и сохраняет индекс max\_i в i.

}

}

Закрывает условие if и цикл for.

float max\_mul = 1;

Объявляет переменную max\_mul типа float и инициализирует ее единицей. Эта переменная будет использоваться для хранения произведения элементов после максимального.

for (int i = max\_i + 1; i < n; i++)

Начинает цикл for, который проходит по массиву, начиная с элемента, следующего после max\_i, до конца массива.

{

max\_mul \*= array[i];

Умножает max\_mul на текущий элемент массива array[i].

}

Закрывает цикл for.

printf("\n2) maximum value with abs = %d on index = %d | multiply of numbers after = %f", max, max\_i, max\_mul);

Выводит на экран максимальное значение max, его индекс max\_i и произведение всех элементов после него.

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

Начинает внешний цикл for для сортировки массива методом пузырька. Этот цикл будет выполнять сравнения между элементами массива.

{

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

Начинает внутренний цикл for, который проходит по массиву, исключая уже отсортированные элементы.

{

if (array[j] > array[j + 1])

Проверяет, больше ли текущий элемент array[j] следующего элемента array[j + 1].

{

int tmp = array[j];

array[j] = array[j + 1];

array[j + 1] = tmp;

Если условие верно, меняет местами элементы array[j] и array[j + 1] с помощью временной переменной tmp.

}

}

}

Закрывает условие if, внутренний цикл for, а затем внешний цикл for.

printf("\n3) sorted array \n");

Выводит сообщение о том, что будет показан отсортированный массив.

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

Начинает цикл for, чтобы вывести отсортированный массив.

{

printf("%d ", array[i]);

Выводит каждый элемент отсортированного массива.

}

Закрывает цикл for, который выводит элементы массива.

return 0;

}