Тема 1.3 Массивы: определение, описание, размещение в памяти, использование

Массивы - это группа элементов одинакового типа (double, float, int и т.п.) упорядоченных индексами. Из объявления массива компилятор должен получить информацию о типе элементов массива и их количестве.

Объявление массива имеет два формата:

спецификатор-типа описатель [константное - выражение];
спецификатор-типа описатель [ ];

Описатель - это идентификатор массива.

Спецификатор-типа задает тип элементов объявляемого массива. Элементами массива не могут быть функции и элементы типа void.

Константное-выражение в квадратных скобках задает количество элементов массива.

Константное-выражение при объявлении массива может быть опущено в следующих случаях:

- при объявлении массив инициализируется,
- массив объявлен как формальный параметр функции,
- массив объявлен как ссылка на массив, явно определенный в другом файле.

В языке СИ определены только одномерные массивы, но поскольку элементом массива может быть массив, можно определить и многомерные массивы. Они формализуются списком константных-выражений следующих за идентификатором массива, причем каждое константное-выражение заключается в свои квадратные скобки.

Каждое константное-выражение в квадратных скобках определяет число элементов по данному измерению массива, так что объявление двухмерного массива содержит два константных-выражения, трехмерного - три и т.д. Отметим, что в языке СИ первый элемент массива имеет индекс равный 0.

Примеры:

```
int a[2][3]; /* представлено в виде матрицы
            a[0][0]  a[0][1] a[0][2]
            a[1][0]  a[1][1] a[1][2]      */

double b[10]; /* вектор из 10  элементов имеющих тип double */

int w[3][3] = { { 2, 3, 4 },
                { 3, 4, 8 },
                { 1, 0, 9 } };
```

В последнем примере объявлен массив w[3][3]. Списки, выделенные в фигурные скобки, соответствуют строкам массива, в случае отсутствия скобок инициализация будет выполнена неправильно.

Объявления массивов

Массивы занимают область в памяти. Программист указывает тип каждого элемента, количество элементов и компилятор может зарезервировать соответствующий объем памяти.

Пример:

```
# include <iostream.h>
# include <iomanip.h>

int main ()
{
        int n[10];
        for (int i=0; i<10; i++)
        n[i]=0;
        cout<< “Элемент” << setw(13) << “Значение” <<\n;
```

```
    for (i=0; i<10; i++)                                    return 0; }
    cout << setw(7) << i << setw(13) << n[i]
<<\n;
```

Первая структура for используется для присвоения начальных нулевых значений элементам массива n.

setw указывает ширину поля, в которое будет выведено следующее значение.

Элементам массива может присваиваться начальное значение в объявлении массива с помощью следующего за объявлением списка (заключенного в {}) одинаковых по смыслу и разделенных запятыми начальных значений.

```
int a[10] = {0, 1, 2, 3, 4, 5, 6, 7, 8, 9}
```

Если начальных значений меньше, чем элементов в массиве, оставшиеся элементы автоматически получают нулевые начальные значения.

```
int n[10] = {0} – присвоение нулевых значений всем элементам массива n.
```

Это объявление явно присваивает нулевое значение первому элементу и неявно присваивает нулевые значения оставшимся девяти элементам.

Массивы автоматически не получают начальные нулевые значения неявно.

Если размер массива не указан в объявлении со списком инициализации, то количество элементов массива будет равно количеству элементов в списке начальных значений.

```
int a[] = {1, 2, 3, 4}
```

```
const int arraysize = 10;
int a [arraysize];
```

Строка const int arraysize = 10 использует спецификатор const для объявления именованной константы arraysize, имеющей значение 10.

Именованные константы должны получать при объявлении в качестве начальных значений постоянные выражения, которые после этого не могут быть модифицированы.

Использование именованных констант для задания размеров массивов делает программу более масштабируемой.

Массиву символов можно задать начальное значение, используя литеральную константу.

```
char st1[] = “first”;
```

Это объявление присваивает элементам массива st1 в качестве начальных значений отдельные символы строки “first”.

Размер массива st1 определяется компилятором на основе длины строки.

Строка “first” содержит пять символов плюс символ окончания строки, называемый нулевым символом. Таким образом st1 содержит шесть элементов. Нулевой символ представляет собой символьную константу ‘\0’. Все строки заканчиваются этим символом.

Символьному массиву можно задать в качестве начальных значений список отдельных символьных констант, указанных в списке инициализации.

```
char st1[] = {‘f’, ‘i’, ‘r’, ‘s’, ‘t’, ‘\0’}
```

Примеры:

```
// Присваивание элементам массива четных начальных значений // от 2 до 20.

# include   <iostream.h>

# tinclude  <iomanip.h>

main()

{
   const  int.  arraySize =  10;
   int   s[arraySize];

   for  (int  j  =  0;  j  < arraySize;  j++)          // задание значений
      s[j]  = 2  + 2  * j;

   cout  «  "Элемент"  «  setw(13)  «  "Значение"  « endl;
   for(j  = 0;  j  < arraySize;  j++)                  // печать значений
   cout  «  setw(7)  «  j  «  setw(13)  «  s[j]  «  endl;
   return  0;
}
```

```
// Вычисление суммы элементов массива

#include <iostream.h>

main ( )

(
   const int arraySize = 12;
   int a [arraySize] = {1, 3, 5, 4, 7, 2, 99, 16, 45, 67, 89, 45};
   int total = 0;
   for (int i = 0; i < arraySize; i++) total += a[i] ;
   cout « "Сумма значений элементов массива равна "
   « total « endl;
   return 0;
 }
```

К сорока студентам обратились с просьбой оценить качество пищи в студенческом кафетерии по десятибалльной шкале (1 означает отвратительное качество, а 10 означает отличное). Поместите сорок ответов в массив целых чисел и просуммируйте результаты опроса.

```
// Программа опроса студентов
#include <iostream.h>
#include <iomanip.h>

const int responseSize = 40, frequencySize = 11;
int responses[responseSize] = {1, 2, 6, 4, 8, 5, 9, 7, 8,
10, 1, 6, 3, 8, 6, 10, 3, 8, 2, 7, 6, 5, 7, 6, 8, 6, 7,
5, 6, 6, 5, 6, 7, 5, 6, 4, 8, 6, 8, 10};
int frequency[frequencySize] = {0};

for (int answer = 0; answer < responseSize; answer++)
++frequency[responses[answer] ] ;

cout « "Рейтинг" « setw(17) « "Частота" « endl;

for (int rating = 1; rating < frequencySize; rating++) cout « setw(6} «
rating « setw(17) « frequency[rating] « endl;
return  0;
```