Одномерный массив

**Массив** – это набор **однотипных** данных, которые располагаются в памяти последовательно друг за другом, объединенные общим **именем**. Количество элементов в массиве называется **размером** массива.

Выглядит массив примерно так:

Объявление массива имеет такую структуру:

тип [] имя\_массива = new тип[размер массива];

Пример:

int[] array = new int[5];

Доступ к элементам осуществляется по **индексу**. Следует помнить, что **индексация начинается с нуля** – первый элемент массива имеет индекс **0**, а последний - *n*−1, где *n* – размер массива.

int[] numbers = new int[5];

numbers[0] = 5;

numbers[1] = 2;

numbers[4] = 3;

numbers[5] = 2;

Console.WriteLine(numbers[0]);

Для того, чтобы получить размерность массива, нужно воспользоваться функцией **Length**:

int[] numbers = new int[5];  
Console.WriteLine(numbers.Length);

В качестве массива можно представить, например, список студентов в группе (имена), показатели температуры воздуха за последние несколько дней и так далее.

Ввод и вывод одномерного массива

В задачах, где вводится последовательность чисел на одной строке, сохранять в массив нужно следующим образом:

int n = Convert.ToInt32(Console.ReadLine());   
string s = Console.ReadLine();   
string[] ss = s.Split(' ');

То есть, если у нас входные данные такие

5

1 3 2 5 4

то переменные будут иметь следующие значения

*n*=5

, *s*="1 3 2 5 4", *ss*= ["1", "3", "2", "5", "4"]

Как видно, в переменной ***ss*** у нас сохранились **числа в виде строк**. Если захотим сохранить числа в массиве **в виде чисел**, а не строк, то нам нужно завести **новый массив целого типа**, и записать туда соответствующие элементы исходного массива, **конвертируя** их в целые числа. Для большего понимания, рассмотрим код:

int[] numbers = new int[n];

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

{

   int number = Convert.ToInt32(ss[i]);

   numbers[i] = number;

}

Теперь в массиве **numbers** хранится [1, 3, 2, 5, 4], то есть в массиве хранятся **числа** и оперировать мы с ними можем как с числами. Чтобы вывести элементы массива на консоль **через пробел** нужно:

for (int i = 0; i < n; i++)  
{  
   Console.Write(numbers[i] + " ");  
}

Нахождение максимума, и индекса максимума

Рассмотрим следующую задачу: в последовательности различных целых чисел, найти максимум и индекс максимума.  Например:

**Входные данные**  
8  
3 5 2 1 8 2 4 5

**Выходные данные**  
8  
4

Алгоритм нахождения максимума мы уже рассматривали. Для нахождения индекса максимума нужно просто при обновлении максимума сохранить еще и индекс максимума.

Для лучшего понимания рассмотрим код:

int n = Convert.ToInt32(Console.ReadLine());

string s = Console.ReadLine();

string[] ss = s.Split(' ');

int[] numbers = new int[n];

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

{

   int number = Convert.ToInt32(ss[i]);

   numbers[i] = number;

}

int max = numbers[0];

int indexMax = 0;

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

{

   if (numbers[i] > max)

   {

      max = numbers[i];

      indexMax = i;

   }

}

Console.WriteLine(max);

Console.WriteLine(indexMax);

Вложенные циклы

Разберем задачу. Дан массив чисел. Нужно вывести какое количество чисел от 0 до 9 встречается в данном массиве.

Например:

**Входные данные:**

5

5 2 3 5 3

**Выходные данные:**

Количество чисел 0 равно 0

Количество чисел 1 равно 0

Количество чисел 2 равно 1

Количество чисел 3 равно 2

Количество чисел 4 равно 0

Количество чисел 5 равно 2

Количество чисел 6 равно 0

Количество чисел 7 равно 0

Количество чисел 8 равно 0

Количество чисел 9 равно 0                    
  
Рассмотрим код, который решает предложенную задачу:

int n = Convert.ToInt32(Console.ReadLine());

string s = Console.ReadLine();

string[] ss = s.Split(' ');

int[] numbers = new int[n];

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

{  
   int number = Convert.ToInt32(ss[i]);

   numbers[i] = number;  
}

for (int i = 0; i < 10; i++)

{  
   int cnt = 0;

   for (int j = 0; j < n; j++)

   {  
      if (numbers[j] == i)

      {

         cnt++;

      }

   }

  Console.WriteLine("Количество чисел " + i + " равно " + cnt);   
}

Двумерные массивы

Объявление

int[] arr  = new int [n];

создает одномерный массив -набор пронумерованных элементов, идущих в памяти последовательно. **На месте элементов массива может быть также другой массив**. Данная конструкция называется **двумерным массивом**. Двумерные массивы можно представлять в виде **таблицы**(матрицы):

У таблицы есть две характеристики: количество **строк** и количество **столбцов**. Чтобы создать двумерный массив, нужно воспользоваться следующим синтаксисом:

тип [,] имя = new тип [количество строк, количество столбцов]

Создадим двумерный массив целых чисел с тремя строками и четырьмя столбцами(как на рисунке):

int[,] a = new int[3, 4];

Данное объявление создает массив из **3** объектов, каждый из которых в свою очередь является массивом типа **int[] размера 4**. Тогда **a[i]**, где **i** принимает значения от **0** до **2** будет одним из **4** созданных обычных одномерных массивов, и обратиться к элементу с номером **j** в этом массиве можно через **a[i,j]**.

Как уже было сказано, двумерные массивы можно представлять в виде **таблицы**, в которой **первый индекс** элемента означает **номер строки**, а **второй индекс** – **номер столбца**. Заметим, что **индексация** строк и столбцов начинается **с** **нуля**.

Чтобы считать значение элемента массива, нужно обратиться к нему по имени, дополнительно указав **индекс строки** и **индекс столбца**. Например, чтобы считать элемент на первой строке и во втором столбце, нужно написать:

int number = a[0, 1];

Также можно изменять элементы массива. Например, изменим тот же элемент:

a[0, 1] = 8;

Многомерные массивы используются гораздо реже одномерных.

Ввод двумерного массива

Вначале любой задачи нам будут задаваться размеры двумерного массив: число **строк** и **столбцов**. По ним мы создаем массив, то есть :

int n = Convert.ToInt32(Console.ReadLine());

int m = Convert.ToInt32(Console.ReadLine());

int[,] a = new int[n, m];

Давайте представим, что у нас **одна** строка и **m** столбцов. Тогда, чтобы ввести(заполнить) двумерный массив с консоли, например:

**Входные данные**

1 4   
2 7 1 4

нужно **считать** строку, **разбить** ее по пробелам, соответствующие элементы **сконвертировать** в числа и записать в первой строке двумерного массива, то есть первый индекс равен 0, так как индексация начинается с **нуля**. Для лучшего понимания рассмотрим код:

string s = Console.ReadLine();

string[] ss = s.Split(' ');   
for (int j = 0; j < m; j++)

{  
  a[0, j] = Convert.ToInt32(ss[j]);

}

Заметим, что если строк будет **n**, то данные операции нужно выполнить **n** раз. Следовательно, нужно обернуть в цикл, который будет отвечать за строки и вставлять каждое значение в соответствующие строки и столбцы. Для лучшего понимания рассмотрим код:

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

{

  string s = Console.ReadLine();

  string[] ss = s.Split(' ');

  for (int j = 0; j < m; j++)

  {

    a[i, j] = Convert.ToInt32(ss[j]);

  }  
}

Давайте нарисуем таблицу изменений переменных **i** и **j** для **n = 3, m = 4**:

Заметим, что таблица будет заполняться **слева направо по столбцам и сверху вниз по строкам**.

Вывод двумерного массива

Как мы знаем, чтобы вывести одномерный массив, нужно пройтись циклом, перебирая индексы от 0 до количество элементов минус 1, и в цикле выводить каждый элемент на одной строке через пробел:

for (int i = 0; i < a.Length - 1; i++)  
{  
  Console.Write(a[i] + " ");  
}

Как известно, **двумерный массив** - это массив таких массивов. То есть, в первой строке одномерный массив, во второй строке - одномерный массив, и так далее. Давайте **перебирать строки**, а внутри каждой строки **выводить** соответствующий ей **одномерный массив**, ведь одномерный массив мы умеем выводить. После того, как выведем очередную строку, нужно сделать **перевод на новую строку**. Для лучшего понимания рассмотрим код:

for (int i = 0; i < n; i++)  
{  
  for (int j = 0; j < m; j++)  
  {  
    Console.Write(a[i, j] + " ");  
  }  
  Console.WriteLine();  
}