

Углубленный курс информатики

Составление программ с использованием
двумерных массивов.



Чузлов Вячеслав Алексеевич

к.т.н., доцент ОХИ ИШПР

ДВУМЕРНЫЕ МАССИВЫ (МАТРИЦЫ)

- В PascalABC.NET двухмерные динамические массивы называются **матрицами**.
- Матрица имеет два измерения, называемые строками и столбцами по аналогии с матрицами в математике.
- Матрица всегда имеет прямоугольную форму: количество элементов в каждой строке постоянно и количество элементов в каждом столбце тоже постоянно. Внешне матрицу всегда можно представить в виде таблицы.
- Пусть имеется матрица, состоящая из m строк и n столбцов. В этом случае говорят, что она имеет размер $m \times n$.
- В матрице размерность (число измерений массива) равна 2.
- Если элемент матрицы A находится на пересечении строки i и столбца j , его записывают как $A_{i,j}$ или $A[i, j]$.

$$a(3,5) = \begin{pmatrix} a_{00} & a_{01} & a_{02} & a_{03} & a_{04} \\ a_{10} & a_{11} & a_{12} & a_{13} & a_{14} \\ a_{20} & a_{21} & a_{22} & a_{23} & a_{24} \end{pmatrix}$$

СТАТИЧЕСКИЕ ДВУМЕРНЫЕ МАССИВЫ

- Как и в случае с одномерными массивами, статический двумерный массив описывается с указанием границ индексов.
- Границы должны быть заданы и для строк, и для столбцов. Память под статический массив распределяется на этапе компиляции, выделяется при загрузке программы, и в дальнейшем **не может быть перераспределена**.

Статический массив описывается в виде

```
var ИмяМассива: array[m1..n1, m2..n2] of Тип;
```

- Конструкция вида m..n описывает минимальное и максимальное значение, которое может принимать индекс массива, причем допускаются и отрицательные значения.
- Эта конструкция задается константой порядкового типа.
- Количество элементов в массиве можно вычислить по формуле $(n1-m1+1) \times (n2-m2+1)$.

```
var a: array[0..12, 1..4] of byte;  
var b, c: array[-5..8, 0..6] of real;
```

СТАТИЧЕСКИЕ ДВУМЕРНЫЕ МАССИВЫ

Описание статического массива можно совместить с инициализацией его элементов:

```
begin
  var a: array[1..3, 1..4] of integer := ((1, 2, 3, 4),
    (5, 6, 7, 8), (9, 10, 11, 12));
  Println(a);

  var b: array[0..2, 1..3] of real := ((1.2, 5, -3.05),
    (-4, -7, 1), (15, 7, 7));
  Println(b);

  var c: array [0..2, 1..2] of char := (('a', 'b'),
    ('c', 'd'), ('e', 'f'));
  Println(c)
end.
```

```
[[1,2,3,4],[5,6,7,8],[9,10,11,12]]
[[1.2,5,-3.05],[-4,-7,1],[15,7,7]]
[[a,b],[c,d],[e,f]]
```

Двумерные статические массивы в PascalABC.NET оставлены в целях совместимости с базовым Паскалем.

ДИНАМИЧЕСКИЕ ДВУМЕРНЫЕ МАССИВЫ (МАТРИЦЫ)

- Пусть у нас имеется двумерный массив размером 4×3.
- В PascalABC.NET динамические массивы нумеруются с нуля, поэтому получаются четыре строки с номерами от 0 до 3 и три столбца, нумерованные от 0 до 2.
- Каждую строку двумерного массива можно представить, как обычный массив. Массив из таких строк образует массив массивов.

```
begin
  var A: array of array of integer;
  var (m, n) := (4, 3); // число строк и столбцов

  SetLength(A, m); // распределим память под m строк
  for var i := 0 to m - 1 do
    SetLength(A[i], n); // в каждой строке создадим массив из n элементов

  A[3][2] := 43; // строка с индексом 3, в ней элемент индексом 2
  A[1, 0] := 21; // строка с индексом 1, в ней элемент индексом 0

  Println(A)
end.
```

```
[[0,0,0],[21,0,0],[0,0,0],[0,0,43]]
```

ДИНАМИЧЕСКИЕ ДВУМЕРНЫЕ МАССИВЫ

Альтернативный вариант:

```
begin  
  var A: array [,] of integer; // обратите внимание на запятую  
  var (m, n) := (4, 3); // число строк и колонок  
  SetLength(A, m, n);  
  A[3, 2] := 43; // писать A[3][2] здесь не допускается  
  A[1, 0] := 21;  
  Println(A)  
end.
```

```
[[0,0,0],[21,0,0],[0,0,0],[0,0,43]]
```

ДИНАМИЧЕСКИЕ ДВУМЕРНЫЕ МАССИВЫ

Для двумерных массивов, подобно одномерным, можно совмещать описание с выделением памяти. Для этого массив создается с использованием ключевого слова **new**:

```
var a: array [,] of integer := new integer[4, 3]; // с описанием типа  
var b := new real[4, 3]; // с авто выводением типа
```

Можно также выполнить инициализацию, добавив **конструктор массива**:

```
var a := new integer[4, 3] ((1, 2, 3), (4, 5, 6), (7, 8, 9), (10, 11, 12));  
var b := new real[2, 3] ((2.1, 3.7, 5), (1, 2, 3));  
var d: array of array of integer := ((1, 2, 3), (4, 5), (6, 7, 8));
```

ГЕНЕРАЦИЯ МАТРИЦ

Заполнение случайными значениями

- **MatrRandom**(m, n, a, b) – заполнение матрицы размера $m \times n$ целыми числами из интервала $[a; b]$. Имеется синоним **MatrRandomInteger** ;
- **MatrRandomReal**(m, n, a, b) – заполнение матрицы размера $m \times n$ вещественными числами из интервала $[a; b]$.

```
begin
  var a := MatrRandom(6, 9, -50, 50);
  a.Println;
  Println;

  var b := MatrRandomReal(4, 3, -5, 5);
  b.Println
end.
```

```
-18  -4  43  19 -39   6  48  -5 -34
-14   7 -43 -17   1  27 -27  19 -47
-25 -30  21  48 -11  21   2 -20  -8
-29  38  -6 -26  48  -5 -37 -48  -2
 33 -24  13  45  14 -33   5 -26  -5
   3  -5  26 -49   3 -37   3  40 -24
```

```
 2.96  -1.57  -4.33
-2.83  -3.34   3.04
-3.57   2.61   0.71
 2.30  -3.82   0.92
```

Если необходимо настроить вывод элементов матрицы в методе **Println** можно явно указывать параметры вывода:

```
begin
  MatrRandom(6, 9, -50, 50).Println(6);
  Println;
  MatrRandomReal(4, 3, -5, 5).Println(11, 7)
end.
```

```
  -3  -25   26  -32   40   29  -42  -40   48
-33   25    0  -35  -43  -49   48   36  -13
  -8   10   37   48  -42    0    4  -39   42
  15   36  -30  -23   10   30    5  -15   17
-17    8  -18   43   21  -19  -35    0  -11
  44   -2  -38   35   37   -6  -29   -3   10
```

```
-3.9536103 -0.0407892 -3.2696012
-2.3519689  1.4215049  2.8354788
 3.7539460 -0.6999880 -2.5765292
 2.7467702  4.0072183 -3.4617646
```


ГЕНЕРАЦИЯ МАТРИЦ

Заполнение фиксированным значением

- **MatrFill**(m, n, x) возвращает матрицу размера $m \times n$, заполненную значением выражения x. Тип элементов матрицы будет совпадать с типом значения x.

```
begin
  MatrFill(3, 4, 1).Println();
  Println;
  MatrFill(4, 3, 0.25).Println
end.
```

```
1    1    1    1
1    1    1    1
1    1    1    1
```

```
0.25  0.25  0.25
0.25  0.25  0.25
0.25  0.25  0.25
0.25  0.25  0.25
```

КЛАВИАТУРНЫЙ ВВОД ЗНАЧЕНИЙ ЭЛЕМЕНТОВ МАТРИЦ

- В базовом Паскале имеется единственный способ ввести значения элементов матрицы: организовать перебор всех элементов во вложенных циклах с последовательным присваиванием введенного значения очередному элементу.
- Порядок ввода можно задать как по строкам (второй индекс меняется быстрее первого), так и по столбцам (первый индекс меняется быстрее).

```
begin
  var (m, n) := (3, 2);
  var a := new real[m, n];

  for var i := 0 to m - 1 do
    for var j := 0 to n - 1 do
      Read(a[i, j]);

  a.Println
end.
```

```
2.5 -1.4 3.9 0.15 2 0.03
2.50 -1.40
3.90 0.15
2.00 0.03
```

КЛАВИАТУРНЫЙ ВВОД ЗНАЧЕНИЙ ЭЛЕМЕНТОВ МАТРИЦ

- Чтобы каждый раз не писать эти типовые вложенные циклы, в PascalABC.NET введены функции **ReadMatrInteger**(m, n) и **ReadMatrReal**(m, n), возвращающие матрицу размера $m \times n$ типа **integer** или **real** соответственно, заполненную принятыми с клавиатуры значениями.
- Для прочих типов данных придется пользоваться приведенным выше решением на базе вложенных циклов.

```
begin
  var (m, n) := (3, 2);
  var a := ReadMatrReal(m, n);
  a.Println
end.
```

```
2.5 -1.4 3.9 0.15 2 0.03
2.50 -1.40
3.90 0.15
2.00 0.03
```

ПОЛУЧЕНИЕ СВЕДЕНИЙ О ТЕКУЩИХ РАЗМЕРАХ МАТРИЦЫ

- **a.RowCount** – расширение, возвращающее количество строк в матрице;
- **a.ColCount** – расширение, возвращающее количество столбцов в матрице

```
begin
  var (m, n) := (3, 2);
  var a := MatrRandom(m, n, 0, 10);

  a.Println;
  a.ColCount.Println;
  a.RowCount.Print
end.
```

```
0  10
2   0
8   1
2
3
```

```
begin
  var a := MatrRandom(3, 2, 0, 10);
  a.Println;
  println;

  for var i := 0 to a.RowCount-1 do
    for var j := 0 to a.ColCount-1 do
      a[i, j] := j * i + 5 * a[i, j];
    end
  end

  a.Println
end.
```

```
5  6
2  4
3  1

25 30
10 21
15  7
```

ВЫБОРКА ЭЛЕМЕНТОВ МАТРИЦЫ

- **a.Col(k)** – возвращает в виде массива колонку матрицы a с номером k (отсчет номеров ведется от нуля);
- **a.Cols** – возвращает последовательность колонок матрицы, в которой каждая колонка, в свою очередь, является последовательностью;
- **a.Row(k)** – возвращает в виде массива строку матрицы a с номером k (отсчет номеров ведется от нуля);
- **a.Rows** – возвращает последовательность строк матрицы, в которой каждая строка, в свою очередь, является последовательностью;

```
begin
  var a := MatrRandom(4, 4, 1, 100);
  a.Println;
  println;

  a.Col(0).Println;
  a.Row(2).Println;
  println;

  foreach var row in a.Rows do
    row.Println
  end.
```

```
40  59  51  77
21  64  68 100
61  92  98  84
61  73  79  39
```

```
40 21 61 61
61 92 98 84
```

```
40 59 51 77
21 64 68 100
61 92 98 84
61 73 79 39
```

Пример 1

Пусть двумерный массив $a(3, 3)$ состоит из следующих элементов:

$$a(3,3) = \begin{pmatrix} 6 & 14 & -13 \\ 26 & 1 & -10 \\ -23 & -5 & 16 \end{pmatrix}$$

Вычислить:

1. Сумму элементов массива;
2. Произведение элементов массива;
3. Минимальный элемент массива и его индексы;
4. Заменить отрицательные элементы массива их модулем.

Пример 1

```
begin
  var a: array of array of integer := ((6, 14, -13), (26, 1, -10), (-23, -5, 16));
  var (sum, product) := (0, 1);
  var min := a[0, 0];
  var (imin, jmin) := (0, 0);

  for var i := 0 to a.High do
    for var j := 0 to a[i].High do
      begin
        sum += a[i, j];
        product *= a[i, j];

        if min > a[i, j] then
          begin
            min := a[i, j];
            (imin, jmin) := (i, j);
          end;

        if a[i, j] < 0 then
          a[i, j] := abs(a[i, j])
        end;

        sum.Println;
        product.Println;
        Println('$'Минимальный элемент массива - a[{imin}, {jmin}] = {min}');
        println;
        a.PrintLines
      end;
    end;
  end.
```

```
12
522412800
Минимальный элемент массива - a[2, 0] = -23
```

```
[6,14,13]
[26,1,10]
[23,5,16]
```

Пример 2

Составьте программу для заполнения массива $a(3,3)$ случайными числами в интервале $[0; 9]$. Найти сумму элементов первой строки. Результат вывести на экран.

Способ 1

```
begin
  var a := MatrRandom(3, 3, 0, 9);
  a.Println;

  a.Row(0).Sum.Print
end.
```

```
9   1   9
6   2   8
3   3   1
19
```

Способ 2

```
begin
  var a: array of array of integer;
  SetLength(a, 3);
  for var i := 0 to a.High do
    a[i] := ArrRandom(3, 0, 9);
  a.Printlines;

  a[0].Sum.Print
end.
```

```
[6, 7, 5]
[4, 3, 7]
[3, 8, 5]
18
```


Задание 1

Найти сумму минимального элемента первой строки матрицы $P(2,3)$ и максимального элемента массива $X(4)$ и поделить ее на количество элементов, больших нуля массива $X(4)$.

$$P(2,3) = \begin{vmatrix} -3 & 10 & 15 \\ 32 & 12 & -5 \end{vmatrix}, \quad X(4) = -3.5, 120.4, -3.9, 6.11.$$

Задание 2

Заполнить матрицу $a(3, 3)$ случайными целыми числами от 1 до 10. Вывести значения ее элементов. Вычислить сумму элементов первой и последней строк данной матрицы и вывести результат вычисления.

Задание 3

Заполнить матрицу $a(4, 4)$ случайными числами от -3 до 6. Вывести значения ее элементов. Вычислить среднее арифметическое значений неотрицательных элементов каждого столбца данной матрицы.


Задание 4


Заполнить матрицу $a(4, 4)$ случайными целыми числами от 1 до 100. Вывести значения ее элементов. Найти максимальный элемент в каждой строке. Среди максимальных элементов каждой строки найти минимальный.


КОНТАКТНАЯ ИНФОРМАЦИЯ

ЧУЗЛОВ ВЯЧЕСЛАВ АЛЕКСЕЕВИЧ

к.т.н., доцент ОХИ ИШПР

 Учебный корпус №2, ауд. 136

 +7-962-782-66-15

 chuva@tpu.ru