Лекция №7

# Двумерные массивы

# План лекции

1. Понятие двумерного статического массива
2. Ввод – вывод элементов массива
3. Заполнение массива случайными числами
4. Квадратные матрицы
5. Транспонирование матриц
6. Умножение матрицы на вектор
7. Умножение матрицы на матрицу
8. Удаление строки
9. Включение столбца
10. Операции с элементами матриц
11. Преобразование матрицы в одномерный массив
12. Многомерные массивы

# Двумерный массив

При решении практических задач часто приходится иметь дело с различными таблицами данных, математическим эквивалентом которых служат матрицы. Такой способ организации данных, при котором каждый элемент определяется номером строки и номером столбца, на пересечении которых он расположен, называется ***двумерным массивом (матрицей)*** или ***таблицей***.

| Планета | Расст. до Солнца | Относ. обьем | Относ. масса |
|---|---|---|---|
| Меркурий | 57.9 | 0.06 | 0.05 |
| Венера | 108.2 | 0.92 | 0.81 |
| Земля | 149.6 | 1.00 | 1.00 |
| Марс | 227.9 | 0.15 | 0.11 |
| Юпитер | 978.3 | 1345.00 | 318.40 |
| Сатурн | 1429.3 | 767.00 | 95.20 |

Их можно занести в память компьютера, используя понятие двумерного массива. Положение элемента в массиве определяется **двумя индексами**. Они показывают **номер строки** и **номер столбца**. Например: A[7,6], D[56,47].

# Пример объявления двумерного массива

Организацию обработки элементов статического двумерного (многомерного) массива можно организовать с использованием сложноциклической структуры. Так как, например, при заполнении массива необходимо произвести обход всех элементов по строкам ( i ) изменяя индексы столбцов ( j ). Таким образом объявленный массив из N строк и M столбцов

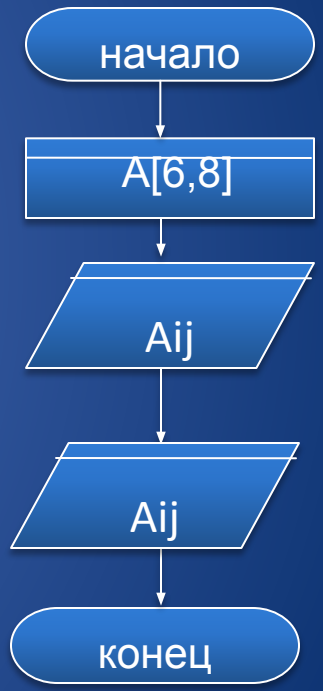
```
Const
      n=6;
      m=8;
Var
      a : array [1..n, 1..m] of integer;
```

будет организован в памяти ЭВМ следующим образом:

| i\j | 1 | 2 | … | m-1 | m |
|---|---|---|---|---|---|
| 1 | A[1,1] | A[1,2] | | | A[1,m] |
| 2 | A[2,1] | A[2,2] | | | |
| … | | | A[i,j] | | |
| n-1 | | | | | |
| n | A[n,1] | | | | A[n,m] |

# Ввод – вывод элементов двумерного массива

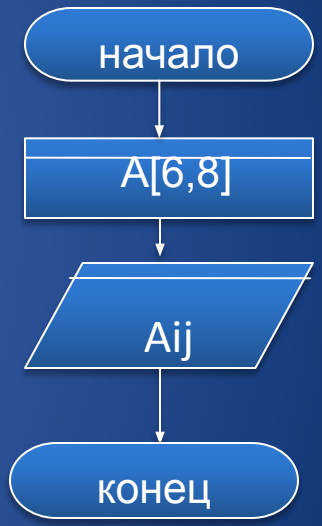
```
Const
        n=6;
        m=8;
Var
        a : array [1..n, 1..m] of integer;
        I, j : integer;
Begin
writeln (‘ Заполнение элементов целочисленного массива A[6,8] ‘);
for i:=1 to n do
         for j:=1 to m do
        begin
        write (‘a[‘, i , ‘, ‘ , j, ‘] =‘);
        read (a[ i , j ]);
        end;
writeln (‘В памяти компьютера сформирован двумерный массив с элементами’);
for i:=1 to n do
        begin
        for  j:=1 to m do
                write (a[ i , j ]:6);
        writeln;
        end
End.
```



# Ввод – вывод элементов двумерного массива

***Генерация элементов двумерного массива случайными числами.***

```
Const
      n=6;
      m=8;
Var
      a : array [1..n, 1..m] of integer;
      I, j : integer;
Begin
randomize;
for i:=1 to n do
      begin
      for  j:=1 to m do
            begin
            a[ i, j ]:=random(101);
            write (a[ i , j ]:6);
            end;
      writeln;
      end
End.
```



# Квадратные матрицы

В квадратной матрице количество строк и столбцов – **одинаково и равно** n.

Любая квадратная матрица имеет элементы **главной и побочной диагонали**.

Диагональные элементы главной диагонали :

a[1 , 1];a[2 , 2];a[3 , 3];….;a[n , n].

Элементами побочной диагонали являются :

a[1 , n];a[2 , n–1];a[3 , n–2];…;a[n–1 , 2];a[n , 1].

# Квадратные матрицы

В качестве примера рассмотрим задачу формирования квадратной матрицы порядка n случайными числами и нахождения произведения элементов главной диагонали и суммы элементов ниже побочной диагонали.

Отметим элементы главной диагонали для нахождения их произведения.

| i\j | 1 | 2 | 3 | … | n-1 | n |
|---|---|---|---|---|---|---|
| 1 | A[1,1] | A[1,2] | A[1,3] | | | A[1,n] |
| 2 | A[2,1] | A[2,2] | A[2,3] | | | |
| 3 | A[3,1] | A[3,2] | A[3,3] | | | |
| … | | | | A[ i, j] | | |
| n-1 | | | | | . . . | |
| n | A[n,1] | | | | | A[n,n] |

# Квадратные матрицы

И элементы ниже побочной диагонали для поиска их суммы.

| i\j | 1 | 2 | 3 | … | n-1 | n |
|---|---|---|---|---|---|---|
| 1 | A[1,1] | A[1,2] | A[1,3] | | | A[1,n] |
| 2 | A[2,1] | A[2,2] | A[2,3] | | | |
| 3 | A[3,1] | A[3,2] | A[3,3] | | | |
| … | | | | A[ i, j] | | |
| n-1 | | | | | . . . | |
| n | A[n,1] | | | | | A[n,n] |

# Квадратные матрицы

```
Const
        n=9;
Var
        a : array [1..n, 1..n] of integer;
        I, j, s, p : integer;
Begin
randomize;
for i:=1 to n do
        begin
        for  j:=1 to n do
                begin
                a[ i, j ]:=random(101);
                write (a[ i , j ]:6);
                end;
        writeln;
        end
p:=1;
for i:=1 to n do
        p:=p*a[ i, i ];
s:=0;
for i:=2 to n do
        for j:=n-i+2 to n do
                s:=s+a[ i, j ];
writeln (p,s);

End.
```

# Транспонирование матриц

В данном алгоритме **транспонирования матрицы** необходимо заменить строки матрицы ее столбцами, а столбцы – строками, т.е. вычислить

b[ i,j ] :=a[ j,i ], где i=1,…,n; j=1,…,m.

Матрица A

| | 1 | 2 | 3 | 4 |
|---|---|---|---|---|
| 1 | 0 | 5 | -7 | 3 |
| 2 | 23 | -45 | 90 | 5 |
| 3 | 102 | 22 | -45 | 21 |
| 4 | -4 | -8 | 0 | 34 |
| 5 | 64 | 4 | 5 | 7 |
| 6 | 10 | -45 | -37 | -23 |
| 7 | -45 | 0 | -3 | 1 |

Матрица B

| | 1 | 2 | 3 | 4 | 5 | 6 | 7 |
|---|---|---|---|---|---|---|---|
| 1 | 0 | 23 | 102 | -4 | 64 | 10 | -45 |
| 2 | 5 | -45 | 22 | -8 | 4 | -45 | 0 |
| 3 | -7 | 90 | -45 | 0 | 5 | -37 | -3 |
| 4 | 3 | 5 | 21 | 34 | 7 | -23 | 1 |

# Транспонирование матриц

```
Const
      n=5;
      m=7;
Var
i, j : integer;
a : array [1..n,1..m] of integer;
b : array [1..m,1..n] of integer;
Begin
randomize;
writeln (‘Сформирована матрица A’);
for i:=1 to n do
      begin
      for  j:=1 to m do
        begin
        a[ i,j ]:=random(31)-15;
        write (a[ i,j ]:6);
        end;
      writeln(‘’);
end;
```

```
for i:=1 to n do
      for j:=1 to m do
        b[ j,i ]:=a[ i,j ];
writeln (‘Получена транспонированная
      матрица B’);
for i:=1 to m do
      begin
      for j:=1 to n do
        write(b[ i,j ]:6);
      writeln(‘’);
end;
End.
```

# Транспонирование матриц

Транспонированную матрицу можно получить в исходном массива A. Для квадратной матрицы `n*n` для этого необходимо поменять местами каждый элемент верхнего треугольника с соответствующим элементом нижнего (диагональные элементы переставлять не нужно).

При этом для каждой строки нужно выполнять перестановку для элементов, расположенных правее главной диагонали, с элементами соответствующего столбца, расположенными ниже главной диагонали. При перестановке используем вспомогательную переменную **tmp**, помещая в нее для временного хранения один из переставляемых элементов, чтобы не потерять его значение.

# Транспонирование матриц

```
Const
      n=9;
Var
i, j, tmp : integer;
a : array [1..n,1..n] of integer;
Begin
randomize;
writeln (‘Сформирована квадратная
      матрица A’);
for i:=1 to n do
      begin
      for  j:=1 to n do
        begin
        a[ i,j ]:=random(101)-50;
        write (a[ i,j ]:6);
        end;
      writeln(‘’);
end;
```

```
for i:=1 to n-1 do
      for j:=i+1 to n do
        begin
        tmp:=a[ i,j ];
        a[ i,j ]:=a[ j,i ];
        a[ j,i ]:=tmp;
        end;
writeln (‘Получена транспонированная
      матрица ’);
for i:=1 to n do
      begin
      for j:=1 to n do
        write(a[ i,j ]:6);
      writeln(‘’);
end;
End.
```

# Умножение матрицы на вектор

Для вычисления произведения C матрицы A размером n*m на вектор B размером m необходимо вычислить $c_i = \sum_{j=1}^{m} a_{ij} b_j$ , i=1,….,n.

$$\begin{pmatrix} a_{11} & a_{12} & a_{13} & a_{14} \\ a_{21} & a_{22} & a_{23} & a_{24} \\ a_{31} & a_{32} & a_{33} & a_{34} \\ a_{41} & a_{42} & a_{43} & a_{44} \\ a_{51} & a_{52} & a_{53} & a_{54} \end{pmatrix} \cdot \begin{pmatrix} b_1 \\ b_2 \\ b_3 \\ b_4 \end{pmatrix} = \begin{pmatrix} c_1 \\ c_2 \\ c_3 \\ c_4 \\ c_5 \end{pmatrix} = \begin{pmatrix} a_{11} \cdot b_1 + a_{12} \cdot b_2 + a_{13} \cdot b_3 + a_{14} \cdot b_4 \\ a_{21} \cdot b_1 + a_{22} \cdot b_2 + a_{23} \cdot b_3 + a_{24} \cdot b_4 \\ a_{31} \cdot b_1 + a_{32} \cdot b_2 + a_{33} \cdot b_3 + a_{34} \cdot b_4 \\ a_{41} \cdot b_1 + a_{42} \cdot b_2 + a_{43} \cdot b_3 + a_{44} \cdot b_4 \\ a_{51} \cdot b_1 + a_{52} \cdot b_2 + a_{53} \cdot b_3 + a_{54} \cdot b_4 \end{pmatrix}$$

Использование вспомогательной переменной s позволяет уменьшить время выполнения программы за счет исключения обращения в цикле по j к элементам массива C.

# Умножение матрицы на вектор

```
Const
      n=6;
      m=9;
Var
i, j, s : integer;
a : array [1..n,1..m] of integer;
b : array [1..m] of integer;
c: array [1..n] of integer;
Begin
randomize;
writeln ('Сформирована матрица
      A');
for i:=1 to n do
      begin
      for  j:=1 to m do
        begin
        a[ i,j ]:=random(101)-50;
        write (a[ i,j ]:6);
                end;
        writeln('');
        end;
writeln ('Сформирован вектор B');
for j:=1 to m do
        begin
        b[ j ]:=random(51)-30;
        write (b[ j ]:6);
        end;
for i:=1 to n do
        begin
        s:=0;
        for j:=1 to m do
                s:=s+a[ i,j ]*b[ j ];
        c[ i ]:=s;
        end;
writeln ('Получен вектор C ');
for i:=1 to n do
      write(c[ i ]:6);
End.
```

# Умножение матрицы на матрицу

Для умножения матрицы A размером n*k на матрицу B размером k*m необходимо вычислить $c_{ij} = \sum_{l=1}^{k} a_{il} b_{lj}$, i=1,…,n; j=1,…,m.

```
Const
    n=3;
    m=4;
    k=5;
Var
i, j, s : integer;
a : array [1..n,1..k] of integer;
b : array [1..k,1..m] of integer;
c : array [1..n,1..m] of integer;
Begin
randomize;
writeln ('Сформирована матрица
    A');
for i:=1 to n do
begin
    for  j:=1 to k do
     begin
        a[ i,j ]:=random(101)-50;
            write (a[ i,j ]:6);
     end;
    writeln;
end;
writeln ('Сформирована матрица B');
for i:=1 to k do
begin
    for  j:=1 to m do
     begin
        b[ i,j ]:=random(351)-85;
```

# Умножение матрицы на матрицу

```
      write (a[ i,j ]:6);
      end;
    writeln();
end;
for i:=1 to n do
    for j:=1 to m do
    begin
    s:=0;
    for l:=1 to k do
     s:=s+a[ i,l ]*b[ l,j ];
    c[ i,j ]:=s;
    end;
```

```
writeln ('Сформирована матрица C');
for i:=1 to n do
        begin
        for j:=1 to m do
                write (c[ i,j ]:6);
        writeln;
        end
End.
```

# Удаление строки матрицы

Алгоритм удаления строки является сходным с алгоритмом удаление элементов одномерного массива, за тем исключением, что операция переноса элементов выполняется для каждого столбца при переборе строк. Рассмотрим программу удаления из матрицы A заданной с клавиатуры строки T.

```
Const
    n=10;
    m=5;
    Var
i, j, t, n : integer;
a : array [1..n,1..m] of integer;
Begin
randomize;
writeln ('Сформирована матрица
    A');
for i:=1 to n do
```

```
begin
    for  j:=1 to m do
      begin
        a[ i,j ]:=random(101)-50;
              write (a[ i,j ]:6);
      end;
    writeln;
end;
writeln ('Введите номер строки для
    удаления');
readln (t);
```

# Удаление строки матрицы

```
k:=n–1;
for i:=t to k do
      for j:=1 to m do
         a[ i,j ]=a[ i+1,j ];
writeln (‘Получена матрица ‘);
for i:=1  to k do
      begin
      for j:=1 to m do
        write (a[ i,j ]);
      writeln;
end
End.
```

# Включение столбца в матрицу

Алгоритм включения столбца в матрицу является сходным с алгоритмом включения элементов одномерного массива, за тем исключением, что операция переноса элементов выполняется для каждой строки при переборе столбцов. Рассмотрим пример, где необходимо в квадратной матрице A вставить столбец, содержащий ее элементы главной диагонали – следующим за столбцом, содержащим минимальный элемент матрицы. Для этого перепишем диагональные элементы в массив B, найдем минимальный элемент и его индекс по столбцу. Перебирая столбцы, сдвинем все столбцы на позицию вправо начиная от найденного индекса столбца минимального элемента. Занесем в следующий столбец за индексом минимального элемента – элементы столбца, которые хранятся в массиве B.

# Включение столбца в матрицу

```
Const
      n=7;
      Var
i, j, min, j_min, m: integer;
a : array [1..n,1..n+1] of integer;
b : array [1..n] of integer;
Begin
randomize;
writeln (‘Сформирована матрица
      A’);
for i:=1 to n do
begin
      for j:=1 to n do
       begin
       a[ i,j ]:=random(101);
       write (a[ i,j ]:6);
       end;
```

```
        b[ i ]:=a[ i,i ];
        writeln;
end;
min:=a[1,1]; j_min:=1;
for i:=1 to n do
        for  j:=1 to n do
                if (a[ i,j ]<min)
                begin
                        min:=a[ i,j ];
                        j_min:=j;
                end;
m:=n+1;
for j:=n downto j_min do
        for i:=1 to n do
                a[ i,j ]:=a[ i,j+1 ];
for i:=1 to n do
        a[ i,j_min+1]:=b[ i ];
writeln (‘Получен массив ‘);
for i:=1 to n do
        begin
        for j:=1 to m do
                write (a[ i,j ]:4);
        writeln;
end
End.
```

# Операции с элементами матриц

Предыдущий пример может служить так же и примером преобразования матрицы, однако довольно часто встречаются задачи на преобразования матриц за счет изменения значений их элементов. Рассмотрим пример на умножение заданной с клавиатуры строки матрицы на элемент, являющийся "*седловой*" точкой, т.е. например, минимальным элементом из максимальных элементов по строкам. Для решения такой задачи следует определить значение "седловой" точки. Это можно сделать, организовав поиск максимального элемента по каждой строке и занесение его в элемент соответствующий индексу этой строки одномерного массива. Далее найти минимальный элемент в получившемся массиве, который и будет являться "седловой" точкой. После алгоритм становится достаточно не сложным. Необходимо ввести с клавиатуры индекс строки для преобразования и в цикле по столбцам преобразовать все элементы заданной строки, умножив их значение на полученный элемент "седловой" точки.

# Операции с элементами матриц

```
Const
      n=7;
      m=6;
      Var
i, j, max, sedlt, t: integer;
a : array [1..n,1..m] of integer;
b : array [1..n] of integer;
Begin
randomize;
writeln ('Сформирована матрица A');
for i:=1 to n do
begin
      for j:=1 to m do
        begin
        a[ i,j ]:=random(101);
        write (a[ i,j ]:6);
        end;
writeln;
end;
writeln ('Введите номер строки матрицы для
ее умножения ее на значение 'седловой'
точки ');
readln(t);
for i:=1 to n do
        begin
        max:=a[i][1];
        for j:=2 to m do
                if (a[ i,j ]>max)
                        max:=a[ i,j ];
        b[ i ]:=max;
        end;
sedlt:=b[1];
for i:=1 to n do
        if (b[ i ]<sedlt)
                sedlt=b[ i ];
for j:=1 to m do
        a[ t,j ]:=a[ t,j ]*sedlt;
printf ('Получена матрица ');
for i:=1 to n do
        begin
        for  j:=1 to m do
                write (a[ i,j ]:5);
        writeln;
        end
End.
```

# Преобразование матрицы в одномерный массив

Обработка одномерных статических массивов осуществляется быстрее, чем двумерных того же размера, что часто требует выполнения указанного преобразования. Требуется переслать элементы матрицы n*m в одномерный массив того же размера по строкам с сохранением порядка следования элементов.

Для этого нужно соответствующим образом согласовать индексы исходной матрицы A и формируемого одномерного массива X.

# Преобразование матрицы в одномерный массив

```
Const
      n=3;
      m=4;
      Var
i, j : integer;
a : array [1..n,1..m] of integer;
x : array [1..n*m] of integer;
Begin
randomize;
writeln ('Сформирована матрица A');
for i:=1 to n do
begin
      for j:=1 to m do
        begin
        a[ i,j ]:=random(101);
        write (a[ i,j ]:6);
        end;
writeln;
end;
for i:=1 to n do
        for j:=1 to m do
                x[ (i-1)*m+j ]=a[ i,j ];
writeln ('Получен массив X');
for i:=1 to n*m do
        write (x[ i ]:6);
End.
```

# Многомерные массивы

**Массивы могут быть более чем двумерными.**

**Пример:**

```
...
a : array [1..5, 1..3, 1..16,  1..4 ] of real;
...
for i:=1 to 5 do
     for j:=1 to 3 do
      for k:=1 to 16 do
            for m:=1 to 4 do
                  a[ i,j,k,m ]:=random(101);
...
```