

# Многомерные массивы

- В математике часто используются многомерные массивы.
- Для реализации их в Паскале достаточно задать массив, компонентами которого также являются массивы.
- В этом случае при задании массива базовый тип тоже массив.
- Так можно задать массив любой размерности.

- Например:  
**var Z : array [ 1..4 ] of array [ 1..5 ] of real** -  
задает матрицу ( двумерный массив ) из  
20 вещественных чисел.
- Это же более компактно можно  
записать так: **var Z : array [1..4, 1..5] of  
real.**

**Z**

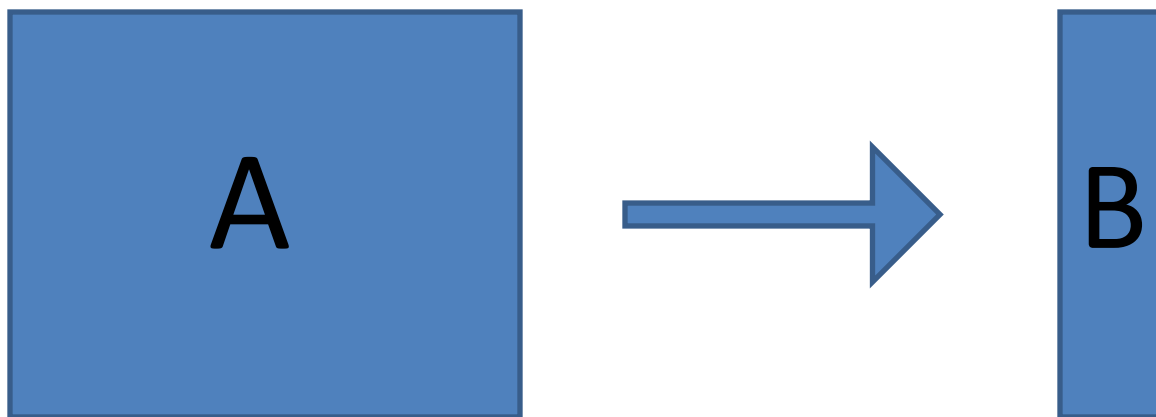
z[1,1]	z[1,2]	z[1,3]	z[1,4]	z[1,5]
z[2,1]	z[2,2]	z[2,3]	z[2,4]	z[2,5]
z[3,1]	z[3,2]	z[3,3]	z[3,4]	z[3,5]
z[4,1]	z[4,2]	z[4,3]	z[4,4]	z[4,5]

- Для обращения к элементам такого массива надо написать  $Z[i,j]$ .
- То есть при обращении к элементу массива, число индексов должно быть равно числу измерений массива.
- В качестве индекса может быть любое выражение получающее значения типа индекса.

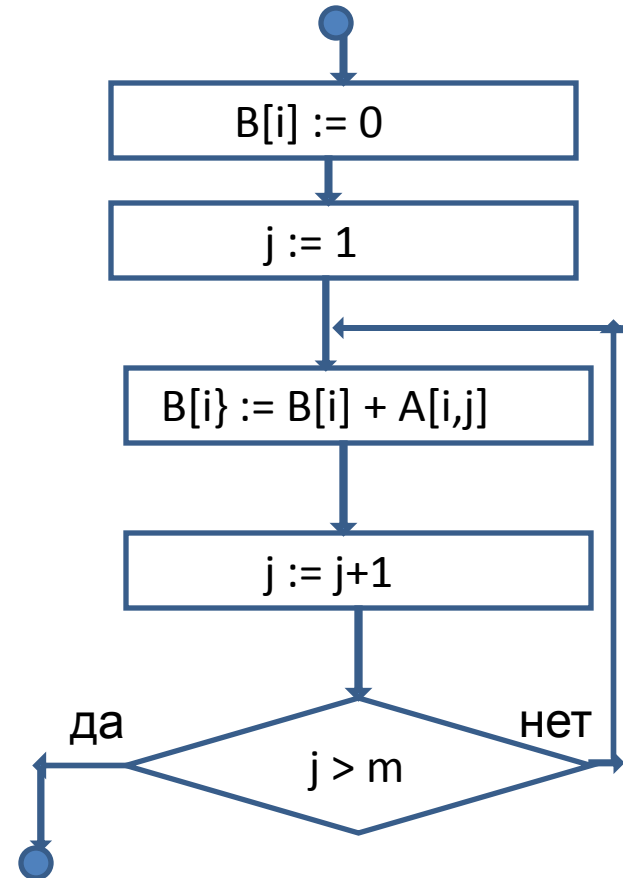
## Пример.

Задана матрица размером  $n \times m$ .

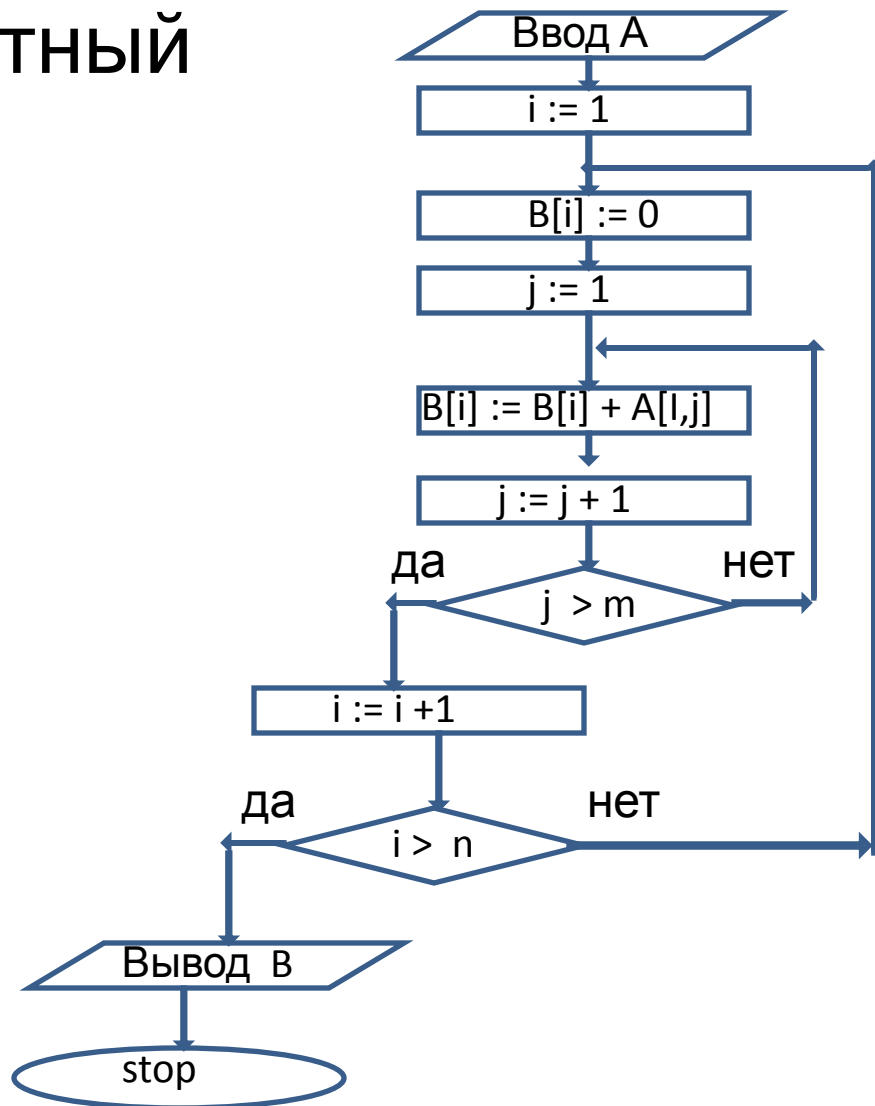
Найти построчные суммы всех строк матрицы.



$$B_i = \sum_{j=1}^m A_{i,j}$$



# Двукратный цикл



```
CONST n=5;
        m=5;
VAR A : array[1..n,1..m] of real;
        B : array[1..n] of real;
        i,j:integer;
Begin
    for i:= 1 to n do    { ввод матрицы }
        begin
            for j:=1 to m do
                read(A[i,j]);
            readln;
        end;
    for i:= 1 to n do    { цикл для перебора строк }
        begin
            B[i]:=0;
            for j:=1 to m do    { суммирование строки }
                B[i]:=B[i] + A[i,j];
            end;
        for i:=1 to n do    { вывод одномерного массива }
            write(B[i]:5:1);
        readln;
    end.
```



Пример.

Задана матрица  $X$  из целых чисел.

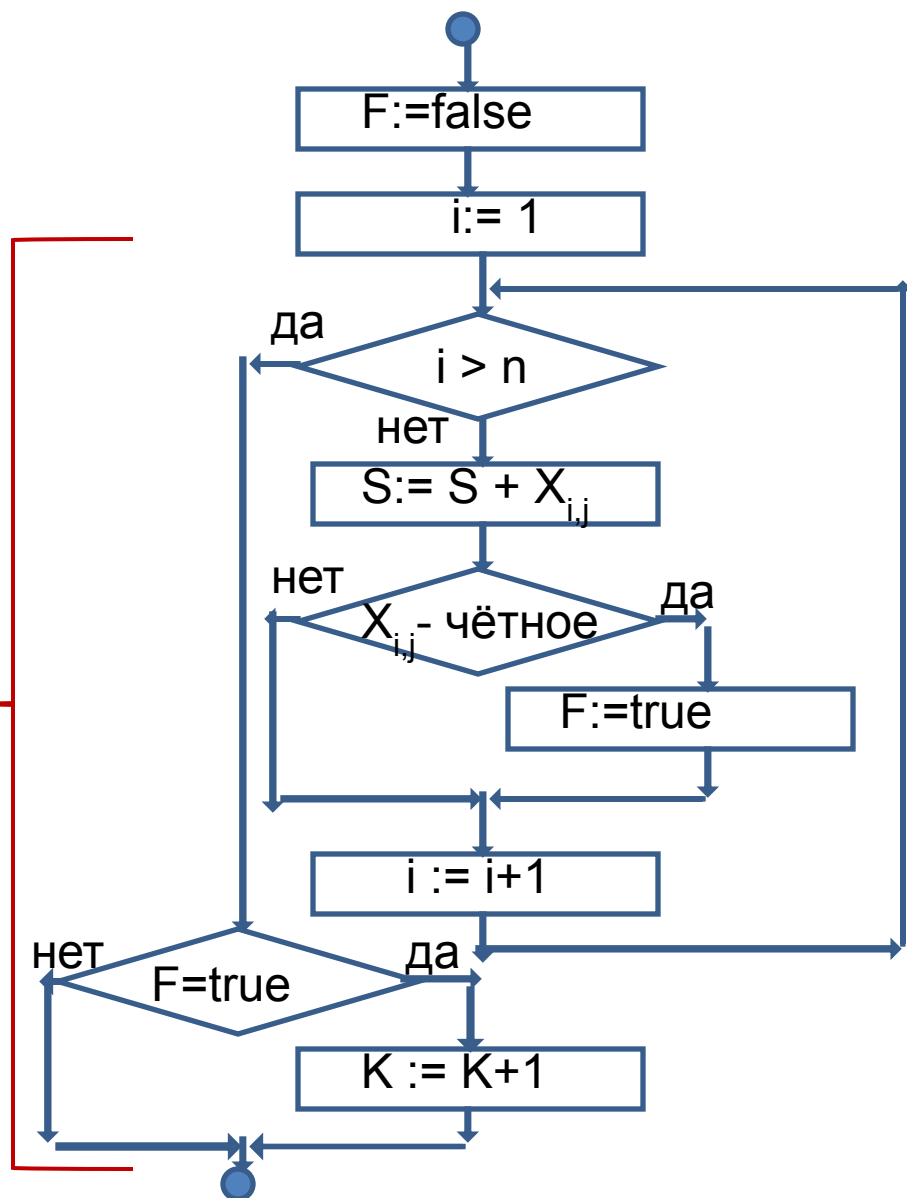
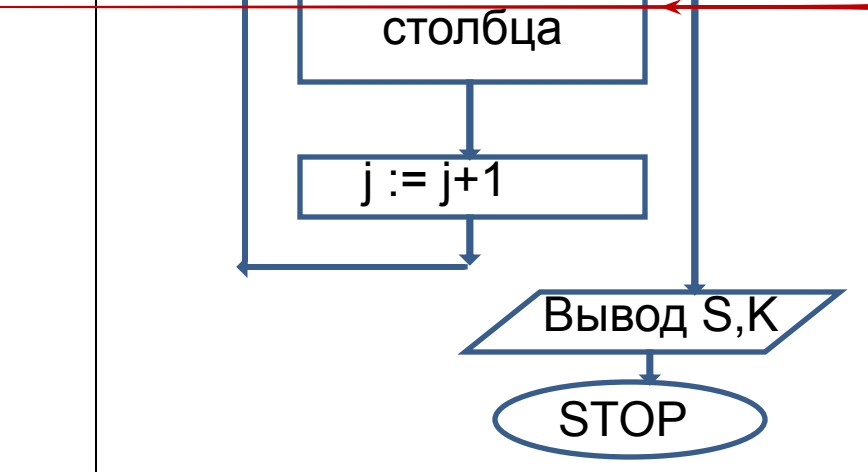
Определить в скольких столбцах матрицы встречаются чётные числа и найти сумму элементов матрицы.

## Метод решения.

Будем просматривать матрицу по столбцам и суммировать элементы.

Для подсчёта количества столбцов, имеющих чётные элементы введём булевскую переменную **F**, которой будем присваивать значение **TRUE**, если в столбце есть чётный элемент, и значение **FALSE**, если таких элементов нет.

Обозначим сумму матрицы через **S**, а количество столбцов с чётными элементами **K**.



```
const n=10;
      m=5;
var X: array[1..n,1..m] of integer;
      S,K,i,j: integer;
      F : boolean;
begin
  for i:=1 to n do
    for j:=1 to m do readln(X[i,j]);
  K:=0;
  S:=0;
  for j:=1 to m do
    begin
      F:= false;
      for i:=1 to n do
        begin
          S:=S + X[i,j];
          if X[i,j] mod 2 = 0 then F:= true;
        end;
      if F then K:= K+1;
    end;
  Writeln('s=',S, ' k=', K);
end.
```

# Комбинированные типы (записи)

- Переменная комбинированного типа содержит фиксированное число полей.
- Каждое поле содержит некоторое значение.
- Тип значения поля может быть любым, кроме файлового.

Для задания переменной типа записи  
используется описатель **RECORD** за ним  
следует перечисление полей записи и  
потом - **END**.

Пример.

TYPE z = **RECORD**

    a:string[20];

    b: Integer;

**END;**

VAR st : z;

    f : **RECORD**

        g,h:real;

        b: boolean;

**END;**

Для обращения к полям записи  
необходимо написать имя переменной  
типа записи и через точку имя поля.

Например:

**st.a**      или      **f.g** .

Такое обращение может стоять в любом  
месте, где допустима переменная такого  
типа.



Пример.

Задана таблица со сведениями о  
полученных в магазин товаров:

Название	Количество	Цена единицы товара
...	...	...
...	...	...
...	...	...
...	...	...

Вводится название товара, подсчитать, на  
какую сумму получено этого товара.

- Исходную таблицу представим как массив записей.
- Запись будет состоять из трёх полей: символьного для названия товара, целочисленного для количества и вещественного для цены.
- Исходную таблицу и товар, суммарную цену которого надо подсчитать, введём с клавиатуры.
- Метод решения состоит из перебора строк таблицы.

```
CONST N = 10;
TYPE  tov = record
        nt : string[20];
        kol : integer;
        c : real;
    end;
VAR tab : array[1..n] of tov;
    t : string(20);
    sum : real;
    i : integer;
begin
    for i := 1 to n do
        readln(tab[i].nt, tab[i].kol, tab[i].c);
        readln( t );
        sum:=0;
        for i := 1 to n do
            if tab[i].nt = t then sum:=sum + tab[i].kol * tab[i].c;
        writeln('sum=', sum);
    end.
```