

# **Примеры разработки программ-функций в системе MATHCAD**

*Лекция 7*

## Пример: задача 1.4.2.30

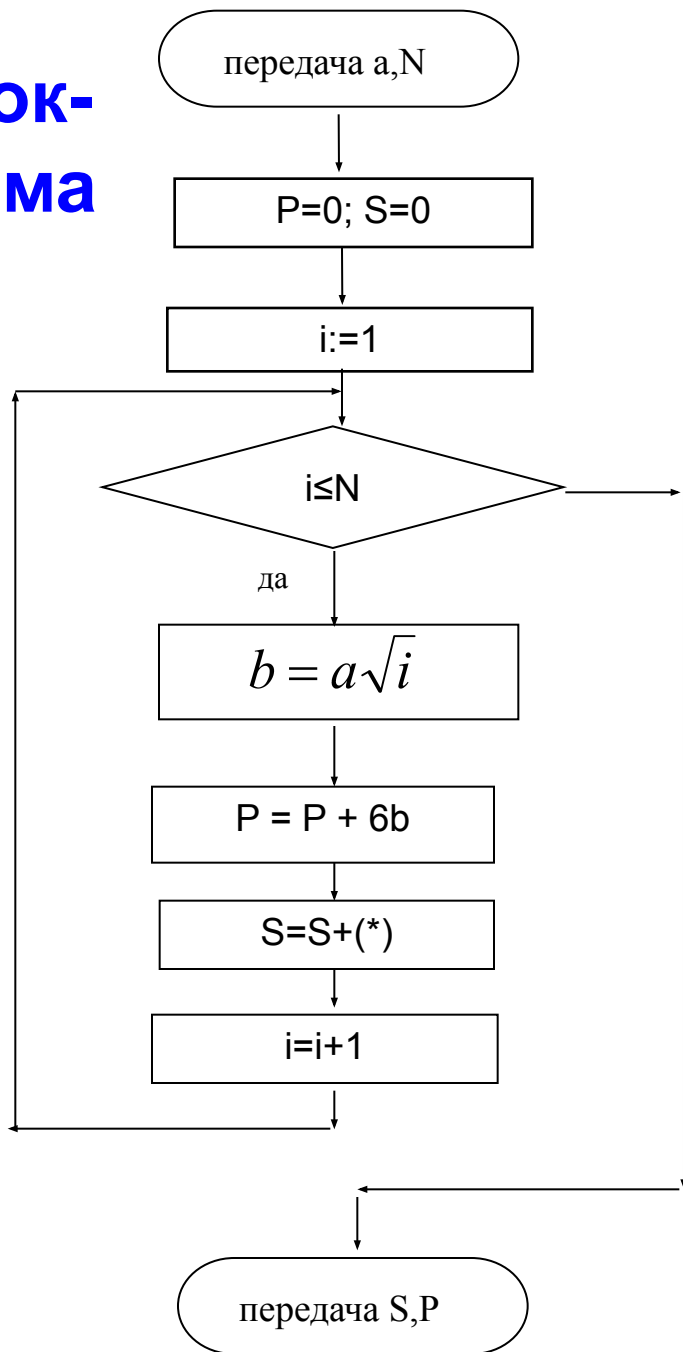
### Условие

Найти суммарную площадь и сумму периметров  $N$  правильных шестиугольников, длины сторон которых образуют последовательность:  $a, a\sqrt{2}, \dots, a\sqrt{N}$ .

### Состав данных

Имя	Смысл	Тип	Структура
<u>Исходные данные</u>			
a	Длина стороны 1-го шестиугольника	Веществ.	Простая переменная
N	Количество шестиугольников	Целый	Простая переменная
<u>Выходные данные</u>			
P	Сумма периметров шестиугольников	Веществ.	Простая переменная
S	Сумма площадей шестиугольников	Веществ.	Простая переменная
<u>Промежуточные данные</u>			
i	Счетчик шестиугольников	Целый	Простая переменная
b	Длина стороны текущего шестиугольника	Веществ.	Простая переменная

# Блок- схема



# Определение функции

```
SummaS_P_6_ugol(a,N) := 

|                                                 |
|-------------------------------------------------|
| S ← 0                                           |
| P ← 0                                           |
| for i ∈ 1..N                                    |
| b ← a·√i                                        |
| P ← P + 6·b                                     |
| S ← S + 3· $\frac{\sqrt{3}}{2}$ ·b <sup>2</sup> |
| (S)                                             |
| (P)                                             |


```

# Примеры вызова функции

Z := SummaS\_P\_6\_ugol(1,5)

$$Z = \begin{pmatrix} 21.778 \\ 8.382 \end{pmatrix} \blacksquare$$

$$\text{SummaS\_P\_6\_ugol}(3,6) = \begin{pmatrix} 84.426 \\ 32.495 \end{pmatrix} \blacksquare$$

Периметр шестиугольника:  $p=6b$ ; площадь шестиугольника:

$$s = \frac{3\sqrt{3}}{2} b^2 \quad (*)$$

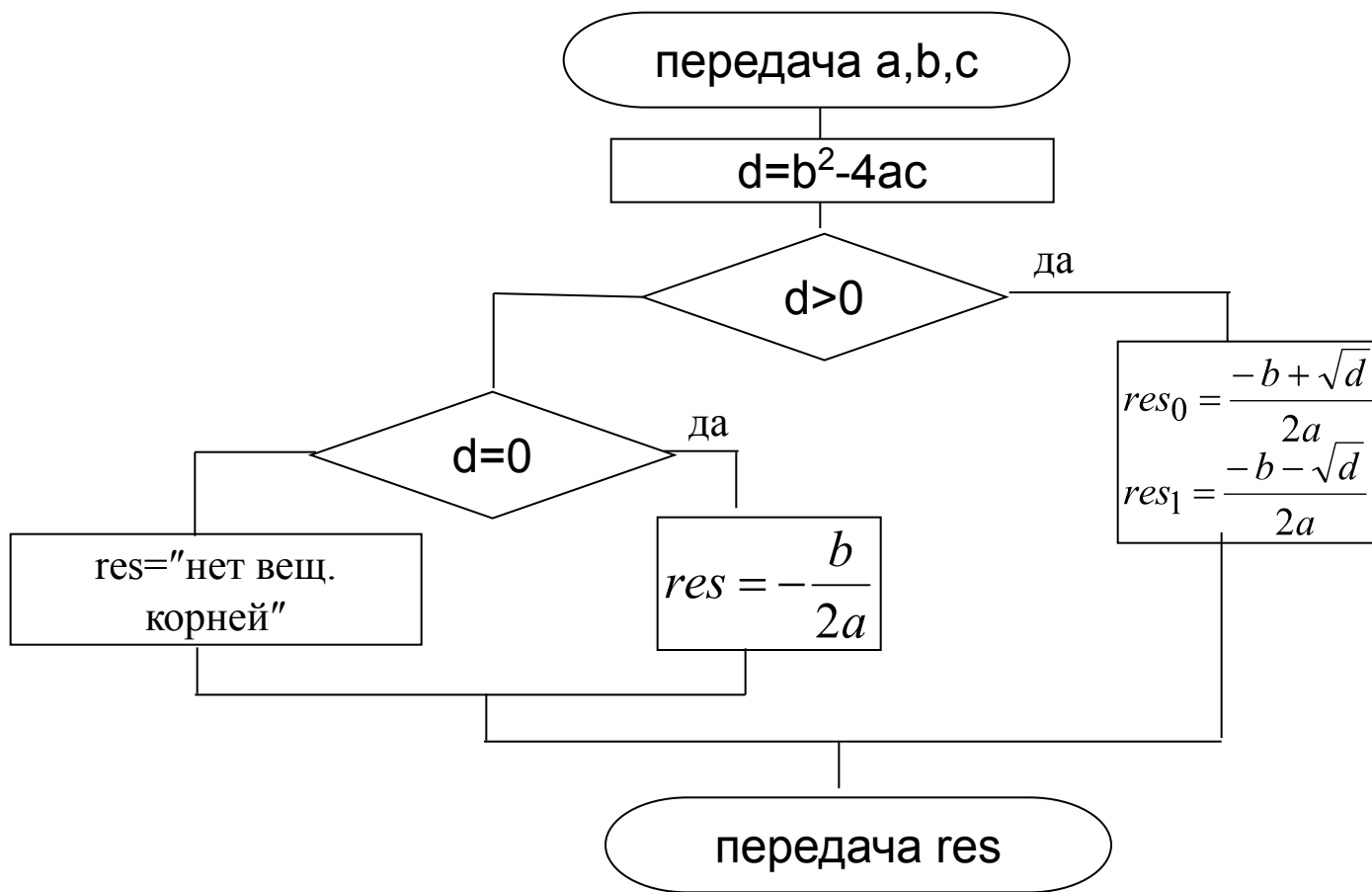
# Найти вещественные корни квадратного уравнения $ax^2+bx+c=0$

## Состав данных

Имя	Смысл	Тип	Структура
<u>Исходные данные</u>			
a			
b	коэффициенты уравнения	Веществ.	Простая переменная
c			
<u>Выходные данные</u>			
res	Корни уравнения	Веществ.	Одномерный массив из двух элементов
	Сообщение об отсутствии вещ. корней	строковый	
<u>Промежуточные данные</u>			
d	Дискриминант уравнения $d=b^2-4ac$	Целый	Простая переменная

*альтернатива*

# Блок-схема алгоритма нахождения вещественных корней квадратного уравнения



# Программа-функция вычисления вещественных корней квадратного уравнения

```
kv_ur1(a,b,c) :=  
  d ← b2 - 4·a·c  
  if d > 0  
    | res0 ←  $\frac{-b + \sqrt{d}}{2 \cdot a}$   
    | res1 ←  $\frac{-b - \sqrt{d}}{2 \cdot a}$   
  otherwise  
    | res ←  $\frac{-b}{2 \cdot a}$  if d = 0  
    | res ← "нет вещественных корней" otherwise  
  res
```

**Обратите внимание:**  
и тип, и структура переменной *res* выбирается динамически, в зависимости от значения *d*

## Примеры вызовов программы-функции kv\_ur1

$kv\_ur1(1, 5, -4) = \begin{pmatrix} 0.702 \\ -5.702 \end{pmatrix}$  ■

$kv\_ur1(1, -2, 5) = \text{"нет вещественных корней"}$  ■

$kv\_ur1(1, -2, 1) = 1$  ■

# Пример решения задачи со сложным условием продолжения цикла

## *Условие*

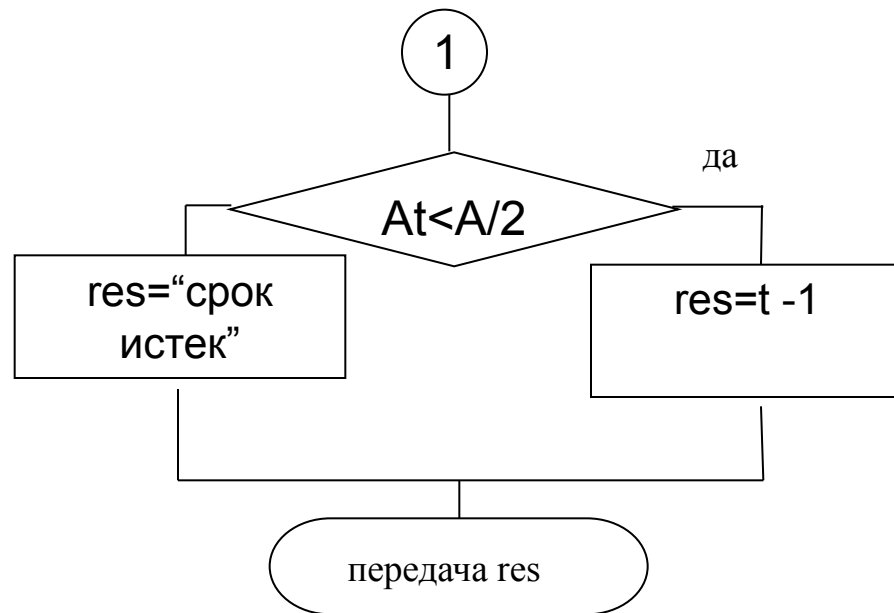
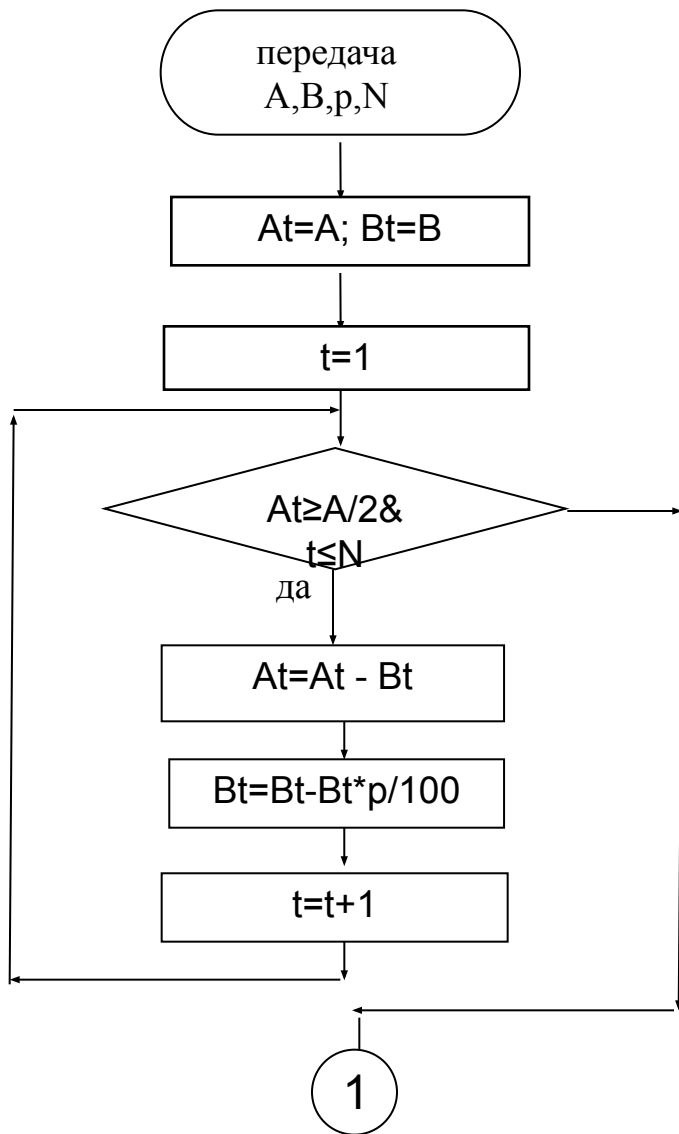
$A$  - начальная стоимость оборудования. В первый год эксплуатации стоимость оборудования снижается на  $B$  руб., а в каждый следующий год снижение стоимости уменьшается на  $p$  % (относительно предыдущего года). Определить, через сколько лет стоимость оборудования станет меньше  $A/2$ . Рассматривать срок не более  $N$  лет.

# Состав данных

Имя	Смысл	Тип	Структура							
<u>Исходные данные</u>										
A	начальная стоимость оборудования	вещественный	простая переменная							
B	начальное снижение стоимости	вещественный	простая переменная							
p	процентное уменьшение снижения стоимости	вещественный	простая переменная							
N	максимальное число лет	целый	простая переменная							
<u>Выходные данные</u>										
res	<table border="0" style="width: 100%;"> <tr> <td style="border: none;">искомый номер года</td> <td style="border: none;">}</td> <td style="border: none;">целый</td> <td style="border: none;">}</td> </tr> <tr> <td style="border: none;">сообщение «срок истек»</td> <td style="border: none;">}</td> <td style="border: none;">строковый</td> <td style="border: none;">}</td> </tr> </table>	искомый номер года	}	целый	}	сообщение «срок истек»	}	строковый	}	простая переменная
искомый номер года	}	целый	}							
сообщение «срок истек»	}	строковый	}							
<u>Промежуточные переменные</u>										
t	текущий номер года	целый	простая переменная							
At	текущая стоимость оборудования	вещественный	простая переменная							
Bt	текущее ежегодное снижение стоимости	вещественный	простая переменная							



# Блок-схема алгоритма



# Программа-функция

```
z1_3_4(A,B,p,N):= At←A
                   Bt←B
                   t←1
                   while (At ≥  $\frac{A}{2}$ ) ^ (t ≤ N)
                       | At←At-Bt
                       | Bt←Bt+0.01·Bt·p
                       | t←t+1
                   res ← t-1 if At <  $\frac{A}{2}$ 
                   res ← "срок прошел" otherwise
                   res
```

## Вызов функции:

$z1\_3\_4(1000,100,1,10)=5$      $z1\_3\_4(1000,100,1,10)="срок\ прошел"$

$y:=z1\_3\_4(1000,100,20,10)$      $y=3$

# Решение задачи методом флажка

Промежуточная переменная:

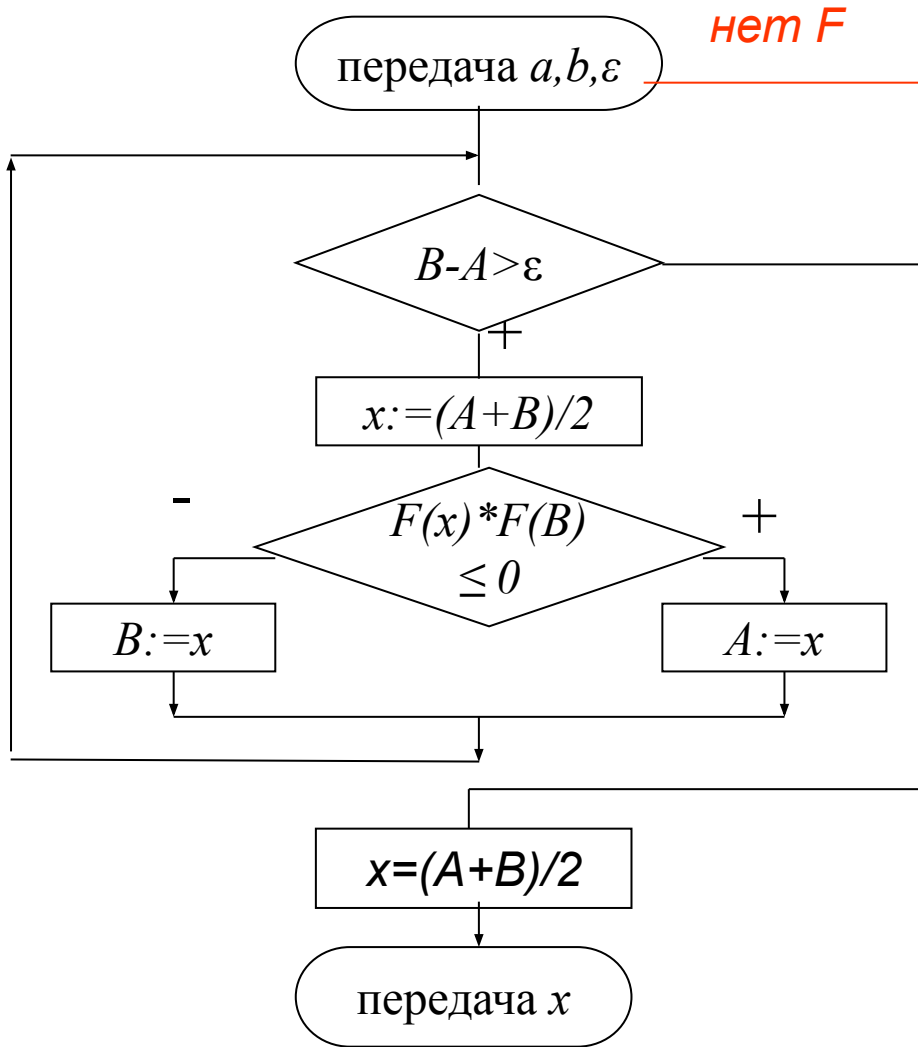
$$f = \begin{cases} 1, & \text{если цикл надо продолжить} \\ 0, & \text{если надо выйти из цикла} \end{cases}$$

# Блок-схема алгоритма с использованием метода флажка





# Решение уравнения методом деления отрезка пополам



*Как в системе Mathcad определить функцию, параметром которой является функция?*

# Программа-функция

$$F(x) := x^2 - 0.25$$

```
Root1(a,b,e) := | while b-a > e
                  |   | x ← (a+b)/2
                  |   | a ← x if F(x) · F(b) ≤ 0
                  |   | b ← x otherwise
                  |   | x ← (a+b)/2
                  |   | x
```

**Функция Root1  
вызывает  
только  
функцию F!  
Нет  
возможности  
подставить  
другую  
функцию!**

**Вызов программы-функции:**

$$\text{Root1}(0.1, 1, 10^{-3}) = 0.5$$

# Встроенная функция `root`

*скаляры*

**`root(Выражение, Имя_переменной)`** – возвращает значение переменной, при котором выражение равно 0, с точностью (погрешностью), задаваемой системной переменной `TOL`.

Функция **`root`** решает уравнение:

**`Выражение=0`**

Например,

**`F(x)=0`**

Или

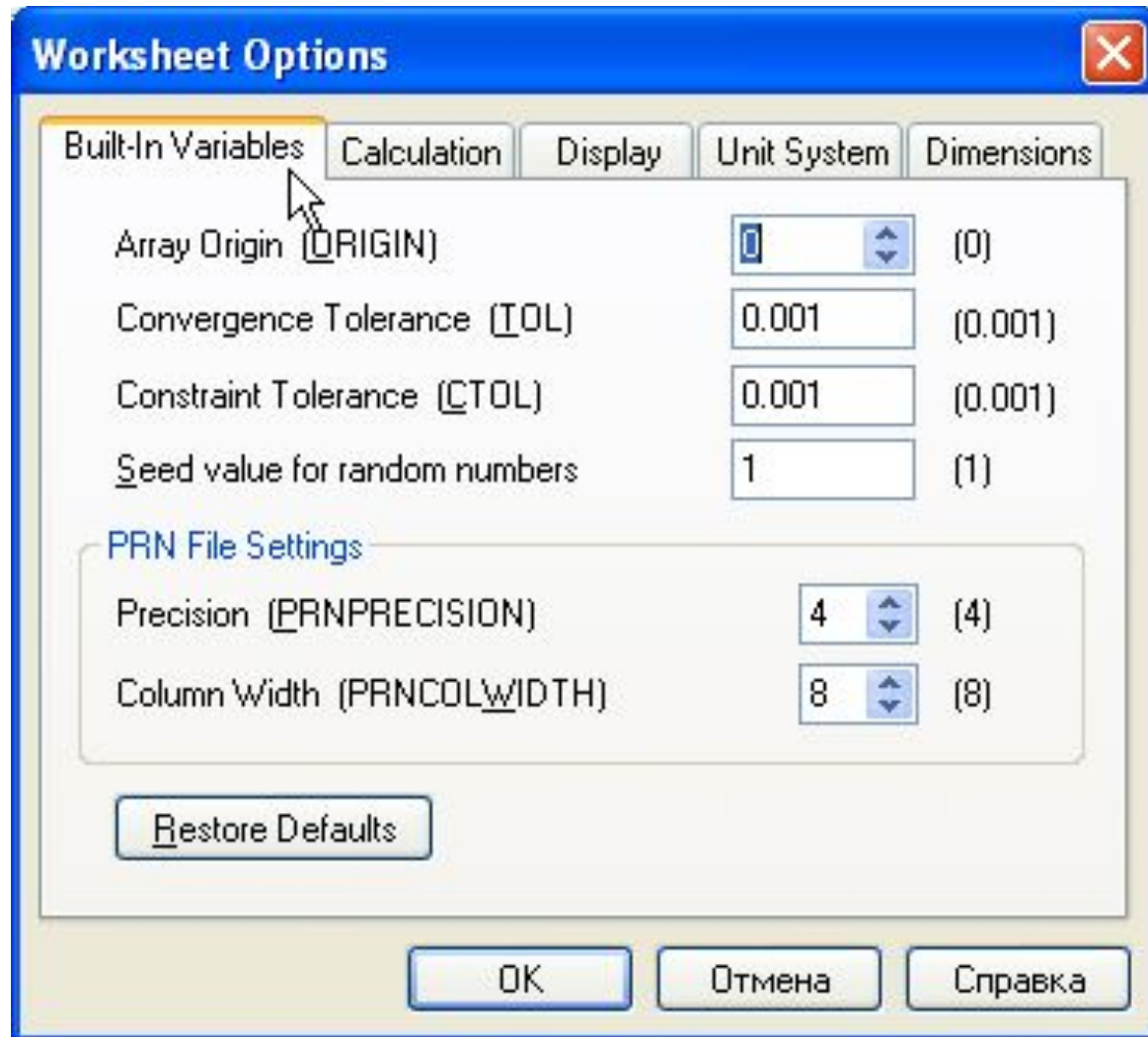
**`x3-6x2+21x-52=0`**

До вызова функции `root` переменной `x` должно быть присвоено начальное значение (начальное приближение корня).



# Установка значений встроенных переменных

Tools / Worksheet Options / Built-in Variables (Сервис / Опции документа / Встроенные переменные)



# Примеры решения уравнения с помощью root

## Пример 1

$$F(x) := x^3 - 6 \cdot x^2 + 21 \cdot x - 52$$

$$x := 0$$

$$x1 := \text{root}(F(x), x) \quad x1 = 4$$

root может вычислять как вещественные, так и комплексные корни

## Пример 2

$$G(a, x) := \text{root}(\exp(x) - a \cdot x^2, x)$$

$$a := 1..5$$

$$x_0 := 0 \quad x_a := G(a, x_{a-1})$$

1
2
3
4
5

x=
0
-0.704
-0.54
-0.459
-0.408
-0.371

# Примеры программ с массивами

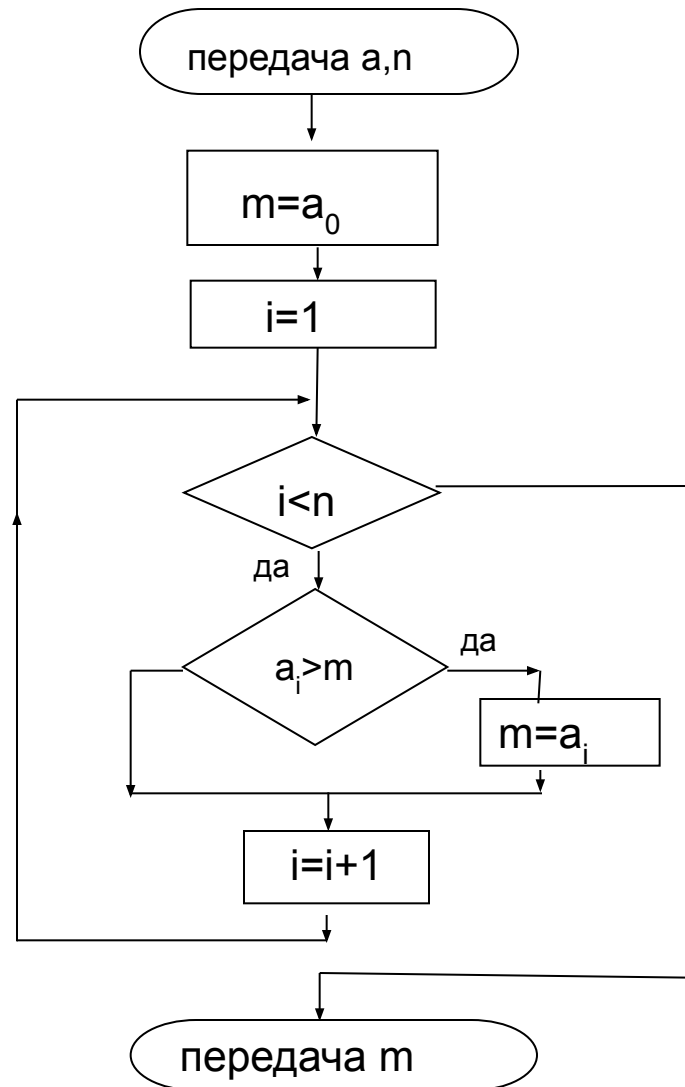
Дан массив **a** из **n** элементов. Найти максимальное значение элементов массива.

## Состав данных

Имя	Смысл	Тип	Структура
<u>Исходные данные</u>			
n	число элементов массива	целый	простая переменная
a	заданный массив	вещественный	одномерный массив
<u>Выходные данные</u>			
m	максимальное значение элементов массива	вещественный	простая переменная
<u>Промежуточные данные</u>			
i	счетчик элементов массива	целый	простая переменная

# Блок-схема алгоритма

Системная (встроенная)  
переменная  $ORIGIN=0$



## Код программы-функции

$$\text{mx}(a, n) := \left\{ \begin{array}{l} m \leftarrow a_0 \\ \text{for } i \in 1..n-1 \\ \quad m \leftarrow a_i \text{ if } m < a_i \\ m \end{array} \right.$$

## Вызов программы-функции

$$a = \begin{pmatrix} 2 \\ -1 \\ 6 \\ 0 \end{pmatrix} \blacksquare$$

$$\text{mx}(a, 4) = 6 \blacksquare$$

С помощью панели  
«Матрицы»

# Примеры программ с массивами

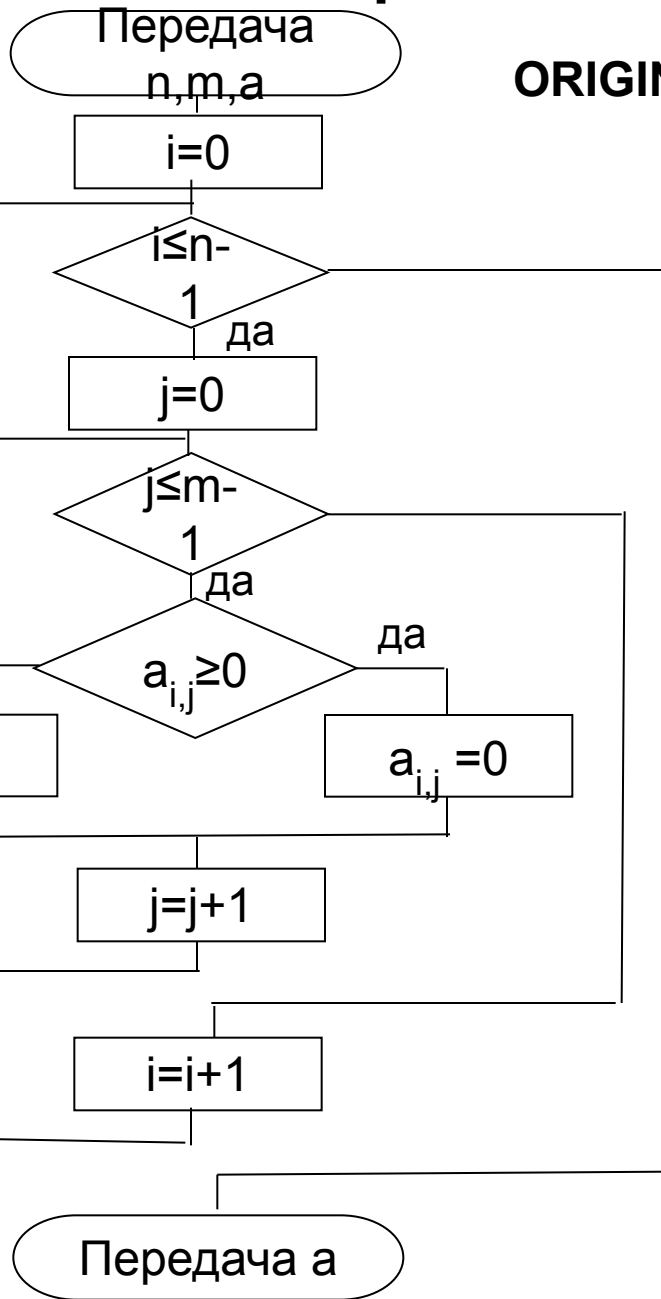
Дана матрица  $a$  из  $n$  строк и  $m$  столбцов.

Неотрицательные элементы матрицы заменить на 0, а отрицательные – на 1.

## Состав данных

Имя	Смысл	Тип	Структура
<u>Исходные данные</u>			
$n$	число строк матрицы	целый	простая переменная
$m$	число столбцов матрицы		число строк матрицы
$a$	заданная матрица	вещественный	двумерный массив
<u>Выходные данные</u>			
$a$	Измененная матрица	вещественный	двумерный массив
<u>Промежуточные данные</u>			
$i$	счетчик строк	целый	простая переменная
$j$	счетчик столбцов	целый	простая переменная

# Блок-схема алгоритма



ORIGIN=0

# Программа-функция и ее

**ВЫЗОВ**

zamena(n,m,a):=

for i ∈ 0..n-1	
for j ∈ 0..m-1	
a <sub>i,j</sub> ← 0	if a <sub>i,j</sub> ≥ 0
a <sub>i,j</sub> ← 1	otherwise
a	

$$D := \begin{pmatrix} -3 & 0 & 2 \\ 1 & -2 & 3 \\ 1 & 2 & 3 \end{pmatrix}$$

T:=zamena(3,3.D)

$$T = \begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 0 \end{pmatrix}$$

# Встроенные функции, определяющие размеры массивов

`rows(A)` – возвращает число строк в матрице **A**. Если **A** – скаляр, возвращается 0.

`cols(A)` – возвращает число столбцов в матрице **A**. Если **A** – скаляр, возвращается 0.

`length(v)` – возвращает число элементов вектора **v**.

`last(v)` – возвращает индекс последнего элемента в векторе **v**.



# Улучшенный код

```
Fun2(a) := | res ← 0
            | n ← rows(a)
            | m ← cols(a)
            | if ORIGIN= 0
            |   | i1 ← 0
            |   | j1 ← 0
            |   | n1 ← n - 1
            |   | m1 ← m - 1
            | otherwise
            |   | if ORIGIN= 1
            |   |   | i1 ← 1
            |   |   | j1 ← 1
            |   |   | n1 ← n
            |   |   | m1 ← m
            |   | res ← "error!" otherwise
            | for i ∈ i1.. n1
            |   for j ∈ j1.. m1
            |     | ai,j ← 0 if ai,j ≥ 0
            |     | ai,j ← 1 otherwise
            | res ← a if res = 0
            | res
```

⊕1. Нет зависимости от значения ORIGIN.

⊕2. Нет необходимости задавать размеры матрицы

# Еще более совершенный код

```
Fun3(a):= | n←rows(a)
           | m←cols(a)
           | i1 ←ORIGIN
           | n1←n+ORIGIN-1
           | j1 ←ORIGIN
           | m1←m+ORIGIN-1
           | for i∈ i1..n1
           |   for j∈ j1..m1
           |     | ai,j ←0 if ai,j≥0
           |     | ai,j ←1 otherwise
           | a
```

⊕1. Нет зависимости от значения ORIGIN (может быть ORIGIN≠0, ORIGIN≠1).

⊕2. Еще короче.

# Цикл for «для каждого элемента массива»

$$\text{Sum}(a) := \left| \begin{array}{l} s \leftarrow 0 \\ \text{for } v \in a \\ \quad | \quad s \leftarrow s + v \\ s \end{array} \right.$$

**НО: изменение  $v$  не приводит к изменению элемента матрицы!**

$$x = \begin{pmatrix} 1 \\ 4 \\ 5 \end{pmatrix} \quad A = \begin{pmatrix} -1 & -2 & -3 \\ 1 & 2 & 3 \\ 1 & 2 & 3 \end{pmatrix}$$

$$\text{Sum}(x) = 10$$

$$\text{Sum}(A) = 6$$

**Одна функция и для вектора, и для матрицы**

# Выделение подпрограмм в процессе решения задачи

## Условие

Проверить, упорядочены ли по возрастанию элементы третьего столбца матрицы  $A(4 \times 6)$  и элементы пятого столбца матрицы  $B(5 \times 5)$

*Решали эту задачу на языке MATLAB*

**1 шаг решения:** выбираем, какую подпрограмму (в MATNCAD программу-функцию) надо разработать: подпрограмма должна вызываться несколько раз с разными данными.

**Для нашей задачи:** функция определяет, упорядочены ли по возрастанию  $q$ -го столбца произвольной матрицы  $X$ .

**Вход функции:** матрица  $X$ , номер столбца  $q$ .

**Выход функции :**  $F$  – признак упорядоченности элементов  $q$ -го столбца:

$$F = \begin{cases} 1, & \text{если элементы } q \text{-го столбца упорядочены;} \\ 0, & \text{если нет.} \end{cases}$$

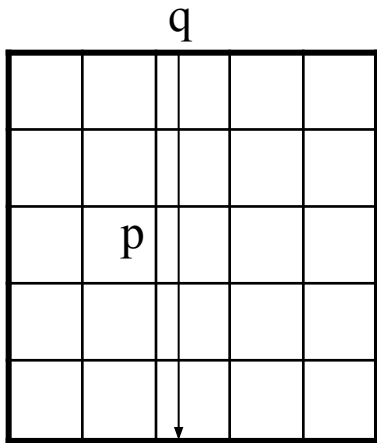
**2 шаг - Пишем заголовок функции:**

`upor_stolb(X,q):=`

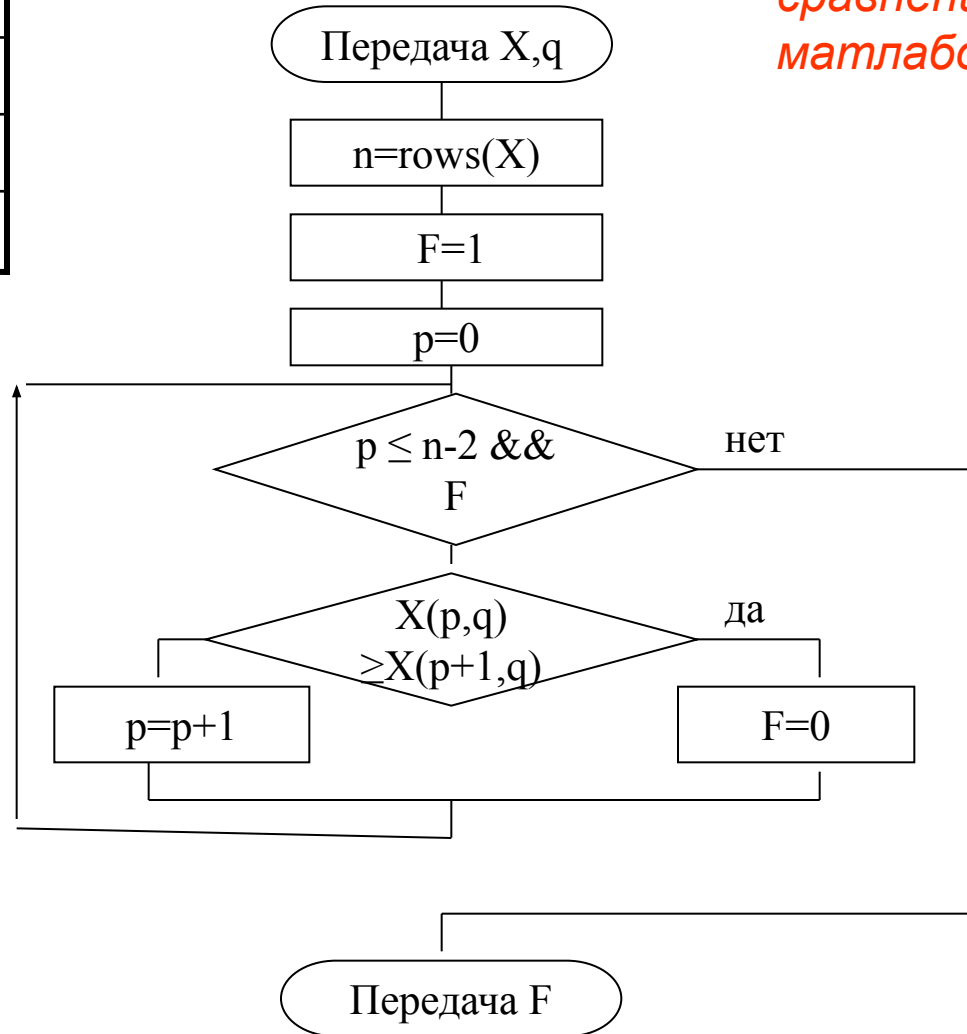
# 3 шаг. Состав данных функции upor\_stolb

Имя	Смысл	Тип	Структура
<u>Входные данные (передаваемые из вызывающего кода в функцию)</u>			
X	Заданная матрица	вещественный	двумерный массив
q	Номер столбца	целый	простая переменная
<u>Выходные данные (передаваемые из функции вызывающий код)</u>			
F	Признак упорядоченности столбца	целый (логический)	простая переменная
<u>Промежуточные данные</u>			
n	Число строк матрицы	целый	простая переменная
m	Число столбцов матрицы	целый	простая переменная
p	Счетчик строк матрицы	целый	простая переменная

# 4 шаг. Блок-схема алгоритма функции



*Что исправили по сравнению с матлабом?*



## 5 шаг. Код функции upor\_stolb

```
Upor_stolb(X,q):= | n ← rows(X)
                   |
                   | F ← 1
                   |
                   | p ← 0
                   |
                   | while (p ≤ n-2) ^ F
                   |   | F ← 0 if  $X_{p,q} > X_{p+1,q}$ 
                   |   | p ← p+1 otherwise
                   |   |
                   |   | F
                   |
```

*Вызов функции – самостоятельно!*



# **Следующая лекция: локальные и глобальные переменные**