

МОДУЛЬНОЕ ПРОГРАММИРОВАНИЕ ИЕ В MATHCAD



Сущность и преимущества модульного программирования

Модульная программа – программа, реализующая алгоритм вычислений в виде модулей и обращения к ним.

Сущность:

- в разбиении алгоритма решения задачи на слабо зависимые фрагменты вычислений (на подзадачи) и реализация каждого фрагмента в виде программных модулей;
- в вызове в нужных местах «основной» программы соответствующих модулей с передачей необходимых данных.

Преимущества:

- возможность распараллеливания разработки программы;
- простота отладки;
- простота сопровождения и модификации;
- возможность использования библиотек «готовых» модулей.



Описание подпрограммы-функции

ПФ предназначены для многократного выполнения фрагментов программы без повторения их записи.

▣ имя подпрограммы-функции

Оригинальное имя, используемое для обращения к ПФ и «возврата» результата ее выполнения.

▣ список формальных параметров

Передаёт данные, необходимые для выполнения вычислений внутри ПФ: переменные, массивы, функции.

Формальные параметры заключаются в круглые скобки и отделяются запятой (список формальных параметров может отсутствовать).

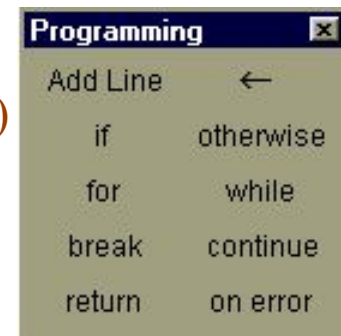
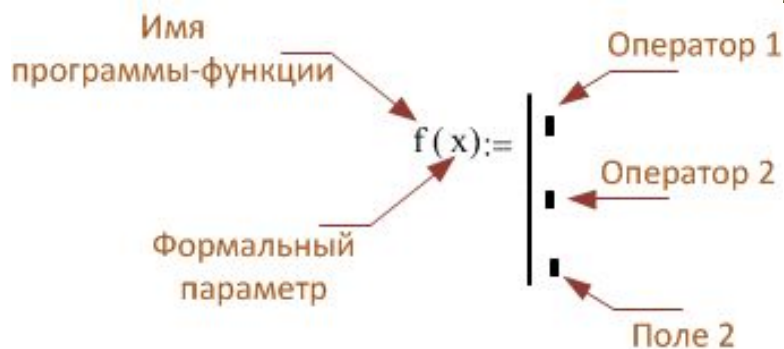
▣ тело подпрограммы-функции

Включает любое число операторов: локальных операторов присваивания, условных операторов и операторов цикла, а также вызовов других ПФ и функций пользователя.



Порядок описания подпрограммы-функции

- Ввести имя ПФ и список формальных параметров, заключенный в круглые скобки.
- Ввести символ присваивания «:=».
- Открыть панель инструментов **Programming** (Программирование) и нажать кнопку **Add Line** (Добавить линию)



- В поле 1 вводится оператор тела ПФ.
- Увеличение числа полей для ввода дополнительных операторов осуществляется с помощью кнопки **Add Line** или с клавиатуры «]»
- Удаление оператора или поля ввода осуществляется нажатием клавиши **Delete**
- Заполнить поле 2 (нижнее поле), введя выражение, определяющее возвращаемое через имя ПФ значение

Локальный оператор присваивания

<Имя переменной> ← <Выражение>

Обращение к подпрограмме функции

< имя П-Ф > (< список фактических параметров >)

Обращение к П-Ф должно находиться после ее описания, и *к моменту обращения фактические параметры должны быть определены.*

$x := 7$

$$f(x) := \begin{cases} x \leftarrow x + 2 \\ z \leftarrow \sqrt{x} \\ z \end{cases}$$

Варианты обращения к ПФ

$f(x) = 3 \quad x = 7$

$f(5) = 2.646$

$z := f(x + 7) \quad z = 4$



Программирование линейных алгоритмов

Под линейным алгоритмом понимается вычислительный процесс, в котором необходимые операции выполняются строго последовательно.

$$\begin{array}{l} \text{root_poly2}(a, b, c) := \\ \text{root_poly2}(a, b, c) := \end{array} \left| \begin{array}{l} d \leftarrow b^2 - 4 \cdot a \cdot c \\ d \leftarrow b^2 - 4 \cdot a \cdot c \\ x1 \leftarrow \frac{-b + \sqrt{d}}{2 \cdot a} \\ x1 \leftarrow \frac{-b + \sqrt{d}}{2 \cdot a}, x2 \leftarrow \frac{-b - \sqrt{d}}{2 \cdot a} \\ x2 \leftarrow \frac{-b - \sqrt{d}}{2 \cdot a} \\ \begin{pmatrix} x1 \\ x2 \end{pmatrix} \\ \begin{pmatrix} x1 \\ x2 \end{pmatrix} \\ \begin{pmatrix} x1 \\ x2 \end{pmatrix} \end{array} \right.$$

$$z := \text{root_poly2}(2, 5, 8) = \begin{pmatrix} -1.25 + 1.561i \\ -1.25 - 1.561i \end{pmatrix}$$

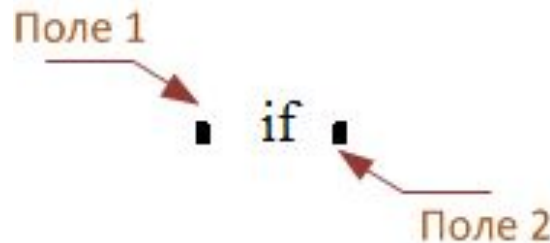


Программирование разветвляющихся алгоритмов

В разветвляющихся алгоритмах присутствует несколько ветвей вычислительного процесса. Выбор конкретной ветви зависит от выполнения (или невыполнения) заданных условий.

Условный оператор *if*

Условная структура ЕСЛИ- ТО



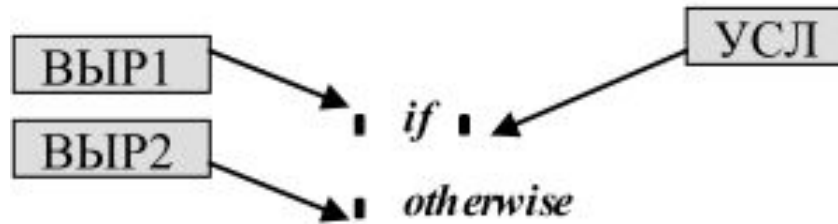
В поле 2 вводится логическое выражение (выражение отношений).

В поле 1 вводится выражение (как правило, арифметическое), значение которого используется, если проверяемое логическое выражение принимает значение 1 (истина)



Программирование разветвляющихся алгоритмов

Условная структура ЕСЛИ- ТО - ИНАЧЕ



ВЫР2 выполняется, если проверяемое логическое выражение (УСЛ) = 0.

Оператор *otherwise* следует непосредственно после условного оператора *if*.

$$y(x) := \begin{cases} \ln(x) & \text{if } x > 0 \\ e^x & \text{otherwise} \end{cases}$$

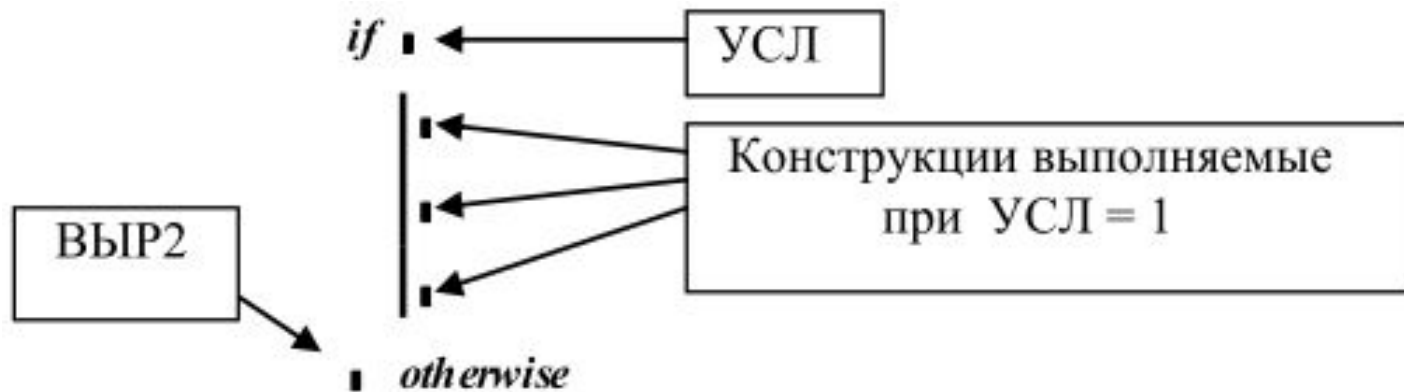
$$y(-3) = 0.05$$

$$y(2.71) = 0.997$$



Программирование сложных разветвляющихся алгоритмов

Вариант 1: При выполнении заданного условия УСЛ необходимо выполнить несколько конструкций

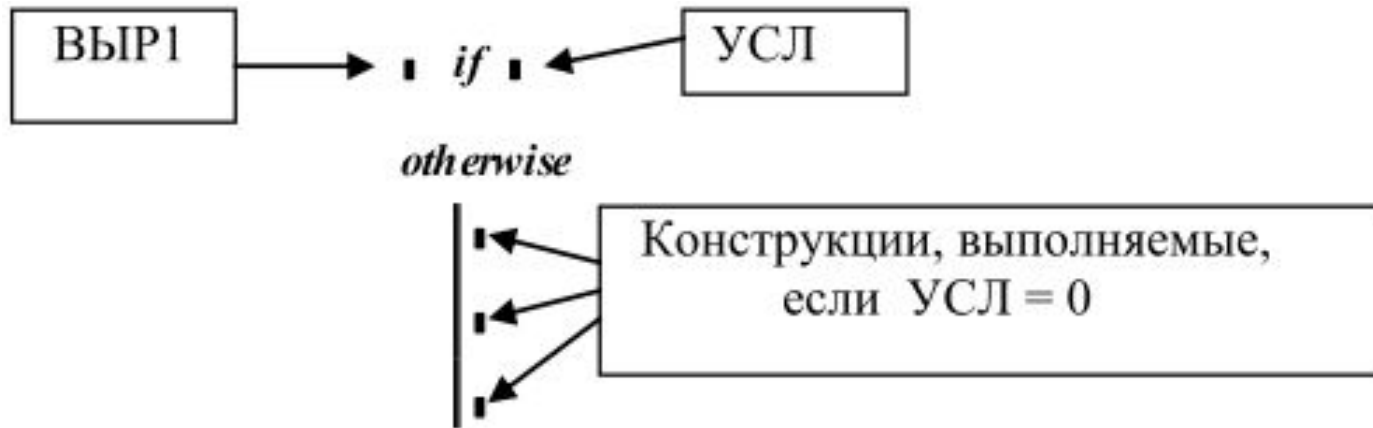


- выделить поле 1 условного оператора *if*
- нажать кнопку *Add line* нужное число раз
- заполнить появившиеся поля



Программирование сложных разветвляющихся алгоритмов

Вариант 2: При невыполнении заданного условия УСЛ необходимо выполнить несколько конструкций




- выделить поле оператора *otherwise*
- нажать кнопку *Add line* нужное число раз
- заполнить появившиеся поля



Программирование сложных разветвляющихся алгоритмов

Составить ПФ, возвращающую значения двух переменных a и b , возведенных во вторую или третью степень. Степень задается переменной n . Если $n < 2$ или $n > 3$, то значения переменных принять равными 0.

$$\text{poly}(n, a, b) := \begin{cases} \text{if } n = 2 & \begin{cases} x \leftarrow a^2 \\ y \leftarrow b^2 \end{cases} & \text{poly}(-1, 2, -3) = \begin{pmatrix} 0 \\ 0 \end{pmatrix} \\ \text{if } n = 3 & \begin{cases} x \leftarrow a^3 \\ y \leftarrow b^3 \end{cases} & \text{poly}(0, 2, -3) = \begin{pmatrix} 0 \\ 0 \end{pmatrix} \\ \text{otherwise} & \begin{cases} x \leftarrow 0 \\ y \leftarrow 0 \end{cases} & \text{poly}(2, 2, -3) = \begin{pmatrix} 4 \\ 9 \end{pmatrix} \\ & & \text{poly}(3, 2, -3) = \begin{pmatrix} 8 \\ -27 \end{pmatrix} \end{cases}$$



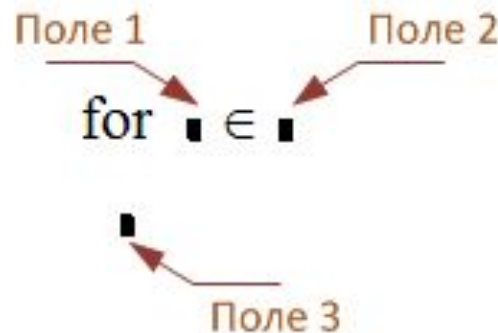
Программирование циклических алгоритмов

Циклические алгоритмы (циклы) содержат повторяющиеся вычисления, зависящие от некоторой переменной.

Такая переменная называется *параметром цикла*, а сами повторяющиеся вычисления составляют *тело цикла*.

Программирование цикла типа арифметической прогрессии

Оператор цикла *for*



- В поле 1 вводится имя переменной, являющейся параметром цикла;
- В поле 2 вводится закон изменения параметра цикла, используя для этого описание дискретной переменной или описание массива;
- В поле 3 вводятся операторы, составляющие тело цикла.

Вычислить значения функции

$$y(x) = \frac{\ln|x|}{a^2 + b^2}$$

для всех x , изменяющихся в интервале $[-0.5, 2.5]$ с шагом $\Delta x = 0.1$;
 a, b – заданные вещественные числа.

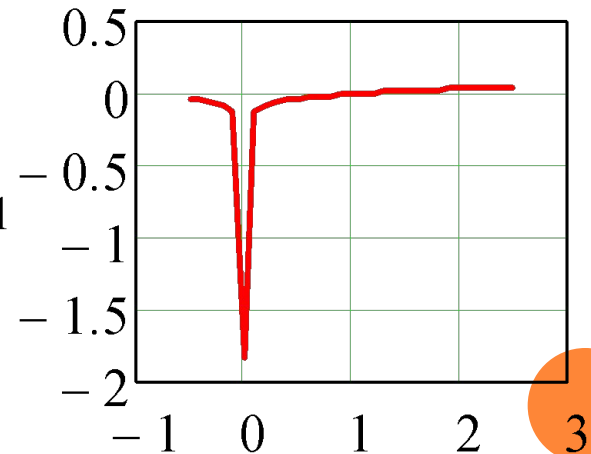
Построить график этой функции

$$x_0 := -0.5 \quad x_k := 2.5 \quad d := 0.1$$

$$y1(a, b, x) := \left| \begin{array}{l} i \leftarrow 0 \\ \text{for } x \in x_0, x_0 + d .. x_k \\ \quad \left| \begin{array}{l} z_i \leftarrow \frac{\ln(|x|)}{a^2 + b^2} \\ X_i \leftarrow x \\ i \leftarrow i + 1 \end{array} \right. \\ \left(\begin{array}{l} X \\ z \end{array} \right) \end{array} \right.$$

$$y1(2, 4, x) = \left(\begin{array}{l} \{31, 1\} \\ \{31, 1\} \end{array} \right)$$

$$y1(2, 4, x)_1$$



$$y1(2, 4, x)_0$$

Составить описание П-Ф, вычисляющей сумму вектора с проекциями |2,5,7|

$sum_vec_3 :=$	$S \leftarrow 0$
	$for\ i \in \begin{pmatrix} 2 \\ 5 \\ 7 \end{pmatrix}$
	$S \leftarrow S + i$
	S
$sum_vec_3 = 14$	

Составить описание П-Ф, вычисляющей сумму следующих слагаемых 2, 3, 5, 9, 11, 17, 21.

$sum_1 :=$	$S \leftarrow 0$
	$for\ i \in 2, 3, 5, 9, 11, 17, 21$
	$S \leftarrow S + i$
	S
$sum_1 = 68$	



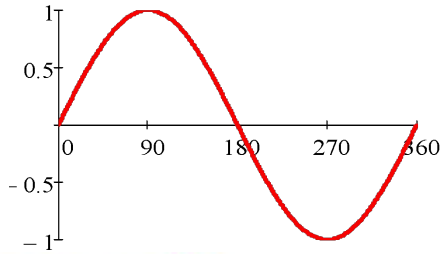
Составить П-Ф, которая формирует двумерный массив из объектов разных типов.

$$\begin{array}{l} \mathit{ORIGIN} := 1 \\ \mathit{Form} := \left| \begin{array}{l} i \leftarrow 1 \\ \mathit{for} \ i_c \in \left(\begin{array}{l} 100.25 \quad \text{"Good"} \\ 20 + 55i \quad \text{"Yes"} \end{array} \right) \\ \left| \begin{array}{l} V_i \leftarrow i_c \\ i \leftarrow i + 1 \end{array} \right. \\ V \end{array} \right. \end{array} \quad \mathit{Form} = \left(\begin{array}{l} 100.25 \\ 20 + 55i \\ \text{"Good"} \\ \text{"Yes"} \end{array} \right)$$

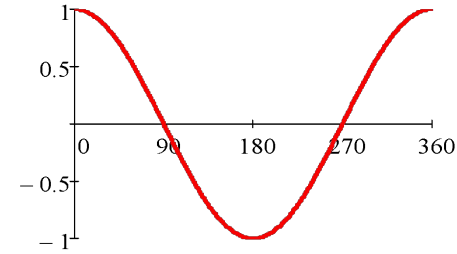

Составить подпрограмму-функцию, осуществляющую построение графика в соответствии с рисунком (выполнить модульное программирование в MathCAD).

Примеры графиков функций $f(t) = \text{Sin}(t)$ (рисунок а) и $f(t) = \text{Cos}(t)$ (рисунок б).

а)

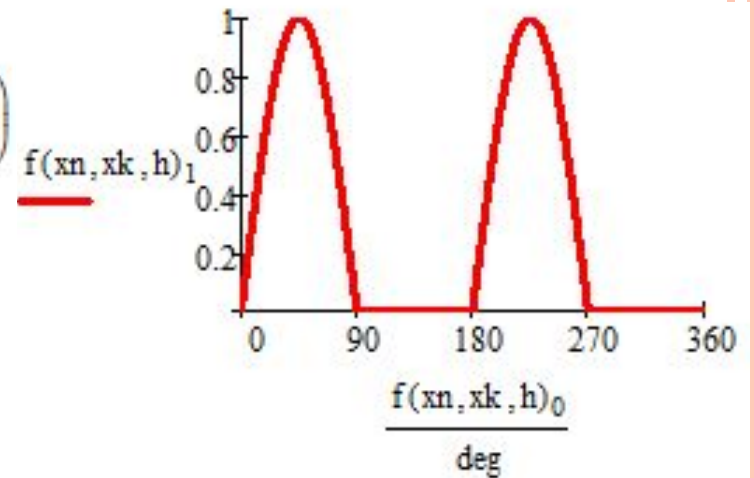


б)



$x_n := 0$ $x_k := 2 \cdot \pi$ $h := 0.001$

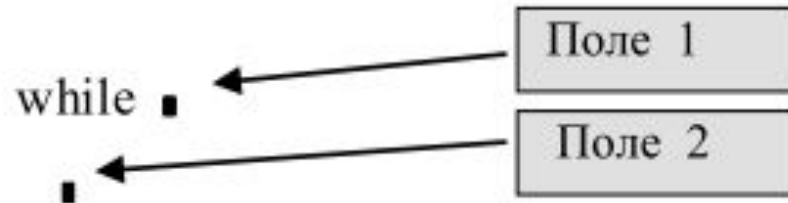
```
f(xn, xk, h) :=
  k ← 0
  for t ∈ xn, xn + h .. xk
    y_k ←
      sin(2t) if (0 ≤ t < π/2) ∨ (π ≤ t < 3·π/2)
      0 otherwise
    x_k ← t
    k ← k + 1
  (
    x
    y
  )
```



Программирование итерационных циклов

Оператор цикла *while*

– внутренние операторы цикла будут исполняться до тех пор, пока будет истинным условие, следующее за ключевым словом *while*.



□ В поле 1 вводится условие выполнения цикла;

□ В поле 2 вводятся операторы тела цикла.

В теле цикла должны присутствовать операторы, которые могут изменить значение условия цикла, иначе цикл будет продолжаться бесконечно.



Решить методом Ньютона

$$f(x) := 5 \cdot \sin(x) - 4 \cdot x^2 + 5.8$$

$$\varepsilon := 10^{-4}$$

$$x_0 := 3$$

```
root_Nyuton := | i ← 1
                 | df(x) ←  $\frac{d}{dx} f(x)$ 
                 |  $x_0 \leftarrow 3, x_1 \leftarrow x_0 - \frac{f(x_0)}{df(x_0)}$ 
                 | while  $|x_i - x_{i-1}| > \varepsilon$ 
                 |   |  $x_{i+1} \leftarrow x_i - \frac{f(x_i)}{df(x_i)}$ 
                 |   | i ← i + 1
                 | x
```

$$\text{root_Nyuton}^T = (3 \quad 1.9811938 \quad 1.6833021 \quad 1.6429796 \quad 1.6421985 \quad 1.6421982)$$

Программирование итерационных циклов

Оператор *break*

— служит для преждевременного завершения цикла, чтобы, например, избежать зацикливания или слишком продолжительных вычислений.

Оператор *break* используется в левом поле ввода условного оператор *if*, а в правом размещается условие, при выполнении которого происходит прекращение работы цикла или программы.



$\text{root_Nyut}(x_0, \varepsilon) :=$ $\left| \begin{array}{l} \text{ierr} \leftarrow 1, x_0 \leftarrow x_0 \end{array} \right.$

$\left. \begin{array}{l} \text{df}(x) \leftarrow \frac{d}{dx} f(x) \end{array} \right|$

$\left. \begin{array}{l} x_1 \leftarrow x_0 - \frac{f(x_0)}{\text{df}(x_0)} \end{array} \right|$

$\left. \begin{array}{l} \text{for } i \in 1..5 \end{array} \right|$

$\left. \begin{array}{l} \left| \text{if } |x_i - x_{i-1}| < \varepsilon \right. \end{array} \right|$

$\left. \begin{array}{l} \left| \left| \text{ierr} \leftarrow 0 \right. \right. \end{array} \right|$

$\left. \begin{array}{l} \left| \text{break} \right. \end{array} \right|$

$\left. \begin{array}{l} \left| x_{i+1} \leftarrow x_i - \frac{f(x_i)}{\text{df}(x_i)} \right. \end{array} \right|$

$\left. \begin{array}{l} x \leftarrow \text{"not solve"} \text{ if } \text{ierr} = 1 \end{array} \right|$

$\left. \begin{array}{l} \left(\begin{array}{l} x \\ \text{ierr} \end{array} \right) \end{array} \right|$

$$\text{root_Nyut}(3, 10^{-4})^T = \begin{bmatrix} 3 \\ 1.9811938 \\ 1.6833021 \\ 1.6429796 \\ 1.6421985 \\ 1.6421982 \end{bmatrix} 0$$

$$\text{root_Nyut}(3, 10^{-10})^T = (\text{"not solve"} \quad 1)$$



Программирование двойных циклов

```
for i ∈ I  
  for j ∈ J  
    ▮
```

```
for i ∈ I  
  while J  
    ▮
```

```
while J  
  for i ∈ I  
    ▮
```

Сформировать матрицу (двумерный массив) B по следующему правилу:

$$B_{i,j} = \frac{1}{i+j+1};$$

ORIGIN := 1

```
form_M(n,m) :=  $\left\{ \begin{array}{l} \text{for } i \in 1..n \\ \quad \text{for } j \in 1..m \\ \qquad B_{i,j} \leftarrow \frac{1}{i+j+1} \end{array} \right.$ 
```

$$\text{form_M}(4,2) = \begin{pmatrix} 0.333 & 0.25 \\ 0.25 & 0.2 \\ 0.2 & 0.167 \\ 0.167 & 0.143 \end{pmatrix}$$

Дополнительные операторы, используемые при программировании циклов

Оператор *continue*

Обычно используется для продолжения выполнения цикла путем возврата в начало тела цикла

Составить описание П-Ф, формирующую новый вектор из положительных элементов исходного вектора.

$\text{form_V}(v) :=$

$i \leftarrow 0, j \leftarrow 0$	
while $i < \text{last}(v)$	
$i \leftarrow i + 1$	
continue if $v_i \leq 0$	
$j \leftarrow j + 1$	
$w_j \leftarrow v_i$	
w	

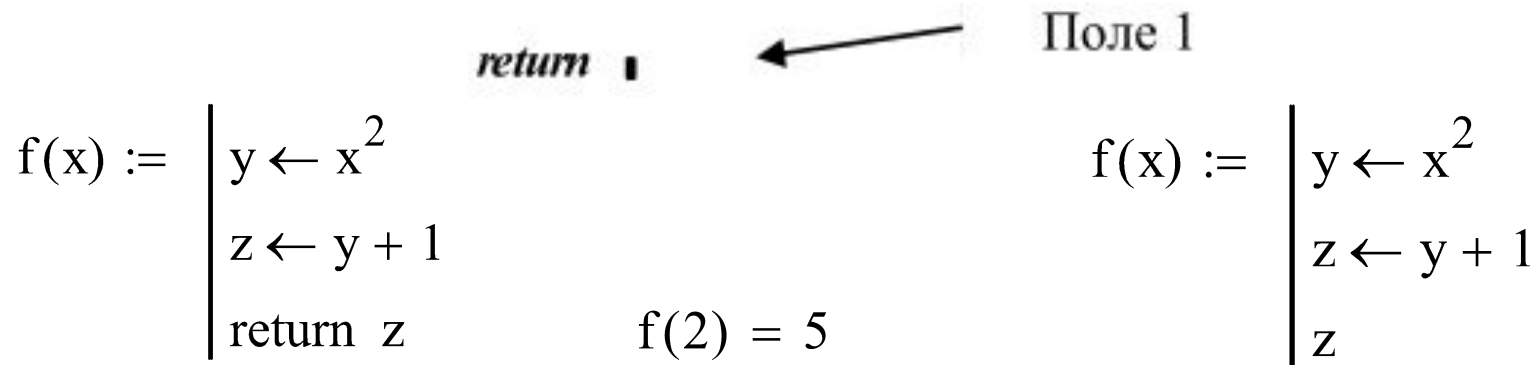
$$v := \begin{pmatrix} 2 \\ -6 \\ -1 \\ 12 \\ 5 \end{pmatrix}$$

$$\text{form_V}(v) = \begin{pmatrix} 2 \\ 12 \\ 5 \end{pmatrix}$$



Оператор *return*

– преждевременное завершение ПФ и возвращение операнда, стоящего в поле 1



Составить описание П-Ф, находящую первую положительную проекцию исходного вектора.

$\text{pol_1}(v) := \left| \begin{array}{l} i \leftarrow 1 \\ \text{while } v_i \leq 0 \\ \quad i \leftarrow i + 1 \\ v_i \end{array} \right.$

$\text{pol_1}(v) = 2$

$\text{pol_2}(v) := \text{for } i \in 1 .. \text{last}(v)$
 $\quad \text{return } v_i \text{ if } v_i > 0$

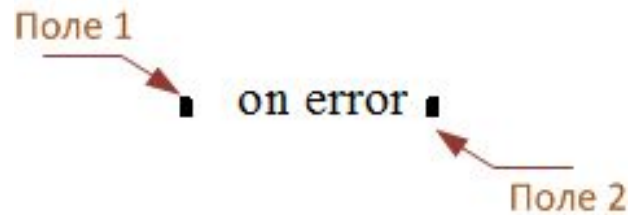
$\text{pol_2}(v) = 2$



Оператор *on error*

– перехват ошибок.

< конструкция 1 > on error < конструкция 2 >



В поле 2 вносится выражение, ошибка в вычислении которого должна быть зарегистрирована.

В поле 1 вносится условие, которое необходимо выполнить при возникновении ошибки.

$$f(x) := \left| \begin{array}{l} z \leftarrow x \\ \text{"деление на ноль"} \end{array} \right. \text{ on error } \frac{1}{z}$$

$f(-2) = -0.5$
 $f(0) = \text{"деление на ноль"}$



Функция *error*

Используется для вывода диагностических сообщений при возникновении в вычислениях ошибки и записывается в виде

error ("*< диагностическое сообщение пользователя >*")

Сформировать вывод диагностического сообщения при попытке спроецировать вектор v на нулевой вектор w

$$\text{proj}(v, w) := \begin{cases} \text{error}(\text{"Проекция на нулевой вектор"}) & \text{if } |w| < 10^{-10} \\ \frac{w}{|w|} \cdot (v \cdot w) & \text{otherwise} \end{cases}$$

$$x := \begin{pmatrix} 2 \\ 4 \\ 6 \end{pmatrix}$$

$$w := \begin{pmatrix} 3 \\ 2 \\ 1 \end{pmatrix}$$

$$\text{proj}(x, w) = \begin{pmatrix} 16.036 \\ 10.69 \\ 5.345 \end{pmatrix}$$

$$x := \begin{pmatrix} 2 \\ 4 \\ 6 \end{pmatrix}$$

$$w := \begin{pmatrix} 0 \\ 0 \\ 0 \end{pmatrix}$$

$$\text{proj}(x, w) = \begin{bmatrix} \blacksquare \\ \blacksquare \end{bmatrix}$$

Проекция на нулевой вектор