

# Операторы цикла

Выполнила Анастасия Корчуганова

# Классификация

---

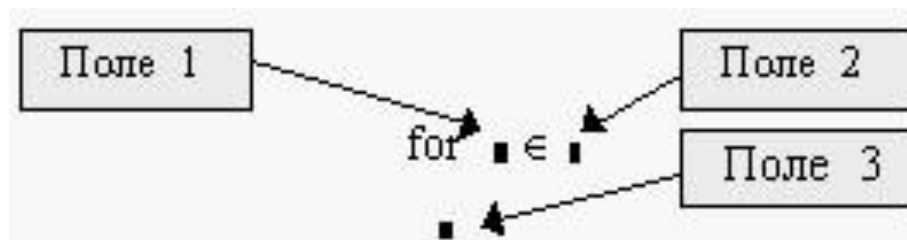
- ? Циклические алгоритмы (или проще циклы) содержат повторяющиеся вычисления, зависящие от некоторой переменной. Такая переменная называется *параметром цикла*, а сами повторяющиеся вычисления составляют *тело цикла*.
- ? Циклы можно условно разделить на две группы:
- ? **циклы типа арифметической прогрессии;**
- ? **итерационные циклы.**
- ? Характерной чертой первой группы циклов является то, что количество повторений тела цикла можно определить до начала выполнения программы, реализующей цикл, т.е. априори.
- ? Для итерационных циклов нельзя априори определить количество повторений тела цикла. Это обусловлено тем, что окончание таких циклов определяется не выходом параметра цикла за конечное значение, а более сложными условиями.



# FOR

? Цикл типа *for* является циклом, число выполнений которого определено заранее. Число выполнений определятся переменной цикла, задаваемой в его начале. Для ввода такого оператора необходимо выполнить следующие действия:

1) щелкнуть на кнопке **for** наборной панели **Программирования**. На экране появятся поля ввода:



- 2) в поле ввода 1 введите имя параметра цикла;
- 3) в поле ввода 2 ввести диапазон значений параметра цикла, используя для этого дискретный аргумент ;
- 4) в поле ввода 3 вводятся операторы, составляющие тело цикла. Если одной строки недостаточно, то дополнительные поля ввода (дополнительные строки) создаются щелчком на кнопке “Add line” в панели программирования и тогда слева от тела цикла появляется вертикальная черта.

# Пример


Mathcad - (Untitled 1)

Файл Правка Текст Математика Графика Символика Окно Книги ?

$\text{sum}(n) := \begin{cases} s \leftarrow 0 \\ \text{for } i \in 1..n \\ \quad s \leftarrow s + i \end{cases}$

$n := 44$   
 $\sum_{i=1}^n i = 990$

$\text{sum}(44) = 990$

 ← Переменная "i" не определена вне программы.

$\text{join}(r, s) := \begin{cases} m \leftarrow 0 \\ \text{for } x \in r, s \\ \quad \begin{cases} v_m \leftarrow x \\ m \leftarrow m + 1 \end{cases} \end{cases}$

$v$

$r := \begin{pmatrix} 100 \\ 101 \\ 102 \end{pmatrix}$

$s := \begin{pmatrix} 1 \\ 2 \end{pmatrix}$

$\text{join}(r, s) = \begin{bmatrix} 100 \\ 101 \\ 102 \\ 1 \\ 2 \end{bmatrix}$

автоматически Стр 1

# WHILE

---

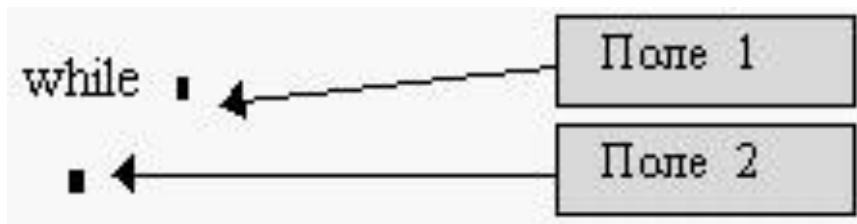
? Цикл типа *while* управляется истинностью некоторого условия, вследствие чего нет необходимости знать заранее число выполнений цикла. Важно только, чтобы где-нибудь внутри цикла или в другом выполняемом участке программы присутствовал оператор, делающий условие цикла ложным. В противном случае цикл будет выполняться бесконечно. Если выполняемая программа *зациклилась*, то ее можно остановить нажатием клавиши **[Esc]**.



# WHILE

? Для ввода этого оператора необходимо выполнить следующие действия:

1) щелкнуть на кнопке while панели **Программирования**. На экране появляются элементы:



2) в поле 1 ввести условие выполнения цикла;

3) в поле 2 ввести операторы тела цикла. В теле цикла должны присутствовать операторы делающие условие цикла ложным иначе цикл будет продолжаться бесконечно.

Оператор цикла while выполняется следующим образом: обнаружив оператор while, Mathcad проверяет указанное условие. Если оно истинно, то выполняется тело цикла и снова проверяется условие. Если оно ложно, то цикл заканчивается.

# Пример

Mathcad - (Untitled 1)

Файл Правка Текст Математика Графика Символика Окно Книги ?

Поиск первого элемента вектора, превосходящего заданное значение

$m := 0..2500$  ← Создание вектора

$v_m := 1 + \sin(m)$

$t(v, \text{thres}) := \begin{cases} j \leftarrow 0 & \leftarrow \text{Инициализация счётчика} \\ \text{while } v_j \leq \text{thres} & \\ \quad j \leftarrow j + 1 & \\ j & \leftarrow \text{Возврат значения} \end{cases}$

$t(v, 1.98) = 8$  ← Величина 1.98 впервые превосходится восьмым элементом вектора.

$v_m$
1
1.841
1.909
1.141
0.243
0.041
0.721
1.657
1.989
1.412
0.456
$9.793 \cdot 10^{-6}$
0.463
1.42
1.991

авто Стр 1

# IF

---

- ? Для того чтобы иметь возможность реализовать логику в программе используются условные операторы. Умозрительно эти операторы можно представить в виде узловых пунктов, достигая которых программа делает выбор по какому из возможных направлений двигаться дальше. Например, требуется определить, содержит ли некоторая переменная `arg` положительное или отрицательное число и вывести соответствующее сообщение на экран. Для этого можно воспользоваться оператором `if` (если), который и выполняет подобные проверки.
  
- ? В самом простом случае синтаксис данного оператора `if` имеет вид:
- ? 

```
if <выражение>  
<операторы>  
end
```
  
- ? 

```
if <выражение>  
<операторы1>      % выполняются, если истинно условие  
else  
<операторы2>      % выполняются, если условие ложно  
end
```





# Пример

---

```
? function my_sign
  x = 5;
  if x > 0
    disp(1);
  else
    if x < 0
      disp(-1);
    else
      disp(0);
    end
  end
end
```



# IF

---

```
? if <выражение1>  
    <операторы1>      % выполняются, если истинно выражение1  
elseif <выражение2>  
    <операторы2>      % выполняются, если истинно выражение2  
...  
elseif <выражениеN>  
    <операторыN>      % выполняются, если истинно выражениеN  
end
```

```
? function my_sign  
    x = 5;  
    if x > 0  
        disp(1);      % выполняется, если x > 0  
    elseif x < 0  
        disp(-1);     % выполняется, если x < 0  
    else  
        disp(0);      % выполняется, если x = 0  
    end
```



# IF

---

- ? С помощью условного оператора `if` можно выполнять проверку более сложных (составных) условий.  
Например, необходимо определить: попадает ли переменная `x` в диапазон значений от 0 до 2? Это можно реализовать одновременной проверкой сразу двух условий: `x >= 0` и `x <= 2`. Если эти оба условия истинны, то `x` попадает в диапазон от 0 до 2.
- ? Для реализации составных условий в MatLab используются логические операторы:
- ? `&` - логическое И  
| - логическое ИЛИ  
~ - логическое НЕ

