**1. Операторы системы MATLAB**

Операторы системы MATLAB делятся на 3 категории:

        *арифметические операторы -*позволяют конструировать арифметиче­ские выражения и выполнять числовые вычисления;

        *операторы отношения -*позволяют сравнивать числовые операнды;

         *логические операторы -*позволяют строить логические выражения.

*Логические операторы*имеют самый низкий приоритет относительно *операторов отношения*и *арифметических операторов.*

**Арифметические операторы.**При работе с массивом чисел установлены сле­дующие уровни приоритета среди арифметических операций:

*уровень 1:*поэлементное транспонирование (.'), поэлементное возведение в сте­пень (.^), транспонирование матрицы ('), воз­ведение матрицы в степень (^);

*уровень 2:*унарное сложение (+), унарное вычитание (—);

*уровень 3:*умножение массивов (.\*), правое деление (./), левое деление массивов (.\), умножение матриц (\*), решение систем линейных уравнений - операция (/), операция (\);

*уровень 4:*сложение (+), вычитание (-);

 *уровень 5:*оператор формирования массивов (:).

Внутри каждого уровня операторы имеют равный приоритет и вычис­ляются в порядке следования слева направо. Заданный по умолчанию порядок следования может быть изменен с помощью круглых скобок.

**Операторы отношения.**В системе MATLAB определено 6 следующих опера­торов отношения:

*<*  *меньше;*

*<=меньше или равно;*

*>   больше;*

>=  *больше или равно;*

==             *равно тождественно;*

~ = *не равно.*

Операторы отношения выполняют поэлементное сравнение двух мас­сивов равных размерностей. Для векторов и прямоугольных массивов оба операнда должны быть одинакового размера, за исключением случая, когда один из них скаляр. В этом случае MATLAB сравнивает скаляр с каждым элементом другого операнда. Позиции, где это соотношение истинно, получают значение 1, где ложно - 0.

Операторы отношения всегда выполняются поэлементно.

**Логические операторы.**В состав логических операторов системы MATLAB входят следующие операторы:

&  И;

          ИЛИ;

   НЕ.

Логические операторы реализуют поэлементное сравнение массивов одинаковых размерностей. Для векторов и прямоугольных массивов оба операнда должны быть одинакового размера, за исключением случая, когда один из них скаляр. В последнем случае MATLAB сравнивает скаляр с каж­дым элементом другого операнда. Позиции, где это соотношение истинно, получают значение 1, где ложно - 0.

Каждому логическому оператору соответствует некоторый набор условий, которые определяют результат логического выражения:

        Логическое выражение с оператором AND (&) является *истинным,*если оба операнда *истинны.*Если элементами логического выражения *явля­ются*числа, то выражение *истинно,*если оба операнда *отличны от нуля.*

        Логическое выражение с оператором OR( ) является *истинным*, если один из операндов или оба операнда логически *истинны*. Выражение *ложно*, только если оба операнда логически *ложны*. Если элементами логического выражения являются числа, то выражение *ложно,* если оба операнда равны *нулю*.

        Логическое выражение с оператором NOT() строит отрицание. Результат логически *ложен*, если операнд *истинен*, и *истинен*, если операнд *ложен.*Если элементами логического выражения являются числа, то любой операнд, *отличный от нуля,* становится *нулем,* и любой *нулевой*операнд становится *единицей.*

**Логические функции.**В дополнение к логическим операторам в состав сис­темы MATLAB включен ряд логических функций:

        Функция хоr(а, b) реализует операцию ИСКЛЮЧИТЕЛЬНОЕ ИЛИ. Выраже­ние, содержащее ИСКЛЮЧИТЕЛЬНОЕ ИЛИ *истинно,*если один из операндов имеет значение TRUE, а другой - FALSE. Для числовых выражений функция возвращает 1, если один из операндов отличен от нуля, а другой - 0.

        Функция all возвращает  1, если все элементы вектора *истинны*или *отличны от нуля.*

        Функция any возвращает 1, если хотя бы один из элементов аргумента отличен от нуля; иначе возвращается 0

Помимо операции возведения в степень, реализуемой с помощью знака  ^ , есть еще функция извлечения квадратного корня sqrt, функция exp для возведения в степень числа е, функция pow2 для возведения в степень числа 2. Также присутствуют обратные к ним функции: log – натуральный логарифм, log10 – логарифм по основанию 10,  log2 - логарифм по основанию 2.

Тригонометрические функции представлены весьма полно: sin, cos, tan (тангенс), cot (котангенс), asin (арксинус), acos (арккосинус), atan (арктан­генс), acot(арккотангенс). Имеются также и менее употребительные функции типа секанса, косеканса, а также гиперболические функции.

Упомянем еще функции, связанные с целочисленной арифметикой. Напри­мер, функции округления: round (округление до ближайшего целого), fix (усе­чение дробной части числа), floor (округление до меньшего целого), ceil (ок­ругление до большего целого).

Кроме того, есть еще функции mod (остаток от деления с учетом знака), rem (остаток в смысле модульной арифметики), sign (знак числа), factor (разложение числа на простые множители), isprime (истинно, если число простое), primes (формирование списка простых чисел), rat (приближение числа в виде рациональной дроби), 1cm (наименьшее общее кратное), gcd (наибольший общий делитель).

**Управление последовательностью исполнения операторов.** Существует 4 основных оператора управления последовательностью исполнения инструкций:

        оператор условия if в сочетании с операторами else и elseif выполняет группу  инструкций в соответствии с некоторыми логическими условиями;

        оператор переключения switch в сочетании с операторами case и otherwise выполняет различные группы инструкций в зависимости от значения некоторого логического условия;

        оператор условия while выполняет группу инструкций неопределенное число раз в соответствии с некоторым логическим условием завершения;

        оператор цикла for выполняет группу инструкций фиксированное число раз. Все операторы управления включают оператор end, чтобы указать конец блока, в котором действует этот оператор управления.

|  |  |
| --- | --- |
|

|  |
| --- |
| **If…else…elseif…end** |

  |

**Оператор  условия**

*Описание:*

Оператор условия if ... end вычисляет некоторое логическое выражение и выполняет соответствующую группу инструкций в зависимости от значения этого выражения. Если логическое выражение *истинно,*то MATLAB выполнит все инструкции между if и end, а затем продолжит выполнение программы в строке после end. Если условие *ложно,*то MATLAB пропусти все утверждения между if и end и продолжит выполнение в строке после end.

Операторы if ... else ... end и if ... elseif ... end создают дополнительные ветвления внутри тела оператора if:

        оператор else не содержит логического условия. Инструкции, связанные с ним, выполняются, если предшествующий оператор if (и, возможно, elseif) *ложен;*

        оператор elseif содержит логическое условие, которое вычисляется, если предшествующий оператор if (и, возможно, elseif) *ложен.*Инструкции, связанные с оператором  elseif,  выполняются,  если соответствующее логическое условие *истинно.*

Оператор elseif может многократно использоваться внутри оператора условия if.

|  |  |
| --- | --- |
|

|  |
| --- |
| **Switch…case…otherwise…end** |

  |

**Оператор переключения**

*Синтаксис:*

switch *<выражение>*

% *выражение - это обязательно скаляр или строка*

case *<значение1>*

*инструкции*

*% выполняются, если <выражение> = <значение1>*

case *<значение2>*

*инструкции*

*% выполняются, если <выражение> = <значение2>*

…

otherwise

*инструкции*

*% выполняются, если <выражение > не совпало ни с одним из значений*

end

Оператор switch ... case 1 ... case k ... otherwise ... end выполняет ветвления, в зависимости от значений некоторой переменной или выражения.

Оператор переключения включает:

        заголовок switch, за которым следует вычисляемое выражение (скаляр или строка);

        произвольное количество групп case. Заголовок группы состоят из слова case, за которым следует возможное значение выражения, распо­ложенное на одной строке. Последующие строки содержат инструкции, которые выполняются для данного значения выражения. Выполнение продолжается до тех пор, пока не встретится следующий оператор case или оператор otherwise. На этом выполнение блока switch завершается;

        группа otherwise. Заголовок включает только слово otherwise, начиная со следующей строки размещаются инструкции, которые выполняются, если значение выражения оказалось не обработанным ни одной из групп case. Выполнение завершается оператором end;

        оператор end - является последним в блоке переключателя.

Оператор switch работает, сравнивая значение вычисленного выражения со значениями групп case. Для числовых выражений оператор case выпол­няется, если *<значение>==<выражение>.*Для строковых выражений оператор case *истинен,*если strcmp*(значение, выражение) истинно.*

|  |  |
| --- | --- |
|

|  |
| --- |
| **While…end** |

  |

**Оператор цикла с  неопределенным  числом операций**

*Синтаксис:*

while *выражение*

*инструкции*

end

*Описание:*

Оператор цикла с неопределенным числом операций while ... end мно­гократно выполняет инструкцию или группу инструкций, пока управляющее выражение *истинно.*

Если выражение использует массив, то все его элементы должны быть истинны, чтобы продолжить выполнение. Чтобы привести матрицу к ска­лярному значению, следует использовать функции any и all.

Выход из white-цикла может быть реализован с помощью оператора break. Если в операторе while управляющее условие является пустым масси­вом, то такое условие *ложно,*то есть оператор вида while A, S1, end никогда **не**выполнит инструкции S1, если А - пустой массив.

|  |  |
| --- | --- |
|

|  |
| --- |
| **For…end** |

  |

**Оператор цикла с определенным числом операций**

*Синтаксис:*

for *<переменная цикла>*= *<нач. знач.>:<приращение>: <конеч. знач.>*

*инструкции*

end

*Описание:*

Оператор цикла for ... end выполняет инструкцию или группу инструк­ций предопределенное число раз. По умолчанию приращение равно 1. Можно задавать любое приращение, в том числе отрицательное. Для поло­жительных индексов выполнение завершается, когда значение индекса пре­вышает*<конечное значение>;*для отрицательных приращений выполнение завершается, когда индекс становится меньше чем *<конечное значение>.*