Общий синтаксис

Общий синтаксис языка прост. Любая инструкция на этом языке записывается в виде команд, за каждым названием команды, после двоеточия следует список параметров, отделяемых запятой. Команда заканчивается точкой с запятой.

Пример команды:

***КОМАНДА: параметр1, параметр2, … , параметр N;***

Команда и её параметры считываются до точки с запятой. Это значит, что в одной строчке можно как записать несколько команд, например:

***VAR: $FIRST, DBL3.14; VAR: $SECOND, NTG1;***

Так же можно и форматировать команду для удобства на несколько строк, например:

***PRINT: ‘строка1’,***

***NTG5,***

***‘строка2’;***

**Типы данных**

В языке существуют следующие типы данных:

|  |  |  |
| --- | --- | --- |
| **Наименование** | **Описание типа** | **Диапазон** |
| UNTG | Беззнаковый целочисленный | 0 … 18 446 744 073 709 551 |
| NTG | Целочисленный | −9 223 372 036 854 775 808 … 9 223 372 036 854 775 807 |
| DBL | Вещественное число двойной точности с плавающей точкой | 1.7E - 308 … 1.7E +308 |
| CHR | Символ ASCII, байт | -128 … 127 |
| UCHR | Беззнаковый байт | 0 … 255 |
| BLN | Логический | TRUE / FALSE |
| STR | Строка из широких, 16 — ти разрядных символов | - |
| NIL | Аналог NULL | NULL |
| ARR | Массив | Массив из любых значений, в том числе других массивов и ассоциативных массивов |
| MAP | Ассоциативный массив с доступом по ключу | Ассоциативный массив из любых значений, в том числе других массивов и ассоциативных массивов |
| UNKNOWN | Внутренний тип, обозначающий ошибку работы с данными интерпретатора. Не должен появляться | UNKNOWN |

**Литералы**

Для понимания типа литерала интерпретатору необходимо сообщить тип данных непосредственно перед значением. Например, чтобы создать переменную с плавающей точкой и значением 6.5, необходимо перед самим числом поставить метку DBL, например:

***VAR: $VAR, DBL6.5;***

Разрешено использовать экспоненциальную запись. Возможно использование как малого так и большого символа Е:

***VAR: $VAR, DBL-1.85633e+11***

А если, к примеру, необходимо создать переменную с целочисленным значением 2, то необходимо поставить метку NTG:

***VAR: $VAR, NTG2;***

Всего существуют следующие метки для литералов:

***NTG*** – целочисленный литерал

***UNTG*** – беззнаковый целочисленный литерал

***DBL*** – число двойной точности с плавающей запятой

***CHR*** – символ, значения от -128 до +127

***UCHR*** – беззнаковый символ от 0 до 255

***TRUE*** – булево значение true, истина

***FALSE*** – булево значение false, ложь

***NIL*** – значение null, ничто

Для литералов типа: NTG, UNTG, DBL, CHR и UCHR возможно использование знака + для явного указания положительного значения, например:

***VAR: $VAR, NTG+13;***

**Строки**

Строки в языке начинаются с одинарной кавычки и заканчиваются ею же. Например, вот строковый литерал:

***‘Это строка а это одинарная кавычка \q‘***

Чтобы использовать символы форматирования, такие, как перевод на новую строку, табуляция, необходимо использовать управляющие последовательности непосредственно в месте, где требуется символ.

Например, данная строка содержит в себе одинарную кавычку, табуляцию и перевод на новую строку

***‘Это одинарная кавычка \q а это перевод на новую строку \n а это табуляция \t пример.‘***

Распечатанная строка будет выглядеть так:

*Это одинарная кавычка ‘ а это перевод на новую строку*

*а это табуляция пример.*

В языке существуют следующие управляющие последовательности:

***\q*** – Одинарная кавычка

***\t*** – Табуляция

***\n*** – Перевод на новую строку

***\a*** – Звуковой сигнал

***\b*** - Backspace

***\f*** – Подача страницы

***\r*** – Возврат каретки

***\v*** – Вертикальная табуляция

***\”*** - Двойная кавычка

***\?*** - Вопросительный знак

***\\*** - Обратная косая черта

**Комментарии**

В языке имеется возможность добавления комментариев. Комментарии вырезаются из исходного когда до разбора кода на составляющие части.

**Однострочный комментарий:**

Однострочный комментарий начинается с символа #. Все, что идет после этого знака, игнорируется до конца строки. Комментарий так же, может занимать всю строку.

Примеры:

NOP:; # Это однострочный комментарий

NOP:; # И это однострочный комментарий

# Это тоже однострочный комментарий

**Многострочный комментарий:**

Многострочный комментарий начинается с двух символов ## подряд без пробелов между ними. Заканчивается многострочный комментарий точно так же, двумя символами ## без пробелов между ними

Примеры:

##

Это

Многострочный

Комментарий

##

## И это многострочный комментарий, но в одной строке ##

**ВНИМАНИЕ:**

Необходимо избегать внутри многострочного комментария последовательности ##, потому что в таком случае это интерпретируется как закрытие многострочного комментарий

Примеры:

## Это неправильный многострочный комментарий с использованием ## внутри, потому что этот текст уже не комментарий, а после весь код будет закомментирован ##

## Закомментирован кусочек кода, это приведет к ошибке

PRINT: ‘## Ой!‘;

##

**Список команд**

**Тестовые**

***NOP*** – тестовая команда, ничего не делающая.

Пример использования:

*NOP:;*

**Прочие**

***END*** *–* команда, завершающая работу основной программы или программный блок (подпрограммы). Принимает один параметр в виде имени переменной или литерала, который возвращается в виде результата выполнения подпрограммы.

Пример использования:

*END: $VAR;*

*END: NTG5;*

**Работа с переменными**

***VAR*** – команда, объявляющая переменную и выделяющая место под нее в динамической памяти. Переменная может быть любого допустимого в языке типа. Принимает два параметра. Первый — имя будущей переменной. Второй — значение переменной. Имя переменной должно начинаться со знака $. Само имя переменной может содержать пробелы, спецсимволы, начинаться с цифры. Если имя переменной уже занято, это приводит к ошибке. Перезаписи значения не происходит. Для перезаписи значения существует команда CHNG.

Пример использования:

*VAR: $PI, DBL3.14;*

*VAR: $FIRST, NTG1665;*

***FREE –*** команда, высвобождающая динамическую память от указанной переменной и удаляющая ее в конечном итоге. Принимает произвольное количество параметров. Это должны быть имена переменных. Необходимо тщательно следить за объявленными переменными и освобождать от них память, если они будут более не нужны, во избежание утечки памяти.

Пример использования:

*FREE: $FIRST, $SECOND, $THIRD;*

***CHNG –*** команда, позволяющая изменить значение уже созданной переменной. Так как команда VAR имеет главное предназначение в создании переменных, команда CHNG – единственный способ присвоить уже существующей переменной другое значение. Принимает два параметра: Первый параметр — имя существующей переменной, значение которой будет изменено. Второй параметр — значение, на которое будет изменено значение первой переменной. Второй параметр может быть как литералом, так и именем другой переменной.

Пример использования:

*CHNG: $SECOND, $FIRST;*

**Условные и безусловные переходы, метки**

***LBL* –** команда, устанавливающая в код метку, на которую после можно ссылаться командами перехода, что переводит исполнение кода на строку с меткой. Принимает один особый параметр — имя метки. Имя метки всегда начинается с символа &. (Техническая информация. Все метки обрабатываются до начала исполнения кода и заносятся в специальную память для меток и содержат номер строки с инструкцией. Метки после занесения в память более неизменяемы. После перехода на метку указатель на исполняемую команду становится именно на МЕТКУ, а не на следующую за ней инструкцию. При выполнении инструкции непосредственно метки, ничего не происходит, как при команде NOP).

Пример использования:

*LBL: &LABEL1;*

***JMP***– команда, выполняющая безусловный переход на указанную метку (указатель на исполняемую команду „прыгает“ на эту метку). Принимает один параметр — имя метки, на которую нужно сделать прыжок.

Пример использования:

*JMP: &LABEL1;*

***JMPIFZ –*** команда, выполняющая условный переход на метку в случае, если в наличествующем параметре значение равно 0. Принимает два параметра. Первый параметр — значение, которое проверяется по условию, может быть как именем переменной, так и литералом. Второй параметр — имя метки, на которое выполняется переход в случае срабатывания условия.

Пример использования:

*JMPIFZ: $FIRST, &LABEL1;*

*JMPIFZ: NTG0, &LABEL1;*

***JMPIFNOTZ*** *-* команда, выполняющая условный переход на метку в случае, если в наличествующем параметре значение НЕ равно 0. Принимает два параметра. Первый параметр — значение, которое проверяется по условию, может быть как именем переменной, так и литералом. Второй параметр — имя метки, на которое выполняется переход в случае срабатывания условия.

Пример использования:

*JMPIFNOTZ: $FIRST, &LABEL1;*

*JMPIFNOTZ: NTG1, &LABEL1;*

**Ввод/вывод**

***INPUT –*** команда, выполняющая запрос на ввод информации с клавиатуры пользователя в консоль. Принимает один параметр, обязательно являющийся именем существующей переменной, в которое кладется введенное значение. Важное примечание: Команда INPUT отдает в переменную строковое значение. Если необходим другой тип данных, например, численный, после ввода необходимо сделать приведение типов до нужного типа.

Пример использования:

*INPUT: $MY\_VAR;*

***PRINT*** – команда, выводящая в консоль (на экран) данные в строковом представлении. Принимает произвольное количество параметров, отделенных запятой (но обязательно не меньше одного). Не может отобразить содержимое обычных массивов и ассоциативных массивов напрямую.

Пример использования:

*PRINT: 'переменная #1 равна: ', $FIRST,*

*'переменная #2 равна: ', $SECOND,*

*'\n';*

**Приведение типов**

Приведение типов производится командой TO.

***TO*** – команда, приводящая иной тип переменной к указанному типу. Принимает 2 либо 3 параметра.

1 параметр: Строковая переменная или строковый литерал с типом данных, к которому приводится значение. Допустимые типы данных: NTG, UNTG, DBL, CHR, UCHR, BLN, STR, ARR

2 параметр: Всегда является именем переменной. В данную переменную заносится результат приведения типов.

3 параметр (опционально): Переменная или литерал, из которых берется значение для приведения.

В случае, если используется 2 параметра, берется значение 2 параметра и приводится к типу, указанному в 1 параметре. Результат кладется в переменную 2 параметра. В случае, если используется 3 параметра, берется значение 3 параметра, приводится к типу, указанному в 1 параметре. Результат кладется в переменную 2 параметра

Пример использования:

c двумя параметрами, тип данных указан в переменной:

*VAR: $CAST, 'NTG';*

*VAR: $VAR, DBL3.14;*

*TO: $CAST, $VAR;*

*c двумя параметрами, тип данных указан в инструкции:*

*VAR: $VAR, DBL3.14;*

*TO: 'NTG', $VAR;*

*с тремя параметрами, тип данных указан в переменной:*

*VAR: $CAST, 'NTG';*

*VAR: $SOURCE, DBL3.14;*

*VAR: $RESULT, NIL;*

*TO: $CAST, $RESULT, $SOURCE;*

*с тремя параметрами, тип данных указан в инструкции:*

*VAR: $SOURCE, DBL3.14;*

*VAR: $RESULT, NIL;*

*TO: 'NTG', $RESULT, $SOURCE;*

Таблицы приведения типов:

**К NTG:**

|  |  |
| --- | --- |
| **Приводимый тип** | **Описание особенностей:** |
| NTG | Приводится к типу NTG без потери данных |
| UNTG | При выходе значения UNTG за пределы NTG, NTG получает отрицательное значение |
| DBL | Отбрасывается дробная часть. При выходе за пределы NTG результат недетерминирован |
| CHR | Приводится к типу NTG без потери данных |
| UCHR | Приводится к типу NTG без потери данных |
| BLN | FALSE приводится к 0  TRUE приводится к 1 |
| STR | Происходит попытка преобразовать строку в число, если строка состоит из цифр. Если же это невозможно, выбрасывается исключение. |
| NIL | Приводится к 0 |
| ARR | Выбрасывается исключение |
| MAP | Выбрасывается исключение |
| UNKNOWN | Приводится к 0 |

**К UNTG:**

|  |  |
| --- | --- |
| **Приводимый тип** | **Описание особенностей:** |
| NTG | При выходе за пределы UNTG (если значение отрицательное) получает положительное значение |
| UNTG | Приводится к типу UNTG без потери данных |
| DBL | Отбрасывается дробная часть. При выходе за пределы UNTG результат недетерминирован |
| CHR | При выходе за пределы UNTG (если значение отрицательное) получает положительное значение |
| UCHR | Приводится к типу UNTG без потери данных |
| BLN | FALSE приводится к 0  TRUE приводится к 1 |
| STR | Происходит попытка преобразовать строку в число, если строка состоит из цифр. Если же это невозможно, выбрасывается исключение. |
| NIL | Приводится к 0 |
| ARR | Выбрасывается исключение |
| MAP | Выбрасывается исключение |
| UNKNOWN | Приводится к 0 |

**К DBL:**

|  |  |
| --- | --- |
| **Приводимый тип** | **Описание особенностей:** |
| NTG | При больших значениях часть мантиссы теряется. |
| UNTG | При больших значениях часть мантиссы теряется. |
| DBL | Приводится к типу DBL без потери данных |
| CHR | Приводится к типу DBL без потери данных |
| UCHR | Приводится к типу DBL без потери данных |
| BLN | FALSE приводится к 0.0  TRUE приводится к 1.0 |
| STR | Происходит попытка преобразовать строку в число. Если же это невозможно, выбрасывается исключение. |
| NIL | Приводится к 0.0 |
| ARR | Выбрасывается исключение |
| MAP | Выбрасывается исключение |
| UNKNOWN | Приводится к 0.0 |

**К CHR:**

|  |  |
| --- | --- |
| **Приводимый тип** | **Описание особенностей:** |
| NTG | При значениях, выходящих за пределы CHR происходит переполнение с потерей данных |
| UNTG | При значениях, выходящих за пределы CHR происходит переполнение с потерей данных |
| DBL | Отбрасывается дробная часть. При значениях, выходящих за пределы CHR происходит переполнение с потерей данных |
| CHR | Приводится к типу CHR без потери данных |
| UCHR | При значениях, выходящих за пределы CHR происходит переполнение. |
| BLN | FALSE приводится к 0  TRUE приводится к 1 |
| STR | Происходит попытка преобразовать строку в число и преобразовать это число к типу CHR. Если же это невозможно, выбрасывается исключение. |
| NIL | Приводится к 0 |
| ARR | Выбрасывается исключение |
| MAP | Выбрасывается исключение |
| UNKNOWN | Приводится к 0 |

**К UCHR:**

|  |  |
| --- | --- |
| **Приводимый тип** | **Описание особенностей:** |
| NTG | При значениях, выходящих за пределы UCHR происходит переполнение с потерей данных |
| UNTG | При значениях, выходящих за пределы UCHR происходит переполнение с потерей данных |
| DBL | Отбрасывается дробная часть. При значениях, выходящих за пределы UCHR происходит переполнение с потерей данных |
| CHR | При значениях, выходящих за пределы UCHR происходит переполнение. |
| UCHR | Приводится к типу UCHR без потери данных |
| BLN | FALSE приводится к 0  TRUE приводится к 1 |
| STR | Происходит попытка преобразовать строку в число и преобразовать это число к типу UCHR. Если же это невозможно, выбрасывается исключение. |
| NIL | Приводится к 0 |
| ARR | Выбрасывается исключение |
| MAP | Выбрасывается исключение |
| UNKNOWN | Приводится к 0 |

**К BLN:**

|  |  |
| --- | --- |
| **Приводимый тип** | **Описание особенностей:** |
| NTG | Любое значение, кроме 0 приводится к TRUE, значение 0 приводится к FALSE |
| UNTG | Любое значение, кроме 0 приводится к TRUE, значение 0 приводится к FALSE |
| DBL | Любое значение, кроме 0 приводится к TRUE, значение 0 приводится к FALSE |
| CHR | Любое значение, кроме 0 приводится к TRUE, значение 0 приводится к FALSE |
| UCHR | Любое значение, кроме 0 приводится к TRUE, значение 0 приводится к FALSE |
| BLN | Приводится к типу BLN без потери данных |
| STR | Пустая строка приводится к значению FALSE. Любая другая к значению TRUE. |
| NIL | Приводится к FALSE |
| ARR | Пустой массив приводится к значению FALSE. Любой другой к значению TRUE. |
| MAP | Пустой ассоциативный массив приводится к значению FALSE. Любой другой к значению TRUE. |
| UNKNOWN | Приводится к FALSE |

**К STR:**

|  |  |
| --- | --- |
| **Приводимый тип** | **Описание особенностей:** |
| NTG | Приводится к типу STR без потери данных |
| UNTG | Приводится к типу STR без потери данных |
| DBL | Приводится к типу STR с возможным сокращением слишком длинной дробной части |
| CHR | Приводится к числу, число приводится к типу STR без потери данных |
| UCHR | Приводится к строковому виду без потери данных |
| BLN | TRUE приводится к строковому значению TRUE. FALSE к FALSE |
| STR | Приводится к типу STR без потери данных |
| NIL | Приводится к пустой строке |
| ARR | Выбрасывается исключение |
| MAP | Выбрасывается исключение |
| UNKNOWN | Приводится к пустой строке |

**К ARR:**

|  |  |
| --- | --- |
| **Приводимый тип** | **Описание особенностей:** |
| NTG | Приводится к массиву из 1 элемента, значение которого равно значению приводимого типа |
| UNTG | Приводится к массиву из 1 элемента, значение которого равно значению приводимого типа |
| DBL | Приводится к массиву из 1 элемента, значение которого равно значению приводимого типа |
| CHR | Приводится к массиву из 1 элемента, значение которого равно значению приводимого типа |
| UCHR | Приводится к массиву из 1 элемента, значение которого равно значению приводимого типа |
| BLN | Приводится к массиву из 1 элемента, значение которого равно значению приводимого типа |
| STR | Приводится к массиву из 1 элемента, значение которого равно значению приводимого типа |
| NIL | Приводится к массиву из 1 элемента, значение которого равно значению приводимого типа |
| ARR | Остается прежним с прежними значениями |
| MAP | Выбрасывается исключение |
| UNKNOWN | Приводится к массиву из 1 элемента, значение которого равно значению приводимого типа |

**Математические операции:**

Приведение типов производится командой CALC.

***CALC*** – команда, совершающая указанную математическую операцию. Может иметь от 2 до 4 параметров.

1 параметр: Строковая переменная или строковый литерал с типом выполняемой математической операции. Допустимые математические операции: +, -, \*, /, % (деление по модулю), ^ (степень), ROOT (корень) INC (инкремент), DEC (декремент), LOG (логарифм), LN (натуральный логарифм), !(факториал)

2 параметр: Всегда является именем переменной. В данную переменную заносится результат математической операции.

3 параметр (опционально): Переменная или литерал, из которых берется значение для произведения математической операции.

4 параметр (опционально): Переменная или литерал, из которых берется значение для произведения математической операции.

Пример использования: