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’;***

**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;***

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 литерала

***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

**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

Пример:

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

**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 $. 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**

***TONTG*** – команда, приводящая иной тип переменной к типу NTG. Принимает либо 1 либо 2 параметра. В случае, если принимается 1 параметр, параметр должен быть именем существующей переменной, а приведенное значение к новому типу записывается в указанную переменную. Если же используется вариант с двумя параметрами, то оба параметра должны быть именами существующих переменных, в таком случае, значение второго параметра приводится к новому типу и записывается в переменную первого параметра. Второй параметр как и первый, НЕ могут быть литералами. В обоих случаях, если привести переменную к нужному типу не удалось, программа выдает ошибку и завершает свою работу.

Пример использования:

*TONTG: $VAR;*

*TONTG: $VAR1, $VAR2;*

***TOUNTG*** – команда, приводящая иной тип переменной к типу UNTG. Работа с командой точно такая же, как и с командой TONTG.

Пример использования:

*TOUNTG: $VAR;*

*TOUNTG: $VAR1, $VAR2;*

***TODBL*** – команда, приводящая иной тип переменной к типу DBL. Работа с командой точно такая же, как и с командой TONTG.

Пример использования:

*TODBL: $VAR;*

*TODBL: $VAR1, $VAR2;*