Heza Documentacion

**Comentarios:**

@comentario

**Palabras clave:**

data //Declara una variable

out //Salida por consola

in // entrada por consola

trace //Activa trace sobre una variable

as //apodo

loop //ciclo while-for

end //termina un bloque

block //function

**Constantes que representan un valor:**

LINE //Salto de linea

TAB //Tabulador

ESP //Espacio en blanco

**Operadores aritmeticos:**

+ //suma

- //resta

\* //multiplicacion

/ //Division

^ //potencia

% //Mod

**operador de asignacion =**

operadores logicos:

not

or

and

**Operadoreas de relacion:**

>

<

==

!=

<=

>=

**Operadores especiales:**

? //Condicional o while

-> //flujo para in/out

---------------------------------------------------------

Uso

---------------------------------------------------------

**Declaracion de variables:**

data var //declara una variable con el nombre var

data trace var //Declara una variable con trace activo (el trace guarda todos los cambios que ha tenido desde la creacion)

declaracion con asignacion:

data var = valor

data trace var = valor

los valores son enteros,decimales,booleanos,constantes de valor,otras variables o string entre comillas,no se aceptan comillas simples

**Entrada:**

in->x //Lee de la consola y guarda en x, debe ser necesariamente una variable declarada por el usuario

in->x,y,z //Multiple lectura

**Salida**

out->x //Imprime el valor de x, pueden ser valores inmediatos

out->x,LINE,y//salida multiple

**Limpiar colsola:**

CLS //Esa linea asi tal cual si mas nada limpia la salida

**Condicionales:**

?condicion? as apodo

//Cuerpo

end:apodo

**Ejemplo:**

?x>5? as mayor

out->"x es mayor a 6"

end:mayor

Nota: No se permiten comandos else por default, deben tener una condicion (contraria a la original obviamente) y un apodo

Todo apodo iniciado debe tener su end:apodo

**Bucles:While**

loop(condicion)? as apodo

//cuerpo

end:apodo

ejecuta cuerpo mientras la condicion se cumpla

**//Bucles:for**

loop(from:to:inc) as alias

//cuerpo

end:alias

from,to e inc son variables numericas, alias representa el valor del iterador en el cuerpo, inicia en fromy termina en to aumentando pasos de inc, ejmplo:

loop(1:10:1) as i

out->i,ESP

end:i

//Salida 1 2 3 4 5 6 7 8 9 10

**Funciones**

block nombre(parametros):

//Cuerpo

end:nombre

Ejemplo:

block saludar(nombre):

out->nombre

end:saludar

llamada: saludar("Juan") //Se aceptan varios parametros

//Hasta ahora no hay librerias ni listas y nada que no se haya nombrado esta implementado, solo exclusivamente este fragmento.

**Funciones integradas:**

Integer(texto) //Convierto string a entero si es posible

String(variable) //Convierte a string si es posible

R(min,max) //Numero random entre min y max

Wait(time) //Espera time segundos

Size(objeto) //Largo de string o cantidad de digitos de un numero

Sin,Cos,Sqrt,Ln //Funciones matematicas de un solo parametron

Log(base,n) //Logaritmo base base de n

ToUpperCase(txt),ToLowerCase(txt) //trasformacion de texto

Pressed(key) //Devuelve true si la tecla key esta presionada en ese momento

Vocalas(txt) //Devuelve solo las vocales del texto

Abs(n) //valor absoluto

Date() //fecha actual

Time() //hora actual

Repeat(text, n) //Repite text n veces en una cadena

**Metodos integrados:**

Reverse //para texto

Equals(c) //devuelve la cantidad de conindicencias del caracter.

**Ejemplos:**

Integer("12") = 12

String(89) = "89"

R(1,8) = entero entre 1 y 8

Wait(4) = espera 4 segundos

Size(56) = 2

ToUpperCase("ho") = "HO"

Abs(-45) = 45

Repeat("a",4) = "aaaa"

data x = "holaa"

x.count("a") = 2

**Como mostrar la evolucion de una variable (trace)**

out->trace:x

data y = trace:x //Se almacena en y el trace de x

//Solo se guarda el trace de las variables que se les activo

ejemplo:

data trace x = 78

x=4

x=5

trace:x = "x = 78 >> 4 >> 5"

**Reasignacion muntiple:**

x,y = 3,2 //x=3,y=2

util para swaps: a,b=b,a //El interprete lo soporta