Lenguaje ZiSan

Declaración de variables

&nombre_

nombre es un conjunto de letras seguido o no de un número de 1 a 3 dígitos

&nombre[]_ Definición de un arreglo

dígitos = {0,1,2,3,4,5,6,7,8,9}

letras = {a,b,c,d,e,f,g,h,i,j,k,l,m,n,ñ,o,p,q,r,s,t,u,v,w,x,y,z}

Palabras reservadas

$$StartZiSan inicia el programa en ZiSan

$$EndZiSan finaliza el programa en ZiSan

ZSPrint() para mostrar en consola

ZSRead() para leer desde consola

; para finalizar una línea de código

ZSLength() para retornar la longitud del atributo

nulle valor nulo

Operadores y Funciones Matemáticas

++ suma

-- resta

****** multiplicación

// división

//· División entera

**** Módulo

^^ exponenciación o elevación

Sn() Seno

Cs() Coseno

Santiago Lechuga Mariana
San Juan Escalona Sarai
Zilli Vazquez David Nicodemo

Tn() Tangente
Sq() Raíz Cuadrada
+++ Incremento en uno
--- decremento en uno
:: concatenación
?= comparador igual
? comparador diferente
?> comparador mayor que
?< comparador menor que
?>= comparador mayor igual que
?<= comparador menor igual que
-> asignación de valor
->-> asigna a la variable su valor propio más el nuevo
, separar atributos
ZSIf Inicio de una condición
ZSOther Caso contrario de una condición
ZSIfOther Para anidar condiciones
ZSLoop Ciclo que puede ser
ZSLoop (condicion) Como el while
ZSLoop (contador) in (arreglo) Como el for
&& and
%% or
**** not
@@ comentario
/Hola/ cadena de caracteres
/H/ un caracter
#
comentario comentario varias lineas
#
¡! Salto de línea
{ } Abrir y cerrar sección de código

Santiago Lechuga Mariana
San Juan Escalona Sarai
Zilli Vazquez David Nicodemo

Estructura Básica

$$StartZiSan nombre_del_programa @@Inico del programa

@@Cuerpo del programa

$$EndZiSan @@Fin del programa

Ejemplo “HOLA MUNDO”

$$StartZiSan Ejem1 @@Inico del programa

ZSPrint(/Hola Mundo/); @@Escritura en consola

$$EndZiSan @@Fin del progama

Ejemplo de Suma de 2 números

$$StartZiSan Ejem2 @@Inico del programa

&num1_ -> 10; @@Asignacion de valor 10 a la variable num1
&num2_ -> 12; @@Asignacion de valor 10 a la variable num2

&resultado_ -> &num1_ ++ &num2_ ;
@@Suma de las variables num1 y num2 y asignacion del
@@resultado a la variable resultado

ZSPrint(/El resultado es:/::&resultado_);
@@Escritura en consola del resultado

$$EndZiSan @@Fin del programa

Santiago Lechuga Mariana
San Juan Escalona Sarai
Zilli Vazquez David Nicodemo

Ejemplos de programa integrados

```
$$StartZiSan Ejem3    @@Inico del programa

#
Retornar a que siglo corresponde un año dado
#

        &year_ -> 0;  @@Declaracion de la variable año

        ZSIf ( ( &year_ \\ 100 ) ?= 0 ){      @@ condicion
                &resultado_ -> &year_ //· 100;   @@ asignacion
        }ZSOther{                               @@ caso contrario
                &resultado_ -> (&year_ //· 100 ) ++ 1;
        }

        ZSPrint( /El siglo de: /::&year_::/ es:/::&resultado_);
                @@Escritura en consola del resultado

$$EndZiSan  @@Fin del programa
```

Santiago Lechuga Mariana
San Juan Escalona Sarai
Zilli Vazquez David Nicodemo

```
$$StartZiSan Ejem4    @@Inico del programa

#
Programa que retorna una cadena con el contenido de un arreglo
exceptuando un valor que sea dado
#
        &valor_-> 0;       @@Declaracion de la variable valor
        &arreglo_ -> [2,4,3,6,8,3,10,13,24,45] ;
                @@Declaracion del vector

        &cad_->nulle;
        @@Declaracion de la variable cadena inicializada en nulo

        &valor_ -> ZSRead(/Ingresa el valor a quitar/);
        @@Lectura de teclado para asignar valor

        ZSLoop (&i_) in (&arreglo_){@@Ciclo para recorrer el arreglo
                ZSIf(&i_ ?\ &valor_){
                        @@Pregunta condicional si i es diferente de valor
                &cad_->->  &i_ ::/¡!/;
                @@Concatenacion de los valores en cad
                }
        }

        ZSPrint( /El valor exceptuado fue: /::&valor_::/¡!/::
        /El arreglo resultado es: /::&cad_);

$$EndZiSan @@Fin del programa
```

Santiago Lechuga Mariana
San Juan Escalona Sarai
Zilli Vazquez David Nicodemo

Gramática

L(G) = {x|x es un programa en lenguaje ZiSan}

G=(Σ,P,Q,V_0)

Σ={a,b,c,d,e,f,g,h,i,j,k,l,m,n,ñ,o,p,q,r,s,t,u,v,w,x,y,z,A,B,C,D,E,F,G, H,I,J,K,L,M,N,Ñ,O,P,Q,R,S,T,U,V,W,X,Y,Z,0,1,2,3,4,5,6,7,8,9,&,_,$,(,),;,+,-/,\,·,^,:,?,>,=,%,@,#,¡,!,{,}}

V_0-> $$StartZiSan NombreClase Cuerpo $$EndZiSan

NombreClase -> Letra
NombreClase -> Letra NombreClase

Letra -> a	**Letra ->j**	**Letra -> r**
Letra -> A	**Letra -> J**	**Letra -> R**
Letra -> b	**Letra -> k**	**Letra -> s**
Letra -> B	**Letra ->K**	**Letra -> S**
Letra -> c	**Letra -> l**	**Letra -> t**
Letra -> C	**Letra ->L**	**Letra -> T**
Letra -> d	**Letra -> m**	**Letra -> u**
Letra -> D	**Letra -> M**	**Letra -> U**
Letra -> e	**Letra -> n**	**Letra -> v**
Letra -> E	**Letra -> N**	**Letra -> V**
Letra -> f	**Letra -> ñ**	**Letra -> w**
Letra -> F	**Letra -> Ñ**	**Letra -> W**
Letra -> g	**Letra ->o**	**Letra -> x**
Letra -> G	**Letra ->O**	**Letra -> X**
Letra -> h	**Letra -> p**	**Letra -> y**
Letra -> H	**Letra ->P**	**Letra -> Y**
Letra -> i	**Letra ->q**	**Letra -> z**
Letra -> I	**Letra ->Q**	**Letra -> Z**

Santiago Lechuga Mariana
San Juan Escalona Sarai
Zilli Vazquez David Nicodemo

Cuerpo -> Declaración Cuerpo

Declaración -> variable -> booleano ;

Declaración -> variable -> entero ;

Declaración -> variable -> flotante ;

Declaración -> variable -> cadena ;

Declaración -> arreglo -> [contenidoB];

Declaración -> arreglo -> [contenidoE];

Declaración -> arreglo -> [contenidoF];

Declaración -> arreglo -> [contenidoC];

variable -> &nombre_

arreglo -> &nombre[]_

nombre -> Letra

nombre -> Letra Número

nombre -> Letra nombre

Número -> dígito

Número -> dígito dígito

Número -> dígito dígito dígito

booleano-> True

booleano -> False

Santiago Lechuga Mariana
San Juan Escalona Sarai
Zilli Vazquez David Nicodemo

entero -> dígito

entero -> dígito entero

flotante -> entero.entero

cadena -> nulle

cadena -> /Letras/

Letras -> Letra

Letras -> Letra Letras

Letras -> Letra ¡!

Letras -> Letra ¡! Letras

Letras ->

Letras -> Letras

cadena -> cadena :: cadena

contenidoB -> booleano

contenidoB -> booleano , contenidoB

contenidoE -> entero

contenidoE -> entero , contenidoE

contenidoF -> flotante

contenidoF -> flotante , contenidoF

contenidoC -> cadena

contenidoC -> cadena , contenidoC

Santiago Lechuga Mariana
San Juan Escalona Sarai
Zilli Vazquez David Nicodemo

dígito -> 0

dígito -> 1

dígito -> 2

dígito -> 3

dígito -> 4

dígito -> 5

dígito -> 6

dígito -> 7

dígito -> 8

digito -> 9

Cuerpo -> Lectura Cuerpo

Lectura -> variable -> ZSRead(cadena);

Cuerpo -> Escritura

Cuerpo -> Escritura Cuerpo

Escritura -> ZSPrint(cadenaP);

cadenaP -> cadena

cadenaP-> variable

cadenaP -> cadena::cadenaP

cadenaP -> cadena :: variable::cadenaP

Santiago Lechuga Mariana
San Juan Escalona Sarai
Zilli Vazquez David Nicodemo

Cuerpo -> Operaciones Cuerpo

Operaciones -> Acumulacion

Operaciones -> variable -> Operación

Acumulación -> variable ->-> operando

Acumulación -> variable ->-> Operación

Operación -> operando operador operando

Operación -> operando operador Operación

Operación -> operando +++

Operación -> operando ---

Operación -> función

Operación -> función operador Operación

función -> Sn(operando)

función -> Sn(Operación)

función -> Cs(operando)

función -> Cs(Operación)

función -> Tn(operando)

función -> Tn(Operación)

función -> Sq(operando)

Santiago Lechuga Mariana
San Juan Escalona Sarai
Zilli Vazquez David Nicodemo

función -> Sq(Operación)

operando -> variable

operando -> entero

operando -> flotante

operador -> ++

operador -> --

operador -> **

operador -> //

operador -> //·

operador -> \\

operador -> ^^

Cuerpo -> Comentario

Cuerpo -> Comentario Cuerpo

Comentario -> @@ Letras

Comentario -> # Letras #

Cuerpo -> Condición Cuerpo

Condición -> ZSIf FunciónLógica CasoOpuesto

FunciónLógica -> (operando operadorL operando){ Cuerpo}

FunciónLógica -> (booleano) { Cuerpo}

CasoOpuesto -> ZSIfOther FunciónLógica

CasoOpuesto -> ZSIfOther FunciónLógica CasoOpuesto

CasoOpuesto -> ZSOther {Cuerpo }

operadorL -> &&
operadorL -> %%
operadorL -> \
operadorL -> ?=
operadorL -> ?\
operadorL -> ?>
operadorL -> ?<
operadorL -> ?>=
operadorL -> ?<=

Cuerpo -> Ciclo Cuerpo

Ciclo -> ZSLoop FunciónLógica

Ciclo -> ZSLoop (varible) in (arreglo) { Cuerpo }

N = { V_0, NombreClase, Letra, Cuerpo, Declaración, variable, arreglo, nombre, Número, booleano, entero, flotante, cadena, Letras, contenidoB, contenidoE, contenidoF, contenidoC, dígito, Lectura, Escritura, cadenaP, Operaciones, Acumulación, Operación, función, operando, operador, Comentario, Condición, FunciónLógica, CasoOpuesto, operadorL, Ciclo}

Santiago Lechuga Mariana
San Juan Escalona Sarai
Zilli Vazquez David Nicodemo

Q = N U Σ = { V_0, NombreClase, Letra, Cuerpo, Declaración, variable, arreglo, nombre, Número, booleano, entero, flotante, cadena, Letras, contenidoB, contenidoE, contenidoF, contenidoC, dígito, Lectura, Escritura, cadenaP, Operaciones, Acumulación, Operación, función, operando, operador, Comentario, Condición, FunciónLógica, CasoOpuesto, operadorL, Ciclo, a, b, c, d, e, f, g, h, i, j, k, l, m, n, ñ, o, p, q, r, s, t, u, v, w, x, y, z, A, B, C, D, E, F, G, H, I, J, K, L, M, N, Ñ, O, P, Q, R, S, T, U, V, W, X, Y, Z, 0, 1, 2, 3, 4, 5, 6, 7, 8, 9, &, _ , $, (,), ;, +, -, /, \, ·, ^, :, ?, >, =, %, @, #, ¡, !, {, }, ,, }

Santiago Lechuga Mariana
San Juan Escalona Sarai
Zilli Vazquez David Nicodemo