Conociendo el lenguaje

Léxico

La siguiente tabla indica las diferentes palabras reconocidas por el lenguaje. El especificar cada tira de texto de acuerdo a lo indicado en esta tabla permitirá un análisis léxico exitoso.

#	Símbolo	Nombre	Patrón	Descripción
1	i	Identificador	letra(letra|digito|_)*	Nombre de variable, o palabra reservada
2	+	operadorsuma	+	Operador aritmético que suma dos operandos
3	-	operadorresta	-	Operador aritmético que resta dos operandos
4	*	operadormultiplicacion	*	Operador aritmético que multiplica dos operandos
5	/	operadordivision	/	Operador aritmético que divide dos operandos
6	^	operadorpotencia	^	Operador aritmético que eleva un operando a la potencia de otro
7	&	operadorand	&	Operador lógico que evalúa la veracidad de dos expresiones
8 |	operadoror	|	Operador lógico que evalúa la veracidad de al menos una de dos expresiones
9	<	menorque	<	Operador relacional que evalúa si la primera expresión es menor que la segunda
10	≤	menorigualque	<=	Operador relacional que evalúa si la primera expresión es menor o igual que la segunda
11	=	asignacion	=	Operador aritmético que asigna el valor del segundo operando al primero
12	¿	igualque	==	Operador relacional que evalúa si la primera expresión es igual que la segunda
13	>	mayorque	>	Operador relacional que evalúa si la primera expresión es mayor que la segunda
14	≥	mayorigualque	>=	Operador relacional que evalúa si la primera expresión es mayor o igual que la segunda
15	#	diferenteque	!=	Operador relacional que evalúa si la primera expresión es diferente que la segunda

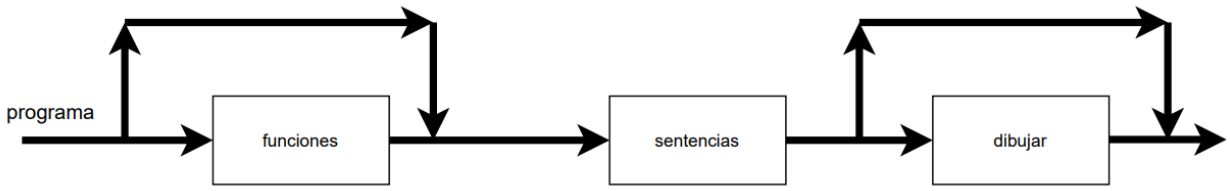
16	%	negacion	!	Operador lógico que invierte el resultado de una expresión
17	a	incremento	++	Operador aritmético (unario) que incrementa por uno un operando
18	d	decremento	--	Operador aritmético (unario) que decrementa por uno un operando
19	[	corcheteizq	[	Usado al lado derecho de un arreglo, seguido por el índice (o los índices) deseado
20	]	corcheteder	]	Usado al lado derecho de un arreglo, precedido por el índice (o los índices) deseado
21	p	puntoycoma	;	Usado en bucles para, delimitando sus partes
22	c	coma	,	Usado para separar varios parámetros o argumentos
23	~	dospuntos	:	Usado para indicar el inicio de funciones, estructuras de control y bucles, tras su declaración
24	(	parentesisizq	(	Usado al lado derecho de llamadas a métodos, para especificar el orden de ejecución de operaciones, para retornar un valor de una función, y para la declaración de funciones, estructuras de control y bucles, seguido por los posibles argumentos, parámetros o expresiones requeridos
25	)	parentesisder	)	Usado al lado derecho de llamadas a métodos, para especificar el orden de ejecución de operaciones, para retornar un valor de una función, y para la declaración de funciones, estructuras de control y bucles, precedido por los posibles argumentos, parámetros o expresiones requeridos
26	l	literaldecimal	(+|-)?(1-9)digito*.digito+ | (+|-)?0.digito+	Un número con precisión decimal
27	l	literalentero	(+|-)?(1-9)digito*|0	Un número sin precisión decimal
28	l	literalcadena	"(letra|digito|simbolos)*"	Un conjunto de caracteres, formando una cadena de texto
29	l	literalchar	'letra|digito|simbolos'	Un carácter alfanumérico o simbólico

Palabras reservadas

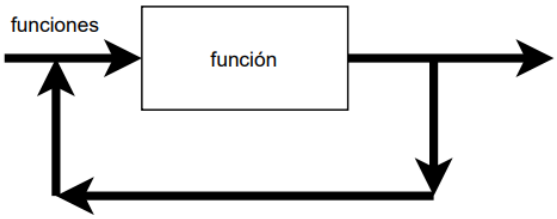
Palabra	Descripción
entero	Tipo de dato entero. (int en C#)
ent	
decimal	Tipo de dato decimal. (decimal en C#)
dec	
carácter	Tipo de dato carácter. (char en C#)
car	
texto	Tipo de dato de cadena de text. (string en C#)
tex	
lógico	Tipo de dato lógico. (bool en C#)
log	
verdad	Literal de tipo booleano
falso	Literal de tipo booleano
textura	Tipo de dato que representa un archivo gráfico.
ras	
col	Tipo de dato que representa un color (COLOR en C#)
color	
si	Estructura de control If
sinosi	Estructura de control else if
sino	Estructura de control else
selección	Estructura de control switch
caso	Parte de la estructura de control switch, case
predeterminado	El caso seleccionado por defecto (default)
fin	Fin de cualquier estructura de control, bucle, o método
mientras	Bucle while
hacer	Bucle do
repetir	Bucle que repite las veces requeridas
para	Bucle for
salir	Sale de un bucle (break)
nada	Tipo de dato retornado (void)
retornar	Retorna un dato de un método
dibujar	Parte del programa que permite dibujar objetos en la pantalla
negro	Color
azul	
verde	
celeste	

rojo	
rosado	
amarillo	
blanco	
pi	Valor de pi
intentar	Bloque de código que se debe intentar
error	Bloque de código que se ejecuta en caso de un error en un bloque previo intentar

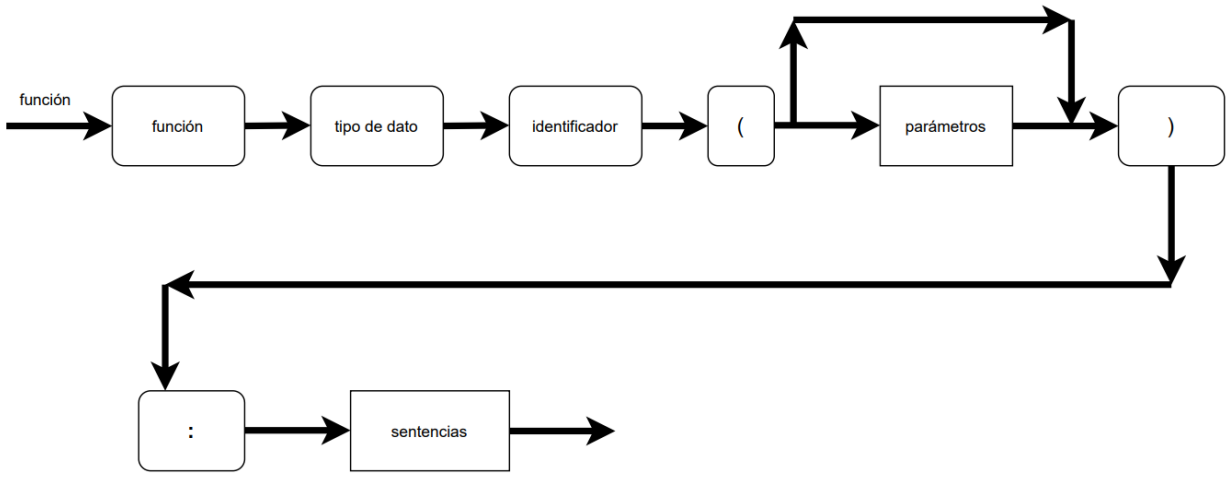
Sintaxis



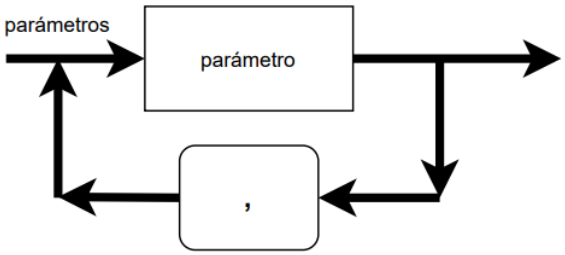
Cada programa *debe* llevar un bloque de sentencias, lo cual *puede* ser precedido por funciones o seguido por un bloque dibujar.



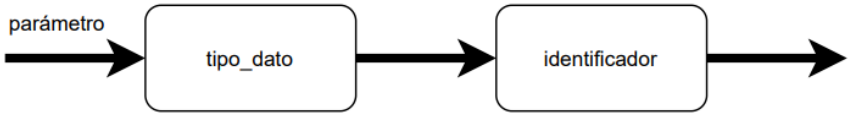
Si existe el espacio de funciones, puede haber una o varias.



Cada función debe comenzar con la palabra “función”, seguida por un tipo de dato (ent, dec, tex, etc.), un identificador (nombre de variable), paréntesis que pueden contener una lista de parámetros, dos puntos y un bloque de sentencias.

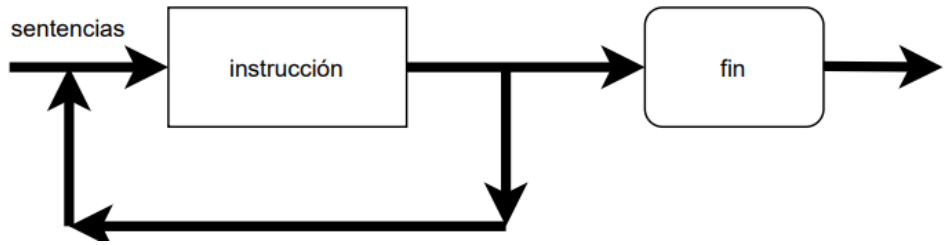


Si se especifica parámetros permitidos en una función, puede haber uno o varios, separados por coma.

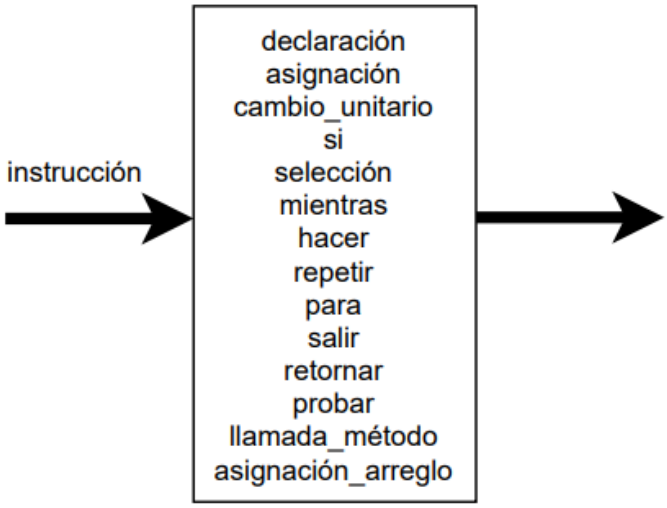


Cada parámetro incluye un tipo de dato y un identificador (nombre).

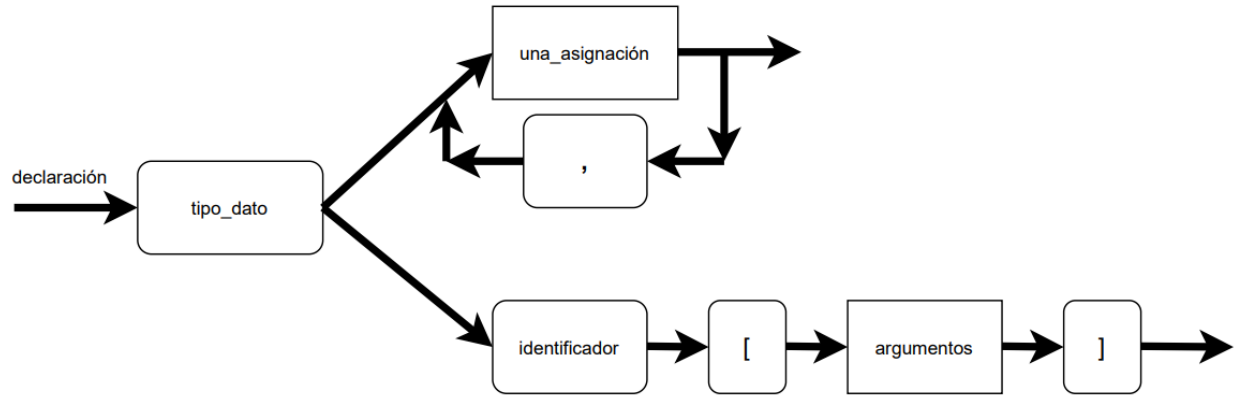
En caso de haber un bloque dibujar, este debe comenzar con la palabra “dibujar” seguido por dos puntos y un bloque de sentencias.

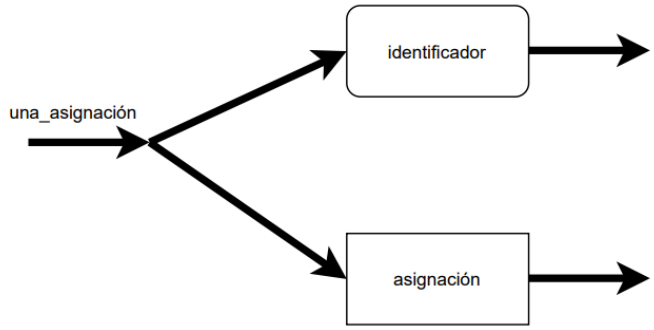


Los bloques de sentencias contienen instrucciones y terminan con la palabra “fin”. Cada instrucción es separada con una línea nueva.

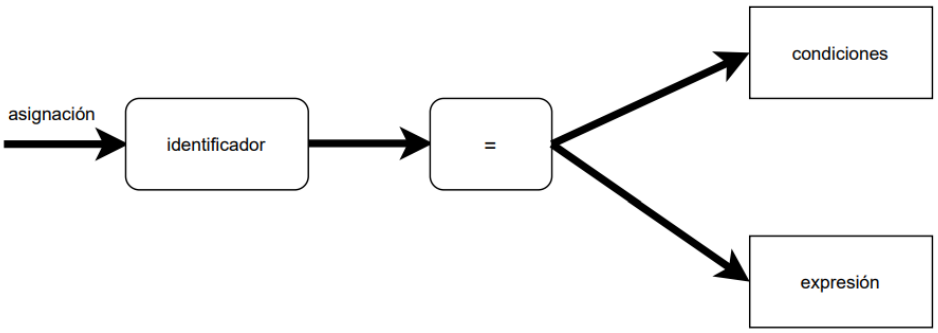


Cada instrucción puede ser una declaración, asignación, cambio unitario, bloque si, bloque selección, bloque mientras, bloque hacer, bloque repetir, bloque para, salir, retornar, bloque probar, llamada a método o asignación de arreglo. A continuación, se especificará cada instrucción.

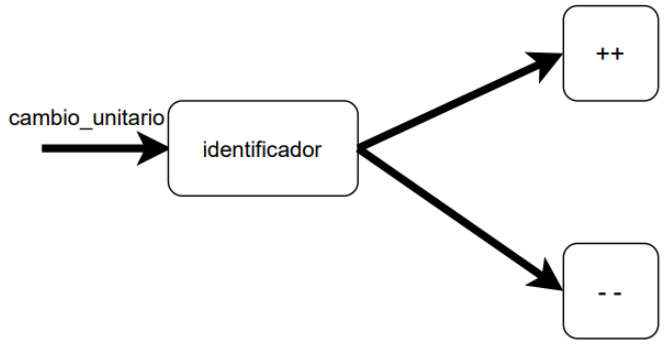




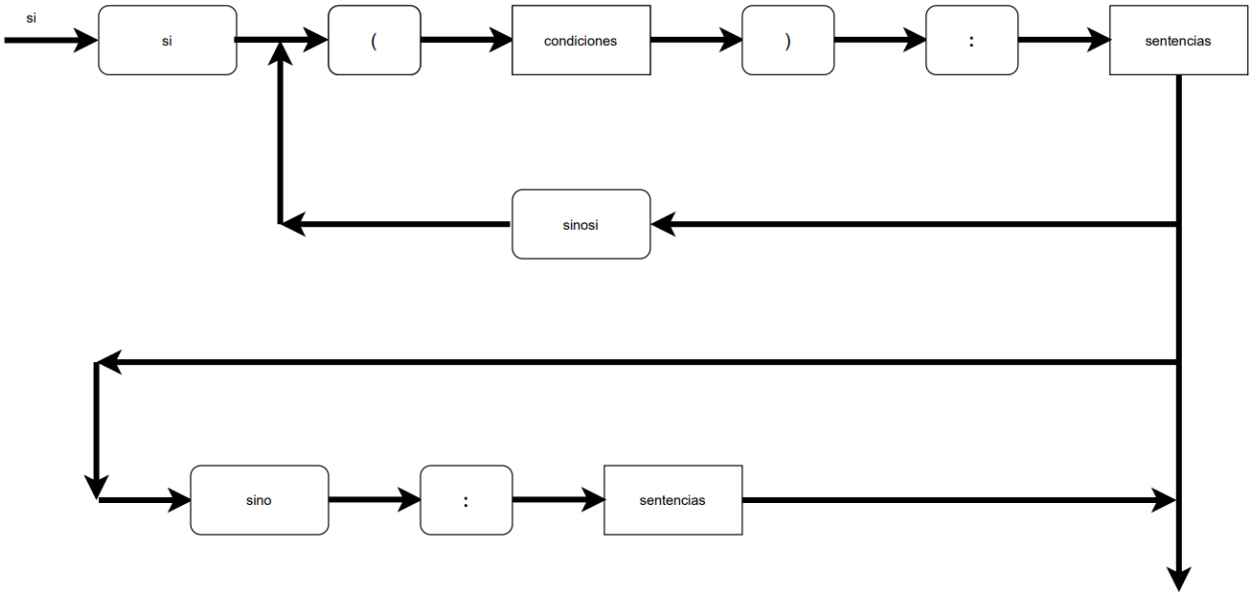
Si se desea declarar variables, se debe comenzar con el tipo de dato seguido por una lista de identificadores con posibles asignaciones, separados por comas. Si se desea declarar un arreglo, se debe escribir un tipo de dato seguido por un identificador (nombre) y se debe especificar una lista de expresiones que determinen el tamaño del arreglo.



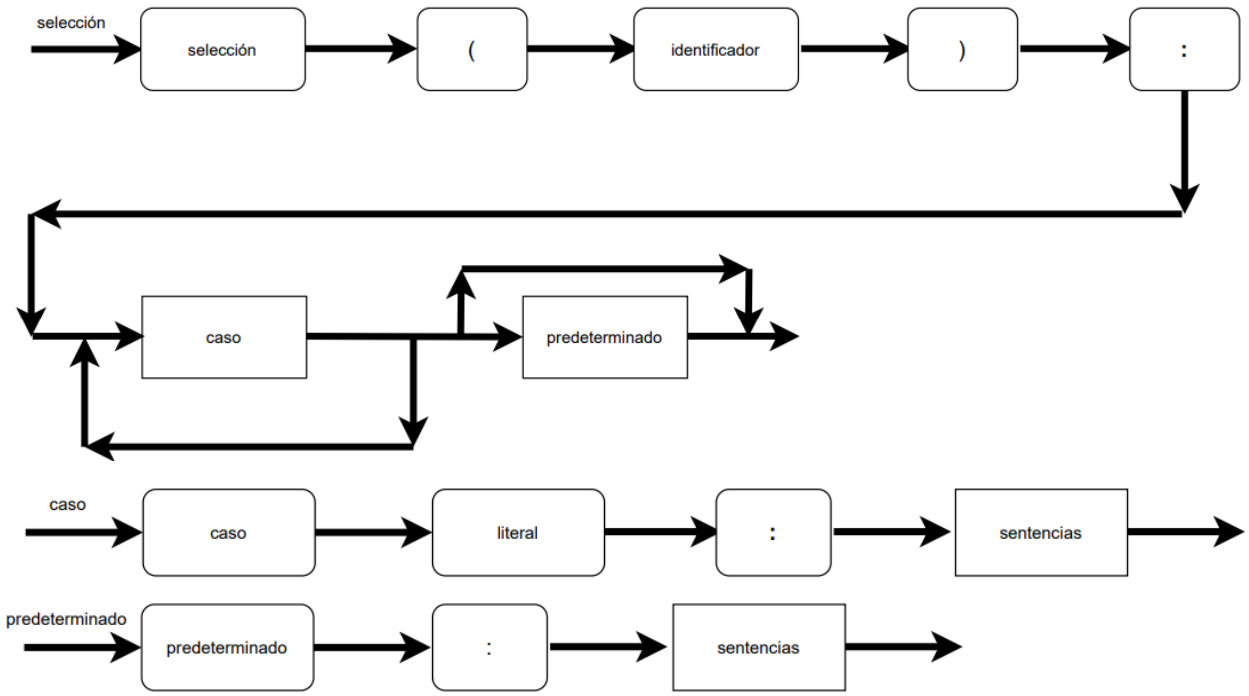
Una asignación empieza con un identificador (nombre de variable), seguido por el signo igual y una lista de condiciones (tipo lógico) o una expresión.



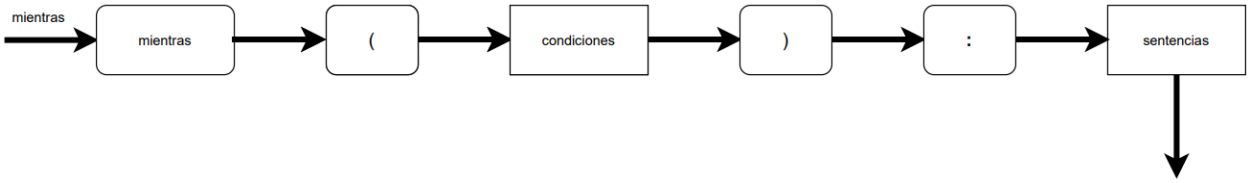
Las instrucciones de cambios unitarios empiezan con un identificador (nombre de variable) y se los siguen con un operador de cambio unitario (++ o - -).



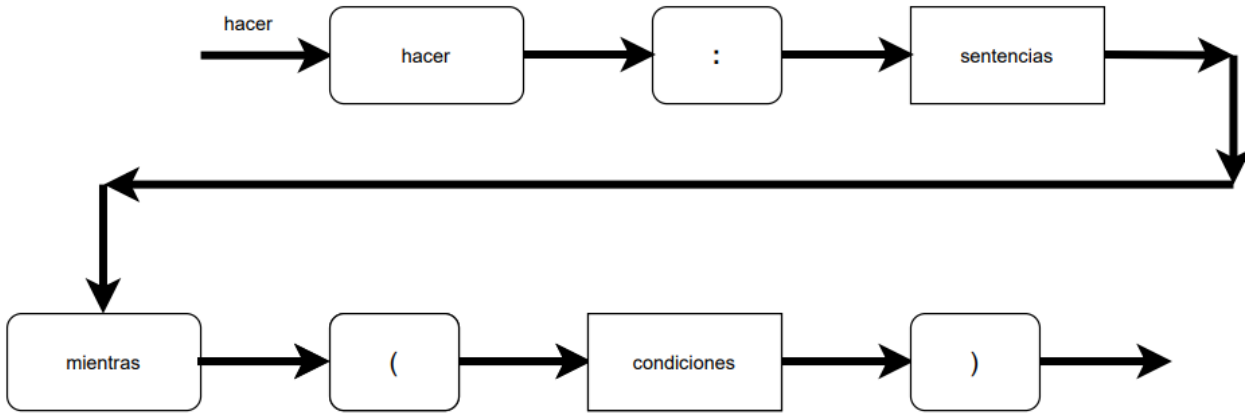
Un bloque si comienza con la palabra “si” y es seguida con una lista de condiciones entre paréntesis, dos puntos y un bloque de sentencias. De forma *opcional*, se puede añadir una lista de bloques sinosi con la palabra “sinosi”, una lista de condiciones entre paréntesis, dos puntos y un bloque de sentencias, y/o un bloque sino, que empieza con la palabra “sino” seguido por dos puntos y un bloque de sentencias.



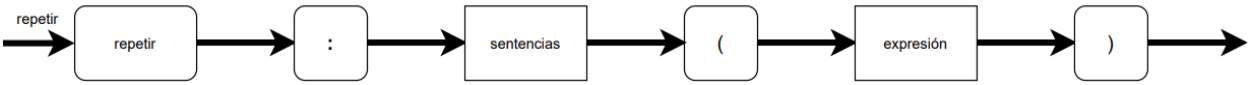
Un bloque selección comienza con la palabra “selección”, un identificador entre paréntesis, dos puntos, una lista de casos y *opcionalmente* un caso predeterminado. Cada caso incluye la palabra “caso”, un literal (un número, texto, un carácter, etc.), dos puntos y un bloque de sentencias. El caso predeterminado es la palabra “predeterminado”, dos puntos y un bloque de sentencias.



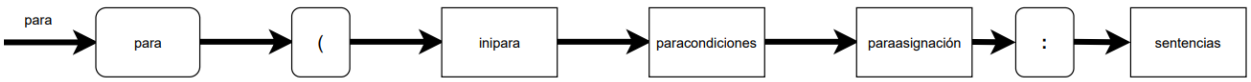
Un bloque mientras comienza con la palabra “mientras” seguida por una lista de condiciones entre paréntesis, dos puntos y un bloque de sentencias.

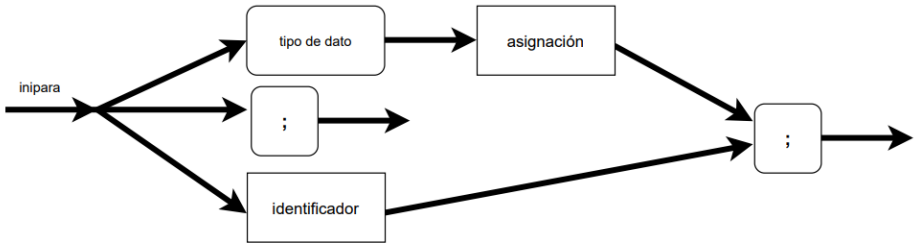


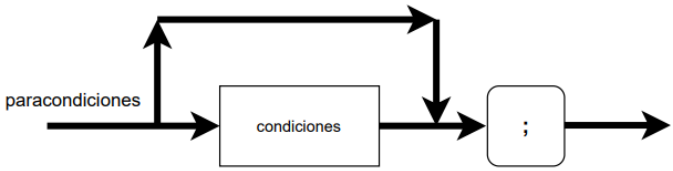
Un bloque hacer comienza con la palabra “hacer” seguido por dos puntos, un bloque de sentencias, la palabra “mientras” y una lista de condiciones entre paréntesis.

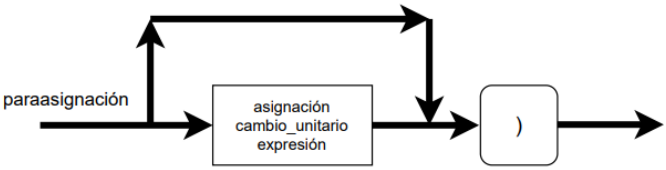


Un bloque repetir empieza con la palabra “repetir” seguido por dos puntos, un bloque de sentencias y una expresión entre paréntesis.

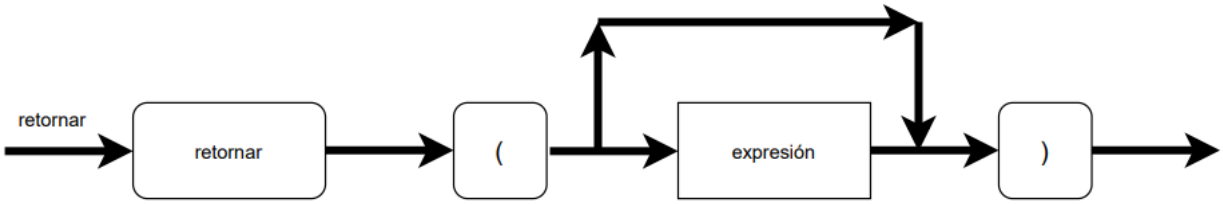






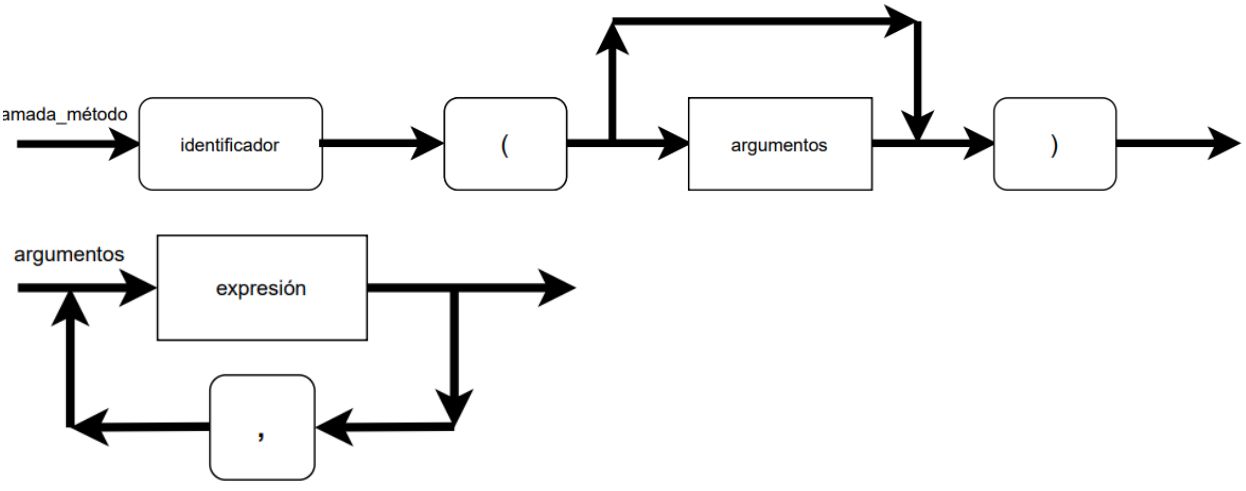


Un bloque para parece ser el más complicado, pero es quizá el más útil. Empieza con la palabra para, seguida por inipara, paracondiciones y paraasignación entre paréntesis, seguidos por dos puntos y un bloque de sentencias. Inipara puede ser un tipo de dato seguido por una asignación y punto y coma, un identificador seguido por punto y coma, o simplemente punto y coma. Paracondiciones *puede* incluir una lista de condiciones, y *debe* incluir al final punto y coma. Paraasignación *puede* incluir una asignación, cambio unitario o expresión, y *debe* terminar con paréntesis derecho.

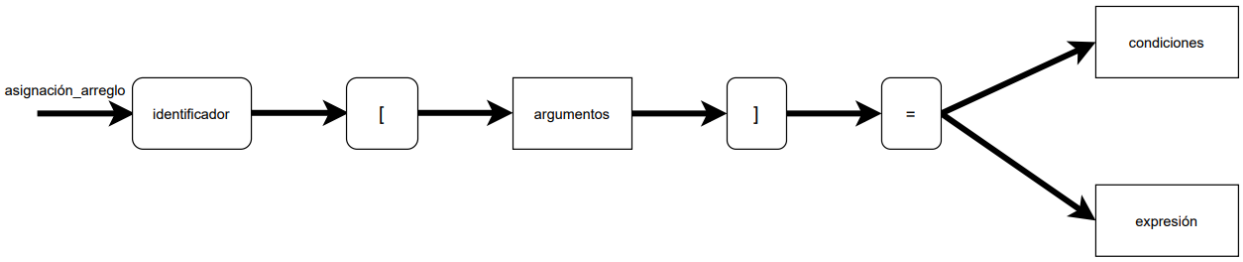


Para una instrucción retornar, se debe comenzar con la palabra “retornar” seguido por paréntesis. Los paréntesis deben incluir una expresión, salvo que la función sea de tipo “nada”; en este caso, no se debe especificar ninguna expresión.

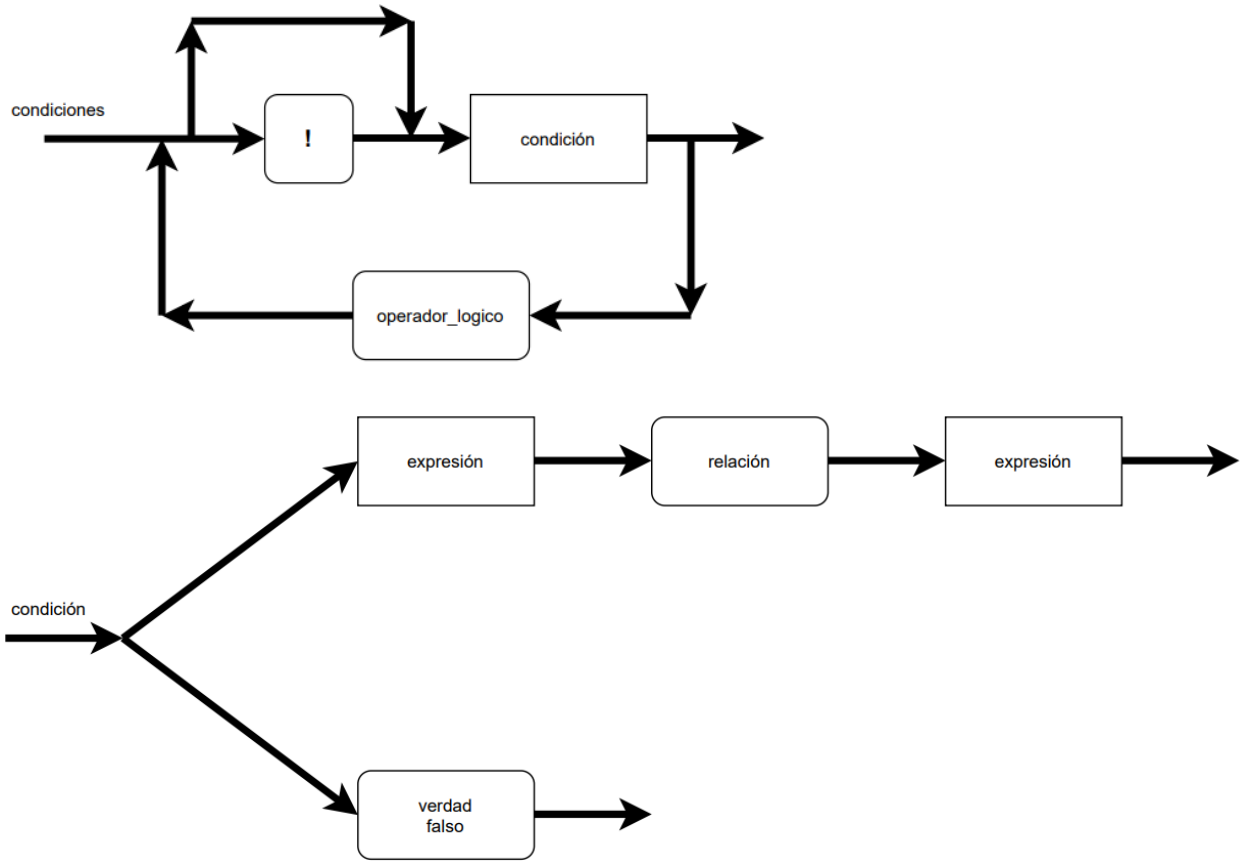
Un bloque probar debe comenzar con la palabra reservada “probar” seguida por dos puntos, un bloque de sentencias, la palabra “error”, dos puntos y un bloque de sentencias.



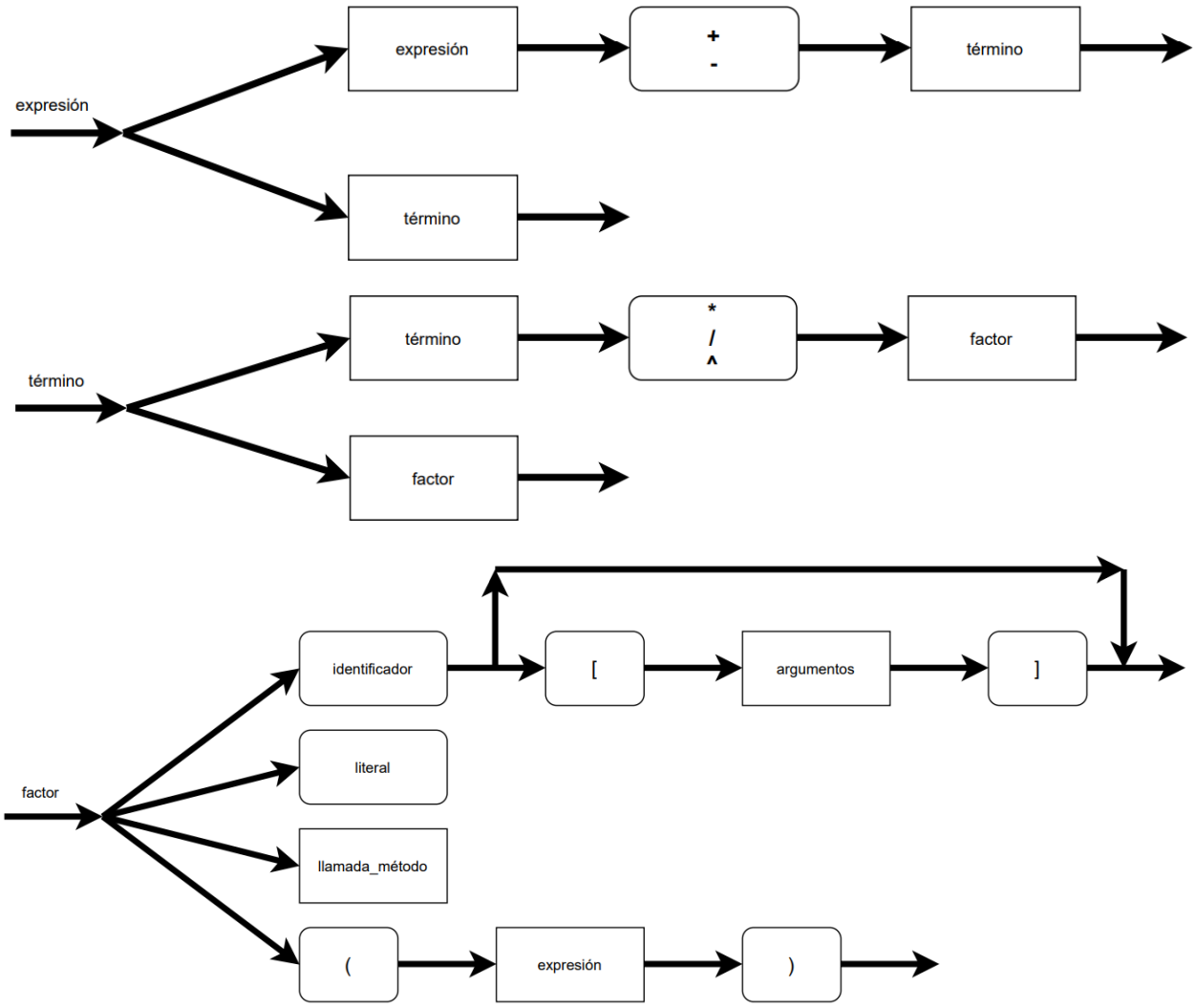
Una llamada a método debe incluir un identificador y una *posible* lista de argumentos entre paréntesis. Los argumentos son expresiones separadas con coma.



Una asignación de arreglo empieza con un identificador seguido por una lista de argumentos entre llaves (que especifiquen el índice correcto del arreglo), el signo igual y una lista de condiciones o una expresión.



Respecto a las listas de condiciones, se las separa mediante operadores lógicos (& y |). *Opcionalmente*, pueden empezar con una negación. Las condiciones pueden ser dos expresiones unidas con un operador relacional (<, <=, ==, !=, >= o >), verdad o falso.



Una expresión puede ser un término o una expresión más o menos un término. Un término puede ser un factor o un término por, para o elevado a un factor. Un factor puede ser un identificador (en caso de una variable), un identificador seguido por argumentos entre llaves (un arreglo), un literal (un número, texto, etc.), una llamada a método o una expresión entre paréntesis.

Semántica

Operaciones permitidas entre operandos

Primer Operando	Operador Aritmético	Segundo Operando	Resultado
entero	+, -, *, /,^,=	entero	entero
		carácter	
	+, -, *, /	decimal	decimal
	+	texto	texto
decimal	+, -, *, /,^,=	entero	decimal
		decimal	
		carácter	
	+	texto	texto
texto	+	entero	texto
		decimal	
		carácter	
	+,=	texto	
carácter	+, -, *, /,^	entero	entero
		decimal	decimal
	+	texto	texto
	+, -, *, /,^	carácter	entero
	=		carácter
lógico	=	lógico	lógico
textura	=	textura	textura
color	=	color	color

Especificaciones sobre el bloque dibujar

Si usted usa el bloque dibujar, debe tomar en cuenta las siguientes consideraciones:

1. No funcionan los métodos “escribir” y “leer” de la consola. Si se puede compilar, pero estos métodos no tendrán ningún efecto sobre la ejecución del programa.
2. El código contenido dentro del bloque principal de instrucciones se ejecuta una vez. Tras su ejecución, el código dentro del bloque dibujar se ejecutará diez veces por segundo.
3. Las variables del código de cada función, el bloque principal de sentencias y el bloque dibujar son independientes de sí, y no hay visibilidad entre ellas, al menos que se especifique variables globales. Para especificar una variable global que funcione entre el bloque principal de instrucciones y el bloque dibujar, simplemente se la debe declarar sin asignación principal dentro del bloque principal.
   Por ejemplo:

```
        ras auto
        auto = leerras(“auto.png”)
        ent x, y
        x = 10
        y = 20
    fin
    dibujar:
        dibujarras(auto, x, y, blanco)
        x++
        y = y + 3
    fin
```

4. Actualmente, solo funcionan los gráficos de tipo .png.
5. No se puede usar el método “dibujarras” fuera del bloque dibujar.
6. El tipo de dato textura/ras y los métodos leerras y dibujarras no se pueden usar en un programa que no contenga bloque dibujar.
7. Si desea dibujar un pixel sobre la pantalla (muy útil para dibujar funciones, líneas, ruido, etc.), se puede cargar “dot.png”.

Funciones preprogramadas

Palabra	Descripción	Entradas	Salida	Resultado
sen	Función que calcula seno	A: Decimal - radianes	Función de seno sobre A (Decimal)	
cos	Función que calcula coseno	A: Decimal - radianes	Función de coseno sobre A (Decimal)	
tan	Función que calcula tangente	A: Decimal - radianes	Función de tangente sobre A (Decimal)	
potencia	Función que calcula potencia	A: Número decimal a elevar, B: número decimal de exponente	A^B (Decimal)	
esperar	Función que espera el tiempo deseado	A: Entero - milisegundos	Ninguna	La ejecución del programa se detiene el tiempo ingresado
raízcuadrada	Función que calcula la raíz cuadrada	A: Decimal	Función de raíz cuadrada sobre A (Decimal)	
leer	Función que lee de la consola	Ninguna	Texto dentro de la línea de la consola (texto)	
escribir	Función que escribe a la consola	A: Texto	Ninguna	El texto indicado se escribe en la consola
convertir	Función que convierte de un tipo de dato a otro	A: (variable o literal de cualquier tipo de dato)	Devuelve la conversión del tipo de dato al tipo detectado automáticamente	
reproducir	Función que reproduce un archivo de audio .wav	A: Texto - Ubicación del archivo de audio a reproducir	Ninguna	Se reproduce el archivo de audio
sonido	Función que hace un sonido	A: Entero - frecuencia del sonido, B: Entero - duración del sonido en milisegundos	Ninguna	Se produce un sonido a frecuencia A durante duración B
leerras	Función que importa un archivo gráfico	A: Text - Ubicación del archivo gráfico a importar	Textura	
leertexto	Función que lee todo el texto de un archivo	A: Texto - Ubicación del archivo de texto a leer	B: Texto - Todo el texto del archivo	

escribirtexto	Función que escribe texto a un archivo	A: Texto - El texto a escribir, B: Texto - La ubicación del archivo de texto	Ninguna	Escribe el texto indicado a un archivo de texto
dibujarras	Función que dibuja una textura en la pantalla	A: Archivo gráfico , B: Decimal - Coordenada X, C: Decimal - Coordenada Y, D: Color	Ninguna	Dibuja una textura 2D en la pantalla en las coordenadas dadas
longitud	Función que calcula la longitud de un objeto (texto, arreglo)	A: Texto o Arreglo	A: Entero - La cantidad de elementos que contenga la entrada	
azar	Función que devuelve un valor aleatorio	A: Ninguna	A: Entero - Un número entero positivo cualquier	
		A: Entero - máximo	A: Entero - Un número entero entre cero (inclusivo) y el número máximo (exclusivo)	
		A: Entero - mínimo, B: Entero - máximo	A: Entero - Un número entero entre el número mínimo (inclusivo) y el número máximo (exclusivo)	
convertirEnt	Función que devuelve un valor entero	A: Dato a convertir	A: Entero	
convertirDec	Función que devuelve un valor decimal		A: Decimal	
convertirTex	Función que devuelve un valor texto		A: Texto	
convertirCar	Función que devuelve un valor carácter		A: Carácter	
convertirLog	Función que devuelve un valor lógico		A: Lógico	