Javier Pellejero y Luis Aguirre

Especificación del lenguaje

Vamos a diseñar un lenguaje basado en C. Se utilizará la sintaxis de C siempre que sea posible y no dé lugar a posibles ambigüedades.
Las principales características de nuestro lenguaje son:

1.-Bloques:

Los bloques van a ser delimitados mediante **“{}”** al igual que en C. Todo programa debe tener obligatoriamente un bloque principal **“main”.** Además se permitirá el anidamiento de bloques.

2.-Declaración de variables y tipos:

Por comodidad vamos a exigir que todas las variables que vayan a ser usadas por un bloque se declaren al principio de cada bloque mediante la palabra reservada **“decVar: { ”.** Seguidamente se declararán todas las variables, separadas por un salto de línea de la siguiente forma: **“tipoVar nombreVar;”.** El bloque de declaración de variables acabará simplemente con **“}”.**

Por supuesto los bloques tienen acceso a las variables de sus bloques “padres” ,si están anidados, pero no al revés.

En cuanto a los tipos, se van a considerar los tipos básicos: **int**, **float** y **bool.** También se permitirá declarar vectores de tamaño fijo mediante la siguiente expresión: **“tipoVar nombreVector[tam]**”. Se da la posibilidad de declarar vectores de vectores. Simplemente: **tipoVar nombreVector[tam][tam],** para un vector de dos dimensiones, por ejemplo.

Por otro lado, si queremos que una variable sea constante, podemos hacerlo mediante la palabra reservada **“const”**. Todas las constantes han de ser declaradas al principio del programa, antes del bloque principal **main.**

3-. Operadores:

Nuestro lenguaje incluye los operadores básicos para los tipos que hemos considerado. En cuanto a los tipos numéricos **int** y **float,** vamos a tener los operadores: multiplicación (*****), división (**/**), suma (**+**) y resta (**-**). La prioridad de éstos es la usual. Multiplicación y división tienen mayor prioridad que suma y resta, y en caso de empate, tendrá mayor prioridad el operador que aparezca antes en el texto del programa. Vamos a exigir que al ser operadores binarios, ambos operandos sean del mismo tipo.
Para los **bool** tenemos los operadores conjunción (**&&**), disyunción (**||**), negación (**!!**) y xor (**^^**).
Además, podremos comparar variables del mismo tipo mediante: igualdad (**==**), no igualdad (**!=**), menor estricto (**<**) y menor o igual (**<=**)

Por último, para expresiones complejas, podremos utilizar "**()**" que es el operador con mayor prioridad de todos los mencionados previamente para indiciar que parte de las expresión debe ser evaluada antes.

4. Instrucciones:

Las principales instrucciones contempladas en nuestro lenguaje son:
-Asignación: nos permite asignar un valor a una variable. Se utilizará "**=**", y se exige que el valor sea del mismo tipo que la variable, y que la variable haya sido previamente declarada.
-Condición: adoptamos la sintaxis de C y denotaremos los bloques condicionales de la siguiente forma:

```
if(condición) {
        instrucciones
} else {
        instrucciones;
```

}

La rama **“else”** puede ser omitida.

-Acceso elemento vector: para acceder a la posición i-ésima de un vector, utilizaremos “[]” de la siguiente forma: **nombreVector[i]**;

-Bucle: sólo vamos a tener un tipo de bucle, “while”, que seguirá la siguiente sintaxis:

while (condición) {
instrucciones;
}

5. Llamadas a funciones:

Tendremos, por último, funciones externas al programa principal. Las denotaremos como:
function (int, bool, float, void) nombreFuncion (tipoVar nombreVar, tipoVar nombreVar) {. A continuación, en las siguientes líneas se añadirían las instrucciones a realizar por dicha función. Para finalizar, si la función no es de tipo **void**, deberá devolver un valor del tipo correspondiente con la instrucción: **return nombreVar;** siempre al final de la función, para acabar con **}**. En caso de no devolver nada (tipo **void**), simplemente acabaremos con **}**.

Cada función tendrá un comportamiento exactamente igual al definido para el programa principal.

Además, si queremos llamar a dichas funciones desde el programa principal lo haremos con la instrucción **start nombreFuncion(...);** asignándola a una variable del mismo tipo si esta devolviera algún valor.

6. Ejemplos:

EJEMPLO 1

```
const int BASE = 2;
main {
	decVar: {
		int b;
		int i;
	}
	b = 1;
	i = 1;
	while (i <= 10) {
		b = b * BASE;
		i = i + 1;
	}
}
```

EJEMPLO 2

```
const int BASE = 2;
function int potencia (int base, int exp) {
	decVar: {
		int i; int resultado;
	}
	i = 0;
	resultado = 1;
	while (i < exp) {
		resultado = resultado * base;
		i = i + 1;
	}
	return resultado;
}
```

```
main {
	decVar: {
		int e; int b;
	}
	e = 10;
	b = start potencia(BASE, e);
}
```