

Traductor de las operaciones con números Romanos.

Sea un lenguaje especializado en operaciones con números romanos. La estructura quedaría de la siguiente manera:

```
programa ()
{
  /* lista de instrucciones separadas por ; */
}
```

Las instrucciones pueden ser de tipo declaración, asignación o impresión por pantalla.

Una declaración de variables se estructura de la forma:

tipo id1, id2, id3, ... ,idn;

Es decir, se especifica primeramente el tipo de datos, luego cada uno de los identificadores de las variables.

Los tipos pueden ser rom o int. Las variables de tipo rom son las que se utilizarán para almacenar los números romanos y realizar con ellas las operaciones básicas entre esos números. Las variables de tipo int para almacenar las variables de tipo enteras.

Existen dos tipos de asignación, la asignación a variables de tipo int y la asignación de variables de tipo rom.

En la instrucción de asignación para el tipo int puede tener la siguiente forma:

variable1 = variable2;

Donde variable1 y variable2 deben haber sido declaradas anteriormente de tipo int.

En la instrucción de asignación para el tipo rom puede tener tres formas:

- Cuando se asignan dos variables :

variable1 = variable2;

donde ambas debe ser del tipo número romano.

- Cuando se le asigna a un número romano un entero casteado a número romano.

variable1 = (rom)variable2;

donde variable1 es de tipo rom y variable2 es de tipo int

- Cuando se le asigna a un número romano una expresión entre números romanos.

Variable1 = <expresión entre números romanos>;

Las operaciones entre números romanos solo se realiza entre variables de ese tipo. Los operadores permitidos son los siguientes:

1. + : Suma
2. - : Resta
3. * : Multiplicación.
4. / : División.

La semántica de los operadores es la misma que la de los operadores aritméticos, se pueden utilizar

paréntesis para encerrar alguna operación prioritaria sobre otra.

La instrucción de impresión por pantalla tiene la siguiente sintaxis:
imprimir variable;

A continuación se muestra un ejemplo de un programa escrito en este lenguaje:

```
programa ()
{
int a , b;
rom num1 , num2, num3, resultado;

a = 1;
b = 2;

num1 = XX;
num2 = (rom)a ;
num3 = (rom)b + num1;

resultado = ((num1 + num2) * num3)/C;
print resultado;

}
```

El objetivo de la tarea extra-clase (TE) es el de diseñar e implementar un intérprete para el lenguaje definido, que incluya:

- Un editor donde codificar el programa.
 - Salvar y cargar de fichero el programa.
 - Funcionalidades para mostrar los errores de cada una de las fases de la compilación.
 - Una consola donde mostrar las salidas del programa.
 - Funciones para ejecutar el programa.
-
- Cuando se ejecute el programa, si tiene errores, el sistema debe señalar la línea del programa donde ocurrió el error y clasificar ese error como léxico, sintáctico o semántico.

Existe bonificaciones adicionales a la nota si se satisfacen algunos de los siguientes elementos:

- El editor podrá señalar en negrita u otro estilo las palabras reservadas del lenguaje.
- Podrá agregar otro tipos de instrucciones según estime conveniente al lenguaje para
- Podrá visualizar el AST que representa el lenguaje usando un componente adecuado o

Otras facilidades del IDE o del lenguaje que estime pueda tener. Ejemplo: un depurador (debugger).