* Un programa es una secuencia de variables y definiciones de funciones. // Pueden ir mezclado
* La sintaxis de la definición de una variable es un tipo seguido de lista, con al menos un elemento, de identificadores separados por comas. Las definiciones de variables terminan mediante el caracter ';' 1 +cs
* Las funciones se definen especificando el tipo de retorno, el identificador de la función y una lista de parametros separados por comas entre '(' y ')'. 1+cs
* El tipo de retorno y de los parámetros debe ser de tipo simple (no se permiten ni arrays ni registros). El cuerpo de la funcion va entre '{' y '}'.
* El cuerpo de las funciones son secuencias de definiciones de variables seguidos por una secuencia de sentencias. //Todas las variables tienen que declararse al principio del método
* En ambos casos terminan con el caracter ';'
* La funcion "main" debe estar la última, retornando void y sin parámetros.
* Los tipos simples son "int", "double" y "char".
* Los arrays se pueden crear con el constructor de tipos "[]", siguiendo la sintaxis Java pero especificando el tamaño con un entero (como en C, es obligatorio).
* Se usa el constructor de tipos "struct" para la creacion de registros. //Dentro de un struct puede ir otro struct
* Los registros no tienen identificador, y los campos son declarados como variables dentro de '{' '}'. No se permite la definicion de tipos (p.e. typedef)
* Una escritura es la palabra reservada "write" seguida de una lista de expresiones separadas por comas.
* Una lectura es la palabra reservada "read" seguida de una lista de expresiones separadas por comas.
* Una asignación está compuesta por dos expresiones separadas por el operador '='.
* La sentencia condicional "if"-"else" y la sentencia iterativa "while" siguen la sintaxis del lenguaje C. // Si solo tienen una linea se omiten los {}
* Se permite la sentencia 'return' <expresion> (la expresion es obligatoria).
* La invocación a una función será una nueva expresión cuando retorne un valor (distinto de void).
* Un procedimiento (retorno void) será siempre sentencia. // A veces sera una sentencia , otras una función
* La conversión explícita a tipos simples (cast) se realizará utilizando la sintaxis del lenguaje C. // Puedes perder informacion, tiene que hacerlo tambien cuando no se indica
* Las expresiones están formadas por:
  + Constantes enteras, reales y caracter sin signo.
  + Identificadores
  + Los siguientes operadores aplicados a una o dos expresiones (por orden de precedencia descendente):
    - ( ) Sin asociatividad
    - . A Izquierdas //Aceder a campos
    - [] Sin asociatividad
    - - (unario) A Derechas
    - \* / % A Izquierdas
    - + - A Izquierdas
    - > >= < <= != == A Izquierdas
    - && || ! A Izquierdas
    - = A derechas

a[4]-b[3].a.c = R[4]\*B+A\*B.C[4]