Práctica 1 - PL

1. Descripción del lenguaje a desarrollar:

El lenguaje se puede dividir en dos secciones: común y específica. La sección común es similar para todos los grupos y la específica es única, siguiendo unas reglas definidas por un código.

En este caso, el código es “BABAD”.

* Común:
  + Es un subconjunto de un lenguaje de programación estructurado.
  + Los identificadores se declaran antes de ser usados.
  + Tipos de datos mínimos: entero, real, carácter, booleano.

Los enteros y reales podrán realizar las siguientes operaciones: suma, resta, producto, división, operaciones de relación.

Los booleanos pues realizar las siguientes operaciones: and, or, not, xor.

* + Todas las expresiones disponen de la sentencia de asignación.
  + Permite expresiones aritméticas lógicas.
  + Tiene una sentencia de entrada y otra de salida.
  + Dispone de las estructuras de control: if-then-else, while.
  + La estructura sintáctica es: <Programa> ::= <Cabecera\_programa> <bloque>
  + Se permite el anidamiento de bloques.
  + La comprobación de tipos es fuertemente tipado.
  + En los argumentos de un subprograma los parámetros se pasan por valor.
  + No se permiten declaraciones fuera de los bloques, teniendo que ir entre marcas de inicio y de fin.
* Específico (lenguaje “BABAD”):
  + La sintaxis está inspirada en lenguaje C.
  + Las palabras reservadas se encuentran en castellano.
  + Debe disponer de un Array de una y dos dimensiones con las operaciones: acceso a un elemento, producto, suma, resta elemento a elemento, producto de un array por un escalar y producto de matrices.
  + Deben existir las constantes de tipo Array.
  + Los subprogramas deben ser funciones.
  + Las estructuras de control adicional son: do-until.

1. Descripción formal de la sintaxis del lenguaje usando BNF:
2. Definición de la semántica en lenguaje natural:

El programa comienza con una cabecera inicial y un bloque. La cabecera inicial está formada por un entero, la palabra reservada *principal* seguida de paréntesis sin argumentos.

1. Identificación de los tokens:
2. Palabras:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| { | = | booleano | \* | verdadero |
| } | [ | caracter | / | falso |
| ; | ] | flotante | == | . |
| , | si | leer | != | principal |
| ( | si\_no | escribir | <= |  |
| ) | hacer | & | >= |  |
| ini\_var\_local | hasta | + | < |  |
| fin\_var\_local | mientras | - | > |  |
| ini\_bloque | devolver | ~ | && |  |
| fin\_bloque | entero | ! | || |  |

1. Tokens:

|  |  |
| --- | --- |
| Tokens | Palabra |
| PRINCIPAL | principal |
| INI\_MATRIZ | { |
| FIN\_MATRIZ | } |
| PUNTO\_Y\_COMA | ; |
| COMA | , |
| PARENT\_IZQUIERDO | ( |
| PARENT\_DERECHO | ) |
| INI\_VAR\_LOCAL | ini\_var\_local |
| FIN\_VAR\_LOCAL | fin\_var\_local |
| INI\_BLOQUE | ini\_bloque |
| FIN\_BLOQUE | fin\_bloque |
| ASIGNACION | = |
| INI\_DIM\_MATRIZ | [ |
| FIN\_DIM\_MATRIZ | ] |
| SI | si |
| SI\_NO | si\_no |
| HACER | hacer |
| HASTA | hasta |
| MIENTRAS | mientras |
| DEVOLVER | devolver |
| TIPO\_BASICO | entero booleano caracter flotante |
| ENTRADA | leer |
| SALIDA | escribir |
| SIGNO | + - |
| OP\_UNARIO | & ~ ! |
| OP\_BINARIO | \* / == != <= >= < > && || |
| VAR\_LOGICA | verdadero falso |
| DECIMAL | . |