Parser de Subset de Python en Prolog

Primer Entregable

En esta primera parte del proyecto deberán de reconocer si un string provisto es o no es una expresión matemática básica en lenguaje Python. Escribir este reconocedor les ayudará a escribir la base del resto del proyecto.

- Un string no es otra cosa que una secuencia de caracteres. Primero deben de poder agrupar estos caracteres basados en ciertas reglas que les permitirán tener ahora una secuencia de tokens. Por ejemplo, la expresión *"result = oldsum - value / 100"* resulta en la siguiente lista de tokens:
    - ["result", "=", "oldsum", "-", "value", "/", "100"]

    Estas reglas para separar tokens no son muy complejas, existe una lista de caracteres bien especifica, en el ejemplo se puede observar que todo está separado limpiamente por espacios pero en general los operadores también pueden separarse de los otros tokens sin necesidad de *whitespace*.

    Deben de escribir este *tokenizer* primero.
- El siguiente paso es lograr identificar cada token. Para los operadores esto no es complicado. En el caso de identificadores tienen que buscar las reglas del lenguaje que les permiten nombrar identificadores (https://www.w3schools.com/python/gloss_python_variable_names.asp). De esta manera logramos añadir información crucial a la lista anterior.
    - Identificador: *result*
    - Operador de Asignación: *=*
    - Identificador: *result*
    - Operador de Resta: -
    - Identificador: *value*
    - Operador de División: */*
    - Literal Entero: *100*
- A continuación las reglas del lenguaje determinan como se combinan ciertos tokens para construir expresiones validas.
    - La siguiente gramática describe operaciones matemáticas, respeta la precedencia de operaciones y permite el uso de paréntesis:

    <assignStmt> --> <id> = <expr>
    <expr> --> <expr> <op1> <expr1> | <expr1>
    <expr1> --> <expr1> <op2> <expr0> | <expr0>
    <expr0> --> <id> | <entero> | <numDecimal> | (<expr>)
    <op2> --> * | /
    <op1> --> + | -

El objetivo de su primer avance es recibir un string e indicar si es o no es un <assignStmt> válido según esta gramática. En los siguientes avances deberán construir las piezas de la gramática que hacen el subset de Python un poco más complejo:

- Soporte para otros tipos de statements (expandir lo anterior para incluir operaciones lógicas y permitir llamadas a funciones).
- Manejo de un programa con múltiples *statements (líneas de código)*
- Soporte para lazos, condicionales, definir y utilizar listas, diccionarios y arreglos.
- Otros por discutir.