Tarea Programada Número 3

Componente 1

Implementar nuevas estructuras de control:

en el compilador de triangle, las siguientes comandos y expresiones

```
For    ::=  for V-name := Expression to Expression
               [by Integer-Literal]
             do  single-Command

Repeat ::= repeat Command until Expression

Do     ::= do Command while Expression

case-Command ::= case V-name of
                   ( (Integer-Literal | Character-Literal)
                        : single-Command )+
                 end

case-Expression ::= case V-name of
                   ( (Integer-Literal | Character-Literal)
                        : Expression)+
                    end
```

El contexto en que se deben incluir estás reglas gramaticales en Triangle es el siguiente:

```
single-Command  ::=  Identifier V-name' := Expression
                  |  Identifier ( Actual-Parameter-Sequence )
                  |  begin Command end
                  |  let Declaration in single-Command
                  |  if Expression
                        then single-Command
                        else single-Command
                  |  while Expression do single-Command
                  |  For
                  |  Repeat
                  |  Do
                  |  case-Command

Expression      ::=  second-Expression
                  |  let Declaration in Expression
                  |  if Expression then Expression else Expression
                  |  case-Expression
```

Note que:

- Todas las estructuras empiezan con una palabra reservada la cual se puede utilizar para guiar el reconocimiento, algunas de ellas ya están parcial o totalmente implementadas en el código de Triangle.
- La sintaxis de estas reglas gramaticales es EBNF, así que los paréntesis utilizados en las estructuras nuevas no son parte de la gramática Triangle, si no de regla gramatical.

Ejemplos

```
for i := 1 to 10 do a := a + 1;
for j := a+2 to a-2 by -1 do
  begin
      x := x + j
  end;

repeat
   i := i + 1
util i > 10;

do
   i := i + 1
while i <= 10;

mes = case hoy.mes of
          1 : 31
          2 : 28
          3 : 31
      end.
```

Componente 2

Debe modificar la declaración y uso de records para que pueda incluírseles métodos (funciones o procedimientos) que tengan acceso a los componentes del record.

```
record-Type          ::=  Identifier
                   |  array Integer-Literal of Type-denoter
                   |  record Record-Type-denoter end
                   |  proc Identififer ( Formal-Param-Seq )
                            ~ single-Command
                   |  func ( Formal-Param-Seq )
                           : Type-denoter ~ Expression
```

Record-Type-denoter ::= Identifier : record-Type (ϵ | , Record-Type-denoter)

Esto le permite entonces poder declarar registros con funciones y procedimientos. No requiere modificar la declaración de constantes de tipo registro. Pero si la declaración de “Primary Expression”

```
primary-Expression   ::=  Integer-Literal
                   |  Character-Literal
                   |  V-name [ ( Actual-Param-Seq ) ]
                   |  Operator primary-Expression
                   |  ( Expression )
                   |  { Record-Aggregate }
                   |  [ Array-Aggregate ]
```

Note que

- En la declaración con V-name los corchetes (paréntesis cuadrados) indican que que Actual-Param-Seq, es opcional, está se refiere a la llamada de un procedimiento o función.
- Por el contrario en Expression, Record-Aggregate y [Array-Aggregate] si son parte de la sintaxis del lenguaje.

esto le permitirá declarar cosas como:

```
record
    x : integer;
    y : integer;
    norm : func () ~ x*x + y*y
end
```

Generalidades

Las indicaciones que se dan en esta tarea deben ser complementadas con su implementación de la semántica de cada extensión, la cual se asume conocida por ustedes para el for, case, repeat y do. La implementación de las funciones en los records puede que requiera de que usted modifique la gramática propuesta, tiene libertad para hacer esto.

Puntos Extra:

Implementar Herencia en los records. El modos y la gramática en que lo hace queda a criterio suyo.