Lenguajes de Programación – Proyecto Semestral

1 Reglas

- El proyecto debe realizarse en grupos de 1-2 personas

- Está prohibido reutilizar, total o parcialmente, código de proyectos similares, con excepción de lo que se muestra en los videos del tutorial de ANTLR

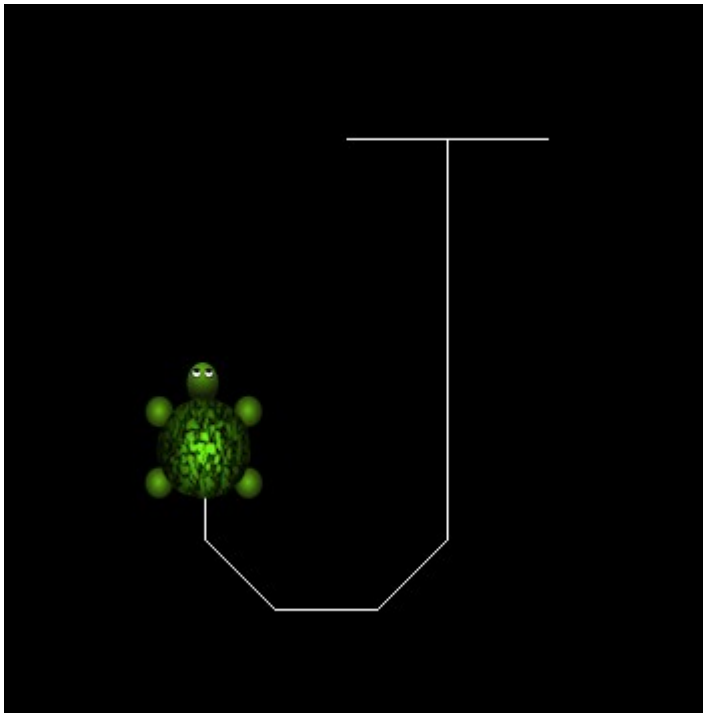
2 Enunciado

El proyecto de este semestre consiste en implementar un intérprete para una variante del lenguaje LOGO. Logo es un lenguaje de programación educativo desarrollado en la década del '60. Su principal característica es la capacidad de desplegar y manejar una tortuga, un agente virtual que es capaz de dibujar en la pantalla.

La tortuga puede moverse por la pantalla, rotar y dibujar con diferentes colores. Por ejemplo, el siguiente programa:

```
set_color 0,0,0,0
mov_forw 50
rot_l 180
set_color 255,255,255,255
mov_forw 100
rot_l 180
set_color 0,0,0,0
mov_forw 50
set_color 255,255,255,255
rot_r 90
mov_forw 200
rot_r 45
mov_forw 50
rot_r 45
mov_forw 50
rot_r 45
mov_forw 50
rot_r 45
mov_forw 50
```

Produce la siguiente salida en pantalla:



Por supuesto, esto es sólo la punta del iceberg. Así como cualquier lenguaje tradicional de programación, LOGO puede procesar expresiones aritméticas, lógicas, sentencias de control (condicionales, ciclos), funciones/subrutinas, etc.

El siguiente ejemplo incluye la mayoría de los elementos del lenguaje:

```
def moveRotate(angle, dist):
      rot_r angle
      move_forw dist
end_def

def makeHorizontalLine():
      set_color 0,0,0,0
      move_forw 50
      rot_l 180
      set_color 255,255,255,255
      move_forw 100
      rot_l 180
      set_color 0,0,0,0
      move_forw 50
end_def

let a = 0
while a < 3 do
      println a
      a = a + 1
end_while

if a == 3 + 2 * (8-1) then
      println "hola"
end_if

makeHorizontalLine()
set_color 255,255,255,255
rot_r 90
move_forw 200
```

```
moveRotate(45,50)
moveRotate(45,50)
moveRotate(45,50)
moveRotate(45,50)
```

3 Elementos del lenguaje

Los elementos del lenguaje son los siguientes:

1. Comandos de la tortuga: `move_forw, move_back, rot_l, rot_r, set_color`
2. Variables: declaración y asignación

   ```
   let miVariable
   miVariable = 5
   let miVariable2 = "hola"
   ```

3. Constantes y tipos de datos: Números decimales, booleanos y strings

   ```
   100.3
   true
   false
   "hola"
   ```

4. Condicionales

   ```
   if condición then  código  end_if
   if condición then  código  else  código  end_if
   ```

5. Ciclos

   ```
   while condición do  código  end_while
   ```

6. Impresión y lectura por pantalla:

   ```
   read miVariable
   println miVariable
   ```

7. Expresiones aritméticas: `+`, `-`, `*`, `/`, inverso aditivo
8. Expresiones lógicas: `&&, ||, !`
9. Expresiones de comparación: `>`, `<`, `<=`, `>=`, `==`, `<>`
10. Declaración de funciones

    ```
    def  miFuncion(param1,  param2,  …  ,  paramN):      código
    end_def
    ```

11. Ejecución de funciones

    ```
    miFuncion(10+3*5, "hola", … , true)
    ```

Para desarrollar el intérprete, Ud. dispondrá de las siguientes herramientas: Una distribución de Eclipse con soporte para ANTLR4 y un proyecto en Eclipse con el esqueleto del proyecto.

Entregas

Primera Entrega:

•Intérprete básico del lenguajes. Deben soportarse solamente los comandos indicados en la sección 3.1

Segunda Entrega:

•Intérprete completo con todos los elementos del lenguaje.