Romel J. Martínez Olvera A01207360

Analizador léxico

Para este semestre decidí realizar un compilador para el lenguaje Scala con una herramienta llamada ANTLR. Scala es un lenguaje hibrido orientado a objetos y funcional.

Scala

Normalmente un programa muy básico en scala luce de esta manera

```
object HelloWorld {
  def main(args: Array[String]){
    println("Hello, world!")
  }
}
```

Se define un objeto, dentro del objeto algunos métodos(def) y dentro de los métodos variables, ciclos, condicionales etc.

Me enfoque en los siguientes elementos para mi gramática.

Objetos

```
object HelloWorld {
}
```

Métodos

```
def main(args: Array[String]){
}
```

Variables

Val es una variable constante y var es una variable que se puede modificar

```
var myVar :Int = 10;
val myVal :String = "Hello Scala with datatype declaration.";
var myVar1 = 20;
val myVal1 = "Hello Scala new without datatype declaration.";
```

Comentarios

```
/*Scala program to print your name*/
//Only a line
```

Condicionales

```
if(a>b){
//something
```

```
}
else if(a<b){
//something
}
else{
//something
}
```

Ciclos

```
while(i<b){
//something
}
```

Instalar ANTLR

Utilice Windows 8 para instalar ANTLR y para trabajar en el proyecto, aquí dejo las instrucciones de como instalar ANTLR en Windows y Linux.

Página web: https://www.antlr.org/index.html

Descargar el archivo

`http://www.antlr.org/download/antlr-4.7.2-complete.jar`

`Linux`

- // 1.
- sudo cp antlr-4.7.2-complete.jar /usr/local/lib/
- // 2. and 3.
- // add this to your .bash_profile
- export CLASSPATH = ".:/usr/local/lib/antlr-4.7.2-complete.jar:$CLASSPATH"
- // simplify the use of the tool to generate lexer and parser
- alias antlr4='java -Xmx500M -cp "/usr/local/lib/antlr-4.7.2-complete.jar:$CLASSPATH" org.antlr.v4.Tool'
- // simplify the use of the tool to test the generated code
- alias grun='java org.antlr.v4.gui.TestRig'

Windows

- // 1. **Copy** antlr-4.7.2-complete.jar **in** C:\Program Files\Java\libs (or wherever you prefer)
- // 2. Create or **append** to the CLASSPATH variable the location of antlr
- // you can **do** to that by pressing WIN + R and typing sysdm.cpl, then selecting Advanced (tab) > Environment variables > System Variables
- // CLASSPATH -> .;C:\Program Files\Java\libs\antlr-4.7.1-complete.jar;%CLASSPATH%
- // 3. Add aliases
- // create antlr4.bat
- java org.antlr.v4.Tool %*
- // create grun.bat
- java org.antlr.v4.gui.TestRig %*
- // put them **in** the system **PATH** or any of the directories included **in** your **PATH**

Gramática

La gramática se encuentra en un archivo dentro de la carpeta Scala_ llamado **Scala.g4**.

Opté por un enfoque top-down al construir mi gramática ya que al analizar el lenguaje scala me di cuenta de que se parecía mucho a java y al conocer java se me hizo más fácil empezar con la organización general del lenguaje.

No me enfoque en la construcción de los objetos ya que eso esta afuera del alcance del proyecto.

Los tokens están en letras mayúsculas y se encuentran hasta el final del documento Scala.g4. Mis tokens son palabras reservadas, operadores, constantes numéricas e identificadores del lenguaje scala.

Casos de prueba

Desarrolle 18 archivos con extensión .txt con los casos anteriores y los nuevos casos

Correr el programa

java Main

En caso de que no funcione utilice los siguientes comandos

antlr4 -no-listener -visitor Scala.g4

javac Main.java Scala.java*

grun Scala prog caso_prueba/caso_1.txt (No es necesario este comando es solo para probar la una gramática)

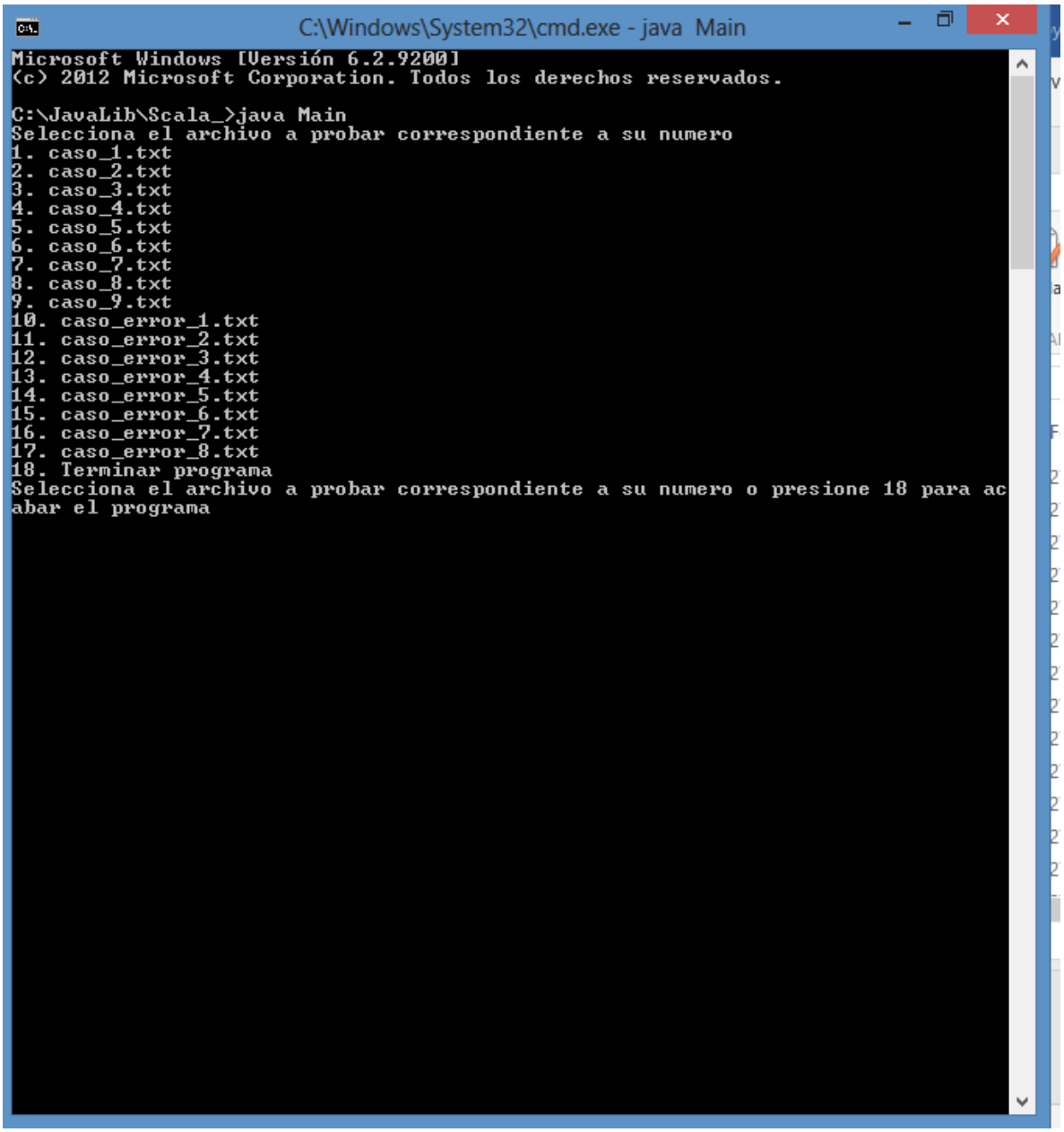
java Main

Analizador Semántico

Para esta entrega construí el árbol sintáctico, realicé comprobaciones semánticas y construí la tabla de símbolos. En las siguientes imágenes se puede visualizar los árboles sintácticos que se requerían para cada error.

Menú de casos

El menú es para evaluar los diferentes casos. Los primeros 9 son los de la evaluación pasada y solo regresa un mensaje de éxito al seleccionarlos. Los otros casos regresan errores correspondientes. Solo hay seleccionar un numero entre el 1 al 17 para probar diferentes casos. El numero 18 es para salir del programa.



Casos de prueba con error

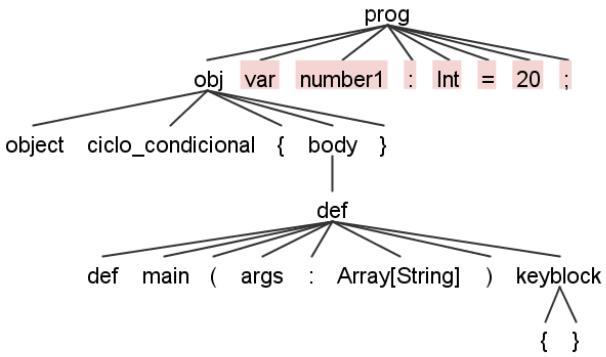
caso_error_1.txt

El caso_error_1.txt marca error porque la variable no fue escrita correctamente

Incorrecto:

```
object ciclo_condicional {
        def main(args: Array[String]) {
 }
}var integral: Int = 10
```

Correcto: Al menos que se encuentre en el main o en algún otro método una variable



caso_error_2.txt

El caso_error_2.txt marca error porque la variable esta al final del documento y en scala tienen que estar en algún objeto o dentro de una definición.

(La imagen también se encuentra en la carpeta para una mejor visualización)

caso_error_3.txt

El caso_error_3.txt marca error porque el ciclo while en lugar de usar paréntesis use celdas eso no está permitido en scala.

Incorrecto

while[]{

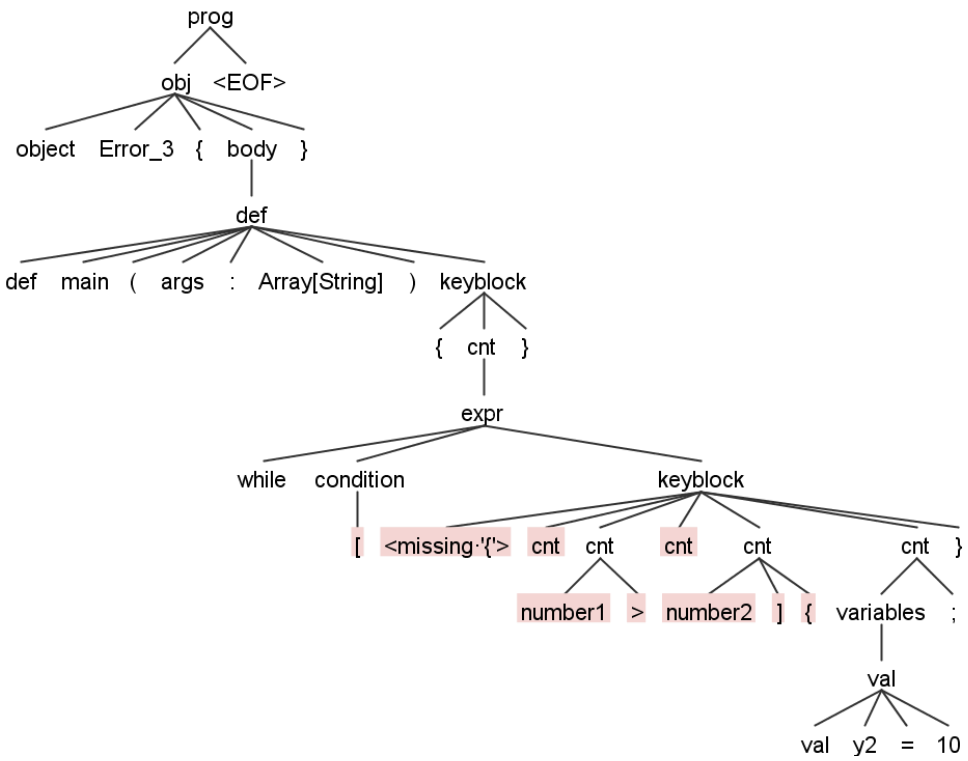
}

while[]

Correcto

while(){

}



caso_error_4.txt y caso_error_5.txt

Es el error de darle a una variable previamente declarada otro tipo de dato que no es el indicado.

caso_error_6.txt

Es el error de darle a una variable que esta fuera del alcance

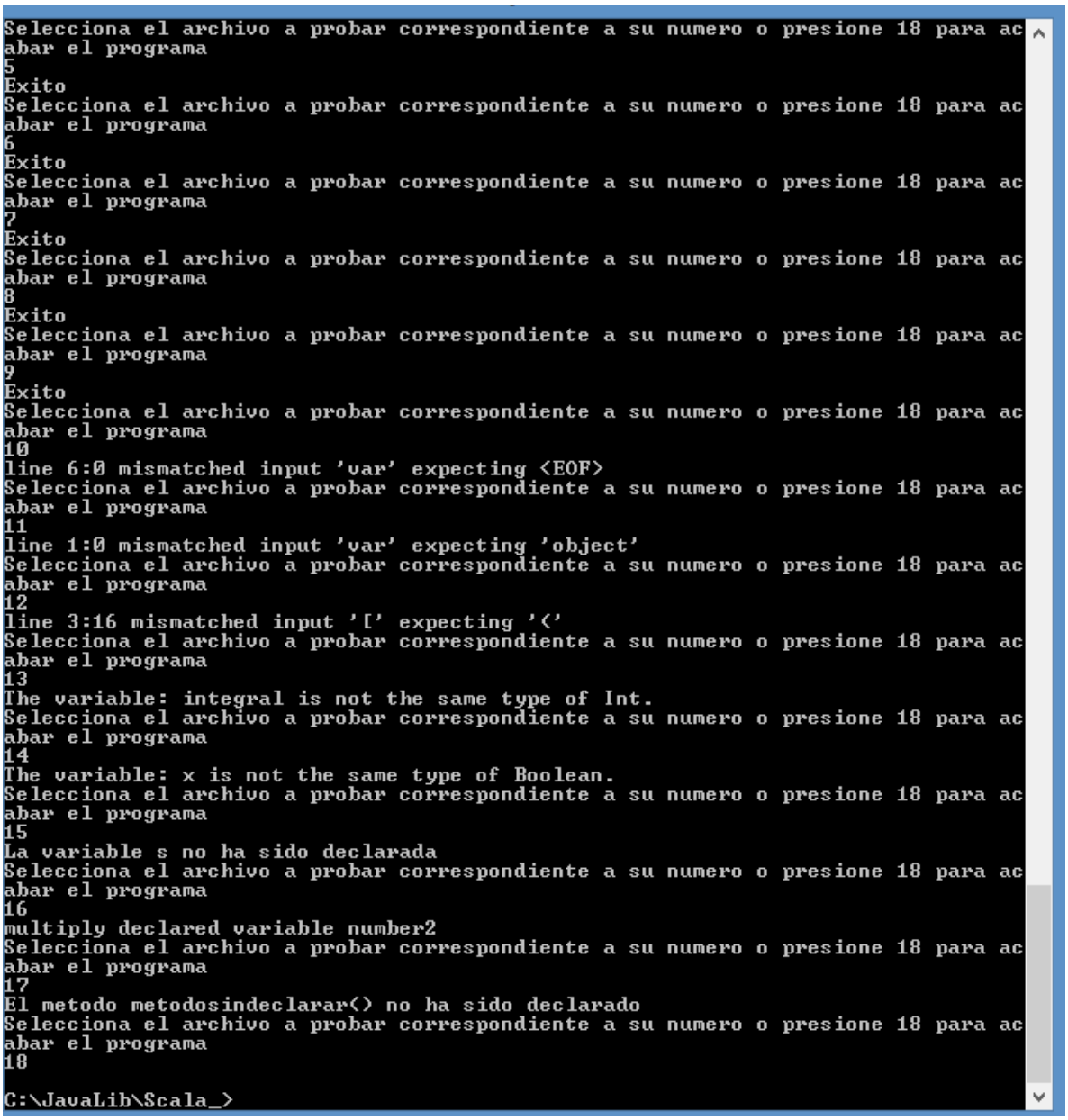
caso_error_7.txt

Error de definir una variable previamente definida.

caso_error_8.txt

Error de un método que no ha sido definido.

Evidencia


Selecciona el archivo a probar correspondiente a su numero o presione 18 para ac
abar el programa
5
Exito
Selecciona el archivo a probar correspondiente a su numero o presione 18 para ac
abar el programa
6
Exito
Selecciona el archivo a probar correspondiente a su numero o presione 18 para ac
abar el programa
7
Exito
Selecciona el archivo a probar correspondiente a su numero o presione 18 para ac
abar el programa
8
Exito
Selecciona el archivo a probar correspondiente a su numero o presione 18 para ac
abar el programa
9
Exito
Selecciona el archivo a probar correspondiente a su numero o presione 18 para ac
abar el programa
10
line 6:0 mismatched input 'var' expecting <EOF>
Selecciona el archivo a probar correspondiente a su numero o presione 18 para ac
abar el programa
11
line 1:0 mismatched input 'var' expecting 'object'
Selecciona el archivo a probar correspondiente a su numero o presione 18 para ac
abar el programa
12
line 3:16 mismatched input '[' expecting '('
Selecciona el archivo a probar correspondiente a su numero o presione 18 para ac
abar el programa
13
The variable: integral is not the same type of Int.
Selecciona el archivo a probar correspondiente a su numero o presione 18 para ac
abar el programa
14
The variable: x is not the same type of Boolean.
Selecciona el archivo a probar correspondiente a su numero o presione 18 para ac
abar el programa
15
La variable s no ha sido declarada
Selecciona el archivo a probar correspondiente a su numero o presione 18 para ac
abar el programa
16
multiply declared variable number2
Selecciona el archivo a probar correspondiente a su numero o presione 18 para ac
abar el programa
17
El metodo metodosindeclarar() no ha sido declarado
Selecciona el archivo a probar correspondiente a su numero o presione 18 para ac
abar el programa
18
C:\JavaLib\Scala_>