Practica 1- Análisis léxico con Flex

Sarah Sophia Olivares García

Preguntas

1. ¿Qué ocurre si en la primera sección se quitan las llaves al nombre de la macro `letra`? (0.5 pts)

   Compiló el programa aún después de quitarle las llaves, pero `flex` interpretó `letra` como una secuencia literal en lugar de una macro. En consecuencia, cambió el comportamiento ya que el texto de entrada lo interpreta de manera distinta, es decir, trata `letra` como una cadena literal en lugar de expandir la macro. Por lo tanto, el análisis no reconoció las palabras correctamente y generó un comportamiento inesperado, en este caso solo encontrando el numero`"100"`.

```
Encontré una palabra: hoy
sophia@Sophie:/mnt/c/Users/sophi/src/Primer programa en Lex$ jflex archivo.flex
Reading "archivo.flex"

Warning : Macro "letra" has been declared but never used.
Constructing NFA : 22 states in NFA
Converting NFA to DFA :
.........
11 states before minimization, 9 states in minimized DFA
Old file "Lexer.java" saved as "Lexer.java~"
Writing code to "Lexer.java"
sophia@Sophie:/mnt/c/Users/sophi/src/Primer programa en Lex$ javac Lexer.java
sophia@Sophie:/mnt/c/Users/sophi/src/Primer programa en Lex$ echo "Hay 100 nubes en el cielo hoy" > input.txt
sophia@Sophie:/mnt/c/Users/sophi/src/Primer programa en Lex$ java Lexer input.txt
HayEncontré un número: 100
nubesenelcielohoysophia@Sophie:/mnt/c/Users/sophi/src/Primer programa en Lex$
```

2. ¿Qué ocurre si en la segunda sección se quitan las llaves a las macros? (0.5 pts)

   Al quitar también las llaves en la segunda sección, `Flex` no interpreta correctamente las macros y las trata como cadenas. Por lo tanto:

   - `Flex` genera advertencias indicando que las macros (`espacio`, `palabra`, `letra`, `digito`) han sido declaradas pero no se están utilizando.
   - Las reglas léxicas que dependen de estas macros no coinciden con ningún texto de entrada. Por lo tanto, el analizador léxico no realiza ninguna acción sobre el texto de entrada y simplemente lo devuelve tal como es.

```
sophia@Sophie:/mnt/c/Users/sophi/src/Primer programa en Lex$ jflex archivo.flex
Reading "archivo.flex"

Warning : Macro "espacio" has been declared but never used.

Warning : Macro "palabra" has been declared but never used.

Warning : Macro "letra" has been declared but never used.

Warning : Macro "digito" has been declared but never used.
Constructing NFA : 46 states in NFA
Converting NFA to DFA :
.....................
23 states before minimization, 22 states in minimized DFA
Old file "Lexer.java" saved as "Lexer.java~"
Writing code to "Lexer.java"
sophia@Sophie:/mnt/c/Users/sophi/src/Primer programa en Lex$ javac Lexer.java
sophia@Sophie:/mnt/c/Users/sophi/src/Primer programa en Lex$ echo "Hay 100 nubes en el cielo hoy" > input.txt
sophia@Sophie:/mnt/c/Users/sophi/src/Primer programa en Lex$ java Lexer input.txt
Hay 100 nubes en el cielo hoy
sophia@Sophie:/mnt/c/Users/sophi/src/Primer programa en Lex$
```

3. ¿Cómo se escribe un comentario en `flex`? (0.5 pts)

```
17  // Este es un comentario de una sola linea
18  /*
19   * Este es un comentario de varias lineas
20   * jeje
21   *  holi
22   */
```

4. ¿Qué se guarda en `yytext`? (0.5 pts)

   En `yytext` se guarda el texto que coincide con la expresión regular definida en la regla léxica que se está ejecutando.

   s decir:

   ```
   {digito}+{System.out.print("Encontre un numero: "+yytext()+"\n");}
   {palabra}{System.out.print("Encontre una palabra:"+yytext() + "\n");}
   ```

   Si la entrada contiene una secuencia de dígitos (como 123), y la regla `{digito}+` coincide, `yytext` contendrá la cadena "123".

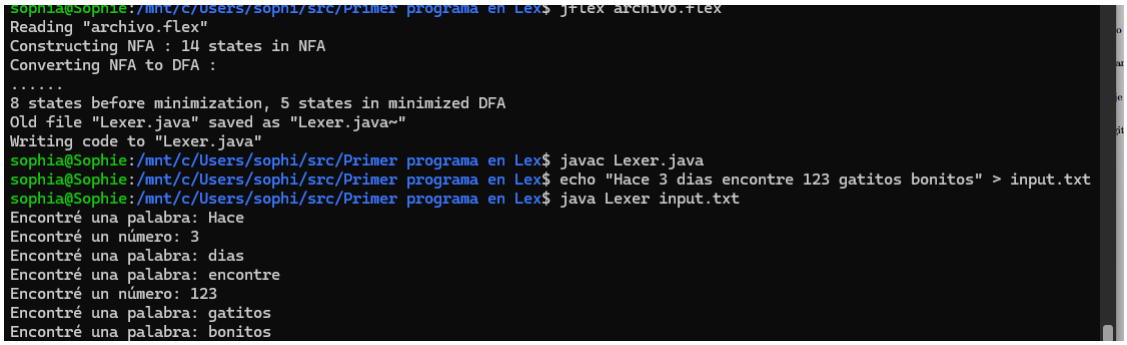
   Si la entrada contiene una palabra, y la regla `{palabra}` coincide, `yytext` contendrá la cadena "Hola".

5. ¿Qué pasa al ejecutar el programa e introducir cadenas de caracteres y de dígitos en el archivo de entrada? (0.5 pts)

   La salida muestra cómo el escáner ha procesado cada token del archivo de entrada y ha clasificado correctamente cada uno como una palabra o un número.

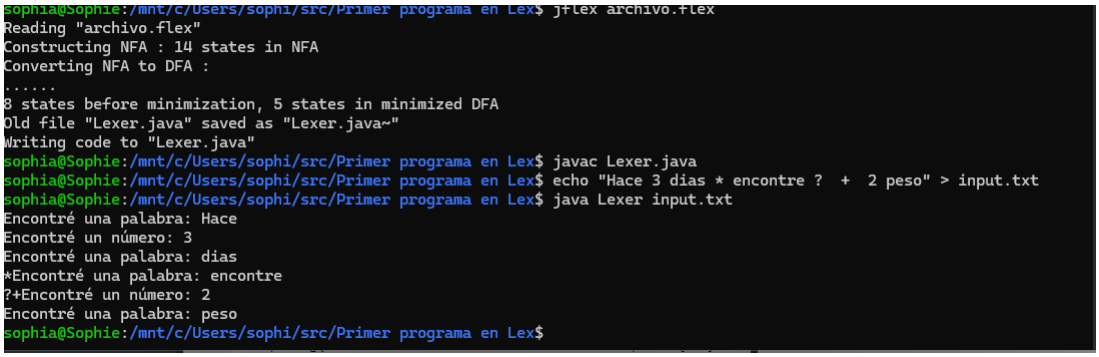
   - **Palabras:** El escáner detecta palabras (cadenas de caracteres) basándose en la expresión regular que define las letras (`letra`) y las combinaciones de letras (`palabra`).

- **Números:** El escáner detecta secuencias de dígitos utilizando la expresión regular que define los dígitos (`digito`).
- **Procesamiento de Espacios:** Los espacios (`espacio`) se ignoran, lo que significa que no aparecen en la salida. Esto se debe a la acción léxica vacía para {`espacio`}.



6. ¿Qué ocurre si introducimos caracteres como `"*"` en el archivo de entrada? (0.5 pts)

   LLos caracteres especiales (`*`, `?`, `+`) no están definidos en las reglas. Por lo tanto, el escáner trata estos caracteres como parte de una secuencia de caracteres que no coinciden con las definiciones de `palabra` o `digito`. Como resultado, la salida muestra que estos caracteres sí se imprimen, pero no se están reconociendo como tokens separados.



7. Modificar al código anterior en un archivo nuevo, de tal manera que reconozca lo siguiente: (2 pts)

   (a) La expresión regular para los hexadecimales en lenguaje `Java`.

   (b) 5 palabras reservadas del lenguaje `Java`.

   (c) Los identificadores válidos del lenguaje `Java`, con longitud máxima de 32 caracteres (Sugerencia: use el operador {`m,n`}).

   (d) Los espacios en blanco.

```
sophia@Sophie:/mnt/c/Users/sophi/src/Primer programa en Lex$ javac Lexer.java
sophia@Sophie:/mnt/c/Users/sophi/src/Primer programa en Lex$ echo "Hay 100 nubes en el cielo hoy public" > input.txt
sophia@Sophie:/mnt/c/Users/sophi/src/Primer programa en Lex$ java Lexer input.txt
Encontré una palabra: Hay
Encontré un número: 100
Encontré una palabra: nubes
Encontré una palabra: en
Encontré una palabra: el
Encontré una palabra: cielo
Encontré una palabra: hoy
Encontré una palabra reservada: public
```