



UNIVERSIDAD
NACIONAL AUTÓNOMA DE MÉXICO
FACULTAD DE CIENCIAS

Preguntas práctica 1

Integrantes:

Yonathan Berith Jaramillo Ramírez. 419004640

Profesor: Adrián Ulises Mercado Martínez

Ayudantes: Yessica Janeth Pablo Martínez

Carlos Gerardo Acosta Hernández

30 Ago, 2024

Compiladores

Ejercicios

1. ¿Qué ocurre si en la primera sección se quitan las llaves al nombre de la macro **letra**? Tras remover las llaves me percaté de que JFlex ya no logró reconocer las palabras que había en el texto y pues las imprime pero no las identifica como tal.

```
[~/unam/currentSemester/compilers/YonathanJaramilloRamirez/Prácticas/P1/src/primer_programa_en_lexer]
[23:37:13 on develop ✘]→ java Lexer input.txt
HayEncontré un número: 100
nubesenelcielohoy%
```

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

Creo que las macros no logran ser reconocidas por JFlex porque sin llaves las macros no están siendo tomadas como referencias a sus respectivas definiciones, y sin éstas macros pues no está sucediendo nada...

```
[~/unam/currentSemester/compilers/YonathanJaramilloRamirez/Prácticas/P1/src/primer_programa_en_lexer]
[23:50:23 on develop ✘]→ java Lexer input.txt
Hay 100 nubes en el cielo hoy
```

3. ¿Cómo se escribe un comentario en Flex?

En Flex, se pueden escribir comentarios de dos formas principales, similares a los estilos utilizados en los lenguajes C y C++. Los comentarios de una sola línea se inician con dos barras inclinadas (//) y continúan hasta el final de la línea. Por ejemplo:

```
// Este es un comentario de una sola línea
```

Por otro lado, los comentarios multilínea comienzan con /* y terminan con */, y pueden extenderse a través de múltiples líneas. Un ejemplo sería:

```
/*
 * Este es un comentario multilínea.
 * Puede abarcar varias líneas.
 */
```

4. ¿Qué se guarda en yytext?

La función yytext() en Flex y JFlex retorna el texto del segmento de entrada que ha sido emparejado por la expresión regular más reciente en las reglas del analizador léxico. Este texto es referido comúnmente como el “lexeme” o token reconocido.

Por ejemplo, si el analizador léxico reconoce un número en la entrada, yytext() contendrá ese número como una cadena de caracteres. Esto es crucial para las acciones subsecuentes que se pueden querer ejecutar con el token reconocido, como convertirlo a un tipo numérico, almacenarlo, o realizar operaciones lógicas dependiendo del contexto del análisis.

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

Pareciera ser que son reconocidos como si fuesen palabras siento que se está enfocando en conjuntos de único tipo de carácter sin importar si son dígitos o letras

```
[~/unam/currentSemester/compilers/YonathanJaramilloRamirez/Practicas/P1/src/primer_programa_en_lex]
(00:08:50 on develop ✘ *)→ 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: ser
ñEncontré una palabra: aeirh
Encontré un número: 2374
```

6. ¿Qué ocurre si introducimos caracteres como “*” en el archivo de entrada?

```
[~/unam/currentSemester/compilers/YonathanJaramilloRamirez/Practicas/P1/src/primer_programa_en_lex]
(00:16:20 on develop ✘ *)→ 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: ser
ñEncontré una palabra: aeirh
Encontré un número: 2374
*%
```

Como no hay ninguna regla definida para ese tipo de carácter simplemente pues no lo procesa y parece ser que cuando no logra procesar algo simplemente lo imprime al final como se ve en la captura de pantalla

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

- La expresión regular para los hexadecimales en lenguaje Java.
- 5 palabras reservadas del lenguaje Java.
- Los identificadores válidos del lenguaje Java, con longitud máxima de 32 caracteres (**Sugerencia:** use el operador $\{m,n\}$).
- Los espacios en blanco.

El archivo lo podrás encontrar en la carpeta segundo_programa_en_lex

```
(~/unam/currentSemester/compilers/VonathanJaramilloRamirez/Practicas/P1/src/segundo_programa_en_lex) ————— (shaman@shaman: ~) [00:45:30 on develop ✘ * ★] → java Lexer input.txt  
Hexadecimal encontrado: 0xFF  
Espacio en blanco detectado  
Hexadecimal encontrado: 0x1A3  
Espacio en blanco detectado  
Hexadecimal encontrado: 0x00  
Espacio en blanco detectado  
Identificador válido encontrado: int  
Espacio en blanco detectado  
Identificador válido encontrado: double  
Espacio en blanco detectado  
Identificador válido encontrado: float  
Espacio en blanco detectado  
Identificador válido encontrado: boolean  
Espacio en blanco detectado  
Identificador válido encontrado: char  
Espacio en blanco detectado  
Identificador válido encontrado: variable1  
Espacio en blanco detectado  
Identificador válido encontrado: _variable2  
Espacio en blanco detectado  
Identificador válido encontrado: count1234  
Espacio en blanco detectado  
Identificador válido encontrado: _1a2b3c  
Espacio en blanco detectado  
Número encontrado: 12345  
Espacio en blanco detectado  
Número encontrado: 67890  
Espacio en blanco detectado  
Identificador válido encontrado: hello  
Espacio en blanco detectado  
Identificador válido encontrado: world  
Espacio en blanco detectado  
Identificador válido encontrado: example  
Espacio en blanco detectado  
Identificador válido encontrado: test  
Espacio en blanco detectado  
*,Espacio en blanco detectado  
;Espacio en blanco detectado  
-Espacio en blanco detectado  
!Espacio en blanco detectado  
?Espacio en blanco detectado  
:Espacio en blanco detectado
```

