

Tema 2. Ejercicios.

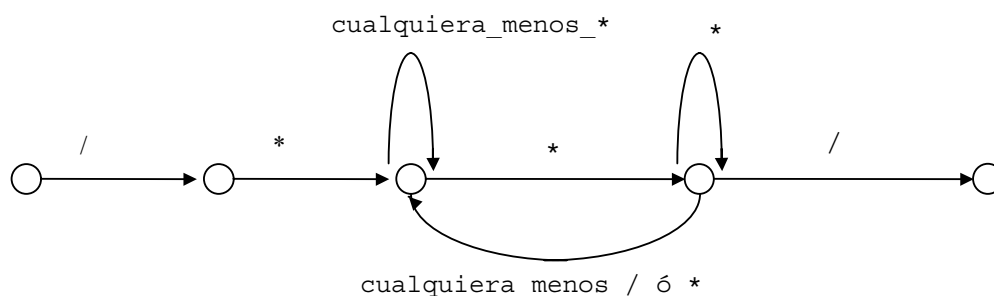
- 2.1** La siguiente expresión regular LEX especifica un tipo de token. A partir de esa expresión desarrolla un autómata finito determinista que especifique lo mismo.

`\{ [^] + \}`

- 2.2** La siguiente expresión regular LEX especifica un tipo de token. A partir de esa expresión desarrolla un autómata finito determinista que especifique lo mismo.

`\(* ([^ * + [^] *) * * + \)`

- 2.3** Dado el siguiente autómata, indica qué tipo de cadenas define y además escribe la definición LEX correspondiente.



- 2.4** Se quiere escribir una expresión regular LEX para describir los identificadores de un nuevo lenguaje. La definición es igual a la ya conocida con la salvedad de que se permite utilizar uno o dos *underscores* para separar caracteres (por ejemplo, Num Num_Entero y Num__Entero son identificadores correctos, mientras que Num_ o Num___Entero no). Una alumna de compilación nos ha dado la siguiente solución:

`[a-zA-Z] ([a-zA-Z0-9] | ("_" ? [a-zA-Z0-9])) *`

¿Es correcta? ¿Por qué? En caso negativo da tu solución.

- 2.5** En un nuevo lenguaje de programación se ha definido un nuevo tipo de comentario: comienza y termina con la llave correspondiente, y no puede incluir una llave de cierre en su interior. En el interior puede aparecer cualquier carácter excepto que no se admite una secuencia de dos ó más asteriscos. Define una expresión LEX que los describa.

Ejemplos:

Válido

`{a bcd fg }`
`{a bcd* fg }`
`{a bcd* fg * }`

Inválido

`{a bcd} fg }`
`{a bcd** fg }`
`{a bcd ***** fg }`