Sesión 04**: Equivalencias de grafos y expresiones regulares**

I. OBJETIVOS

- Convertir AF en expresiones regulares
- Convertir expresiones regulares en AF
- Utilizar las funcionalidades básicas de flex para programar un analizador léxico a partir de un grafo.

II. TEMAS A TRATAR

- Flex
- Equivalencias de AF con expresiones regulares
- Análisis léxico de grafos y expresiones regulares

III. MARCO TEÓRICO

Un AFD es una máquina que consta de un conjunto finito de estados, también toma el nombre de Máquina de estados. Tiene un estado inicial o de entrada de la máquina y uno o varios estados de aceptación.

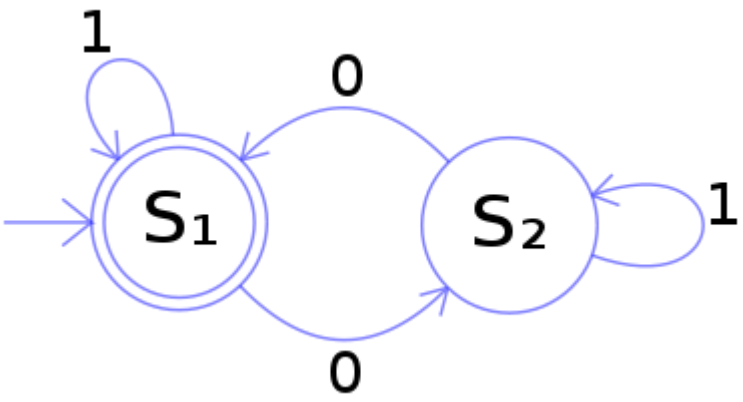
Un AFND es una máquina que tiene transiciones vacías o que por cada símbolo desde un estado de origen, se llega a un estado distinto. Uno de estos estados debe ser la entrada de la máquina, y de igual modo, debe existir uno o más estados de aceptación.

Una Expresión regular, es una forma de representar una máquina de estados. Se considera más descriptica que las máquinas de estados.

Los autómatas y las expresiones regulares con congruentes entre sí, ya que de ambas maneras se pueden representar una infinidad de máquinas de estados.

Operaciones de las expresiones regulares

- **Unión** Se representa con el signo más ‘+’ , es decir que la unión de los conjuntos (A U B) se representa como A+B en una expresión regular.
- **La concatenación** Se representa con un punto ‘.’. Normalmente el punto no se escribe y se coma como concatenación de A y B cuando se escribe “AB”, siendo A y B dos conjuntos.
- **La clausura o cláusula de Kleene** Se representa como un asterisco ‘*’ que se escribe seguido del conjunto al que se le aplica. Por ejemplo A*. Lo que la clausura nos indica es que habrán 0 o más repeticiones de la cadena A

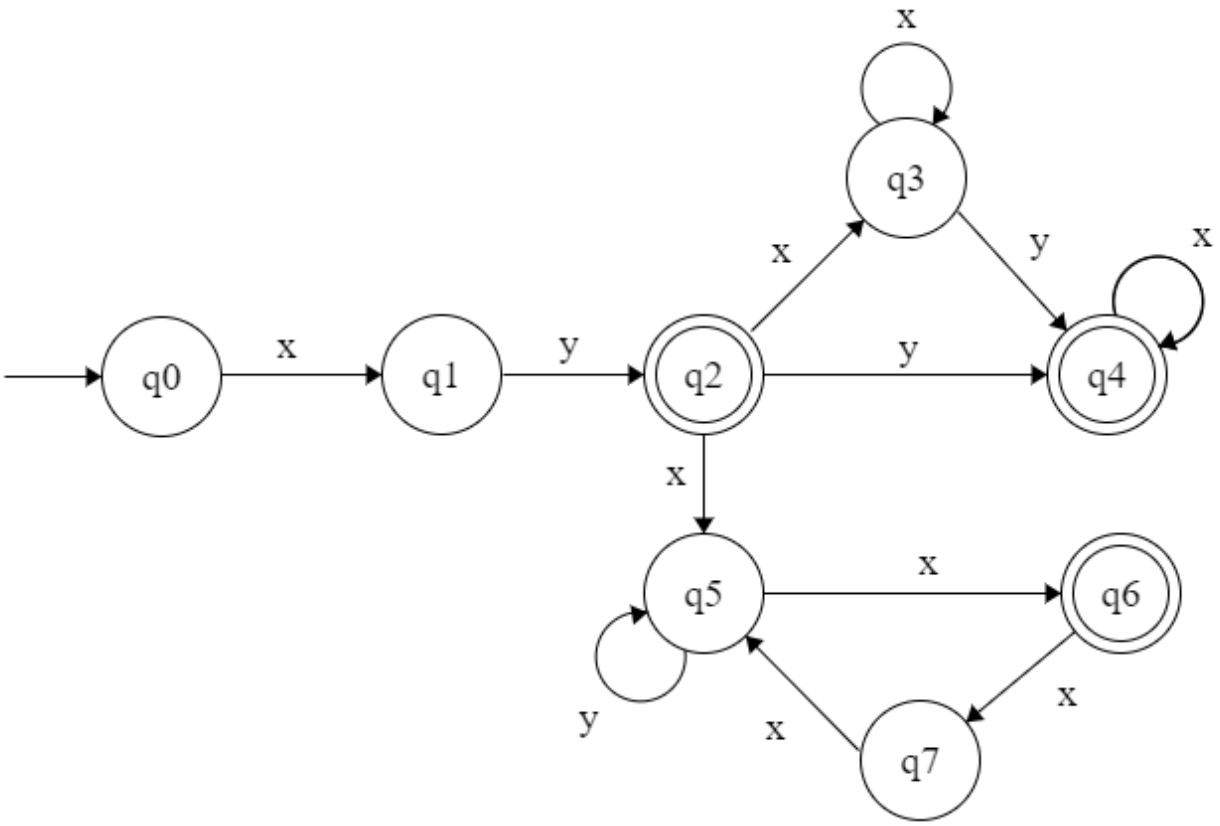


Consideraciones

- La expresión regular está formada por varios sumandos
- El número de sumandos dependerá de:
  - El número de estados de aceptación
  - El número de caminos hasta los mismos
- Los nodos que aparecen “en serie” en el AF, van concatenados en la ER.
- Los caminos paralelos de un nodo a otro nodo, aparecen en la ER, sumados y entre paréntesis (es un “o” lógico).
- Un * indica un bucle sobre un nodo (es una cerradura de Kleene)

Ejemplo de conversión:

Dado el AF:



Primero observamos que tenemos tres estados de aceptación, luego verificaremos todos los caminos disponibles para cada estado:

Tenemos:

Para llegar a q2:

Partida y Final	Estados que atraviesa	Expresión regular
De q0 a q2	Q0, q1, q2	xy

Para llegar a q4:

Partida y Final	Estados que atraviesa	Expresión regular
De q0 a q4	Q0, q1, q2, q3, q4	Xyxx*yx*
De q0 a q4	Q0, q1, q2, q4	Xyyx*

Para llegar a q6:

Partida y Final	Estados que atraviesa	Expresión regular
De q0 a q4	Q0, q1, q2, q5, q6, (q7, q5, q6)*	xyxy*x(xxy*x)

La expresión regular finalmente, es la suma de todos los caminos posibles:

Xy+ Xyxx*yx*+ Xyyx*+ xyxy*x(xxy*x)*

Podemos factorizar:

Xy+ xy(xx*yx* + yx* + xy*x(xxy*x)*)

En flex cada símbolo + representará un o lógico (|) y reconocerá una expresión regular distinta.
La construcción de un analizador sintáctico para estas expresiones regulares será entonces:

El programa que reconocerá el AF en flex es por tanto:

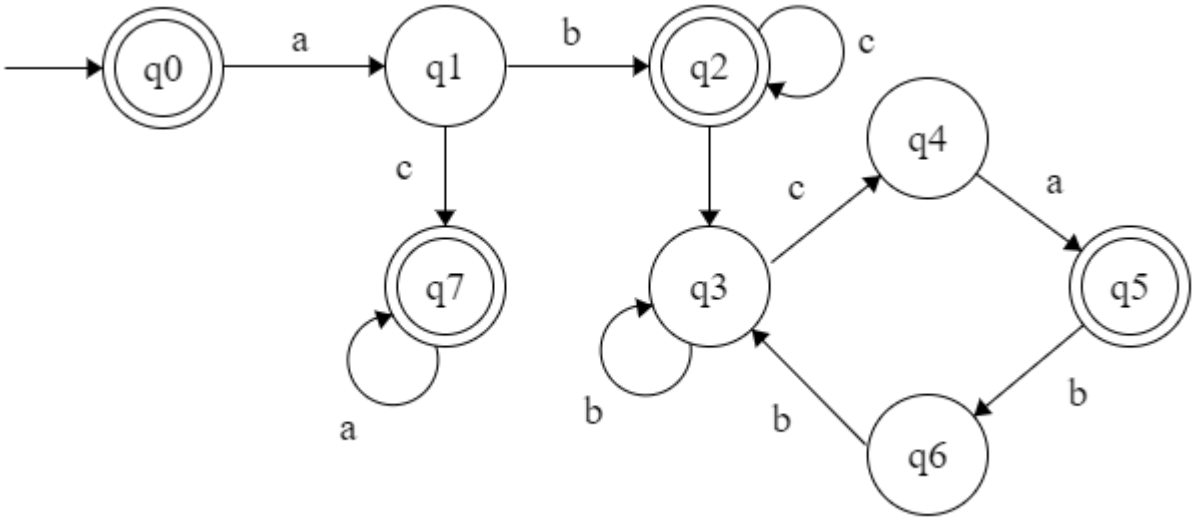
```
/* seccion de definiciones */
%{

%}

/* seccion de reglas*/
%%
xy              {printf("Primera expresion regular \n");}
xyxx*yx*        {printf("Segunda expresion regular \n");}
xyyx*           {printf("Tercera expresion regular \n");}
xyxy*x(xxy*x)*  {printf("Cuarta expresion regular \n");}
 .              {printf("No se reconoce");}

%%
/* seccion de codigo de usuario*/
int yywrap(){}
int main(){
    printf("Equivalencia de AF a ER \n");
    printf("Ingrese la cadena a evaluar \n");
    yylex();
    return 0;
}
```

IV. ACTIVIDADES

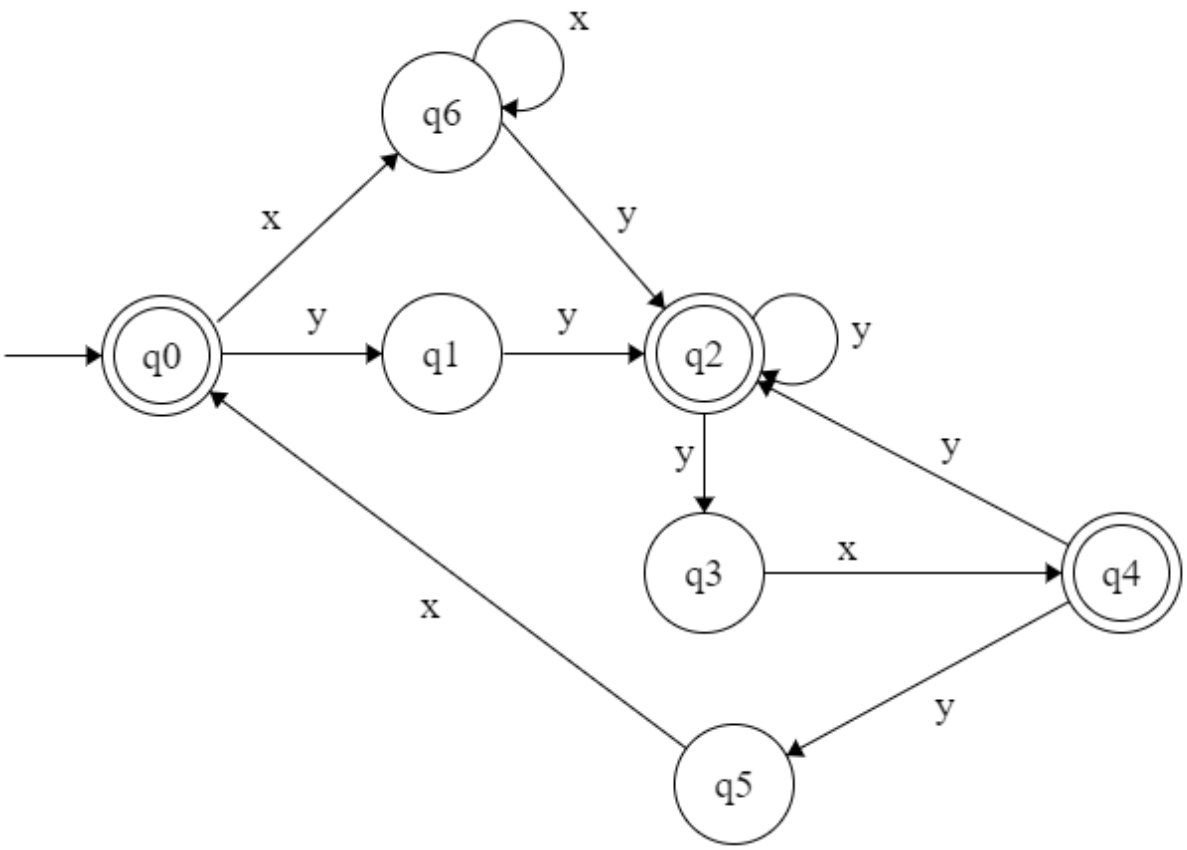
Realiza las siguientes actividades:

1. Escribe los programas en Flex que reconozcan los siguientes AF:

a)


q0
a
q1
b
q2
c
c
q7
a
q3
c
q4
a
q5
b
b
b
q6

b)


x
q6
x
y
y
q0
y
q1
y
q2
y
y
y
q3
x
q4
x
y
q5

2. Diseñe un AF que reconozca múltiplos de 3 y programe su analizador sintáctico en Flex

VII. Ejercicios

1. Cuáles son los pasos para ir de una expresión regular a un programa un AF
2. Diseñe un AF y su programa en Flex que reconozcan números pares en binario.
3. Diseñe un AF y su programa en Flex que reconozca las horas de un reloj digital.
4. Diseñe un AF y su programa en Flex que reconozca la declaración de variables en lenguaje c (tipo int, float, y char)

VII. Bibliografía y referencias

1. https://es.qwe.wiki/wiki/Flex_(lexical_analyser_generator)
2. https://docplayer.es/46330679-Lex-flex-generacion-de-analizador-lexico-p-1.html
3. http://web.archive.org/web/20130627190411/
4. http://wwwdi.ujaen.es:80/~nacho/Flex.htm