



# **INSTITUTO POLITÉCNICO NACIONAL**

## **ESCUELA SUPERIOR DE CÓMPUTO**



### **Práctica 1:**

Diseño e implementación de las clases AFN, AFD

### **Alumno:**

Meza Vargas Brandon David

### **Grupo:**

3CM13

### **Profesor:**

Sánchez Juárez José

# Índice

<b>Objetivo.....</b>	<b>3</b>
<b>Actividades .....</b>	<b>3</b>
<b>Cuestionario .....</b>	<b>5</b>
<b>Conclusiones.....</b>	<b>15</b>

## Objetivo

Se crearán las clases, para el AFN y para el AFD

## Actividades

Se tiene la expresión regular  $c^* b (c \mid bc^* b)^*$ . Usar el siguiente código para programar los tokens que acepta el AFD.

Programa AFD. Para ejecutar este programa se hace con la línea de comando: java DriverAFD cadena de entrada

```
import java.util.*;

public class DriverAFD {
    public static void main(String[] args) {
        ArgumentosToken tm = new ArgumentosToken(args);
        AFD m = new AFD(tm);
        m.activarAFD();
    }
}

class ArgumentosToken {
    private int indice;
    String entrada;
    public ArgumentosToken(String[] args) {
        if(args.length > 0)
            entrada = args[0];
        else
            entrada = "";
        indice = 0;
        System.out.println("entrada = " + entrada);
    }
    public char getSiguienteToken() {
        if(indice < entrada.length())
            return entrada.charAt(indice++);
        else
            return '$';
    }
}

class AFD {
    ArgumentosToken tm;
    private char tokenActual;
    public AFD(ArgumentosToken tm) {
```

```

        this.tm = tm;
    }
    public void avanzar() {
        tokenActual = tm.getSiguienteToken();
    }
    public void activarAFD() {
        int estadoActual = 0;
        avanzar();

        while(tokenActual != '$') {
            switch(estadoActual) {
                case 0:
                    if(tokenActual == 'b') estadoActual = 1;
                    else if(tokenActual == 'c') estadoActual = 1;
                    break;
                case 1:
                    if(tokenActual == 'b') estadoActual = 2;
                    else if(tokenActual == 'c') estadoActual = 3;
                    break;
                case 2:
                    if(tokenActual == 'b') estadoActual = 4;
                    else if(tokenActual == 'c') estadoActual = 4;
                    break;
                case 3:
                    if(tokenActual == 'b') estadoActual = 4;
                    else if(tokenActual == 'c') estadoActual = 3;
                    break;
                case 4:
                    if(tokenActual == 'b') estadoActual = 4;
                    else if(tokenActual == 'c') estadoActual = 4;
                    break;
            }
            avanzar();
        }
        if(estadoActual == 2 || estadoActual == 3)
            System.out.println("Acep");
        else
            System.out.println("rechazado");
    }
}

```

## Cuestionario

1. Realizar todo el proceso de transformación de la expresión regular hasta la tabla de transiciones del AFD. De tal manera que se compruebe el programa DriverAFD

$$c^*b(c|bc^*b)^*$$

Lo primero que se hará será obtener el AFN de la ER para así obtener el AFN, reducirlo y obtener su tabla de transiciones.

### AFN

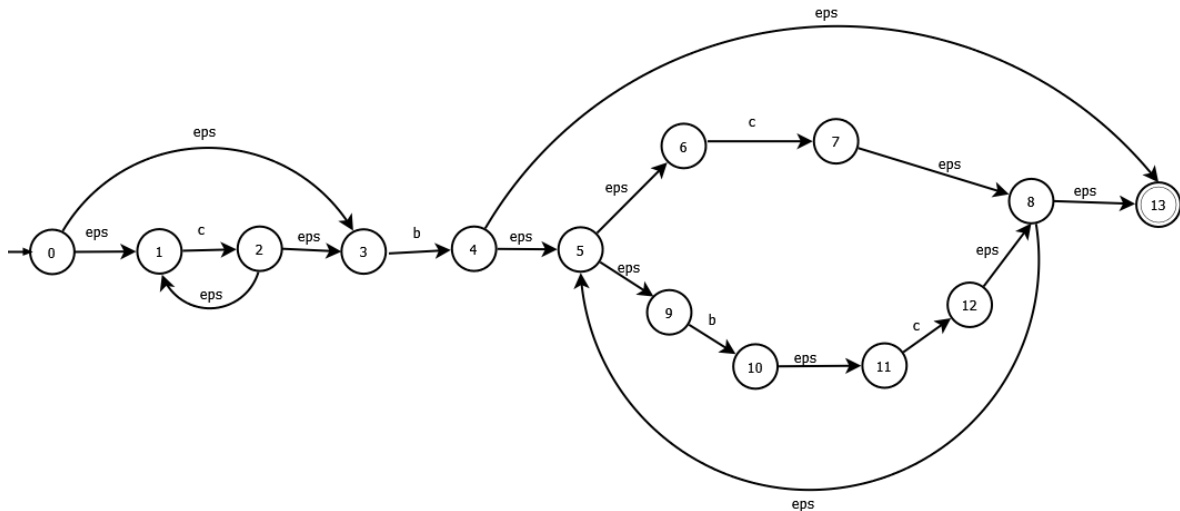


Ilustración 1. AFN de ER

Ahora haremos el proceso para obtener el AFN

$$\Sigma = \{b, c\}$$

Aplicando cerr-eps

$$\text{Cerr-eps}(\{0\}) = \{0, 3, 1\} = q_0$$

q0:

$$q_0 = \{0, 3, 1\}$$

$$\text{cerr-eps}(\text{mov}(q_0, b)) = \text{cerr-eps}(\{4\}) = \{4, 5, 6, 9, 13\} = q_1$$

$$\text{cerr-eps}(\text{mov}(q_0, c)) = \text{cerr-eps}(\{2\}) = \{2, 3\} = q_2$$

q1:

$$q_1 = \{4, 5, 6, 9, 13\}$$

$$\text{cerr-eps}(\text{mov}(q_1, b)) = \text{cerr-eps}(\{10\}) = \{10, 11\} = q_3$$

$$\text{cerr-eps}(\text{mov}(q_1, c)) = \text{cerr-eps}(\{7\}) = \{7, 8, 5, 6, 9, 13\} = q_4$$

q2:

$$q_2 = \{2, 3\}$$

$$\text{cerr-eps}(\text{mov}(q_2, b)) = \text{cerr-eps}(\{4\}) = \{4, 5, 6, 9, 13\} = q_1$$

$$\text{cerr-eps}(\text{mov}(q_2, c)) = \text{cerr-eps}(\{\emptyset\}) = \{\emptyset\}$$

q3:

$$q_3 = \{10, 11\}$$

$$\text{cerr-eps}(\text{mov}(q_3, b)) = \text{cerr-eps}(\{\emptyset\}) = \{\emptyset\}$$

$$\text{cerr-eps}(\text{mov}(q_3, c)) = \text{cerr-eps}(\{12\}) = \{12, 8, 13, 5, 6, 9\} = q_5$$

q4:

$$q_4 = \{7, 8, 5, 6, 9, 13\}$$

$$\text{cerr-eps}(\text{mov}(q_4, b)) = \text{cerr-eps}(\{10\}) = \{10, 11\} = q_3$$

$$\text{cerr-eps}(\text{mov}(q_4, c)) = \text{cerr-eps}(\{7\}) = \{7, 8, 5, 6, 9, 13\} = q_4$$

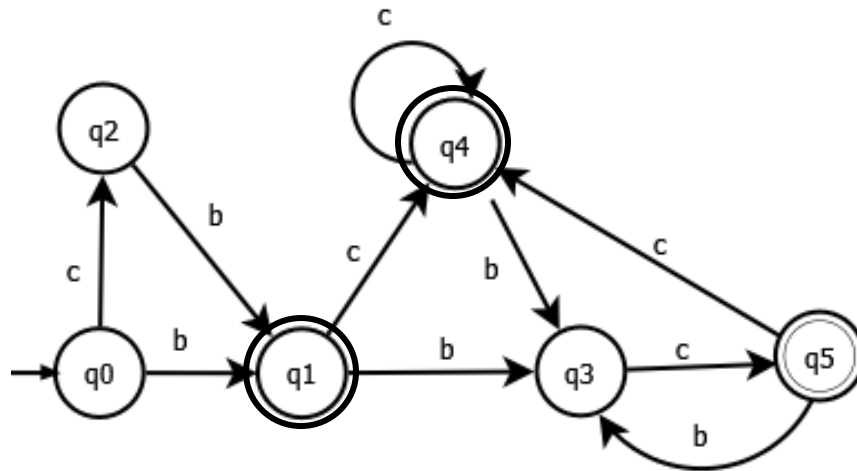
q5:

$$q_5 = \{12, 8, 13, 5, 6, 9, 13\}$$

$$\text{cerr-eps}(\text{mov}(q_5, b)) = \text{cerr-eps}(\{10\}) = \{10, 11\} = q_3$$

$$\text{cerr-eps}(\text{mov}(q_5, c)) = \text{cerr-eps}(\{7\}) = \{7, 8, 5, 6, 9, 13\} = q_4$$

**AFD**



*Ilustración 2. AFD*

**Minimizando AFD**

$$M_0 = \{q_2\}$$

$$M_1 = \{q_1, q_4, q_5\}$$

$$M_2 = \{q_3\}$$

$$M_3 = \{q_0\}$$

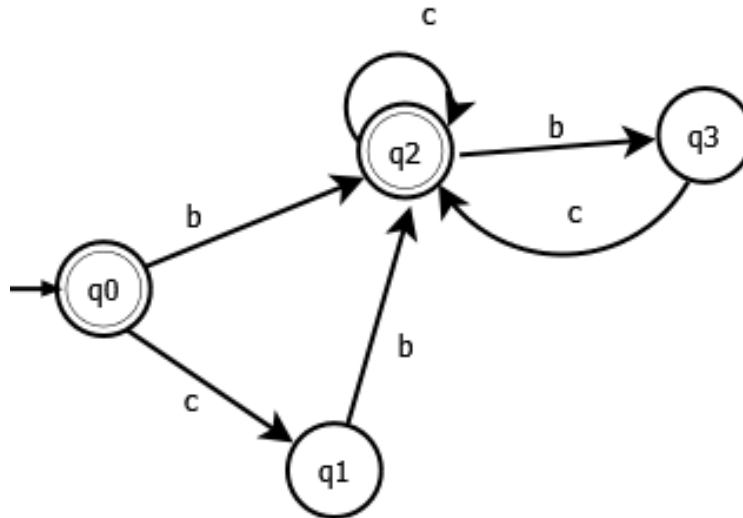


Ilustración 3. AFD minimizado

Así podemos poner a prueba el programa con distintas cadenas

```

PS D:\PC\ESCOM\ESCOM\Septimo Semestre ESCOM\Compiladores\Prácticas\Práctica 1\Programa> java .\DriverAFD.java bccccc
entrada = bccccc
Acep
PS D:\PC\ESCOM\ESCOM\Septimo Semestre ESCOM\Compiladores\Prácticas\Práctica 1\Programa> java .\DriverAFD.java cb
entrada = cb
Acep
PS D:\PC\ESCOM\ESCOM\Septimo Semestre ESCOM\Compiladores\Prácticas\Práctica 1\Programa> java .\DriverAFD.java cbcbc
entrada = cbcbc
rechazado
PS D:\PC\ESCOM\ESCOM\Septimo Semestre ESCOM\Compiladores\Prácticas\Práctica 1\Programa> java .\DriverAFD.java bb
entrada = bb
Acep
PS D:\PC\ESCOM\ESCOM\Septimo Semestre ESCOM\Compiladores\Prácticas\Práctica 1\Programa> java .\DriverAFD.java cbb
entrada = cbb
rechazado
  
```

Ilustración 4. Poniendo a prueba el programa

## 2. Obtener el programa para reconocer las palabras clave if, else y return

Para esta parte vamos a realizar el procedimiento del árbol con todas las palabras clave para llegar a la tabla de transiciones y poder construir el programa.

### IF

Aumentando la ER

**if#**

Creando árbol, calculando la última y primera posición y obteniendo anulables

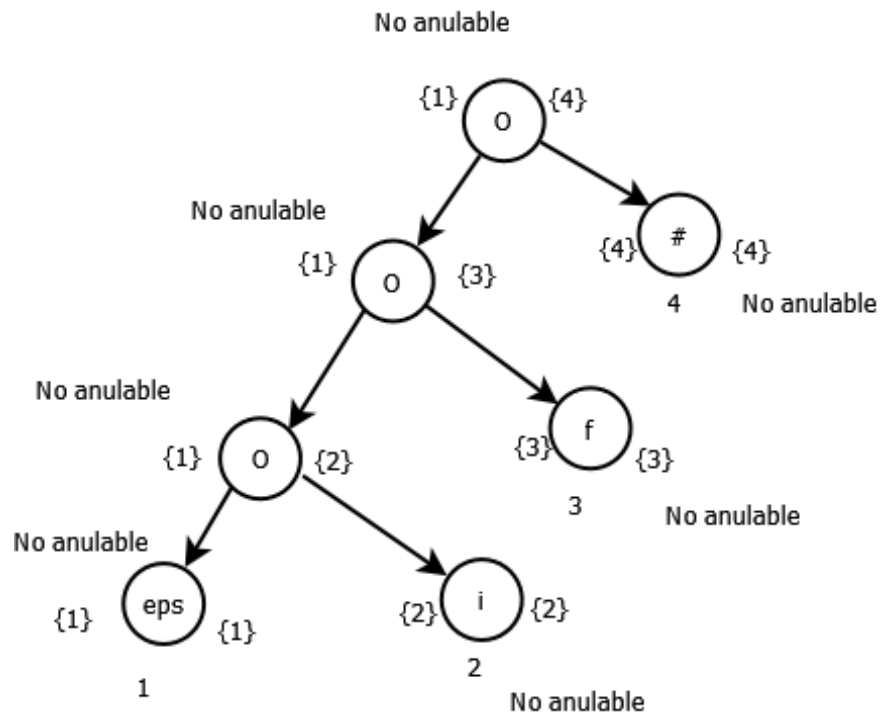


Ilustración 5. Árbol

Obteniendo siguiente posición

Nodo i	Siguiente pos
1	{2}
2	{3}
3	{4}
4	{5}
5	-

Calculando estados y tran y tranD de todo el vocabulario

**Primerapos(raíz) = {2} = q0**

**Tran[q0, i] = siguientepos(2) = {3} = q1**

**Tran[q0, f] = { ∅ }**

Con q1

**Tran [q1, i] = { ∅ }**

**Tran [q1,f] = {3}**

**tranD[q1, f] = siguientepos(3) = {4} = q2**

Obteniendo AFD mínimo



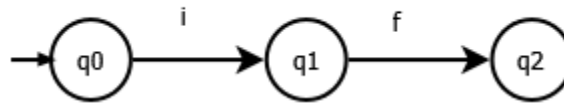


Ilustración 6. AFD

Tabla de transiciones:

Estado	I	F
q0	q1	-
q1	-	q2
q2	-	-

Lo siguiente que se muestra es la parte del código modificada para esta expresión regular.

```

while(tokenActual != '$') {
    switch(estadoActual) {
        case 0:
            if(tokenActual == 'f') break;
            else if(tokenActual == 'i') estadoActual = 1;
            break;
        case 1:
            if(tokenActual == 'f') estadoActual = 2;
            else if(tokenActual == 'i') break;
            break;
        case 2:
            if(tokenActual == 'f') estadoActual = 3;
            else if(tokenActual == 'i') estadoActual = 3;
            break;
    }
    avanzar();
}
if(estadoActual == 2)
    System.out.println("Acep");
else
    System.out.println("rechazado");
,

```

Ilustración 7. Código if

Y al probarlo tenemos:

```

PS D:\PC\ESCOM\ESCOM\Septimo Semestre ESCOM\Compiladores\Prácticas\Práctica 1\Programa> javac .\DriverAFD.java
PS D:\PC\ESCOM\ESCOM\Septimo Semestre ESCOM\Compiladores\Prácticas\Práctica 1\Programa> java DriverAFD.java ifi
entrada = ifi
rechazado
PS D:\PC\ESCOM\ESCOM\Septimo Semestre ESCOM\Compiladores\Prácticas\Práctica 1\Programa> java DriverAFD.java if
entrada = if
Acep
PS D:\PC\ESCOM\ESCOM\Septimo Semestre ESCOM\Compiladores\Prácticas\Práctica 1\Programa> java DriverAFD.java ifif
entrada = ifif
rechazado
PS D:\PC\ESCOM\ESCOM\Septimo Semestre ESCOM\Compiladores\Prácticas\Práctica 1\Programa>

```

Ilustración 8. Probando programa

Para las otras palabras clave, no será necesario realizar todo el proceso ya que al ser puras concatenaciones resulta en un afd muy sencillo con una tabla de transiciones igual de sencilla, por lo que solo se incluirá el AFD mínimo y su tabla de transiciones con sus pruebas del programa correspondientes.

**else**



*Ilustración 9. AFD mínimo*

Estado	e	l	s
q0	q1	-	-
q1	-	q2	-
q2	-	-	q3
q3	q4	-	-
q4	-	-	-

El programa modificado queda de la siguiente manera:

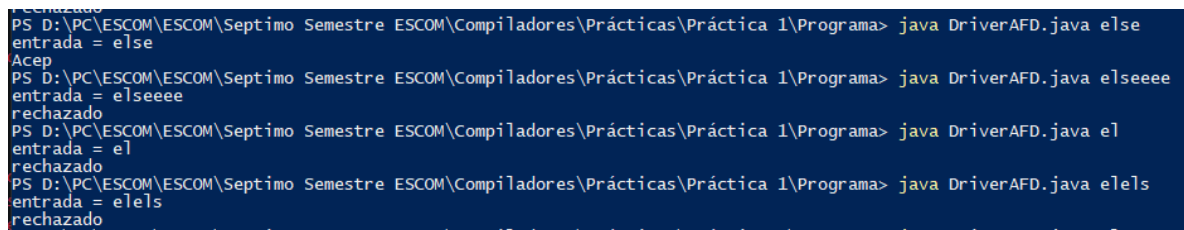
```

while(tokenActual != '$') {
    switch(estadoActual) {
        case 0:
            if(tokenActual == 'e') estadoActual = 1;
            else if(tokenActual == 'l') estadoActual = 5;
            else if(tokenActual == 's') estadoActual = 5;
            break;
        case 1:
            if(tokenActual == 'l') estadoActual = 2;
            else if(tokenActual == 'e') estadoActual = 5;
            else if(tokenActual == 's') estadoActual = 5;
            break;
        case 2:
            if(tokenActual == 's') estadoActual = 3;
            else if(tokenActual == 'l') estadoActual = 5;
            else if(tokenActual == 'e') estadoActual = 5;
            break;
        case 3:
            if(tokenActual == 'e') estadoActual = 4;
            else if(tokenActual == 'l') estadoActual = 5;
            else if(tokenActual == 's') estadoActual = 5;
        }
        avanzar();
    }
    if(estadoActual == 4)
        System.out.println("Acep");
    else
        System.out.println("rechazado");
}
}

```

Ilustración 10. Programa else

Y las pruebas del programa:



```

PS D:\PC\ESCOM\ESCOM\Septimo Semestre ESCOM\Compiladores\Prácticas\Práctica 1\Programa> java DriverAFD.java else
entrada = else
Acep
PS D:\PC\ESCOM\ESCOM\Septimo Semestre ESCOM\Compiladores\Prácticas\Práctica 1\Programa> java DriverAFD.java elseeee
entrada = elseeee
rechazado
PS D:\PC\ESCOM\ESCOM\Septimo Semestre ESCOM\Compiladores\Prácticas\Práctica 1\Programa> java DriverAFD.java el
entrada = el
rechazado
PS D:\PC\ESCOM\ESCOM\Septimo Semestre ESCOM\Compiladores\Prácticas\Práctica 1\Programa> java DriverAFD.java elels
entrada = elels
rechazado

```

Ilustración 11. Pruebas else

return

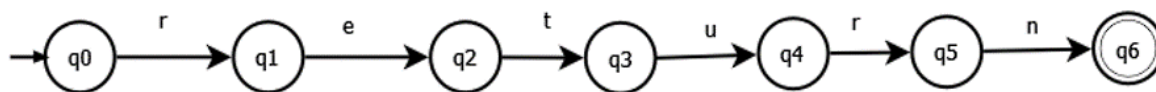


Ilustración 12. AFD mínimo

Aquí la tabla de transiciones

Estado	r	e	t	u	N
q0	q1	-	-	-	-
q1	-	q2	-	-	-
q2	-	-	q3	-	-
q3	-	-	-	q4	-
q4	q5	-	-	-	-
q5	-	-	-	-	q6
q6	-	-	-	-	-

El programa modificado queda de la siguiente manera:

```

while(tokenActual != '$') {
    switch(estadoActual) {
        case 0:
            if(tokenActual == 'r') estadoActual = 1;
            else if(tokenActual == 'e') estadoActual = 7;
            else if(tokenActual == 't') estadoActual = 7;
            else if(tokenActual == 'u') estadoActual = 7;
            else if(tokenActual == 'n') estadoActual = 7;
            break;
        case 1:
            if(tokenActual == 'e') estadoActual = 2;
            else if(tokenActual == 'r') estadoActual = 7;
            else if(tokenActual == 't') estadoActual = 7;
            else if(tokenActual == 'u') estadoActual = 7;
            else if(tokenActual == 'n') estadoActual = 7;
            break;
        case 2:
            if(tokenActual == 't') estadoActual = 3;
            else if(tokenActual == 'r') estadoActual = 7;
            else if(tokenActual == 'e') estadoActual = 7;
            else if(tokenActual == 'u') estadoActual = 7;
            else if(tokenActual == 'n') estadoActual = 7;
            break;
        case 3:
            if(tokenActual == 'u') estadoActual = 4;
            else if(tokenActual == 'r') estadoActual = 7;
            else if(tokenActual == 'e') estadoActual = 7;
            else if(tokenActual == 't') estadoActual = 7;
            else if(tokenActual == 'n') estadoActual = 7;
            break;
        case 4:
            if(tokenActual == 'r') estadoActual = 5;
            else if(tokenActual == 'l') estadoActual = 7;
            else if(tokenActual == 's') estadoActual = 7;
            else if(tokenActual == 's') estadoActual = 7;
            else if(tokenActual == 's') estadoActual = 7;
            break;
        case 5:
            if(tokenActual == 'n') estadoActual = 6;
            else if(tokenActual == 'r') estadoActual = 7;
            else if(tokenActual == 'e') estadoActual = 7;
            else if(tokenActual == 't') estadoActual = 7;
            else if(tokenActual == 'u') estadoActual = 7;
            break;
    }
}

```

```

        if(tokenActual == 'n') estadoActual = 7;
        else if(tokenActual == 'r') estadoActual = 7;
        else if(tokenActual == 'e') estadoActual = 7;
        else if(tokenActual == 't') estadoActual = 7;
        else if(tokenActual == 'u') estadoActual = 7;
        break;

    }
    avanzar();
}
if(estadoActual == 6)
    System.out.println("Acep");
else
    System.out.println("rechazado");
}

```

*Ilustración 13. Programa return*

Y las pruebas del programa:

```

PS D:\PC\ESCOM\ESCOM\Septimo Semestre ESCOM\Compiladores\Prácticas\Práctica 1\Programa> java DriverAFD.java return
entrada = return
Acep
PS D:\PC\ESCOM\ESCOM\Septimo Semestre ESCOM\Compiladores\Prácticas\Práctica 1\Programa> java DriverAFD.java returnn
entrada = returnn
rechazado
PS D:\PC\ESCOM\ESCOM\Septimo Semestre ESCOM\Compiladores\Prácticas\Práctica 1\Programa> java DriverAFD.java retu
entrada = retu
rechazado
PS D:\PC\ESCOM\ESCOM\Septimo Semestre ESCOM\Compiladores\Prácticas\Práctica 1\Programa> java DriverAFD.java retur
entrada = retur
rechazado

```

*Ilustración 14. Pruebas else*

## Conclusiones

Gracias a esta práctica reforcé mis conocimientos sobre pasar una  $er$  a  $afd$  y obtener su tabla de transiciones, esta última resulta muy importante pues gracias a esta se puede realizar un programa el cual nos indica que cadena acepta o no la expresión regular original, fue muy interesante implementar este programa y probarlo con varias expresiones regulares, entre ellas palabras clave que son usadas siempre cuando programamos.