```
/*
 * To change this license header, choose License Headers in Project
Properties.
 * To change this template file, choose Tools | Templates
 * and open the template in the editor.
 */
package utm;

/**
 *
 * @author daniel
 */
public class StateTable {
    private String[][] state;

    /**
     * Inicializamos la clase
     * @param state Recibimos la cadena en binario de la MT
     */
    public StateTable(String state) {
        this.state = new String[ state.length()/16 ][6];
        this.toMatrix(state);
    }

    /**
     * Imprime la matriz de estados
     * @return Val de la matriz en forma de String
     */
    public String getState() {
        String res="EA|O| M| SE|O| M| SE|\n";
        res += "--------------------\n";

        for (int i = 0; i < state.length; i++) {
            res += i+"  ";
            for (int j = 0; j < state[0].length; j++)
                res += state[i][j]+"  ";
            res += "\n";
```

```
        }

        return res;
    }

    /**
     * Covertimos la cadena binaria en una tabla de dos dimensiones
     * @param bin Cadena de la MT en binario
     */
    private void toMatrix(String bin) {
        int cont = 0;

        for (int i = 0; i < bin.length(); i+=8 ) {
            // Obtenemos todos los datos necesarios para insertar
en la matriz
            int write = Integer.parseInt( bin.charAt( i )+"" );
//Bit de escritura, qué poner
            int mov = Integer.parseInt( bin.charAt( i+1 )+"" ); //
Bit de movimiento, Der o Izq
            String next = bin.substring( i+2, i+8 );
            int nextState = this.bin2Dec(next);

            // Recordar que los inputs PARES de la MT es cuando In
= 0
            // e IMPAR cuando In = 1
            int colTmp = ( (i/8)%2==0 ) ? 0 : 3; // Ve si es par o
impar
            // if ( i%2==0 )
            //     colTmp = 1; // In = 0
            // else
            //     colTmp = 4; // In = 1
            this.state[ cont ][ colTmp ] = write+"";
            this.state[ cont ][ colTmp+1 ] = ( mov==0 ) ? "R" :
"I";
            this.state[ cont ][ colTmp+2 ] = (nextState==63 ) ? "H"
: nextState+"";
            cont = ( (i/8)%2==1 ) ? cont+1 : cont; // Ve si es par
o impar
```

```
        }
    }

    /**
     * Convierte el binario a decimal
     * @param bin Cadena con el número binario
     * @return Su conversión a decimal
     */
    private int bin2Dec(String bin){
        int res = 0;

        for (int i = 0; i < bin.length(); i++)
            res += Math.pow(2, i)*Integer.parseInt( bin.charAt(
bin.length()-i-1 )+"" );

        return res;
    }

    /**
     * Regresa el próximo estado dado el estado actual y el bit que
se recibe
     * @param car Dígito actual de la MT
     * @param state Número del estado actual
     * @return
     */
    public String[] nextStep(char car, int state){
        String[] fila = this.state[ state ];
        String[] res;

        if ( car=='0' ) {
            res = new String[]{ fila[0], fila[1], fila[2] };
        } else{
            res = new String[]{ fila[3], fila[4], fila[5] };
        }

        return res;
    }
}
```