

# Reporte practicas Teoría Computacional

## **ESCUELA SUPERIOR DE CÓMPUTO**

Colin Heredia Luis Antonio

Juárez Martínez Genaro

Grupo: 2CM5

20 de junio de 2020

# Programar una Expresión Regular

En esta practica vamos a programar la expresión regular  $L((0+10)^*(\epsilon +1))$  para generar 10 cadenas. Usaremos aleatoriamente las copias generadas por la cerradura de kleene. Las cadenas generadas las guardaremos en un archivo texto así como sus estados que paso por la expresión regular para generar esa cadena.

## Código main

ExpresionRegular.java

```
package expresionregular;

import javafx.application.Application;
import javafx.fxml.FXMLLoader;
import javafx.scene.Parent;
import javafx.scene.Scene;
import javafx.stage.Stage;

/**
 *
 * @author colin
 */
public class ExpresionRegular extends Application {

    @Override
    public void start(Stage stage) throws Exception {
        Parent root = FXMLLoader.load(getClass().getResource("FXMLMain.fxml"));

        Scene scene = new Scene(root);

        stage.setScene(scene);
        stage.show();
    }

    /**
     * @param args the command line arguments
     */
    public static void main(String[] args) {
        launch(args);
    }
}
```

## Control de vistas

FXMLMainController.java

```
package expresionregular;

import com.jfoenix.controls.JFXButton;
import com.jfoenix.controls.JFXTextArea;
import java.io.IOException;
import java.net.URL;
import java.util.ResourceBundle;
import javafx.event.ActionEvent;
import javafx.fxml.FXML;
import javafx.scene.Node;
import javafx.stage.Stage;

public class FXMLMainController {

    @FXML
    private ResourceBundle resources;

    @FXML
    private URL location;

    @FXML
    private JFXButton btnGenerarCadenas;

    @FXML
    private JFXButton btnSalir;

    @FXML
    private JFXTextArea areaText;

    @FXML
    private JFXTextArea areaTextEstados;

    private RE expresionR;

    @FXML
    void generarCadenas(ActionEvent event) throws IOException {
        this.expresionR.generar();
    }

    @FXML
    void salir(ActionEvent event) {
        Node source = (Node) event.getSource();
        Stage stage = (Stage) source.getScene().getWindow();
        stage.close();
    }

    @FXML
    void initialize() throws IOException {
        this.expresionR = new RE(areaTextEstados, areaText);
    }
}
```

## Código Archivos

ArchivoRutas.java

```
package expresionregular;

import java.io.BufferedWriter;
import java.io.File;
import java.io.FileWriter;
import java.io.IOException;

public class ArchivoRutas {
    private String ruta;
    private String nombreArchivo;
    private String pathCompleto;
    // variables para lectura y escritura
    private File file;
    private FileWriter fw;
    private BufferedWriter bw;

    public ArchivoRutas(String nombreArchivo) throws IOException
    {
        this.ruta = "./src/expresionregular/";
        this.nombreArchivo = nombreArchivo;
        this.pathCompleto = ruta+nombreArchivo+".txt";

        this.file = new File(this.pathCompleto);

        // si el archivo no existe crearlo
        if(!file.exists())
            file.createNewFile();
    }
    // escribe el texto en el archivo
    public void escribirArchivo(String contenido) throws IOException
    {
        this.fw = new FileWriter(this.file, true);
        this.bw = new BufferedWriter(fw);
        this.bw.write(contenido);
        this.bw.close();
    }
    // deja el archivo en blanco
    public void borrarContenido() throws IOException
    {
        this.fw = new FileWriter(this.file);
        this.bw = new BufferedWriter(fw);
        this.bw.write("");
        this.bw.close();
    }

    // getter para derivar del nombre un nuevo archivo con rutas validas
    public String getNameArchivo()
    {
        return this.nombreArchivo;
    }
}
```

## Código de la expresión regular

RE.java

```
package expresionregular;

import com.jfoenix.controls.JFXTextArea;
import java.io.IOException;
import java.util.Random;

/**
 *
 * @author colin
 */
public class RE {
    private Random rand;
    private ArchivoRutas estados;
    private ArchivoRutas cadenas;
    private JFXTextArea areaEstados;
    private JFXTextArea areaCadenas;

    /**
     * Constructor
     */
    public RE(JFXTextArea estados, JFXTextArea cadenas) throws IOException
    {
        this.estados = new ArchivoRutas("estados");
        this.cadenas = new ArchivoRutas("cadenas");
        this.areaCadenas = cadenas;
        this.areaEstados = estados;
    }

    /**
     * Generando la cadena en base a la expresion
     * regular
     */
    private void generar(int contador) throws IOException
    {
        boolean parteIzq;
        boolean parteDer;
        int numCad = contador+1;
        boolean cerradura;
        int copias;
        String cadena = "";
        this.rand = new Random();

        // determinamos si entramos a la cerradura
        cerradura = this.rand.nextBoolean();
        this.estados.escribirArchivo("Generando la cadena: " + numCad + "\n");
        this.areaEstados.appendText("Generando la cadena: " + numCad + "\n");

        if(cerradura)
        {
            // se entra a la parte de la cerradura (0+01)*
            // las copias de la cerradura seran
            copias = rand.nextInt(1001)+1; // maximo de copias 1000
            this.areaEstados.appendText("Entramos a la cerradura\n");
        }
    }
}
```

```

this.estados.escribirArchivo("Entramos a la cerradura\n");

// de la parte izquierda (0+01)* saber que vamos a aadir a la cadena
// si es verdarero sera 01 su no solo 0
parteIzq = rand.nextBoolean();
// de la parte derecha (e + 1) donde si es verdadero se pondra 1
// si es falso sera e
parteDer = rand.nextBoolean();
if(parteIzq)
{
    this.estados.escribirArchivo("Valor a escribir 10 -> se haran " +
        copias + " copias en la cadena concatenandolas\n");
    this.areaEstados.appendText("Valor a escribir 10 -> se haran " +
        copias + " copias en la cadena concatenandolas\n");
    for(int i=1; i<=copias; i++)
        cadena += "01";
} else
{
    this.estados.escribirArchivo("Valor a escribir 0 -> se haran " +
        copias + " copias en la cadena concatenandolas\n");
    this.areaEstados.appendText("Valor a escribir 0 -> se haran " +
        copias + " copias en la cadena concatenandolas\n");
    for(int i=1; i<=copias; i++)
        cadena += "0";
}
// para la parte ( e + 1)
if(parteDer)
{
    this.estados.escribirArchivo("Valor a escribir 1 en la cadena.\n");
    this.areaEstados.appendText("Vamos a escribir un 1 en la cadena.\n");
    cadena += "1";
} else
{
    this.estados.escribirArchivo("Valor a escribir e, como es vacio no lo pondremos.\n");
    this.areaEstados.appendText("Valor a escribir e, como es vacio no lo pondremos.\n");
}
} else
{
    // no entra a la cerradura
    this.estados.escribirArchivo("No entramos a la cerradura se considera e.\n");
    this.areaEstados.appendText("No entramos a la cerradura se considera e.\n");
    parteDer = rand.nextBoolean();
    if(parteDer)
    {
        this.estados.escribirArchivo("Valor a escribir 1 en la cadena.\n");
        ;
        this.areaEstados.appendText("Vamos a escribir un 1 en la cadena.\n");
        cadena += "1";
    } else

```

```

        {
            this.estados.escribirArchivo("Toca el valor 'e' vacío, no lo pondremos en la cadena.\n");
            this.areaEstados.appendText("Toca el valor 'e' vacío, no lo pondremos en la cadena.\n");
        }
    }
    // colocamos la cadena

    if(cadena != "")
    {
        this.cadenas.escribirArchivo(numCad+"-"+cadena+"\n");
        this.areaCadenas.appendText(numCad+"-"+cadena+"\n");
    } else
    {
        this.cadenas.escribirArchivo(numCad+"-e\n");
        this.areaCadenas.appendText(numCad+"-e\n");
    }
}

public void generar() throws IOException
{
    for(int i = 0; i<10; i++)
        generar(i);
}
}

```

## Capturas de programa



## Salida archivos de texto

```
estados: Bloc de notas
Archivo Edición Formato Ver Ayuda
Generando la cadena: 1
No entramos a la cerradura se concidera e.
Valor a escribir 1 en la cadena.
Generando la cadena: 2
No entramos a la cerradura se concidera e.
Valor a escribir 1 en la cadena.
Generando la cadena: 3
Entramos a la cerradura
Valor a escribir 10 -> se haran 12 copias en la cadena concatenandolas
Valor a escribir e, como es vacio no lo pondremos.
Generando la cadena: 4
No entramos a la cerradura se concidera e.
Valor a escribir 1 en la cadena.
Generando la cadena: 5
No entramos a la cerradura se concidera e.
Valor a escribir 1 en la cadena.
Generando la cadena: 6
No entramos a la cerradura se concidera e.
Valor a escribir 1 en la cadena.
Generando la cadena: 7
Entramos a la cerradura
Valor a escribir 0 -> se haran 6 copias en la cadena concatenandolas
Valor a escribir e, como es vacio no lo pondremos.
Generando la cadena: 8
Entramos a la cerradura
Valor a escribir 10 -> se haran 7 copias en la cadena concatenandolas
Valor a escribir e, como es vacio no lo pondremos.
Generando la cadena: 9
Entramos a la cerradura
Valor a escribir 0 -> se haran 9 copias en la cadena concatenandolas
Valor a escribir e, como es vacio no lo pondremos.
Generando la cadena: 10
Entramos a la cerradura
Valor a escribir 0 -> se haran 2 copias en la cadena concatenandolas
Valor a escribir e, como es vacio no lo pondremos.

cadenas: Bloc de notas
Archivo Edición Formato Ver Ayuda
1.-1
2.-1
3.-01010101010101010101
4.-1
5.-1
6.-1
7.-000000
8.-01010101010101
9.-00000000
10.-00
```

## Programa Gramática libre de contexto

Vamos realizar un programa que construya palindromes de un lenguaje binario. El lenguaje deberá solicitar únicamente la longitud del palindrome a calcular, de esta manera el programa deberá construir el palindrome de manera aleatoria. La longitud máxima que podría alcanzar un palindrome será de 100,000 caracteres. La salida del programa se irá a un archivo de texto y ahí especificarán qué regla se selecciono y la cadena resultante hasta llegar a la cadena final. El programa deberá ofrecer dos opciones: que el usuario defina la longitud del palindrome o que lo genere todo de manera automática. Usaremos la siguientes reglas:

- (1)  $P \rightarrow e$
- (2)  $P \rightarrow 0$
- (3)  $P \rightarrow 1$
- (4)  $P \rightarrow 0P0$
- (5)  $P \rightarrow 1P1$

## Código

## Código control de vistas



## FXMLMainControllerP.java

```

package generadorpalindromos;

import java.io.IOException;
import java.net.URL;
import java.util.Optional;
import java.util.ResourceBundle;
import javafx.event.ActionEvent;
import javafx.fxml.FXML;
import javafx.scene.Node;
import javafx.scene.control.Alert;
import javafx.scene.control.Button;
import javafx.scene.control.ButtonType;
import javafx.scene.control.CheckBox;
import javafx.scene.control.RadioButton;
import javafx.scene.control.Spinner;
import javafx.scene.control.SpinnerValueFactory;
import javafx.scene.control.TextArea;
import javafx.stage.Stage;

public class FXMLMainController {

    @FXML
    private TextArea areaPalindromos;

    @FXML
    private TextArea areaEstados;

    @FXML
    private Spinner<Integer> spinerTam;

    @FXML
    private Spinner<Integer> spinerCant;

    @FXML
    private Button btnGenerar;

    @FXML
    private Button btnSalir;

    @FXML
    private Button btnReinicio;

    @FXML
    private CheckBox chkAuto;

    // Generador class
    Generador gen;

    @FXML
    void automaticoEnable(ActionEvent event) {
        if(chkAuto.isSelected())
            this.spinerTam.setDisable(true);
        else
            this.spinerTam.setDisable(false);
    }

```

```

}
@FXML
void generar(ActionEvent event) throws IOException {
    if(!this.chkAuto.isSelected())
        this.gen.generar(this.spinerCant.getValue(),this.spinerTam.getValue());
    else
        this.gen.generar(this.spinerCant.getValue());

    this.btnReinicio.setDisable(false);
}

```

```

@FXML
void reinicio(ActionEvent event) throws IOException {
    Alert alert = new Alert(Alert.AlertType.CONFIRMATION);
    alert.setHeaderText(null);
    alert.setTitle("Confirmaci n");
    alert.setContentText("Esta accion borrara el archivo de estados y palindromos anteriores, Estas seguro?");
    Optional<ButtonType> action = alert.showAndWait();

    // Si hemos pulsado en aceptar
    if(action.get() == ButtonType.OK){
        this.gen.reiniciarTodo();
        this.btnReinicio.setDisable(true);
    }
}

```

```

@FXML
void salir(ActionEvent event) {
    Node source = (Node) event.getSource();
    Stage stage = (Stage) source.getScene().getWindow();
    stage.close();
}

```

```

@FXML
void initialize() throws IOException {
    // inicializamos los spinners
    SpinnerValueFactory<Integer> valueF = new SpinnerValueFactory.IntegerSpinnerValueFactory(1, 10); // numero de cadenas
    SpinnerValueFactory<Integer> valueF2 = new SpinnerValueFactory.IntegerSpinnerValueFactory(2, 100000); // minimo 3 hasta 100,000
    this.spinerTam.setValueFactory(valueF2);
    this.spinerCant.setValueFactory(valueF);
    this.spinerCant.setEditable(false);
    this.spinerTam.setEditable(false);
    this.btnReinicio.setDisable(true);
    this.gen = new Generador(this.spinerCant.getValue(), this.spinerTam.getValue(), areaEstados, areaPalindromos);
}
}

```

## Código main

GeneradorPalindromos.java

```
package generadorpalindromos;

import javafx.application.Application;
import javafx.fxml.FXMLLoader;
import javafx.scene.Parent;
import javafx.scene.Scene;
import javafx.stage.Stage;

/**
 *
 * @author colin
 */
public class GeneradorPalindromos extends Application {

    @Override
    public void start(Stage stage) throws Exception {
        Parent root = FXMLLoader.load(getClass().getResource("FXMLMain.fxml"));

        Scene scene = new Scene(root);

        stage.setScene(scene);
        stage.show();
    }

    /**
     * @param args the command line arguments
     */
    public static void main(String[] args) {
        launch(args);
    }
}
```

## Código para los archivos

## ArchivoRutasP.java

```

package generadorpalindromos;

import java.io.BufferedWriter;
import java.io.File;
import java.io.FileWriter;
import java.io.IOException;

public class ArchivoRutas {
    private String ruta;
    private String nombreArchivo;
    private String pathCompleto;
    // variables para lectura y escritura
    private File file;
    private FileWriter fw;
    private BufferedWriter bw;

    public ArchivoRutas(String nombreArchivo) throws IOException
    {
        this.ruta = "./src/generadorpalindromos/txts/";
        this.nombreArchivo = nombreArchivo;
        this.pathCompleto = ruta+nombreArchivo+".txt";

        this.file = new File(this.pathCompleto);

        // si el archivo no existe crearlo
        if(!file.exists())
            file.createNewFile();
    }
    // escribe el texto en el archivo
    public void escribirArchivo(String contenido) throws IOException
    {
        this.fw = new FileWriter(this.file, true);
        this.bw = new BufferedWriter(fw);
        this.bw.write(contenido);
        this.bw.close();
    }
    // deja el archivo en blanco
    public void borrarContenido() throws IOException
    {
        this.fw = new FileWriter(this.file);
        this.bw = new BufferedWriter(fw);
        this.bw.write("");
        this.bw.close();
    }

    // getter para derivar del nombre un nuevo archivo con rutas validas
    public String getNameArchivo()
    {
        return this.nombreArchivo;
    }
}

```

## Código Generador de palíndromos(Gramática libre de contexto)

Generador.java

```
package generadorpalindromos;

import java.io.IOException;
import java.util.Random;
import javafx.scene.control.Spinner;
import javafx.scene.control.TextArea;

/**
 *
 * @author colin
 */
public class Generador {
    // Esta clase es la que hace la generacion del palindromo
    private int tamPal;
    private int cantPals;
    private TextArea areaEstados;
    private TextArea areaPalindromos;
    private ArchivoRutas archivoEstados;
    private ArchivoRutas archivoPalindromos;
    private Random rand;
    private String palindromo;

    public Generador(int tamPal, int cantPals, TextArea areaEstados, TextArea
        areaPalindromos) throws IOException {
        this.tamPal = tamPal;
        this.cantPals = cantPals;
        this.areaEstados = areaEstados;
        this.areaPalindromos = areaPalindromos;
    }

    public Generador(int cantPals, TextArea areaEstados, TextArea areaPalindromos)
        throws IOException {
        this.cantPals = cantPals;
        this.areaEstados = areaEstados;
        this.areaPalindromos = areaPalindromos;
        this.tamPal = rand.nextInt(11)+1; // generar automaticamente el tam de
            cadena
    }

    /**
     Comenzaremos a generar
     */
    public void generar(int cantPals, int tam) throws IOException
    {
        this.archivoEstados = new ArchivoRutas("estados");
        this.archivoPalindromos = new ArchivoRutas("palindromos");
        boolean prodFiveFour; // si es verdadero tomara la regla 5 en otro caso la
            4
        this.cantPals = cantPals;
        this.tamPal = tam;

        for(int j = 0; j<this.cantPals; j++){
            this.palindromo = "P"; // (0)-> P
            this.archivoEstados.escribirArchivo("(0)->P.␣\n");
        }
    }
}
```

```

this.areaEstados.appendText("(0)->P.␣\n");
this.rand = new Random();
// se toma el tama o menos uno para al final
// tomar una de las reglas 1,2 o 3
for(int i = 0; i<this.tamPal; i++)
{
    prodFiveFour = this.rand.nextBoolean();
    if(prodFiveFour)
        reglasProd(5);
    else
        reglasProd(4);
    // tomamos el nuevo tama o de la cadena
    i = this.palindromo.length();
}
// cerramos nuestro palindromo
int cerrar = rand.nextInt((3-2)+1)+2; // force el 2 y 3 para que
    cumpliera con el rango pedido del usuario
// ya que si sale e faltaria un caracter para que cumpla con el rango.
reglasProd(cerrar);
this.areaEstados.appendText("Se cre ␣la␣cadena:"+this.palindromo+"\n");
;
this.archivoEstados.escribirArchivo("Se cre ␣la␣cadena:"+this.
    palindromo+"\n");
this.archivoPalindromos.escribirArchivo(this.palindromo+"\n");
this.areaPalindromos.appendText(this.palindromo+"\n");
}
}
public void reiniciarTodo() throws IOException
{
    this.areaEstados.clear();
    this.areaPalindromos.clear();
    this.archivoEstados.borrarContenido();
    this.archivoPalindromos.borrarContenido();
}
private void reglasProd(int regla) throws IOException
{
    switch(regla)
    {
        case 1:
            // regla P -> e
            // a adimos el texto a los arhcivos
            this.palindromo=this.palindromo.replace("P", "e");
            this.archivoEstados.escribirArchivo("(" +regla+")->" +this.palindromo
                +".␣\n");
            this.areaEstados.appendText("(" +regla+")->" +this.palindromo+" .␣\n")
                ;
            this.archivoEstados.escribirArchivo("Tenemos:" +this.palindromo+"
                Quitamos␣la␣e.␣\n");
            this.palindromo = this.palindromo.replace("e", "");

            break;
        case 2:
            // regla P -> 0
            this.palindromo = this.palindromo.replace("P", "0");
            this.archivoEstados.escribirArchivo("(" +regla+")->" +this.palindromo

```

```

        +".\n");
        this.areaEstados.appendText("(" + regla + ") -> " + this.palindromo + ".\n");

        break;
    case 3:
        // regla P -> 1
        this.palindromo = this.palindromo.replace("P", "1");
        this.archivoEstados.escribirArchivo("(" + regla + ") -> " + this.palindromo + ".\n");
        this.areaEstados.appendText("(" + regla + ") -> " + this.palindromo + ".\n");
        break;
    case 4:
        // regla P -> 0P0
        this.archivoEstados.escribirArchivo("(" + regla + ") -> 0" + this.palindromo + "0.\n");
        this.areaEstados.appendText("(" + regla + ") -> 0" + this.palindromo + "0.\n");
        this.palindromo = "0" + this.palindromo + "0";
        break;
    case 5:
        // regla P -> 1P1
        this.archivoEstados.escribirArchivo("(" + regla + ") -> 1" + this.palindromo + "1.\n");
        this.areaEstados.appendText("(" + regla + ") -> 1" + this.palindromo + "1.\n");
        this.palindromo = "1" + this.palindromo + "1";
        break;
    default:
        // ninguna regla

        break;
    }
}

void generar(int cantidad) throws IOException {
    this.archivoEstados = new ArchivoRutas("estados");
    this.archivoPalindromos = new ArchivoRutas("palindromos");
    boolean prodFiveFour; // si es verdadero tomara la regla 5 en otro caso la
    4
    this.cantPals = cantidad;
    this.rand = new Random();
    this.tamPal = rand.nextInt(100000)+1;

    for(int j = 0; j<this.cantPals; j++){
        this.palindromo = "P"; // (0) -> P
        this.archivoEstados.escribirArchivo("(0) -> P.\n");
        this.areaEstados.appendText("(0) -> P.\n");
        this.rand = new Random();
        // se toma el tama o menos uno para al final
        // tomar una de las reglas 1,2 o 3
        for(int i = 0; i<this.tamPal; i++)
        {
            prodFiveFour = this.rand.nextBoolean();
            if(prodFiveFour)

```

```

        reglasProd(5);
    else
        reglasProd(4);
    // tomamos el nuevo tamaño de la cadena
    i = this.palindromo.length();
}
// cerramos nuestro palindromo
int cerrar = rand.nextInt((3-2)+1)+2; // force el 2 y 3 para que
    cumpliera con el rango pedido del usuario
// ya que si sale e faltaria un caracter para que cumpla con el rango.
reglasProd(cerrar);
this.areaEstados.appendText("Se creó la cadena:"+this.palindromo+"\n");
;
this.archivoEstados.escribirArchivo("Se creó la cadena:"+this.palindromo+"\n");
this.archivoPalindromos.escribirArchivo(this.palindromo+"\n");
this.areaPalindromos.appendText(this.palindromo+"\n");
}
}
}

```

## Capturas versión manual

**Generador de palindromos**

Tamaño del palindromo:

☐ GenerarAutomatico

Cantidad de cadenas:

**Reglas**

- 1-  $P \rightarrow e$
- 2-  $P \rightarrow O$
- 3-  $P \rightarrow 1$
- 4-  $P \rightarrow OPO$
- 5-  $P \rightarrow 1P1$

Palindromos generados

Reglas de produccion

Luis Antonio Colin Heredia



Para las pruebas solo generaremos 5 cadenas de tamaño de 10.

# Generador de palindromos

Tamaño del palindromo:

☐ GenerarAutomatico

Cantidad de cadenas :

Reiniciar todo

Salir

Generar

110111011  
011010110  
111010111  
101010101  
111000111

Reglas


- 1.-  $P \rightarrow e$
- 2.-  $P \rightarrow 0$
- 3.-  $P \rightarrow 1$
- 4.-  $P \rightarrow 0P0$
- 5.-  $P \rightarrow 1P1$

(0)->P.  
(5)->1P1.  
(4)->01P10.  
(5)->101P101.  
(5)->1101P1011.  
(3)->110111011.  
Se creó la cadena:110111011  
(0)->P.  
(4)->0P0.  
(5)->10P01.  
(5)->110P011.  
(4)->0110P0110.  
(3)->011010110.  
Se creó la cadena:011010110  
(0)->P.  
(4)->0P0.

Luis Antonio Colin Heredia


16

## Archivos de salida

 estados: Bloc de notas

Archivo Edición Formato Ver Ayuda

```
(0)->P.  
(5)->1P1.  
(4)->01P10.  
(5)->101P101.  
(5)->1101P1011.  
(3)->110111011.  
Se creó la cadena:110111011  
(0)->P.  
(4)->0P0.  
(5)->10P01.  
(5)->110P011.  
(4)->0110P0110.  
(3)->011010110.  
Se creó la cadena:011010110  
(0)->P.  
(4)->0P0.  
(5)->10P01.  
(5)->110P011.  
(5)->1110P0111.  
(3)->111010111.  
Se creó la cadena:111010111  
(0)->P.  
(4)->0P0.  
(5)->10P01.  
(4)->010P010.  
(5)->1010P0101.  
(3)->101010101.  
Se creó la cadena:101010101  
(0)->P.  
(4)->0P0.  
(5)->10P01.  
(5)->110P011.  
(5)->1110P0111.  
(2)->111000111.  
Se creó la cadena:111000111
```

 palindr... — □ ×

Archivo Edición Formato Ver Ayuda

```
110111011  
011010110  
111010111  
101010101  
111000111
```

< >

UNIX (LF) UTF-8

## Pruebas automáticas

# Generador de palindromos

Tamaño del palindromo:

☒ GenerarAutomatico

Cantidad de cadenas :

Reiniciar todo

Salir

Generar

Palindromos generados

Reglas de produccion

### Reglas

- 1.-  $P \rightarrow e$
- 2.-  $P \rightarrow 0$
- 3.-  $P \rightarrow 1$
- 4.-  $P \rightarrow 0P0$
- 5.-  $P \rightarrow 1P1$

Luis Antonio Colin Heredia

## Generando palíndromos automático

# Generador de palindromos

Tamaño del palindromo:  ☒ GenerarAutomatico

Cantidad de cadenas :

1001001  
0100010  
1010101  
1101011  
0110110  
1010101  
1100011

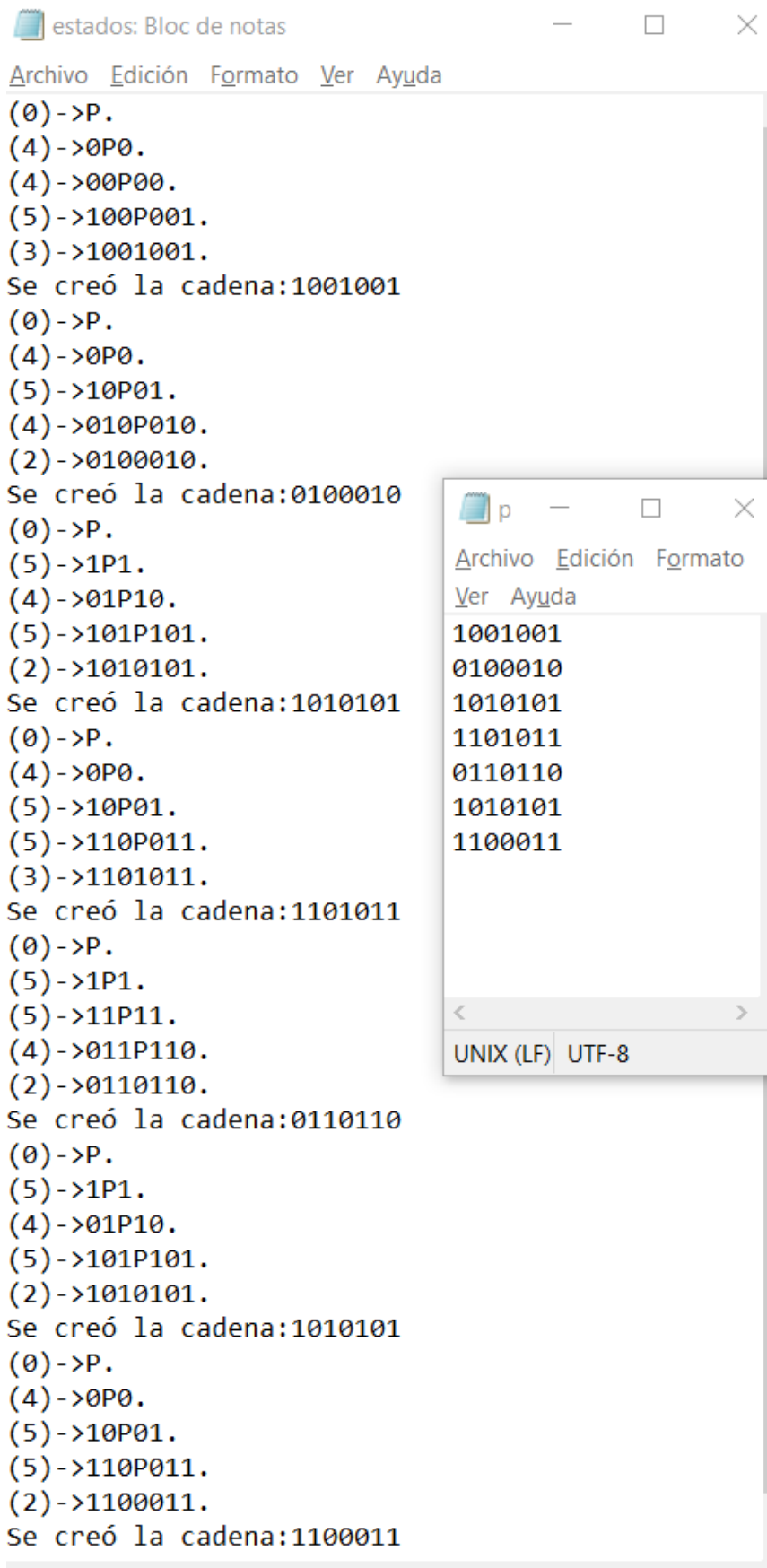
(0)->P.  
(4)->0P0.  
(4)->00P00.  
(5)->100P001.  
(3)->1001001.  
Se creó la cadena:1001001  
(0)->P.  
(4)->0P0.  
(5)->10P01.  
(4)->010P010.  
(2)->0100010.  
Se creó la cadena:0100010  
(0)->P.  
(5)->1P1.  
(4)->01P10.  
(5)->101P101.

Luis Antonio Colin Heredia

### Reglas

- 1.-  $P \rightarrow e$
- 2.-  $P \rightarrow 0$
- 3.-  $P \rightarrow 1$
- 4.-  $P \rightarrow 0P0$
- 5.-  $P \rightarrow 1P1$

## Archivos de salida automático



```
estados: Bloc de notas
Archivo Edición Formato Ver Ayuda
(0)->P.
(4)->0P0.
(4)->00P00.
(5)->100P001.
(3)->1001001.
Se creó la cadena:1001001
(0)->P.
(4)->0P0.
(5)->10P01.
(4)->010P010.
(2)->0100010.
Se creó la cadena:0100010
(0)->P.
(5)->1P1.
(4)->01P10.
(5)->101P101.
(2)->1010101.
Se creó la cadena:1010101
(0)->P.
(4)->0P0.
(5)->10P01.
(5)->110P011.
(3)->1101011.
Se creó la cadena:1101011
(0)->P.
(5)->1P1.
(5)->11P11.
(4)->011P110.
(2)->0110110.
Se creó la cadena:0110110
(0)->P.
(5)->1P1.
(4)->01P10.
(5)->101P101.
(2)->1010101.
Se creó la cadena:1010101
(0)->P.
(4)->0P0.
(5)->10P01.
(5)->110P011.
(2)->1100011.
Se creó la cadena:1100011
```

# Autómata de pila (PDA)

El autómata de pila debe de reconocer el lenguaje libre de contexto  $0^n1^n, n \geq 1$ .

Se debe recibir una cadena ingresada manualmente o generada una por la máquina de máximo 1000 caracteres.

## Código

### Main

AutomataPila.java

```
package automatapila;

import javafx.application.Application;
import javafx.fxml.FXMLLoader;
import javafx.scene.Parent;
import javafx.scene.Scene;
import javafx.stage.Stage;

/**
 *
 * @author colin
 */
public class AutomataPila extends Application {

    @Override
    public void start(Stage stage) throws Exception {
        Parent root = FXMLLoader.load(getClass().getResource("FXMLMain.fxml"));

        Scene scene = new Scene(root);

        stage.setScene(scene);
        stage.show();
    }

    /**
     * @param args the command line arguments
     */
    public static void main(String[] args) {
        launch(args);
    }
}
```

## Archivos

Realmente es la misma clase de todas las practicas anteriores. Solo cambia la ruta donde se guarda el archivo.

ArchivosRutasPila.java

```
package automataPila;

import java.io.BufferedWriter;
import java.io.File;
import java.io.FileWriter;
import java.io.IOException;

/**
 *
 * @author colin
 */
public class ArchivosRutas {
    private String ruta;
    private String nombreArchivo;
    private String pathCompleto;
    // variables para lectura y escritura
    private File file;
    private FileWriter fw;
    private BufferedWriter bw;

    public ArchivosRutas(String nombreArchivo) throws IOException
    {
        this.ruta = "./src/automataPila/";
        this.nombreArchivo = nombreArchivo;
        this.pathCompleto = ruta+nombreArchivo+".txt";

        this.file = new File(this.pathCompleto);

        // si el archivo no existe crearlo
        if(!file.exists())
            file.createNewFile();
    }
    // escribe el texto en el archivo
    public void escribirArchivo(String contenido) throws IOException
    {
        this.fw = new FileWriter(this.file, true);
        this.bw = new BufferedWriter(fw);
        this.bw.write(contenido);
        this.bw.close();
    }
    // deja el archivo en blanco
    public void borrarContenido() throws IOException
    {
        this.fw = new FileWriter(this.file);
        this.bw = new BufferedWriter(fw);
        this.bw.write("");
        this.bw.close();
    }
}
```

```

    // getter para derivar del nombre un nuevo archivo con rutas validas
    public String getNameArchivo()
    {
        return this.nombreArchivo;
    }
}

```

## Generador de cadenas

### GeneradorCadenas.java

```

/*
    Cree esta clase para tambien usarla en
    la maquina de turing junto con la de ArchivosRutas
*/
package practica9.maquinaturing;

import java.util.Random;

/**
 *
 * @author colin
 */
public class GeneradorCadenas {
    private Random rand;
    private String cad;

    public GeneradorCadenas()
    {
        this.rand = new Random();
        this.cad = "";
    }

    /*
        Usaria las reglas de produccion
        1)S->0S1
        2)S->01
        Pero solo generaria cadenas validas
        lo cual mejor lo hare aleatoriamente asi pueden ser
        cadenas valiadas y no validas.
    */
    public String generarCadena(int tam)
    {
        this.cad = "";
        int cantCeros = rand.nextInt(tam)+1;
        int cantUnos = tam-cantCeros;
        // ceros
        for(int i = 0; i < cantCeros; i++)
            this.cad += "0";
        // unos
        for(int i = 0; i < cantUnos; i++)
            this.cad += "1";

        return this.cad;
    }
}

```



```

    }
}

```

## Control de vista

FXMLMainControllerPila.java

```

package automatapila;

import java.io.IOException;
import java.net.URL;
import java.util.ResourceBundle;
import javafx.event.ActionEvent;
import javafx.fxml.FXML;
import javafx.scene.control.Button;
import javafx.scene.control.CheckBox;
import javafx.scene.control.Spinner;
import javafx.scene.control.SpinnerValueFactory;
import javafx.scene.control.TextArea;
import javafx.scene.control.TextField;
import javafx.scene.paint.Color;
import javafx.scene.paint.Paint;
import javafx.scene.text.Text;

public class FXMLMainController {
    @FXML
    private CheckBox chkAuto;
    @FXML
    private TextField textCadena;
    @FXML
    private Spinner<Integer> longitudCadena;
    @FXML
    private Button btnGenearCadena;
    @FXML
    private Text textCadenaEvaluar;
    @FXML
    private TextArea textAreaProceso;
    @FXML
    private Text textMensaje;
    @FXML
    private Button btnStart;

    //Automata de pila
    private PDA automataDePila;
    //generador de cadenas
    private GeneradorCadenas generador;
    private String cad;

    @FXML
    void iniciar(ActionEvent event) throws IOException {

        if(this.chkAuto.isSelected()) // generacion automatica
            this.automataDePila = new PDA(this.cad, this.textAreaProceso);
        else{

```

```

        // cadena manual
        this.textCadenaEvaluar.setText(this.textCadena.getText().trim());
        this.automataDePila = new PDA(this.textCadena.getText().trim(), this.
            textAreaProceso);
    }

    if(this.automataDePila.empezarEvaluacion()){
        this.textMensaje.setText("La cadena es valida");
        this.textMensaje.setFill(Color.web("#40FF00"));
    }
    else{
        this.textMensaje.setText("La cadena es invalida");
        this.textMensaje.setFill(Color.web("#FF0000"));
    }
}

@FXML
void isAuto(ActionEvent event) {
    this.textCadena.setDisable(this.chkAuto.isSelected());
    this.btnGenearCadena.setDisable(!this.chkAuto.isSelected());
}

@FXML
void generarCad(ActionEvent event)
{
    this.cad = this.generador.generarCadena(this.longitudCadena.getValue());
    this.textCadenaEvaluar.setText(this.cad);
}

@FXML
void initialize() {
    SpinnerValueFactory values = new SpinnerValueFactory.
        IntegerSpinnerValueFactory(2, 10000,1);
    this.longitudCadena.setValueFactory(values);
    this.generador = new GeneradorCadenas();
    this.btnGenearCadena.setDisable(!this.chkAuto.isSelected());
    this.btnStart.setDisable(false);
}
}

```

## Código autómatas de pila (PDA)

PDA.java

```
package automatapila;
import java.io.IOException;
import java.util.Stack;
import javafx.scene.control.TextArea;
/**
 *
 * @author colin
 */
public class PDA {
    // aqui estara escrito el codigo para el automata de pila.
    private final String letraG;
    private String estado;
    private String cadena;
    private Stack<String> pila;
    private boolean isValida;
    private ArchivosRutas secuencia;
    private TextArea area;

    public PDA(String cadena , TextArea area)
    {
        // constructor
        this.letraG = "\u03B4";
        this.area = area;
        this.cadena = cadena;
        this.estado = "q";
        this.pila = new Stack();
        this.isValida = false;
    }
    /**
     Empezara el proceso del automata
     para determinar si la cadena es valida
     devolvera un true o false si la cadena es o no valida
     */
    boolean empezarEvaluacion() throws IOException
    {
        // metemos la Z0 y nuestro estado es q
        this.pila.push("Z");
        String cadAux = this.cadena;
        this.secuencia = new ArchivosRutas("secuencia");

        // Primero verifico que empiece con 0 y termine con 1
        this.area.appendText("\n"+this.letraG+"["+this.estado+" , "+this.cadena+" , "+this.pila.toString()+"\n");
        this.secuencia.escribirArchivo("\n"+this.letraG+"["+this.estado+" , "+this.cadena+" , "+this.pila.toString()+"\n");

        if(this.cadena.endsWith("1") && this.cadena.startsWith("0"))
        {
            this.cadena += " "; // agregamos el caracter nulo para leerlo
            for(int i=0; i < this.cadena.length(); i++)
            {

```

```

// si leemos un 0
if(this.cadena.charAt(i) == '0')
{
    // checamos que sea estado q, si es un estado f o p
    // no seria una cadena valida ya que viene de sacar un 1
    // o viene de leer el final de la cadena
    if(this.estado.equals("q"))
    {
        this.estado = "q"; // el estado se mantiene en q
        // hacemos push a la pila con X
        this.pila.push("X");
        cadAux = cadAux.replaceFirst("0", ""); // remplzamos el
            // cero por vacio
        // imprimimos
        this.imprimirDatos(cadAux);
    } else
        break;
}
// si leemos un 1
if(this.cadena.charAt(i) == '1')
{
    // verificamos que sea de un estado q o p
    if(this.estado.equals("q") || this.estado.equals("p"))
    {
        // checamos que al hacer pop no sea z
        if(!this.pila.pop().equals("Z"))
        {
            this.estado = "p";
            cadAux = cadAux.replaceFirst("1", "");
            // imprimimos
            this.imprimirDatos(cadAux);
        } else // si saca la Z sera una cadena invalida
            break;

        } else // si viene de f la funcion no esta definca
            break;
    }

// si llega el caracter vacio ya leyo toda la cadena
if(this.cadena.charAt(i) == '_')
{
    this.estado = "f";
    // si la pila al inicio tiene la Z
    if(this.pila.peek().equals("Z") && this.estado.equals("f"))
    {
        this.isValida = true;
        // imprimimos
        this.imprimirDatos(cadAux);
    }
    else // si no es la Z entonces la cadena es invalida
        break;
}
}
else

```

```

        this.isValida = false;

    return this.isValida;
}

private void imprimirDatos(String datos) throws IOException
{
    StringBuilder pilaDatos;
    String auxPila = this.pila.toString();
    auxPila = auxPila.replace("[", "");
    auxPila = auxPila.replace("]", "");
    auxPila = auxPila.replaceAll(",", "");
    pilaDatos = new StringBuilder(auxPila);
    pilaDatos = pilaDatos.reverse();

    this.area.appendText(this.letraG+"["+this.estado+" , "+datos+" , "+pilaDatos
        +"]\n");
    this.secuencia.escribirArchivo(this.letraG+"["+this.estado+" , "+datos+" , "+
        pilaDatos+"]\n");
}
}

```

## Pruebas (modo manual)

**AUTÓMATA DE PILA**

Ingresar cadena:

Longitud cadena:

Cadena:

**Proceso**

Proceso de evaluación de cadena

Ingresando la cadena manualmente.

**AUTÓMATA DE PILA**

Ingresar cadena:

Longitud cadena:

☐ Automático

Cadena:

**Proceso**

Proceso de evaluación de cadena

## Iniciando el programa de modo manual

The screenshot shows the 'AUTÓMATA DE PILA' application window. At the top, the title 'AUTÓMATA DE PILA' is displayed in large, bold, black letters. Below the title, there is a text input field labeled 'Ingresar cadena:' containing the string '0000011111'. To the right of this field is a dropdown menu for 'Longitud cadena:' set to '2'. Further right are three buttons: 'Automático' (disabled), 'Generar' (disabled), and 'Iniciar' (active). Below these controls, the text 'Cadena: 0000011111' is shown. A large section titled 'Proceso' contains a scrollable list of states and transitions. The states listed are:  $\delta[q, 0000011111, [Z]]$ ,  $\delta[q, 000011111, XZ]$ ,  $\delta[q, 00011111, XXZ]$ ,  $\delta[q, 0011111, XXXZ]$ ,  $\delta[q, 011111, XXXXZ]$ ,  $\delta[q, 11111, XXXXXZ]$ ,  $\delta[p, 1111, XXXXZ]$ ,  $\delta[p, 111, XXXXZ]$ ,  $\delta[p, 11, XXXZ]$ ,  $\delta[p, 1, XZ]$ ,  $\delta[p, ., Z]$ , and  $\delta[f, ., Z]$ . To the right of the 'Proceso' section, the text 'La cadena es valida' is displayed in green.

**AUTÓMATA DE PILA**

Ingresar cadena: 0000011111

Longitud cadena: 2

☐ Automático

Cadena: **0000011111**

**Proceso**

- $\delta[q, 0000011111, [Z]]$
- $\delta[q, 000011111, XZ]$
- $\delta[q, 00011111, XXZ]$
- $\delta[q, 0011111, XXXZ]$
- $\delta[q, 011111, XXXXZ]$
- $\delta[q, 11111, XXXXXZ]$
- $\delta[p, 1111, XXXXZ]$
- $\delta[p, 111, XXXXZ]$
- $\delta[p, 11, XXXZ]$
- $\delta[p, 1, XZ]$
- $\delta[p, ., Z]$
- $\delta[f, ., Z]$

**La cadena es valida**

## Modo Automático activando

Generando una cadena de 10.

The screenshot shows the 'AUTÓMATA DE PILA' application window. The title 'AUTÓMATA DE PILA' is at the top. Below it, the 'Ingresar cadena:' field contains '0000011111'. The 'Longitud cadena:' dropdown is set to '10'. The 'Automático' checkbox is now checked, and the 'Generar' button is active. The 'Iniciar' button remains active. The 'Proceso' section is not visible in this screenshot.

**AUTÓMATA DE PILA**

Ingresar cadena: 0000011111

Longitud cadena: 10

☒ Automático

## Cadena Generada automáticamente

# AUTÓMATA DE PILA

Ingresar cadena:

Longitud cadena:

Cadena: **0000000111**

☒ Automático

## Iniciando con cadena generada

# AUTÓMATA DE PILA

Ingresar cadena:

Longitud cadena:

Cadena: **0000000111**

☒ Automático

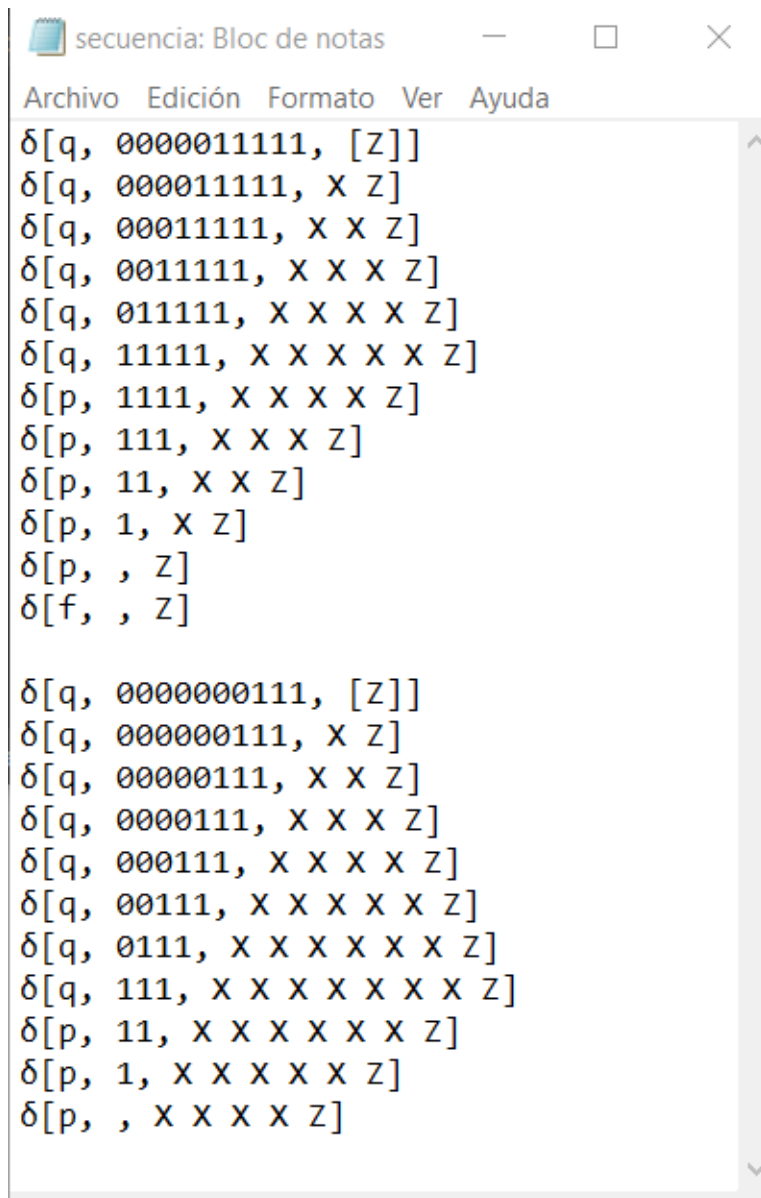
Proceso

$\delta[q, 0000000111, [Z]]$   
 $\delta[q, 000000111, XZ]$   
 $\delta[q, 00000111, XXZ]$   
 $\delta[q, 0000111, XXXZ]$   
 $\delta[q, 000111, XXXXZ]$   
 $\delta[q, 00111, XXXXXZ]$   
 $\delta[q, 0111, XXXXXXZ]$   
 $\delta[q, 111, XXXXXXXZ]$   
 $\delta[p, 11, XXXXXXXZ]$   
 $\delta[p, 1, XXXXXXZ]$   
 $\delta[p, ., XXXXXZ]$

**La cadena es invalida**



## Salida en archivo



```
secuencia: Bloc de notas
Archivo  Edición  Formato  Ver  Ayuda
delta[q, 0000011111, [Z]]
delta[q, 000011111, X Z]
delta[q, 00011111, X X Z]
delta[q, 0011111, X X X Z]
delta[q, 011111, X X X X Z]
delta[q, 11111, X X X X X Z]
delta[p, 1111, X X X X Z]
delta[p, 111, X X X Z]
delta[p, 11, X X Z]
delta[p, 1, X Z]
delta[p, , Z]
delta[f, , Z]

delta[q, 0000000111, [Z]]
delta[q, 000000111, X Z]
delta[q, 00000111, X X Z]
delta[q, 0000111, X X X Z]
delta[q, 000111, X X X X Z]
delta[q, 00111, X X X X X Z]
delta[q, 0111, X X X X X X Z]
delta[q, 111, X X X X X X X Z]
delta[p, 11, X X X X X X Z]
delta[p, 1, X X X X X Z]
delta[p, , X X X X Z]
```

# Máquina de Turing

Programar una máquina de turing que pueda reconocer  $0^n 1^n, n \geq 1$ .

Las cadenas son ingresadas por el usuario o generadas automáticamente con una longitud máxima de 1000.

## Código

### Main

Practica9MaquinaTuring.java

```
package practica9.maquinaturing;

import javafx.application.Application;
import javafx.fxml.FXMLLoader;
import javafx.scene.Parent;
import javafx.scene.Scene;
import javafx.stage.Stage;

/**
 *
 * @author colin
 */
public class Practica9MaquinaTuring extends Application {

    @Override
    public void start(Stage stage) throws Exception {
        Parent root = FXMLLoader.load(getClass().getResource("FXMLMain.fxml"));

        Scene scene = new Scene(root);

        stage.setScene(scene);
        stage.show();
    }

    /**
     * @param args the command line arguments
     */
    public static void main(String[] args) {
        launch(args);
    }
}
```

# ArchivosRutasMT.java

```
package practica9.maquinaturing;

import java.io.BufferedWriter;
import java.io.File;
import java.io.FileWriter;
import java.io.IOException;

/**
 *
 * @author colin
 */
public class ArchivosRutas {
    private String ruta;
    private String nombreArchivo;
    private String pathCompleto;
    // variables para lectura y escritura
    private File file;
    private FileWriter fw;
    private BufferedWriter bw;

    public ArchivosRutas(String nombreArchivo) throws IOException
    {
        this.ruta = "./src/practica9/maquinaturing/";
        this.nombreArchivo = nombreArchivo;
        this.pathCompleto = ruta+nombreArchivo+".txt";

        this.file = new File(this.pathCompleto);

        // si el archivo no existe crearlo
        if(!file.exists())
            file.createNewFile();
    }
    // escribe el texto en el archivo
    public void escribirArchivo(String contenido) throws IOException
    {
        this.fw = new FileWriter(this.file, true);
        this.bw = new BufferedWriter(fw);
        this.bw.write(contenido);
        this.bw.close();
    }
    // deja el archivo en blanco
    public void borrarContenido() throws IOException
    {
        this.fw = new FileWriter(this.file);
        this.bw = new BufferedWriter(fw);
        this.bw.write("");
        this.bw.close();
    }

    // getter para derivar del nombre un nuevo archivo con rutas validas
    public String getNameArchivo()
    {
        return this.nombreArchivo;
    }
}
```

```
}
```

## GeneradorCadenas

GeneradorCadenas.java

```
/*
    Cree esta clase para tambien usarla en
    la maquina de turing junto con la de ArchivosRutas
*/
package practica9.maquinaturing;

import java.util.Random;

/**
 *
 * @author colin
 */
public class GeneradorCadenas {
    private Random rand;
    private String cad;

    public GeneradorCadenas()
    {
        this.rand = new Random();
        this.cad = "";
    }

    /*
        Usaria las reglas de produccion
        1)S->0S1
        2)S->01
        Pero solo generaria cadenas validas
        lo cual mejor lo hare aleatoriamente asi pueden ser
        cadenas valiadas y no validas.
    */
    public String generarCadena(int tam)
    {
        this.cad = "";
        int cantCeros = rand.nextInt(tam)+1;
        int cantUnos = tam-cantCeros;
        // ceros
        for(int i = 0; i < cantCeros; i++)
            this.cad += "0";
        // unos
        for(int i = 0; i < cantUnos; i++)
            this.cad += "1";

        return this.cad;
    }
}
```

## Control de vistas

FXMLMainControllerMT.java

```
package practica9.maquinaturing;

import java.io.IOException;
import java.net.URL;
import java.util.ResourceBundle;
import javafx.event.ActionEvent;
import javafx.fxml.FXML;
import javafx.scene.control.Button;
import javafx.scene.control.CheckBox;
import javafx.scene.control.Spinner;
import javafx.scene.control.SpinnerValueFactory;
import javafx.scene.control.TextArea;
import javafx.scene.control.TextField;
import javafx.scene.paint.Color;
import javafx.scene.text.Text;

public class FXMLMainController {

    @FXML
    private TextField txtCadena;

    @FXML
    private Spinner<Integer> sppinerNum;

    @FXML
    private CheckBox chkAuto;

    @FXML
    private Button btnGenerear;

    @FXML
    private Text txtCadAna;

    @FXML
    private Text txtValida;

    @FXML
    private TextArea textAreaDatos;

    @FXML
    private Button btnReset;

    // variables para la Maquina de turing
    MaquinaTuring maquinaT;
    GeneradorCadenas gen;
    String cadena;

    @FXML
    void automatico(ActionEvent event) {
        this.btnGenerear.setDisable(!this.chkAuto.isSelected());
        this.txtCadena.setDisable(this.chkAuto.isSelected());
        this.sppinerNum.setDisable(!this.chkAuto.isSelected());
    }
}
```

```

}

@FXML
void generarCadena(ActionEvent event) {
    this.cadena = this.gen.generarCadena(this.sppinerNum.getValue());
    this.txtCadAna.setText(cadena);
}

@FXML
void iniciarMaquina(ActionEvent event) throws InterruptedException, IOException
{
    if(this.chkAuto.isSelected())
    {
        // modo automatico
        this.txtCadAna.setText(this.cadena);
        this.maquinaT = new MaquinaTuring(this.cadena, this.textAreaDatos);
        this.validar();
    }else
    {
        // modo manual
        this.txtCadAna.setText(this.txtCadena.getText());
        this.maquinaT = new MaquinaTuring(this.txtCadena.getText(), this.
            textAreaDatos);
        this.validar();
    }
}

@FXML
void limpiar(ActionEvent event) {
    this.textAreaDatos.clear();
}

@FXML
void initialize() {
    this.btnGenerear.setDisable(!this.chkAuto.isSelected());
    this.txtCadena.setDisable(this.chkAuto.isSelected());
    this.sppinerNum.setDisable(!this.chkAuto.isSelected());
    SpinnerValueFactory value = new SpinnerValueFactory.
        IntegerSpinnerValueFactory(2, 10000);
    this.sppinerNum.setValueFactory(value);
    this.gen = new GeneradorCadenas();
}

public void validar() throws InterruptedException, IOException
{
    if(this.maquinaT.iniciarMaquina()){
        this.txtValida.setText("cadena_valida");
        this.txtValida.setFill(Color.web("#24FC0D"));
    }else
    {
        this.txtValida.setFill(Color.web("#FF0000"));
        this.txtValida.setText("Cadena_invalida");
    }
}
}

```

```
}
```

## Código de la máquina de Turing

MaquinaTuring.java

```
package practica9.maquinaturing;

import java.io.IOException;
import javafx.scene.control.TextArea;
import javafx.scene.text.Text;

/**
 *
 * @author colin
 */
public class MaquinaTuring {
    private String cadena;
    private int pos;
    private TextArea area;
    private ArchivosRutas archivo;
    private String estado;
    private String letraG;

    private boolean isValida;

    public MaquinaTuring(String cadena, TextArea area) {
        this.cadena = cadena+"B";
        this.area = area;
        this.pos=0;
        this.isValida = false;
        this.estado = "q0";
        this.letraG = "\u03B4";
    }

    /**
     Algoritmo para la maquina de turing
     vamos terminando primero el estado ene l que estamos
     y dependiendo de cual es el caracter qye lee actuamos en
     los cambios y movimiento de la maquina de turing
    */
    public boolean iniciarMaquina() throws InterruptedException , IOException
    {
        this.archivo = new ArchivosRutas(" analisis");
        StringBuilder movimientos = new StringBuilder(this.cadena);
        boolean procesando=true;
        while(procesando)
        {
            movimientos = new StringBuilder(this.cadena);
            switch(this.estado)
            {
                case "q0":
                    switch(this.cadena.charAt(pos))
                    {
                        case '0':
                            // escribimos

```

```

        this.area.appendText(this.letraG+"(q0,0)->(q1,X,R)⌞\t⌞"
            +movimientos.replace(pos, pos, "("+this.estado+")")+
            "\n");
        this.archivo.escribirArchivo(this.letraG+"(q0,0)->(q1,X
            ,R)⌞\t⌞"+movimientos+"\n");
        // (q0,0)->(q1,X,R)
        this.estado = "q1";
        this.cadena = this.cadena.replaceFirst("0", "X");
        // mover a la derecha
        this.pos++;
        break;
    case 'Y':
        // escribimos
        this.area.appendText(this.letraG+"(q0,Y)->(q3,Y,R)⌞\t⌞"
            +movimientos.replace(pos, pos, "("+this.estado+")")+
            "\n");
        this.archivo.escribirArchivo(this.letraG+"(q0,Y)->(q3,Y
            ,R)⌞\t⌞"+movimientos+"\n");
        // (q0,Y) -> (q3,Y,R)
        this.estado = "q3";
        // el caracter se queda en Y
        // nos movemos a la derecha
        this.pos++;
        break;
    default:
        this.area.appendText(this.letraG+"("+this.estado+", "+
            this.cadena.charAt(pos)+")\t"+ movimientos.replace(
            pos, pos, "("+this.estado+")\n\n"));
        this.archivo.escribirArchivo(this.letraG+"("+this.
            estado+", "+this.cadena.charAt(pos)+")\t"+
            movimientos+")\n\n");
        // (q0,1) -> muere
        // (q0,X)-> muere
        // (q0,B) -> muere
        procesando = false;
        break;
    }
    break;
case "q1":
    switch(this.cadena.charAt(pos))
    {
        case '0':
            // escribimos
            this.area.appendText(this.letraG+"(q1,0)->(q1,0,R)⌞\t⌞"
                +movimientos.replace(pos, pos, "("+this.estado+")")+
                "\n");
            this.archivo.escribirArchivo(this.letraG+"(q1,0)->(q1
                ,0,R)⌞\t⌞"+movimientos+"\n");
            // (q1,0)->(q1,0,R)
            // el estado no cambia ni remplazamos caracteres
            // solo nos movemos a la derecha
            this.pos++;
            break;
        case '1':
            this.area.appendText(this.letraG+"(q1,1)->(q2,Y,L)⌞\t⌞"

```



```

        +movimientos.replace(pos, pos, "("+this.estado+")")+
        "\n");
    this.archivo.escribirArchivo(this.letraG+"(q1,1)->(q2,Y
    ,L)\t"+movimientos+"\n");
    // (q1,1) -> (q2,Y,L)
    // cambiamos estado a q2 reemplazamos 1 por Y
    // y nos movemos a la izq
    this.estado = "q2";
    this.cadena = this.cadena.replaceFirst("1", "Y");
    this.pos--;
    break;
case 'Y':
    this.area.appendText(this.letraG+"(q1,Y)->(q1,Y,R)\t"+
    +movimientos.replace(pos, pos, "("+this.estado+")")+
    "\n");
    this.archivo.escribirArchivo(this.letraG+"(q1,Y)->(q1,Y
    ,R)\t"+movimientos+"\n");
    // (q1,Y) -> (q1,Y,R)
    // nos quedamos en el mismo estado
    // no reemplazamos la Y solo nos movemos a la der
    this.pos++;
    break;
default:
    // (q1,X) -> Muere
    // (q1,B) -> Muere
    this.area.appendText(this.letraG+"("+this.estado+", "+
    this.cadena.charAt(pos)+")\t"+ movimientos.replace(
    pos, pos, "("+this.estado+")\n\n"));
    this.archivo.escribirArchivo(this.letraG+"("+this.
    estado+", "+this.cadena.charAt(pos)+")\t"+
    movimientos+"\n\n");
    procesando = false;
    break;
}
break;
case "q2":
    switch(this.cadena.charAt(pos))
    {
        case '0':
            this.area.appendText(this.letraG+"(q2,0)->(q2,0,L)\t"+
            +movimientos.replace(pos, pos, "("+this.estado+")")+
            "\n");
            this.archivo.escribirArchivo(this.letraG+"(q2,0)->(q2
            ,0,L)\t"+movimientos+"\n");
            // (q2,0)->(q2,0,L)
            // solo nos movemos a la izquierda
            this.pos--;
            break;
        case 'X':
            this.area.appendText(this.letraG+"(q2,X)->(q0,X,R)\t"+
            +movimientos.replace(pos, pos, "("+this.estado+")")+
            "\n");
            this.archivo.escribirArchivo(this.letraG+"(q2,X)->(q0,X
            ,R)\t"+movimientos+"\n");
            // (q2,X) -> (q0,X,R)

```

```

        // cambiamos estado a q0
        // nos movemos a la derecha
        this.estado = "q0";
        this.pos++;
        break;
    case 'Y':
        this.area.appendText(this.letraG+"(q2,Y)->(q2,Y,L)␣␣"
            +movimientos.replace(pos, pos, "("+this.estado+")")+
            "␣\n");
        this.archivo.escribirArchivo(this.letraG+"(q2,Y)->(q2,Y
            ,L)␣␣"+movimientos+"\n");
        // (q2,Y) -> (q2,Y,L)
        // solo nos vamos a mover a la izq
        this.pos--;

        break;
    default:
        this.area.appendText(this.letraG+"("+this.estado+", "+
            this.cadena.charAt(pos)+")"+ movimientos.replace(pos
            , pos, "("+this.estado+")\n\n"));
        this.archivo.escribirArchivo(this.letraG+"("+this.
            estado+", "+this.cadena.charAt(pos)+")"+ movimientos+
            "\n\n");
        // (q2,1) -> muere
        // (q2,B) -> muere
        procesando = false;
        break;
    }
    break;
case "q3":
    switch(this.cadena.charAt(pos))
    {
        case 'Y':
            this.area.appendText(this.letraG+"(q3,Y)->(q3,Y,R)␣␣"
                +movimientos.replace(pos, pos, "("+this.estado+")")+
                "␣\n");
            this.archivo.escribirArchivo(this.letraG+"(q3,Y)->(q3,Y
                ,R)␣␣"+movimientos+"␣\n");
            // (q3,Y) -> (q3,Y,R)
            // solo nos movemos a la derecha
            this.pos++;
            break;
        case 'B':
            this.area.appendText(this.letraG+"(q3,B)->(q4,B,R)␣␣"
                +movimientos.replace(pos, pos, "("+this.estado+")")+
                "␣\n");
            this.archivo.escribirArchivo(this.letraG+"(q3,B)->(q4,B
                ,R)␣␣"+movimientos+"␣\n");
            // (q3,B) -> (q4,B,R)
            // cambiamos estado. movemos a la derecha
            this.estado = "q4";
            this.pos++;
            break;
        default:
            // (q3,0) -> muere

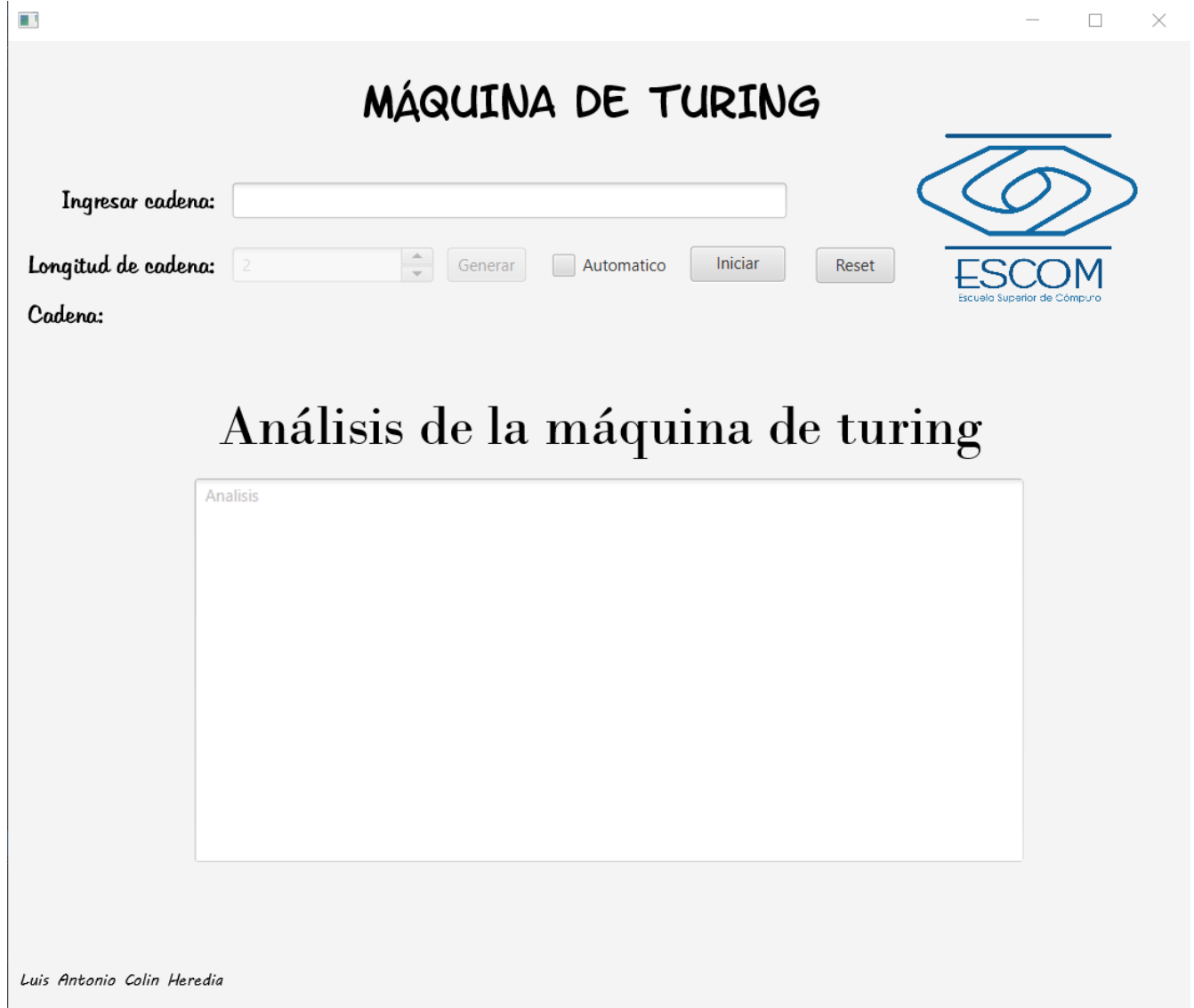
```

```

        // (q3,1) -> muere
        // (q3,X) -> muere
        this.area.appendText(this.letraG+"("+this.estado+", "+
            this.cadena.charAt(pos)+")\t" + movimientos.replace
            (pos, pos, "("+this.estado+")\n\n"));
        this.archivo.escribirArchivo(this.letraG+"("+this.
            estado+", "+this.cadena.charAt(pos)+")\t" +
            movimientos+"\n\n");
        procesando = false;
        break;
    }
    break;
case "q4":
    this.area.appendText(this.letraG+"("+this.estado+", "+this.
        cadena.charAt(pos-1)+")\t" + movimientos.replace(pos, pos,
            "("+this.estado+")\n\n"));
    this.archivo.escribirArchivo(this.letraG+"("+this.estado+", "+
        this.cadena.charAt(pos-1)+")\t" + movimientos+"\n\n");
    procesando = false;
    this.isValida = true;
default:
    break;
}
}
return this.isValida;
}
}

```

# Pruebas de la Maquina de Turing



**MÁQUINA DE TURING**

Ingresar cadena:

Longitud de cadena:   ☐ Automatico

Cadena:

**Análisis de la máquina de turing**

Analisis

*Luis Antonio Colin Heredia*

Ingresando la cadena

MÁQUINA DE TURING

Ingresar cadena: 00001111

Longitud de cadena: 2

Generar

☐ Automatico

Iniciar

Reset

Cadena:

ESCOM

Escuela Superior de Cómputo

Análisis de la máquina de turing

Analisis

Luis Antonio Colin Heredia

## Iniciando análisis manual

Se puede hacer scroll para ver toda la salida desde el programa. Aunque también esa salida se guardara en el archivo de texto.

# MÁQUINA DE TURING

Ingresar cadena:

00001111

Longitud de cadena:

2

Generar

☐ Automatico

Iniciar

Reset

Cadena:00001111



## Análisis de la máquina de turing

```
δ(q0,0)→(q1,X,R) (q0)00001111B
δ(q1,0)→(q1,0,R) X(q1)0001111B
δ(q1,0)→(q1,0,R) X0(q1)001111B
δ(q1,0)→(q1,0,R) X00(q1)01111B
δ(q1,1)→(q2,Y,L) X000(q1)1111B
δ(q2,0)→(q2,0,L) X00(q2)0Y111B
δ(q2,0)→(q2,0,L) X0(q2)00Y111B
δ(q2,0)→(q2,0,L) X(q2)000Y111B
δ(q2,X)→(q0,X,R) (q2)X000Y111B
δ(q0,0)→(q1,X,R) X(q0)000Y111B
δ(q1,0)→(q1,0,R) XX(q1)00Y111B
δ(q1,0)→(q1,0,R) XX0(q1)0Y111B
δ(q1,Y)→(q1,Y,R) XX00(q1)Y111B
δ(q1,1)→(q2,Y,L) XX00Y(q1)111B
δ(q2,Y)→(q2,Y,L) XX00(q2)YY11B
δ(q2,0)→(q2,0,L) XX0(q2)0YY11B
δ(q2,0)→(q2,0,L) XX0(q2)0YY11B
```

**cadena valida**

Luis Antonio Colin Heredia

## Salida en archivo de texto del modo manual

 analisis: Bloc de notas

Archivo Edición Formato Ver Ayuda

$\delta(q_0, 0) \rightarrow (q_1, X, R)$	$(q_0)00001111B$
$\delta(q_1, 0) \rightarrow (q_1, 0, R)$	$X(q_1)0001111B$
$\delta(q_1, 0) \rightarrow (q_1, 0, R)$	$X0(q_1)001111B$
$\delta(q_1, 0) \rightarrow (q_1, 0, R)$	$X00(q_1)01111B$
$\delta(q_1, 1) \rightarrow (q_2, Y, L)$	$X000(q_1)1111B$
$\delta(q_2, 0) \rightarrow (q_2, 0, L)$	$X00(q_2)0Y111B$
$\delta(q_2, 0) \rightarrow (q_2, 0, L)$	$X0(q_2)00Y111B$
$\delta(q_2, 0) \rightarrow (q_2, 0, L)$	$X(q_2)000Y111B$
$\delta(q_2, X) \rightarrow (q_0, X, R)$	$(q_2)X000Y111B$
$\delta(q_0, 0) \rightarrow (q_1, X, R)$	$X(q_0)000Y111B$
$\delta(q_1, 0) \rightarrow (q_1, 0, R)$	$XX(q_1)00Y111B$
$\delta(q_1, 0) \rightarrow (q_1, 0, R)$	$XX0(q_1)0Y111B$
$\delta(q_1, Y) \rightarrow (q_1, Y, R)$	$XX00(q_1)Y111B$
$\delta(q_1, 1) \rightarrow (q_2, Y, L)$	$XX00Y(q_1)111B$
$\delta(q_2, Y) \rightarrow (q_2, Y, L)$	$XX00(q_2)YY11B$
$\delta(q_2, 0) \rightarrow (q_2, 0, L)$	$XX0(q_2)0YY11B$
$\delta(q_2, 0) \rightarrow (q_2, 0, L)$	$XX(q_2)00YY11B$
$\delta(q_2, X) \rightarrow (q_0, X, R)$	$X(q_2)X00YY11B$
$\delta(q_0, 0) \rightarrow (q_1, X, R)$	$XX(q_0)00YY11B$
$\delta(q_1, 0) \rightarrow (q_1, 0, R)$	$XXX(q_1)0YY11B$
$\delta(q_1, Y) \rightarrow (q_1, Y, R)$	$XXX0(q_1)YY11B$
$\delta(q_1, Y) \rightarrow (q_1, Y, R)$	$XXX0Y(q_1)Y11B$
$\delta(q_1, 1) \rightarrow (q_2, Y, L)$	$XXX0YY(q_1)11B$
$\delta(q_2, Y) \rightarrow (q_2, Y, L)$	$XXX0Y(q_2)YY1B$
$\delta(q_2, Y) \rightarrow (q_2, Y, L)$	$XXX0(q_2)YYY1B$
$\delta(q_2, 0) \rightarrow (q_2, 0, L)$	$XXX(q_2)0YYY1B$
$\delta(q_2, X) \rightarrow (q_0, X, R)$	$XX(q_2)X0YYY1B$
$\delta(q_0, 0) \rightarrow (q_1, X, R)$	$XXX(q_0)0YYY1B$
$\delta(q_1, Y) \rightarrow (q_1, Y, R)$	$XXXX(q_1)YYY1B$
$\delta(q_1, Y) \rightarrow (q_1, Y, R)$	$XXXXY(q_1)YY1B$
$\delta(q_1, Y) \rightarrow (q_1, Y, R)$	$XXXXYY(q_1)Y1B$
$\delta(q_1, 1) \rightarrow (q_2, Y, L)$	$XXXXYYY(q_1)1B$
$\delta(q_2, Y) \rightarrow (q_2, Y, L)$	$XXXXYY(q_2)YYB$
$\delta(q_2, Y) \rightarrow (q_2, Y, L)$	$XXXXY(q_2)YYYB$
$\delta(q_2, Y) \rightarrow (q_2, Y, L)$	$XXXX(q_2)YYYYB$
$\delta(q_2, X) \rightarrow (q_0, X, R)$	$XXX(q_2)XYYYYB$
$\delta(q_0, Y) \rightarrow (q_3, Y, R)$	$XXXX(q_0)YYYYB$
$\delta(q_3, Y) \rightarrow (q_3, Y, R)$	$XXXXY(q_3)YYYB$
$\delta(q_3, Y) \rightarrow (q_3, Y, R)$	$XXXXYY(q_3)YYB$
$\delta(q_3, Y) \rightarrow (q_3, Y, R)$	$XXXXYYY(q_3)YB$
$\delta(q_3, B) \rightarrow (q_4, B, R)$	$XXXXYYYY(q_3)B$
$\delta(q_4, B)$	$XXXXYYYYB(q_4)$

## Modo automático activado

# MÁQUINA DE TURING

Ingresar cadena:

Longitud de cadena:   ☒ Automatico

Generamos una cadena de 9 caracteres.

Longitud de cadena:

Cadena:000001111



## Iniciando el análisis con modo automático

MÁQUINA DE TURING

Ingresar cadena: 00001111

Longitud de cadena: 9

Generar

☒ Automático

Iniciar

Reset

ESCOM  
Escuela Superior de Cómputo

Cadena:000001111

Análisis de la máquina de turing

$\delta(q_0, 0) \rightarrow (q_1, X, R)$	(q0)000001111B
$\delta(q_1, 0) \rightarrow (q_1, 0, R)$	X(q1)00001111B
$\delta(q_1, 0) \rightarrow (q_1, 0, R)$	X0(q1)0001111B
$\delta(q_1, 0) \rightarrow (q_1, 0, R)$	X00(q1)001111B
$\delta(q_1, 0) \rightarrow (q_1, 0, R)$	X000(q1)01111B
$\delta(q_1, 1) \rightarrow (q_2, Y, L)$	X0000(q1)1111B
$\delta(q_2, 0) \rightarrow (q_2, 0, L)$	X000(q2)0Y111B
$\delta(q_2, 0) \rightarrow (q_2, 0, L)$	X00(q2)00Y111B
$\delta(q_2, 0) \rightarrow (q_2, 0, L)$	X0(q2)000Y111B
$\delta(q_2, 0) \rightarrow (q_2, 0, L)$	X(q2)0000Y111B
$\delta(q_2, X) \rightarrow (q_0, X, R)$	(q2)X0000Y111B
$\delta(q_0, 0) \rightarrow (q_1, X, R)$	X(q0)0000Y111B
$\delta(q_1, 0) \rightarrow (q_1, 0, R)$	XX(q1)000Y111B
$\delta(q_1, 0) \rightarrow (q_1, 0, R)$	XX0(q1)00Y111B
$\delta(q_1, 0) \rightarrow (q_1, 0, R)$	XX00(q1)0Y111B
$\delta(q_1, Y) \rightarrow (q_1, Y, R)$	XX000(q1)Y111B
$\delta(q_1, 1) \rightarrow (q_2, Y, L)$	XX0000(q1)1111B

**Cadena invalida**

Luis Antonio Colin Heredia

## Salida del archivo cadena automática

Archivo	Edición	Formato	Ver	Ayuda
$\delta(q_0, \emptyset) \rightarrow (q_1, X, R)$			(q0)000001111B	
$\delta(q_1, \emptyset) \rightarrow (q_1, \emptyset, R)$			X(q1)00001111B	
$\delta(q_1, \emptyset) \rightarrow (q_1, \emptyset, R)$			X0(q1)0001111B	
$\delta(q_1, \emptyset) \rightarrow (q_1, \emptyset, R)$			X00(q1)001111B	
$\delta(q_1, \emptyset) \rightarrow (q_1, \emptyset, R)$			X000(q1)01111B	
$\delta(q_1, 1) \rightarrow (q_2, Y, L)$			X0000(q1)1111B	
$\delta(q_2, \emptyset) \rightarrow (q_2, \emptyset, L)$			X000(q2)0Y111B	
$\delta(q_2, \emptyset) \rightarrow (q_2, \emptyset, L)$			X00(q2)00Y111B	
$\delta(q_2, \emptyset) \rightarrow (q_2, \emptyset, L)$			X0(q2)000Y111B	
$\delta(q_2, \emptyset) \rightarrow (q_2, \emptyset, L)$			X(q2)0000Y111B	
$\delta(q_2, X) \rightarrow (q_0, X, R)$			(q2)X0000Y111B	
$\delta(q_0, \emptyset) \rightarrow (q_1, X, R)$			X(q0)0000Y111B	
$\delta(q_1, \emptyset) \rightarrow (q_1, \emptyset, R)$			XX(q1)000Y111B	
$\delta(q_1, \emptyset) \rightarrow (q_1, \emptyset, R)$			XX0(q1)00Y111B	
$\delta(q_1, \emptyset) \rightarrow (q_1, \emptyset, R)$			XX00(q1)0Y111B	
$\delta(q_1, Y) \rightarrow (q_1, Y, R)$			XX000(q1)Y111B	
$\delta(q_1, 1) \rightarrow (q_2, Y, L)$			XX000Y(q1)111B	
$\delta(q_2, Y) \rightarrow (q_2, Y, L)$			XX000(q2)YY11B	
$\delta(q_2, \emptyset) \rightarrow (q_2, \emptyset, L)$			XX00(q2)0YY11B	
$\delta(q_2, \emptyset) \rightarrow (q_2, \emptyset, L)$			XX0(q2)00YY11B	
$\delta(q_2, \emptyset) \rightarrow (q_2, \emptyset, L)$			XX(q2)000YY11B	
$\delta(q_2, X) \rightarrow (q_0, X, R)$			X(q2)X000YY11B	
$\delta(q_0, \emptyset) \rightarrow (q_1, X, R)$			XX(q0)000YY11B	
$\delta(q_1, \emptyset) \rightarrow (q_1, \emptyset, R)$			XXX(q1)00YY11B	
$\delta(q_1, \emptyset) \rightarrow (q_1, \emptyset, R)$			XXX0(q1)0YY11B	
$\delta(q_1, Y) \rightarrow (q_1, Y, R)$			XXX00(q1)YY11B	
$\delta(q_1, Y) \rightarrow (q_1, Y, R)$			XXX00Y(q1)Y11B	
$\delta(q_1, 1) \rightarrow (q_2, Y, L)$			XXX00YY(q1)11B	
$\delta(q_2, Y) \rightarrow (q_2, Y, L)$			XXX00Y(q2)YY1B	
$\delta(q_2, Y) \rightarrow (q_2, Y, L)$			XXX00(q2)YY1B	
$\delta(q_2, \emptyset) \rightarrow (q_2, \emptyset, L)$			XXX0(q2)0YYY1B	
$\delta(q_2, \emptyset) \rightarrow (q_2, \emptyset, L)$			XXX(q2)00YYY1B	
$\delta(q_2, X) \rightarrow (q_0, X, R)$			XX(q2)X00YYY1B	
$\delta(q_0, \emptyset) \rightarrow (q_1, X, R)$			XXX(q0)00YYY1B	
$\delta(q_1, \emptyset) \rightarrow (q_1, \emptyset, R)$			XXXX(q1)0YYY1B	
$\delta(q_1, Y) \rightarrow (q_1, Y, R)$			XXXX0(q1)YY1B	
$\delta(q_1, Y) \rightarrow (q_1, Y, R)$			XXXX0Y(q1)YY1B	
$\delta(q_1, Y) \rightarrow (q_1, Y, R)$			XXXX0YY(q1)Y1B	
$\delta(q_1, 1) \rightarrow (q_2, Y, L)$			XXXX0YYY(q1)1B	
$\delta(q_2, Y) \rightarrow (q_2, Y, L)$			XXXX0YY(q2)YYB	
$\delta(q_2, Y) \rightarrow (q_2, Y, L)$			XXXX0Y(q2)YYB	
$\delta(q_2, Y) \rightarrow (q_2, Y, L)$			XXXX0(q2)YYYB	
$\delta(q_2, \emptyset) \rightarrow (q_2, \emptyset, L)$			XXXX(q2)0YYYB	
$\delta(q_2, X) \rightarrow (q_0, X, R)$			XXX(q2)X0YYYB	
$\delta(q_0, \emptyset) \rightarrow (q_1, X, R)$			XXXX(q0)0YYYB	
$\delta(q_1, Y) \rightarrow (q_1, Y, R)$			XXXXX(q1)YYYB	
$\delta(q_1, Y) \rightarrow (q_1, Y, R)$			XXXXXY(q1)YYYB	
$\delta(q_1, Y) \rightarrow (q_1, Y, R)$			XXXXXY(q1)YYB	
$\delta(q_1, Y) \rightarrow (q_1, Y, R)$			XXXXXYYY(q1)YB	
$\delta(q_1, B) \rightarrow (q_1, Y, R)$			XXXXXYYY(q1)YB	
$\delta(q_1, B) \rightarrow (q_1, Y, R)$			XXXXXYYY(q1)YB	

## Conclusiones de las prácticas

En general estas practicas me parecieron más fáciles que las anteriores al momento de programar, pero me pareció un concepto mas interesante en las aplicaciones de estos autómatas. La practica de generar palíndromo se me había complicado un poco por que no entendía como era que siguiendo unas cuantas reglas de producción podíamos generar siempre cadenas validas para el lenguaje. Eso me voló la cabeza. La máquina de Turing fue una de mis favoritas, quisiera animarla y darme un poco mas de tiempo para su análisis y funcionamientos mas complejos. El autómata de pila me encantó programarla, fue muy sencillo pero me gustó la lógica que se debe aplicar para determinar si es valida o no. Me recordó a una de mis practicas de estructura de datos donde teníamos que ingresar una ecuación con variables A,B,C,..., esa ecuación debíamos validar si era una ecuación correcta y darles valores a esas variables para después convertirla de Infija a Postfija y evaluar la expresión.

## Conclusión de la materia

Sinceramente esta materia fue mucho mejor de lo que esperaba. Este semestre vengo de problemas de salud y de una baja temporal. El tomar esta materia me ayudo a darme cuenta que de verdad esta carrera me encanta. La programación, los conceptos , aplicaciones en realidad todo en general me pareció súper interesante. Yo creí que seria una materia equis de programación, donde normalmente los programas que nos dejan son muy cuadrados o demasiado sencillos. Las prácticas que desarrollamos en esta materia desde la primera ya tenían un análisis muy profundo en ciencia de datos. A mi parecer la mas dificil fue la del Chess. De verdad me costó mucho encontrar una manera de generar todos los caminos posibles. Pero al final al terminarla con todo y animaciones me gustó muchísimo; quisiera buscar como implementar la inteligencia artificial y ver que más puedo agregarle a esa practica. Creo que usted fue un gran profesor y es de los mejores que he tenido en mi trayectoria escolar y a pesar de esta contingencia dio su clase de una manera fantástica. Saludos profesor y gracias por la materia.