

Instituto Politécnico Nacional

Escuela Superior de Cómputo

Teoría Computacional

Reporte segundo bloque de programas

Alumno: Javier Said Naranjo Miranda

Grupo: 2CM4

Índice

1. Automata Web-Ebay	2
1.1. Descripción del programa	2
1.2. Código	2
1.3. Pruebas	8
2. Balanceo de Paréntesis	11
2.1. Descripción del programa	11
2.2. Código	11
2.3. Pruebas	15
3. Palindromos-Gramática libre de contexto	17
3.1. Descripción del programa	17
3.2. Código	17
3.3. Pruebas	21
4. Autómata de Pila	24
4.1. Descripción del programa	24
4.2. Código	24
4.3. Pruebas	32
5. Maquina de Turing	35
5.1. Descripción del programa	35
5.2. Código	35
5.3. Pruebas	40

1. Automata Web-Ebay

1.1. Descripción del programa

El siguiente programa reconoce todas las palabras que contengan web o ebay, esto se realiza por medio de un automata finito determinista, podrá reconocer las palabras solo en textos en inglés.

Consta de un modo automático y manual, además se podrá visualizar el diagrama del autómata.

El programa guarda todas las palabras que encontró y las guarda en un archivo de texto indicando el número de fila y palabra en la que se encuentra.

1.2. Código

El código utilizado para la resolución del problema se muestra a continuación:

Código: webay.py

```
def automataWebay(caracter, estado, archivo):
    if(estado==0):
        estado=estadoCero(caracter, archivo)
    elif(estado==1):
        estado=estadoUno(caracter, archivo)
    elif(estado==2):
        estado=estadoDos(caracter, archivo)
    elif(estado==3):
        estado=estadoTres(caracter, archivo)
    elif(estado==4):
        estado=estadoCuatro(caracter, archivo)
    elif(estado==5):
        estado=estadoCinco(caracter, archivo)
    elif(estado==6):
        estado=estadoSeis(caracter, archivo)
    elif(estado==7):
        estado=estadoSiete(caracter, archivo)

    return estado

def estadoCero(caracter, archivo):
    if(caracter=='w'):
        archivo.write('q0—w—>q1\t')
        return 1
    elif(caracter=='e'):
        archivo.write('q0—e—>q4\t')
        return 4
    else:
        archivo.write('q0— %s—>q0\t' %caracter)
        return 0
```

```

def estadoUno(caracter, archivo):
    if(caracter=='w'):
        archivo.write('q1—w—>q1\t')
        return 1
    elif(caracter=='e'):
        archivo.write('q1—e—>q2\t')
        return 2
    else:
        archivo.write('q1— %s—>q0\t' % caracter)
        return 0

def estadoDos(caracter, archivo):
    if(caracter=='w'):
        archivo.write('q2—w—>q1\t')
        return 1
    elif(caracter=='e'):
        archivo.write('q2—e—>q4\t')
        return 4
    elif(caracter=='b'):
        archivo.write('q2—b—>q3\t')
        return 3
    else:
        archivo.write('q2— %s—>q0\t' % caracter)
        return 0

def estadoTres(caracter, archivo):
    if(caracter=='w'):
        archivo.write('q3—w—>q1\t')
        return 1
    elif(caracter=='e'):
        archivo.write('q3—e—>q4\t')
        return 4
    elif(caracter=='a'):
        archivo.write('q3—a—>q6\t')
        return 6
    else:
        archivo.write('q3— %s—>q0\t' % caracter)
        return 0

def estadoCuatro(caracter, archivo):
    if(caracter=='w'):
        archivo.write('q4—2—>q1\t')
        return 1
    elif(caracter=='e'):
        archivo.write('q4—e—>q4\t')
        return 4
    elif(caracter=='b'):
        archivo.write('q4—b—>q5\t')
        return 5
    else:
        archivo.write('q4— %s—>q0\t' % caracter)

```



```

        Leer_texto\n3.–Diagrama\n4.–Salir\nElija_una_opcion:_')
    opcion=int(opcion)
except:
    print('\nIntroduzca_una_opcion_correcta\n')

return opcion

def IniciarArchivo():
    archivo=open("Palabras.txt","w")
    archivo.close
    archivoH=open("Historia.txt","w")
    archivo.close

def AbrirArchivo():
    try:
        archivo=open("Palabras.txt","a")
    except:
        print("\nError_al_abrir_el_archivo")
        exit()
    return archivo

def AbrirHistoria():
    try:
        archivo=open("Historia.txt","a")
    except:
        print("\nError_al_abrir_el_archivo")
        exit()
    return archivo

def escribir(archivo,palabra_aux,no_palabra,no_fila):
    archivo.write("Palabra:_"+palabra_aux+"_Numero_de_palabra:_"+str(
        no_palabra)+"_Numero_de_fila:_"+str(no_fila))
    archivo.write("\n")

def Evaluar(texto):
    archivo=AbrirArchivo()
    historia=AbrirHistoria()
    palabra_aux=''
    final=False
    estado=0
    no_palabra=1
    no_fila=1

    historia.write("\n\n\n\n")
    for caracter in texto:
        caracter=caracter.lower()
        estado=webay.automataWebay(caracter,estado,historia)
        if(estado==3 or estado==7):
            final=True
        if(caracter=='\n'):
            if(final):

```

```

        escribir (archivo , palabra_aux , no_palabra , no_fila)
        historia . write ( "\n" )
        palabra_aux = ''
        final = False
    else :
        palabra_aux = ''
        no_palabra = 1
        no_fila = no_fila + 1
    continue
if (caracter == '_') :
    if (final) :
        escribir (archivo , palabra_aux , no_palabra , no_fila)
        historia . write ( "\n" )
        palabra_aux = ''
        final = False
    else :
        palabra_aux = ''
        no_palabra = no_palabra + 1

    continue
palabra_aux = palabra_aux + caracter
if (final) :
    no_palabra = no_palabra + 1
    escribir (archivo , palabra_aux , no_palabra , no_fila)
    historia . write ( "\n" )

archivo . close
historia . close
def leer_Archivo () :
    try :
        archivo = open ( " archivo . txt " , " r " )
        texto = str ( archivo . read () )
    except :
        print ( "\nError_al_abrir_el_archivo" )
        exit ()
    return texto

def main () :
    IniciarArchivo ()
    while True :
        eleccion = menu ()
        if (eleccion == 1) :
            texto = input ( " Introduzca_un_pequenio_texto : _ " )

            Evaluar ( texto )
            print ( " Evaluacion_terminada , _cheque_el_archivo_de_texto " )
            while True :
                reop = input ( " Desea_regresar_al_menu\n1. - Si\n2. - No\nEleccion : _ " )
                if (reop == '1') :
                    break
                elif (reop == '2') :
                    exit ()

```

```

        else:
            continue
    elif (eleccion==2):
        texto=leer_Archivo()

        Evaluar(texto)
        print("Evaluacion_terminada,_cheque_el_archivo_de_texto")
        while True:
            reop=input("Desea_regresar_al_menu\n1.-Si\n2.-No\nEleccion:_")
            if (reop=='1'):
                break
            elif (reop=='2'):
                exit()
            else:
                continue
    elif (eleccion==3):
        diagrama.mostrarDiagrama()
        while True:
            reop=input("Desea_regresar_al_menu\n1.-Si\n2.-No\nEleccion:_")
            if (reop=='1'):
                break
            elif (reop=='2'):
                exit()
            else:
                continue
    elif (eleccion==4):
        exit()
    else:
        continue
main()

```

Código:diagrama.py

```

from tkinter import *

def mostrarDiagrama():
    ventana=Tk()
    ventana.geometry("900x650")
    ventana.resizable(width=False,height=False)
    ventana.title("Automata_WEBAY")
    AreaDibujo=Canvas(ventana,width=900,height=650)
    AreaDibujo.pack()

    #Creacion de circulos de estados
    #inicio
    AreaDibujo.create_oval(50,275,150,375,fill="sky_blue")

    #web
    AreaDibujo.create_oval(200,150,300,250,fill="sky_blue")

```



```

AreaDibujo.create_oval(400,150,500,250,fill="sky_blue")
AreaDibujo.create_oval(590,140,710,260,fill="white")
AreaDibujo.create_oval(600,150,700,250,fill="sky_blue")

#ebay
AreaDibujo.create_oval(200,400,300,500,fill="sky_blue")
AreaDibujo.create_oval(400,400,500,500,fill="sky_blue")
AreaDibujo.create_oval(600,400,700,500,fill="sky_blue")
AreaDibujo.create_oval(770,390,890,510,fill="white")
AreaDibujo.create_oval(780,400,880,500,fill="sky_blue")

#creacion de lineas de transicion
#inicio
AreaDibujo.create_line(20,325,50,325)
#web
AreaDibujo.create_line(149,318,203,210)
AreaDibujo.create_line(300,200,400,200)
AreaDibujo.create_line(500,200,590,200)

AreaDibujo.create_line(650,260,650,400)
AreaDibujo.create_line(296,180,404,180)
AreaDibujo.create_line(407,230,253,402)
AreaDibujo.create_line(600,230,269,408)
#ebay
AreaDibujo.create_line(149,332,203,440)
AreaDibujo.create_line(300,450,400,450)
AreaDibujo.create_line(500,450,600,450)
AreaDibujo.create_line(700,450,770,450)

AreaDibujo.create_line(250,400,250,250)
AreaDibujo.create_line(405,424,265,247)
AreaDibujo.create_line(298,430,403,430)
AreaDibujo.create_line(618,410,280,240)
AreaDibujo.create_line(787,405,294,225)

#creacion de arcos de transicion
#inicio
AreaDibujo.create_arc(60,240,110,305,start=-8,extent=219,style=ARC)
#web
AreaDibujo.create_arc(210,125,260,185,start=2,extent=210,style=ARC)
AreaDibujo.create_arc(110,190,400,450,start=112,extent=50,style=ARC)
AreaDibujo.create_arc(150,100,750,650,start=55,extent=72,style=ARC)
AreaDibujo.create_arc(20,55,720,395,start=26,extent=181,style=ARC)
AreaDibujo.create_arc(78,85,480,500,start=41,extent=137,style=ARC)

#ebay
AreaDibujo.create_arc(110,190,400,460,start=200,extent=48,style=ARC)
AreaDibujo.create_arc(20,240,880,600,start=157,extent=173,style=ARC)
AreaDibujo.create_arc(200,385,890,550,start=45,extent=95,style=ARC)
AreaDibujo.create_arc(70,85,770,571,start=189,extent=128,style=ARC)
AreaDibujo.create_arc(90,85,560,520,start=199,extent=98,style=ARC)
AreaDibujo.create_arc(250,390,625,550,start=200,extent=146,style=ARC)

```

```
AreaDibujo.create_arc(208,442,285,530,start=172,extent=185,style=ARC)
```

```
#creacion de circulos de indicacion
```

```
#inicio
```

```
AreaDibujo.create_oval(47,322,53,328,fill="black")
```

```
#web
```

```
AreaDibujo.create_oval(52,296,58,302,fill="black")
```

```
AreaDibujo.create_oval(61,285,67,291,fill="black")
```

```
AreaDibujo.create_oval(76,278,82,284,fill="black")
```

```
AreaDibujo.create_oval(115,275,121,281,fill="black")
```

```
AreaDibujo.create_oval(198,209,204,215,fill="black")
```

```
AreaDibujo.create_oval(397,197,403,203,fill="black")
```

```
AreaDibujo.create_oval(587,197,593,203,fill="black")
```

```
AreaDibujo.create_oval(210,166,216,172,fill="black")
```

```
AreaDibujo.create_oval(293,177,299,183,fill="black")
```

```
AreaDibujo.create_oval(269,152,275,158,fill="black")
```

```
AreaDibujo.create_oval(647,397,653,403,fill="black")
```

```
AreaDibujo.create_oval(253,397,259,403,fill="black")
```

```
AreaDibujo.create_oval(268,403,274,409,fill="black")
```

```
#ebay
```

```
AreaDibujo.create_oval(53,346,59,352,fill="black")
```

```
AreaDibujo.create_oval(72,365,78,371,fill="black")
```

```
AreaDibujo.create_oval(102,372,108,378,fill="black")
```

```
AreaDibujo.create_oval(117,368,123,374,fill="black")
```

```
AreaDibujo.create_oval(199,436,205,442,fill="black")
```

```
AreaDibujo.create_oval(206,476,212,482,fill="black")
```

```
AreaDibujo.create_oval(297,447,303,453,fill="black")
```

```
AreaDibujo.create_oval(259,495,265,501,fill="black")
```

```
AreaDibujo.create_oval(278,411,284,417,fill="black")
```

```
AreaDibujo.create_oval(247,247,253,253,fill="black")
```

```
AreaDibujo.create_oval(399,427,405,433,fill="black")
```

```
AreaDibujo.create_oval(263,245,269,251,fill="black")
```

```
AreaDibujo.create_oval(597,447,603,453,fill="black")
```

```
AreaDibujo.create_oval(276,238,282,244,fill="black")
```

```
AreaDibujo.create_oval(768,447,774,453,fill="black")
```

```
AreaDibujo.create_oval(292,223,298,229,fill="black")
```

```
#etiquetas
```

```
#inicio
```

```
inicio=Label(ventana, text=" inicio ", font="Verdana_6") . place (x=20,y=305)
```

```
#estados
```

```
q0=Label(ventana, text="q0", font="Verdana_10", background="sky_blue") . place (
    x=90,y=317)
```

```

q1=Label(ventana , text="q1" , font="Verdana_10" , background="sky_blue" ) . place (
    x=240,y=190)
q2=Label(ventana , text="q2" , font="Verdana_10" , background="sky_blue" ) . place (
    x=440,y=190)
q3=Label(ventana , text="q3" , font="Verdana_10" , background="sky_blue" ) . place (
    x=640,y=190)
q4=Label(ventana , text="q4" , font="Verdana_10" , background="sky_blue" ) . place (
    x=240,y=440)
q5=Label(ventana , text="q5" , font="Verdana_10" , background="sky_blue" ) . place (
    x=440,y=440)
q6=Label(ventana , text="q6" , font="Verdana_10" , background="sky_blue" ) . place (
    x=640,y=440)
q7=Label(ventana , text="q7" , font="Verdana_10" , background="sky_blue" ) . place (
    x=820,y=440)
#web
w1=Label(ventana , text="w" , font="Verdana_7" ) . place (x=165,y=285)
w2=Label(ventana , text="no_es_w,e" , font="Verdana_7" ) . place (x=125,y=212)
w3=Label(ventana , text="e" , font="Verdana_7" ) . place (x=165,y=345)
w4=Label(ventana , text="w" , font="Verdana_7" ) . place (x=390,y=160)
w5=Label(ventana , text="e" , font="Verdana_7" ) . place (x=310,y=190)
w6=Label(ventana , text="w" , font="Verdana_7" ) . place (x=220,y=120)
w7=Label(ventana , text="w" , font="Verdana_7" ) . place (x=455,y=105)

w8=Label(ventana , text="b" , font="Verdana_7" ) . place (x=510,y=190)
w9=Label(ventana , text="no_es_w,e,a" , font="Verdana_7" ) . place (x=455,y=40)
w10=Label(ventana , text="no_es_w,e,b" , font="Verdana_7" ) . place (x=245,y=95)
w11=Label(ventana , text="e" , font="Verdana_7" ) . place (x=390,y=215)
w12=Label(ventana , text="e" , font="Verdana_7" ) . place (x=580,y=215)
w13=Label(ventana , text="a" , font="Verdana_7" ) . place (x=660,y=270)
#ebay
e1=Label(ventana , text="no_es_w,e,b" , font="Verdana_6" ) . place (x=120,y=385)
e2=Label(ventana , text="e" , font="Verdana_7" ) . place (x=263,y=525)
e3=Label(ventana , text="b" , font="Verdana_7" ) . place (x=310,y=412)
e4=Label(ventana , text="e" , font="Verdana_7" ) . place (x=380,y=455)
e5=Label(ventana , text="a" , font="Verdana_7" ) . place (x=510,y=430)
e6=Label(ventana , text="y" , font="Verdana_7" ) . place (x=710,y=430)

e7=Label(ventana , text="e" , font="Verdana_7" ) . place (x=710,y=400)
e8=Label(ventana , text="e" , font="Verdana_7" ) . place (x=580,y=476)
e9=Label(ventana , text="w" , font="Verdana_7" ) . place (x=710,y=355)
e10=Label(ventana , text="w" , font="Verdana_7" ) . place (x=550,y=355)
e11=Label(ventana , text="w" , font="Verdana_7" ) . place (x=370,y=360)
e12=Label(ventana , text="w" , font="Verdana_7" ) . place (x=230,y=370)

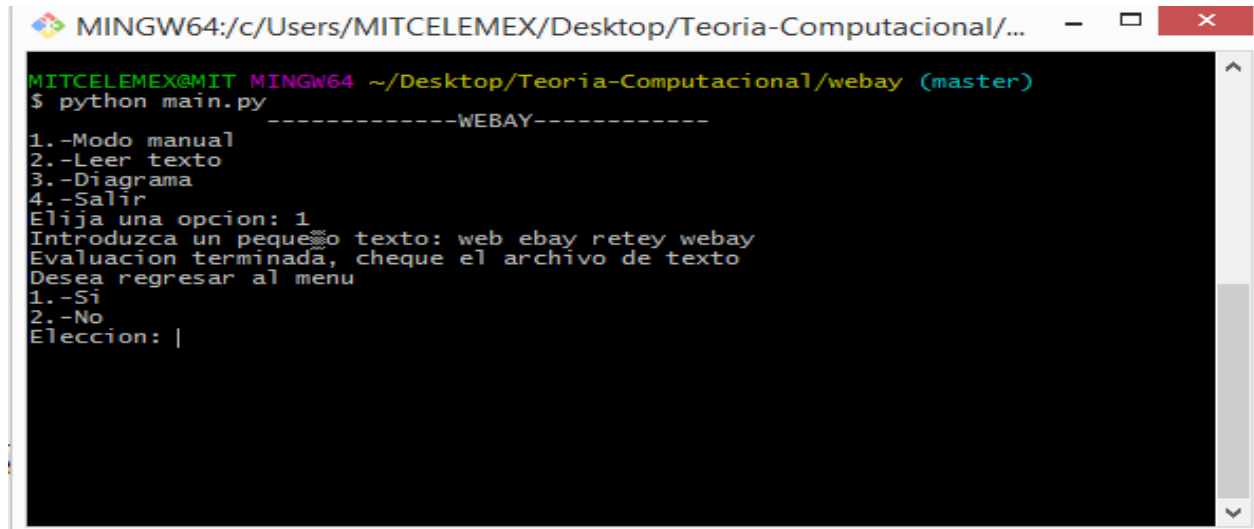
e13=Label(ventana , text="no_es_w,e" , font="Verdana_7" ) . place (x=450,y=605)
e14=Label(ventana , text="no_es_w,e,a" , font="Verdana_7" ) . place (x=425,y=505)
e15=Label(ventana , text="no_es_e,y,w" , font="Verdana_7" ) . place (x=470,y=570)
e16=Label(ventana , text="no_es_w,e" , font="Verdana_6" ) . place (x=35,y=220)

```

```
ventana.mainloop()
```

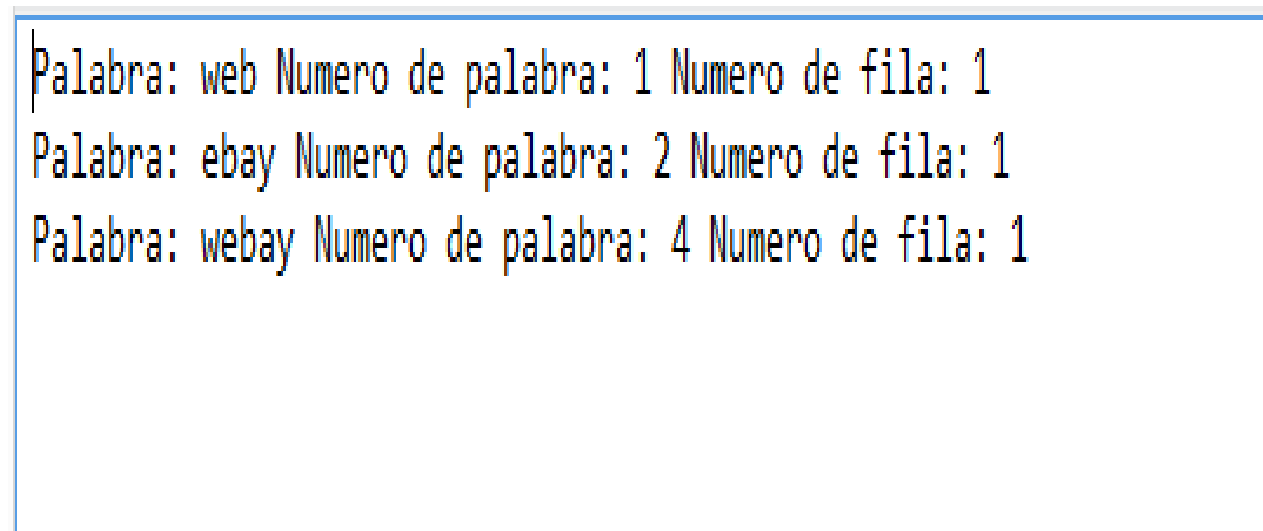
1.3. Pruebas

A continuación se mostraran algunas imágenes capturadas al momento de ejecutar el programa, dichas imágenes mostraran los resultados obtenidos.
Para el modo manual:



```
MINGW64:/c/Users/MITCELEMEX/Desktop/Teoria-Computacional/...
MITCELEMEX@MIT MINGW64 ~/Desktop/Teoria-Computacional/webay (master)
$ python main.py
-----WEBAY-----
1.-Modo manual
2.-Leer texto
3.-Diagrama
4.-Salir
Elija una opcion: 1
Introduzca un pequeño texto: web ebay retey webay
Evaluacion terminada, cheque el archivo de texto
Desea regresar al menu
1.-Si
2.-No
Eleccion: |
```

Figura 1: Palabras de prueba: web ebay retey webay



```
Palabra: web Numero de palabra: 1 Numero de fila: 1
Palabra: ebay Numero de palabra: 2 Numero de fila: 1
Palabra: webay Numero de palabra: 4 Numero de fila: 1
```

Figura 2: Salida del archivo de palabras encontradas

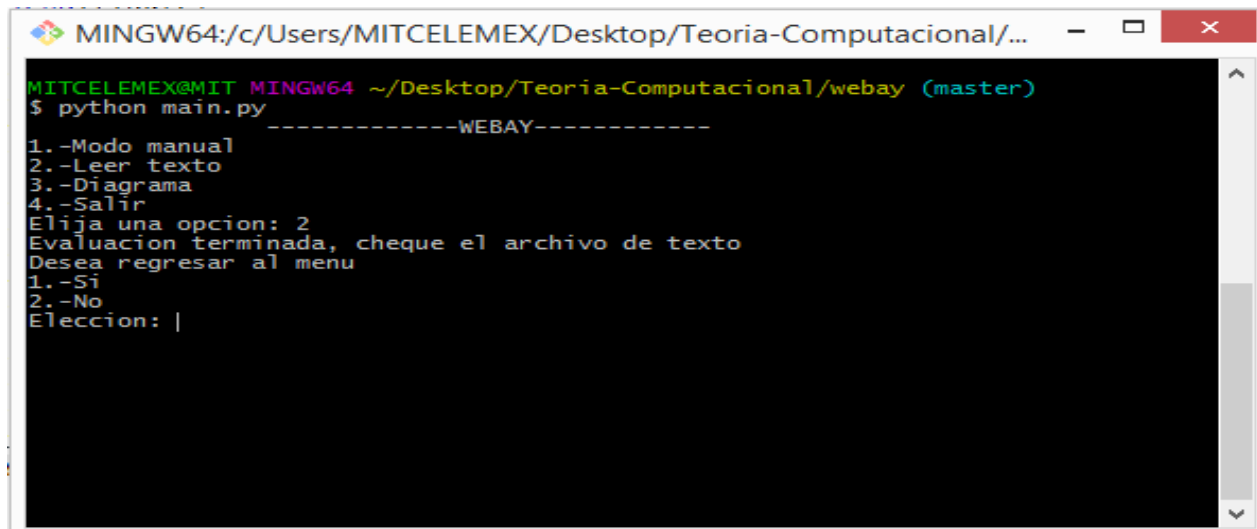
```

q0--w-->q1    q1--e-->q2    q2--b-->q3    q3-- -->q0
q0--e-->q4    q4--b-->q5    q5--a-->q6    q6--y-->q7    q7-- -->q0
q0--r-->q0    q0--e-->q4    q4--t-->q0    q0--e-->q4    q4--y-->q0    q0-- -->q0
q0--w-->q1    q1--e-->q2    q2--b-->q3    q3--a-->q6    q6--y-->q7

```

Figura 3: Historia de la evaluación del autómata

Para el modo de lectura de un archivo:



```

MINGW64:/c/Users/MITCELEMEX/Desktop/Teoria-Computacional/...
MITCELEMEX@MIT MINGW64 ~/Desktop/Teoria-Computacional/webay (master)
$ python main.py
-----WEBAY-----
1.-Modo manual
2.-Leer texto
3.-Diagrama
4.-Salir
Elija una opcion: 2
Evaluacion terminada, cheque el archivo de texto
Desea regresar al menu
1.-Si
2.-No
Eleccion: |

```

Figura 4: Lectura de un texto con palabras WEB

```

Palabra: web Numero de palabra: 4 Numero de fila: 1
Palabra: web) Numero de palabra: 9 Numero de fila: 1
Palabra: web Numero de palabra: 18 Numero de fila: 1
Palabra: web Numero de palabra: 46 Numero de fila: 1
Palabra: web Numero de palabra: 5 Numero de fila: 2
Palabra: web Numero de palabra: 4 Numero de fila: 4
Palabra: web Numero de palabra: 29 Numero de fila: 4
Palabra: web Numero de palabra: 48 Numero de fila: 4
Palabra: web Numero de palabra: 64 Numero de fila: 4
Palabra: web Numero de palabra: 79 Numero de fila: 4
Palabra: web Numero de palabra: 2 Numero de fila: 5
Palabra: website. Numero de palabra: 17 Numero de fila: 5
Palabra: website Numero de palabra: 18 Numero de fila: 5
Palabra: websites Numero de palabra: 43 Numero de fila: 5
Palabra: web Numero de palabra: 78 Numero de fila: 5
Palabra: web-like Numero de palabra: 71 Numero de fila: 7
Palabra: "web" Numero de palabra: 109 Numero de fila: 7
Palabra: web Numero de palabra: 91 Numero de fila: 9
Palabra: web Numero de palabra: 99 Numero de fila: 9
Palabra: worldwideweb, Numero de palabra: 101 Numero de fila: 9
Palabra: web:[12] Numero de palabra: 117 Numero de fila: 9
Palabra: web Numero de palabra: 120 Numero de fila: 9

```

Figura 5: Palabras encontradas del texto

q0--t-->q0	q0--h-->q0	q0--e-->q4	q4-- -->q0	q0--w-->q1	q1--o-->q0	q0--r-->q0	q0--l-->q0
q0--d-->q0	q0-- -->q0	q0--w-->q1	q1--i-->q0	q0--d-->q0	q0--e-->q4	q4-- -->q0	q0--w-->q1
q1--e-->q2	q2--b-->q3	q3-- -->q0					
q0--(-->q0	q0--a-->q0	q0--b-->q0	q0--b-->q0	q0--r-->q0	q0--e-->q4	q4--v-->q0	q0--i-->q0
q0--a-->q0	q0--t-->q0	q0--e-->q4	q4--d-->q0	q0-- -->q0	q0--w-->q1	q1--w-->q1	q1--w-->q1
q1-- -->q0	q0--o-->q0	q0--r-->q0	q0-- -->q0	q0--t-->q0	q0--h-->q0	q0--e-->q4	q4-- -->q0
q0--w-->q1	q1--e-->q2	q2--b-->q3	q3--)-->q0	q0-- -->q0			
q0--i-->q0	q0--s-->q0	q0-- -->q0	q0--a-->q0	q0--n-->q0	q0-- -->q0	q0--i-->q0	q0--n-->q0
q0--f-->q0	q0--o-->q0	q0--r-->q0	q0--m-->q0	q0--a-->q0	q0--t-->q0	q0--i-->q0	q0--o-->q0
q0--n-->q0	q0-- -->q0	q0--s-->q0	q0--p-->q0	q0--a-->q0	q0--c-->q0	q0--e-->q4	q4-- -->q0
q0--w-->q1	q1--h-->q0	q0--e-->q4	q4--r-->q0	q0--e-->q4	q4-- -->q0	q0--d-->q0	q0--o-->q0
q0--c-->q0	q0--u-->q0	q0--m-->q0	q0--e-->q4	q4--n-->q0	q0--t-->q0	q0--s-->q0	q0-- -->q0
q0--a-->q0	q0--n-->q0	q0--d-->q0	q0-- -->q0	q0--o-->q0	q0--t-->q0	q0--h-->q0	q0--e-->q4
q4--r-->q0	q0-- -->q0	q0--w-->q1	q1--e-->q2	q2--b-->q3	q3-- -->q0		
q0--r-->q0	q0--e-->q4	q4--s-->q0	q0--o-->q0	q0--u-->q0	q0--r-->q0	q0--c-->q0	q0--e-->q4
q4--s-->q0	q0-- -->q0	q0--a-->q0	q0--r-->q0	q0--e-->q4	q4-- -->q0	q0--i-->q0	q0--d-->q0
q0--e-->q4	q4--n-->q0	q0--t-->q0	q0--i-->q0	q0--f-->q0	q0--i-->q0	q0--e-->q4	q4--d-->q0
q0-- -->q0	q0--b-->q0	q0--y-->q0	q0-- -->q0	q0--u-->q0	q0--n-->q0	q0--i-->q0	q0--f-->q0
q0--o-->q0	q0--r-->q0	q0--m-->q0	q0-- -->q0	q0--r-->q0	q0--e-->q4	q4--s-->q0	q0--o-->q0
q0--u-->q0	q0--r-->q0	q0--c-->q0	q0--e-->q4	q4-- -->q0	q0--l-->q0	q0--o-->q0	q0--c-->q0
q0--a-->q0	q0--t-->q0	q0--o-->q0	q0--r-->q0	q0--s-->q0	q0-- -->q0	q0--(-->q0	q0--u-->q0

Figura 6: Historia de la evaluación del autómata

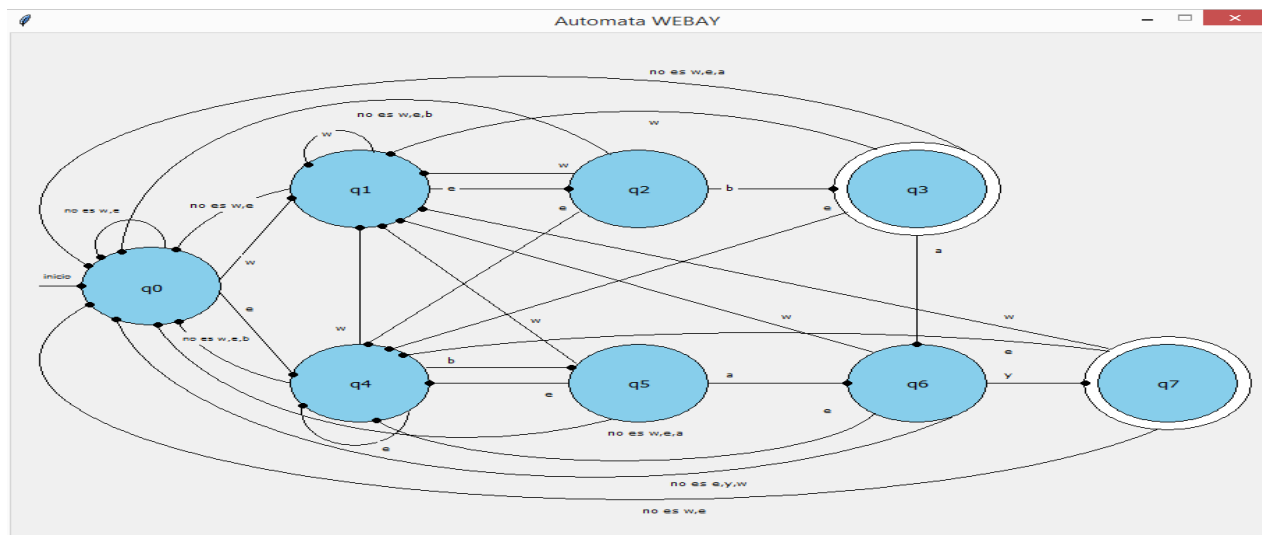


Figura 7: Diagrama del autómata

2. Balanceo de Paréntesis

2.1. Descripción del programa

El siguiente programa verificar que los paréntesis de una cadena esten balanceados, el programa tiene un modo manual, la entrada de este modo acepta todo tipo de caracteres ASCII. De igual forma cuenta con un modo automático con el cual se genera una cadena de paréntesis aleatoria con una longitud que puede estar entre 0 y 1000.

En un archivo se muestra la historia que se siguió al momento de hacer la evaluación de los paréntesis, así como las reglas que se utilizaron al momento de evaluar cada carácter.

2.2. Código

El código utilizado para la resolución del problema se muestra a continuación:

Código:balanceo.py

```
def VerificarBalanceo(cadena, archivo):
    exp= 'B'
    i=-1;
    cadena_aux=''
    archivo.write(exp)
    while True:
        try:
            i=i+1
            archivo.write('\n')
            if(cadena[i]=='('):
                cadena_aux=cadena_aux+'('
                if(exp[0]=='B'):
                    exp=exp.replace('B', 'RB', 1)
                    cadena_aux=cadena_aux.replace('B', 'RB', 1)
                    archivo.write(cadena_aux+exp)
                    archivo.write('\t\t\tB->(RB')
                    continue
                if(exp[0]=='R'):
                    exp=exp.replace('R', 'RR', 1)
                    cadena_aux=cadena_aux.replace('R', 'RR', 1)
                    archivo.write(cadena_aux+exp)
                    archivo.write('\t\t\tR->(RR')
                    continue
            if(cadena[i]==')'):
                if(exp[0]=='R'):
                    exp=exp[1:]
```

```

        cadena_aux=cadena_aux+' '
        archivo.write(cadena_aux+exp)
        archivo.write('\t\t\tR->_')
        continue
    elif(exp[0]== 'B') :
        exp= ''
        print('Cadena_no_balanceada')
        break
    else :
        exp= ''
        continue
except:
    if(exp=='B') :
        archivo.write(cadena_aux)
        archivo.write('\t\t\tB->_e_\n')
        print('Cadena_balanceada')
    else :
        print('Cadena_no_balanceada')
    break

```

Código:main.py

```

import random
import balanceo

def IniciarArchivo():
    archivo=open("Gramatica.txt","w")
    archivo.close

def Menu():
    print("-----Menu-----")
    print("1.-Modo_Manual")
    print("2.-Modo_Automatico")
    print("3.-Salir")

def Eleccion():
    op=input("Elige_una_opcion:_")
    try:
        op=int(op)
    except:
        print("Introduzca_una_opcion_valida")
    return op

def longitud():
    lon=random.randint(1,100)
    return lon

def generarCadena():
    lon=longitud()
    cadena= ''
    for c in range(1,lon+1):
        o=random.randint(1,2)

```

```

        if(o==1):
            cadena=cadena+'('
        else:
            cadena=cadena+')'
    return cadena

def Manual():
    try:
        archivo=open("Gramatica.txt","a")
    except:
        print("Error_al_abrir_el_archivo")
        exit()

    cadena=input("Introduce una cadena de parentesis: ")
    balanceo.VerificarBalanceo(cadena, archivo)

def Automatico():
    try:
        archivo=open("Gramatica.txt","a")
    except:
        print("Error_al_abrir_el_archivo")
        exit()
    cadena=generarCadena()
    print(cadena)
    balanceo.VerificarBalanceo(cadena, archivo)

def VerificarDeNuevo():
    opcion=input("Desea ingresar una nueva cadena [s/n]: ")
    return opcion

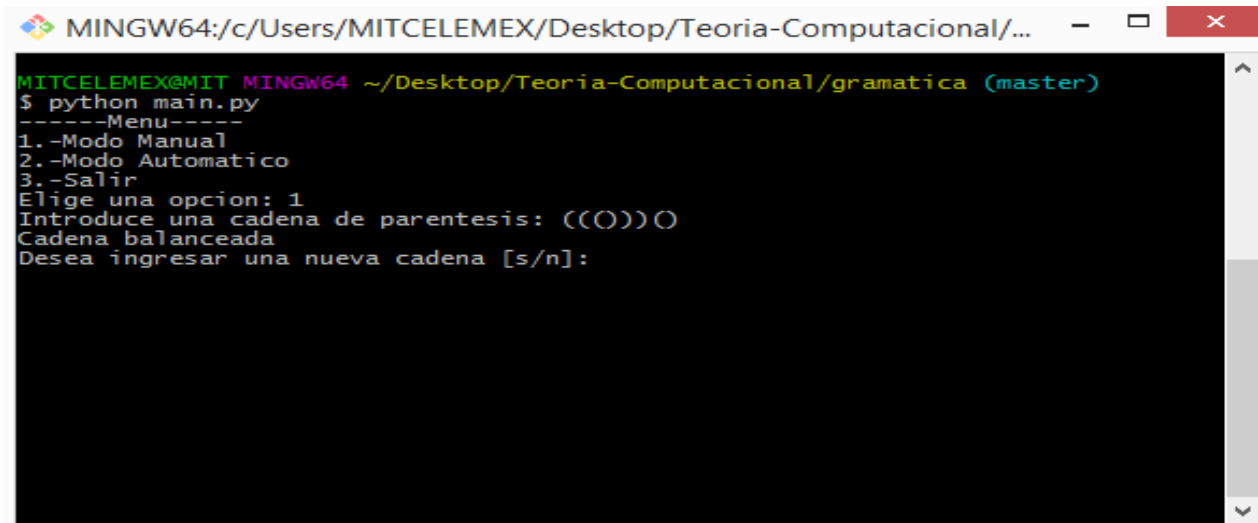
def main():
    IniciarArchivo()
    Menu()
    op=Eleccion()
    while True:
        if(op==1):
            Manual()
            while True:
                rop=VerificarDeNuevo()
                if(rop=='s'):
                    break
                elif(rop=='n'):
                    exit()
                else:
                    continue
            elif(op==2):
                Automatico()
                while True:
                    rop=random.randint(1,2)
                    if(rop==1):
                        break
                    elif(rop==2):
                        exit()

```

```
        else:  
            continue  
elif (op==3):  
    exit ()  
else:  
    continue  
  
main ()
```

2.3. Pruebas

A continuación se mostraran algunas imágenes capturadas al momento de ejecutar el programa, dichas imágenes mostraran los resultados obtenidos.
Para el modo manual:



```
MINGW64:/c/Users/MITCELEMEX/Desktop/Teoria-Computacional/...
MITCELEMEX@MIT MINGW64 ~/Desktop/Teoria-Computacional/gramatica (master)
$ python main.py
-----Menu-----
1.-Modo Manual
2.-Modo Automatico
3.-Salir
Elige una opcion: 1
Introduce una cadena de parentesis: ((() ))
Cadena balanceada
Desea ingresar una nueva cadena [s/n]:
```

Figura 8: Cadena de paréntesis de prueba: ((()))0

B		
(RB	B -> (RB	
((RRB	R -> (RR	
(((RRRB	R -> (RR	
((() RRB	R ->)	
((())RB	R ->)	
((()))B	R ->)	
((())) (RB	B -> (RB	
((())) ()B	R ->)	
((())) ()	B -> e	

Figura 9: Historia de la evaluación de la cadena

Para el modo automático:

```
MITCELEMEX@MIT MINGW64 ~/Desktop/Teoria-Computacional/gramatica (master)
$ python main.py
-----Menu-----
1.-Modo Manual
2.-Modo Automatico
3.-Salir
Elige una opcion: 2
))O(((O((O O)O)))((O))O((O(O O O)))((O))O((O))O((O O))
(O
Cadena no balanceada
(((O)((((O O((O O)(O O)(O O((O)))))OOO((O O)O))O((O)O))O(((
O)))O)O
Cadena no balanceada
))O)))
Cadena no balanceada
O O)))(O O)(O)((O O)))((O))O)))(((O((O(O(O O O O)O)))O((((
(O(O O(
Cadena no balanceada
```

Figura 10: Cadena generada automáticamente

[illegible]

Figura 11: Historia de la evaluación del modo automático

3. Palindromos-Gramática libre de contexto

3.1. Descripción del programa

El siguiente programa genera palindromos de cadenas binarias, esto se logra gracias a las siguientes reglas de producción de la gramática:

- 1.-S→e
- 2.-S→0
- 3.-S→1
- 4.-S→0S0
- 5.-S→1S1

El programa cuenta con un modo manual y automático, con el cual se podrá elegir el tamaño del palindromo, como salidas se obtendrá en un archivo la forma en que se fue generando y en otro archivo la regla que se utilizó en cada caso.

3.2. Código

El código utilizado para la resolución del problema se muestra a continuación:

Código:palindromo.py

```
import random;

def IniciarArchivo():
    archivo=open("palindromo.txt","w")
    archivo.close
    historia=open("historiaPal.txt","w")
    historia.close

def Run(pal,longitud,par):
    try:
        archivo=open("palindromo.txt","a")
        historia=open("historiaPal.txt","a")
    except:
        exit()
    archivo.write("S")
    exito=generar_palindromo1(archivo,pal,longitud,par,historia)
    archivo.write("\n\n")
    historia.write("————\n\n")
    archivo.close
    historia.close
    return exito

def opcion_fin(par):
```

```

        if(par==True):
            op=1
        else:
            op=random.randint(2,3)
        return op

def opcion_inicio():
    op=random.randint(4,5)
    return op

def regla1(pal, historia):
    pal=pal.replace("S", "")
    historia.write("\n1.-S-->e")
    return pal

def regla2(pal, historia):
    pal=pal.replace("S", '0')
    historia.write("\n2.-S-->0")
    return pal

def regla3(pal, historia):
    pal=pal.replace("S", '1')
    historia.write("\n3.-S-->1")
    return pal

def regla4(pal, historia):
    pal=pal.replace("S", "0S0")
    historia.write("\n4.-S-->0S0")
    return pal

def regla5(pal, historia):
    pal=pal.replace("S", "1S1")
    historia.write("\n5.-S-->1S1")
    return pal

def generar_palindromo1(archivo, pal, longitud, par, historia):

    if(longitud>1):

        opcion=opcion_inicio()
        if(opcion==4):
            pal=regla4(pal, historia)
            archivo.write("\n"+pal)
        if(opcion==5):
            pal=regla5(pal, historia)
            archivo.write("\n"+pal)

    if(longitud==1):
        opcion=opcion_fin(par)
        if(opcion==1):
            pal=regla1(pal, historia)
            archivo.write("\n"+pal)
        if(opcion==2):

```



```

        pal=regla2(pal, historia)
        archivo.write("\n"+pal)
    if(opcion==3):
        pal=regla3(pal, historia)
        archivo.write("\n"+pal)
if (longitud==0):
    return 1
longitud=longitud-1
pal=generar_palindromo1(archivo, pal, longitud, par, historia)

```

Código:main.py

```

import palindromo
import random

def longitud():
    x=random.randint(0,1000);
    return x

palindromo.IniciarArchivo()
def menu():
    print("——Menu——")
    print("1.-Modo_Manual")
    print("2.-Modo_Automatico")
    print("3.-Salir")

while True:
    menu()
    opcion=input("Seleccione_una_opcion:_")
    if(opcion=="1"):
        tamaño=input("Ingrese_un_tamaño_de_cadena:_")
        print(tamaño)
        tamaño=int(tamaño)
        if(tamaño%2==0):
            g=palindromo.Run("S", (tamaño/2)+1, True)
        else:
            tamaño=int(tamaño/2)+1
            g=palindromo.Run("S", tamaño, False)

        while True:
            try:
                seln=input("Desea_regresar_al_menu\n1.-Si\n2.-No\n_")
                seln=int(seln)
            except:
                exit()

            if(seln==1):
                break

```

```

        elif (seln==2):
            exit ()
        else :
            continue
elif (opcion=="2") :
    tamaño=longitud ()
    print (tamaño)
    if (tamaño%2==0):
        g=palindromo.Run("S" ,(tamaño/2)+1,True)
    else :
        tamaño=int (tamaño/2)+1
        g=palindromo.Run("S" ,tamaño, False)

    while True:
        try:
            seln=input("Desea_regresar_al_menu\n1.- Si\n2.- No\n-")
            seln=int (seln)
        except:
            exit ()

        if (seln==1):
            break
        elif (seln==2):
            exit ()
        else :
            continue

elif (opcion=="3") :
    exit ()

else :
    print ("Seleccione_una_opcion_correcta")

```

3.3. Pruebas

A continuación se mostraran algunas imágenes capturadas al momento de ejecutar el programa, dichas imágenes mostraran los resultados obtenidos.

Para el modo manual:

```
MITCELEMEX@MIT MINGW64 ~/Desktop/Teoria-Computacional/Palindromo (master)
$ python main.py
-----Menu-----
1.-Modo Manual
2.-Modo Automatico
3.-Salir
Seleccione una opcion: 1
Ingrese un tamaño de cadena: 5
5
11011
Desea regresar al menu
1.-Si
2.-No
- |
```

Figura 12: El tamaño de la cadena sera de 5

```
5
151
11511
11011
```

Figura 13: Evaluación de la gramática

```
5.-S-->151
5.-S-->151
2.-S-->0-----
```

Figura 14: Historia de la evaluación de la gramática

Para el modo automático:

```
MITCELEMEX@MIT MINGW64 ~/Desktop/Teoria-Computacional/Palindromo (master)
$ python main.py
-----Menu-----
1.-Modo Manual
2.-Modo Automatico
3.-Salir
Seleccione una opcion: 2
413
10110100001000101011111000001001101111101011001000000011100001000101111111001111
11101001001111111011110001001011110100111001110111001100010110011111000010101101
00101101100110000000110001010110111011110000000000000111101110110101000110000000
11001101101001011010100001111100110100011001110111001110010111101001000111101111
1110010010111111100111111010001000011100000001001101011111011001000001111101010
0010000101101
Desea regresar al menu
1.-Si
2.-No
- 2
```

Figura 15: Tamaño generado automáticamente

```

5
1S1
10S01
101S101
1011S1101
10110S01101
101101S101101
1011010S0101101
10110100S00101101
101101000S000101101
10110100001S10000101101
101101000010S010000101101
1011010000100S0010000101101
10110100001000S00010000101101
101101000010000S00010000101101
1011010000100001S100010000101101
10110100001000010S0100010000101101
101101000010000101S10100010000101101
1011010000100001010S010100010000101101
10110100001000010101S11010100010000101101
101101000010000101011S111010100010000101101
1011010000100001010111S1111010100010000101101
10110100001000010101111S11111010100010000101101
101101000010000101011111S011111010100010000101101
10110100001000010101111100S001111010100010000101101
101101000010000101011111000S0001111010100010000101101
1011010000100001010111110000S00001111010100010000101101
10110100001000010101111100000S000001111010100010000101101
101101000010000101011111000001S10000011111010100010000101101
1011010000100001010111110000010S010000011111010100010000101101
10110100001000010101111100000100S0010000011111010100010000101101
101101000010000101011111000001001S10010000011111010100010000101101
1011010000100001010111110000010011S110010000011111010100010000101101
10110100001000010101111100000100110S0110010000011111010100010000101101
101101000010000101011111000001001101S10110010000011111010100010000101101
1011010000100001010111110000010011011S110110010000011111010100010000101101

```

Figura 16: Historia de la evaluación del modo automático

```

|
5.-S-->1S1
4.-S-->0S0
5.-S-->1S1
5.-S-->1S1
4.-S-->0S0
5.-S-->1S1
4.-S-->0S0
4.-S-->0S0
4.-S-->0S0
4.-S-->0S0
5.-S-->1S1
4.-S-->0S0
4.-S-->0S0
4.-S-->0S0
5.-S-->1S1
4.-S-->0S0
5.-S-->1S1
5.-S-->1S1
5.-S-->1S1

```

Figura 17: Historia de la evaluación de la gramática

4. Autómata de Pila

4.1. Descripción del programa

El siguiente programa verifica la siguiente gramática 0^n1^n con $n > 0$, cuenta con un modo automático y manual, en la salida se imprimirá la historia de la evaluación de la cadena y si esta es aceptada o no, además se guardará de igual forma en un archivo la historia de esta misma evaluación.

El siguiente es un autómata de pila, es decir al recibir un 0 el autómata meterá en la pila una X y al recibir un 1 el autómata sacará esta X. La cadena será aceptada si la pila se encuentra vacía.

4.2. Código

El código utilizado para la resolución del problema se muestra a continuación:

Código:Nodo.java

```
package Stack;

public class Nodo {
    private String dato;
    private Nodo siguiente;

    public Nodo() {
        this.dato="";
        this.siguiente = null;
    }

    public String getDato() {
        return dato;
    }

    public void setDato(String dato) {
        this.dato = dato;
    }

    public Nodo getSiguiente() {
        return siguiente;
    }

    public void setSiguiente(Nodo siguiente) {
        this.siguiente = siguiente;
    }
}
```

```
}
```

Código:Pila.java

```
package Stack;

public class Pila {
    private Nodo tope;

    public Pila () {
        this.tope = null;
    }

    public void push(String dato){
        Nodo nuevo = new Nodo();
        nuevo.setDato(dato);

        if(is_empty()){
            tope=nuevo;
        }
        else{
            nuevo.setSiguiente(tope);
            tope=nuevo;
        }
    }

    public String pop(){
        Nodo auxiliar;
        String dato="";
        if(!is_empty()){
            auxiliar=tope;
            tope=tope.getSiguiente();
            dato=auxiliar.getDato();
            auxiliar=null;
        }
        return dato;
    }

    public boolean is_empty(){
        if(tope==null){
            return true;
        }
        else{
            return false;
        }
    }
}
```

```
}
```

Código:AutomataPila.java

```
package automata;

import Stack.Pila;

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

public class AutomataPila {
    private Pila automata;
    private int tamaño;
    private String cadena;

    public AutomataPila () {
        automata=new Pila ();
        cadena="Z0";
    }

    public String evaluar(char caracter,String estado){
        //System.out.println(estado);
        if(estado.equals("q0")){
            if(caracter=='1'){
                if(!automata.is_empty()){
                    estado="q1";
                    automata.pop();
                    cadena=cadena.substring(1);
                    tamaño--;
                }
                else{
                    return "";
                }
            }
            else if(caracter=='0'){
                estado="q0";
                automata.push("X");
                cadena="X"+cadena;
                //System.out.println(automata.imprimirPila());
                tamaño++;
            }
            else{
```



```

        return "";
    }
}
else if(estado.equals("q1")){
    if(!automata.is_empty()){
        if(caracter=='1'){
            estado="q1";
            automata.pop();
            cadena=cadena.substring(1);
            tamano--;
        }
        else if(caracter=='0'){
            return "";
        }
        else{
            return "";
        }
    }
    else{
        return "";
    }
}
return estado;
}

public boolean fin () {
    if(automata.is_empty()){
        return true;
    }
    else{
        return false;
    }
}

public void vaciarPila () {

    for(int i=0;i<tamano;i++){
        automata.pop();
    }

}

public void IniciarArchivo(String texto){
    File f = new File ("archivo.txt");

    if ( !( f.exists ( ) ) ){
        try {
            FileWriter w = new FileWriter ( f, true );
            f.createNewFile ( );
            w.write (texto);
        }
    }
}

```

```

        }
        catch ( Exception e ) {
            e.printStackTrace ( );
        }
    }
    else {
        try{
            FileWriter w = new FileWriter ( f, true );
            w.write (texto);
            w.close ( );
        }

        catch (Exception e ) {
            e.printStackTrace ( );
        }
    }
}

public String getCadena() {
    return cadena;
}

public void setCadena(String cadena) {
    this.cadena = cadena;
}

public int GenerarNumero() {
    int numero;
    numero=(int) (Math.random() *100)+1;
    return numero;
}

public int Generarbit() {
    int numero;
    numero=(int) (Math.random() *2);
    return numero;
}

public String generarCadena() {
    String cadena="";
    int numero=GenerarNumero();
    for (int i = 0; i < numero; i++) {
        cadena=cadena+Generarbit();
    }

    return cadena;
}
}

```

Código:main.java

```
package automata;
import java.util.Scanner;
import javax.swing.JOptionPane;

public class Main {

    public static void main(String[] args) {
        String cadena;
        String bin_aux;
        String estado="q0";
        char cadena_aux;
        AutomataPila automata=new AutomataPila();
        String opcion;
        String opcion2;
        //Menu

        System.out.println("Menu\n1.-Modo_manual\n2.-Modo_Automatico\n3.-Salir\nElija_una_opcion:_");
        Scanner entrada = new Scanner(System.in);
        opcion=entrada.nextLine();

        if(opcion.equals("1")){
            while(true){
                cadena=JOptionPane.showInputDialog(null,"Introduzca_una_cadena_binaria");
                bin_aux=cadena;
                for(int i=0;i<cadena.length();i++){
                    cadena_aux=cadena.charAt(i);
                    automata.IniciarArchivo("(" +estado+" ,_" +bin_aux+" ,_" +
                        automata.getCadena()+")");
                    System.out.println("(" +estado+" ,_" +bin_aux+" ,_" +automata.
                        getCadena()+")");
                    estado=automata.evaluar(cadena_aux,estado);

                    bin_aux=bin_aux.substring(1);
                    //System.out.println(estado);
                    if(estado.equals("")){
                        break;
                    }
                }
                if(automata.fin() && estado.equals("q1")){
                    automata.IniciarArchivo("(" +estado+" ,_e_, "+automata.getCadena()+")");
                    System.out.println("(" +estado+" ,_e_, "+automata.getCadena()+")"
```

```

        );
        estado="f";
        automata.IniciarArchivo (" (" +estado+" ,_e_, "+automata.getCadena
            ()+" )\n\n");
        System.out.println (" (" +estado+" ,_e_, "+automata.getCadena ()+" ) "
            );
        System.out.println ("Cadena_aceptada");
    }
    else{
        System.out.println ("Cadena_no_aceptada");
    }
    estado="q0";
    automata.vaciarPila ();
    automata.setCadena ("Z0");
    while(true){
        System.out.println ("Desea_evaluar_otra_cadena\n1.- Si\n2.- No\n
            nElija_una_opcion:_");
        Scanner entrada = new Scanner(System.in);
        opcion2=entrada.nextLine ();
        if(opcion2.equals ("1")){
            break;
        }
        else if(opcion2.equals ("2")){
            System.exit (0);
        }
    }
}
else if(opcion.equals ("2")){
    while(true){
        cadena=automata.generarCadena ();
        bin_aux=cadena;
        System.out.println (bin_aux);
        for(int i=0;i<cadena.length (); i++){
            cadena_aux=cadena.charAt (i);
            automata.IniciarArchivo (" (" +estado+" ,_ "+bin_aux+" ,_ "+
                automata.getCadena ()+" )");
            System.out.println (" (" +estado+" ,_ "+bin_aux+" ,_ "+automata.
                getCadena ()+" )");
            estado=automata.evaluar (cadena_aux, estado);

            bin_aux=bin_aux.substring (1);
            //System.out.println (estado);
            if(estado.equals ("")){
                break;
            }
        }
        if(automata.fin () && estado.equals ("q1")){
            automata.IniciarArchivo (" (" +estado+" ,_e_, "+automata.getCadena
                ()+" )");
            System.out.println (" (" +estado+" ,_e_, "+automata.getCadena ()+" ) "

```

```

        );
        estado="f";
        automata.IniciarArchivo (" (" +estado+" ,_e_, "+automata.getCadena
            ()+" )\n\n");
        System.out.println (" (" +estado+" ,_e_, "+automata.getCadena ()+" ) "
            );
        System.out.println ("Cadena_aceptada");
    }
    else{
        System.out.println ("Cadena_no_aceptada");
    }
    estado="q0";
    automata.vaciarPila ();
    automata.setCadena ("Z0");
    while(true){
        System.out.println ("Desea_evaluar_otra_cadena\n1.- Si\n2.- No\n
            nElija_una_opcion:_");
        int reop=automata.Generarbit ();
        System.out.println (reop+1);
        if (reop==0){
            break;
        }
        else if (reop==1){
            System.exit (0);
        }
    }
}
else if (opcion.equals ("3")){
    System.exit (0);
}
else{
    System.out.println ("Opcion_Invalida");
}
}
}

```

4.3. Pruebas

A continuación se mostraran algunas imágenes capturadas al momento de ejecutar el programa, dichas imágenes mostraran los resultados obtenidos.
Para el modo manual:

```
run:
Menu
1.-Modo manual
2.-Modo Automatico
3.-Salir
Elija una opcion:
1
```

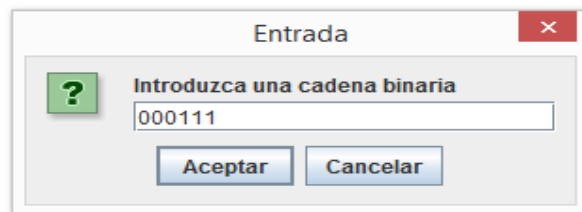


Figura 18: Cadena de prueba: 000111

```
run:
Menu
1.-Modo manual
2.-Modo Automatico
3.-Salir
Elija una opcion:
1
(q0, 000111, Z0)
(q0, 00111, XZ0)
(q0, 0111, XXZ0)
(q0, 111, XXXZ0)
(q1, 11, XXZ0)
(q1, 1, XZ0)
(q1, e, Z0)
(f, e, Z0)
Cadena aceptada
Desea evaluar otra cadena
1.-Si
2.-No
Elija una opcion:
```

Figura 19: Cadena de prueba: 000111

$(q_0, 000111, Z_0)(q_0, 00111, XZ_0)(q_0, 0111, XXZ_0)(q_0, 111, XXXZ_0)(q_1, 11, XXZ_0)(q_1, 1, XZ_0)(q_1, e, Z_0)(f, e, Z_0)$

Figura 20: Historia de la evaluación de la cadena

Para el modo automático:

```
run:
Menu
1.-Modo manual
2.-Modo Automatico
3.-Salir
Elija una opcion:
2
00100101111011110100000110011011001001011100000001011100000110
(q0, 00100101111011110100000110011011001001011100000001011100000110, Z0)
(q0, 0100101111011110100000110011011001001011100000001011100000110, XZ0)
(q0, 100101111011110100000110011011001001011100000001011100000110, XXZ0)
(q1, 00101111011110100000110011011001001011100000001011100000110, XZ0)
Cadena no aceptada
Desea evaluar otra cadena
1.-Si
2.-No
Elija una opcion:
1
100011111011110001001100101000101
(q0, 100011111011110001001100101000101, Z0)
Cadena no aceptada
Desea evaluar otra cadena
1.-Si
2.-No
Elija una opcion:
2
```

Figura 21: Cadena generada automáticamente

```

(q0, 000111, Z0)(q0, 00111, XZ0)(q0, 0111, XXZ0)(q0, 111, XXXZ0)(q1, 11, XXZ0)(q1, 1, XZ0)(q1, e ,Z0)(f, e ,Z0)|
(q0, 00100101111011110100000110011011001001011100000001011100000110, Z0)(q0,
0100101111011110100000110011011001001011100000001011100000110, XZ0)(q0,
100101111011110100000110011011001001011100000001011100000110, XXZ0)(q1,
00101111011110100000110011011001001011100000001011100000110, XZ0)(q0, 100011111011110001001100101000101, Z0)

```

Figura 22: Historia de la evaluación del modo automático

5. Maquina de Turing

5.1. Descripción del programa

El siguiente programa es la Maquina de Turing, esta evalua si una cadena cuenta con la condición de ser $0^n 1^n$ con $n \geq 1$.

Se mostrara en un archivo de texto la forma en la que esta se evalua y como van cambiando entre sus estados, asimismo se mostrara en otro archivo las cadenas que fueron aceptadas por la maquina.

La maquina de Turing es la siguiente.

Estado	Símbolo				
	0	1	X	Y	B
q_0	(q_1, X, R)	—	—	(q_3, Y, R)	—
q_1	$(q_1, 0, R)$	(q_2, Y, L)	—	(q_1, Y, R)	—
q_2	$(q_2, 0, L)$	—	(q_0, X, R)	(q_2, Y, L)	—
q_3	—	—	—	(q_3, Y, R)	(q_4, B, R)
q_4	—	—	—	—	—

Figura 23: Maquina de Turing que acepta $0^n 1^n$ con $n \geq 1$

5.2. Código

El código utilizado para la resolución del problema se muestra a continuación:

Código:MaquinaTuring.py

```
def maquinaTuring(cadena, archivo):
    estado=[0, "", 0]
    cadena=list(cadena)
    index=0
    while True:
        for e in cadena:
            archivo.write(e)
            archivo.write("\n")
        for i in range(index):
            archivo.write("_")
```

```

archivo.write("|")
archivo.write("\n")
for i in range(index):
    archivo.write("_")
archivo.write("q"+str(estado[0])+"\n")
try:
    if(index<0):
        raise Exception
    elif(estado[0]==0):
        estado[0]=estadoCero(cadena,index,estado)
        index=estado[2]
    elif(estado[0]==1):

        estado[0]=estadoUno(cadena,index,estado)
        index=estado[2]

    elif(estado[0]==2):
        estado[0]=estadoDos(cadena,index,estado)
        index=estado[2]
    elif(estado[0]==3):
        estado[0]=estadoTres(cadena,index,estado)
        index=estado[2]
    elif(estado[0]==4 or estado[0]==-1):
        break
    else:
        break
except Exception as ex:
    if(estado[0]==3):
        estado[0]=4
    else:
        break
return estado[0]
def estadoCero(cadena,index,estado):
    if(cadena[index]=='0'):
        cadena[index]='X'
        estado[1]="X"
        estado[2]=estado[2]+1
        return 1
    elif(cadena[index]=='Y'):
        cadena[index]='Y'
        estado[1]="Y"
        estado[2]=estado[2]+1
        return 3
    else:
        return -1

def estadoUno(cadena,index,estado):
    if(cadena[index]=='0'):
        cadena[index]='0'
        estado[1]="0"
        estado[2]=estado[2]+1
        return 1

```

```

elif(cadena[index]== '1') :
    cadena[index]= 'Y'
    estado[1]= "Y"
    estado[2]=estado[2]-1
    return 2
elif(cadena[index]== 'Y') :
    cadena[index]= 'Y'
    estado[2]=estado[2]+1
    return 1
else:
    return -1

def estadoDos(cadena, index, estado) :
    if(cadena[index]== '0') :
        cadena[index]= '0'
        estado[1]= "0"
        estado[2]=estado[2]-1
        return 2
    elif(cadena[index]== 'X') :
        cadena[index]= 'X'
        estado[1]= "X"
        estado[2]=estado[2]+1
        return 0
    elif(cadena[index]== 'Y') :
        cadena[index]= 'Y'
        estado[2]=estado[2]-1
        return 2
    else:
        return -1

def estadoTres(cadena, index, estado) :
    if(cadena[index]== 'Y') :
        cadena[index]= 'Y'
        estado[1]= "Y"
        estado[2]=estado[2]+1
        return 3
    elif(cadena[index]== 'B') :
        cadena[index]= 'B'
        estado[1]= "B"
        estado[2]=estado[2]+1
        return 4
    else:
        return -1

```

Código:main.py

```

import random
import MaquinaTuring

```

```

def longitud():
    num=random.randint(1,1000)
    return num

def num_bin():
    bin=random.randint(0,1)
    return bin

def generar_cadena():
    lon=longitud()
    cadena=''
    for i in range(1,lon+1):
        bin=str(num_bin())
        cadena=cadena+bin
    return cadena

def menu():
    print("-----MAQUINA_DE_TURING-----")
    print("1.-Modo_Manual")
    print("2.-Modo_Automatico")
    print("3.-Salir")
    op=input("Elija_una_opcion:_")
    return op

def iniciarArchivo():
    archivo=open("historia.txt","w")
    archivo.close
    validas=open("Validas.txt","w")
    validas.close

def manual():
    cadena=input("Ingresa_una_cadena_binaria:_")
    return cadena

def main():
    iniciarArchivo()
    estado=[0,'',0]
    try:
        archivo=open("historia.txt","a")
        validas=open("Validas.txt","a")
    except:
        print("Error_al_abrir_el_archivo")
    while True:
        opcion=menu()
        if(opcion=='1'):
            while True:
                cadena=manual()
                estado=MaquinaTuring.maquinaTuring(cadena, archivo)
                if(estado==4):
                    validas.write(cadena)
                    validas.write("\n")

```

```

        print ("Cadena_Valida")
    else:
        print ("Cadena_invalida")
    while True:
        print ("Desea_evaluar_otra_cadena_[s/n]:_")
        reop=input("")
        if(reop=='s'):
            break
        if(reop=='n'):
            exit()
    elif(opcion=='2'):
        while True:
            cadena=generar_cadena()
            estado=MaquinaTuring.maquinaTuring(cadena, archivo)
            if(estado==4):
                validas.write(cadena)
                validas.write("\n")
                print ("Cadena_Valida")
            else:
                print ("Cadena_invalida")
            archivo.write("\n\n")
        while True:
            print ("Desea_regresar_al_menu_[s/n]:_")
            reop=num_bin()
            if(reop==0):
                print ("s")
                break
            if(reop==1):
                print ('n')
                exit()
    elif(opcion=='3'):
        archivo.close
        validas.close
        exit()
    else:
        print ("Selecciona_una_opcion_correcta")
archivo.close
validas.close

```

```
main()
```

5.3. Pruebas

A continuación se mostraran algunas imágenes capturadas al momento de ejecutar el programa, dichas imágenes mostraran los resultados obtenidos.
Para el modo manual:

```
MITCELEMEX@MIT MINGW64 ~/Desktop/Teoria-Computacional/MaquinaTuring (master)
$ python main.py
-----MAQUINA DE TURING-----
1.-Modo Manual
2.-Modo Automatico
3.-Salir
Elija una opcion: 1
Ingresa una cadena binaria: 00001111
Cadena Valida
Desea evaluar otra cadena [s/n]:
```

Figura 24: La cadena a evaluar será 00001111

```
00001111
|
q0
x0001111
|
q1
x0001111
|
q1
x0001111
|
q1
x0001111
|
q1
x000Y111
|
q2
x000Y111
|
q2
x000Y111
```

Figura 25: Fragmento de la evaluación de la Maquina de Turing

Para el modo automático:

```
MITCELEMEX@MIT MINGW64 ~/Desktop/Teoria-Computacional/MaquinaTuring (master)
$ python main.py
-----MAQUINA DE TURING-----
1.-Modo Manual
2.-Modo Automatico
3.-Salir
Elija una opcion: 2
1011000011101011100011111000010110001010001000101010111111001110001001001110111
00111011101101010111110010111001111011111010101100110111111101110101000001011101
101111110111001010101010001010010110111010001001110101111000100011011001101000
11001000011101110101010111011000001100011010010101001110100110001011000110111
111111111011001111000010001111010111000100011101010110001011100010111101001111
100111011101111100000110110000110111101111010100100000110000101000000101110110
101110111000010011101000001001011000000101111010111111101101000010110110010111
01100101101010100110010000011010111100101111000011101011110001110000010010011101
0011100001011001010010100101101111110011011100001000111010000001110111101101001
10011001001110001000110100100001010100000110000000001010101001010111000110110101
00110111000001100111
Cadena invalida
Desea regresar al menu [s/n]:
n
MITCELEMEX@MIT MINGW64 ~/Desktop/Teoria-Computacional/MaquinaTuring (master)
$
```

Figura 26: Tamaño y cadena generado automáticamente

```
10110000111010111000111110000101100010100010001010111111001110001001001110111001110111011011111110010111001111011111110101011001111111011111101101000001011101101111
1101110010110110100010100101101110100010011101101111000100011011101010101101100000110001101001010011101001100010110001011111111111111110110
01111000010001111010111000100011101011000101110001011110100111100111011111000001101100001101111011110101001000001100000101101101110111000010011101
00000100101100000010111110101111110110100010110110011110110010110101001100100000110101110001110000100100111010011100001011001010010100101
101111111001101110000100011101000001110111010011001100111000100011010010001100000000101010010101100011011010100110111000001100111
|
q0
1011000011101011100011111000010110001010001000101011111100111000100100111011100111011101101111110010111001111011111101011100111101111110101011001111111011010000010111011111
110111001011011010001010010110111010001001110110111000100011011101010101101100000110001101001010011101001100010110001011111111111111110110
01111000010001111010111000100011101011000101110001011110100111100111011111000001101100001101111011101001000001100000101101101110111000010011101
000001001011000000101111101011111101101000101101100101110110010110101001100100000110101110010111100011100011100001011001011001010010100101
101111111001101110000100011101000001110111010011001100111000100011010010001100000000101010010101100011011010100110111000001100111
|
q-1
```

Figura 27: Historia de la evaluación del modo automático

00001111

Figura 28: Cadenas validas aceptadas