
SEGUNDA ENTREGA

TEORÍA COMPUTACIONAL

BY

GARCÍA DÍAZ RICARDO AXEL

PROFESSOR: GENARO JUAREZ MARTINEZ

*Escuela Superior de Cómputo
Instituto Politécnico Nacional*

Index

2018

Índice

1. Generador de cadenas a partir de una Expresión Regular	3
1.1. Descripción del problema	3
1.2. Código	3
1.3. Capturas	5
1.3.1. Consola	5
1.3.2. Archivo de Texto	8
2. Autómata de Pila, Número par de 1's y 0's	10
2.1. Descripción del problema	11
2.2. Código	11
2.3. Capturas	17
2.3.1. Consola en automatico con error	17
2.3.2. Graficacion error	18
2.3.3. Archivo de texto error	19
2.3.4. Consola manual con éxito	20
2.3.5. Graficacion exito en proceso	21
2.3.6. Graficacion exito en proceso	22
2.3.7. Archivo de Texto Exito	23
3. Generador de Palíndromos	23
3.1. Descripción del problema	23
3.2. Código	24
3.2.1. Consola Manual	26
3.2.2. Archivo de texto modo Manual	27
3.2.3. Consola Automático	28
3.2.4. Archivo de texto modo Automático	33

4. Arbol de Derivaciones, Balanceo de paréntesis	34
4.1. Descripción del problema	34
4.2. Código	34
4.3. Capturas	37
4.3.1. Consola Éxito	37
4.3.2. Archivo de texto Éxito	38
4.3.3. Consola Fracaso	39
4.3.4. Archivo de texto Fracaso	40
5. Maquina de Turing	41
5.1. Descripción del problema	41
5.2. Código	41
5.3. Capturas	47
5.3.1. Consola Manual con Éxito	47
5.3.2. Graficacion parcial Éxito	48
5.3.3. Archivo de texto Éxito	49
5.3.4. Consola Automática Fracaso	50
5.3.5. Graficacion parcial Fracaso	52
5.3.6. Archivo de texto Fracaso	53

1. Generador de cadenas a partir de una Expresión Regular

. Programa:8.py | Texto:Datos8.txt

1.1. Descripción del problema

Dada la expresión "(0 U 10)*(e U 1)", generar 10 cadenas aleatorias que pertenezcan a esta. Cada decisión se realizará con un aleatorio distinto.

1.2. Código

```
#EXPRESION REGURAL A AUTOMATA
import sys
import time
import random
sys.setrecursionlimit(1000)
from tkinter import*

archivo = open("Datos8.txt", "w")

expresiones = []

print ("Generador de cadenas de la expresion:
(0 U 10)*(e U 1)")

archivo.write("\n"+"Generador de cadenas de laexpresion:
(0 U 10)*(e U 1)")

for i in range(0,10):
    print(i+1,".-")
    archivo.write("\n"+str(i+1)+".-")
    expresiones.append("")

    ck = random.choice(["e", "n"]) #random para
    cerradura de kleene para ver si es e o n
    print ("ck=",ck)
    archivo.write("\n"+"ck="+ck)

    if ck == "n":
```

```

        n = random.randint(1, 10)#si es n otro
random para encontrar ese n
        print("n:",n)
        archivo.write("\n"+"n:"+str(n))
        for j in range(0,n):
            a1 = random.choice(["0", "10"])
#random para ver si se escoje 0 o 10

            print("a1=",a1)
            archivo.write("\n"+"a1="+a1)
            expresiones[i] = expresiones[i]+a1


        print ("Expresion antes de la multiplicacion:"
,expresiones[i])

archivo.write("\n"+"Expresion antes de la
multiplicacion:"+expresiones[i])

        a2 = random.choice(["", "1"])
#random para escoger entre e o 1

        print("a2="+a2)
        archivo.write("\n"+"a2="+a2)

        if(a2=="1"):
            print("Hay 1 al final , se suma")
            archivo.write("\n"+"Hay 1 al final ,
se suma")

            expresiones[i] = expresiones[i]+a2

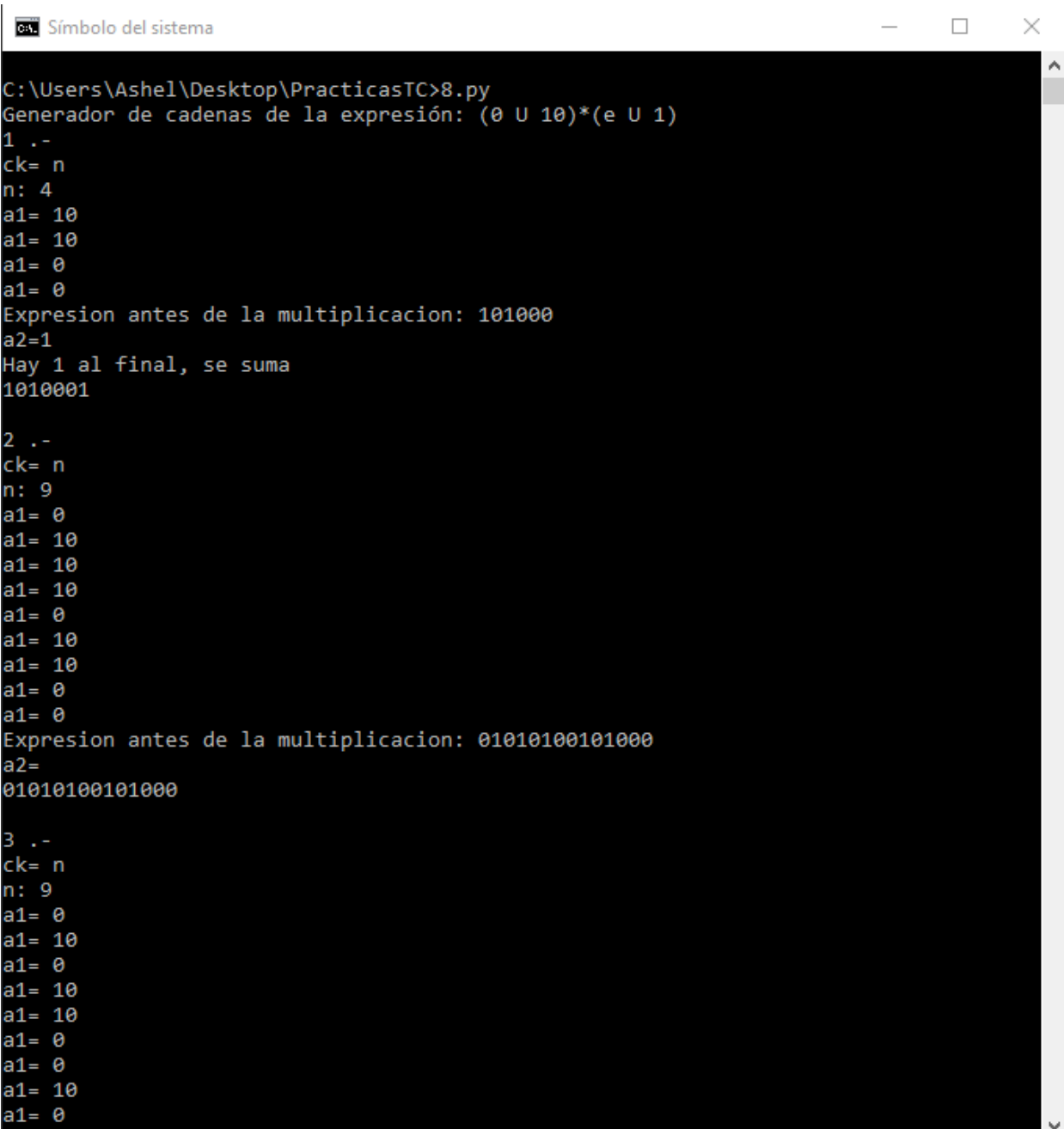
        print (expresiones[i], "\n")
        archivo.write("\n"+expresiones[i]+ "\n")

print (expresiones)
archivo.write("\n"+str(expresiones))

```

1.3. Capturas

1.3.1. Consola



```
ca. Símbolo del sistema

C:\Users\Ashel\Desktop\PracticasyTC>8.py
Generador de cadenas de la expresión: (0 U 10)*(e U 1)
1 .-
ck= n
n: 4
a1= 10
a1= 10
a1= 0
a1= 0
Expresion antes de la multiplicacion: 101000
a2=1
Hay 1 al final, se suma
1010001

2 .-
ck= n
n: 9
a1= 0
a1= 10
a1= 10
a1= 10
a1= 0
a1= 10
a1= 10
a1= 0
a1= 0
Expresion antes de la multiplicacion: 01010100101000
a2=
01010100101000

3 .-
ck= n
n: 9
a1= 0
a1= 10
a1= 0
a1= 10
a1= 10
a1= 0
a1= 0
a1= 10
a1= 0
```

```
Símbolo del sistema
0100101000100
4 .-
ck= n
n: 9
a1= 10
a1= 10
a1= 0
a1= 0
a1= 0
a1= 10
a1= 10
a1= 0
a1= 10
Expresion antes de la multiplicacion: 10100001010010
a2=
10100001010010
5 .-
ck=
Expresion antes de la multiplicacion:
a2=
6 .-
ck=
Expresion antes de la multiplicacion:
a2=1
Hay 1 al final, se suma
1
7 .-
ck=
Expresion antes de la multiplicacion:
a2=
8 .-
ck= n
n: 2
a1= 10
a1= 10
Expresion antes de la multiplicacion: 1010
```

```
ca. Símbolo del sistema
Expresión antes de la multiplicación:
a2=

8 .-
ck= n
n: 2
a1= 10
a1= 10
Expresión antes de la multiplicación: 1010
a2=1
Hay 1 al final, se suma
10101

9 .-
ck=
Expresión antes de la multiplicación:
a2=1
Hay 1 al final, se suma
1

10 .-
ck= n
n: 3
a1= 10
a1= 10
a1= 10
Expresión antes de la multiplicación: 101010
a2=1
Hay 1 al final, se suma
1010101

['1010001', '01010100101000', '0100101000100', '10100001010010', '', '1', '', '1010
1', '1', '1010101']

C:\Users\Ashel\Desktop\PracticasTC>
```


1.3.2. Archivo de Texto

```
Datos8: Bloc de notas
Archivo Edición Formato Ver Ayuda

Generador de cadenas de la expresión: (0 U 10)*(e U 1)
1.-
ck=n
n:4
a1=10
a1=10
a1=0
a1=0
Expresion antes de la multiplicacion:101000
a2=1
Hay 1 al final, se suma|
1010001

2.-
ck=n
n:9
a1=0
a1=10
a1=10
a1=10
a1=0
a1=10
a1=10
a1=0
a1=0
Expresion antes de la multiplicacion:01010100101000
a2=
01010100101000

3.-
ck=n
n:9
a1=0
a1=10
a1=0
```

3. -

ck=n

n:9

a1=0

a1=10

a1=0

a1=10

a1=10

a1=0

a1=0

a1=10

a1=0

Expresion antes de la multiplicacion:0100101000100

a2=

0100101000100

4. -

ck=n

n:9

a1=10

a1=10

a1=0

a1=0

a1=0

a1=10

a1=10

a1=0

a1=10

Expresion antes de la multiplicacion:10100001010010

a2=

10100001010010

5. -

ck=

Expresion antes de la multiplicacion:

a2=

```
Datos8: Bloc de notas
Archivo Edición Formato Ver Ayuda

7.-
ck=
Expresion antes de la multiplicacion:
a2=

8.-
ck=n
n:2
a1=10
a1=10
Expresion antes de la multiplicacion:1010
a2=1
Hay 1 al final, se suma
10101

9.-
ck=
Expresion antes de la multiplicacion:
a2=1
Hay 1 al final, se suma
1

10.-
ck=n
n:3
a1=10
a1=10
a1=10
Expresion antes de la multiplicacion:101010
a2=1
Hay 1 al final, se suma
1010101

['1010001', '01010100101000', '0100101000100', '10100001010010', '', '1', '', '10']
```

2. Autómata de Pila, Número par de 1's y 0's

Programa:9.py | Texto:Datos9.txt

2.1. Descripción del problema

El programa verificará si una cadena (ingresada o generada) con tiene un numero par de 1's y 0's siempre comenzando en 1 y terminando en 0.

2.2. Código

```
#AUTOMATA DE PILA
import sys
import time
import random
sys.setrecursionlimit(1000)
from tkinter import*

archivo = open("Datos9.txt", "w")

class Pila(object):

    def __init__(self):
        super(Pila, self).__init__()
        self.arreglo = []
        self.cnt = 0

    def push(self, objeto):
        self.cnt += 1
        self.arreglo.append(objeto)
    def pop(self):
        if self.cnt == 0:
            return False
        else:
            self.cnt -= 1
            del self.arreglo[self.cnt]
            return True
    def top(self):
        return self.arreglo[self.cnt-1]

print("Cadenas numero par de 1 y 0 \n 1)Modo manual. 2)Modo automatico")
archivo.write("Cadenas numero par de 1 y 0 \n 1)Modo manual.
2)Modo automatico")
```

```

opc = input()

if (opc=="1"):
    cad = input("Cadena a evaluar:\n") #cadena a evaluar
    archivo.write("Cadena a evaluar: \n")

else:
    rand = random.randrange(10000)
    cad = str(bin(rand)[2:])
    print("Cadena generada:", cad)
    archivo.write("\n Cadena generada:"+cad)
cadc = cad[::-1] #cadena invertida para meterla a la pila

p = Pila()

#Declaracion interfaz grafica
ventana = Tk()
canv = Canvas(ventana,width=800,height=600)
ventana.geometry("800x600")

#
canv.pack()
xspeed=5
yspeed=0
tk= Tk()

e = "a" #a son 0's b son 1's
aux1 = 0 #auxiliar para ver los 1 que se van llenando

cuenta = 0

o= Label(ventana,text="Cadena a verificar. \n Cadena:"+cad)
.place(x=10,y=10)

cont = 0

conta = 0 #auxiliar para contabilizar los 0 que llegan adecuadamente
contb = 0 #auxiliar para contabilizar los 1 que llegan adecuadamente

```

```

conterr = 0 #auxiliar para contar los errores

canv.create_rectangle(300, 50, 400, 550, width=0, fill='black')

for i in cadc:

    u= Label(ventana,text="Cadena evaluandose \n "+
cad[0:len(cad)-cont]).place(x=10,y=100)

    tk.update()
    time.sleep(.5)
    cont = cont + 1

    if(cadc[cuenta]=="1"):
        cuenta = cuenta + 1
        p.push("X")
        print(p.arreglo)
        archivo.write("\n"+str(p.arreglo))
        aux1= aux1+1
        e = "c"

        #grafico
        canv.create_rectangle(300, 500-conterr*25, 400
, 550-conterr*25, width=1, fill='red', outline='white')

        conterr= conterr+1
        time.sleep(.5)
        conterr = conterr + 1
        continue

    if(i=="0" and e == "a"):

        p.push(i)
        print(p.arreglo)
        archivo.write("\n"+str(p.arreglo))
        #grafico
        canv.create_rectangle(300, 500-conta*25, 400,
550-conta*25,
width=1, fill='blue', outline='white')

```

```

        conta= conta+1
        time.sleep(.5)
        conta = conta + 1
        continue

    if(i=="0" and e == "b"): #el estado c sirve cuando se comienza
con un 1 y solo se van aniadiendo X's cuando pasan nuevos 1,
cuando pasa un 0 no pasa nada

        e = "c"
        p.push("X")
        print(p.arreglo)
        archivo.write("\n"+str(p.arreglo))
        continue

    if(i=="0" and e == "c"):
        p.push("X")
        print(p.arreglo)
        archivo.write("\n"+str(p.arreglo))

        continue

    if(i=="1" and e =="c"):
        p.push("X")
        print(p.arreglo)
        archivo.write("\n"+str(p.arreglo))

        canv.create_rectangle(300, 500-conterr*25, 400,
550-conterr*25, width=1, fill='red', outline='white')
        conterr= conterr+1
        time.sleep(.5)
        conterr = conterr + 1
        continue

    if(i=="1" and e=="a"):
        e="b"
        p.pop()
        print(p.arreglo)
        archivo.write("\n"+str(p.arreglo))
        aux1= aux1+1

```

```

        #grafico
        canv.create_rectangle(300, 550-((conta)*25), 400,
600-(conta)*25, width=1, fill='black', outline='white')
        contb= contb+1
        time.sleep(.5)
        continue

if(i=="1" and len(p.arreglo)<aux1):
    p.push("X")
    print(p.arreglo)
    archivo.write("\n"+str(p.arreglo))
    aux1= aux1+1

    #grafico
    canv.create_rectangle(300, 500-conterr*25, 400,
550-conterr*25, width=1, fill='red', outline='white')

    conterr= conterr+1
    time.sleep(.5)
    conterr = conterr + 1
    continue

if(i=="1" and e=="b"):
    p.pop()
    print(p.arreglo)
    archivo.write("\n"+str(p.arreglo))

    #grafico
    canv.create_rectangle(300, 550-((conta)*25)+contb*50,
400, 600-(conta)*25+contb*50, width=1, fill='black',
outline='white')

    time.sleep(.5)
    contb = contb + 1
    continue

print("\n"+p.arreglo)
archivo.write("\n"+str(p.arreglo))

```



```

a=len(p.arreglo)
if(a==0):
    print("Cadena valida")
    archivo.write("\nCadena valida")
    o= Label(ventana,text="Cadena VALIDA :D!").place(x=500,y=300)
    canv.create_rectangle(300, 50, 400, 550, width=0, fill='green')

else:
    print("Cadena invalida")
    archivo.write("\nCadena invalida")
    o= Label(ventana,text="Cadena INVALIDA :(").place(x=500,y=300)
    canv.create_rectangle(300, 50, 400, 550, width=0, fill='red')


canv.place(x=0,y=0)
ventana.mainloop()

```

[frame=single, linewidth=17cm]

2.3. Capturas

2.3.1. Consola en automatico con error

 Símbolo del sistema - 9.py

```
C:\Users\Ashel\Desktop\PracticasTC>9.py
Cadenas numero par de 1 y 0
1)Modo manual. 2)Modo automatico
1
Cadena a evaluar:
2
Traceback (most recent call last):
  File "C:\Users\Ashel\Desktop\PracticasTC\9.py", line 177, in <module>
    print("\n"+p.arreglo)
TypeError: must be str, not list

C:\Users\Ashel\Desktop\PracticasTC>9.py
Cadenas numero par de 1 y 0
1)Modo manual. 2)Modo automatico
2
Cadena generada: 101011011010
['0']
[]
['X']
['X']
['X', 'X']
['X', 'X', 'X']
['X', 'X', 'X', 'X']
['X', 'X', 'X', 'X', 'X']
['X', 'X', 'X', 'X', 'X', 'X']
['X', 'X', 'X', 'X', 'X', 'X', 'X']
['X', 'X', 'X', 'X', 'X', 'X', 'X', 'X']
['X', 'X', 'X', 'X', 'X', 'X', 'X', 'X', 'X']
Cadena inválida
```

2.3.2. Graficacion error

 tk

Cadena a verificar.
Cadena:101011011010

Cadena evaluandose
1



Cadena INVALIDA :(

2.3.3. Archivo de texto error

 Datos9: Bloc de notas

Archivo Edición Formato Ver Ayuda

Cadenas numero par de 1 y 0

1)Modo manual. 2)Modo automatico

Cadena generada:101011011010

['0']

[]

['X']

['X', 'X']

['X', 'X', 'X']

['X', 'X', 'X', 'X']

['X', 'X', 'X', 'X', 'X']

['X', 'X', 'X', 'X', 'X', 'X']

['X', 'X', 'X', 'X', 'X', 'X', 'X']

['X', 'X', 'X', 'X', 'X', 'X', 'X', 'X']

['X', 'X', 'X', 'X', 'X', 'X', 'X', 'X', 'X']

['X', 'X', 'X', 'X', 'X', 'X', 'X', 'X', 'X', 'X']

Cadena inválida

2.3.4. Consola manual con éxito

```
C:\Users\Ashel\Desktop\PracticastC>9.py
Cadenas numero par de 1 y 0
  1)Modo manual. 2)Modo automatico
1
Cadena a evaluar:
11110000
['0']
['0', '0']
['0', '0', '0']
['0', '0', '0', '0']
['0', '0', '0']
['0', '0']
['0']
[]
Cadena válida
```

2.3.5. Graficacion exito en proceso

 tk

Cadena a verificar.
Cadena:11110000

Cadena evaluandose
111100



2.3.6. Graficacion exito en proceso

 tk

Cadena a verificar.
Cadena:11110000

Cadena evaluandose
1



Cadena VALIDA :D!

2.3.7. Archivo de Texto Exito



Datos9: Bloc de notas

Archivo Edición Formato Ver Ayuda

Cadenas numero par de 1 y 0

1)Modo manual. 2)Modo automaticoCadena a evaluar:

['0']

['0', '0']

['0', '0', '0']

['0', '0', '0', '0']

['0', '0', '0']

['0', '0']

['0']

[]

Cadena válida

3. Generador de Palíndromos

Programa:10.py | Texto:Datos10.txt

3.1. Descripción del problema

Con base a las reglas de derivación descritas a continuación, se generará un palíndromo de longitud ingresada manual o automáticamente.

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

3.2. Código

```
#GENERADOR DE PALINDROMOS
ar = open("Datos10.txt", "w")
import random
from random import randint

print("Generador de palindromos")
op = input("1)Longitud manual o 2)Generar un palindromo de longitud
aleatoria ")

if op == "1":
    lon = int(input("Ingrese la longitud deseada:"))
else:
    lon = randint(0,1000)

ar.write("Longitud: "+str(lon)+"\n")
l = int(lon/2)
if(lon%2!=0):
    print ("inpar")
    inpar=1
else:
    inpar=0

palin = "P" #palindromo generado

for i in range(0,l+1):
    if i == l:
        final =palin.split()

        if inpar == 0:
            print ("Se aplico la regla 1 (P -> e)")
            final.remove("P")
            print (final)
            ar.write(str(final)+"\n")

        if inpar == 1:
            reglas1 = randint(2,3)
            if reglas1 == 2:
                print ("Se aplico la regla 2 (P -> 0)")
                final[l] = "0"
```

```

        print (final)
        ar.write(str(final)+"\n")
    if reglas1 == 3:
        print ("Se aplico la regla 3 (P -> 1)")
        final[1] = "1"
        print (final)
        ar.write(str(final)+"\n")


else:
    reglas2 = randint(4,5)
    if reglas2 == 4:
        print ("Se aplico regla 4 (P -> 0P0)")
        palin = "0 "+palin+" 0"
        print (palin)
        ar.write(str(palin)+"\n")

    if reglas2 == 5:
        print("Se aplico regla 5 (P -> 1P1)")
        palin = "1 "+palin+" 1"
        print(palin)
        ar.write(str(palin)+"\n")

print("\n")

```

3.2.1. Consola Manual

 Símbolo del sistema

```
C:\Users\Ashel\Desktop\PracticasTC>10.py
Generador de palindromos
1)Longitud manual o 2)Generar un palíndromo de longitud aleatoria 1
Ingrese la longitud deseada:11
inpar
Se aplicó regla 4 (P -> 0P0)
0 P 0

Se aplicó regla 5 (P -> 1P1)
1 0 P 0 1

Se aplicó regla 5 (P -> 1P1)
1 1 0 P 0 1 1

Se aplicó regla 5 (P -> 1P1)
1 1 1 0 P 0 1 1 1

Se aplicó regla 5 (P -> 1P1)
1 1 1 1 0 P 0 1 1 1 1

Se aplicó la regla 3 (P -> 1)
['1', '1', '1', '1', '0', '1', '0', '1', '1', '1', '1']

C:\Users\Ashel\Desktop\PracticasTC>
```

3.2.2. Archivo de texto modo Manual



Datos10: Bloc de notas

Archivo Edición Formato Ver Ayuda

Longitud: 11

0 P 0

1 0 P 0 1

1 1 0 P 0 1 1

1 1 1 0 P 0 1 1 1

1 1 1 1 0 P 0 1 1 1 1

['1', '1', '1', '1', '0', '1', '0', '1', '1', '1', '1']

3.2.3. Consola Automático

```
C:\Users\Ashel\Desktop\PracticascTC>10.py
Generador de palindromos
1)Longitud manual o 2)Generar un palíndromo de longitud aleatoria 2
Se aplicó regla 4 (P -> 0P0)
0 P 0

Se aplicó regla 5 (P -> 1P1)
1 0 P 0 1

Se aplicó regla 4 (P -> 0P0)
0 1 0 P 0 1 0

Se aplicó regla 5 (P -> 1P1)
1 0 1 0 P 0 1 0 1

Se aplicó regla 5 (P -> 1P1)
1 1 0 1 0 P 0 1 0 1 1

Se aplicó regla 4 (P -> 0P0)
0 1 1 0 1 0 P 0 1 0 1 1 0

Se aplicó regla 4 (P -> 0P0)
0 0 1 1 0 1 0 P 0 1 0 1 1 0 0

Se aplicó regla 5 (P -> 1P1)
1 0 0 1 1 0 1 0 P 0 1 0 1 1 0 0 1

Se aplicó regla 5 (P -> 1P1)
1 1 0 0 1 1 0 1 0 P 0 1 0 1 1 0 0 1 1

Se aplicó regla 5 (P -> 1P1)
1 1 1 0 0 1 1 0 1 0 P 0 1 0 1 1 0 0 1 1 1
```

```

Se aplicó regla 5 (P -> 1P1)
1 1 0 0 1 1 0 1 0 P 0 1 0 1 1 0 0 1 1

Se aplicó regla 5 (P -> 1P1)
1 1 1 0 0 1 1 0 1 0 P 0 1 0 1 1 0 0 1 1 1

Se aplicó regla 4 (P -> 0P0)
0 1 1 1 0 0 1 1 0 1 0 P 0 1 0 1 1 0 0 1 1 1 0

Se aplicó regla 5 (P -> 1P1)
1 0 1 1 1 0 0 1 1 0 1 0 P 0 1 0 1 1 0 0 1 1 1 0 1

Se aplicó regla 4 (P -> 0P0)
0 1 0 1 1 1 0 0 1 1 0 1 0 P 0 1 0 1 1 0 0 1 1 1 0 1 0

Se aplicó regla 4 (P -> 0P0)
0 0 1 0 1 1 1 0 0 1 1 0 1 0 P 0 1 0 1 1 0 0 1 1 1 0 1 0 0

Se aplicó regla 4 (P -> 0P0)
0 0 0 1 0 1 1 1 0 0 1 1 0 1 0 P 0 1 0 1 1 0 0 1 1 1 0 1 0 0 0

Se aplicó regla 4 (P -> 0P0)
0 0 0 0 1 0 1 1 1 0 0 1 1 0 1 0 P 0 1 0 1 1 0 0 1 1 1 0 1 0 0 0 0
0

Se aplicó regla 5 (P -> 1P1)
1 0 0 0 0 0 1 0 1 1 1 0 0 1 1 0 1 0 P 0 1 0 1 1 0 0 1 1 1 0 1 0 0 0
0 0 1

```

```

Se aplicó regla 5 (P -> 1P1)
1 0 0 0 0 0 1 0 1 1 1 0 0 1 1 0 1 0 P 0 1 0 1 1 0 0 1 1 1 0 1 0 0 0
0 0 1

Se aplicó regla 4 (P -> 0P0)
0 1 0 0 0 0 0 1 0 1 1 1 0 0 1 1 0 1 0 P 0 1 0 1 1 0 0 1 1 1 0 1 0 0
0 0 0 1 0

Se aplicó regla 4 (P -> 0P0)
0 0 1 0 0 0 0 0 1 0 1 1 1 0 0 1 1 0 1 0 P 0 1 0 1 1 0 0 1 1 1 0 1 0
0 0 0 0 1 0 0

Se aplicó regla 5 (P -> 1P1)
1 0 0 1 0 0 0 0 0 1 0 1 1 1 0 0 1 1 0 1 0 P 0 1 0 1 1 0 0 1 1 1 0 1
0 0 0 0 0 1 0 0 1

Se aplicó regla 4 (P -> 0P0)
0 1 0 0 1 0 0 0 0 0 1 0 1 1 1 0 0 1 1 0 1 0 P 0 1 0 1 1 0 0 1 1 1 0
1 0 0 0 0 0 1 0 0 1 0

Se aplicó regla 5 (P -> 1P1)
1 0 1 0 0 1 0 0 0 0 0 1 0 1 1 1 0 0 1 1 0 1 0 P 0 1 0 1 1 0 0 1 1 1
0 1 0 0 0 0 0 1 0 0 1 0 1

Se aplicó regla 5 (P -> 1P1)
1 1 0 1 0 0 1 0 0 0 0 0 1 0 1 1 1 0 0 1 1 0 1 0 P 0 1 0 1 1 0 0 1 1
1 0 1 0 0 0 0 0 1 0 0 1 0 1 1

Se aplicó regla 5 (P -> 1P1)
1 1 1 0 1 0 0 1 0 0 0 0 0 1 0 1 1 1 0 0 1 1 0 1 0 P 0 1 0 1 1 0 0 1
1 1 0 1 0 0 0 0 0 1 0 0 1 0 1 1 1

```

```

Se aplicó regla 5 (P -> 1P1)
1 1 1 0 1 0 0 1 0 0 0 0 0 1 0 1 1 1 0 0 1 1 0 1 0 P 0 1 0 1 1 0 0 1
1 1 0 1 0 0 0 0 0 1 0 0 1 0 1 1 1

Se aplicó regla 5 (P -> 1P1)
1 1 1 1 0 1 0 0 1 0 0 0 0 0 1 0 1 1 1 0 0 1 1 0 1 0 P 0 1 0 1 1 0 0
1 1 1 0 1 0 0 0 0 0 1 0 0 1 0 1 1 1 1

Se aplicó regla 4 (P -> 0P0)
0 1 1 1 1 0 1 0 0 1 0 0 0 0 0 1 0 1 1 1 0 0 1 1 0 1 0 P 0 1 0 1 1 0
0 1 1 1 0 1 0 0 0 0 0 1 0 0 1 0 1 1 1 1 0

Se aplicó regla 5 (P -> 1P1)
1 0 1 1 1 1 0 1 0 0 1 0 0 0 0 0 1 0 1 1 1 0 0 1 1 0 1 0 P 0 1 0 1 1
0 0 1 1 1 0 1 0 0 0 0 0 1 0 0 1 0 1 1 1 1 0 1

Se aplicó regla 5 (P -> 1P1)
1 1 0 1 1 1 1 0 1 0 0 1 0 0 0 0 0 1 0 1 1 1 0 0 1 1 0 1 0 P 0 1 0 1
1 0 0 1 1 1 0 1 0 0 0 0 0 1 0 0 1 0 1 1 1 1 0 1 1

Se aplicó regla 5 (P -> 1P1)
1 1 1 0 1 1 1 1 0 1 0 0 1 0 0 0 0 0 1 0 1 1 1 0 0 1 1 0 1 0 P 0 1 0
1 1 0 0 1 1 1 0 1 0 0 0 0 0 1 0 0 1 0 1 1 1 1 0 1 1 1

Se aplicó regla 4 (P -> 0P0)
0 1 1 1 0 1 1 1 1 0 1 0 0 1 0 0 0 0 0 1 0 1 1 1 0 0 1 1 0 1 0 P 0 1
0 1 1 0 0 1 1 1 0 1 0 0 0 0 0 1 0 0 1 0 1 1 1 1 0 1 1 1 0

Se aplicó regla 5 (P -> 1P1)
1 0 1 1 1 0 1 1 1 1 0 1 0 0 1 0 0 0 0 0 1 0 1 1 1 0 0 1 1 0 1 0 P 0
1 0 1 1 0 0 1 1 1 0 1 0 0 0 0 0 1 0 0 1 0 1 1 1 1 0 1 1 1 0 1

```



```


Se aplicó regla 5 (P -> 1P1)
1 0 1 1 1 0 1 1 1 1 0 1 0 0 1 0 0 0 0 0 1 0 1 1 1 0 0 1 1 0 1 0 P 0
1 0 1 1 0 0 1 1 1 0 1 0 0 0 0 0 1 0 0 1 0 1 1 1 1 0 1 1 1 0 1

Se aplicó la regla 1 (P -> e)
['1', '0', '1', '1', '1', '0', '1', '1', '1', '1', '1', '0', '1', '0', '0', '0',
'1', '0', '0', '0', '0', '0', '0', '1', '0', '1', '1', '1', '0', '0', '0',
'1', '1', '0', '1', '0', '0', '1', '0', '1', '1', '0', '0', '1', '1',
, '1', '0', '1', '0', '0', '0', '0', '0', '0', '1', '0', '0', '1', '0', '
1', '1', '1', '1', '0', '1', '1', '1', '0', '1']

C:\Users\Ashel\Desktop\PracticasTC>

```

3.2.4. Archivo de texto modo Automático

 Datos10: Bloc de notas

[Archivo](#) [Edición](#) [Formato](#) [Ver](#) [Ayuda](#)

Longitud: 64

 $\theta \quad P \quad \theta$

1 0 P 0 1

0 1 0 P 0 1 0

1 0 1 0 P 0 1 0 1

1 1 0 1 0 P 0 1 0 1 1

0 1 1 0 1 0 P 0 1 0 1 1 0

0 0 1 1 0 1 0 P 0 1 0 1 1 0 0

1 0 0 1 1 0 1 0 P 0 1 0 1 1 0 0 1

1 1 0 0 1 1 0 1 0 P 0 1 0 1 1 0 0 1 1

1 1 1 0 0 1 1 0 1 0 P 0 1 0 1 1 0 0 1 1 1

0 1 1 1 0 0 1 1 0 1 0 P 0 1 0 1 1 0 0 1 1 1 0

1 0 1 1 1 0 0 1 1 0 1 0 P 0 1 0 1 1 0 0 1 1 1 0 1

0 1 0 1 1 1 0 0 1 1 0 1 0 P 0 1 0 1 1 0 0 1 1 1 0 1 0

0 0 1 0 1 1 1 0 0 1 1 0 1 0 P 0 1 0 1 1 0 0 1 1 1 0 1 0 0

0 0 0 1 0 1 1 1 0 0 1 1 0 1 0 P 0 1 0 1 1 0 0 1 1 1 0 1 0 0 0

0 0 0 0 1 0 1 1 1 0 0 1 1 0 1 0 P 0 1 0 1 1 0 0 1 1 1 0 1 0 0 0 0

0 0 0 0 0 1 0 1 1 1 0 0 1 1 0 1 0 P 0 1 0 1 1 0 0 1 1 1 0 1 0 0 0 0 0

1 0 0 0 0 0 1 0 1 1 1 0 0 1 1 0 1 0 P 0 1 0 1 1 0 0 1 1 1 0 1 0 0 0 0 0 :

0 1 0 0 0 0 0 1 0 1 1 1 0 0 1 1 0 1 0 P 0 1 0 1 1 0 0 1 1 1 0 1 0 0 0 0 0

0 0 1 0 0 0 0 0 1 0 1 1 1 0 0 1 1 0 1 0 P 0 1 0 1 1 0 0 1 1 1 0 1 0 0 0 0 0

1 0 0 1 0 0 0 0 0 1 0 1 1 1 0 0 1 1 0 1 0 P 0 1 0 1 1 0 0 1 1 1 0 1 0 0 0

0 1 0 0 1 0 0 0 0 0 1 0 1 1 1 0 0 1 1 0 1 0 P 0 1 0 1 1 0 0 1 1 1 0 1 0 1

1 0 1 0 0 1 0 0 0 0 0 1 0 1 1 1 0 0 1 1 0 1 0 P 0 1 0 1 1 0 0 1 1 1 0 1 0

1 1 0 1 0 0 1 0 0 0 0 0 1 0 1 1 1 0 0 1 1 0 1 0 P 0 1 0 1 1 0 0 1 1 1 0 :

1 1 1 0 1 0 0 1 0 0 0 0 0 1 0 1 1 1 0 0 1 1 0 1 0 P 0 1 0 1 1 0 0 1 1 1 0

1 1 1 1 0 1 0 0 1 0 0 0 0 0 1 0 1 1 1 0 0 1 1 0 1 0 P 0 1 0 1 1 0 0 1 1 :

0 1 1 1 1 0 1 0 0 1 0 0 0 0 0 1 0 1 1 1 0 0 1 1 0 1 0 P 0 1 0 1 1 0 0 1 :

1 0 1 1 1 1 0 1 0 0 1 0 0 0 0 0 1 0 1 1 1 0 0 1 1 0 1 0 P 0 1 0 1 1 0 0 :

1 1 0 1 1 1 1 0 1 0 0 1 0 0 0 0 0 1 0 1 1 1 0 0 1 1 0 1 0 P 0 1 0 1 1 0

1 1 1 0 1 1 1 1 0 1 0 0 1 0 0 0 0 0 1 0 1 1 1 0 0 1 1 0 1 0 P 0 1 0 1 1

0 1 1 1 0 1 1 1 1 0 1 0 0 1 0 0 0 0 0 1 0 1 1 1 0 0 1 1 0 1 0 P 0 1 0 1

1 0 1 1 1 0 1 1 1 1 0 1 0 0 1 0 0 0 0 0 1 0 1 1 1 0 0 1 1 0 1 0 P 0 1 0

```
['1' '0' '1' '1' '1' '0' '1' '1' '1' '1' '0' '1' '0' '0' '']
```

$(-1, 0), (0, -1), (1, -1), (-1, 1), (0, 1), (1, 0), (-1, 0), (0, -1), (1, -1), (-1, 1), (0, 1), (1, 0)$

```

1
0 1 0
0 0 1 0 0
0 0 0 1 0 0 1
0 0 0 0 1 0 0 1 0
0 0 0 0 0 1 0 0 1 0 1
1 0 0 0 0 0 1 0 0 1 0 1 1
0 1 0 0 0 0 0 1 0 0 1 0 1 1 1
1 0 1 0 0 0 0 0 1 0 0 1 0 1 1 1 1
1 1 0 1 0 0 0 0 0 1 0 0 1 0 1 1 1 1 0
1 1 1 0 1 0 0 0 0 0 1 0 0 1 0 1 1 1 1 0 1
0 1 1 1 0 1 0 0 0 0 0 1 0 0 1 0 1 1 1 1 0 1 1
0 0 1 1 1 0 1 0 0 0 0 0 1 0 0 1 0 1 1 1 1 0 1 1 1
1 0 0 1 1 1 0 1 0 0 0 0 0 1 0 0 1 0 1 1 1 1 0 1 1 1 0
1 1 0 0 1 1 1 0 1 0 0 0 0 0 1 0 0 1 0 1 1 1 1 0 1 1 1 0 1
1', '0', '0', '0', '0', '0', '0', '1', '0', '1', '1', '1', '0', '0', '1', '1', '0', '1', '0', '1', '0', '1', '0', '1'

```

4. Arbol de Derivaciones, Balanceo de paréntesis

Programa:l1.py | Texto:Datosl1.txt

4.1. Descripción del problema

En este se pondrá a prueba una cadena (ingresada o generada) con las reglas de producción:

$$B \rightarrow (RB \mid e R \rightarrow) \mid (RR$$

4.2. Código

```

#ARBOL DE DERIVACIONES
ar = open("Datosl1.txt", "w")

from random import randint
print("Bienvenido al arbol de Derivaciones:")
op = input("Que modo desea utilizar? 1)Manual 2)Automatico")

if op == "1":
    print("Modo manual, ingrese una cadena compuesta por '(' y ')'")

```

```

        cad = input() #cadena a evaluar
        ar.write(cad)

else:
    print("Modo automatico")
    ar.write("\n"+"Modo automatico")
    cad=""
    largo=randint(1,4)#rand()%1001
    for i in range(0,largo+1):
        rand1=randint(0,1)
        if(rand1==1):
            cad=cad+')'
        else:
            cad=cad+'('

    print(cad)
    ar.write("\n"+cad)

final = "B"
resultado = ""
print("")
ar.write("\n"+"")

longitud = len(cad)
print("B")
ar.write("\n"+"B")

for i in range(0,longitud):
    #print(i)
    if(final[0] == "B"):

        if (cad[i] == "("):
            final = "R"+final
            resultado = resultado + "("
            print (resultado+final)
            ar.write("\n"+resultado+final)
            continue

        if(cad[i] == ")"):
            print("Condicion de derivacion no valida")
            ar.write("\n"+"Condicion de derivacion no
valida")

```

```

        final =final+"X"
        break

if(final[0] == "R"):

    if(cad[i]=="("):
        final = "R"+final
        resultado = resultado + "("
        print (resultado+final)
        ar.write("\n"+resultado+final)
        continue

    if(cad[i]==")"):
        final = final[1:len(final)]
        resultado = resultado + ")"
        print (resultado+final)
        ar.write("\n"+resultado+final)
        continue

if(final=="B"):
    print("\n"+resultado)
    print("\nEsta balanceada")
    ar.write("\n"+" \n"+resultado)
    ar.write("\n"+" \nEsta balanceada")

if(final!="B"):
    print("\nNo esta balanceada")
    ar.write("\n"+" \nNo esta balanceada")

```

4.3. Capturas

4.3.1. Consola Éxito

 Símbolo del sistema

```
C:\Users\Ashel\Desktop\PracticasTC>11.py
Bienvenido al arbol de Derivaciones:
¿Qué modo desea utilizar? 1)Manual 2)Automático2
Modo automático
()((
B
(RB
()B
()(RB
()((RRB

No está balanceada

C:\Users\Ashel\Desktop\PracticasTC>
```

4.3.2. Archivo de texto Éxito



Datos11: Bloc de notas

Archivo Edición Formato Ver Ayuda

Modo automático

()((

B

(RB


()B

()(RB

()((RRB

No está balanceada

4.3.3. Consola Fracaso

 Símbolo del sistema

```
C:\Users\Ashel\Desktop\PracticasyTC>11.py
Bienvenido al arbol de Derivaciones:
¿Qué modo desea utilizar? 1)Manual 2)Automático1
Modo manual, ingrese una cadena compuesta por '(' y ')'
((()))

B
(RB
((RRB
(())RB
(())B
(())(RB
(())()B

((()))

Está balanceada
```


4.3.4. Archivo de texto Fracaso



Datos11: Bloc de notas

Archivo

Edición

Formato

Ver

Ayuda

|(())()

B

(RB

((RRB

(()RB

(())B

(())(RB

(())()B

(())()

Está balanceada

5. Maquina de Turing

Programa:12.py | Texto:Datos12.txt

5.1. Descripción del problema

Según las reglas dadas en clase, programar la máquina de Turing. Al analizarla como resultado se puede observar que al meter cadenas con terminación '0' la cadena hace que estas crezcan por un tiempo indeterminado (podría ser que nunca pare). Por otro lado, cuando se reciben cadenas con terminación '1' la máquina las rechaza dado que no existen reglas para este tipo de situación.

5.2. Código

```
#MAQUINA DE TURING

import sys
import time
ar = open("Datos12.txt", "w")
import random
from tkinter import*
from random import randint
arc = open("Datos12.txt", "w")

print("MAQUINA DE TURING:")
op = input("Que modo desea utilizar? 1)Manual 2)Automatico\n")
if op == "1":
    cadena = input("Ingrese la cadena deseada: ")
else:
    rand = random.randrange(10000)
    cadena = str(bin(rand)[2:])
    print ("Cadena generada: ",cadena)
cad = []
for x in cadena:
    cad.append(x)

arc.write(str(cad))
print ("\n")
```

```

#Declaracion interfaz grafica-----
ventana = Tk()
canv = Canvas(ventana,width=1200,height=600)
ventana.geometry("1200x600")
#
canv.pack()
xspeed=5
yspeed=0
tk= Tk()
e = "a" #a son 0's b son 1's
aux1 = 0 #auxiliar para ver los 1 que se van llenando
cuenta = 0
o= Label(ventana,text="Cadena a verificar.\n
Cadena:"+str(cad)).place(x=10,y=10)

cuad= (len(cad)*50)/2 #Tamanio inicial de nuestra barra
#canv.create_rectangle(50, 500, 1150, 550, width=0, fill='black')
#posicion auxiliar para los cuadritos

#-----
try:
    estado = "q0" #estado inicial
    i=0 #Lleva el indice de nuestra cadena
    while(estado != "q5"): #Analizando toda la cinta
        pos=0
        posaux=0
        for x in cad:
            posicua = 600-50*(pos/2)#posicion inidel
            cuadrito actual

            canv.create_rectangle(600-(len(cad)/2)*50
+pos*50,
500, 650-(len(cad)/2)*50+pos*50, 550, width=1,
fill='green')

```

```

        if(pos==i):
            canv.create_rectangle(600-(len(cad)/2)
*50+
pos*50, 500, 650-(len(cad)/2)*50+pos*50,
550, width=1,fill='blue ')
            pos = pos+1

u= Label(ventana ,text=cad ). place (x=600,y=450)
tk.update()
time.sleep(1.5)

print(i)
print(len(cad))
if(i==-1):
    cad.insert(0,"")
    i=0
if(estado=="q0"): #Condiciones q0

    if(cad[i]=="0"):
        estado = "q0"
        cad[i] = "0"
        i= i+1
        print("(q0,0,R)")
        print(cad)
        arc.write("(q0,0,R)\n"+str(cad)+"\n")
        o= Label(ventana ,text="(q0,0,R)"). place
(x=600,y=350)

    elif(cad[i]=="1"):
        estado = "q1"
        cad[i] = "1"
        i=i+1
        print("(q1,1,R)")
        print(cad)
        arc.write("(q1,1,R)\n"+str(cad)+"\n")
        o= Label(ventana ,text="(q1,1,R)"). place
(x=600,y=350)

    elif(cad[i]==""):

```

```

cad[i]="0"
estado = "q3"
i=i-1
print("(q3,0,L)")
print(cad)
arc.write("(q3,0,L)\n"+str(cad)+"\n")
o= Label(ventana, text="(q3,0,L)"). place
(x=600,y=350)

if(estado=="q1"): #Condiciones q1

    if(cad[i]=="0"):
        estado = "q0"
        cad[i] = "1"
        i= i+1
        print("(q0,1,R)")
        print(cad)
        arc.write("(q0,1,R)\n"+str(cad)+"\n")
        o= Label(ventana, text="(q0,1,R)"). place
(x=600,y=350)

    elif(cad[i]=="1"):
        estado = "q2"
        cad[i] = "1"
        i= i+1
        print("(q2,1,R)")
        print(cad)
        arc.write("(q2,1,R)\n"+str(cad)+"\n")
        o= Label(ventana, text="(q2,1,R)"). place
(x=600,y=350)

    elif(i==len(cad)-1):
        print("Cadena no aceptada")
        break

if(estado=="q2"): #Condiciones q2

    if(cad[i]=="0"):

```

```

estado = "q0"
cad[i] = "1"
i= i+1
print("(q0,1,R)")
print(cad)
arc.write("(q0,1,R)\n"+str(cad)+"\n")
o= Label(ventana, text="(q0,1,R)"). place
(x=600,y=350)

elif(cad[i]=="1"):
    estado = "q2"
    cad[i] = "0"
    i= i+1
    print("(q2,0,R)")
    print(cad)
    arc.write("(q2,0,R)\n"+str(cad)+"\n")
    o= Label(ventana, text="(q2,0,R)"). place
(x=600,y=350)

elif(i==len(cad)-1):
    print("Cadena no aceptada")
    break

if(estado=="q3"): #Condiciones q3
    if(cad[i]=="0"):
        estado = "q3"
        cad[i] = "0"
        i= i-1
        print("(q3,0,L)")
        print(cad)
        arc.write("(q3,0,L)\n"+str(cad)+"\n")
        o= Label(ventana, text="(q3,0,L)"). place
(x=600,y=350)

elif(cad[i]=="1"):
    estado = "q3"
    cad[i] = "1"
    i= i-1

```

```

        print("(q3,1,L)")
        print(cad)
        arc.write("(q3,1,L)\n"+str(cad)+"\n")
        o= Label(ventana, text="(q3,1,L)"). place
(x=600,y=350)

        elif(cad[i]==""):
            cad[i]="0"
            estado = "q0"
            i= i+1
            print("(q0,0,R)")
            print(cad)
            arc.write("(q0,0,R)\n"+str(cad)+"\n")
            o= Label(ventana, text="(q0,0,R)"). place
(x=600,y=350)

        if(i==len(cad)):
            cad.append("")


        posaux=posaux+1

    canv.place(x=0,y=0)
    ventana.mainloop()
except:
    print("Cadena no aceptada")

```

5.3. Capturas

5.3.1. Consola Manual con Éxito

 Símbolo del sistema - 12.py

```
C:\Users\Ashel\Desktop\PracticasyTC>12.py
MAQUINA DE TURING:
¿Qué modo desea utilizar? 1)Manual 2)Automático
1
Ingrese la cadena deseada: 1010

0
4
(q1,1,R)
['1', '0', '1', '0']
(q0,1,R)
['1', '1', '1', '0']
2
4
(q1,1,R)
['1', '1', '1', '0']
(q0,1,R)
['1', '1', '1', '1']
4
5
(q3,0,L)
['1', '1', '1', '1', '0']
(q3,1,L)
['1', '1', '1', '1', '0']
```

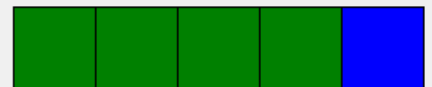

5.3.2. Graficacion parcial Éxito

tk


Cadena a verificar.
Cadena:['1', '0', '1', '0']

(q0,1,R)

1 1 1 1 {}




5.3.3. Archivo de texto Éxito

 Datos12: Bloc de notas

Archivo	Edición	Formato	Ver	Ayuda
['1', '0', '1', '0']				
(q1,1,R)				
['1', '0', '1', '0']				
(q0,1,R)				
['1', '1', '1', '0']				
(q1,1,R)				
['1', '1', '1', '0']				
(q0,1,R)				
['1', '1', '1', '1']				
(q3,0,L)				
['1', '1', '1', '1', '0']				
(q3,1,L)				
['1', '1', '1', '1', '0']				
(q3,1,L)				
['1', '1', '1', '1', '0']				
(q3,1,L)				
['1', '1', '1', '1', '0']				
(q3,1,L)				
['1', '1', '1', '1', '0']				
(q0,0,R)				
['0', '1', '1', '1', '1', '0']				
(q1,1,R)				
['0', '1', '1', '1', '1', '0']				
(q2,1,R)				
['0', '1', '1', '1', '1', '0']				
(q2,0,R)				
['0', '1', '1', '0', '1', '0']				
(q2,0,R)				
['0', '1', '1', '0', '0', '0']				
(q0,1,R)				
['0', '1', '1', '0', '0', '1']				
(q3,0,L)				
['0', '1', '1', '0', '0', '1', '0']				

5.3.4. Consola Automática Fracaso

 Símbolo del sistema - 12.py

```
C:\Users\Ashel\Desktop\PracticasTC>12.py
MAQUINA DE TURING:
¿Qué modo desea utilizar? 1)Manual 2)Automático
2
Cadena generada: 11100111111111

0
13
(q1,1,R)
['1', '1', '1', '0', '0', '1', '1', '1', '1', '1', '1', '1', '1']
(q2,1,R)
['1', '1', '1', '0', '0', '1', '1', '1', '1', '1', '1', '1', '1']
(q2,0,R)
['1', '1', '0', '0', '0', '1', '1', '1', '1', '1', '1', '1', '1']
3
13
(q0,1,R)
['1', '1', '0', '1', '0', '1', '1', '1', '1', '1', '1', '1', '1']
4
13
(q0,0,R)
['1', '1', '0', '1', '0', '1', '1', '1', '1', '1', '1', '1', '1']
5
13
(q1,1,R)
['1', '1', '0', '1', '0', '1', '1', '1', '1', '1', '1', '1', '1']
(q2,1,R)
['1', '1', '0', '1', '0', '1', '1', '1', '1', '1', '1', '1', '1']
```

```
(q2,1,R)
['1', '1', '1', '0', '0', '1', '1', '1', '1', '1', '1', '1', '1']
(q2,0,R)
['1', '1', '0', '0', '0', '1', '1', '1', '1', '1', '1', '1', '1']
3
13
(q0,1,R)
['1', '1', '0', '1', '0', '1', '1', '1', '1', '1', '1', '1', '1']
4
13
(q0,0,R)
['1', '1', '0', '1', '0', '1', '1', '1', '1', '1', '1', '1', '1']
5
13
(q1,1,R)
['1', '1', '0', '1', '0', '1', '1', '1', '1', '1', '1', '1', '1']
(q2,1,R)
['1', '1', '0', '1', '0', '1', '1', '1', '1', '1', '1', '1', '1']
(q2,0,R)
['1', '1', '0', '1', '0', '1', '1', '0', '1', '1', '1', '1', '1']
3
13
(q2,0,R)
['1', '1', '0', '1', '0', '1', '1', '0', '0', '1', '1', '1', '1']
9
13
(q2,0,R)
['1', '1', '0', '1', '0', '1', '1', '0', '0', '0', '1', '1', '1']
10
13
(q2,0,R)
['1', '1', '0', '1', '0', '1', '1', '0', '0', '0', '0', '1', '1']
11
13
(q2,0,R)
['1', '1', '0', '1', '0', '1', '1', '0', '0', '0', '0', '0', '1']
12
13
(q2,0,R)
['1', '1', '0', '1', '0', '1', '1', '0', '0', '0', '0', '0', '0']
13
14
Cadena no aceptada
```

5.3.5. Graficacion parcial Fracaso

tk


Cadena a verificar.
Cadena:['1', '1', '1', '0', '0', '1', '1', '1', '1', '1', '1', '1']

(q2,0,R)

1101011000000{}



5.3.6. Archivo de texto Fracaso

 Datos12: Bloc de notas

Archivo	Edición	Formato	Ver	Ayuda
['1', '1', '1', '0', '0', '1', '1', '1', '1', '1', '1', '1', '1']				
(q1,1,R)				
['1', '1', '1', '0', '0', '1', '1', '1', '1', '1', '1', '1', '1']				
(q2,1,R)				
['1', '1', '1', '0', '0', '1', '1', '1', '1', '1', '1', '1', '1']				
(q2,0,R)				
['1', '1', '0', '0', '0', '1', '1', '1', '1', '1', '1', '1', '1']				
(q0,1,R)				
['1', '1', '0', '1', '0', '1', '1', '1', '1', '1', '1', '1', '1']				
(q0,0,R)				
['1', '1', '0', '1', '0', '1', '1', '1', '1', '1', '1', '1', '1']				
(q1,1,R)				
['1', '1', '0', '1', '0', '1', '1', '1', '1', '1', '1', '1', '1']				
(q2,1,R)				
['1', '1', '0', '1', '0', '1', '1', '1', '1', '1', '1', '1', '1']				
(q2,0,R)				
['1', '1', '0', '1', '0', '1', '1', '0', '1', '1', '1', '1', '1']				
(q2,0,R)				
['1', '1', '0', '1', '0', '1', '1', '0', '0', '1', '1', '1', '1']				
(q2,0,R)				
['1', '1', '0', '1', '0', '1', '1', '0', '0', '0', '1', '1', '1']				
(q2,0,R)				
['1', '1', '0', '1', '0', '1', '1', '0', '0', '0', '0', '1', '1']				
(q2,0,R)				
['1', '1', '0', '1', '0', '1', '1', '0', '0', '0', '0', '0', '1']				
(q2,0,R)				
['1', '1', '0', '1', '0', '1', '1', '0', '0', '0', '0', '0', '0']				