REPORTE
Programa 3 - tablero
Materia : Teoría de la Computación
Grupo : 4CM1
Alumno : Julio Cesar Hernández Reyes
Docente : Juárez Martínez Genaro

---

1. Introducción

En este reporte se explicara como se realizo el programa 3 -protocolo, el cual es un programa que simula un tablero de 4 x 4 cuadrados, y la pieza del rey colocada en el cuadrado 1, a partir de este cuadrante el rey puede realizar movimientos hacia los otros cuadrantes de color negro(black) o rojo(red). La entrada del programa es una secuencia de movimientos de acuerdo al color al que se movera la pieza del rey, r para rojo, n para negro. El programa evalúa la secuencia ingresada con un autómata finito no determinista que evalúa todas los posibles movimientos del rey de acuerdo a la secuencia ingresada. La salida del programa son dos archivos, uno para todos los movimientos posibles que el autómata encuentre, y el archivo de los movimientos ganadores, si es que encuentra.

Resumen del programa:
Entrada: secuencia de caracteres r y b, ejemplo (rrrb")
Salida: Dos archivos de texto:
1-TodosLosMovimientos
2-MovimientosGanadores

Al final del programa se debe poder escoger dos salidas de los movimientos ganadores y mostrarlos en una animación del tablero con el rey moviendose de acuerdo a los movimientos ganadores.

Para la realización del programa se uso Python, por sus funciones de graficacion y animacion

Se uso un IDE en vez de compilar y ejecutar por consola, esto para una mayor facilidad a la hora de corregir errores y algunos detalles del programa.

PyCharm Community Edition
2021.2.1
Aplicación

2. Desarrollo

2.1. Código del programa

Código creado en Python en el IDE de Pycharm:

```
# Programa3
# Tablero 4 * 4

from threading import Thread
import os
import random



def limpiarpantalla():  # Definimos la funci n estableciendo el nombre
    que queramos
    if os.name == "posix":
        os.system("clear")
    elif os.name == "ce" or os.name == "nt" or os.name == "dos":
        os.system("cls")


def verificarsecuencia(secuencia):
    secuencia = secuencia + "."
    todoslosmovimientos(secuencia)
    largo = len(secuencia)
    for i in range(0, largo):
        if secuencia[i] == '1':
            if secuencia[i+1] == '6':
                if secuencia[i+2] == ' ':
                    if secuencia[i+3] == '.':
                        movimientosganadores(secuencia)


def movimientosganadores(cadena):
    archivo = open("MovimientosGanadores.txt", "a")
    archivo.write(cadena)
    archivo.write("\n")
    archivo.close()


def generarsecuencia(conta):
    secuencia = ""
    for i in range(0, conta):
```

```
        i = random.randrange(0, 2)
        if i == 0:
            secuencia = secuencia + 'r'
        elif i == 1:
            secuencia = secuencia + 'b'
    return secuencia

def todoslosmovimientos(cadena):
    archivo = open("TodosLosMovimientos.txt", "a")
    archivo.write(cadena)
    archivo.write("\n")
    archivo.close()

def manual():
    limpiarpantalla()
    print("Manual")
    cadena = input("Favor de ingresar secuencia\n")
    if len(cadena) <= 20:
        automata(cadena, 0, 0, "")
    elif len(cadena) > 20:
        print("ERROR mas de 20 caracteres en la cadena")
    print(" ")
    os.system("pause")


def automatico():
    limpiarpantalla()
    print("Automatico")
    cantcaracteres = random.randrange(0,100)
    secuencia = generarsecuencia(cantcaracteres)
    print("Secuencia generada: " + secuencia)
    automata(secuencia, 0, 0, "")
    os.system("pause")


def menu():

    if os.path.isfile("MovimientosGanadores.txt"):
        os.remove("MovimientosGanadores.txt")
    if os.path.isfile("TodosLosMovimientos.txt"):
        os.remove("TodosLosMovimientos.txt")

    while True:
        limpiarpantalla()

```

```
84          print("Menu Programa 3")
85          print("1.Forma manual")
86          print("2.Forma automatica")
87          print("3.Salir")
88          respuesta = input("Favor de escoger una opci n\n")
89
90          if respuesta == '1':
91              manual()
92          elif respuesta == '2':
93              automatico()
94          elif respuesta == '3':
95
96              break
97          else:
98              print("Ingrese un valor entre 1 y 3")
99      print("Fin del programa")
100
101
102 def automata(cadena, actual, conta, camino):
103     q0 = 0
104     q1 = 1
105     q2 = 2
106     q3 = 3
107     q4 = 4
108     q5 = 5
109     q6 = 6
110     q7 = 7
111     q8 = 8
112     q9 = 9
113     q10 = 10
114     q11 = 11
115     q12 = 12
116     q13 = 13
117     q14 = 14
118     q15 = 15
119     largo = len(cadena)
120
121     camino = camino + str(actual + 1) + " "
122     #print(camino)
123
124     if conta == largo + 1:
125         camino = camino + "."
126         return
127     verificarsecuencia(camino)
128
```

```
129
130
131
132
133     conta = conta + 1
134
135     if conta < largo + 1:
136
137         # estado q0
138         if actual == q0:
139             if cadena[conta-1] == 'r':
140                 h1 = Thread(target=automata, args=(cadena, q1, conta,
                        camino,))
141                 h1.start()
142                 h1.join()
143                 h2 = Thread(target=automata, args=(cadena, q4, conta,
                        camino,))
144                 h2.start()
145                 h2.join()
146
147             if cadena[conta-1] == 'b':
148                 h3 = Thread(target=automata, args=(cadena, q5, conta,
                        camino,))
149                 h3.start()
150                 h3.join()
151
152         # estado q1
153         if actual == q1:
154             if cadena[conta-1] == 'r':
155                 h4 = Thread(target=automata, args=(cadena, q4, conta,
                        camino,))
156                 h4.start()
157                 h4.join()
158                 h5 = Thread(target=automata, args=(cadena, q6, conta,
                        camino,))
159                 h5.start()
160                 h5.join()
161
162             if cadena[conta-1] == 'b':
163                 h6 = Thread(target=automata, args=(cadena, q0, conta,
                        camino,))
164                 h6.start()
165                 h6.join()
166                 h7 = Thread(target=automata, args=(cadena, q2, conta,
                        camino,))
```

```
                    h7.start()
                    h7.join()
                    h8 = Thread(target=automata, args=(cadena, q5, conta,
                        camino,))
                    h8.start()
                    h8.join()

            # estado q2
            if actual == q2:
                if cadena[conta-1] == 'r':
                    h9 = Thread(target=automata, args=(cadena, q1, conta,
                        camino,))
                    h9.start()
                    h9.join()
                    h10 = Thread(target=automata, args=(cadena, q3, conta,
                        camino,))
                    h10.start()
                    h10.join()
                    h11 = Thread(target=automata, args=(cadena, q6, conta,
                        camino,))
                    h11.start()
                    h11.join()

                if cadena[conta-1] == 'b':
                    h12 = Thread(target=automata, args=(cadena, q5, conta,
                        camino,))
                    h12.start()
                    h12.join()
                    h13 = Thread(target=automata, args=(cadena, q7, conta,
                        camino,))
                    h13.start()
                    h13.join()

            # estado q3
            if actual == q3:
                if cadena[conta-1] == 'r':
                    h14 = Thread(target=automata, args=(cadena, q6, conta,
                        camino,))
                    h14.start()
                    h14.join()

                if cadena[conta-1] == 'b':
                    h15 = Thread(target=automata, args=(cadena, q2, conta,
                        camino,))
                    h15.start()
```

```
                h15.join()
                h16 = Thread(target=automata, args=(cadena, q7, conta,
                    camino,))
                h16.start()
                h16.join()

        # estado q4
        if actual == q4:
            if cadena[conta-1] == 'r':
                h17 = Thread(target=automata, args=(cadena, q1, conta,
                    camino,))
                h17.start()
                h17.join()
                h18 = Thread(target=automata, args=(cadena, q9, conta,
                    camino,))
                h18.start()
                h18.join()

            if cadena[conta-1] == 'b':
                h19 = Thread(target=automata, args=(cadena, q0, conta,
                    camino,))
                h19.start()
                h19.join()
                h20 = Thread(target=automata, args=(cadena, q5, conta,
                    camino,))
                h20.start()
                h20.join()
                h21 = Thread(target=automata, args=(cadena, q8, conta,
                    camino,))
                h21.start()
                h21.join()

        # estado q5
        if actual == q5:
            if cadena[conta-1] == 'r':
                h22 = Thread(target=automata, args=(cadena, q1, conta,
                    camino,))
                h22.start()
                h22.join()
                h23 = Thread(target=automata, args=(cadena, q4, conta,
                    camino,))
                h23.start()
                h23.join()
                h24 = Thread(target=automata, args=(cadena, q6, conta,
                    camino,))
```

```
240                     h24.start()
241                     h24.join()
242                     h25 = Thread(target=automata, args=(cadena, q9, conta,
                            camino,))
243                     h25.start()
244                     h25.join()
245                 if cadena[conta-1] == 'b':
246                     h26 = Thread(target=automata, args=(cadena, q0, conta,
                            camino,))
247                     h26.start()
248                     h26.join()
249                     h27 = Thread(target=automata, args=(cadena, q2, conta,
                            camino,))
250                     h27.start()
251                     h27.join()
252                     h28 = Thread(target=automata, args=(cadena, q8, conta,
                            camino,))
253                     h28.start()
254                     h28.join()
255                     h29 = Thread(target=automata, args=(cadena, q10, conta,
                             camino,))
256                     h29.start()
257                     h29.join()
258             # estado q6
259             if actual == q6:
260                 if cadena[conta-1] == 'r':
261                     h30 = Thread(target=automata, args=(cadena, q1, conta,
                            camino,))
262                     h30.start()
263                     h30.join()
264                     h31 = Thread(target=automata, args=(cadena, q3, conta,
                            camino,))
265                     h31.start()
266                     h31.join()
267                     h32 = Thread(target=automata, args=(cadena, q9, conta,
                            camino,))
268                     h32.start()
269                     h32.join()
270                     h33 = Thread(target=automata, args=(cadena, q11, conta,
                             camino,))
271                     h33.start()
272                     h33.join()
273                 if cadena[conta-1] == 'b':
274                     h34 = Thread(target=automata, args=(cadena, q2, conta,
                            camino,))
```

```
                    h34.start()
                    h34.join()
                    h35 = Thread(target=automata, args=(cadena, q5, conta,
                       camino,))
                    h35.start()
                    h35.join()
                    h36 = Thread(target=automata, args=(cadena, q7, conta,
                       camino,))
                    h36.start()
                    h36.join()
                    h37 = Thread(target=automata, args=(cadena, q10, conta,
                        camino,))
                    h37.start()
                    h37.join()

            # estado q7
            if actual == q7:
                if cadena[conta-1] == 'r':
                    h38 = Thread(target=automata, args=(cadena, q3, conta,
                       camino,))
                    h38.start()
                    h38.join()
                    h39 = Thread(target=automata, args=(cadena, q6, conta,
                       camino,))
                    h39.start()
                    h39.join()
                    h40 = Thread(target=automata, args=(cadena, q11, conta,
                        camino,))
                    h40.start()
                    h40.join()
                if cadena[conta-1] == 'b':
                    h41 = Thread(target=automata, args=(cadena, q2, conta,
                       camino,))
                    h41.start()
                    h41.join()
                    h42 = Thread(target=automata, args=(cadena, q10, conta,
                        camino,))
                    h42.start()
                    h42.join()
            # estado q8
            if actual == q8:
                if cadena[conta-1] == 'r':
                    h43 = Thread(target=automata, args=(cadena, q4, conta,
                       camino,))
                    h43.start()
```

```
311                     h43.join()
312                     h44 = Thread(target=automata, args=(cadena, q9, conta,
                           camino,))
313                     h44.start()
314                     h44.join()
315                     h45 = Thread(target=automata, args=(cadena, q12, conta,
                            camino,))
316                     h45.start()
317                     h45.join()
318                 if cadena[conta-1] == 'b':
319                     h46 = Thread(target=automata, args=(cadena, q5, conta,
                           camino,))
320                     h46.start()
321                     h46.join()
322                     h47 = Thread(target=automata, args=(cadena, q13, conta,
                            camino,))
323                     h47.start()
324                     h47.join()
325             # estado q9
326             if actual == q9:
327                 if cadena[conta-1] == 'r':
328                     h48 = Thread(target=automata, args=(cadena, q4, conta,
                           camino,))
329                     h48.start()
330                     h48.join()
331                     h49 = Thread(target=automata, args=(cadena, q6, conta,
                           camino,))
332                     h49.start()
333                     h49.join()
334                     h50 = Thread(target=automata, args=(cadena, q12, conta,
                            camino,))
335                     h50.start()
336                     h50.join()
337                     h51 = Thread(target=automata, args=(cadena, q14, conta,
                            camino,))
338                     h51.start()
339                     h51.join()
340
341                 if cadena[conta-1] == 'b':
342                     h52 = Thread(target=automata, args=(cadena, q5, conta,
                           camino,))
343                     h52.start()
344                     h52.join()
345                     h53 = Thread(target=automata, args=(cadena, q8, conta,
                           camino,))
```

```
346                     h53.start()
347                     h53.join()
348                     h54 = Thread(target=automata, args=(cadena, q10, conta,
                            camino,))
349                     h54.start()
350                     h54.join()
351                     h55 = Thread(target=automata, args=(cadena, q13, conta,
                            camino,))
352                     h55.start()
353                     h55.join()
354
355             # estado q10
356             if actual == q10:
357                 if cadena[conta-1] == 'r':
358                     h56 = Thread(target=automata, args=(cadena, q6, conta,
                           camino,))
359                     h56.start()
360                     h56.join()
361                     h57 = Thread(target=automata, args=(cadena, q9, conta,
                           camino,))
362                     h57.start()
363                     h57.join()
364                     h58 = Thread(target=automata, args=(cadena, q11, conta,
                            camino,))
365                     h58.start()
366                     h58.join()
367                     h59 = Thread(target=automata, args=(cadena, q14, conta,
                            camino,))
368                     h59.start()
369                     h59.join()
370                 if cadena[conta-1] == 'b':
371                     h60 = Thread(target=automata, args=(cadena, q5, conta,
                           camino,))
372                     h60.start()
373                     h60.join()
374                     h61 = Thread(target=automata, args=(cadena, q7, conta,
                           camino,))
375                     h61.start()
376                     h61.join()
377                     h62 = Thread(target=automata, args=(cadena, q13, conta,
                            camino,))
378                     h62.start()
379                     h62.join()
380                     h63 = Thread(target=automata, args=(cadena, q15, conta,
                            camino,))
```

```
                h63.start()
                h63.join()
        # estado q11
        if actual == q11:
            if cadena[conta-1] == 'r':
                h64 = Thread(target=automata, args=(cadena, q6, conta,
                   camino,))
                h64.start()
                h64.join()
                h65 = Thread(target=automata, args=(cadena, q14, conta,
                    camino,))
                h65.start()
                h65.join()

            if cadena[conta-1] == 'b':
                h67 = Thread(target=automata, args=(cadena, q7, conta,
                   camino,))
                h67.start()
                h67.join()
                h68 = Thread(target=automata, args=(cadena, q10, conta,
                    camino,))
                h68.start()
                h68.join()
                h69 = Thread(target=automata, args=(cadena, q15, conta,
                    camino,))
                h69.start()
                h69.join()

        # estado q12
        if actual == q12:
            if cadena[conta-1] == 'r':
                h70 = Thread(target=automata, args=(cadena, q9, conta,
                   camino,))
                h70.start()
                h70.join()

            if cadena[conta-1] == 'b':
                h71 = Thread(target=automata, args=(cadena, q8, conta,
                   camino,))
                h71.start()
                h71.join()
                h72 = Thread(target=automata, args=(cadena, q13, conta,
                    camino,))
                h72.start()
                h72.join()
```

```
418
419         # estado q13
420         if actual == q13:
421             if cadena[conta-1] == 'r':
422                 h73 = Thread(target=automata, args=(cadena, q9, conta,
                        camino,))
423                 h73.start()
424                 h73.join()
425                 h74 = Thread(target=automata, args=(cadena, q12, conta,
                        camino,))
426                 h74.start()
427                 h74.join()
428                 h75 = Thread(target=automata, args=(cadena, q14, conta,
                        camino,))
429                 h75.start()
430                 h75.join()
431
432             if cadena[conta-1] == 'b':
433                 h76 = Thread(target=automata, args=(cadena, q8, conta,
                        camino,))
434                 h76.start()
435                 h76.join()
436                 h77 = Thread(target=automata, args=(cadena, q10, conta,
                        camino,))
437                 h77.start()
438                 h77.join()
439         # estado q14
440         if actual == q14:
441             if cadena[conta-1] == 'r':
442                 h78 = Thread(target=automata, args=(cadena, q9, conta,
                        camino,))
443                 h78.start()
444                 h78.join()
445                 h79 = Thread(target=automata, args=(cadena, q11, conta,
                        camino,))
446                 h79.start()
447                 h79.join()
448             if cadena[conta-1] == 'b':
449                 h80 = Thread(target=automata, args=(cadena, q10, conta,
                        camino,))
450                 h80.start()
451                 h80.join()
452                 h81 = Thread(target=automata, args=(cadena, q13, conta,
                        camino,))
453                 h81.start()
```

```
                    h81.join()
                    h82 = Thread(target=automata, args=(cadena, q15, conta,
                        camino,))
                    h82.start()
                    h82.join()
            # estado q15
            if actual == q15:
                if cadena[conta-1] == 'r':
                    h83 = Thread(target=automata, args=(cadena, q11, conta,
                        camino,))
                    h83.start()
                    h83.join()
                    h84 = Thread(target=automata, args=(cadena, q14, conta,
                        camino,))
                    h84.start()
                    h84.join()

                if cadena[conta-1] == 'b':
                    h85 = Thread(target=automata, args=(cadena, q10, conta,
                        camino,))
                    h85.start()
                    h85.join()


if __name__ == '__main__':
    menu()
```

Explicación: En el código use hilos para simular el autómata finito no determinista el cual puede estar en varios estados a la vez. Cuando se crea una nueva posibilidad para los caminos que se pueden crear se crea un hilo, en total para el automata cree 85 hilos de acuerdo a la tabla de estados cuando es un AFN:

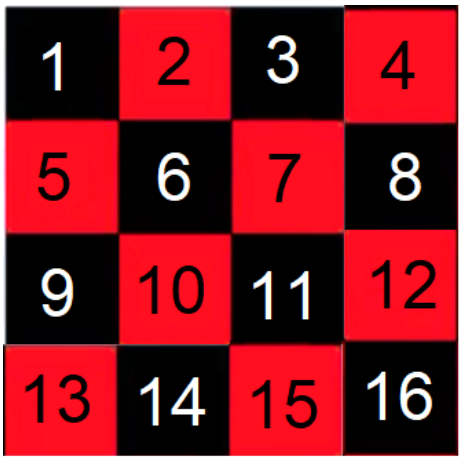
	r	b
->1	2,5	6
2	5,7	1,3,6
3	2,7,4	6,8
4	7	3,8
5	2,10	1,6,9
6	2,5,7,10	1,3,9,11
7	2,4,10,12	3,6,8,11
8	4,7,12	3,11
9	5,10,13	6,14
10	5,7,13,15	6,9,11,14
11	7,10,12,15	6,8,14,16
12	7,15	8,11,16
13	10	9,14
14	10,13 ,15	9,11
15	10,12	11, 14, 16
*16	12,15	11

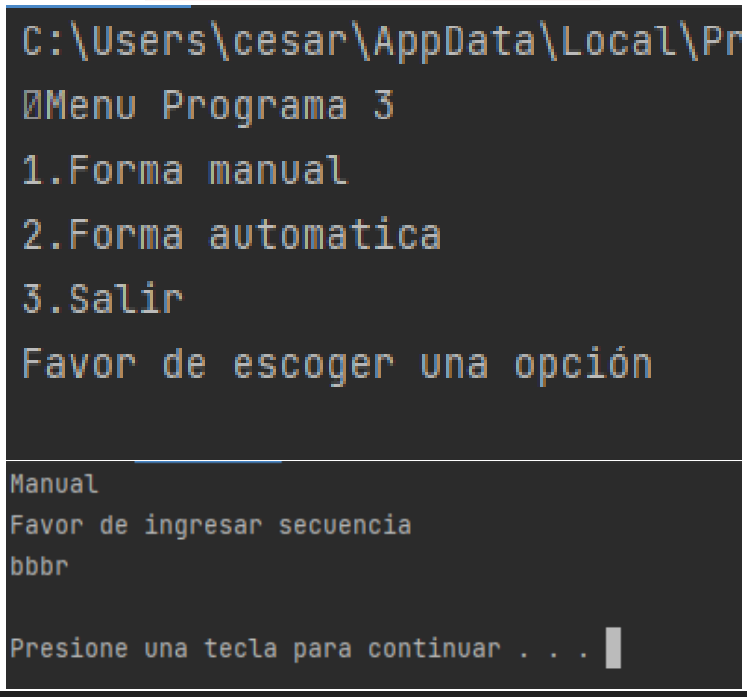
Tabla de estados AFN

AFND	r	b
->q0	q1, q4	q5
q1	q4, q6	q0, q2, q5
q2	q1, q3, q6	q5, q7
q3	q6	q2, q7
q4	q1, q9	q0, q5, q8
q5	q1, q4, q6, q9	q0, q2, q8, q10
q6	q1, q3, q9, q11	q2, q5, q7, q10
q7	q3, q6, q11	q2, q10
q8	q4, q9, q12	q5, q13
q9	q4, q6, q12, q14	q5, q8, q10, q13
q10	q6, q9, q11, q14	q5, q7, q13, *q15
q11	q6, q14	q7, q10, *q15
q12	q9	q8, q13
q13	q9, q12, q14	q8, q10
q14	q9, q11	q10, q13, *q15
*q15	q11, q14	q10

Tabla de estados usando notacion con q0...qn

2.2. Ejecución del programa, en Phyton





```
Automatico
Secuencia generada: rrbbrrrrbrrbrrbbrbrrrrrbbrrrbbbrrbbbbbbbbrbbrbbrbrbbbrrbbbrrbbrrr
```

```
Menu Programa 3
1.Forma manual
2.Forma automatica
3.Salir
Favor de escoger una opción
3
Fin del programa
```

3. Conclusiones

El programa fue muy complicado al grado de que no complete la ultima parte de la animacion solo pude realizar el funcionamiento interno del AFN y generar los archivos. Espero que para los otros programas si los pueda terminar a tiempo.