

Ejercicios sobre máquinas de Turing

Julian Esteban Ballesteros Ortiz

Programa de ingeniería de sistemas y computación, Universidad de Cundinamarca

Lenguajes y Autómatas 801 ISC TRAN

Docente

Ing. Fabio Alejandro Sastoque Rincón

Miércoles 1 de octubre de 2025

Problema 1



```
2025 - 2 > Lenguajes y automatas > 🐍 turing1.py > ...
1  class TuringMachineComplemento:
2      def __init__(self, tape):
3          self.tape = list(tape + "_")
4          self.head = 0
5          self.state = "q0"
6          self.final_states = {"halt"}
7          self.transition_function = {
8              ("q0", "0"): ("1", "R", "q0"),
9              ("q0", "1"): ("0", "R", "q0"),
10             ("q0", "_"): ("_", "R", "halt")
11         }
12
13     def step(self):
14         symbol = self.tape[self.head]
15         key = (self.state, symbol)
16
17         if key not in self.transition_function:
18             self.state = "halt"
19             return False
20
21         new_symbol, direction, new_state = self.transition_function[key]
22         self.tape[self.head] = new_symbol
23
24         if direction == "R":
25             self.head += 1
26         elif direction == "L":
```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS

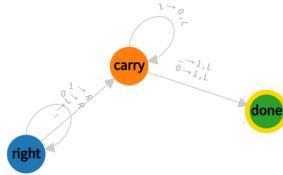
```
PS C:\Users\Julia\Documents\Universidad> & C:/Users/Julia/AppData/Local/Programs/Python/Pyt
py"
Entrada: 10110
Complemento: 01001
```

Problema 2

```

1 input: '111'      # prueba '1011', '011' ...
2 binario: '_'
3 start_state: right
4 table:
5   right:
6     1: R
7     0: R
8     _: {L: carry}
9   carry:
10    1: (write: 0, L)
11    0: (write: 1, L; done)
12    _: (write: 1, L; done)
13 done:
14

```



```

2025 - 2 > Lenguajes y automatas > turing2.py > ...
1  class TuringMachineSucesor:
2      def __init__(self, tape):
3          ("q0", "1"): ("0", "R", "q0"), # 1 + carry -> 0 y seguimos
4          ("q0", "0"): ("1", "R", "halt"), # 0 + 1 -> 1 y terminamos
5          ("q0", "_"): ("1", "R", "halt") # overflow -> agregamos 1
6      }
7
8      def step(self):
9          symbol = self.tape[self.head]
10         key = (self.state, symbol)
11
12         if key not in self.transition_function:
13             self.state = "halt"
14             return False
15
16         new_symbol, direction, new_state = self.transition_function[key]
17         self.tape[self.head] = new_symbol
18
19         if direction == "R":
20             self.head += 1
21         elif direction == "L":
22             self.head = max(0, self.head - 1)
23
24         self.state = new_state
25         return True
26
27     def run(self):
28         while self.state not in self.final_states:
29             self.step()
30         result = "".join(self.tape).strip("_")
31         return result[::-1] # devolvemos el n&uacute;mero en orden normal
32
33
34     # PRUEBA
35     binario = "111"
36     tm = TuringMachineSucesor(binario)
37     print("Entrada:", binario)
38     print("Sucesor:", tm.run())
39
40
41
42
43
44
45

```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS

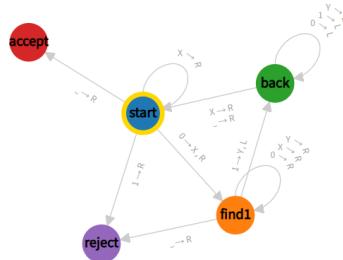
PS C:\Users\Julia\Documents\Universidad> & C:/Users/Julia/AppData/Local/Programs/Python/Python

Entrada: 111

Sucesor: 1000

PS C:\Users\Julia\Documents\Universidad>

Problema 3



```

1 # Acepta cadenas de la forma 0^n1^n
2 # o que 0^n1^n
3 blank:
4 start state: start
5 accept states: [accept]
6 reject states: [reject]
7
8 + table:
9 > start
10 0: (write: X, R; find1)
11  X: (R; start)
12  1: (R; accept)
13  blank: (R; reject)
14
15  find1:
16  0: (R; find1)
17  X: (R; find1)
18  Y: (R; find1)
19  blank: (write: Y, L; back)
20  1: (R; reject)
21
22  back:
23  0: (L; back)
24  1: (L; back)
25  blank: (L; back)
26  X: (R; accept)
27  Y: (R; accept) ↳ Ask AI
28  blank: (R; accept)
29
30  accept:
31  reject:
  
```



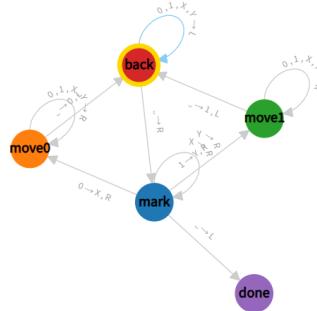
```

2025 - 2 > Lenguajes y automatas > turing3.py > ...
 1  class TuringMachine0n1n:
 2      def __init__(self, tape):
 3          ("q1", "_"): ("_", "R", "reject"),
 4
 5          ("q2", "0"): ("0", "L", "q2"),
 6          ("q2", "_"): ("_", "R", "q0")
 7      }
 8
 9      def step(self):
10          symbol = self.tape[self.head]
11          key = (self.state, symbol)
12
13          if key not in self.transition_function:
14              self.state = "reject"
15              return False
16
17          new_symbol, direction, new_state = self.transition_function[key]
18          self.tape[self.head] = new_symbol
19
20          if direction == "R":
21              self.head += 1
22          elif direction == "L":
23              self.head = max(0, self.head - 1)
24
25          self.state = new_state
26          return True
27
28      def run(self):
29          while self.state not in self.final_states:
30              self.step()
31          return self.state == "accept"
32
33
34      # PRUEBA
35      cadena = "0011"
36      tm = TuringMachine0n1n(cadena)
37      print("Entrada:", cadena)
38      print("¿Pertenece a {0^n1^n}?:", tm.run())
39
40
41
42
43
44
45
46
47
48
49
  
```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS

PS C:\Users\Julia\Documents\Universidad> & C:/Users/Julia/AppData/Local/Programs/Python/Python37-32/turing3.py
Entrada: 0011
¿Pertenece a {0^n1^n}?: False

Problema 4



```

1 # Reverso de una palabra binaria
2 input: "101100"
3 start state: mark
4 table:
5   mark:
6     0: {write: X, R: move0}
7     1: {write: Y, R: move1}
8     X: R
9     Y: R
10    : {L: done}
11
12 move0:
13   0,[0,1,X,Y]: R
14   1: {write: 0, L: back}
15
16 move1:
17   0,[0,1,X,Y]: R
18   1: {write: 1, L: back}
19
20 back:
21   0,[0,1,X,Y]: L
22   1: {R: mark}
23
24 done:
  
```



2025 - 2 > Lenguajes y automatas > turing4.py > ...

```

1 class TuringMachineReverso:
2     def __init__(self, tape):
3         self.tape = tape
4
5     def run(self):
6         return self.tape[::-1]
7
8
9     # PRUEBA
10    cadena = "101100"
11    tm = TuringMachineReverso(cadena)
12    print("Entrada:", cadena)
13    print("Reverso:", tm.run())
14
  
```

PROBLEMS

OUTPUT

DEBUG CONSOLE

TERMINAL

PS C:\Users\Julia\Documents\Universidad> &
 /Julia/Documents/Universidad/2025 - 2/Lenguajes y Automatas> turing4.py
 Entrada: 101100
 Reverso: 001101

