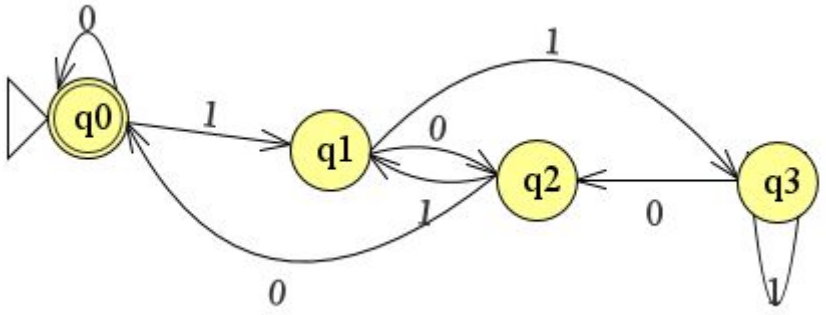
Projeto 1 - Interpretador de Autômatos Determinísticos
ECM253 – Linguagens Formais, Autômatos e Compiladores
Grupo Kali

João Guilherme Martins Jatobá 18.01790-8
André Lebrão Aigner Ribeiro 18.00356-7

Explicação dos autômatos escolhidos para implementar no interpretador em Python:

I-) m1.dfa



Este é um autômato que reconhece qualquer cadeia binária divisível por 4.

```
{
'states' : set([0,1,2,3]),
'initial_state' : 0,
'sigma' : set(['0','1']),
'delta' : { (0,'0'):0, (0,'1'):1,
(1,'0'):2, (1,'1'):3,
(2,'0'):0, (2,'1'):1,
(3,'0'):2, (3,'1'):3,
},
'final_states' : set([0])
}
```

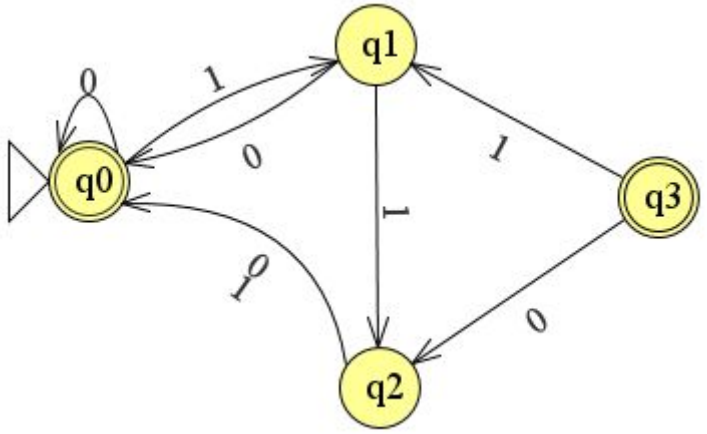
Linguagem:
$L=\{0^*+[(0^*10111^*000^*)^n(0^*1000^*)^m]^* \mid n,m \in N\}$

Cadeias aceitas:
100_2 (4_{10})

100100_2 (36_{10})

Cadeias rejeitadas:
010_2 (2_{10})
00110010_2 (50_{10})

II-) m2.dfa

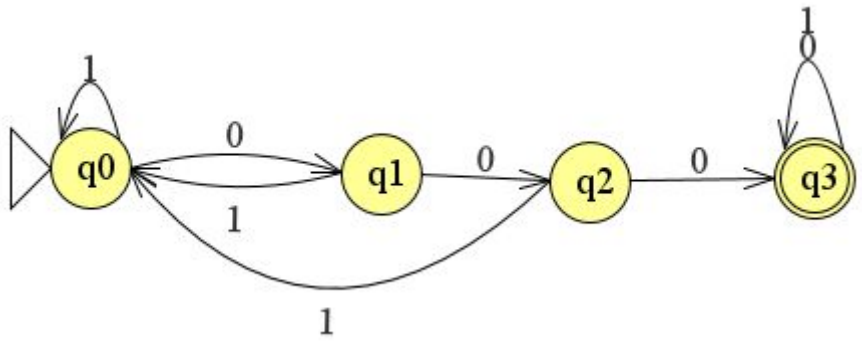


{
'states' : set([0,1,2,3]),
'initial_state' : 0,
'sigma' : set(['0','1']),
'delta' : { (0,'0'):0, (0,'1'):1,
(1,'0'):0, (1,'1'):2,
(2,'0'):0, (2,'1'):0,
(3,'0'):2, (3,'1'):1,
},
'final_states' : set([0, 3])
}
Linguagem:
L = {0* + 0*(10*)* + 0*11(0+1)0*}

Cadeias aceitas:
0010
011110

Cadeias rejeitadas:
1001
011

III-) m3.dfa



Reconhece qualquer qualquer cadeia que contenha uma sequência de 000.

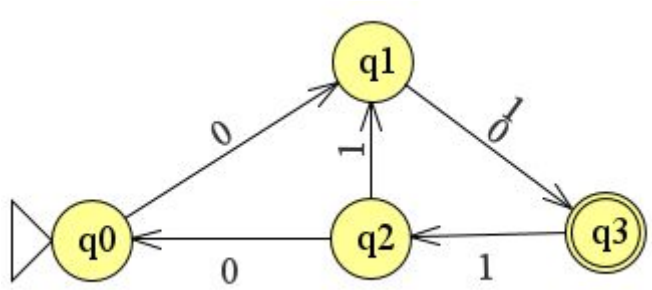
```
{
'states' : set([0,1,2,3]),
'initial_state' : 0,
'sigma' : set(['0','1']),
'delta' : { (0,'0'):1, (0,'1'):0,
(1,'0'):2, (1,'1'):0,
(2,'0'):3, (2,'1'):0,
(3,'0'):3, (3,'1'):3,
},
'final_states' : set([3])
}
```

Linguagem:
L={(01)*000(01)*}

Cadeias aceitas:
00010110011
10111001001000

Cadeias rejeitadas:
1100110101100
1101110

IV-) m4.dfa



{
'states' : set([0,1,2,3]),
'initial_state' : 0,
'sigma' : set(['0','1']),
'delta' : { (0,'0'):1, (0,'1'):'',
(1,'0'):3, (1,'1'):3,
(2,'0'):0, (2,'1'):1,
(3,'0'):'', (3,'1'):2,
},
'final_states' : set([3])
}

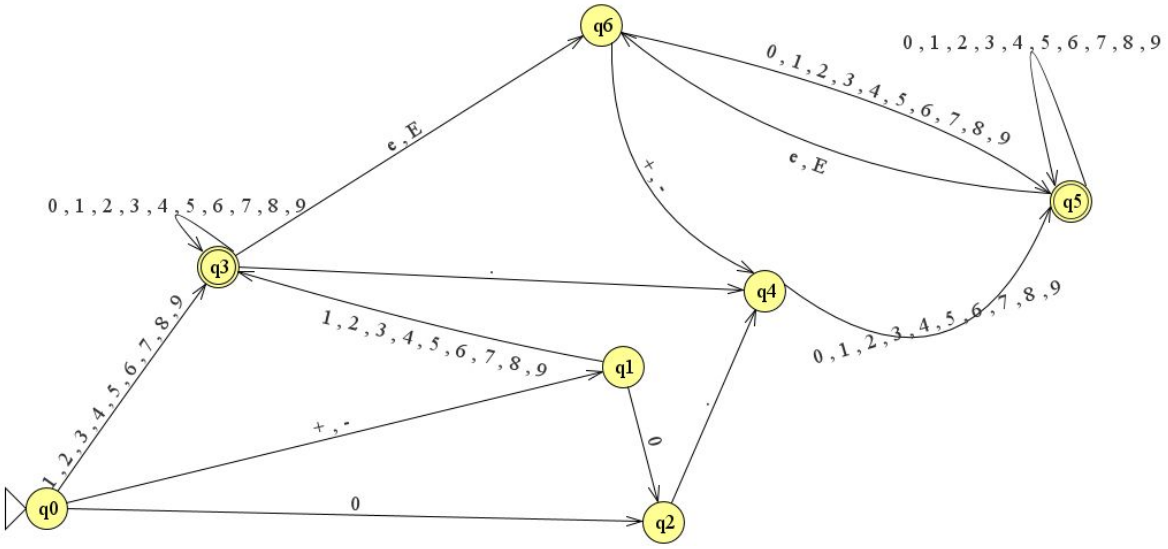
Linguagem:
$L=\{0(0+1) + 0[(0+1)11(0+1)]^n \mid n \in N\}$

Cadeias aceitas:
01110
011000

Cadeias rejeitadas:
01100
11011

Teste obrigatório:

Teste obrigatório: elaborar um autômato finito determinístico que reconheça números reais tais como 57, -123.89, 6.02E23, 0.33, -9.21e-21. Incluir um diagrama de estados para esse autômato.




```
{
'states' : set([0,1,2,3,4,5,6]),
'initial_state' : 0,
'sigma' : set(['0','1','.','+','-','2','3','4','5','6','7','8','9','e','E']),
'delta' : { (0,'0'):2,
(0,'1'):3,(0,'2'):3,(0,'3'):3,(0,'4'):3,(0,'5'):3,(0,'6'):3,(0,'7'):3,(0,'8'):3,(0,'9'):3,(0,'-'):1,(0,'+'
):1,
(1,'0'):2, (1,'1'):3, (1,'2'):3, (1,'3'):3, (1,'4'):3, (1,'5'):3, (1,'6'):3, (1,'7'):3, (1,'8'):3,
(1,'9'):3,
(2,'.'):4,
(3,'.'):4,(3,'0'):3,(3,'1'):3,(3,'2'):3,(3,'3'):3,(3,'4'):3,(3,'5'):3,(3,'6'):3,(3,'7'):3,(3,'8'):3,(3,'9'
):3,(3,'e'):6,(3,'E'):6,
(4,'0'):5,(4,'1'):5,(4,'2'):5,(4,'3'):5,(4,'4'):5,(4,'5'):5,(4,'6'):5,(4,'7'):5,(4,'8'):5,(4,'9'):5,
(5,'e'):6,(5,'E'):6,(5,'0'):5,(5,'1'):5,(5,'2'):5,(5,'3'):5,(5,'4'):5,(5,'5'):5,(5,'6'):5,(5,'7'):5,(5,'8
'):5,(5,'9'):5,
(6,'0'):5,(6,'1'):5,(6,'2'):5,(6,'3'):5,(6,'4'):5,(6,'5'):5,(6,'6'):5,(6,'7'):5,(6,'8'):5,(6,'9'):5,(6,'-'
):4,(6,'+'):4,
},
'final_states' : set([3,5])
}
```