

AUTOMI (FSM)-Esercizi In Python

Prof. Fischetti Pietro

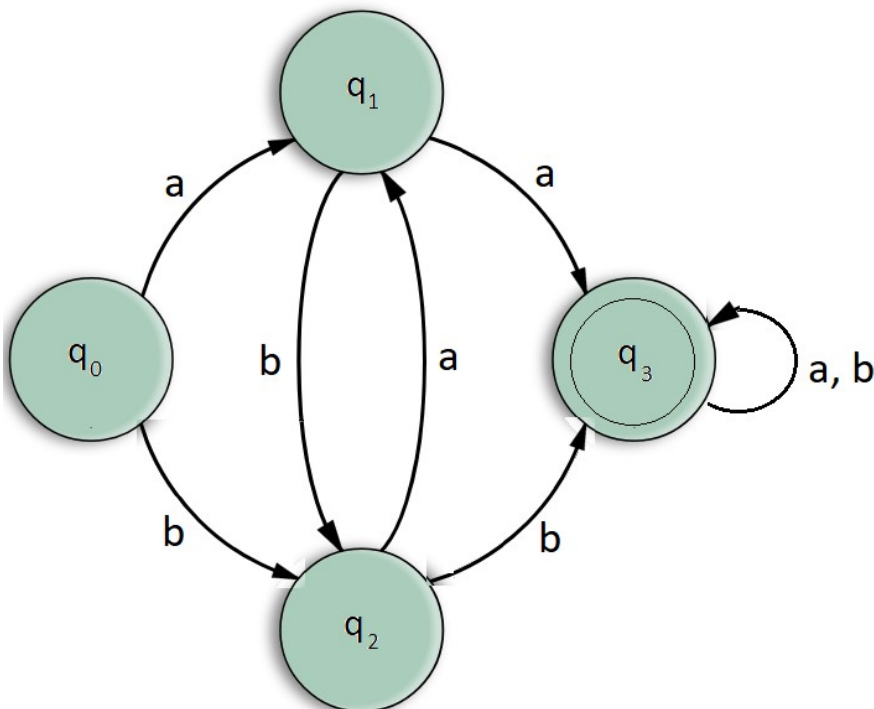
Creare un file di testo Json contenente la descrizione dell'automa, vedi figura seguente:

```
FSM_MAP1="""[
  {'name':'q0',
   'events':[
     {'data':'a','n':'q1','out':'0'},
     {'data':'b','n':'q2','out':'0'}
   ]
 },
.....
```

Dove l'elemento 'name' indica il nome del nodo, la lista 'events' i possibili eventi per quello stato, nell'esempio se si presenta in input allo stato 'q0' il dato 'a' si passa allo stato indicato da 'n' cioè 'q1' nell'esempio, con uscita specificata in 'out'.

ESEMPIO 1

Realizzare un automa che riconosca tutte e sole le stringhe costituite da due 'a' oppure da due 'b' consecutivi:
es. valido: aab, babb, aaa,...
es. non valido: ab, a, babab,...



Soluzione (q₀ - stato iniziale):

```
FSM_MAP1="""[
  {'name':'q0',
```

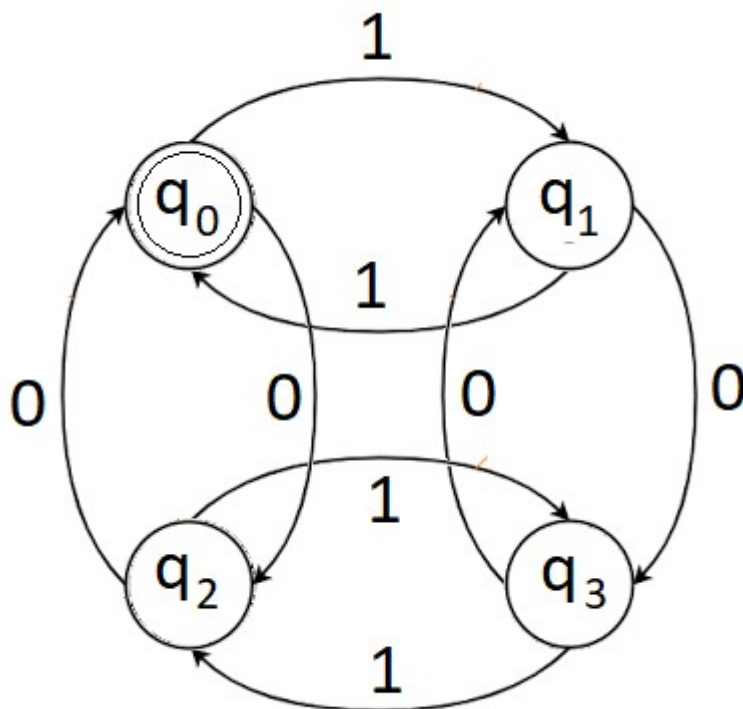
```

'events':[
  {'data':'a','n':'q1','out':'0'},
  {'data':'b','n':'q2','out':'0'}
],
{'name':'q1',
'events':[
  {'data':'a','n':'q3','out':'1'},
  {'data':'b','n':'q2','out':'0'}
]},
{'name':'q2',
'events':[
  {'data':'a','n':'q1','out':'0'},
  {'data':'b','n':'q3','out':'1'}
]},
{'name':'q3',
'events':[
  {'data':'a','n':'q3','out':'0'},
  {'data':'b','n':'q3','out':'0'}
]}
]"""

```

ESEMPIO 2

Riconoscere tutte e sole le stringhe costituite da un numero pari di 0 e da un numero pari di 1, come ad esempio 1001, 10100011.



Soluzione (q_0 - stato iniziale):

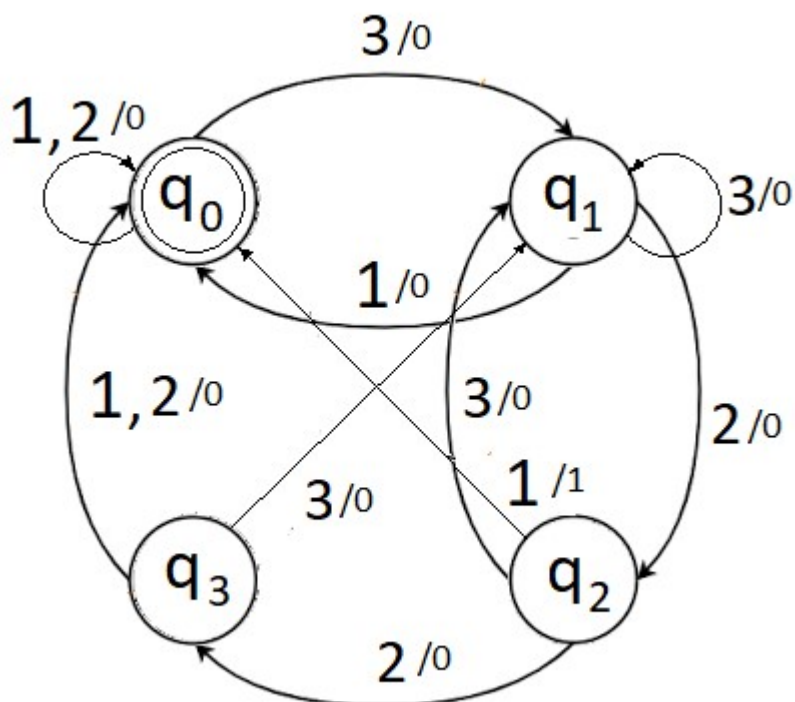
```

FSM_MAP2=""[
  {'name':'q0',
  'events':[
    {'data':'1','n':'q1','out':'0'},
    {'data':'0','n':'q2','out':'0'}
  ]
},
  {'name':'q1',
  'events':[
    {'data':'0','n':'q3','out':'0'},
    {'data':'1','n':'q0','out':'1'}
  ]
},
  {'name':'q2',
  'events':[
    {'data':'0','n':'q0','out':'1'},
    {'data':'1','n':'q3','out':'0'}
  ]
},
  {'name':'q3',
  'events':[
    {'data':'0','n':'q1','out':'0'},
    {'data':'1','n':'q2','out':'0'}
  ]
}
]""

```

ESEMPIO 3

Dati i simboli di input:1,2,3 riconoscere la sequenza '321'.

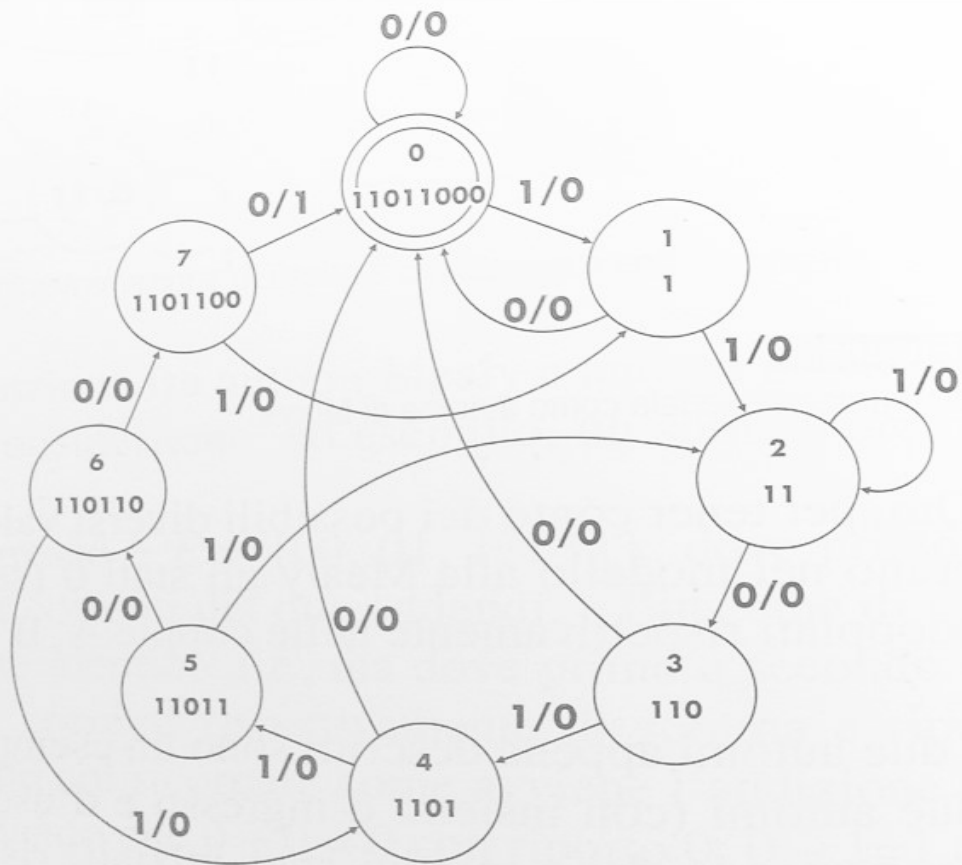


Soluzione (q_0 - stato iniziale):

```
FSM_MAP3=""[
  {'name':'q0',
   'events':[
     {'data':'3','n':'q1','out':'0'},
     {'data':'2','n':'q0','out':'0'},
     {'data':'1','n':'q0','out':'0'}
   ]
  },
  {'name':'q1',
   'events':[
     {'data':'1','n':'q0','out':'0'},
     {'data':'2','n':'q2','out':'0'},
     {'data':'3','n':'q1','out':'0'}
   ]
  },
  {'name':'q2',
   'events':[
     {'data':'3','n':'q1','out':'0'},
     {'data':'2','n':'q3','out':'0'},
     {'data':'1','n':'q0','out':'1'}
   ]
  },
  {'name':'q3',
   'events':[
     {'data':'1','n':'q0','out':'0'},
     {'data':'2','n':'q0','out':'0'},
     {'data':'3','n':'q1','out':'0'}
   ]
  }
]""
```

ESEMPIO 4

Riconoscere la sequenza:1101000



SOLUZIONE

```
#mealy
FSM_MAP4=""[
  {'name':'q0',
  'events':[
    {'data':'0','n':'q0','out':'0'},
    {'data':'1','n':'q1','out':'0'}
  ]
},
{'name':'q1',
'events':[
  {'data':'0','n':'q0','out':'0'},
  {'data':'1','n':'q2','out':'0'}
]
},
{'name':'q2',
'events':[
  {'data':'0','n':'q3','out':'0'},
  {'data':'1','n':'q2','out':'0'}
]
},
{'name':'q3',
'events':[
  {'data':'0','n':'q0','out':'0'},
  {'data':'1','n':'q4','out':'0'}
]
},
],
```

```

{'name':'q4',
 'events':[
   {'data':'0','n':'q0','out':'0'},
   {'data':'1','n':'q5','out':'0'}
 ]
},
{'name':'q5',
 'events':[
   {'data':'0','n':'q6','out':'0'},
   {'data':'1','n':'q2','out':'0'}
 ]
},
{'name':'q6',
 'events':[
   {'data':'0','n':'q7','out':'0'},
   {'data':'1','n':'q4','out':'0'}
 ]
},
{'name':'q7',
 'events':[
   {'data':'0','n':'q0','out':'1'},
   {'data':'1','n':'q1','out':'0'}
 ]
}
]"""

```

Realizzazione in Python: FSM.py

```

import json
class event:
    data=""
    n=""
    def __init__(self,d,n,out=None):
        self.data=d
        self.n=n
        self.out=out
    def __repr__(self):
        return self.data + '->' + self.n + ' (' + self.out + ')'
class state:
    name=""
    events=[]
    def __init__(self,automa):
        self.automa=automa
    def toJSON(self):
        return json.dumps(self, default=lambda o: o.__dict__,
            sort_keys=True, indent=4)
    def fromJSON(self,d):
        self.events=[]
        self.name=d["name"]
        for element in d['events']:
            self.events+=[event(element["data"],element["n"],element["out"])]
    def __repr__(self):

```

```

s=""
for e in self.events:
    s+=str(e)
return self.name + ':' + s + " "
def getStateByName(self,name):
    i=0
    for s in self.automa.states:
        if s.name == name:
            return self.automa.states[i]
        i+=1
    return None
def switch(self,data):
    for s in self.automa.state.events:
        if s.data == data:
            print self.automa.state.name,':',data,
            self.automa.state=self.getStateByName(s.n)
            print '->',self.automa.state.name,'(',s.out,')'
            return
    raise Exception('Unknown Symbol: ' + data )

class automa:
    states=[]
    state=None
    def __init__(self,jsonDict):
        for item in jsonDict:
            x = state(self)
            x.fromJSON(item)
            self.states+=[x]
        self.state=self.states[0]
    def switch(self,data):
        self.state.switch(data)
##### ESECUZIONE #####
##ESEMPIO 1
####json_acceptable_string = FSM_MAP1.replace("","\\")
####stringa="babbab"

##ESEMPIO 2
####json_acceptable_string = FSM_MAP2.replace("","\\")
####stringa="10100011"

##ESEMPIO 3
####json_acceptable_string = FSM_MAP3.replace("","\\")
####stringa="13211332131233212222113213232131322132111"

##ESEMPIO 4
json_acceptable_string = FSM_MAP4.replace("","\\")
stringa="11011000"

d = json.loads(json_acceptable_string)
obj=automa(d)
for c in stringa:
    obj.switch(c)

```