Mujtaba Shahid Faizi

BSCS-5A

#131818

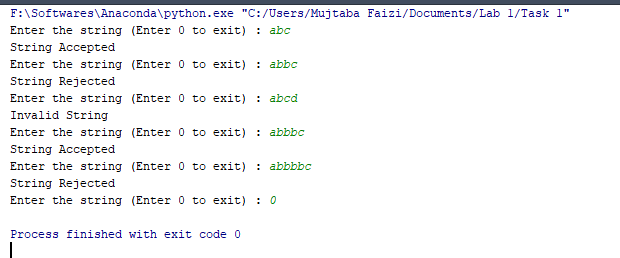
Lab 1 of CC

Task 1:

**Code:**

**class** DFA:  
 current\_state = **None  
  
 def** \_\_init\_\_(self, states, alphabet, transition\_function, start\_state, accept\_states): *#5-tupple* self.states = states  
 self.alphabet = alphabet  
 self.transition\_function = transition\_function  
 self.start\_state = start\_state  
 self.accept\_states = accept\_states  
 self.current\_state = start\_state  
 **return  
  
 def** transition\_to\_state\_with\_input(self, input\_value):  
 **if** ((self.current\_state, input\_value) **not in** self.transition\_function.keys()):  
 self.current\_state = **None  
 return** self.current\_state = self.transition\_function[(self.current\_state, input\_value)]  
 **return  
  
 def** in\_accept\_state(self):  
 **if** self.current\_state **in** accept\_states:  
 print(**"String Accepted"**)  
 **else**:  
 print(**"String Rejected"**)  
  
 **def** go\_to\_initial\_state(self):  
 self.current\_state = self.start\_state  
 **return  
  
 def** run\_with\_input\_list(self, input\_list):  
 self.go\_to\_initial\_state()  
 **for** inp **in** input\_list:  
 self.transition\_to\_state\_with\_input(inp)  
 **continue  
 return** self.in\_accept\_state()  
  
 **pass  
  
 def** validity(self,input\_list): *#checking whether the input string is valid* **for** a **in** input\_list:  
 **if** a **in** alphabet:  
 **continue  
 else**:  
 print(**"Invalid String"**)  
 **return** 0  
 **return** 1  
  
states = {0, 1, 2, 3, 4}  
alphabet = {**'a'**, **'b'**, **'c'**}  
  
tf = dict() *#defining the transition rules*tf[(0, **'a'**)] = 1  
tf[(0, **'b'**)] = 4  
tf[(0, **'c'**)] = 4  
tf[(1, **'a'**)] = 4  
tf[(1, **'b'**)] = 2  
tf[(1, **'c'**)] = 4  
tf[(2, **'a'**)] = 4  
tf[(2, **'b'**)] = 1  
tf[(2, **'c'**)] = 3  
tf[(3, **'a'**)] = 4  
tf[(3, **'b'**)] = 4  
tf[(3, **'c'**)] = 4  
tf[(4, **'a'**)] = 4  
tf[(4, **'b'**)] = 4  
tf[(4, **'c'**)] = 4  
  
start\_state = 0  
accept\_states = {3}  
  
d = DFA(states, alphabet, tf, start\_state, accept\_states)  
  
**for** a **in** range(1000):  
 string = input(**"Enter the string (Enter 0 to exit) : "**)  
 **if** string==**'0'**:  
 exit()  
 val=d.validity(string)  
 **if** val==0:  
 **continue** d.run\_with\_input\_list(string)

**Screenshot:**



Task 2:

**Code:**

**class** DFA:  
 current\_state = **None  
  
 def** \_\_init\_\_(self, states, alphabet, transition\_function, start\_state, accept\_states): *#5-tupple* self.states = states  
 self.alphabet = alphabet  
 self.transition\_function = transition\_function  
 self.start\_state = start\_state  
 self.accept\_states = accept\_states  
 self.current\_state = start\_state  
 **return  
  
 def** transition\_to\_state\_with\_input(self, input\_value):  
 **if** ((self.current\_state, input\_value) **not in** self.transition\_function.keys()):  
 self.current\_state = **None  
 return** self.current\_state = self.transition\_function[(self.current\_state, input\_value)]  
 **return  
  
 def** in\_accept\_state(self):  
 **if** self.current\_state **in** accept\_states:  
 print(**"String Accepted"**)  
 **else**:  
 print(**"String Rejected"**)  
  
 **def** go\_to\_initial\_state(self):  
 self.current\_state = self.start\_state  
 **return  
  
 def** run\_with\_input\_list(self, input\_list):  
 self.go\_to\_initial\_state()  
 **for** inp **in** input\_list:  
 self.transition\_to\_state\_with\_input(inp)  
 **continue  
 return** self.in\_accept\_state()  
  
 **pass  
  
 def** validity(self,input\_list): *#checking whether the input string is valid* **for** a **in** input\_list:  
 **if** a **in** alphabet:  
 **continue  
 else**:  
 print(**"Invalid String"**)  
 **return** 0  
 **return** 1  
  
states = {0, 1, 2, 3}  
alphabet = {**'a'**, **'b'**}  
  
tf = dict() *#defining the transition rules*tf[(0, **'a'**)] = 1  
tf[(0, **'b'**)] = 2  
tf[(1, **'a'**)] = 0  
tf[(1, **'b'**)] = 3  
tf[(2, **'a'**)] = 3  
tf[(2, **'b'**)] = 0  
tf[(3, **'a'**)] = 2  
tf[(3, **'b'**)] = 1  
  
start\_state = 0  
accept\_states = {0}  
  
d = DFA(states, alphabet, tf, start\_state, accept\_states)  
  
**for** a **in** range(1000):  
 string = input(**"Enter the string (Enter 0 to exit) : "**)  
 **if** string==**'0'**:  
 exit()  
 val=d.validity(string)  
 **if** val==0:  
 **continue** d.run\_with\_input\_list(string)

**Screenshot:**

