```
# implementasi NFA
class NonDeterministicFiniteAutomata: # class/objek dengan nama N.F.A yang nanti
akan di panggil di file bahasa2
   def __init__(self,Q,Sigma,delta,S,F):# atribut dari class NFA semua ampir sama
kaya DFA kecuali di state awal yang berupa set karena nfa state awal bsa banyak
       self.states = Q
       self.inputs = Sigma
       self.FTransisi = delta
       self.StateAwals = S
       self.StateAkhirs = F

   def do_delta(self,q,x): # function untuk nemuin next state
       try:
           return self.FTransisi[(q,x)] # ngembaliin value dari key q,x yaitu next
next
       except KeyError:
           return set ({}) # kalo g ada next state kan bakal error jadi biar
ketangkep errornya dan ngembaliin set kosong

   def delta_nfa(self,word):
       P = self.StateAwals # variabel P nyimpen set state awal
       while(word!=""): # iterasi input user selama kata blm abis
           try :
                   Pnew = set({}) #set kosong untuk nyimpen next states
                   # set itu bsa dibilang tipe data structure dmn gaad elemen yang
berulang dalam set
                   for i in P : #iterasi states awal
                       Pnew = Pnew | self.do_delta(i,word[0]) # Pnew di updet
dengan dirinya sendiri dan gabungan (|)
                   print("state ",i, "-->",word[0],'-->
state',self.FTransisi[(i,word[0])])
                   word = word[1:] # katanya di potong satu persatu selagi ber
iterasi

           except KeyError:
               return False
           P = Pnew

       return (P & self.StateAkhirs) != set({}) # cek apakah ada intersection
antara P dan Pnew klo g kosong berarti di acc


   def hat_delta_nfa(self,curr,word):
       #base case
       if word == "":
```

```
            return curr #return base case

        next_states = set({})
        for state in curr:
            next_states = next_states | self.do_delta(state, word[0])
        return self.hat_delta_nfa(next_states,word[1:])
        #rekursif









NL1 = NonDeterministicFiniteAutomata({1,2,3,4,5},
                                     {"0","1"},
                                     {(1,"1"):{2},
                                      (2,"0"):{3},
                                      (3,"1"):{3},(3,"0"):{3,4},
                                      (4,"1"):{5},(4,"0"):{4}
                                     },
                                     {1},
                                     {5})
NL2 = NonDeterministicFiniteAutomata({0,1,2,3,4,5},
                                     {"0","1"},
                                     {(0,"1"):{0},(0,"0"):{1,0},
                                      (1,"1"):{0},(1,"0"):{2},
                                      (2,"1"):{0},(2,"0"):{3},
                                      (3,"1"):{3,4},(3,"0"):{3},
                                      (4,"1"):{5},(4,"0"):{3},
                                      (5,"1"):{5},(5,"0"):{3},
                                      },
                                     {0},
                                     {4}
                                     )

NL3 = NonDeterministicFiniteAutomata({0,1,2,3,4,5},
                                     {"0","1"},
                                     {(0,"0"):{1},(0,"1"):{3},(0,""):{5},
                                      (1,"1"):{1,2},(1,"0"):{1},
                                      (2,"1"):{2},(2,"0"):{1},
                                      (3,"1"):{3},(3,"0"):{3,4},
                                      (4,"1"):{3},(4,"0"):{4},
                                      (5,"1"):{0},
                                      },
```

```
                                    {0},
                                    {2,4}
                                    )

NL4 = NonDeterministicFiniteAutomata({0,1,2,3,4,5,6,7,8,9,10},
                                    {"0","1"},
                                    {(0,"1"):{1},(0,"0"):{6},
                                     (1,"1"):{1},(1,"0"):{2},
                                     (2,"1"):{3},(2,"0"):{5},
                                     (3,"1"):{3},(3,"0"):{4},
                                     (4,"1"):{3},(4,"0"):{4},
                                     (5,"0"):{5},(5,"1"):{1},
                                     (6,"0"):{6},(6,"1"):{6,7},
                                     (7,"0"):{8},
                                     (8,"1"):{9},
                                     (9,"0"):{9,10},(9,"1"):{9},
                                     },
                                    {0},
                                    {3,10}
                                    )
```