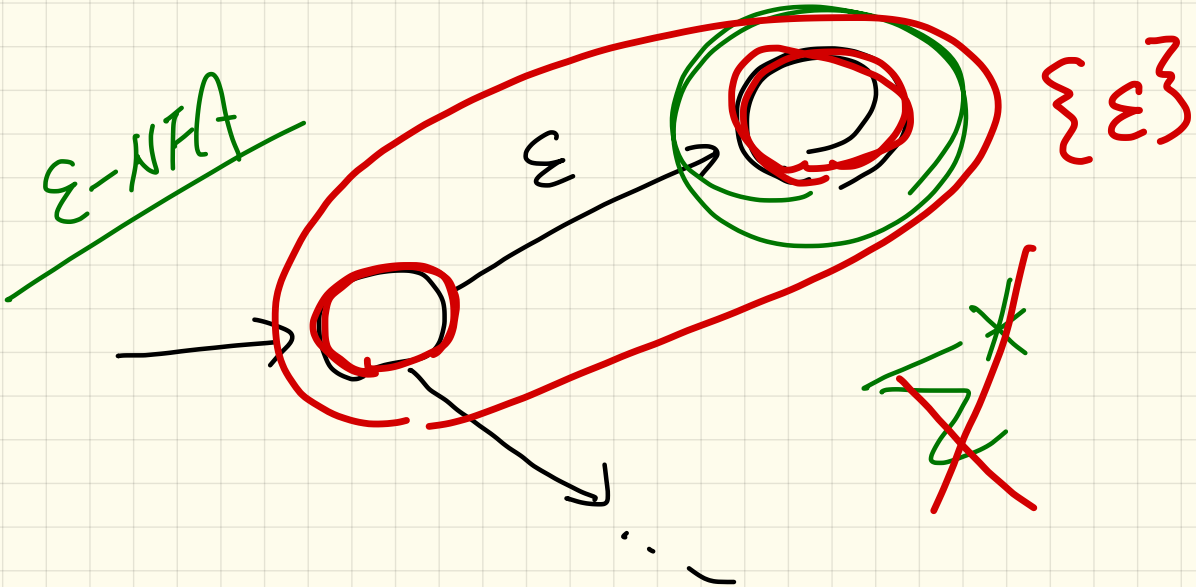
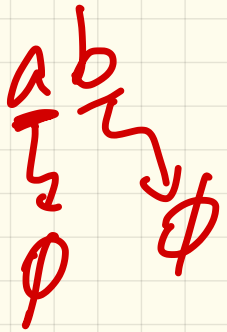
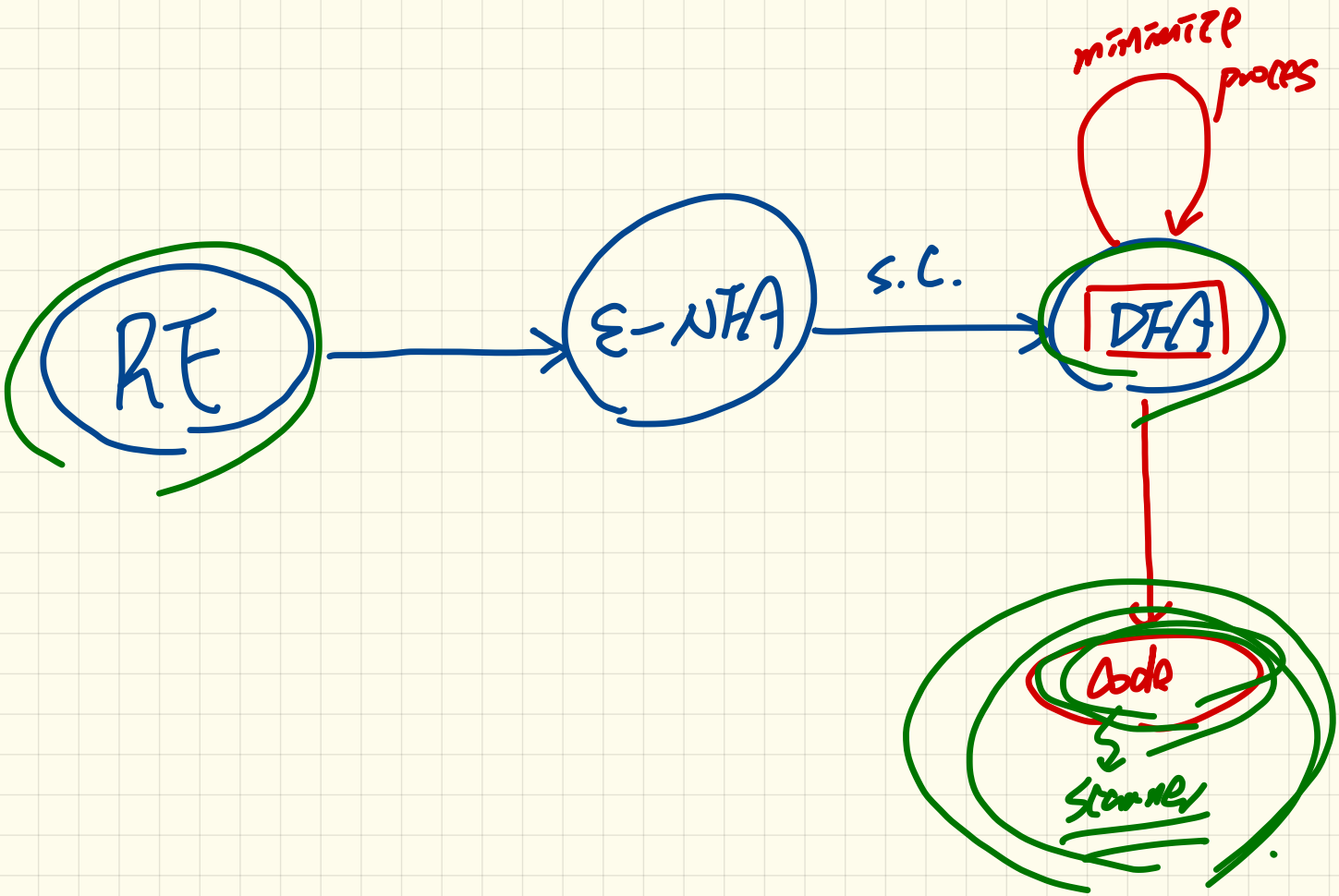
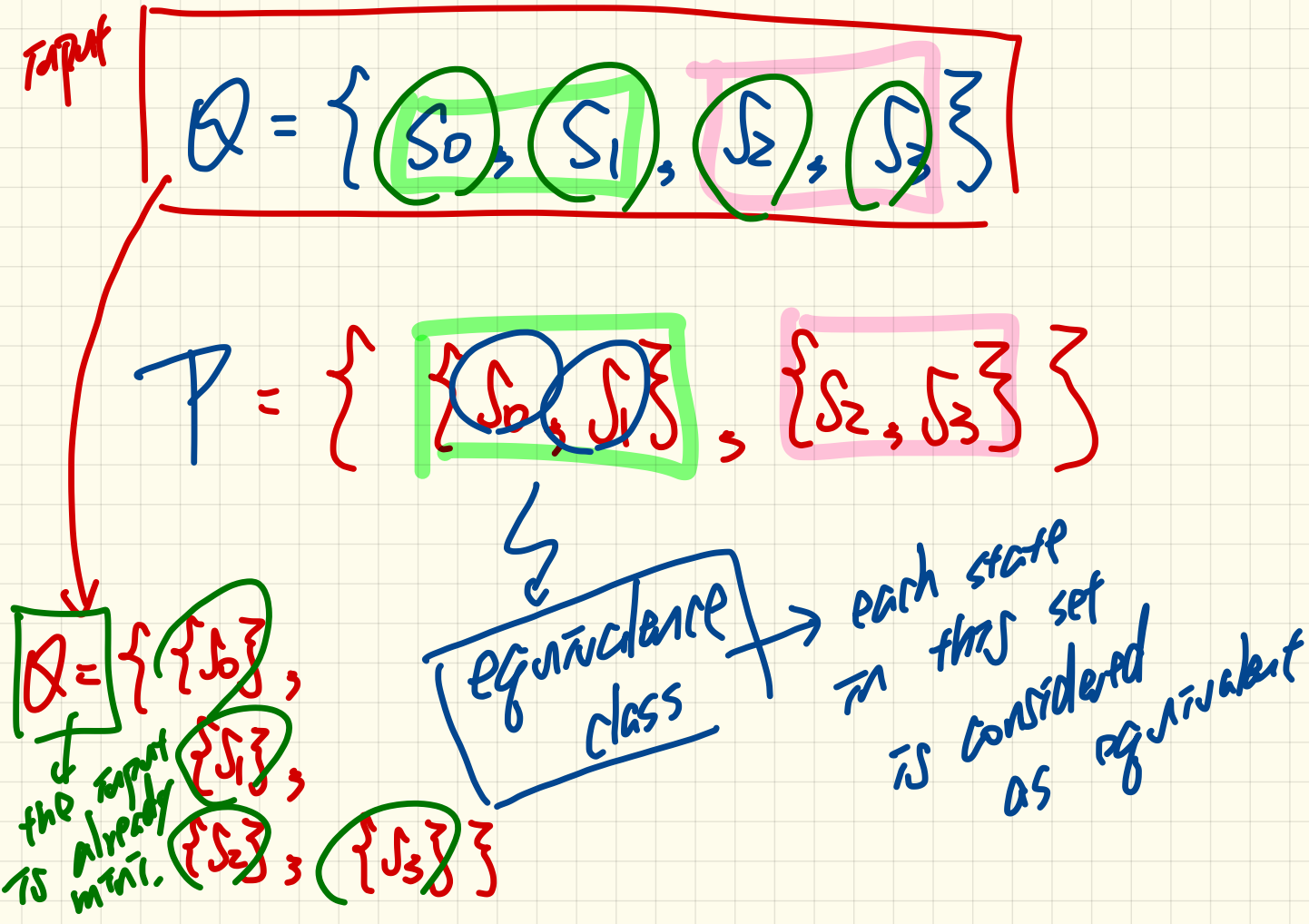
Lecture 5

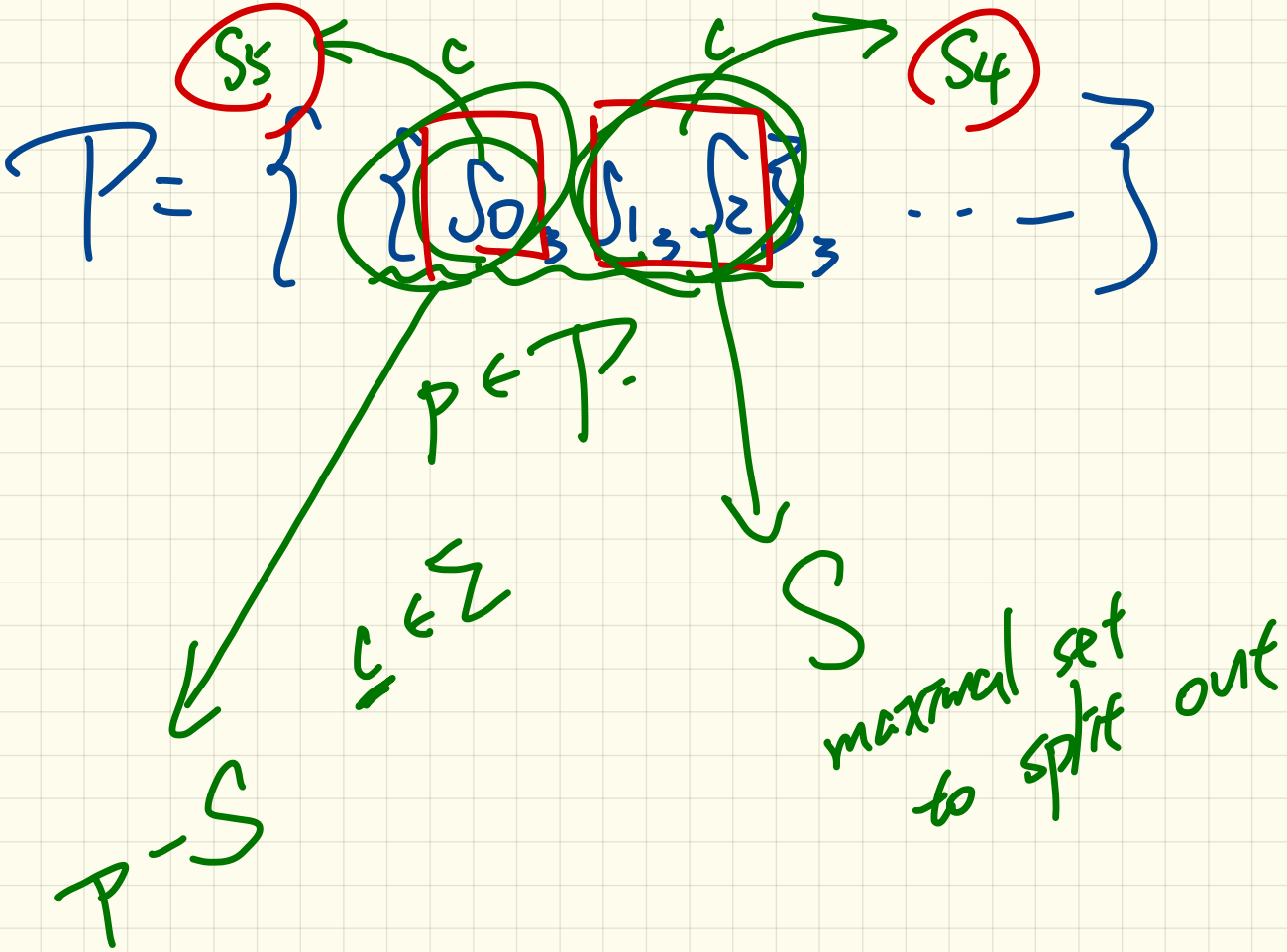
Monday January 20


ε-NFA
ε
{ε}


a
b
ϕ
ϕ


RE
ε-NFA
s.c.
DFA
Code
Scanner


Output
Q = { S0, S1, S2, S3 }
P = { { S0, S1 }, { S2, S3 } }
equivalence class
each state in this set is considered as equivalent
Q = { { S0 }, { S1 }, { S2 }, { S3 } }
the output already initi.


S5
c
c
S4
P =
S0
S1
S2
S3
p ∈ P
c ∈ S
S
maximal set
to split out
P - S

Minimizing DFA: Algorithm

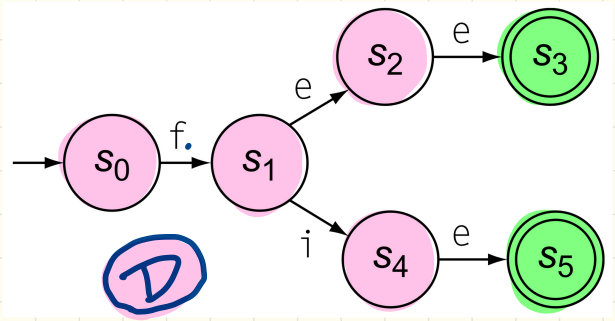
```
ALGORITHM: MinimizeDFAStates
  INPUT: DFA M = (Q, Σ, δ, q0, F)
  OUTPUT: M' s.t. minimum |Q| and equivalent behaviour as M
PROCEDURE:
  P := ∅ /* refined partition so far */
  T := { F, Q − F } /* last refined partition */
  while (P ≠ T):
    P := T
    T := ∅
    for(p ∈ P s.t. |p| > 1):
      find the maximal S ⊆ p s.t. splittable(p, S)
      if S ≠ ∅ then
        T := T ∪ {S, p − S}
      else
        T := T ∪ {p}
      end
```

splittable(p, S) holds iff there is $c \in \Sigma$ s.t.

- Transition c leads all $s \in S$ to states in the ***same partition*** $p1$.
- Transition c leads some $s \in p - S$ to a ***different partition*** $p2$ ($p2 \neq p1$).

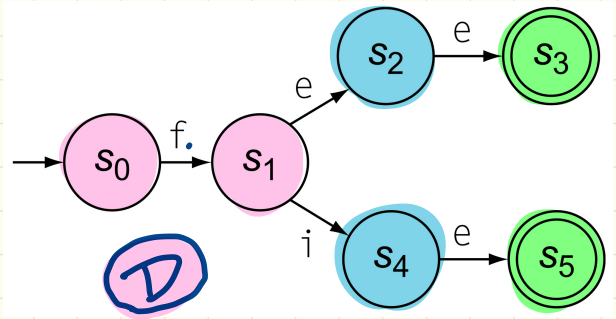
Minimizing DFA: Example (1)

fee | fie



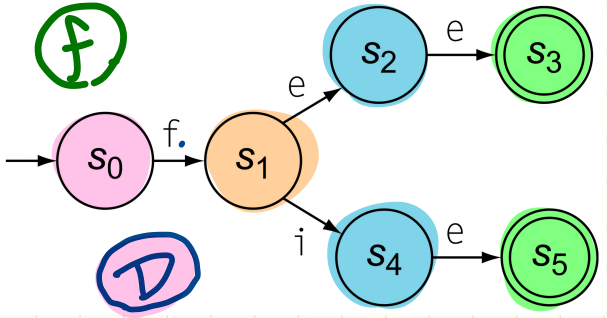
f ✓] ∀s ∈ · s $\xrightarrow{c}$

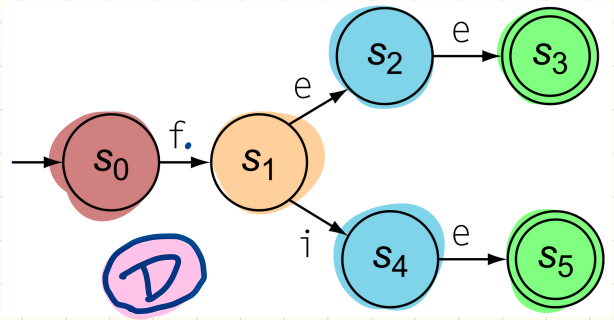
e × { {s0, s1}, {s2, s4}, {s3, s5} }



e × { {s0}, {s1}, {s2, s4}, {s3, s5} }

f



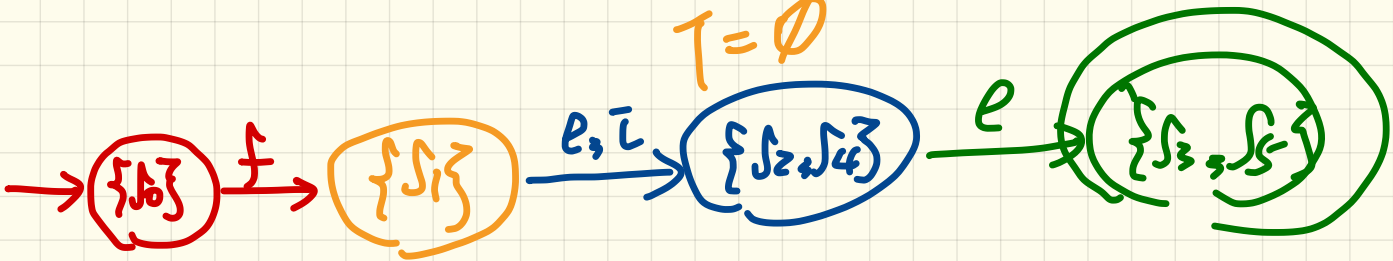


T

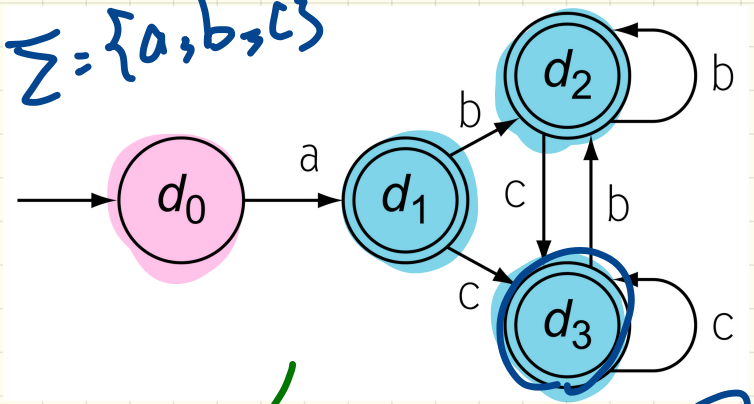
{ {s_0}, {s_1}, {s_2, s_4},
{s_3, s_5} }

P=T

T=∅



Minimizing DFA: Example (2)



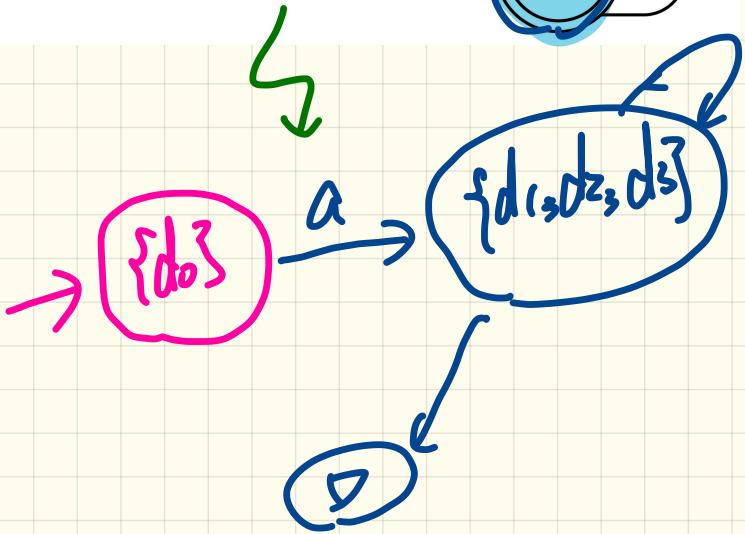
a → all dead state

b

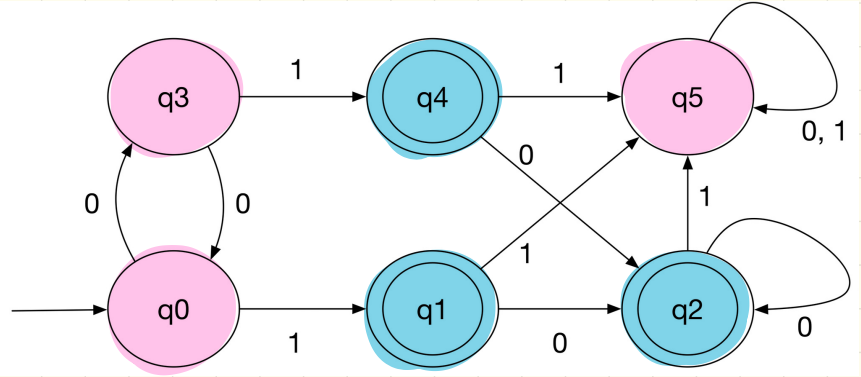
d_1	b →	d_2
d_2	b →	d_2
d_3	b →	d_2

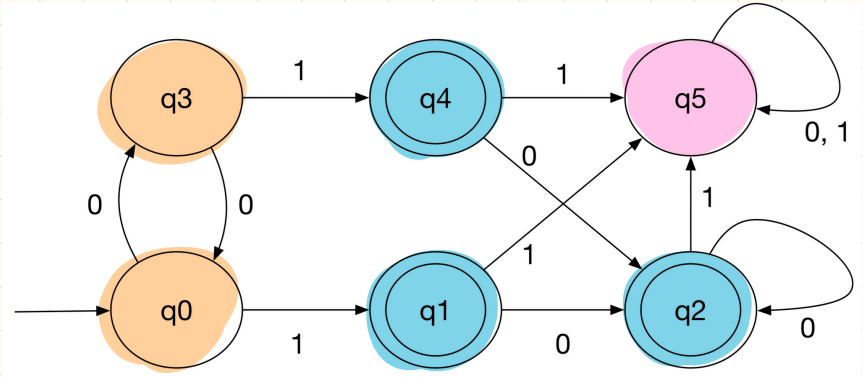
c

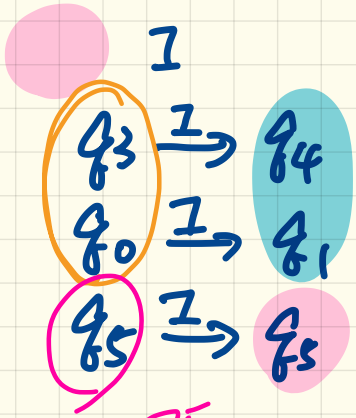
d_1	c →	d_3
d_2	c →	d_3
d_3	c →	d_3

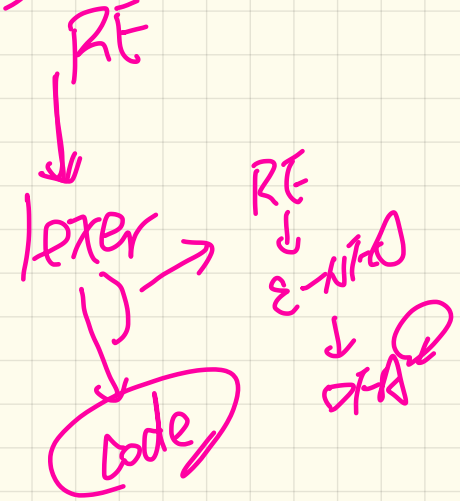


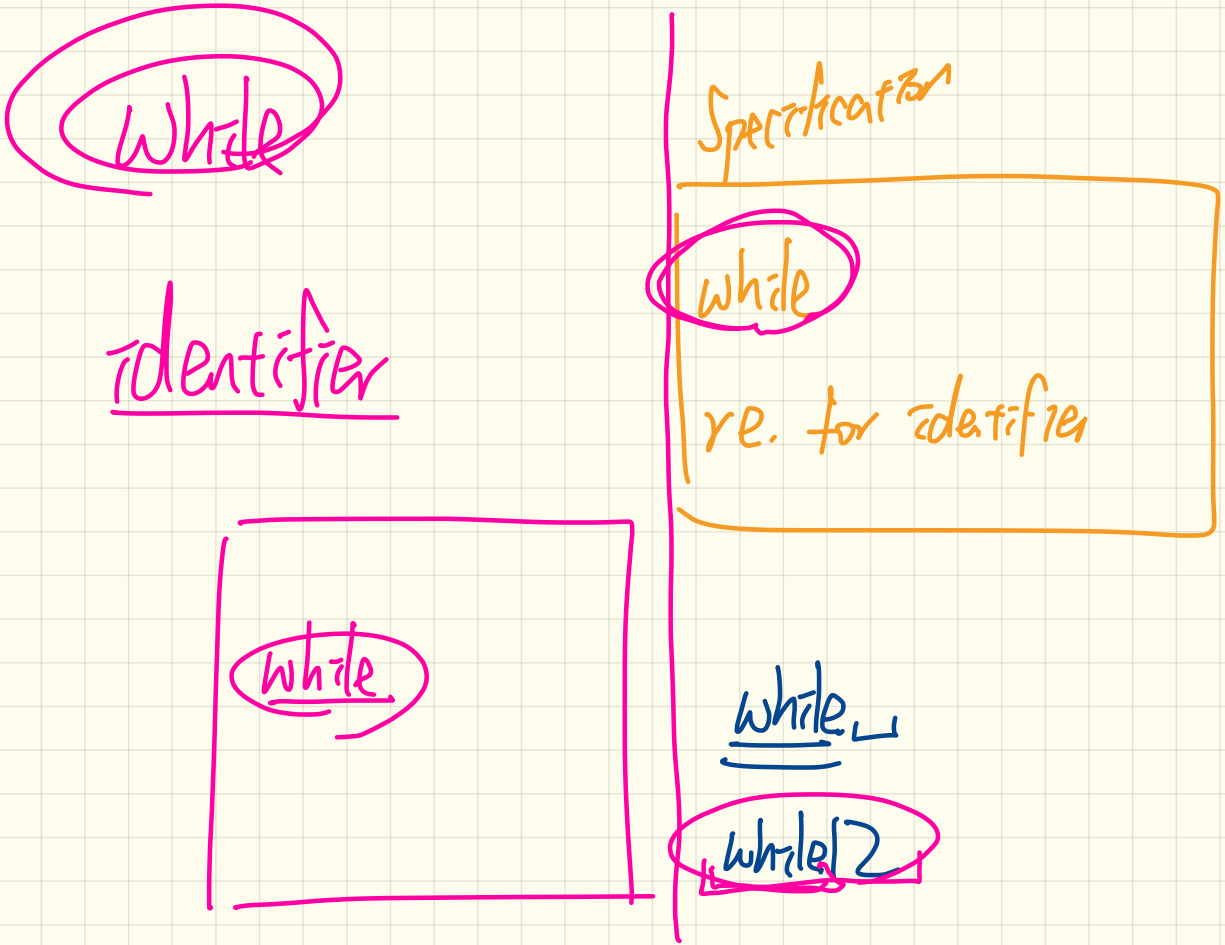
Minimizing DFA: Example (3)

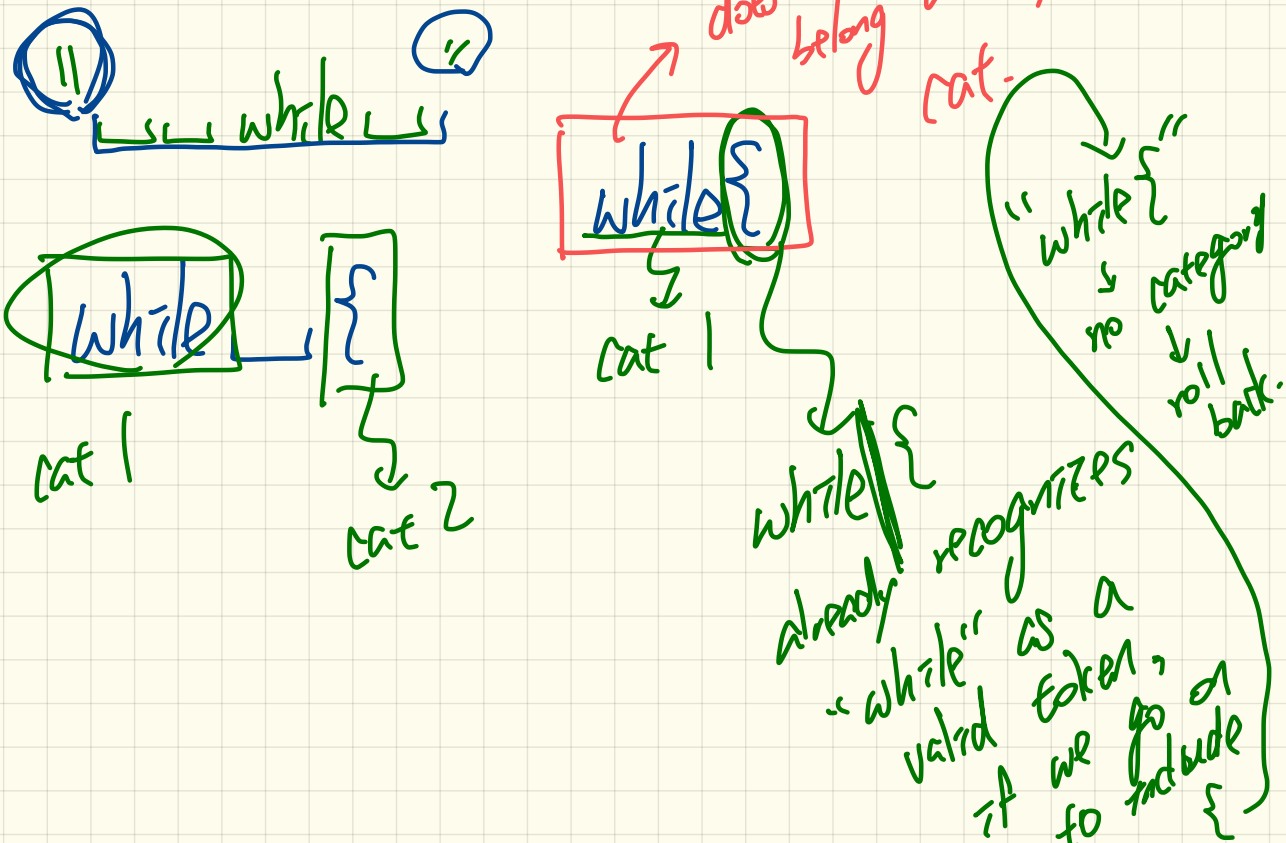









while
identifier
while
Specification
while
re. for identifier
while
while12


"while"
while
while
{
cat 1
cat 2
does not belong to any cat.
while
{
cat 1
while
{
already recognizes "while" as a valid token, if we go on to include
"while"
no category
roll back.