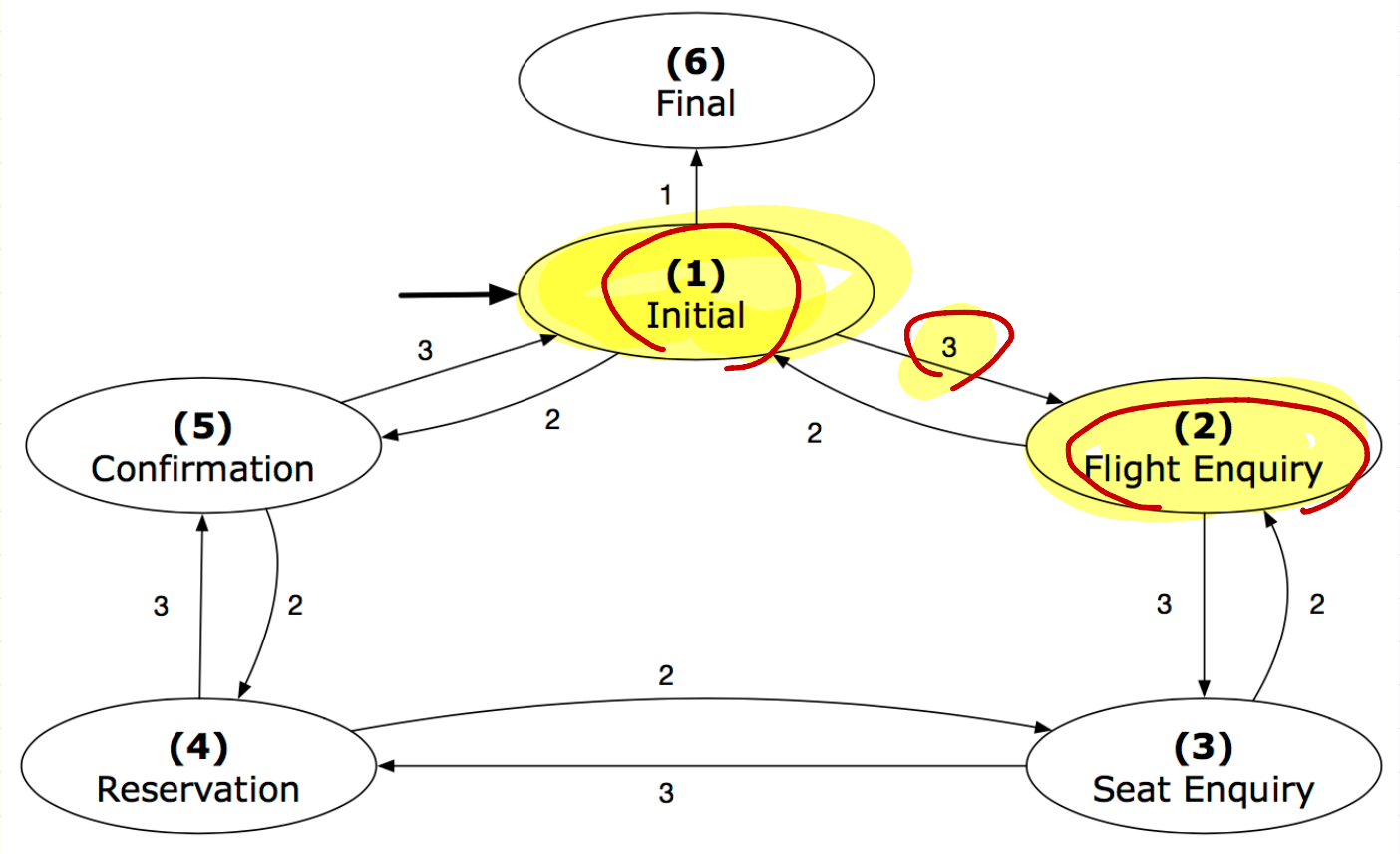
EECS 3311 Winter 2018

Wednesday Feb. 28

Finite State Machine



1st Design

```
1_Initial_panel:
  -- Actions for Label 1.
2_Flight_Enquiry_panel:
  -- Actions for Label 2.
3_Seat_Enquiry_panel:
  -- Actions for Label 3.
4_Reservation_panel:
  -- Actions for Label 4.
5_Confirmation_panel:
  -- Actions for Label 5.
6_Final_panel:
  -- Actions for Label 6.
```

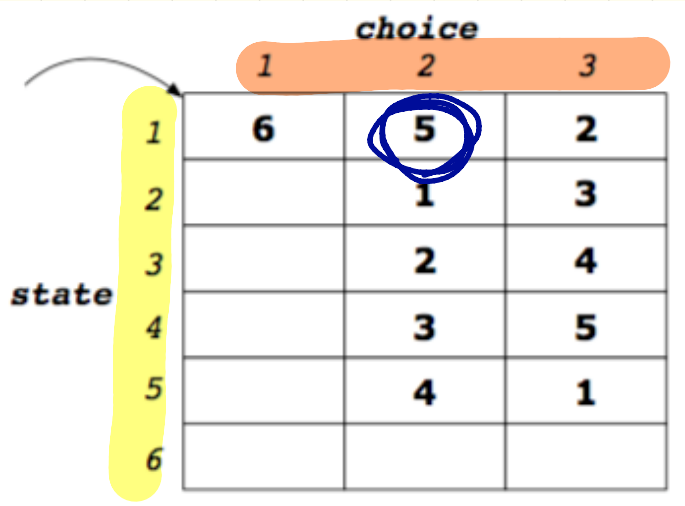
```
3_Seat_Enquiry_panel:
from
  Display Seat Enquiry Panel
until
  not (wrong answer or wrong choice)
do
  Read user's answer for current panel
  Read user's choice C for next step
  if wrong answer or wrong choice then
    Output error messages
  end
end
Process user's answer
case C in
  2: goto 2_Flight_Enquiry_panel
  3: goto 4_Reservation_panel
end
```

State Transition Program

SRC STATE \ CHOICE	1	2	3
1 (Initial)	6	5	2
2 (Flight Enquiry)	–	1	3
3 (Seat Enquiry)	–	2	4
4 (Reservation)	–	3	5
5 (Confirmation)	–	4	1
6 (Final)	–	–	–

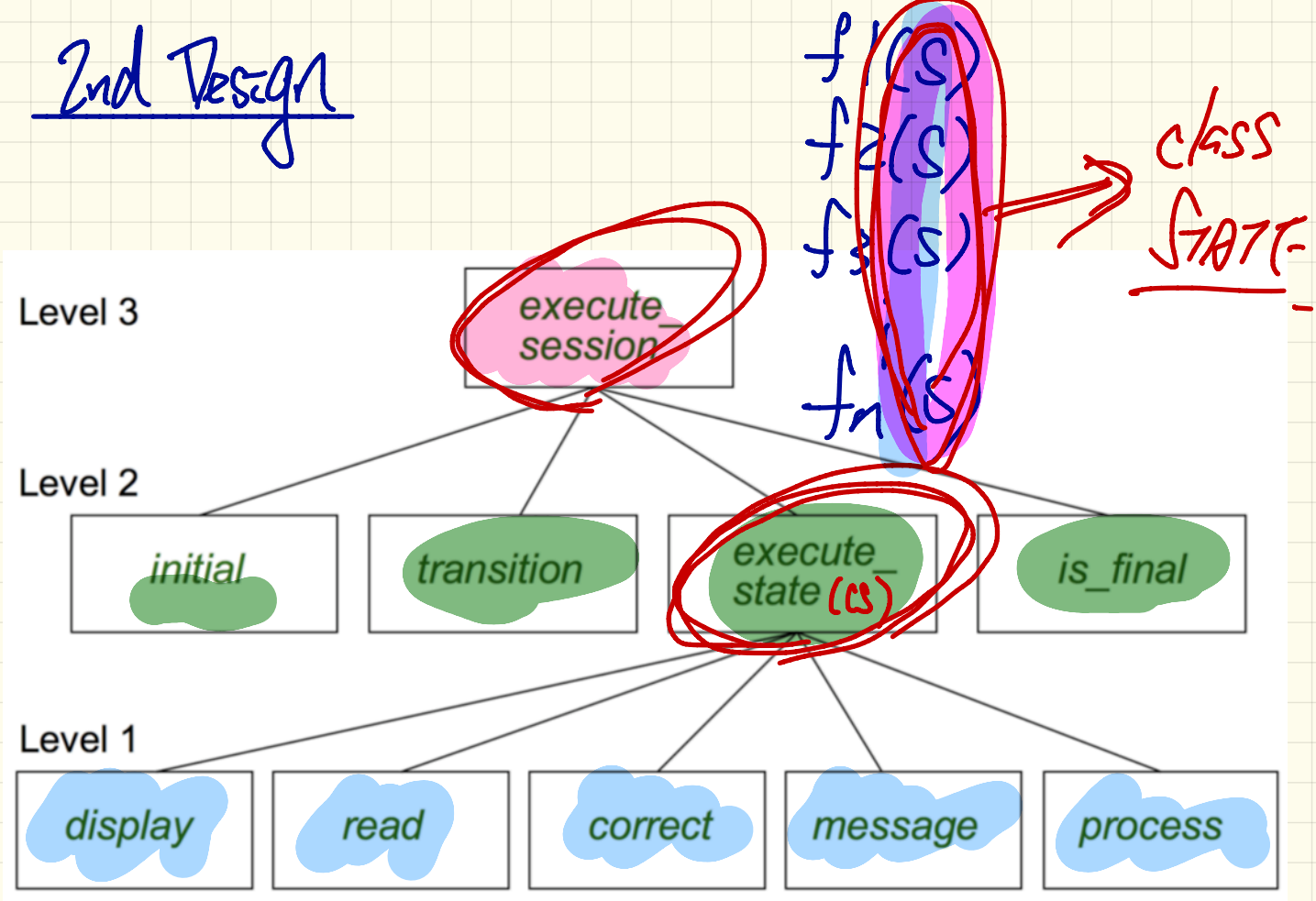
Implementing a transition diagram: 2-D Array

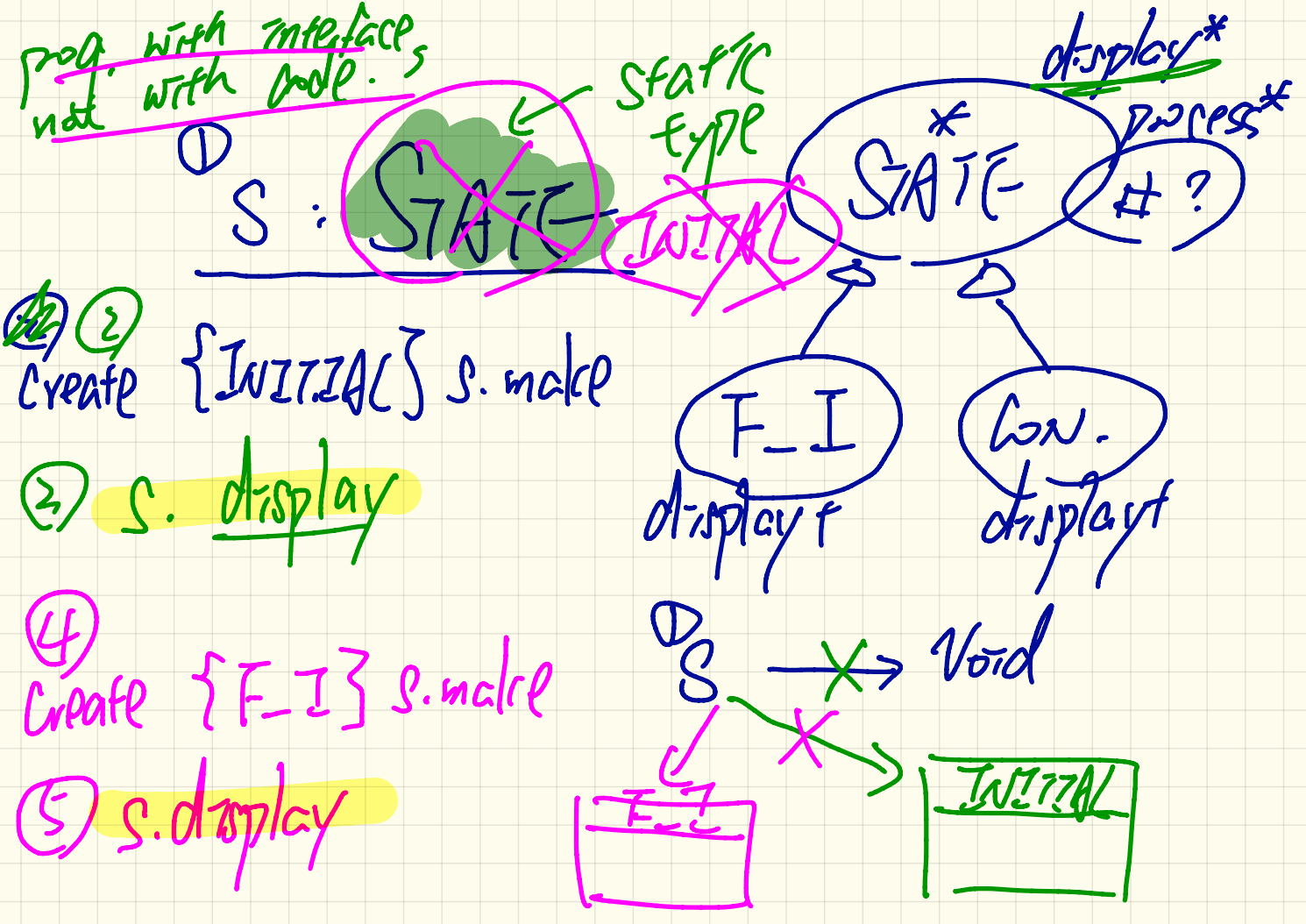
transition[1][2]



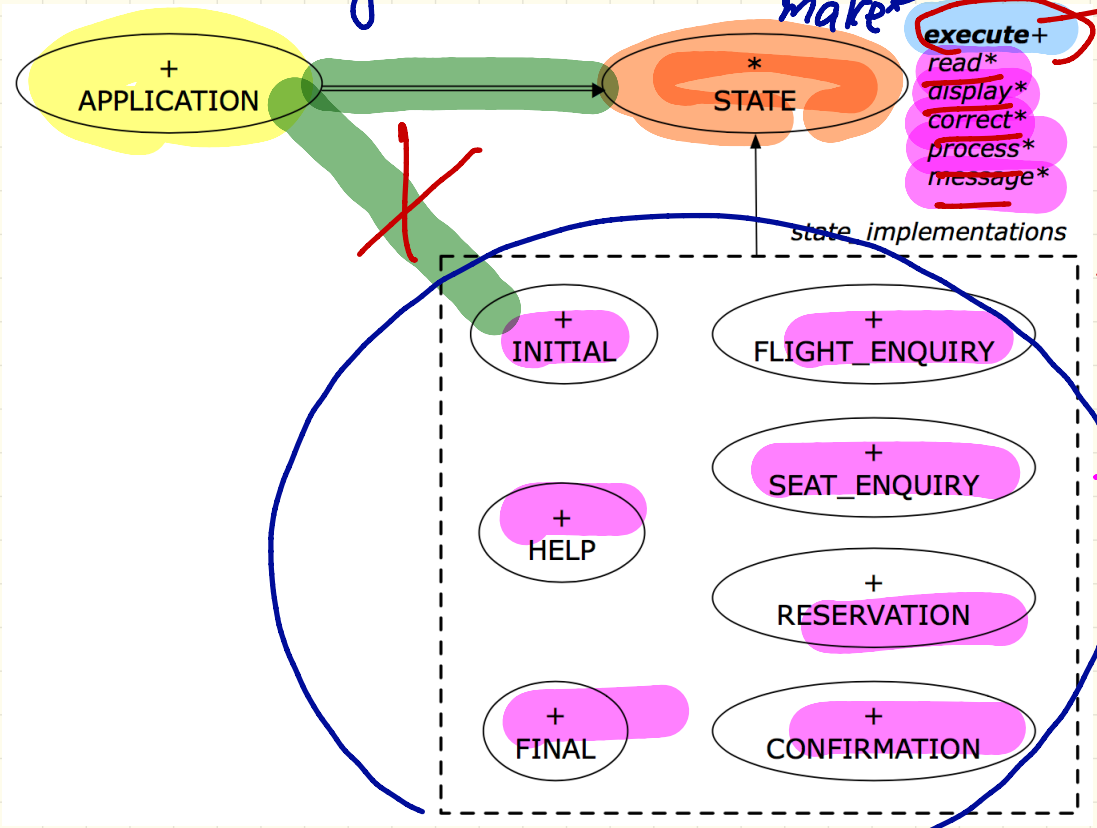
state \ choice	1	2	3
1	6	5	2
2		1	3
3		2	4
4		3	5
5		4	1
6			

2nd Design


f1(s)
f2(s)
f3(s)
fn(s)
class
STATE
Level 3
execute_
session
Level 2
initial
transition
execute_
state (cs)
is_final
Level 1
display
read
correct
message
process


prog. with interface's not with code.
① S : STATIC
static type
INITIAL
STATIC
display*
process*
#?
② Create {INITIAL} S. make
③ S. display
F_I
display f
CON.
display f
④ Create {F_I} S. make
⑤ S. display
① S
Void
F_I
INITIAL

State Design Pattern: Architecture



make*

call all the deferred features, each of which to be imp. in descendant classes.

template pattern

Template Design Pattern

```
deferred class STATE
  read
    -- Read user's inputs
    -- Set 'answer' and 'choice'
    deferred end
  answer: ANSWER
    -- Answer for current state
  choice: INTEGER
    -- Choice for next step
  display
    -- Display current state
    deferred end
  correct: BOOLEAN
    deferred end
  process
    require correct
    deferred end
  message
    require not correct
    deferred end
```

```
  execute
    local
      good: BOOLEAN
    do
      from
      until
        good
      loop
        display
        -- set answer and choice
        read
        good := correct
        if not good then
          message
        end
      end
      process
  end
end
```

template

1. ③ and ④: the same version (template) of execute is called

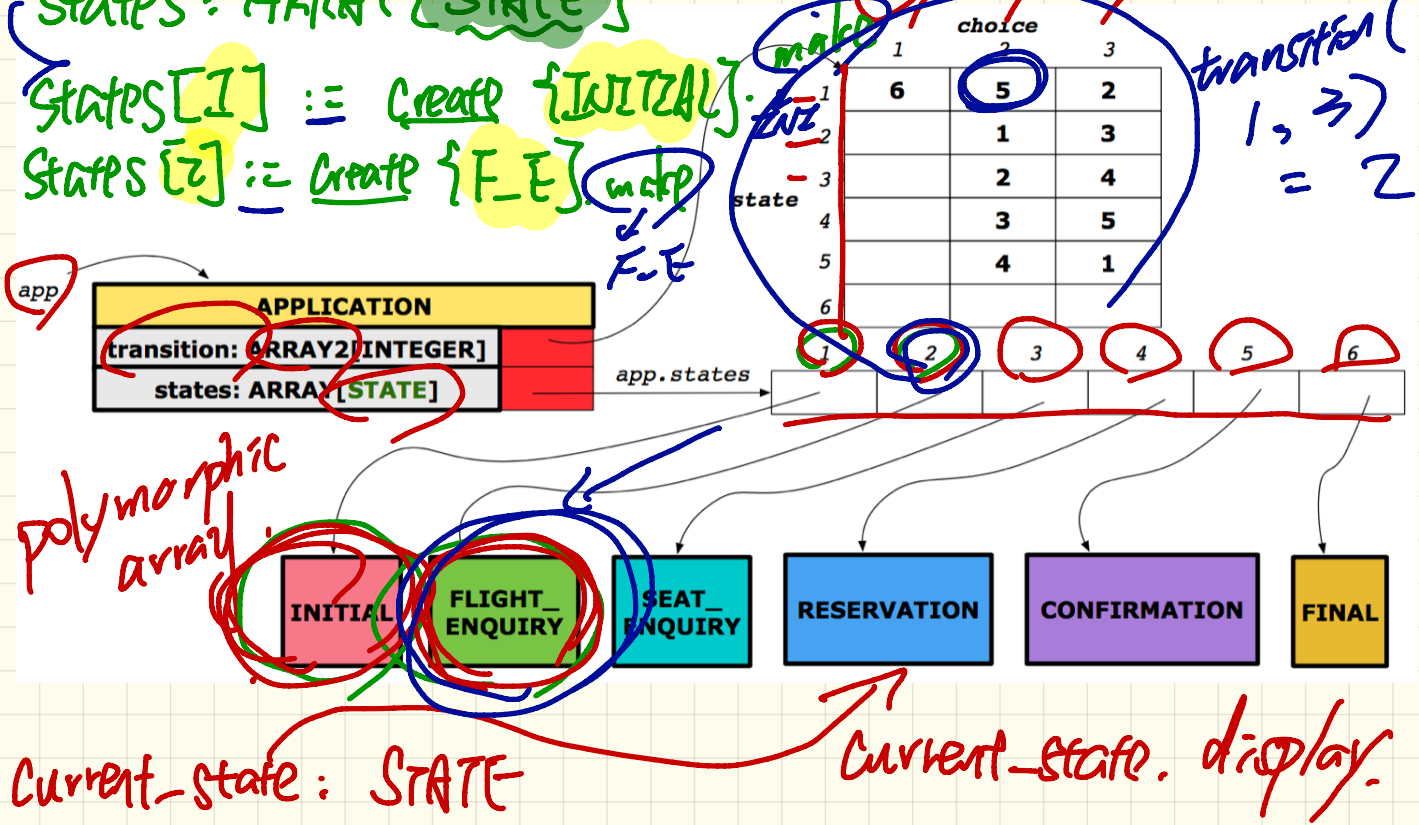
2. ③ and ④ different versions of display, read... will be executed (dynamic binding)

S : STATE

① ② create {INITIAL} S.make
S.execute

③ ④ create {CONFIRM} S.make
S.execute

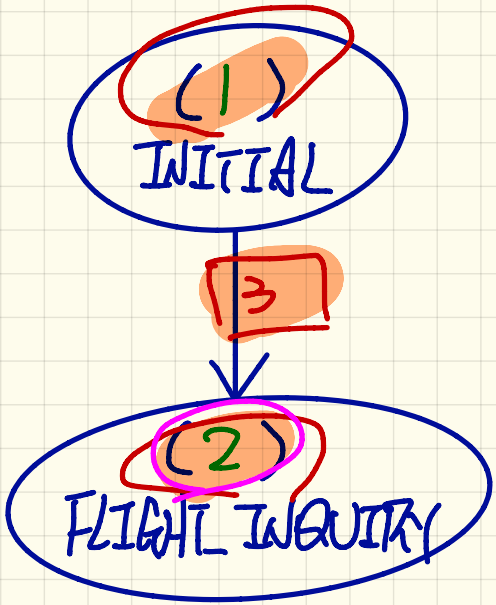
State Pattern: Runtime

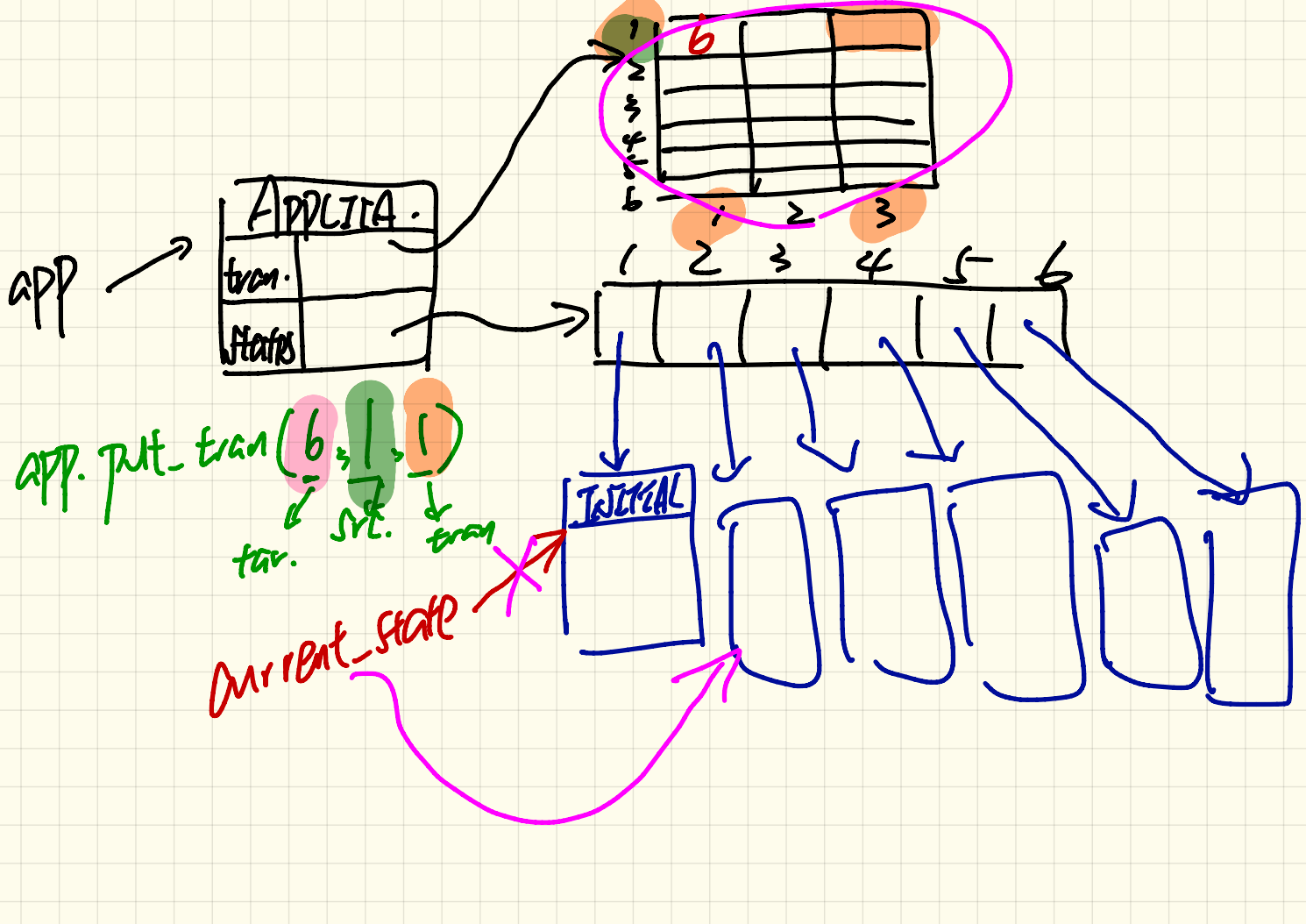


Testing the State Pattern

```
test_application: BOOLEAN
  local
    app: APPLICATION ; current_state: STATE ; index: INTEGER
  do
    create app.make (6, 3)
    app.put_state (create {INITIAL}.make, 1)
    -- Similarly for other 5 states.
    app.choose_initial (1)
    -- Transit to FINAL given current state INITIAL and choice 1.
    app.put_transition (6, 1, 1)
    -- Similarly for other 10 transitions.

    index := app.initial
    current_state := app.states [index]
    Result := attached {INITIAL} current_state
    check Result end
    -- Say user's choice is 3: transit from INITIAL to FLIGHT_STATUS
    index := app.transition.item (index, 3)
    current_state := app.states [index]
    Result := attached {FLIGHT_ENQUIRY} current_state
end
```




app
APPLICA.
tran.
states
app.put_tran(6, 1, 1)
tar.
src.
tran
INITIAL
current_state