Topics Covered:

- **Black box testing**
  - State transition testing

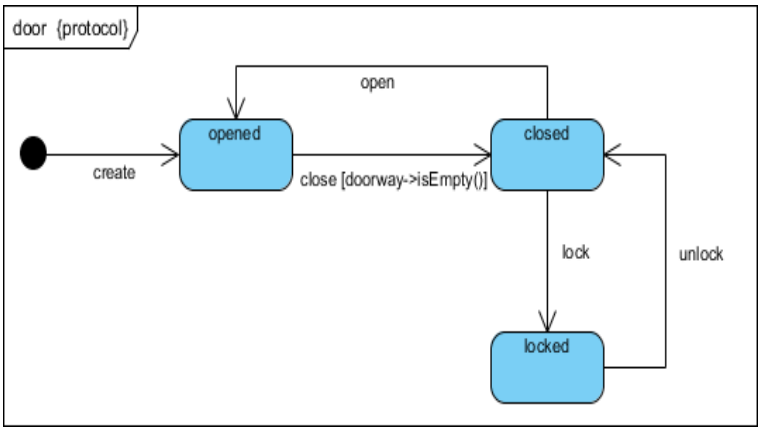
- **White box testing**
  - Statement testing
  - Decision testing

- **Experience-based test techniques (self-study)**

Black box (state transition testing)

Used to test finite systems, embedded software, menu-based, workflows,

- State transition diagram
- State transition table
- State transition coverage criteria
- State transition test case



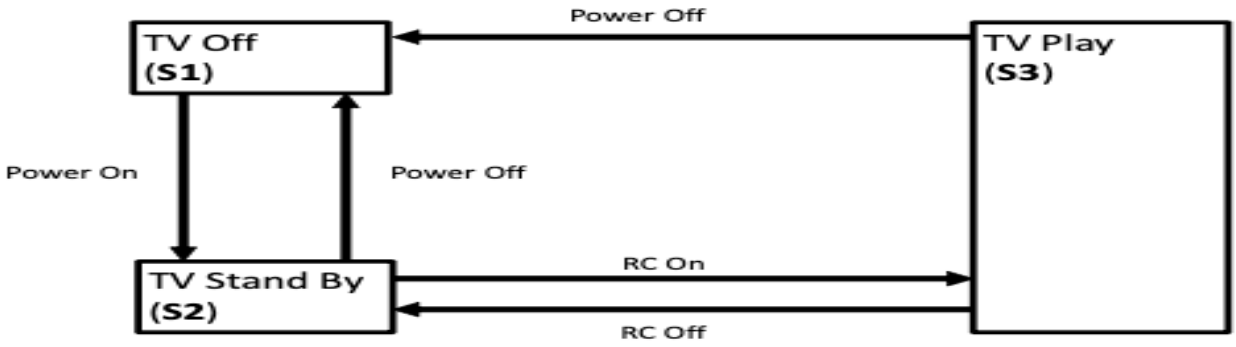
State Transition Testing

- **States** that the software may occupy (On, Off).
- **Events** that cause a transition.
- **Transitions** from one state to another.
- **Actions** that result from a transition.

Each Transition = Test Condition

Exercise

1. Which of the following statements about the given state transition diagram and table of test cases is TRUE?



Test Case	1	2	3	4	5
Start State	S1	S2	S2	S3	S3
Input	Power On	Power Off	RC On	RC Off	Power Off
Expected Final State	S2	S1	S3	S2	S1

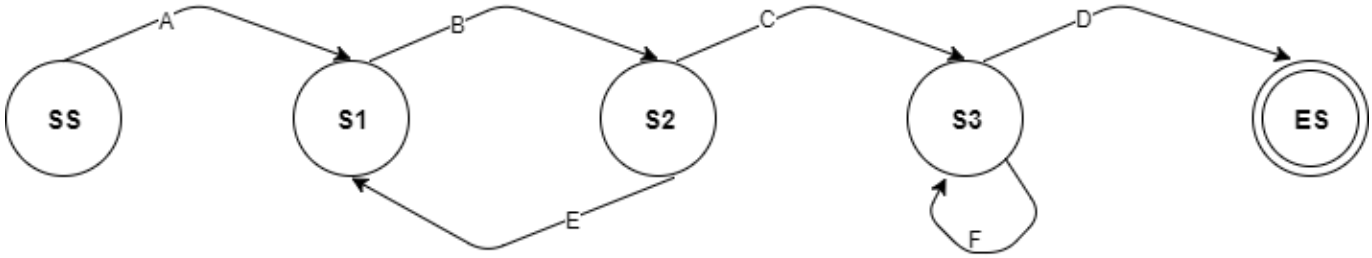
a) The given test cases can be used to cover both valid and invalid transitions in the state transition diagram

b) The given test cases represent all possible valid transitions in the state transition diagram

c) The given test cases represent only some of the valid transitions in the state transition diagram

d) The given test cases represent sequential pars of transitions in the state transition diagram

2. The following state transition diagram represents a series of valid transitions between 5 states, which answer gives a full set of 0-switch test cases?



a) ABCDFE
b) ABEBCFD
c) ABECFD
d) ABCD

White box testing

Statement Testing and Coverage

- Test executable statements (Line of code) per test case.
- Statement coverage (Number of executable statements executed / Total number of executable statements) * 100

Decision Testing and Coverage

- Test decision outcomes (true/false) per test case.
- Decision coverage = (Number of decision outcomes executed / Total number of decision outcomes) * 100

Branch Testing and Coverage

- Test decision outcomes (true/false) per test case.
- Decision coverage = (Number of branches outcomes executed / Total number of decision outcomes) * 100

Notes:

- When 100% statement coverage is achieved, it ensures that all executable statements in the code have been tested at least once.
- Statement coverage find defects that was not exercised by other tests.
- Decision coverage find defects where other tests have not taken both true and false outcomes.
- 100% decision coverage guarantees 100% statement coverage.

Exercise

---

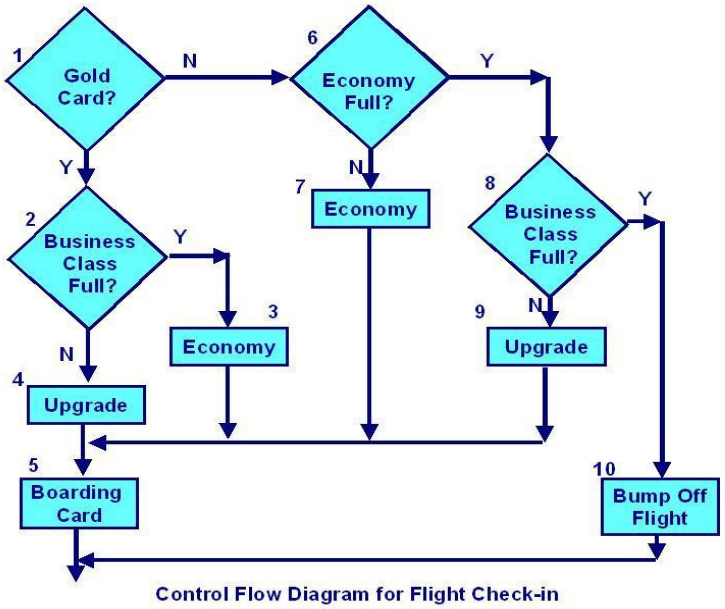
1. If you are flying with an economy ticket, there is a possibility that you may get upgraded to business class, especially if you hold a gold card in the airline's frequent flier program. If you don't hold a gold card, there is a possibility that you will get bumped off the flight if it is full and you checking in late. This is shown in following Figure. Note that each box (i.e. statement) has been numbered.

   Three tests have been run:

   Test 1: Gold card holder who gets upgraded to business class.

   Test 2: Non-gold card holder who stays in economy.

   Test 3: A person who is bumped from the flight



Control Flow Diagram for Flight Check-in

2.

```
Read P
Read Q
IF P+Q > 100 THEN
Print "Large"
ENDIF
If P > 50 THEN
Print "P Large"
ENDIF
```

Calculate the statement, decision coverage

Participation:

Find the statement and branch coverage.