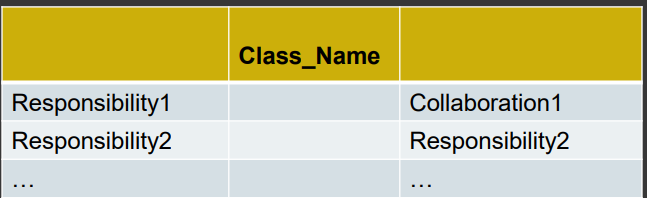
Quiz on CRC Cards and UML State Diagrams

# CRC Cards

Class Responsibility Collaborator Card

* Developed by Beck & Cunningham
* Help explore objects
* Provide ‘easy’ introduction
* Starting point of many methodologies
* Used in industry and teaching

Format:

* Index Cards
* Post-It Notes
* Boards
* String and blu-tack
* Responsibility = What class does or knows
* Collaborators = Which classes help perform responsibility

Knowing classes:

* Read specification
  + If don’t have one, write one
  + Should:
    - Describe goals
    - Discuss things system should do
* Work through requirements, highlighting nouns to give candidate classes
  + Convert plurals to singular
  + Discard nonsense classes
  + Remove synonyms
  + Beware adjectives and hidden nouns
* Candidate classes:
  + Physical objects
  + Cohesive entities
  + Categories (may become abstract superclasses)
  + Interfaces
  + Attribute values
* Problems:
  + Warnings:
    - Adjectives
    - Passive voice
  + Reject:
    - Attributes
    - Verbs
    - Objects outside system
* Identifying responsibilities:
  + Concerned with:
    - Mainenance of knowledge
    - Actions of object
  + Technique:
    - Highlight verbs in requirements
    - Walkthroughs
    - Spread intelligence
    - Keep behavior and knowledge close
  + Assigning:
    - Distribute system intelligence
    - State responsibilities as generally as possible
    - Keep behavior with related information
    - Look at relationships
      * Is-kind-of
      * Is-anagolous-to
      * Is-part-of
  + Problems:
    - Missing classes
    - Uncertain assignment of responsibilities
* Collaborations:
  + Class needs another to perform responsibilities
  + NB this is a one-way relationship
  + Class may have:
    - No colorations
    - One collaboration
    - Many collaborations

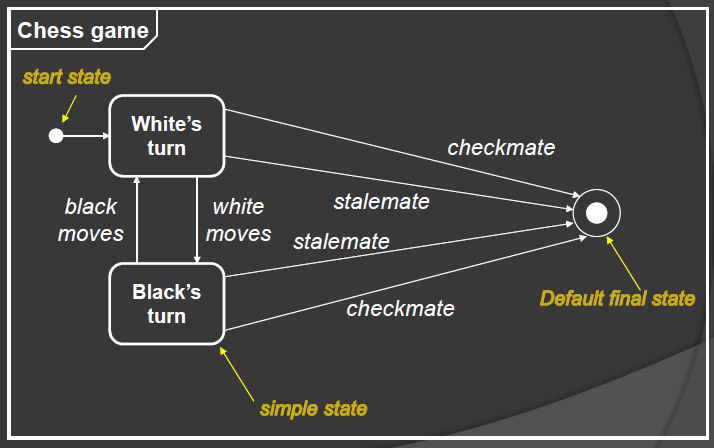
# UML State Diagrams

Used to show lifetime behavior of single pobject

* States it can be in
* What causes it to change states

Finite-state models

* Represent behavior of system over time
* Specify control
  + In terms of states, events, and transitions
  + Transitions occur when enabled
* Many variants:
  + Graphical: State charts, UML
  + Textual: FSP

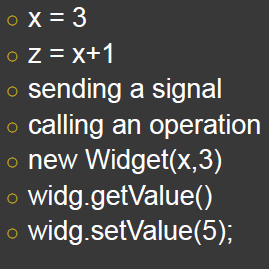
State machine view

* Describes dynamic behavior of objects over time
* States objects may hold, and how they react to events when in those states
* Localized view
* Example ---------------------------🡪

Event:

* Occurrence at point in time
* Instant
* Verbs of past tense
* Onset of a condition
* Call:
  + Receipt of an explicit synchronous call request by an object
* Change:
  + Change in value of a Boolean expression
* Signal:
  + Receipt of an explicit, named, asynchronous communication among objects
* Time:
  + Arrival of absolute time or passage of a relative amount of time

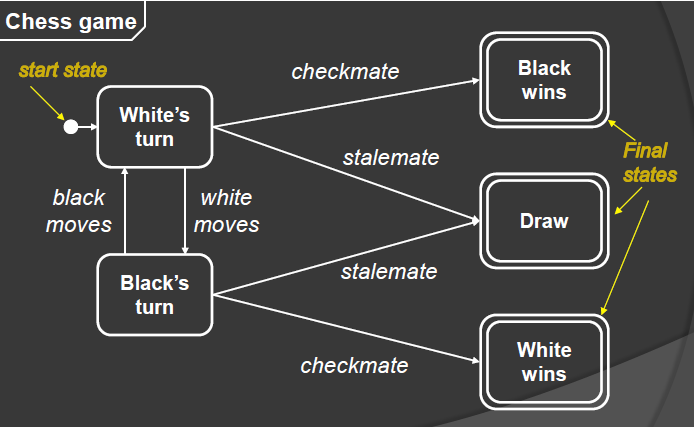
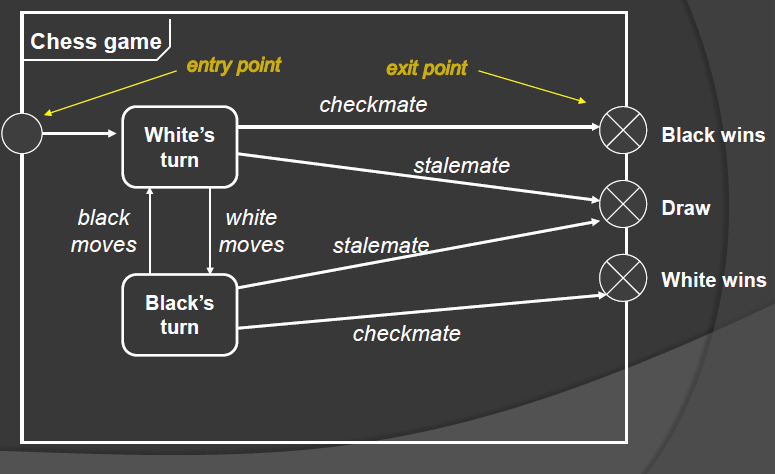
Transition:

* Instant change in state
* 4 elements:
  + Event trigger
  + Guard condition
  + Effect
  + Target state
* Kinds:
  + Entry
  + Exit
  + External
  + Internal

Activity:

* Activity
  + Primitive computation
* Do-activity
  + List of simpler actions
  + Specification of excludable behavior as the coordinated sequential and concurrent execution of subordinate units

State:

* Behavioral condition that persists in time
* Corresponds to verbs with suffix of ‘ing’
* Abstraction of values of attributes and configuration of objects
  + A set of object with similar values
  + Object waits for events
  + Object performs ongoing activity
* Kinds:
  + Simple [S]
  + Initial (O)
  + Final ((O)), [[S]]
  + Entry Point 🡪o[T]
  + Exit point [U](x)🡪
  + Submachine [s:M]
* Composite
  + Introdruces abstract “superstate”
    - Decomposed into multiple sub-states
    - When active, one sub-state active