1. Types of Frames: Role, Trajectory, Transition.

a. **Role Frames:** an action-oriented representation focused on identifying the roles played by various objects.
   - **Actor** (or Agent) – The actor is the cause of the action. Volition is generally implied. e.g., Robbie hit the ball.
   - **Act** – The action. Robbie hit the ball.
   - **Coagent** – The coagent is the partner in the cause of the action. e.g., Jack ran up the hill with Jill.
   - **Beneficiary** – The action is performed for the beneficiary. e.g., Robbie bought the gift for Suzie.
   - **Object** – What the sentence is about; typically the object undergoing a change. e.g., Robbie hit the ball.
   - **Instrument** – The action is performed on the object with the instrument. e.g., Robbie hit the ball with the bat.
   - **Conveyance** – Something on which or by which one travels. e.g., Robbie always travels by air.
   - **Trajectory** – Motion from source to destination takes place over a trajectory. e.g., Robbie and Suzie went through the front door. (This can be represented using a trajectory frame; see below.)
   - **Location** – Where the action occurs. e.g., Robbie and Suzie studied in the library.
   - **Time** – When an action occurs. e.g., Robbie and Suzie left before noon.
   - **Duration** – Duration specifies how long an action takes. e.g., Robbie and Suzie studied for an hour.

b. **Trajectory Frames:** an action-oriented representation focused on identifying the paths through which objects move.
   - **Agent**
   - **Object**
   - **Source**
   - **Away-from**
   - **Via** – A waypoint along the path. e.g., Robbie went around the corner to the store.
   - **Toward**
   - **Destination**
   - **Result** – Often represented using a transition frame; see below.

c. **Transition Frames:** an action-oriented representation focused on identifying changes in variable values over time, and supporting reasoning about cause and effect relationships.
   - Presence vs. absence for attributes
     - (Not) A = (Not) Appear
     - (Not) D = (Not) Disappear
   - Change for qualitative attributes
     - (Not) Δ = (Not) Change
   - Change for quantitative attributes
     - (Not) ↑ = (Not) Increase
     - (Not) ↓ = (Not) Decrease

2. Linearization of Inheritance Hierarchies

Constraints:
   - sub-classes before super-classes
   - classes on the left before classes on the right

Topological sorting algorithm:
   - Enumerate constraints as pairs: write a sub-class before super-class constraint, followed by the left-to-right constraints for the super-classes for that sub-class
   - Find a symbol appearing on the left side of a pair that doesn’t appear on the right side of any pair. That symbol is next in the order of linearization
   - Cross out all constraint pairs containing that symbol.
   - Repeat.