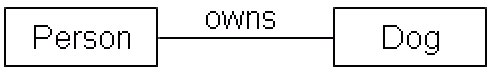
Modeling

Exercises

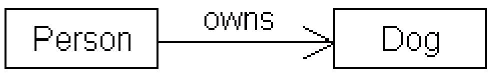
1. All models are simplifications built by excluding details. When constructing a model, how do you decide which details to exclude?

> **Ans.** The intended audience for the model and its purpose determine what details to exclude.

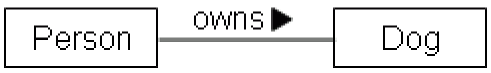
2. During a modeling exercise you create the following model:



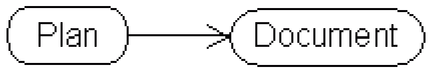
Recalling your experience with some dogs and their owners you decide to make the direction of the relationship explicit. Is the following the correct way of making the direction of the relationship explicit?

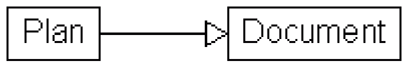


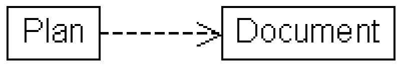
> **Ans.** No. The proposed solution says that Dog is visible from Person. It doesn't say anything about how to read the relationship. The correct answer is:



3. Match the UML model with the appropriate interpretation.

___ A plan is dependent on a document. A. 

___ Plan first, then document. B. 

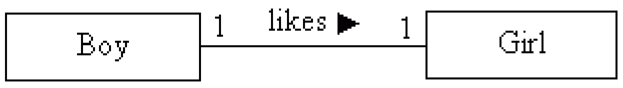
___ A plan is a special type of document. C. 

> **Ans.** C, A, B

4. Give an example of a non-graphical model used during software development.

> **Ans.** The narrative form of a use case, a data dictionary, a process description.

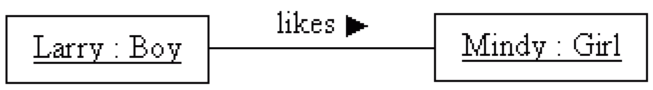
5. Consider the following class diagram:



Based on the class diagram above, is there any way to express with an object diagram that Jane (a girl) likes Brian (a boy)?

Ans. No. The relationship is boy-likes-girl. With the class diagram above there is no way to express that a girl likes a boy.

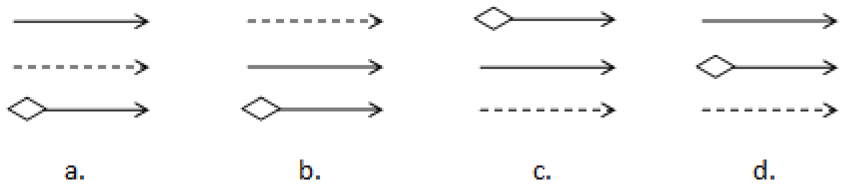
6. According to the class diagram in question #5, if Larry likes Mindy:



Can there be another boy, besides Larry, that also likes Mindy?

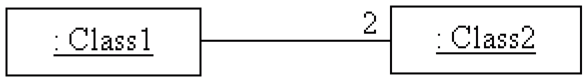
Ans. No. It is impossible because the model says that every girl is liked by exactly one boy. If Larry likes Mindy it's impossible for Mindy to be liked by another boy.

7. Which of the following properly sorts the given UML relations from the more general to the more specific?

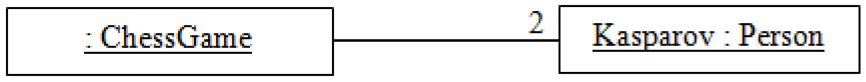


Ans. b.

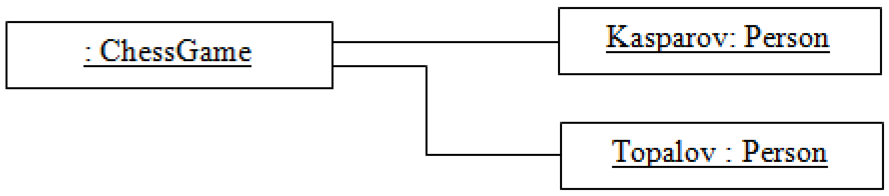
8. Is the following UML model correct? If so, what does it mean?



Ans. It is invalid. Links can't have multiplicities. To see why multiplicity on links do not make sense, consider an example with more meaningful names:



The example above is incorrect. If the intent is to suggest a link to two objects, the links can be shown separately:

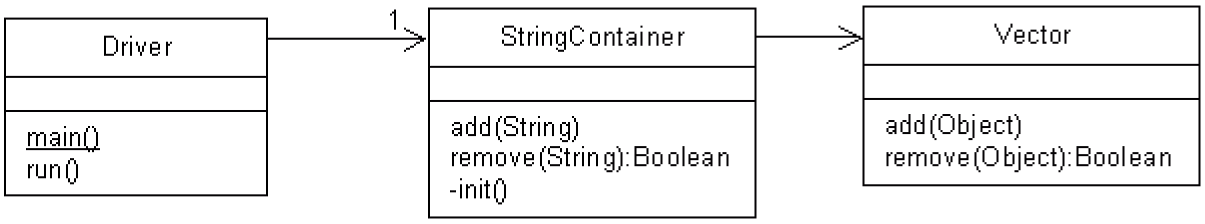


9. Draw an UML class diagram to express the structural relationships in the following program and draw an UML sequence diagram to express the dynamic behavior.

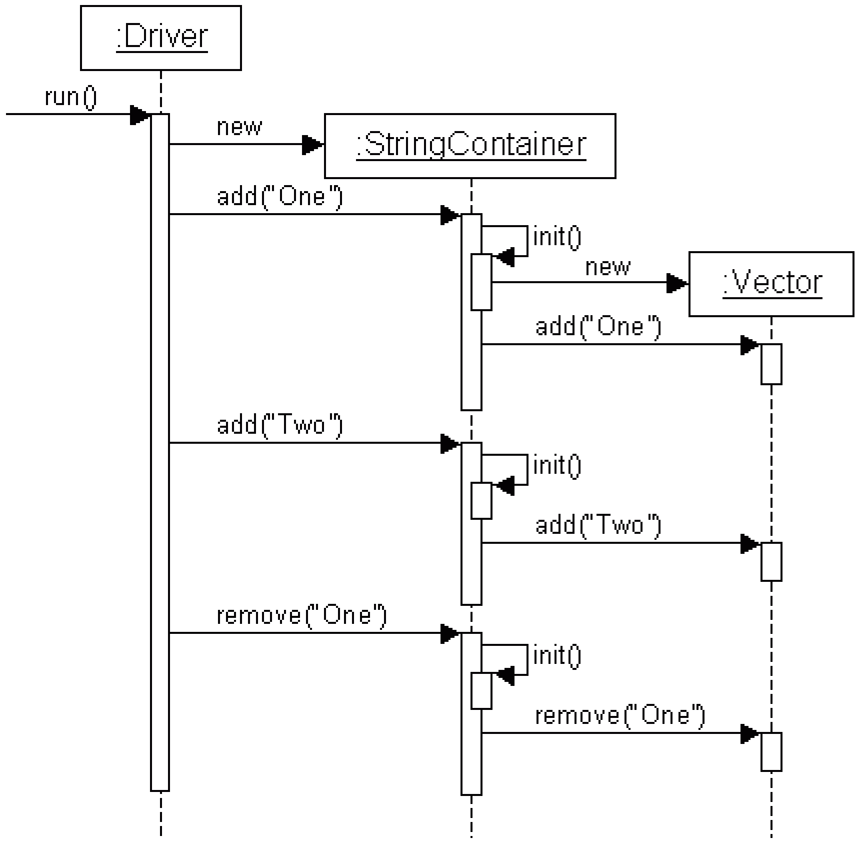
```
import java.util.Vector;

public class Driver {
    private StringContainer b = null;

    public static void main(String[] args){
        Driver d = new Driver();
        d.run();
    }

    public void run() {
        b = new StringContainer();
        b.add("One");
        b.add("Two");
        b.remove("One");
    }
}

class StringContainer {
    private Vector v = null;

    public void add(String s) {
        init();
        v.add(s);
    }

    public boolean remove(String s) {
        init();
        return v.remove(s);
    }

    private void init() {
        if (v == null)
            v = new Vector();
    }
}
```

Ans.

Class Diagram:

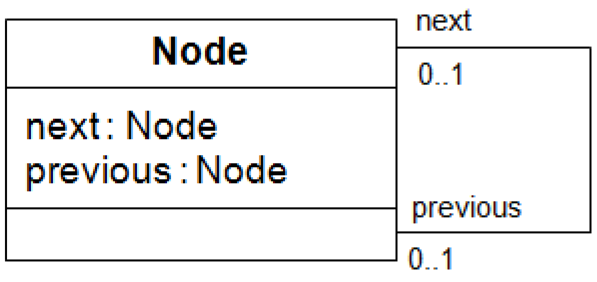


Sequence Diagram:

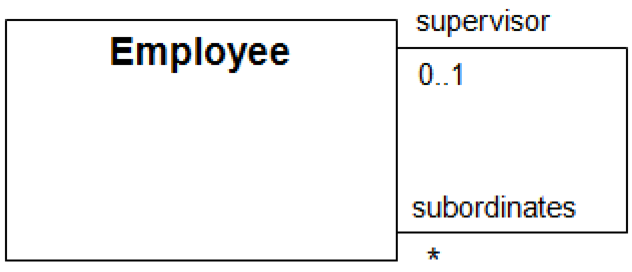


10. Give an example of a binary association where both ends attach to the same classifier.

Ans. Both ends may attach to the same classifier when one instance refers to another instance or when one instance refers to itself. For example, consider a node in a linked list:



Or, an employee class that keeps track of supervisors and subordinates:



11. Does the UML prescribe a process?

Ans. No, the UML evolved from modeling notations that were process or method specific, but the UML doesn't specify or require a particular process. The UML specification does, however, recommend a developmental process that is use-case driven, architecture-centric and iterative and incremental.

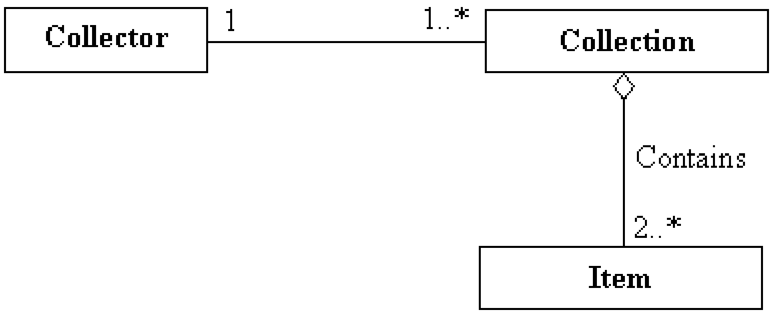
12. What are the dangers of creating models that use intuitive symbols but don't follow the syntax and semantics of a formal (or semi-formal) modeling language?

Ans. The biggest danger is that it will be interpreted incorrectly. Formal modeling languages have precise semantics, which allows them to be more expressive.

13. Create an UML class diagram that models the data relationships described in the following paragraph.

> To be a collector you have to have one or more collections. Each collection must have 2 or more items. Each collection belongs to one collector. A collection is made up of items owned. A particular item may be in more than one collection (i.e. an old Coke sign may be in both a Coke memorabilia collection and a sign collection.)

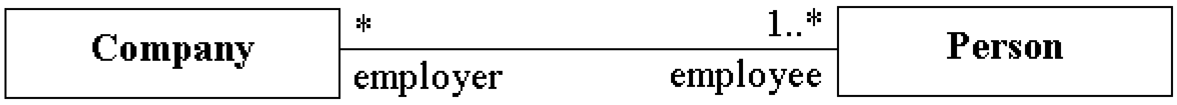
Ans.



14. T/F In an UML class diagram the default value for attribute and operation visibility is public. If the visibility isn't specified for an attribute or operation you should assume it is public.

Ans. False. The UML doesn't define defaults for visibility. In general, very few unspecified attributes of a model can be assumed. Having default values for elements of a UML model isn't practical because it makes it impossible to distinguish between the case when something is left out because it's not known or isn't important for the purpose of the model and the case when the modeler expected the default values to apply.

15. Use the following class diagram to answer the questions that following it. If there isn't enough information in the diagram to answer the question, state as much.



1. Can you have a company without any employees?
2. Can a person be unemployed?
3. Can a person be self-employed?

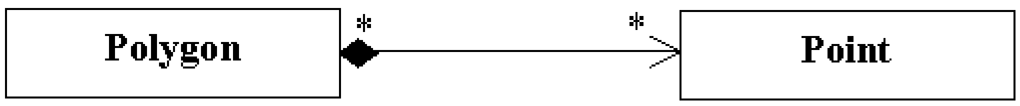
Ans.

1. No, the multiplicity on the employee side is 1..*.

2. Yes, the multiplicity on the employer side is * so 0 is a valid value.

3. According to the diagram if a person was self-employed that person would have to work for the company he or she owned.

16. What, if anything, is wrong with the following diagram:



Ans. It is incorrect. The part side of a composition can be associated with at most one composite. The diagram above says that a point can be included in more than one Polygon (* = 0 or more). If this is the semantics wanted, composition (filled diamond) should be replaced with aggregation (unfilled diamond).

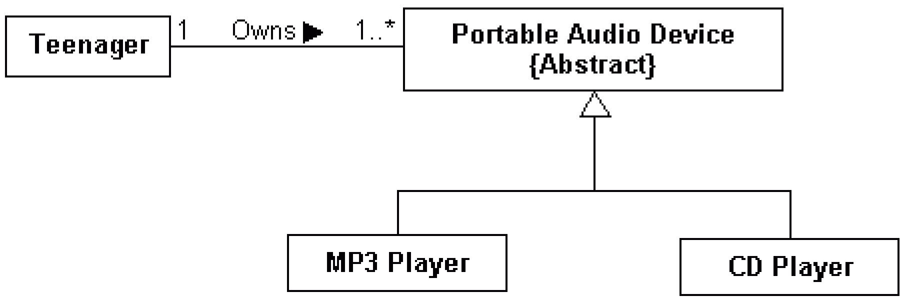
17. What is the difference between a behavioral relationship and a structural relationship between classes?

Ans. A behavioral relationship is temporary. At the analysis level a behavior relationship might be: a student services representative processes an application. A structural relationship is more permanent, such as: a student is a member of a department.

In implementation terms, a behavioral relationship exists if a method in one class receives a reference to another class as a parameter. A structural relationships exists if one class maintains a reference to another class. With an association (structural relationship) the reference might change or be null at times but it almost always exists.

In the UML a behavioral relationship is modeled as a dependency $\dashrightarrow$. A structural relationship is modeled as an association $\longrightarrow$.

18. Use the following figure to answer the questions that follow.



a. How many portable audio devices does each teenager own?
b. Can there be a portable audio device not owned by any teenager?
c. Can there be an MP3 player player not owned by any teenager?

Ans. (a) 1 or more. (b) no. (c) no. An MP3 player is a portable audio device. All portable audio devices have to be owned by exactly one teenager, so any MP3 player must be owned by exactly one teenager.

19. (True / False) The model in question 18 says that there is at least one teenager and at least one portable audio device.

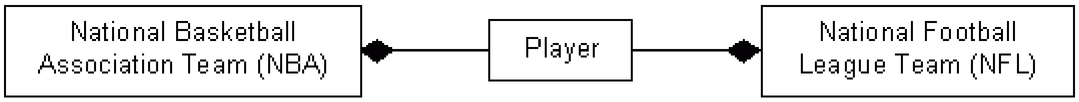
Ans. False. The reason is both subtle and obvious. The image above is a class diagram. It doesn't indicate there are any actual objects. The 1's next to Teenager and PortableAudioDevice don't imply there are actual teenagers and portable audio devices.

The model is a static model. It describes the structural relationships between classes. It's a class diagram. There are no objects in the diagram. It only says that *if* you have a teenager then you have a portable audio device, or *if* you have a portable audio device then you have a teenager. As an analogy, just because you have a recipe for potato salad doesn't mean there are potatoes.

20. tbd.

Ans. tbd.

21. Mark each of the statements below true or false, according to the following diagram.


National Basketball Association Team (NBA)
Player
National Football League Team (NFL)

a. (True / False) A player must play for either an NBA team or an NFL team but can't play for both at the same time.
b. (True / False) A player can play for two NBA or NFL teams simultaneously.
c. (True / False) A player can be traded among teams.
d. (True / False) A player must always be a part of a one team.

Ans.

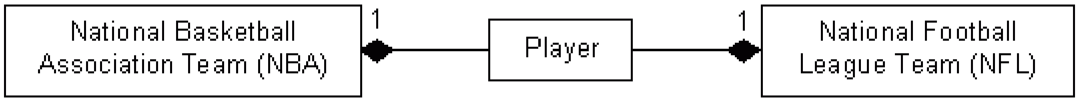
a. True. The semantics of the composition relationship is that a component may be part of at most one composition at a time.

b. False. The multiplicity implied on the composite side of a composition relationship is 0..1. A player can't play for more than one team.

c. True. A component can be a part of at most one composition but can be traded between compositions. The attitude of the statement is correct too. The whole side of a whole-part relationship is thought of as owning the components on the part side.

d. False. A player may assume responsibility for himself. While a player is on a team that player is controlled by the team.

22. What, if anything, is wrong with the following model?


National Basketball Association Team (NBA)
1
Player
1
National Football League Team (NFL)

Ans. The multiplicities of each composite should be 0..1 rather than 1. A component can be a part of at most one composition at a time. The diagram above says that Player is always a part of both composites.

23. If any method of a class is abstract the class must be abstract as well. Is the reverse true? That is, if a class is abstract must it have an abstract method?

Ans. No. You can have an abstract class that has no abstract methods. A class that has no abstract methods doesn't have to be abstract, but you may want to make it abstract as a way of preventing clients from creating instances of it.

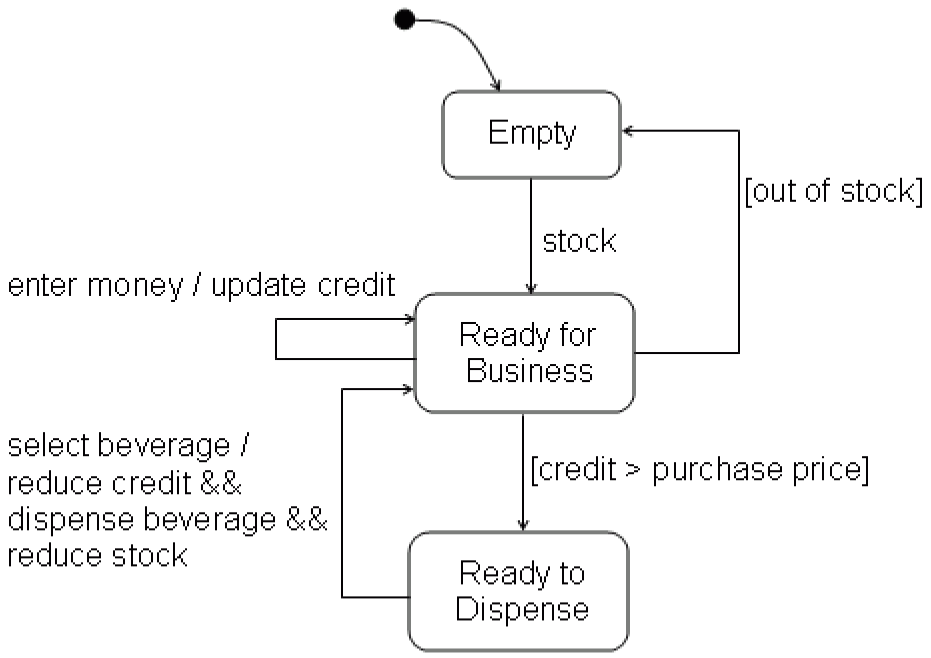
24. The UML can be used during all phases of the software life cycle. Give some examples of how UML is used during the software life cycle.

Ans. During the requirements phase you might create an analysis model to verify your understanding of the problem domain. Analysis models should reflect the problem domain so that end users can review and verify them. During the design phase you might create a design model to experiment with alternative solution ideas. It's much less expensive to explore solution options with models than with code. During implementation you might document program implementation with implementation models. During maintenance, programs must be understood before they can be changed. Programs are, of course, fully specified by their source code, but source code usually contains too much detail to be efficiently read and understand. Implementation models ease the task of reading and understanding source code.

25. What does the "width" of a sequence number on a message in a communication diagram represent. For example, the sequence number on the message "4.5.6 : method()" is 3 decimal digits wide.

Ans. The depth of the call stack.

26. According to the soda machine statechart in figure 29 (repeated below for convenience) is there any way for the machine to take money without giving credit towards a beverage?


Empty
stock
[out of stock]
enter money / update credit
Ready for Business
select beverage / reduce credit && dispense beverage && reduce stock
[credit > purchase price]
Ready to Dispense

Ans. Yes. If money is entered when the machine is empty or when it is ready to dispense, these events will be ignored and credit will not be updated.

27. What is the difference in perspective between a use case model and a data flow model?

Ans. A use case model captures the interactions between the system and its environment. A data flow model captures the flow of data through the system. A use case model has an external perspective, and a data flow model has an internal perspective.

28. Which UML model is best capable of modeling the life-cycle of a butterfly?

Ans. Statechart diagram. Butterflies go through 4 stages of metamorphosis: egg, larva (caterpillar), pupa (chrysalis or cocoon), and adult butterfly. A statechart diagram is probably the best diagram for documenting the activities of each stage and the transitions between stages.

29. Models may differ in the amount of detail and type of information they expose. Provide an example model that makes a clear distinction between these two concepts.

> **Ans.** Consider a class diagram. The type of information it represents is the static structure of the system. This type of informaiton is different from the information on the dynamic behavior of the system. A class diagram may include more or less detail. A detailed class diagram would include method names, parameters, parameter types, return values, attribute names, attribute types, etc.

30. (True / False) The default multiplicity for attributes is [1] in the UML meta model (the document that defines the meaning of UML). If you see an attribute in a class diagram without explicit multiplicity, it might be reasonable to assume the multiplicity is [1] but why can't you be absolutely certain it is?

> **Ans.** It's possible the multiplicity of the attribute wasn't specified because it wasn't important for the purposes of the model.

31. Of the 4 relationships: association, realization, generalization, and dependency, which is the weakest?

> **Ans.** Dependency. The other three relationships imply dependency.

32. tbd?

> **Ans.** tbd.

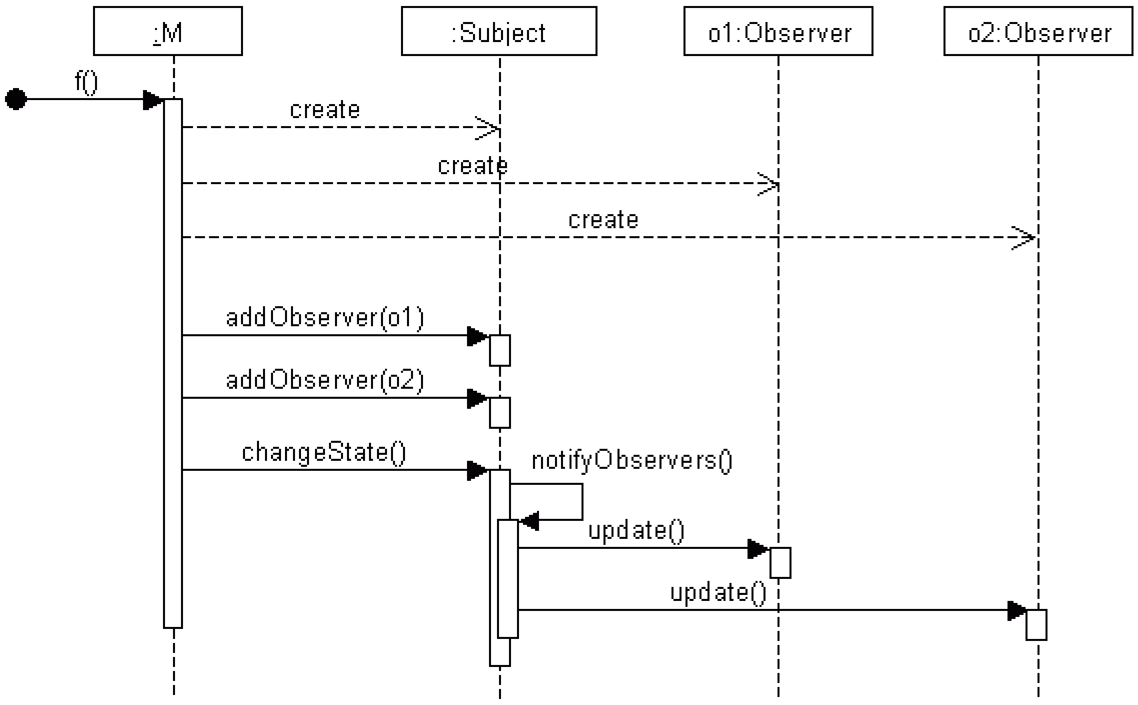
33. Draw the UML class diagram and UML sequence diagram for the following code fragments. You can assume class M already exists and start the sequence diagram with the method f(). You don't have to document calls to any methods defined for classes not shown here.

```
public class M {

  public static void main(String[] args) {
    M m = new M();
    m.f();
  }
  public void f() {
    Subject s = new Subject();
    Observer o1 = new Observer();
    Observer o2 = new Observer();
    s.addObserver(o1);
    s.addObserver(o2);

    s.changeState();
  }
}
```

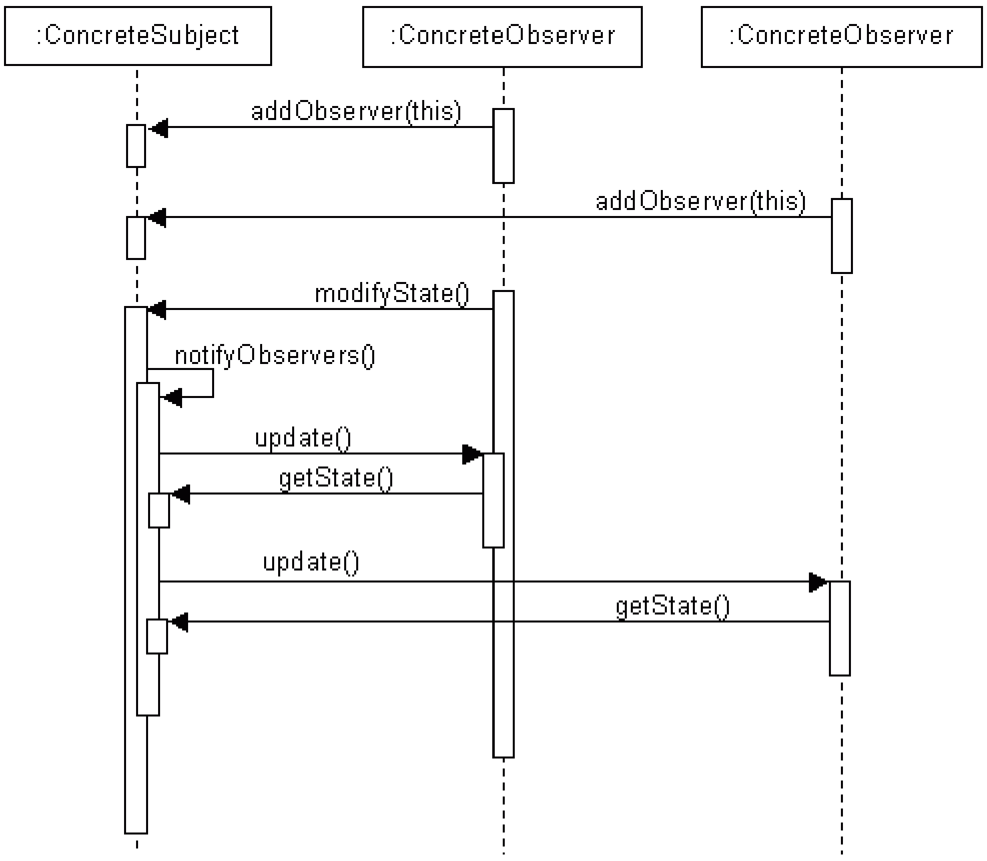
```
class Observer {
    public void update(){}
}
```

```
class Subject {
  private Collection c;

  public void addObserver(Observer o) {
    c.add(o);
  }

  public void changeState() {
    // Change state of subject
    notifyObservers();
  }

  public void notifyObservers() {
    Iterator i = c.iterator();
    while (i.hasNext()) {
      Observer o = (Observer)i.next();
      o.update();
    }
  }
}
```

> **Ans.**



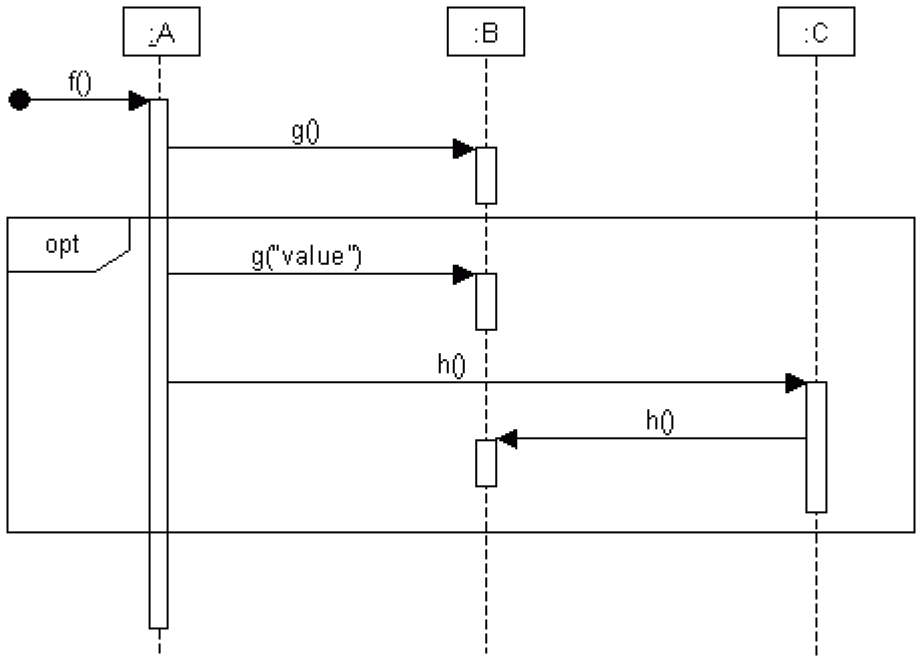
34. Write the code fragment that corresponds to the following sequence diagram.



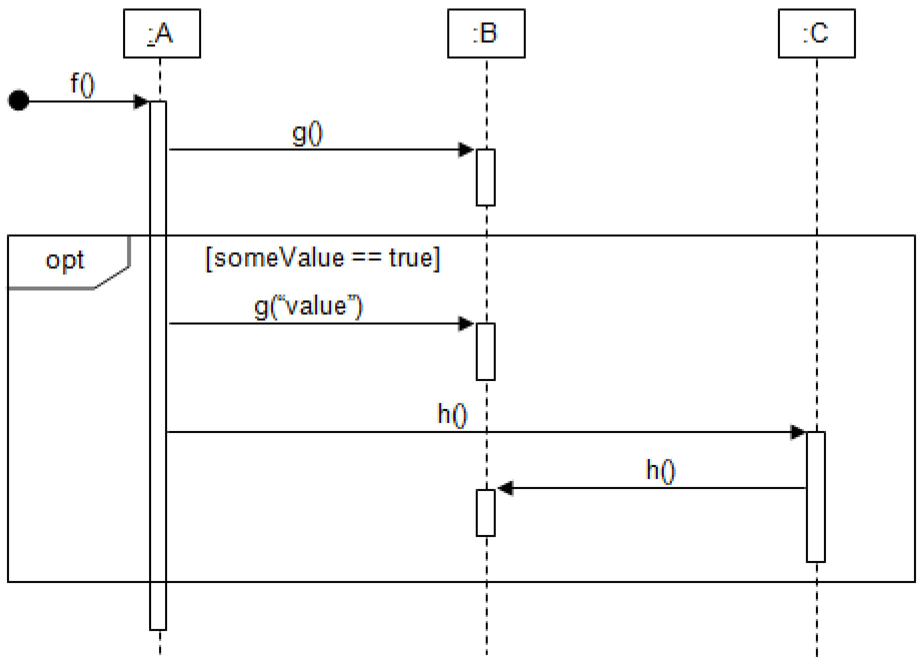
Ans. While not complete, the code fragment below reflects the essence of the sequence diagram above.

```
public class ConcreateSubject {

    public void addObserver(ConcreateObserver o) {...}
    public void modifyState() {
        notifyObservers();
    }

    public void notifyObservers() {
        for each observer o {
            o.update(this);
        }
    }

    public Object getState() { ...}
}

public class ConcreteObserver {

    public void update(ConcreateSubject s) {
        s.getState();
    }
}
```

35. Is there anything wrong with or missing from the following sequence diagram? If so, what is wrong or missing?



Ans. The sequence diagram above includes a combined fragment with the opt ("optional") interaction operator but doesn't include a guard condition saying when the optional fragment should be executed. It should include a guard condition of the form: [*boolean expression*]. The optional fragment executes only if the boolean condition is true.



36. Roughly speaking, a communication diagram contains a mixture of features and characteristics from what three other diagram types?

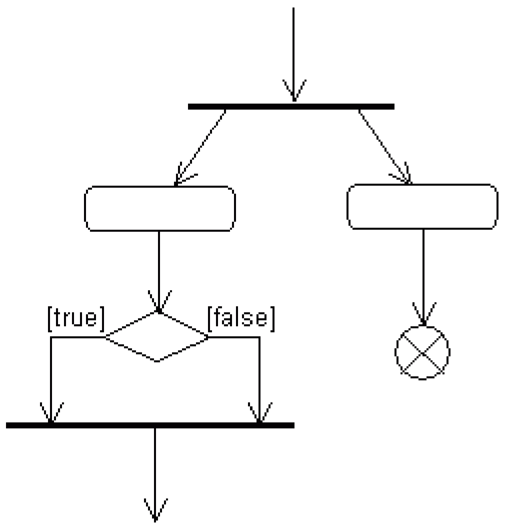
> **Ans.** A communication diagram has structure like a class diagram, objects like an object diagram, and object interaction like a sequence diagram.

37. What does an edge in a UML activity diagram represent?

a. Control flow only
b. Data flow only
c. Both control flow and data flow
d. Control flow or data flow or both

> **Ans.** d. Edges on an activity diagram represent the flow of control or the flow of data (objects) or both.

38. What, if anything, is wrong with the following activity diagram (other than missing action node names)?



Ans. It never ends. There is a decision node with both of its outputs going directly into a join point. Control flow will follow one of the paths from the decision node but never both. The join point, however, needs control flow on both inputs before execution is allowed to proceed.

39. What are partitions (or swim lanes) used for in activity diagrams?

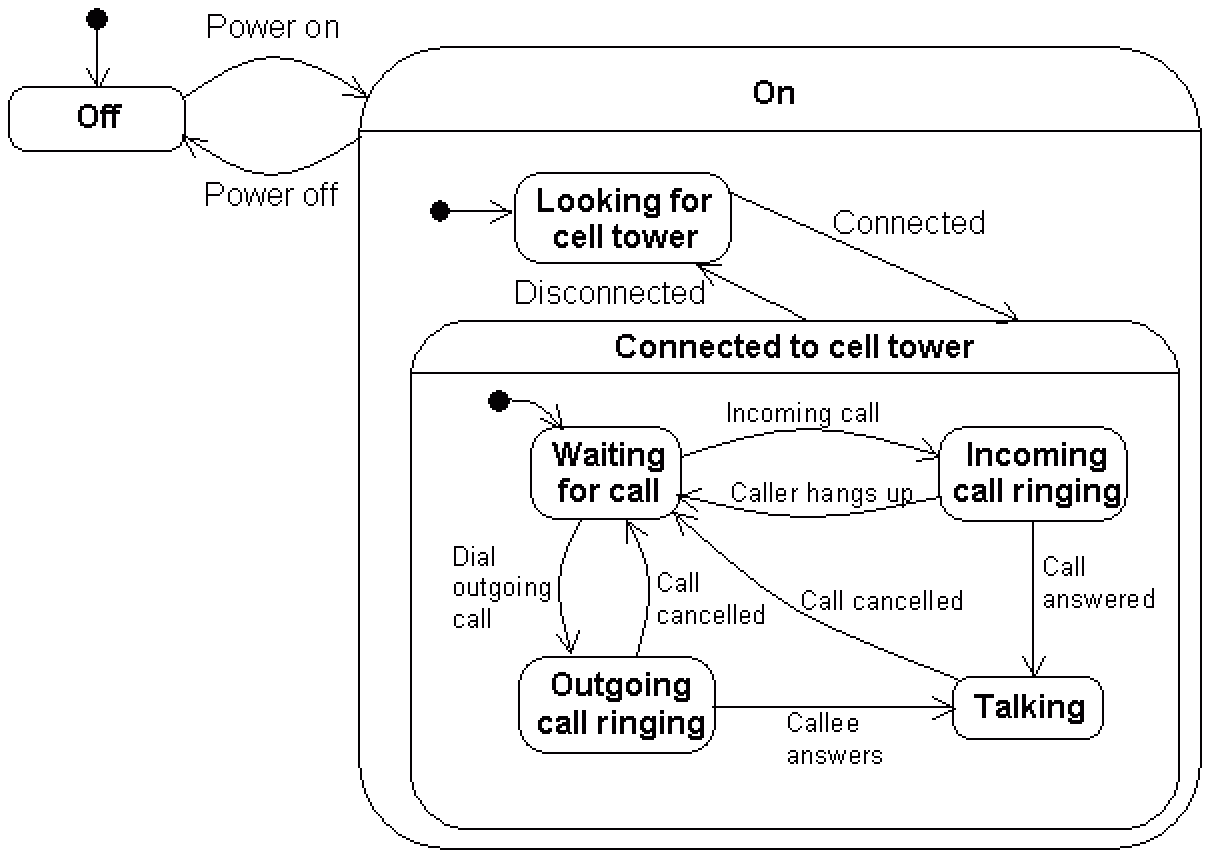
Ans. Partitions, also called swim lanes, can be used to show who or what is responsible for certain activities.

40. What is the syntax for expressing object flow between two activities in an activity diagram?

Ans. Put an object node between two action nodes to make it clear that data is flowing over the edge between the two nodes. You can also use pins at the start and end of an edge to state expl that data is flowing over the edge. Pins can be labeled with a data type that indicates the type of data flowing over the edge.

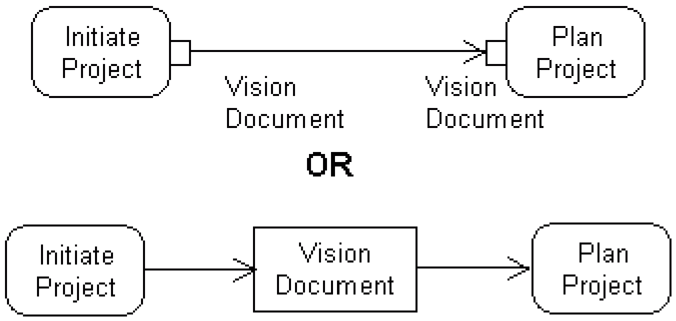
41. Model the operation of a cell phone using a state machine diagram. Show the main states a cell phone can be in and the events that cause it to transition between these states. Use substates if it helps simplify your diagram.

Ans. Substates reduce the number of connections that have to be shown on a model. A transition to/from a composite state represents a transition to/from any state within the composite state. For example, in the graph below it's possible to transition to the Off state from any of the other states. Rather than show 5 different edges with the label "Power off", there is one edge from the composite state On to the Off state.

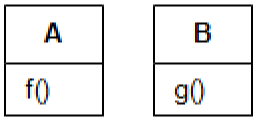


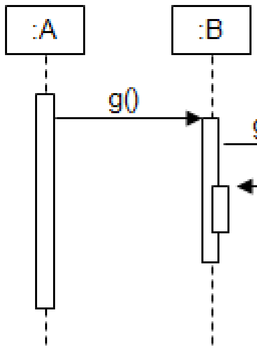
42. Create an activity diagram with two nodes: Initiate Project and Plan Project. Use object flow notation to show an instance of a Vision Document flowing between the two nodes.

Ans.

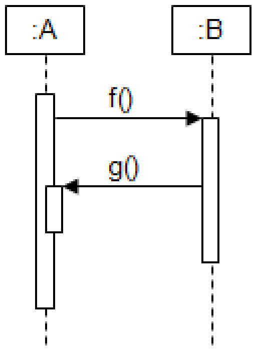


43. Which of the following sequence diagrams are potentially valid for the given class diagram? Check all that apply.

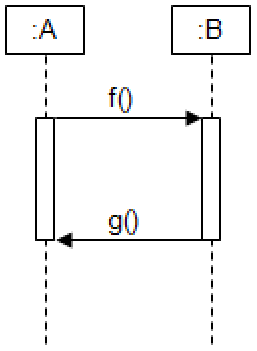




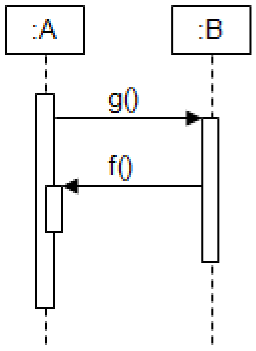
(a)



(b)



(c)

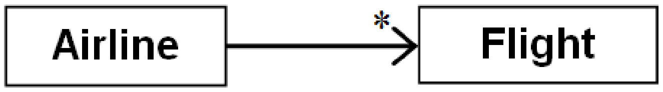


(d)

Ans. a and d.

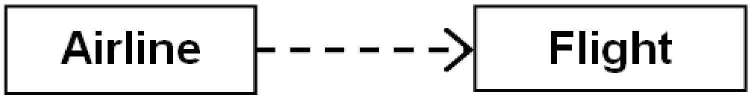
44. Is the following class diagram valid for the given code fragment? Explain. If invalid, show a valid class diagram.

```
class Airline {
   Flight [] getFlights() { . . . }
}
```



Ans. Invalid. The class diagram shows a structural relationship between Airline and Flight. In other words, if you have an instance of an Airline you are likely to have an instance of zero or more flights. The code shows a weaker dependency between the two classes. Airline has a method that returns zero or more Flights. If the interface of Flight changes, Airline may need to change also.

The correct class diagram for the code above would show a dependency relationship between Airline and Flight:



A valid code fragment for the class diagram shown in the question is:

```
class Airline {
   private Flight [] flights;
   . . .
}
```