Answers to Questionnaires:

1. Define & discuss OO characteristics: Identity, Classification, Inheritance, and Polymorphism.

Answer:

(a)Identity (Thinking about objects):

An object is an entity which is having real world existence and it is distinguishable from other objects(by considering other properties of object) .

For objects, we have to consider both aspect:

Real world & Programming aspect.

Both type of object s are having existence & Identity, but there is a conceptual difference between them.

Eg. TABLE occupies space in "SPACE"(Real World),

While,

TABLE occupies space in "MEMORY"(Computer World)

(b)Classification (Grouping of objects/Thinking about Class):

It means objects with same data structure (attributes) and behavior (code/operation) can be grouped

Together and a cover is provided, called a class.

Considering both aspect:

Real world & Programming aspect.

Both type of object s are having existence & Identity, but there is a conceptual difference between them.

Eg. TABLE occupies space in “SPACE”(Real World),

While,

TABLE occupies space in “MEMORY”(Computer World)

(c)Inheritance:

It is the sharing of attributes and operations among no. of other classes using hierarchal relationship.

It means attributes & method defined In a class (Super Class) can be used by its Subclass without redefinition of them. That’s why we can say, Inheritance provides us code reuse facility.

(d)Polymorphism:

It means same operation may behave differently for different classes.

Eg.

Assume ,

one class CIRCLE, second class SQUARE, third class RECTANGLE………

Now ,it may that operation AREA() may behave differently for all three classes.

As we want area of circle to be (3.14*r*r),

Similarly, for square (l*l) ,for rectangle(l*b),

so, methods to implement this the different will be three, but we can give same name to the method.

Each object knows its class as well as which method to select, when an operation is polymorphic.

2. Explain: Various OO Themes with example.

Answer:

(a)Abstraction:

It focuses in only important portion of the application, we don't capture all the possibilities. It means, we focus on "what an object is and does", before deciding how to implement it.

(b)Encapsulation:

It encapsulates data & code together and separates external aspect of objects from implementation details.

You can change object's implementation without affecting application which use that object.

(c)Inheritance:

It is the sharing of attributes and operations among no. of other classes using hierarchal relationship.

It means attributes & method defined In a class (Super Class) can be used by its Subclass without redefinition of them. That's why we can say, Inheritance provides us code reuse facility.

(d)Polymorphism:

It means same operation may behave differently for different classes.

E.g. Assume ,one class CIRCLE, second class SQUARE, third class RECTANGLE………Now ,it may that operation AREA() may behave differently for all three classes .As we want area of circle to be (3.14*r*r),lly, for square (l*l) ,for rectangle(l*b),so, methods to implement this the different will be three, but we can give same name to the method.

Each object knows its class as well as which method to select, When an operation is polymorphic.

3. Define Model. Explain Importance of Modeling. Explain three models & relationship between them.

Answer:

A model:

A model is an abstraction of something for the purpose of understanding it before building it.

Any of the models will be useful to at most the two persons.

1. Developer, who will build the system.

It will let a person to Think upon different aspects of the system before actual development, so that if any updation is required , one can do it and once the model become accurate ,we can go for next stage, called "Developing" the system.

2. User, who will use the system after complete development.

so that a person can predict(visualize) the system that "How it will look alike and facilitate them" because it human nature tendency that it can grasp Graphical Form faster & better.

Also, user can suggest other modifications, which can be useful in improvement of the system.

It means both the aspects provides us with early corrections.

Another important reason is Reduction of complexity, As it will be too complex to make change in a system directly ,but if we separate out the important aspects in a model and whenever we need to modify the system, before directly do change in a system ,if we find out the place where we need to change in the model ,then it will be more easier to understand.

THREE MODELS & THEIR RELATIONSHIP:

There are three types of models to describe a system from different viewpoints.

So, you can say that A full description requires all three models.

CLASS MODEL:

It covers all possible & necessary classes (with attributes & operations) and their relationships.

Facility like Generalization allows us to share attributes and operation with other classes (subclasses).

Eg. As far as our Hostel Management System (HMS) is concerned class model may contain classes like,

Owner,

Staff & its generalization

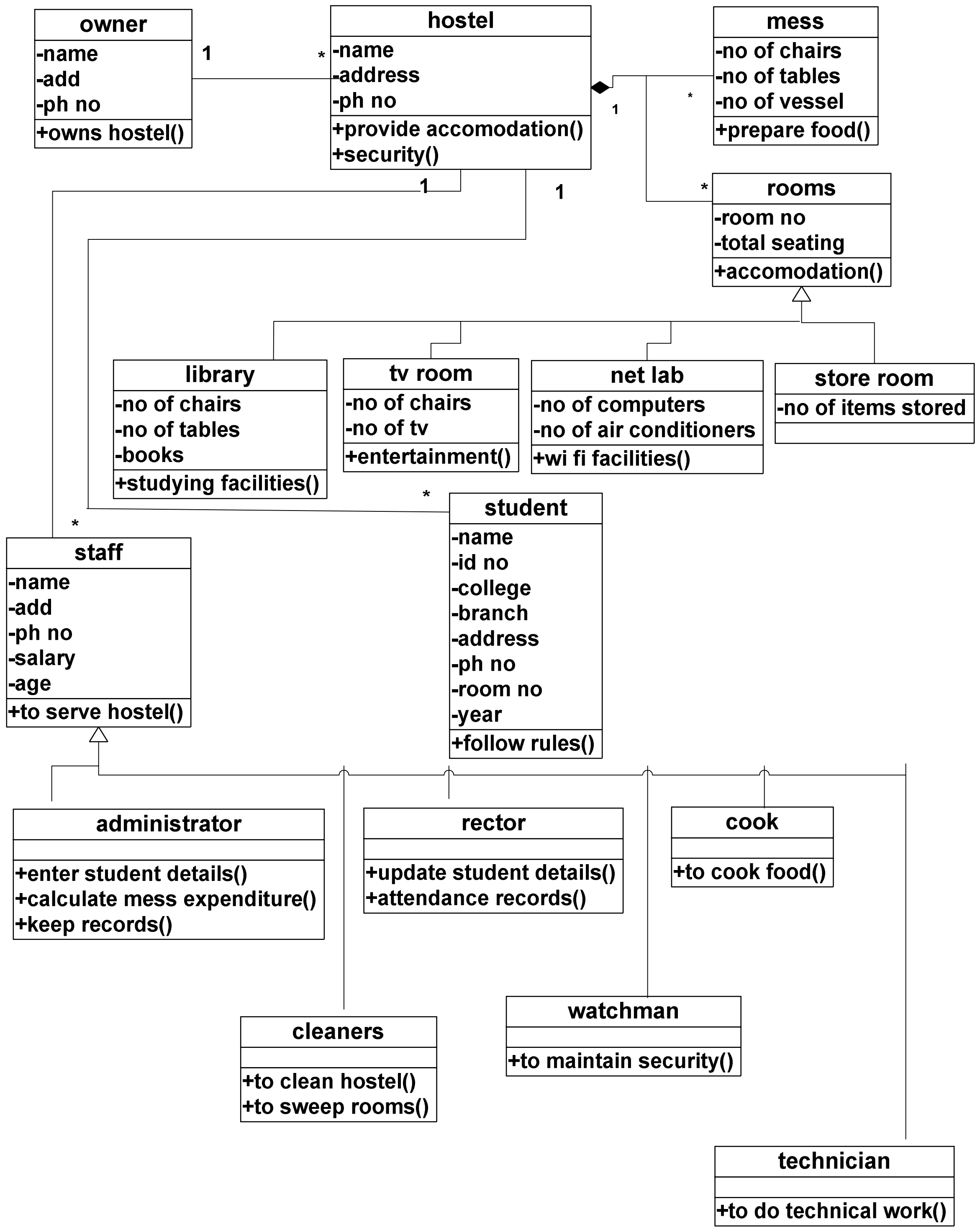
Students & its Generalisation.

Rooms & its Generalisation

Building with composition,etc.

It's obvious that As time passes Staff may change, students may change ,It simply means objects are getting changed ,but class is not. And these classes and their relationship will not change till system exist. So we can say that class is a STATIC (which doesn't change frequently) STRUCTURE.

Class Diagram:



STATE MODEL:

As far as HMS is concerned, In class STUDENT we have operations like ,

Taking Admission(),

AllowedtoUseRoom()

Allowed toUseMess()

AllowedtoUseLibrary(),

etc....

State Model will show the whole Life cycle of the object. It means ,from when object is created to when it got destroyed.i.e.When student gets admission and be a member a hostel and then uses all the facilities ,then its studying period finishes and he left the hostel.........

So ,from starting to end whichever possible actions are there they will be covered here.

Hence, You can say that

Sate model describes those aspects of objects which are concerned with Time & Sequencing of Operation.

Here, At a particular time some action will happen(called EVENT) and in response to that occurrence another action/no. of actions will be performed.(called STATE).

The State Model captures control without regard for What operation do, What they operate on, or how they are implemented.

State & event of State diagram becomes Operation of class diagram.

Reference between state diagrams becomes interactions interaction model.

STATE DIAGRAM:

5.4 State Diagrams 97

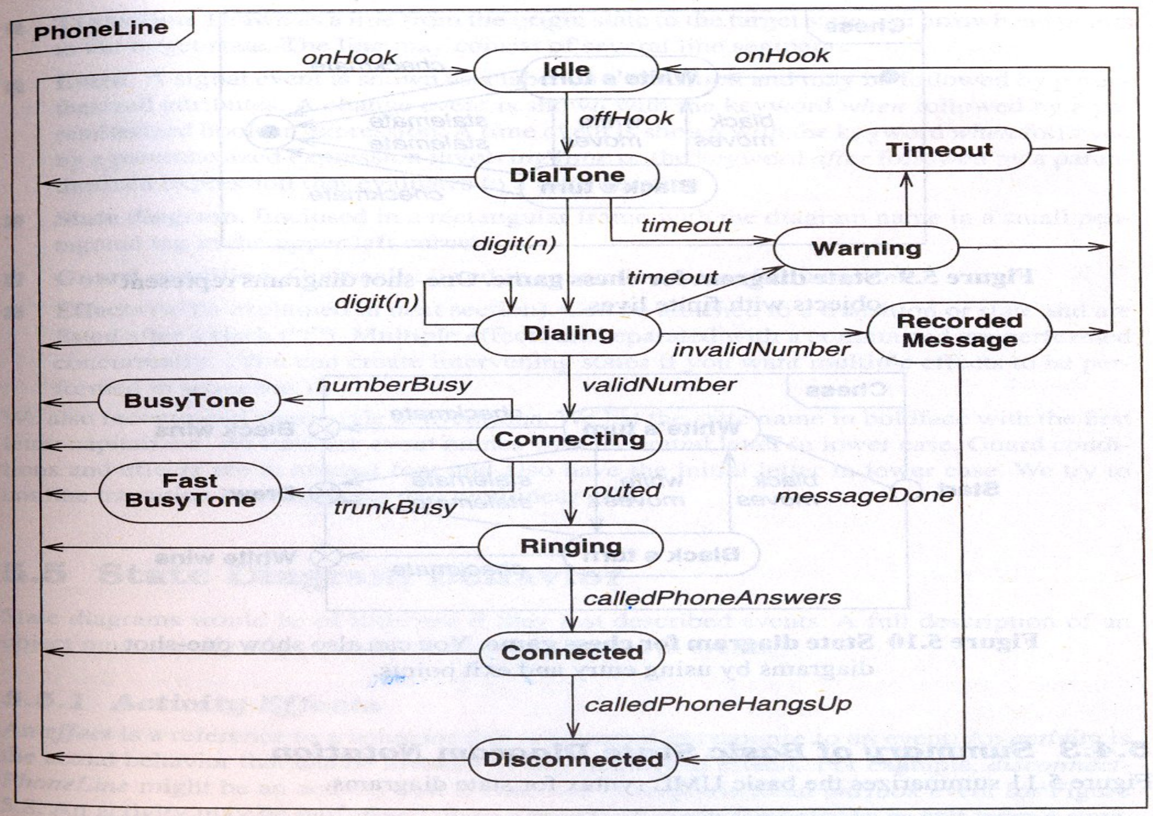


Figure 5.8 State diagram for a telephone line. A state diagram specifies the state sequences caused by event sequences.

INTERACTION Model:

it describes interactions between objects that how they will communicate when system is working.

Interactions can be modeled at different levels of abstraction.

- HIGH LEVEL-USE CASE:How system WILL interact with outside users.
- SEQUENCE DIAGRAM:PROVIDE MORE DETAIL and shows the messages exchanged among a set of objects over time.

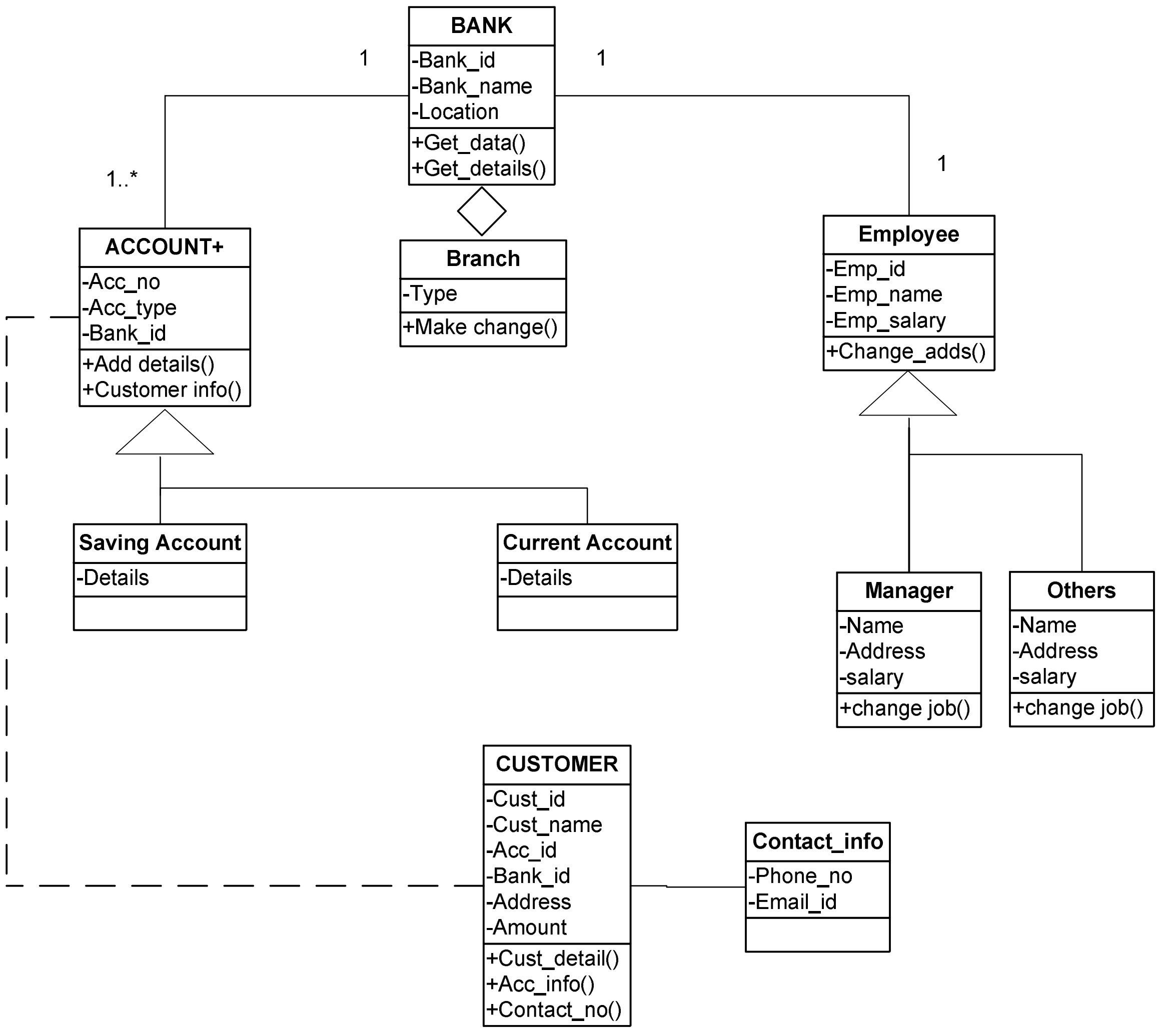
- Activity diagram:
    - Provide further details and show the flow of control among the steps of computation.
    - It shows Data flow as well as Control flow.

4. Is UML a programmming Language? Is it Process Dependent or Independent?Pen Down the names of all the UML diagram.Identify each of the UML diagram belong to structural and which of these belong to behavioral group.

Answer:No, UML is not a programming Language language.It is an Unique Modelling Language.

DIAGRAM NAME	GROUP
Class Diagram	Structural
State Diagram	Behavioral
Use Case Diagram	Behavioral
Sequence Diagram	Behavioral
Activity Diagram	Behavioral

5. Draw a class diagram for given system.
For example BANK MANAGEMENT SYSTEM



6. Explain Ordered, Sequence and Bag used in UML diagram.

Answer:

ORDERED:

- Often the objects on a “many” association end have no explicit order and we regard them as a set.
- Sometimes the objects have an explicit order.
- For example, a workstation screen containing a number of overlapping windows.
- The windows have an explicit order so only the topmost window is visible at any point on the screen.
- It is an inherent part of the association and indicated by writing “{ordered}" next to the association end.

BAG:

- A bag is a collection of elements with duplicates allowed.

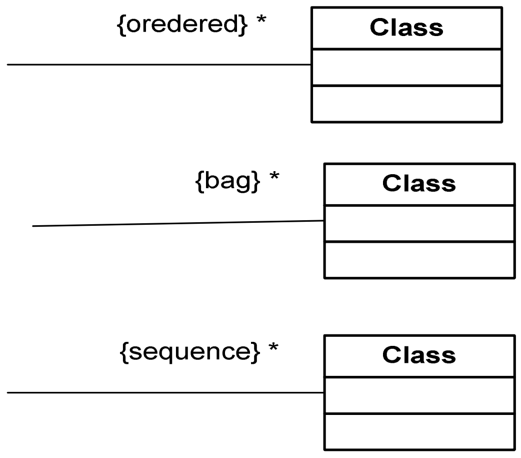
SEQUENCE:

- A sequence is an ordered collection of elements with duplicates allowed.

➢ Both are permitted only for binary association.

➢ The {ordered} and the {sequence} annotations are the same, except that the first disallows duplicates and other allows them.

SYNATAX:



7. Explain Association Class with Example.

Answer:

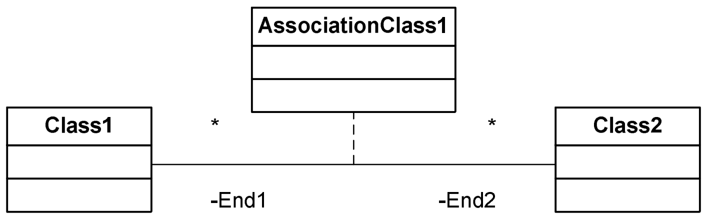
“An association class is an association that is also a class.”

Like a class an association class can have attributes and operations and participate in associations.

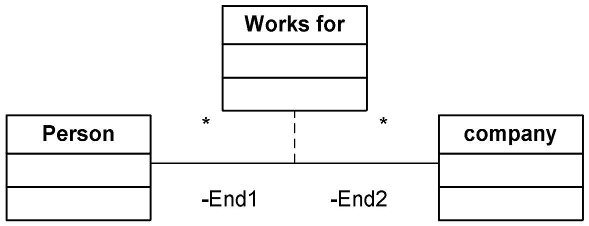
→We can find association classes by looking for adverbs in a problem statement.

→The UML notation for an association class is a box attached to the association by a dashed line.

Syntax:



Example:



8. What is Association End Name & Why is it needed? Give an Example.

Answer:

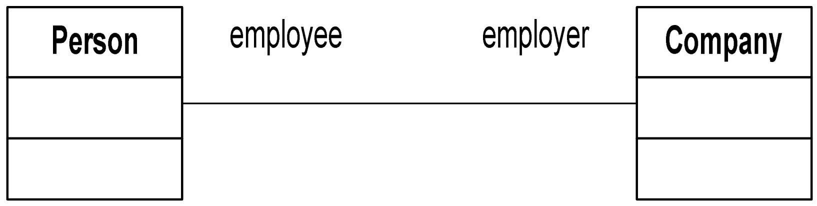
If we consider one to many association then it has two ends:

- an end with a multiplicity of “one”
- an end with a multiplicity of “many”.

We can also give the name to both the ends.

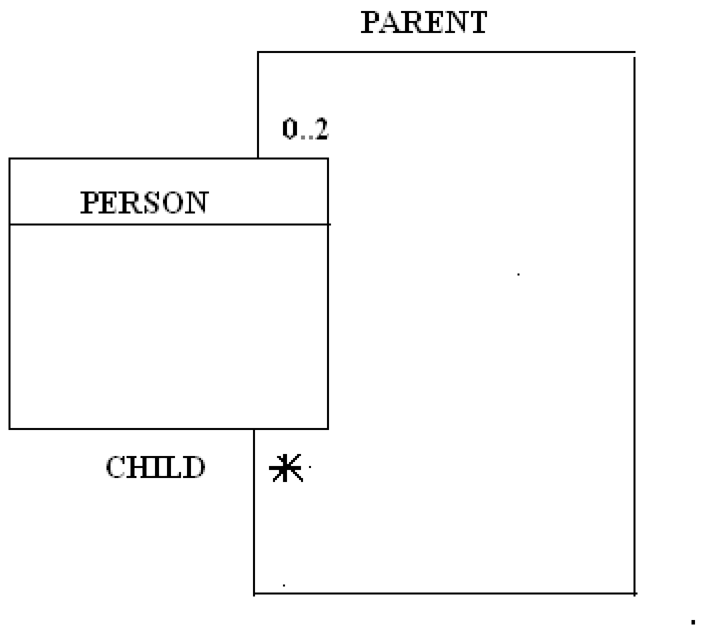
Association end names often appear as nouns in problem descriptions.

Example:



Association end names are necessary for associations between two objects of the same class.

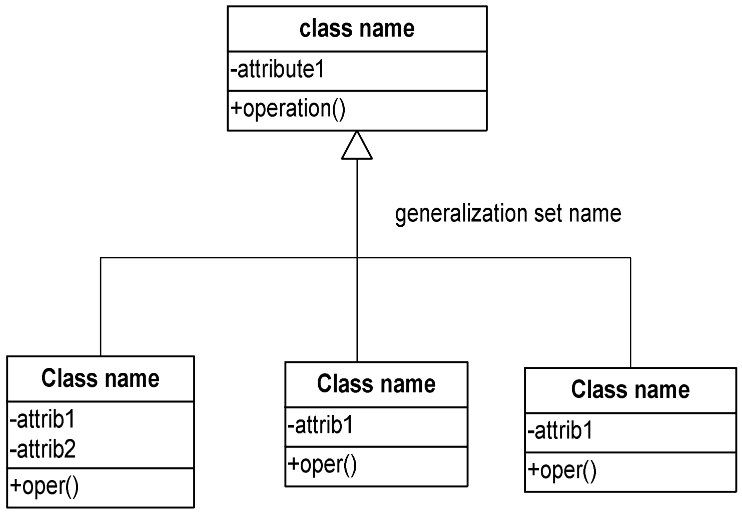
Eg.



9. Define:Genealization & its importance.

Answer:

Syntax:



- Inheritance is the mechanism for sharing attributes, operations and associations via the generalization/specialization relationship.
- Generalization is the relationship between a class (super class) and one or more variations of the class (subclasses).
- Super Class (Base class)
  - Provides common functionality and data members
- Subclass (Derived class)
  - Inherits public and protected members from the *super class*
  - Can extend or change behavior of super class by *overriding* methods
- Overriding
  - Subclass may override the behavior of its super class.

A generalization set name is an enumerated attribute that indicates which aspect of an object HAS BEEN CONSIDERED IN DOING generalization.

Advantages OF GENERALISATION:

- Polymorphism
- Find common characteristics among classes
- Define hierarchies
- Reuse of the code

10. Which UML diagram gives a Static View of a system and which give a dynamic view of a system.

Answer:See answer of ques.(4)

11. Differentiate between attribute & Association Class? When is it useful to model an association class?

Answer:

- An attribute is a named property of a class that describes a value held by each object of the class.
- E.g.
  - Name,birthdate and weight are attributes of Person class.
  - Color,model_year and weight are att. Of Car class.
- Each attribute name is unique within a class.
- In short, attribute is a member of class.

WHILE,

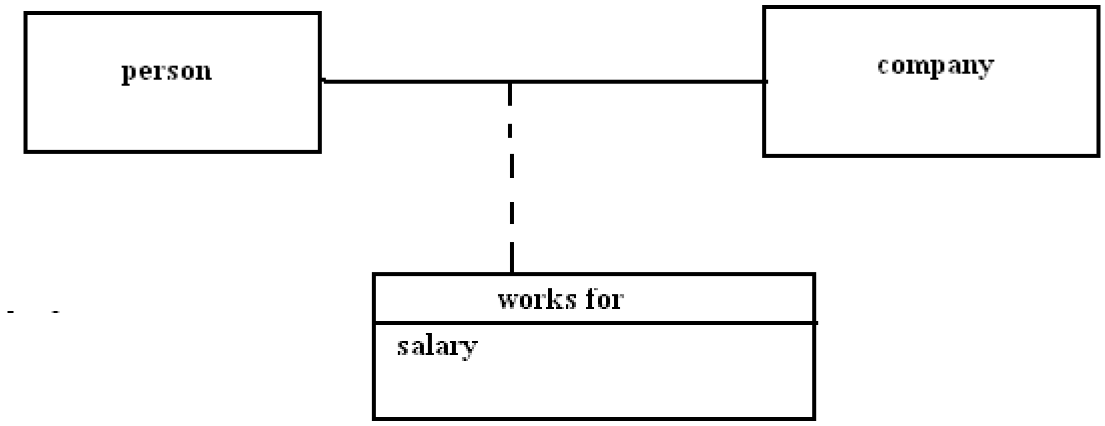
An association class is an association that is also a class.

Like a class an association class can have attributes and operations and participate in associations.

We can find association classes by looking for adverbs in a problem statement.

The UML notation for an association class is a box attached to the association by a dashed line.

Example:



IMPORTANCE:

When we want that we are not able add any attribute to either the class than we need to include that one as Association class. So, it is joint property of both classes.

12 What is the purpose of models in designing? Which diagrams show the concept of Inheritance? What are the challenges in designing with Inheritance?

Answer: There are three types of models to describe a system from different viewpoints.

So, you can say that A full description requires all three models.

A model is an abstraction of something for the purpose of understanding it before building it.

Class diagram shows the concept of Inheritance.

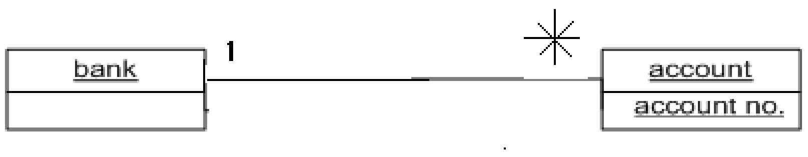
CHALLENGES:

13.What is Qualified Association? Eplain with Example.

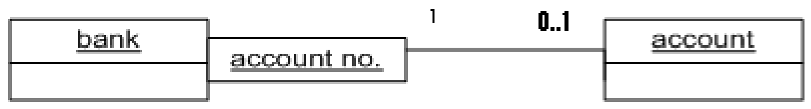
Answer: A qualified association is an association in which an attribute called the qualifier disambiguates the objects for a “many” association end.

→A qualifier selects among the target objects, reducing multiplicity from “many” to “one”.

→ Example:- A bank services multiple accounts.



Not Qualified Model



Qualified Model

→Both models are acceptable but the qualified model adds information.

→The qualified model adds a multiplicity constraint that the combination of bank and an account number yields at most one account.

→The qualified model conveys the significance of account number in traversing the model.

13 Explain: State Chart with an example.

Answer:

→The state model consists of multiple state diagrams,one for each class with temporal behavior that is important to an application.

A state diagram specifies the state sequences caused by event sequences

State names must be unique within the scope of a state diagram.

14 Draw a State Chart for a given system.

Answer: A state diagram is a graph whose nodes are states and whose directed arcs are transitions between states.

→A state diagram specifies the state sequences caused by event sequences.

→State names must be unique within the scope of a state diagram.

The UML notation for a state diagram is a rectangle with its name in a small pentagonal tag in the upper left corner.

→The constituent states and transitions lie within the rectangle.

15 Explain Terms: Event, State, Transition, Signal event, Time event, Change event.

Answer:

An Event is an occurrence at a point in time, such as user depresses left button of the mouse or flight 123 departs from Chicago.

→Events often correspond to verbs in the past tense like

- power turned on
- alarm set

Events include error conditions as well as normal occurrences. For example

- motor jammed
- transaction aborted
- timeout

→There are several kinds of events but following are most common events.

- signal event
- change event
- time event

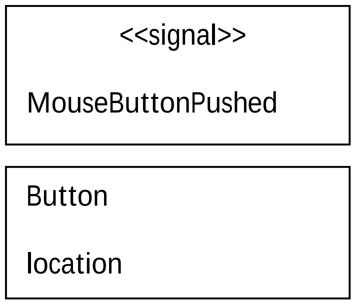
A signal is an explicit one-way transmission of information from one object to another.

→If object want to reply then use the separate signal.

→A signal event is the event of sending or receiving a signal.

→We are more concern about the receipt of a signal because it causes effects in the receiving object.

The UML notation is the keyword signal in guillemets(<<>>) above the signal class name in the top section of a box.



Change Event:

A change event is an event that is caused by the satisfaction of a boolean expression.

→The intent of a change event is that the expression is continually tested whenever the expression changes from false to true, the event happens.

→The UML notation for a change event is the keyword "*when*" followed by a parenthesized boolean expression.

→Examples:

- When (room_temp<heating_set_pt)
- When (bat_power < lower_limit)
- When (tire_pressure < min.pressure)

Time Event:

A time event is an event caused by the occurrence of an absolute time or the elapse of a time interval.

→The UML notation for an absolute time is the keyword when followed by a parenthesized expression involving time.

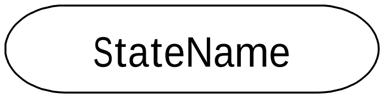
→The notation for a time interval is the keyword after followed by a parenthesized expression that evaluates to a time duration.

→Examples:

- When (date=Jan 1,2000)
- After (10 seconds)

STATE:

- A state is an abstraction of the values and links of an object.
- Sets of values and links are grouped together into a state according to the gross behavior of objects.
- States often correspond to verbs with a suffix of "ing" (Waiting, Dialing) or the duration of some condition (Powered, BelowFreezing).
- A state specifies the response of an object to input events.
- The response may include the invocation of behavior or a change of state.
- UML notation for a state is a rounded box containing an optional state name.



- examples:

Transitions and conditions:

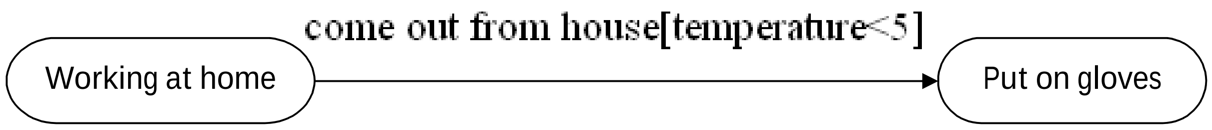
A transition is an instantaneous change from one state to another.

→For example, when a called phone is answered, the phone line transition from the Ringing state to the Connected state.

→The transition is said to fire upon the change from the source state to the target state.

→The choice of the next state depends on both the original state and the event received.

The UML notation for a transition is a line from origin state to target state.



A guard condition is a boolean expression that must be true in order for a transition to occur.

→A guarded transition fires when its event occurs, but only if the guard condition is true.

→For example "when you go out in the morning (event), if the temperature is below freezing (condition), then put on your gloves (next state)."

A guard condition is checked only once while a change event is in effect checked continuously.

An event may label the transition and be followed by an optional guard condition in square brackets.

16 Define:Use case,Actor

Answer:

<u>A use case</u> is a coherent piece of functionality that a system can provide by interacting with actors.

Use cases describe how a system interacts with outside actors.

<u>An actor</u> is a direct external user of a system.

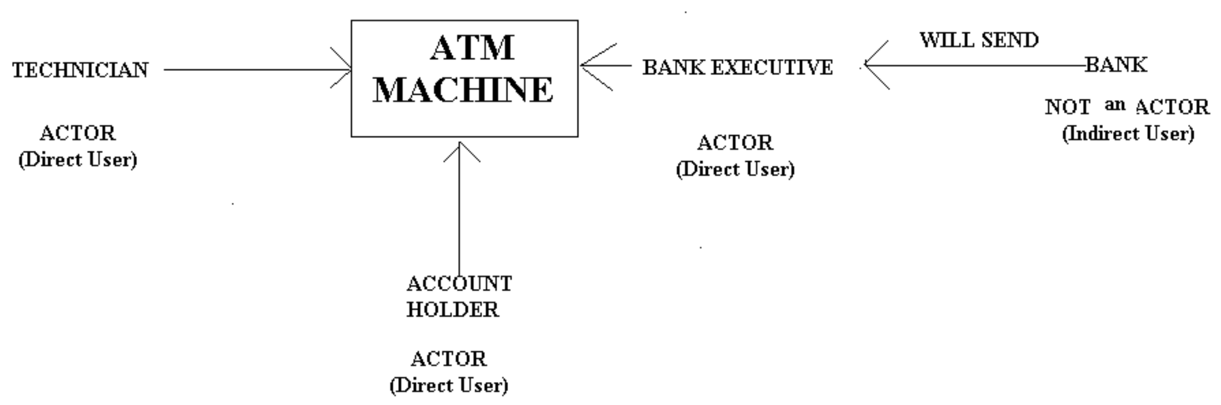
An object or set of objects that communicates directly with the system but that is not part of the system.

Actors can be persons, devices, and other systems means anything that interacts directly with the system.

An actor is directly connected to the system and the object indirectly connected to the system is not called an actor.

Eg.

.



17 Draw Usecase for given system.

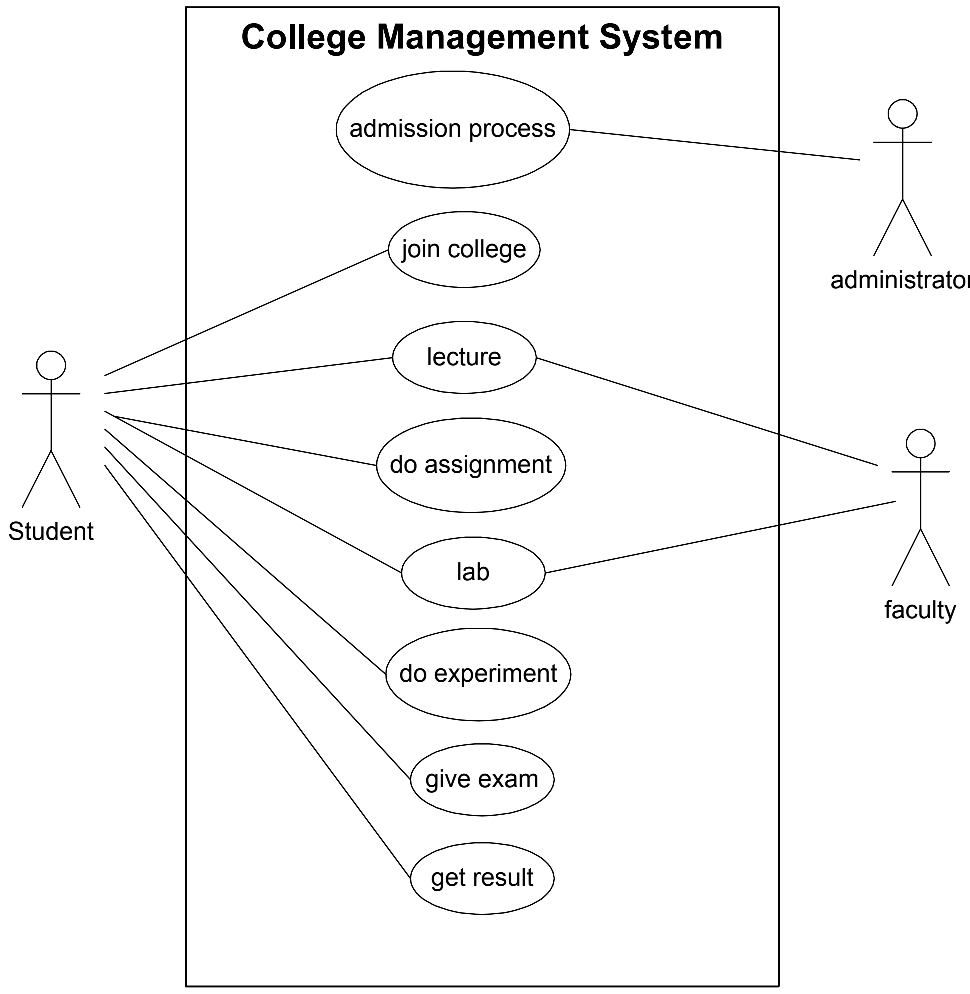
Answer: Use Case Diagrams:

A system involves a set of use cases and a set of actors.

Each use case represents the functionality OF the system .

Each actor represents one type of object for which the system can perform behavior(USECASE).

DIAGRAM of college management system:


College Management System
admission process
join college
lecture
do assignment
lab
do experiment
give exam
get result
Student
administrator
faculty

18 Draw Scenario for given system.

Answer:

There are two kinds of sequence models:

- scenarios
- sequence diagrams.

A scenario is a sequence of events that occurs during one particular execution of a system such as for a use case.

A Scenario can be displayed as a list of text statements

Each scenario has a sequence of steps.

Eg.

<u>SCENARIO of hostel management system:</u>

- Student applies to take admission in hostel.
- Rector collects details to give admission.
- Student submits all details to get admission.
- Administrator gives admission after verifying availability of rooms.
- Rector takes attendance daily.
- Administrator provides mess facilities.
- Administrator request for mess bill report, containing details of monthly mess expenditure.
- Rector gives detailed mess report.
- Rector gives monthly bill to respective students.
- Students pays bill if no fine is charged.
- Students pay additional charge along with bill if fined due to late payment.
- Rector gives receipt on receiving payment to keep record of payment.
- Rector gives the collected amount.

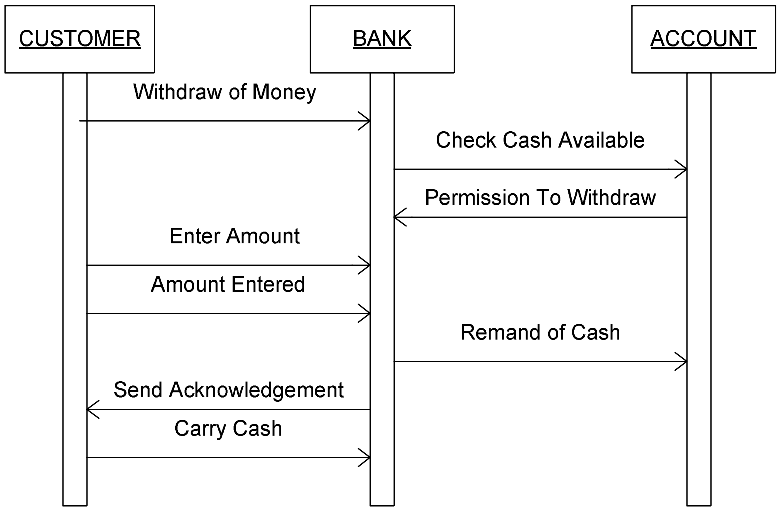
19 Draw a Sequence diagram for given system.

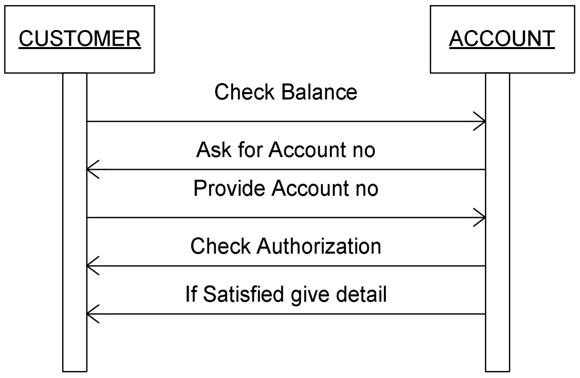
Answer: A text format is convenient for writing but it cumbersome & slower procedure when you attempt to understand .

→A sequence diagram shows the objects (participants) and the sequence of messages passing among them.

→In seq. diagram, each actor as well as the system is represented by a vertical line called a lifeline and each message by a horizontal arrow from the sender to the receiver.

BANK SEQUENCE DIAGRAM:





23.what do you mean by Interaction Modelling?How it is useful?

Answer:

INTERACTION Model:

it describes interactions between objects that how they will communicate when system is working.

Interactions can be modeled at different levels of abstraction.

USE CASE(High level):How system WILL interact with outside users.

SEQUENCE DIAGRAM:PROVIDE MORE DETAIL and shows the messages exchanged among a set of objects over time.

Activity diagram:

Provide s further details and show the flow of control among the steps of computation.

It shows Data flow as well as Control flow.

11.All the useful detail of class diagram and usefull notations.

A class model captures the static structure of a system by characterizing the objects in the system, the relationships between the objects and the attributes and operations for each class of objects.

- → Class model provides a graphic representation of a system and are used for communicating with customers.

Class diagrams provide a graphic notation for modeling classes and their relationships thereby describing possible objects.

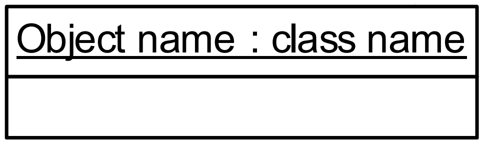
- → Useful for abstract modeling and designing actual programs.
- → They are concise ,easy to understand

<u>Notation used for class & object diagram:</u>

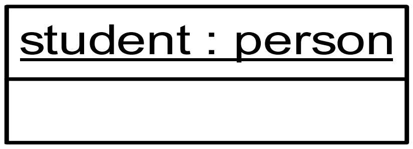
- ❖ <u>Object:</u>

An object is a concept, abstract or any thing that has meaning for an application. All object has identity and distinguishable.

Syntax:



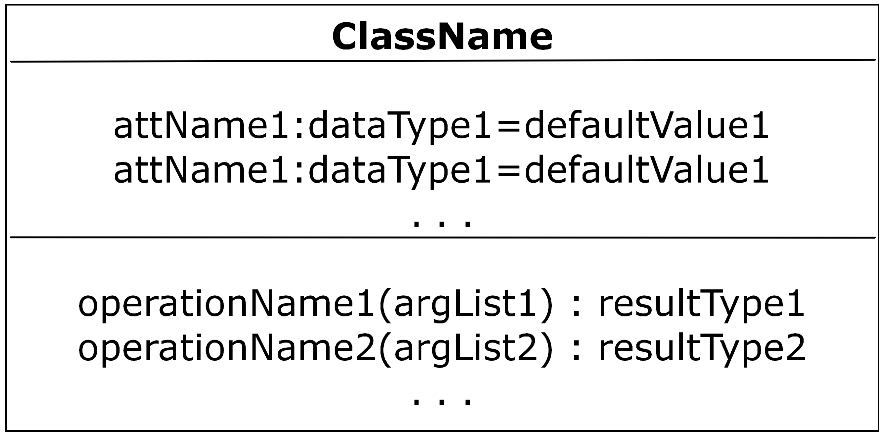
Example:



❖ Class:

A class describes a group of objects with the same properties, behaviors, and semantics.person, company, and windows are all classes.

Syntax:


ClassName
attName1:dataType1=defaultValue1
attName1:dataType1=defaultValue1
. . .
operationName1(argList1) : resultType1
operationName2(argList2) : resultType2
. . .

An attribute is a named property of a class that describes a value held by each object of the class.

→ E.g. Name,birthdate and weight are attributes of Person class.

- Color,modelyear and weight are att. Of Car class.

→ Each attribute name is unique within a class.

So Person and Car class have attribute called weight.

An Operation is a procedure that an object performs or is subject to performs.

→ E.g. Open, close , hide and redisplay are operations on class window.

→ All objects in a class share the same operations.

→ Each operation has a target object as an implicit argument.

→ The behavior of the operation depends on the class of its target.

→ The same operation may apply to many different classes such an operation is polymorphic. (same operation takes different forms in different classes.)

→ The attribute and operation compartments are optional.

→ A missing attribute / operation compartment means that attributes / operations are unspecified.

→ An empty compartment means attributes/operations are specified and that are none.

A link is a physical or conceptual connection among objects.
→Most link relate two objects but some links relate three or more objects.

e.g. Smith works for simplex company.

- → A link is an instant of an association.
- → An Association is a description of a group of links with common structure and common semantics.
- → A semantic relationship between two or more classes that specifies connections among their instances.
- → A structural relationship, specifying that objects of one class are connected to objects of a second class.

Ambiguity arises when a model has multiple associations among the classes.

For example "person works for a company and person owns stocks in company"

- → To solve it we must have to use association name.
- → Associations are inherently bidirectional.
- → Multiplicity specifies the number of instances of one class that may relate to a single instance of an associated class.
- → UML diagrams explicitly list multiplicity at the end of the association lines.
- → Multiplicity Indicators

Exactly one	1
Zero or more (unlimited)	* (0..*)
One or more	1..*
Zero or one (both inclusive)	0..1
Specified range	2..4
Multiple, disjoint ranges	2, 4..6, 8

- → is the WorksFor association between Person and Company one-to-many or many-to-many ?

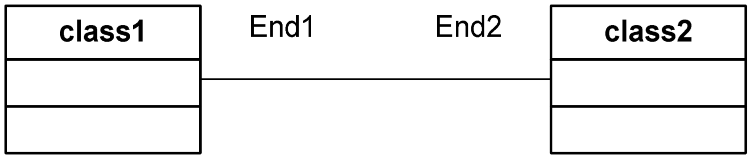
It depends on the context (situation).

❖ Association:

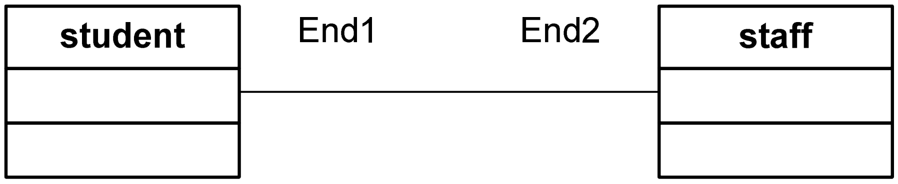
An association is a description of a group of links with common structure and common semantics.

An association describes a set of potential link.

Syntax:



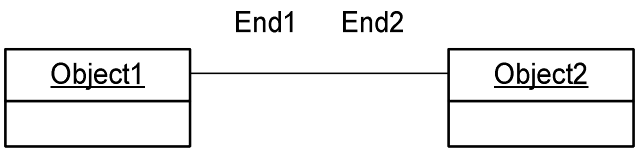
Example:



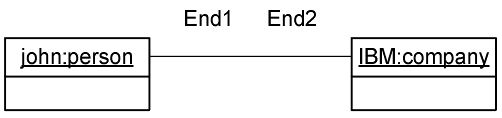
❖ Link:

A link is a physical or conceptual connection among objects.

Syntax:



Example:

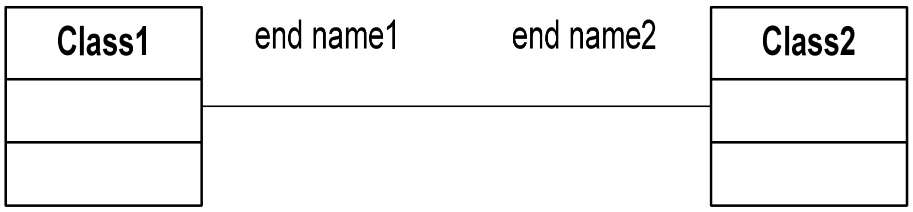


❖ <u>Association End Names:</u>

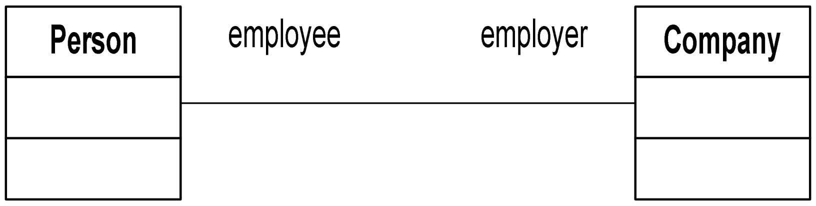
Association end names are necessary for association between two objects of the same class.

It also distinguishes multiple association s between the same pair of the class.

Syntax:



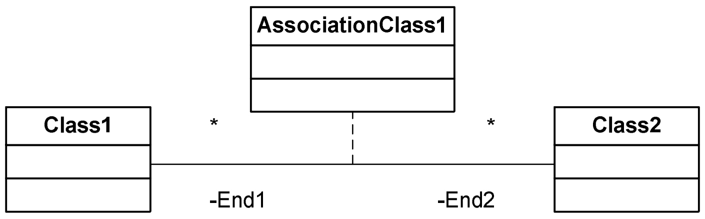
Example :



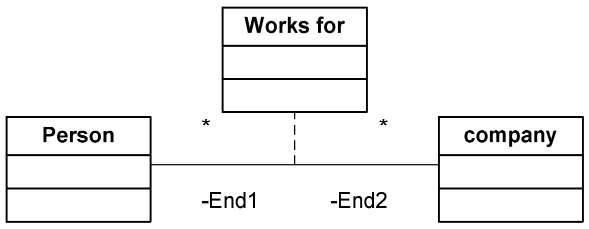
❖ <u>Association Class:</u>

An association class is an association that is also a class. like a class ,an association class can have attributes and operation and participate in association.

Syntax:



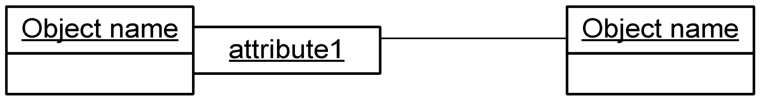
Example:



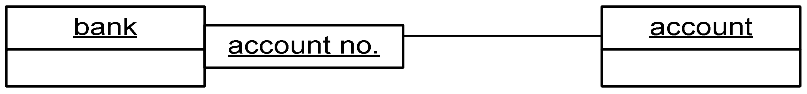
❖ Qualified Association:

A qualified association is an association in which an attribute called the qualifier disambiguates the object for a "many" association end. It is possible to define qualifier for one-to-many and many-to-many associations.

Syntax:



Example:



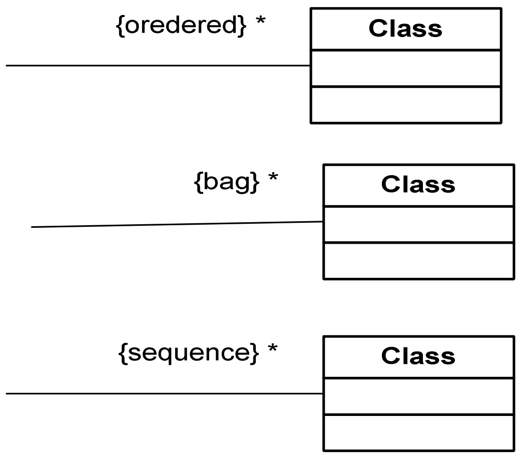
❖ Order , Bag and Sequence:

- Often the objects on a “many” association end have no explicit order and we regard them as a set. Sometimes the objects have an explicit order.
- A bag is a collection of elements with duplicates allowed.
- A sequence is an ordered collection of elements with duplicates allowed.

Both are permitted only for binary association.

- The {ordered} and the {sequence} notations are the same, except that the first disallows duplicates and other allows them.

Syntax :

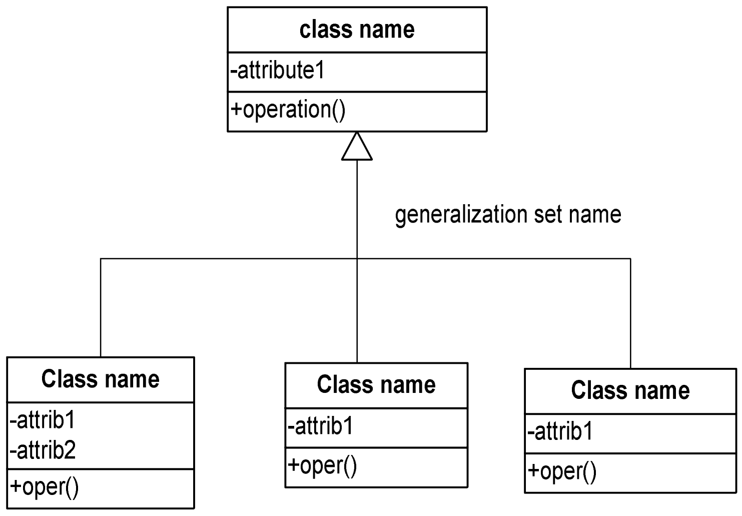


❖ <u>Generalization:</u>

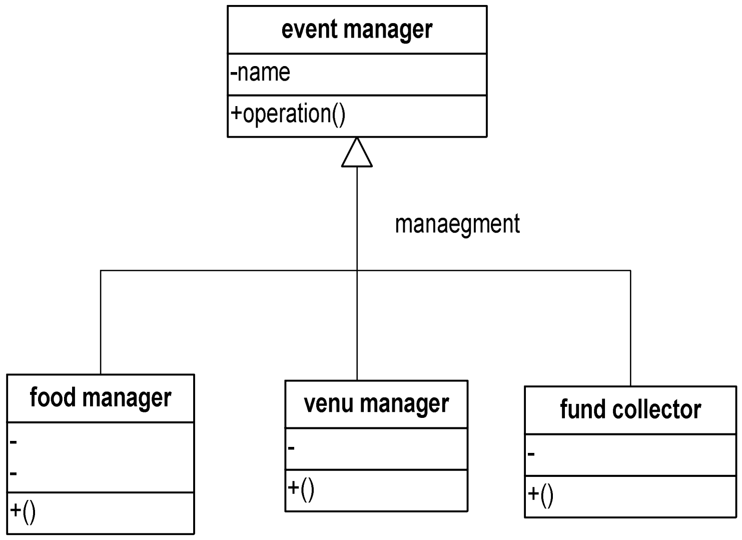
Generalization is a relationship between a class and one or more variation of the class.

The super class holds common attributes, operations, and association the sub class add specific attributes, operation.

Syntax:



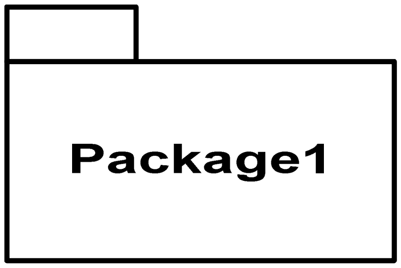
Example:



❖ <u>Package :</u>

It is a group of class.

Syntax:


Package1