**object-oriented development**

object-oriented analysis ooa is concerned with developing an object model of the application domain.

object-oriented design ood is concerned with developing an object-oriented system model to implement requirements.

object-oriented programming oop is concerned with realising an ood using oo programming language such as java or c++

**characteristics of ood**

objects are abstractions of real-world or system entities and manage themselves.

objects are independent and encapsulate state and representation information.

system functionality is expressed in terms of object services

objects communicate by message passing

objects may be distributed and may execute sequentially or parallel

**advantages of ood**

easier maintenance.objects may be understood as stand-alone entities.

objects are potentially reusable components

for some systems, there may be obvious mapping from real world entities to system objects.

**objects and object classes**

objects are entities in a software system which represent instances of real-world and system entities.

object classes are templates for objects.they may be used to create objects.

object classes may inherit attributes and services from other object classes.

an object is an entity that has a state and a defined set of operations which operate on that state.the state is represented as a set of object attributes. the operations associated with the object provied services to other objects (clients) which request these services when some computation is required.

object are created according to some object class definition.an object class definition serves as template for objects.it includes declarations of all the attributes and services which should be associated with an object of that class.

**object approach**

a natural way to view design

children first talk with things(objects)

you deal with your environment as objects

we model a system as a number of objects that interact with each other

similar to entity-relationship approach

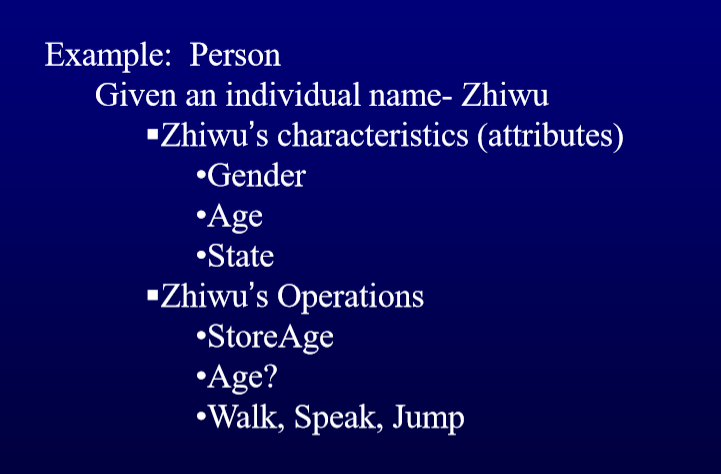
object orientation is a technique for system modeling

**object's characteristics**

identity:a way to distinguish between occurrences of objects

behavior:how the object acts and reacts in terms of state changes and messages(method)

state:the state of an object encompasses all of the (usually static) properties of the object plus the current (usually dynamic) values of each of these properties



**principles of object orientation**

**major elements**

**abstraction**

an abstraction denotes the essential characteristics of an object that distinguish it from all of other kinds of object that distinguish it from all of other kinds of objects and thus provide clear defined conceptual boundaries, relative to the persepective of the viewer.

**encapsulation**

encapsulation is the process of compartmentalizing the elements of an abstraction that constitutes its structure and behavior; encapsulation serves to separate the contractual interface of an abstraction and its implementation

**modularity**

modularity is the property of a system that has been decomposed into a set of cohesive and loosely coupled modules

**hierarchy**

hierarchy is a ranking or ordering of abstractions

**two types of hierarchy**

"is a" -- Inheritance

inheritance defines a relationship among classes ,where in one class shared the structure or behavior defined in one or more objects.

sementically,inheritance denotes an "is-a" relationship

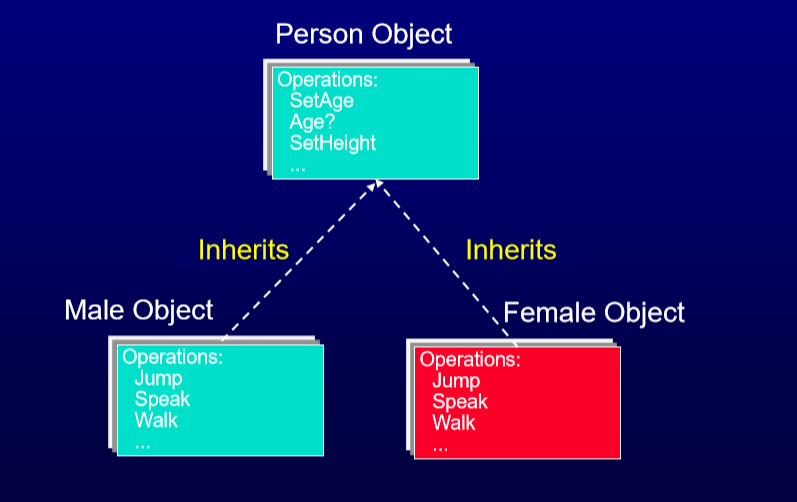
inheritance implies generalization/specialization

"part of" -- aggregation

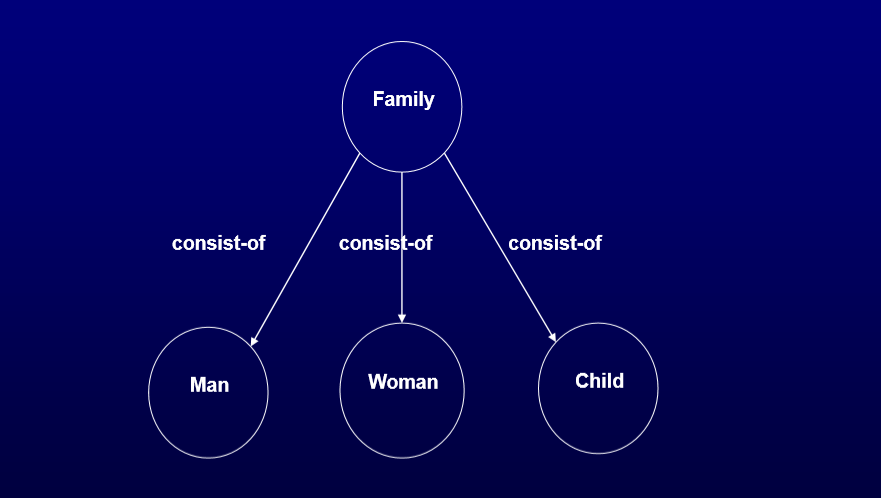
permits the physical grouping

aggregation raises the issue of ownership

object inheritance - hierarchy



aggregation - hierarchy

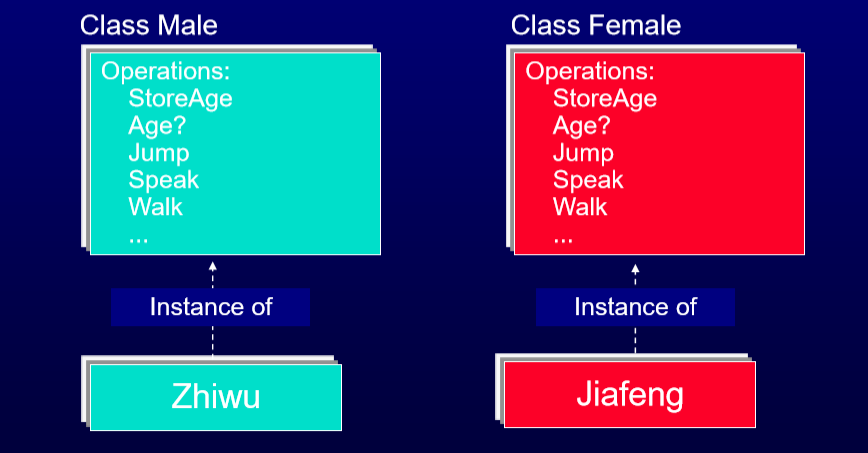


**class and object**

a class represents a template for several objects and describes how these objects are structuresd internally

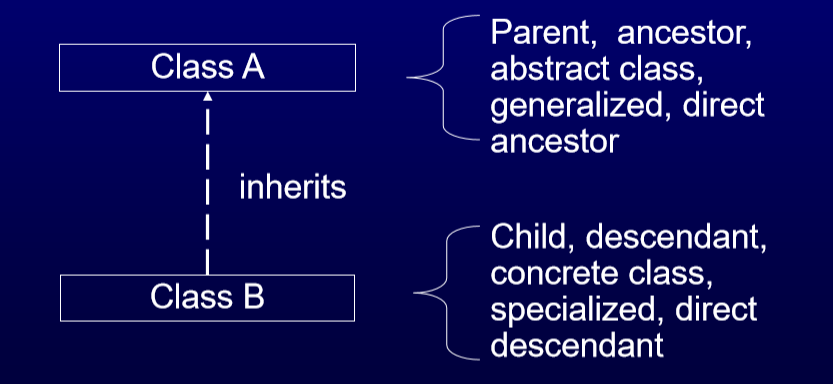
an instance is an object created from a class

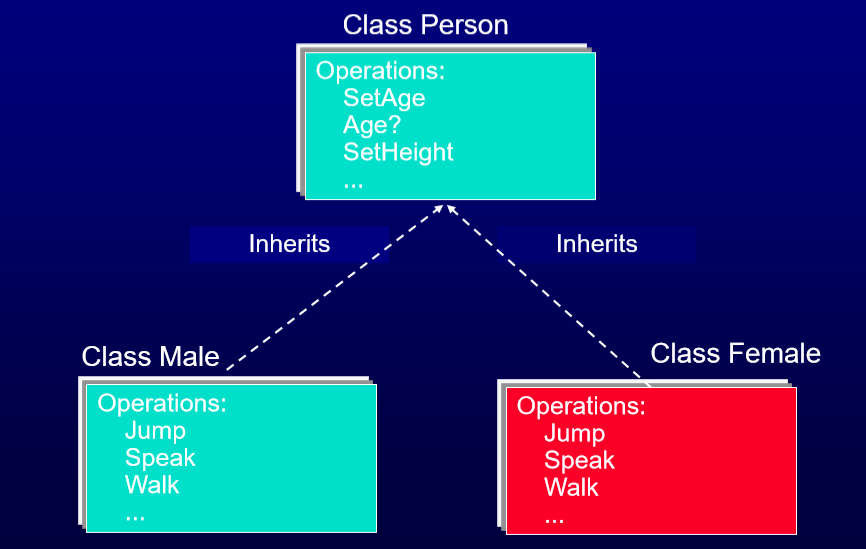
the class describes the (behavior and information) structure of the instance, while the current state of the instance is defined by the operations performed on the instance

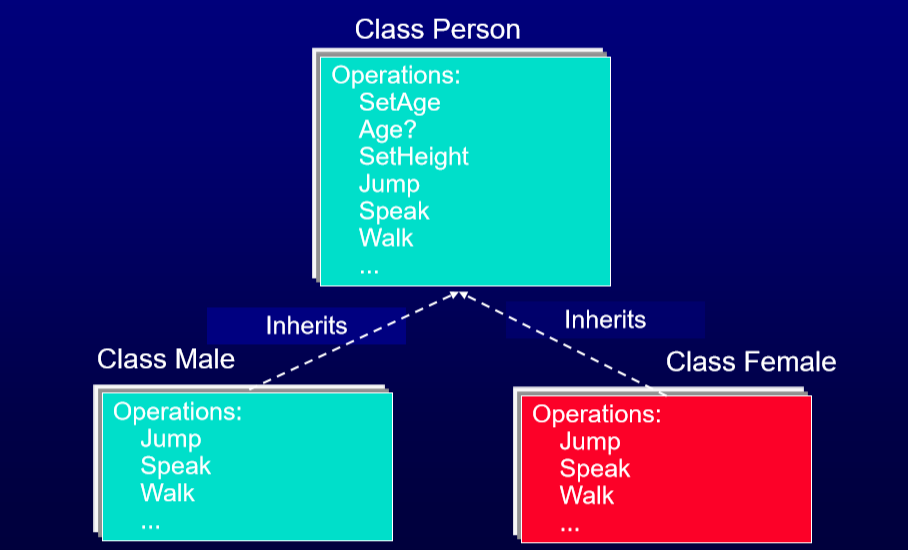


generalization of a group of classes

collect the common characteristics into one specific class and let the original classes inherit this class







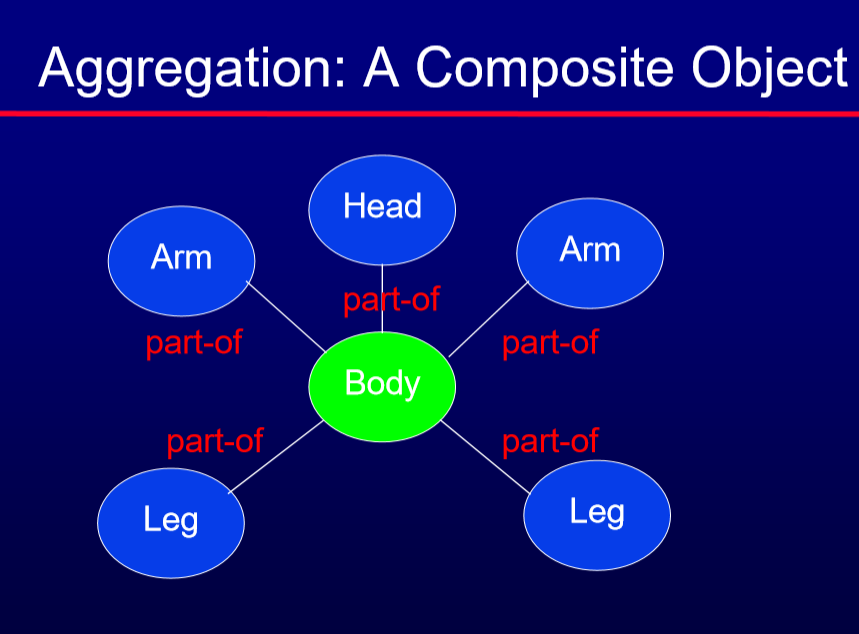
**object relationships types**

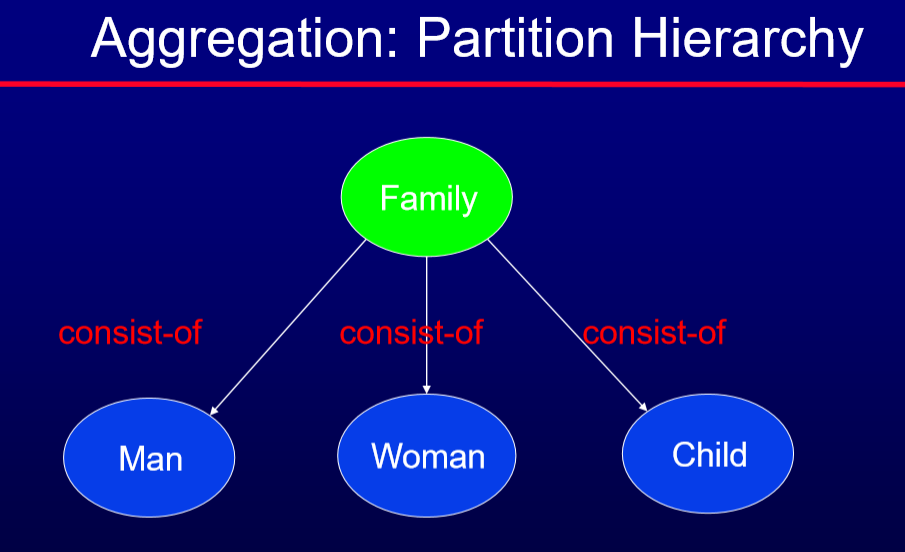
association: has the same characteristics as the relationships in an e-r diagram.

inheritance: a static generalization

aggregation: a set of objects that comprise a larger whole.

using: a peer link where one object supplies a set of services to another





**an object-oriented desgin process**

structured design processes involve developing a number of different system models.

they require a lot of effort for development and maintenance of these models and, for small systems, this may not be cost-effective

however, for large systems developed by different groups design models are an essential communication mechanism

**process stages**

highlights key activities without being tied to any proprietary process

define the context and modes of use of the system.

design the system architecture

identify the prinicipal system objects

develop design models

specify object interfaces