Topic2: Object oriented Principles and Concepts

-Features of OO approach

-OO principles

-OO abstraction techniques

-OO concepts

-Benefits of OO approach

2.1 Features of OO Approach

- Shared data areas are eliminated.
  – Objects communicate by exchanging messages and reduces *system coupling*.
- Objects are independent entities.
  – can exist separately and enhances *reuse*.
- Directly maps problem domain into a model.
  – instead to functions or data flows.
- Clear mapping between real-world concepts and objects within the system.
  – improves understandability of solution

2.2 Object oriented Principles

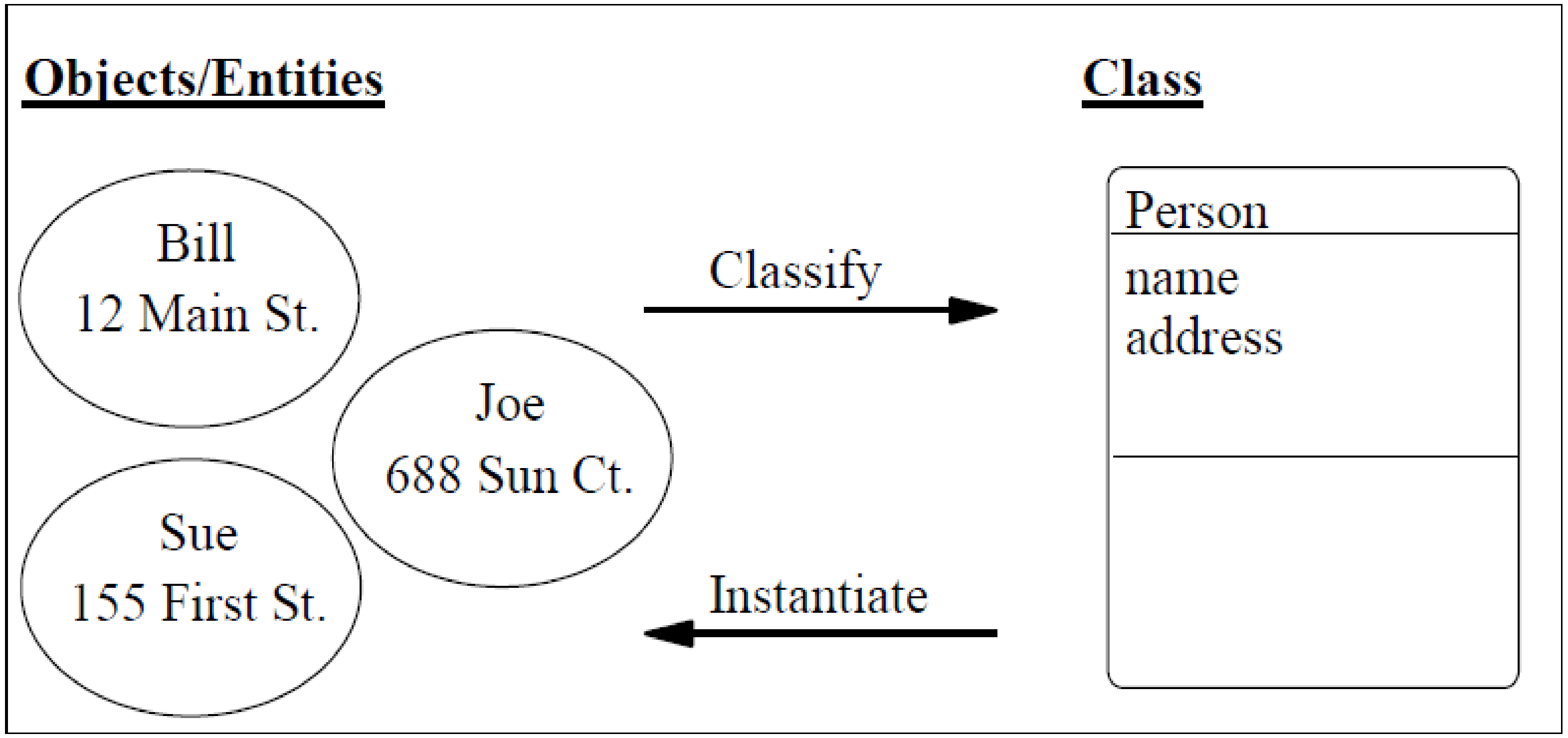
- Abstraction –*foussing on most important aspects while ignoring less important details*
- Encapsulation –*hiding implementation details using an interface from users*
- Modularity –*breaking complex system into small self-contained pieces that can be managed independently*
- Hierarchy –*ordering abstractions into a tree like structure*

2.3 Object oriented Abstraction Techniques

- Classification
- Inheritance
- Encapsulation
- Polymorphism
- Aggregation
- Association
- Collaboration

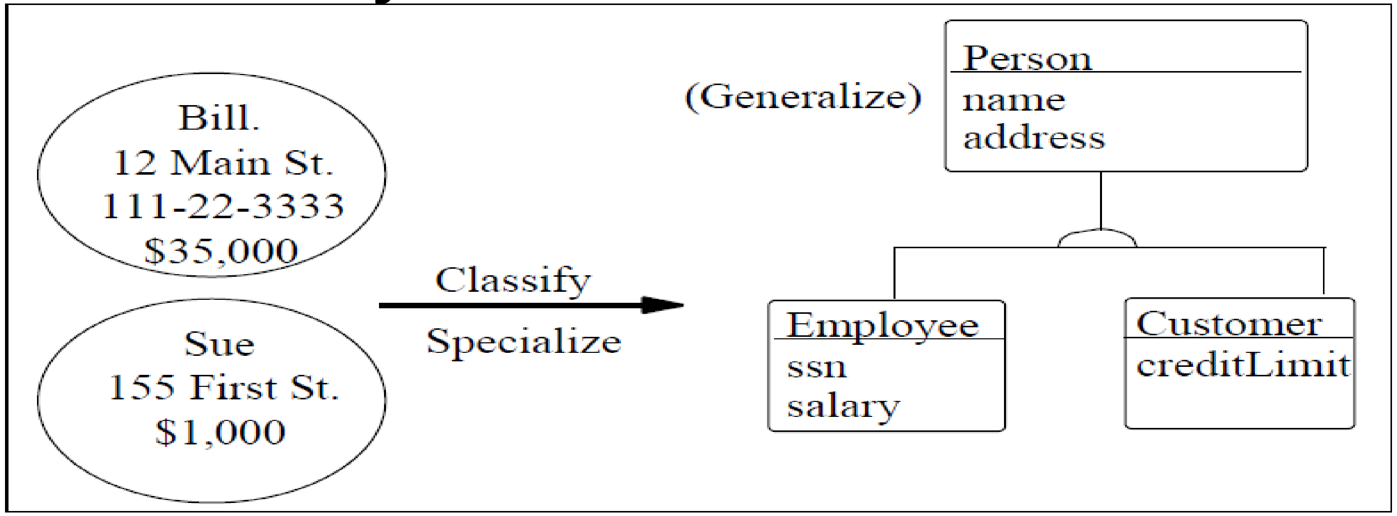
2.31 Classification

- Classification is used to group entities that share common characteristics into a class over which uniform conditions hold.



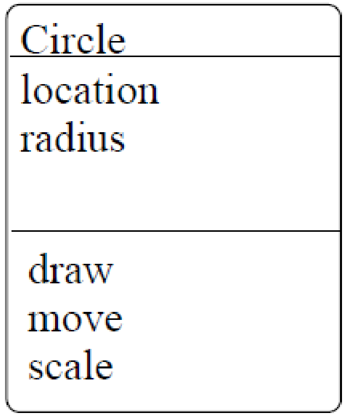
2.32 Inheritance

- A mechanism for expressing similarity among classes.
- It portrays generalization (**What is the same?**) and specialization (**What is different?**), making common attributes and services explicit within a class hierarchy.



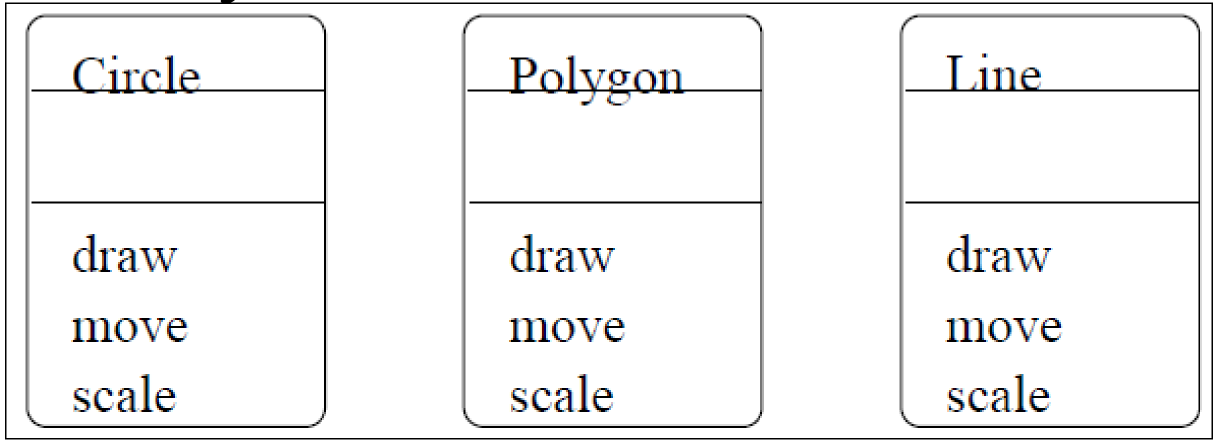
2.33 Encapsulation

- Encapsulation is a mechanism that binds/wraps together code and data it manipulates and keeps both safe from outside interference and misuse.
- In the following, only the services move(l) and scale(r) modify the attributes of Circle.
- The service draw() performs some computation based on the values of the attributes.



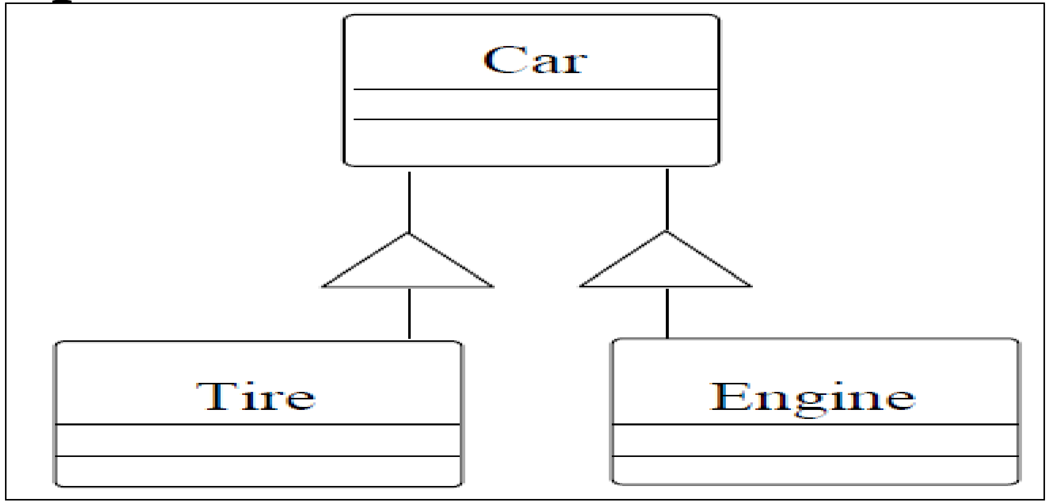
2.34 Polymorphism

- Polymorphism is the quality that allows one name to be used for two or more related but technically different purposes.
- In the following, each graphical object has the same services, although they are implemented differently.



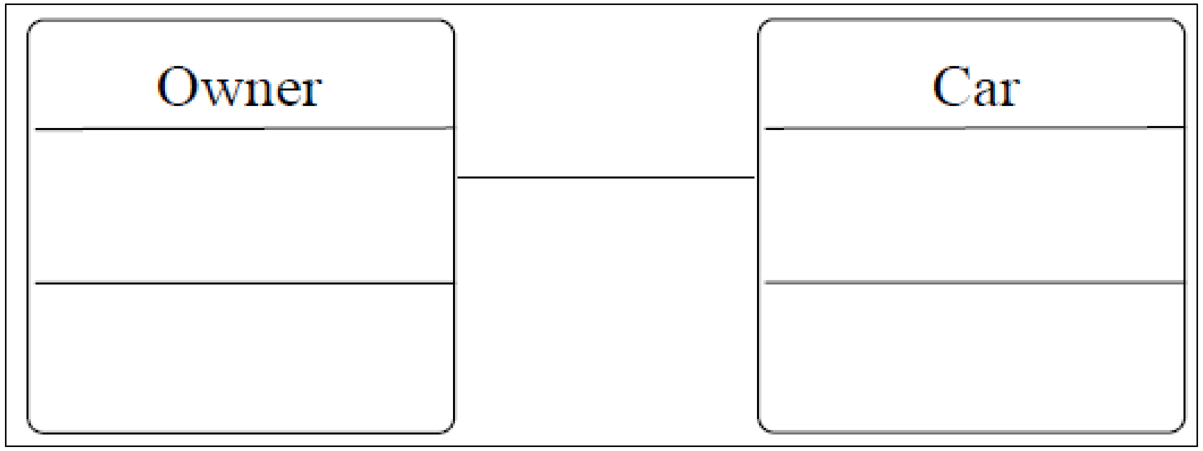
2.35 Aggregation

- Aggregation is used to treat a collection of objects as a single object.
- For example, among other things, a car consists of tyres and an engine.
- Note that the opposite of aggregation is decomposition.



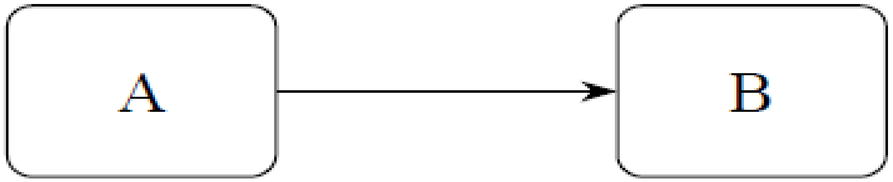
2.36 Association

- An association is a data-oriented relationship between two entities that signals one uses the other.
- For example, the following relationship models the concept that if there is a car, it must be associated with an owner i.e. owner uses car.



2.37 Collaboration

- Collaboration is co-operation between classes that is achieved through message passing.
- This documents dependencies between classes by answering the questions.
    - -What help do I need?
    - -Who needs my help?
- At some point, class **A** sends one or more messages to class **B**.



2.4 Object oriented Concepts

- *objects*
- *classes*
- *attributes*
- *operations*
- *interfcaes*
- *relationships*

2.41 object

- Any concept that represents a single thing or a specific entity in the real world.
- An object is a unique entity with a unique state and behavior that determine its identity.
- may be tangible (physical entity) or intangible
- is graphically denoted by a a rectangle with three partitions indicating objectName, state, behavior.

Contd..

objectName
State
Behavior

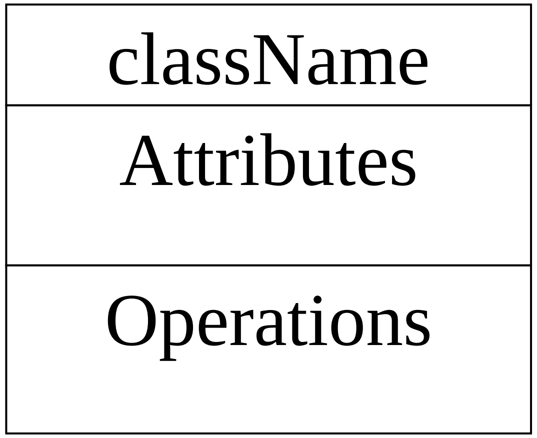
e.g

Tom
A male 21 years
Can draw Can teach

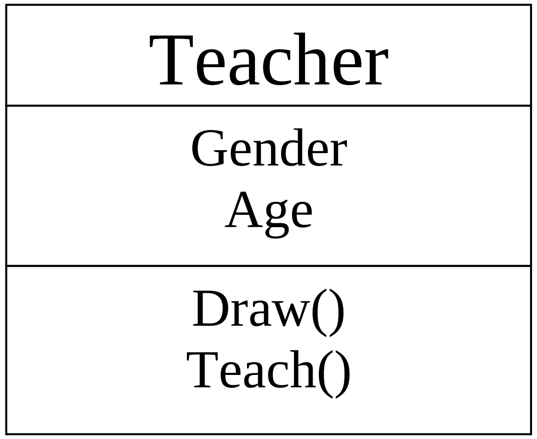
2.42 class

- Concept that represents a set of logically related objects that share similar characteristics
- A definition or template that describes accurate representation of specific type of objects
- Objects are created using class definitions as templates.
- a class is graphically denoted by a a rectangle with three partitions indicating name of class, attributes, operations

Contd….

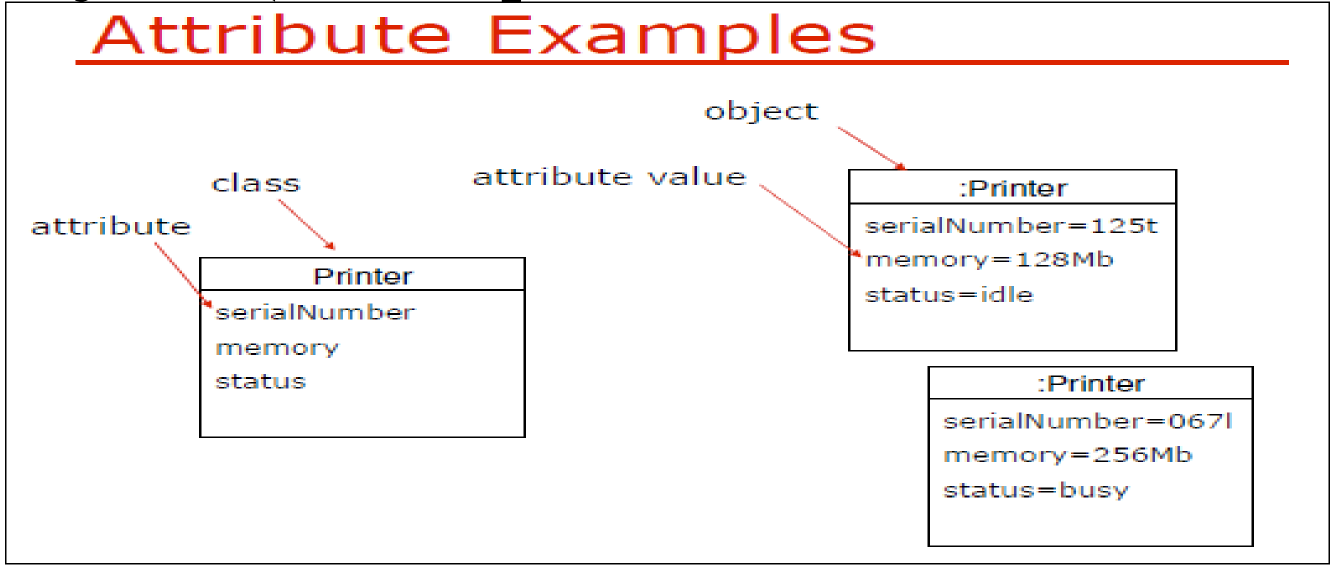

className
Attributes
Operations

E,g


Teacher
Gender
Age
Draw()
Teach()

2.43 attribute

- A named property of a class describing a range of values that instances/objects of the class may hold as state for that property.
- The set of attribute values defines the state of the object. (i.e. implemented as data members)



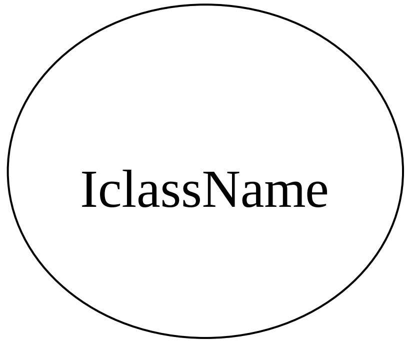
2.44 operation

- A concept that models behavior/service that can be requested from any object of a given class (i.e. implemented as method members)
- An operation could be:

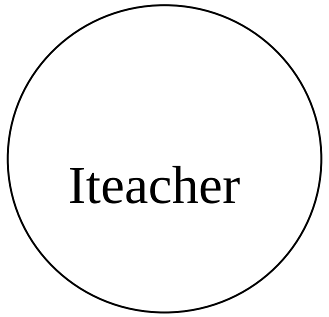
1. a question - does not change the values of the attributes
2. a command – may change the values of the attributes

2.45 interface

- Collection of operations that specifies externally visible behaviour/service of a class
- Defines a set of operation signatures but not their implementations (methods)
- Denoted as a circle with a name that reflects the name of the class to which the interface belongs and prefix I i.e. IclassName

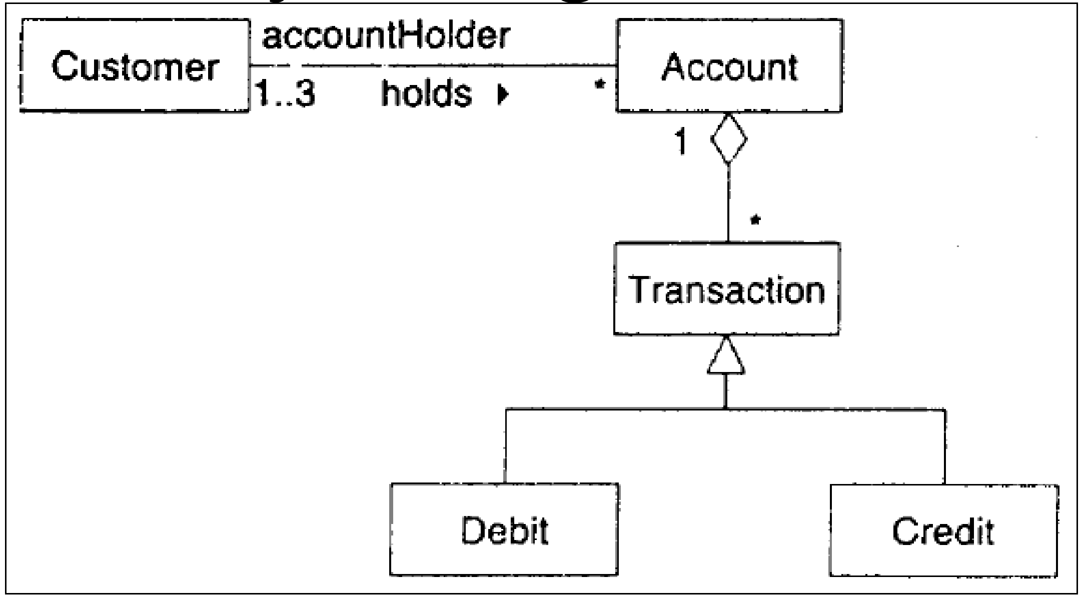


e.g



2.46 Relationship

- Connections between classes and objects that help to bind them together.
- Relationship is a concept that helps to declare inheritance or signal potential association or collaboration through message passing between classes or objects. E.g



2.5 Benefits of Object Oriented Approach

- *maintainability* – modularity ensures errors are localized in objects and easy to fix
- *reusability* –self-contained and independence property of objects makes them transferable
- *productivity* –direct mapping of design concepts into features in the programming languages
- *reliability* –object encapsulation ensures no inteference of software units
- *security* –information hiding ensures safety and integrity of data in the software