CS F213 - Object Oriented Programming

J. Jennifer Ranjani

email: jennifer.ranjani@pilani.bits-pilani.ac.in

Chamber: 6121 P, NAB

Consultation: Appointment by e-mail

https://github.com/JenniferRanjani/Object-Oriented-Programming-with-Java

BITS Pilani

Pilani Campus

Solution to Review Questions

1. Is Container is a subtype of Component ?. *TRUE*
2. Is JButton is a subtype of Component ? *TRUE*
3. Is FlowLayout is a subtype of LayoutManager? *TRUE*
4. Is ListIterator is a subtype of Iterator ? *TRUE*
5. Is Rectangle[ ] is a subtype of Shape[ ] ? *TRUE*
6. Is int[ ] is a subtype of Object ? *TRUE*
7. Is int is subtype of long ? *NO*
8. Is long is a subtype of int ? *NO*
9. Is int is a subtype of Object ? *NO*

Primitive Types are not implemented as Objects

Enum Type

- Used for representing a group of named constants in programming
- Enum in java is more powerful than C/C++
- In Java, we can add variables, methods and constructors to it.
- Enum can be declared outside the class or inside the class but not inside the method.

```
class Test{
enum Color{
 RED, GREEN, BLUE;
 }
 public static void main(String[] args) {
    }
}
```

Features of enum

- Enum is internally implemented using class

```
/* internally above enum Color is converted to
class Color {
public static final Color RED = new Color();
public static final Color BLUE = new Color();
public static final Color GREEN = new Color(); }*/
```

- Constants represents an object of type enum
- Constants are always implicitly public static final
  - It can be accessed using enum name
  - Child enums can not be created.
- It can be passed as an argument to switch statements

Features of enum

- All enums implicitly extend java.lang.Enum class
- toString() returns the enum constant name
- values() method can be used to return all values present inside enum
- ordinal() method is used to retrieve the constant index
- Enum can contain constructor and it is executed separately for each enum constant at the time of class loading.
- We cant create enum objects explicitly and hence we cannot invoke the enum constructor directly
- Enum can contain concrete method and not abstract methods.

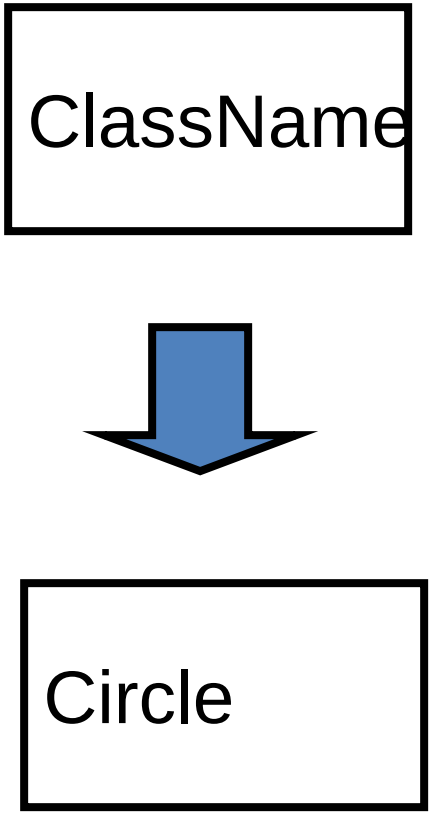
Class Diagram

Modelling a class in UML

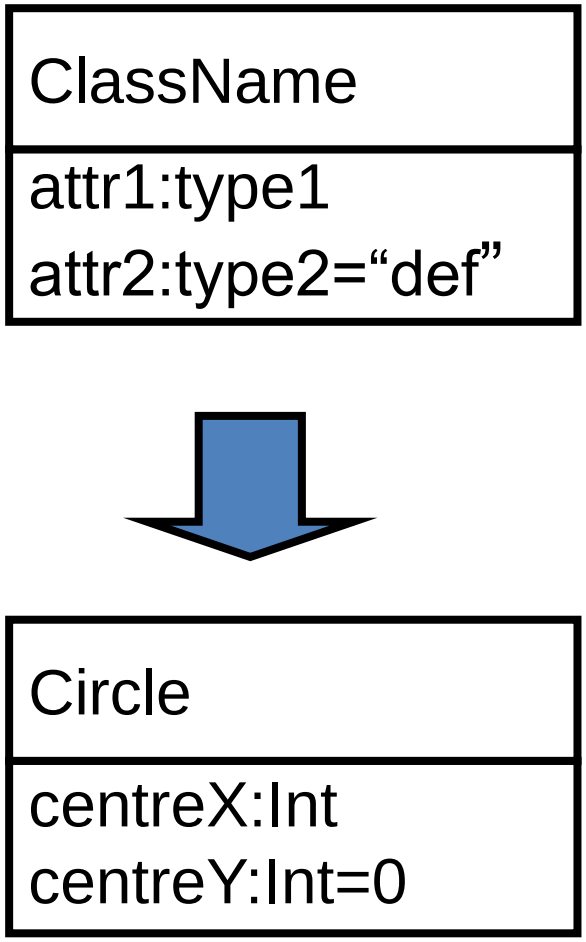
- Class can be modeled in four different ways:
  - With no attributes or operations shown
  - With only the attributes shown
  - With only the operations shown
  - With both the attributes and operations shown

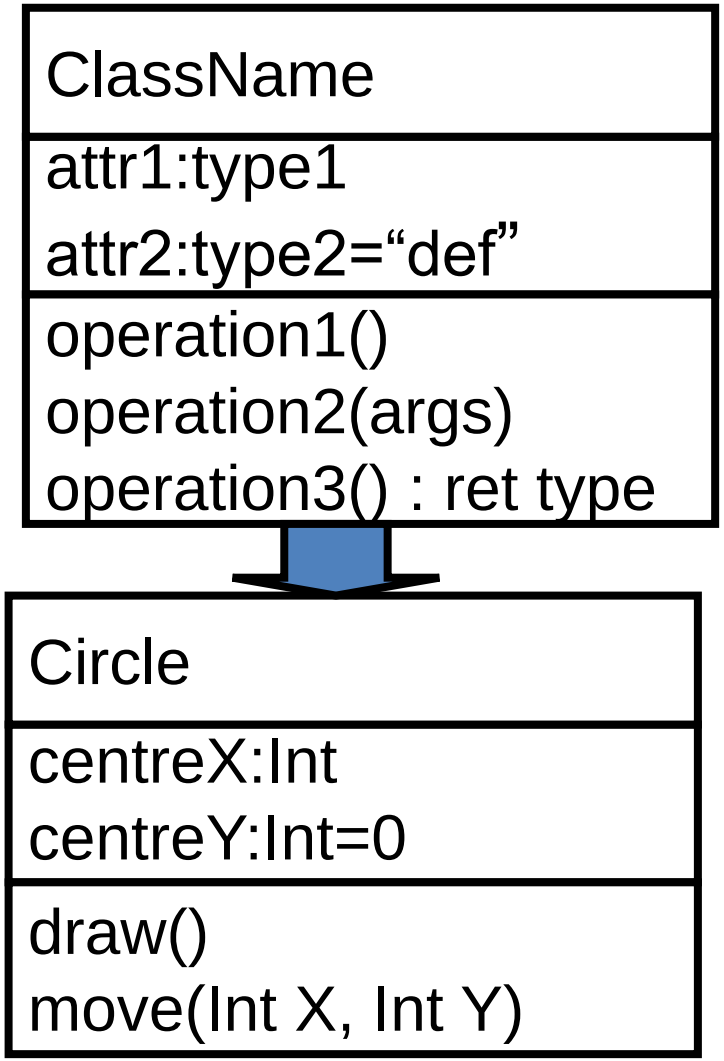
Class with no members

Class - The rectangle is the icon for the class. The name of the class is, by convention, a word with an initial uppercase letter. It appears near the top of the rectangle. If your class name has more than one word name, then join the words together and capitalize the first letter of the every word.



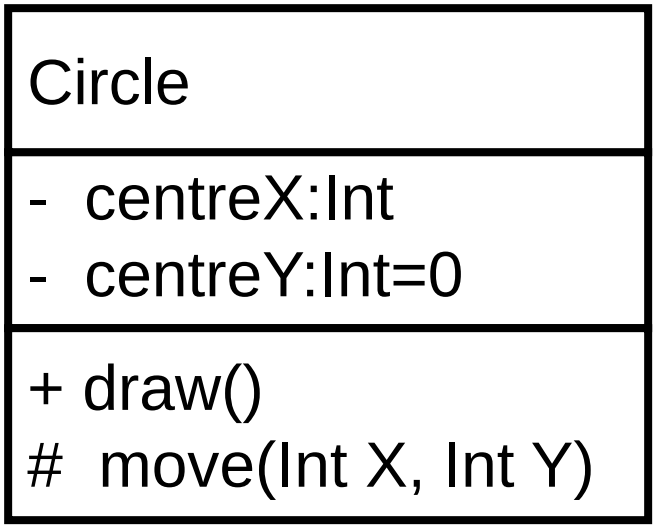
Classes with Attributes and Operations



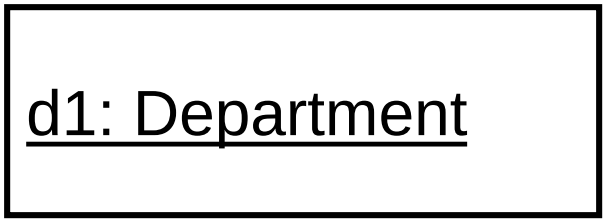


Class Visibility

public level	+
protected level	#
private level	-

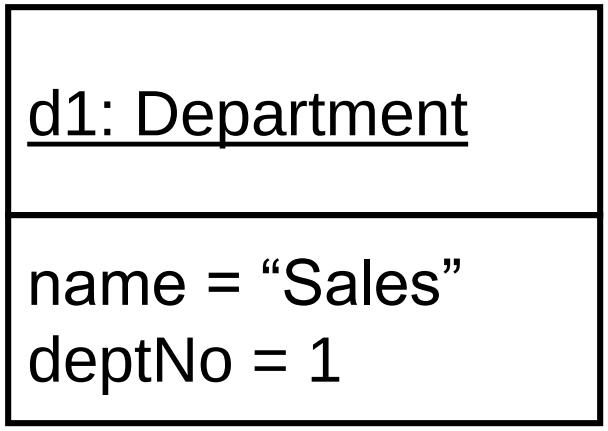


Objects

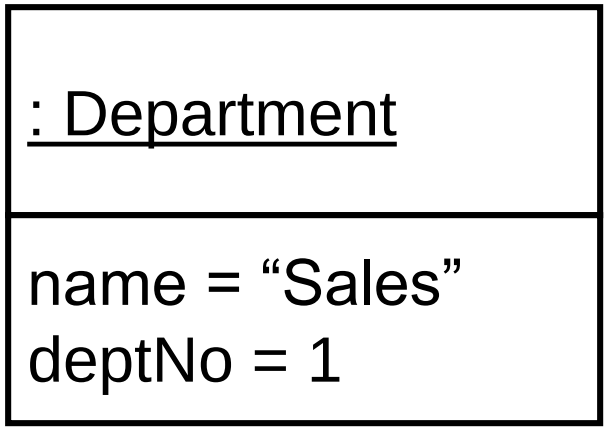


(a)

(b)



(c)



(d)

Multiplicity

- A multiplicity in a class specifies the number of instances (objects) of that class that can exist simultaneously.

- Only one Library object is allowed in the system (referred to as a *singleton* object).

Classes, Objects, and Packages

- A package is expressed by appending the name of package and a double colon before the class name in either a class or an object.

```
PackageName::ClassName
```

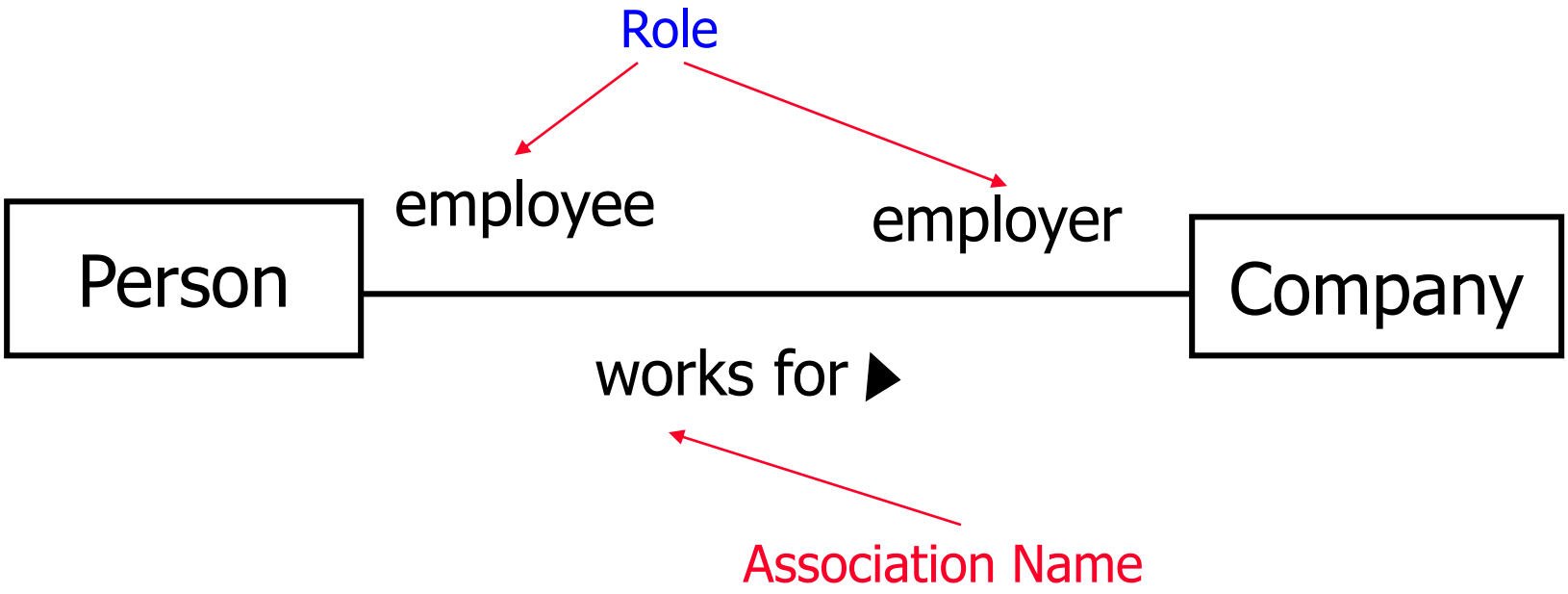
```
ObjectName:Packagename::ClassName
```

Association

- Association is a relation between two separate classes that is established through their objects.
- It is shown by a solid line between classes.
- Association can be one-to-one, one-to-many, many-to-many.
- Composition and aggregation are two forms of association.

Example

A Person works for a Company.

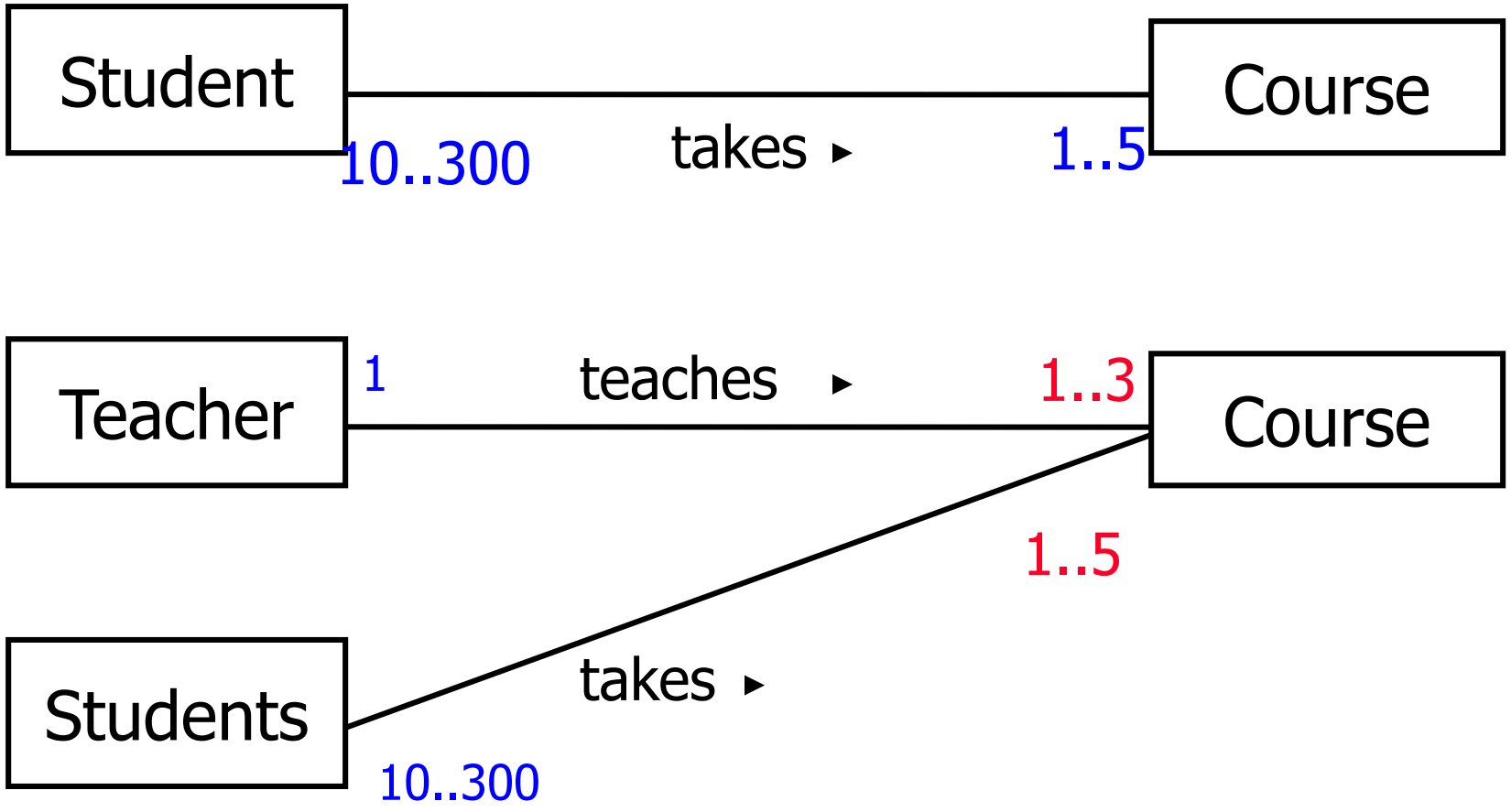


Properties

- Name
  - *Name of the association*
- Role
  - *The specific role of the association*
- Multiplicity
  - *Indicates the number of objects that are connected*
- Type
  - *Plain association, aggregation, composition*
- Direction
  - *Direction can also be shown for a association*

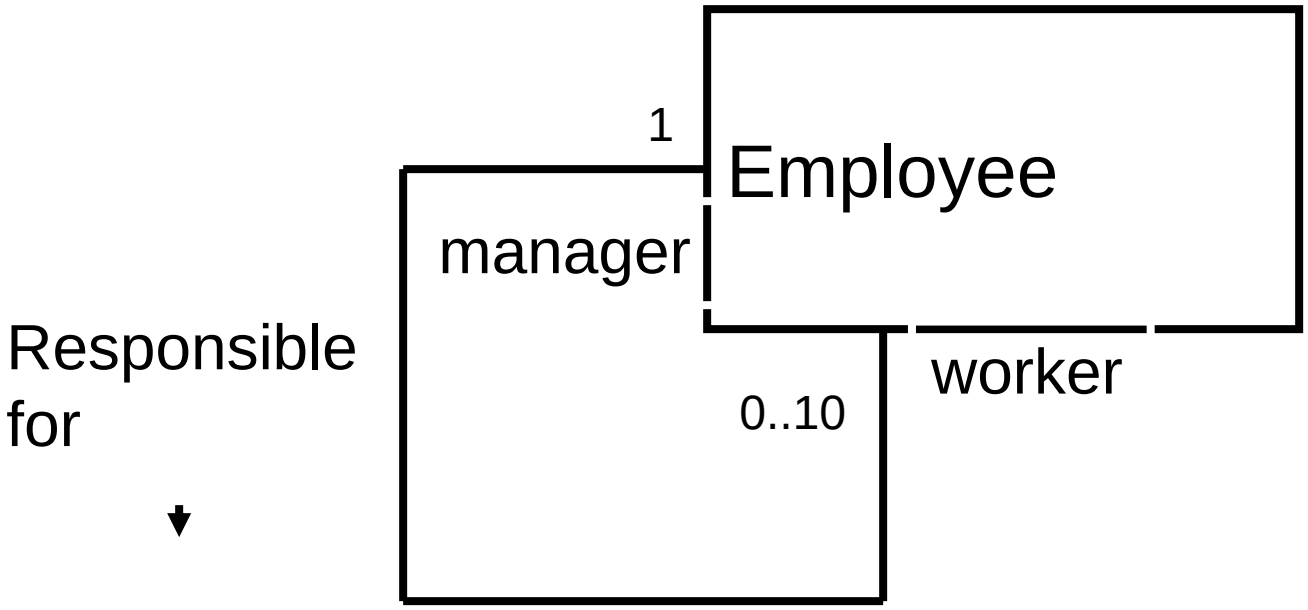
Association - Multiplicity

- It specifies the number of links that can exist between instances of the associated classes.

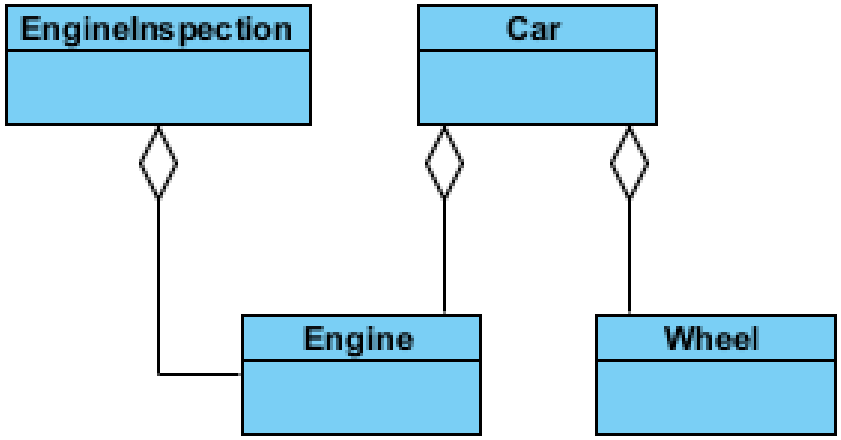


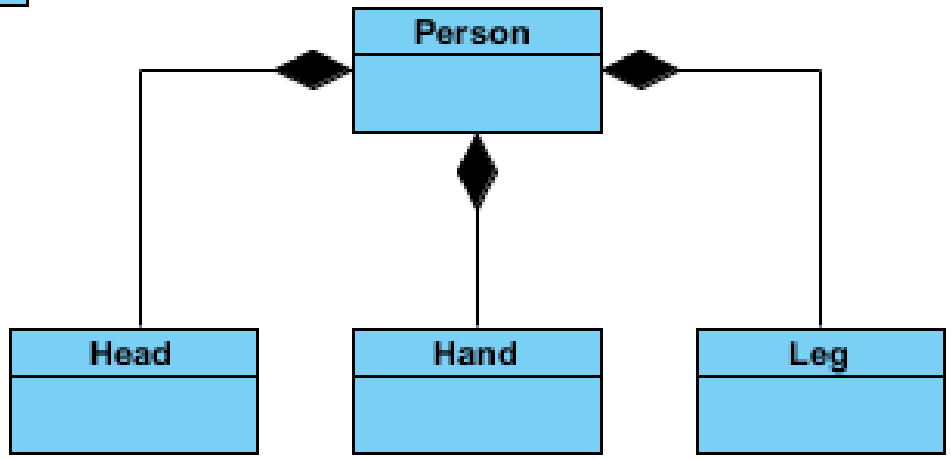
Self Association

- A Company has Employees.
- A single manager is responsible for up to 10 workers.



Association Types





Association - Types

innovate achieve lead

Aggregation

- It represents a **has-a** relationship.
- Both entities can survive individually which means ending one entity will not effect the other.
- Weak Association
- Eg. Bank and Employee

Composition

- It represents a **part-of** relationship.
- The entities are dependent on each other and one cannot exist without the other.
- Strong Association
- Eg. Library and books