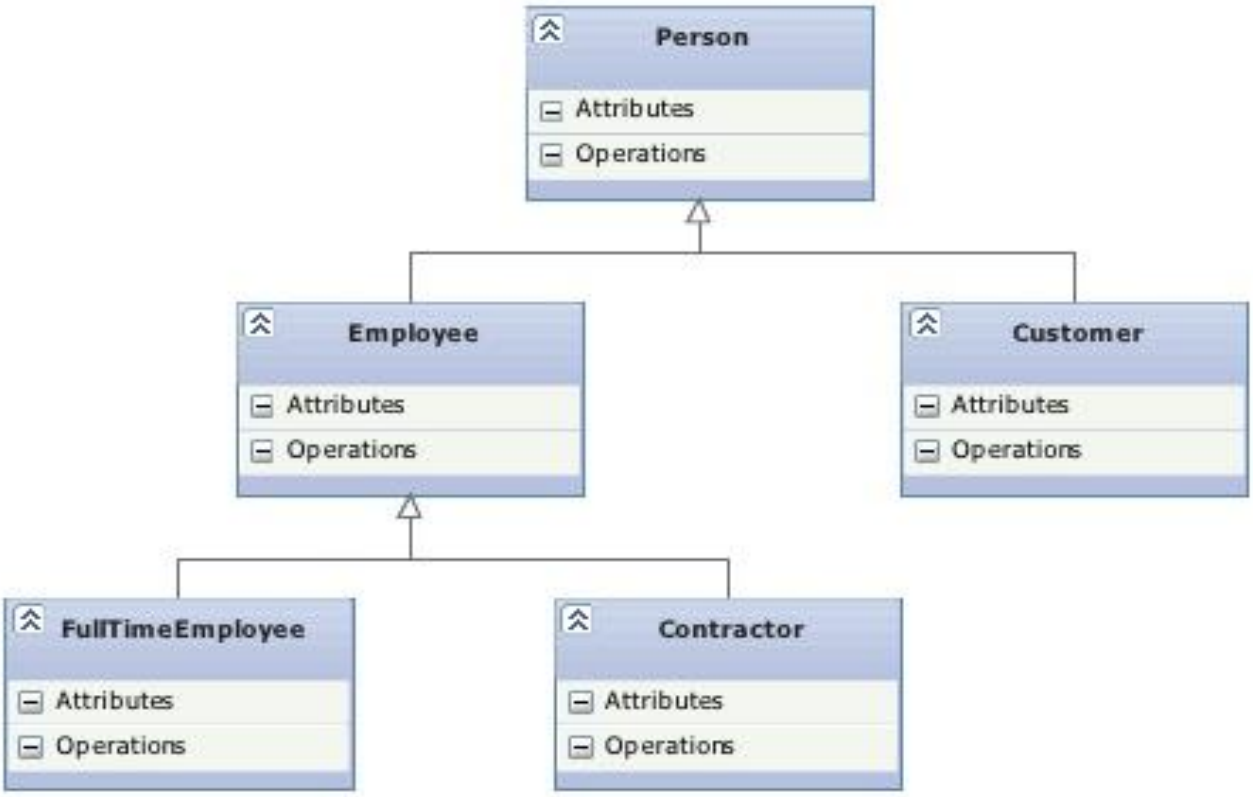
**LEC3 OOP 2018-2019**

# Types of Relationships In (OOP)

One of the advantages of Object-Oriented programming language **is code reuse**. This reusability is possible due to the relationship b/w the classes. Object oriented programming generally support 4 types of relationships that are: inheritance, association, composition and aggregation. All these relationships are based on:
"is a" relationship, "has-a" relationship and "part-of" relationship.

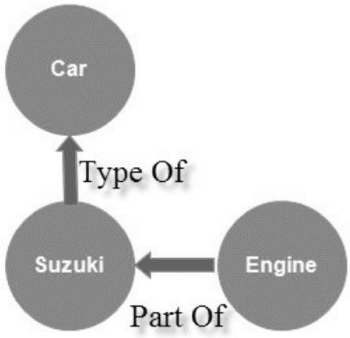
1-**Inheritance:**

Inheritance is “IS-A” type of relationship. It means that it creates a new class by using existing class code. It is just like saying that “A is type of B”. For example:
**Apple is a type of fruit**, so **Apple is a fruit**. **Ferrari is a type of car,** so **Ferrari is a car**.



## 2- Composition:

Composition is a "part-of" relationship. Simply composition means mean use of instance variables that are references to other objects. In composition relationship both entities are interdependent of each other for example “engine (المحرك) is part of car”, “heart is part of body”. Let us take an example of car and engine. Engine is a part of each car and both are dependent on each other.



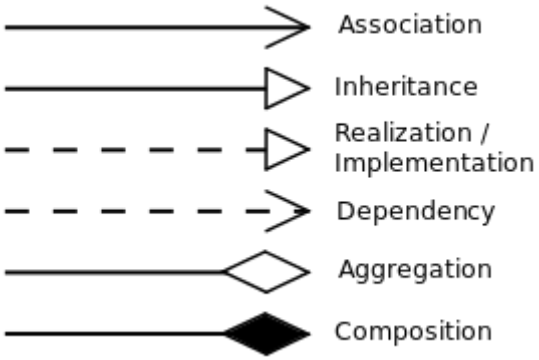

## 3- Association (الاقتران)

Association is a **“has-a”** type relationship. Association establish the relationship between two classes using through their objects. Association relationship can be one to one, One to many, many to one and many to many. Association is the OOPS concept to define the relationship between objects. Association defines the multiplicity between objects. For example Teacher and Student objects. There is one to many relationship between a teacher and students. Similarly a student can have one to many relationship with teacher objects. However both student and teacher objects are independent of each other.

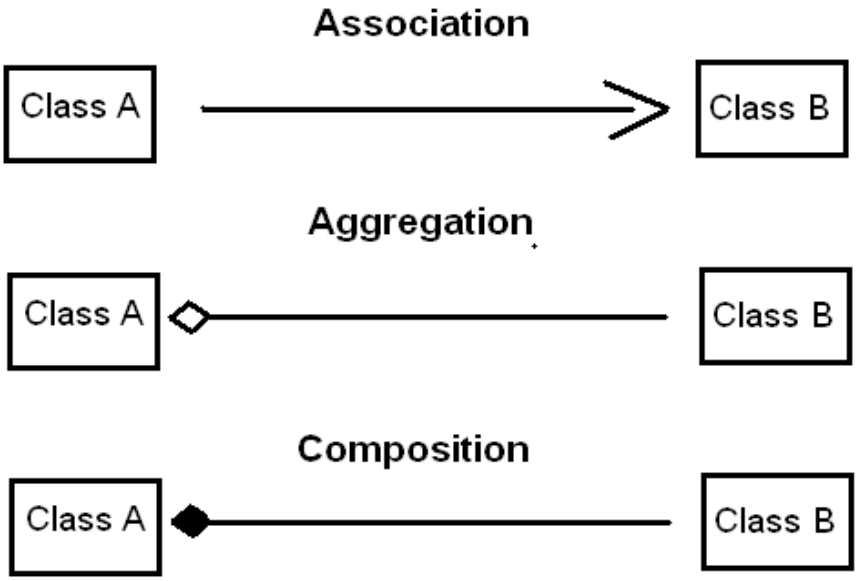
## 4- Aggregation (التجميع)

Aggregation is based is on "has-a" relationship. Aggregation is a special form of association. In association there is not any classes (entity) work as owner but in aggregation one entity work as owner. In aggregation, objects have their own life cycle but there is an ownership. Whenever we have “HAS-A” relationship between objects and ownership then it’s a case of aggregation. Composition is a special case of aggregation. Composition is a more restrictive form of aggregation. When the contained object in “HAS-A” relationship can’t exist on it’s own, then it’s a case of composition. For example, House has-a Room. Here room can’t exist without house.

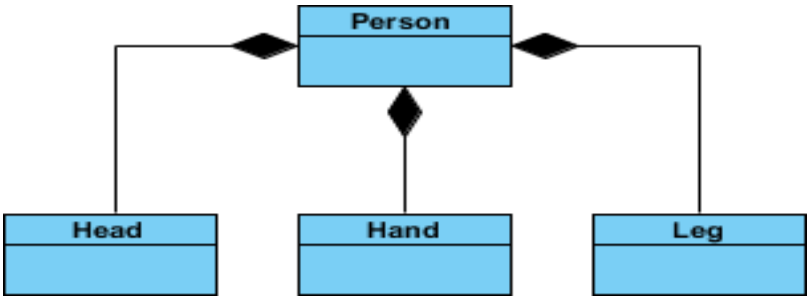
The following notations are used to express the relationships graphically



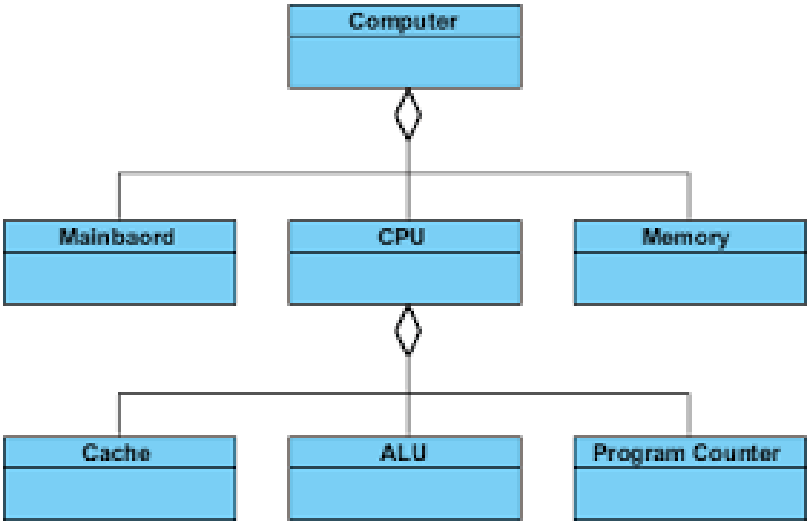

**Ex1:**



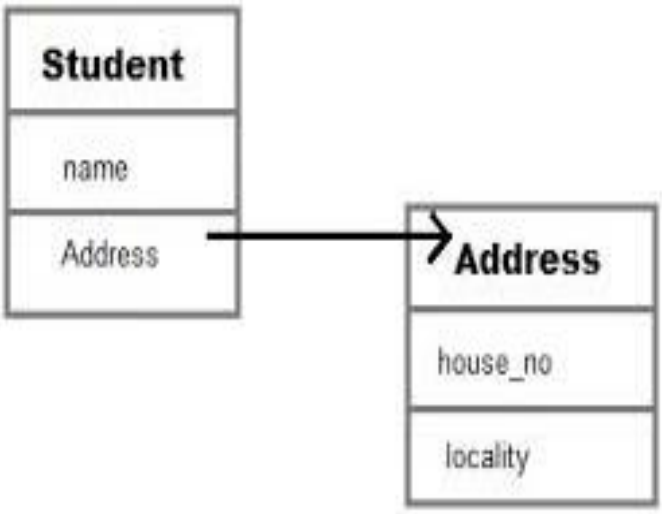

**Ex2:**



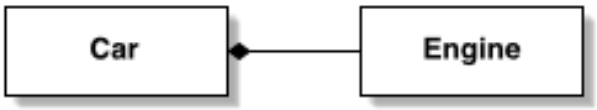
## Ex3:



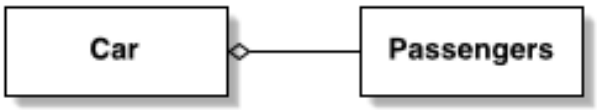

## Ex4:




## Ex5:

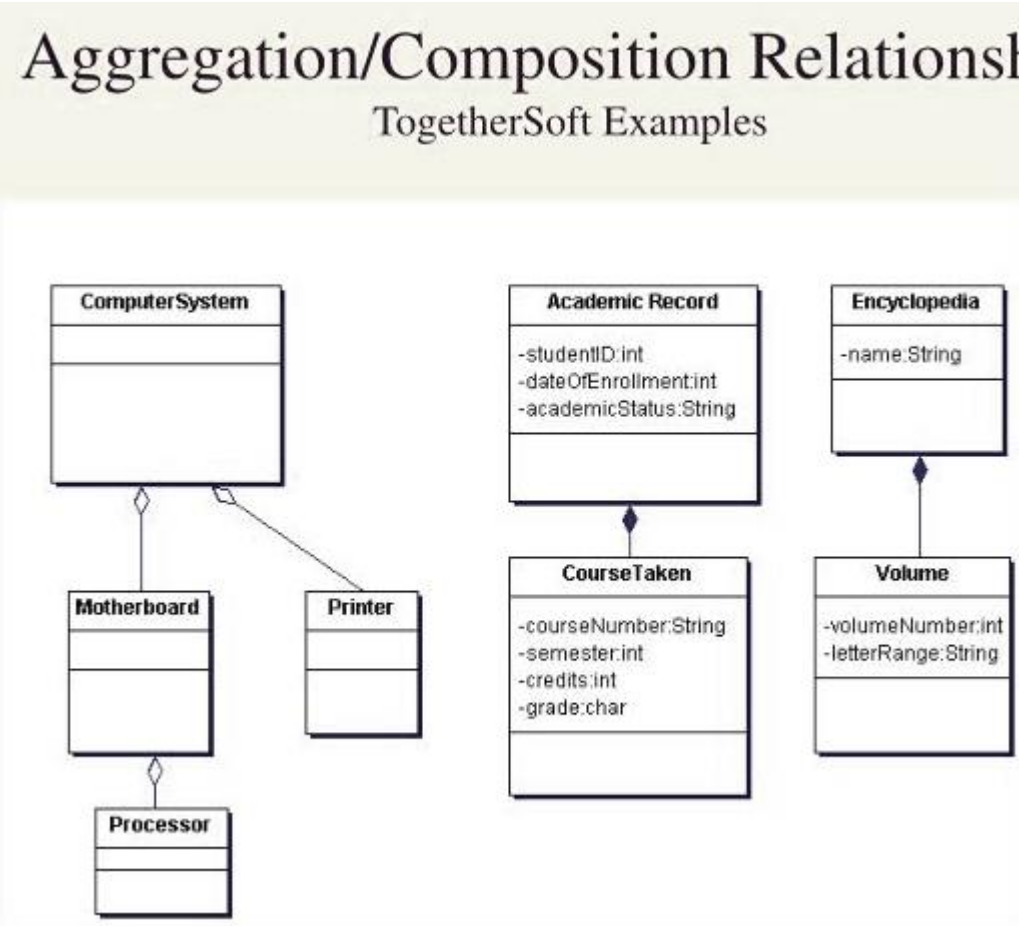



Composition: every car has an engine.

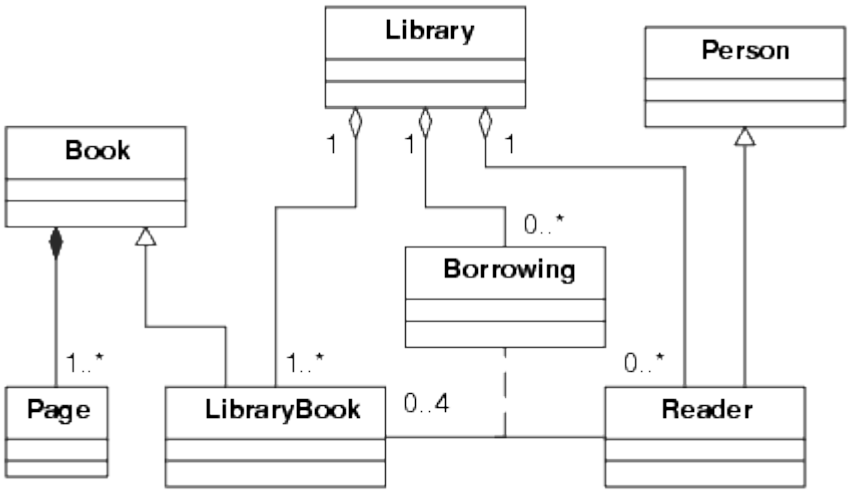



Aggregation: cars may have passengers, they come and go

**Ex6:**


Aggregation/Composition Relationsl
TogetherSoft Examples
ComputerSystem
Academic Record
-studentID:int
-dateOfEnrollment:int
-academicStatus:String
Encyclopedia
-name:String
Motherboard
Printer
CourseTaken
-courseNumber:String
-semester:int
-credits:int
-grade:char
Volume
-volumeNumber:int
-letterRange:String
Processor

**Ex7:**


Library
Person
Book
1
1
1
0..*
Borrowing
1..*
1..*
0..*
Page
LibraryBook
0..4
Reader