EXPERIMENT NO-10

Title: Write a program to implement Aggregation and Composition.

Objectives:

1. study of Association, Aggregation and Composition.
2. implementation of Aggregation and Composition.

Theory:

An object is a basic unit of Object-Oriented Programming and represents real-life entities. Complex objects are objects that are built from smaller or a collection of objects. For example, a mobile phone is made up of various objects like a camera, battery, screen, sensors, etc. This process of building complex objects from simpler ones is called object composition.

In object-oriented programming languages, object composition is used for objects that have a “has-a” relationship with each other. Therefore, the complex object is called the whole or a parent object whereas a simpler object is often referred to as a child object.

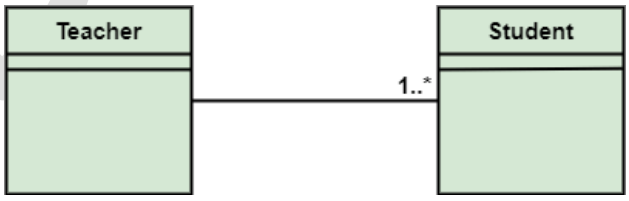
Association:

In object-oriented general software design, the relationship between one object's functionality and another’s is known as an association. Association in C++ is a relationship between two classes where one class uses the functionalities provided by the other class. In other words, an association represents the connection or link between two classes. In an association, one class instance is connected to one or more instances of another class.

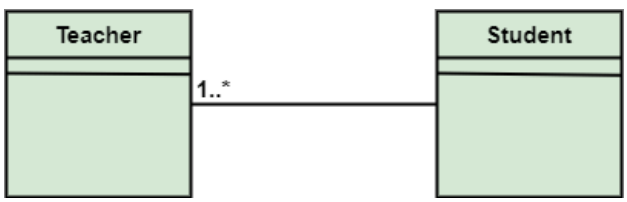
The relationship between doctors and patients is a great example of an association. The doctor clearly has a relationship with his patients, but conceptually it’s not a part/whole (object composition) relationship. A doctor can see many patients in a day, and a patient can see many doctors (perhaps they want a second opinion, or they are visiting different types of doctors). Neither of the object’s lifespans are tied to the other.

Representation of Association using the UML Notation:

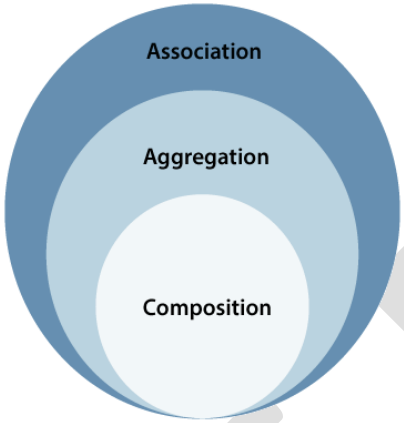
1) A single teacher has multiple students.



2) A single student can associate with many teachers.

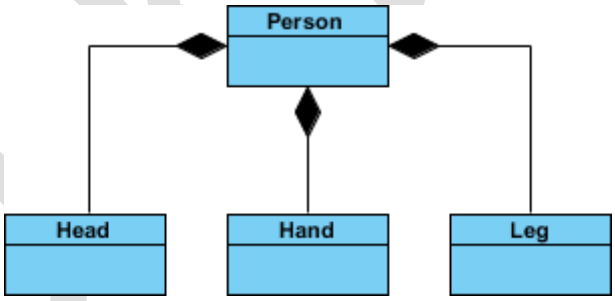


The composition and aggregation are two subsets of association. In both of the cases, the object of one class is owned by the object of another class; the only difference is that in composition, the child does not exist independently of its parent, whereas in aggregation, the child is doe exists independently i.e., standalone. An aggregation is a special form of association, and composition is the special form of aggregation.



Composition:

Composition relationship is also called a part-whole relationship in which the part component can only be a part of a single object simultaneously. In composition relationships, the part component will be created when the object is created, and the part will be destroyed when the object is destroyed. A person's body and heart is a good example of a part-whole relationship where if a heart is part of a person's body, then it cannot be a part of someone else's body at one time.



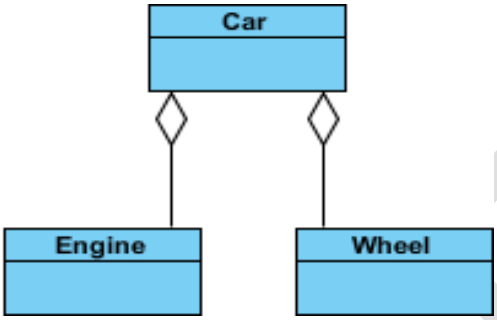
To qualify as a composition, the object and a part must have the following relationship-

- The part (member) is part of the object (class).
- The part (member) can only belong to one object (class).
- The part (member) has its existence managed by the object (class).

Aggregation:

Unlike composition, in the aggregation process, the part component can simultaneously belong to more than one object. It is also a part-whole relationship. The object(class) will not be responsible for the existence or presence of the parts. To be qualified as aggregation, the part and object must follow the relationship described below:

- The part component does not show its presence with the help of an object.
- The part (member) can belong to more than one object (class) at a time.
- The part (member) does not have its existence managed by the object (class).
- In aggregation, the lifetime of the owned object does not depend on the lifetime of the owner.



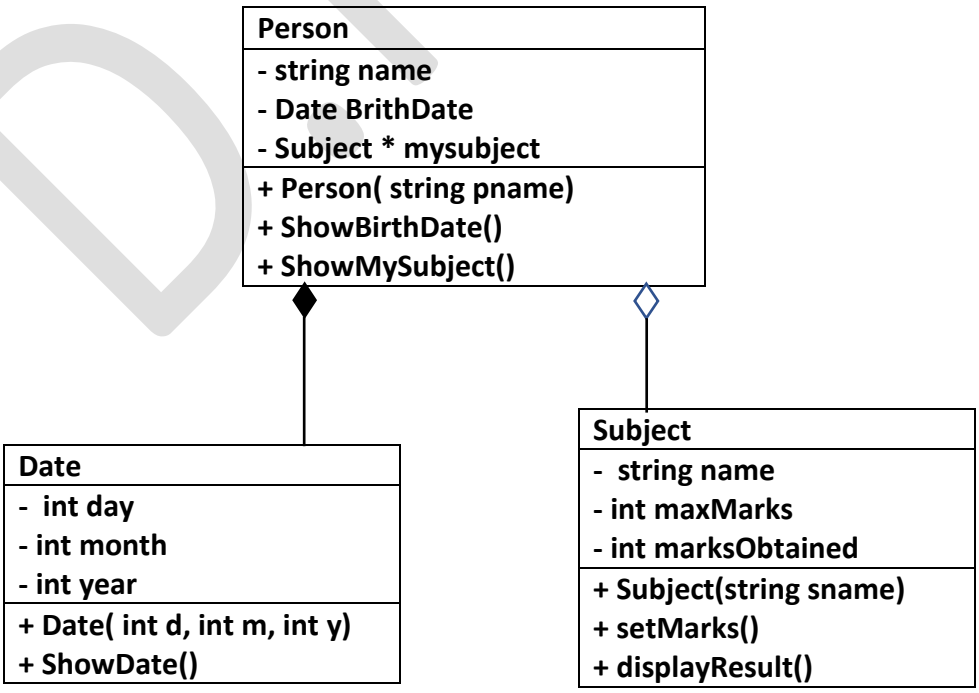
Benefits of Object Composition:

Using object composition can provide the following benefits:

- Reuse existing codes: Object composition allows to reuse of the existing code without a need to model an is-a relationship as usually done in inheritance.
- Composition also allows making code easier to change and adapt if necessary. The internal classes can be changed without any side effects and changes can be handled internally.

Procedure:

Write program to implement aggregation and composition for classes as shown in figure.



Keywords: Object Composition, Association, aggregation.