Принципы отношений между классами (объектами) ООП

Существует всего три основных принципа отношений между классами ООП:

1. Ассоциация [“Has a” - обобщение].
2. Агрегация и композиция [“Has a”];
3. Обобщение/Расширение (наследование) [“Is a”].

Ассоциация

Ассоциация означает, что объекты двух различных классов могут ссылаться один на другой. То есть иметь некоторую связь между друг другом. К примеру, object Manager может “выписать” object TheCheck (счёт). То бишь ассоциация указывает на то, что между объектами есть какая-то связь.

Агрегация и композиция

На самом деле агрегация и композиция являются частными случаями ассоциации. Эти определения конкретизируют тип связи между объектами.

Агрегация – отношения, когда один объект является частью другого. Пример: студент входит в группу любителей группы (студент может входить в множество различных групп). То есть, объект создаётся вне объекта, которому он принадлежит.

Композиция – более жесткие отношения: объект не только является частью другого, но и вообще не может принадлежать ещё кому-то. К примеру, двигатель может быть и без машины, но он вряд ли сможет быть в двух или трех машинах одновременно.

http://java-course.ru/begin/relations/ - источник.

Реализация в Java:

Композиция

```
///Composition
class Engine{
    private int time_of_working;

    /** Default. */
    public Engine(){time_of_working = 60;}

    public void setTime_of_working(int time_of_working) {
        this.time_of_working = time_of_working;
    }
    public int getTime_of_working(){
        return time_of_working;
    }
}

class Car{
    Engine engine;

    public Car(){
        Engine engine = new Engine();
        engine.setTime_of_working(65);
    }
}
```

Как можно заметить, объект engine класса Engine создаётся в классе Car и полностью контролируется именно им.

Агрегация

```java
///Aggregation
class Student{
    String name;
    int age;

    /** Default. */
    public Student(){
        name = "Ivan Ivanov";
        age = 25;
    }

    public Student(String name, int age){
        this.name = name;
        this.age = age;
    }
}

class Group{

    Student[] student;

    public Group(Student[] student){
        this.student = new Student[student.length];
        for (int i = 0; i < student.length; i++)
            this.student[i] = student[i];
    }
}

public class Main {

    public static void main(String[] args) {
        Student[] students = new Student[10];
        for (int i = 0; i < 10; i++)
            students[i] = new Student();
    }
}
```

Здесь же объект класса (массив объектов класса [в данном случае не важно объект/контейнер объектов]) Student создаётся в классе Main и передаётся в класс Group. Таким образом, данным объектом никак не управляет класс Group, он лишь использует его.