Agregación

- Una clase es parte de otra clase (Composición débil).
- La destrucción del compuesto no conlleva la destrucción de los componentes

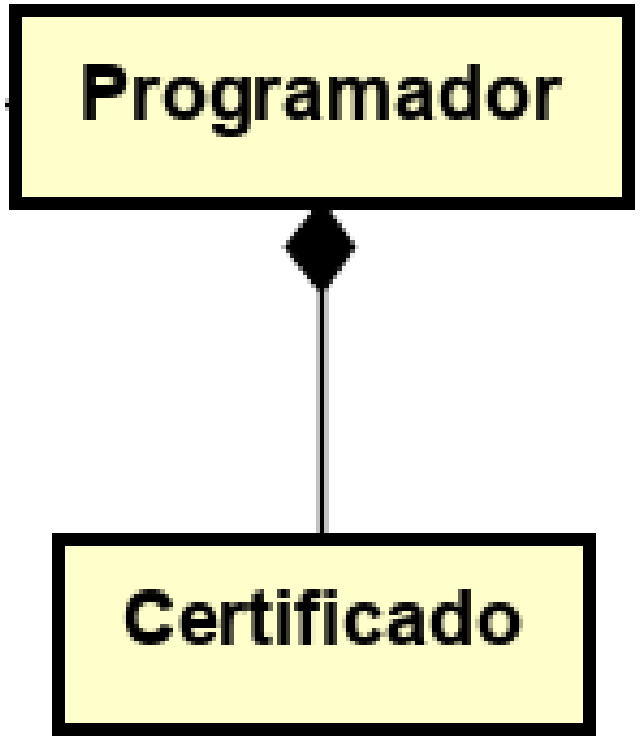
Agregación

```
public class Team
{
    private Developer developer;
    /**
     * Constructor for objects of class Team
     */

    public Team(Developer developer){
        this.developer = developer;
    }
```

```
public class Team
{
    private ArrayList<Developer> developers;

    /**
     * Constructor for objects of class Team
     */
    public Team()
    {
        developers = new ArrayList<Developer>();
    }

    /**
     * Add developer
     * @param developer -  developer to add
     */
    public void addDeveloper(Developer developer)
    {
        developers.add(developer);
    }
}
```

Composición

- La existencia de una clase contenida depende de la existencia de la clase contenedora (Composición fuerte).
- La destrucción del compuesto conlleva la destrucción de los componentes



Composición

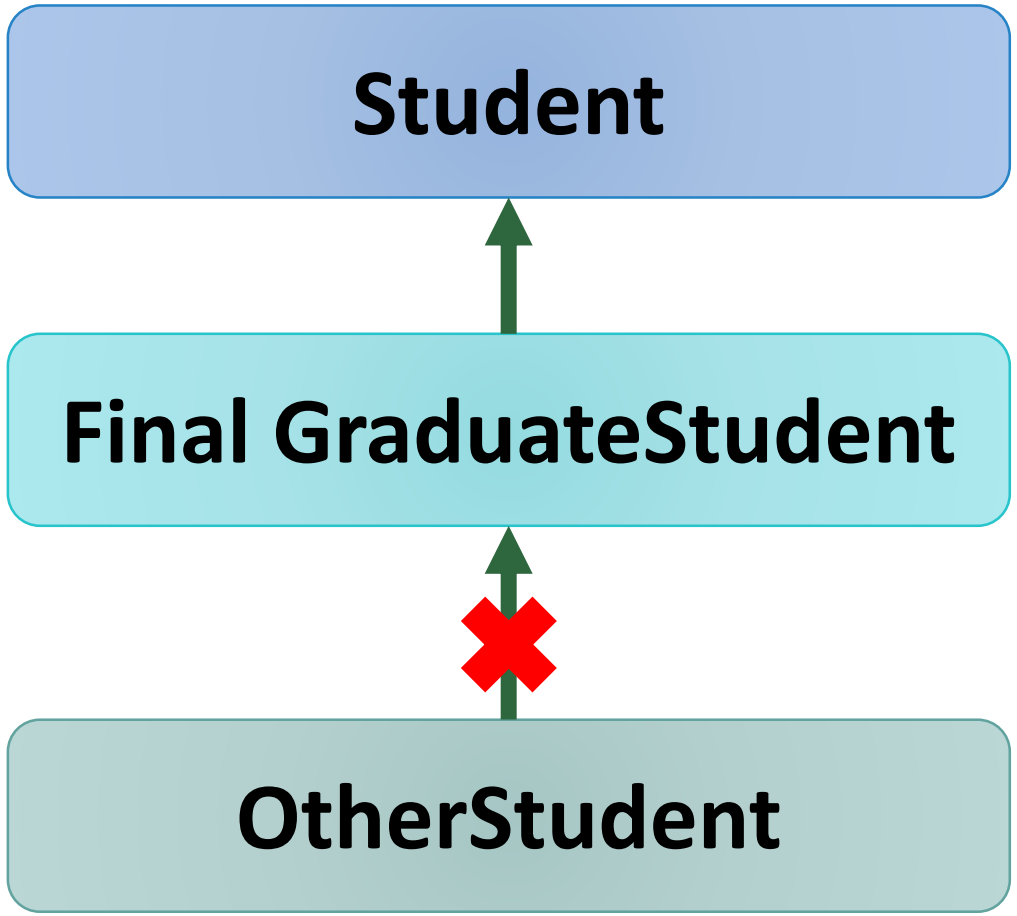
```
public class Developer
{
    private Certificate certificate;

    /**
     * Constructor for objects of class Developer
     */
    public Developer()
    {
        certificate = new Certificate(); // SE CREA UNA INSTANCIA
    }
}
```

Modificadores de acceso

Modificador	Clase	Subclase	Todos
Private	SI	NO	NO
Protected	SI	SI	NO
Public	SI	SI	SI

Niveles de visualización.

Clase final



- Una clase final **no** puede ser heredada
- Una clase final **->** Hoja de un árbol
- Máximo nivel de especialización