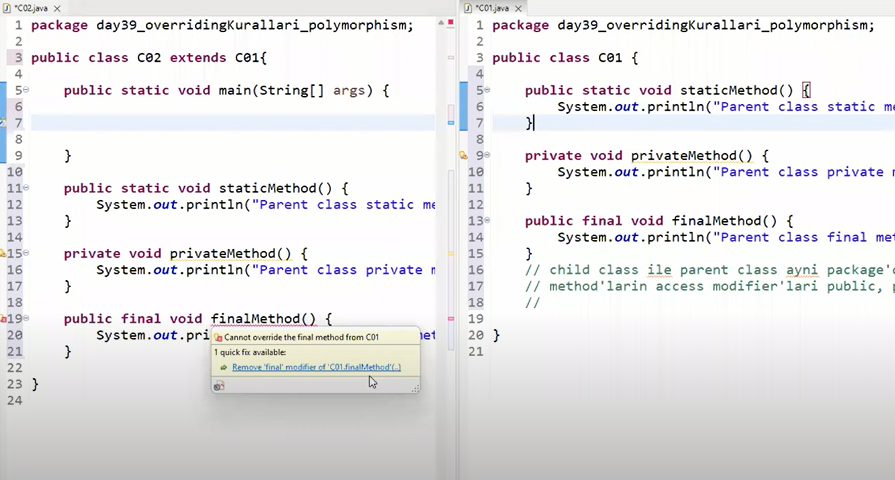
**DAY 39 OVERRİDİNG KURALLARI VE POLYMORPHİSM**

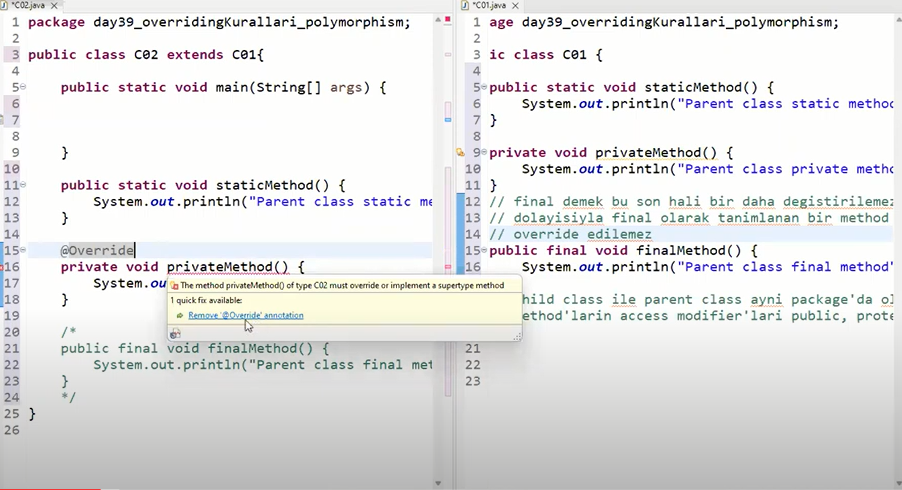
1. Method Signature’i **(isim ve parametreler)** ayni olmalidir.
2. Child class’daki method’un (overriding method) Access Modifier’i parent class’daki method’un (overridden) modifier’indan daha dar olamaz.(Baba çocuğu kısıtlayabilir fakat çocuk babayı kısıtlayamaz.)
3. Overriding method **covariant** return type kullanmalidir.
4. private, static and final method’lar overriding yapilamazlar

not: child ve parent class lar aynı package de olduklarında methodlarımız public, protected veya default olması durumunda herhangi bir sorunla karşılaşmayız.

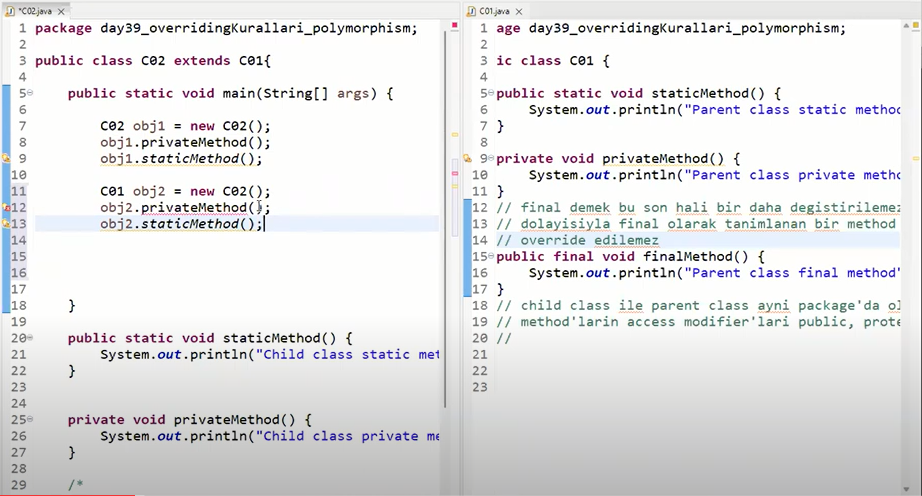


Üç method yapmış olmamıza rağmen java sadece finalMethod un altını çizdi. Çünkü final’ın anlamı bu methodun son hali artık herhangi bir değişiklik yapılamaz. Bu nedenle parent class ta son halini almış bir methodu child class ta override yapamayız. Bu sebebden dolayı cte verdi. finalMethodun üzerine geldiğimizde de hatayı düzeltmek için finalMethodunu parent classtan kaldırmamızı söyler.

Static ve private için herhangi bir hata vermedi.



Ancak private olarak tanımlama yaptığımızda private özelliğinden dolayı privateMethod cte verdi. Çünkü private methodlar sadece bulunduğu class ta çalışırlar. Hatayı düzeltmek içinde bizden override ı kaldırmamızı ister.

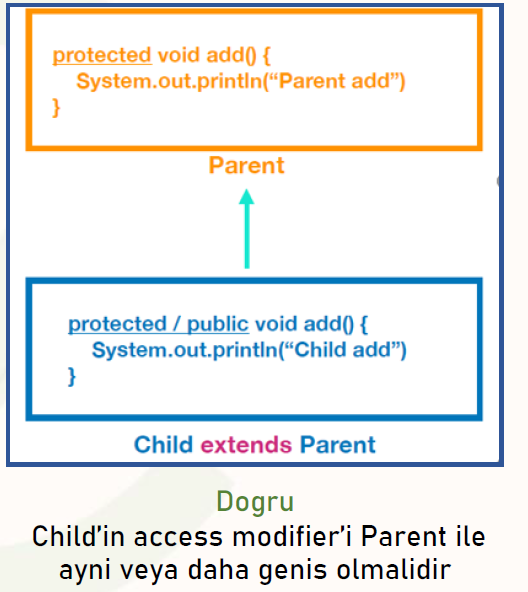
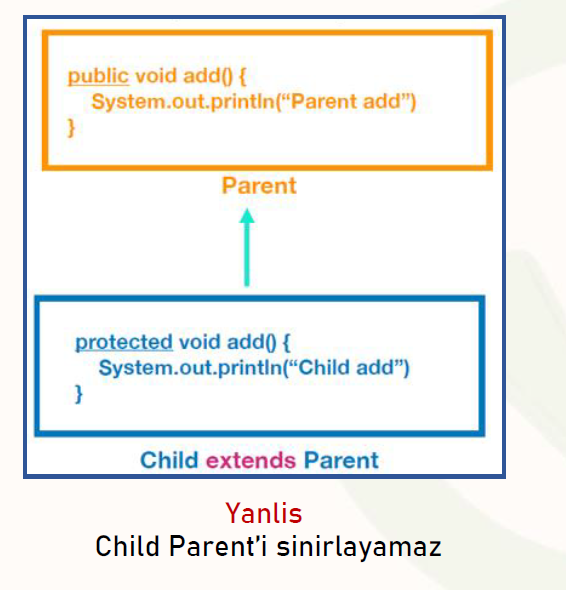


Obj2 nesnesinin data türü parent class dan olduğu için **obj2.privateMethod()** methodu call yapıldığında parent a gider. Ancak method private olduğu için ulaşamaz ve cte verir. Bu method a ulaşamadığımız için override da yapamayız.

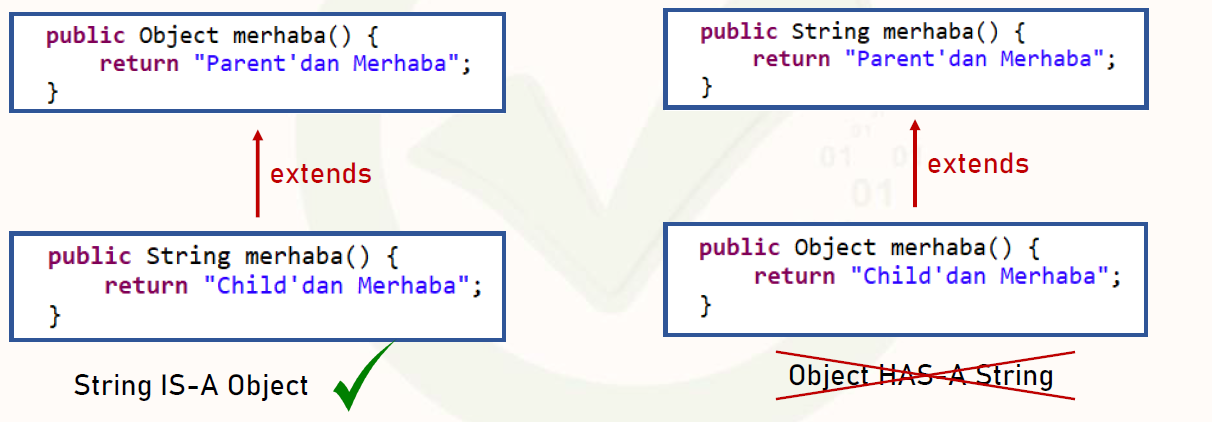
Bu iki private methodun signature ları aynıı olmasına rağmen bu methodlar farklı iki method olarak çalışır.

Aynı durum static method larda da geçerlidir. Yani signature aynı olmasına rağmen bu method lar farklı iki method olarak çalışır.

Access modifier lar açısından da parent class lar child class ları sınırlayabilir. Child class parent class ı sınırlayamaz



Eğer parent class ve child class taki methodların return type ları aynı değil ise aralarında İS-A RELATİON olmalıdır. Yani parent daki return type child daki return type tan daha kapsayıcı olmalıdır. Primitive data türlerinde böyle bir ilişki yoktur. Eğer return type mız primitive ise türler aynı olmak zorundadır.



Not: object en tepedeki parent class olduğu için child class ta hangi return type istersek kullanabiliriz.

**POLYMORPHİSM**

Aynı isimdeki bir methodu farklı biçimlerde kullanmaktır.

**Polymorphism = Overloading + Overriding**

Isim aynı parametreler faklı

Static tir. Compile time

Signature aynı parent ve child class larda oluşturulup body değiştirilir.

Dinamiktir. Run time

Bir nesnenin birden fazla nesne gibi davranmasıdır.

**Overloading ile Overriding karşılaştırılması**

1. Overloading’de method signature degisir, Overriding’de degismez.
2. Overloading’de body istenirse degistirilebilir, Overriding’de body %99 degistirilir.
3. final, static ve private methodlar Overload edilebilir, ama Override edilemezler.
4. Overloading Compile Time Polymorphism (static)’dir, Overriding is Run Time Polymorphism’(dynamic)dir.
5. Overloading’de inheritance gerekmez, Overriding’de gerekir.

6) Overloading’de istedigimiz sekilde access modifier ve return type kullanabiliriz ama Overriding’de access modifier ve return type kullanma belli kurallara baglidir.