

1- (Inheritance and Software Reuse) Discuss the ways in which inheritance promotes software reuse, saves time during app development and helps prevent errors.

Kalıtım sayesinde yazılımcılar; daha **az kod** yazar, **daha hızlı** sonuç alır ve bir kez yazdıkları **güvenilir kodları** her projede tekrar tekrar kullanabilirler.

2- (Chain of Constructor Calls) When the object of a derived class is instantiated, discuss the chain of constructor calls that begin in the program. Discuss which constructor will be executed at the end. Can constructors be inherited?

1. Yapıcı Metot Çağrı Zinciri (Chain of Constructor Calls)

C# veya Java gibi dillerde türetilmiş bir sınıftan nesne oluşturduğunuzda, arka planda bir zincirleme reaksiyon başlar. Bu süreci şu adımlarla özetleyebiliriz:

- **Yukarı Doğru Çağrı:** Türetilmiş sınıftan (alt sınıf) bir nesne oluşturulduğunda, alt sınıfın yapıcı metodu çalışmadan önce **otomatik olarak** üst sınıfın (base/parent class) yapıcı metodunu çağırır. Bu işlem hiyerarşinin en tepesindeki Object sınıfına kadar gider.
- **İşlem Sırası:** Çağrılar "aşağıdan yukarıya" doğru gider, ancak kodun asıl yürütülmesi (execution) "**yukarıdan aşağıya**" doğru gerçekleşir.

2. En Son Hangi Yapıcı Metot Yürütülür?

Zincir en tepedeki sınıfa kadar ulaştıktan sonra, yapıcı metotların içindeki kodlar çalışmaya başlar:

1. Önce en tepedeki (temel) sınıfın yapıcı metodu tamamlanır.
2. Ardından bir altındaki sınıfın kodu çalışır.
3. **En son**, nesnesi oluşturulmak istenen **türetilmiş sınıfın (derived class)** kendi yapıcı metodu yürütülür ve tamamlanır.

3. Yapıcı Metotlar Miras Alınabilir mi?

Hayır, yapıcı metotlar miras alınamaz.

- **Nedeni:** Her sınıfın yapıcı metodu o sınıfın kendi nesnesini başlatmak (initialize) ve kendi verilerine ilk değerlerini vermek için tasarlanmıştır. Alt sınıfın adı üst sınıftan farklı olduğu için, üst sınıfın yapıcı metodu alt sınıfta bir "yapıcı metot" olarak işlev göremez.

