1- (Key Feature of OOP) One of the key features of object-oriented programming is polymor phism. What does polymorphism mean? Does this require altering the whole app, if some new class es are added to the app?

## 1. Çok Biçimlilik (Polymorphism) Nedir?

**Polymorphism**, kelime anlamı olarak "çok formluluk" veya "çok biçimlilik" demektir. Programlama dünyasında ise; farklı nesnelerin, aynı mesaja (metot çağrısına) kendilerine özgü farklı şekillerde yanıt verebilme yeteneğidir.

## 2. Uygulamanın Tamamını Değiştirmek Gerekir mi?

**Hayır, gerekmez.** Çok biçimliliğin en büyük avantajı budur.

- **Esneklik:** Eğer kodunuzu bir üst sınıfa (veya arayüze/interface) göre yazdıysanız, sisteme yeni sınıflar eklediğinizde mevcut kodun büyük bir kısmına dokunmanıza gerek kalmaz.
- **Örnek:** Bir ödeme sisteminiz olduğunu düşünün. `OdemeYontemi` isimli bir üst sınıfınız var ve `KrediKarti` ile `Nakit` alt sınıfları mevcut. Yarın bir gün sisteme `KriptoPara` ile ödeme eklemek isterseniz, ana ödeme döngünüzü (kodun geri kalanını) değiştirmeniz gerekmez. Sadece `OdemeYontemi` sınıfından türeyen yeni bir `KriptoPara` sınıfı oluşturmanız yeterlidir.

2- (Polymorphism and device drivers) How is polymorphism effective for implementing device drivers in layered software systems?

## Aygıt Sürücülerinde Çok Biçimliliğin Rolü

Katmanlı bir sistemde (örneğin bir işletim sistemi), en üstte kullanıcı uygulamaları, en altta ise donanımlar bulunur. Çok biçimlilik, bu iki dünya arasında bir **"standart arayüz"** oluşturarak şu avantajları sağlar:

### *1. Standart Arayüz (Soyutlama)*

İşletim sistemi, her yazıcı veya her ekran kartı için farklı bir kod yazmak yerine, genel bir "Arayüz" (Interface) tanımlar. Örneğin, her yazıcı sürücüsü için standart bir `Print()` metodu belirlenir.

- İşletim sistemi sadece `yazici.Print()` komutunu gönderir.
- Bu yazıcının markası HP de olsa, Canon da olsa sistem için fark etmez. Çok biçimlilik sayesinde ilgili sürücü (driver) bu komutu kendi donanımının anlayacağı dile çevirir.

### *2. Donanım Bağımsızlığı*

Çok biçimlilik sayesinde işletim sistemi katmanı, donanımın fiziksel detaylarından kurtulur.

- Bir kullanıcı yeni bir fare (mouse) taktığında, işletim sisteminin çekirdek (kernel) kodlarını yeniden yazmanız gerekmez.
- Yeni donanımın sürücüsü, önceden belirlenmiş olan standart metotları (örneğin `Click()`, `Move()`) kendi içinde uygular (override eder) ve sistemle anında uyum sağlar.

### *3. Kolay Güncellenebilirlik ve Genişletilebilirlik*

Eski sistemlerde her yeni donanım için tüm sistemin güncellenmesi gerekebilirdi. Çok biçimlilik kullanan katmanlı mimarilerde ise:

- **Tak-Çalıştır (Plug-and-Play):** Yeni bir donanım eklendiğinde, sadece o donanıma özel sınıfları içeren bir sürücü dosyası sisteme tanıtılır.