Bölüm 4: Soyut Sınıflar

JAVA ile Nesne Yönelimli Programlama

Soyut Sınıflar (Abstract Classes)

- abstract anahtar kelimesi kullanılarak tanımlanır.
- Soyut sınıflar, soyut veya soyut olmayan metotlar içerebilir.

```
public abstract class Sekil {
  // Soyut olmayan metot
  void tasi(int x, int y) {
    // Metot içeriği
  }
  // Soyut yöntem (alt sınıflarca implemente edilmeli)
  abstract void ciz();
}
```

Soyut Sınıflar

- *An abstract class is not allowed to have any instances.*
- Soyut sınıflardan nesne oluşturulamaz.
- Başka bir sınıf tarafından kalıtım yoluyla türetilebilir.
- Alt sınıf soyut metotları gerçeklemek zorundadır.
- Ortak durumları ve davranışları birleştirir, yeniden kullanım sağlar.
- Alt sınıflar, soyut metotları kendi ihtiyaçlarına göre özelleştirebilir.
- Yapıcı metotlara (constructor) sahip olabilirler.

Soyut Metotlar

- Gerçekleştirim (implementasyon) olmadan tanımlanır.
- Süslü parantez içermeyen ve noktalı virgülle biten bir tanımlamadır.
- Soyut sınıflar genellikle kalıtım hiyerarşisinde yer alır.

```
public abstract void ciz();
```

Soyut Metotlar

```
public abstract class Sekil {
   // Soyut metot
   public abstract void ciz();
}

public class Cember extends Sekil {
   // ciz metodunu implemente etmek zorunlu
   public void ciz() {
      // Çemberi çiz
   }
}
```

Arayüzler (Interfaces)

- Sistemin belirli işlevlere sahip olacağını garanti eden sözleşmedir.
- İki sistem arasındaki entegrasyon noktasını temsil eder.
- Arayüz tanımlamak için interface anahtar kelimesi kullanılır.
- *Methods in interface are public, whether you mark them public or not!*
- *Unless marked static or default, method will be automatically abstract*
- Metotlar public ve abstract olmalıdır.
- public static final değişkenleri (sabitleri) içerebilir.

Arayüzler

```
public interface GeometrikSekil {

   // Soyut yöntem (public abstract)
   void ciz();

   // Sabit (public static final)
   int KENAR_SAYISI = 4;
}
```

Arayüzler

- *interfaces can also have default methods.*
- Arayüz, varsayılan metotlara sahip olabilir.

```
interface Collection {
   void add(Object o);
   void remove(Object o);
   int size();
   default boolean isEmpty() {
      return size() == 0;
   }
}
```

Arayüzler

- *Interfaces can also have static methods.*
- Arayüzler static methodlara sahip olabilir.

```
interface Geometry {
  static double circleArea(double r) { return PI * r * r; }
  static double squareArea(double s) { return s * s; }
  static double perimeter(double s) { return s * 4.0; }
  static double boxVolume(double w, double h, double d) {
    return w * h * d;
  }
}
```

seal, non-seal ve permits

- *Classes and interfaces can be sealed.*
- Sınıf ve arayüzler mühürlenebilir.
- sealed, bir sınıfı veya arayüzü sadece izin verilen sınıflar kullanabilir.
- non-sealed, bir sınıfı veya arayüzü bazı sınıflara yasaklar.
- permits anahtar kelimesi ile hangi alt sınıflara izin verildiği belirtilir.

seal, non-seal ve permits

```
public sealed interface Hayvan permits Kedi, Kopek {

   void hareketEt();
   // ...
}

final class Kedi implements Hayvan {

   public void hareketEt() {
      // ...
   }
}
```

Arayüz Gerçekleme

- Bir sınıf, bir arayüzü implements anahtar kelimesiyle gerçekler.
- Arayüzdeki tüm metotları gerçeklemeli veya soyut olarak tanımlamalıdır.
- *A class is only allowed to extend at most one superclass, but it can implement zero or more interfaces.*
- Bir sınıf birden fazla arayüzü gerçekleyebilir.

```
public class Kare implements GeometrikSekil {
  // ciz metodunu implemente etmek zorunlu
  public void ciz() {
    // Kareyi çiz
  }
}
```

Arayüzden Kalıtım

- Bir arayüz, başka bir arayüzü extends anahtar kelimesi ile genişletebilir.

```
public interface KareSekli extends GeometrikSekil {
  // Ek metodlar veya sabitler eklenebilir
  void ozelFonksiyon();
}
```

Arayüz Soyut Sınıf Karşılaştırma

- **Arayüz**
  - Yalnızca soyut metotlar içerir.
  - Birden çok arayüzden türeyebilir.
  - Çoklu kalıtım özellikleri sağlar.
  - *interfaces can not specify instance fields!*
- **Soyut Sınıf**
  - Hem soyut hem de somut metotları içerebilir.
  - Tek bir soyut sınıftan türetilir.
  - Ortak niteliklere sahip sınıfları gruplamak için kullanılır.

SON