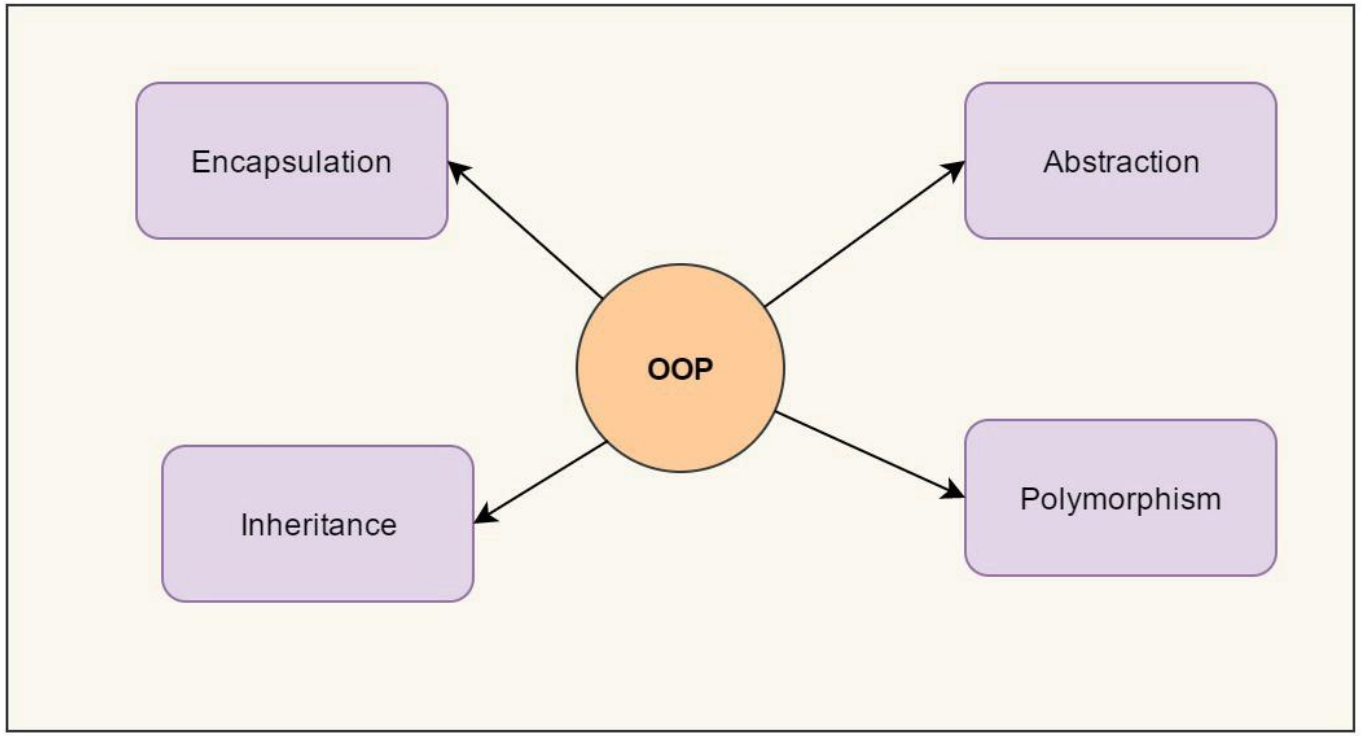
NESNE YÖNELİMLİ PROGRAMLAMA(Object Oriented Programming/OOP)

Öğr. Gör. Celil ÖZTÜRK
Marmara Üniversitesi
Teknik Bilimler Meslek Yüksekokulu

İçerik

- ✓Polymorphism(Çok Biçimlilik)
- ✓Soyut Sınıflar(Abstract Class)
- ✓Soyut Metotlar(Abstract Methods)

Nesne Yönelimli Programlama Temel Bileşenler



Four Pillars of Object Oriented Programming

https://medium.com/@sumeyyeturkmen/4-temel-oop-prensibi-9bd4a36d35a1

Polymorphism(Çok Biçimlilik)

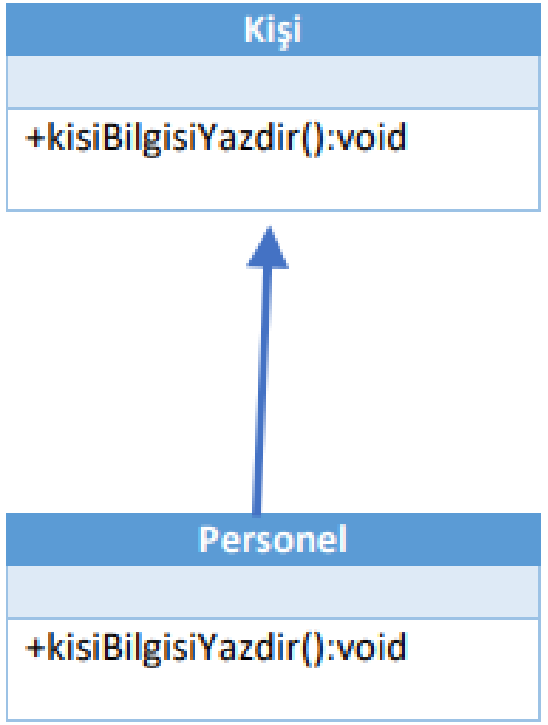
- Çok biçimlilik nesne tabanlı programlamaya biyolojiden aktarılmış bir kavramdır.
- Biyolojide bir türdeki canlılar aynı türden olmalarına rağmen kendilerine has bazı davranışlar sergileyebilirler.
- Java'da bir sınıftan türetilen canlıların farklı şekillerde davranmalarını sağlayabilir.

Polymorphism(Çok Biçimlilik)

- Çok biçimlilik, bir nesnenin davranış şekillerini duruma göre değiştirebilme yeteneği, başka bir tanıma göre ise, nesnelerin içeride farklı çalışmalarına rağmen dışarıdan aynı biçimde görülmelerine verilen addır.

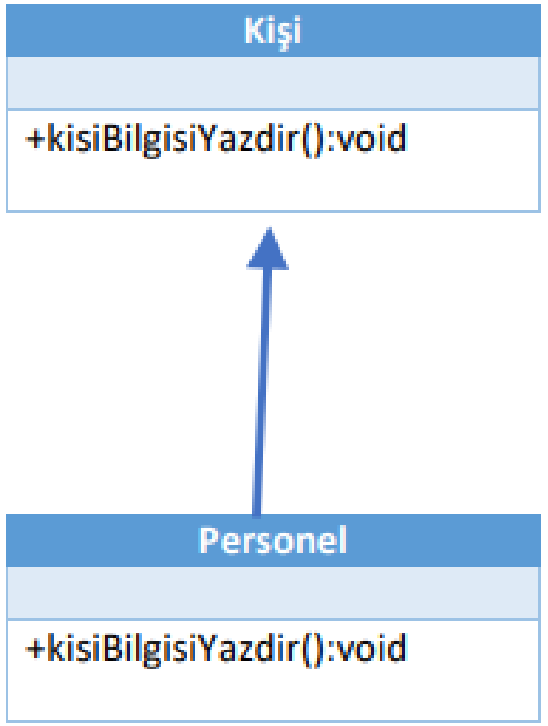
Polymorphism(Çok Biçimlilik)

- Bir nesnenin davranış şekillerinin duruma göre değişebilmesidir.
- Bir çok sınıfın ortak kullanacağı bir metot gerekli ise, her birinin temel alacağı bir ana sınıf içerisinde tanımlanabilir.
- Bir katılım hiyerarşisine sahip sınıflarda aynı metot imzası olan metotlar var ise, hangi sınıfa ait metot çalıştırılacak bunu Java dinamik olarak belirleyebilir.
- Bu özelliğe Polymorphism(çok-biçimlilik) denir.



Polymorphism(Çok Biçimlilik)

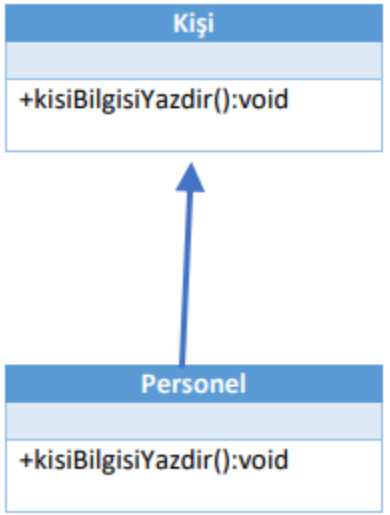
- Çok biçimlilikte, kişiBilgisiYazdir() metodu, Personel sınıfı altında kullanılıyor ise, Kisi sınıfında bulunan kisiBilgisiYazdir() metodunu ezmelidir.(Override).
- Override kavramı, ezme, devre dışı bırakma anlamında kullanılabilir.
- Override kullanılan metot aynı imzaya sahip olan, kalıtımda kullanılan üst sınıftaki metodu devre dışı bırakır.



Polymorphism(Çok Biçimlilik)

```java
public class Kisi {

    public void kisiBilgisiYazdir()
    {
        System.out.println("Kisi sinifi kisi bilgisi yazdir metodu çalıştı.");
    }

}
```

```java
public class Personel extends Kisi {

    @Override
    public void kisiBilgisiYazdir()
    {
        System.out.println("Personel sınıfı kisi bilgisi yazdır metodu çalıştı");
    }
}
```



Polymorphism(Çok Biçimlilik)

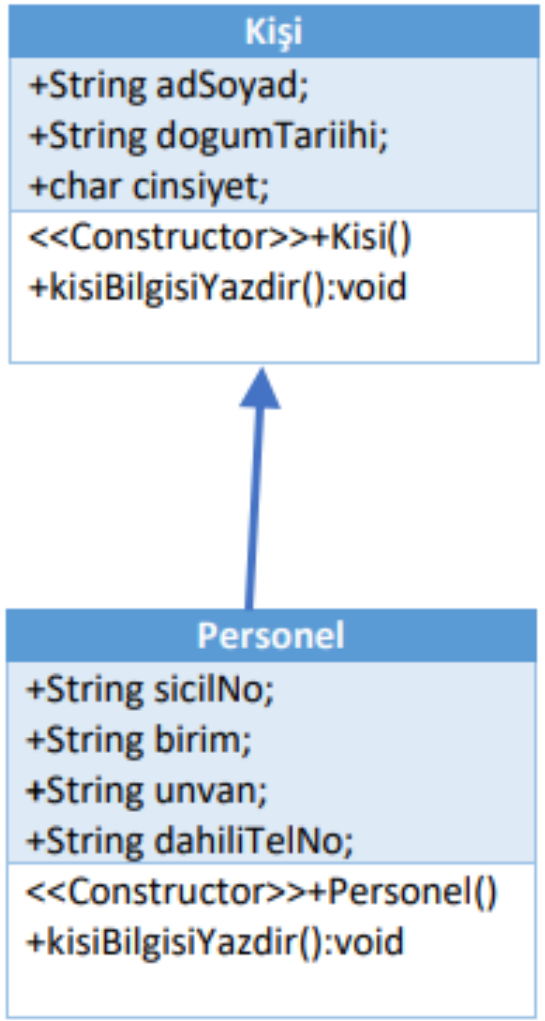
```
public class PolymorphismMU {


    public static void main(String[] args) {
        Personel personel1 = new Personel();
        personel1.kisiBilgisiYazdir();
        Kisi kisi1=new Kisi();
        kisi1.kisiBilgisiYazdir();
    }

}
```

```
run:
Personel sınıfı kisi bilgisi yazdır metodu çalıştı
Kisi sinifi kisi bilgisi yazdir metodu çalıştı.
BUILD SUCCESSFUL (total time: 0 seconds)
```

Polymorphism(Çok Biçimlilik)

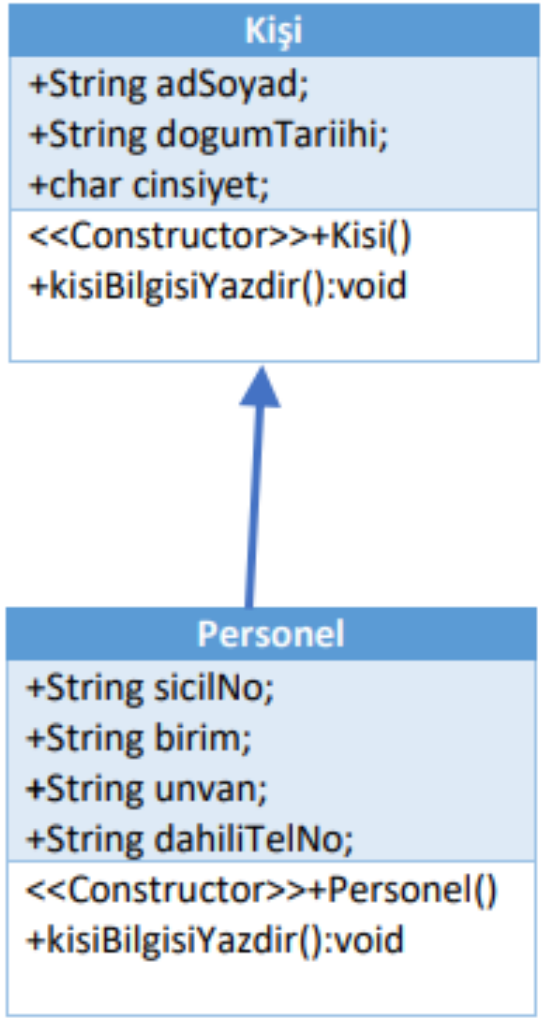
```java
public class Kisi {
    public String adSoyad;
    public String dogumTarihi;
    public char cinsiyet;

    public void kisiBilgisiYazdir()
    {
        System.out.println("Ad Soyad:"+adSoyad
    }

}
```

```java
public class Personel extends Kisi {
    public String sicilNo;
    public String birim;
    public String unvan;
    public String dahiliTelNo;

    public Personel() {

    }
    @Override
    public void kisiBilgisiYazdir()
    {
        System.out.println(unvan+" "+adSoyad)
    }
}
```



Polymorphism(Çok Biçimlilik)

```
public class PolymorphismMU {


    public static void main(String[] args) {

        Kisi kisi1=new Kisi();
        kisi1.adSoyad="Ahmet ÖZ";
        kisi1.kisiBilgisiYazdir();

        Personel personel1 = new Personel();
        personel1.adSoyad ="Ali Akbay";
        personel1.unvan="Doç. Dr.";
        personel1.kisiBilgisiYazdir();
    }

}
```



Polymorphism(Çok Biçimlilik)

```
class Hayvan{
public void Konus(){
System.out.println("Ben bir hayvanım.");
}}
class Inek extends Hayvan{
public void Konus(){
System.out.println("MÖÖ");
}}
class Kedi extends Hayvan{
public void Konus(){
System.out.println("Miyav");
}}
class Kopek extends Hayvan{
public void Konus(){
System.out.println("Hav Hav");
}}
public class Alem {
public static void main(String [] args){
Hayvan[]a=new Hayvan [3];
int indis;
a[0]=new Kedi();
a[1] =new Kopek();
a[2]=new Inek ();
for (indis=0; indis <a.length;indis++){
a[indis].Konus();
}
}}
```

Abstract(Soyut) Sınıflar

- Sonradan genişletilmek üzere yaratılan fakat kendisinden nesne oluşturulmayan sınıflara soyut sınıf (abstract class) denir.
- Bazı programlama modellerinde üst sınıfların direkt olarak kullanılması istenmez.
- Üst sınıflardan türetilmiş sınıflardan oluşturulmuş nesnelerin kullanılması istenir.**
- Soyut sınıflar bir ana sınıf sağlar.
- Soyut sınıflar oluşturulurken **abstract** kelimesi kullanılır.
- Soyut sınıftan nesne yaratılamaz.

Abstract(Soyut) Sınıflar(Oracle Java Documentation)

- An abstract class is a class that is declared abstract—it may or may not include abstract methods.
- Abstract classes cannot be instantiated, but they can be subclassed.
- An *abstract method* is a method that is declared without an implementation (without braces, and followed by a semicolon(;)), like this:

```
abstract void moveTo(double deltaX, double deltaY);
```

Abstract(Soyut) Sınıflar(Oracle Java Documentation)

- If a class includes abstract methods, then the class itself must be declared abstract, as in:

```
public abstract class GraphicObject {
   // declare fields
   // declare nonabstract methods
   abstract void draw();
}
```

Abstract(Soyut) Sınıflar

- Bazı methodlarını implement etmiş, bazılarının imlementation'unun kendisini extend eden class'a bırakmış olan class'a abstract class denir. Bu tip class'lar en çok, iki class'ın ortak methodlarından bazılarının implementation'u da aynı olması durumunda kullanılır.

Abstract(Soyut) Metotlar

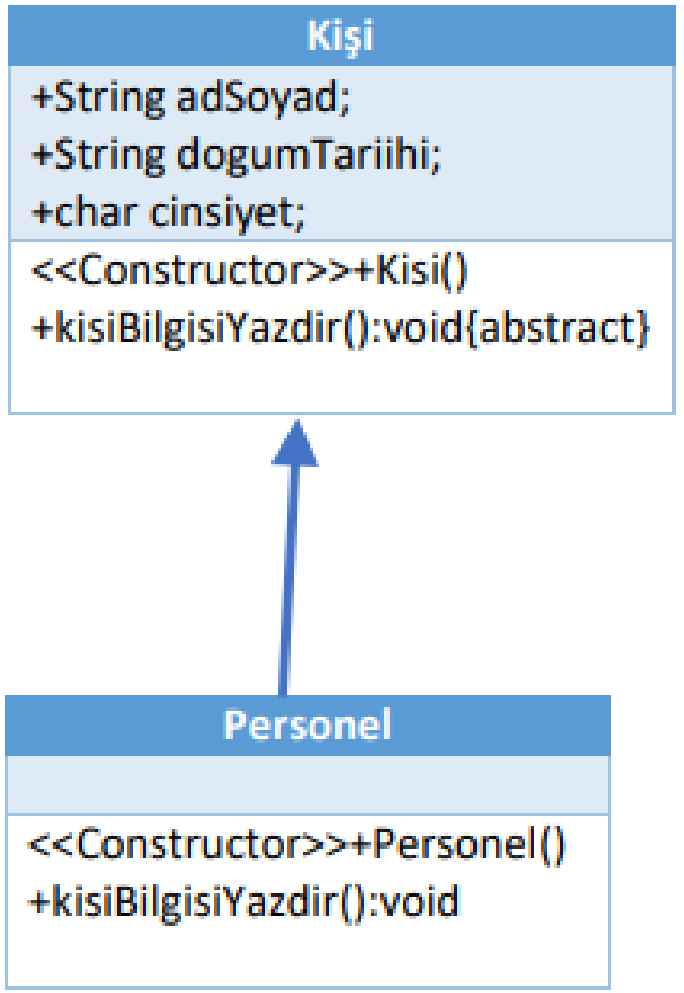
- Soyut sınıflar genellikle soyut metotlar (abstract methods) içerirler.
- Soyut bir metot hiçbir metot ifadesi içermez; bu sınıftan türetilen sınıflar, bu metotları ezmelidir(Override).
- Soyut metotlarda kod gövdesi bulunmaz. Bu metotların gövdeleri kalıtılan(türetilen) sınıfta tanımlanır.

```
public abstract void kisiBilgisiYazdir();  **
```

- Soyut bir sınıftan miras alınarak oluşturulan sınıflarda, yani türetilen sınıflarda; override kelimesi kullanılarak Soyut sınıfta bulunan soyut metotların gövdeleri oluşturulmalıdır.
- **Abstract tanımlanmış metotlar ezilmelidir(override).

Abstract(Soyut) Sınıflar

```
public abstract class Kisi {
        String adSoyad;
        String dogumTarihi;
        char cinsiyet;

        public abstract void kisiBilgisiYazdir();

}
```



**Önemli: Soyut bir metot tanımlamadan da soyut bir sınıf oluşturulabilir ancak soyut sınıf oluşturulmadan soyut bir metot oluşturulamaz.

Abstract(Soyut) Sınıflar

```
public class Personel extends Kisi {

    @Override
    public void kisiBilgisiYazdir() {
        System.out.println("Personel sınıfı kişi bilgisi yazdır...");
    }

}
```

Abstract(Soyut) Sınıf Örnek

```
public abstract class Body{
        public double density=1.0;
        public Body(double d)
        {
                density=d;  //
        }
        public double getDensity(){
                return density;
        }
        public double getMass(){
        return density*getVolume();          //Hacim
        }
         public abstract double getVolume();
}
```

Abstract(Soyut) Sınıf Örnek

```
public class CubeBody extends Body{
        public double edge=1.0;
        public CubeBody(double d,double e){
                super(d);
                edge=e;   //kenar
        }
        public double getVolume(){
                return edge*edge*edge;
        }
}
```

Abstract(Soyut) Sınıf Örnek

```
public class SphereBody extends Body{
        public double radius=1.0;
        public SphereBody(double d,double r){
                super(d);
                radius=r;   //yarıçap
        }
        public double getVolume(){
                return (3.14 * radius * radius * radius )/3;
        }
}
```

Kaynaklar

- Java ve Java Teknolojileri, *Tevfik KIZILÖREN* – Kodlab Yayınları
- Dr Öğr. Üyesi Aysun Zehra ALTIKARDEŞ- Nesne Yönelimli Programlama Notları
- http://web.firat.edu.tr/iaydin/bmu112/week_8_polymorphism.pdf
- Doç. Dr. Deniz KILINÇ CBÜ–Nesne Yönelimli Programlama Ders Notları
- https://docs.oracle.com/javase/tutorial/java/IandI/abstract.html#:~:text=An%20abstract%20class%20is%20a,but%20they%20can%20be%20subclassed.