

6. Ders: JAVA ile Nesne Yönelimli Programlama Çok biçimlilik (Polymorphism)

Fırat Üniversitesi Teknoloji Fakültesi Yazılım Mühendisliği Bölümü

YMH112 Algoritma ve Programlama-II

Dr. Öğr. Üyesi Yaman Akbulut

JAVA ile Nesne Yönelimli Programlama

- <http://www.kriptarium.com/algoritma.html> (Yardımcı kaynak)
- JAVA ile Nesne Yönelimli Programlama (Ders: 24-34) video.
 - Ders 24: this Anahtar Sözcüğünün Kullanımı I (izle)
 - Ders 25: this Anahtar Sözcüğünün Kullanımı II (izle)
 - Ders 26: Kalıtım/Miras (Inheritance) (izle)
 - Ders 27: Kalıtım Türleri (izle)
 - Ders 28: Java'da Neden Çoklu Kalıtım Desteklenmiyor? (izle)
 - Ders 29: Java Polimorfizmi / Yöntem Aşırı Yükleme (Method Overloading) (izle)
 - Ders 30: Java Polimorfizmi / Yöntem Geçersiz Kılma (Method Overriding) (izle)
 - Ders 31: super Anahtar Sözcüğünün Kullanımı (izle)
 - Ders 32: final Anahtar Sözcüğünün Kullanımı (izle)
 - Ders 33: Soyut Sınıf (Abstract Class) (izle)
 - Ders 34: Arayüz (Interface) (izle)

Java Keywords

abstract	double	int	super
assert	else	interface	switch
boolean	enum	long	synchronized
break	extends	native	this
byte	final	new	throw
case	finally	package	throws
catch	float	private	transient
char	for	protected	try
class	goto	public	void
const	if	return	volatile
continue	implements	short	while
default	import	static	
do	instanceof	strictfp	

Metot Geçersiz Kılma (Method Overriding)

Bir metodu geçersiz kılmak için, o metodun alt sınıfta, üst sınıfındakiyle aynı imza ve aynı dönüş türü kullanılarak tanımlanması gerekir.

Bir alt sınıf, metotlarını bir üst sınıftan kalıtım yoluyla miras alır.

Bazen alt sınıfın, üst sınıfta tanımlanan bir metodun uygulamasını (implementation) değiştirmesi gerekir.

Bu, **metodu geçersiz kılma** olarak adlandırılır.

Metot Geçersiz Kılma (Method Overriding)

```
1 public class TemelGeometrikSekil {  
2     private String renk = "beyaz";  
3     private boolean dolgu;  
4     private java.util.Date olusturulmaTarihi;  
5 }
```

```
45  
46 /** Bu nesnenin string sunumunu dondur */  
47 public String toString() {  
48     return "Olusturulma Tarihi: " + olusturulmaTarihi  
49     + "\nrenk: " + renk + " ve dolgu: " + dolgu;  
50 }  
51 }
```

```
1 public class TemelGeometrikSekildenCember  
2     extends TemelGeometrikSekil{  
3     private double yaricap;  
4  
5     public TemelGeometrikSekildenCember() {  
6     }  
7 }
```

```
49  
50 // Üst sınıfta tanımlanan toString metodunu gecersiz kılma  
51 public String toString() {  
52     return super.toString() + "\n yari cap: " + yaricap;  
53 }  
54 }
```

```
C:\Program Files\Java\jdk-15.0.1\bin\yeni2>java TestCemberDikdortgen  
Bir cember: Olusturulma Tarihi: Sun Apr 04 13:21:51 TRT 2021  
renk: beyaz ve dolgu: false  
Renk: beyaz  
Yaricap: 1.0  
Alan: 3.141592653589793  
Cap: 2.0
```

```
Bir dikdortgen: Olusturulma Tarihi: Sun Apr 04 13:21:51 TRT 2021  
renk: beyaz ve dolgu: false  
Alan: 8.0  
Cevre: 12.0
```

```
C:\Program Files\Java\jdk-15.0.1\bin\yeni2>java TestCemberDikdortgen  
Bir cember: Olusturulma Tarihi: Sun Apr 11 00:02:23 TRT 2021  
renk: beyaz ve dolgu: false  
yari cap: 1.0  
Renk: beyaz  
Yaricap: 1.0  
Alan: 3.141592653589793  
Cap: 2.0
```

```
Bir dikdortgen: Olusturulma Tarihi: Sun Apr 11 00:02:23 TRT 2021  
renk: beyaz ve dolgu: false  
Alan: 8.0  
Cevre: 12.0
```

Metot Geçersiz Kılma (Method Overriding)

```
1 public class TestCemberDikdortgen {
2     public static void main(String[] args) {
3         TemelGeometrikSekildenCember cember = new TemelGeometrikSekildenCember(1);
4         System.out.println("Bir cember: " + cember.toString());
5         System.out.println("Renk: " + cember.getRenk());
6         System.out.println("Yaricap: " + cember.getYaricap());
7         System.out.println("Alan: " + cember.getAlan());
8         System.out.println("Cap: " + cember.getCap());
9
10        TemelGeometrikSekildenDikdortgen dikdortgen = new TemelGeometrikSekildenDikdortgen(2, 4);
11        System.out.println("\nBir dikdortgen: " + dikdortgen.toString());
12        System.out.println("Alan: " + dikdortgen.getAlan());
13        System.out.println("Cevre: " + dikdortgen.getCevre());
14    }
15 }
```

```
49
50 // Üst sınıfta tanımlanan toString metodunu gecersiz kılma
51 public String toString() {
52     return super.toString() + "\n yari cap: " + yaricap;
53 }
```

C:\Program Files\Java\jdk-15.0.1\bin\yeni2>java TestCemberDikdortgen

Bir cember: Olusturulma Tarihi: Sun Apr 04 13:21:51 TRT 2021

renk: beyaz ve dolgu: false

Renk: beyaz

Yaricap: 1.0

Alan: 3.141592653589793

Cap: 2.0

Bir dikdortgen: Olusturulma Tarihi: Sun Apr 04 13:21:51 TRT 2021

renk: beyaz ve dolgu: false

Alan: 8.0

Cevre: 12.0

C:\Program Files\Java\jdk-15.0.1\bin\yeni2>java TestCemberDikdortgen

Bir cember: Olusturulma Tarihi: Sun Apr 11 00:02:23 TRT 2021

renk: beyaz ve dolgu: false

yari cap: 1.0

Renk: beyaz

Yaricap: 1.0

Alan: 3.141592653589793

Cap: 2.0

Bir dikdortgen: Olusturulma Tarihi: Sun Apr 11 00:02:23 TRT 2021

renk: beyaz ve dolgu: false

Alan: 8.0

Cevre: 12.0

Geçersiz kılma vs. Aşırı yükleme (Overriding vs. Overloading)

Geçersiz kılma (overriding), alt sınıftaki bir metot için yeni bir uygulama (implement) sağlamak anlamına gelir.

Aşırı yükleme (overloading), aynı isme ve farklı imzalara sahip birden çok metodu tanımlamak anlamına gelir.

```
1 public class Test1 {
2     public static void main(String[] args) {
3         A1 a = new A1();
4         a.p(10);
5         a.p(10.0);
6     }
7 }
8 class B1 {
9     public void p(double i) {
10        System.out.println(i * 2);
11    }
12 }
13 class A1 extends B1 {
14     // Bu metot B1'deki metodu geçersiz kılar
15     public void p(double i) {
16        System.out.println(i * 3);
17    }
18 }
```

```
1 public class Test2 {
2     public static void main(String[] args) {
3         A2 a = new A2();
4         a.p(10);
5         a.p(10.0);
6     }
7 }
8 class B2 {
9     public void p(double i) {
10        System.out.println(i * 2);
11    }
12 }
13 class A2 extends B2 {
14     // Bu metot B2'deki metodu aşırı yükler
15     public void p(int i) {
16        System.out.println(i * 3);
17    }
18 }
```

Geçersiz kılma vs. Aşırı yükleme (Overriding vs. Overloading)

```
C:\Program Files\Java\jdk-15.0.1\
bin\yeni2>java Test1
30.0
30.0
```

```
1 public class Test1 {
2     public static void main(String[] args) {
3         A1 a = new A1();
4         a.p(10);
5         a.p(10.0);
6     }
7 }
8 class B1 {
9     public void p(double i) {
10        System.out.println(i * 2);
11    }
12 }
13 class A1 extends B1 {
14     // Bu metot B1'deki metodu geçersiz kılar
15     public void p(double i) {
16        System.out.println(i * 3);
17    }
18 }
```

```
C:\Program Files\Java\jdk-15.0.1\
bin\yeni2>java Test2
30
20.0
```

```
1 public class Test2 {
2     public static void main(String[] args) {
3         A2 a = new A2();
4         a.p(10);
5         a.p(10.0);
6     }
7 }
8 class B2 {
9     public void p(double i) {
10        System.out.println(i * 2);
11    }
12 }
13 class A2 extends B2 {
14     // Bu metot B2'deki metodu aşırı yükler
15     public void p(int i) {
16        System.out.println(i * 3);
17    }
18 }
```


Aşırı yükleme (Overloading)

```
1 public class Test2 {
2     public static void main(String[] args) {
3         A2 a = new A2();
4         a.p(10);
5         a.p(10.0);
6         a.p("Yazilim Muhendisligi");
7     }
8 }
9 class B2 {
10    public void p(double i) {
11        System.out.println(i * 2);
12    }
13 }
14 class A2 extends B2 {
15    // Bu metot B2'deki metodu aşırı yükler
16    public void p(int i) {
17        System.out.println(i * 3);
18    }
19    // Bu metot B2'deki metodu aşırı yükler
20    public void p(String yazi) {
21        System.out.println("Merhaba " + yazi
22        + "\nMerhaba " + yazi );
23    }
24 }
```

Aşırı yükleme (overloading), aynı isme ve farklı imzalara sahip birden çok metodu tanımlamak anlamına gelir.

```
C:\Program Files\Java\jdk-15.0.1\bin\yeni2>
javac Test2.java
```

```
C:\Program Files\Java\jdk-15.0.1\bin\yeni2>
java Test2
30
20.0
Merhaba Yazilim Muhendisligi
Merhaba Yazilim Muhendisligi
```

Çok biçimlilik (Polymorphism)

Çok biçimlilik, bir üst tip değişkeninin bir alt tip nesneye başvurabileceği anlamına gelir.

Nesne yönelimli programlamanın üç ayağı
 kapsülleme (encapsulation),
 kalıtım (inheritance)
 ve **çok biçimlilik** (polymorphism).

Üst tip ve alt tip (supertype and subtype)

Bir sınıf, bir tipi tanımlar.

Bir alt sınıf (subclass) tarafından tanımlanan bir tipe **alt tip** (subtype) denir.

Bir üst sınıf (superclass) tarafından tanımlanan bir tipe **üst tip** (supertype) denir.

Bir önceki derste oluşturduğumuz *Cember* bir *GeometrikSekil*'in alt tipi veya *GeometrikSekil Cember*'in üst tipi olduğunu söyleyebiliriz.

Kalıtım ilişkisi, bir alt sınıfın ek yeni özelliklerle üst sınıfından özellikleri devralmasını sağlar.

Çok biçimlilik (Polymorphism)

Bir alt sınıf, üst sınıfının bir özelleşmiş (uzmanlaşmış) halidir;

Bir alt sınıfın her örneği (nesnesi) aynı zamanda üst sınıfının bir örneğidir, ancak bunun tersi geçerli değildir.

Örneğin, her çember geometrik bir nesnedir, ancak her geometrik nesne bir çember değildir.

Bu nedenle, bir alt sınıfın bir örneğini her zaman üst sınıf türündeki bir parametreye geçirebilirsiniz.

alt sınıf
subclass

üst sınıf
superclass



```
public class TemelGeometrikSekildenCember extends TemelGeometrikSekil
```

alt sınıf
subclass

üst sınıf
superclass



```
public class TemelGeometrikSekildenDikdortgen extends TemelGeometrikSekil
```

CokbicimlilikDemo.java

```
1 public class CokbicimlilikDemo {
2     /** Main metot */
3     public static void main(String[] args) {
4         // cember ve dikdortgen ozelliklerini goster
5         nesneGoster(new TemelGeometrikSekildenCember(1, "kirmizi", false));
6         nesneGoster(new TemelGeometrikSekildenDikdortgen(1, 1, "siyah", true));
7     }
8
9     /** Geometrik nesne ozelliklerini goster */
10    public static void nesneGoster(TemelGeometrikSekil nesne) {
11        System.out.println(nesne.getOlusturulmaTarihi() +
12            " tarihinde nesne olusturuldu. Rengi " +
13            nesne.getRenk() + " dir.");
14    }
15 }
```

```
C:\Program Files\Java\jdk-15.0.1\bin\yeni2>java CokbicimlilikDemo
Sat Apr 10 23:19:38 TRT 2021 tarihinde nesne olusturuldu. Rengi kirmizi dir.
Sat Apr 10 23:19:39 TRT 2021 tarihinde nesne olusturuldu. Rengi siyah dir.
```

Alt sınıfın bir nesnesi, üst sınıf nesnesinin kullanıldığı her yerde kullanılabilir. Bu genellikle **çok biçimlilik** (polimorfizm, polymorphism) olarak bilinir.

Basit bir ifadeyle, **çok biçimlilik**, bir süper tip değişkeninin bir alt tip nesneye atıfta bulunabileceği anlamına gelir.