4. Ders: JAVA ile Nesne Yönelimli Programlama
Nesne Yönelimli Düşünme (Object-Oriented Thinking)

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: 14-25) video.

    - Ders 14: Java Programları Yazarken Kullanılan Yaygın Yaklaşım (izle)
    - Ders 15: Nesneleri Hazırlamanın Farklı Yolları (izle)
    - Ders 16: Anonim Nesneler (Anonymous Object) (izle)
    - Ders 17: Örnek Bir Uygulama (izle)
    - Ders 18: Yapıcı Metotlar (Constructor Methods) (izle)
    - Ders 19: Java'da Constructor Aşırı Yüklemesi (izle)
    - Ders 20: Static Anahtar Sözcüğü (izle)
    - Ders 21: Statik Değişkenler ile Program Sayacı (izle)
    - Ders 22: Statik Yöntemler (izle)
    - Ders 23: Statik Yöntemler Kullanılırken Dikkat Edilecek Unsurlar (izle)
    - Ders 24: this Anahtar Sözcüğünün Kullanımı I (izle)
    - Ders 25: this Anahtar Sözcüğünün Kullanımı II (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	

Kaynak: Introduction to JAVA Programming 10th Edition, Y. Daniel Liang

Static anahtar sözcüğü

Bir sınıfın tüm örneklerinin (nesnelerinin) verileri paylaşmasını istiyorsanız, sınıf değişkenleri olarak da bilinen statik değişkenler kullanın.

Statik değişkenler, değişkenler için değerleri ortak bir bellek konumunda depolar. Bu ortak konum nedeniyle, bir nesne statik bir değişkenin değerini değiştirirse, aynı sınıftaki tüm nesneler etkilenir.

Java, statik değişkenlerin yanı sıra statik metotları da destekler. Statik metotlar, sınıfın bir örneğini oluşturmadan çağrılabilir.

Static anahtar sözcüğü

Static (Statik) bir değişken sınıfın tüm nesneleri tarafından paylaşılır.

Statik bir metot, sınıfın örnek (nesne) üyelerine erişemez veya üyelerini kullanamaz.

BasitCember cember1 = new BasitCember();
BasitCember cember2 = new BasitCember(5);

cember1.yaricap
cember2.yaricap

```
26  // İki yapılandırıcı ile Cember sınıfı tanimalama
27  class BasitCember {
28      double yaricap;
29
30      /** Yaricapi 1 olan bir cember yapilandir */
31      BasitCember() {
32      yaricap = 1;
33      }
34
35      /** Yaricapi belirlenebilen bir cember yapilandir */
36      BasitCember(double yeniYaricap) {
37      yaricap = yeniYaricap;
38      }
39
40      /** Bu cemberin alanini geri döndür */
41      double getAlan() {
42      return yaricap * yaricap * Math.PI;
43      }
44
45      /** Bu cemberin cevresini geri döndür */
46      double getCevre() {
47      return 2 * yaricap * Math.PI;
48      }
49
50      /** Bu cember icin yeni bir yaricap ata */
51      void setYaricap(double yeniYaricap) {
52      yaricap = yeniYaricap;
53      }
54  }
```

Örnek 1a: Static anahtar sözcüğü (yokken)

```java
public class Sayici {
    int sayac;

    Sayici(){
        sayac++;
        System.out.println(sayac);
    }
}
```

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

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

C:\Program Files\Java\jdk-15.0.1\bin\yeni2>java TestSayici
1
1
1
1
1
```

```java
public class TestSayici{
    public static void main(String arg[]){
        Sayici nesne1 = new Sayici();
        Sayici nesne2 = new Sayici();
        Sayici nesne3 = new Sayici();
        Sayici nesne4 = new Sayici();
        Sayici nesne5 = new Sayici();
    }
}
```

Örnek 1b: Static anahtar sözcüğü (varken)

```java
public class Sayici {
    static int sayac;

    Sayici(){
        sayac++;
        System.out.println(sayac);
    }
}
```

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

C:\Program Files\Java\jdk-15.0.1\bin\yeni2>java TestSayici
1
2
3
4
5
```

```java
public class TestSayici{
    public static void main(String arg[]){
        Sayici nesne1 = new Sayici();
        Sayici nesne2 = new Sayici();
        Sayici nesne3 = new Sayici();
        Sayici nesne4 = new Sayici();
        Sayici nesne5 = new Sayici();
    }
}
```

Örnek 1c: Static anahtar sözcüğü (nesne tanımlamadan)

```java
public class Sayici {
    static int sayac = 10;

    Sayici(){
        sayac++;
        System.out.println(sayac);
    }
}
```

```java
public class TestSayici{
    public static void main(String arg[]){
        System.out.println(Sayici.sayac);
        Sayici nesne1 = new Sayici();
    }
}
```

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

C:\Program Files\Java\jdk-15.0.1\bin\yeni2>
java TestSayici
10
11
```

Static anahtar sözcüğü

Bir sınıfın tüm örneklerinin (nesnelerinin) verileri paylaşmasını istiyorsanız, sınıf değişkenleri olarak da bilinen statik değişkenler kullanın.

Statik değişkenler, değişkenler için değerleri ortak bir bellek konumunda depolar. Bu ortak konum nedeniyle, bir nesne statik bir değişkenin değerini değiştirirse, aynı sınıftaki tüm nesneler etkilenir.

Java, statik değişkenlerin yanı sıra statik metotları da destekler. Statik metotlar, sınıfın bir örneğini oluşturmadan çağrılabilir.

Örnek 2a: Static degisken, metot

```
1  public class A {
2      int i = 5;
3      static int k = 2;
4
5      public static void main(String[] args) {
6      int j = i; // hatali: i ornek degiskeni
7      m1(); // hatali: m1() metot
8      }
9
10     public void m1() {
11         // dogru: ornek, statik degisken ve metot
12         i = i + k + m2(i, k);
13     }
14
15     public static int m2(int i, int j) {
16     return (int)(Math.pow(i, j));
17     }
18 }
```

Math sınıfındaki bütün metotlar statiktir.

main metodu da statiktir.

Örnek 2b: Static degisken, metot

```
1  public class A {
2      int i = 5;
3      static int k = 2;
4
5      public static void main(String[] args) {
6          A a = new A();
7          int j = a.i; // i ornek degiskeni
8          a.m1(); // m1() metot
9      }
10
11     public void m1() {
12         // dogru: ornek, statik degisken ve metot
13         i = i + k + m2(i, k);
14     }
15
16     public static int m2(int i, int j) {
17     return (int)(Math.pow(i, j));
18     }
19 }
```

Math sınıfındaki bütün metotlar statiktir.

main metodu da statiktir.

Örnek 3:

```java
public class Hesapla{
    static int kupAl(int a){
        return a*a*a;
    }
    static int kareAl(int a){
        return a*a;
    }
}
```

Bir sınıftaki static metotları nesne oluşturmadan doğrudan kullanabiliriz.

```java
public class TestHesapla{
    public static void main(String arg[]){
        System.out.println(Hesapla.kupAl(5));
        System.out.println(Hesapla.kupAl(10));
        System.out.println(Hesapla.kareAl(5));
        System.out.println(Hesapla.kareAl(10));
    }
}
```

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

C:\Program Files\Java\jdk-15.0.1\bin\yeni2>
java TestHesapla
125
1000
25
100
```

package, public, default, private

package, sınıfları organize etmek ve düzenlemek için kullanılır.
package paketAdi;
package (paket) en üstte olacak şekilde tanımlanır.

public, sınıflara, metotlara ve veri alanlarına herhangi bir sınıftan erişilebilmesini sağlar.

private, metotların ve veri alanlarının sadece kendi sınıfı içinde erişilebilmesini sağlar.

Bir şey belirtilmediyse (public, private, protected gibi) varsayılan (default) olarak aynı paketin içindeki tüm sınıflara, metotlara ve veri alanlarına herhangi bir sınıf tarafından erişilebilir.

package, public, default, private

```
package p1;

public class C1 {
  public int x;
  int y;
  private int z;

  public void m1() {
  }
  void m2() {
  }
  private void m3() {
  }
}
```

```
package p1;

public class C2 {
  void aMethod() {
    C1 o = new C1();
    can access o.x;
    can access o.y;
    cannot access o.z;

    can invoke o.m1();
    can invoke o.m2();
    cannot invoke o.m3();
  }
}
```

```
package p2;

public class C3 {
  void aMethod() {
    C1 o = new C1();
    can access o.x;
    cannot access o.y;
    cannot access o.z;

    can invoke o.m1();
    cannot invoke o.m2();
    cannot invoke o.m3();
  }
}
```

FIGURE 9.14 The private modifier restricts access to its defining class, the default modifier restricts access to a package, and the public modifier enables unrestricted access.

public/public olmayan sınıf

```
package p1;

class C1 {
  ...
}
```

```
package p1;

public class C2 {
  can access C1
}
```

```
package p2;

public class C3 {
  cannot access C1;
  can access C2;
}
```

FIGURE 9.15 A nonpublic class has package-access.

package, public, default, private

private olduğunda metotlar ve veri alanları sadece kendi sınıfı içinde kullanılabilir.

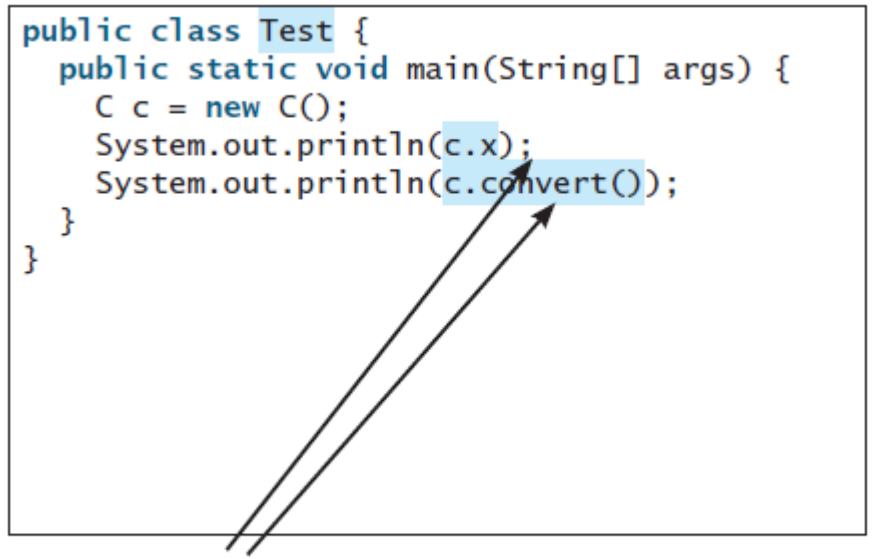
```
public class C {
  private boolean x;

  public static void main(String[] args) {
    C c = new C();
    System.out.println(c.x);
    System.out.println(c.convert());
  }

  private int convert() {
    return x ? 1 : -1;
  }
}
```

(a) This is okay because object c is used inside the class C.

```
public class Test {
  public static void main(String[] args) {
    C c = new C();
    System.out.println(c.x);
    System.out.println(c.convert());
  }
}
```



(b) This is wrong because x and convert are private in class C.

FIGURE 9.16 An object can access its private members if it is defined in its own class.

Ödev 1a:

Aşağıdaki programların çıktısı nedir?

```
import java.util.Date;

public class Test {
  public static void main(String[] args) {
    Date date = null;
    m1(date);
    System.out.println(date);
  }

  public static void m1(Date date) {
    date = new Date();
  }
}
```

(a)

```
import java.util.Date;

public class Test {
  public static void main(String[] args) {
    Date date = new Date(1234567);
    m1(date);
    System.out.println(date.getTime());
  }

  public static void m1(Date date) {
    date = new Date(7654321);
  }
}
```

(b)

Ödev 1b:

Aşağıdaki programların çıktısı nedir?

```
import java.util.Date;

public class Test {
  public static void main(String[] args) {
    Date date = new Date(1234567);
    m1(date);
    System.out.println(date.getTime());
  }

  public static void m1(Date date) {
    date.setTime(7654321);
  }
}
```

(c)

```
import java.util.Date;

public class Test {
  public static void main(String[] args) {
    Date date = new Date(1234567);
    m1(date);
    System.out.println(date.getTime());
  }

  public static void m1(Date date) {
    date = null;
  }
}
```

(d)

Ödev 2a:

Aşağıdaki programların çıktısı nedir?

```
public class Test {
  public static void main(String[] args) {
    int[] a = {1, 2};
    swap(a[0], a[1]);
    System.out.println("a[0] = " + a[0]
      + " a[1] = " + a[1]);
  }

  public static void swap(int n1, int n2) {
    int temp = n1;
    n1 = n2;
    n2 = temp;
  }
}
```

(a)

```
public class Test {
  public static void main(String[] args) {
    int[] a = {1, 2};
    swap(a);
    System.out.println("a[0] = " + a[0]
      + " a[1] = " + a[1]);
  }

  public static void swap(int[] a) {
    int temp = a[0];
    a[0] = a[1];
    a[1] = temp;
  }
}
```

(b)

Ödev 2b:

Aşağıdaki programların çıktısı nedir?

```
public class Test {
  public static void main(String[] args) {
    T t = new T();
    swap(t);
    System.out.println("e1 = " + t.e1
      + " e2 = " + t.e2);
  }

  public static void swap(T t) {
    int temp = t.e1;
    t.e1 = t.e2;
    t.e2 = temp;
  }
}

class T {
  int e1 = 1;
  int e2 = 2;
}
```

(c)

```
public class Test {
  public static void main(String[] args) {
    T t1 = new T();
    T t2 = new T();
    System.out.println("t1's i = " +
      t1.i + " and j = " + t1.j);
    System.out.println("t2's i = " +
      t2.i + " and j = " + t2.j);
  }
}

class T {
  static int i = 0;
  int j = 0;

  T() {
    i++;
    j = 1;
  }
}
```

(d)

Ödev 3:

Banka Örneği

- Bir banka müşterisinin hesap numarası, müşteri adı ve bankada ne kadar parası olduğu bilgilerini saklamaktadır.
- Müşteri bankaya para yatırabilir veya para çekebilir.
- Para yatırma ve çekme işlemleri sonucunda hesaptaki para miktarının tutarlı olduğunu gösteren kontrol isimli bir yöntem olmalıdır.
- Göster metodu müşteri no, adı ve tutar bilgileri yazdırmaktadır.