

Java Programlama Dili ve Algoritmadan Kodlamaya Geçiř

Dr. Fatih KALEMKUŐ

Kafkas Üniversitesi

Java'da Veri Tipleri

Java'da kullanılacak olan deęişkenler ve veri tipleri önceden bildirilmelidir.

1. Tamsayı Veri Tipleri

Veri Tipi	Aktarılabilecek En Küçük Deęer	Aktarılabilecek En Büyük Deęer	Bellekte Kapladığı Alan (byte)
byte	-2^7	2^7-1	1
short	-2^{15}	$2^{15}-1$	2
int	-2^{31}	$2^{31}-1$	4
long	-2^{63}	$2^{63}-1$	8

2. Ondalıklı Sayı Veri Tipleri

Veri Tipi	Aktarılabilecek En Küçük Deęer	Aktarılabilecek En Büyük Deęer	Bellekte Kapladığı Alan (byte)
float	$3,4 \cdot 10^{-38}$	$3,4 \cdot 10^{38}$	4
double	$1,7 \cdot 10^{-308}$	$1,7 \cdot 10^{308}$	8

Java'da Veri Tipleri

3. Alfasayısal Veri Tipleri

Tek karakter ve karakter grubu (kelime, cümle vb.) için kullanılacak iki veri tipi olup aşağıda gösterilmiştir:

Veri Tipi	Anlamı
char	Tek tırnak içinde bir karakter aktarılabilir.
String	Çift tırnak içinde birden fazla karakter aktarılabilir.

Java Program Yapısı

Program Başlığı:

Program hakkındaki açıklamaları ya da ismini içeren ifade veya ifadelerdir.

```
// açıklamalar veya program başlığı  
.....
```

Sınıf Çağırma Bölümü:

Java dilinde “sınıf”lar (*class*), “paket” (*package*) olarak adlandırılan dosyalarda toplanmışlardır. Diğer sınıfların, yazılacak programda kullanılabilmesi için önceden çağırılması gerekir. Herhangi bir Java programı yazıldığında, Java standart kütüphanesi (*java.lang* paketi) otomatik olarak çağırılır. Fakat kullanılacak diğer paketlere ait sınıflar, nesnelere, fonksiyonlar kullanılacaksa bunların “*import*” ile çağırılması gerekir.

```
import paket.sınıf;  
.....
```

Java Program Yapısı

Örneğin `"import java.util.Scanner"` komutu ile Scanner sınıfı ilgili programda artık kullanılabilir veya `"import java.util.*"` ile de "java.util" paketindeki tüm sınıflar çağrılıp kullanılabilir.

Paket	Sınıfları
java.lang	Java programlama dilinin temel sınıfları
java.applet	Applet uygulamaları sınıfları
java.awt	Grafiksel arayüz uygulamaları sınıfları
java.io	Sistem giriş/çıkış sınıfları
java.sql	Veritabanı programlama sınıfları
javax.net	Ağ uygulamaları sınıfları

Java Program Yapısı

Sınıflar:

Java ile geliştirilen uygulamaların bileşenleri “.class” uzantılı dosyalarda saklanırlar. Java’da sınıf tanımlama en genel haliyle aşağıdaki gibi yapılır.

```
denetleyiciler class sınıfAdı{  
.....  
  
    program kodları  
    .....  
}
```

Java Program Yapısı

Değişken Tanımlama:

Java'da kullanılacak değişkenler önceden bildirilmelidir.

```
veri tipi değişken adı;  
.....
```

Sabit Tanımlama:

Java'da sabit tanımlamak için "final" kullanılmaktadır.

```
final veri tipi sabit adı = sabit değeri;  
.....
```

Java Program Yapısı

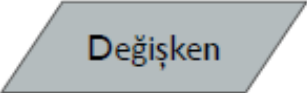

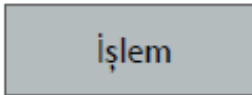
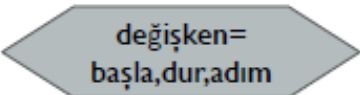
Ekrana “merhaba” yazan programı inceleyelim.

```
//Merhaba
public class Merhaba {
    public static void main (String[ ] args) {
        System.out.println (“Merhaba”);
    }
}
```

Dosya adı ile sınıf adının aynı olması gerekir. “main”deki “public” deyimi, sınıfın veya yöntemin herkese açık (dışarıdan erişilebilir) olduğunu belirtir. “static” deyimi sınıf tarafından paylaşıldığını, “void” de bir değer geri göndermediğini (dönmediğini) belirtir.

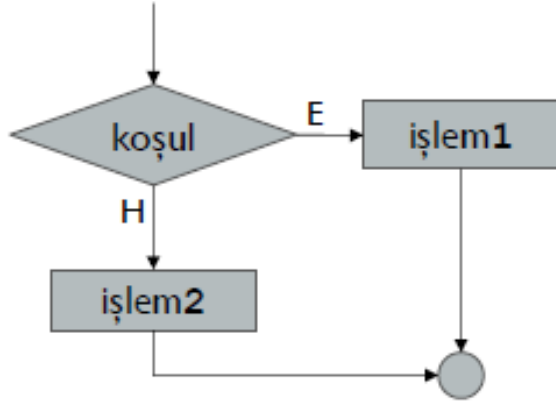
Akış Diyagramından Kodlamaya Geçiř

Sembollerin Karřılıkları

řekil	Java'daki Karřılıđı
 Bařla	Açıklamalar, bildirimler
 Deđiřken	<code>deđiřken=nextInt();</code> <code>deđiřken=nextLine(); ...</code>
 Deđiřken	<code>System.out.println(deđiřken);</code> <code>System.out.print(deđiřken); ...</code>
 iřlem	iřlem
 deđiřken= bařla,dur,adım	<pre>for (bařla; řart; adım) { }</pre>

Sembollerin Karşılıkları

Şekil



Java'daki Karşılığı

```
if (koşul) {  
    işlem1;  
} else {  
    işlem2;  
}
```

Veri Giriş Komutları

Java'da klavyeden veri girişi için "java.util" paketindeki "Scanner" sınıfının yöntemleri (System.in) kullanılır. Bu nedenle programın başında "`import java.util.Scanner`" ile sınıf çağrılır. "Scanner" sınıfının bazı yöntemleri aşağıdaki gibi özetlenebilir.

Paket	Sınıfları
<code>next()</code>	Klavyeden girilen ifadeyi ilk özel karakterine (boşluk) kadar alır.
<code>nextBoolean()</code>	Klavyeden girilen ifadeyi boolean tipinde alır.
<code>nextByte()</code>	Klavyeden girilen ifadeyi byte tipinde alır.
<code>nextDouble()</code>	Klavyeden girilen ifadeyi double tipinde alır.
<code>nextInt()</code>	Klavyeden girilen ifadeyi int tipinde alır.
<code>nextLine()</code>	Klavyeden girilen tüm satırı alır.
<code>nextLong()</code>	Klavyeden girilen ifadeyi long tipinde alır.
<code>nextShort()</code>	Klavyeden girilen ifadeyi short tipinde alır.

Veri Giriş Komutları

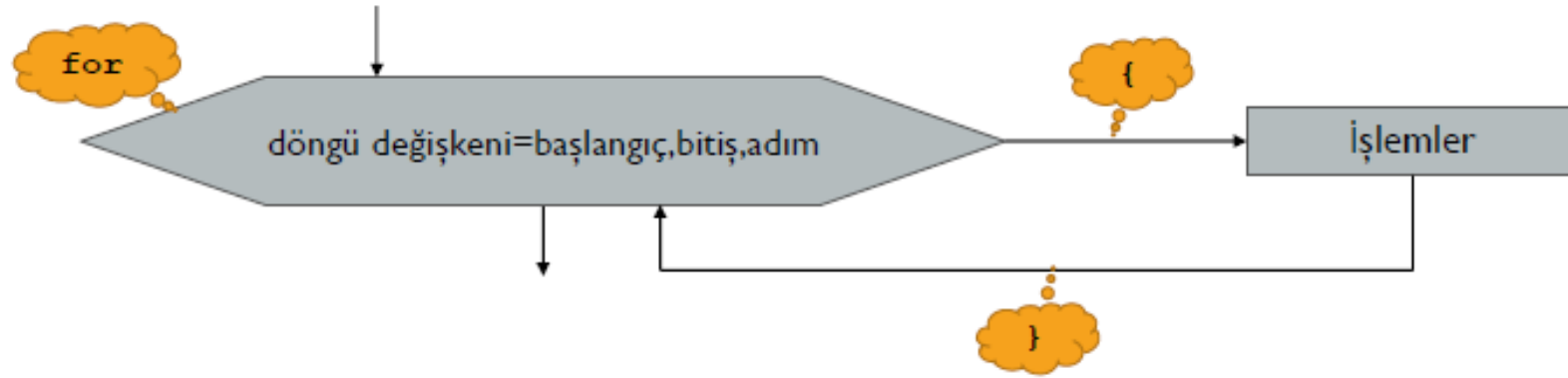
```
//Veri Girişleri
import java.util.Scanner;
public class VeriGiris {
    public static void main (String[ ] args) {
        String a;
        int b;
        Scanner klavye = new Scanner(System.in);
        System.out.print ("Bir cümle giriniz: ");
        a=klavye.nextLine();
        System.out.println ("Girdiğiniz cümle: "+a);
        System.out.print ("Bir sayı giriniz: ");
        b=klavye.nextInt();
        System.out.println ("Girdiğiniz sayı: "+b);
    }
}
```

Döngü Komutları

Tekrarlı işlemlerin yapılmasını sağlarlar. Döngüler üçe ayrılırlar:

- Sayıcı döngüler: Döngü işlemleri bir sayaca bağlı olarak gerçekleştirilir.
- Ön koşullu döngüler: Döngü işlemleri, döngü öncesinde kontrol edilen koşula bağlı olarak gerçekleştirilir.
- Son koşullu döngüler: Döngü işlemleri, döngü sonunda kontrol edilen koşula bağlı olarak gerçekleştirilir. Bu durumda döngü en az bir kez çalıştırılır.

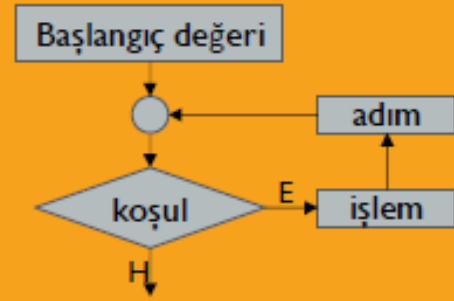
Döngü Komutları – For Döngüsü



```
for (tip başlangıç değeri; koşul; adım) {  
    .....  
    .....  
}
```

} işlemler

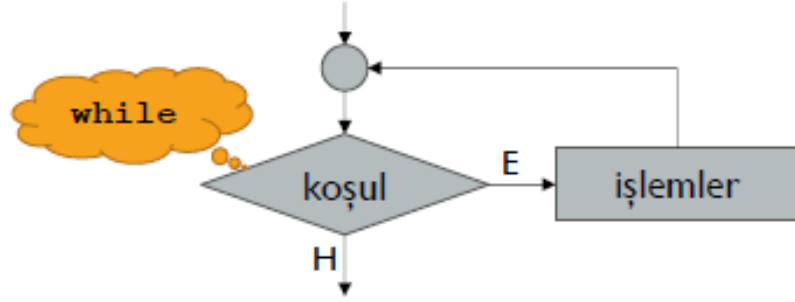
Alternatif gösterim



Döngü Komutları – For Döngüsü

```
//Döngü 1
public class ornek {
    public static void main (String[ ] args) {
        int t=0;
        int N=0;
        for (int i=1; i<=N; i++){
            t+=i;
        }
        System.out.println ("Birden N'e kadar sayıların
        toplamı: "+t);
    }
}
```

Döngü Komutları – While Döngüsü



```
while ( koşul ) {
```

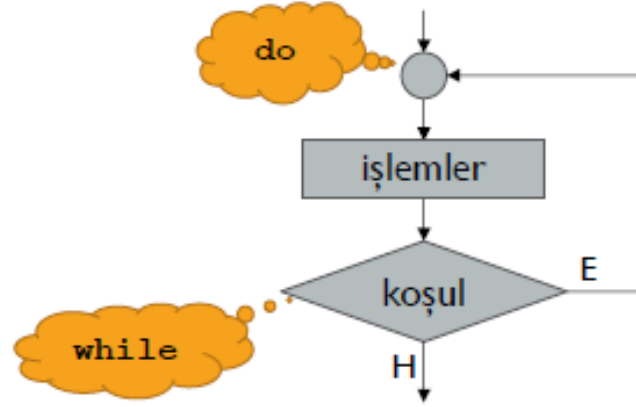
```
..... } işlemler
```

```
}
```

Döngü Komutları – While Döngüsü

```
//Döngü 2
import java.util.Scanner;
public class ornek {
    public static void main (String[ ] args) {
        float t=0;
        Scanner klavye = new Scanner(System.in);
        System.out.print ("Tek sayıların üst sınırı: ");
        int N=klavye.nextInt();
        int i=1;
        while (i<=N) {
            t+=i;
            i+=2;
        }
        System.out.println("Toplam: "+t);
    }
}
```

Döngü Komutları – do While Döngüsü



```
do {  
    .....  
    ..... } işlemler  
} while ( koşul );
```

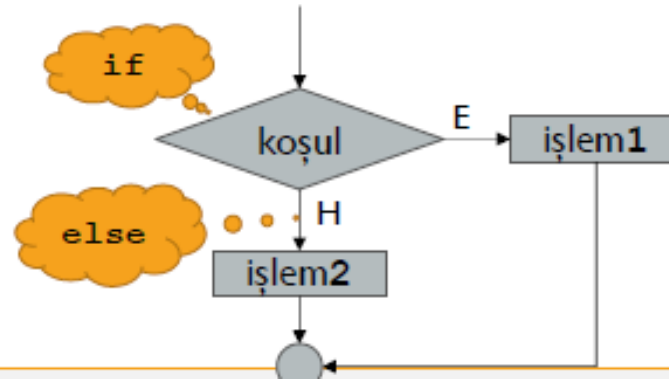
Döngü Komutları – do While Döngüsü

```
//Döngü 2
import java.util.Scanner;
public class ornek {
    public static void main (String[ ] args) {
        float t=0;
        Scanner klavye = new Scanner(System.in);
        System.out.print ("Çift sayıların üst sınırı: ");
        int N=klavye.nextInt();
        int i=2;
        do {
            t+=i;
            i+=2;
        } while (i<=N);
        System.out.println("Toplam: "+t);
    }
}
```

Karar (Karşılaştırma) Komutları – IF ELSE YAPISI

Koşulların kontrolünde kullanılan komutlardır. Karar komutları dört farklı yapıda olabilirler:

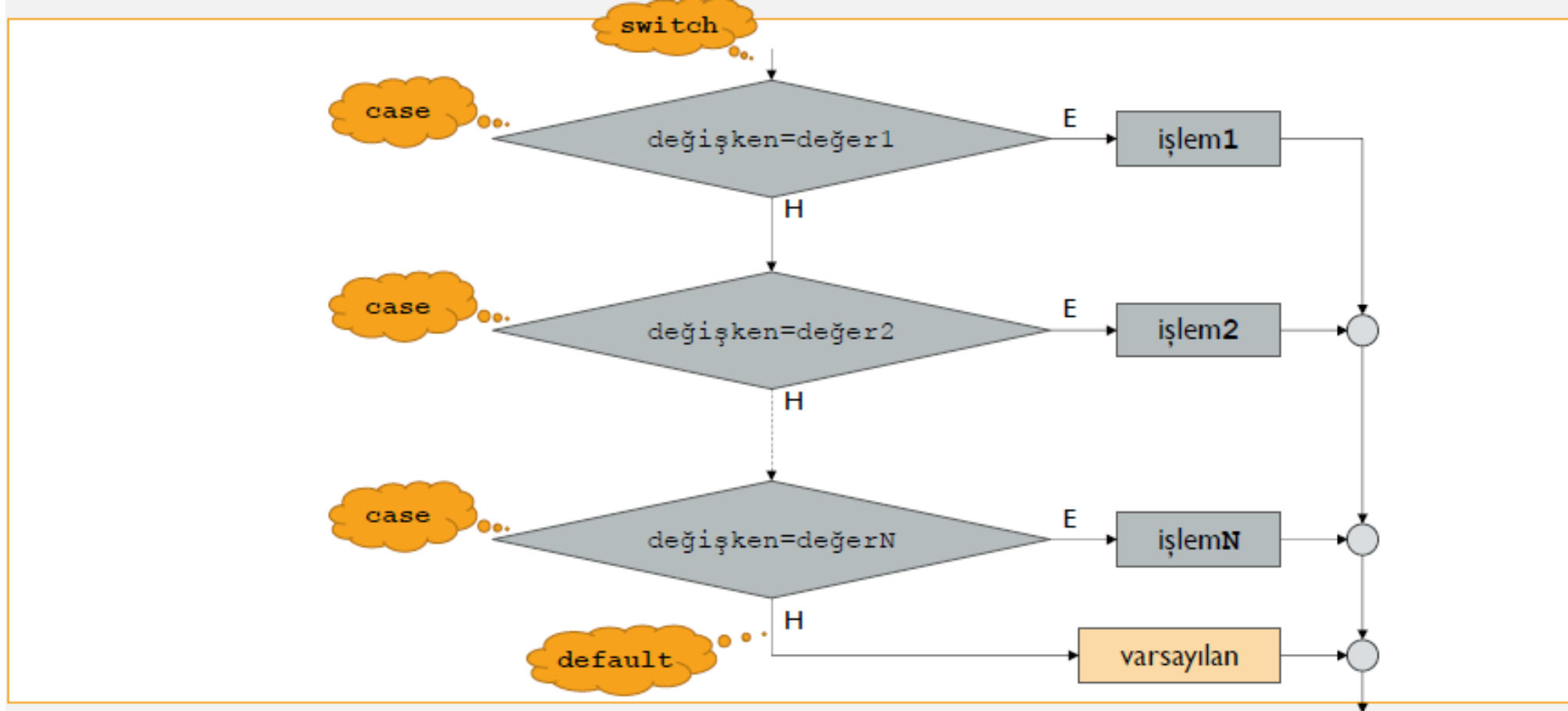
- **Yarım Form:** Sadece koşul doğru ise yapılacak işlemler vardır.
- **Tam Form:** Koşul doğru olduğunda ve koşul yanlış olduğunda yapılacak işlemler vardır.
- **Çok Koşullu Form:** Birçok koşulun durumuna göre yapılacak işlemler vardır.
- **Seçimli Form:** Kontrol değişkeninin değerine göre yapılacak işlemler vardır. Çok koşullu formun sade biçimidir.



Karar (Karşılaştırma) Komutları – IF ELSE YAPISI

```
//Karar 1
import java.util.Scanner;
public class ornek {
    public static void main (String[] args) {
        Scanner klavye = new Scanner(System.in);
        System.out.print ("Bir tamsayı giriniz: ");
        int a=klavye.nextInt();
        if (a>0) {
            System.out.println ("Pozitif");
        } else if (a<0){
            System.out.println("Negatif");
        } else{
            System.out.println("Sıfır");
        }
    }
}
```

Karar (Karşılaştırma) Komutları – SWITCH-CASE



Karar (Karşılaştırma) Komutları – SWITCH-CASE

```
//Karar 2
import java.util.Scanner;
public class ornek {
    public static void main (String[] args) {
        Scanner klavye = new Scanner(System.in);
        System.out.print ("Notunuzu (1-5) giriniz: ");
        int a=klavye.nextInt();
        switch (a) {
            case 1: {
                System.out.println ("Çok zayıf");
                break;
            } case 2: {
                System.out.println ("Zayıf");
                break;
            } case 3: {
                System.out.println ("Orta");
                break;
            } case 4: {
                System.out.println ("İyi");
                break;
            } case 5: {
                System.out.println ("Çok iyi");
                break;
            } default: {
                System.out.println("Geçersiz Not");
                break;
            }
        }
    }
}
```

Bazı Matematiksel İşlem Komutları

π : `Math.PI()`

e : `Math.E()`

x^y : `Math.pow(x, y)`

\sqrt{x} : `Math.sqrt(x)`

Rastgele(x): `Math.random() * (x+1)`

Radyan \rightarrow Derece: `Math.toDegrees()`

Derece \rightarrow Radyan: `toRadians()`

e^x : `Math.exp(x)`

Üste Yuvarla: `Math.ceil()`

Aşağı Yuvarla: `Math.floor()`

Ondalıklı Kısmı At: `Math.round()`

Mutlak Değer: `Math.abs()`

Mod: `%`

En Büyük: `Math.max()`

En Küçük: `Math.min()`

Sırala: `Arrays.sort()`

$\ln(x)$: `Math.log(x)`

$\log(x)$: `Math.log10(x)`

$\sin(x)$: `Math.sin(x)`

$\cos(x)$: `Math.cos(x)`

$\tan(x)$: `Math.tan(x)`

Bazı Alfasayısal İşlem Komutları

Uzunluk: `.length()`

Büyüt: `.toUpperCase()`

Küçült: `.toLowerCase()`

Ters: `.reverse()`

Bul: `.indexOf()` veya `.contains()`

Değiştir: `.replace`

Dönüştür Sayısal Tam: `.parseInt()`

Dönüştür Sayısal Ondalıklı: `.parseFloat()`

Dönüştür Alfasayısal: `.toString()`

Sonu



Dr. Fatih KALEMKUŐ

Sorular



Dr. Fatih KALEMKUŞ

TEŐEKKÜRLER

Dr. Fatih KALEMKUŐ