

BBS515 Nesneye Yönelik Programlama

Ders 3

Zümra Kavafoğlu
<https://zumrakavafoglu.github.io/>

Bağıntısal Operatörler

operatör	tanımı
<	küçüktür
>	büyüktür
<=	küçük eşittir
>=	büyük eşittir
=	eşittir
!=	eşit değildir

Bağıntısal Operatörler

- Bağıntısal ifadelerin değeri boolean tipinde yani true ya da false'dur.

$3 < 4$ → true

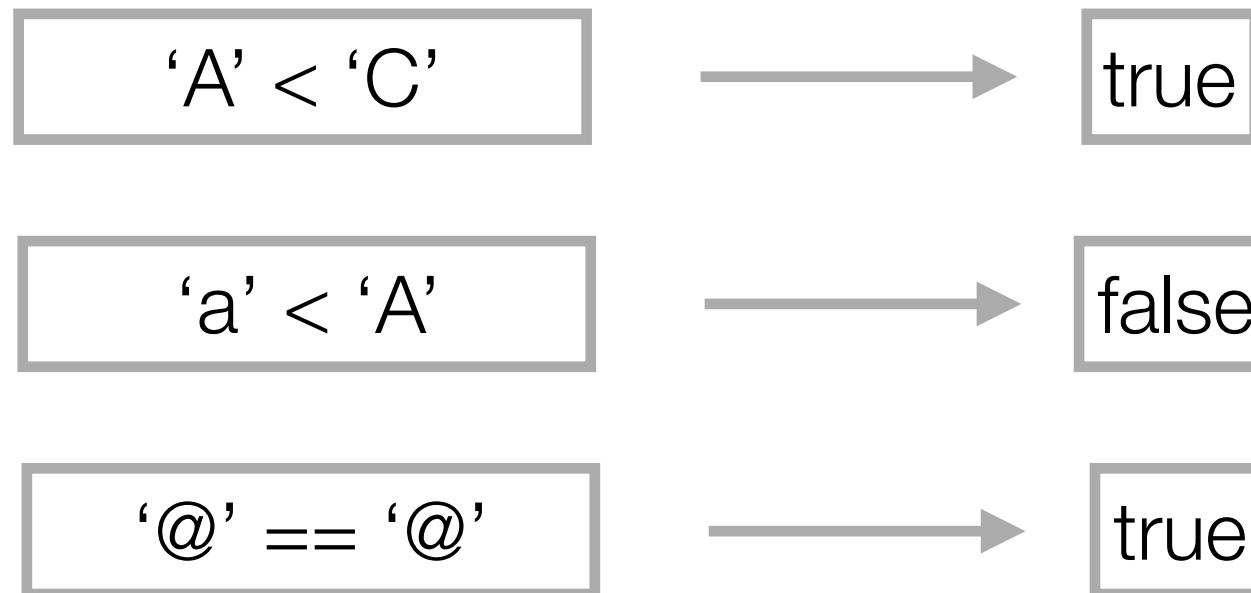
$2 > 3$ → false

$4 == 4$ → true

- double ve float için == işleci her zaman beklenen sonucu vermeyebilir

Bağıntısal Operatörler : karakter karşılaştırması

- char tipinde değişkenler de bağıntısal operatörlerle karşılaştırılabilirler. Bunun için bu değişkenlerin decimal kodları baz alınır.



Karakterlerin decimal kodları

Dec	Hx	Oct	Char	Dec	Hx	Oct	Html	Chr	Dec	Hx	Oct	Html	Chr	Dec	Hx	Oct	Html	Chr
0	0	000	NUL (null)	32	20	040	 	Space	64	40	100	@	@	96	60	140	`	`
1	1	001	SOH (start of heading)	33	21	041	!	!	65	41	101	A	A	97	61	141	a	a
2	2	002	STX (start of text)	34	22	042	"	"	66	42	102	B	B	98	62	142	b	b
3	3	003	ETX (end of text)	35	23	043	#	#	67	43	103	C	C	99	63	143	c	c
4	4	004	EOT (end of transmission)	36	24	044	$	\$	68	44	104	D	D	100	64	144	d	d
5	5	005	ENQ (enquiry)	37	25	045	%	%	69	45	105	E	E	101	65	145	e	e
6	6	006	ACK (acknowledge)	38	26	046	&	&	70	46	106	F	F	102	66	146	f	f
7	7	007	BEL (bell)	39	27	047	'	'	71	47	107	G	G	103	67	147	g	g
8	8	010	BS (backspace)	40	28	050	((72	48	110	H	H	104	68	150	h	h
9	9	011	TAB (horizontal tab)	41	29	051))	73	49	111	I	I	105	69	151	i	i
10	A	012	LF (NL line feed, new line)	42	2A	052	*	*	74	4A	112	J	J	106	6A	152	j	j
11	B	013	VT (vertical tab)	43	2B	053	+	+	75	4B	113	K	K	107	6B	153	k	k
12	C	014	FF (NP form feed, new page)	44	2C	054	,	,	76	4C	114	L	L	108	6C	154	l	l
13	D	015	CR (carriage return)	45	2D	055	-	-	77	4D	115	M	M	109	6D	155	m	m
14	E	016	SO (shift out)	46	2E	056	.	.	78	4E	116	N	N	110	6E	156	n	n
15	F	017	SI (shift in)	47	2F	057	/	/	79	4F	117	O	O	111	6F	157	o	o
16	10	020	DLE (data link escape)	48	30	060	0	0	80	50	120	P	P	112	70	160	p	p
17	11	021	DC1 (device control 1)	49	31	061	1	1	81	51	121	Q	Q	113	71	161	q	q
18	12	022	DC2 (device control 2)	50	32	062	2	2	82	52	122	R	R	114	72	162	r	r
19	13	023	DC3 (device control 3)	51	33	063	3	3	83	53	123	S	S	115	73	163	s	s
20	14	024	DC4 (device control 4)	52	34	064	4	4	84	54	124	T	T	116	74	164	t	t
21	15	025	NAK (negative acknowledge)	53	35	065	5	5	85	55	125	U	U	117	75	165	u	u
22	16	026	SYN (synchronous idle)	54	36	066	6	6	86	56	126	V	V	118	76	166	v	v
23	17	027	ETB (end of trans. block)	55	37	067	7	7	87	57	127	W	W	119	77	167	w	w
24	18	030	CAN (cancel)	56	38	070	8	8	88	58	130	X	X	120	78	170	x	x
25	19	031	EM (end of medium)	57	39	071	9	9	89	59	131	Y	Y	121	79	171	y	y
26	1A	032	SUB (substitute)	58	3A	072	:	:	90	5A	132	Z	Z	122	7A	172	z	z
27	1B	033	ESC (escape)	59	3B	073	;	;	91	5B	133	[[123	7B	173	{	{
28	1C	034	FS (file separator)	60	3C	074	<	<	92	5C	134	\	\	124	7C	174	|	
29	1D	035	GS (group separator)	61	3D	075	=	=	93	5D	135]]	125	7D	175	}	}
30	1E	036	RS (record separator)	62	3E	076	>	>	94	5E	136	^	^	126	7E	176	~	~
31	1F	037	US (unit separator)	63	3F	077	?	?	95	5F	137	_	_	127	7F	177		DEL

Mantıksal Operatörler

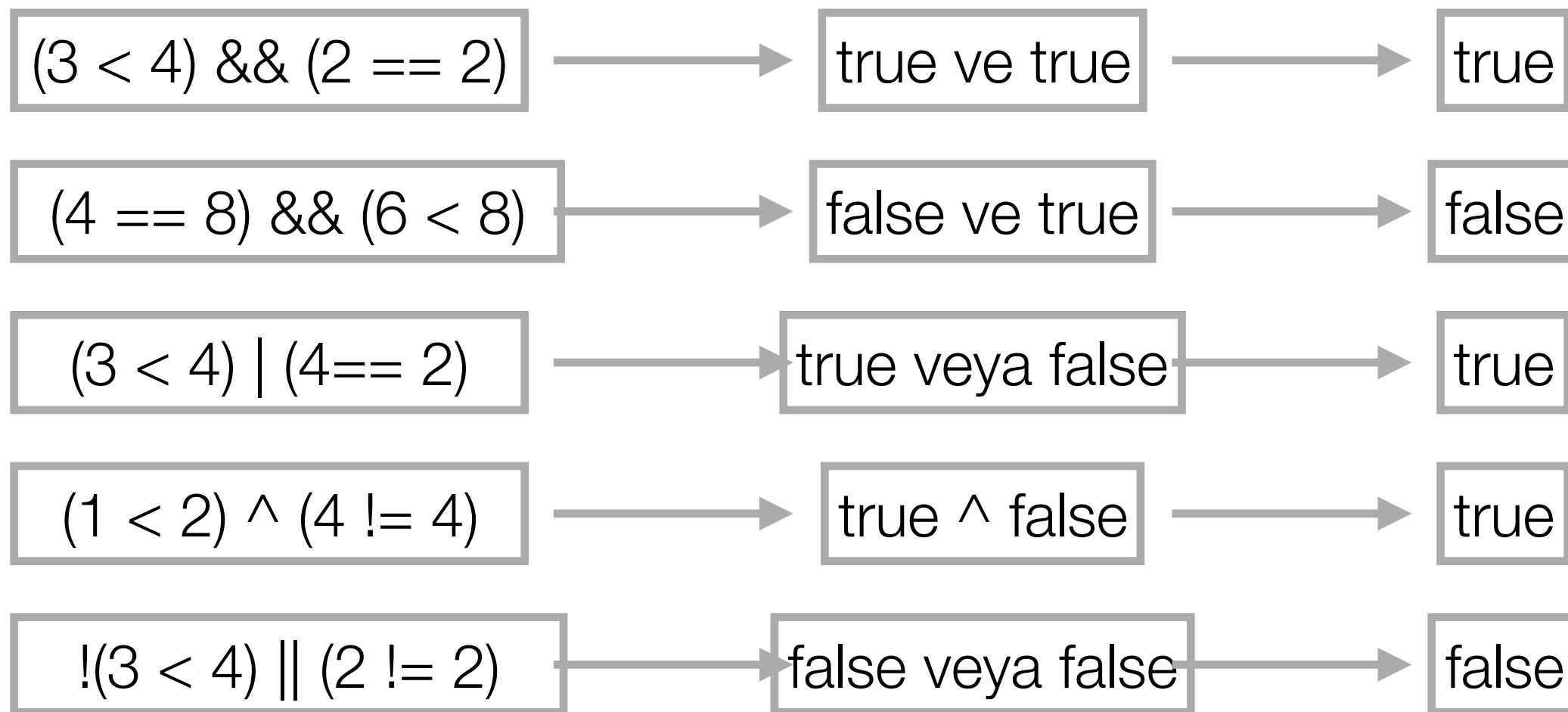
operatör	tanımı
&&	ve
&	ve
 	veya
 	veya
!	değil
^	exclusive veya

Mantıksal Operatörler için Doğruluk Tablosu

p	q	$p \& q$ ($p \& q$)	$p \mid q$ ($p \mid q$)	$\neg p$	$p \wedge q$
TRUE	TRUE	TRUE	TRUE	FALSE	FALSE
FALSE	TRUE	FALSE	TRUE	TRUE	TRUE
TRUE	FALSE	FALSE	TRUE	FALSE	TRUE
FALSE	FALSE	FALSE	FALSE	TRUE	FALSE

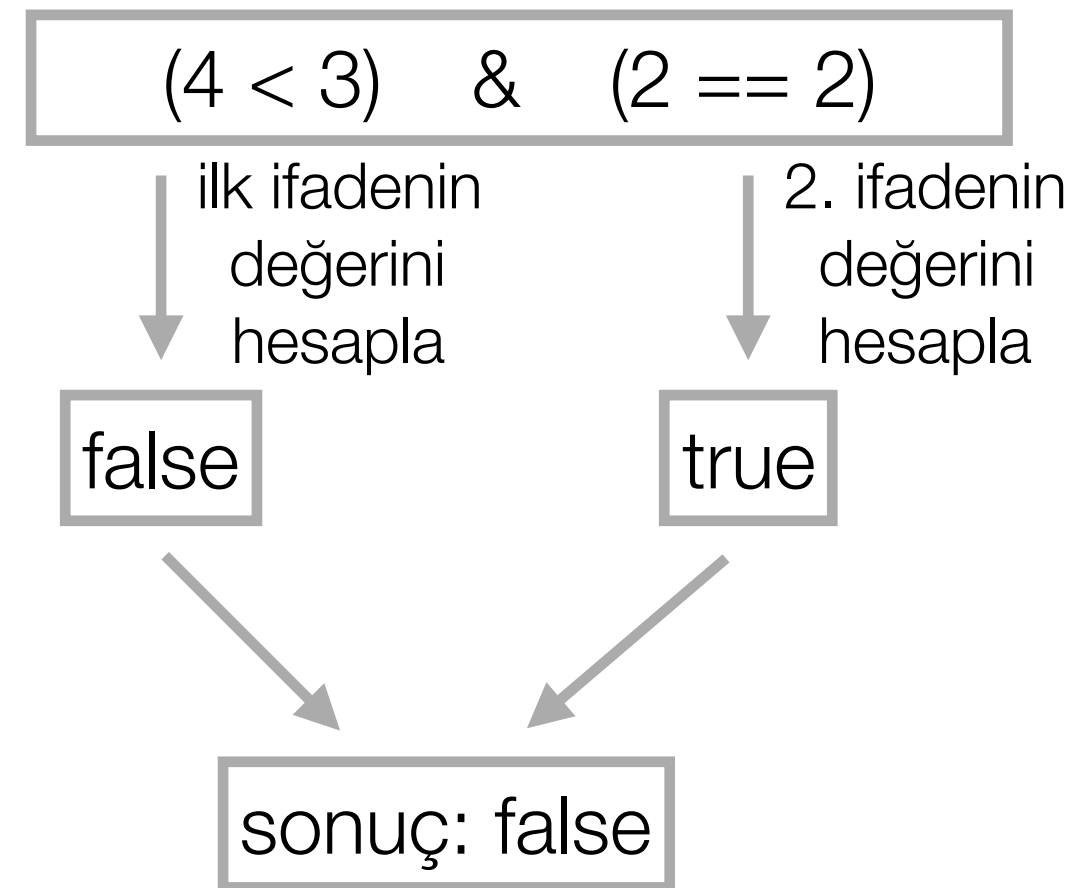
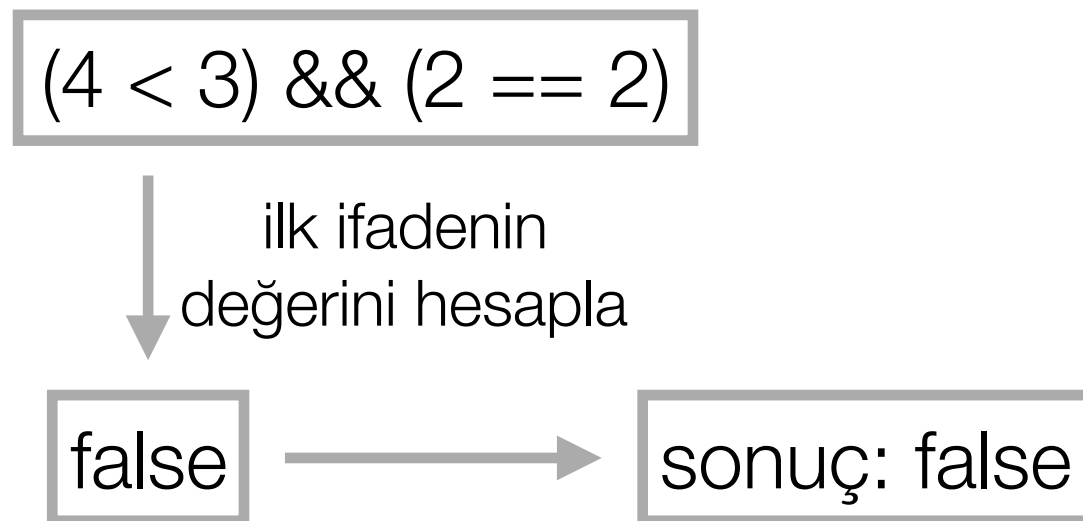
Mantıksal Operatörler

- Exclusive veya \wedge
- ifade1 \wedge ifade2
 - ifade1 ve ifade2 **aynı** değere sahipse **false**
 - ifade1 ve ifade2 **farklı** değerlere sahipse **true**



Mantıksal Operatörler: && ile & arasındaki fark

- && kullanıldığında, ilk ifade false ise ikinci ifadenin değeri hiç hesaplanmadan sonuç false olarak bulunur.
- & kullanılırsa, ilk ifade false olsa bile iki ifadenin de değeri hesaplanır.



Operatör öncelik tablosu

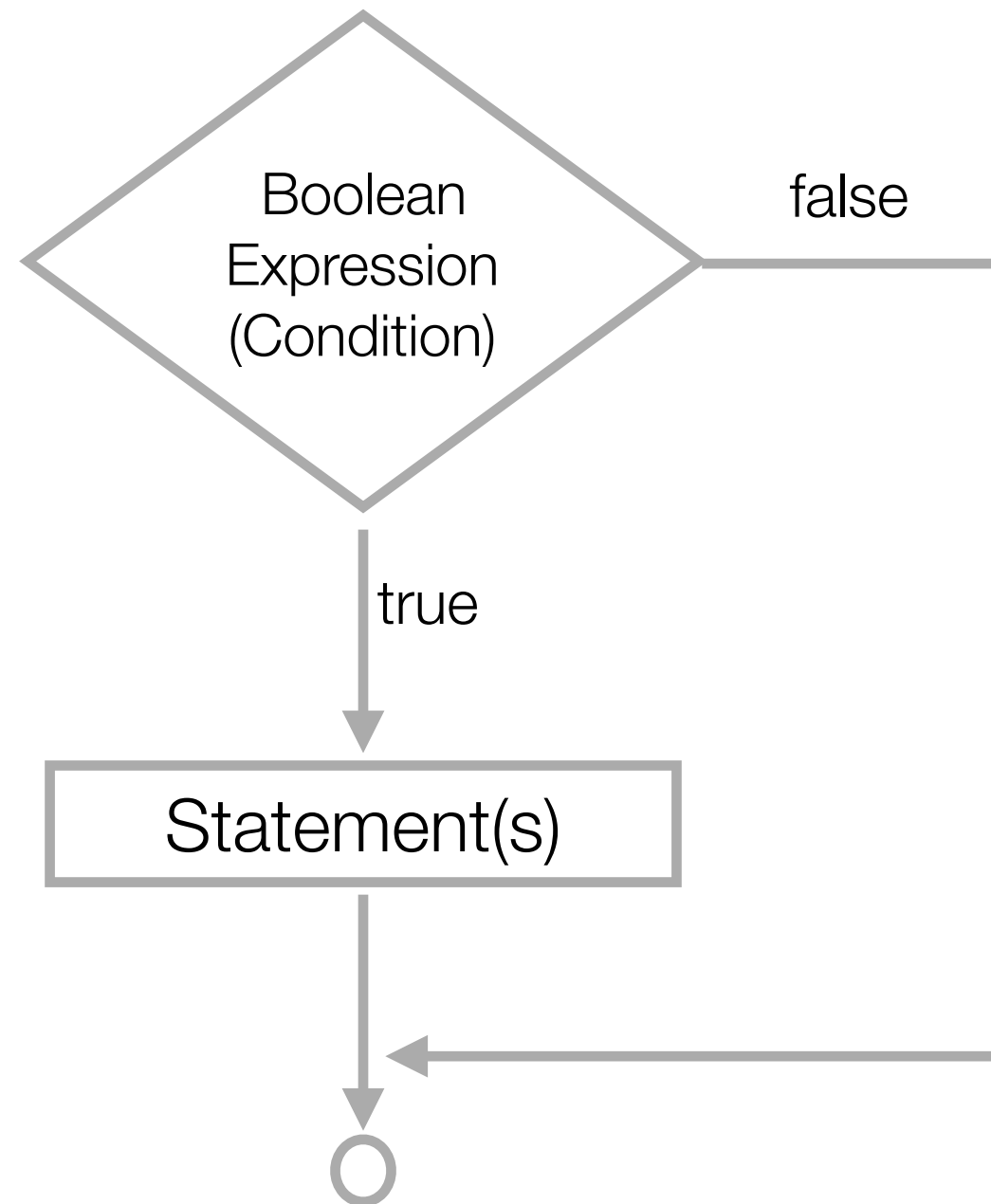
TABLE 2.10 Operator Precedence Chart	
<i>Precedence</i>	<i>Operator</i>
Highest Order	casting
	++ and -- (prefix)
	! (not)
	*, /, %
	+, -
	<, <=, >, >=
	==, !=
	& (Unconditional AND)
	^ (Exclusive OR)
	(Unconditional OR)
	&&
Lowest Order	=, +=, -=, *=, /=, %=

if koşul ifadesi

```
if(koşul) {  
    koşul doğruysa çalıştırılacak ifade;  
}
```

Burada koşul boolean değerli yani değeri true veya false olan bir ifade olmalıdır.

if koşul ifadesi



if koşul ifadesi : Örnek Program

Problem: Kullanıcıdan öğrencinin notunu isteyen ve geçme notunu geçtiyse sınavdan geçtiğini ekrana yazdıran bir program yazınız.

if koşul ifadesi : Örnek Program

```
1  import java.util.Scanner;
2
3  ▶ public class StudentPassExam {
4
5  ▶      public static void main(String[] args) {
6
7          final double passingGrade = 55;
8
9          double grade;
10
11          Scanner input = new Scanner(System.in);
12
13          System.out.println("Enter student's grade: ");
14
15          grade = input.nextDouble();
16
17          if(grade >= passingGrade) {
18              System.out.println("Student passed the exam");
19          }
20      }
21  }
```

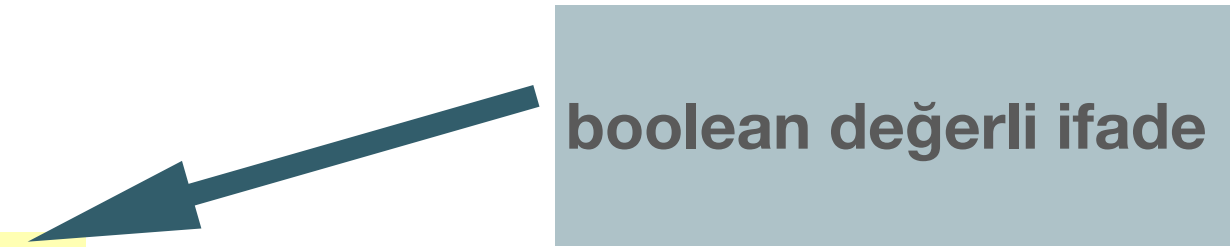
if koşul ifadesi : Örnek Program

```
1  import java.util.Scanner;
2
3  ▶ public class StudentPassExam {
4
5  ▶      public static void main(String[] args) {
6
7          final double passingGrade = 55;
8
9          double grade;
10
11          Scanner input = new Scanner(System.in);
12
13          System.out.println("Enter student grade:");
14
15          grade = input.nextDouble();
16
17          if(grade >= passingGrade) {
18              System.out.println("Student passed the exam");
19          }
20      }
21  }
```

grade değeri, passingGrade'den büyükse konsola *Student passed the exam* yazdır.

if koşul ifadesi : Yazım kuralları

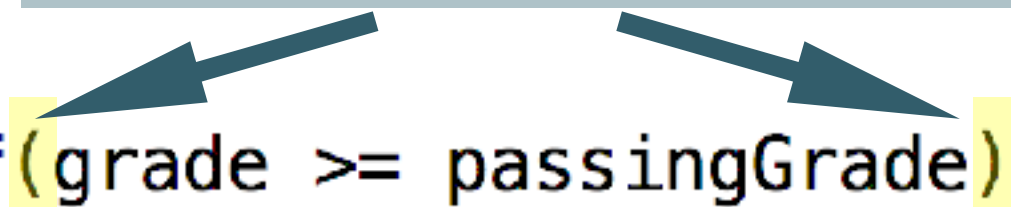
```
if(grade >= passingGrade) {  
    System.out.println("Student passed the exam");  
}
```



boolean değerli ifade

boolean değerli ifade her zaman
parantezlerin arasına yazılmalı

```
if(grade >= passingGrade) {  
    System.out.println("Student passed the exam");  
}
```



if koşul ifadesi : Parantezler

if bloğunun içindeki ifade tek satırdan oluşuyorsa süslü parantezler yazılmayabilir.

```
if(grade >= passingGrade) {  
    System.out.println("Student passed the exam");  
}
```

=

```
if(grade >= passingGrade)  
    System.out.println("Student passed the exam");
```

if koşul ifadesi : Parantezler

if bloğunun içindeki birden fazla satırdan oluşuyorsa blok süslü parantezle açılıp kapatılmalıdır.

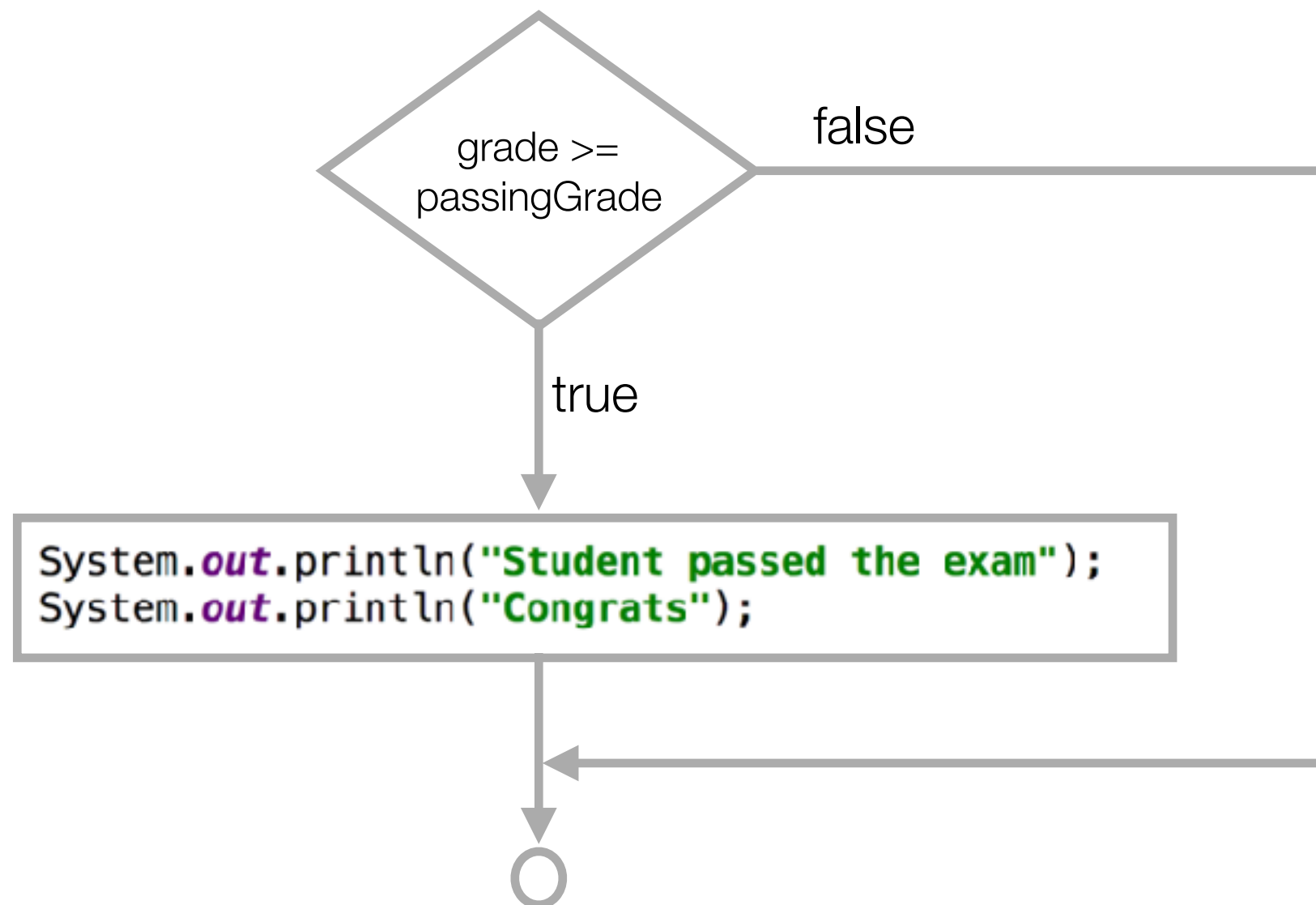
```
if(grade >= passingGrade) {  
    System.out.println("Student passed the exam");  
    System.out.println("Congrats");  
}
```



```
if(grade >= passingGrade)  
    System.out.println("Student passed the exam");  
    System.out.println("Congrats");
```

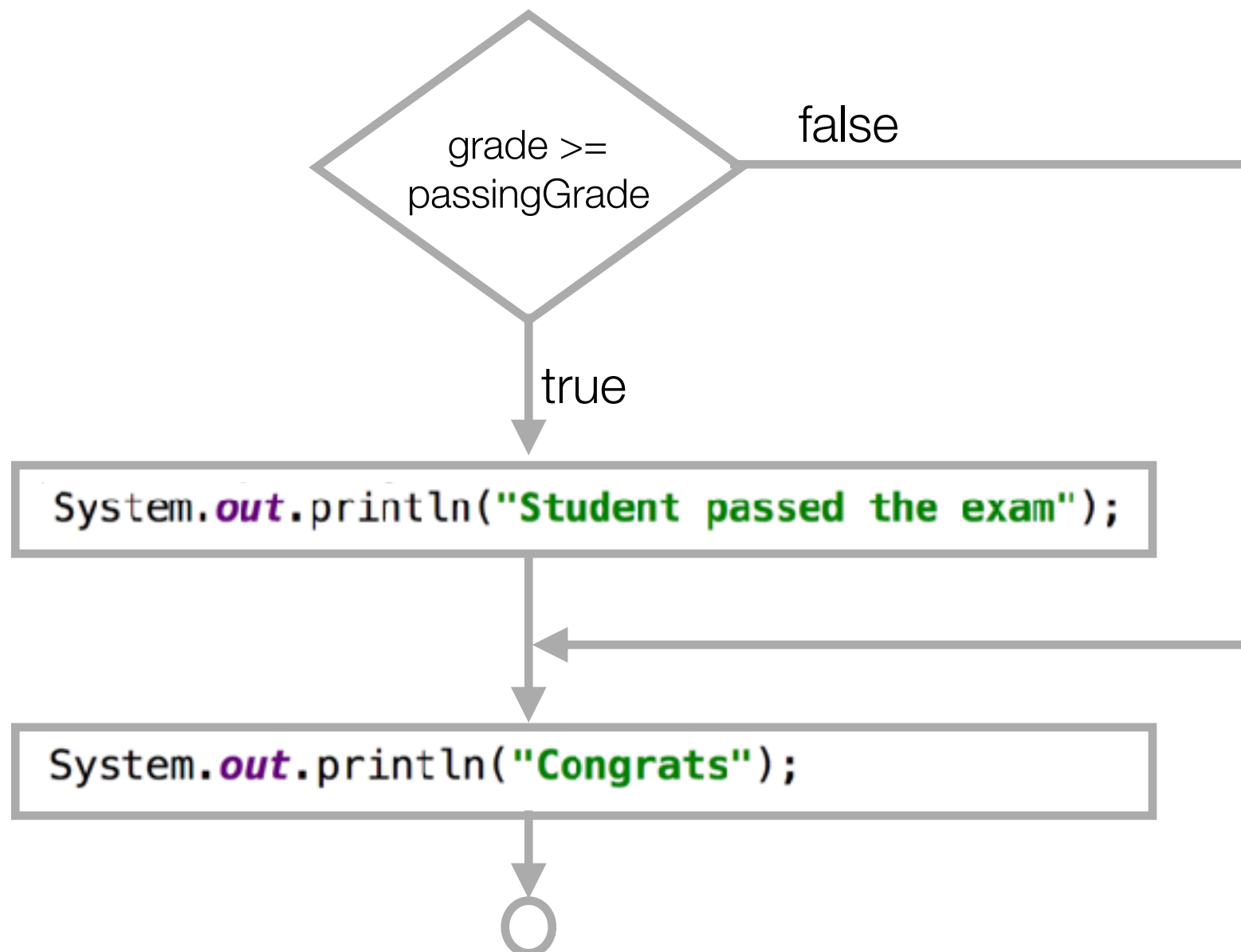
if koşul ifadesi: Parantezler

```
if(grade >= passingGrade) {  
    System.out.println("Student passed the exam");  
    System.out.println("Congrats");  
}
```



if koşul ifadesi: Parantezler

```
if(grade >= passingGrade)  
    System.out.println("Student passed the exam");  
    System.out.println("Congrats");
```



if koşul ifadesi: Parantezler

```
if(grade >= passingGrade)
    System.out.println("Student passed the exam");
    System.out.println("Congrats");
```

```
/Library/Java/JavaVirtualMachines/jdk-9.jdk/
Connected to the target VM, address: '127.0.0.1:63556'
Enter student's grade:
75
/Library/Java/JavaVirtualMachines/jdk-9.jdk/Contents/
Connected to the target VM, address: '127.0.0.1:63556'
Enter student's grade:
75
Disconnected from the target VM, address: '127.0.0.1:63556'
Student passed the exam
Congrats

Process finished with exit code 0
|
```

```
/Library/Java/JavaVirtualMachines/jdk-9.jdk/Contents/
Connected to the target VM, address: '127.0.0.1:63556'
Enter student's grade:
35
/Library/Java/JavaVirtualMachines/jdk-9.jdk/Contents/
Connected to the target VM, address: '127.0.0.1:63556'
Enter student's grade:
35
Disconnected from the target VM, address: '127.0.0.1:63556'
Congrats

Process finished with exit code 0
```

if- else koşul ifadesi

```
if(koşul) {
```

```
    koşulun değeri true ise çalıştırılacak ifade;
```

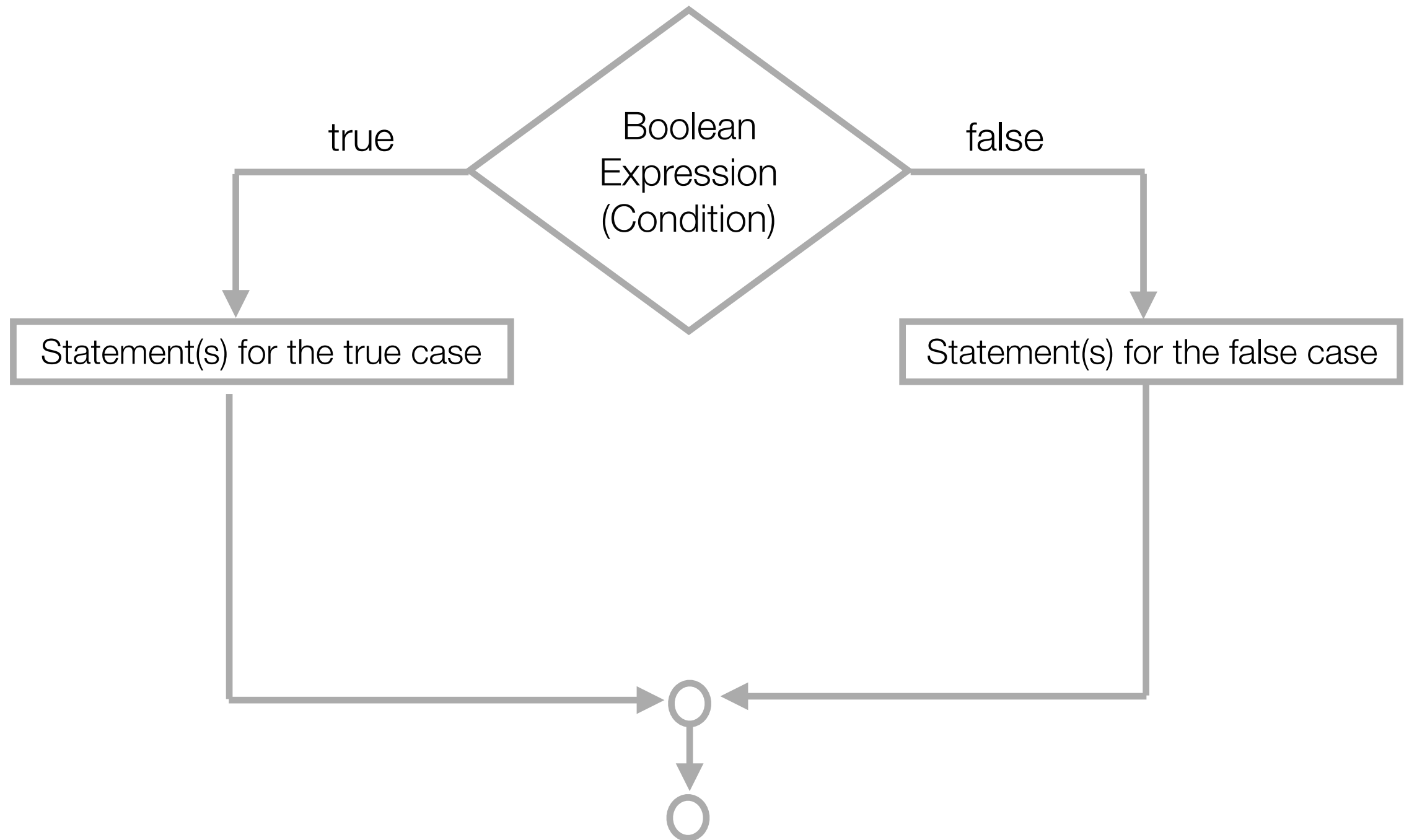
```
} else{
```

```
    koşulun değeri false ise çalıştırılacak ifade;
```

```
}
```

- Burada koşul boolean değerli yani değeri true veya false olan bir ifade olmalıdır.
- else bloğu için süslü parantezlerin kullanımı if bloğunda olduğu gibidir.

if- else koşul ifadesi



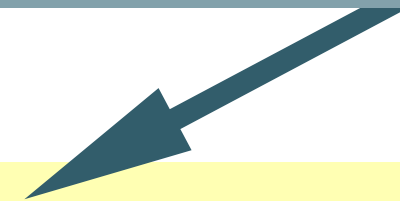
if- else koşul ifadesi

```
1  import java.util.Scanner;
2
3  ▶ public class StudentPassFailExam {
4
5  ▶      public static void main(String[] args) {
6
7          final double passingGrade = 55;
8
9          double grade;
10
11          Scanner input = new Scanner(System.in);
12
13          System.out.println("Enter student's grade: ");
14
15          grade = input.nextDouble();
16
17          if(grade >= passingGrade){
18              System.out.println("Student passed the exam");
19          }else{
20              System.out.println("Student failed the exam");
21          }
22      }
23  }
```


if- else koşul ifadesi

```
1  import java.util.Scanner;
2
3  ▶ public class StudentPassFailExam {
4
5  ▶      public static void main(String[] args) {
6
7          final double passingGrade = 55;
8
9          double grade;
10
11          Scanner input = new Scanner(System.in);
12
13          System.out.println("Enter student's grade:");
14
15          grade = input.nextDouble();
16
17          if(grade >= passingGrade){
18              System.out.println("Student passed the exam");
19          }else{
20              System.out.println("Student failed the exam");
21          }
22      }
23  }
```

grade değeri, passingGrade'den büyükse
Student passed the exam
yazdır, değilse;
Student failed the exam
yazdır.



İç içe *if-else* yapıları


- else-if yapısı çok esnek bir kullanım şekline sahiptir. Çünkü istediğimiz sayıda else-if yapısını birbirinin içine yerleştirebiliriz. Bu şekilde kullanılan else-if yapılarına içiçe else-if yapıları denir.
- İçiçe else-if yapıları çok kullanışlı olmasına rağmen bir çok hataya da neden olabilir. Bunlardan en önemlisi else komutunun yanlış if komutuyla eşleştirilmesidir.

İç içe if yapıları

Bu soruna bir örnek verelim: x ve y isimli iki değişken verilsin ve

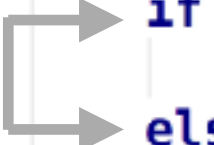
- eğer değişkenlerin ikisi de pozitifse konsola "x ve y pozitiftir"
 - eğer x negatifse konsola "x negatiftir"
- yazılsın

Doğru ifade



```
if(x>0) {  
    if (y > 0)  
        System.out.println("x and y are positive");  
}  
else  
    System.out.println("x is negative");
```

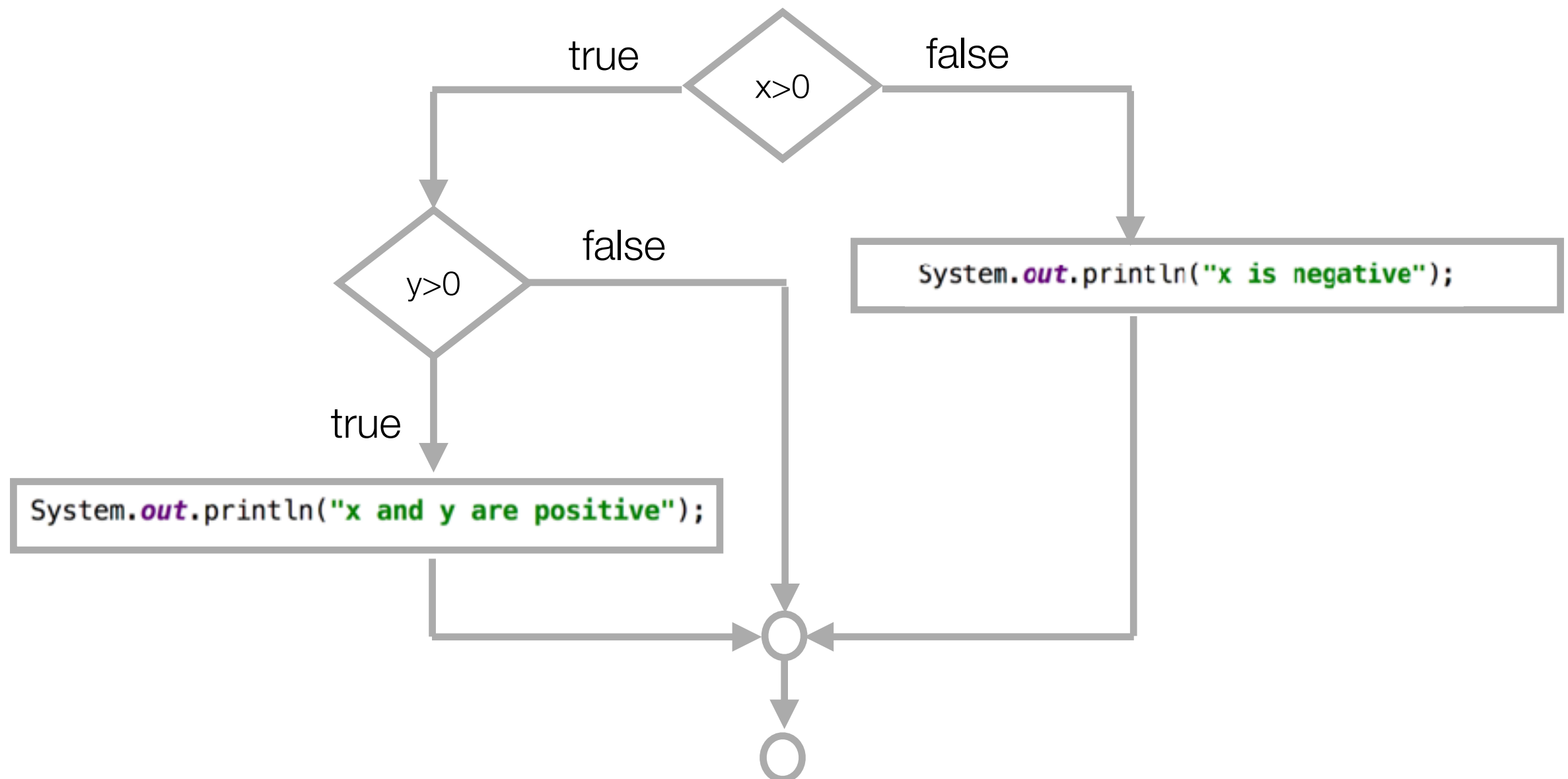
Yanlış ifade



```
if(x>0)  
    if (y > 0)  
        System.out.println("x and y are positive");  
else  
    System.out.println("x is negative");
```

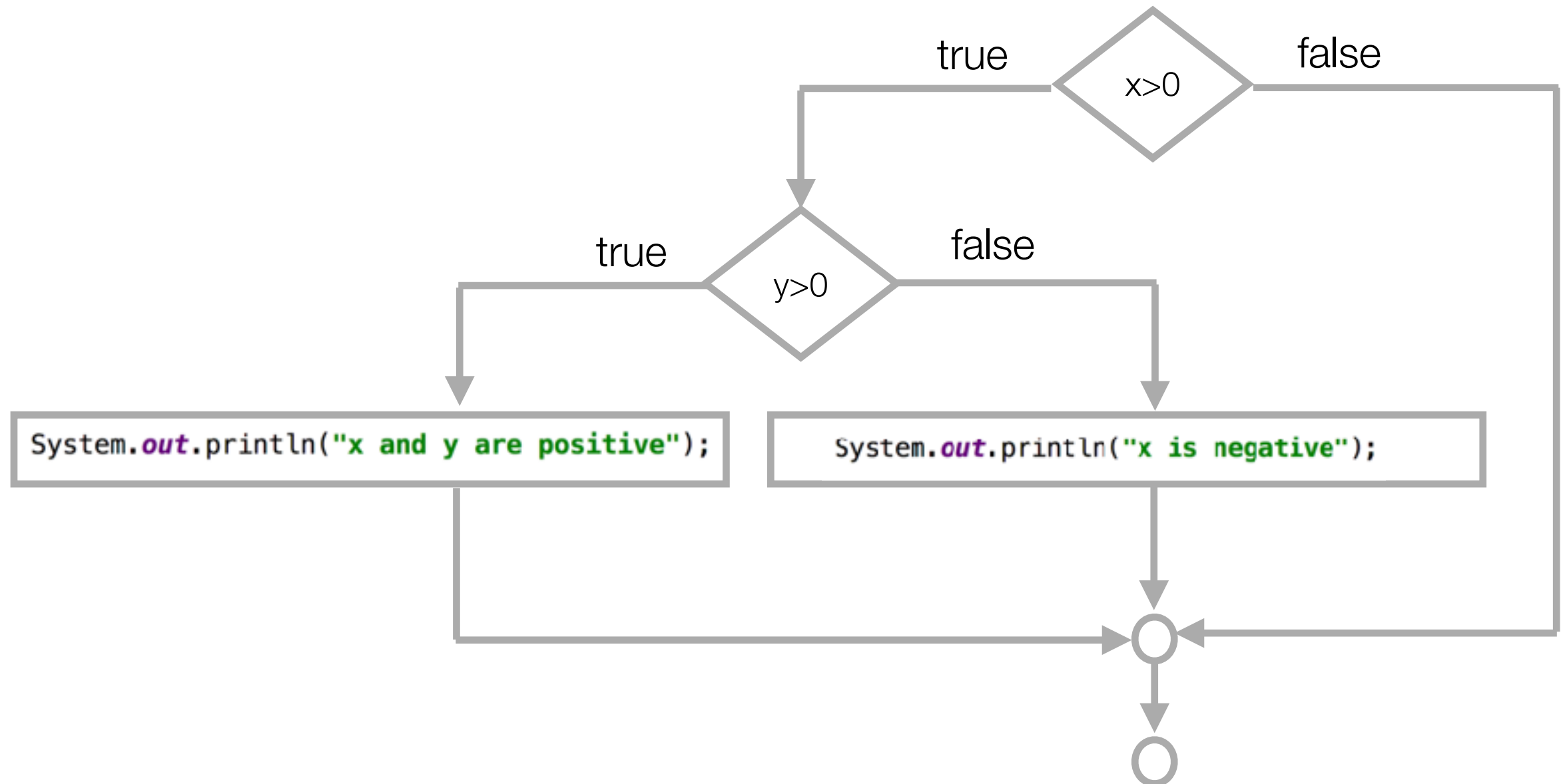
İç içe if yapıları

```
if(x>0) {  
    if (y > 0)  
        System.out.println("x and y are positive");  
}  
else  
    System.out.println("x is negative");
```



İç içe if yapıları

```
if(x>0)
    if (y > 0)
        System.out.println("x and y are positive");
    else
        System.out.println("x is negative");
```



else if komutu

```
if (testscore >= 90) {  
    grade = 'A';  
} else if (testscore >= 80) {  
    grade = 'B';  
} else if (testscore >= 70) {  
    grade = 'C';  
} else if (testscore >= 60) {  
    grade = 'D';  
} else {  
    grade = 'F';  
}
```

=

```
if (testscore >= 90) {  
    grade = 'A';  
} else {  
    if (testscore >= 80) {  
        grade = 'B';  
    } else {  
        if (testscore >= 70) {  
            grade = 'C';  
        } else {  
            if (testscore >= 60) {  
                grade = 'D';  
            } else {  
                grade = 'F';  
            }  
        }  
    }  
}
```

İç içe *if* yapıları

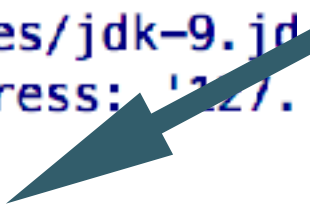
- İç içe if/else yapılarında karşılaşılan bir başka hata ise programcının test edilecek koşulların sırasını iyi ayarlayamamasıyla ortaya çıkar.
- Örneğin Celcius cinsinden girilen sıcaklık değerine göre ekranda bir mesaj yazdırmak istiyoruz.
 - `temp > 15` ise "ılık"
 - `temp > 25` ise "sıcak"

İç içe *if* yapıları

```
if(temperature > 15)
    System.out.println("It's warm!");
else if(temperature > 25)
    System.out.println("It's hot!");
```

Mantık hatası: 15'den büyük her derece için (dolayısıyla 25'den de büyük) "It's warm" mesajını yazdıracak.

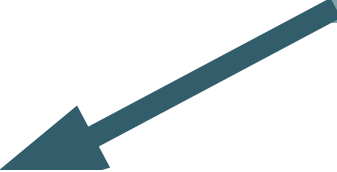
```
/Library/Java/JavaVirtualMachines/jdk-9.jd
Connected to the target VM, address: '127.
Enter current temperature:
30
Disconnected from the target VM, address:
It's warm!
```



İç içe *if* yapıları

```
if(temperature > 25)
    System.out.println("It's hot!");
else if(temperature > 15)
    System.out.println("It's warm!");
```

Doğru sıralamayla yazıldığında mantık hatası ortadan kalkar.



```
/Library/Java/JavaVirtualMachines/jdk-9.jdk/Contents/Home/bin/java
Connected to the target VM, address: '127.0.0.1:64058', transport:
Enter current temperature:
30
Disconnected from the target VM, address: '127.0.0.1:64058', transp
It's hot!
```

```
Process finished with exit code 0
```

Koşullu operatör (Conditional Operator)

Koşullu operatör (?:) esas olarak sıkıştırılmış bir if/else yapısıdır. Koşullu operatör üçlü bir operatördür , yani koşullu bir ifade yaratmak için üç tane argüman alır.

```
if(grade >= 70){  
    System.out.println("Passed");  
}else{  
    System.out.println("Failed");  
}
```

=

```
System.out.println(grade >= 70 ? "Passed" : "Failed");
```

Koşullu operatör (Conditional Operator)

```
if(x>2){  
    y = 5;  
}else{  
    y = 8;  
}
```

=

```
y = (x > 2 ? 5 : 8);
```

switch komutu


- Kullanıcıdan haftanın kaçınıcı gününde olduğumuzu girmesin isteyen ve o günün adını yazan bir program yazınız.

```
1  import java.util.Scanner;
2
3  public class SwitchDays {
4
5      public static void main(String[] args) {
6
7          int dayNumber;
8          String dayName = "";
9
10         System.out.print("Enter the day number: ");
11         Scanner input = new Scanner(System.in);
12         dayNumber = input.nextInt();
13
14         if(dayNumber == 1)
15             dayName = "Monday";
16         else if(dayNumber == 2)
17             dayName = "Tuesday";
18         else if(dayNumber == 3)
19             dayName = "Wednesday";
20         else if(dayNumber == 4)
21             dayName = "Thursday";
22         else if(dayNumber == 5)
23             dayName = "Friday";
24         else if(dayNumber == 6)
25             dayName = "Saturday";
26         else if(dayNumber == 7)
27             dayName = "Sunday";
28         else
29             System.out.print("You entered wrong number!");
30
31         System.out.print(dayName);
32     }
33 }
34
```

switch komutu

- Kullanıcıdan haftanın kaçınıcı gününde olduğumuzu girmesini isteyen ve o günün adını yazan bir program yazınız.

```
1  import java.util.Scanner;
2
3  public class SwitchDays {
4
5      public static void main(String[] args) {
6
7          int dayNumber;
8          String dayName = "";
9
10         System.out.print("Enter the day number: ");
11         Scanner input = new Scanner(System.in);
12         dayNumber = input.nextInt();
13
14         if(dayNumber == 1)
15             dayName = "Monday";
16         else if(dayNumber == 2)
17             dayName = "Tuesday";
18         else if(dayNumber == 3)
19             dayName = "Wednesday";
20         else if(dayNumber == 4)
21             dayName = "Thursday";
22         else if(dayNumber == 5)
23             dayName = "Friday";
24         else if(dayNumber == 6)
25             dayName = "Saturday";
26         else if(dayNumber == 7)
27             dayName = "Sunday";
28         else
29             System.out.print("You entered wrong number!");
30
31         System.out.print(dayName);
32     }
33 }
34
```



```
switch (dayNumber) {
    case 1:
        dayName = "Monday";
        break;
    case 2:
        dayName = "Tuesday";
        break;
    case 3:
        dayName = "Wednesday";
        break;
    case 4:
        dayName = "Thursday";
        break;
    case 5:
        dayName = "Friday";
        break;
    case 6:
        dayName = "Saturday";
        break;
    case 7:
        dayName = "Sunday";
        break;
    default:
        System.out.print("You entered wrong number");
        break;
}
```

switch komutu

```
switch(switch değişkeni){  
    case değer1:  
        ifade(ler)1;  
        break;  
    case değer2:  
        ifade(ler)2;  
        break;  
    .  
    .  
    .  
    case değerN:  
        ifade(ler)N;  
        break;  
    default: default durum için ifade(ler);  
}
```

switch komutu

```
switch (dayNumber) {  
    case 1:   
        dayName = "Monday";  
        break;  
    case 2:   
        dayName = "Tuesday";  
        break;  
    case 3:   
        dayName = "Wednesday";  
        break;  
    case 4:   
        dayName = "Thursday";  
        break;  
    case 5:   
        dayName = "Friday";  
        break;  
    case 6:   
        dayName = "Saturday";  
        break;  
    case 7:   
        dayName = "Sunday";  
        break;  
    default:   
        System.out.print("You entered wrong number");  
        break;  
}
```

switch değişkeni

değer1

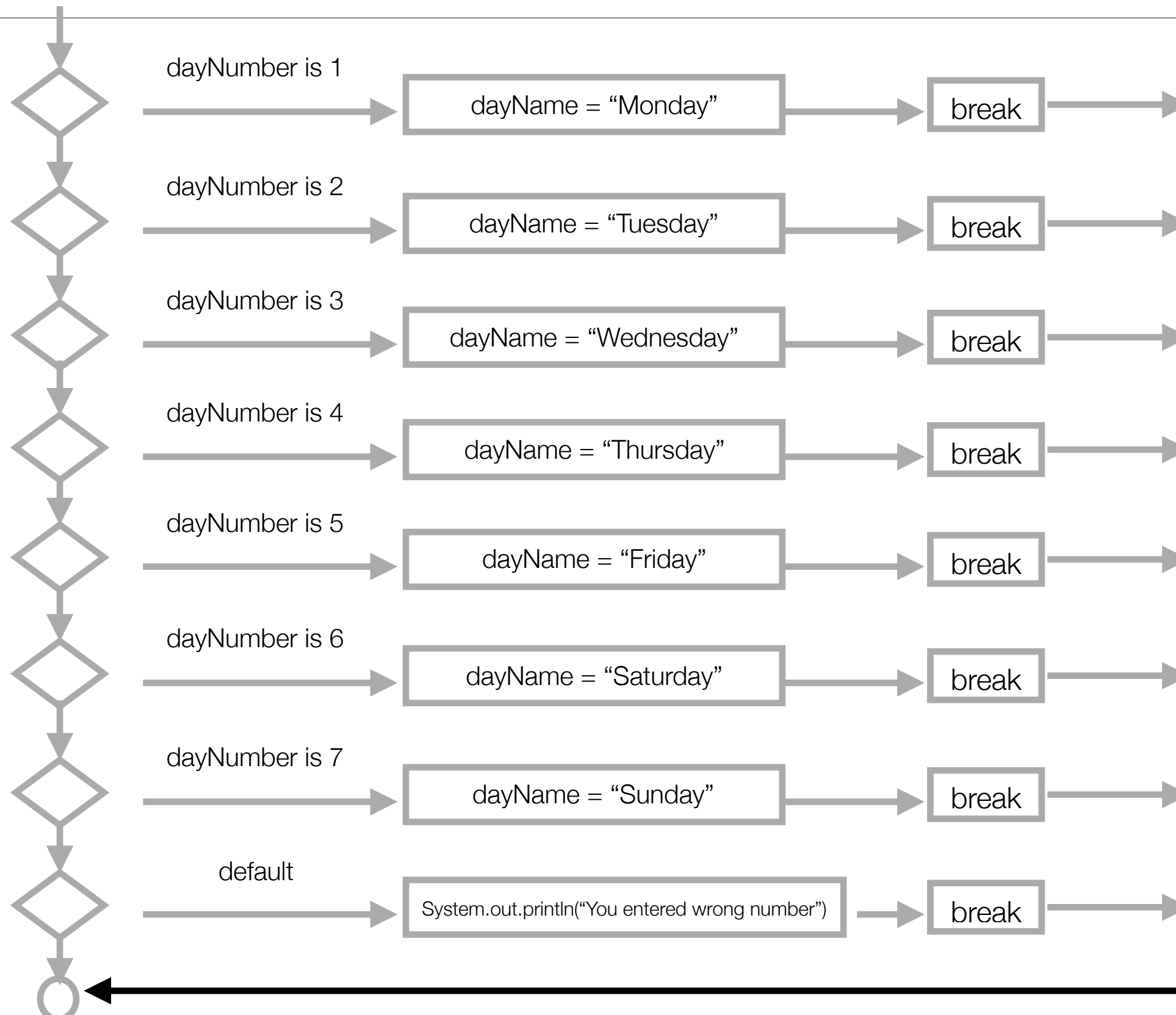
ifade1

değer2

ifade2

default ifade

switch komutu



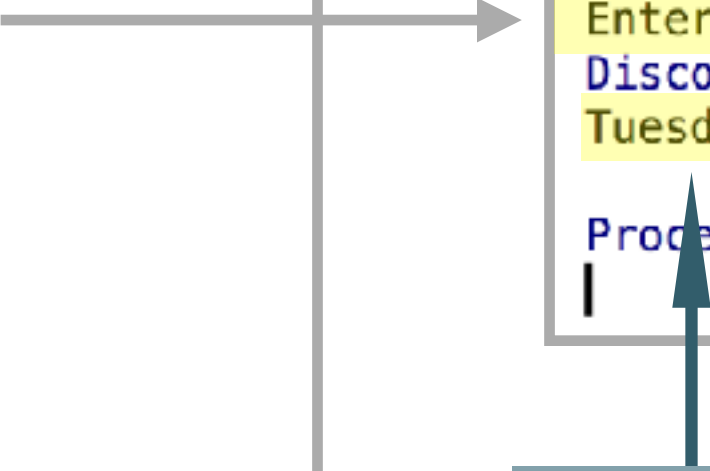
switch komutu

- **switch** değişkeni char, byte, short, int ya da String tipinde olmalı ve her zaman parantez içinde olmalı
- değer1 ... değerN switch değişkeni ile aynı tipte olmalıdır.
- **break** ifadesi bütün switch bloğunu sonlandırır. Eğer kullanılmazsa durumun ne olduğuna bakılmaksızın sonraki ifadeler çalıştırılır.
- **default** durumu belirlenen durumlardan hiçbirinin doğru olmadığı durumlar için kullanılır. **default** her zaman switch bloğunun en sonunda yer alır.

switch komutu: break

Bir durumun sonunda break ifadesi kullanılmazsa synthax hatası oluşmaz ama önemli bir mantık hatası oluşabilir.

```
switch (dayNumber) {  
    case 1:  
        System.out.println("Monday");  
        break;  
    case 2:  
        System.out.println("Tuesday");  
        break;  
    case 3:  
        System.out.println("Wednesday");  
        break;  
    case 4:  
        System.out.println("Thursday");  
        break;  
    case 5:  
        System.out.println("Friday");  
        break;  
    case 6:  
        System.out.println("Saturday");  
        break;  
    case 7:  
        System.out.println("Sunday");  
        break;  
    default:  
        System.out.println("You entered wrong number");  
        break;  
}
```



```
/Library/Java/JavaVirtualMachines/jdk-9.  
Connected to the target VM, address: '12  
Enter the day number: 2  
Disconnected from the target VM, address  
Tuesday  
  
Process finished with exit code 0  
|
```

doğru sonuç

switch komutu: break

Bir durumun sonunda break ifadesi kullanılmazsa synthax hatası oluşmaz ama önemli bir mantık hatası oluşabilir.

```
switch (dayNumber) {  
    case 1:  
        System.out.println("Monday");  
        break;  
    case 2:  
        System.out.println("Tuesday");  
    case 3:  
        System.out.println("Wednesday");  
        break;  
    case 4:  
        System.out.println("Thursday");  
        break;  
    case 5:  
        System.out.println("Friday");  
        break;  
    case 6:  
        System.out.println("Saturday");  
        break;  
    case 7:  
        System.out.println("Sunday");  
        break;  
    default:  
        System.out.println("You entered wrong number");  
        break;  
}
```

İfadeden sonra break yok

```
/Library/Java/JavaVirtualMachines/  
Connected to the target VM, address  
Enter the day number: 2  
Disconnected from the target VM, address  
Tuesday  
Wednesday  
Process finished with exit code 0
```

hatalı sonuç

while döngüsü

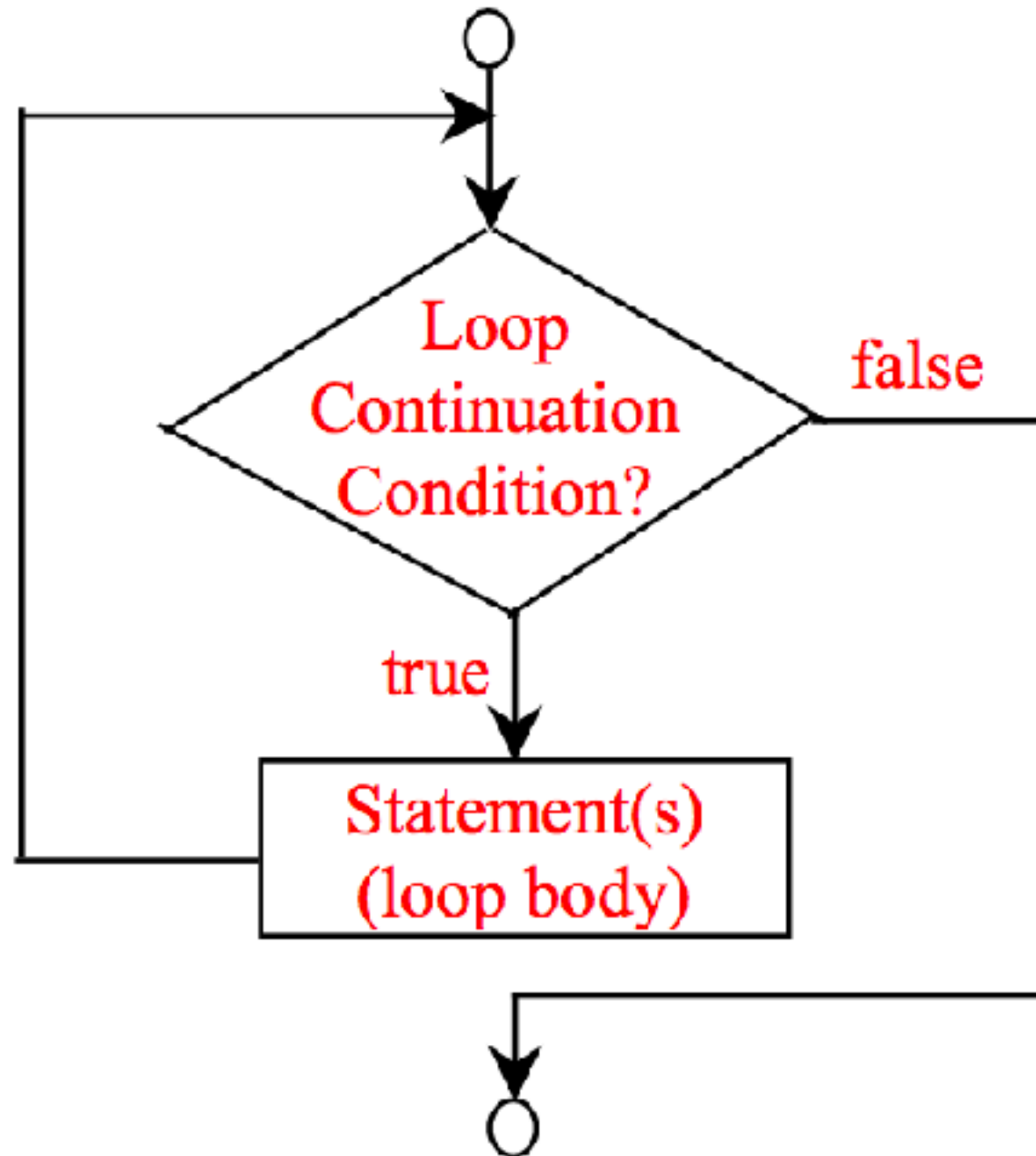
```
while(koşul){  
    döngü ifadeleri  
}
```

Koşul boolean değerli olmalıdır.

Koşulun değeri true olduğu sürece döngü ifadeleri tekrar tekrar çalıştırılır.

Sonsuz döngüyü engellemek için koşulun eninde sonunda false olacağından emin olun.

while döngüsü



while döngüsü

```
1 ▶ public class WhileDemo {  
2 ▶     public static void main(String[] args){  
3         int count = 1;  
4         while (count < 11) {  
5             System.out.println("Count is: " + count);  
6             count++;  
7         }  
8     }  
9 }  
10
```

```
/Library/Java/JavaVirtualMachines/jdk-9.jdk  
Connected to the target VM, address: '127.0.0.1:5055'  
Disconnected from the target VM, address: '127.0.0.1:5055'
```

```
Count is: 1  
Count is: 2  
Count is: 3  
Count is: 4  
Count is: 5  
Count is: 6  
Count is: 7  
Count is: 8  
Count is: 9  
Count is: 10
```

```
Process finished with exit code 0
```

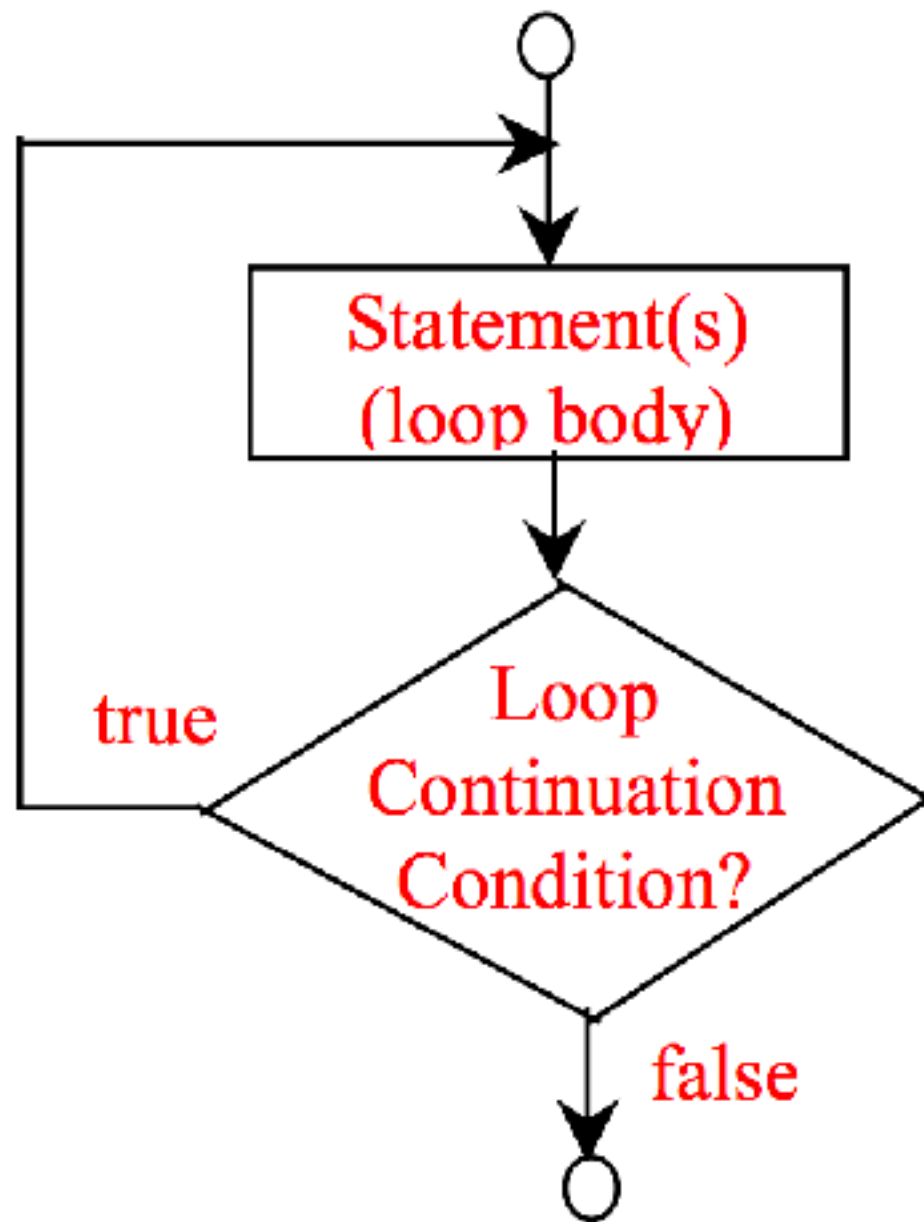
do-while döngüsü

```
do{  
    döngü ifadeleri  
}while(koşul)
```

while döngüsünden farkı:

- while döngüsünde önce koşulun değerine bakılır, true ise döngü ifadeleri çalıştırılır.
- do-while döngüsünde önce döngü ifadeleri çalıştırılır, sonra koşulun değerine bakılır, true ise bir sonraki döngü ifadesi çalıştırılır. **Yani do-while döngüsünde döngü ifadeleri en az bir kez çalıştırılır.**

do-while döngüsü

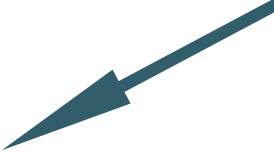


do-while döngüsü

Kullanıcıdan negatif bir tamsayı girene kadar sürekli tamsayı girmesini istediğimiz bir programı hem while hem de do-while döngüsü kullanarak yazalım:

while döngüsü ile:

```
1  import java.util.Scanner;
2
3  public class DoWhileDemo {
4
5      public static void main(String[] args) {
6
7          int number;
8
9          Scanner input = new Scanner(System.in);
10
11          System.out.print("Input a positive number to continue, a negative number to stop: ");
12          number = input.nextInt();
13
14          while (number >= 0){
15              System.out.print("Input a positive number to continue, a negative number to stop: ");
16              number =input.nextInt();
17          }
18      }
19  }
20 }
```



ilk sayıyı döngünün dışında alıyoruz

do-while döngüsü

Kullanıcıdan negatif bir tamsayı girene kadar sürekli tamsayı girmesini istediğimiz bir programı hem while hem de do-while döngüsü kullanarak yazalım:

do-while döngüsü ile:

```
1  import java.util.Scanner;
2
3  public class DoWhileDemo {
4
5      public static void main(String[] args) {
6
7          int number;
8
9          Scanner input = new Scanner(System.in);
10
11         do {
12             System.out.print("Input a positive number to continue, a negative number to stop: ");
13             number = input.nextInt();
14         } while(number > 0);
15     }
16 }
```

Döngünün içindeki ifade ilk sefer mutlaka çalıştırılacağı için döngünün dışında bu ifadeleri tekrar yazmaya ihtiyaç yok.

for döngüsü

```
for(initialization; termination; adjustment){  
    döngü ifadeleri  
}
```

- **initialization(ilk değer verme)**: kontrol değişkenine ilk değer verilir. Sadece bir defa döngünün başlangıcında çalıştırılır.
- **termination(sonlandırma)**: değeri false olduğunda döngü sonlanır.
- **adjustment(ayarlama)**: kontrol değişkeninin değerini değiştirir.

for döngüsü

Konsola alt alta yüz defa Welcome to Java yazdırmak için:

kontrol değişkeni

```
for(int i=0; i<100; i++){  
    System.out.println("Welcome to Java");  
}
```

for döngüsü

Konsola alt alta yüz defa Welcome to Java yazdırmak için:

i kontrol değişkenine 0
ilk değeri verilir.

```
for(int i=0; i<100; i++){  
    System.out.println("Welcome to Java");  
}
```

for döngüsü

Konsola alt alta yüz defa Welcome to Java yazdırmak için:

(i<100) ifadesi false
olduğunda yani (i>=100)
olduğunda döngü sonlanır

```
for(int i=0; i<100; i++){  
    System.out.println("Welcome to Java");  
}
```

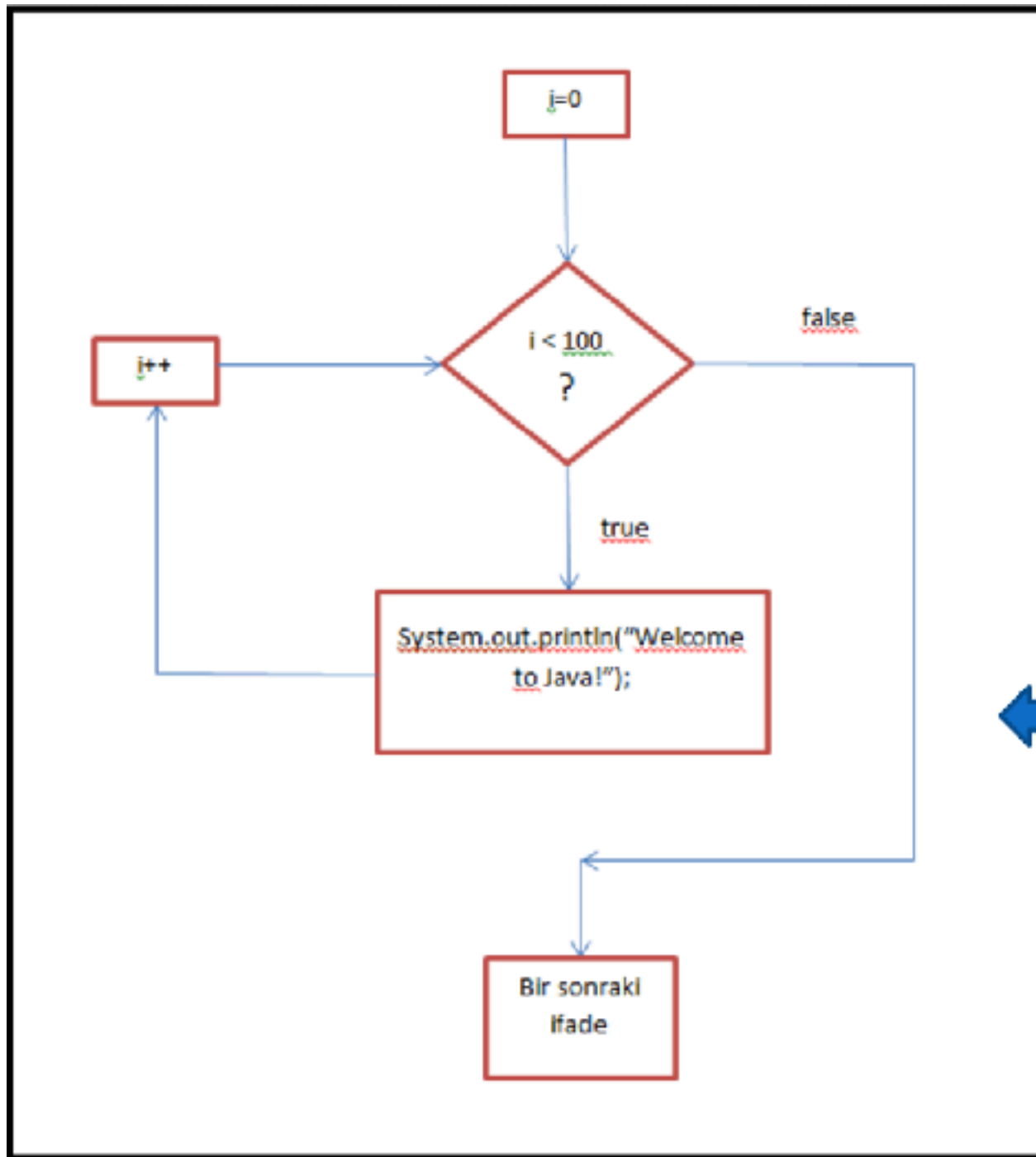
for döngüsü

Konsola alt alta yüz defa Welcome to Java yazdırmak için:

i değeri döngünün her adımında 1 arttırılır

```
for(int i=0; i<100; i++){  
    System.out.println("Welcome to Java");  
}
```

for döngüsü



```
for(int i=0; i<100; i++){  
    System.out.println("Welcome to Java");  
}
```


for, while, do-while

Konsola alt alta yüz defa *Welcome to Java* yazdırmak için:

```
for(int i=0; i<100; i++){  
    System.out.println("Welcome to Java");  
}
```

=

```
int i = 0;  
  
while(i<100){  
    System.out.println("Welcome to Java");  
    i++;  
}
```

=

```
int i = 0;  
  
do{  
    System.out.println("Welcome to Java");  
    i++;  
}while(i < 100);
```

break ve continue

break: Bu komut onu kapsayan en içteki döngüyü hemen sonlandırır.

continue: Bu komut onu kapsayan en içteki döngünün o adımını hemen sonlandırır, döngü bir sonraki adımdan devam eder.

break

```
1
2 ▶ public class TestBreak {
3
4 ▶     public static void main(String[] args) {
5         int sum = 0;
6         int item = 0;
7
8         while(item < 5)
9         {
10             item++;
11             System.out.println("Item is now: "+item);
12             sum += item;
13             System.out.println("Sum is now: "+sum);
14             if(sum >= 6)
15                 break;
16         }
17
18         System.out.println("The last sum is: "+sum);
19     }
20 }
```

sum 6'dan büyük eşit olduğunda döngüden çık.

```
/Library/Java/JavaVirtualMachines/jdk-
Connected to the target VM, address: '
Item is now: 1
Sum is now: 1
Item is now: 2
Sum is now: 3
Item is now: 3
Sum is now: 6
The last sum is: 6
Disconnected from the target VM, address:
Process finished with exit code 0
```

break komutu olmadan aynı döngü

```
1
2 ▶ public class TestBreak {
3
4 ▶     public static void main(String[] args) {
5         int sum = 0;
6         int item = 0;
7
8         while(item < 5)
9         {
10             item++;
11             System.out.println("Item is now: "+item);
12             sum += item;
13             System.out.println("Sum is now: "+sum);
14         }
15
16         System.out.println("The last sum is: "+sum);
17     }
18 }
```

```
/Library/Java/JavaVirt
Connected to the target
Item is now: 1
Sum is now: 1
Item is now: 2
Sum is now: 3
Item is now: 3
Sum is now: 6
Item is now: 4
Sum is now: 10
Item is now: 5
Sum is now: 15
The last sum is: 15
```

continue

```
1
2 ▶ public class TestContinue {
3 ▶     public static void main(String[] args) {
4         for(int i=0; i<8; i++)
5         {
6             if(i==2)
7                 continue;
8             System.out.println("i is "+ i);
9         }
10    }
11
12 }
```

i 2'ye eşit olduğunda gövdedeki takip eden komutlar(bu örnekte print) atlanır ve hemen döngüdeki bir sonraki adıma geçilir

```
/Library Java/JavaVi
i is 0
i is 1
i is 3
i is 4
i is 5
i is 6
i is 7

Process finished with
```

continue olmadan aynı döngü

```
1
2 ▶ public class TestContinue {
3 ▶     public static void main(String[] args) {
4         for(int i=0; i<8; i++)
5         {
6             System.out.println("i is "+ i);
7         }
8     }
9 }
```

/Library/Java/

i is 0
i is 1
i is 2
i is 3
i is 4
i is 5
i is 6
i is 7

Process finish

İç içe döngülerde break komutu

```
1 ▶ public class BreakNestedLoops {  
2 ▶     public static void main(String args[]){  
3  
4         for(int i=0; i<3; i++){  
5             System.out.println("\nouter value: " + i);  
6             for(int j=0; j<7; j++){  
7                 if(j==5)  
8                     break;  
9                 System.out.println("inner value: " + j);  
10            }  
11        }  
12    }  
13 }
```

ait olduğu en içteki
döngüden çıkış sağlar

/Library/Java/JavaVi

outer value: 0
inner value: 0
inner value: 1
inner value: 2
inner value: 3
inner value: 4

outer value: 1
inner value: 0
inner value: 1
inner value: 2
inner value: 3
inner value: 4

outer value: 2
inner value: 0
inner value: 1
inner value: 2
inner value: 3
inner value: 4

Çalışma zamanı hatası(Run-time error)

- Derleyicinin algılayamadığı ama programın çalışması sırasında ortaya çıkan hatalardır.
- Bazı örnekler:
- **InputMismatchException:** Scanner ile kullanıcıdan istenen verinin tipinde uyumsuzluk olduğunda ortaya çıkar.

```
1  import java.util.Scanner;
2
3  ▶ public class RunTimeErrorDemo {
4  ▶      public static void main(String[] args) {
5
6          int inputValue;
7
8          Scanner input = new Scanner(System.in);
9
10         System.out.print("Enter an integer: ");
11
12         inputValue = input.nextInt();
13     }
14 }
```

```
/Library/Java/JavaVirtualMachines/jdk-9.jdk/Contents/Home/bin/j
Connected to the target VM, address: '127.0.0.1:55277', transpo
Enter an integer: 3.7
Exception in thread "main" java.util.InputMismatchException
    at java.base/java.util.Scanner.throwFor(Scanner.java:860)
    at java.base/java.util.Scanner.next(Scanner.java:1497)
    at java.base/java.util.Scanner.nextInt(Scanner.java:2161)
    at java.base/java.util.Scanner.nextInt(Scanner.java:2115)
    at RunTimeErrorDemo.main(RunTimeErrorDemo.java:12)
Disconnected from the target VM, address: '127.0.0.1:55277', tr
Process finished with exit code 1
|
```


Çalışma zamanı hatası(Run-time error)

- **ArithmeticException:** Bir sayının 0 ile bölümünde ortaya çıkar.

```
1
2 ▶ public class ArithmeticExceptionDemo {
3 ▶     public static void main(String[] args) {
4
5         int x = 3/0;
6
7         System.out.println(x);
8     }
9 }
```

```
/Library/Java/JavaVirtualMachines/jdk-9.jdk/Contents/Home/bin/java -a
Connected to the target VM, address: '127.0.0.1:57477', transport: 's
Disconnected from the target VM, address: '127.0.0.1:57477', transpo
Exception in thread "main" java.lang.ArithmeticException: / by zero
    at ArithmeticExceptionDemo.main(ArithmeticExceptionDemo.java:5)
```

```
Process finished with exit code 1
```

Mantık hatası

- Mantık hataları programa yaptırmak istediğiniz bir görevin yanlış yapılması sonucu ortaya çıkar. Derleme zamanında veya çalışma zamanında herhangi bir hata verilmemesine rağmen programın çıktısı istediğiniz çıktı değildir. Bu istemsiz durumun kodun hangi bölümünden kaynaklandığını, yani mantık hatasının yerini bulmak özellikle geniş kapsamlı programlarda çok zor olabilir.
- Belli başlı mantık hatalarına örnekler:
 - Operatör önceliklerinde hata
 $5+4*3$ ile $(5+4)*3$ farklı sonuçlar verir
 - Bir koşulun yanlış olduğu halde doğru olduğunu varsaymak
 - Kayar noktalı sayılarla(floating point numbers, double / float) eşitlik kontrolü yapmak
 - İki tamsayı tipinde değişkenin bölümünün ondalık sayı çıkacağını varsaymak
 - Noktalı virgülü yanlış yere koymak

float ve double tipinde değişkenlerin eşitliğinin karşılaştırılması

- floating-point değerleriyle aritmetik işlemler yapılırken bazı küçük yuvarlamalar sonucu teorik olarak eşit olan iki floating-point değişkeni çok küçük bir değer farkı nedeniyle eşit değilmiş gibi anlaşılabilir. Bu yanlış anlaşılma sonucu doğru olması gereken bir koşul yanlış olarak alınır ve programda ayıklanması çok güç mantık hataları oluşur.
- Dolayısıyla floating-point tipi değişkenlerle program yazarken eşitlik yerine yaklaşık eşitliği kontrol etmek daha iyi bir fikirdir. Örneğin double tipinde bir değişken olan x'in 10.0 'a eşit olup olmadığına bakmaktansa , $|x-10.0| \leq 1E-10$ eşitsizliğine bakmak daha mantıklıdır.

float ve double tipinde değişkenlerin eşitliğinin karşılaştırılması

```
1 ▶ public class FloatingPointEquality {
2
3 ▶     public static void main(String[] args)
4     {
5         final double EPS = 1.0E-14;
6
7         double sinX;
8
9         sinX = Math.sin(2*Math.PI);
10
11         System.out.print("Equality test : ");
12
13         if(sinX==0)
14             System.out.println("sin(2*PI) is equal to zero");
15         else
16             System.out.println("sin(2*PI) is NOT equal to zero");
17
18         System.out.print("Approximate equality test : ");
19
20         if(Math.abs(sinX-0.0) < EPS )
21             System.out.println("sin(2*PI) is equal to zero");
22         else
23             System.out.println("sin(2*PI) is NOT equal to zero");
24
25         System.out.println("\nReal values :");
26         System.out.println("sin(2*PI) = " + sinX);
27
28     }
29 }
```

float ve double tipinde değişkenlerin eşitliğinin karşılaştırılması

/Library/Java/JavaVirtualMachines/jdk-9.jdk/Contents/Home

Equality test : $\sin(2\pi)$ is NOT equal to zero

Approximate equality test : $\sin(2\pi)$ is equal to zero

Real values :

$\sin(2\pi) = -2.4492935982947064E-16$

Process finished with exit code 0

Mantık hatası

- Noktalı virgülü yanlış yere koymak

```
1 ▶ public class IfError {  
2 ▶     public static void main(String args[]) {  
3  
4         int x = 10;  
5  
6         if (x < 0);  
7             System.out.println("x is negative");  
8  
9     }  
10 }
```

x, 0'dan küçükse boş satır çalıştır.

```
/Library/Java/JavaVirtualMachines/jdk-9.  
Connected to the target VM, address: '12  
Disconnected from the target VM, address:  
x is negative
```

```
Process finished with exit code 0
```

Örnek 1

Kullanıcının girdiği herhangi bir x değeri için $y(x) = \ln(1/(1-x))$ fonksiyonunu hesaplayan Java programını yazınız. Programda fonksiyonun tanımsız olduğu bir nokta girilirse ekrana hata mesajı yazdırılmalıdır.

LnFunction.java

Örnek 2

Kullanıcı tarafında girilen negatif olmayan bir tamsayının faktoriyelini hesaplayan Java programını yazınız.(Kullanıcı negatif olmayan bir tamsayı girene kadar sayı girmesini istemeye devam edilmelidir.)

Factorial.java

Örnek 3

Kullanıcıdan pozitif bir tamsayı isteyen ve 1'den bu sayıya kadar tüm tek sayıların çarpımını hesaplayan bir program yazınız.

OddMultiplication.java