

True or False?

Boolean expressions and relational operators



Yes, no, or something else?

PAUSE AND TRY

Which of these questions can only be answered yes or no?

A GAME ASKS...

1. Is the guess equal to the secret number?
2. How many lives are left?
3. Is the temperature below 0?
4. What is the player's name?

Talk with a partner. Which questions could a program use to decide what to do next?

By the end of class you can...

- 1 Use `==` and `!=` to ask whether two values are the same, and predict true or false.
- 2 Use `<`, `>`, `<=`, and `>=` to compare numbers, including values right on the boundary.
- 3 Work out a comparison that contains arithmetic: the arithmetic happens first.
- 4 Store a true-or-false answer in a boolean variable and print it.
- 5 Explain why `==` compares references for Strings, and use `equals` and `compareTo`.

A large, white, rounded square containing the number '1' in a bold, dark teal font.

1

PART 1

Same or different?

A comparison answers true or false



A **relational operator** compares two values and produces a **boolean**. An expression that evaluates to true or false is a **Boolean expression**.

Operator	Read it as
<code>a == b</code>	a is equal to b
<code>a != b</code>	a is not equal to b

CHECKING A GUESS

```
int guess = 3;
int secret = 3;
System.out.println(guess == secret);
System.out.println(guess != secret);
guess = 8;
System.out.println(guess == secret);
System.out.println(guess != secret);
```

OUTPUT

```
true
false
false
true
```

One = stores. Two == asks.



Writing = when you mean == is the most common slip of the day. Java doesn't guess what you meant.

DOES NOT COMPILE

```
int lives = 3;  
boolean full = lives = 5;
```

COMPILER SAYS

```
error: incompatible types:  
int cannot be converted to boolean
```

FIXED: ASK, DON'T STORE

```
int lives = 3;  
boolean full = lives == 5;  
System.out.println(full);
```

OUTPUT

```
false
```

2

PART 2

Bigger or smaller?

The boundary is where they disagree

Operator	True when the left value is...
<	smaller
>	larger
<=	smaller or the same
>=	larger or the same



These four work on numbers. Only the **"or equal"** versions are true when the two values are the same.

EXACTLY ON THE LINE

```
int pages = 100;  
System.out.println(pages > 100);  
System.out.println(pages >= 100);  
System.out.println(pages < 100);  
System.out.println(pages <= 100);
```

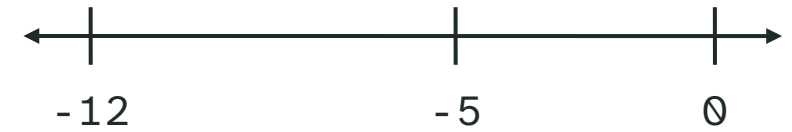
OUTPUT

```
false  
true  
false  
true
```

From words to operators

Words	Boundary counts?	Operator
more than, over, above	No	$>$
at least, no fewer than	Yes	$\geq$
less than, under, below	No	$<$
at most, no more than	Yes	$\leq$

NEGATIVE NUMBERS



$-5 > -12$ is **true**: -5 is farther right.



Type the angle bracket first: $\geq$ and $\leq$. Java has no $\Rightarrow$ or $\Leftarrow$ operator, and $> =$ with a space won't compile either.

Is the book overdue enough?

PAUSE AND TRY

A library fines books more than 14 days late. Write each value.

```
int daysLate = 14;
```

Which expression is the right test for a fine?
Why?

Expression	Value
daysLate > 14	
daysLate >= 14	
daysLate != 14	
-3 > -10	

A large white square with rounded corners, tilted slightly to the right, containing the number '3' in a bold, teal, sans-serif font.

3

PART 3

Math first, then compare

Arithmetic happens before comparison

SHARING AND BUYING

```
int cookies = 17;  
int kids = 4;  
System.out.println(cookies % kids == 0);  
System.out.println(cookies / kids >= 4);  
double price = 6.25;  
System.out.println(price * 3 > 18);
```

OUTPUT

```
false  
true  
true
```

Arithmetic first	Then compare
$17 \% 4 \rightarrow 1$	$1 == 0 \rightarrow \text{false}$
$17 / 4 \rightarrow 4$	$4 \geq 4 \rightarrow \text{true}$
$6.25 * 3 \rightarrow 18.75$	$18.75 > 18 \rightarrow \text{true}$



A number is **even** when $n \% 2 == 0$ is true. The whole expression is still one boolean.

== on double results



== compares the value Java **actually stored**. A double can be a tiny bit off.

```
double total = 1.1 + 2.2;  
System.out.println(total);  
System.out.println(total == 3.3);  
System.out.println(Math.abs(total - 3.3) < 0.000001);
```

OUTPUT

```
3.3000000000000003  
false  
true
```

Safer: ask "very close?" with `Math.abs`, or keep exact amounts, like cents, in an `int`.

A large, white, rounded square containing the number '4' in a bold, teal, sans-serif font.

4

PART 4

Storing the answer

A boolean holds the answer, not the question

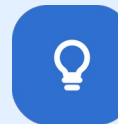
BATTERY CHECK

```
int battery = 25;  
boolean low = battery < 20;  
battery -= 10;  
System.out.println(low);  
System.out.println(battery < 20);
```

OUTPUT

```
false  
true
```

Line	battery	low
1	25	
2	25	false
3	15	false



Java evaluates the comparison **once**, when it is assigned. The variable changes only if you assign to it again.

Labels need parentheses around a comparison



+ happens **before** any relational operator. Java follows operator order, not what you meant.

DOES NOT COMPILE

```
int battery = 15;  
System.out.println("Low: " + battery < 20);
```

COMPILER SAYS

```
error: bad operand types  
for binary operator '<'
```

FIXED

```
int battery = 15;  
System.out.println("Low: " + (battery < 20));
```

OUTPUT

```
Low: true
```

A large, bold, teal number '5' is centered within a white rounded square that has a slight 3D effect and a shadow.

PART 5

Comparing Strings

KEY IDEA

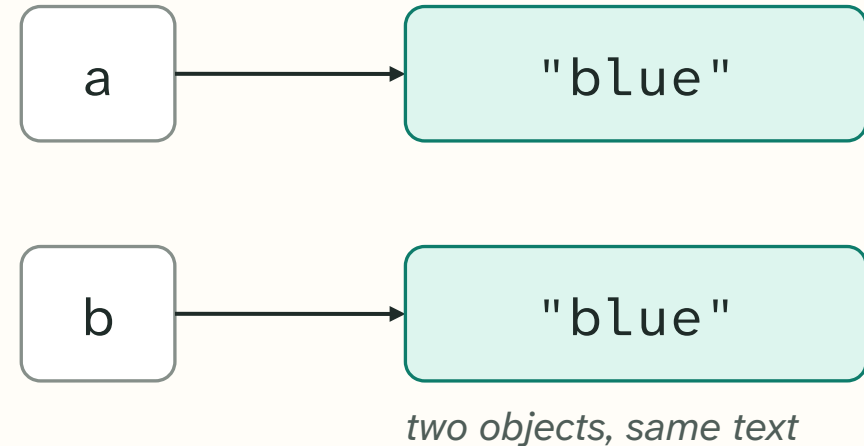
== compares references, equals compares text

```
String a = new String("blue");  
String b = new String("blue");  
System.out.println(a == b);  
System.out.println(a.equals(b));  
System.out.println(a != b);
```

OUTPUT

```
false  
true  
true
```

IN MEMORY



For reference types, == and != compare **references**. Use equals for text.

Alphabetical order: compareTo, then 0

```
String x = "kiwi";  
String y = "mango";  
System.out.println(x.compareTo(y) < 0);  
System.out.println(y.compareTo(x) < 0);  
System.out.println(x.length() < y.length());
```

OUTPUT

```
true  
false  
true
```

compareTo is...	Meaning
negative	x comes first
zero	same text
positive	x comes after



`x < y` won't compile: **bad operand types**. The size operators only work on numbers.

Same password?

PAUSE AND TRY

Write the exact output. Which line uses the wrong tool?

```
String saved = new String("tiger");  
String typed = new String("tiger");  
boolean match = saved.equals(typed);  
typed = new String("Tiger");  
System.out.println(match);  
System.out.println(saved == typed);  
System.out.println(saved.equals(typed));
```

YOUR OUTPUT

Words to know

Relational operator

`==, !=, <, >, <=, >=`: compares two values and gives a boolean.

Boolean expression

An expression that evaluates to true or false.

boolean

The primitive type with exactly two values: true and false.

Boundary value

The value right on the line, like exactly 100. Test it.

Even

A whole number with no remainder when divided by 2:
`n % 2 == 0`.

equals

Compares the text of two Strings. `==` compares references.

Ask Java a question



Explain to your partner when `==` gives the answer you expect, and when it doesn't.

SENTENCE STARTER

With int values, `==` compares _____, but with two Strings it compares _____, so I use _____.

Next: the True or False? puzzle worksheet.