

NAME _____

DATE _____

PERIOD _____

UNIT 1 · TOPIC 1.10 · WORKSHEET

Calling on the Class

Read the headers, rebuild a fence program, and settle an argument between two calls to the same method.

About 30 minutes

10 questions

Cumulative through 1.10

TOPIC

1.10**DIRECTIONS**

Work each puzzle in pencil. For code, write exactly what prints, one line per `println`. You may use the Java Quick Reference. Some puzzles include pieces you won't need.

**THINK FIRST**

Does `Math.sqrt(16)` print 4 or 4.0? Guess in the margin; question 1 settles it.

PART A · 2 QUESTIONS

A**Warm up your brain****1**

YOU'RE THE JVM

You are the Java Virtual Machine. `Math.abs` returns the absolute value of its argument (the number without its minus sign), and `Math.sqrt` returns the square root. What prints?

```
int low = 3;
int high = 12;
System.out.println(Math.abs(low - high) + Math.sqrt(16));
```

A 13**B** 13.0**C** -5.0**D** 17.0



FAIR QUESTION

The square root of 16 is a whole number. Why the .0?

The type of the result comes from the header, not from the numbers. The header of `sqrt` says it returns a `double`, so every call gives back a `double`; even `Math.sqrt(100)` prints `10.0`.

2

PENCIL CHECK

Each row is a real method header from the Java library. Decide whether it is a *class* method or an *instance* method, and write the type of value it returns (write *nothing* if it returns nothing). `public` is explained in a later unit; read past it.

Class	Header	Class or instance?	Returns
Math	<code>public static double sqrt(double a)</code>		
String	<code>public int length()</code>		
Scanner	<code>public double nextDouble()</code>		
your class	<code>public static void main(String[] args)</code>		
Math	<code>public static double pow(double a, double b)</code>		



WATCH OUT

`static` does not mean "the answer never changes." `Math.abs(-2)` and `Math.abs(-50)` return different values. `static` only says who the method belongs to: the class, not one object. The return type is the word right before the method name.

PART B · 4 PUZZLES

B Puzzles

3 CODE TILES

A garden program fell off the fridge. It reads the area of a square garden, then prints the length of one side and the length of fence that goes all the way around. `Math.sqrt` returns a square root. Put tiles in order inside `main` so that when the user types 49, it prints the output below. Two tiles are decoys.

Fence.java (already in place)

```
import java.util.Scanner;
public class Fence {
    public static double perimeter(double side) {
        return side * 4;
    }
    public static void main(String[] args) {
        Scanner console = new Scanner(System.in);
        // your tiles go here
    }
}
```

<code>System.out.println(perimeter(side));</code>	<code>double side = sqrt(area);</code>
<code>int area = console.nextInt();</code>	<code>System.out.println(perimeter(area));</code>
<code>System.out.println(side);</code>	<code>double side = Math.sqrt(area);</code>

Target output

7.0

28.0

YOUR PROGRAM

4 SNIPPET POOL

These are two separate classes. `Trip` calls a class method that lives in `Convert`. `Math.abs` returns the absolute value of its argument. Fill each numbered hole with a snippet from the pool so that `Trip` prints `40.0`.

`Convert.java`

```
public class Convert {
    public static double toKm(double miles) {
        return miles * 1.6;
    }
}
```

`Trip.java`

```
public class Trip {
    public static void main(String[] args) {
        (1) km = (2).toKm((3));
        System.out.println(km);
    }
}
```

THE POOL

Each snippet can be used only once, and not every snippet is needed.

<code>int</code>	<code>Trip</code>	<code>Math.abs(-25)</code>	<code>double</code>
<code>-25</code>	<code>convert</code>	<code>abs(-25)</code>	<code>Convert</code>

5 FACE-OFF

The class `Recipe` has a class method with the header `public static double toCups(double spoons)`. Tonight, two calls to it argue: the **Short Call** `toCups(8)` and the **Full Call** `Recipe.toCups(8)`. Fill in the blanks.

FULL CALL	I say my class name up front, then a _____, then the method. Java never has to guess where to look.
SHORT CALL	I skip all that, and I still work, as long as I'm written inside <code>Recipe</code> , the method's _____.
FULL CALL	Try that inside a class called <code>Menu</code> , and see how far you get.
SHORT CALL	Fine. In <code>Menu</code> , Java looks for <code>toCups</code> in _____, finds nothing, and the program doesn't _____.
FULL CALL	Inside <code>Recipe</code> , though, we're twins. We both run the _____ method and get the same value back.
SHORT CALL	And because <code>toCups</code> has the word _____ in its header, neither of us needs an _____ to run it.

6 WHO AM I?

Some ideas from today are playing a party game. Each gives a clue about itself. Name it.

Clue	I am...
I'm the class a method is written in. Inside me, you may leave my name out of the call.	
I'm a class method in every program you've written. Java runs me first, with no object.	
I'm the error you get when you call <code>pow(2, 3)</code> in your own program without <code>Math.</code> in front.	
I'm a <code>Math</code> method with two versions: one takes an <code>int</code> , the other takes a <code>double</code> .	
I'm the value inside the parentheses of a call. Java works me out before the method runs.	
When a non-void method finishes, the whole call is replaced by me.	

PART C · 4 QUESTIONS

C

Catch the mistake

7

YOU'RE THE COMPILER

You are the compiler. A class named `Helper` has the two class methods below. Each snippet in the table is in the main method of a *different* class, not in `Helper`. Decide whether it compiles. If it does, write what it prints. If it doesn't, say why in a few words.

`Helper.java` (headers only)

```
public static int square(int n)    // returns n * n
public static void shout()        // prints HEY
```

Snippet	Compiles?	Output, or why not
1 <code>int x = Helper.square(4);</code> <code>System.out.println(x);</code>		
2 <code>int x = square(4);</code> <code>System.out.println(x);</code>		
3 <code>int x = Helper.shout();</code>		
4 <code>Helper.shout();</code>		
5 <code>double d = Helper.square(3);</code> <code>System.out.println(d);</code>		
6 <code>int y = Helper.square(2.5);</code>		

8

ONE-TOKEN FIX

Nine square meters of garden are shared equally by four people. This program should print the side of each person's square plot, 1.5, but it prints 1.4142135623730951. Change exactly one token to fix it.

```
1 int area = 9;
2 int people = 4;
3 double side = Math.sqrt(area / people);
4 System.out.println(side);
```

LINE, OLD TOKEN, NEW TOKEN

9 TWO TRUTHS AND A LIE

Two of these claims about the code are true and one is a lie. Circle the lie, then rewrite it so it's true.

```
1 double balance = -12.5;  
2 Math.abs(balance);  
3 double fixed = Math.abs(balance);
```

- **1** Line 2 compiles, even though nothing stores its result.
- **2** After line 2 runs, `balance` holds 12.5.
- **3** After line 3 runs, `fixed` holds 12.5.

THE LIE, REWRITTEN

**FAIR QUESTION**

If line 2 changes nothing, why does Java allow it?

Java lets you call any method as a statement on its own. For a method that returns a value, the value is simply dropped. That's useful for a `void` method, which does its job and returns nothing. For a method like `abs`, whose whole job is its return value, it's almost always a mistake, so store the result or use it in an expression.

10 WRAP-UP**REFLECTION**

In one or two sentences: when can you leave the class name out of a class method call, and why does Java need it everywhere else?

YOUR ANSWER