

NAME _____

DATE _____

PERIOD _____

UNIT 1 · TOPIC 1.11 · WORKSHEET

Math Class Puzzles

Rebuild a power program, rig a random number, and settle an argument between two Math methods.

About 30 minutes

10 questions

Cumulative through 1.11

TOPIC

1.11

**DIRECTIONS**

Work in pencil. When code prints, 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****Guess before you read on**

Does `Math.pow(2, 3)` give 8 or 8.0? 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. Run this code in your head. What prints?

```
System.out.println(Math.pow(2, 3) + Math.abs(-7));
```

A 15**B** 15.0**C** 1.0**D** 13.0

2 PENCIL CHECK

Write the value of each expression exactly as `System.out.println` would show it. Watch for `.0`.

Expression	Value
<code>Math.abs(-12)</code>	
<code>Math.abs(-2.5)</code>	
<code>Math.abs(3 - 10)</code>	
<code>Math.pow(3, 2)</code>	
<code>Math.sqrt(49)</code>	
<code>Math.sqrt(9 / 2)</code>	
<code>(int) Math.sqrt(30)</code>	



FAIR QUESTION

Why isn't `Math.sqrt(7 / 2)` about 1.87?

Java works out every argument *before* it calls the method. `7 / 2` uses two `int` values, so it's integer division and gives 3. `sqrt` never sees 3.5. It gets 3 and returns about 1.73. Want the square root of 3.5? Write `Math.sqrt(7.0 / 2)` so one of the numbers is a `double`.

PART B · 3 PUZZLES

B Puzzles

3 CODE TILES

A power calculator fell off the fridge and its tiles scattered. The start of the program is still stuck in place. Put the tiles back in order underneath it so that when the user types 2 5, the program prints the output below. One tile is a decoy and isn't used.

PowerUp.java (already in place)

```
import java.util.Scanner;

Scanner console = new Scanner(System.in);
// your tiles go here
```

System.out.println((int) result);	int exp = console.nextInt();
int result = Math.pow(base, exp);	System.out.println(result);
double result = Math.pow(base, exp);	int base = console.nextInt();

Target output

32.0

32

YOUR PROGRAM

4 SNIPPET POOL

A board game moves a token a random number of squares, from -3 through 3, with both -3 and 3 possible. Fill each numbered hole with a snippet from the pool so that every run prints one of -3, -2, -1, 0, 1, 2 or 3, and nothing else.

Move.java

```
int move = (1) (Math.random() * (2)) (3);
System.out.println(move);
```

THE POOL

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

(double)	6	- 3	(int)
+ 3	7	- 4	



WATCH OUT

Where the cast goes matters

`(int) Math.random() * 100` is always 0. A cast grabs only the value right after it, so `Math.random()` gets cast first. Every value it returns is less than 1.0, so the cast makes it 0, and `0 * 100` is still 0. Put the multiplication in parentheses so the cast happens last.

5 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.

```
double r = Math.random();
int pick = (int) (r * 10);
```

- **1** `r` can be exactly 0.0.
- **2** `r` can be exactly 1.0.
- **3** `pick` is always one of 0 through 9.

THE LIE, REWRITTEN

PART C · 5 QUESTIONS

C

Catch the mistake

6

ONE-TOKEN FIX

This program should roll a six-sided die: every run prints one of 1, 2, 3, 4, 5 or 6. It runs, but it never prints 6. Change exactly one token to fix it.

```
1 int roll = (int) (Math.random() * 5) + 1;
2 System.out.println(roll);
```

LINE, OLD TOKEN, NEW TOKEN

7

YOU'RE THE COMPILER

You are the compiler. For each snippet, decide whether it compiles. If it does, write what it prints. If it doesn't, say why in a few words.

Snippet	Compiles?	Output, or why not
1 int p = Math.pow(2, 3); System.out.println(p);		
2 int p = (int) Math.pow(2, 3); System.out.println(p);		
3 int d = Math.abs(-4); System.out.println(d);		
4 int d = Math.abs(-4.5); System.out.println(d);		
5 double y = sqrt(81); System.out.println(y);		



FAIR QUESTION

None of these snippets imports Math. Why do they compile?

Math lives in the `java.lang` package, and every class in `java.lang` is available to every program automatically. `Scanner` lives in `java.util`, which is why the program in question 3 needs `import java.util.Scanner;` and these snippets need nothing.

8 FACE-OFF

Tonight: `Math.abs` and `Math.pow` argue about which one is easier to use. Fill in the blanks.

MATH.ABS	Hand me an <code>int</code> and you get an _____ back. Hand me a <code>double</code> and you get a _____ back. Two versions, one name.
MATH.POW	I don't need two versions. I always return a _____. Even <code>Math.pow(2, 3)</code> comes back as _____.
MATH.ABS	So when someone wants a whole number from you, they're stuck?
MATH.POW	Not stuck. They just add a _____. <code>(int) Math.pow(2, 3)</code> gives them 8.
MATH.ABS	Nobody has to cast me. When they call <code>Math.abs(-4)</code> , the compiler picks my <code>int</code> version by looking at the _____.

9 WHO AM I?

The Math gang is playing a party game. Each one gives a clue about itself. Name it.

Clue	I am...
I take no arguments. Every value I return is at least <code>0.0</code> and less than <code>1.0</code> .	
I take two values and return the first one raised to the power of the second.	
I come in two versions, one for <code>int</code> and one for <code>double</code> .	
I take one value and return its nonnegative square root.	
I'm a class full of class methods. You call each one through my name.	

10 WRAP-UP



REFLECTION

In one or two sentences: `Math.random()` never returns `1.0`. Explain why that means `(int) (Math.random() * 4)` can never be 4.

YOUR ANSWER