Introduction to Java

Today, we'll take a first look at Java programs. We will also see some ways to use generative artificial intelligence (GenAI) tools in this course, especially if you're new to Java.

Throughout the course, we will occasionally use GenAI tools, being careful to still build foundational understanding and skills. In both education and industry, we want GenAI tools to ***support***, not undermine, our goals. We'll also discuss how GenAI tools are currently used in industry, including best practices and ethical considerations.

You'll work in pairs today; next time, you'll work in larger teams. If you're new to Java, work with someone else who is new to Java. And if you already have some Java experience, find a partner with a similar level of Java experience.

Content Learning Targets

After completing this activity, you should be able to say:

- I can identify components of simple Java programs.
- I can write Java code to declare int and double variables.
- I can explain what it means to assign a value to a variable.
- I can evaluate Java expressions that use the % operator with integers.
- I can explain the difference between integer and floating-point division.
- I can identify operator precedence for addition, division, and assignment.
- I can name Java's primitive data types and give examples of each one.
- I can identify illegal assignment statements, and explain why they are illegal.
- I can use GenAI tools to learn about Java syntax and features.

Process Skill Goals

During the activity, you should make progress toward:

- Leveraging prior knowledge and experience of other students. (Teamwork)

Model 1 Welcome to Java!

Questions (10 min) Start time:

Find the file `WelcomeToJava.java` in the `src` folder.

1. What do you notice about the name of the class? What happens when you try changing it?

2. What is the purpose of the first 14 lines? What is the purpose of the blank lines?

3. Describe in your own words what `System.out.println` does. Be very specific.

Model 2 Variables and Operators

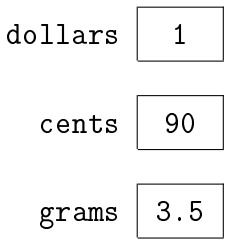
Most programs store and manipulate data values. We use ***variables*** to give values meaningful names. The following code ***declares*** three variables and ***assigns*** them (using the = operator). Each variable is stored in the computer's memory, represented by the boxes on the right.

Java code

```
int dollars;
int cents;
double grams;

dollars = 1;
cents = 90;
grams = 3.5;
```

Computer memory



Questions (15 min) Start time:

4. Identify the Java ***keyword*** used in a variable declaration to indicate

a) an integer:

b) a real number:

5. What would you expect the following statements to print out?

a) `System.out.println(dollars);`

b) `System.out.println(cents);`

c) `System.out.println(grams);`

6. In the previous question, how does the third printed line (c) differ from the first two?

7. What do you think is the purpose of a variable declaration?

8. In your own words, explain how you should read the = sign in Java. For example, the Java statement `x = a + b;` should be read out loud as "x _____ a plus b".

9. Find the `// === Model 2 ===` comment in `WelcomeToJava.java`. Work through the predict-and-run exercises.

a) Notice the Java order of operations.

b) For the dividend and divisor example, try changing their values and see what you get. What do you think the / and % Java operators do, given two integers?

c) For the directly-printed division examples that follow, do any results surprise you? What do you think is the general pattern for how Java handles division?

Model 3 Primitive Types

Keyword	Size	Min Value	Max Value	Example
`byte`	1 byte	-128	127	`(byte) 123`
`short`	2 bytes	$-32,768$	$32,767$	`(short) 12345`
`int`	4 bytes	-2^{31}	$2^{31}-1$	`1234567890`
`long`	8 bytes	-2^{63}	$2^{63}-1$	`123456789012345L`
`float`	4 bytes	-3.4×10^{38}	3.4×10^{38}	`3.14159F`
`double`	8 bytes	-1.8×10^{308}	1.8×10^{308}	`3.141592653589793`
`boolean`	1 byte	N/A	N/A	`true`
`char`	2 bytes	0	65,535	`'A'`

Note that 1 byte is 8 bits, i.e., eight "ones and zeros" in computer memory. Since there are only two possible values for each bit, you can represent $2^8 = 256$ possible values with 1 byte.

Questions (15 min)

Start time:

10. Which of the primitive types are integers? Which are floating-point?

11. Why do primitive types have ranges of values? What determines the range of a type?

12. Since a `byte` can represent 256 different numbers, why is its max value 127 and not 128?

13. What is the data type for each of the following values?

1.14159		7.2E-4		-128	
0		0.0		'0'	
-1.0F		-13L		false	
123		'H'		true	

14. Based on these examples, when does Java let you assign one type of primitive to another?

```
int int_ = 3;
long long_ = 3L;
float float_ = 3.0F;
double double_ = 3.0;

int_ = int_;
int_ = long_;      // illegal
int_ = float_;     // illegal
int_ = double_;    // illegal

long_ = int_;
long_ = long_;
long_ = float_;    // illegal
long_ = double_;   // illegal

float_ = int_;
float_ = long_;
float_ = float_;
float_ = double_;  // illegal

double_ = int_;
double_ = long_;
double_ = float_;
double_ = double_;

int_ = '0';
int_ = false;      // illegal
double_ = '0';
double_ = false;   // illegal
```

15. Given the following variable declarations, which of the assignments are not allowed?

```
byte miles;
short minutes;
int checking;
long days;
float total;
double sum;
boolean flag;
char letter;

checking = 56000;
total = 0;
sum = total;
total = sum;
checking = miles;
sum = checking;
flag = minutes;
days = '0';
```

Model 4 Generative AI Tools for Software Development

Questions (20 min) Start time:

If you are **new to Java**, complete #16. If you are **already familiar** with Java, complete #17.

16. GenAI tools can help you switch from another programming language to Java.

a) Open the provided `ConvertToJava.java` file.

b) Find a short function or code snippet (ideally, one you wrote) in a non-Java language with which you are familiar.

c) Have a GenAI tool (e.g., Claude, ChatGPT, Gemini) convert your code into Java (sample).

d) Test the Java version: copy/paste into `ConvertToJava.java` and modify the method call.

e) If the response doesn't explain the changes, ask. **What similarities and differences do you notice?** (**Tip**: Repeat for other non-Java code to help you adapt to Java syntax.)

f) When you finish this activity, take a look at `JavaExamples.java` and review.

17. GenAI tools can help you learn about different ways to solve a problem using Java.

a) Open `JavaExamples.java` and look through the provided example methods.

b) Choose one method, and try to solve the same problem in a different way using your current knowledge of Java. Write your new version as a separate method with a similar name, and test it by calling it from the `main` method. (If you have trouble coming up with a completely new approach, it's OK to just make a small change.)

c) Copy/paste the original method into a GenAI chat tool such as Claude, ChatGPT, or Gemini, along with the prompt, "Give me different ways to implement this Java method." Review the responses. As time allows, repeat for other methods.

d) **What new Java features/syntax did you learn?** Jot down some notes below. Which alternative do you like the best, and why? (**Tip**: Repeat with other Java code you've written.)

e) When you finish this activity, take a look at the first homework assignment.