



IN1007 Programing in Java

Lecture 1: Introduction

DR MAI ELSHEHALY

MAI.ELSHEHALY@CITY.AC.UK

Today's Lecture

- Module information
- What is Java?
- Our first Java program: HelloWorld
- Some core concepts

Academic Team (1/2)

Module Leader:



Dr Mai Elshehaly (Mai.Elshehaly@city.ac.uk)

Office: A302 (College Building)

Web: www.invisai.com/mai

Office hours: Thursdays 15:00 – 16:00

Fridays 12:00 – 13:00

Academic Team (2/2)

Tutorials are led by a team of six wonderful teaching assistants:

- Aravin (aravin.naren.1@city.ac.uk),
- Charlie (charles.watson.1@city.ac.uk),
- Kam (k.pal@city.ac.uk),
- Olga (olga.herrero.1@city.ac.uk),
- Hossam (Hossam.Youssef@city.ac.uk), and
- Sandamali (Sandamali.Wickramasinghe@city.ac.uk).

Our Timetable

Every Monday:

- Lecture slides (and sometimes additional material) uploaded on Moodle – **PLEASE CHECK!**

Every Tuesday:

- **Lecture:** Two hours of discussion, Q&A, hands-on work **please** have your laptops charged & your questions / thoughts ready
- **Tutorials:** One hour – guided technical work

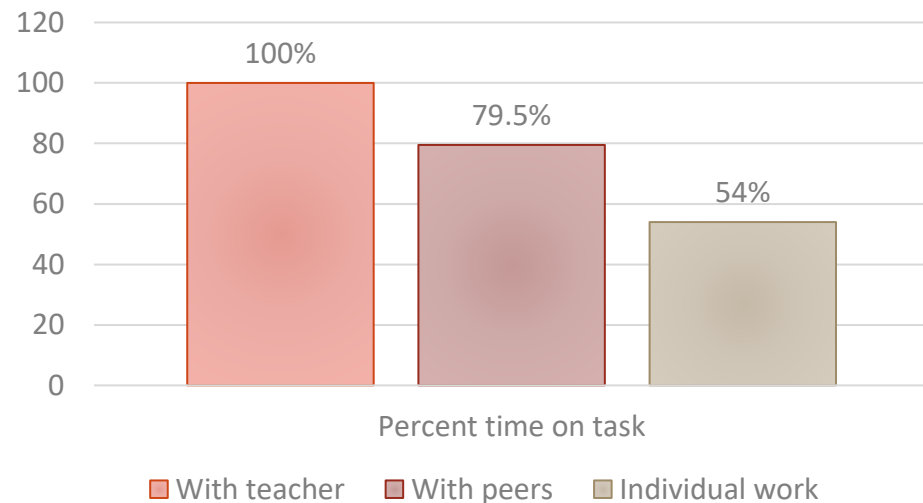
Every week: between 8 – 12 hours of self-guided work

Attendance!!!!

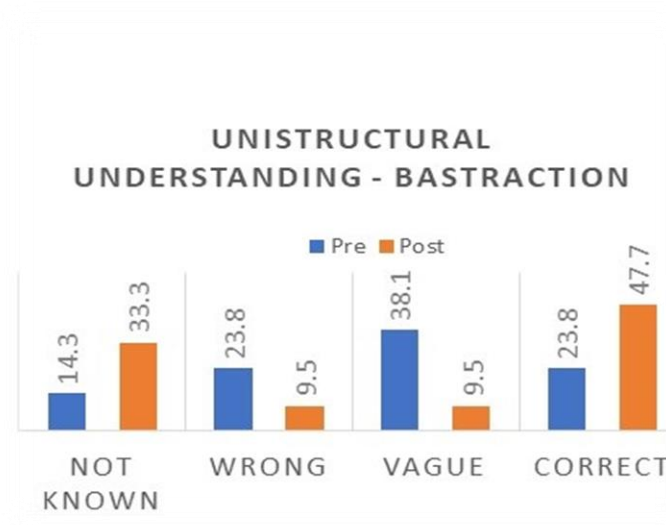
Long commutes? I hear you BUT...

The impact of “social mediation” on learning in CS

Participation



Learning



Increased:

- ✓ Motivation
- ✓ Confidence
- ✓ Sense of belonging

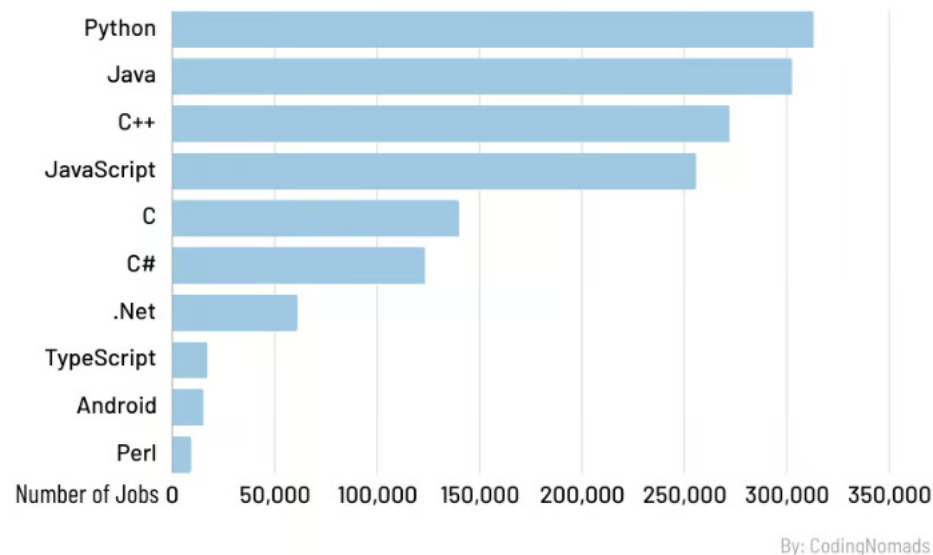
Katherine Hiley, Hannah Cebolla, and Mai Elshehaly. 2023. The Impact of Non-Formal Computer Science Outreach on Computational Thinking in Young Women. In Proceedings of the 2023 Conference on Innovation and Technology in Computer Science Education V. 2 (ITiCSE 2023). Association for Computing Machinery, New York, NY, USA, 642. <https://doi.org/10.1145/3587103.3594197>



Motivation (1/2) - Why Learn Java?

Most in-demand programming languages of 2024

Based on LinkedIn job postings in the US



- ✓ 90% of Fortune 500 companies use Java for their backend architecture
- ✓ Many startups too
- ✓ Runs on billions of devices, including smart phones, automobiles, medical devices, etc.
- ✓ Prominent websites like eBay, LinkedIn, Google, etc.

Motivation (2/2): Why Learn Java?

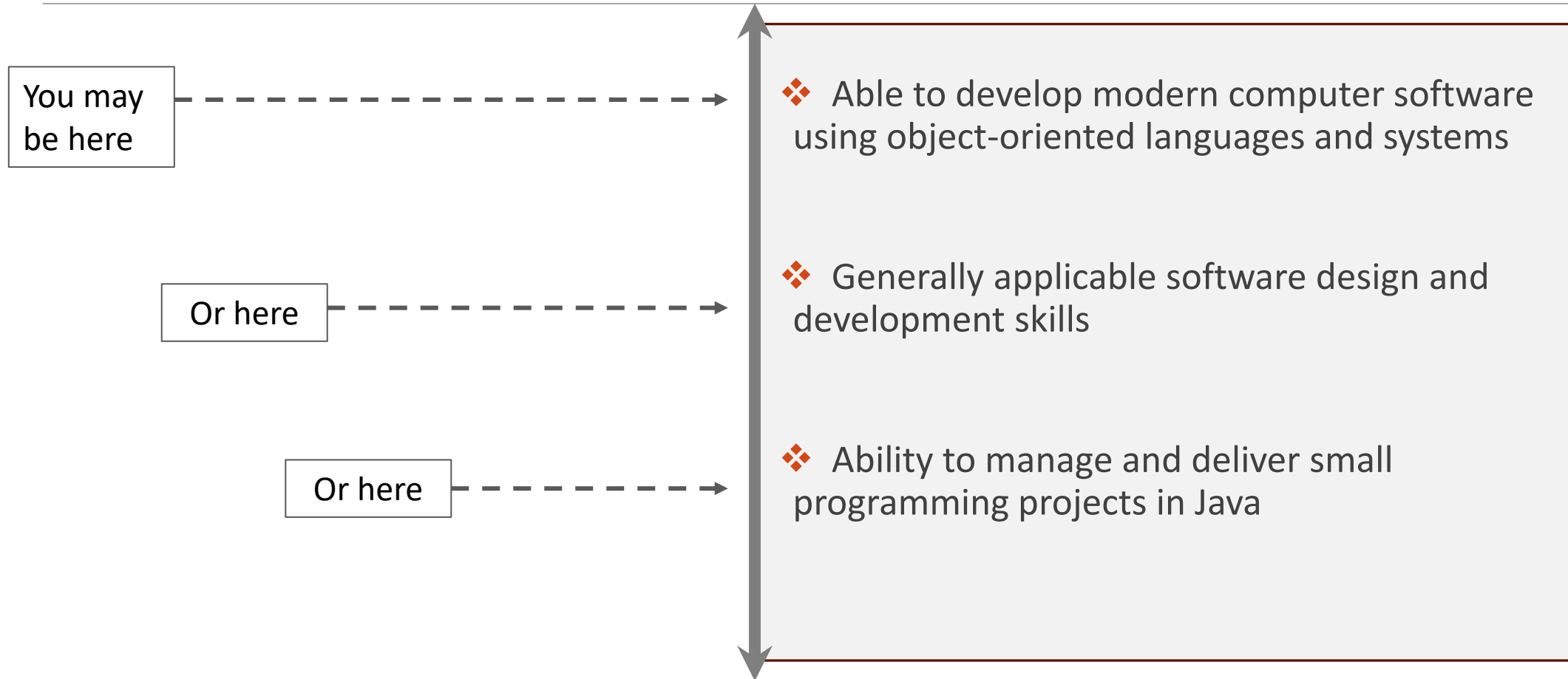
- Develop professional skills
- Deepen problem solving and software engineering skills
- Master object-oriented programming
- Prepare for next year's modules

Class Activity (3 minutes)

<https://shorturl.at/cqMX9>



Module Aims



Module Outline

Weeks
3-6:

- Building blocks: data structures, conditions, loops, methods and recursion

Weeks
7-9:

- Classes & Objects

Week
10

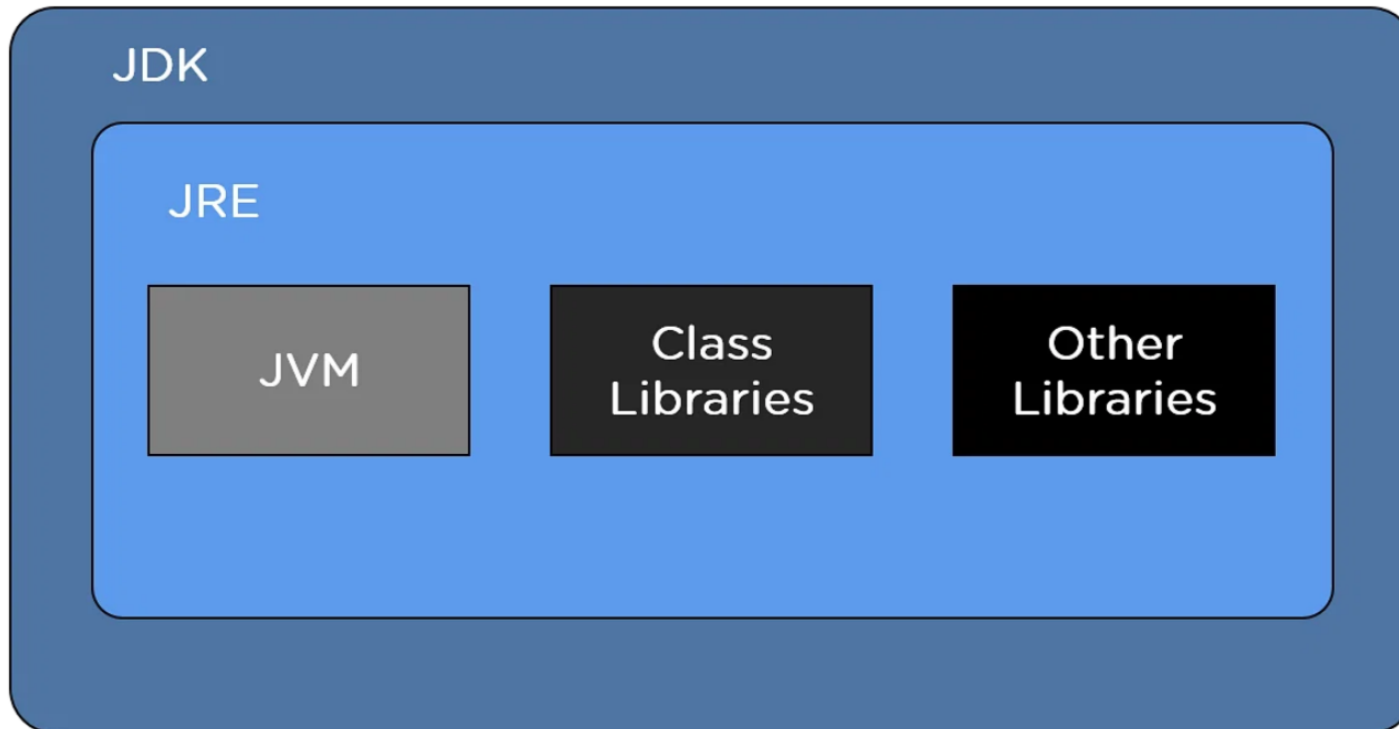
- Collections

Module Assessment

Timing	Assessment	Weight
Week 7	Multiple Choice Questions (MCQ) Quiz Released: 12 November 2024 at 2pm. Deadline: 17 November 2024 at 5pm	10 points
Week 8	Java program to submit on Moodle Released: 21 November 2024. Deadline: 1st December 2024 at 5pm	20 points
Week 11	Oral assessment (viva) Scheduled during Week 11. The exact schedule will be made accessible in due time.	20 points

Today's Lecture

- Module information
- **What is Java?**
- Our first Java program: HelloWorld
- Some core concepts

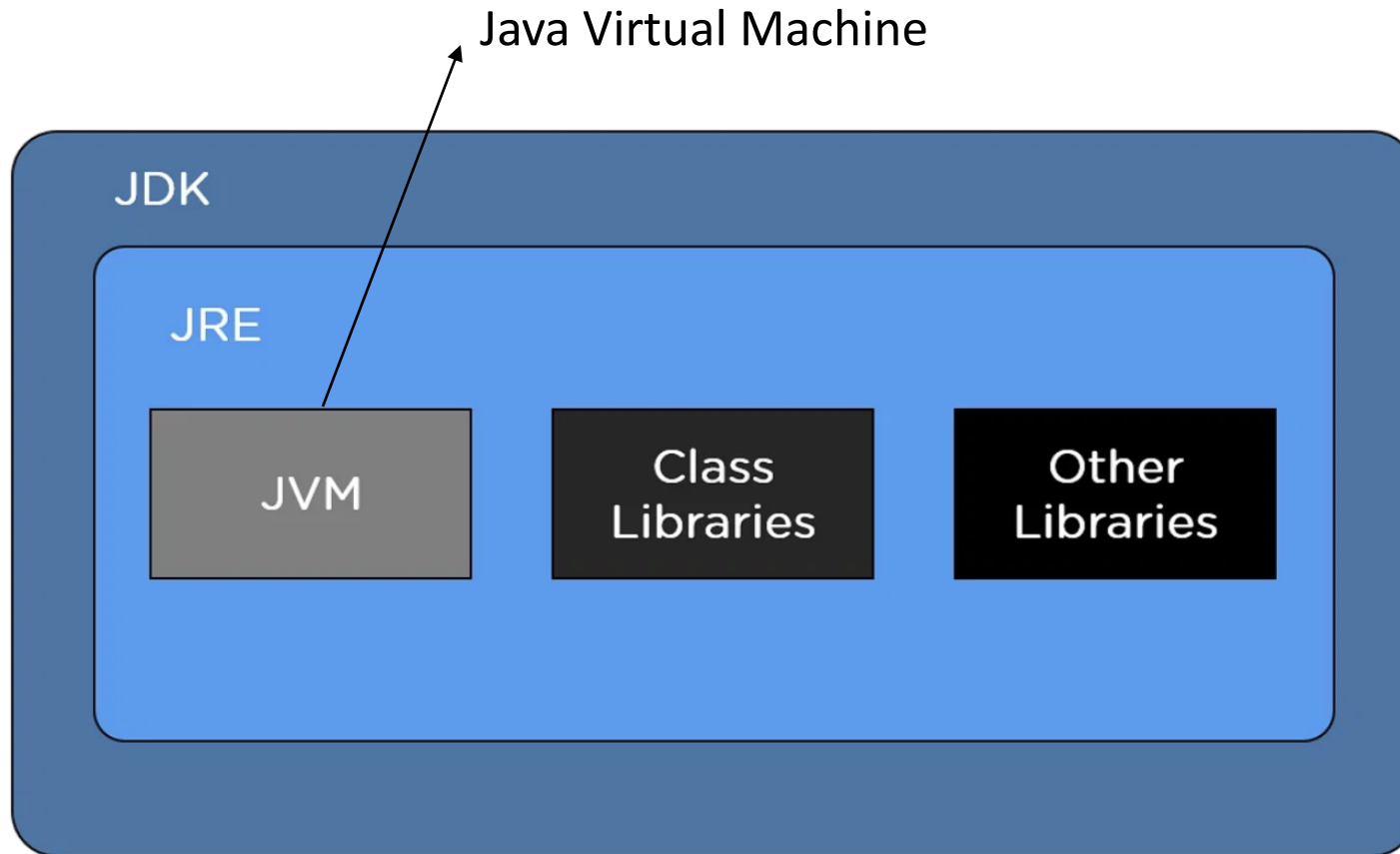


What is Java?

A Programming Language

A Computing Platform for
executing **bytecode**

[What is a Java Development Kit \(JDK\)? Why Do We Need It? \(simplilearn.com\)](https://www.simplilearn.com/What-is-a-Java-Development-Kit-JDK-Why-Do-We-Need-It-?tid=US-MCQ&cid=US-MCQ)

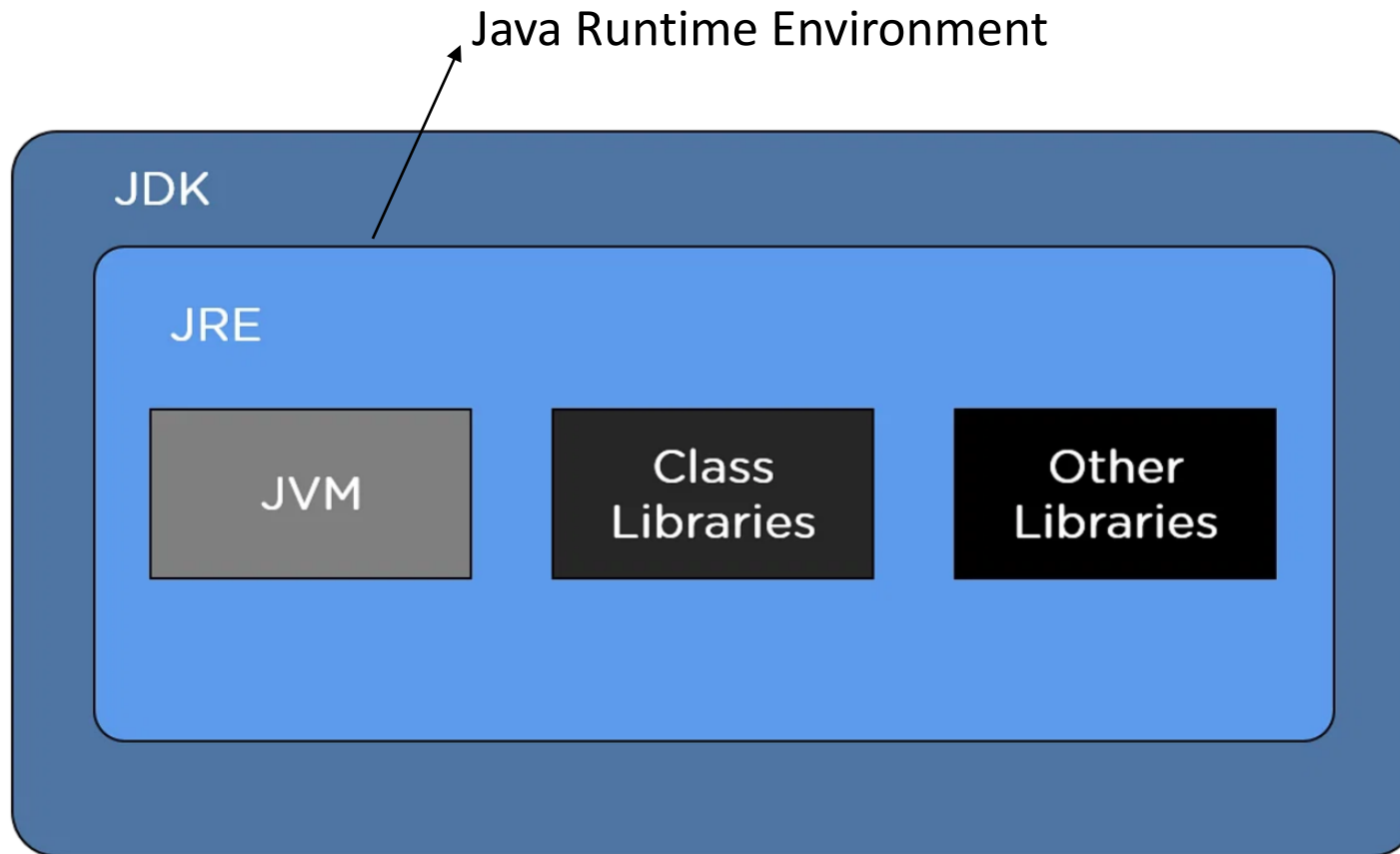


What is Java?

A Programming Language

A Computing Platform for
executing **bytecode**

[What is a Java Development Kit \(JDK\)? Why Do We Need It? \(simplilearn.com\)](https://www.simplilearn.com/What-is-a-Java-Development-Kit-(JDK)-Why-Do-We-Need-It-?)

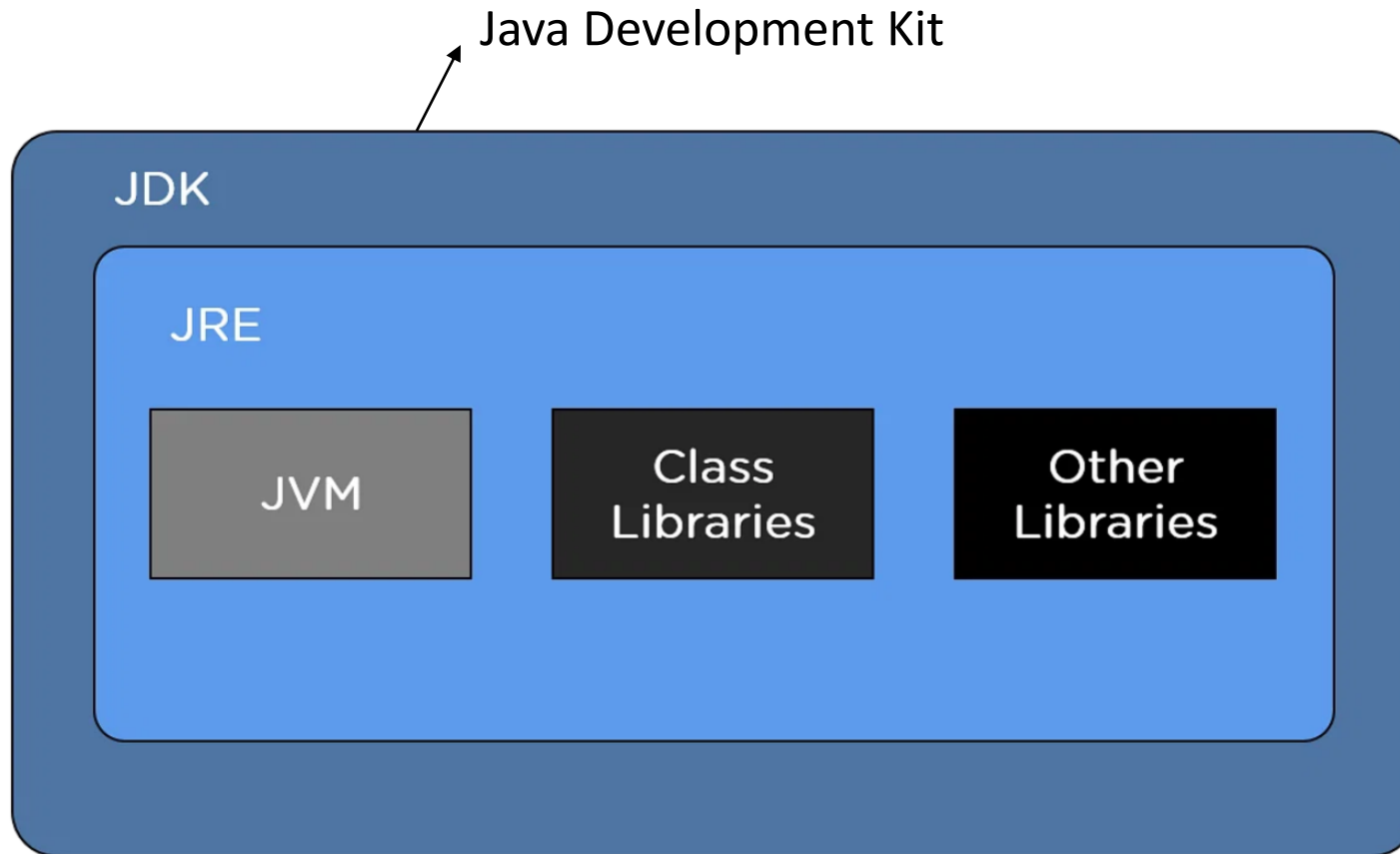


[What is a Java Development Kit \(JDK\)? Why Do We Need It? \(simplilearn.com\)](https://www.simplilearn.com/What-is-a-Java-Development-Kit-JDK-Why-Do-We-Need-It-?utm_source=organic&utm_medium=search&utm_campaign=What-is-a-Java-Development-Kit-JDK-Why-Do-We-Need-It-)

What is Java?

A Programming Language

A Computing Platform for
executing **bytecode**



What is Java?

A Programming Language

A Computing Platform for
executing **bytecode**

[What is a Java Development Kit \(JDK\)? Why Do We Need It? \(simplilearn.com\)](https://www.simplilearn.com/What-is-a-Java-Development-Kit-(JDK)-Why-Do-We-Need-It/)

Java Compiler and Virtual Machine

The Java Compiler

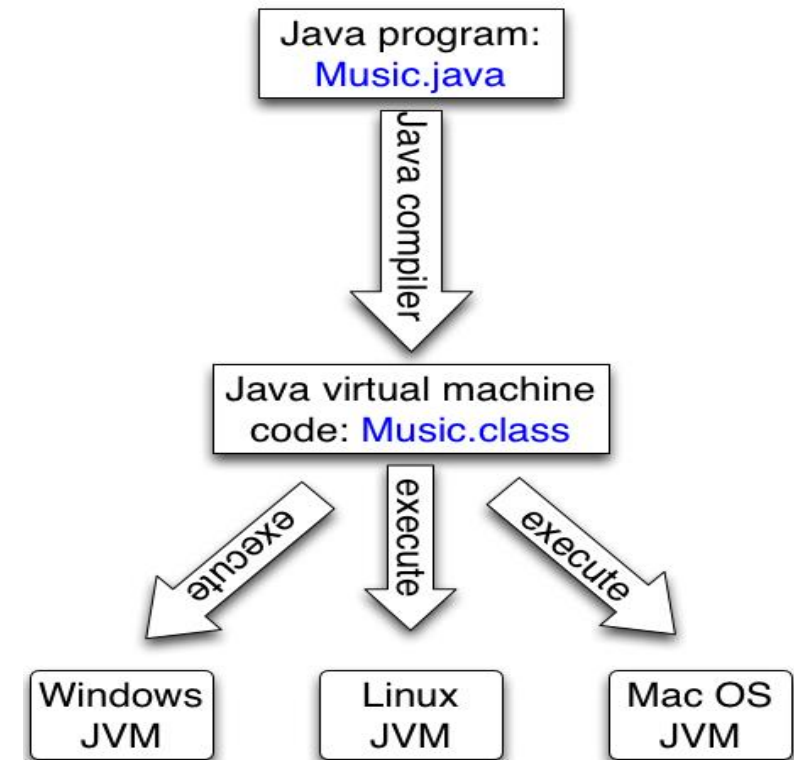
- Reads file with extension *.java*
- Checks syntax / grammar
- Creates a .class file which contains byte code (machine code) independent of any machine

Java Byte Code

- *Is portable*

The JVM (Java Virtual Machine)

- Translates *byte code* into instructions for a particular processor, which “runs the program”



If you haven't already...

✓ Setting up Java and IntelliJ

Please follow these instructions to install Java and IntelliJ before the first tutorial session. Don't worry too much if you don't fully understand what you are doing; if you just follow these instructions step-by-step it will allow us to start writing Java programs during the first session.

For Mac:

Follow the instructions in the following videos (in this order), in the link below:

1. [install_jdk_mac.mp4](#)
2. [install_idea_mac.mp4](#)

For Windows:

Follow the instructions in the following videos (in this order), in the link below:

1. [install_jdk_win.mp4](#)
2. [install_idea_win.mp4](#)

Your Process for Writing a Java Program

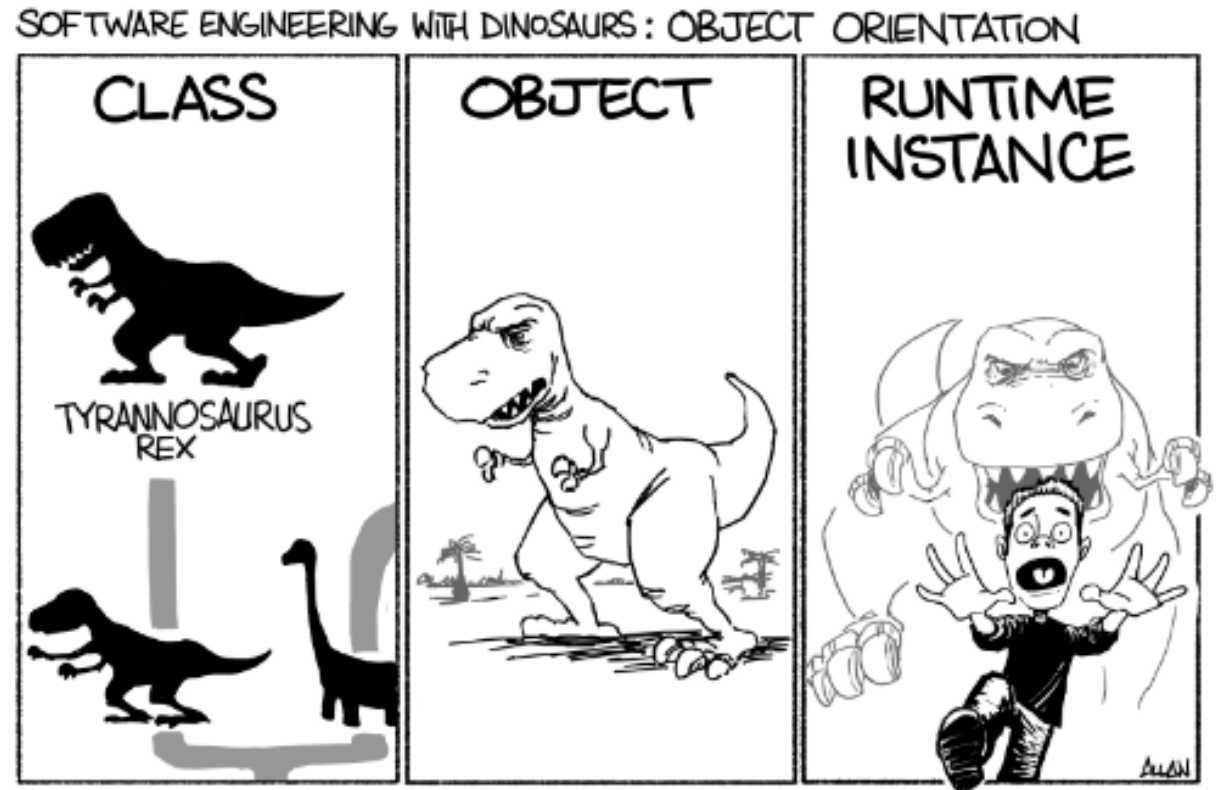
1. Think about the problem
2. Design the **components** of a solution
3. Write code
4. Test code
5. Correct / debug code
6. Look for ways to improve / optimise performance (?)

Designing Components of a Solution

Think algorithmically:

Ask yourself (details in Week 7)

- What types of objects (i.e., classes) need to exist?
- What behaviours / functions does each type of object exhibit?
- Which objects will need to interact with each other?
- How to structure your code such that they know about each other?

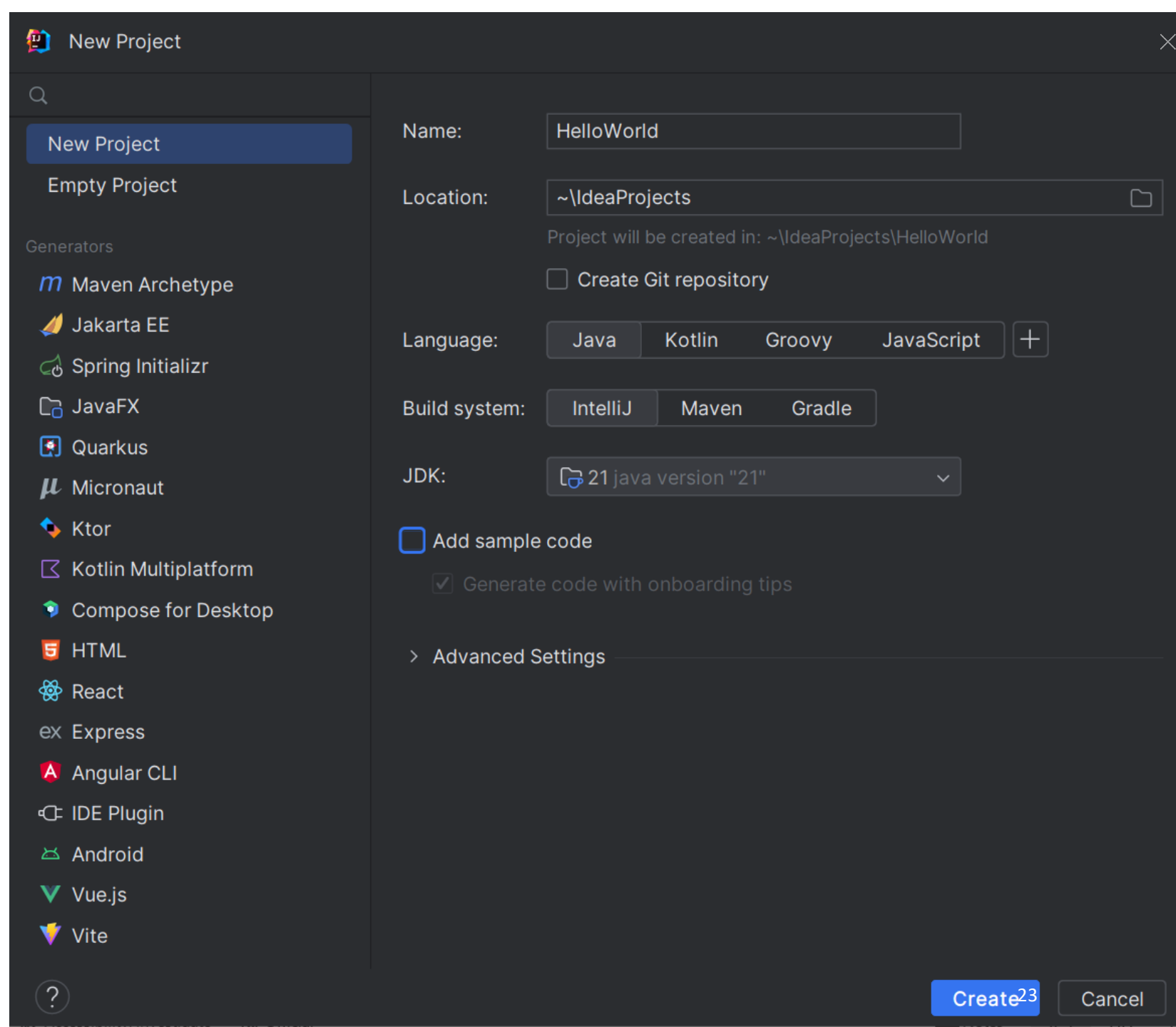


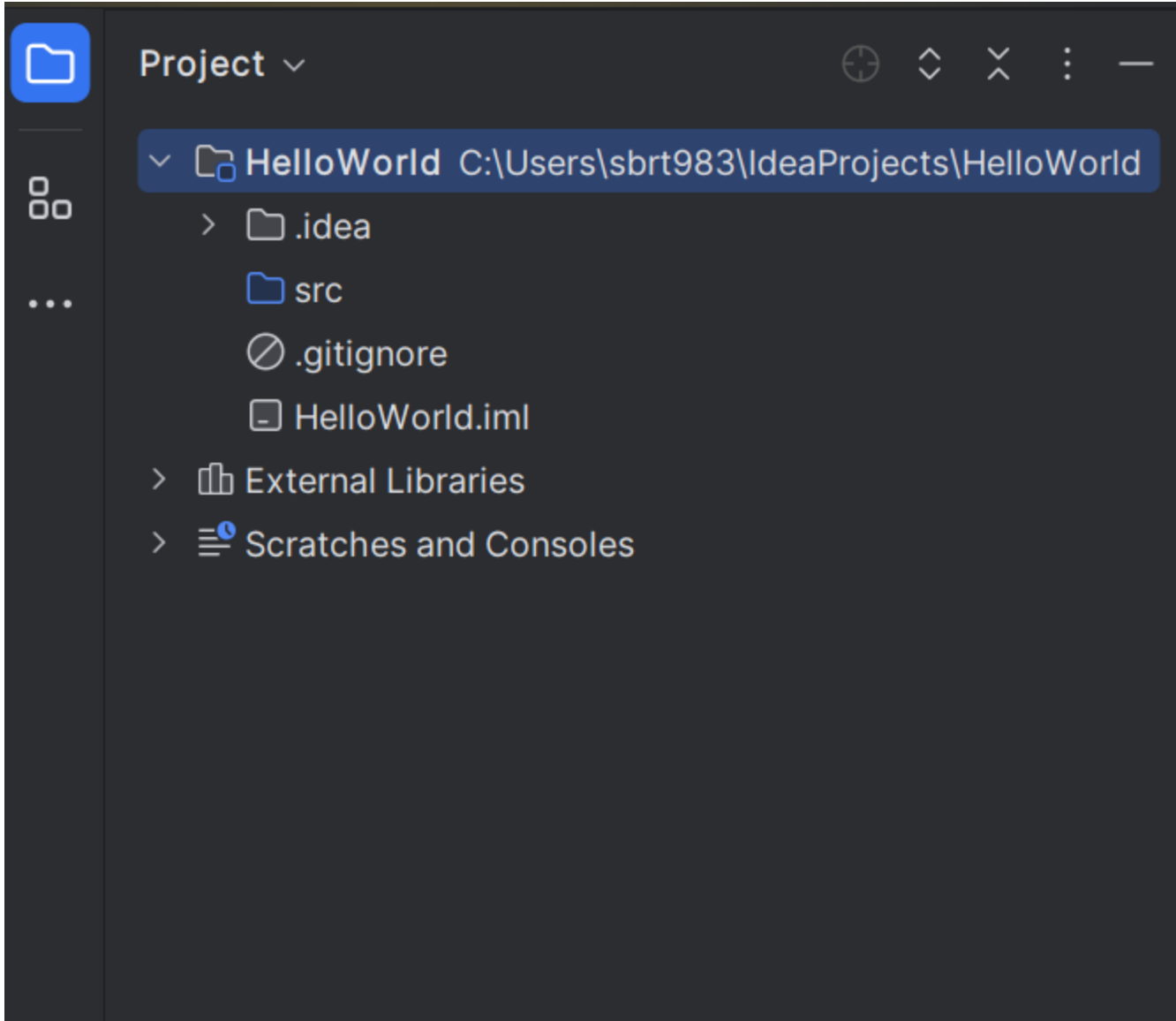
Your Process for Writing a Java Program

1. Think about the problem
2. Design the *components* of a solution
- 3. Write code**
4. Test code
5. Correct / debug code
6. Look for ways to improve / optimise performance (?)

Let's Write Code!

- Start IntelliJ
- Click **New Project**
- Enter project name: **HelloWorld**
- Uncheck **Add sample code** – so we can write code from scratch
- Click the **Create** button





Search Everywhere Double Shift

Go to File Ctrl+Shift+N

Recent Files Ctrl+E

Navigation Bar Alt+Home

Drop files here to open them

HW

Project

▼ HelloWorld

▼

...

17

54

18

19

20

21

22

23

Slide 23 of 23

New

Ctrl+X

Ctrl+C

Copy Path/Reference...

Ctrl+V

Alt+F7

Ctrl+Shift+F

Ctrl+Shift+R

Analyze

Refactor

Bookmarks

Ctrl+Alt+L

Ctrl+Alt+O

Delete

Ctrl+Shift+F9

Open In

Local History

Repair IDE on File

Reload from Disk

Ctrl+D

F4

Mark Directory as

Java Class

Kotlin Class/File

File

Scratch File

Ctrl+Alt+Shift+Insert

Package

package-info.java

module-info.java

HTML File

Stylesheet

JavaScript File

TypeScript File

package.json

Dockerfile

API HTTP Request

OpenAPI Specification

Resource Bundle

EditorConfig File

Swing UI Designer

Data Source in Path

Current File

Double Shift

me

them

Notes

63%

20:56

15/10/2023

First program: HelloWorld.java

```
public class HelloWorld {  
    public static void main(String[] args)  
    {  
        System.out.println("Hello World!");  
    }  
}
```

First program: HelloWorld.java

Name of your class

```
public class HelloWorld {  
    public static void main(String[] args)  
    {  
        System.out.println("Hello World!");  
    }  
}
```

First program: HelloWorld.java

Name of method

```
public class HelloWorld {  
    public static void main(String[] args)  
    {  
        System.out.println("Hello World!");  
    }  
}
```

First program: HelloWorld.java

Method parameters

```
public class HelloWorld {  
    public static void main(String[] args)  
    {  
        System.out.println("Hello World!");  
    }  
}
```

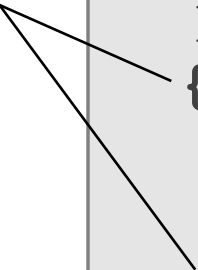
First program: HelloWorld.java

Beginning and end of class

```
public class HelloWorld {  
    public static void main(String[] args)  
    {  
        System.out.println("Hello World!");  
    }  
}
```

First program: HelloWorld.java

Beginning
and end of
method



```
public class HelloWorld {  
    public static void main(String[] args)  
    {  
        System.out.println("Hello World!");  
    }  
}
```

First program: HelloWorld.java

```
public class HelloWorld {  
    public static void main(String[] args)  
    {  
        System.out.println("Hello World!");  
    }  
}
```

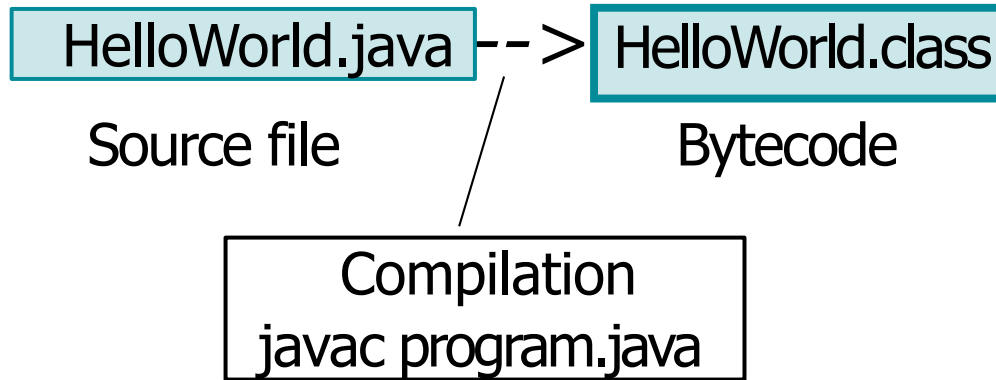
Instruction to print a message: **Hello World!**

I have written a program in Java, and now what...?

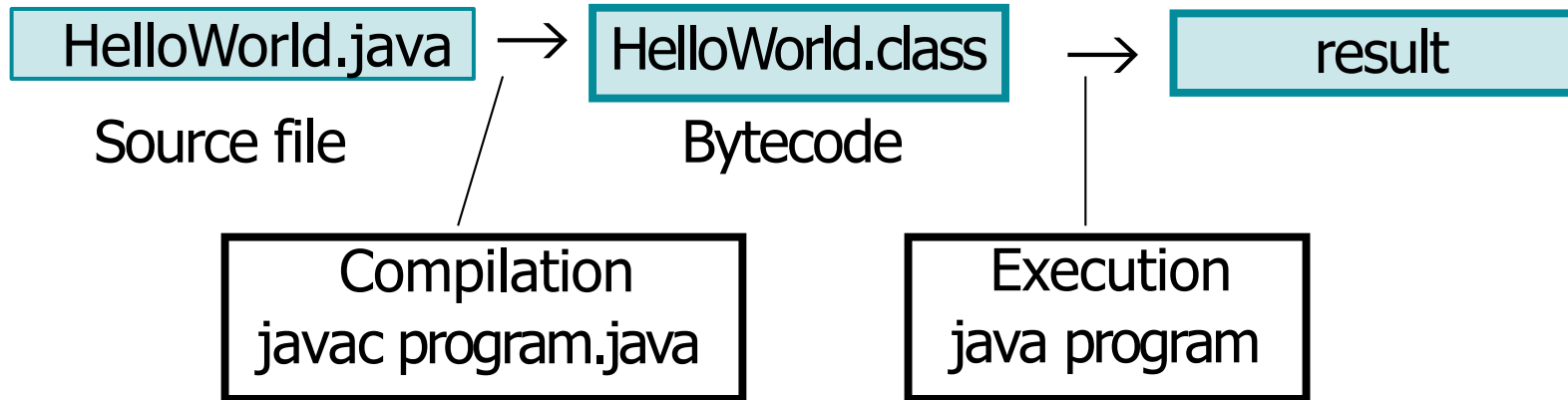
HelloWorld.java

Source file

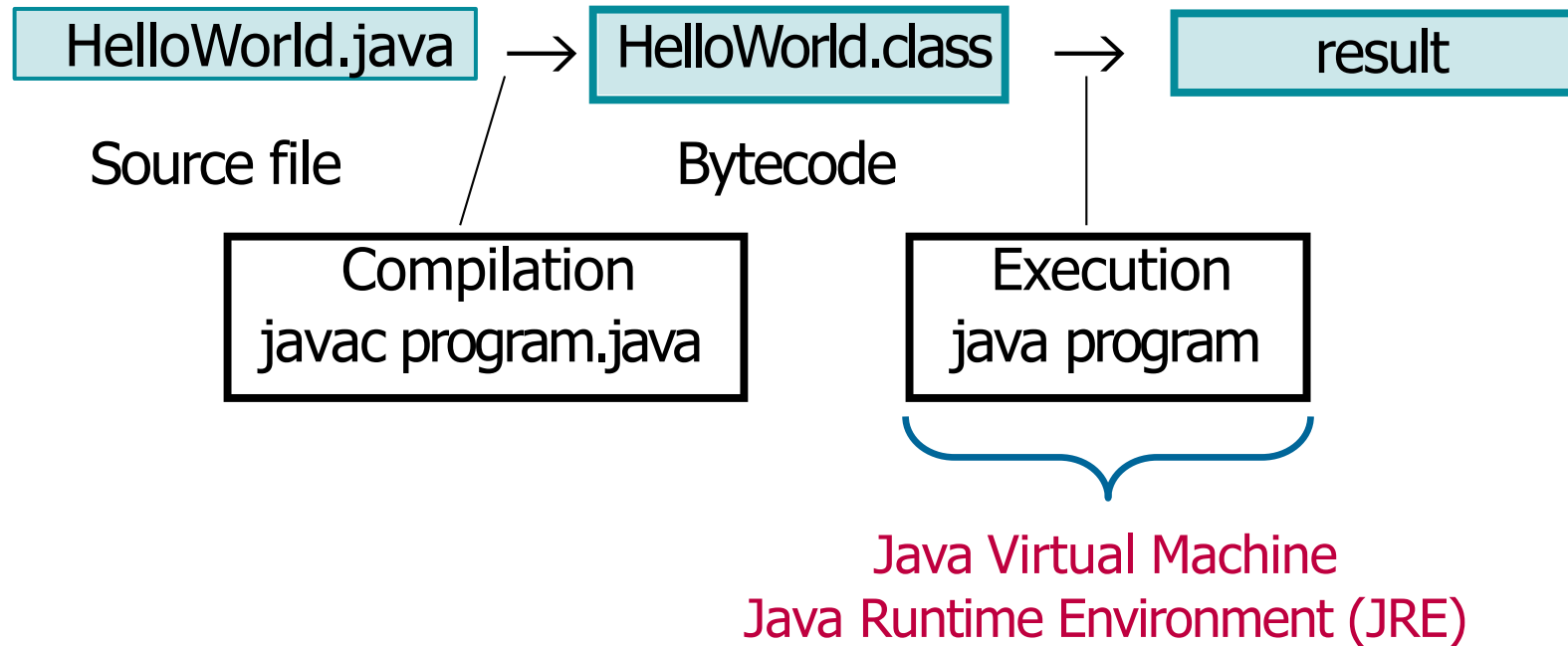
I have written a program in Java, and now what...?



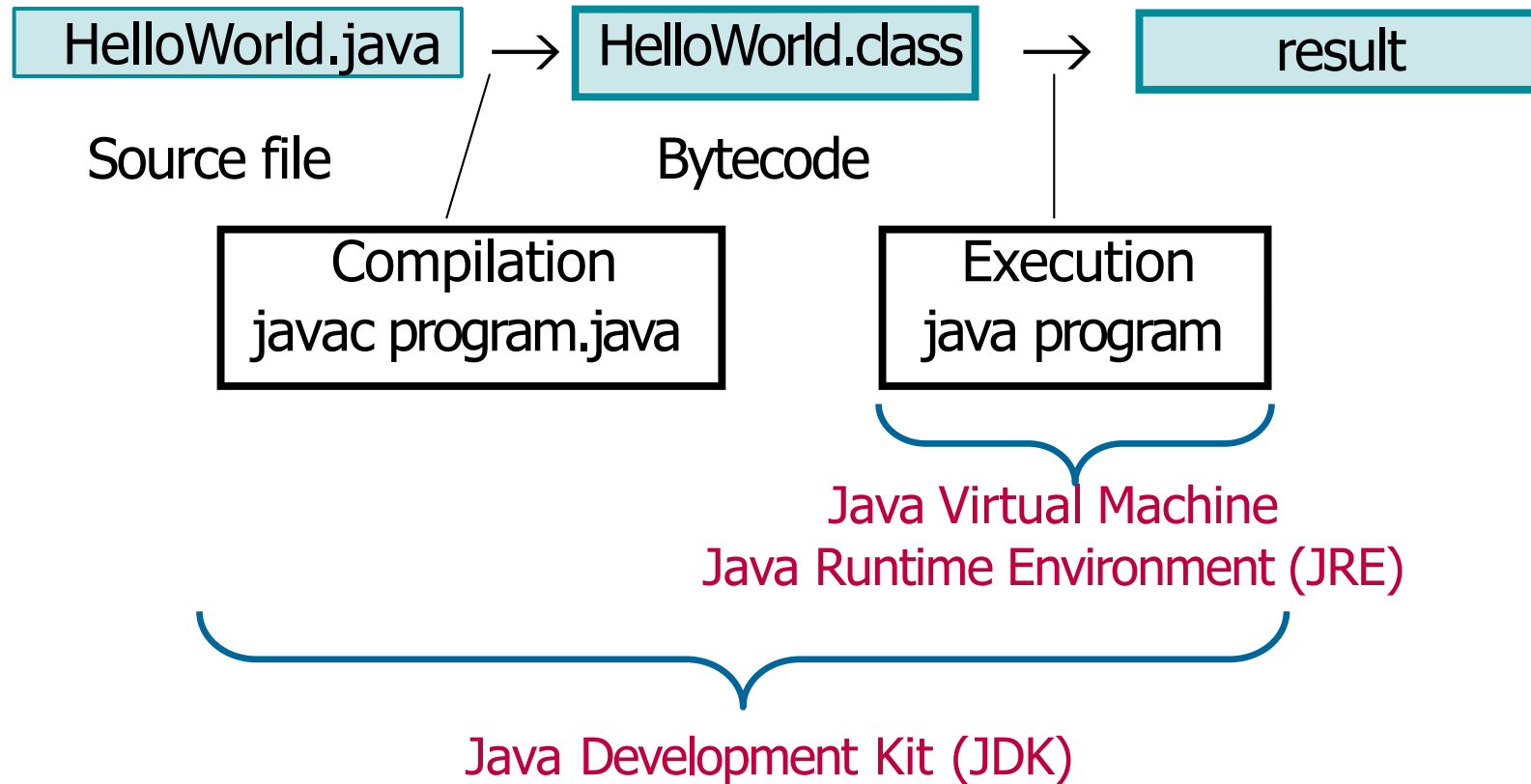
I have written a program in Java, and now what...?



I have written a program in Java, and now what...?



I have written a program in Java, and now what...?



Demo:

Using IntelliJ versus command prompt to compile HelloWorld.

- Where does each of them store the HelloWorld.class file?
- What happens when we change the output string?

Class Exercise: Discuss with 1 person & Vote

What does the following program display?

```
public class Number {  
    public static void main(String[] args) {  
        int number = 2;  
        number = 3 + number;  
        // System.out.print(number);  
        System.out.print(3 + number);  
        System.out.print(number);  
        System.out.print("number");  
    }  
}
```

A - 585number

B - 855

C - 52number

D - 3numbernumbernumber

E - 85number

F - 88number

[Java Lecture Quiz 15-10-24 | Present mode \(sli.do\)](#)

<https://tinyurl.com/3rps9pw3>

Today's Lecture

- Module information
- What is Java?
- Our first Java program: HelloWorld
- **Some core concepts:**
 - Variables
 - Arrays
 - Naming conventions
 - Comments & documentation

Variables

- . To use a new variable: declare its type and name.

```
int number;
```

- . To assign a value:

```
number = 2;
```

- . Or at the same time:

```
int number = 2;
```

Primitive Data Types

- . byte: 8-bit, signed, $[-128; 127]$
 - . short: 16-bit, signed, $[-32767; 32768]$
 - . int: 32-bit, signed, $[-2^{31}; 2^{31} - 1]$ or $[0; 2^{32}]$
 - . long: 64-bit, signed, $[-2^{63}; 2^{63} - 1]$ or $[0; 2^{64}]$
 - . float: 32-bit, floating point
 - . double: 64-bit, floating point
 - . boolean: true or false
 - . char: 16-bit Unicode character
- String

[Unsigned char \(8-bit\) Converter \(binaryconvert.com\)](https://www.binaryconvert.com/)

Integer literals

- `byte b = 100;`
Default value: 0
- `short number = 1000;`
Default value: 0
- `int number = 100000;`
Default value: 0
- `long number = 512839L;`
Default value: 0L

Other number systems

The number 26 in:

- decimal: `int decVal = 26;`
- hexadecimal: `int hexVal = 0x1A;`
- binary: `int binVal = 0b11010;`

Floating-point literals

- `float number = 123.4f;`

Default value: 0.0f

- `double number = 123.4;`

- `double number = 1.234e2;`

Default value: 0.0

Arithmetic Operations

Binary operators

- . + Additive operator
- . – Subtraction operator
- . * Multiplication operator
- . / Division operator
- . % Remainder operator

Unary operators

- . ++ Increments a value by 1
- . -- Decrements a value by 1

Character and String literals

```
char letter = 'a';
```

```
char letter = '\u0061';
```

?

```
char letter = '\u0108';
```

```
Default value: char letter = '\u0000'
```

```
String word = "something";
```

```
String spanishSiSenor = "S\u00ED Se\u00F1or";
```

```
Default value: null
```

Operators on strings: + concatenates two strings

Character and String Literals

Some special characters:

- `\b` (backspace)
- `\n` (line feed)
- `\r` (carriage return)
- `\'` (single quote)
- `\t` (tab)
- `\f` (form feed)
- `\"` (double quote)
- `\\` (backslash)

PI Computations Example

What does the following program display?

```
public class Computations {  
    public static void main(String[] args){  
        int a = 2;  
        double b = Math.PI;  
        double c = a + b;  
        System.out.print(a); System.out.print(",");  
        System.out.print(b); System.out.print(",");  
        System.out.print(c);  
    }  
}
```

A - 2,PI,2+PI

C - 2,3.141592653589793,5

B - 2,3.14,5.14

D - 2,3.141592653589793,5.141592653589793

Display String Example

What does the following program display?

```
public class Display {  
    public static void main(String[] args){  
        int a = 2;  
        String s = "java is cool";  
        System.out.print(a + a + s);  
        System.out.print("a" + "a");  
        System.out.print(a + "a");  
    }  
}
```

A - 4java is coolaa2a

B - 22java is coolaa2a

C - aaajava is coolaaaa

D - aaajava is coolaa2a

Boolean Data Type and Boolean Operators:

- . `boolean b = true;`
`boolean b = false;`
Default value: false

Boolean operators:

- . AND → `&&`
- . OR → `||`
- . NOT → `!`

<code>&&</code>	true	false
true	true	false
false	false	false

<code> </code>	true	false
true	true	true
false	true	false

<code>!</code>	true	false
	false	true

TrueAndFalse Example (Discuss!)

What does the following program display?

```
public class TrueAndFalse {  
    public static void main(String[] args){  
        boolean b = true;  
        boolean c = false;  
        System.out.print(b && c);  
        System.out.print((b && c) || (!c));  
    }  
}
```

A - true true
B - true false

C - false true
D - false false

Relational Operators

- . == Equal
- . != Not equal
- . < Less than
- . > Greater than
- . <= Less than or equal to
- . >= Greater than or equal to

- . Condition ? Value1 : Value2
→ *If Condition is true, takes the value Value1 otherwise takes the value Value2.*

Operators: (discuss!)

What does the following program display?

```
public class Operators {  
    public static void main(String[] args){  
        String s = "java";  
        String t = "java";  
        System.out.print(0.1 + 0.2 == 0.3);  
        System.out.print(s == t);  
        System.out.print((!true) ? 2 : 3);  
    }  
}
```

A - true true 2

B - true true 3

C - false true 2

D - false true 3

E - false false 2

F - false false 3

Casting: Conversion from a primitive type to another.

→ Automatic:

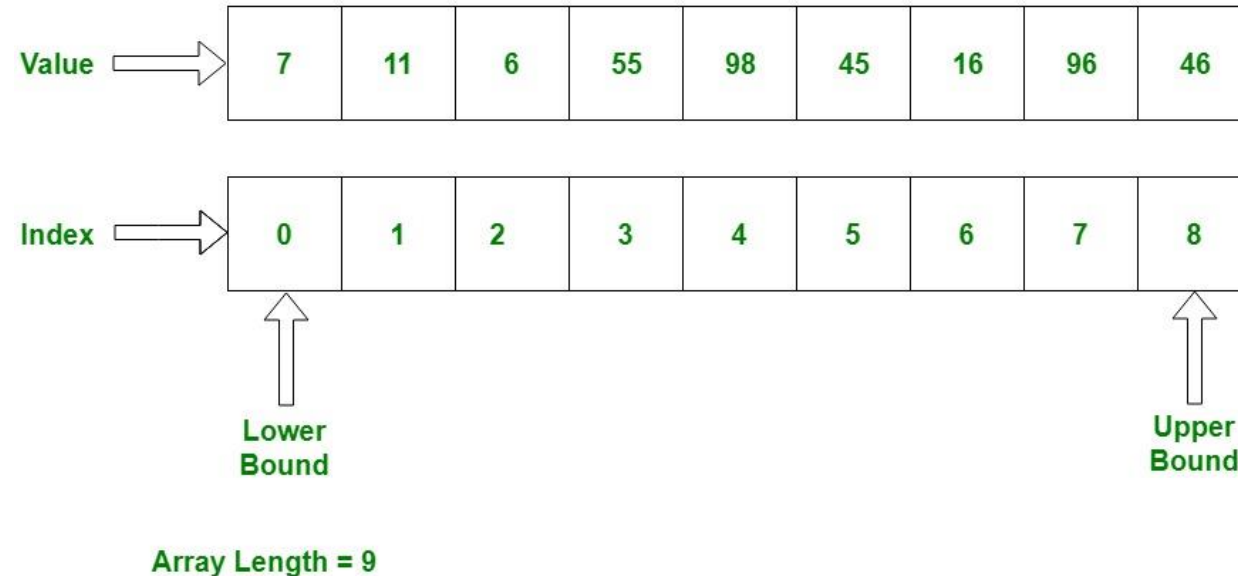
byte → short → char → int → long → float → double

→ Non automatic: the other way around

```
double d = 3.4;  
int i = (int) d;
```

Arrays

- Arrays are stored in consecutive memory locations in main memory.
- We can pass entire arrays to methods
- We can access the individual array elements.
- We can process the individual elements like other simple variables.



Declaring arrays

Array type Array name Array size

↓ ↓ ↘

```
double[] salary = new double[50];
```



Declares an array of type double that can hold 50 values

Q: Why might we need this to be an array of type “double”?
What does this tell us about the values stored in each element?

Declaring arrays

Array type Array name Array size

↓ ↓ ↓

```
double[] salaries = new double[50];
```



Alternatively, one can initialise the array with numbers:

```
double[] salaries = {56.0, 210.5, ... };
```

Array Example

```
public class Array {  
    public static void main(String[] args){  
        int[] anArray = {  
            1, 2, 3  
        };  
        System.out.print(anArray[2] + " ");  
        System.out.print(anArray + " ");  
        System.out.print(anArray[3]);  
    }  
}
```

- . Get the length of an array

`anArray.length`

```
public class MultiDimArrayDemo {  
    public static void main(String[] args){  
        String[][] tab = {  
            {"a", "b", "c"},  
            {"d", "e"},  
            {"f", "g", "h"}  
        };  
        System.out.print(tab[0][0] + tab[1][0]);  
        System.out.print(tab[0][2] + tab[1][1]);  
    }  
}
```

2D Array

See if you can find a way to get the number of rows and columns in a 2D array.

Conventions (1/4)

Identifiers Type	Naming Rules	Examples
Class	<p>It should start with the uppercase letter.</p> <p>It should be a noun such as Color, Button, System, Thread, etc.</p> <p>Use appropriate words, instead of acronyms.</p>	<pre>public class Employee { //code snippet }</pre>

[Java Naming Conventions - Javatpoint](#)

Conventions (2/4)

Method	<p>It should start with lowercase letter.</p> <p>It should be a verb such as main(), print(), println().</p> <p>If the name contains multiple words, start it with a lowercase letter followed by an uppercase letter such as actionPerformed().</p>	<pre>class Employee { // method void draw() { //code snippet } }</pre>
--------	--	---

[Java Naming Conventions - Javatpoint](#)

Conventions (3/4)

Variable	<p>It should start with a lowercase letter such as id, name.</p> <p>It should not start with the special characters like & (ampersand), \$ (dollar), _ (underscore).</p> <p>If the name contains multiple words, start it with the lowercase letter followed by an uppercase letter such as firstName, lastName.</p> <p>Avoid using one-character variables such as x, y, z.</p>	<pre>class Employee { // variable int id; //code snippet }</pre>
----------	--	---

[Java Naming Conventions - Javatpoint](#)

Conventions (4/4)

Constant	<p>It should be in uppercase letters such as RED, YELLOW.</p> <p>If the name contains multiple words, it should be separated by an underscore(_) such as MAX_PRIORITY.</p> <p>It may contain digits but not as the first letter.</p>	<pre>class Employee { //constant static final int MIN_AGE = 18; //code snippet }</pre>
----------	--	---

[Java Naming Conventions - Javatpoint](#)

Comments

Comments

```
// comment until the end of the line  
/* comment */
```

Documentation comments

```
/** documentation */
```

Ignored by the compiler, used by the javadoc tool

Why commenting your code?

- . for yourself: remember your code when coming back to it later
- . for your colleagues: help them understand and use your code
- . for me! when I will have to mark your assessment

Today's Lecture

- **Module information**
- **What is Java?**
- **Our first Java program: HelloWorld**
- **Some core concepts**