جامعة الجلالة
GALALA UNIVERSITY

CSE110: Principles of Programming

Lecture 1: What is Java?

Professor: Shaker El-Sappagh

Shaker.elsappagh@gu.edu.eg

Fall 2023

Outline

1. Welcome and introduce yourself.
2. The interactive learning tools:
   - CANVAS
   - ZyBooks
   - Java how to program
   - HackerRank
3. The syllabus of the course
4. The used software
5. Computer systems
6. Java programming language

1. Welcome and introduce yourself

2. The interactive learning tools:

- CANVAS
- ZyBooks
- Java how to program
- HackerRank

3. The used software

Java JDK

Java IDE

https://www.educative.io/blog/best-java-ides-2021

4. The syllabus of the course

5. Computer systems

5. Computer systems

Computers: Hardware and Software

- Computer—Device that can perform computations and make logical decisions phenomenally faster than human beings *can*.
- Today's personal computers can perform billions of calculations in one second—more than a human can perform in a lifetime.
- *Supercomputers* are already performing thousands of trillions (quadrillions) of instructions per second!
- Computers process data under the control of sets of instructions called computer programs.
- **Programs** guide the computer through orderly sets of CPU instructions specified by people called programmers to fulfill a task.
- The programs that run on a computer are referred to as software.

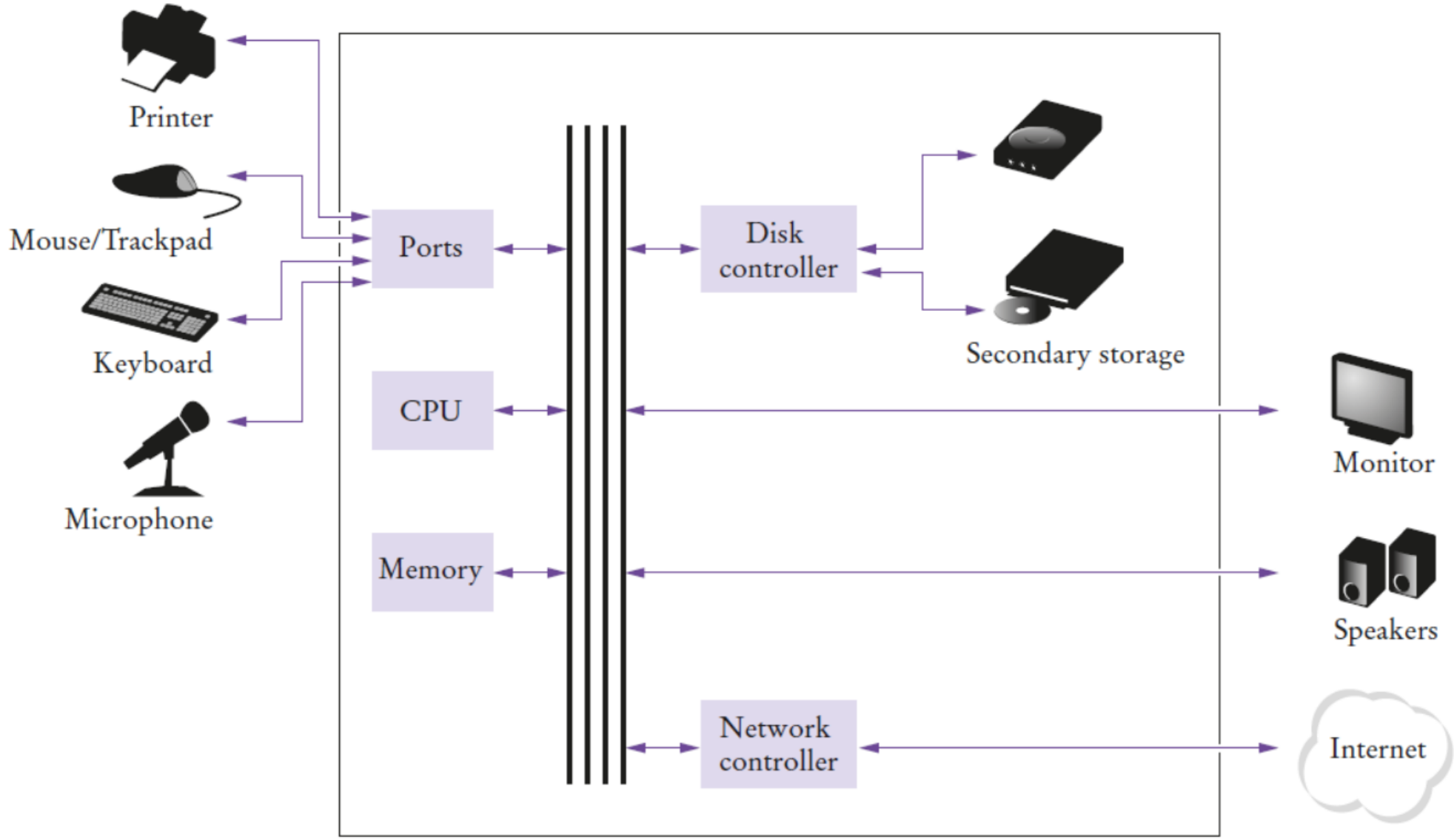
5. Computer systems

Computers: Hardware and Software (Cont.)

- You'll learn today's **key programming methodology** that's enhancing programmer productivity, thereby reducing software-development costs—*object-oriented programming*.
- A computer consists of various devices referred to as hardware
  - (e.g., the keyboard, screen, mouse, hard disks, memory, DVDs and processing units).
- Computing costs are dropping dramatically, owing to rapid developments in hardware and software technologies.
- Computers that might have filled large rooms and cost millions of dollars decades ago are now inscribed on silicon chips smaller than a fingernail, costing perhaps a few dollars each.
- Moore's (*co-founder of Intel*) Law
  - For many decades, hardware costs have fallen rapidly.
  - Every year or two, the capacities of computers have approximately *doubled without any increase in price*.

5. Computer systems

Schematic Design of a Personal Computer

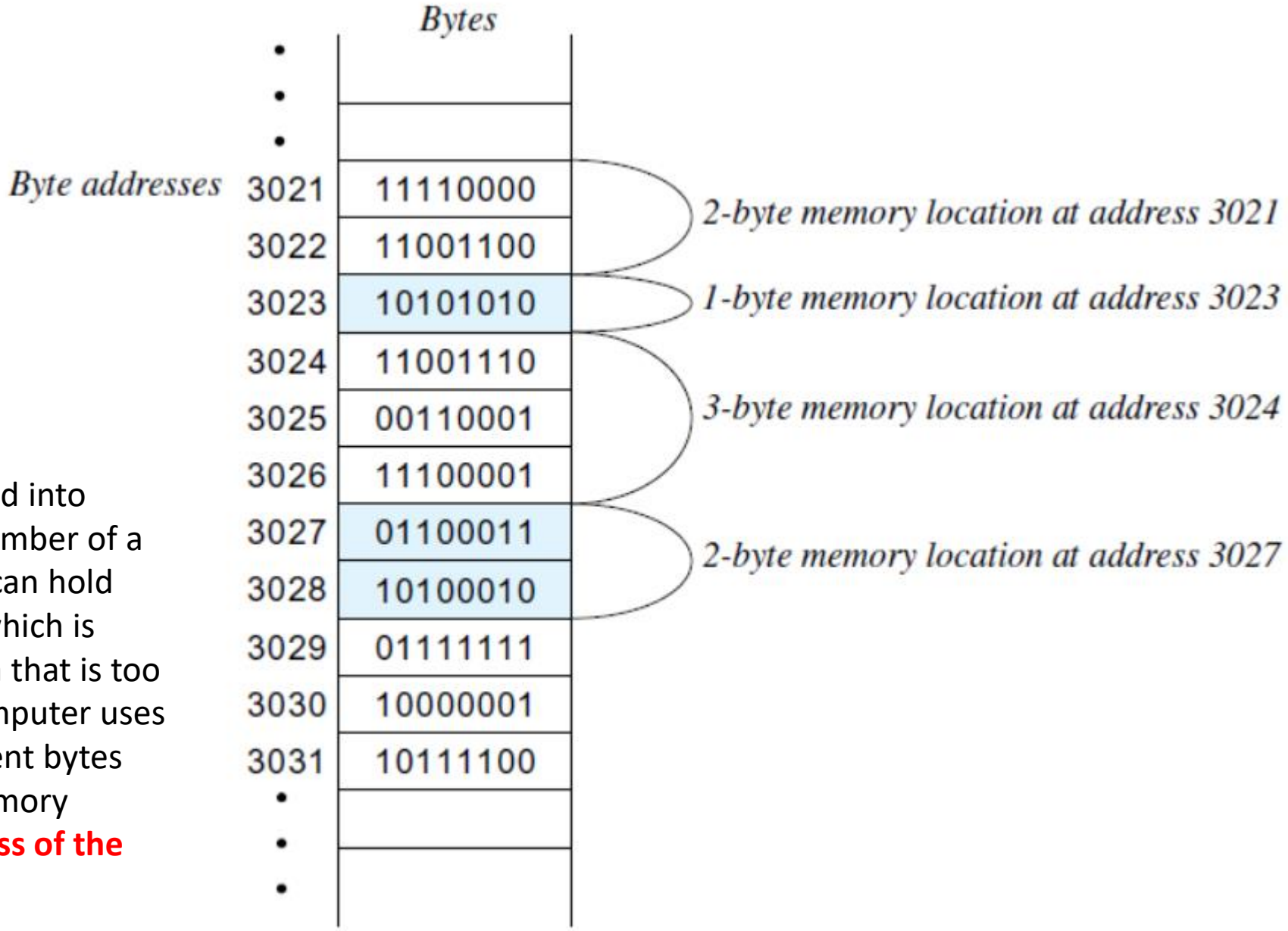


5. Computer systems

Main memory

1. Main memory consists of addressable eight-bit bytes.
2. Groups of adjacent bytes can serve as a single memory location.

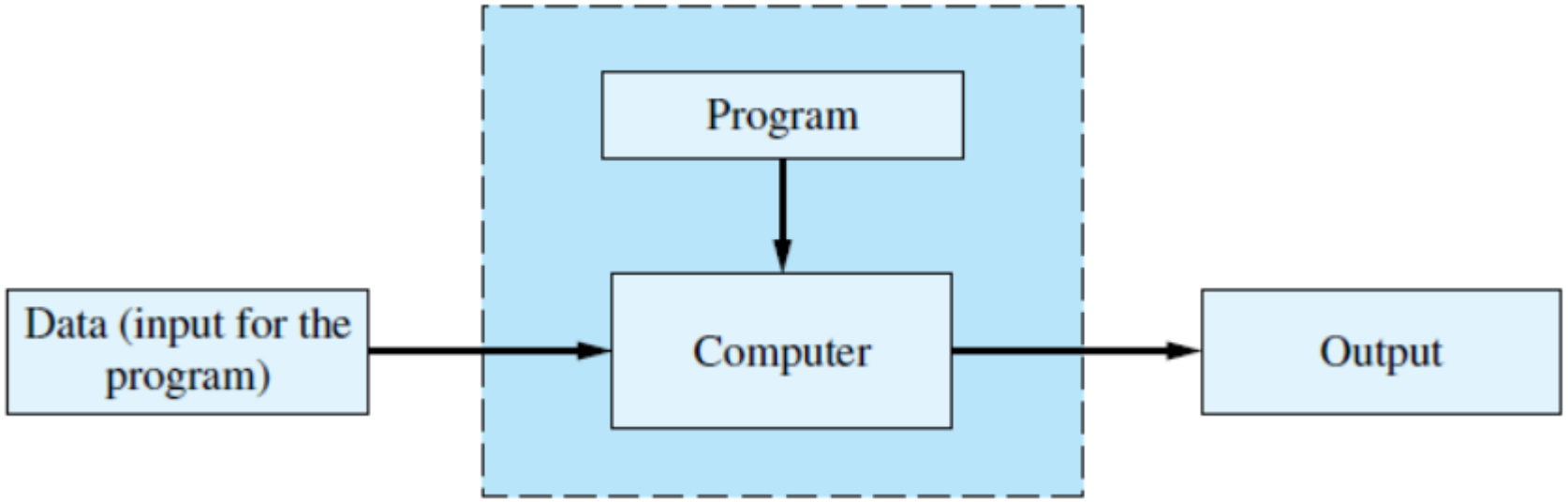
A computer's main memory is divided into numbered units called **bytes**. The number of a byte is called its **address**. Each byte can hold **eight binary digits**, or bits, each of which is either 0 or 1. To store a piece of data that is too large to fit into a single byte, the computer uses **several adjacent bytes**. These adjacent bytes are thought of as a single, larger memory location **whose address is the address of the first of the adjacent bytes**.



5. Computer systems

Program

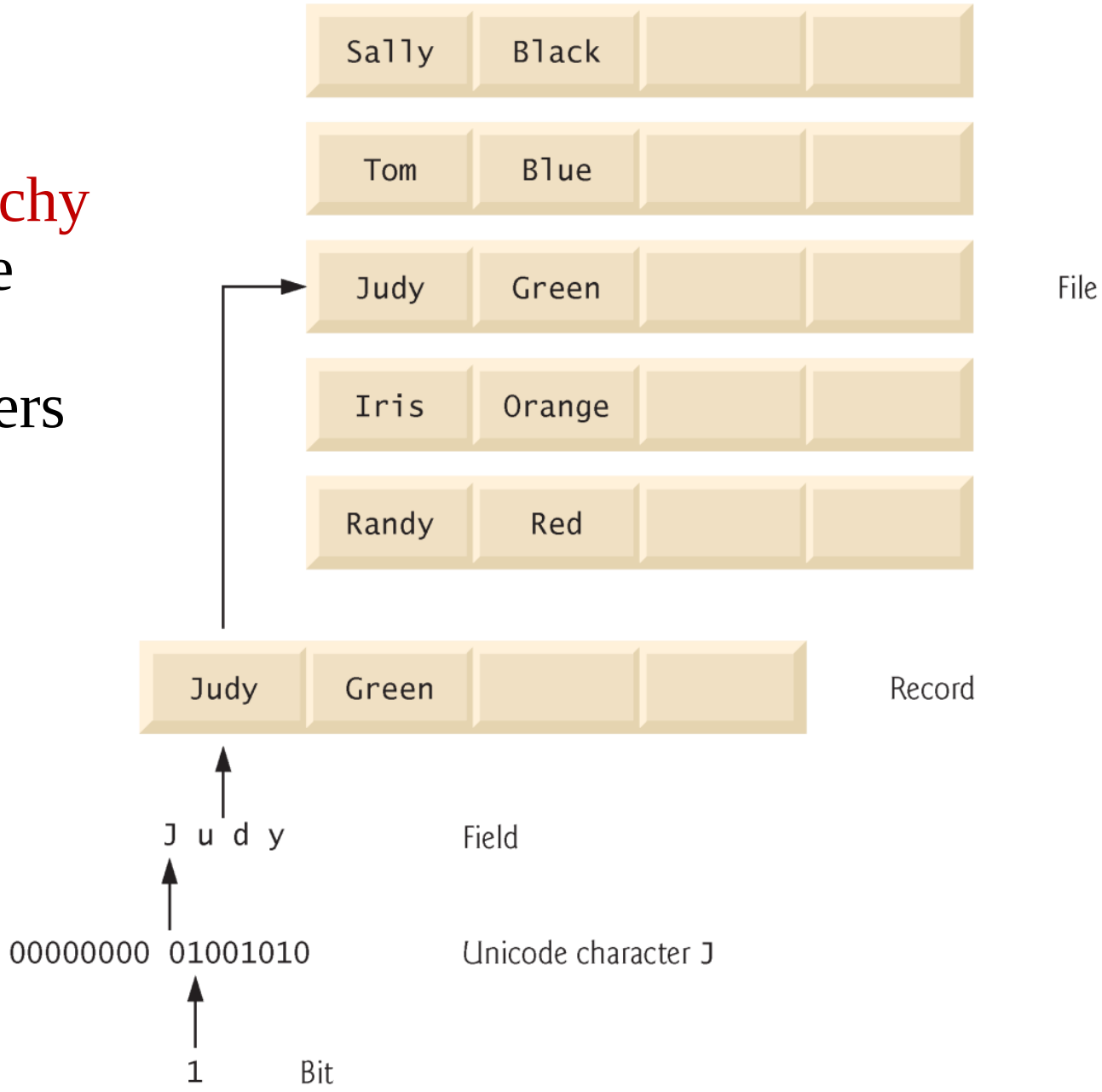
A program is a set of computer instructions written in specific language to perform a specific task.



5. Computer systems

Data Hierarchy

- Data items processed by computers form a data hierarchy that becomes larger and more complex in structure as we progress from bits to characters to fields, and so on.



5. Computer systems

Machine Languages, Assembly Languages and High-Level Languages

- Programmers write instructions in **various programming languages**, some directly understandable by computers and others requiring intermediate translation steps.
- These may be divided into three general types:
  - **Machine languages**
  - **Assembly languages**
  - **High-level languages**

5. Computer systems

Machine Languages, Assembly Languages and High-Level Languages

- Any computer can directly understand only its own machine language, *defined by its hardware design.*
  - Generally, consist of strings of numbers (ultimately reduced to 1s and 0s) that instruct computers to perform their most elementary operations one at a time.
  - Machine dependent—a particular machine language can be used on only one type of computer.

- English-like abbreviations that represent elementary operations formed the basis of assembly languages.
- Translator programs called assemblers convert early assembly-language programs to machine language.

5. Computer systems

Machine Languages, Assembly Languages and High-Level Languages

- High-level languages
  - Single statements accomplish substantial **tasks**.
  - Compilers convert high-level language programs into machine language.
  - Allow you to write instructions that look almost like **everyday English** and contain commonly used mathematical notations.
- Compiling a high-level language program into machine language can take a considerable amount of computer time. This is done by compiler program.
- **Interpreter** translates program statements from a high-level language to a low-level language. But unlike a compiler, an interpreter executes a portion of code right after translating it, rather than translating the entire program at once. Using an interpreter means that when you run a program, translation alternates with execution. Moreover, translation is done each time you run the program. Recall that compilation is done once, and the resulting object program can be run over and over again without engaging the compiler again. **This implies that a compiled program generally runs faster than an interpreted one.**

6. Java programming language

6. Java programming language

- Java is the world's **most widely used** computer programming language.
- You'll learn to **write instructions** commanding computers to perform tasks.
- ***Software*** (i.e., the instructions you write) controls **hardware** (i.e., computers).
- You'll learn ***object-oriented*** **programming**—today's key programming methodology.
- Java is the **preferred language** for meeting many organizations' enterprise programming needs.
- Java has become the language of choice for implementing **Internet-based applications** and **software for devices** that communicate over a network.
- In use today are **billions** of general-purpose computers and billions more Java-enabled cell phones, smartphones and handheld devices (such as tablet computers).

6. Java programming language

- Java Editions: SE, EE and ME
  - **SE:** Used for developing cross-platform, general-purpose applications.
  - **EE:** Geared toward developing large-scale, distributed networking applications and web-based applications.
  - **ME:** geared toward developing applications for small, memory-constrained devices, such as smartphones. Google’s Android operating system.

6. Java programming language

Java apps

- Standalone applications
  - Console Applications
  - Swing Applications (GUI)
- Web applications
  - Applet
  - Servlet
  - JSP
- Mobile applications
  - J2ME Applications
  - Android Applications
- Distributed Applications
  - Enterprise Java Beans (EJB Technology)
- Embedded Systems Applications
- Java Card Applications
  - smart card programing

6. Java programming language

Java and a Typical Java Development Environment

- 1991
  - Recognizing this, **Sun Microsystems** funded an internal corporate research project led by James Gosling, which resulted in a C++-based object-oriented programming language Sun called Java.
  - Key goal of Java is to be able to write programs that will run on a great variety of computer systems and computer-control devices.
  - This is sometimes called "**write once, run anywhere**."

6. Java programming language

Java and a Typical Java Development Environment

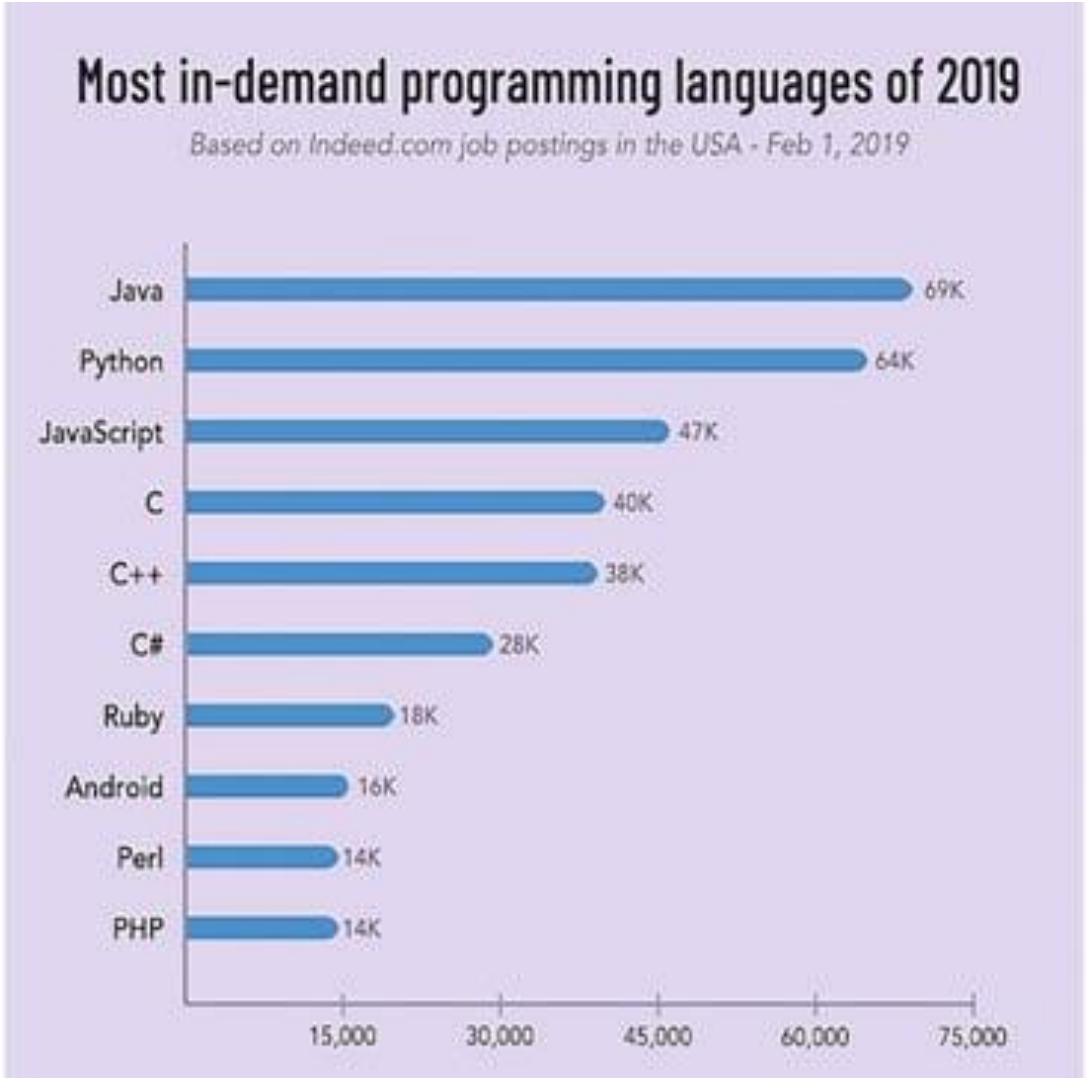
- 1993
  - The **web** exploded in popularity
  - Sun saw the potential of using Java to add **dynamic content to web pages**.
- Java got the attention of the business community because of the phenomenal **interest in the web**.
- Java is used to develop large-scale enterprise applications, to enhance the functionality of **web servers**, to provide applications for consumer devices and for many other purposes.

6. Java programming language

Java and a Typical Java Development Environment

- Sun Microsystems was acquired by **Oracle in 2009**.
- Java is the most widely used software development language in the world.
- Our world moved by Java:

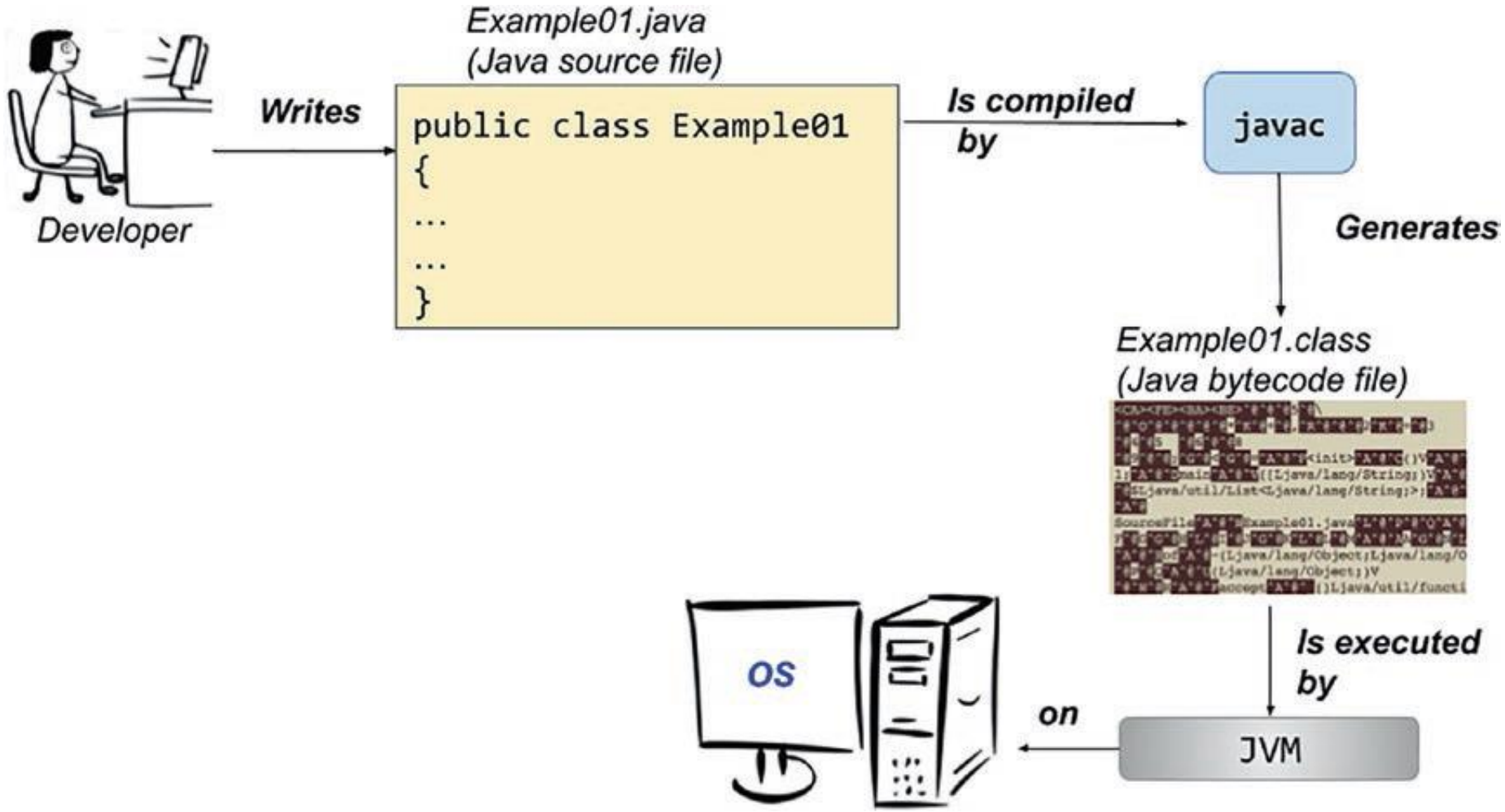
  https://www.oracle.com/java/moved-by-java/timeline/
- Java **object-oriented programming** (OOP) language.
- Java improved program **portability**.
- Java is Simple, Distributed, Robust, Secure, Multithreaded, and Dynamic.
- Java provide **automatic garbage collection**.
- **Java provides huge Class Libraries**
  - Rich collections of existing classes and methods.
  - Also known as the Java APIs (Application Programming Interfaces).



6. Java programming language

Java and a Typical Java Development Environment

- **Java programs normally go through five phases**
  1. Edit
  2. Compile
  3. Load
  4. Verify
  5. Execute



6. Java programming language

Java and a Typical Java Development Environment

- **Phase 1** consists of editing a file with an *I*
  - Type a Java program (source code) using the editor.
  - Make any necessary corrections.
  - Save the program.
  - A file name ending with the .java extension indicates that the file contains Java source code.

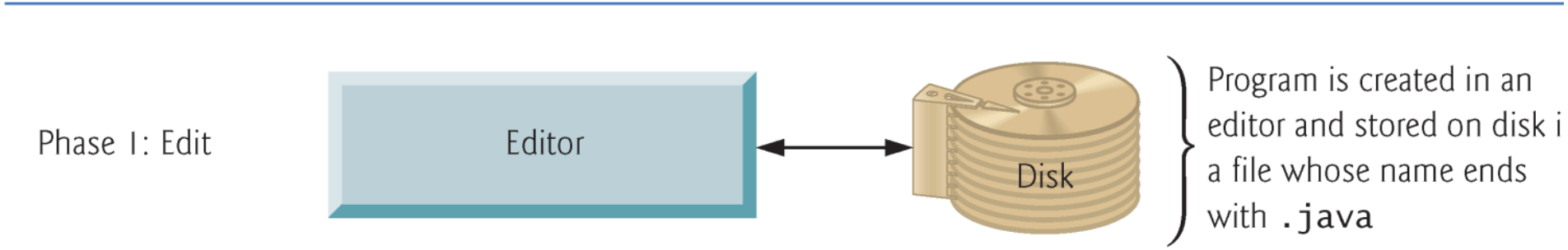


Fig. 1.6 | Typical Java development environment—editing phase

6. Java programming language

Java and a Typical Java Development Environment

- Linux editors: `vi` and `emacs`.
- Windows editors:
  - Notepad
  - EditPlus (`www.editplus.com`)
  - TextPad (`www.textpad.com`)
  - jEdit (`www.jedit.org`).
- Integrated development environments (IDEs)
  - Provide tools that support the software development process, including editors for writing and editing programs and debuggers for locating logic errors—errors that cause programs to execute incorrectly.

6. Java programming language

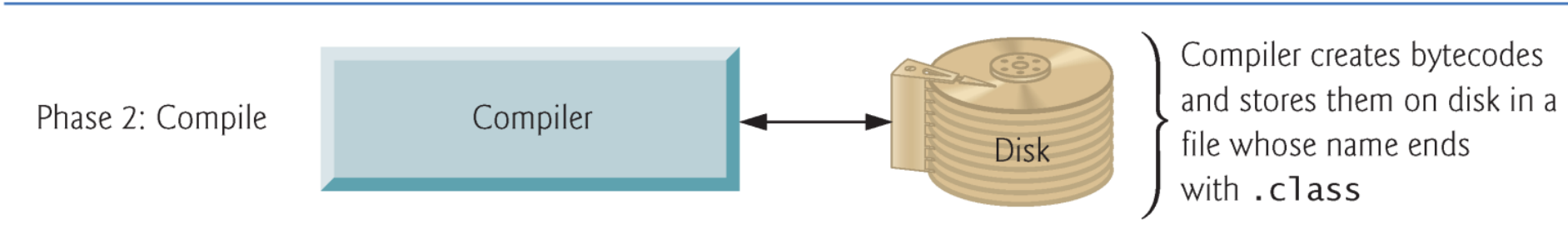
Java and a Typical Java Development Environment

- Popular IDEs
  - Eclipse (`www.eclipse.org`)
  - NetBeans (`www.netbeans.org`).
  - jGRASP™ IDE (`www.jgrasp.org`)
  - DrJava IDE (`www.drjava.org/download.shtml`)
  - BlueJ IDE (`www.bluej.org/`)
  - TextPad® Text Editor for Windows® (`www.textpad.com/`)

6. Java programming language

Java and a Typical Java Development Environment

- **Phase 2**: Compiling a Java Program into Bytecodes
  - Use the command javac (the Java compiler) to compile a program. For example, to compile a program called `Welcome.java`, you'd type
    - `javac Welcome.java`
  - If the program compiles, the compiler produces a .class file called `Welcome.class` that contains the compiled version of the program.



RECAP Compiler

A compiler is a program that translates a program written in a high-level language, such as Java, into a program in a simpler language that the computer can more or less directly understand.

6. Java programming language

Java and a Typical Java Development Environment

- Java compiler translates Java source code into bytecodes that represent the tasks to execute.
- Bytecodes are executed by the Java Virtual Machine (JVM)—a part of the JDK and the foundation of the Java platform.
- Virtual machine (VM)—a software application that simulates a computer
  - Hides the underlying operating system and hardware from the programs that interact with it.
- If the same VM is implemented on many computer platforms, applications that it executes can be used on all those platforms.

6. Java programming language

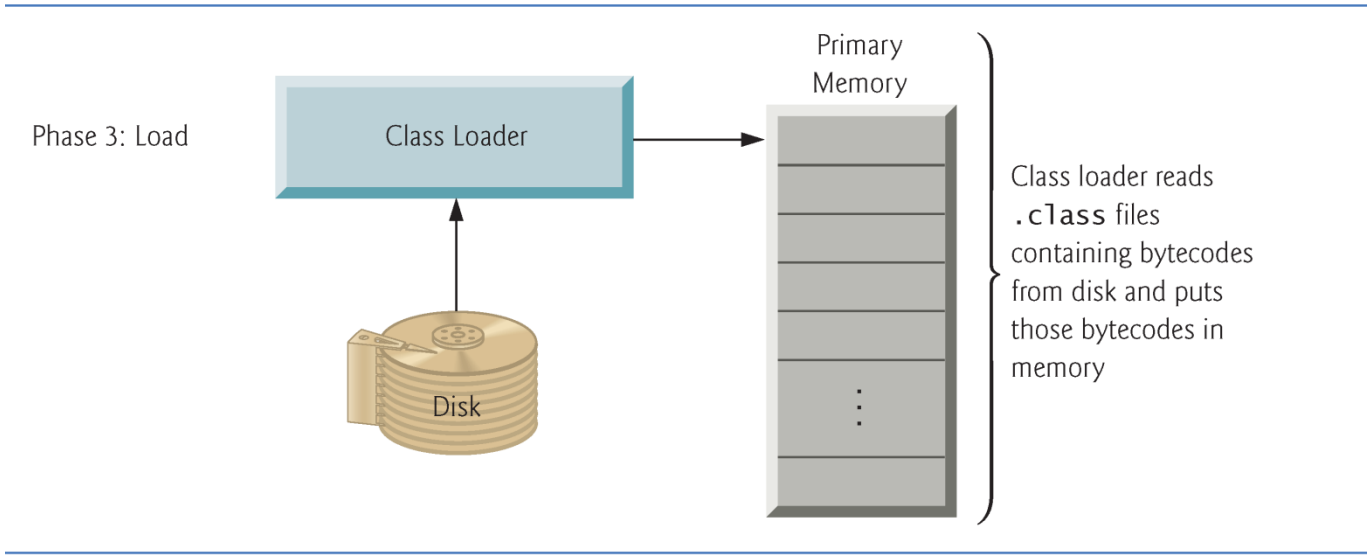
Java and a Typical Java Development Environment

- Bytecodes are platform independent
  - They do not depend on a particular hardware platform.
- Bytecodes are portable
  - The same bytecodes can execute on any platform containing a JVM that understands the version of Java in which the bytecodes were compiled.
- The JVM is invoked by the java command. For example, to execute a Java application called `Welcome`, you'd type the command
    - `java Welcome`

6. Java programming language

Java and a Typical Java Development Environment

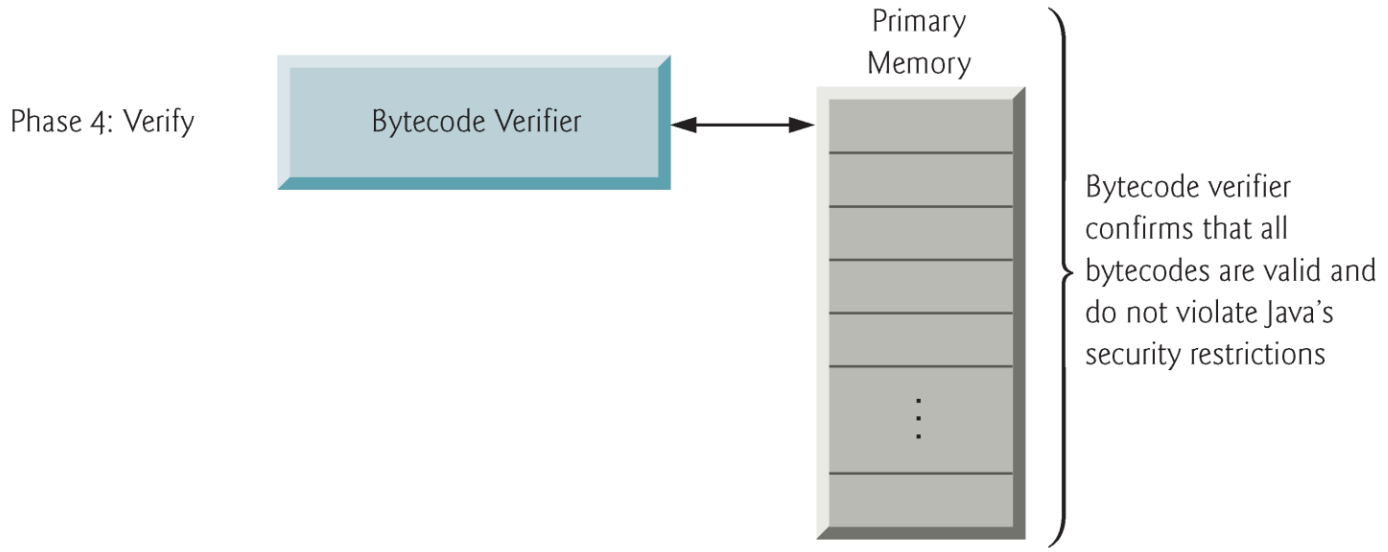
- **Phase 3:** Loading a Program into Memory
  - The JVM places the program in memory to execute it—this is known as loading.
  - Class loader takes the `.class` files containing the program's bytecodes and transfers them to primary memory.
  - Also loads any of the `.class` files provided by Java that your program uses.
- The `.class` files can be loaded from a disk on your system or over a network.



6. Java programming language

Java and a Typical Java Development Environment

- **Phase 4:** Bytecode Verification
  - As the classes are loaded, the bytecode verifier examines their bytecodes
  - Ensures that they're valid and do not violate **Java's security restrictions**.
- Java enforces **strong security** to make sure that Java programs arriving over the network do not damage your files or your system (as computer viruses and worms might).



6. Java programming language

Java and a Typical Java Development Environment

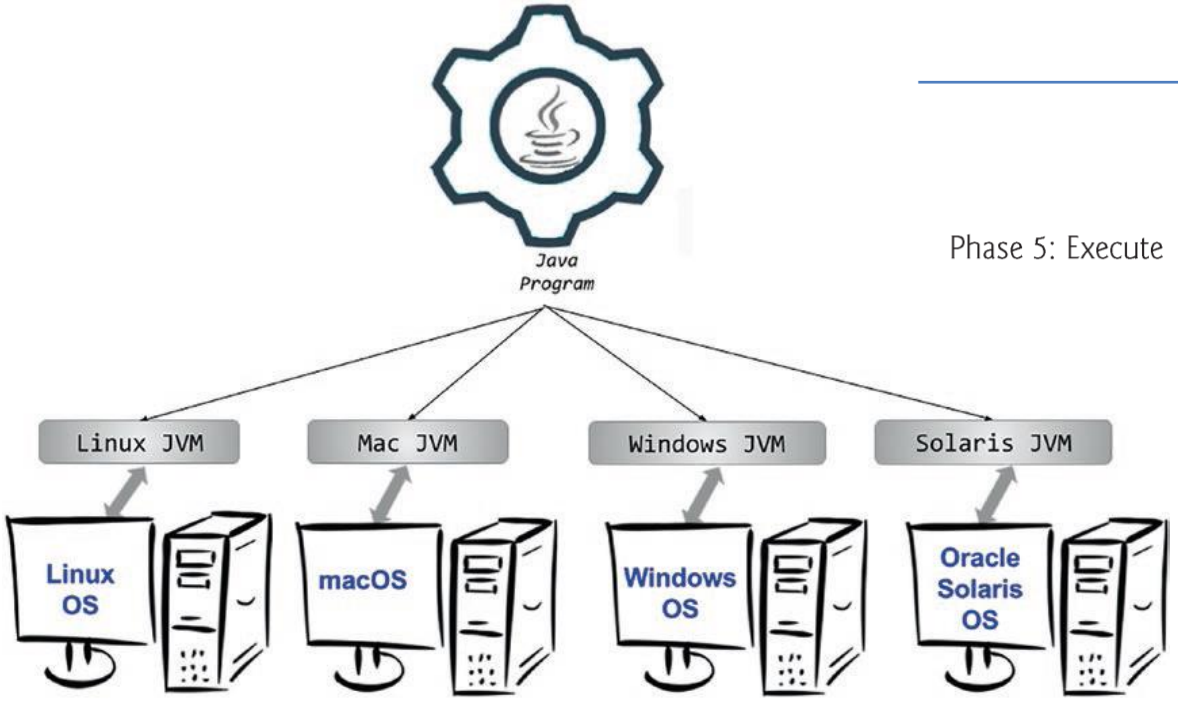
- **Phase 5:** Execution
  - The JVM executes the program's bytecodes.
  - Java programs are distributed as instructions for a virtual machine, making them platform-independent.
  - JVMs typically execute bytecodes using a combination of interpretation and so-called just-in-time (JIT) compilation.
  - Analyzes the bytecodes as they're interpreted
  - A just-in-time (JIT) compiler—known as the Java HotSpot compiler—translates the bytecodes into the underlying computer's **machine language**.

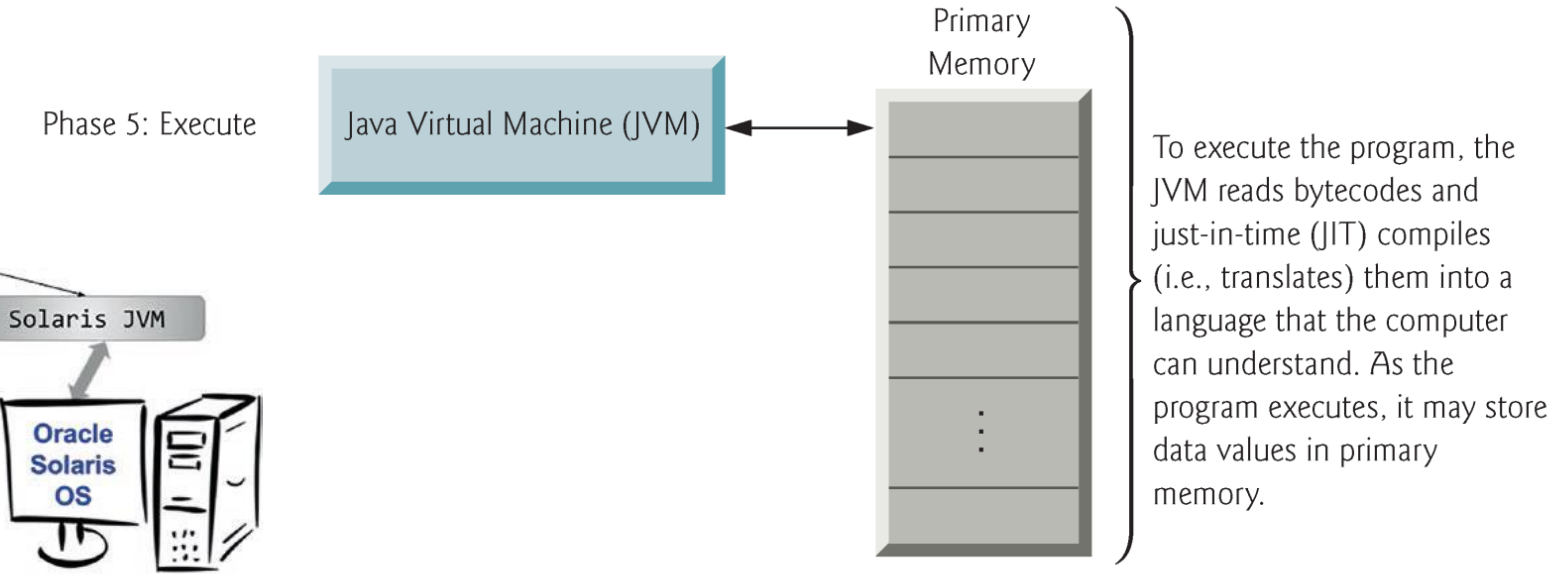
6. Java programming language

Java and a Typical Java Development Environment

- Java programs go through *two compilation phases*

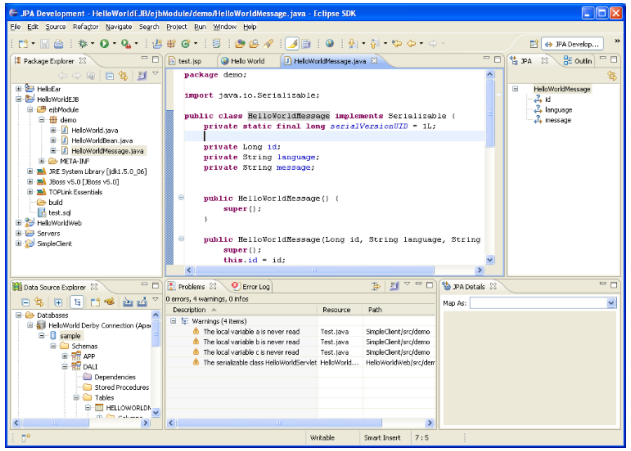
1. One in which source code is translated into bytecodes (for portability across JVMs on different computer platforms) and
2. A second in which, during execution, the bytecodes are translated into machine language for the actual computer on which the program executes.

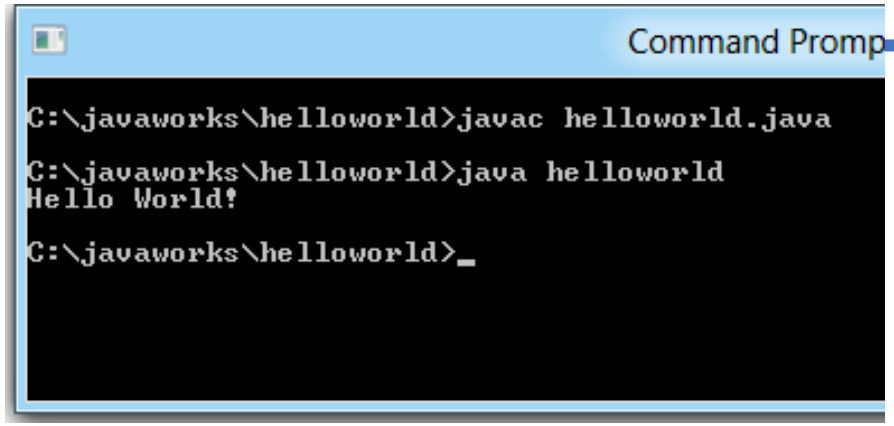


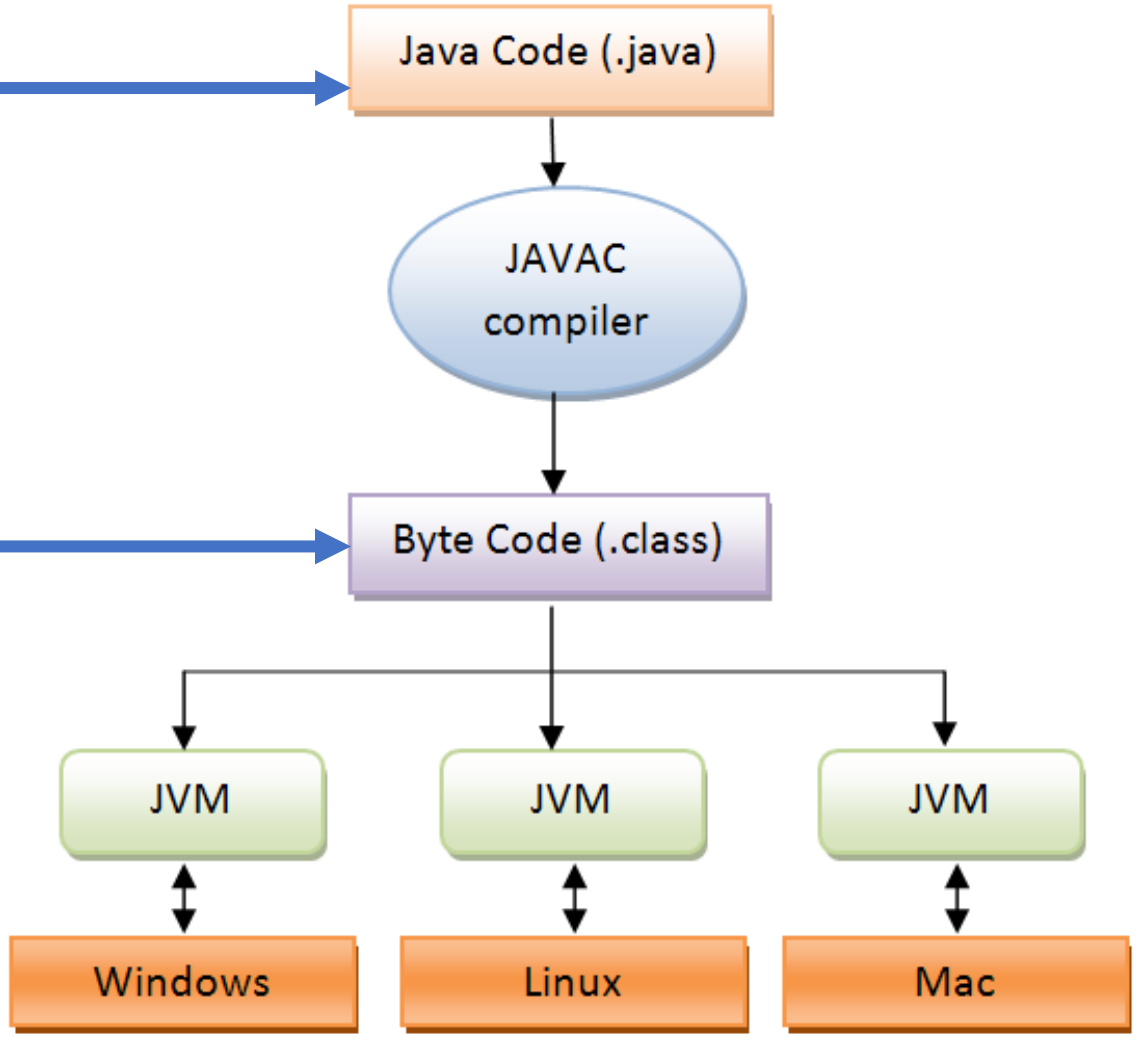


6. Java programming language

Java and a Typical Java Development Environment



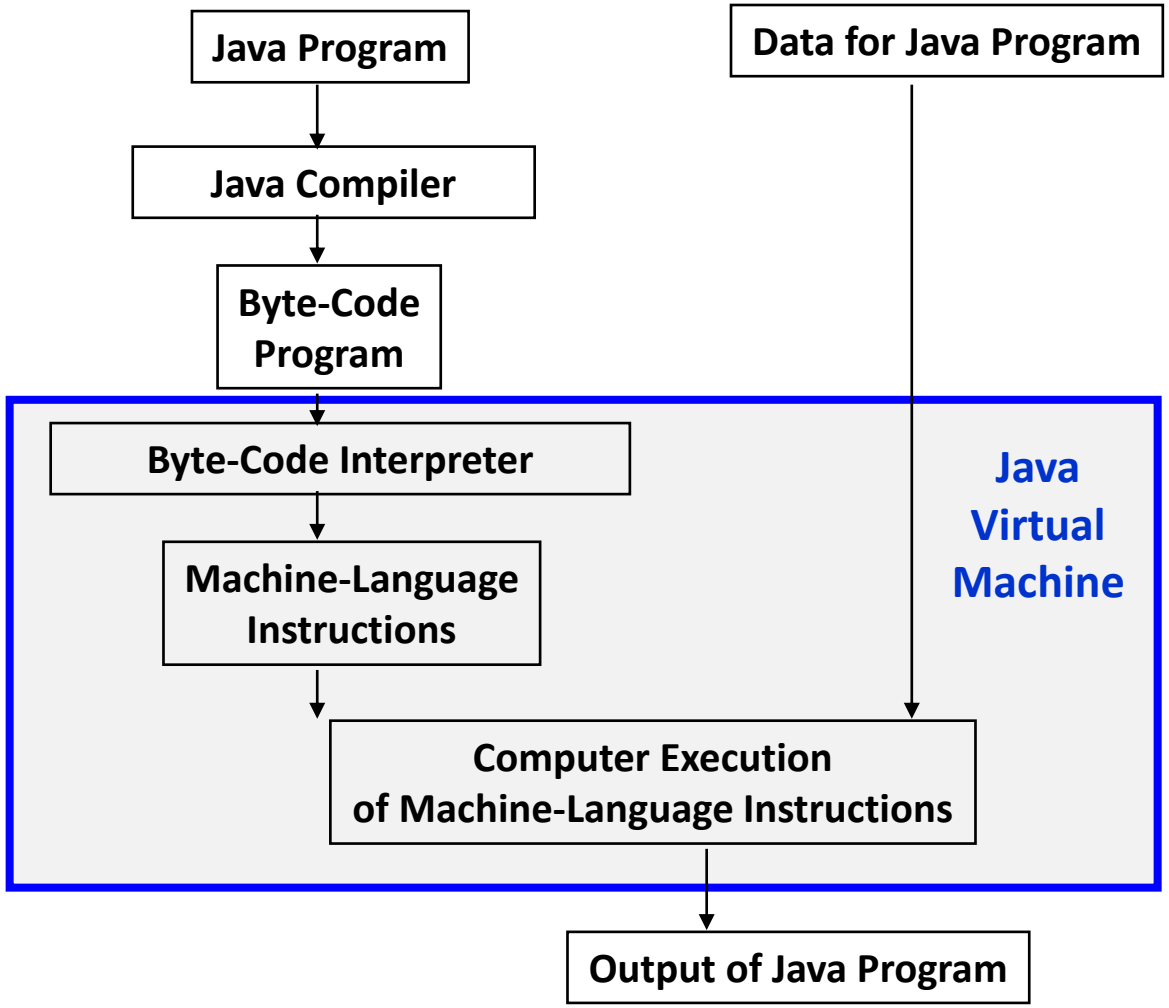




6. Java programming language

Java and a Typical Java Development Environment

- Both Compilation and Interpretation
- Intermediate Code: *"Byte Code"*
  - similar to assembly code, but hardware *independent*
- Interpreter translates from generic byte code to hardware-specific machine code



6. Java programming language

Java Byte Code?

- **Generated** by Java compiler
  - Instead of generating machine language as most compilers do, the Java compiler generates byte code.
- **Translated** to machine language of various kinds of computers
- **Executed** by Java interpreter
- Invisible to programmer
  - You don't have to know anything about how byte code works to write a Java program.

6. Java programming language

Why Use Byte Code?

Disadvantages:

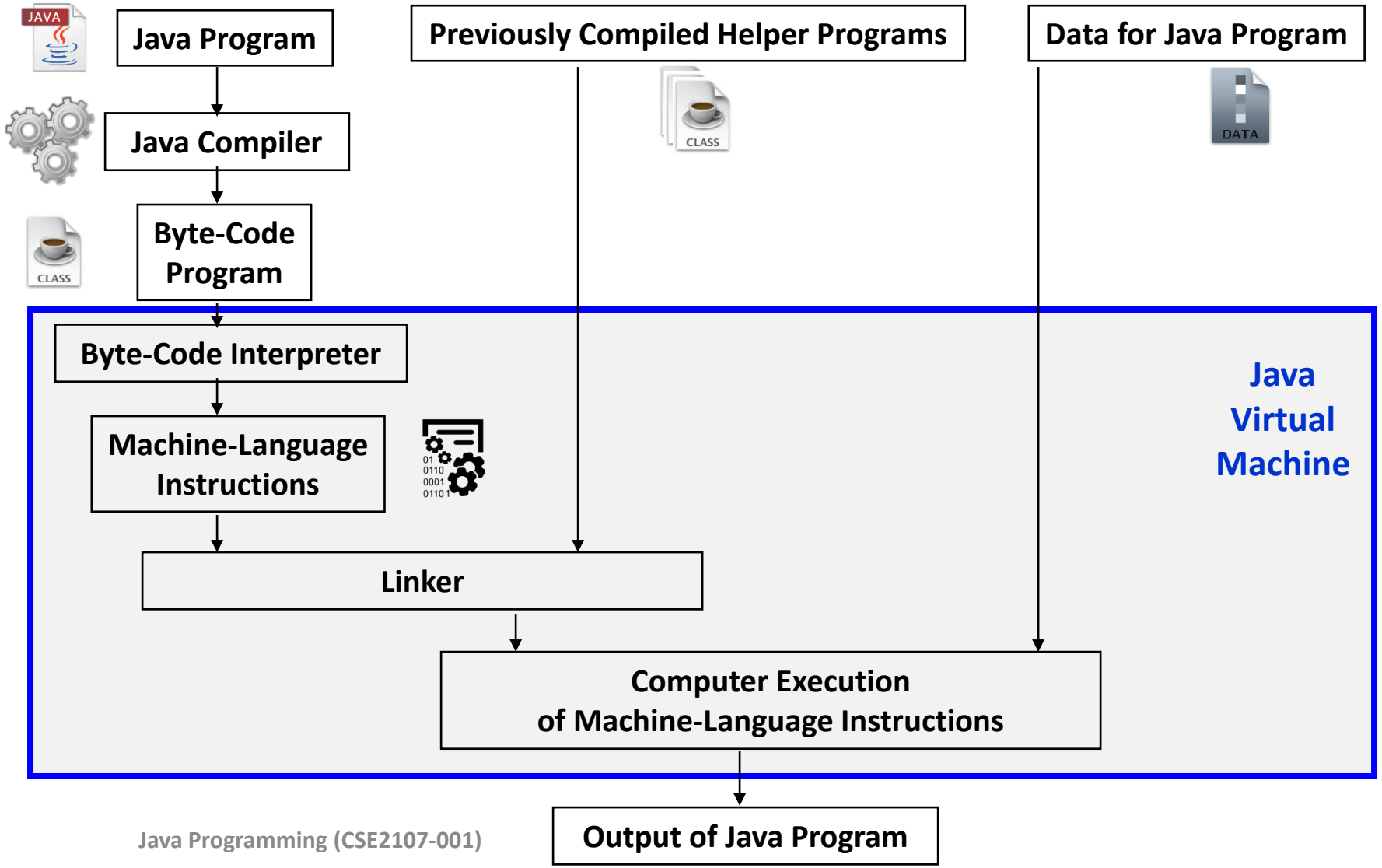
- requires both compiler and interpreter
- slower program execution

Advantages:

- portability
  - very important
  - same program can run on computers of different types (useful with the Internet)
  - Java compiler for new types of computers can be made quickly

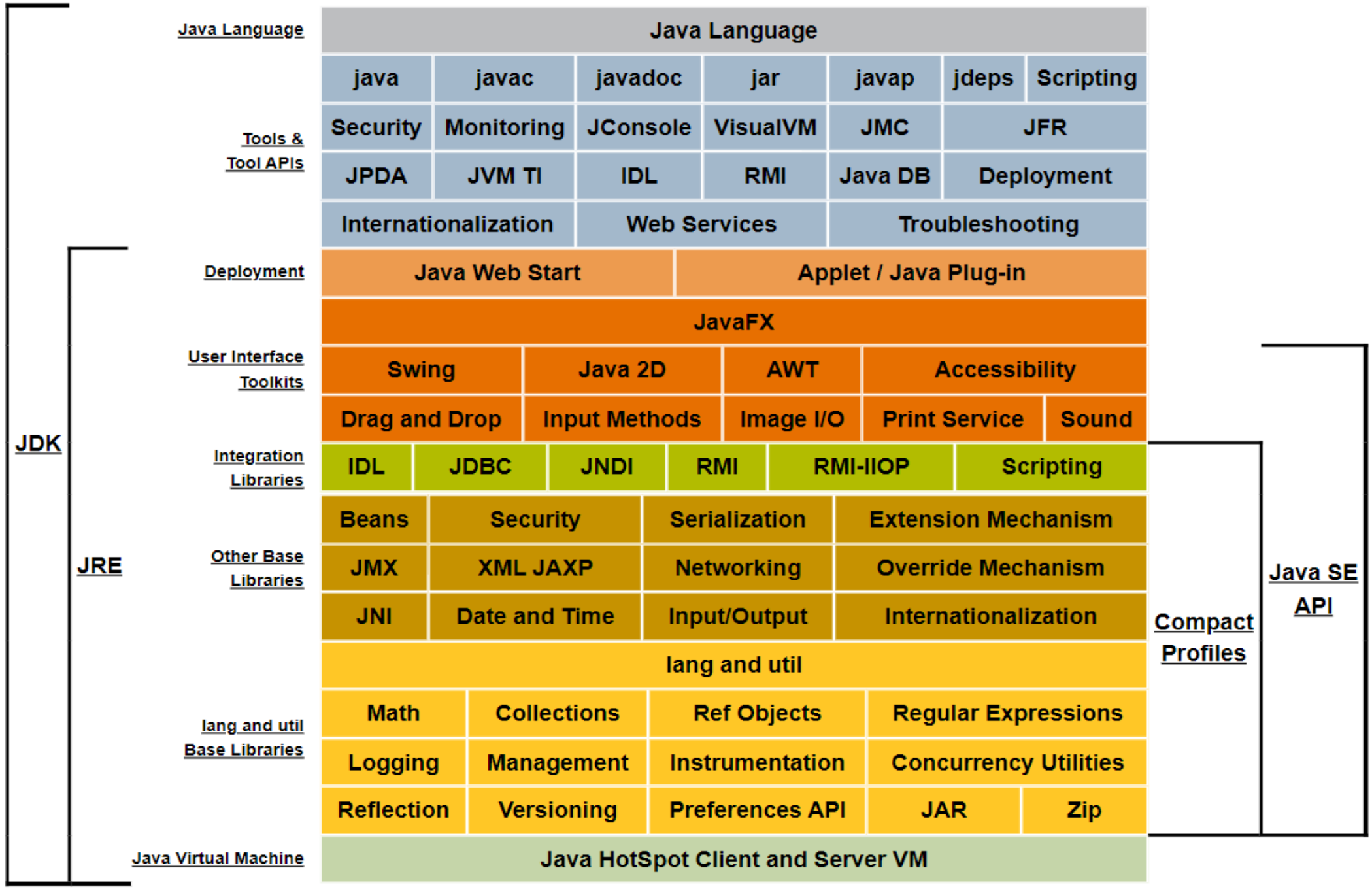
6. Java programming language

Java Program Translation Including Linker



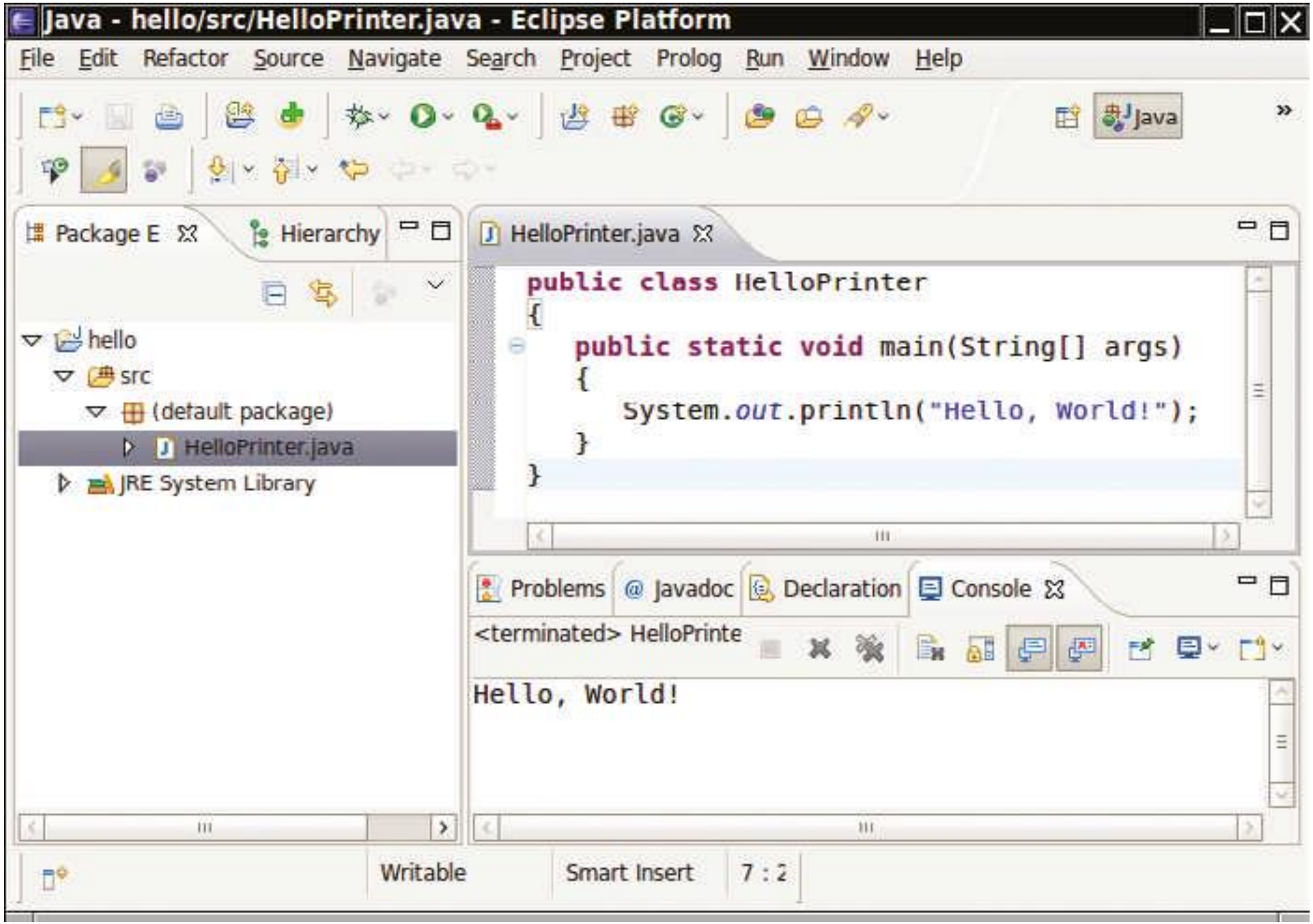
JDK 19 is the latest release

[Java Downloads | Oracle]



Eclipse IDE

The Environment

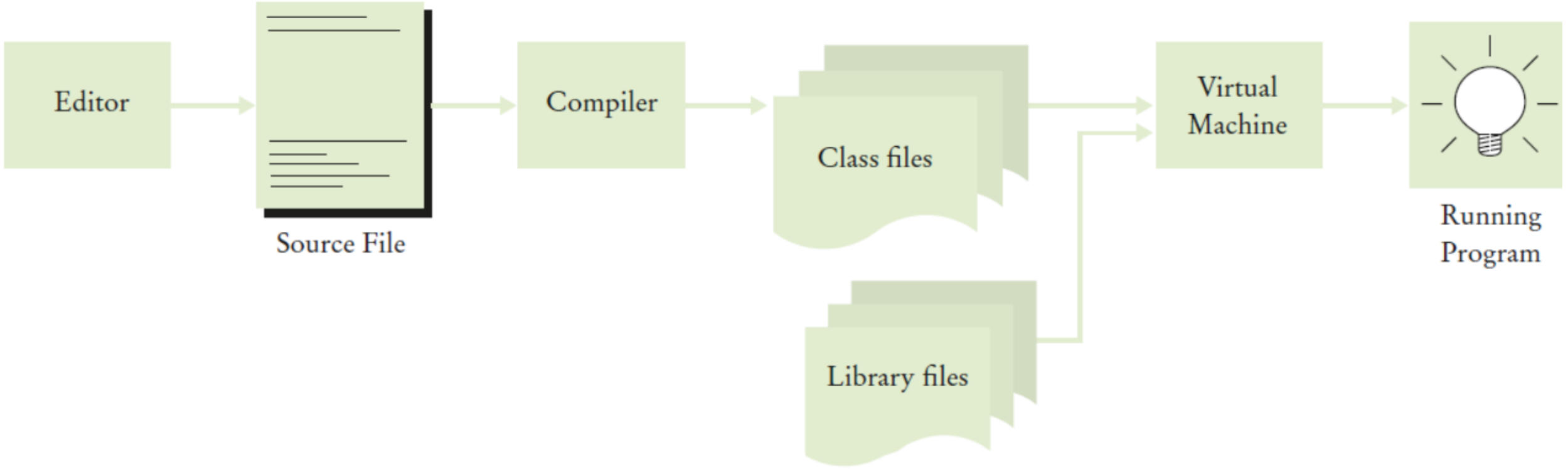


Eclipse IDE

Writing a simple program

Java is case sensitive.

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



Thank you