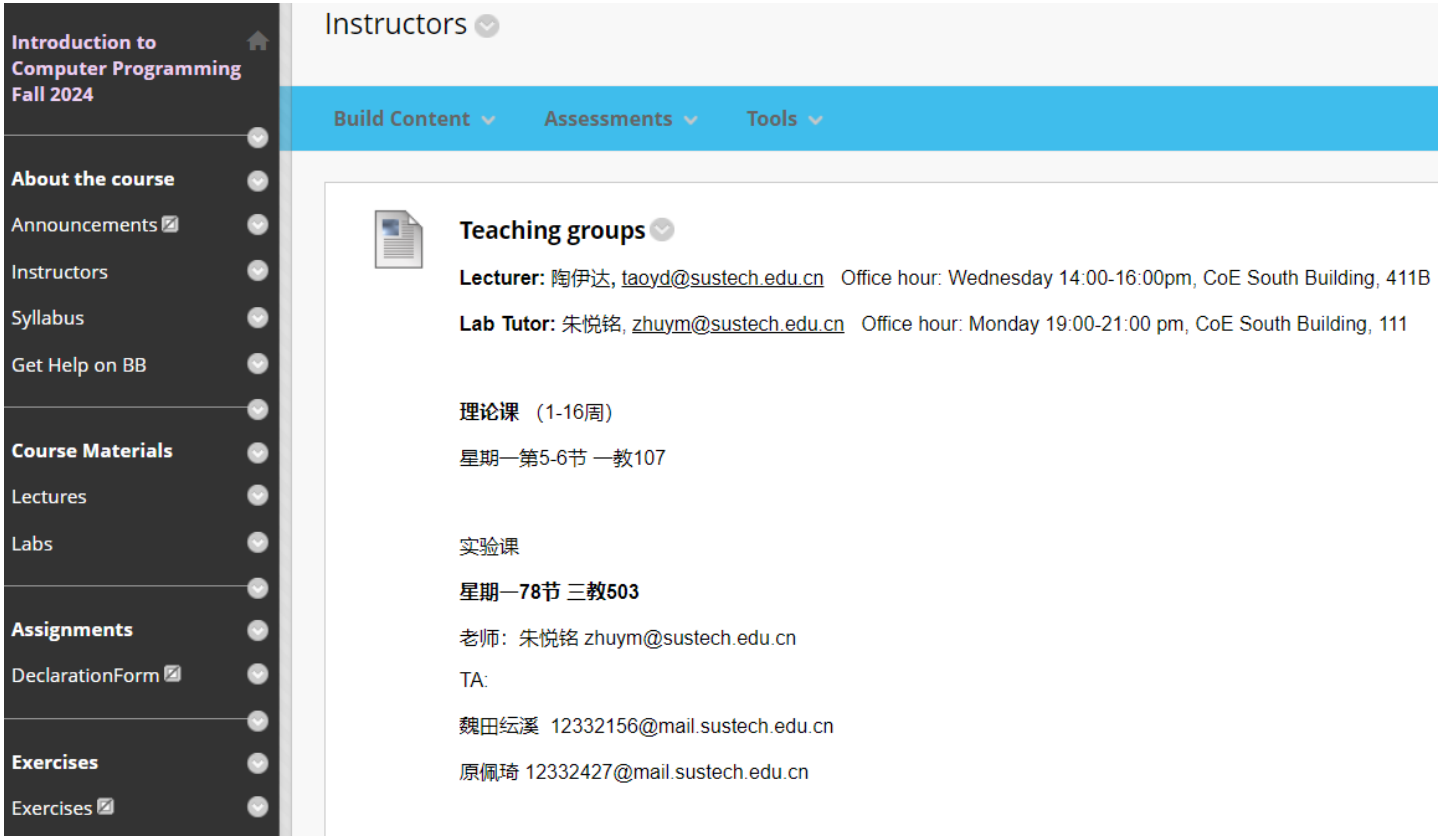
CS109: Introduction to Computer Programming

Yida Tao (陶伊达)

taoyd@sustech.edu.cn

Course Website

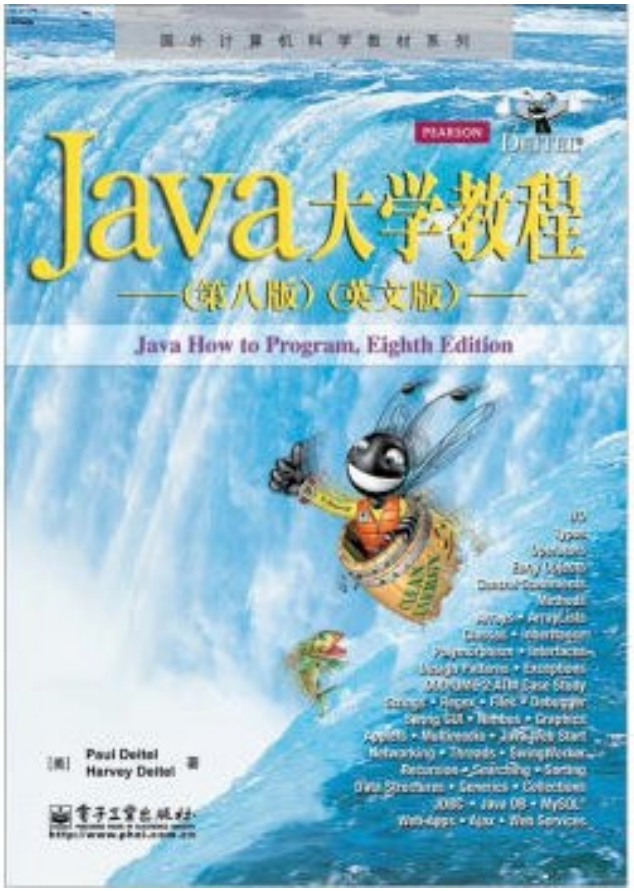
- Available at the **Blackboard** course site: https://bb.sustech.edu.cn/



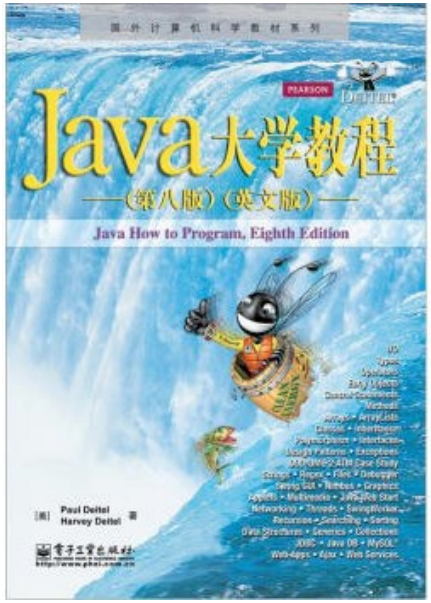
Textbook

- Main textbook:
  - P. Deitel, H. Deitel, **Java: How to Program（Java大学教程，第八版），电子工业出版社**

- Reference books:
  - Y. Daniel Liang. **Introduction to Java Programming**, 12e, Pearson, Prentice Hall, 2020.
  - Allen B. Downey and Chris Mayfield. **Think Java, How to Think Like a Computer Scientist**, O'Reilly, 2016.



Course Syllabus



- Introduction to Computers and Java
- Primitive Data Types
- Control Statements and Structured Programming
- Array
- Procedural Programming: Methods and APIs
- Introduction to Classes, Objects, Methods
- Strings and Wrapper Classes
- Classes, Objects and Methods: A Deeper Look
- Object-Oriented Programming: Inheritance
- Object-Oriented Programming: Polymorphism
- Graphical User Interface (GUI)
- Generic Classes and Methods
- Exception Handling: A Deeper Look

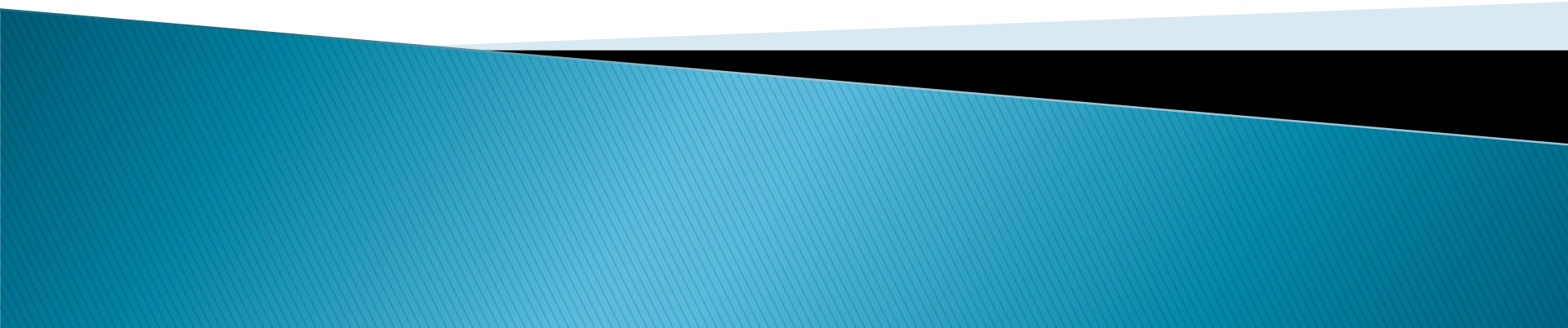
Grading Scheme

- Final exam: 40%
- Project: 20%
- Labs: 5%
- Assignments: 30%
  - 6 assignments, starting from week 3
- Quiz, exercises, and participation: 5%

Programming!

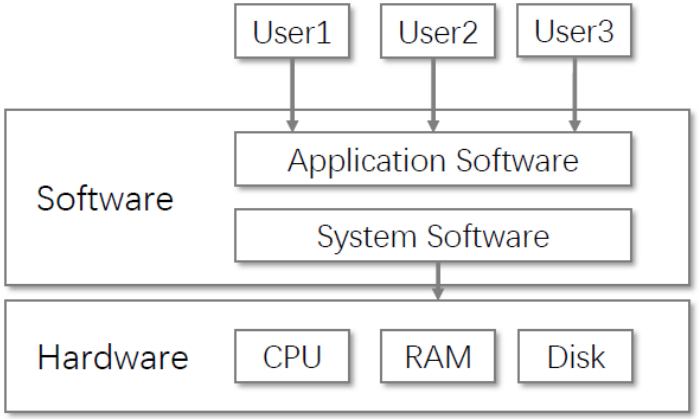
You will pass the course if your overall grade >= 60

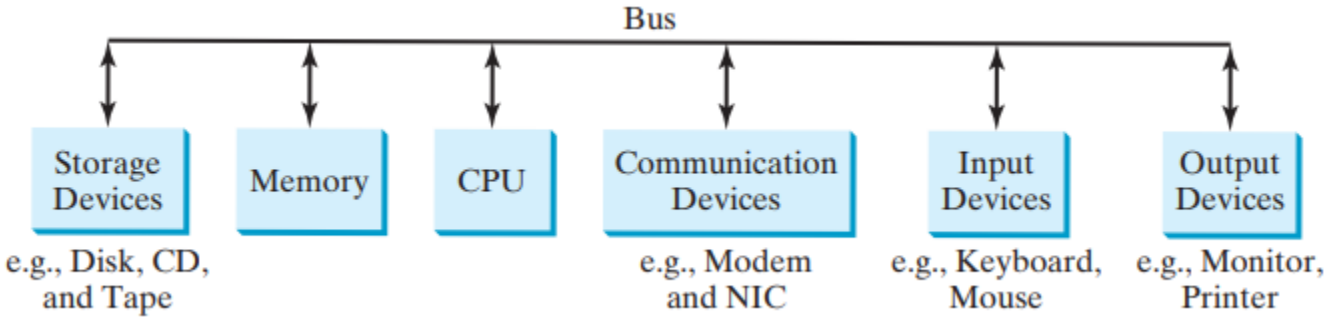
Chapter 0:
Introduction to Computers, Programs, and Java



What is a computer?

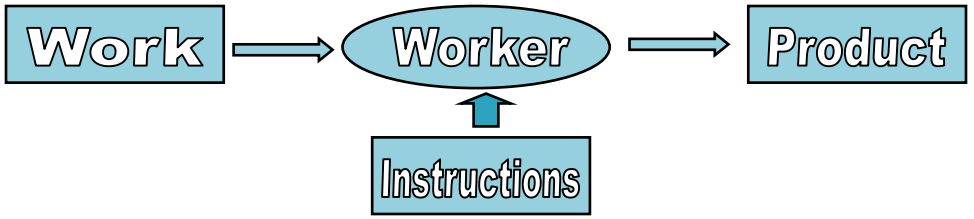
- **Software**: a set of programs, which could be viewed as a set of instructions
- **Hardware**: physical parts (e.g., keyboard, mouse, hard disk, memory, CPU). Hardware is directed by software to execute commands or instructions



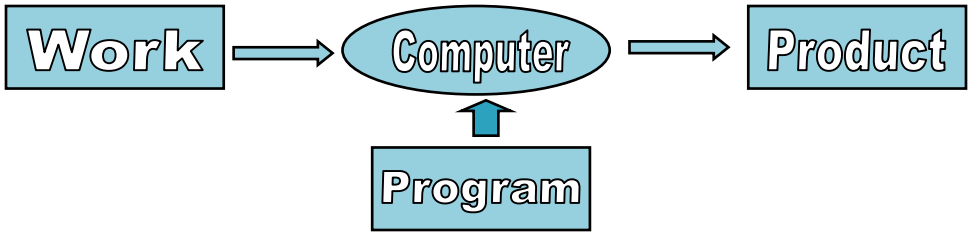


What is a computer program?

- Human work model



- Computer work model



- A computer program is a set of machine-readable instructions that tells a computer how to preform a specific task.

What is a (programming) language?

A sequence of instructions

An algorithm (算法)
(in human language)

A program
(in computer language)

- Programs are written in programming languages
- There are many programming languages
  - Low-level (低级语言), understandable by a computer
  - High-level (高级语言), understandable by human

Can you understand this?

0000100100101110011001100110100101101100011001010000100100100010011011000110
0101011000110111010001110101011100100110010100110001001011100110001100100010
0000101001100111011000110110001100110010010111110110001101101111011011010110
0000110100101101100011001010110010000101110001110100000101000101110011100110
1100101011000110111010001101001011011110110111000001001001000100010111001110
1000110010101111000011101000010001000001010000010010010111001100001011011000
1101001011001110110111000100000001101000000101000001001001011100110011101101
1000110111101100010011000010110110000100000011011010110000101101001011011100
0001010000010010010111001110100011110010111000001100101000010010010000001101
1010110000101101001011011100010110000100011011001100111010101101110011000110
1110100011010010110111101101110000010100000100100101110011100000111001001101 1
1011000110000100100110000001101000000101001101101011000010110100101101110001
1110100000101000001001001000010010001101010000010100100100111101001100010011
1110100011101010101010001010010001100100000001100000000101000001001011100 11
01100001011101 1001

How about this?

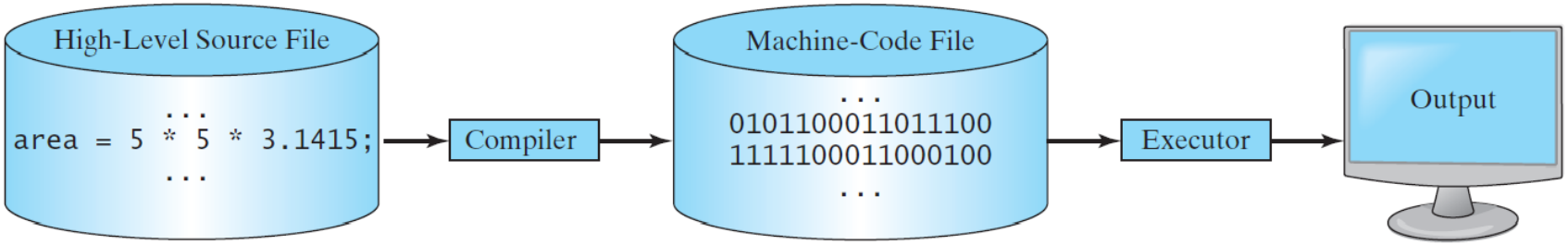
```
main:
        !#PROLOGUE# 0
        save %sp,-128,%sp
        !#PROLOGUE# 1
        mov 1,%o0
        st %o0,[%fp-20]
        mov 2,%o0
        st %o0,[%fp-24]
        ld [%fp-20],%o0
        ld [%fp-24],%o1
        add %o0,%o1,%o0
        st %o0,[%fp-28]
        mov 0,%i0
        nop
```

Is it better now?

```
int valueofz( )
{
        int x, y, z;
        x = 1;
        y = 2;
        z = x+y;
        return z;
}
```

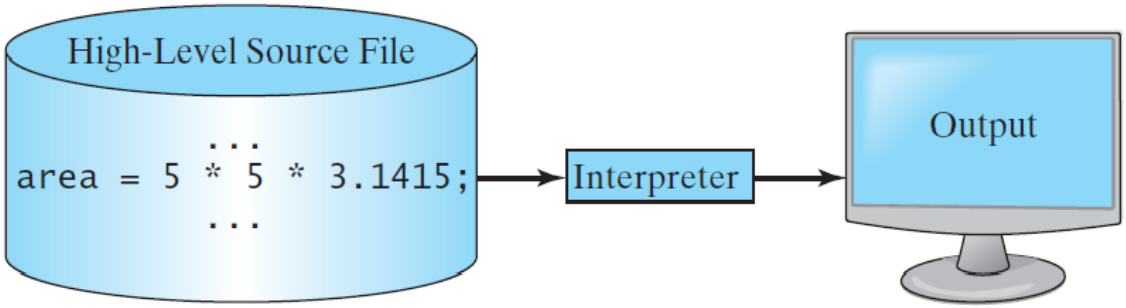
Compilation: from source to executables

- A complier (编译器) translates source programs written in high-level languages into machine codes that can run directly on the target computer.



Interpreter

- An **interpreter** (解释器) reads one statement from the source code, translates it to the machine code or virtual machine code, and then executes it right away



A brief history of Java

- In 1991, Sun Microsystems (acquired by Oracle in 2009) funded an internal research project, aiming to achieve the goal of "write once, run anywhere". This resulted in a C++-based language named Java.
- Why called "Java"? Java is an island in Indonesia where the first coffee was produced (Java coffee)

The father of Java:
James Gosling

We learn Java, why?

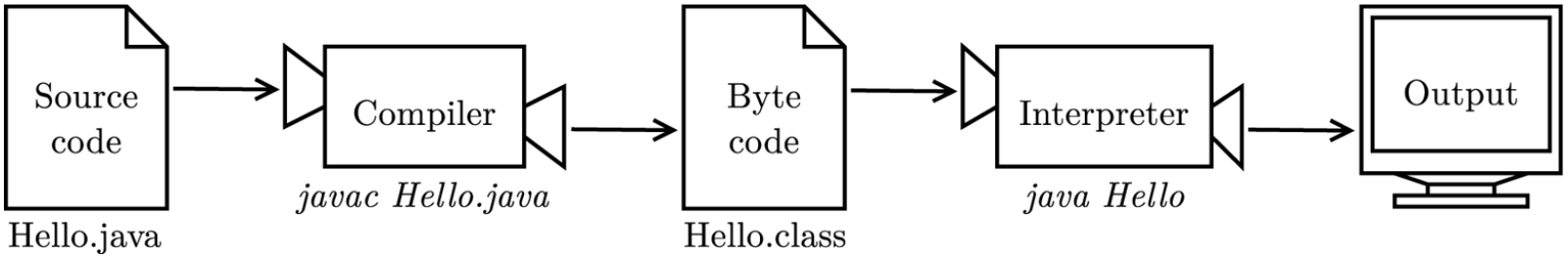
- Java is a full-featured, general-purpose programming language that can be used to develop standalone applications across platforms on servers, desktop computers, and mobile devices.

从笔记本电脑到数据中心，从游戏控制台到科学超级计算机，从手机到互联网，Java 无处不在！

- 97% 的企业桌面运行 Java
- 美国有 89% 的桌面（或计算机）运行 Java
- 全球有 900 万 Java 开发人员
- 开发人员的头号选择
- 排名第一的部署平台
- 有 30 亿部移动电话运行 Java
- 100% 的蓝光盘播放器附带了 Java
- 有 50 亿张 Java 卡在使用
- 1.25 亿台 TV 设备运行 Java
- 前 5 个原始设备制造商均提供了 Java ME

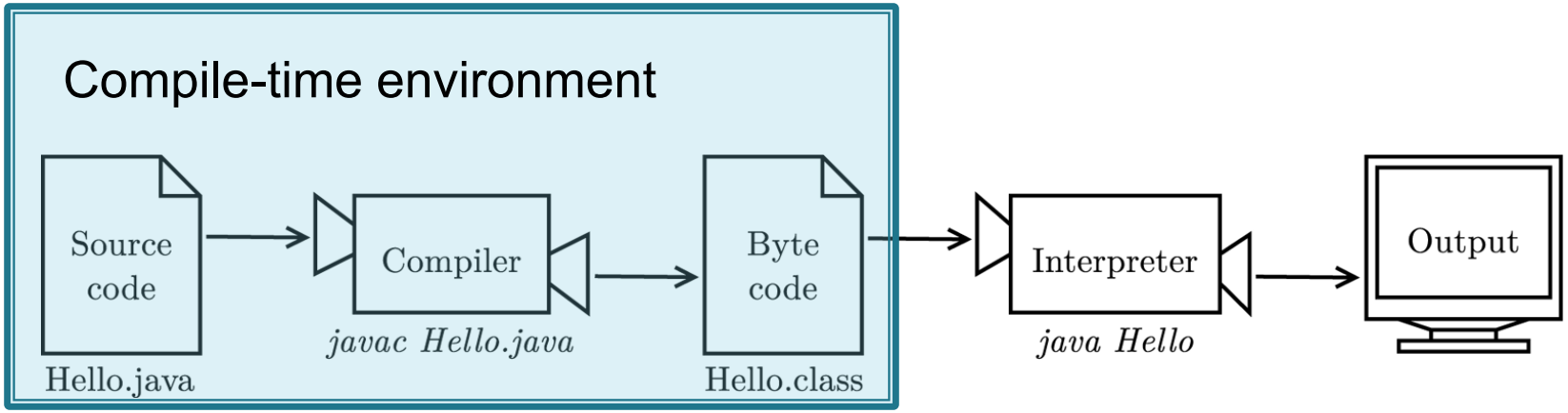
https://www.java.com/zh-CN/about/

Java is both compiled and interpreted



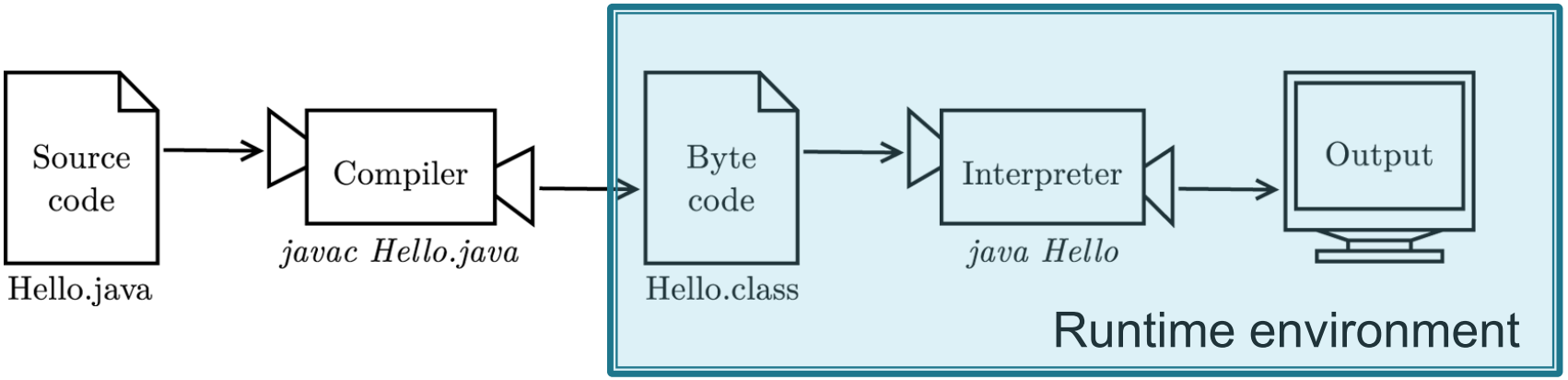
Java programming steps

- Step 1: Edit (write the program and store it in the disk `.java`)
- Step 2: Compile (create bytecode and store it in a file `.class`)



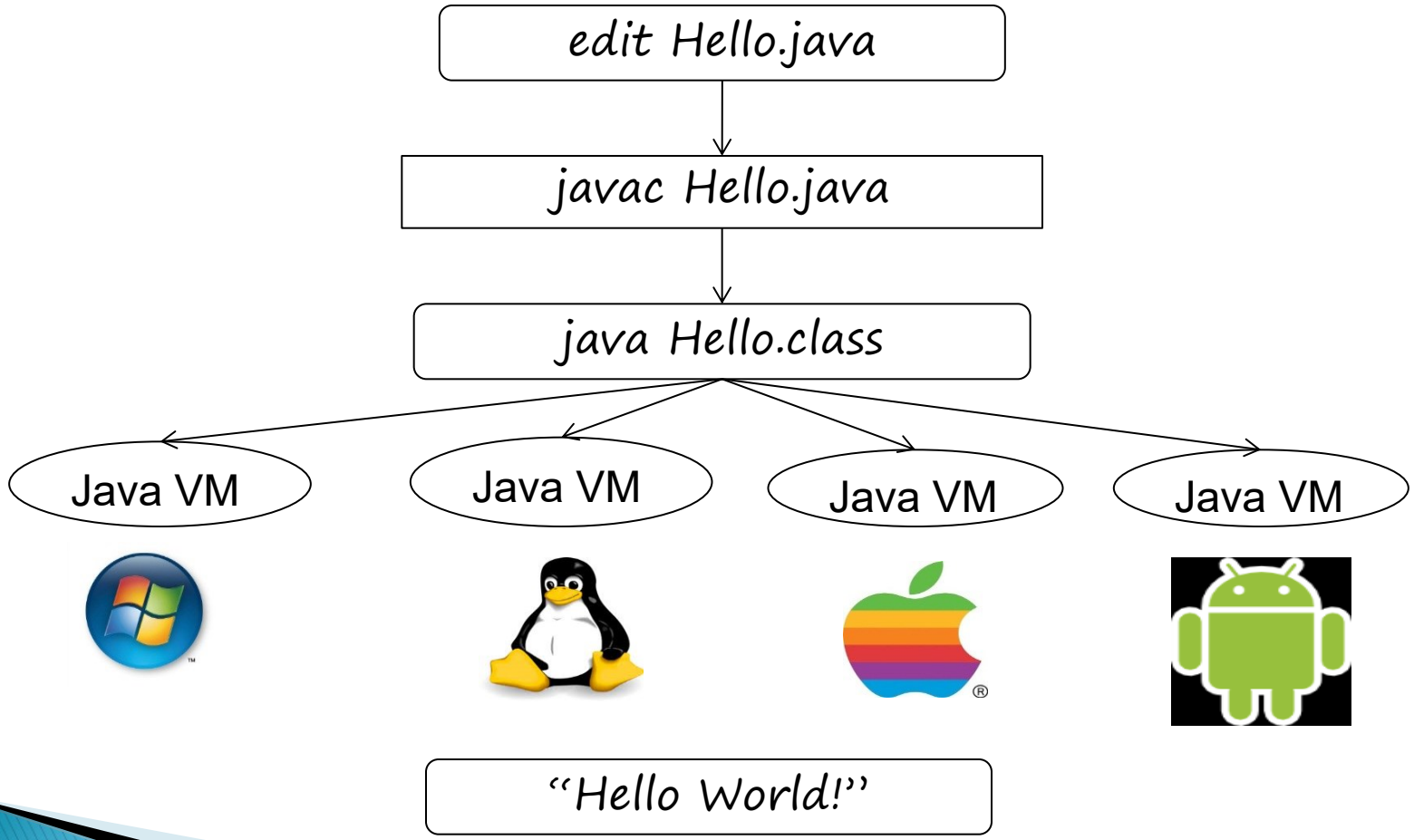
Java programming steps

- Step 3: the .class bytecode is read, verified, interpreted, and executed in JVM (Java Virtual Machine)



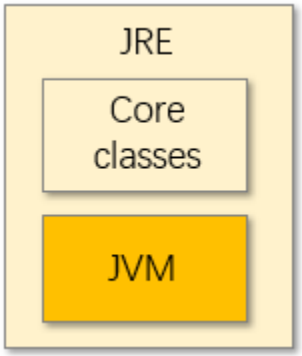
Write Once and Run Anywhere

Java is platform independent



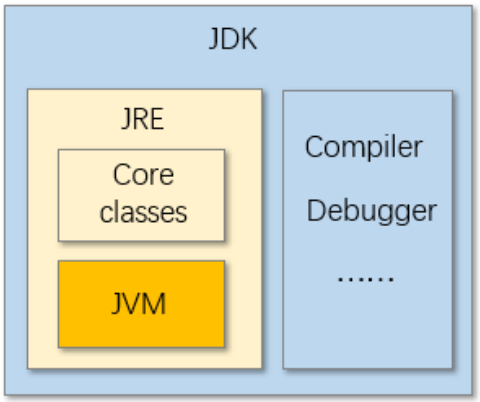
JRE and JVM

- A **Java Virtual Machine (JVM)** is an abstract computing machine that enables a computer to run a Java program.
- The **Java Runtime Environment (JRE)** provides the minimum requirements for executing a Java application. It consists of the Java Virtual Machine (JVM), core classes, and supporting files.
- In short, JRE = JVM + Library classes



JDK（开发套件）

- The **Java Development Kit (JDK)** is a software development environment for developing Java programs. It includes:
  - A Java Runtime Environment (JRE, 运行环境)
  - A compiler (javac)
  - An interpreter/loader (java)
  - An archiver (jar)
  - A documentation generator (javadoc)
  - Other tools needed in Java development.



- In short, JDK = JRE + Development tools

Our First Java Program

```
public class Welcome1 {
    // main method begins the execution of a Java application
    public static void main(String[] args) {
        System.out.println("Welcome to Java Programming!");
    }
}
```

`Welcome1` prints the following text in the command window (console):

```
Welcome to Java Programming!
```

Class Declaration

```
public class Welcome1 {
    // main method begins the execution of a Java application
    public static void main(String[] args) {
        System.out.println("Welcome to Java Programming!");
    }
}
```

- Every Java program consists of at least one class (类) that you define
- The `class` keyword introduces a class declaration and is immediately followed by the class name
- Keywords are reserved for use by Java and are always spelled with all lowercase letters (we will see more later)

Identifiers (标识符)

```
public class Welcome1 {
    // main method begins the execution of a Java application
    public static void main(String[] args) {
        System.out.println("Welcome to Java Programming!");
    }
}
```

- A name in a Java program is called an **identifier**, which is used for identification purpose.
  - "Welcome1" is an identifier. It is the name for the class we just defined.

Identifiers (标识符)

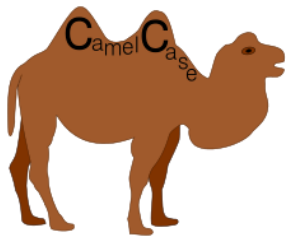
```
public class Welcome1 {
    // main method begins the execution of a Java application
    public static void main(String[] args) {
        System.out.println("Welcome to Java Programming!");
    }
}
```

- The only allowed characters in Java identifiers are **a to z, A to Z, 0 to 9, $** and _ (underscore).
- Identifiers can't start with digits, e.g., **123name** is invalid.
- Java Keywords cannot be identifiers (can’t compile).
- Java is case sensitive—uppercase and lowercase letters are distinct (not in comments). “main” and “Main” are different identifiers.

Class Names

```
public class Welcome1 {
    // main method begins the execution of a Java application
    public static void main(String[] args) {
        System.out.println("Welcome to Java Programming!");
    }
}
```

- By convention, class names begin with a capital letter and capitalize the first letter of each word they include (upper camel case, 大驼峰式命名规范)



Comments (注释)

```
public class Welcome1 {
    // main method begins the execution of a Java application
    public static void main(String[] args) {
        System.out.println("Welcome to Java Programming!");
    }
}
```

- Comments help document programs to improve their readability.
- Compiler ignores comments.

Comments (注释)

```
public class Welcome1 {
    // main method begins the execution of a Java application
    public static void main(String[] args) {
        System.out.println("Welcome to Java Programming!");
    }
}

    // This is a line comment (行注释)

    /* This is a block comment (块注释或段注释). It
       can be spread over multiple lines */
```

Method declaration

```
public class Welcome1 {
    // main method begins the execution of a Java application
    public static void main(String[] args) {
        System.out.println("Welcome to Java Programming!");
    }
}
```

- Java class declarations normally contain one or more methods
- The main method is the starting point of Java applications

Braces (花括号)

```
public class Welcome1 {
    // main method begins the execution of a Java application
    public static void main(String[] args) {
        System.out.println("Welcome to Java Programming!");
    }
}
```

- A pair of curly braces { } in a program forms a block (块) that groups the program's components.
- A left brace { begins the declaration of every class and method
- A corresponding right brace } ends the declaration of each class and method

The `main` method body

```
public class Welcome1 {
    // main method begins the execution of a Java application
    public static void main(String[] args) {
        System.out.println("Welcome to Java Programming!");
    }
}
```

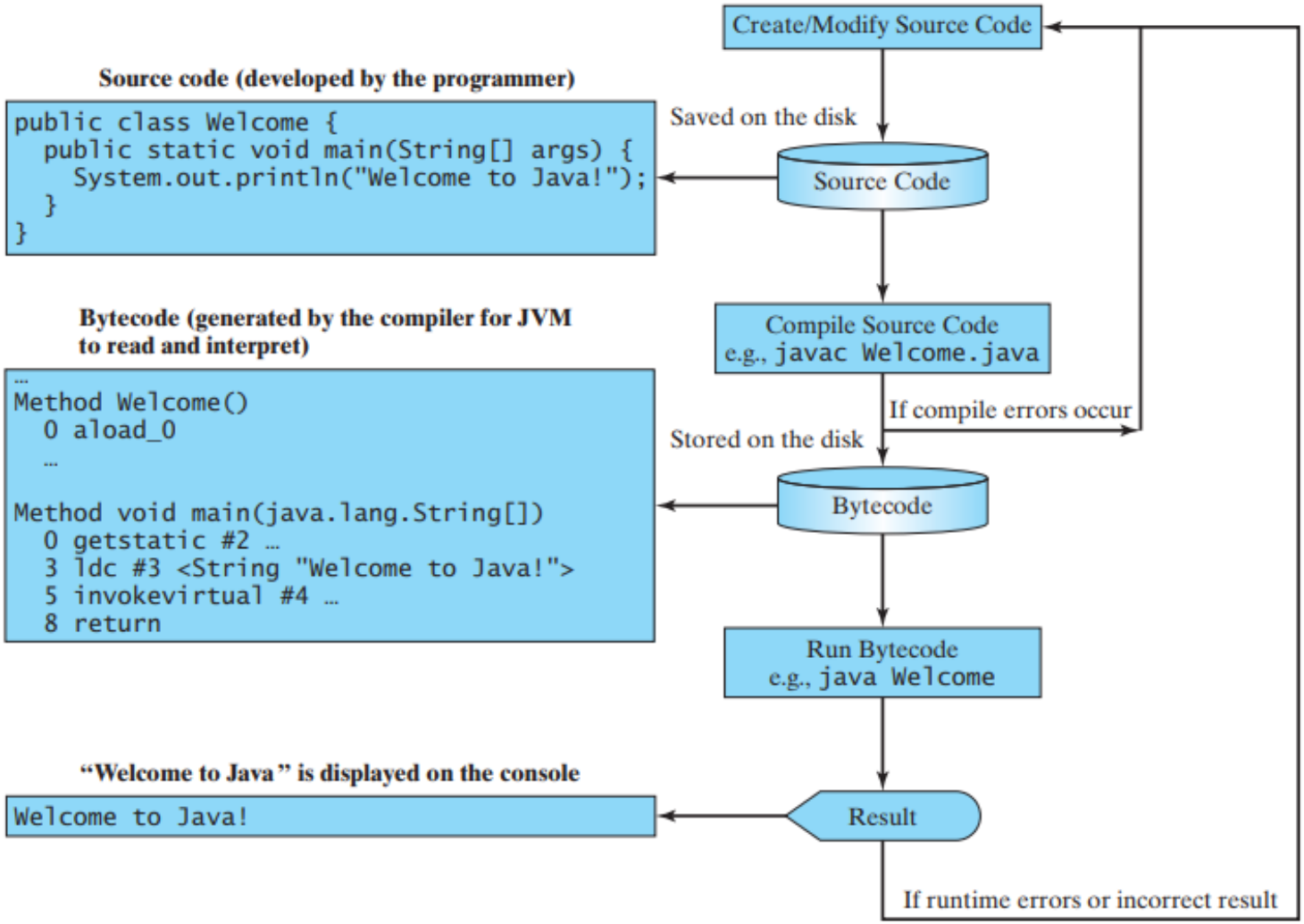
- A method is a construct that contains **statements** (语句).
- The `System.out.println` statement displays the input string on the console
- String is a programming term meaning a sequence of characters. A string must be enclosed in double quotation marks (“ xxx ”).
- Every statement in Java ends with a semicolon (;),

Indentation (缩进)

```
public class Welcome1 {
    // main method begins the execution of a Java application
    public static void main(String[] args) {
        System.out.println("Welcome to Java Programming!");
    }
}
```

- Code between braces should be indented (good practice)
- Indentation doesn't affect the execution of code; why use indentation?
- But for some programming languages (e.g., Python), indentation matters a lot

Compile & Execute `Welcome1.java`



Modify the code

```
// Print multiple lines of text using a single statement
public class Welcome2 {
    public static void main(String[] args) {
        System.out.println("Welcome\nto\nJava\nProgramming!");
    }
}
```

`Welcome2` prints the following text on the console:

```
Welcome
to
Java
Programming!
```

The newline character \n

- Newline characters (换行符) instruct `System.out`'s `println` method to position the output cursor at the beginning of the next line in the command window

- Newline characters are white-space characters, which represent horizontal or vertical space in typography and do not correspond to visible marks (辅助排版)

```
System.out.println("Welcome\nto\nJava\nProgramming!");
```

Escape character

- The backslash (\) is an escape character (转义字符, a case of metacharacters), which invokes an alternative interpretation on subsequent characters (转换意义)

- Backslash \ is combined with the next character to form an escape sequence (转义序列)

- The escape sequence \n represents the newline character

Common Escape Sequences

Sequence	Description
`\n`	Newline (换行符). Position the cursor at the beginning of the next line.
`\t`	Horizontal tab. Move the cursor to the next tab stop.
`\"`	Used to print a double-quote character. `System.out.println("\"in quotes\"");` displays `"in quotes"`