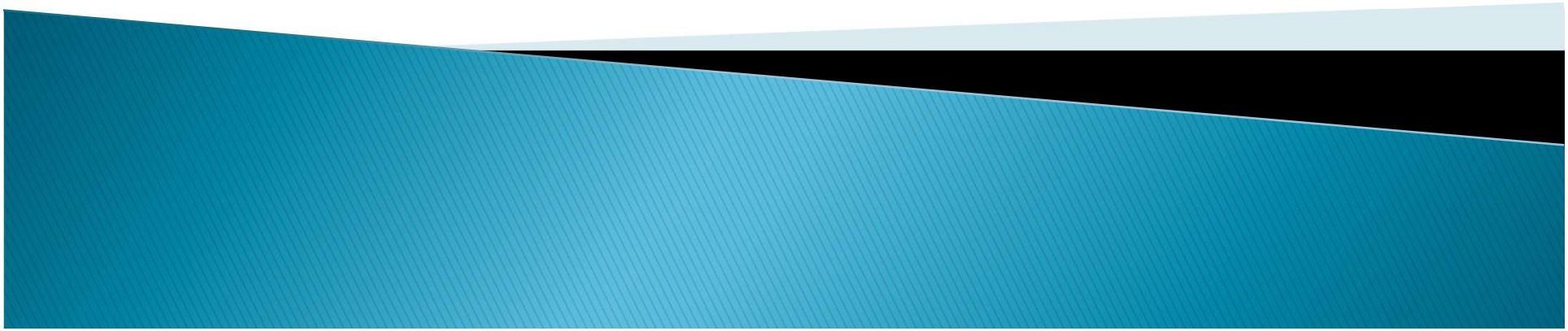
CS102A: Introduction to Computer Programming

Xuetao Wei (危学涛)

weixt@sustech.edu.cn

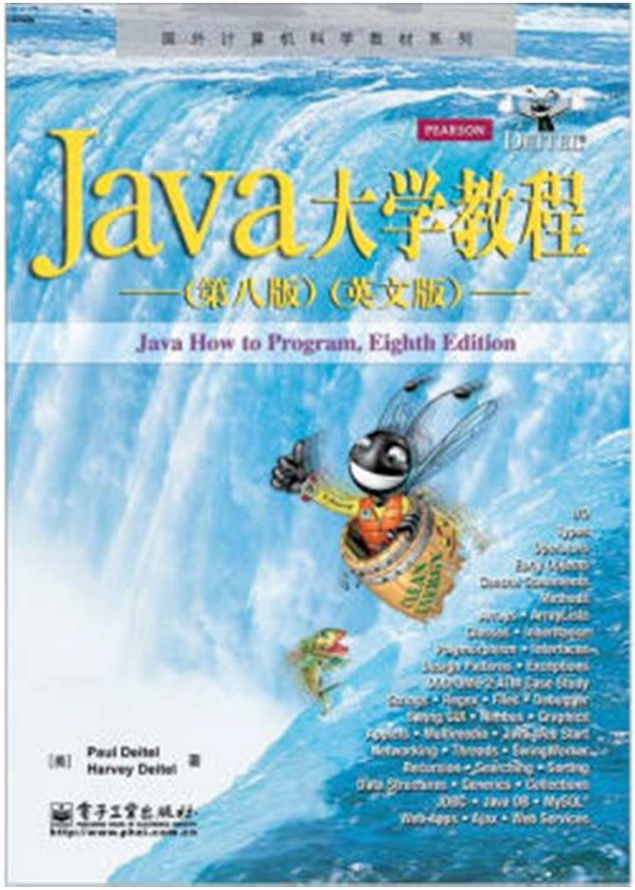


Course Instructor

- Dr. Xuetao Wei (Associate Professor in CSE)
- Office: Room 406, Block 10, Innovation Park (创园)
- Email: weixt@sustech.edu.cn

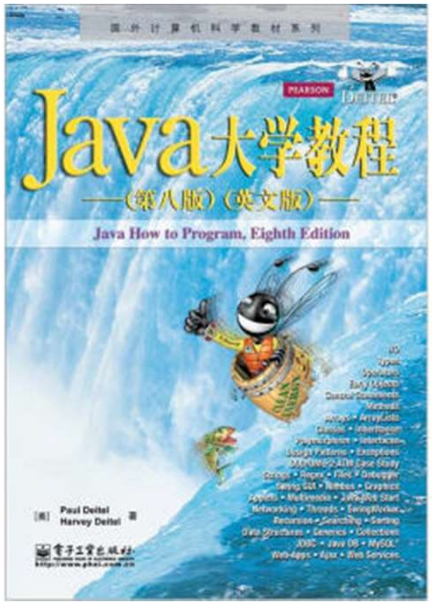
Textbook

- 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**, 10e, Pearson, Prentice Hall, 2015.
  - 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
- Introduction to Java Applications
- Data Types
- Control Statements
- Array
- Procedural Programming
- Introduction to Classes, Objects, Methods
- Strings and Wrapper Classes
- Classes, Objects and Methods: A Deeper Look
- Object-Oriented Programming: Inheritance
- Object-Oriented Programming: Polymorphism
- GUI Programming
- Generic Classes and Methods
- Generic Collections
- Exception Handling: A Deeper Look

Lecture Notes

- Available at the Blackboard course site
- Computing technologies advance quickly. Search online to learn more by yourself.
  - Google, Baidu, Bing
  - Stack Overflow: https://stackoverflow.com/
  - GitHub: https://github.com/

Course Objectives

- Learn how to solve problems by writing computer programs
- Learn how to design a computer program
- Learn the basics of the Java programming language
- Learn the basic concepts of object-oriented programming
- Prepare you for future courses and career

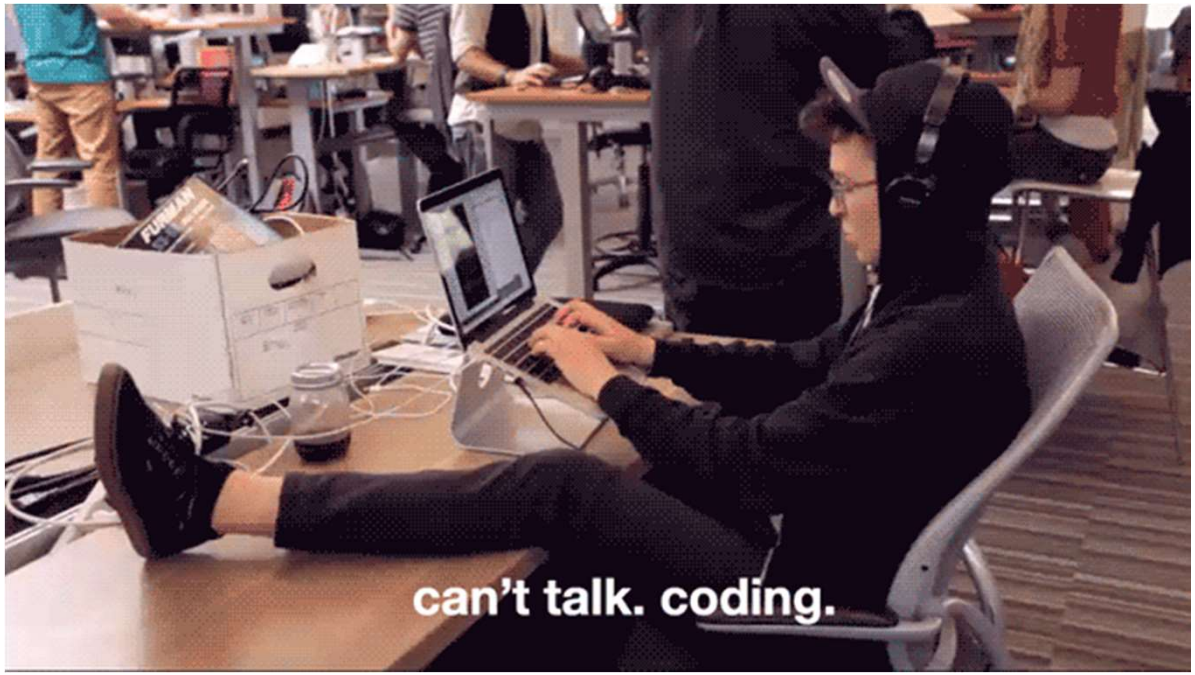
Grading Scheme

- Final exam: 30%
- Project: 20%
- Lab attendance and exercise: 10%
- Lab assignments: 30%

Programming!

- Lecture attendance and quizzes: 10%
- You will pass the course if your overall grade >= 60

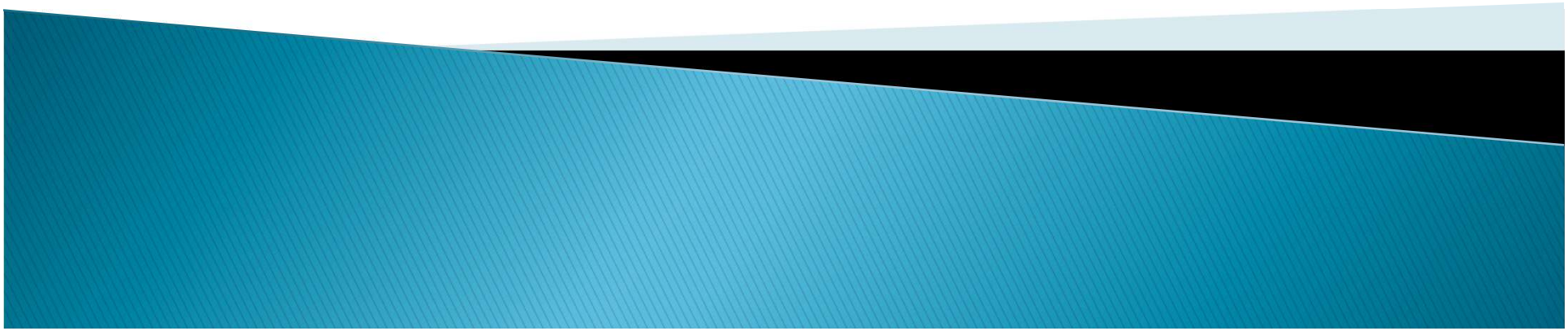
Let's Start & Have fun ☺



Practice Makes Perfect!

Chapter 1: Introduction to Computers and Java

Java™ How to Program, 8/e

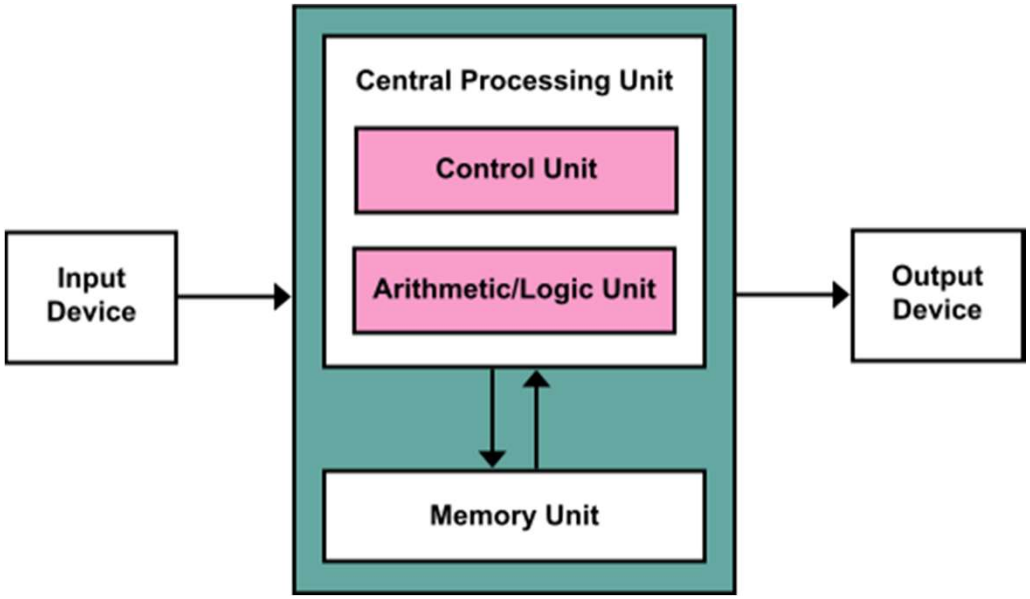


Computer System

- Hardware (physical parts, e.g., keyboard, mouse, hard disk, memory, processing units)
- Software (computer programs, libraries, non-executable data, e.g., documentation)
- Hardware is directed by software to execute commands or instructions. A combination of hardware and software forms a usable computer system.

The von Neumann Architecture

- A design model for a stored-program digital computer



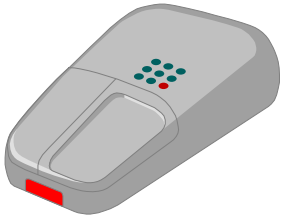
John von Neumann
(1903-1957)
Hungarian-American
mathematician, physicist

Computer Organization

- Following the von Neumann architecture, modern computers consist of the following logical units:
  - Input unit
  - Output unit
  - Memory unit (内存，主存)
  - Arithmetic and logic unit (ALU, 算术逻辑单元)
  - Central processing unit (CPU, 中央处理器)
  - Secondary storage unit (辅助存储单元，二级存储器)

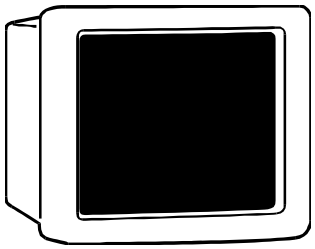
Input Unit

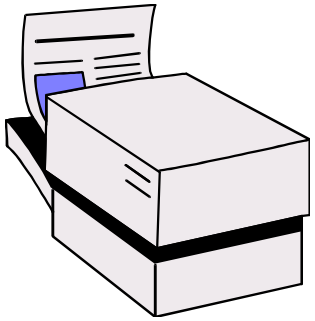
- The “receiving” section of a computer
- Obtains information (data and programs…) for other units to process.

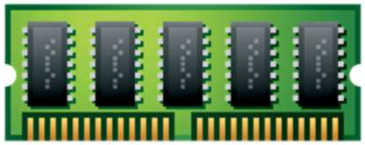
Output Unit

- The "shipping" section of a computer
- Takes the information that the computer has processed and makes it available for use outside the computer.





Memory Unit



- Rapid-access, relatively low-capacity “warehouse” section
- Retains information entered through the input unit, making it immediately available for processing when needed.
- Retains processed information until it is placed on output devices.
- Information in the memory unit is **volatile** (易失) and will be lost when the computer’s power is turned off.
- Also known as main memory, primary memory, memory, or RAM (Random Access Memory).

http://people.scs.carleton.ca/~armyunis/notes/ram.htm

Arithmetic and Logic Unit (ALU)

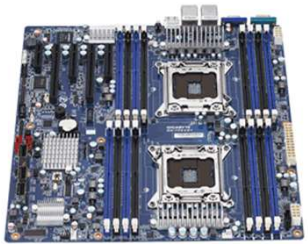
- "Manufacturing" section that performs calculations, such as addition, subtraction, multiplication and division.
- Contains the mechanisms that allow the computer to make decisions, e.g., comparing two items from the memory to determine whether they are equal.
- In today's computer systems, the ALU is usually implemented as part of a CPU.

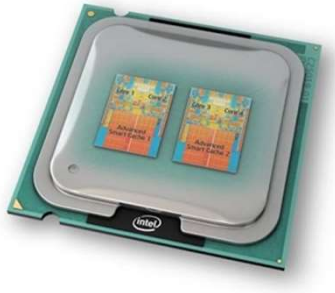
Central Processing Unit (CPU)

- "Administrative" section that coordinates the operations of the other units (the brain/heart of a computer).
  - Tells the input unit when information should be read into the memory unit
  - Tells the ALU when information in the memory unit should be used in calculations
  - Tells the output unit when to send information from the memory unit to output devices

Central Processing Unit (CPU)

- Many of today's computers have multiple CPUs (can perform operations simultaneously). They are called multiprocessors.
- A multicore processor implements multiprocessing on a single integrated circuit chip (e.g., dual-core, quad-core, octa-core)





How many cores in your phone?

Secondary Storage Unit

- Long-term, high-capacity “warehousing” section
- Programs or data not actively being used by the other units normally are placed on the secondary storage units (e.g., hard drive)
- Information on secondary storage devices is persistent and will be preserved even when the computer is turned off
- Storage devices are typically much cheaper than main memory.

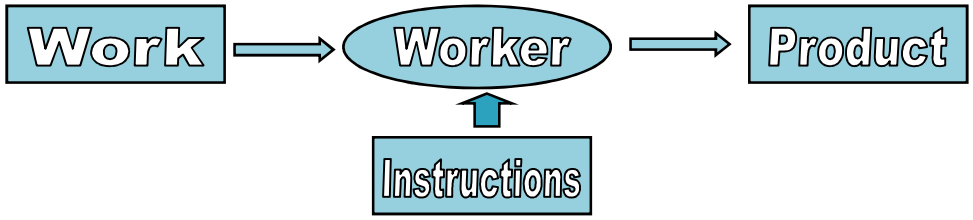


Are They Computers

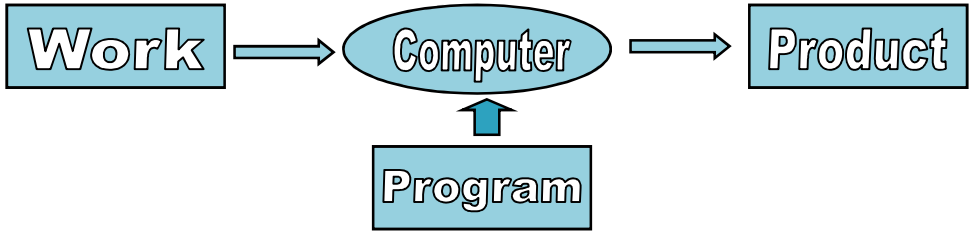
- What is the input unit?
- What is the output unit?
- Do they have CPU, RAM and disk?

What is a computer program?

- Human work model

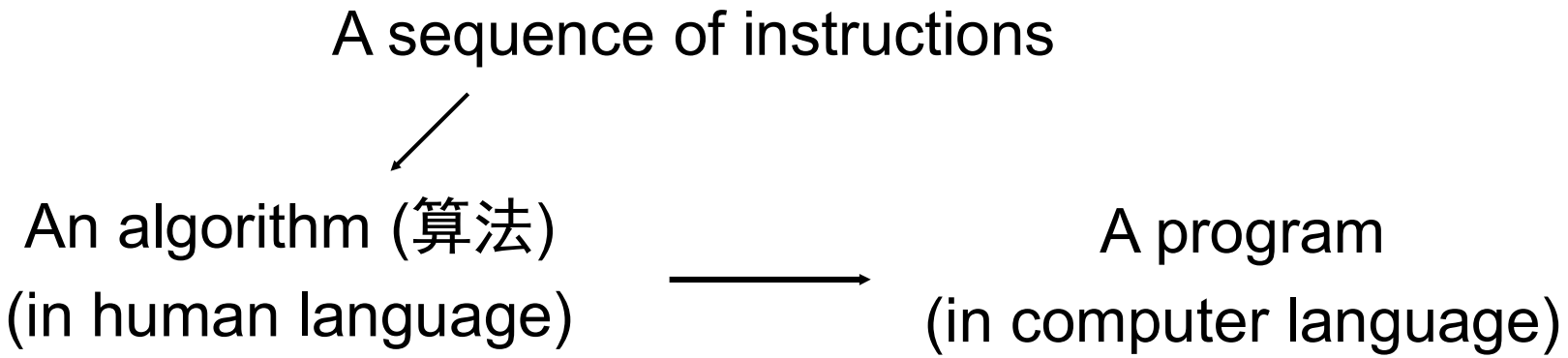


- Computer work model



- 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?



- 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), needs a translator!

Can you understand this?

0000100100101110011001100110100101101100011001010000100100100010011011000110
0101011000110111010001110101011100100110010100110001001011100110001100100010
0000101001100111011000110110001100110010010111110110001101101111011011010111
0000110100101101100011001010110010000101110001110100000101000101110011100110
1100101011000110111010001101001011011110110111000001001001000100010111001110
1000110010101111000011101000010001000001010000010010010111001100001011011000
1101001011001110110111000100000001101000000101000000100100101110011001110110 1
1000110111101100010011000010110110000100000011011010110000101101001011011100
0001010000010010010111001110100011110010111000001100101000010010010000001101
1010110000101101001011011100010110000100011011001100111010101101110011000110
1110100011010010110111101101110000010100000100100101110011100000111001001101 1
1011000110000100100110000001101000000101001101101011000010110100101101110000
1110100000101000001001001000010010001101010000010100100100111101001100010010
1110100011101010101010001010010001100100000001100000000010100000100101110011
011000010111011001

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 beter now?

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

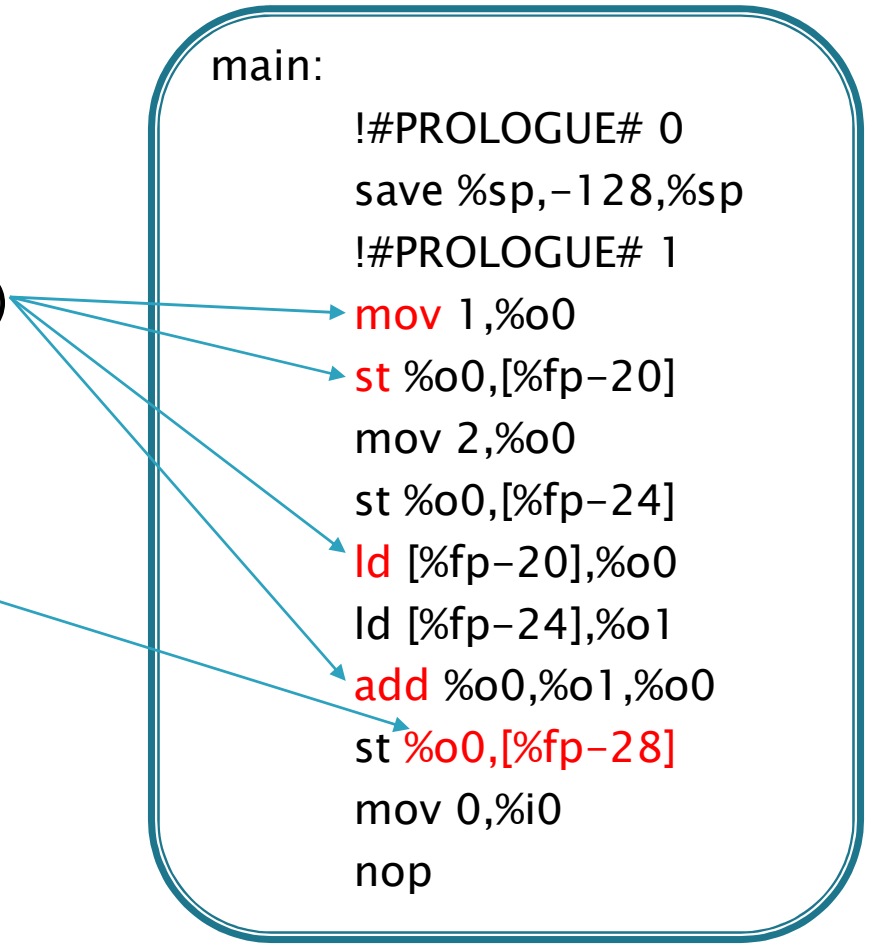
Levels of programming languages

- Machine (binary) language is unintelligible (bits)

0000100100101110011001100110100101101100011001010000100100100010011011000110
0101011000110111010001110101011100100110010100110001001011100110001100100010
0000101001100111011000110110001100110010010111110110001101101111011011010111 0
0000110100101101100011001010110010000101110001110100000101000101110011100110
1100101011000110111010001101001011011110110111000000100100100010001011100111 0
1000110010101111000011101000010001000000101000000100100101110011000010110110 00
1101001011001110110111000100000001101000000101000000100100101110011001110110 1
1000110111101100010011000010110110000100000001101101011000010110100101101110 0
0001010000001001001011100111010001111001011100000110010100001001001000000110 1
1010110000101101001011011100010110000100011011001100111010101101110011000110
1110100011010010110111101101110000001010000001001001011100111000000111001001101 1

Levels of programming languages

- Assembly language (汇编语言) is low level
  - Mnemonic names (助记符) for machine operations
  - Explicit manipulation of memory addresses and contents
  - Machine-dependent

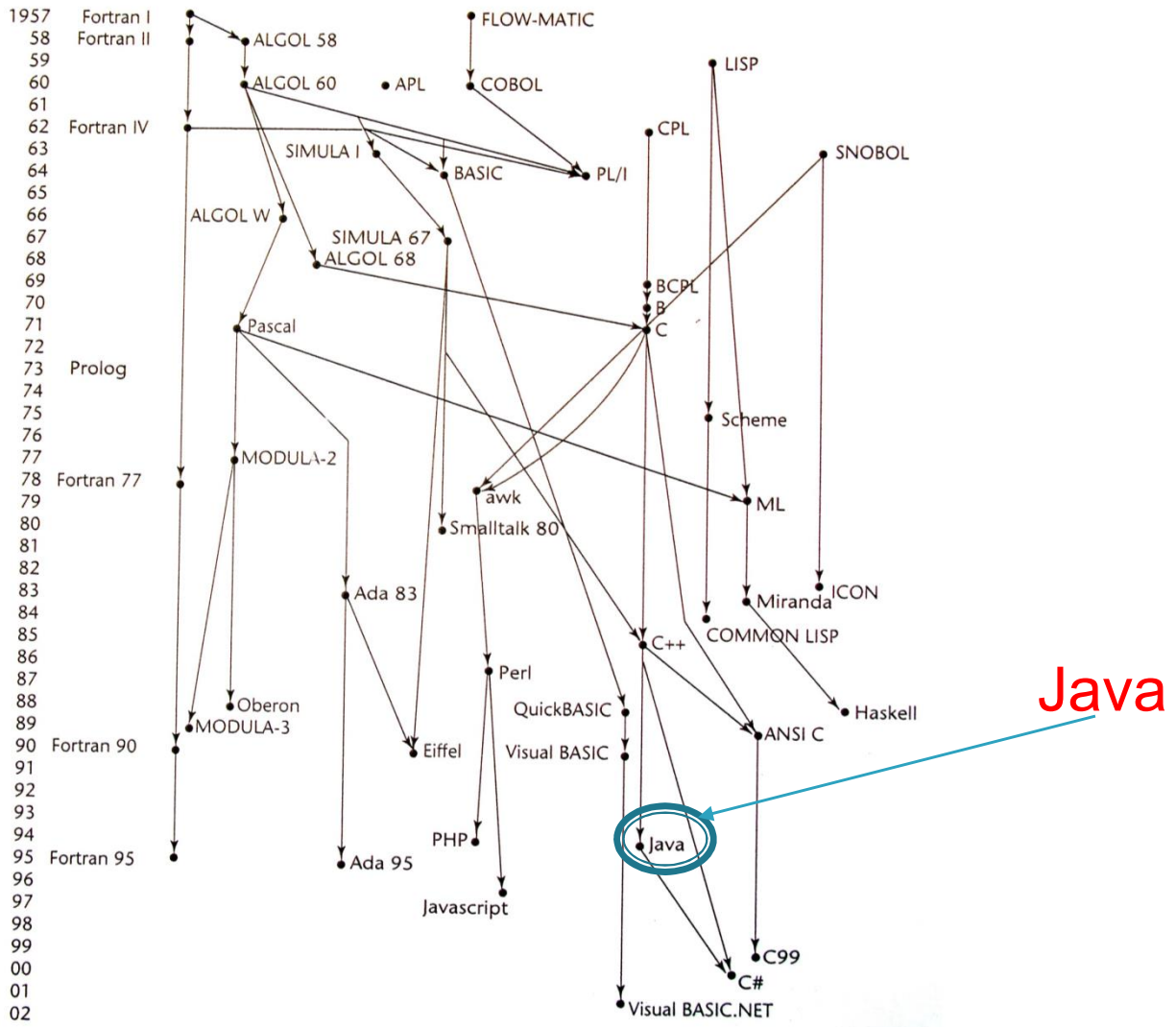


Levels of programming languages

- High-level language
  - Readable: instructions are easy to remember (faster coding)
  - Less error-prone
  - No mention of memory locations
  - Machine-independent = portable

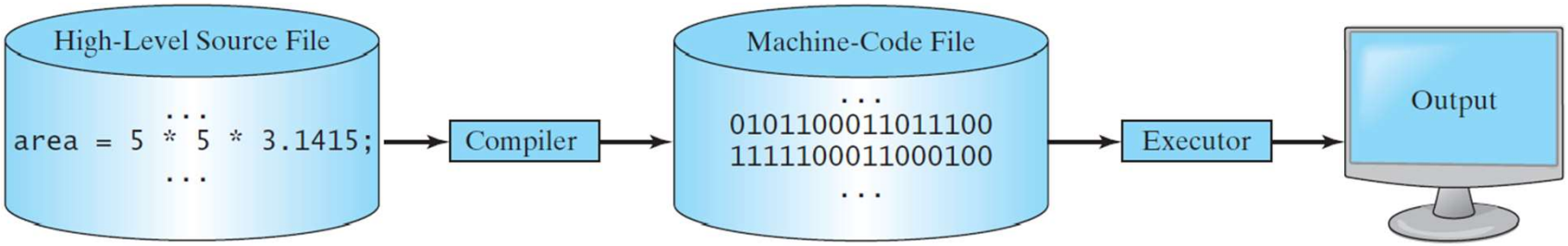
```
int valueofz( )
{
        int x, y, z;
        x = 1;
        y = 2;
        z = x+y;
        return z;
}
```

Genealogy of programming languages



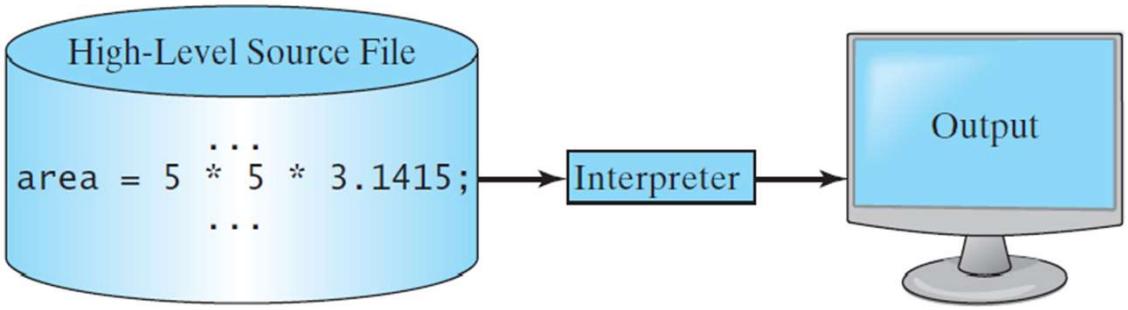
Compilation: from source to executables

- 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) directly executes the statements from source code, without requiring the programs to have been compiled into machine codes.



Compiler vs. Interpreter

Interpreter	Compiler
Interprets and executes one statement at a time.	Scans the entire program and translates it as a whole into machine codes.
Takes less time to analyze the source code but the overall execution is usually slower.	Takes more time to analyze the source code but the overall execution is typically faster.
Continues executing a program until the first error is met, in which case it stops.	Programs are executable only after they are successfully compiled.
Programming languages like Python, Ruby use interpreters.	Programming languages like C, C++ use compilers.

What is software?

A set of programs (also including libraries and non-executable data, e.g., documentation)

- Application software (应用软件): Programs designed for specific tasks. They are typically easy to use.
  - MS Word, PowerPoint, Chrome, Photoshop, WeChat etc.
- System software (系统软件): Programs that support the execution and development of other programs.
  - Operating systems (e.g., Windows, Mac OSX, Linux for desktops, and iOS & Android for mobile devices)
  - Translation systems (e.g., compilers, assemblers)

We learn Java, why?

- An object-oriented computer programming language – today's key methodology
- The most widely used computer programming language – billions of devices run Java programs
- Preferred for Internet-based applications and devices over a network

A brief history of Java

- Microprocessors have a profound impact in intelligent consumer-electronic devices. Personal computers and hand-held devices become possible.
- 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.

The father of Java:
James Gosling

A brief history of Java

- In 1993, Sun saw the potential of using Java to add dynamic content to web pages. Java's connection to the Internet began.
- In 1995, Java was officially released and the Netscape browser (网景浏览器) started to support Java.

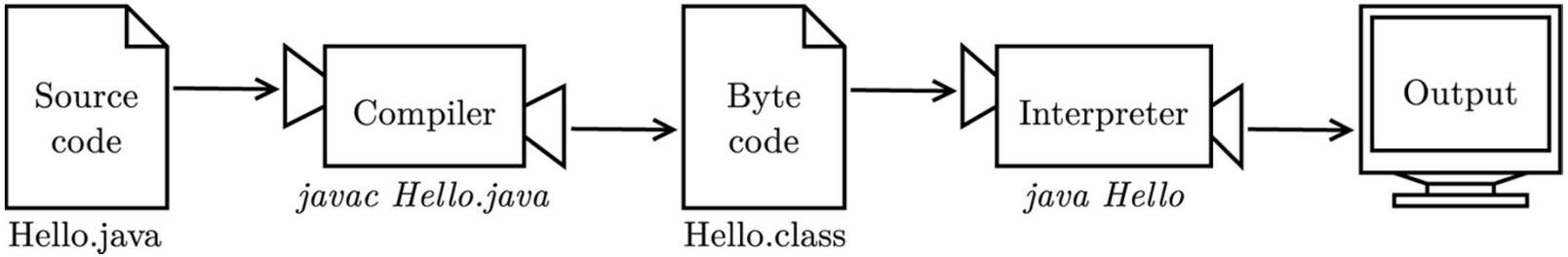
Java Editions

- Java Standard Edition (Java SE)
  - Java SE 11 (long term support) was released in Sept. 2018
- Java Enterprise Edition (Java EE)
  - For large-scale, distributed networking and web-based applications
- Java Micro Edition (Java ME)
  - For small, memory-constrained devices, e.g., micro controllers, sensors, TV boxes etc.

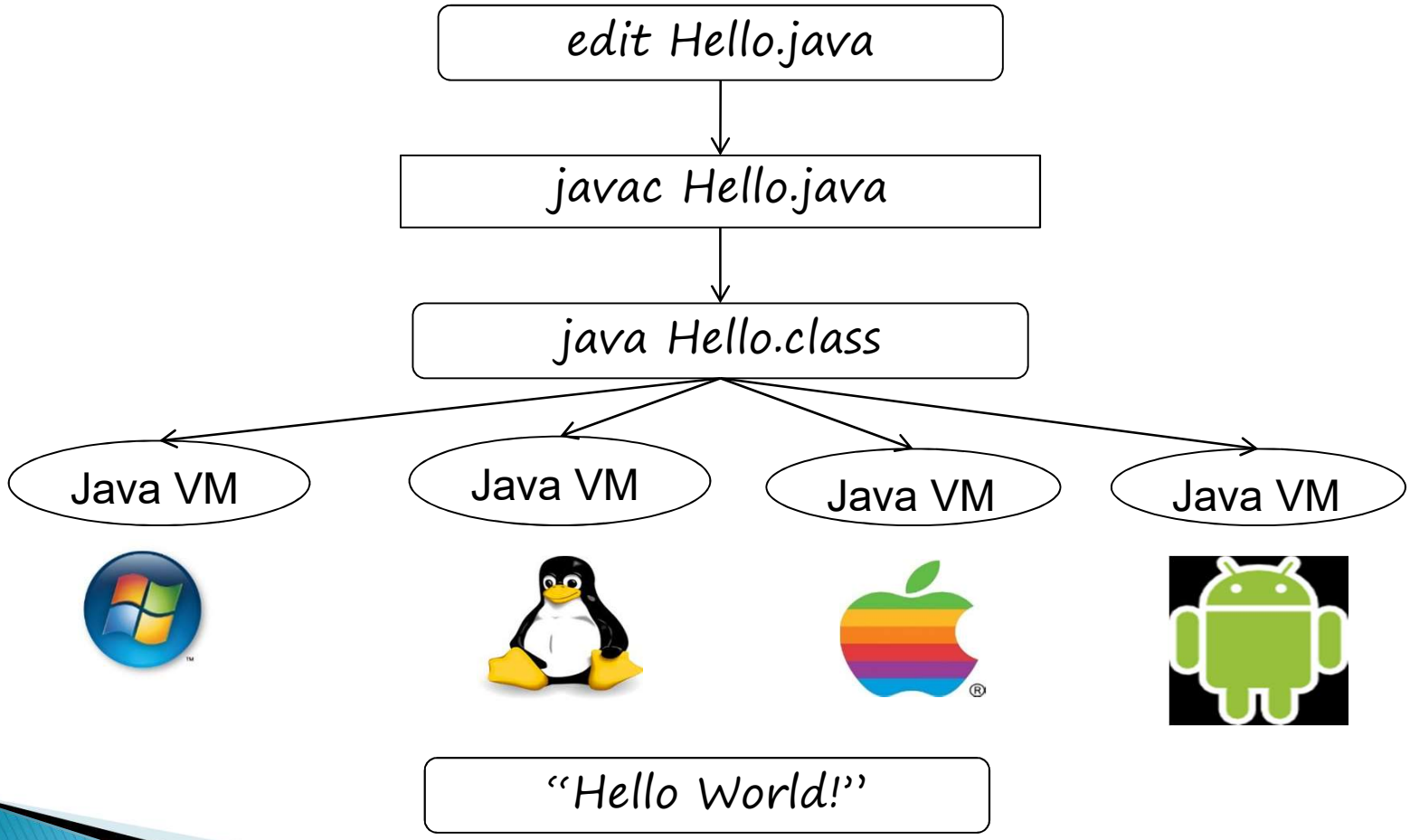
Java programming steps

- Edit (write the program and store it in the disk `.java`)
- Compile (create bytecodes and store them in a file `.class`)
- Load (read `.class` files and put those bytecodes in memory)
- Verify (confirm the bytecodes are valid and secure)
- Execute (run the program in Java Virtual Machine or JVM)

Java is both compiled and interpreted



Java is platform independent



Integrated Development Environment (IDE)

- Combine all the capabilities that a programmer would want while developing software (Eclipse, IntelliJ IDEA, BlueJ, etc.)
  - We will use IDEA in this course (https://www.jetbrains.com/idea/)
  - BlueJ is good for beginners (https://www.bluej.org/)
- Before you begin programming, install JDK (Java SE Software Development Kit) and set the PATH Environment Variable properly (attend the first lab to learn this)
  - http://www.oracle.com/technetwork/java/javase/downloads

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)
  - An interpreter/loader (java)
  - A compiler (javac),
  - An archiver (jar),
  - A documentation generator (javadoc)
  - Other tools needed in Java development.
- In short, JDK = JRE + Development tools

JRE and JVM（虚拟机）

- 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.
- A **Java Virtual Machine (JVM)** is an abstract computing machine that enables a computer to run a Java program.
- In short, JRE = JVM + Library classes

What is debugging?

- The process of tracking down and correcting bugs (errors) in your programs
  - Syntax Errors（语法错误）: Syntax refers to the structure of your program and the rules about that structure (e.g., missing a semicolon at the end of a statement)
  - Runtime Errors (运行时错误, 异常): Runtime errors or exceptions occur when the interpreter is running the byte code and something goes wrong, e.g., an infinite recursion (无限递归) causes a StackOverflowException
  - Logic Errors (逻辑错误): The semantics or meaning of your program are wrong (e.g., it yields an unexpected result)