03 Programming Languages

Instructor: Ke Wei（柯韋）

➽ A319 ✆ Ext. 6452 ✉ wke@ipm.edu.mo

http://brouwer.ipm.edu.mo/COMP112/18/

Bachelor of Science in Computing, School of Public Administration, Macao Polytechnic Institute

September 3, 2018

Outline

1. Programming Languages
2. Imperative Languages
3. Euclid's Algorithm
4. History of Programming Languages
5. Reading Homework

Programming Languages

A programming language is a standardized communication technique for expressing instructions to computers.

- It is a set of *syntactic* and *semantic* rules used to define computer programs.
- It enables programmers to specify what data a computer will act upon, how these data will be stored/transmitted, and what actions to take under various circumstances.

Each programming language is a set of formal specifications concerning syntax, vocabulary, and meaning. A programming language usually includes:

- data manipulation and program sequencing *statements*,
- a *type system*,
- a *module system*,
- design philosophy.

Concepts of Programming Languages

Type system

- All data in modern digital computers are stored as binary numbers.
- Low-level binary data are organized by a programming language into high-level objects.
- How a piece of data is organized and interpreted is determined by the *data type*.

  For example, a 32-bit binary number 01010010110111010101001000101001 can be interpreted
  – as an *integer* 1390236201,
  – as a *floating point number* $4.75283096 \times 10^{11}$, or
  – as two Unicode *characters* “勝利”.

Module system

- Large programs need to be divided into smaller units (modules).
- It simplifies the program complexity.
- It enables program *re-use*.

Imperative Languages

- Review the functions of digital computers:
  - Data transfer between memory and registers
  - Arithmetic and logic operations on data
  - Program sequencing and control
  - Input and output operations
- Imperative programs are a sequence of *commands* for the computer to perform.
  - It describes computations in terms of program states, and statements that change the states.
- Programming languages that enable imperative programming efficiently are called imperative languages. Most *high-level* languages are imperative languages. They support four basic types of statements:

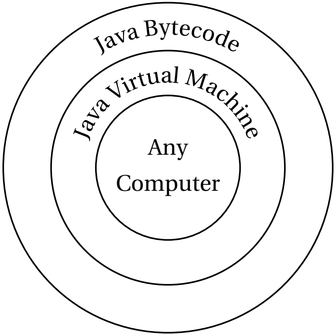
  1) assignments, 2) loops, 3) conditionals, and 4) method invocations.
- In addition, a language also provides means to call system services, typically for input and output.

Compilers and Interpreters

- High-level programming languages, such as Java, simplify your programming life in several ways.
  - You don’t have to express your instructions in a set of numerical code (machine language).
  - The statements you use are much closer to how you might think about a problem than they are to the detailed approach a computer uses.
- However, a computer understands only machine language. Translation is needed.
- A *compiler* is a computer program that translates a program written in a high-level programming language (called the *source* language) into an *equivalent* program in another language (called the *target* language, usually a machine language).
- An *interpreter* interprets a computer program into machine language at runtime. Thus, it enables a program to be executed from its source form. In general, a language can be both compiled and interpreted.
- Compiled programs are usually faster. There are also *Just-In-Time* (JIT) compilers, such as in the *Java Virtual Machine*.

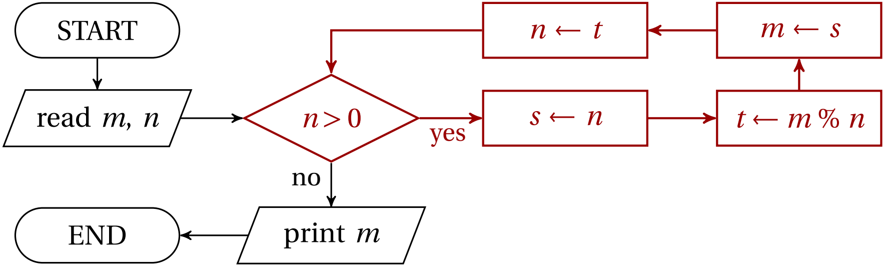
Java Virtual Machine

- A program in Java must be *compiled* to Java *bytecode* to run on the Java Virtual Machine (JVM).
- The bytecode is similar to machine instructions but is *architecture neutral.*
- A bytecode program can run on any platform that has a JVM.
- Java bytecode is usually *interpreted* by the JVM.
- A JIT compiler compiles a segment of bytecode when it is about to be *executed* (hence the name “just-in-time”), and then caches and reuses the result later without recompiling.



Computing the Greatest Common Divisor — Euclid's Algorithm

- The *modulo* operation between two integers m and n, written as $m\,\%\,n$, finds the remainder of m divided by n. We simply read as "m mod n".
- The Euclid's Algorithm is based on the fact, that for $m>0$ and $n>0$, $\gcd(m,n)=\gcd(n,m\,\%\,n)$ and $\gcd(m,0)=m$.
- Because $0 \le (m\,\%\,n) < n$, this substitution will terminate.



Euclid's Algorithm — Java Code

```
public static void main(String[] args) {
    int m, n;
    try ( Scanner scanner = new Scanner(System.in) ) {
        System.out.print("Input␣two␣positive␣integers:␣");
        m = scanner.nextInt();
        n = scanner.nextInt();
    }
    while ( n > 0 ) {
        int s = n;
        int t = m%n;
        m = s;
        n = t;
    }
    System.out.println(m);
}
```

The First Programmer

- Ada Byron, Lady Lovelace (1815-1852)
  - It was at a dinner party at Mrs. Somerville's in November, 1834 that Ada heard Babbage's ideas for a new calculating machine, the Analytical Engine.
  - In 1842-1843, Ada specified in complete detail a method for calculating Bernoulli numbers with the Engine, recognized as the world's first "computer program".
  - A programming language developed by the U.S. Department of Defense was named "Ada" in her honor in 1979.
- The first digital computer was built in 1945: ENIAC (Electronic Numerical Integrator and Computer).
  - 17468 vacuum tubes, 70000 resistors, 10000 capacitors, 1500 relays, and 6000 manual switches.

History of Imperative Programming Languages

Fortran The first widely used high level programming language, was developed during 1954–57 by an IBM team led by John W. Backus.

Pascal Niklaus Wirth developed Pascal in 1970. It is one of the landmark programming languages, variants of which are still widely used today (Delphi).
The popular typesetting system TeX and much of the original Macintosh operating system were written in Pascal.

C Dennis Ritchie developed the C programming language during 1969–1972, initially for porting the Unix operating system to a DEC PDP-11.

Python Created by Guido van Rossum and first released in 1991, that emphasizes code readability and features a dynamic type system and automatic memory management.

Java Sun Microsystems developed Java in 1995.

C# Microsoft presented the C# programming language, which is similar to Java and Delphi, in 2001 to fight Java's popularity.

Reading Homework

Textbook

- Section 2.1 – 2.5, 3.3.

Internet

- History of programming languages (http://en.wikipedia.org/wiki/History_of_programming_languages).

Self-test

- 2.1 – 2.9 (http://tiger.armstrong.edu/selftest/selftest9e?chapter=2).