嵌入式系统实践

王彬

西主楼3-120

课堂形式

考核方式

■ 考查

任一项没有达到则记不通过（F）：

1、无故缺勤。

2、未经同意，没有提交课堂练习（可以比规定时间延后1天补交）。

3、未经同意，没有提交课后作业（可以比规定时间延后1天补交）。

4、作业copy。

评分标准：

- 课程作业完成情况（每日练习+专题作业），占 60%
- 课堂参与情况（课堂抽检），占 20%
- 大作业，占 20%

大作业：

- 开发一款手机APP，思路（供参考）：
  - 通过蓝牙，实现手机与单片机（或智能家具）的多种交互。
  - 获取互联网信息， 在手机端展示。
  - 手机小游戏（或其他应用）
- 作业要求：
  - 不少于3个页面、5个交互操作。
  - 代码行数不少于500行。
- 提交时间：周一（8.29）晚上前。
- 提交形式：程序代码包 + 程序说明文档 + 使用操作文档（或演示视频）。

本节概要

- Java基础知识
- Java数据类型
- Java流程控制
- Java输出输入

清華大學 Tsinghua University

Java基础知识

编程语言的种类

◆程序由编程语言编写，编程语言是计算机能够“读懂”的语言

◆编程语言的发展历程：

代	编程语言
第一代	机器语言 (Machine Language)
第二代	汇编语言 (Assembly Language)
第三代	高级语言 (High Level Language)
第四代	应用程序产生的语言 (Application-Generation Language) 或者查询语言 (Query Language)
第五代	面向逻辑的语言 （人工智能、专家系统的逻辑分析语言）

◆越高级的语言，运行效率越低，开发效率越高

◆越低级的语言，运行效率越高，开发效率越低

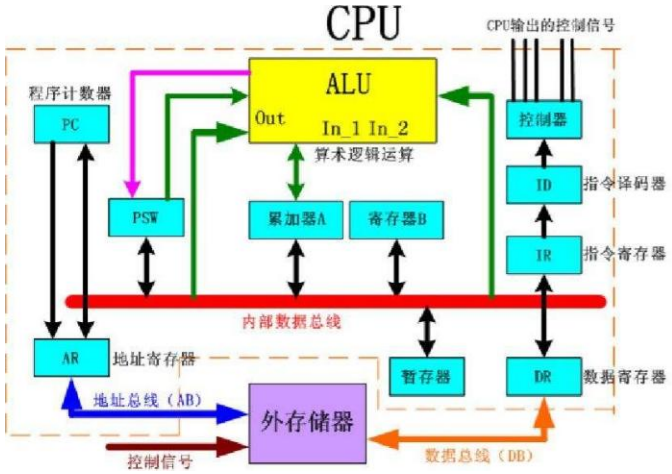
◆低级语言：

1. 机器语言 (Machine Language)

机器语言使用二进制的0和1来表示程序代码，计算机可以直接执行机器语言的程序代码，所以执行效率最高

例如：
0111 0001 0000 1111
1001 1101 1011 0001

学过数电和微机原理就可以理解机器语言的含义



2. 汇编语言 (Assembly Language)

汇编语言使用一些有简单符号组成的指令集来代表机器语言0和1表示的二进制程序代码，汇编语言十分接近机器语言

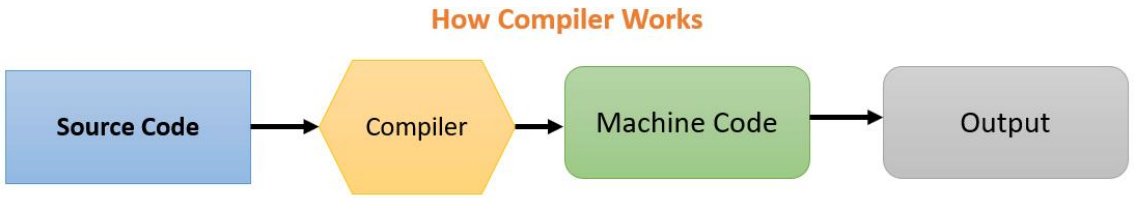
例如：
MOV AX 01
MOV BX 02

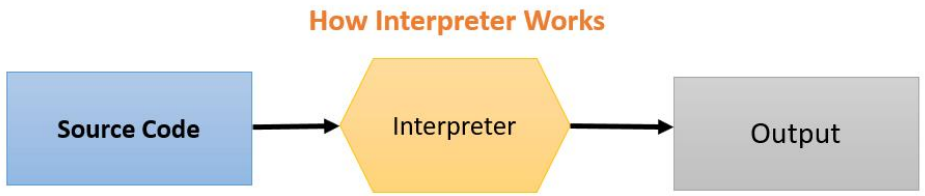
◆高级语言

高级语言是一种接近人类语言的编程语言，由于计算机不能直接读懂高级语言，需要进一步转换翻译成机器语言，计算机才能执行

常见的高级语言：Basic、C/C++、C#、Java、Python等

- 高级语言转译成机器语进行执行的两种方式：使用编译程序(典型的为C)和解释程序(典型的为Javascript)





◆例：C语言程序编译的过程

1. 源代码 ⟶ 2. 汇编代码 ⟶ 3. 机器代码

code.c

```
1  int accum = 0;
2
3  int sum(int x, int y)
4  {
5      int t = x + y;
6      accum += t;
7      return t;
8  }
```

code.s

```
1      .section    __TEXT,__text,regular,pure_instructions
2      .macosx_version_min 10, 11
3      .globl  _sum
4      .align  4, 0x90
5  _sum:                                   ## @sum
6      .cfi_startproc
7  ## BB#0:
8      pushq   %rbp
9  Ltmp0:
10     .cfi_def_cfa_offset 16
11 Ltmp1:
12     .cfi_offset %rbp, -16
13     movq    %rsp, %rbp
14 Ltmp2:
15     .cfi_def_cfa_register %rbp
16     movl    %edi, -4(%rbp)
17     movl    %esi, -8(%rbp)
18     movl    -4(%rbp), %esi
19     addl    -8(%rbp), %esi
20     movl    %esi, -12(%rbp)
21     movl    -12(%rbp), %esi
22     addl    _accum(%rip), %esi
23     movl    %esi, _accum(%rip)
24     movl    -12(%rbp), %eax
25     popq    %rbp
26     retq
27     .cfi_endproc
28
29     .globl  _accum                  ## @accum
30 .zerofill __DATA,__common,_accum,4,2
31
32 .subsections_via_symbols
```

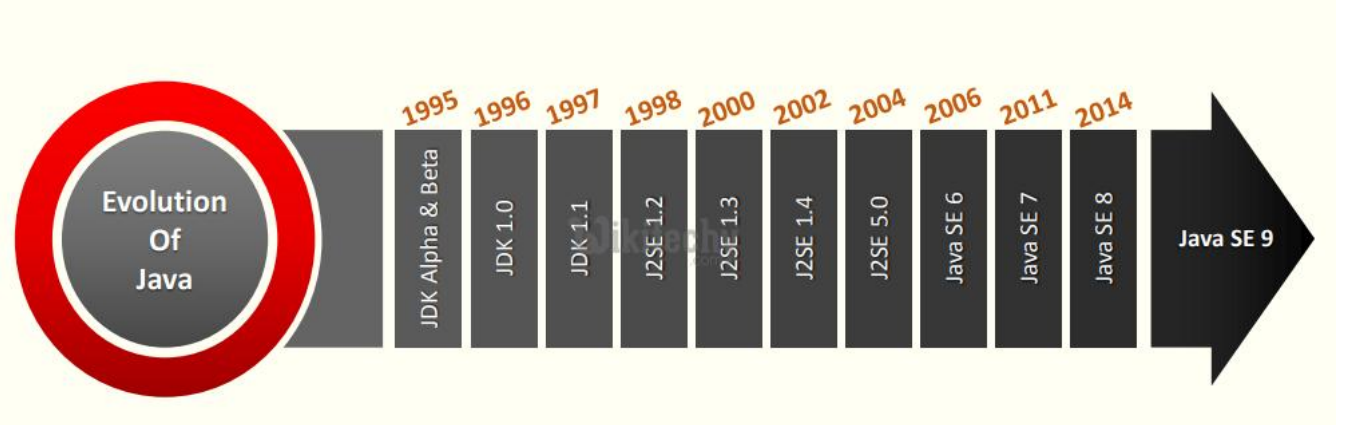
code.o

```
1  cffa edfe 0700 0001 0300 0000 0100 0000
2  0400 0000 0002 0000 0020 0000 0000 0000
3  1900 0000 8801 0000 0000 0000 0000 0000
4  0000 0000 0000 0000 0000 0000 0000 0000
5  8c00 0000 0000 0000 2002 0000 0000 0000
6  8800 0000 0000 0000 0700 0000 0700 0000
7  0400 0000 0000 0000 5f5f 7465 7874 0000
8  0000 0000 0000 0000 5f5f 5445 5854 0000
9  0000 0000 0000 0000 0000 0000 0000 0000
10 2700 0000 0000 0000 2002 0000 0400 0000
11 a802 0000 0200 0000 0004 0080 0000 0000
12 0000 0000 0000 0000 5f5f 636f 6d6d 6f6e
13 0000 0000 0000 0000 5f5f 4441 5441 0000
14 0000 0000 0000 0000 8800 0000 0000 0000
15 0400 0000 0000 0000 0000 0000 0200 0000
16 0000 0000 0000 0000 0100 0000 0000 0000
17 0000 0000 0000 0000 5f5f 636f 6d70 6163
18 745f 756e 7769 6e64 5f5f 4c44 0000 0000
19 0000 0000 0000 0000 2800 0000 0000 0000
20 2000 0000 0000 0000 4802 0000 0300 0000
21 b802 0000 0100 0000 0000 0002 0000 0000
22 0000 0000 0000 0000 5f5f 6568 5f66 7261
23 6d65 0000 0000 0000 5f5f 5445 5854 0000
24 0000 0000 0000 0000 4800 0000 0000 0000
25 4000 0000 0000 0000 6802 0000 0300 0000
26 0000 0000 0000 0000 0b00 0068 0000 0000
27 0000 0000 0000 0000 2400 0000 1000 0000
28 000b 0a00 0000 0000 0200 0000 1800 0000
29 c002 0000 0200 0000 e002 0000 1000 0000
30 0b00 0000 5000 0000 0000 0000 0000 0000
31 0000 0000 0200 0000 0200 0000 0000 0000
32 0000 0000 0000 0000 0000 0000 0000 0000
```

Java的历史和发展

- Sun公司GreenTeam小组，Oak@StarSeven
- 开发一种能够在各种消费性电子产品（如机顶盒、冰箱、收音机等）上运行的程序架构
- 从Oak到爪哇岛特产的咖啡Java （三选一 Silk， DNA， Java ）
- 第一个浏览器Mosaic诞生，Java进入网络应用
- HotJava和Java之父James Gosling
- 1995.5.23，JDK 1.02a版本正式发布，Java正式诞生

Write Once，Run Anywhere



Java SE 18

感兴趣的同学可以阅读https://www.cnblogs.com/CodeMagicMaster/p/8064077.html

Java 不只是一门编程语言

Java technology is both
a programming language and a platform.

◆ 开发语言

“C++” - “复杂性和奇异性” + “安全性和可移植性”

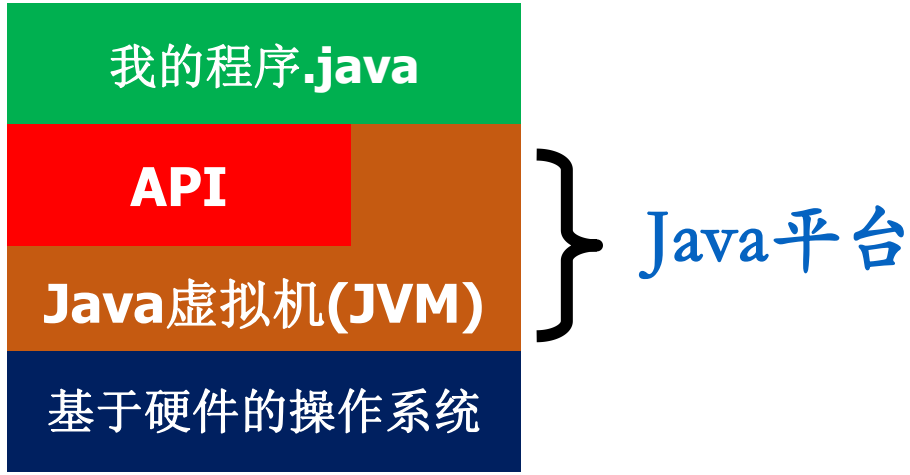
□面向对象	□简单安全
□跨平台	□多线程
□高性能	□鲁棒可靠
□分布式	□动态性能

http://www.java.com/en/everywhere/

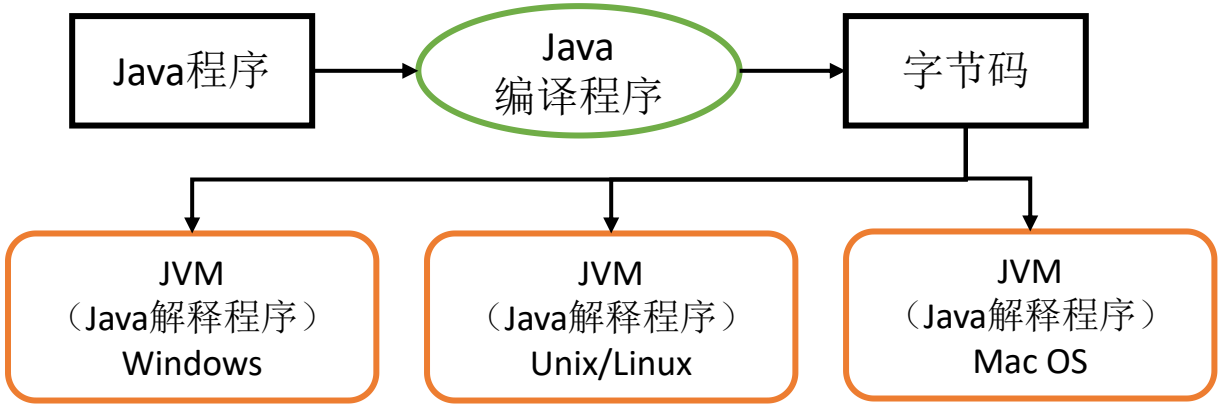
Java 不只是一门编程语言

◆ Java 平台

Java平台是一个纯软件平台，由于运行Java程序，它可以在各种基于硬件的平台上运行，与硬件无关，主要由JVM和Java API两部分组成。



1. JVM （Java虚拟机）



Java 程序由Java虚拟机程序执行（或解释执行）

2. Java APIs (应用程序接口)

经过编译的，可在程序中使用的Java代码标准库.

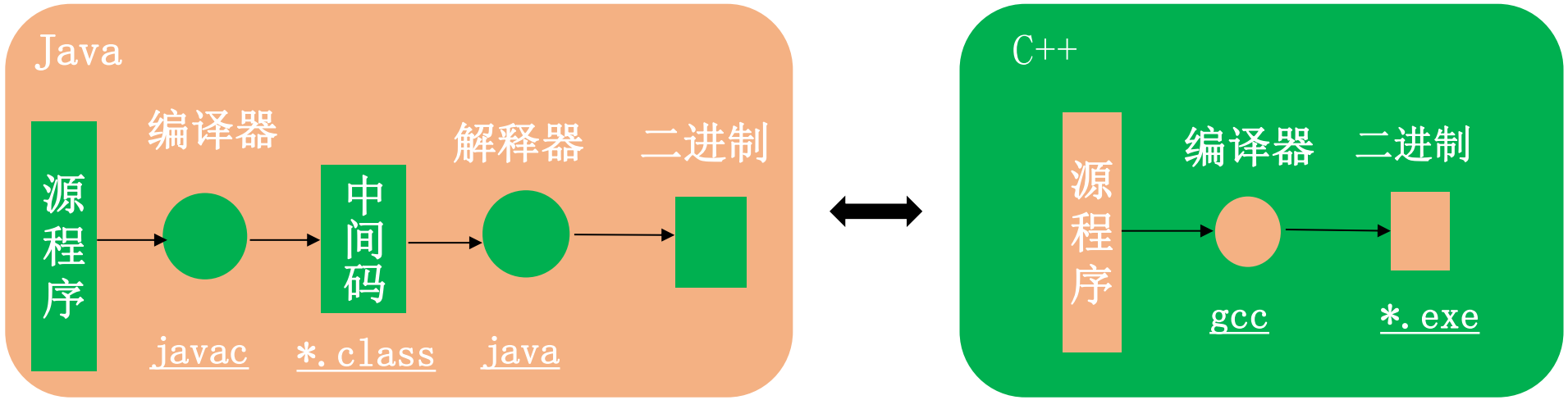
Java语言基本原理

◆Java 设计目标

- 创建一种面向对象的程序设计语言
- 提供一个程序运行的解释环境，使程序代码独立于平台
- 吸收C和C++的优点，使程序员容易掌握
- 去掉C和C++中影响程序健壮性的部分，如：指针，内存申请和释放

Java语言编译和运行

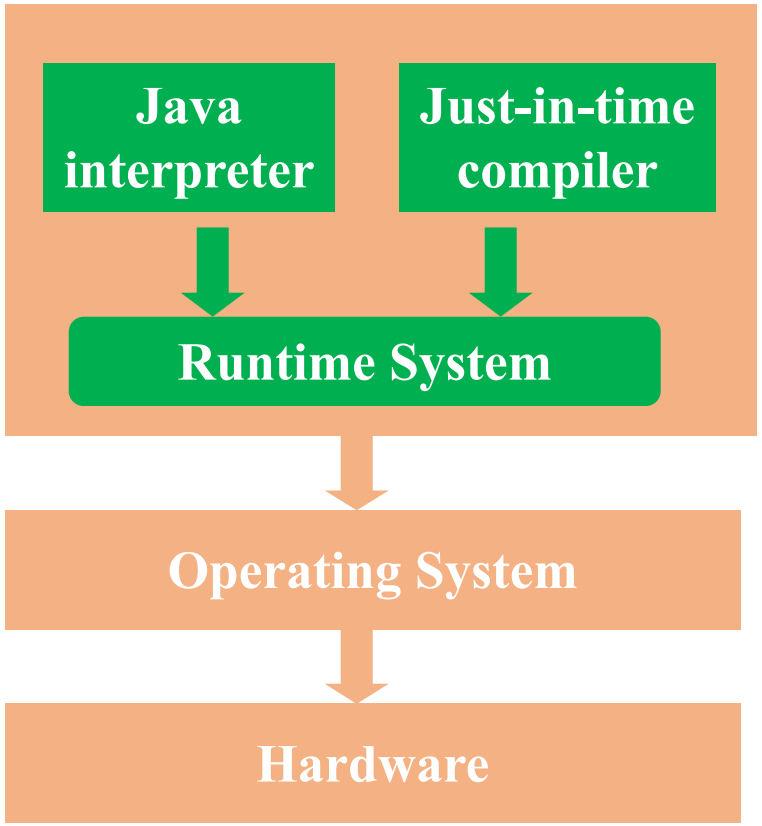
◆ **Java中的编译和解释**



Java语言中源程序以*.java*为后缀。源代码先被编译成一个中立的字节码（*byte code*），然后再被装载到有Java虚拟机（*Java Virtual Machine, JVM*） 的硬件中执行。

Java语言编译和运行

◆ **Java虚拟机**

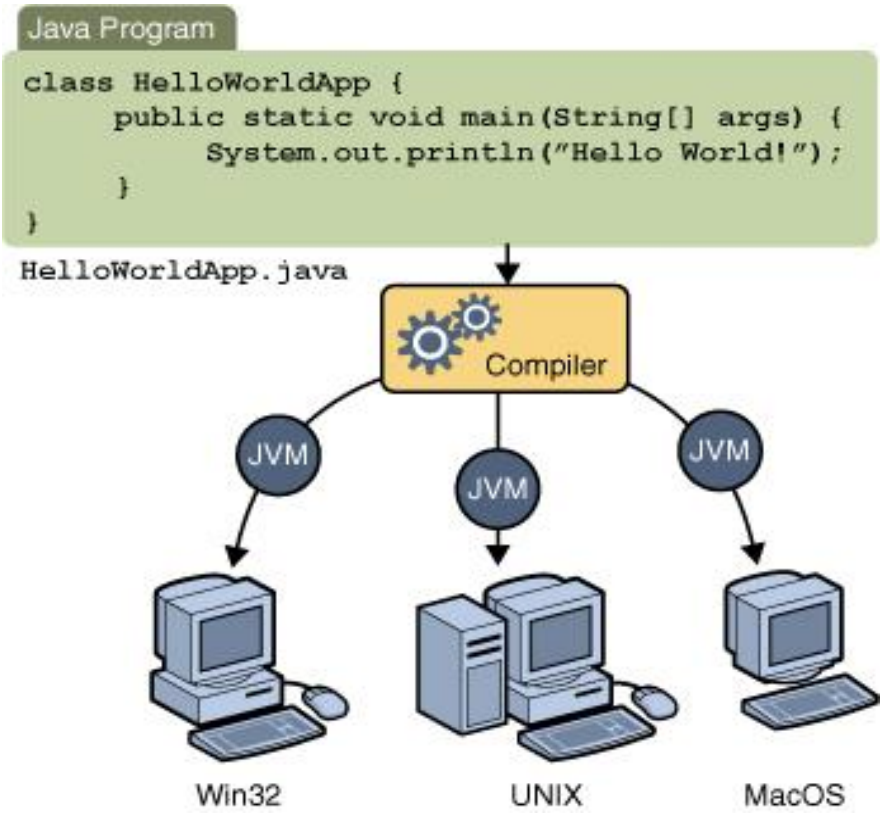


- 不同的操作系统有不同的虚拟机。它类似一个小巧而高效的CPU
- Bytecode代码是与平台无关的是虚拟机的机器指令
- Java字节代码运行的两种方式
  - Interpreter(解释方式)
  - Just-in-time(即时编译):由代码生成器将字节代码转换成本机的机器代码，然后可以以较高速度执行.

Java语言编译和运行

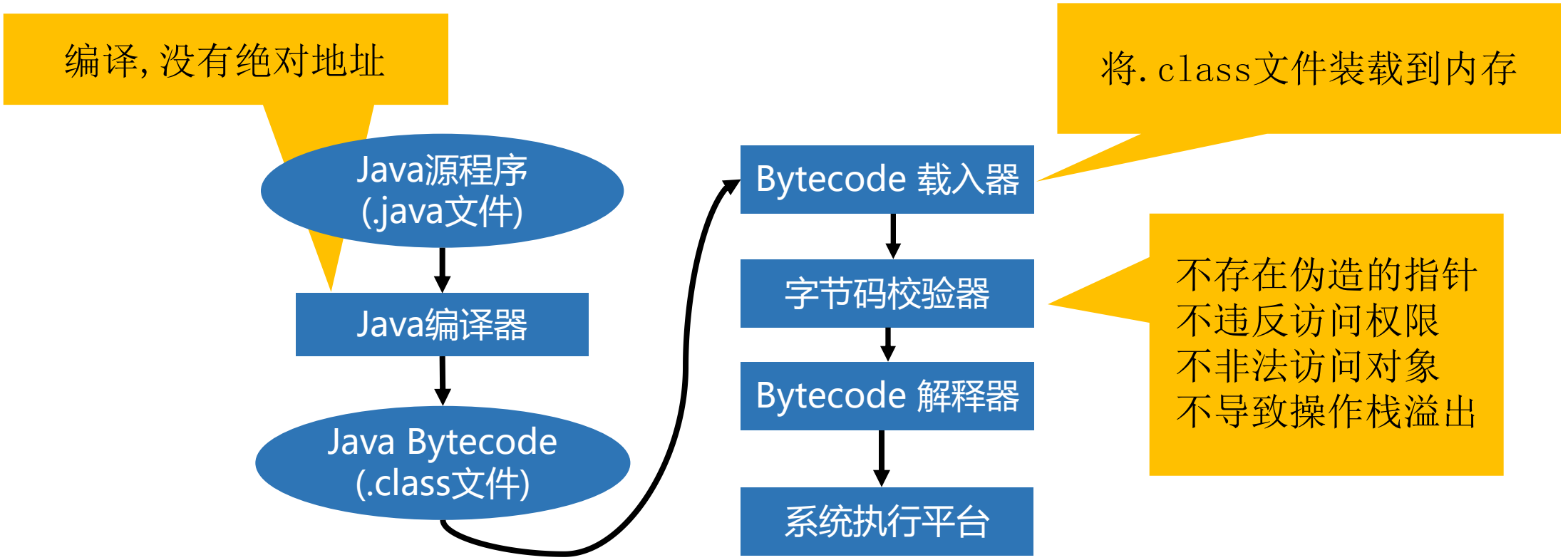
◆**一次编译，各处使用**

可以针对不同的硬件和操作系统设计不同的JVM，并使这些JVM都可对标准的二进制中间码解释执行。这样，Java程序就可以实现**“一次编译，各处使用”**



超强的跨平台性能

Java语言编译和运行



◆ **Java是一个安全的高级语言，适合网络应用，自己管理内存**

Java语言编译和运行

◆为什么Java语言如此的优秀？

三个底层实现机制：

- **虚拟机** Java virtual machine (JVM)
  —— 一次编译，各处运行！
- **垃圾内存管理** Garbage collection(GC)
  —— 设计简化，编写快捷
- **代码安全** Code security
  —— 没有指针，内存安全，多线程

◆此外，Java语言还具有这些特点

- 分布式
  支持各种网络协议，能够建立分布式(Distributed) 主从架构的应用程序，轻松访问网络上其他主机的资源
- 多线程
  Java语言原生支持多线程(Multi-Threading)，在同一程序能够创建多条“轻量级进程”(Light Weight Process)，并且支持同步功能(Synchronized 关键字)
- 异常处理
  Java具备完善的异常处理(Exception Handling) 机制，使得程序更加“强壮”

Java语言的开发和运行环境

◆编程开发方式

- 终端机模式的开发环境

  这种方式也可以称为“命令行”模式，一般在服务器上使用。使用Vim、Emacs等纯编辑器进行代码开发，调用命令行工具进行程序的编译和调试。

- 集成开发环境（Integrated Development Environment）

  在同一个应用程序中对所开发程序进行编辑、编译、执行和调试，典型的IDE为开发C/C++/C#语言程序的Visual Studio。常见的Java IDE有Eclipse、Netbeans IDE

◆Java 开发环境

- JRE是Java运行环境，通常不同的操作系统和硬件平台有不同的JRE，要想运行Java程序必须先安装JRE
- JDK是Java开发工具，编译器和调试器等

Java语言的开发和运行环境

◆Java Runtime Environment (JRE)

- 包括:
  - Java 虚拟机
  - 组成Java 2 平台API的类
  - 帮助文档
- 不包括:
  - 编译器和调试器等开发工具

◆Java SE Development Kit (JDK)

开发工具
运行环境
附加库
Applets 和 Applications 的演示
……

◆ 本课程使用的JDK版本为JDK 8（也称1.8）

JDK的安装与使用

- 直接运行“jdk-8u211-windows-x64.exe”，按照安装向导进行安装
- JDK安装完毕之后，需要将JDK中的\bin目录添加到系统环境变量中

JDK 环境简要说明

- \bin目录：Java开发工具，包括Java编译器、解释器等
- \lib目录：Java开发类库
- \jre目录： Java运行环境，包括Java虚拟机、运行类库等

JDK的安装与使用

◆JDK 中包含的Java开发工具

- Javac：

  Java编译器，用来将java程序编译成 Bytecode
- Java：

  Java解释器，执行已经转换成Bytecode的java应用程序
- Jdb：

  Java调试器，用来调试java程序
- Javap：

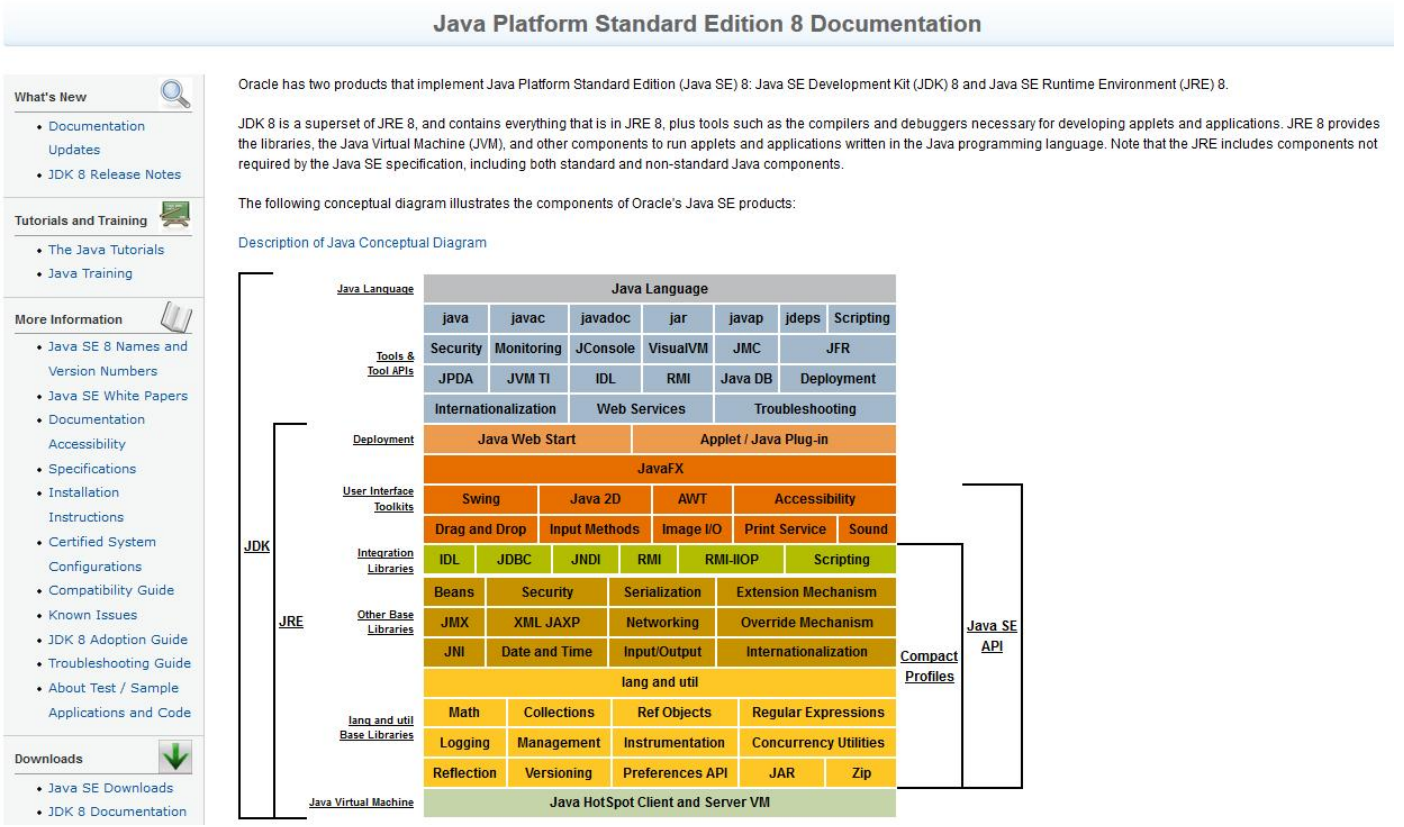
  反编译，将类文件还原回方法和变量
- Javadoc：

  文档生成器，创建HTML文件
- Appletviwer：

  Applet解释器， 用来解释已经转换成Bytecode的java小应用程序

Java 开发必备说明书——Java API Doc

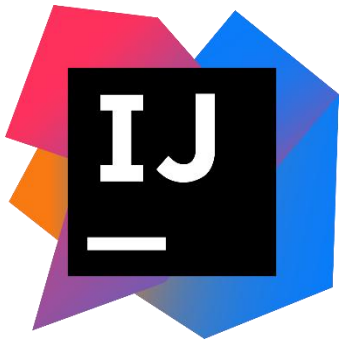
◆学会看API Doc进行开发是程序员的必备素养



◆JDK 8 API Doc： https://docs.oracle.com/javase/8/docs/api/

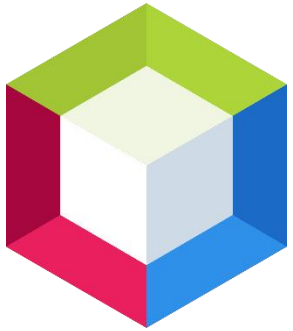
Java 集成开发环境

- 由开源社区进行维护
- 插件众多，功能强大



IntelliJ IDEA

- 快捷技巧多
- 开发效率高



Netbeans

- 由SUN维护
- 高效图形界面开发

◆ 由于Eclipse属于轻量级IDE，配合Android开发插件即可进行Android App开发，本课程使用Eclipse IDE

第一个Java程序

◆ 第一个Java程序： Hello World 源码：

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

- 编辑存盘： 主类名(含有main方法)和文件名要一致性 HelloWorldApp.java
- 编译程序： javac HelloWorldApp.java
- 运行程序： java HelloWorldApp
- 运行结果： ***Hello World***

练习1-1

思考：Java程序编译、执行和C/C++程序有什么不同？

第一个Java程序

◆如果是C/C++? C++的Hello World程序源码：

```
int main() {
        printf("Hello World");
        return 0;
}
```

- 存盘到helloworld.cpp
- 编译程序 VS6.0中按F5编译，或是用gcc helloworld.cpp
- 执行程序 helloworld.exe
- 运行结果 Hello World

Java 中编译出来的是*.class文件，而C++编译出来的是*.exe文件

Java：真正的面向对象的程序语言

◆ Java程序的结构与C语言或者Basic语言不一样，因为Java语言是一门真正的面向程序语言。Java的程序架构是一个“类”(Class)的定义。

类的主体部分

```
public class HelloWorldApp{
    public static void main(String args[]){
        System.out.println(“Hello World”);
    }
}
```

◆命名

如果.java 文件包含一个public 类，它必需按该类名命名

◆类个数

一个源文件中最多只能有一个public类。其它类的个数不限

Java程序结构与基本输出

- package 语句　　//0或1条，必须放在文件开始
- import 语句　　/*0或多条，必须放在所有类定义之前*/
- public class Definition　　//0或1个 文件名必须与该类的类名完全相同
- Class Definition　　//0或多个
- Interface Definition　　//0或多个

Java程序结构与基本输出

◆变量与常量

■ 变量

基本存储单元，它的定义包括变量名、变量类型和作用域几个部分，变量的值可以被改变

```
1   int total = 1234;
2   double rate = 0.05;
```

■ 常量

常量一旦被初始化以后就不可改变，用保留字final来实现常量的定义与声明

```
17  final int NUM1 = 100;
18  final int NUM2 = 200;
```

Java程序结构与基本输出

◆标识符

- 标识符：程序员对程序中的各个元素加以命名时使用的命名记号称为标识符（identifier）
- Java中，标识符是以字母、下划线（_）、美元符($)开始的一个字符序列，后面可以跟字母、下划线、美元符、数字
  - identifier
  - username
  - user_name
  - _sys_var1
  - $change

◆思考：能否使用中文作为标识符？为什么？

Java程序结构与基本输出

◆ Java 关键字

具有专门的意义和用途，不能当作一般的标识符使用，这些标识符称为保留字：

abstract break byte boolean catch case
class char continue default double do else
extends false final float for finally if import
implements int interface instanceof long
length native new null package private
protected public return switch synchronized
short static super try true this throw throws
threadsafe transient void while

Java程序结构与基本输出

◆main()方法

- **Java Application**程序的入口是**main()**方法。它有固定的书写格式：

```
public static void main(String args[]) {
……
}
```

注意：并不是每个类都必须有main方法，往往大多数类没有main方法，而由有main方法的类来调用

Java程序结构与基本输出

◆ 程序内容

在本例中，程序内容只有一条语句：

```
System.out.println(“Hello World”);
```

这个语句实际上调用了System.out类中的println方法，将文字内容输出至控制台(Console)来显示。在Eclipse中进行运行时，为Eclipse工作界面中的“Console”窗口。

◆ 在未介绍对象和方法之前，同学们可以先把该语句看作标准的**Java输出程序代码**。

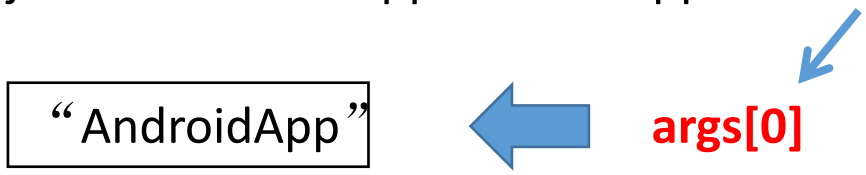
Java 命令行参数

◆在main函数的声明中，我们可以发现main函数具有一个形参

public static void main (**String[] args**)

String[] args是命令行中传进的参数

例如，命令行执行 java HelloWorldApp AndroidApp　**xxx**



练习1-2

◆ 修改HelloWorldApp程序，程序运行后Console显示”Hello xxx”，xxx为命令行第一个参数。

Java 编程规范

◆程序语句

程序语句的范例：

```
1    int total = 1234;
2    double rate = 0.05;
3    double interest = total * rate;
4    System.out.println("Hello!");
```

1. 程序语句必须以英文分号结束
2. 注意运算符号的前后都需要有一个空格
3. 变量命名需要清晰可读，避免“i, j, k, a, b, c”这样指代不明的变量名称，做到“代码即文档”

Java 编程规范

◆程序注释

程序注释是程序的重要部分，良好的注释文字不仅利于别人阅读你写的代码，还利于自己调试，快速定位bug。

Java语言的注释以“//”符号开始（本行有效）

```
6   // 这是一句注释
7   int ID1 = 12345;
8
9   int ID2 = 23456; //这也是一句注释
```

如果注释较长，可以使用 /* 和 */ 来进行多行注释

```
11  /*
12  这是一段注释
13  这是一段注释
14  这是一段注释
15  */
```

Java 编程规范

◆变量命名规则

- **包名**:全小写,一般为名词,如:package shipping.objects
- **类名**:首字母大写,每个单词的首字母大写, 如:class HelloWorldApp
- **接口名**:同类名,如: interface AccountBook
- **方法名**:一般为动词,小写字母开头,每个单词的首字母大写,
  如:balanceAccount
- **变量名**:小写字母开头,每个单词的首字母大写,一般为名词, 如
  length,userAccount
- **常量名**:全大写,用下划线隔开，例如:HEAD_COUNT

清華大學 Tsinghua University

Java数据类型

Java的数据类型

◆Java语言是一种“强类型”（Strongly Typed）程序语言，对变量需要指定数据类型。

◆Java语言的数据类型分为“基本” (Primitive) 和“引用” (Reference) 两种数据类型：

- 基本数据类型：Java提供byte、short、int、long、float、double、char和boolean共8种基本数据类型。
- 引用数据类型：此类型的变量值是内存地址，即对象保存的地址，关于引用数据类型的详细说明在类和对象一节中详细介绍。

Java的数据类型

◆整数类型（Integral Types）

1. 整数类型的种类

整数类型按照所能表示整数的范围，可以分为四种

整数类型	位数	范围
byte	8	-128~127
short	16	-32768~32767
int	32	-2147483648~2147483647
long	64	-9223372036854775808~ 9223372036854775807

Java的数据类型

◆整数类型（Integral Types）

2. 整数字面值

Java程序代码中可以直接使用“整数字面值”(Integral Literal)，并且可以使用十进制、八进制（“0”开头）和十六进制（“0x”开头）

整数字面值	十进制值	说明
44	44	十进制整数
o256	174	八进制整数
0xef	239	十六进制整数
0x3e6	998	十六进制整数

Java的数据类型

◆浮点型

1. 浮点型的种类

浮点型按照所能表示浮点数的范围，可以分为两种

浮点型	位数	作用域
float	32	1.4023e-45~3.4028e38
double	64	4.94066e-324~1.79769e308

Java的数据类型

◆浮点型

2. 浮点字面值

Java程序代码中可以直接使用“浮点字面值”(Floating Point Literal)，默认是double。可以使用科学计数法来表示浮点数。

浮点数字面值	十进制值	说明
0.0123	0.0123	浮点数
.00567	0.00567	浮点数
1.25e4	12500.0	使用科学计数法表示的浮点数

Java的数据类型

◆数值型字面值的下划线记号

Java程序代码允许在数值型字面值使用下划线“_”进行数字的分割（不影响其含义），用于提高数字的可读性，比如：

```
20    long creditCardNumber = 1234_5678_9012_3456L;
21    float pi = 3.14_15F;
22    long hexBytes = 0x00_EC_FF_5E;
```

Java的数据类型

◆布尔类型和布尔值

- 布尔类型表示一个逻辑量， 有两个取值：true和false

- 例如:

```
boolean is_salaried;
boolean is_hourly;
is_salaried = true; //将 is_salaried设置为true
is_hourly = false; //将is_hourly设置为false
```

Java的数据类型

◆char型

- 表示字符，Java使用Unicode字符
- **字符常量：**
  字符常量是用单引号括起来的一个字符，如'a'，'A';
- **字符型变量：**
  类型为char，它在机器中占16位，其范围为0～65535。字符型变量的定义如：
  char c=‘a’; //变量c为char型，初值为'a’
  char c=‘1’; //变量c为char型，初值为’a’
  Int　c = 1

◆ 思考：char型变量中能不能存贮一个中文汉字?为什么?

Java的数据类型

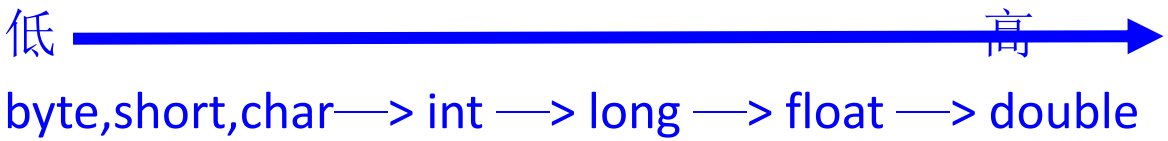
◆基本数据类型例程

```
public class Assign {
        public static void main (String args [ ] ) {
        int x , y ;                    //定义x，y两个整型变量
        float z = 1.234f ;             //指定变量z为float型，且赋初值为1.234
        double w = 1.234 ;             //指定变量w为double型，且赋初值为1.234
        boolean flag = true ;          //指定变量flag为boolean型，且赋初值为true
        char c ;                       //定义字符型变量c
        String str ;                   //定义字符串变量str
        String str1 = " Hi " ;         //指定变量str1为String型，且赋初值为Hi
        c = ' A ' ;                    //给字符型变量c赋值'A'
        str = " bye " ;                //给字符串变量str赋值"bye"
        x = 12 ;                       //给整型变量x赋值为12
        y = 300;                       //给整型变量y赋值为300
        }
    }
```

Java的数据类型转换

◆自动类型转换

整型,浮点型,字符型数据可以混合运算。运算中，不同类型的数据先转化为同一类型，然后进行运算，转换从低级到高级；



◆强制类型转换

高级数据要转换成低级数据，需用到强制类型转换，如：

```
int i;
byte b=(byte)i; /*把int型变量i强制转换为byte型*/
```

Java 的运算符

- **按操作数的数目划分**
  - 一元运算符：＋＋，－－，＋，－
  - 二元运算符：＋，－，>
  - 三元运算符：？：
- **按功能划分**
  - 算术运算符 +，—，*，/，%，++，——
  - 关系运算符 >，<，>=，<=，==，!=
  - 布尔逻辑运算符：!，&&，||
  - 位运算符：>>，<<，>>>，&，|，^，~
  - 赋值运算符 =，及其扩展赋值运算符如+=，—=，*=，/=
  - 条件运算符 ?：
  - 其它

算数运算符

算术运算符

运算符	运算	范例	结果
+	正号	+3	3
-	负号	b=4;-b;	-4
+	加	5+5	10
-	减	6-4	2
*	乘	3*4	12
/	除	5/5	1
%	取模	5%5	0
++	自增（前）	a=2;b=++a;	a=3;b=3
++	自增（后）	a=2;b=a++;	a=3;b=2
--	自减（前）	a=2;b=--a	a=1;b=1
--	自减（后）	a=2;b=a--	a=1;b=2
+	字符串相加	"He"+"llo"	"Hello"

算数运算符

◆算术运算符的注意问题

- ■ 如果对负数取模，可以把模数负号忽略不记，如：5%-2=1。但被模数是负数就另当别论。
- ■ 对于除号“/”，它的整数除和小数除是有区别的：整数之间做除法时，只保留整数部分而舍弃小数部分。
  - 例如：**int x=3510;x=x/1000*1000; x的结果是？**

赋值运算符

- 符号：

  = , +=, -=, *=, /=, %=

- 示例：

  int a,b,c; a=b=c =3;

  int a = 3;   a+=5;等同运算a=a+5;

- 思考：

  short s = 3;

  s=s+2;

  s+=2;

  有什么区别？

比较运算符

比较运算符

运算符	运算	范例	结果
==	相等于	4==3	false
!=	不等于	4!=3	true
<	小于	4<3	false
>	大于	4>3	true
<=	小于等于	4<=3	false
>=	大于等于	4>=3	false
instanceof	检查是否是类的对象	"Hello" instanceof String	ture

- 注1：比较运算符的结果都是boolean型，也就是要么是true，要么是false。
- 注2：比较运算符“==”不能误写成“=” 。

逻辑运算符

逻辑运算符

运算符	运算	范例	结果
&	AND(与)	false&true	false
|	OR（或）	false|true	true
^	XOR（异或）	true^false	true
!	Not（非）	!true	false
&&	AND(短路)	false&&true	false
||	OR(短路)	false||true	true

- 逻辑运算符用于连接布尔型表达式，在Java中不可以写成3<x<6，应该写成x>3 & x<6 。
- “&”和“&&”的区别：
  - 单&时，左边无论真假，右边都进行运算；
  - 双&时，如果左边为真，右边参与运算，如果左边为假，那么右边不参与运算。

  “|”和“||”的区别同理，双或时，左边为真，右边不参与运算。
- 异或(^)与或(|)的不同之处是：当左右都为true时，结果为false。

位运算符

位运算符		
运算符	运算	范例
<<	左移	3 << 2 = 12 --> 3*2*2=12
>>	右移	3 >> 1 = 1 --> 3/2=1
>>>	无符号右移	3 >>> 1 = 1 --> 3/2=1
&	与运算	6 & 3 = 2
|	或运算	6 | 3 = 7
^	异或运算	6 ^ 3 = 5
~	反码	~6 = -7

◆ 位运算是直接对二进制进行运算。

◆ 例如：

a=10011101; b=00111001；则有如下结果：

1. a<<3 =11101000; 左移三位补0
2. a>>3 =11110011; 右移三位补1
3. a>>>3=00010011;右移三位补1
4. a&b=00011001; a 与 b
5. a|b=10111101; a 或 b
6. ~a=01100010; a 取反
7. a^b=10100100; a 异或 b

位运算符

位运算符的细节	
<<	空位补0，被移除的高位丢弃，空缺位补0。
>>	被移位的二进制最高位是0，右移后，空缺位补0； 最高位是1，空缺位补1。
>>>	被移位二进制最高位无论是0或者是1，空缺位都用0补。
&	二进制位进行&运算，只有1&1时结果是1，否则是0;
|	二进制位进行 | 运算，只有0 | 0时结果是0，否则是1;
^	任何相同二进制位进行 ^ 运算，结果是0；1^1=0 ， 0^0=0 不相同二进制位 ^ 运算结果是1。1^0=1 ， 0^1=1

条件运算符

- 表达式1？表达式2：表达式3

1. 首先计算表达式1
2. 如果表达式1的值为 true, 则选择表达式2的值
3. 如果表达式1的值为 false,则选择表达式3的值

例：　　int answer = 10 > 5 ? 1 : 2;

- 思考：如何获取两个变量a和b中的大数，赋值给变量c？

取最大数，c = a > b ? a: b
取最小数，c = a > b ? : b: a

运算符的优先级

优先级	运算符	结合性
1	( ) [ ] .	从左到右
2	! ~ ++ --	从右到左
3	* / %	从左到右
4	+ -	从左到右
5	<< >> >>>	从左到右
6	< <= > >= instanceof	从左到右
7	== !=	从左到右
8	&	从左到右
9	^	从左到右
10	|	从左到右
11	&&	从左到右
12	||	从左到右
13	? :	从左到右
14	= += -= *= /= %= &= |= ^= ~= <<= >>= >>>=	从右到左
15	,	从右到左

清華大學 Tsinghua University

Java流程控制

Java 的表达式

- 表达式是由一系列变量、运算符、方法调用构成的，表达式可以计算出一个值来
- 程序中的很多工作是通过计算表达式的值来完成的
  - 有时需要的是表达式的副作用，例如赋值表达式将数值赋给变量
  - 更多时候起作用的是表达式的值，这个值可以用作函数的参数，或更大的表达式的操作数，或者影响语句的执行顺序

Java的流程控制结构

◆流程控制结构主要分成三种

- 判断结构
- 选择结构
- 循环结构

Java的流程控制结构

◆判断结构

if语句的三种格式：

1.
```
if(条件表达式)
{
    执行语句;
}
```

2.
```
if(条件表达式)
{
    执行语句;
}
else
{
    执行语句;
}
```

3.
```
if(条件表达式)
{
    执行语句;
}
else if (条件表达式)
{
    执行语句;
}
……
else
{
    执行语句;
}
```

Java的流程控制结构

◆if语句特点：

a）每一种格式都是单条语句。

b）第二种格式与三元运算符的区别：三元运算符运算完要有值出现。好处是：可以写在其他表达式中。

c）条件表达式无论写成什么样子，只看最终的结构是否是true 或者 false;

Java的流程控制结构

◆选择语句

■ switch语句格式：

```
switch(表达式)
{
  case 取值1:
    执行语句;
    break;
  case 取值2:
    执行语句;
    break;
  ……
  default:
    执行语句;
    break;
}
```

■ switch语句特点：

a) switch语句选择的类型只有四种：byte，short，int ， char。

b) case之间与default没有顺序。先执行第一个case，没有匹配的case执行default。

c) 结束switch语句的两种情况：遇到break，执行到switch语句结束。

d) 如果匹配的case或者default没有对应的break，那么程序会继续向下执行，运行可以执行的语句，直到遇到break或者switch结尾结束。

Java的流程控制结构

◆循环结构

代表语句：while ， do while ， for

■while语句格式：

```
while(条件表达式)
{
   执行语句;
}
```

■ do while语句格式：

```
do
{
    执行语句;
}while(条件表达式);
```

■ for循环格式：

```
for(初始化表达式；循环条件表达式；循环后的操作表达式)
{
        执行语句;
}
```

Java的流程控制结构

◆循环结构

a）do while特点是条件无论是否满足，循环体至少被执行一次。

b）for里面的连个表达式运行的顺序，初始化表达式只读一次，判断循环条件，为真就执行循环体，然后再执行循环后的操作表达式，接着继续判断循环条件，重复找个过程，直到条件不满足为止。

c）while与for可以互换，区别在于for为了循环而定义的变量在for循环结束就是在内存中释放。而while循环使用的变量在循环结束后还可以继续使用。

d）最简单无限循环格式：while(true) ， for(;;),无限循环存在的原因是并不知道循环多少次，而是根据某些条件，来控制循环。

Java的流程控制结构

◆其他流程控制语句

break(跳出)，　continue(继续)

1. break语句：应用范围：选择结构和循环结构。

2. continue语句：应用于循环结构。

注：

a) 这两个语句离开应用范围，存在是没有意义的。

b) 这个两个语句单独存在下面都不可以有语句，因为执行不到。

c) continue语句是结束本次循环继续下次循环。

d) 标号的出现，可以让这两个语句作用于指定的范围。

清華大學 Tsinghua University

Java输出和输入

输出

- 在前面的代码中，我们总是使用System.out.println()来向屏幕输出一些内容。println是print line的缩写，表示输出并换行。因此，如果输出后不想换行，可以用print()。
- 观察以下代码的执行效果：

```
public class Main {
    public static void main(String[] args) {
        System.out.print("A,");
        System.out.print("B,");
        System.out.print("C.");
        System.out.println();
        System.out.println("END");
    }
}
```

格式化输出

- 如果要把数据显示成我们期望的格式，就需要使用格式化输出的功能。格式化输出使用System.out.printf()，通过使用占位符%?，printf()可以把后面的参数格式化成指定格式：

```
public class Main {
    public static void main(String[] args) {
        double d = 3.1415926;
        System.out.printf("%.2f\n", d);
        System.out.printf("%.4f\n", d);
        int n = 12345000;
        System.out.printf("n=%d, hex=%08x", n, n);
    }
}
```

占位符	说明
%d	格式化输出整数
%x	格式化输出十六进制整数
%f	格式化输出浮点数
%e	格式化输出科学计数法表示的浮点数
%s	格式化字符串

注意，由于%表示占位符，因此，连续两个%%表示一个%字符本身。

输入

- 和输出相比，Java的输入就要复杂得多。我们先看一个从控制台读取一个字符串和一个整数的例子：

```
import java.util.Scanner;

public class Main {
    public static void main(String[] args) {
        Scanner scanner = new Scanner(System.in); // 创建Scanner对象
        System.out.print("Input your name: "); // 打印提示
        String name = scanner.nextLine(); // 读取一行输入并获取字符串
        System.out.print("Input your age: "); // 打印提示
        int age = scanner.nextInt(); // 读取一行输入并获取整数
//        System.out.printf("Hi, %s, you are %d\n", name, age); // 格式化输出
        System.out.println("Hi, " + name  + " you are " + age); //输出
    }
}
```

1. 首先，通过import语句导入java.util.Scanner，import是导入某个类的语句，必须放到Java源代码的开头；
2. 然后，创建Scanner对象并传入System.in
3. 有了Scanner对象后，要读取用户输入的字符串，使用scanner.nextLine()；要读取用户输入的整数，使用scanner.nextInt()。Scanner会自动转换数据类型，因此不必手动转换。

清華大學 Tsinghua University

课堂练习

课堂练习(1)

1. 利用for循环：
   - for循环从大到小输出1 - 100
   - for循环从小到到输出100 - 1
   - while循环从大到小输出1 - 100
   - while循环从小到到输出100 – 1
2. 利用while循环输出：
   - 使用 while 循环实现输出 1,2,3,4,5, 7,8,9, 11,12
   - 使用while 循环输出100-50，从大到小，如100，99，98…，到50时再从0循环输出到50，然后结束
   - 使用 while 循环实现输出 1-100 内的所有奇数
   - 使用 while 循环实现输出 1-100 内的所有偶数
   - 使用while循环实现输出2-3+4-5+6…+100 的和

课堂练习(2)

1. 编程计算：1+2+…+n，n为一个从键盘输入的数字。
2. 假设一年期定期利率为3.25%，计算一下需要过多少年，一万元的一年定期存款连本带息能翻番？
3. 一球从100米高度自由落下，每次落地后反跳回原高度的一半；求它在第10次落地时，共经过多少米？第10次反弹多高？
4. 编程实现：从键盘接收一百分制成绩（0~100），要求输出其对应的成绩等级A~E。
   - 其中，90分以上为'A'，80~89分为'B'，70~79分为'C'，60~69分为'D'，60分以下为'E'，若不是0-100之间，则报错并要求再次输入。
5. 输入一年份，判断该年份是否是闰年并输出结果。
   - 注：凡符合下面两个条件之一的年份是闰年。（1） 能被4整除但不能被100整除。（2） 能被400整除。
6. 制作趣味模板程序
   - 需求：根据用户输入的名字、地点、爱好，进行任意显示，如：敬爱可爱的xxx，最喜欢在xxx地方YYY
7. 使用while,完成以下图形的输出

```
*
**
***
****
*****
****
***
**
*
```

课堂练习(3)

1. 猜年龄游戏。
   - 首先预设1个年龄。
   - 用户输入猜测的年龄，如果大了，则提示“大”，如果小了，则提示“小”，如果相同，则提示“猜对了”。
   - 允许用户最多猜测3次，3次均猜不中，则提示：“3次猜错，游戏失败！”
2. 猜4位数字游戏。

   1、输入一个4位数字a（不能重复）

   2、猜一个数b，程序自动答复：mAnB

   m:数字和位次全对的数量。

   n:只有数字对但位不对个数的数量。

   m+n <= 4

   3、4A，代表完全猜出来了。
3. 实现用户输入用户名和密码，并登录验证测试：
   - 当用户名为 seven 且 密码为 123 时,显示登陆成功,否则显示登陆失败!
   - 失败时，允许重复输入三次。
   - 失败3次后，则提醒出错，程序运行退出。