# java命令

## 使用Java命令运行程序

第一种 无第三方java包，使用命令运行程序

准备工作：两个类：1）User类：包路径：com.wwl.bean,2）主程序：A类 com.wwl包中

User类

Public class User{

Private String name;

Private Integer age;

Setget方法省略。。。

重写toString方法省略。。。

A类

package com.wwl;

import com.wwl.bean.User;

public class A {

public static void main(String[] args) {

User user = new User();

user.setAge(33);

System.out.println(user.toString());

}

}第一步 进行编译成class（字节码）文件 命令：javac 类路径(相对路径或绝对路径都可以)

1. 先进行编译User类，因为此java没有依赖任何的类

命令：javac com/wwl/bean/User.java

1. 编译A类

命令：javac com/wwl/A.java

第二步 运行此工程

只要运行主入口即可：移动到程序目录的根：如我的是com,那么就要在com目录的上一层进行操作命令

命令：java com.wwl.A 结果：User [name=null, age=33]

第二种 依赖第三方jar包，使用命令运行程序

准备工作：一个jar包一个主程序类：1）B.jar中就一个User.class：包路径：com.wwl.bean,2）主程序：A类 com.wwl包中

A类和第一种的一样，B.jar包中的User也是第一种方式的User类

第一步 进行编译成class（字节码）文件 命令：javac –classpath .; 类路径(相对路径或绝对路径都可以)

把jar包先加载到classpath环境中，然后带上需要编译的A类

命令：javac -classpath 。：lib/B.jar A/A.java

此命令意思是：先加载lib包中的B.jar到classpath中，再编译A包中的A.java

注意：1、windows系统下用：分号；Linux用冒号：

2、解析编译时候必须要带上：.:，不然编译好的字节码在第二步运行会出现找不到类

第二步 运行工程

命令：java -classpath .:lib/B.jar A.A

此命令意思是：先加载jar至classpath环境中：再运行 A包中的主程序A.class

# Java类型

## 八大基本数据类型

数据类型转换要注意的细节

1. 凡是byte、short 、 char数据类型数据在运算的时候都会自动转换成int类型的数据再运算。

2. 两个不同数据类型的数据在运算的时候，结果取决于大的数据类型。

3. sun给我们提供一个功能 Integer.tobinaryString() 查看一个数据的二进制数据形式的。

System.out.println(Integer.toBinaryString(-7)); // 11111001（补码）

第一种情况 类型自动转换，下面情况是正确的

byte b= 11; // 1个字节 一两碗

short s = b; // 把b变量存储的值赋予给s变量。 2字节 二两的碗

int i = s; // i是4字节

long l = i; // l 是8个字节。

System.out.println(l);

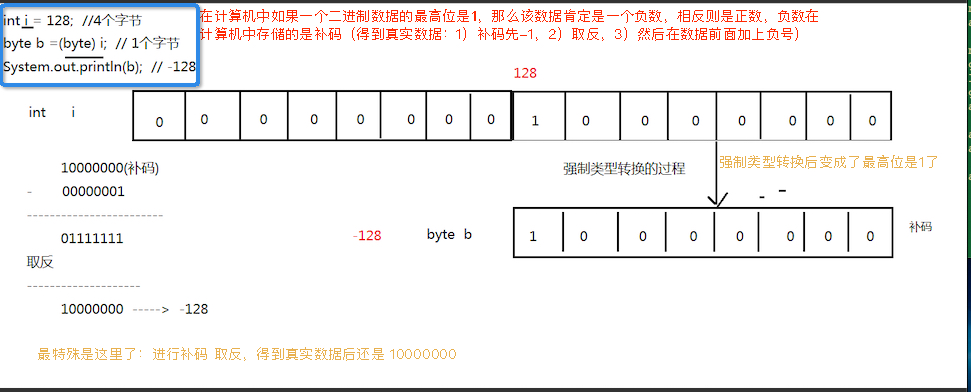
第二种情况 关于进行强制类型转换后数据不一致问题

int i = 128; //4个字节（32位）

byte b =(byte) i; // 1个字节 (8位) 进行强转是把24位进行去除

System.out.println(b); // 结果却变成了负128 -128

为什么结果是-128呢？原理解析如下：



第三种情况 java的整数类型默认是int

凡是byte、short 、 char数据类型数据在运算的时候都会自动转换成int类型的数据再运算。

byte b=1;

byte b1=2;

byte b3=1+3;//这样是没问题的，编译器会先检查此类型赋值有没有超出范围，再解析为字节码class

byte b4=(byte) (b+b1);//需要强制类型转换，虚拟机编译检查不了有没有超出范围，它是一个变量相加

第四种情况 两个不同数据类型的数据在运算的时候，结果取决于大的数据类型

int i =10;

long l = 20; //不加L也可以，不知道啥情况

i = (int)(i+l);

System.out.println(i);

最后住意

int i = 10; //

byte b = i; // 一个整数没有加上任何标识的时候，默认是int类型的数据。

//10 是一个常量， 编译器在编译的时候就能够确认常量的值了，byte b = 10,在编译到的时候

java编译器就会检查到10并没有超出byte的表示范围，所以允许赋值。java编译器在编译 的时候并不能确认变量所存储的值，变量存储的值是在运行的时候才在内存中分配空间的。

System.out.println(b);

# 运算符

逻辑运算符

^ (异或)

规律： 只要两边的布尔表达式 结果不一致，那么结果就为true，如果左右两边 的布尔表达式一致，那么就为false.

System.out.println(true^true); // false 两边相同 所以false

System.out.println(true^false); // true

System.out.println(false^true); // true

System.out.println(false^false); / / false 两边相同 所以false

&& (短路 与（双与）)

短路与和单与符号的相同与不同点：

相同点： 短路与和单与运算 的结果是一样的。

不同点： 使用短路与的时候，如果左边的布尔表达式为false，则不会在运算右边的布尔表达式，从而提高了

效率：使用单与的时候，即使发现左边的布尔表达式为false，还是会运算右边的布尔表达式的。

只有左边的布尔表达式为false时，双与的效率才要高于单与的.

|| (短路 或（双或）)

短路或与单或的相同点与不同点：

相同点：运算的结果是一致 的。

不同点：使用短路或的时候，当发现左边的布尔表达式为true时，则不会运算右边的布尔表达式。

使用单或的时候 发现左边的布尔表达式为true，还是会运算右边布尔表达式。

！ (非)

# Java转义字符

转义字符：特殊字符使用”\”把其转化成字符的本身输出，那么使用”\”的字符称作为转移字符。

常见的转义字符有：

\b Backspace （退格键）

\t Tab 制表符(制表符的作用就是为了让一列对齐) 一个tab一般等于四个空格（区别就是对不齐）。:老师拿九九乘法表列子进行对齐

\n 换行

\r 回车 把光标移动到一行的首位置上。

System.out.println("hello, \rworld");//在windows演示是光标不换行回车到本行的第一位置然后覆盖其内容：打印：world 没有hello

注意： 如果是在windows系统上操作文件的时候需要换行，是需要\r\n一起使用的。

如果是在其他的操作系统上需要换行，仅需要\n即可。