1.1〉准备：：：编java时，如果没有NetBeans IDE时，我们只能通过设置环境变量（前提是jdk-7u51-windows-x64存在于电脑，我们安装了虚拟机,他不会在桌面上显示，但是它是存在的（自己想的））

方法——〉开始——〉系统——〉高级系统设置——〉环境变量——〉path——〉编辑(后面+；+（安装jdk-7u51-windows-x64之后bin文件夹所在位置）例：我存在D盘中，那就找他的bin吧D:\Program Files\Java\jdk1.7.0\_51\bin)——〉 ；D:\Program Files\Java\jdk1.7.0\_51\bin

或者 ；%JAVA\_HOME%\bin

2〉编辑：：：新建记事本文档(一般是)或者写字板之类都行编译代码——〉将文件名字改为起始“类”（的名字）.java(扩展名) 例如：TestJava.java

windos8中不可以直接改扩展名

——〉我的电脑——〉查看——〉显示文件扩展名——〉此时我们就可以改变扩展名了。（最后我们打开看的时候，采用记事本形式大可。

Win7|xp

——〉察看文件扩展名——〉打开我的电脑——〉工具——〉文件夹选项——〉隐藏文件的扩展名

3〉编程时：类的名字一般大写，例如上面2〉TestJava，都是大写，函数都是小写，例如void main 都是小写的，但是大写也行，只不过太另类。

4〉运行时先是主函数，再是其他函数，但是如果直接将其他函数的写在后面不会运行。此时我们需要将后面的函数——〉调用

——〉类，函数都要调上去

public class TestJava

{

public static void main(String[] args)

{

System.out.printf("Hello,World!\n %d",num);

A aa = new A(); //此时我自己认为是将A作为一个调用函数名 aa为变量 将新的对象A（）给了aa变量 通过aa来寻找函数

aa.b();

}

}

class A

{

public void b()

{

//int num = 2;

System.out.println("hello,world "+2014+num);

System.out.println(""+2014);

System.out.println(2014);

}

}

5〉" "外面如果想加东西直接“+”就可以了，不需要" "

例如："hello,world "+2014

但是不可以加字符，如果想加那就得上’’(单引号)了

6〉" "中的都是字符串，如果想在外面加数字，那么数字也是字符串

例如：println("5"+2014)

输出 52014

除非没有" " 例如：println(2014) 那么它就是你数字了。

7〉使用dos进行编译——〉找到编程的那个文件——〉属性——〉看他是在那个文件之下——〉记住——〉dos界面要先进入它的起始基本文件——〉

输入命令 ：“cd Desktop(记住的哪个文件名)——〉javac TestJava.java(如果程序正确，则什么也看不到，它只是默默的编译，如果错误，他会提示我们)

此时如果想执行看到

——〉java TestJava

8> printf 输出时不会自动换行所以我们需要像c语言一样输入\n才可以换行

println则是自动换行

9〉执行完之后会在编程文件后面形成文件（类的文件，打不开，没有用，删除即可）

10>在编译java时我们中间使用了\n那么后面加上’+’的在最后的输出的后面

？？？dos

1〉cd转移到编程文件的上一个目录

有时dos执行时不再c盘所以我们要转到其他的盘 d：（转移到d盘了）

再转移到需要的目录即可

2〉编译：

javac可以将java源码编译成字符串（之前我们设置了path路径中有bin下的javac.exe,所以dos中可以识别出来）

javac （souse files（源文件））就可以执行了

3〉文件上多了一个文件，这就是自解码文件（打开也不认识，这时给虚拟机看的，我们看不懂）

4〉执行：

使用虚拟机执行（此时，如果我们改变源文件的东西再次执行也会出现之前的结果，那么我们必须再次编译一次javac （souse files））

java 源文件名

2.大小写必须区分，例如Ｓｙｓｔｅｍ，Ｓｔｒｉｎｇ［］．

1>void main 是程序入口

3.一个源文件可以对应多个，很多个class

1〉如上文的class A 它就是另外的一个

2〉在进行编译之后，会出现一个类class对应一个自解码文件

3〉哪个类class前面放了public，那么这个类的类名必须与源文件名保持一致

public class只能有一个，而且源文件名必须与它的class类保持一致

public class Welcome

{

public static void main(String[] args)

{

System.out.println("我叫张宇晋"+'？');

}

如右面的编码

编译之后会出现：

Welcome.class

A.class

Man.class

(3个自解码文件)

}

class A{

}

class Man{

}

5.{ }一定要合理的放置

1〉 {

}

2〉 {

}

6.注释

有时为了让别人看懂或者为了添加注释//

1〉例如： //args是argument的缩写

那么虚拟机不会执行它的 ，如果在此出现错误，在格式中改变

为本地编码——〉 转为ANSI编码格式

2〉/\* 我叫张宇晋 \*/

7.虚拟机命令

1〉 tab命令

命令补齐，在打了几个字时，tab命令看看。有收获的哦

2> cd e:

cd Desktop/新建文件夹/……

8.标识符Identifier

1〉必须是字母、下划线、美元符￥、 开头哦、

2〉其他位置字母、下划线，美元符￥、数字

3〉不可以是关键词，但是可以使用汉字，因为Java使用了Unicode字符集

universial 通用的缩写——〉国际通用字符集

9.整数类型

1〉 八进制 以0开头的数

例如 int a=0开头 10

输出 8

十六进制 0x开头

例如 int a=0x10 b=0xf

输出 16 15

1 2 3 4 5 6 7 8 9 a b c d e f 10

2>转换进制

System.out.println(Integer.toBinaryString(a));

Octal

Hex

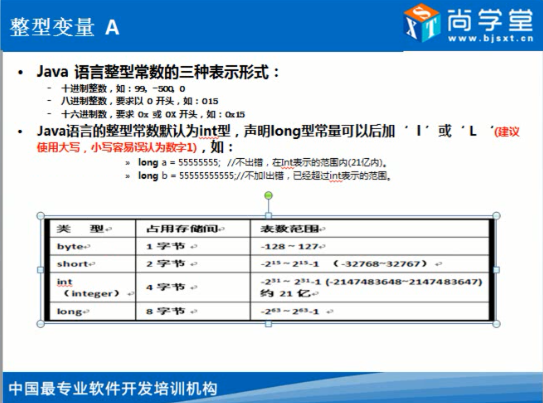
Binary是二进制即将a转换为二进制数，八进制，十六进制

3〉申明long类型的常量（观察int（21亿大））

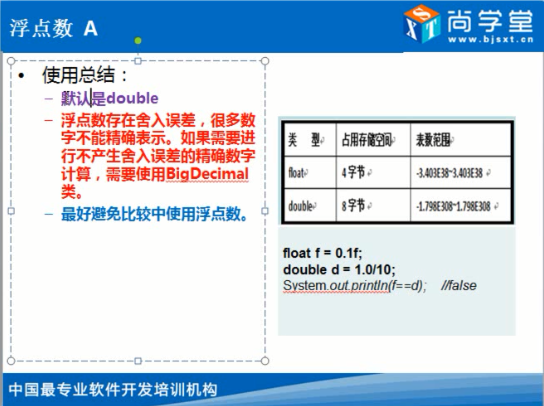
long a=123456789456(超过了21亿)

虚拟机运行时会出现数字过大的错误 所以我们要在123456789456L（加一个L）

4〉有时数字太大，BigInteger



10.浮点型



2〉例如

double a=3.14 正确

float a=6.28（超出了4个字节）所以会出现精度缺失的错误

float a=6.28f/F（此时精度已经损失了，但是表示的是单精度）

3〉浮点数有误差，所以我们不要去比较浮点数

例如： double d=1.0/10

float a=0.1f

System.out.println(a==d);

输出错误false

4>有时如果一定要精确

使用 BigDecimal

11.字符

1〉输出“’ /” “ /’ // ”

2> boolean ture/false

例 boolean b=ture（比如！0则是ture）

例

2.

boolean w=true&false;

System.out.println(w); //输出 false

2.3

boolean d=1>2&&3<(3/0);

System.out.println(d);

输出 flase

3> 强制转型

比如char（字符可以直接当作整数来用，但是整数要通过强制转型才可以用作字符型，因为整数是四个字节，char是2个字节）

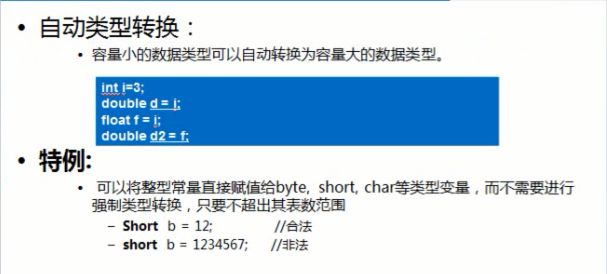
char c3=’a’

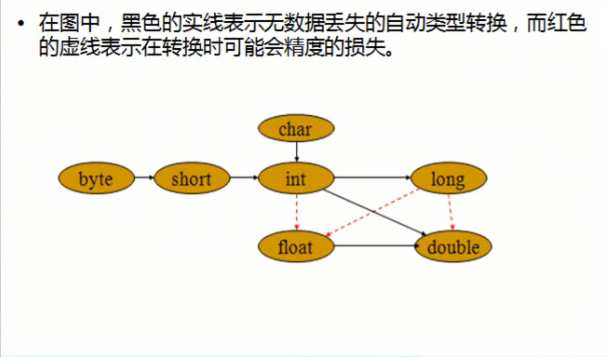
int c2=’a’+1

char c1=c2 (出错) //应该写为char c1=(char)c2

12.自动转型和强制转型

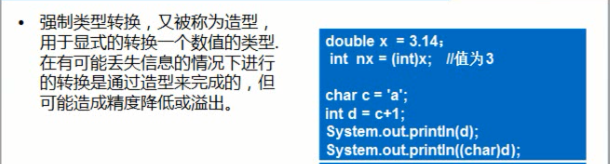
1〉自动类型转换（容量**小的可以自动转换为大的）**



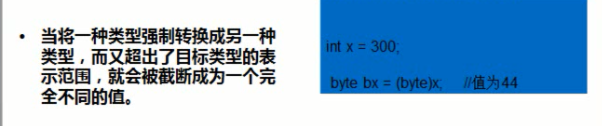


2〉强制转型

1.精度缺失



2.随便打出



3〉转型中的类型提升问题

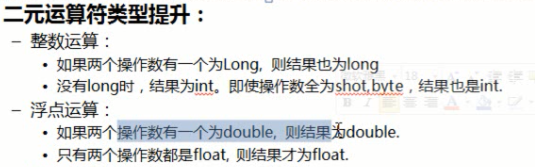
1.例1

int a=1

long b=2

int c=a+b 出错

（所有的二元运算符+-\*/类型会有提升，此时的提升之后位long型） 得强制转型



int c=(int)(a+b)

2.同级进行运算

1.

int money=1000000000

int today=10000000

int a=today\*money

输出是负数（a的值已经超出了21亿）

修正1〉此时我们想将他变为long型不就可以了吗

long a=today\*money

(也是负数，因为我们将二者进行乘法已经是负数，负数再复制给a)

修正2〉此时我们进行强制转型吧

long a=(long)today\*money

千万不可以写为long a=(long)（today\*money）

否则出现的还是负数，原因自己想吧

2.long a=10\*45\*7989\*4615\*1515（）

输出a是负数，又是超出int了记得前面的l/L

此时我们long a=10\*45\*7989\*4615\*1515L

或者 long a=10L\*45\*7989\*4615\*1515

尽量将L加在前面的数，防止在前面的运算中就已经超出了

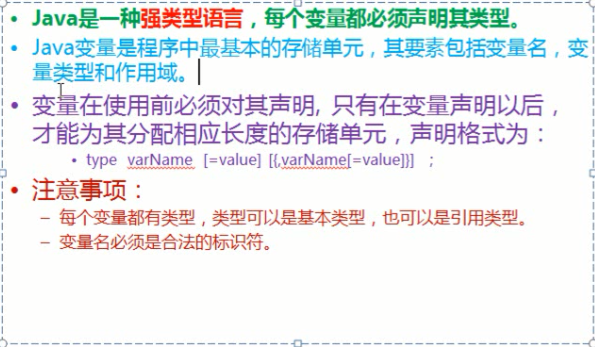
13.二进制与分隔符

1〉 int a=0b0000\_0000\_0000\_0000\_0000\_0000\_0000\_0011

此时表示的便是二进制数了

2〉分隔符对于数字没有影响只是为了让我们好看

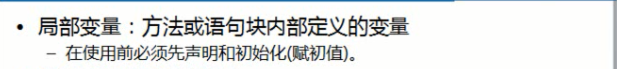
14.变量 ，成员变量 ，局部变量



2〉实例变量



3〉局部变量



15.常量

1〉final常量 （最终）在一个变量前加个final那么这个变量以后就不可以更改了，就是只能被初始化一次

final int a=1；

a=2； (错误的)

2〉常量一般大写，大写字母加下划线（小写的一般为变量）

例如：final int MAX\_VALUE=1235(加分隔符好区分)

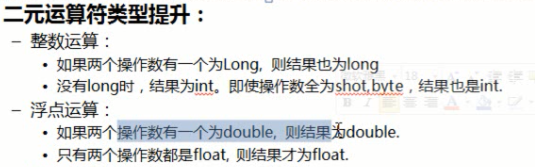
16.大小写

变量 首字母小写，驼峰原则 ageNew

类名 首字母大写。驼峰原则 Man GoodMan

[17.运算符](mycode/13集_运算符_01_算术_逻辑_位运算符_扩展运算符)

1〉算数运算符 + - / % ++ --（一元运算符）



2〉例

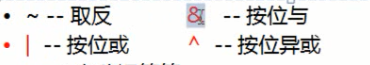
boolean d=1>2&&3<(3/0);

System.out.println(d);

输出 flase

（先执行前面的，前面1>2为正确才会3<(3/0)不可以做除数，此时只是为了检测他的运算） boolean 的值就是只有ture 和false （!0则是ture，0则是false）

2.了解即可 ，位运算符（对于二进制来说的，运算中是以二进制来进行的，但是结果还是10进制的数）



1〉取反 1元运算符

按位,或,异或 是2元运算符（ture和false也可以参与）

（按位异或 一样的为0，不一样的为1）

8和4：转化为2进制分别为

1 0 0 0 1.取反 0 1 1 1

0 1 0 0 1 0 1 1

2. & 0 0 0 0

3. | 1 1 0 0

4. 1 1 0 0

2> 例1

int e=8;

int f=4;

System.out.println(e&f); 输出 0

System.out.println(e|f); 输出 12

System.out.println(e^f); 输出 12

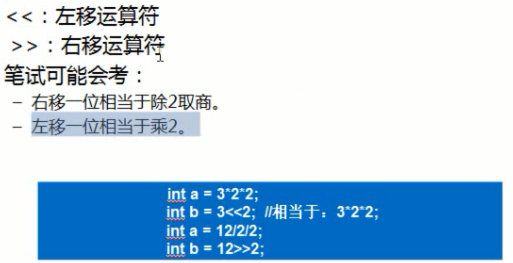
例2

boolean w=true&false;

System.out.println(w); //输出 false

进行为运算符时都是以二进制运算的但是结果不是二进制的，8 1000 4 0100

2.



10<<2(这个数表示的是移动的位数，左移2位)

10〉〉3（右移3位）

3.扩展运算符 += , -= , /= , %=



18.eclipse开发java（下面讲的都是eclipse开发，否则将会有声明）

1〉新建——〉 java项目——〉填写包名（project name）——〉选择Java-SE的版本——〉点击下一步即可

2〉——〉（放置源码）

类库

3〉点击 src——〉右键——〉新建——〉类——〉name（填写你想要的类的名字）

4> 点击scr——〉包的名字——〉上面类的名字（点击这里可以编程）

5〉编程之后——〉右键——〉运行方式——〉Java应用程序

（此时控制台上将会有运行的结果）

6〉打开窗口——〉显示视图——〉大纲（这里我们可以看到很多有用的东西）

补充

7〉运行之后也会像jdk中的一样，出class文件（自解码文件，提供给计算机看的文件）他是位于——〉工作空间——〉bin文件夹下（源文件位于src文件夹下），

以前我们是手动编译，现在呢是eclipse帮助我们编译

8> 调试（双击我们想要进行调试的位置，可以多选几行）

——〉右键——〉运行调试——〉（点击——> 进入下一调试点，点击——）停止调试直接运行完，点击——〉会一步步来）

9〉

要想在打开的视图中来回切换——〉点击（右上角我们已经打开的视图即可）

19.输出一个随机数

double d=Math.random() //这个数是在[0，1）之间的随机数

例：[置筛子：](mycode/17集if语句_单选择_双选择_多选择结构/TestIf.java)（输出一个在[1，6]之间的数）

**public** **class** TestIf {

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

{

**double** d=Math.*random*(); //随机数[0，1)之间的数；

**int** a=1+(**int**)(d\*6); //d的取值范围在[0，1)之间，所以 d\*6的值为[0，5)

**if**(a>3)

{

System.*out*.println("庄家赢");

}

**else**

{

System.*out*.println("庄客赢");

}

}

}

20。[定义函数，调用函数（不需要像c语言中一样声明）](mycode/24集方法_方法的本质_形参_实参_return语句/TestMethod.java)

1〉只能从主函数main调用函数值

例1 void 无返回值

public class TestMethod{

public static void main(String[] args){

System.out.println("我是谁");

test01(); //进行调用

}

public static void test01() //不调用时不会进行编译的

{ //void不会返回值

System.out.println("我叫张宇晋");

}

}

例2.有返回值，形参，实参等；

public class TestMethod{

public static void main(String[] args){

System.out.println("我是谁");

int a=max(3,4); //进行调用

System.out.println(a);

}

public static int max(int x,int y) //定义一个函数

{

int z;

z=x>y?x:y;

return (z);

}

}

2〉return 的两个作用，返回值

例如1 return (z);

例如2 return 0；结束方法的运行

2[1.调用（递归，调用自己）](mycode/25集递归算法详解/TestDigui.java)

1〉

public class TestDigui{

public static void test01(){

System.out.println("我在这里");

test01();

}

public static void main(String[] args){

test01();

}

}

//输出的结果为 "我在这里” 连续执行，没有结束

2〉有时我们不需要他一直执行下去（内存有限）

public class TestDigui{

static int a=0; //方法外部的变量可以供下面方法的共同使用

public static void test01(){

a++;

System.out.println("我在这里"+a);

if(a<=10)

{

test01();

}

}

public static void main(String[] args){

test01();

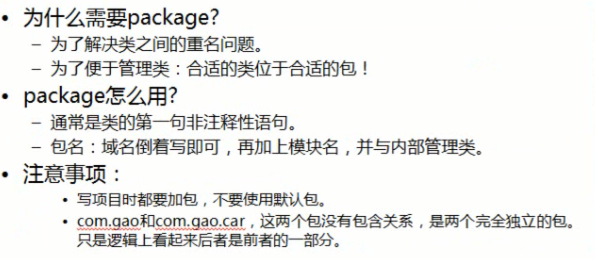
}

} //一共运行11次 10+1

3〉static int a=0; 定义静态的常量

22.API文档

1〉包（管理类，合适的类位于合适的包）



2〉点击scr——〉右键-〉新建——〉包

将写好的直接拉入——〉点击确定即可

3〉包名——〉域名倒着写

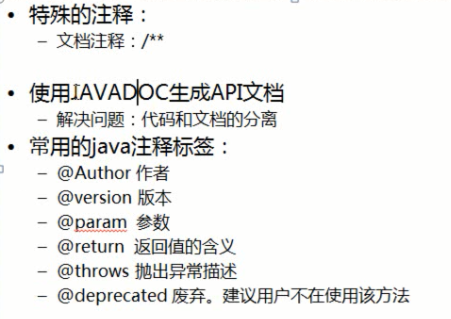
com.taobao.test(项目名)

4〉学会使用（观看类的作用）

API-〉概述——〉选择一些常用的包——〉java.long(包含很多的类)

5〉拥有自己的API文档

使用JAVADOC



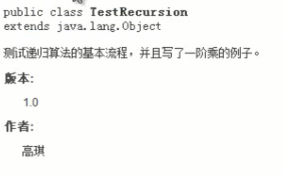
1.解释我们的:输入下面的开始描述



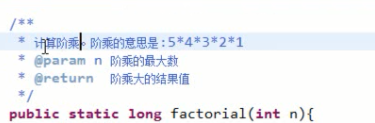
2〉点击包——〉右键——〉导出——〉java——〉java.doc——〉找到

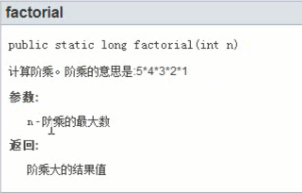
（D:\Program Files\Java\jdk1.7.0\_51\bin\javadoc.exe）——〉选择存储位置——》完成即可





输入我们对于类的描述





23.对于不是long(其他包)包中的类的输入（要进行import调用，因为long包中没有这个类）

1〉下面以以scanner为例：

2〉

**public** **class** TestScanter {

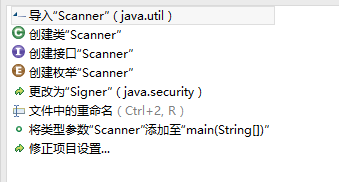
**public** **static** **void** main(String[] args){

Scanner s=**new** Scanner()

}

}

3> 输入Scanner时左面会有——〉点击它出现——〉



4〉点击导入

5〉会在java程序的开头出现

如下面的程序

**import** java.util.Scanner;

**public** **class** TestScanter {

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

Scanner s=**new** Scanner()

}

}

24.使用键盘输入东西

**Scanner s = new Scanner(System.in);**

//String.in是输入的意思，与System.out相对应

例1：输入字符串

**Scanner s = new Scanner(System.in);**

**String str=s.next();**

//next:程序运行到next会阻塞，等待键盘的输入

例2：输入数字

**Scanner s = new Scanner(System.in);**

**int a=s.nextInt();**  同上等待键盘的输入：

例3：输入两个数并且计算二者的和：

**import java.util.Scanner;**

**public class TestScanter {**

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

***test01*();**

**}**

**public static void test01(){**

**Scanner s = new Scanner(System.*in*);**

**System.*out*.println("请输入一个数字：");**

**int a=s.nextInt(); //接收键盘的值**

**System.*out*.println("请输入又一个数字：");**

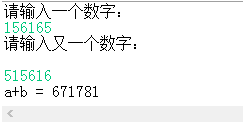
**int b=s.nextInt();**

**int sum=a+b;**

**System.*out*.println("a+b = "+sum);**

**}**

**}**



25.面向对象：（这里测试非静态的如调用）

**public** **class** TestOverload {

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

{

Mymath aa = **new** Mymath();

**int** h=aa.add(4,5)；

System.*out*.println(h+" "+m);

}

}

**class** Mymath{

**public** **int** add(**int** a,**int** b){

**int** z=a+b;

**return** z;

}

}

调用有参数无返回值得方法

调用有参数，且有返回值的的方法

**public** **class** TestMianxiang {

//静态的数据 供下面的方法使用

String name; //一般由主函数设置值

**int** id; //学号

**int** age;

String gener;

**int** weight;

//动态的行为

**public** **void** study()

{

System.*out*.println(name+"留学生"); //使用name

}

**public** **void** shouzhang(String sname) //使用 形参

{

System.*out*.println(name+"向"+sname+" 说首长好");

}

**public** **static** **void** main(String[] args){ //必须有这句，否则不会执行

TestMianxiang aa=**new** TestMianxiang(); //调用 class

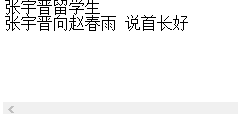
aa.name="张宇晋"; //给字符串赋值

aa.study(); //调用study类

aa.shouzhang("赵春雨"); //调用shouzhanglei ，输入形参的值

}

输出结果

} 

26.