Java知识

**Day3**

1. **标识符的定义**
2. 类和接口：每个单词首字母要大写
3. 方法：首字母小写，其后用大写字母分割每个单词
4. 常量：全部大写，单词之间用下划线分割
5. 变量: 通常使用名词，首字母小写，其后大写字母分隔每个单词
6. **三元运算符**

变量=条件 判断 ？ 表达式1： 表达式2

（解析）

如果符合条件判断则执行表达式1 ，否则执行表达式2

**三、算术运算符（++，--）**

++a 先进行自加在运算

1. - 先赋值运算在自加

**四、位运算符（&，| ，^,~）**

（1） << 左移 >> 右移 >>> 无符号位右移

（2） 位运算符首先要把数据转换成为二进制补码

（3）当两边是数据做位运算（3 & 4），当两边是布尔值做的逻辑运算（ true & flase）

3的二进制 0011

4的二进制 0100

1. &位与运算：有0则0

结果：0000 结果十进制为0

1. |位或运算：有1则1

结果：0111 十进制为7

（6)^ 位异或运算 ：相同为0，不同为1

结果：0111 十进制为7

1. ~ 按位取反 ：0变1，1变0

1. 一个数据对另一个数据异或两次，该数本身不变

1. **符号位移动**

【1】<< 左移 ：左边最高为丢弃，右边补齐0 (把<<左边的数据乘以2的移动次幂)

3<<2 相当于 3\*2\*2

00000011

(00)00001100 得十进制为12

【2】>> 右移 ：最高位是0 ，左边补齐0，最高为1，左边补齐1

（把>>左边的数据除以2的移动次幂）仅限正数

【3】 >>> 无符号位右移 ： 无论最高是0还是1，左边均补齐0

Day4

1. **Switch选择语句**

【1】、switch语句后的表达式的取值是有限制的：byte,short,int char

Jdk7 版本后可以每句以及String 类型

【2】break :中断结束的标志，可以控制switch语句的结束

【3】当所有case后的值都与表达式值不匹配时，执行default语句

【4】Switch语句先走case，遇见break结束，如果没有break那么执行到末尾在结束

【5】case 后可以接收short ,byte, int,char,string 类型

二、**if语句与switch语句运用场景**

【1】if语句：针对结果是Boolean类型的判断

针对一个范围的判断

针对几个常量值的判断

【2】针对几个常量值的判断

1. **循环语句**

【1】for循环 ：for(初始话语句；判断语句；控制条件语句){

循环体语句；

}

1. 执行初始化语句
2. 进行判断语句：满足则继续，不满足则退出
3. 满足后：执行循环体语句，在执行控制条件语句
4. 回到步骤2 继续执行

【2】for 循环可以嵌套使用

1. **while语句**

【1】while循环：初始化语句

While（判断条件）{

循环体语句

控制条件

}

1. **do……while语句**

【1】do……while循环： 初始化语句

Do{

循环体语句；

控制条件语句；

}while（判断语句条件）；

1. **for循环的嵌套**

【1】for循环第一个循环是行的变换

第二个循环是列的变化

【2】只有当第二个循环的所有满足判断条件的执行完以后，才可以接着进行第一个for循环满足条件的其他语句的执行；（当内循环的语句执行完一次后，跳出内循环，用 System.*out*.println();实现换行）

1. **break语句的使用**

【1】、用于放在switch语句中实现=程序的结束

【2】用于放在for循环的If语句判断中进行中断程序

【3】嵌套在for循环的内层结构中，实现内层循环的结束，腰想实现外层语句的中断，结构如下： 标签名：语句

wc: **for**(**int** i=1;i<=5;i++) {

nc: **for**(**int** j=1;j<=4;j++) {

**if**(j==3) {

**break** wc;

}

System.*out*.print("\*");

}

System.*out*.println();

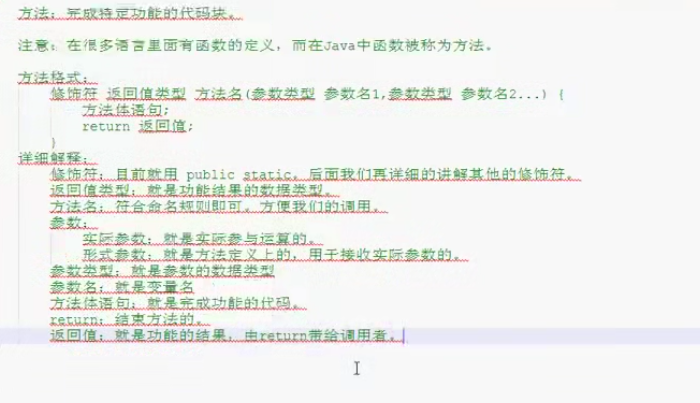
}

1. break：结束单层循环

Continue：结束本次循环

Return : 退出整个main方法

Day5



**Day9 抽象——接口——多态**

**一、this:**  封装 ，对数据进行初始化，，访问本类的内容

**super：**访问父类的构造函数，访问父类的东西

当变量同名时， 输出局部变量，输出成员变量用this

\*打印父类成员变量用super

\*this：代表本类中的使用

\*super：代表父类存储空间的标识（可以理解为父类引用，操作父类成员）

\*

\*A：调用成员变量

\* this.成员变量 调用本类

\* super.成员变量 调用父类

\* B:调用构造方法：

\* this(...) 调用本类的构造方法

\* super(...) 调用父类的构造方法

\* C:调用成员方法：

\* this.方法名

\* super.方法名

**二、final** 标记的类不能被继承

final标注的变量是常量。值不可以被改动

final 修饰的方法不能被重写

final修饰的对象，只能改变对象的内容，不能改变对象的地址值（即就是对象的名字）

**三、继承**

java继承的特点:

\* a:java中只允许单继承尴尬，不允许多继承;

\* b:java支持多层继承(也就说一个类即可以作为父类，也可以继承其他类)

\* c:子类只能继承父类所有非私有的成员变量及方法

\* d:子类不能继承父类的构造方法，但可以通过super关键字来调用父类的构造方法

\* e:不要为了部分功能去继承；

\*

\* 成员变量：子类和父类中的成员变量名一样，采用就近原则；

\* （先在子类的局部中找，在子类的成员中找，在父类中找，否则报错）

**四、多态**：同一个对象在不同时期不同的状态

（进行方法的重写）

向上转型: 父 对象 = new 子类（）；

成员变量：编译看左边，结果看左边

构造方法：只跟类有关，默认父类的构造

成员方法：编译看左边，结果看右边

向下转型： 子类 对象名 = （子类） 父类对象；

**五、接口：**

interface 标记

用implements 实现

接口中的变量只能是常量，就算没有表明，系统也默认是常量

接口中的方法只能被定义成抽象方法，无具体的语句块，无其他的成员方法

接口中无构造方法

**六、类与类：**继承关系

**类和接口**：实现关系，类实现接口中的其他辅助功能，类中是具体存在的共性

**接口和接口：**继承关系

**七、抽象类**：abstract关键字

成员变量：既可以是常量也可以是变量

构造方法：可以无参也可以有参

成员方法：可以抽象也可以非抽象，，，抽象方法中没有实体

**八、抽象类不能实例化：**

抽象类可以没有抽象方法，但是抽象方法的类一定是抽象类

抽象类的子类：

是一个具体类：必须重写抽象类的抽象方法

是一个抽象类：

**static**： 可以标识一个共有的特征变量

可以对象引用，也可以类名进行引用

**Day10 形式参数——修饰符——权限——导包——内部类**

**一、形式参数**

基本类型：

引用类型：

类：（匿名对象时）方法中定义引用类型的参数，必须定该类的对象（用该类对象调用方法）

抽象类：必须定义该抽象类的子类对象（必须是具体类，并且向上转型）

接口：定义实现接口的类的对象, 用实现类去调用方法

**二、返回类型**

基本类型：

引用类型：

类：返回该类的对象

抽象类：返回抽象类的子类对象

接口：返回接口实现类的对象

链式编程：每次调用完后返回的是一个对象，才能继续调用其他的方法

LeiDemo le = new LeiDemo();

le.method().study();

1. **package ：**

**1、手动式**

A：其实就是文件夹

B：作用

a:把相同的类名放在不同的包内

b:把类进行分类管理

C: 定义

package 包名

多级包用 . 分开即可

D: 注意事项

a ：package语句必须在程序的第一条可执行的代码

b ：package语句在第一个Java文件中只能有一个

c ： 如果没有package，默认表示无包名

E ：带包的编译和运行

手动式：

a: 编译一个带包的Java文件夹

b: 通过Javac命令编译该Java文件

c: 手动创建包名

d:把b步骤的class文件放到c步骤的最底层包中

e:回到和包根目录在同一目录的地方，然后运行 （带包运行）

2、 自动式：

a:编写一个带包的Java文件

b:javac编译的时候带上 -d 即可

Javac - d . helloword.java

c: 带包运行

Java cn.itcast.helloword

3、不同包类的访问：

cn.itcast.Demo d = new cn.itcast.Demo ( );

导包：import 包名

导入是类的名称的级别：import cn.itcast.Demo ;

4、package import class 有没有顺序关系：

有

package >import >class

package: 只能有一个

import：可以有多个

class：可以有多个，以后建议是一个

**五、权限修饰符：**

本类 同一个包类（子类和父类） 不同包下(子类) 不同包下（父类）public : ok ok ok ok

private: ok no

protected: ok ok

默认： ok ok

**六、修饰符**：

权限修饰符：默认，private，protected, public

状态修饰符：static , final

抽象修饰符：abstract

1、类：

权限修饰符：默认修饰符，public

状态修饰符：final

抽象修饰符：abstract

（用的最多的是 public）

成员变量：final表示常量，static 表示 类名调用 四种权限修饰符都可以

（用的最多的是private）

构造方法：只能使用四种常见修饰符 ( 用的最多是public)

成员方法：四种权限修饰符，，static (类名调用) ，final( 不能被重写)，abstract

（用的最多是public）

**七、内部类**：把类定义在其他类的内部，这个类称为是内部类

访问特点：

A：内部类可以直接访问外部类的成员，包括私有的

B: 外部类要访问内部类的成员，必须创建对象

C:对象创建格式

外部类类名. 内部类类名 对象名 = new 外部类类名（）. new 内部类类名（）；

D：内部类加了private修饰，只能内部类访问，外部类不能够通过C方式进行创建对象名来访问内部类 ;

成员内部类的修饰符：

private： 通过父类写一个同等的方法。创建内部类的对象，来访问内部类的方法，，，，

在通过外部类创建对象，通过对象来访问同级方法。

static : 内部类用static修饰可以看成是内部类是外部类的成员

注意： 被static修饰的内部类只能访问外部类的静态数据

成员内部类被静态修饰的访问方式是：

外部类名 . 内部类名 对象名 = new 外部类名. 内部类( ）；

**注意 ：**

1、 内部类和外部类没有继承关系

2、通过外部类名限定this对象

外部类名. this

**八、 局部内部类:**

在外部类的方法里定义了一个类

1、可以直接访问外部类的成员

2、在局部位置，可以创建内部类对象，通过对象调用内部类的方法，来使用内部类的特定功能

**面试题**：

局部内部类访问局部变量的注意事项？

-----局部内部类访问局部变量，必须用final修饰

为什么呢？

--------局部变量随着方法事物调用而调用，随着调用结束而消失，而堆内存（内部类）的内容并不会立即消失

加final修饰后就变成了 常量，，（把值留下了，变量名消失了）

**九、匿名内部类：**

前提：存在一个类或者接口

这里的类可以是具体类也可以是抽象类

格式：

new 类名或者接口名（）{

重写方法 ;

}

本质是什么？

是一个继承该类或者实现了该接口的子类的匿名对象

调用方法：多态

**十、接口的特点：**

1、用关键字interface表示：

interface 接口名 { }

2、类实现接口用implements 表示：

class 类名 implements 接口名 { }

3、接口是抽象的，无法实例化；按照多态的方式进行实例化

4、具体类多态，抽象类多态，接口多态

5、接口的子类:

要重写抽象类的抽象方法，或者定义为抽象类（但是此方法意义不大）

**接口的成员特点：**

A：成员变量：接口中的变量默认为常量（值不可以改）：

默认为静态的（不写也默认）可以用接口调用 接口名.变量名

默认修饰符： public static final 变量名

B：构造方法：无构造方法 （子类中的super( ) 其实调用的是Object类）

C：成员方法：接口的方法不能有主体，默认为抽象类（只能是抽象方法 public abstract void show ( ) ;）

所有的类都默认继承一个类 Object

类与类的关系：

继承关系：只能单继承，可以多层继承；

**类与接口：**

实现关系：一个类可以实现多个接口（ 多个接口之间用逗号分割）

并且可以在继承一个类的同时，实现多个接口；

注意事项：对应的接口实现对应的方法

**接口与接口：**

继承关系：可以单继承，也可以多继承

**抽象类和接口的区别：**

A：成员区别

抽象类：

成员变量：可以常量，也可以是变量

构造方法：有

成员方法：可以抽象，也可以非抽象

接口：

成员变量：只能是常量

构造方法：无

成员方法：只能是抽象的

设计理念区别：

抽象类：被继承体现是 “ is a”的关系。抽象类所定义的是该继承体系的共性功能

接口类；被实现是“ like a " 的关系。 接口中定义的是继承体系中的扩展功能

**Day11 debug断点调试**

**1、什么是断点：**

就是一个标记，从哪里开始

**2、如何设置断点：**

你想看哪里的程序，你就在那个有效程序的左边双击即可

**3、在哪里设置断点：**

哪里不会点哪里

目前每个方法的第一条有效语句都加上

**4、如何运行设置断点后的程序**

右键----Debug as ---java Application

**5、看那些地方：**

Debug:断点测试的地方 一次看一行的执行结果

Variables：查看程序的变量变化

ForDemo: 被查看的源文件

Console: 控制台

**Day12 String类的使用**

**Scanner类**

基本格式：

\* public boolean hasNextxxx();判断是否是某种类型的数据

\* public xxx nextXxx();获取该元素

\*

\* public int nextInt();

\* public String nextLine();输入string类型的数据

\*

\* 先获取一个数值在获取一个字符串，回车默认给了数据给String类

**String 类**

字符串：就是由多个字符组成的一串数据，也可以看成是一个字符数组

（字符串是常量，一旦被赋值，就不能被改变）

**一、构造方法：**

public String(); 空构造

public String(byte[] bytes); //把字节数组转换成字符串

public String(byte[] bytee,int index ,int length); 把字节数组的一部分转换成字符串

public String(char[] value);//把字符数组转换成字符串

public String(char[] value,int index,int length);//把字符数组的一部分转换成字符串

public String(String original );//把字符串常量值转换成字符串

字符串一旦被赋值就不能被修改（指的是值不变，而不是引用）

**面试题：**

String s1= new String("hello");

String s2 = "hello"; 有什么区别

解答：

==比较对象的地址值，比较引用类型

equals：比较引用类型，也是比较地址值，而string类重写了equals方法，比较的是内容是否相同

前者 中其实有两个对象，堆内存和方法区里分别一个，（先是常量池----然后堆内存指向方法区改变地址值，最后栈内存指向堆内存）

后者直接在方法区常量池中创建一个对象

**面试题：**

String s1 = "hello";

String s2 = "word";

String s3 = "helloword";

s3==s1+s2;

s3=="hello" + "word";

字符串如果变量相加,先开空间，在拼接

字符串如果是常量相加，先加，然后在常量池中找，如果有直接返回，否则创建

**二、String类的判断功能 :**

boolean equals(Object obj);//比较字符串内容是否相同，区分大小写

boolean equalsIgnoreScae(String str);//比较字符串内容是否相同，忽略大小写

boolean contains(String str);//比较大字符串是否包含小字符串

boolean stratWith(String str);//判断字符串是否以某个指定的字符串开头

boolean endsWith(String str);//判断字符串是否以某个指定的字符串结尾

boolean isEmpty();//判断字符串是否为空

注意：

字符串内容为空，字符串对象为空

String s = " "; //对象所表示得到内容为空，但是对象存在

String s = null; // 表示对象不存在

**三、String 类的获取功能：**

int length();//获取字符串长度

char CharAt(int index); //获取指定索引位置的字符

int indexOf(int ch); //返回指定字符在此字符串中第一次出现的索引

为什么此处是int类型，而不是char类型？

原因是 'a' 和97 其实都可以代表 ‘a’ index 返回-1 表示不存在该字符

但是如果单写成char类型，，，97就会无法识别

int indexOf(String str); //返回指定字符串在此字符串中第一次出现的索引值

int indexOf(int ch,int fromIndex);//返回指定字符在此字符串从指定位置后第一次出现的索引位置

String substring(int start)；//从指定位置开始获取字符串，默认到末尾结束

String substring(int start ,int end);//从指定位置开始到指定的位置结束

[ strat ，end } 返回的字符串只包含start索引处的字符，但是不包含end所在索引位置的字符

**字符串的遍历：**

A：如何遍历？

\* char charAt(int index)

B:如何知道字符到底有多少呢？

int length();

**字符判断是大写还是小写还是数字：**

if(ch>='0'&&ch<='9') {

number ++;//数字

}else if(ch>='A'&&ch<='Z') {

maxCount ++;//大写

}else if(ch>='a'&&ch<='z') {

minCount ++;//小写

}

**四、String的转换功能：**

byte[ ] getBytes( )；//把字符串转换成字节数组

char[ ] toCharArray();//把字符串改变成字符数组

static String valueOf(char[ ] chs);//把字符数组转换成字符串

static String valueOf(int i) ;//把int类型的数据转换成字符串

注意：String类型的valueOf能把任意类型的数据转换成字符串

String toLowerCase( );//把字符串全部变成小写

String toUpperCase( );//把字符串全部变成大写

String concat(String str);// 把字符串进行拼接

**五、String的其他功能：**

替换功能：

String replace(char old , char new) //将一个字符换成另一个字符

String replace(String old,String new)//将一个字符串换成另一个字符串

去除字符串空格：(只能去除两端的空格)

String trim( )

按字典顺序比较两个字符串：

int compareTo(String str)

int compareToIgnoreCase(String str)

分割字符串：

String str[]=s.split(" ");//将一个字符串分割成子字符串，然后结果作为字符串数组返回

Copy方法

1、 创建一个与指定的字符串具有相同值的字符串的新实例，其语法格式如下。

public static string Copy (string str)；

其中：str是指要复制的字符串；返回值是与str具有相同值的字符串。

2．CopyTo方法

CopyTo方法可以将字符串的某一部分复制到另一个数组中