JAVA复习

1. 概述

**1 常用命令行指令**

dir 查看当前目录下所有文件

cd 切换路径 （绝对路径和相对路径）

cd.. 返回上一级目录

cd. 当前路径

cls 清屏

exit 退出

ipconfig 查看本机ip地址

ipconfig /all详细信息

**2 Java语言发展简史**

1991 年Sun公司的James Gosling等人开始开发名称为 Oak 的语言，希望用于控制嵌入在有线电视交换盒、PDA等的微处理器；

1994年将Oak语言更名为Java；

**JAVAEE：**Java Platform Enterprise Edition，开发企业环境下的应用程序，主要针对web程序开发；

**JAVASE：**Java Platform Standard Edition，完成桌面应用程序的开发，是其它两者的基础；

**JAVAME：**Java Platform Micro Edition，开发电子消费产品和嵌入式设备，如手机中的程序；

**3 Java语言特性**

简单性、面向对象、可移植性、健壮性、多线程

**4 Java的加载与执行**

.java（源文件）－－编译－－.class（字节码文件）－－类装载器－－java虚拟机（JVM）－－操作系统－－硬件平台

**5 JDK和JRE**

**JDK：Java Development Kit，**java的开发和运行环境，java的开发工具和jre。

**JRE：Java Runtime Environment，**java程序的运行环境，java运行的所需的类库+JVM(java虚拟机)。

**6 配置环境变量**

电脑右键-属性-高级-环境变量，新建用户变量

让java jdk\bin目录下的工具，可以在任意目录下运行，原因是，将该工具所在目录告诉了系统，当使用该工具时，由系统帮我们去找指定的目录。

**环境变量的配置：**

1**）：**永久配置方式：JAVA\_HOME=%安装路径%\Java\jdk

path=%JAVA\_HOME%\bin

2**）：**临时配置方式：set path=%path%;C:\Program Files\Java\jdk\bin

特点：系统默认先去当前路径下找要执行的程序，如果没有，再去path中设置的路径下找。

**classpath的配置:**

1**）：**永久配置方式：classpath=.;c:\;e:\

2**）：**临时配置方式：set classpath=.;c:\;e:\

**注意：**在定义classpath环境变量时，需要注意的情况

如果没有定义环境变量classpath，java启动jvm后，会在当前目录下查找要运行的类文件；

如果指定了classpath，那么会在指定的目录下查找要运行的类文件。

还会在当前目录找吗？两种情况：

1**）：**如果classpath的值结尾处有分号，在具体路径中没有找到运行的类，会默认在当前目录再找一次。

2**）：**如果classpath的值结果出没有分号，在具体的路径中没有找到运行的类，不会再当前目录找。

一般不指定分号，如果没有在指定目录下找到要运行的类文件，就报错，这样可以调试程序。

**7 javac 和 java命令**

**javac命令和java命令做什么事情呢？**

要知道java是分两部分的：一个是编译，一个是运行。

**javac：**负责的是编译的部分，当执行javac时，会启动java的编译器程序。对指定扩展名的.java文件进行编译。 生成了jvm可以识别的字节码文件。也就是class文件，也就是java的运行程序。

**java：**负责运行的部分.会启动jvm.加载运行时所需的类库,并对class文件进行执行.

一个文件要被执行,必须要有一个执行的起始点,这个起始点就是main函数.

**8 public class 和 class**

一个.java源文件中可以定义多个class，并且一个class会生成一个.class文件；

一个.java源文件中可以没有public class，若要定义public class，则public class的名字必须和文件名保持一致，且只能有一个。

1. Java语言基础

**1 关键字**

其实就是某种语言赋予了特殊含义的单词。

保留字：其实就是还没有赋予特殊含义，但是准备日后要使用过的单词。

**2 标示符**

其实就是在程序中自定义的名词。比如类名，变量名，函数名。

只能用数字、字母、下划线和美元符号组成。

注意：

**1），数字不可以开头。**

**2），不可以使用关键字。**

**3 常量和变量**

**常量：**是在程序中的不会变化的数据。

**变量：**其实就是内存中的一个存储空间，用于存储常量数据。

作用：方便于运算。因为有些数据不确定。所以确定该数据的名词和存储空间。

特点：变量空间可以重复使用。

**什么时候定义变量？**只要是数据不确定的时候，就定义变量。

**变量空间的开辟需要什么要素呢？**

1，这个空间要存储什么数据？数据类型。

2，这个空间叫什么名字啊？变量名称。

3，这个空间的第一次的数据是什么？ 变量的初始化值。

**变量的作用域和生存期:**

变量的作用域：

作用域从变量定义的位置开始，到该变量所在的那对大括号结束；

生命周期：

变量从定义的位置开始就在内存中活了；

变量到达它所在的作用域的时候就在内存中消失了；

**4 数据类型**

①基本数据类型

整数型：byte（1）、short（2）、int（4）、long（8）

浮点型：float（4）、double（8）

字符类型：char（2）

布尔类型：boolean（1）

byte的取值范围为-128~127，占用1个字节（-2的7次方到2的7次方-1）；   
 short的取值范围为-32768~32767，占用2个字节（-2的15次方到2的15次方-1）；   
 int的取值范围为（-2147483648~2147483647），占用4个字节（-2的31次方到2的31次方-1）；   
 long的取值范围为（-9223372036854774808~9223372036854774807），占用8个字节（-2的63次方到2的63次方-1）；

float的取值范围为3.402823e+38 ~ 1.401298e-45，（e+38表示是乘以10的38次方，同样，e-45表示乘以10的负45次方）占用4个字节；   
 double的取值范围为1.797693e+308~ 4.9000000e-324 占用8个字节；

char用于存放字符的数据类型，占用2个字节，采用unicode编码，它的前128字节编码与ASCII兼容，字符的存储范围在\u0000~\uFFFF；

②引用数据类型

数组、类、接口

**5 字符编码**

Java采用Unicode编码

常见的编码方式有：UTF-8、UTF-16、ASCII（‘a’=97,‘A’=65，‘o’=48）

**6 转义字符**

\t---制表 \’---’ \\----\

\”---” \n---转行 \u----unicode码

bin目录下 nativezascii.exe命令可获得字符的unicode码

**7 整数型详解**

十进制

八进制 以O开始

十六进制 以Ox开始

**8 类型转换**

规则1：

级别从低到高：byte-->char,short-->int-->float-->long-->double

自动类型转换：从低级别到高级别，系统自动转的；

强制类型转换：把一个高级别的数赋给一个比该数的级别低的变量；

规则2：

在定义long型整数时，如果数字后面没有加‘L’且在没有超过int类型的范围时，则先将这个整数视为int型，再转为long型

如：long a = 123;//123视为int型，再自动转为long类型

而：long a = 123L;//123视为long型，不存在自动类型转换

规则3：

如果一个整数没有超过byte的范围（-127-128），则可以直接将其赋值给byte类； 如果一个整数没有超过short的范围，则可以直接将其赋值给short类；

如果一个整数没有超过char的范围（-127-128），则可以直接将其赋值给char类；

如：char a = 97;//实际是char a = ‘a’;

规则4：

字面值为小数的浮点型默认为double型 ；

如：double a = 1.2;//正确

而：float a = 1.2;//错误，必须加强制类型转换

即float a = (float)1.2;或 float a = 1.2F;

规则5：

在多种类型混合运算时，先将所有数据转换为容量最大的那种再运算；

规则6：

byte\short\char之间计算不会互相转换，全部转换成int进行计算。

**9 运算符**

①算术运算符。

+ - \* / % **%:**任何整数模2不是0就是1，所以只要改变被模数就可以实现开关运算。

+:连接符。

++,--

②赋值运算符。

= += -= \*= /= %=

③比较运算符。

> < >= <=

特点：该运算符的特点是：运算完的结果，要么是true，要么是false。

④逻辑运算符。

& | ^ ! && ||

逻辑运算符除了 ! 外都是用于连接两个boolean类型表达式。

&: 只有两边都为true结果是true。否则就是false。

|:只要两边都为false结果是false，否则就是true

^:异或：和或有点不一样。

两边结果一样，就为false。

两边结果不一样，就为true.

**& 和 &&区别：** & ：无论左边结果是什么，右边都参与运算。

**&&:短路与**，如果左边为false，那么右边不参数与运算。

**| 和|| 区别：**|：两边都运算。

**||：短路或**，如果左边为true，那么右边不参与运算。

⑤位运算符:用于操作二进制位的运算符。

& | ^

<< >> >>>(无符号右移)

⑥条件运算符（三目运算符）

<boolean表达式>?<表达式1>:<表达式2>

\*赋值运算符与运算符的区别：赋值运算符不改变运算结果类型。（E:\学习\JAVA\代码\第2章Java语言基础\Test.java）

**10 控制语句**

①条件控制语句：

if-else:

在if语句中只要有一个分支语句执行，则整个if语句结束。

分支语句中，如果只有一条Java语句，大括号可以省略。

switch:

break是可以省略的，如果省略了就一直执行到遇到break为止；

default可以写在switch结构中的任意位置；如果将default语句放在了第一行，则不管expression与case中的value是否匹配，程序会从default开始执行直到第一个break出现。

②循环语句：

for(表达式1;表达式2;表达式3){}

表达式1是初始化表达式，最先执行，只执行一次

表达式2是boolean类型的表达式

表达式1、2、3都可以没有，但“;”不可以省略

while

do-while

**break:**作用于switch ，和循环语句，用于跳出，或者称为结束。

break语句单独存在时，下面不要定义其他语句，因为执行不到，编译会失败。当循环嵌套时，break只跳出当前所在循环。要跳出嵌套中的外部循环，只要给循环起名字即可，这个名字称之为**标号**。

**continue:**只作用于循环结构，继续循环用的。

作用：结束本次循环，继续下次循环。该语句单独存在时，下面不可以定义语句，执行不到。

**11 方法初步**

为了提高代码的复用性，可以将其定义成一个单独的功能，该功能的体现就是java中的函数。函数就是体现之一。

**java中的函数的定义格式：**

**修饰符 返回值类型 函数名(参数类型 形式参数1，参数类型 形式参数1，…){**

**执行语句；**

**return 返回值；**

**}**

当函数没有具体的返回值时，返回的返回值类型用void关键字表示。

如果函数的返回值类型是void时，return语句可以省略不写的，系统会帮你自动加上。

**return的作用：**结束函数。结束功能。

return语句后不能再有别的语句，否则编译不能通过，但要注意程序分为编译期和运行期，编译期不会进行判断和运算。（E:\学习\JAVA\代码\第2章Java语言基础\MethodTest2.java）

**如何定义一个函数？**

函数其实就是一个功能，定义函数就是实现功能，通过两个明确来完成：

1）、明确该功能的运算完的结果，其实是在明确这个函数的返回值类型。

2）、在实现该功能的过程中是否有未知内容参与了运算，其实就是在明确这个函数的参数列表(参数类型&参数个数)。

**函数的作用：**

1）、用于定义功能。

2）、用于封装代码提高代码的复用性。

注意：函数中只能调用函数，不能定义函数。

**主函数：**

1）、保证该类的独立运行。

2）、因为它是程序的入口。

3）、因为它在被jvm调用。

**函数定义名称是为什么呢？**

答：1）、为了对该功能进行标示，方便于调用。

2）、为了通过名称就可以明确函数的功能，为了增加代码的阅读性。

**重载（Overload）**的定义是：在一个类中，如果出现了两个或者两个以上的同名函数，只要它们的参数的个数、类型或顺序不同，即可称之为该函数重载了。

**如何区分重载：**当函数同名时，只看参数列表。和修饰符列表、返回值类型没关系。

**12 方法的执行原理**

方法在调用的时候，才会给该方法在内存中分配空间；

如果这个方法只是定义没有调用，则不会在内存中分配空间；

方法在调用的时候，在“栈”中分配空间；

方法调用其实就是“压栈”，方法结束其实就是“弹栈”。

**13 递归**

方法调用方法本身。

1. 面向对象

1 面向对象的三大特征

封装、继承和多态

2 类与对象的概念

类：是对具有共性实物的抽象描述，是在概念上的一种定义。

对象：也叫实例，也就是一个类的具体化，就是对象或实例。

基于面向对象可以分为三个阶段：

OOA 面向对象的分析

OOD 面向对象的设计

OOP 面向对象的编程

在类中定义其实都称之为成员。成员有两种：

1**：成员变量**：也叫实例变量、非静态变量，其实对应的就是事物的属性。

2**：成员函数**：其实对应的就是事物的行为。修饰符中不加static，称为成员方法，实例方法，非静态方法。

3 成员变量和局部变量的区别

1：成员变量直接定义在类中。

局部变量定义在方法中，参数上，语句中。

2：成员变量在这个类中有效。

局部变量只在自己所属的大括号内有效，大括号结束，局部变量失去作用域。

3：成员变量存在于堆内存中(的对象中)，随着对象的产生而存在，消失而消失。

局部变量存在于栈内存中，随着所属区域的运行而存在，结束而释放。

4 类的定义

<类的修饰符> class <类名> (extends <父类型名称> implements <接口名称>){

<类体>;

//属性和方法组成

}

5 创建对象

<类名> <对象名> = new <类名>();

①<对象名>保存的是该对象的堆内存地址，相当于指针，属于局部变量，存储在栈区；

②new出来的对象本身存储在堆内存空间中；

③访问成员变量，必须采用<引用>.<属性>

④private修饰的数据只能在本类中访问，对外提供公共的set和get方法。

6 构造函数

<修饰符列表> <构造方法名> (<形式参数列表>){

<方法体>;

}

①没有返回值类型；

②构造方法的方法名必须与类名一致；

③作用：创建对象；给成员变量赋值（初始化成员）

④调用：new <构造方法名>(<实参>);//在堆中开辟空间存储对象

⑤如果一个类没有提供任何构造方法，系统默人提供无参数的构造方法，如果一个类已经编写了构造方法，那么系统不会再提供任何构造方法。成员变量只有在调用构造方法时，才会给成员变量赋值。

Person p = new Person();

创建一个对象都在内存中做了什么事情？

1：先将硬盘上指定位置的Person.class文件加载进内存。

2：执行main方法时，在栈内存中开辟了main方法的空间(压栈-进栈)，然后在main方法的栈区分配了一个变量p。

3：在堆内存中开辟一个实体空间，分配了一个内存首地址值。new

4：在该实体空间中进行属性的空间分配，并进行了默认初始化。

5：对空间中的属性进行显示初始化。

6：进行实体的构造代码块初始化。

7：调用该实体对应的构造函数，进行构造函数初始化。（）

8：将首地址赋值给p ，p变量就引用了该实体。(指向了该对象)

7 Java内存的划分

堆区：存放成员变量、new出来的对象，此对象由垃圾收集器收集；

栈区：每调用一个方法，就会创建一个栈帧，存放局部变量，包括引用；

方法区：类的所有信息，包括所有的方法、静态变量和常量等。

关于垃圾收集器：如果堆中的对象没有引用指向它，那么该对象就变成了垃圾，等待垃圾回收器的回收。

8 this关键字

this是一个引用类型，保存了内存地址。在堆中的每一个对象都有一个指向自身的this引用类型。（ThisTest.java）

this到底代表什么呢？哪个对象调用了this所在的函数，this就代表哪个对象，就是哪个对象的引用。

开发时，什么时候使用this呢？

在定义功能时，如果该功能内部使用到了调用该功能的对象，这时就用this来表示这个对象。

①this可以用在成员方法中；

②this可以用在构造方法中；

③this不可以用在静态方法中；

④this 还可以用于构造函数间的调用。

调用格式：this(实际参数)；

this对象后面跟上 . 调用的是成员属性和成员方法(一般方法)；

this对象后面跟上 () 调用的是本类中的对应参数的构造函数。

注意：用this调用构造函数，必须定义在构造函数的第一行。因为构造函数是用于初始化的，所以初始化动作一定要执行。否则编译失败。

9 static关键字

一个修饰符，用于修饰成员(成员变量和成员函数)。

①static还可以定义静态语句块

static{

<java语句>;

}

static 定义的静态语句块在类加载阶段执行，并且是自上而下的顺序执行的。**可以完成类的初始化。静态代码块随着类的加载而执行，而且只执行一次（new 多个对象就只执行一次）。如果和主函数在同一类中，优先于主函数执行。**

②关于实例语句块

{

<java语句>;

}

每次调用构造方法前会执行一次,注意调用构造方法时，会将类进行加载，但只加载一次，因此第一次调用构造方法会先执行静态语句块，再执行实例语句块，再执行构造方法，第二次调用时不再执行静态语句块，但是会执行实例语句块，再执行构造方法。

③静态方法

静态方法可以采用“类名.”的方式进行调用，也可以采用“引用.”的方式，但底层还是采用“类名.”的方式

静态方法中不能直接访问非静态变量

静态方法中不能使用this

④静态变量

静态变量被存储在方法区，所有的JAVA对象共享这一份，所以静态变量是类级别的，采用“类名.”的方式访问。

如果这个属性所有的对象都有，并且这个属性的值是相同的，则该属性声明称静态变量。

成员变量：在创建对象时候初始化

静态变量：在类加载阶段赋值，并只赋值一次

⑤关于代码的顺序

静态语句块和静态变量都在类加载时就执行了，因此有顺序区别。

特点：

①想要实现对象中的共性数据的对象共享。可以将这个数据进行静态修饰。

②被静态修饰的成员，可以直接被类名所调用。也就是说，静态的成员多了一种调用方式。类名.静态方式。

③静态随着类的加载而加载。而且优先于对象存在。

弊端：

①有些数据是对象特有的数据，是不可以被静态修饰的。因为那样的话，特有数据会变成对象的共享数据。这样对事物的描述就出了问题。所以，在定义静态时，必须要明确，这个数据是否是被对象所共享的。

②静态方法只能访问静态成员，不可以访问非静态成员。

因为静态方法加载时，优先于对象存在，所以没有办法访问对象中的成员。

③静态方法中不能使用this，super关键字。

因为this代表对象，而静态在时，有可能没有对象，所以this无法使用。

④主函数是静态的。

**成员变量和静态变量的区别：**

1，成员变量所属于对象。所以也称为实例变量。

静态变量所属于类。所以也称为类变量。

2，成员变量存在于堆内存中。

静态变量存在于方法区中。

3，成员变量随着对象创建而存在。随着对象被回收而消失。

静态变量随着类的加载而存在。随着类的消失而消失。

4，成员变量只能被对象所调用 。

静态变量可以被对象调用，也可以被类名调用。

**所以，成员变量可以称为对象的特有数据，静态变量称为对象的共享数据。**

**生成Java帮助文档：**命令格式：javadoc –d 文件夹名 –auther –version \*.java

/\*\*

\*类描述

\*@author 作者名

\*@version 版本号

\*/

/\*\*

\*方法描述

\*@param 参数描述

\*@return 返回值描述

\*/

10 设计模式