·**第一章:**Java 运行平台：JavaSE(J2SE)标准版平台（桌面开发和低端商务应用、Applet）；Java EE（企业）API、开发网络、web 应用 JavaEE(J2EE)企业版平台、JavaME(J2ME)(嵌入式设备)（微型）移动设备；Java CARD SIM、ATM 卡。JRE（运行环境）JDK（开发工具包）javac.exe 编译器 java.exe 解释器 jdb.exe 调试器 javap.exe 反编译。Java 特点：简单、面向对象、分布式、解释型、健壮、安全、架构中立、可移植、高性能、多线程、动态。 描述Java 技术的主要特性:java 虚拟机, 垃圾回收, 代码安全性。Java 虚拟机：字节代码，垃圾回收有一定滞后性. **命令行编译：**javac Hello.java →生成 Hello.class→运行 java Hello。**简述如何搭建Java 开发环境:**首先下载安装JDK 然后配置环境（1）配置PATH， 操作系统运行环境的路径（2）配置CLASSPATH JAVA 运行应用程序时所需要的类包的路径 （3）配置JAVA_HOME 供需要运行 JAVA 的程序使用.

·**第二章:**1.java 文件->编译->class 文件->java 虚拟机,解释机器码(java.exe)->程序 2.标识符:字母数字_$不能数字开头 3.关键字:enum、transient、strictfp、volatile、native、(goto,const 保留字)4.数据范围:077(八)0x3ABC（十六）5.Byte1 字节 8 位 $-2^7 \sim 2^7-1$ (byte 运算必须转化 byte s = byte（a+b）) 6.short2 字节 16 位$-2^{15} \sim 2^{15}-1$ int4 字节 32 位$-2^{31} \sim 2^{31}-1$ int32 位$-2^{31} \sim 2^{31}-1$ 7.long8 字节 64 位$-2^{63} \sim 2^{63}-1$ (后面加 l 或 L) boolean 单个 32 位、数组中 8 位 8.char2 字节 0~65535(0~2^16-1)('1'、转义字符、'\u????'(四位十六进制)使用时是一个数字)'t'水平制表符'n'换行'f'表格符'r'回车'\"'双引号'\''单引号'\\'反斜线 9.float4 字节 1.4E-45 ~ 3.4E+38(后面要加 f) 10.double8 字节 4.9E-324~1.7E308(后面加 d/D 可缺省)Long.Size 获得位数 11.默认值:数据的都是 0,char 是'\u0000',String 是 null,boolean 是 false 精度级别:byte short int long float double(左到右自动转换，右到左显式转换)(130 转化为 byte00000000010000010->10000010(负数)->绝对值 01111110(126)->真实值-126)(float->int 不会四舍五入) 12.System.out.printf("%5.2f")输出小数部分最多 6 位的 float 数据，小数点后 2 位(会四舍) 常量值 1 到常量值 n 也必须是整型或字符型 14.数组声明不可，要使用数组要分配空间。数组下标访问 ArrayIndexOutOfBoundsException 15.%可以对浮点数用，3.5%2 = 1.5，++可以对浮点型用 17.Instanceof 确定某个对象是否属于某个类 18.<<左移右补 0>>右移左按符号补 0 或 1>>>不带符号右移左补 0 19.switch 中表达式是整型/字符型；20.A.基本（简单）数据（注意 String 不是）：1.数值：整型 btye、short、int、long；浮点 float、double；2.字符（char）；3.布尔（boolean）B.复合数据：类接口数组

·**第三章:** 1.面向对象特性:封装继承多态 3.没有实体的对象使用会有 NullPointerException(运行出错) 4.final 修饰的常量,不占内存,不能通过类名访问。用 static 修饰的成员变量称为静态变量（类变量） 5.重载:参数列表不同即可.(返回值可以不同) 6. package cn.edu.szu.javapd.pwk; 13.java.lang.Number 中有基本类型的类包装 ByteShortLongInteger FloatDouble Character. 7.面向对象好处：模块化、信息隐藏、代码重用、易调试.8.Java 中只有按值传递，没有按引用传递 9.this 代表对当前对象的引用，不能出现 10.在类方法中：在构造方法中使用 this 调用其他构造方法时须放在方法第一 11.用 static 修饰的成员方法称为静态方法（类方法），不使用则称为实例方法；类方法可通过类名或对象名来调用

·**第四章:**继承与接口 1.重写/覆盖：名字返回类型参数个数和类型全相同。2.重载是编译时处理覆盖是运行时处理.3.访问权限要大于等于被覆盖方法的权限，例外列表要小于等于被覆盖方法的例外列表。9. abstract 类可有构造方法 10.abstract 类中有 abstract 方法(只声明不实现)和非 abstract 方法 10. interface 声明，只含常量、方法声明，不含方法体，可以声明标量，不可实例化 11.子类不能继承父的初始化方法。当用子类的初始化方法创建一个子类的实例对象时，这个子类声明的所有成员变量都被分配了内存空间，直接父类和所有的祖先类的成员变量也都分配了内存空间。17.如果父类实现了某个接口，则其子类也就自然实现这个接口。18. 非 static 内部类的不可以声明静态变量和静态方法，只可以申请静态常量 20.在类 A 中的内部类 B,若 B 中要使用 B 创建对象，则需要 B obj = new A().new B(); 21 匿名类一定是内部类，可以继承，匿名类不可以声明静态成员变量和静态方法。常用:方法参数（new 类名 or 接口名{重写方法}）22.异常类: Exception 的相应子类 23.**上转型变量:**不能通过上转型对象变量访问子类对象实体中的成员变量和成员方法；通过上转型对象变量访问子类对象实体重写的父类成员方法，执行的代码是子类重写的方法体；可通过上转型对象变量访问父类被隐藏的成员变量；可通过强制类型转换将上转型对象变量转换为子类对象变量 24.**super**:子类方法通过使用 super 调用父类中被子类隐藏的成员变量和覆盖的成员方法/子类的初始化方法中使用 super 调用父类的初始化方法(必须在子类的初始化方法的第一)25.类方法的方法体中不能有与类的对象有关的内容 1.类方法中不能引用对象变量不能调用类的对象方法不能调用 superthis 关键字；4.类方法不能被覆盖。

· **第五章:**String 表示一个 UTF-16 格式的 2byte 字符串 2. java.lang.String 包 3.构造函数:**String(char[] a) String(char[] a,startpos,length)** 4.String 的长度是 **str.length()** 数组元素个数是 **arrey.length**（无括号）5.比较字符串字典序大小用 **str.comparTo(String str)**返回 0 负数代表等于小于 7.**str.equalsIgnoreCase(str2)**忽略大小写 **equals,str.toUpperCase()str.toLowerCase()** 字符串全字符转为大小写 6.**str.startWith(String string) .endWith(String string)**判断 str 是否以 string 开头或者结尾 7. **str.contains(子串)**判断有无子串 **int indexOf(String s)**判断字串第一次出现的位置没有该串返回 8.String substring(int startpoint,int endpoint)截取从 stratpoint 到 endpoint-1 的子串 9. **String replaceAll(String s1, String s2)**将 str 中的所有 s1 子串改为 s2 10. **String trim()**返回去除前后空格的串 11.字符串转化为数据类型**:int a=Integer.parseInt(String s) Integer a=Interger.valueOf(String s)** 12.数据类型转化为字符串,**String.valueOf(数据值)** 或者 str = "”+数 或者 **str = Integer.toString(int a)**12.进制转换:Integer.toString(值,进制);返回一个值在该进制下的字符串 13. **public void getChars(int start, int end, char c[], int offset)**截取 string 中 start 到 end 到从 c 的 offset 位置开始（不包括 end） 14. public char[] toCharArray() 字符串转字符数组 15. 用字节数组构建字符串 **String(byte[]),String(byte[],start,length)str.getBytes** 获得字符串字节表示 15.**str.charAt(i)**获得 i 位置字符 16.**str.concat(String str2)**字符串连接,返回一个新的字符串 17.**str.replace(char a,char b)**返回一个将 str 中 all 字符 a 换成 b 的串 18.StringBuilder 和 StringBuffer 构造函数 **StringBuffer(int capacity)**容量为 capacity,StringBuffer()默认容量 16 StringBuffer(String s)容量为 len(s)+16 19. 常用方法 **append(num/char[]/String) delete(start,end)**(删除 start~end-1) **deleteCharAt(index) insert(offset**, 数据字符数组字符串) 字符数组有 **insert(index,char[]str,start,len) replace(start,end,String)**指定位置替换字符串(即删除 start 到 end-1 的串并换成新串)**setCharAt(index,char) reverse()**反转并返回 StringBuilder、toString 返回 String 对象 19.String[] **split(String regex)** 用 regex 作为分隔符 20.**Scanner(String str),Scanner.useDelimiter(regex)**用正则表达式作为分隔符 "[^0123456789.]+"匹配非(数字和小数点)String.format("%d",123); 21.线程安全 StringBuffer 22.StringBuffer 的 setLength(int len)，设置序列长度，大了就增加空字符，小了就截。

· 第六章:泛型 java.util.*;： 1.Java 中集合分为两大类,实现了 Collection 接口(包括 List:ArrayList,LinkedList,Vector 和 Set:HashSet,TreeSet,LinkedHashSet)和实现了 Map 接口(包括 HashMap,TreeMap,HashTable) 2 **ArrayList(int capaticy=16)** 3.ArrayList<E> :**add(E) add(index,E) get(index) Object clone()**强转 **remove(int i) bool contains(Object) indexOf(Object)(-1) isEmpty() lastIndexOf() set(index,value) removeRange(start,end) size() Object[] toArray()** LinkedList: **addFirst(),getFirst(),removeFirst()**（or Last）4.Vector 最安全。线程安全版 **List list = Collections.synchronizedList(new ArrayList<>()/或者为 LinkedList<>())**5.HashSet<E>(不保证元素顺序不变化)如果是 E 是自定义对象,则需要在对象中重写 int hashCode{return 自定义 hash 编码 ;}boolean equals(Object obj){if(obj.hashCode()==this.hashCode())return true;return false;} 线程安全 :Set set = Collections.synchronizedSet(new HashSet<>()); 操作:add,remove,contains,size,clone 无 get,**用迭代器获得元素(Treeset<String>is=new TreeSet<String>();Itertator<String> it=is.iterator()返回迭代器 while(it.hasNext){it.next();}**还有集合操作:A.**addAll(B)**并 **retainAll** 交 **removeAll()**差 结果在 A 中 6.TreeSet<E>是有序的集合若 E 为自定义类,需要实现 Comparable<E>接口重写 ComparTo 方法 public int compareTo(E obj) { return obj.attr >this.attr?-1:(obj.attr==this.attr?0:1);}升序. String 的比较：标点最前，然后大写字母，然后小写。线程同步 **SortedSet s = Collections.synchronizedSortedSet(new TreeSet<>());**7.HashMap 无序 快 TreeMap 有序 慢 LinkedHashMap 有序 快 7.HashMap<T,E>注意若 T 为自定义类,要跟 HashSet<>一样重写两个方法 操作:**put(Key,Value) get(Key) containsKey(Key) containsValue(Value) remove(Key) Set set = hashmap.KeySet() Collection values = hashmap.values()**可用迭代器 Iterator 或 ArrayList 包装强制转换 values 8.TreeMap 要在自定义的 Key 类中实现 Comparable 接口(同上)或者在构造 TreeMap 时提供一个比较器对象参数,比较器为实现泛型接口 Comparator<E>的类,方法:**public int compare(E o1,E o2) { return o1.attr>o2.attr?1:(o1.attr==o2.attr?0:-1);} 线程安全：Map m=Collections.synchronizedMap(new HashMap(..))**

· 第七章:异常处理 java.lang.*; java.io.*; java.util.*; java.net.*：2.异常-生成异常对象-交给 JVM(过程称为抛出异常)JVM 寻找(遍历栈)能处理该异常的方法,交给这个方法(捕获异常)(默认处理是输出异常信息终止程序)3.标准异常类都是 Exception 的子类 4.Throwable(java.lang 包)直接子类:Error:OutOfMemoryError, ThreadDeath(致命错误,与 JVM 相关,由系统处理入线程死亡,内存溢出)Exception(异常) 4.Exception(String)Exception()5.异常的方法 printStackTrace()输出异常的信息 6. try- catch-finally 语句块 finally 块是个可选项(一定执行)7.常见异常:运行异常 ArithmeticException、 ArrayIndexOutOfBoundsException、 NullPointerException、 ClassCastException(类型转换异常)NumberFormatException 非运行异常:IOException、FileNotFoundException、SQLException8.自定义异常类:继承 Exception,构造函数(String str),并且要 super(str) 8.Exception 分为运行时异常 RuntimeException(不检查异常)和非运行时异常(检查异常) 10.子类重写父类的带抛出异常的方法:1.不声明抛出异常 2 声明抛出的异常必须是其父类方法声明抛出的那种异常或者其子类异常

·第八章:3.没有结束 run()方法之前不能再 start()否则 IllegalThreadStateException 5.创建线程两种方法(可以同时用):1.继承 Thread 类(已经实现 Runnable 接口)重写 run()方法 2.实现 Runnable 接口实现 run()方法 6.方法:name()start()run()setName()getName()toString()(名称,优先级,所属线程组)currentThread(当前运行的线程引用)isAlive()(start-run 为 true,其余为 false)sleep(int ms)(静态方法)interrupt()(中断休眠进入就绪)wait()(等待 notify())notify()(唤醒正在等待的线程)notifyAll(唤醒所以等待的线程) join()t.join()让 t 运行,t 运行完当前线程再继续(wait 和 sleep 和 join 放在 try-catch 中异常为 InterruptedException) 4.优先级(启动前设置)t.setPriority(0~10 值大级大)(超出范围 IllegalArgumenException)多个线程排队优先级高的先运行相同优先”先到先得”,getPriority() 5.synchronized 关键字(常在方法中 synchronized(this){语句})方法声明 synchronized(与 static 同位置)6.协作:wait()暂停线程执行并释放锁,notify()会唤醒一个等待的线程(两线程协作,常需要轮询等待 while(balance>=5000){try{wait()}catch(InterruptedException){}})balance 是一个必要的控制元素。进行一系列操作后再 notify()(注意,**有 notify()的方法必须是同步的**)7.join 挂起:B.start()等到 B 运行就把 B 加入 while(threadB.isAlive()==false){} try{ threadB.join(); } catch(InterruptedException e){} 7. 具有相同优先级的多个线程的调度不一定是分时的 9. 如有高优先级的线程就绪，正在运行的低优先级的线程将暂停执行 10.多个线程的运行顺序其实与线程的优先级有关，与线程的启动顺序无关。

· 第九章:IO import java.io.*; import java.util.*;(Scanner 类) 文件流注意 FileNotFoundException： 3.File 类:File(“pathname”)File(URI) **getName canRead canWrite exists length getAbsolutePath getParent isFile isDirectroy mkdir** 创建目录 list,listFiles 返回目录下的文件名数组,文件对象数组 **file.delete()**删除文件 **renameTo(File newName) long lastModified()**最后一次修改时间 4.InputStream 类(以字节为基本处理单位):**read()**(读一个字节)**read(byte[])**(读取一定量的字节)**read(byte[],start,len)**(读取 len 个字节入 byte)read 到达结尾返回(-1)**skip(int)**放弃 n 个字节 **available()**返回可读取字节数 **close()**关闭 5.**OutputStream**:write(int b)把字节写入,**write(byte[]),write(byte[].start,len) flush()**刷新输出流并强制写出所有缓存的输出字节 close()关闭 6.FileInputStream(String)/(File)(FileNotFoundException)通常是 **byte buffer[] = new byte[2];while(in.read(buffer,0,2)!=-1){String str = new String (byte,0,2) ;System.out.print(str);}in.close();catch..**7.FileOutputStream(文件夹 **FileNotFoundException** 不存在自动创建)常 **int count,n=512;byte buffer[]=new byte[n];count=System.in.read(buffer); FileOutputStream out=newFileOutputSteam(filename); out.write(byte,0,count); out.close();** 8.DataInputStream(InputStream)DataOutPutStream(同上)常: **DataInputStream(new FileInputStream(File/Name))**方法:**readInt**,**readChar**,**readLine**,**readShort**,**readDouble**,**readFloat**. DataOutputStream: write 数据类型 writeChars()输出字符串.
9. 字符流:Reader:**read()**读取单个字符 **int read(char[])**(需要强制转化 char)**int read(char[],start,len)**(返回读取的个数,无法读取返回-1)**skip(n) close()** Writer: **write(int c) write(cjar[],start,len)write(String)flush()close()**10.**FileReader(String/File)FileWriter(String/File)FileReader fr = new FileReader("out.txt");int c;char[] buffer = new char [message.length()] ; while((c=fr.read(buffer,0,buffer.length))!=-1) {String str = new String(buffer,0,c);System.out.println(str);}**10.缓冲流:**BufferReader(Reader)BufferWriter(Writer)**常用 FileReader 和 FileWriter 对象为参数构建,**br.readLine() (while((str=br.readLine())!=null && str.length()!=0)){int read()(需强制转换) bw.write(String) bw.newLine()}bw.flush(), bw.close()**11. **InputStreamReader:BufferReader br =new BufferReader(new InputStreamReader(System.in));** 常用代码 **:while(scanner.hasNext()){ try{ int p = scanner.nextInt();} catch(InputMismatchException e){ String t = scanner.next(); } }**

当前类	同一pack	子孙类	其他pack
public	√	√	√
protected	√	√	×
friendly	√	×	×
private	√	×	×

第十章: java.net.*;： 1.URL 三部分:协议地址资源<URL 的访问协议>://<主机名或 IP 地址>:<端口号(可选)>/<路径> http、ftp、file 协议 InputStream openStream()获取 URL 资源 getProtocot,Host,Port,Path.Query,Ref 获得协议,主机,端口,路径,查询,URL 对象的引用 .2.JEditorPane 类可以解释执行 html 文件 3.JEditorPane 对象调用 addHyperlinkListener(HyperlinkListener listener)获得监视器 4. InetAddress 对象含 Internet 主机地址的域名和 IP 地址:getByName(主机名或地址) getLocalHost()主机地址(UnknownHostException) 套接字:端口号与 IP 地址的组合得出一个网络套接字 5.UDP **无连接不可靠** 16 位端口(0~65535)(0~1023 被占) datagrampacket =newDatagramPacket (str.getBytes(),str.length(),InetAddress.getByName(“...”),8000); datagramsocket.send(datagrampacket);|接收 UDP： byte[] b = new byte[1024]; dgs = new DatagramSocket(8000); dgp = new DatagramPacket(b,b.length); dgs.receive(dgp); .发送 **ClientSocket.send(Packet(参数多的 package))**接受 **ServerSocket.receive(参数少的 package)**发送或者接收完之后要 **Socket.close();** 6.TCP 面向连接,先建立连接再通信 **Socket(“域名如 time.nist.gov”,”端口”).ServerSocket(端口)ServerSocket.accept()**返回一个 **Socket|(Server)Socket.getInputStream** 通常包装为 **Scanner(InputStream)**或者 **StringReader(new InputStreamReader(InputStream))**用于接收信息|**(Server) Socket.getOutPutStream** 通常包装为 **PrintStream(OutputStream), print(Obj) flush() close()**还有 **PrintWriter(OutputStream,true) println(String) flush() close()**(注意 IOException)|半关闭:**shutdownOutput()可**以关闭输出流,只接收。多线程:**while(true)去接受 socket**,开启一个 **Handler**(实现了 Runnable 接口)线程去处理。注意流的关闭与 socket 的关闭。**TCP 连接实现服务器** InetAddress address = InetAddress.getByName("127.0.0.1"); DatagramPacket data = new DatagramPacket(b,b.length,address,4000); DatagramSocket mail = new DatagramSocket();mail.send(data);// 发送信息 pack = new DatagramPacket(b,b.length);mail = new DatagramSocket(4001);Socket socket = new Socket(hostname,port); InputStream in = socket.getInputStream(); BufferedReader br=new BufferedReader (new InputStreamReader(socket.getInputStream()));DataOutputStream dos = new DataOutputStream(os); dos.writeUTF("Hello," +s1.getInetAddress() + "port#" + s1.getPort() + "\nbye!"); PrintStream ps=new PrintStream(socket.getOutputStream());

· 第十一章 javax.swing.*; 1.setDefaultCloseOperation JFrame.DISPOSE_ON_CLOSE JFrame.EXIT_ON_CLOSE JFrame.DO_NOTHING_ON_CLOSE JFrame.HIDE_ON_CLOSE2. 顶层容器:JFrame,JDialog,Japplet 3.中间层容器:Jpanel 4.线程调度安排任务 Java.swing.SwingUtilities.invokeLater(new Runnable(){public void run(){...}});