数组声明 = new 类型[长度]; 总数.lenth 二维[][] 二维长度[0].length

字符串池——对象池，用的时候直接取出，使用new会开辟一个新的，如果使用匿名的””就直接从池子中取出字符串对象;

字符串方法 构造 通过字符数组 ，通过字节数组， 生成字符数组，生成字节数组

长度length() 子串索引indexOf 截取返回新串 substring 拆分成串数组split

大写 是否以串开头 是否相等 替换

This关键字 构造 this() 指代对象实例 this 指向对象属性this.name

枚举就是一组有序排列的命名常量，从0到n，把常量包装在一个**枚举类**中

枚举可以从值获得名字 Color[2]

5.11 JAVA可以定义代码块 并加上限制符，不推荐 {}

13对象数组 一开始是Null 字符串数组也是对象数组，数组本身也是对象

14内部类 15、模拟系统登录 单向链表实现操作封装

六、 1、对象实例化的过程：先调用父构造（无参），再调用子 构造方法不同名所以不是重写而是调用 可以写个父的功能print看看

2、方法覆写 属性覆盖 修改访问级别不算覆盖 使用方法是调用父方法

3、super super()无参构造 带参构造super(a,b)

动态数组的实现，数组游标 4、final

5、向上转型是为了实现多态（但是子类独有的属性和方法被屏蔽） 向下转型是为了恢复对象的全部功能，没有向上转型就不能向下转

多态代替if swith 多态代替重载 多态实现不同功能。。。。多态实现对象替换，各种模式，代理 委托

Instanceof 是哪个类实例（向下转型前先验证）

**不要继承实现好的类，而是继承抽象类， 抽象类是设计，实现类是实现，多个实现类是平行关系，如果继承实现好的类，那就是层级嵌套，形成了父子关系**

抽象类，模板方法 **接口：制定标准** ，api 工厂模式 代理模式 适配器

内部类

11、Object equals hashCode() toString equals()默认比较的是地址，比较不了内容 ——String除外

**所有对象都可以转型成Object,包括数组**

**包装类：一切都是对象 继承Number类或者是Charactor**

**装箱方法 new A(int) 拆箱 .intValue()**

**自动装箱，通过赋值符， 类型转换 int = Integer Integer =20**

**字符串转基本类型 parseXXX**

12、 匿名内部类——必须要有接口 ，利用接口直接实现方法

13、私有构造函数 单例模式

七、异常类

异常处理程序thow

2、 可以写多个不同类型的catch（） 捕获不同类型的异常，进行不同的处理

3、带throws的方法，**方法不处理异常**，由用户自己捕获处理，如果不处理方法就不能执行 （写了throws的方法内部所有代码都可以不处理异常）

4、异常和运行时异常 运行时异常可以不处理

静态导入

jar

九、线程

同步 ：需要顺序执行，需要先获得某个属性，才能向下执行

同步代码块 同步方法

死锁：都在等彼此

生产者消费者

十、泛型 泛型数组？ 只能使用包装类

泛型方法 泛型变量 泛型对象

安全警告：对于类级泛型 构造的时候应该直接传入类型<String>

可以有一个泛型或者多个泛型 使用同样泛型符号的必须传入相同类型 A,B,C

泛型上下限 extends 类和其子类 super 类和其父类

十一、常用类 StringBuffer 有很多追加操作 toString()

2、运行时 Runtime jvm进程 内存空间

Process

3、国际化 资源/属性文件

系统类 垃圾回收

日期 日历 日期格式化

数学

随机数类

数字格式化

大整数 大浮点数

对象壳隆

数组操作 Arrays

比较器

观察者

正则 和匹配器 String类

定时器（线程）

断言

正则

十二、JAVA IO 包io

File(path) 常量 多路径分隔符.路径中分隔符 创建新文件createNewFile

删除文件 .delete() 是否存在？exsist 是否是目录isDirectory 大小length() 创建文件夹mkdir列举文件名 list 列举所有文件完整路径名listFiles

**带throws的方法和类是要求必须进行异常捕获的**

File.seperotor可以适应多个平台

列举所有文件可以用递归法 判断是否目录

2、随机访问类 用的不多 一般使用流

3、流 标准输入 System.in 标准输出System.out cout 把内容向某处输出（写文件）

Cin 从某处读入内容 流实际上就是一个用缓冲数组保存内容的对象，然后对内容进行处理 读取文件到数组或字符传， 或者把字符写入文件，

**字节流只能处理字节数组，适合二进制文件读写**

OutputSteam(写入到的文件，或者屏幕 ) write close flush刷新缓冲（清除）

**写入方法需要try异常** 字节数组写入到文件 write(byte[] )

子类向上转型 new FileOutputStream 文件输出流 out= fout

不存在的文件会被自动创建

字符串转byte数组 .getBytes() 追加方式输出new FileOutputStream(file,true)

InputStream (被读出的文件) 字节输入流

Available()文件大小可以获取 read() 每次读取一个字节，需要循环

read(byte[])读入到字节数组中（**返回读取到的长度**，很有用，不过最好用file.length()确定文件大小） read(b)

2、判断读取到了文件尾 .read()每次一字节 最后会返回-1

3、字符流， 一个字符两个字节？

Writer close write(char[])写入字符数组 String 写入字符串 清空缓存flush 关闭close FileWiter(f) 追加FileWriter(f,true) .write()

Reader FileReader() read读取一个字符 read(char[] ) 读取到字符数组(不能读取到字符串，因为字符串不能变) 返回长度

**字符数组转字符串：new String(char,start,length)**

**字符流处理的是Unicode**

字节流不缓冲，是直接写入文件的 字符流先缓冲再写 所以必须关闭或者flush();

2、写入方式： 边读边写节省内存 for read(); write();

3、转换流 输出流写入 输入流读取 OutputStreamWriter InputReader

字节转字符 OutputWriter(OutputStream(f) )

.write(string) InputStreamReader(FileInputStream(f)) read( c[])

4、内存操作流 输入到内存 从内存输出，写入到其他文件

字节数组流 out.write(int ) ByteArrayInputStream(byte[] )

？？？？？逻辑都混乱了

5、管道流 piped 线程通信 PipedOutputStream() out.write(b) str.getBytes()写入到管道 Pos .read(b) 从管道读入到数组 pis

管道需要连接才能用 pos.connect(pis)

6、打印流 print printf println() 用来包装输出流，方便输出

New PrintStream(outputStrem)

7、系统打印流 System.out in

err System.err.print(e) err和out功能没区别

in是一个InputStream对象， 对象本身连接键盘标准输入 System.in.read(b)

把键盘输入 读出 到byte数组

8、重定向标准输入流，输出流 System.setOut( out ) 参数是新的流比如FileOutputStream，必须用PrintStream包裹 setError

9、缓冲区 读取(一次写一行) BufferedReader（Reader in） 必须是一个字符流对象 ，可以是文件字符流 ，或者字符转换流转换来的 包括标准输入 (new InputStreamReader(System.in))

可以按行读取 str= readLine() **注意读取的类型都是字符串，如果要数字需要转换**

Integer.parInt()

常见输入小数，日期，字符串，整数， 可以一一解析成相应类型 buf.readLine

2、循环读取输入，某个输入可以break read本身就会阻塞循环

日期格式化类

3、输入数据类 Scanner 更好用， 方便验证

Scanner(文件 )（需要抛出） （InputStream）字节流

hasNext( pattern ) 输入验证正则 是否整数 hasNextInt Float

开始输入，返回字符串 next() 输入并验证next(pattern) 输入数字，返回Int nextInt() nextFloat 设置分隔符（默认是回车）

New Scanner(System.in) scan.next() .hasNextInts

日期格式用正则验证，输入字符串，然后自己转换 文件中读取Scanner(new File)

11、数据操作流 格式化输出 格式化读入 属于包装类

DataOutStream(out) 写入到输出流中 写入整数writeInt() Double 写入字符串Chars(String) 写入字符Char(int)

从输入流中读取 比如文件输入流 readInt 读取为整数 读取为浮点 读取为字符

12、合并流 合并文件 SequenceInputStream(fis,fis ) 两个输入流合并

Available() read()

13、压缩流 zip jar gzip

输出流 zipoutputstream(out) 包装流 zip实体 new ZipEntry(path)

需要用File新建一个zip文件，获取文件名.getName() zipout.putNextEntry(entry) .setCommet(“”)注释

Zipout.write(char) 用一个输入流打开要压缩的文件然后循环read,写入到zipout中

如果要压缩文件夹，需要遍历各文件，每个文件一个实体

2、解压 ZipFile(f) getEntry(name) 由文件名获取实体 获取路径名getName

获取输入流 getInputStream input.read

3、zip输入流 可以方便地获取实体 包装 ZipInputStream(FileOut)

遍历getNextEntry() entry.getName()

4、回退流： 输入的可以退回去 读取一个 读取到数组 读取到任意部分数组

输入 回退一个 回退一个数组 回退 Pushback InputStream( is) 包装输入流

Read unread(byte[])

5、编码 System.getProperty(“file.encoding”)

6、对象序列化 把对象变为二进数据流？ 继承接口 Serializable

使用对象输入流 对象输出流

包装字节流 ObjectOutputStream( out ) 可以是fileoutputstream 输出到文件

.writeObject(obj) 被序列化的只有属性，所有同类对象的方法是相同的

输入流反序列化 从文件（或者其他地方）中读入对象 ObjectInputStream( fis )

readObject() 返回值是Object类型 ，**需要强制向下转型才能使用**

7、自定义扩展序列化 需要自己重写 **调用参数对象的方法读取对象的属性——如FileOutPutStream fos.write(obj.name)**

**Transient修饰符：不被序列化的属性**

序列化多个对象，需要使用对象数组

十三、集合框架 泛型 **集合中的数据都是包装类型，或者对象类型**

Listiterator 可以双向迭代的迭代器

加入对象add() 加入另一个集合addAll 清除 clear 是否存在 contains() 一组对象是否存在containsAll() 比较对象是否相等（只有包装类型和字符串） .equals(o)

哈希码 hashCode 生成迭代器对象iterator(一般是内部类) 是否为空 isEmpty 移除remove 全部移removeAll 数量 size() 转变为数组toArray

List get(i) 获取索引 （查找） indexOf() 取出子集

数组列表 add addAll add(index,value) remove(o) removeAll remove(index)

Size get(index) toArray

**foreach**

链表 LinkedList addFirst 加到尾addLast() offer()

Add 找表头peek() element() 删除表头poll() removeFirst removeLast

Set.add Treeset 会有序 add **被排序的对象必须实现Comparable接口**

**分辨同一对象需要重写HashCode equals() 方法**

2、Map

HashMap是异步。性能高线程不安全 hashtable同步 性能低

Map.put(“”，“”)

3、对象引用强度 Weak

4、集合工具类Collections

5、栈和 属性文件（是字符串键值对，方便读取配置），可以读取XML

6、一对一， 包含一个对象型属性 一对多 包含多个对象属性，或者对象数组（集合）属性 多对多，包含别人，别人也包含你

十四、枚举 命名排序常量 ，abcde 12345 也能从序号获得常量名

。values() 获取所有数， 可以用foreach获取

. name() 枚举名 .ordinal序号

十五、反射

Class类 所有类对象都是Class类的对象

对象.getClass()反射获得包类名 Class.forName(“包类名”)

获得构造器对象数组 获得类全部属性 全部方法 全部接口 类名 父类

是否数组 实例化newInstance()

2、调用类中的方法 Method类 Field类 Constructor类

使用setter getter 操作属性 操作数组

5、动态代理 类加载器

工厂模式解耦

十六、注解

重写 废弃（如果使用编译时会警告） 压制警告（警告不出现）suppresswarning

**自定义注解**

**使用反射获取注解 class.getAnnotations() isAnnotation() isAnnotationPresent()**

十七 、JDBC

十八、GUI

十九、网络编程 IP类 URL类 编码类 套接字

Echo服务器回应 accept只能一次，需要建一个循环====不断地读**输入流**

**输入流需要用缓冲包装 因为是不断输入字符串的**

**循环会阻塞主机， 需要多线程**

UDP不可靠连接，用于聊天，可以对方接受不到 包装信息

二十、新IO newio 无阻塞

Read() readLine() accept都要等待输入数据，

文件锁 内存映射

NIO的操作不用流，用缓冲区BUFFER 容量 限制 操作位置 设置操作位置 清空 重设

分配 获取当前 获取指定 写入 写入一组

状态变量：position当前位置 limit 还有多少 capacity

字节缓冲区

3、文件通道

内存映射

文件锁

4、无阻塞服务器 选择器selector

二十一、JUNIT