Java

1、 printf：格式化输出

标准形式为“%-0m.n？”格式字符

“%”：表示格式说明的占位符。

“-“：有-表示左对齐输出，如省略表示右对齐输出。

“0”：有0表示指定空位填0,如省略表示指定空位不填。

“m.n“ ： m指域宽，若数据的位数小于m，则左端补空格，若大于m，则按实际位数输出。

n指精度，用于说明输出的实型数的小数位数。未指定n时，隐含的精 度为n=6位。
“？”：c：一个字符(char)

d有符号十进制整数(int)

f单精度浮点数(默认float)、十进制记数法（%.nf 这里n表示精确到小数位后n位.十进制计数）

o:无符号八进制整数

s：对应字符串String

x：使用十六进制数字 0 f的无符号十六进制整数

2、 if-else

if后面若有两个语句，一定要{}，否则编译不成功；若else后面有或者单独的if语句后面多于一个语句，有且仅有第一个语句属于if子块，其他属于子块之外。

3、 switch的机制：

从上往下寻找符合的case，寻找到之后将全部case（包括defalt）隐身，随后往后执行语句，直至遇到break跳出循环，case：{}这个括号可加可不加，case:后面无论有没有语句都是可以的，不需要加分号；switch可针对的类：byte，short，int（long不行），Byte，Short，Integer，char，Character，String，erum类；

4、 split用法：把字符串中的字符存入字符串组；

String[] b=a.split(“”)意思是将字符串a中的每个字符都进行分割，分别存入字符串相应

的位置，后面需要用的话可以直接用b[i]表示b中的第i个字符

String a=as df gh;

String[] b=a.split(“ “);---b[1]=as,b[2]=df,b[3]=gh;

5、 b=a.substring.(i,i+k)

表示b是字符串a的下标为i到下标为i+k-1组成的新字符串

6、 String的不可变性：

1、 所有字符串方法（concat，replace，trim等等）都不会对字符串本身做出改变，而是创建一份相同的字符串再进行改变。

2、 关于新建字符串：String s1=”ww”,s2=”ww”,s3=new String(“ww”),s4=new String(“ww”),s5=s1,s6=s1+””,s7=s1+””;在这里，s1==s2,s1==s5为true，s1==s3,s3==s4,s1==s6,s6==s7都为false，首先s1，s2都是在常量池里面的，所以节约内存，会让他们地址相同，s3，s4是新建对象，每一个new会在堆中开辟一块新的内存地址；s6认为是对字符串的更改，所以会使用s1的复制品上进行更改(s6!=s1)，当然每次复制品的地址不一样（s6!=s7）；

7、 boolean：

一种数据类型，可能值只有true，false

8、 类型投影与提升

1、 可以隐式转换：byte到short，short到int，char到int，int到float，int到long，long到float，float到double，反过来转要丢失精度，需要强转

2、 变量赋值为常数时，注意范围，要在对应变量类型的范围之内

3、 类型提升：在组和运算中，java习惯见到double到float到long到int，在这组排序中，只要组合运算有一个数是上面的类型，最终会变成这个类型，不然默认是int；典例：short i=1;short j=i+1;编译错误；short j=i+（short）1;还是编译错误，i虽然是short，但是在组合运算中默认变成了int；不过short j=i；是没有问题的

4、 (+=等运算符等价形式)：T a+=x;等价于Ta=(T)(a+x);

9、 类型提升和类型装箱不可以同时进行：

long l=9;对的； Long l=9;错的； Long l=9L; Long l=(long)9;都是对的；但是注意，float要强制类型转换，float ff=5.0;错的（因为5.0默认是double），要改成5.0f；类型提升：float ff=5；对的（int自动提升为float）； Integer[] l={1,2,3,3,4}没问题（每一个int可以自动装箱）； Integer [] l=new int[]{1,2,3}编译错误，int[]不能提升为Integer[]!

10、 浮点数（单精度或者双精度）除以0

会出现infinity，整数除以零的时候是ArithemeticException:\ by 0;

11、 charAt：char a=b.charAt[i]

a是新定义的一个字符，b是一串字符串，表示a是b字符串中的第i+1位数字

12、 字符串相等：

a.equals(b) 不相等：！a.equals.(b)

13、 单字符相等：

char a==char b；char a==‘Y’（表示这个字符是Y）

14、 布尔字符：

boolean a==boolean b；boolean a==false

15、 随机抽取random：

Import java.util.Random;

Random a=new.Random();

B=a.nextInt(bound)；返回一个[0, bound)的整数

11、break跳出循环：

不加任何代码,最里层的break,只能跳出最内层的循环, 如果想要跳出最外面的,就要新增一个标记.在想要跳出循环的地方加标记（不一定是tab，其他也可）,然后break标记.同理换成continue 标记; 效果一样.注意是直接打破到目标循环为止，中间的循环因为会被打破，而Continue则是直接跳到判断条件处。 int a = 3;

tab: for (int i = 0; i < a; i++) {System.out.println("我是i...." + i);

for (int j = 0; j < a; j++) {System.out.println("我是j...." + j);if (j == 1) {break tab;}}}

16 contain函数：检测字符串中是否含有每段子字符串；

String num = "WKCON190400111";

if (num.contains("CON")) {System.out.println(1);}

else {System.out.println(2);

17、 打印转义字符“\“:

在任何格式的打印当中都要用[\,e.g.Systerm.out.print(“yes\\”)输出yes\](file://,e.g.Systerm.out.print();

18、 在printf中打印%，类似的，要用%%

19、 在打印中，有引号的表示是直接打出内部的字符串（一般而言），没有引号的，是变量

20、 nextLine和next的输入区别：

Java中Scanner类中的方法next()和nextLine()都是吸取输入台输入的字符，区别：next()不会吸取字符前/后的空格/Tab键，只吸取字符，开始吸取字符(字符前后不算)直到遇到空格/Tab键/回车截止吸取； nextLine()吸取字符前后的空格/Tab键，回车键截止。（即nextline字符串里面可能有空格，而next不可能有）

21、 重要的基本数据类型转换：

\1. 由 基本数据型态转换成 String
String 类别中已经提供了将基本数据型态转换成 String 的 static 方法
也就是 String.valueOf() 这个参数多载的方法
有下列几种
String.valueOf(boolean b) : 将 boolean 变量 b 转换成字符串
String.valueOf(char c) : 将 char 变量 c 转换成字符串

以下：a为char数组名称
String.valueOf(char[] a) : 将 char 数组a转换成字符串
String.valueOf(char[] a, int i, int n) :
将 char 数组a 中 由 a[i] 开始取n 个元素 转换成字符串
String.valueOf(double d) : 将 double 变量 d 转换成字符串
String.valueOf(float f) : 将 float 变量 f 转换成字符串
String.valueOf(int i) : 将 int 变量 i 转换成字符串
String.valueOf(long l) : 将 long 变量 l 转换成字符串
String.valueOf(Object obj) : 将 obj 对象转换成 字符串, 等于 obj.toString()
用法如: int i = 10; String str = String.valueOf(i); 这时候 str 就会是 "10"
\2. 由 String 转换成 数字的基本数据型态
要将 String 转换成基本数据型态转
大多需要使用基本数据型态的包装类别
比如说 String 转换成 byte
可以使用 Byte.parseByte(String s)
这一类的方法如果无法将 s 分析 则会丢出 NumberFormatException
byte :
Byte.parseByte(String s) : 将 s 转换成 byte
Byte.parseByte(String s, int radix) : 以 radix 为基底 将 s 转换为 byte
比如说 Byte.parseByte("11", 16) 会得到 17
double :
Double.parseDouble(String s) : 将 s 转换成 double
float :
Double.parseFloat(String s) : 将 s 转换成 float
int :
Integer.parseInt(String s) : 将 s 转换成 int
long :
Long.parseLong(String s)

```
22、System.out.println(""+v+v);输出两个v的值；
23、System.out.println(v+“v”);输出v的值和一个字母v
```

24、 Math.round(a):

对a取整，可以赋值于整型或浮点型（储存为a.0）

25、 浮点型储存规律：

尽量少保留的记录小数的信息；如float a=5；实际上储存为5.0；float a=510/100,则a==5.1。

26、 声明数组需要注意的地方：两种方式：

静态初始化：有长度有元素

int[]a=new int[]{1,2,3,4};对的—既然在一起，

简写：int[]a={1,2,3,4}对的，

如果不写在一起：int[]a;a={1,2,3}；错的

简写只有与声明在同一语句的时候才是对的，其他情况不管原本是null还是什么，都是错的

int[]a=new int[4]{1,2,3,4};多余了，错的

int[]a;a=new int[]{1,2,3}； 对的（想想如果中间隔一百行，为表明数组是新建的，所以这样写）

动态初始化：有长度没元素（记住，说了长度就不能提元素），没长度没元素必错

int []a=new int[3];对的；

int[]a;a=new int[3];对的；

int[]a=new int[3];a={1,2,3}；错的；

27、 数组的赋值"="： a=b表示使a与b都指向b原本指向的东西，

int[]a=new int[3],b=new int[2];b=a;

（它本身的长度不等也可以这样指向赋值，都变为长度为3的了；）

for(int i=0;i<3;i++){b[i]=1;System.out.print（a[i]）;}输出结果为111

28、 数组元素使用中的enhance for循环：

Int []b={1,2,3,4}For(int a:b){语句}

等价于Int []b={1,2,3,4}For(int I;i<b.length;i++){a=b[i];语句}

29、 二维数组a[i][j]可以理解为数组的叠加，a[i]中存储的也是一个子数组，：

示例一：

Int[][] a=new int[3][3];int c=0;

For(int b:a){b[c]=0;c++;}省略了b=a[1]、a[2]、a[3],b是数组！

所以a为：

1 0 0

0 1 0

0 0 1

示例二：

Int[][] a=new int[3][3];int c=0;

For(i=0;i<3;i++){

For(int b:a[i]){b=0;c++;}}

a为

0 0 0

0 0 0

0 0 0

30、 Math方法：提供数学函数与特殊值

1、 指对数函数方法：Math.pow(a,b)返回a的b次幂，Math.exp(a);Math.log(x)返回lnx；Math.log10(x)返回10为底的对数;Math.sqrt(a)返回a的平方根；

2、 a=Math.random();产生[0，1）随机的双精度数并赋值给a；

若是想在[0,i](i为整数)中抽取一个整数：int a=（int）（Math.random()*(i+1）);

3、 在弧度制下：Math.sin(a)，Math.cos(a),Math.tan(a)返回正弦余弦正切值；
Math.asin(a),Math.acos(a),Math.atan(a)返回相应的反三角函数值；

4、 Math.toradians(d),Math.todegrees(r),角度与弧度进行转换

5、 取大，取小，取绝对值方法：Math.max(a,b);Math.min(a,b)只能是两个数;Math.abs(a);

6、 舍入方法：Math.ceil(a)上取整；Math.floor(a)下取整;Math.rint(a)四舍六入五取偶；
Math.round(a)a+0.5后下取整，即大于零时四舍五入，小于零时五舍六入；

注意：a=（int）a是直接抹去小数部分的意思；

7、 特殊值：Math.PI,Math.E

31 If语句加boolean赋值可以直接写成boolean赋值

if(条件句)even=true；

可写成Boolean even=（条件句）；

32、 三元操作符：

形式 （Boolean expression）？ e1：e2对的话结果就是e1，不对的话结果就是e2，这个结果可以是数字，字符，字符串都可以！可以直接赋值，可以直接放入打印的括号中！

33、 在打印的双引号中，\表示单斜杠，\”表示双引号

34、 Int Y=1,z=Y++*++Y;

*理清楚过程，首先是传值和自增自减，然后是赋值（“=”），首先，前缀++，--是先自己变化再传值，后缀++，--是先传值再自己变化：Y=1，Y++，所以是现在这个Y++位置上传入一个1，然后Y++变成了2；++Y，所以是先++Y变成了3，然后再传到这个位置，最后是赋值，所以z=（1*3）=3；

35、 使用以下boolean方法（即返回值是true活false）对字符a进行检测：

Character.isDigit(a)判断是不是数字；Character.isLetter（a）判断是不是字母；Character.isLetterOrDigit（a）；Character.isLowerCase（a）判断是不是小写；Character.isUpperCase（a）判断是不是大写；

而以下则是返回字符的大写或小写：Character.toUpperCase（a）变大；Character.toLowerCase（a）变小；

36、 字符串对象的简单方法

1、 s=s1+s2+。。。；表示s是s1和s2等字符串的连接；

2、 s.toUpperCase（）全部变为大写；s.toLowerCase（）全部变为小写；

3、 boolean值返回的：a.startwith(“abc”)判断前缀；a.endwith(“abc”)判断后缀；a.contains(“abc”)判断包含；

4、 截取子字符串：s.substring.(a,a+b)返回s中从a+1到a+b-1的子串

5、 寻找子串出现的序号：

s.indexOf（a，k）返回首次出现a的下标，a可以是字符或子字符串（内部第一个字符在原字符串中的坐标），k可有可无，若有则表示从k+1个字符开始往后查找；

s.lastIndexOf(a,k)返回末次出现a的下标，a可以是字符或子字符串（内部第一个字符在原字符串中的坐标），k可有可无，若有则表示从k+1个字符开始往前查找；

如果没有，则返回值-1；

37、 nextLine的读取特点为：可以为空，只有回车表示读取结束。

几个特殊的例子：如果前面先用next读入了某个数，表示完成时用了空格：

如：22 333，则nextLine读取为“ 333”；说明nextLine读取是从上一次读取结束就立刻开始的。

若表示完成时用了回车，则nextLine读取为“”，即空字符

38、 数组复制：

System.arraycopy（a，k，b，u，l）;表示将a数组从第k位开始复制到b从u开始的位置，复制长度为l，注意是复制，不是让他们共同指向；

39、 java.util.Arrays类中的一些方法（静态方法）

1、 排序方法（仅针对于int[],char[]）：

java.util.Arrays.sort(a,k,k+u); java.util.Arrays.parallelsort(a,k,k+u);对a数组的k到k+u进行升序排序，当然如果没有后面两个参数，那就是对整个数组进行排序；

2、 查找方法

java.util.Arrays.binarysearch(a,keywords);针对升序的int[],char[]数组a进行的查找，返回keywords的位置，若找不到返回一个负数；

3、 审等方法

java.util.Arrays.equals(list1,list2);list可以是任意数组；此方法检查严格相等，返回布尔值true、false；

4、 填充(更改)方法

java.util.Arrays.fill(list，8)；把数组list全部字符改编为8（其实只要同一个类型都可以）

java.util.Arrays.fill(list，1，5，8) 把list[1]-list[5]字符改编为8

5、 转s方法

java.util.Arrays.toString（list）；表示将list数组的元素加上一个头“[“，一个尾”]”并在任意两个元素中间加上逗号，返回该字符串；

40、 一些java中API的类（引用类型）

1、 java.util.Random类：

1、创建对象：new Random（）；.....以当前时间作为种子创建Random对象；

New Random（long seed）；以一个long型种子创建Random对象；如果两个Random对象具有相同的种子，他们会产生相同的序列。

2、方法：nextInt（）；返回一个随机int；

nextInt（int n）；从[0,n）返回一个int；

nextLong（）；返回一个随机long；

nextDouble（）；返回[0,1）中一个随机Double；

nextFloat（）；返回[0,1.0F)中一个随机float；

nextBoolean（）；返回一个随机boolean；

2、 javafx.geometry.Point2D类

1、 创建对象：Point2D（double x，double y）用x，y坐标创建一个Point2D对象。

2、 方法调用：

distance（double x，double y）返回一个double值表示对象和(x,y)的距离。

distance（Point2D p2）返回一个double表示原对象和新对象p2的距离。

getx（）；返回double表示该点的x坐标。

gety（）；返回double表示该点的y坐标。

midpoint（Point2D p2）；返回一个Point2D对象，表示原对象和p2的中点；

toString（）；返回Point2D对象的字符串形式；

打印p1的输出结果是(举例)：”Point2D [x =2.0, y = 3.0]”

41、 引用类型中的对象的创建：

1、用new；2、直接指向原有对象的指向（这种可以避免立刻向构造方法传参）；ex：int[]a=new int[]{1,2};int[]b=a;

42、 String.format(“%x”,a);

将十进制的a以十六进制返回成字符串；

Remark：针对的是正数a；

43、 Idea中编译和运行：

1.在idea的terminal中或cmd中，首先要“cd ”加上这个.java文件的上一层目录的地址,如：

1、 C:\Users\86180\IdeaProjects\Demo\src\lab7> javac Food.java（成功编译）

2、 C:\Users\86180\IdeaProjects\Demo\src> javac lab7.Food.java

javac: 找不到文件: lab7.Food.java（在src内目录中没有找到Food类）

随后运行，首先要确保有main方法，然后注意，此时这个Food的文件的全名变成了包含它所有包（不是目录，所以从src往下的第一个包开始）次第展开，用.连接最后连上原名。比如说这里，运行命令：java lab7.Food 注意cd到src中输入命令，否则依然无法找到主类。

FileReader与FileWriter的注意点：

1、首先后面读取的文件，双引号中要输入地址从C:\开始知道所要读和写的文件，还有另外一种方式，在run-edit configuration-working directory中修改路径到这个Java文件的上一级目录。

第二种方式注意：若读入写出的文件在与java文件同级的其它包中（如lab7），有两个办法1、双引号中文件名前加上lab7\2、将java文件移入lab7中，再次修改路径

2、要有异常处理try{这里读写} catch（IOException e）{如果读不到或写不到就运行这个}

3、要记得关闭变量名.close;否则失败

4、变量名.read()方法返回int，就是读的字符转的，读完之后会跳转到下一位，读完所有字符后继续读取时返回-1；

5、.write()方法直接写在文件中

38、BigInterger与BigDemical类（实现任意大整数，或者任意精度）注意对象不是基础数据类型。

BigInterger类：创建对象BigInteger a=new BigInteger（“任意数”）或（int+“”）

方法：a.add(b),a.substract(b),a.multiply(b),a.divide(b),a.remainder(b)来实现加减乘整除取余操作,a,b都是该类的对象。

BigDemical类：创建对象BigDemical a=new BigDemical(String)或（Double）但是String比double更准确，因为double是近似的；

方法和上面类似：注意divide方法重载：divide（b，scale，roundingmode），scale是小数位数，roundingmode是舍入方式

44、 erum枚举类

1、 特点：既是一个类，有类的member（构造函数，字段，方法）；又是枚举，对象只有确定的有限个，这决定了构造函数必须是private，当然这个是redundent的；

2、 所有对象之间用“，”隔开，每个对象后面可以加上构造函数并进行传参

3、 这个类不能用new来创建新对象，因该直接用类名+对象的变量名+“=”+“某一个对象”来进行赋值；

4、 去使用对象的元素：假设枚举类的类名为a；

1、 用a.values()返回a类的数组，表示a中所有对象排列后形成的数组；

2、 用Erumset.range(a.对象x，a.对象y)生成从对象x到对象y的一个数组

3、 Erum对象的变量名.toString返回对象的名字，这仅仅是对于erum类才成立的，对于一般的object.toString返回的是对象的类名+@+哈希值；

4、 .name()方法返回枚举类中调用方法的对象的名字；

5、 .ordinary()方法返回枚举类中的序号；

5、 在枚举类中，我们时常会使用final的字段，注意若这个字段描述的是每一个对象自己的性质，这个final字段显然不是static的，所以我们要使用构造函数定义final字段的方法。

6、 枚举类的审等方法，直接==即可，因为对象只有有限个

45、 排序数组：

针对问题：

1、对于数组内为基础数据类型，如果不想改变其原有排序，或是有几种可能要使用的顺序，那么会使用排序数组，用来储存调用数组内容的几种顺序；（当然你也可以改变数组元素的顺序（x=a[i];a[i]=a[j];a[j]=x）

2、对于数组类型为引用数据类型，数组之间元素不方便轻易改变，因为是指针，而且new对象也不一定能完全复制对象属性；此时应对该种问题必须要使用排序数组了，因为引用类型数组按其属性进行排序是非常常见的问题；

具体代码：

```
//建立初始的排序数组

//设引用类型的数组是object1[];

Int [] order=new int [object.length];

For(int i=0;i<object1.length;i++){order[i]=i;}

//针对object1更改排序数组，以X性质从小到大为例；

For(int i=0;i<object1.length-1;i++){

  For(int j=i+1;j<object1.length;j++){

    If(object1[order1[i]].getX()>object1[order1[j]].getX()){

      //交换顺序

      Int a=order1[i];order1[i]=order1[j];order1[j]=a;

}}}
```

但是该代码有一个小问题，它并没有处理相等的情况；如果要求相等的时候保持默认排序；那么if的条件要做一点小的改动：

If(object1[order1[i]].getX()>object1[order1[j]].getX()||object1[order1[i]].getX()=object1[order1[j]].getX()&&order1[i]>order1[j])

//若他们相等，那么就应该按出现的先后顺序进行的赋予：即两个相同性质的object1，那应该是先出现order1[i]小的那个,再出现order1[j]大的那个，所以如果order1[i]>order1[j], 那么就要调换顺序。

46、 继承与多态

1、 方法重写：签名和参数列表相同，返回值必须是父类方法返回值的子类

2、 搜索方法的机制：1、优先找参数类型相同的，再找可以隐式转换的；2、在1、的基础上，先在子类寻找，再去父类寻找。

3、 有override的标注，如果并没有实现重写，会标红；

4、 多态：父类变量可以指向子类对象。几种具体表现：1：new一个对象时；2、方法传参可以传参数的子类；3、Instance of 的目的，是判断对象具体形式，所以只有后面的类的对象或子类的对象才能返回true。

5、 instanceof A，验证是否是A或者其子类（子类可以是父类的对象）

6、 如果要调用子类有，父类没有的方法，那么要向下投影，这是一种显式转型，这与向上投影（隐式转型，例如多态初始化）不同，需要前面加上括号中有子类实现，此时常用instanceof；

7、 对象成员访问操作符（.）优先于向下投影：（B）A.methodName可能发生错误，此时要（(B)A）.methodName;

8、 Protected可以不同包中的子类访问，也可以同一个包内访问

9、 子类重写父类方法不可以降低其可访问性（ex：知道人可以讲话，却不能知道人的实体（如学生）能不能讲话，这是不合理的；同时父类既然想让子类继承，那么肯定也想让“孙类“继承）

10、 子类重写父类方法不可以改变参数列表，即使是变成参数的子类也不行

11、 向下转型要保证括号内的类型是变量实际指向类型的子类

12、 多态中调用实例变量和静态变量时，都是变量声明类型的变量

13、 多态中方法调用时，编译器会检查声明变量类型内有没有这个方法（如果没有，编译器会认为你不能保证实际类型一定有该方法，视为编译错误）

14、 转型合法（（circle）Obj1）：检验Obj1的实际类型是否可以转型（可以无条件向上转型，向下转型可能运行时错误）；赋值合法（obj1=obj2）：obj2的声明类型如果高于obj1的声明类型一定会引起编译错误，只要obj1声明类型高于obj2的声明类型，一定是可以的；

47、 静态绑定和动态绑定

1、 绑定是什么：将方法的调用和方法的主体（所在类）关联起来。

2、 静态绑定：在编译期间，编译器已经知道调用方法的类是什么，一个明显的特征是调用方法于变量的声明类型有关，java中只有final方法（不允许被重写），static（不允许被重写），private，constructor（不能被继承）是静态绑定

因为觉得很完善，final方法不允许被重写，final类不能被继承，调用方法时关注的是声明变量类型；在多态中，调用final方法：与一般多态类似，先从调用者往下寻找，知道找到实际指向类型的方法（途遇到了final之后子类不能重写，但是还是可以继承的）；

static方法：1用类名.方法：可以继承，这意味着用子类类名.父类静态方法是可以的，子类同名同参的方法是对父类的覆盖，并非重写（不能标注@Override），且子类中不能用super.父类的静态方法名实现调用（不过还是可以用父类类名静态方法调用的），2如果用一种不好的习惯：即用对象名字.静态方法的时候，调用取决于变量的声明类型；

private：调用private方法，只能是同一个类之中的（想想为什么，结合权限继承只升不降，其它类中不能调用），本身具有final特性，只不过子类可以重写，但是直接与声明类型直接相关；子类不能继承父类的private方法，但是子类实际变量能否使用private取决于1、调用的时候和private方法在不在同一个类中，2、编译器认为这个变量是父类（所以说是静态绑定），就像如果是父类方法中，利用多态，使父类变量指向子类，用变量名.private方法是可以的，但是在子类中这样做不行（不在同一个类，既是在子类也有该方法签名的方法也不行，静态绑定！！），在父类中直接构建子类对象（不使用多态）也不行；

constructor与类名相同（与类直接关联），而同类不同constructor的参数不同，所以调用时编译器也是直接绑定了类和方法

3、 动态绑定：除了以上四个其余都是动态绑定，即运行时才知道要调用哪个方法；工作机制：jvm提取对象实际类型（注意与声明类型的区别）开始往上的方法表，搜索签名，找到后调用并执行：

两个经典的例子：继承：调用方法时是不知道调用到哪个类的方法的，只能运行时不断往上查找；多态，编译时变量的类型是声明类型，直到运行时才知道该变量是指向某一个对象的，随后调用实际对象类型的方法列表；

48、 抽象类和接口：

1、 类的构造器不能是static的，因为构造器是与对象直接联系的；类的构造器不能为final，因为final的目的是为了避免被子类继承，但是构造器是不会被继承的；构造器不能用abstract修饰，因为abstract目的是要求子类重写；以上这三种情况，编译器会ce来提醒你。

2、 一个类继承另一个类（包括抽象类也可以继承实体类），若没有构造器，会隐性调用无参构造器，且构造器内会隐性调用父类无参构造器；若有构造器但是没有调用super（）或者this（），会自动调用super中无参构造器。所以在这种情况下，父类无参构造器为private，这个时候就会报错；同时super和this只能有一个，而且都必须写在第一行，否则会ce；

3、 Inner class：定义在一个类的内部；存在原因：只有某一个类需要用到，没有必要在外面定义；特点：inner class与outer class的private的字段方法可以直接访问；一个特殊的inner class：anonymous class。匿名类的作用：可以化简一般的innerclass，因为不需要新建类名，目的仅仅是使用接口。代码使用方式如下：

New InterfaceName（）{..};代表了使用该接口的一个类（使接口这种抽象的东西具体化了），InterfaceName是接口的名称，表示使用的接口，Braket内的内容和原本类中的内容完全一致。

4、 接口：特点1：字段是隐性public（protected不能被使用者调用）；static的，因为如果一个类使用有相同变量名的不同接口，会有歧义，解决方法为使用类名加以区分，这要求static；final的，接口是一份合同，不能因为一个使用者更改而导致其他使用者受影响；基于此，属性不能改变与不能被实例化，决定了不能有constructor（ce）； 字段与方法定义为除public外，会ce；使用者在没有歧义的情况下，可以直接使用变量名对接口中字段进行调用；

特点2：不继承于object类，只能继承于接口；

特点3：继承只能针对唯一父类，但是使用者可以以使用很多个接口；

特点4：三种方法，defalt方法使用者的对象可以直接调用，可以被重写（覆盖），abstract方法必须被重写（否则ce）,static方法要用接口名加方法调用，这里与父类子类不同的地方在于，子类类名加父类静态方法名可以调用这个方法（如果没有被隐藏），但是接口不能用使用子类类名加接口中静态方法名进行调用（会ce）

5、 抽象类的字段和方法可以是普通的（对static，final，和权限都没有要求），抽象方法对于非抽象子类必须重写，否则ce

6、 Final的特点：final class只能是子类，不能被继承，类中的方法是隐性final，但是字段不一定；final method：子类可以调用，但是不能重写，会ce（不允许被覆盖）；这一点与static有点不一样：父类的static的字段和方法是可以通过子类类名（当然父类类名也可以）进行调用的

7、 抽象类的一点理解：1、抽象和具体是根据实际情况而定的，所以java并没有限制具体类只能是抽象类的子类，比如说object是实体类，但是设计关系时，一般实体类都作为抽象类的子类。2、子类是抽象类可以不重写父类抽象方法，但是孙类还是要重写父类方法的。

49、 泛型

1、 泛型：类和方法不仅实际数据由调用者决定，连数据类型也有外部决定

2、 泛型类：类的完整名字为名字加上<E>，意义是可以存储任意类型的数据，内部实际帮助你存储数据时就可以用泛型E指代，如常用ArrayList<E> arr=new ArrayList<>();常用的数组E[] a=new E[3];常用的与类有关的变量E b；泛型类中可以有一般的泛型方法，使用规则和第二点相同；也可以有针对与类名中相同泛型的方法，此时就不需要再返回值之前表明泛型尖括号，在参数列表和方法体中调用同一个泛型符号即可；

3、 泛型方法：需要在返回值之前限制方法的参数与返回值，用<T extends a>来表示，其中a是T的类上界，表示T必须继承于a类，或使用a接口；

4、 ？：表示不确定的泛型，由于我们传参需要传入参数的对应类型或其子类，但是对于序列（如数组，链表等），List<String>显然不是List<object>的子类，（判断方法，子类必须具有父类所有的成员，比如说后者可以添加new Integer对象，但是前者显然不行），这个时候我们可以使用List<?>,代替未知类型的数据。

5、 链表ArrayList的一些方法：针对整数，浮点数，字符串或者任何使用Comparable接口的类的链表，可以调用Collections.sort排序（用Compare To比较，小的排在前面），当然类似原理有Collections.max, Collections.min返回最大最小对象。

50、 异常处理

1、 throw和throws：方法体中抛出异常使用throw，这是一定发生的；方法名中提示可能抛出异常用throws，这是可能发生的；方法体中throw不会受方法头中throws异常种类的影响，方法头中的throws主要用于提示，以及监测方法体中可能出现该种形式的异常。

2、 必须要进行异常处理才能继续运行程序，异常处理的方式有两种，一种是throws/throw，将方法中的该种异常延栈抛回到方法调用栈中，等待处理；try-catch用于捕获某种特定的异常，并进行处理。如果最终并没有处理，main方法的异常会自动抛回到jvm，随后运行终止并反馈异常信息；注意一遇到异常后，剩下的语句（无论是方法体中，还是try中）都不再执行，如果没有抛出或捕获，则会终止程序