简答题题库

1：什么是注释?注释分为几种?注释有什么作用?
注释就是对代码的解释说明性文字，分为三种：单行注释、多行注释、文档注释。注释可以对程序进行说明，给人看，还可以对程序进行排错。

2：什么是关键字?有什么特点?
关键字就是被 java 语言赋予了特殊含义的单词。特点就是所有的关键字都是小写。

3：什么是标识符?由哪些部分组成?常见的命名规则有哪些?
标识符就是给类、接口、方法、变量名起的名字。常见的规则有：有数字、字母、_以及$符号组成，不能以数字开头，不能是 java 中的关键字。

4：什么是常量?常量的分类?字面值常量表现形式有哪些?
在程序的运行过程中其值不发生改变的量。

字面值常量： 1 14

自定义常量： int MAX = 100;

5：什么是变量?变量的定义格式?要使用变量需要注意什么?
在程序运行过程中，其值是可以在某个范围内发生改变的量。

变量其实就是内存中一小块区域。

由 3 部分组成：

1，数据类型：限定变量的取值

2，变量名：方便使用。

3，值： 如果没有值就没有意义。

6：Java 中的数据类型分几类?基本数据类型有哪些?
基本类型：4 类 8 种。基本数据类型有：

整数(byte,short,int,long)、浮点数(float,double)、字符(char)、布尔(boolean)。

7：算术运算有哪些，各自的作用?
有：+，-，*，/，%，++，--。

+:正号,加法,字符串连接符

%:取得余数

++,--:

单独使用：在操作数据的前后效果一致。

参与操作使用：

在操作数据的前面，是先++或者--，然后参与操作。

在操作数据的后面，是先参与操作，后++或者--。

8：+运算符需要注意的问题?

当把+号作为加法和字符串连接符一起使用的时候，注意把运算的地方（）起来。

9：赋值运算符常见的有哪些，各自的作用?

=:把右边的内容赋值给左边

+=,-=,*=,/=,%=：把左边的和右边的操作后赋值给左边。

注意：

+=隐含了强制类型转换。

x+=y;等价与：x = (x 的数据类型)(x + y);

10：逻辑运算符有哪些都有什么作用？

&:有 false 则 false

|:有 true 则 true

^:相同为 false，不同为 true。通过情侣关系来理解。

!:非 false 则 true,非 true 则 false。偶数次不改变以前的真假。

&&:和&的结果一样,但是具有短路效果。左边有 false,右边不执行。

||:和|的结果一样,但是具有短路效果。左边有 true,右边不执行。

11：位移运算符有什么不同？

要想操作，就必须把所有的数据转换为二进制。然后操作。

<<:把数据向左移动，后边补 0。其实相当与乘以 2 的移动位数次幂。

>>:把数据向右移动，左边补是最高位的值。其实相当与除以 2 的移动位数次幂。

>>>:把数据向右移动，左边补 0。其实相当于除以 2 的移动位数次幂。

12：什么是函数重载？

函数名相同，参数列表不同，跟返回值不关，就是函数重载。

13：类是什么?类的组成是哪些?
类是抽取了同类对象的共同属性和行为形成的对象或实体的"模版".

类是由成员变量,成员方法,构造函数组成

14：对象是什么?什么时候需要使用对象?
对象是现实世界中实体的描述,是其自身所具有的状态特征及可以对这些状态施加的操作结合在一起所

构成的独立实体.

需要描绘具体事物的时候要使用对象.

15：封装是什么?自己概述
隐藏对象的属性和具体的实现细节,仅对外提供公共的访问方式.

类、方法其实也是封装的体现。

16：继承是什么?自己概述
概念:把多个类中相同的内容抽取出来,单独定义到一个类(父类)中,再定义其他类(子类)的时候,继承父类即可.

好处:1.提高了代码的复用性,降低了代码的重复率.

2.让类与类之间产生了关系,是多态的前提.

17：匿名对象是什么?应用场景是什么?
匿名对象就是没有名字的对象，由于没有指向，所以效率较高一些。

应用场景：A：调用方法。但是仅仅只限调用一次而已。

B：匿名对象用作实际参数传递。

18：构造方法的作用是什么?构造方法的特点是什么?构造方法的注意事项是什么?
构造方法的作用是初始化数据。

特点是：名称和类名一致，并且没有返回值类型的修饰和返回值。

注意：如果我们没有给构造方法，系统将给出一个无参构造方法。如果我们给出了构造方法，系统将不再提供构造方法。

19：给成员变量赋值有几种方式?
1，通过 set/get 方法。

2，通过带参构造。

20：方法重写和方法重载的区别?重载可以改变返回值类型吗?（*面试题）

方法重写：

子类中出现和父类中一模一样的方法声明的方法。

方法重载：

同一个类中，出现方法名相同，参数列表不同的方法。

跟返回值无关。

重载可以改变返回值类型，因为他跟返回值无关。

21：static 关键字是什么?有什么特点?什么时候用呢?

static 关键字：是静态的意思，可以修饰类中成员变量和成员方法。

静态的特点：随着类的加载而加载、优先与对象而存在、被所有对象所共享、可以通过类名.静态的内容调用。

22：this 和 super 的区别,以及应用场景? (*面试题)

this:当前类的引用对象。谁调用代表谁。

super:父类的存储空间标识。可以理解为父类对象，谁调用代表谁父亲。

应用场景：

A:成员变量

this.成员变量 本类的成员变量

super.成员变量　父类的成员变量

B:构造方法

this(...) 本类的构造方法

super(...)　父类的构造方法

C:成员方法

this.成员方法()　本类的成员方法

super.成员方法() 父类的成员方法

23：代码块是什么?代码块的分类有哪些及每种代码块的作用是什么?

用{}括起来的代码，就叫代码快。

分为：

A：局部代码快：就是在方法中用{}括起来的代码。作用是限定变量的生命周期，提高效率。

B：构造代码快：在类中，方法外。用{}括起来的代码。作用是把所有构造方法中相同的内容抽取出来，

定义到构造代码块中，将来在调用构造方法的时候，会去自动调用构造代码块。构造代码快优先于构造方法。

C：静态代码快：在类中、方法外。用{}括起来的代码。只不过加了 staric 修饰。

作用是：在整个系统中，只加载一次的代码。一般做整个系统的初始化。

24：一个类的实例化过程有哪些步骤？（*面试题）

Student s = new Student(); 在内存中到底执行了哪些步骤。

1，加载 Sutdent.class 文件进内存(类加载器)

2，在栈内存为 s 变量申请一个空间

3，在堆内存为 Student 对象申请空间

4，对类中的成员变量进行默认初始化

5，对类中的成员变量进行显示初始化

6，有构造代码块就先执行构造代码块，如果没有，则省略

7，执行构造方法，通过构造方法对对对象数据进行初始化

8，堆内存中的数据初始化完毕，把内存值复制给 s 变量

26：继承是什么?继承的好处是什么?Java 中的继承特点是什么?

继承就是：把多个类中相同的内容提取出来，定义到一个类中。然后让这多个类和这个类产生一个关系，这多个类就具备该类的数据了。这种关系叫：继承。

继承的好处是：提高代码的复用性，让类与类之间产生了一个关系，是多态的前提。

继承的特点是：Java 中类只能够单继承，不能多继承，但是可以多层继承。

27：方法重写和重载有什么区别?重载可以改变返回值类型吗？（*面试题）

方法重写：子类中出现和父类一模一样的方法声明的方法。

方法重载：同一个类中，出现的方法名相同，参数列表不同的方法。

重载可以改变返回值类性，因为跟返回值无关。

28：子父类中构造方法的执行有什么特点?为什么要这样?

子类构造方法的执行，首先会去执行父类的构造方法。

因为子类中可能直接访问了父类的数据，父类的数据要优先于子类的数据进行初始化。

29：静态代码块，构造代码块，构造方法的执行顺序是什么?

静态代码块 -- 构造代码块 -- 构造方法

30：final 关键字是什么，可以修饰什么，有什么特点?

final 关键字：是最终意思，可以修饰类、方法、变量。

修饰类：类不可以被继承。

修饰方法：方法不可别重写

修饰变量：变量为常量。

31：多态是什么，前提是什么?

多态：对象在不同时刻表现出来的多种状态。是一种编译时期状态和运行时期状态不一致的现象。

成员变量：编译看左边，运行看左边。

成员方法：编译看左边，运行看右边。因为，普通成员方法可以重写，变量不可以。

静态方法：编译看左边，运行看左边。

前提：类与类之间要有继承关系。要有方法的重写。父类引用指向子类对象。

32：多态的好处及弊端？如何解决多态的弊端。

好处：提高了程序的可维护性(前提要有继承保证)，和扩展性。

弊端：不能使用子类的特有功能。

33：什么是抽象类？抽象类的特点和好处是什么?

相同的方法，有相同方法的声明，但是方法体不一样，只抽取方法声明的方法，叫做抽象方法，有抽象方法的类，叫做抽象类。

特点：

A：类或者方法必须用 abstract 修饰。

B：具体类在继承抽象类的时候，要么本身也是抽象类，要么实现抽象类中的所有抽象方法。

C：抽象类不能被实例化。要想使用，必须按照多态的方式使用。

D：成员特点：

a：成员变量

可以是变量，也可以是常量。

b：构造方法

有构造方法，但是不能实例化。

用于子类访问父类数据的初始化。

c：成员方法

可以有抽象方法也可以有非抽象方法。

好处：

A：限定子类必须实现某些功能。

B：提高代码的复用性。

抽象类的几个问题：

A:抽象类不能实例化，构造方法有什么用。

用于子类访问父类数据的初始化。

B:抽象类没有抽象方法，有什么意义？

限制创建对象。

C:abstract 不能和哪些关键字共存?

final：冲突

private：冲突

static：无意义

34：什么是接口？接口的特点?

如果一个抽象类中的方法全部是抽象方法，那么 java 就针对这种类型的抽象类，给出了一个更抽象的表达式：接口。

特点：

A：所有的方法，都是抽象方法。

B：类实现接口

要么本身是抽象类，要么重写接口中的抽象方法。

C：接口不能被实例化，要想使用，用多态。

D：成员特点

a:成员变量

只有常量，并且是静态常量。默认修饰符：publci static final

b:构造方法

没有构造方法，子类数据的初始化默认走的是 Ojbect 类的构造方法

c:成员方法

全部是抽象的，有默认修饰符：public abstract

35：抽象类和接口的区别？（*面试题）

A：成员区别

抽象类：

成员变量:可以是变量，也可以是常量。

构造方法:有。

成员方法:有，可以是抽象的，也可以是非抽象的。

接口：

成员变量:只能是常量。默认修饰符：publci static final

构造方法:没有，子类数据的初始化默认走的是 Ojbect 类的构造方法。

成员方法:只能是抽象的，默认修饰符是:publci abstract

B：类与接口的关系区别

类与类：

继承关系，单继承。

类与接口：

实现关系，单实现、多实现。

接口与接口：

继承关系，单继承、多继承。

C：设计理念的区别

抽象类被继承体现的是：is a 的关系。抽象类中一般定义的是整个继承结构的共性功能。

接口被实现体现的是：like a 的关系。接口中一般定义的是整个继承结构的扩展功能。

36：什么是内部类？有什么特点？

把类定义在其他类的内部，就被称为内部类。

内部类的访问特点：

A：内部类可以直接访问外部类的成员，包括私有。

B：外部类要访问内部类的成员，必须创建对象。

37:为什么内部类访问局部变量必须加 final 修饰？(*面试题)

A：防止在使用后数据发生改变。

B：延长变量的生命周期。

38：什么是匿名内部类？本质是什么？

匿名内部类就是没有名字的内部类。

格式：

```
new 类名或者接口名() {
    重写方法;
};
```

本质：是继承类或者实现接口的子类匿名对象。

39:==和 equals()的区别？(*面试题)

A:==

a:基本类型 比较的是基本类型的值

b:引用类型 比较的是引用类型的地址值

B:equals()

只能比较引用类型。

默认比较地址值。

40:什么是字符串？字符串的特点是什么？

字符串：由多个字符组成的一串数据。

特点：一旦被赋值就不能被改变。（*面试题）

注意：这里指的是字符串的内容不能发生改变。而字符串的引用是可以再次赋值的。

41：String s1 = new String("hello");和 String s2 = "hello";的区别？（*面试题）

有区别，区别是：前者创建了两个对象，后者创建了一个对象。

42：String、StringBuffer、StringBuilder 的区别？（*面试题）

String:字符长度是固定的。

StringBuffer/StringBuilder:字符长度是可变的。

StringBuffer:安全的，但效率较低一些。

StringBuilder:不安全的，但效率较高一些。

StringBuffer 和 StringBuilder 兼容。

43：什么是基本数据包装类？

为了对基本类型的数据进行更多的操作的，java 就针对每种基本类型的数据提供了对应的包装类类型。

对应的类型

byte	Byte
short	Short
int	Integer
long	Long
float	Float
double	Double
char	Character
boolean	Boolean

44：什么是正则表达式？

正则表达式就是符合一定规则的字符串。

45：集合和数组有什么区别？（*面试题）

集合：

长度可以发生改变。

只能存储对象类型，引用类型。

可以存储任意类型的对象。

数组：

长度固定。

可以存储基本类型，也可以存储对象类型。

只能存储同一种类型的元素。

46：集合有多少种？ 各自的特点是什么？

Collection

|--List 有序(存入和取出的顺序一致)，元素可重复

|--ArrayList

底层数据结构是数组，查询快，增删慢。

线程不安全，效率高。

|--Vector

底层数据结构是数组，查询快，增删慢。

线程安全，效率低。

|--LinkedList

底层数据结构是链表，查询慢，增删快。

线程不安全，效率高。

|--Set 无序的 元素唯一

|--HashSet

|--TreeSet

47：泛型是什么?有什么用?在哪里用?泛型有什么好处和弊端？

泛型是一种把明确数据类型的工作推迟到创建对象或者调用方法的时时候才去明确的特殊的数据类型。

能优化程序设计，解决了黄色警告线问题。

把运行时期的异常提前到了编译时间。

避免了强制类型转换。

好处：

优化了程序的设计，解决了黄色警告线的问题。

把运行时期的问题提前到了编译时期间解决了。

避免了强制类型转换。

弊端：

让类型统一了，不能存储不同的数据类型了。

48：用迭代器遍历集合的时候，用集合修改集合有没有问题？如果有，怎么解决？（*面试题）

有问题，因为会出现并发修改异常。

解决方法有多种，比如，我们可以不通过集合来修改集合，而使用迭代器来修改集合。

像 ListIterator 迭代器就有添加方法。

49:HashSet 如何保证元素唯一性的呢?

底层数据结构是哈希表(散列表)。具体的是由一个元素是单向链表的数组组成。

它依赖于两个方法：hashCode()和 equals()方法。

执行顺序：

先判断 hashCode()是否相同，

如果相同

继承执行 equals()方法，看其返回值：

true:元素重复，不存储。

false:元素不重复，存储。

如果不同

存储。

记住：

看到 HashXxx 结构的集合，就要知道，被该集合存储的元素要重写 hashCode()和 equals()方法。

而且，是自动生成的。

50:TreeSet 底层数据结构是什么？如何保证元素的唯一性的呢?

底层数据结构是二叉树。

根据比较的返回值是否是 0 来决定。

如何保证元素的排序的呢?

A:自然排序 元素具备比较性

让集合中被存储的元素所属的类实现 Comparable 接口。

B:比较器排序 集合具备比较性

在创建集合对象的时候，让构造方法接收一个 Comparator 接口的子类对象。

51:LinkedHashSet 底层数据结构是什么？如何保证元素的唯一性的呢?

底层由链表和哈希表组成。

由链表保证有序。

由哈希表保证唯一。

52:什么是可变参数?

针对在写一个方法的时候，不知道具体要有多少个形式参数的时候。

java 提供了可变参数的用法。

注意：

A:变量其实是该数据类型的一个数组。

B:形式参数如果有多个的话，可变参数只能是最后一个。

C:数据类型要一致。

53:Map 和 Collection 集合的区别？(*面试题)

A:Map 集合存储的是键值对形式的元素。

Collection 集合存储的是单个的元素。

B:Map 集合的键是唯一的。

Collection 的儿子 Set 集合元素是唯一的。

C:Map 集合的值是可以重复的。

Collection 的儿子 List 集合的元素是可以重复的。

54:Hashtable 和 HashMap 的区别?(面试题)

A:HashMap 线程不安全，效率高。允许 null 键和 null 值。

B:Hashtable 线程安全，效率低。不允许 null 键和 null 值。

55:Collection 和 Collections 有什么区别？

Collection:是集合的顶层接口，定义了集合的通用方法。

Collections:是一个工具类，里面定义了对集合进行操作的一些常见方法。

56:什么是异常？异常有哪几种？各有什么特点？

程序出现了不正常的情况，就是异常。

异常的继承体系结构：

Throwable

|--Error 严重的问题，一般我们解决不了。

|--Exception

|--RuntimeException运行时期异常，这种问题一般要修正代码。

|--非 RuntimeException 编译时期异常，必须进行处理，否则代码不能够通过。

57:throws 和 throw 的区别？（*面试题）

throws

位置：在方法()后面，跟的是类名。

如果后面根据的是 RuntimeException 及其子类，那么，该方法可以不用处理。

如果后面根据的是 Exception 及其子类，那么，必须要编写代码进行处理，或者调用的时候抛出。

throw

位置：在方法中，跟的对象名称。

如果方法中，有 throw 抛出 RuntimeException 及其子类，那么，声明上可以没有 throws。

如果方法中，有 throw 抛出 Exception 及其子类，那么，声明上必须有 throws。

58:final、finally、finalize 的区别？（*面试题）

final：是最终的意思，用于修饰类、变量、和方法。修饰类的时候，类是最终类，不可以被继承。

修饰变量的时候，变量为常量，不可以被改变。修饰方法的时候，方法不能被重写。

finally:是异常处理的一部分,它里面的代码永远会执行(前提是 jvm 没退出),一般用于释放资源。

finalize:是 object 类的一个方法，用于垃圾处理。

59:什么是递归？使用递归需要注意哪些？
递归就是方法定义中调用方法本身的现象。

A:递归一定要有出口，否则就是死递归。

B:递归的次数不能太多，否则内存溢出。

C:构造方法不能递归使用。

60:基本的 IO 流有哪些？(*面试题)
字节流：InputStream、OutputStream、FileInputStream、FileOutputStream

字符流：Reader、Writer、FileReader、FileWriter

高效字节流：BufferedInputStream、BufferedOutputStream

高效字符流： BufferedReader、 BufferedWriter

最开始的时候，只有字节流，但是后来由于中文字符或者其他字符的出现，用两个字节才能表示。

如果用字节流也是可以读写字符文件的数据的，但是比较麻烦。为了简化这种操作，就提供了字符流。

61:flush()和 cloas()的区别？(*面试题）
flush():刷新缓冲区，流对象还可以继续使用。

close():释放流资源，但是会先刷新一次缓冲区，操作完毕后，流对象不可以再使用。

62:什么是多线程？进程和线程的区别是什么？
多线程就是应用程序的多条执行路径。

进程：正在运行的应用程序，每个进程的具备独立的运行空间。

线程：是进程的执行单元，执行路径。如果是多个线程，那么，这多个线程共享同一个进程资源。

63:启动线程调用的是 run()还是 start()方法？run()和 start()的区别是什么？（*面试题）
启动线程调用的是 start()

run()封装了被线程执行的代码，start()是启动线程并调用 run()方法。

64:多线程有几种实现方案？分别是什么？如何操作？（*面试题）

多线程有两种实现，分别是：

1，继承 Thread 类

自定义类继承 Thread 类，在类中重写 run()方法，测试类中创建自定义类对象，并调用 start()方法

2，实现 Runnable 接口

自定义类实现 Runnable 接口，重写 run()方法，测试类中创建自定义对象，

创建 Thread 对象，把自定义对象作为构造参数传递。调用 Thread 类的 start()方法。

65:线程的生命周期？ （*面试题）

新建： 创建线程对象

就绪： 具备 cup 执行资格，没有执行权，随时准备执行

运行： 具备执行资格，执行权，执行 run()中的代码

堵塞： 是当线程运行到符合某个我们定义的条件时，它会停止下来等待唤醒

死亡： run()结束了

66:多线程为什么会出现安全问题？怎么解决呢？（*面试题）

如果满足以下条件，那么就会出现安全问题：

A：是多线程程序。

B：有共享的数据。

C：针对共享数据有多条语句操作。

只要我们把多线程环境中，把操作共享数据的操作，变成单线程就没有问题了。

Java 针对这种情况，就提供了同步技术：

A：同步代码块

B：同步方法

C：JDK5 以后的 Lock 锁

67:同步的锁对象分别是？（*面试题）

代码块：任意对象

方法：this

静态方法：类名.class

68:sleep()和 wait()的区别？（*面试题）

sleep():必须指定时间，不释放锁对象。

wait():可以指定时间，也可以不指定。释放锁对象。

69:线程死锁是什么，为什么有死锁，怎么解决死锁？（*面试题)

为了解决程序因占用资源，出现资源争抢，而出现的程序进入等待的状态（死锁）。

举例：有 A 和 B 两个线程，有 CD 两把锁， A 和 B 嵌套 CD 锁，A 线程中有 C，D 锁，B 线程中有 D C 两把锁，当两个线程运行时，就可能会出现死锁导致

程序停滞的情况。

怎么解决：真正意义上来说，死锁是不能被解决的，死锁是多线程中的一个需要避免的重大的问题，当我们在编写程序时，可以给共享的资源加上另外一

个把锁，控制资源的动态，同时可以设置线程的优先级使线程之间协调合理的利用 CPU 的时间。

70:线程间的通信是什么？

不同种类的线程针对同一个资源的操作。

71:什么是网络编程？

用编程语言来实现计算机的资源共享和信息传递，就叫做网络编程。

72:网络通信三要素是什么？ （*面试题）

A:IP 地址

计算机在网络中的唯一标识。

现在使用的是："点分十进制"

B:端口

应用程序的的标记。

C:协议

通信的规则。

73:UDP 和 TCP 的区别？ （*面试题）

UDP:不建立连接，数据打包传输，数据有限制，数据不可靠，速度快。

TCP:建立连接，数据无限制，数据可靠，速度慢。

74:反射是什么？反射获取字节码文件的三种方式？反射的好处？（*面试题）

在运行状态下，通过 class 文件对象（Class 的对象），去使用构造方法，成员变量，成员方法。就是反射。

3 种方法：

A.用 Object 类的 getClass 方法得到。

B.用任意数据类型的静态 class 属性可以得到

C.用 Class 类的静态方法 forName（String className）方法得到

好处：只要有一个类或者一个类的对象，就可以得到这个类或对象的所有属性和方法。包括私有的。

2，单词中的面试题

CHAR 型变量中能不能存贮一个中文汉字?为什么? 一个字符占几个字节?
答:能存储一个中文汉字,char 数据类型占两个字节,一个中文汉字就是两个字节,一个字符占两个字节.

Person p = new Person();它在内存中做了哪些事情?
答案:

A：将 Person.class 文件加载到内存中。

B：在堆内存中创建一个对象 Person。

C：把 Person 中的属性进行默认初始化。

D：把 Person 中的属性进行显示初始化。

E：调用构造代码块(如果没有，不执行这个操作)。

F：调用构造函数进行初始化。

G：在栈内存中声明 Person 类型的变量 P。

H：把堆内存的地址(引用)赋给了栈内存中 P。

接口特点：

A：接口是对外暴露的规则

B：接口是功能的扩展

C：接口降低了程序的耦合性。

**内聚(自己实现功能的能力)

**高内聚，低耦合。

举例：主板和 CPU,USB 接口,电源插座。

D：扩展说了下接口的理解

**狭义的理解就是 java 中的接口。

**广义的理解就是：任何定义的规范都是接口。

接口和抽象类的区别：

A：抽象类只能被单继承；接口可以被多实现。

B：抽象类中的成员：

成员变量:可以是常量，也可以是变量。

成员方法:可以是抽象的，也可以是非抽象的。

构造方法:虽然不可以创建对象，但是可以给子类实例化用。

C：接口中的成员：

成员变量：只能是常量。默认修饰符 public static final

成员方法：只能是抽象的。默认修饰符 public abstract

D：抽象类中定义的是体系结构中的共性的内容。

接口中定义的是对象的扩展功能。

E：抽象类被继承表示的是："is a"的关系。xx 是 yy 中的一种。

接口被实现表示的是："like a"的关系。xx 像 yy 中的一种。

构造器 Constructor 是否可被 override?
答:不可以

接口是否可继承接口? 抽象类是否可实现(implements)接口? 抽象类是否可继承具体类(concrete class)? 抽象类中是否可以有静态的 main 方法?
答:接口可以继承接口,抽象类可以实现接口,

是否可以从一个 static 方法内部发出对非 static 方法的调用?
答：不可以。因为非 static 方法是要与对象关联在一起的，必须创建一个对象后，才可以在该对象上进行方法调用，而 static 方法调用时不需要创建对象，可以直接调用。也就是说，当一个 static 方法被调用时，可能还没有创建任何实例对象，如果从一个 static 方法中发出对非 static 方法的调用，那个非 static 方法是关联到哪个对象上的呢？这个逻辑无法成立，所以，一个 static 方法内部发出对非 static 方法的调用。

wait 和 sleep 的区别?
答:

String 是最基本的数据类型吗?
答:不是,string 是引用数据类型.

是否可以继承 String 类?
答:不可以.因为 string 类被 final 修饰的,被 final 修饰的类不能被继承.

StringBuffer 与 StringBuilder 的区别?
答：StringBuffer 是 jdk1.0 版本的,是线程安全的,效率低

StringBuilder 是 jdk1.5 版本的,是线程不安全的,效率高

"=="和 equals 方法究竟有什么区别?
答: A:==

a:基本类型 比较的是基本类型的值

b:引用类型 比较的是引用类型的地址值

B:equals()

只能比较引用类型。

默认比较地址值。

Integer 与 int 的区别?
答:Integer 是 int 类型的包装类,是引用数据类型,int 是基本数据类型

Math.round(11.5)等於多少? Math.round(-11.5)等於多少?
答:12,-11

ArrayList 和 Vector 的区别?
答: ArrayList:底层的数据结构使用的是数组结构。特点：查询速度很快。但是增删稍慢。线程不同步。

Vector:底层是数组数据结构。线程同步。被 ArrayList 替代了。因为效率低。

LinkedList:底层使用的链表数据结构。特点：增删速度很快，查询稍慢。线程不同步。

说出 ArrayList,Vector, LinkedList 的存储性能和特性?
答:同上

说说 is a 和 has a 的区别?
答:

jdk 中哪些类是不能继承的?
答:string 等被 final 修饰的类

HashMap 和 Hashtable 的区别?
答: Hashtable:底层是哈希表数据结构，不可以存入 null 键 null 值。该集合是线程同步的。jdk1.0.效率低。

HashMap：底层是哈希表数据结构，允许使用 null 值和 null 键，该集合是不同步的。将 hashtable 替代，jdk1.2.效率高。

List 和 Map 区别?
答: List(子接口):元素是有序的，元素可以重复。因为该集合体系有索引。

Map 集合:该集合存储键值对。一对一对往里存。而且要保证键的唯一性。

java 中实现多态的机制是什么？
答:父类引用指向子类对象

IO 操作中为什么要释放资源?
答:IO 操作后如果不释放资源,会导致资源耗尽,最后系统崩溃.

flush()和 close()有什么区别?
答: flush()方法:用来刷新缓冲区的,刷新后可以再次写出

close()方法:用来关闭流释放资源的的,如果是带缓冲区的流对象的 close()方法,不但会关闭流,还会再关闭流之前刷新缓冲区,关闭后不能再写出

List、Map、Set 三个接口，存取元素时，各有什么特点？
答: List:可以存取重复元素

Map 不可以存储重复的键值,

Set 不可以存储重复元素

List 是有索引,有序的

Map 是无序的

Set 也是无序的

你所知道的集合类都有哪些？主要方法？
答: 答：集合分为单列集合和双列集合。

单列集合的顶层是 Collection 接口，包括 List 和 Set 集合。

（1.1）List 集合的特点是元素可重复，有序，有索引，能够有角标操作集合，有特有的迭代方式 ListIterator。包括 ArrayList、LinkedList 和 Vector。

ArrayList 集合底层采用的是数组数据结构，查询速度比较快，因为数组有索引，在内存中分配的空间是连续的，但是增删比较慢。线程不同步，效率高。初始容量为 10。

LinkedList 集合的底层采用的是链表数据结构，增删速度比较快，查询速度比较慢。线程不同步。

Vector 底层数据结构也是数组数据结构，但是线程同步，效率低，特有取出元素的方式是枚举。因为效率低，逐步被 ArrayList 替代。

（1.2） Set 集合的特点元素是无序的（存入和取出的顺序不一致），元素不可以重复。包括 HashSet 和 TreeSet。

HashSet 的底层数据结构是哈希表，线程不同步，效率高。保证元素的唯一性额有的依据是元素的 hashCode 和 equals 方法。如果 hashCode 不同，不调用 equals 方法。如果 hashCode 相同，才会调用 equals 方法判断元素是否相同。

TreeSet 的底层数据结构是二叉树，线程不同步，效率高。能够给元素进行排序。保证元素唯一性的依据是 compareTo 和 return0。排序的两种方式：第一种元素自身实现 Comparable 接口，重写 compareTo（）方法。这种排序方式叫元素的自热排序，也叫默认排序。第二种是当元素自身不具备比较性或者具备的比较性不是所需要的，这时就让集合自身具备比较性，当集合初始化时就有了比较性。定义一个比较器实现 Comparator 接口，重写 compare 方法，定义集合的时候将比较器作为参数传递给 TreeSet 的构造函数，这样集合就具有了比较性。

（2）Map 是双列集合的顶层接口，该集合存储的是键值对，一对一对的往里存，而且要保证键的唯一性。包括 Hashtable、HashMap、TreeMap。

Hashtable 的底层数据结构是哈希表，不可以存储 null 键和 null 值，线程同步，效率低。JDK1.0.

HashMap 的底层数据结构是哈希表，可以存储 null 键和 null 值，线程不同步，将 Hashtable 替代，JDK1.2 效率高。保证键的唯一性的 依据是 hashCode 和 equals 方法。

TreeMap 的底层数据结构是二叉树，线程不安全，能够给集合中的键排序。

Collection 和 Collections 的区别。
答:

字节流与字符流的区别?
答:

描述一下 JVM 加载 class 文件的原理机制?
答： JVM 中类的装载是由 ClassLoader 和它的子类来实现的，Java ClassLoader 是一个重要的 Java 运行时系统组件。它负责在运行时查找和装入类文件的类。

3，STRING 面试题

a.java 中创建的字符串都会存在一个叫做常量池的地方.

b.用 new 创建的 String 对象

1.此字符串已经存在常量池中,则直接拿过来用,这时只在堆中创建了一个对象.

2.此字符串不在常量池中,则在常量池中创建该字符串,然后在把副本值给堆中 String 对象

此时创建了两个对象(堆和常量池)

c.字面值常量的字符串相加(+号)

java 中会有字符串常量优化机制,在编译的时候会直接在常量池中完成相加操作,

如果有不再创建对象,直接拿来用.

d.字符串变量相加(+号)

字符串变量相加,低层其实用 StringBuffer(StringBuild)对象在堆中完成相加操作,

相加后存的其实 StringBuffer.toString()地址值.

简单记忆:

java 字符串常量优化机制:有就使用,没有就创建.

如果是字符串变量和字符串常量拼接,会先创建一个字符串缓冲区(StringBuffer)的对象,

拼接完成之后,会将该对象转成 String,最后把地址值赋值给 String 引用.

String 作为参数传递

java 中只有值传递,引用数据类型传递的是地址值.

形参的类型:

基本数据类型:形参的改变不影响实参.

引用数据类型:形参的改变直接影响实参.(String 类有点特殊)

String 类虽然是引用数据类型,但是他当作参数传递时和基本数据类型是一样的,是值传递!!!

String 类虽然是引用数据类型,但是他当作参数传递时和基本数据类型是一样的,是值传递!!!

String 类虽然是引用数据类型,但是他当作参数传递时和基本数据类型是一样的,是值传递!!!