Tarena
达内科技

arena eaching ystem

JAVA 企业面试题精选

Java企业应用及互联网
高级工程师培训课程

达内集团教学研发部 编著

目　录

第一部分

1. Java 基础

1.1.这你认为 Java 与其他（你所了解的）语言相比，有什么优点和缺点?

参考答案：

首先，Java 与 C/C++相比。Java 语言是一种完全的面向对象语言，虽然它的底层（运行时库）是用C语言开发的，可是并不依赖于C。因为 Java 的运行是在运行时库的支持下进行的，所以运行效率比起可以更接近底层的 C/C++来说效率会有所影响，不过 Java 的类库采用很好的设计理念，非常好用，也非常实用，已经成为业界的一种标准开发语言。它的跨平台的特性受到开发者的青睐，只需要开发一次就能在所有安装了 Java 运行时库的环境上运行。

其次，Java 与 C#相比。C#是微软开发的一种编程语言，语法类似 Java，几乎就是 Java 的翻版。运行原理和 Java 也类似，也是通过运行时库的支持运行。不过支持的平台还很有限。Java 几乎被所有平台支持，而C＃目前只被 Windows 和 linux 支持，Windows 下的支持当然是由微软自己开发的，而 Linux 下的支持则有 mono 支持。实际上，mono 也是把C＃应用转化为 Java 应用而已，所以本质上，C＃仍然只是被微软自己的操作系统支持。应用平台受到限制，是它最大的缺点。

1.2.请回答以下几个名词的意思：JVM、JDK、JRE、JavaSE、JavaEE，JavaME，GC

参考答案：

JVM：Java 虚拟机，Java Virtual Machine 的缩写。是一个虚构出来的计算机，通过在实际的计算机上仿真模拟各种计算机功能来实现的。Java 虚拟机有自己完善的硬体架构，如处理器、堆栈、寄存器等，还具有相应的指令系统。JVM 屏蔽了与具体操作系统平台相关的信息，使得 Java 程序只需生成在 Java 虚拟机上运行的目标代码（字节码），就可以在多种平台上不加修改地运行。

JDK：Java 开发工具包，Java Development Kit 的缩写。JDK 是整个 Java 的核心，包括了 Java 运行环境、Java 工具和 Java 基础类库。

JRE：Java 运行环境，Java Runtime Environment 的缩写。运行 JAVA 程序所必须的环境的集合，包含 JVM 标准实现及 Java 核心类库。

JavaSE：Java Standard Edition，标准版，是我们常用的一个版本，从 JDK 5.0 开始，改名为 Java SE，主要用于桌面应用软件的编程。

JavaEE：Java Enterprise Edition，企业版。JavaEE 是 J2EE 的一个新的名称 ，主要

用于分布式的网络程序的开发。

JavaME：Java Micro Edition,是为机顶盒、移动电话和 PDA 之类嵌入式消费电子设备提供 Java 语言平台，包括虚拟机和一系列标准化的 Java API。

GC：垃圾回收，Garbage Collection 的缩写。当 Java 虚拟机发觉内存资源紧张时，则会自动地去清理无用对象（没有被引用到的对象）所占用的内存空间。

1.3.JVM 能有几个实例？

参考答案：

每个 Java 程序对应于一个 JVM 实例，当一个 Java 程序运行时就创建一个 JVM 实例，因此 JVM 实例的个数取决于同时执行的程序个数。

1.4.Java 跨平台是如何实现的？

参考答案：

Java 是利用 JVM（Java 虚拟机）实现跨平台的。

Java 源代码（*.java）经过 Java 编译器编译成 Java 字节码（*.class）,执行 Java 字节码，Java 字节码经过 JVM 解释为具体平台的具体指令，并执行。不同平台有不同的 JVM，主流平台都提供了 JVM，所以 Java 字节码可以在主流平台上能够解释执行。在这个意义上 Java 是跨平台的，也就是说：Java 的字节码是跨平台的。

1.5.简述 TCP/UDP 协议的区别？

参考答案：

TCP/UDP 协议的区别如下表所示。

TCP/UDP 协议比较

比较项	TCP	UDP
是否可连接	面向连接	面向非连接
传输可靠性	可靠的	不可靠的
应用场合	传输大量的数据	少量数据
速度	慢	快

1.6.阐述一下类的命名规则、方法的命名规则、变量的命名规则、包名的命名规则、常量的命名规范？

参考答案：

在 Java 中，类的命名、方法的命名、变量的命名、包名的命名以及常量的命名首先必须符合 Java 标识符的命名规则，规则如下：

1） 可以以字母、数字、“_”和“$”符组成；

2） 首字符不能以数字开头；

3） 中文可以作为变量名，但不提倡使用；

4） Java 大小写敏感，命名变量时需要注意；

5） 不能使用 Java 保留字（一些 Java 语言规定好的，有特殊含义的字符），如：int、if、for、break 等。

其次，类的命名、方法的命名、变量的命名、包名的命名以及常量的命名要符合如下规范：

1） 类命名规范：首字母大写，如果由多个单词合成一个类名，要求每个单词的首字母也要大写，如：HelloWorld。

2） 方法命名规范：首字母小写，中间的每个单词的首字母都要大写,如：getName。

3） 变量的命名规范：变量的命名规范和方法一样，首字母小写，中间的每个单词的首字母都要大写，如：name。

4） 包的命名规范：Java 包的名字都是由小写单词组成。但是由于 Java 面向对象编程的特性，每一名 Java 程序员都可以编写属于自己的 Java 包，为了保障每个 Java 包命名的唯一性，在最新的 Java 编程规范中，要求程序员在自己定义的包的名称之前加上唯一的前缀。由于互联网上的域名称是不会重复的，所以程序员一般采用自己在互联网上的域名称作为自己程序包的唯一前缀。例如：“ com.sun.swt”一般公司命名会以“com.公司名.项目名.模块名“开头,所以会长一点，如 com.land.oa.emp.struts.action。

5） 常量的命名规范：基本数据类型的常量名为全大写，如果是由多个单词构成，可以用下划线隔开，如：WEEK_OF_MONTH。

1.7.阐述一下 Java 共有几种注释方法？

参考答案：

在 Java 中有三种注释类型：

1） 单行注释符号是 “// “，只能注释一行。

2） 块注释符号是 “ /* */ “，可以跨多行。

3） javadoc 注释符号是 /** */，可以跨多行，而且生成 javadoc 时，这样的注释会被生成标准的 Java API 注释。

1.8.如何增加代码的清晰度和可观性？

参考答案：

增加代码的清晰度和可观性常用的方式如下：

1） 给代码添加注释。

2） 类名包名等命名规范化。

3） 缩进排版规范。

4） 添加异常的处理。

5） 使用测试类和测试方法。

1.9.Java 中有两个关键字： void 和 null ，它们有什么区别？

参考答案：

在 Java 中，void 仅用于无返回值的方法上，例如：

```
public void a(){}
```

该方法不需要返回数据，故返回值类型设置为 void。
null 则代表对象/变量的值，例如：

```
String a = null;
```

表示变量 a 没有被实例化，没有指向具体的内存地址。

1.10.Java 中结构化程序设计有哪三种基本流程，分别对应那些语句？

参考答案：

Java 中结构化程序设计有三种基本流程，分别是顺序、选择、循环。其中，顺序表示程序中的各操作是按照它们出现的先后顺序执行的；选择对应 Java 语言中的 if 语句和 switch 语句；循环对应 Java 语言中的 for 语句、do-while 语句以及 while 语句。

1.11.&和&&的区别？

参考答案：

&和&&都可以执行关系判断。二者的区别是：&运算是把逻辑表达式全部计算完，而&&运算具有短路计算功能。所谓短路计算，是指系统从左至右进行逻辑表达式的计算，一旦出现计算结果已经确定的情况，则计算过程即被终止。

1.12.写出 Java 中 8 种原始类型及其字节长度？

参考答案：

Java 中 8 种原始类型及其字节长度如下表所示.

Java 中的 8 种原始类型介绍

类型名称	字节空间	说明
byte	1 字节（8 位）	存储 1 个字节数据
short	2 字节（16 位）	兼容性考虑，一般不用
int	4 字节（32 位）	存储整数（常用）
long	8 字节（64 位）	存储长整数（常用）
float	4 字节（32 位）	存储浮点数
double	8 字节（64 位）	存储双精度浮点数（常用）

char	2 字节（16 位）	存储一个字符
boolean	1 字节（8 位）	存储逻辑变量（true、false）

1.13.请描述一下 JVM 加载 class 文件的原理机制？

参考答案：

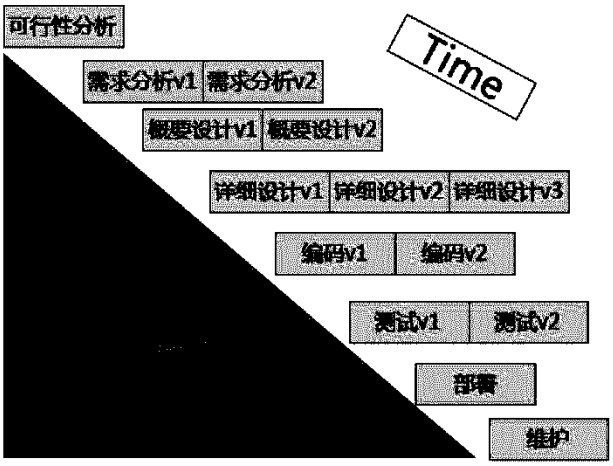
JVM 中类的装载是由 ClassLoader 和它的子类来实现的,Java ClassLoader 是一个重要的 Java 运行时系统组件，它负责在运行时查找和装入类文件中的类。

1.14.你对软件开发中迭代的含义的理解？

参考答案：

软件开发中，各个开发阶段不是顺序执行的，而各个阶段都进行迭代，然后在进入下一阶段的开发。这样对于开发中的需求变化，及人员变动都能得到更好的适应。

软件开发过程中迭代模型如下图所示。



1.15.什么是进程？

参考答案：

进程是操作系统结构的基础，是一个计算机中正在运行的程序实例。可以分配给处理器并由处理器执行的一个实体，由单一顺序的执行显示，一个当前状态和一组相关的系统资源所描述的活动单元。

1.16.什么是垃圾回收？什么时候触发垃圾回收？如何降低垃圾回收的触发频率？它能保证程序有足够的可用内存吗？

参考答案：

各问题的参考答案如下：

第一问：垃圾回收(GC)是 Java 语言的一个重要特性，作用是释放不再被使用的内存。

第二问：垃圾回收由系统进行管理。在系统认为需要的时候自动启动一个线程进行处理。

第三问：尽量减少垃圾内存，也就是新建对象的数量，可以降低垃圾回收的频率。

第四问：垃圾回收机制无法保证有足够的内存。

1.17.Java 中会存在内存泄露吗，请简单描述？

参考答案：

会出现内存泄漏。

一般来说内存泄漏有两种情况。一是在堆中分配的内存，在没有将其释放掉的时候，就将所有能访问这块内存的方式都删掉；另一种情况则是在内存对象已经不需要的时候，还仍然保留着这块内存和它的访问方式（引用）。第一种情况，在 Java 中已经由于垃圾回收机制的引入，得到了很好的解决。所以，Java 中的内存泄漏，主要指的是第二种情况。

下面给出了一个简单的内存泄露的例子。在这个例子中，我们循环申请 Object 对象，并将所申请的对象放入一个 List 中，如果我们仅仅释放引用本身，那么 List 仍然引用该对象，所以这个对象对 GC 来说是不可回收的。代码如下所示：

```
List list=new ArrayList(10);
for (int i=1;i<100; i++)
{
    Object o=new Object();
    list.add(o);
    o=null;
}
```

此时，所有的 Object 对象都没有被释放，因为变量 list 引用这些对象。

1.18.Java 源文件中是否可以包括多个类,有什么限制？

参考答案：

一个 Java 源文件中可以包含多个类，每个源文件中至多有一个 public 类，如果有的话，那么源文件的名字必须与之相同。如果源文件中没有 public 类，则源文件用什么名字都可以，但最好还是具有特定的意义，免得自己都不记得里面写的是什么了。一般建议一个源文件中只写一个 Java 类。

1.19.列出自己常用的 jdk 包？

参考答案：

常用的 JDK 包如下：

1．java.lang 包：这个包中包含了 JDK 提供的基础类，比如 String 等都是这里面的，这个包是唯一一个可以不用导入就可以使用的包；

2．java.io 包：这个包中包含了与输入输出相关的类，比如文件操作等；

3．java.net 包：这个包中包含了与网络有关的类，比如 URL,URLConnection 等；

4． java.util 包：这个是系统辅助类，特别是集合类 Collection,List,Map 等；

5． java.sql 包：这个是数据库操作的类，Connection, Statememt，ResultSet 等。

1.20.简单说明什么是递归？什么情况会使用？并使用 Java 实现一个简单的递归程序？

参考答案：

1．递归做为一种算法在程序设计语言中广泛应用，是指函数/过程/子程序在运行过程中直接或间接调用自身而产生的重入现象。

2．递归算法一般用于解决三类问题：

1）数据的定义是按递归定义的。(Fibonacci（斐波那契）函数)

2）问题解法按递归算法实现。(回溯)

3）数据的结构形式是按递归定义的。(树的遍历，图的搜索)

3．下面是使用递归算法实现计算某个整数在二进制中的个数，代码如下所示：

```
**
* 计算二进制中 1 的个数，
* N 为奇数，二进制中 1 的个数等于 N/2 的个数
* 例子：
* num=13
* 1.getBinary(13/2=6)+1;
* 2.getBinary(6/2=3)+1
* 3.getBinary(3/2=1)+1+1
* 4.getBinary(1)+1+1;  getBinary(1) 返回 1 ，与后边两个 1 相加得结果 3
*/
public static int getBinary(int num){
   if(num==1)
       return 1;
    if(0==num%2){//是否为偶数
       return getBinary(num/2);
    }else{
       return getBinary(num/2)+1;
    }
}
```

1.21.请写出求 n！的算法？

参考答案：

求 n!的算法的代码如下所示：

```
public class Factorial {
 public static void main(String[] args) {
     long n = 5;
     System.out.println(doFactorial(n));
 }
 public static long doFactorial(long n) {
     if (n < 1) {
         System.out.println("ERROR");
         return 0;
     } else if (n == 1 || n == 2) {
         return n;
     } else {
```

```
            return n * doFactorial(n - 1);
        }
    }
}
```

1.22.排序都有哪几种方法？

参考答案：

排序的方法有：插入排序（直接插入排序、希尔排序）、交换排序（冒泡排序、快速排序）、选择排序（直接选择排序、堆排序）、归并排序、分配排序（箱排序、基数排序）。

1.23.写一个排序算法，将 10 个 1-100 随机数字进行排序？

参考答案：

选择排序算法实现 10 个 1-100 随机数字的排序，代码如下所示：

```
public class SelectSort {
// 选择排序方法
public static void selectionSort(int[] number) {
    for (int i = 0; i < number.length - 1; i++) {
        int m = i;
        for (int j = i + 1; j < number.length; j++) {
            if (number[j] < number[m])
                m = j;
        }
        if (i != m)
            swap(number, i, m);
    }
}

// 用于交换数组中的索引为 i、j 的元素
private static void swap(int[] number, int i, int j) {
    int t;
    t = number[i];
    number[i] = number[j];
    number[j] = t;
}

public static void main(String[] args) {
    // 定义一个数组
    int[] num = new int[10];
    for (int i = 0; i < num.length; i++) {
        num[i] = (int) (Math.random() * 100) + 1;
    }
    // 排序
    selectionSort(num);
    for (int i = 0; i < num.length; i++) {
        System.out.println(num[i]);
    }
}
}
```

1.24.请用 Java 语言编写一个完成冒泡排序算法的程序？

参考答案：

使用 Java 语言实现的冒泡排序算法代码如下所示：

```
import java.util.Random;
import java.util.Arrays;

public class BubbleSort {
 public static void main(String[] args) {
     int[] arr = new int[10];
     for (int i = 0; i < arr.length; i++) {
         Random ran = new Random();
         arr[i] = ran.nextInt(100);
     }
     System.out.println(Arrays.toString(arr));
     // 冒泡排序
     System.out.println("----------冒泡排序 开始----------");
     for (int i = 0; i < arr.length - 1; i++) {
         for (int j = 0; j < arr.length - i - 1; j++) {
             if (arr[j] > arr[j + 1]) {
                 int t = arr[j];
                 arr[j] = arr[j + 1];
                 arr[j + 1] = t;
             }
         }
         System.out.println(Arrays.toString(arr));
     }
     System.out.println("----------冒泡排序 结束----------");
     System.out.println(Arrays.toString(arr));
 }
}
```

1.25.有一数组 a[1000]存放了 1000 个数，这 1000 个数取自 1-999，且只有两个相同的数,剩下的 998 个数不同，写一个搜索算法找出相同的那个数的值？

参考答案：

下列代码中使用二分搜索算法实现了从数组 a[1000]中，查找两个相同的数，代码如下所示：

```
import java.util.Arrays;
public class SearchDemo {
 /** 被搜索数据的大小 */
 private static final int size = 1000;
 public static void main(String[] args) {
     int[] data = new int[size];
     // 添加测试数据
     for (int k = 0; k < data.length; k++) {
         data[k] = k + 1;
     }
     data[999] = 567;
     result(data);

 }
 /**
  * 调用二分搜索算法的方法实现查找相同元素
  * @param data
```

```
     */
    public static void result(int data[]){
        Arrays.sort(data);
        for (int i = 0; i < data.length; i++) {
            int target = data[i];
            data[i] = 0;
            int result = binaryFind(data, target);
            if (result != -1) {
                System.out.println("相同元素为: "+data[result]);
                break;
            }
        }
    }
    /**
     * 二分搜索算法实现
     *
     * @param data
     *          数据集合
     * @param target
     *          搜索的数据
     * @return 返回找到的数据的位置，返回-1 表示没有找到。
     */
    public static int binaryFind(int[] data, int target) {
        int start = 0;
        int end = data.length - 1;
        while (start <= end) {
            int middleIndex = (start + end) / 2;
            if (target == data[middleIndex]) {
                return middleIndex;
            }
            if (target >= data[middleIndex]) {
                start = middleIndex + 1;
            } else {
                end = middleIndex - 1;
            }
        }
        return -1;
    }
    }
```

1.26.现有一个 32 位的整型变量 value 和一个有 32 个元素的数组 a[32] ,要求 :1、对 value 随机赋值；2、让数组 a[n]的值等于 value “位 n” 的值，0<=n<=31。举例：如果 value 的 “位 0” (Bit0)=0 ,那么 a[0]=0 ;如果 value 的 “位 10” (Bit10)=1 ,那么 a[10]=1 。

参考答案：

参考答案的代码如下所示：

```
public class Foo {
public static void main(String[] args) {
    //产生随机数
    int random = (int) (Math.random() * Integer.MAX VALUE + 1);
    //转成二进制字符串
    String str=Integer.toBinaryString(random);
    //转成二进制时最前面的零被省略，补上省略的 0
    if(str.length()<32){
        for(int j=0;j<=32-str.length();j++){
        str="0"+str;
        }
    }
    //给数组赋值
    int[] a=new int[32];
```

```
        for(int i=0;i<str.length();i++){
            a[i]=Integer.parseInt(String.valueOf(str.charAt(i)));
            System.out.println("a["+i+"]="+a[i]);
        }
    }
}
```

1.27.有 1~100 共一百个自然数，已随机放入一个有 98 个元素的数组 a[98]。要求写出一个尽量简单的方案，找出没有被放入数组的那 2 个数，并在屏幕上打印这 2 个数。注意：程序不用实现自然数随机放入数组的过程。

参考答案：

参考答案的代码如下所示：

```
public static void main(String[] args){
  int[] b = new int[]{....存入 98 个随机的 1~100 的整数};
  int[] a = new int[100];
  for(int t : b) {
    a[t-1]=t;
  }
  for(int t=0; t < a.length; t++){
    if(a[t]==0) {
      System.out.println(t+1);
    }
  }
}
```

1.28.用 1，2， 2， 3， 4，5 这六个数字，用 Java 写一个 main 函数，打印出所有不同的排列，如: 512234 ，412345 等，要求：”4“ 不能在第三位，”3“与”5”不能相连

参考答案：

参考答案的代码如下所示：

```
import java.util.ArrayList;
import java.util.List;
public class Q028 {
 public static List<String> list = new ArrayList<String>();
 public static void group(String str, String nstr) {
     if (str.length() != nstr.length()) {
         String rest = getRest(str, nstr);
         for (int i = 0; i < rest.length(); i++) {
             String temp = str + rest.substring(i, i + 1);
             if (temp.indexOf("4") != 2 && temp.indexOf("35") == -1
                     && temp.indexOf("53") == -1) {
                 //把符合条件的字符串添加到集合中
                 if (!list.contains(temp)&&temp.length()==nstr.length()) {
                     list.add(temp);
                 }
                 group(temp, nstr);
             }
         }
     }
 }
 public static String getRest(String str, String nstr) {
     String rest = "";
```

```
        if (nstr.length() > str.length()) {
            rest = nstr;
            for (int i = 0; i < str.length(); i++) {
                // 注意此处的 replaceFirst，而不是 replaceAll
                rest = rest.replaceFirst(str.substring(i, i + 1), "");
            }
        }
        return rest;
    }
    public static void main(String[] args) {
        group("", "122345");
        System.out.println(list.toString());
    }
}
```

1.29.编写一个命令行程序，提示让用户输入 2 个整数 ，然后计算这 2 个整数之间能被 5 整除的所有整数的和，并打印显示？

参考答案：

参考答案的代码如下所示：

```
import java.util.Arrays;
import java.util.Scanner;

public class Q029 {
    public static void main(String[] args) {
        Scanner sc = new Scanner(System.in);
        int[] nums = new int[2];
        System.out.println("请输入一个整数：");
        nums[0] = sc.nextInt();
        System.out.println("请输入第二个整数：");
        nums[1] = sc.nextInt();
        Arrays.sort(nums);
        int sum = 0;
        for (int begin = nums[0] + 1; begin < nums[1]; begin++) {
            if (begin % 5 == 0) {
                sum += begin;
            }
        }
        System.out.println(sum);
    }
}
```

1.30.编写一个命令行程序，提示让用户输入用户名和密码。如果用户名和密码都是 admin，则显示登录成功；如果不是则显示登录失败，让用户重新输入。如果用户连续 3 次认证失败，则锁定终止程序？

参考答案：

参考答案的代码如下所示：

```
import java.util.Scanner;

public class Q030 {
    public static void main(String[] args) {
        Scanner sc = new Scanner(System.in);
        int count=0;
```

```
        while(true){
            System.out.println("请输入账号：");
            String name=sc.nextLine();
            System.out.println("请输入密码：");
            String pwd=sc.nextLine();
            if("admin".equals(name)&&"admin".equals(pwd)){
                System.out.println("登录成功.");
                break;
            }else{
                count++;
                if(count<3)
                    System.out.println("登录失败，请重新输入。");
                else{
                    System.out.println("登录次数过多，认证失败。");
                    break;
                }
            }
        }
    }
}
```

1.31.写 Java 代码，打印如下的杨辉三角：

```
        1
       1 1
      1 2 1
     1 3 3 1
    1 4 6 4 1
  1 5 10 10 5 1
```

参考答案：

参考答案的代码如下所示：

```
public class Q031 {
 public static void main(String[] args) {
     int n = 6;
     for (int i = 1; i <= n; i++) {
         for (int j = 1; j <= n - i; j++) {
             System.out.print(" ");
         }
         // 打印空格后面的字符,从第 1 列开始往后打印
         for (int j = 1; j <= i; j++) {
             System.out.print(num(i, j) + " ");
         }
         System.out.println();
     }
 }

 public static int num(int x, int y) {// 第 x 行，第 y 列
     if (y == 1 || y == x) {
         return 1;
     }
     int c = num(x - 1, y - 1) + num(x - 1, y);
     return c;
 }
}
```

1.32.使用 Java 完成快速排序算法？

参考答案：

使用 Java 实现快速排序算法的代码如下所示：

```
package com.stone.algorithm;

public class QuickSort {
 // 排序方法，接受一个 int[]参数，将会调用快速排序方法进行排序
 public static void sort(int[] number) {
     quickSort(number, 0, number.length - 1);
 }
 // 快速排序方法
 private static void quickSort(int[] number, int left, int right) {
     if (left < right) {
         int s = number[left];
         int i = left;
         int j = right + 1;
         while (true) {
             // 向右找大于 s 的数的索引
             while (i + 1 < number.length && number[++i] < s)
                 ;
             // 向左找小于 s 的数的索引
             while (j - 1 > -1 && number[--j] > s)
                 ;
             // 如果 i>=j，退出循环
             if (i >= j){
                 break;
             }
             // 否则交换索引 i 和 j 的元素
             swap(number, i, j);
         }
         number[left] = number[j];
         number[j] = s;
         // 对左边进行递归
         quickSort(number, left, j - 1);
         // 对右边进行递归
         quickSort(number, j + 1, right);
     }
 }

 // 交换数组 number 中的索引为 i、j 的元素
 private static void swap(int[] number, int i, int j) {
     int t;
     t = number[i];
     number[i] = number[j];
     number[j] = t;
 }

 public static void main(String[] args) {
     int[] num = { 34,1,23,345,12,546,131, 54, 78, 6543, 321, 85,
             1234, 7, 76, 234 };
     sort(num);
     for (int i = 0; i < num.length; i++) {
         System.out.println(num[i]);
     }
 }
}
```

1.33.编程题：设有n个人依围成一圈，从第1个人开始报数，数到第m个人出列，然后从出列的下一个人开始报数，数到第m个人又出列，…，如此反复到所有的人全部出列为止。设n个人的编号分别为1，2，…，n，打印出出列的顺序；要求用 Java 实现？

参考答案：

参考答案的代码如下所示：

```
public class Q033 {
 private static boolean same(int[] p, int l, int n) {
     for (int i = 0; i < l; i++) {
         if (p[i] == n) {
             return true;
         }
     }
     return false;
 }

 public static void play(int playerNum, int step) {
     int[] p = new int[playerNum];
     int counter = 1;
     while (true) {
         if (counter > playerNum * step) {
             break;
         }
         for (int i = 1; i < playerNum + 1; i++) {
             while (true) {
                 if (same(p, playerNum, i) == false)
                     break;
                 else
                     i = i + 1;
             }
             if (i > playerNum)
                 break;
             if (counter % step == 0) {
                 System.out.print(i + " ");
                 p[counter / step - 1] = i;
             }
             counter += 1;
         }
     }
     System.out.println();
 }

 public static void main(String[] args) {
     play(10, 7);
 }

}
```

1.34.写一段小程序检查数字是否为质数？

参考答案：

参考答案的代码如下所示：

```
public class Q034 {
 public boolean prime(int n) {
     if (n <= 0)
         System.exit(0);
     for (int i = 2; i <= n; i++)
```

```
            for (int j = 2; j <= Math.sqrt(i); j++)
                if ((n % j == 0) && (j != n))
                    return false;
        return true;
    }
}
```

1.35.1 到 100 自然数，放入到 a[99]这个数组中，写一个函数找出没有被放进数组的那个数字？

参考答案：

参考答案的代码如下所示：

```
public class Q035 {
 public static void main(String[] args) {
     // 模拟 a[99]数组
     int[] b = new int[99];
     for (int i = 0; i < 99; i++) {
         b[i] = i + 1;
     }

     int[] a = new int[100];
     for (int t : b)
         a[t - 1] = t;
     for (int t = 0; t < a.length; t++)
         if (a[t] == 0)
             System.out.println(t + 1);
 }
}
```

1.36.找出 101 到 200 自然数中的质数，for 循环越少越好（我用到 2 个 for 循环）？

参考答案：

参考答案的代码如下所示：

```
public class Q036 {
 public static void main(String[] args) {
     for (int i = 101; i <= 200; i++) {
         boolean b = true;
         for (int n = 2; n < i; n++) {
             if (i % n == 0) {// 成立则不是素数
                 b = false;
                 break;
             }
         }
         if (b) {
             System.out.println(i);
         }
     }
 }
}
```

1.37.用数组实现一个栈（Stack），至少有 入栈方法 push 和出栈方法 pop？

参考答案：

参考答案的代码如下所示：

```
import java.util.Arrays;

class Stack {
 private Object[] data;// 栈的内容
 private int size = 0; // 栈内的元素个数

 public Stack() {
     data = new Object[0];
 }

 // 判断栈是否满
 public boolean isFull() {
     // 当 数组长度与栈内元素个数相同 或者 数组长度为 0 并且元素个数为 0 时
     return data.length == size || (data.length == 0 && size == 0);
 }

 // 判断栈是否 empty
 public boolean isEmpty() {
     return size == 0;
 }

 // 数组扩容 10 个
 public void increData() {
     data = Arrays.copyOf(data, data.length + 10);
 }

 // 入栈操作
 public void push(Object obj) {
     if (isFull()) {
         increData();
     }
     size++;
     data[size - 1] = obj;
 }

 // 出栈操作
 public Object pop() {
     Object o = data[size - 1];
     data[size - 1] = null;
     size--;
     return o;
 }
}

public class Q037 {
 public static void main(String[] args) {
     Stack stack = new Stack();
     stack.push(1);
     stack.push("123");
     Object o = stack.pop();
     System.out.println(o);
     o = stack.pop();
     System.out.println(o);

 }
```

```
}
```

1.38.a、b、c 为 3 个整型变量 ,在不引入第 4 个变量的前提下写一个算法实现 a=b、b=c、c=a？

参考答案：

参考答案的代码如下所示：

```
public class Q038 {
 public static void main(String[] args) {
     int a=1;
     int b=2;
     int c=3;
     a=b-a;
     b=b-a;
     a=a+b;
     b=c-b;
     c=c-b;
     b=c+b;
     System.out.println("a:"+a+"  b:"+b+"  c:"+c);
 }
}
```

1.39.Java 编程写出 1000-2000 可以能被 3 整除的数？

参考答案：

参考答案的代码如下所示：

```
public class Q039 {
 public static void main(String[] args) {
     for (int i = 1000; i <= 2000; i++) {
         if (i % 3 == 0) {
             System.out.println(i);
         }
     }
 }
}
```

1.40.编程：有一对兔子，从出生后第 3 个月起每个月都生一对兔子，小兔子长到第四个月后每个月又生一对兔子，假如兔子都不死，问每个月的兔子总数为多少？

参考答案：

参考答案的代码如下所示：

```
public class Q040 {
 public static void main(String[] args) {
     System.out.println("第 1 个月的兔子对数:    1");
     System.out.println("第 2 个月的兔子对数:    1");
     int f1 = 1, f2 = 1, f, M = 24;
     for (int i = 3; i <= M; i++) {
         f = f2;
         f2 = f1 + f2;
         f1 = f;
```

```
            System.out.println("第" + i + "个月的兔子对数: " + f2);
        }
    }
}
```

1.41.查找有哪几种方法：试写其中一种方法的小例子？

参考答案：

参考答案的代码如下所示：

有顺序查找，二分查找，分块查找，二叉排序树查找等。

下面的 sequelSearch 方法是顺序查找的案例（顺序查找适合于存储结构为顺序存储或链接存储的线性表）。

```
public int sequelSearch(String[] s, String key, int n)
/* 在 s[0]-s[n-1]中顺序查找关键字为 key 的记录 */
/* 查找成功时返回该记录的下标序号；失败时返回-1 */
{
    int i;
    i = 0;
    while (i < n && s[i]!= key) {
        i++;
        if (s[i] == key){
            return i;
        }
    }
    return -1;
}
```

1.42.写代码判断两个数字(x. y)的大小，并返回大数能否被小数整除？

参考答案：

参考答案的代码如下所示：

```
public class Q042 {
 public static void main(String[] args) {
     boolean b=judge(10,21);
     System.out.println(b);
 }
 public static boolean judge(int x, int y) {
     return (x < y ? y % x : x % y) == 0;
 }
}
```

1.43.将一个整型十进制数转化为二进制数(不能使用 Java 的类库)？

参考答案：

参考答案的代码如下所示：

```
public class Q043 {
 public static void main(String[] args) {
     String hex=hexConvert(7);
     System.out.println(hex);
```

```
    }
    public static String hexConvert(int d) {
        String s = "";
        do {
            int f = d % 2;
            if (f == 1) {
                s = "1" + s;
            } else if (f == 0) {
                s = "0" + s;
            }
            d /= 2;
        } while (d != 0);
        return s;
    }
    }
```

1.44.编一个函数 1000 以内，可以被 5 整除，可以被 7 整除，但不被 5 和 7 同时整除，输出符合结果的所有数？

参考答案：

参考答案的代码如下所示：

```
    public class Q044 {
    public static void main(String[] args) {
         divide();
    }
    public static void divide() {
        for (int i = 0; i < 1000; i++) {
            if (i % 5 == 0 && i % 7 != 0 || i % 5 != 0 && i % 7 == 0)
                System.out.println(i);
        }
    }
    }
```

1.45.生成一个六位数的验证码 包括大写字母 小写字母和数字？

参考答案：

参考答案的代码如下所示：

```
    import java.util.Arrays;
    public class Q045 {
    public static void main(String[] args) {
        String[] letters = { "A", "B", "C", "D", "E", "F", "G", "H", "I", "J",
"K",
                "L", "M", "N", "O", "P", "Q", "R", "S", "T", "U", "V", "W",
                "X", "Y", "Z", "a", "b", "c", "d", "e", "f", "g", "h", "i",
                "j", "k", "l", "m", "n", "o", "p", "q", "r", "s", "t", "u",
                "v", "w", "x", "y", "z", "1", "2", "3", "4", "5", "6", "7",
                "8", "9", "0" };
        boolean[] flags = new boolean[letters.length];
        String[] chs =new String[6];
        for (int i = 0; i < chs.length; i++) {
            int index;
            do {
                index = (int)(Math.random()*(letters.length));
            } while (flags[index]);
            chs[i] = letters[index];
            flags[index] = true;
        }
```

```
        String code=Arrays.toString(chs);
        System.out.println(code);
    }
}
```

1.46.有这样一类数字，它们顺着看和倒着看是相同的数，例如：121、656、2332 等，这样的数字就称为回文数字。编写一个 Java 程序，判断从键盘接收的数字是否为回文数字？

参考答案：

参考答案的代码如下所示：

```
public class Q046 {
 public static void main(String[] args) {
     Scanner scanner = new Scanner(System.in);
     System.out.println("请输入数字：");

     long inputValueLong = scanner.nextLong();
     long temp = inputValueLong;
     long reverseLong = 0L;
     while (inputValueLong != 0) {
         reverseLong = reverseLong * 10 + inputValueLong % 10;
         inputValueLong = inputValueLong / 10;
     }
     if (reverseLong == temp) {
         System.out.println("你输入的是回文数");
     } else {
         System.out.println("你输入的不是回文数");
     }
 }
}
```

1.47.编写程序输出 9*9 乘法口诀？

参考答案：

参考答案的代码如下所示：

```
public class Q047 {
 public static void main(String[] args) {
     for (int i = 1; i <= 9; i++) {
         for (int j = 1; j <= i; j++) {
             System.out.print(i + "*" + j + "=" + (i * j) + "\t");
         }
         System.out.print("\n");
     }
 }
}
```

1.48.有五个人坐在一起，问第 5 个人多少岁？他说比第 4 个人大 2 岁。问第 4 个人多少岁？他说比第 3 个人大 2 岁。问第 3 个人多少岁？他说比第 3 个人大 2 岁。问第 2 个人多少岁？他说比第 1 个人大 2 岁。最后问第 1 个人多少岁？他说是 10 岁。请问第 5 个人多大？（用递归方式实现）？

参考答案：

参考答案的代码如下所示：

```
public class Q048 {
 public static int getAge(int a) {
     if (a == 1)
         return 10;
     else
         return getAge(a - 1) + 2;
 }

 public static void main(String[] args) {
     int a = 5;
     int b = getAge(a);
     System.out.println(b);
 }
}
```

1.49.公鸡每只 3 元，母鸡每只 5 元，小鸡三只一元，问 100 元买 100 只鸡有几种买法，请编程？

参考答案：

参考答案的代码如下所示：

```
public class Q049 {
 public static void main(String[] args) {
     int count = 0;
     for (int i = 0; i <= 100 * 3; i += 3) {// 买小鸡的只数只能是三的倍数
         for (int j = 0; j <= 100 / 3; j++) {// 买公鸡的只数
             for (int k = 0; k <= 100 / 5; k++) {// 买母鸡的只数
                 if (i / 3 + j * 3 + k * 5 == 100 // 买鸡花了 100 元
                         && i + j + k == 100) { // 买了 100 只鸡
                     count++;
                     System.out.println("小" + i + "  公" + j + "  母" + k);
                 }
             }
         }
     }
     System.out.println("共" + count + "种买法！");
 }
}
```

1.50.编程：有 1020 个西瓜，第一天卖一半多两个，以后每天卖剩下的一半多两个，问几天以后能卖完？

参考答案：

参考答案的代码如下所示：

```
public class Q050 {
 public static void main(String[] args) {
     int i;
     int num = 1020;
     for (i = 1;; i++) {
         num = num - (num / 2 + 2);// 当天卖后剩下多少个
         System.out.println("第" + i + "天后剩下" + num + "个");
         if (num == 0)
             break;
     }
     System.out.println(i + "天卖完！");
 }
}
```

2. OOP

2.1.什么是 OOAD？OOAD 怎么实现？

参考答案：

OOAD（Object Orient Analysis Design，面向对象的分析与设计）是现代软件企业广为采用的一项有效技术。OOAD 方法要求在设计中要映射现实世界中指定问题域中的对象和实体，例如：顾客、汽车和销售人员等。这就需要设计要尽可能地接近现实世界，即以最自然的方式表述实体。

使用 UML 建模语言创建系统的分析模型与设计模型是 OOAD 实现的主要手段。

2.2.请分述类及对象的创建模式？

参考答案：

总共有五种创建模式，分别是单例模式，工厂方法模式，抽象工厂模式，建造者模式，原型模式，其中工厂方法模式是类创建模式，其余四种是对象创建模式。下面是对各个模式的介绍。

单例模式（Singleton ，对象创建模式）：单例模式确保某一个类只有一个实例，而且自行实例化并向整个系统提供这个实例。

工厂方法模式（Factory Method， 类创建模式）：核心工厂类不再负责所有产品的创建，而是将具体创建的工作交给子类去做，成为一个抽象工厂角色，仅负责给出具体工厂类必须实现的接口，而不接触哪一个产品类应当被实例化这种细节。

抽象工厂模式（Abstract Factory，对象创建模式）：抽象工厂模式是指当有多个抽象角色时，使用的一种工厂模式。抽象工厂模式可以向客户端提供一个接口，使客户端在不必指定产品的具体的情况下，创建多个产品族中的产品对象。简单来说是创建一组相关或相互依赖的复杂对象。

建造模式（Builder，对象创建模式）：将产品的内部表象和产品的生成过程分割开来，从而使一个建造过程生成具有不同的内部表象的产品对象。建造模式使得产品内部表象可以独立的变化，客户不必知道产品内部组成的细节。建造模式可以强制实行一种分步骤进行的建造过程。

原型模式（Prototype，对象创建模式）：通过给出一个原型对象来指明所要创建的对象的类型，然后用复制这个原型对象的方法创建出更多同类型的对象。原始模型模式允许动态的增加或减少产品类，产品类不需要非得有任何事先确定的等级结构，原始模型模式适用于任何的等级结构。缺点是每一个类都必须配备一个克隆方法。

2.3.什么时候需要改写 hashCode 方法？为什么？

参考答案：

在改写 equals 方法的时候总是要改写 hashCode 方法。如果不这样的话，就会违反 Object 类的 hashCode 方法的通用约定，导致这个类无法与所有基于散列值的集合类结合在一起正常工作，包括 HashMap,HashSet 和 Hashtable。

2.4.继承和重载的区别？

参考答案：

Java 的继承是子类对象继承父类对象的成员属性和成员方法，只允许单继承。

在继承的过程中可以实现方法的重写（也称为覆盖），即子类定义一个方法，覆盖从父类那里继承来的同名的方法。每当子类对象调用该方法时都是子类自己定义的方法，只有使用 super 关键字或父类名为前缀时，才会调用父类原来的方法。方法覆盖时，子类应与父类有完全相同的方法名，返回值类型和参数列表，子类中的覆盖方法不能使用比父类中被覆盖的方法更严格的访问权限。

方法重载要求是有两个或多个方法名相同，只是方法参数列表不同的方法，它对返回值类型没有限定。

2.5.请在下表对应的作用域可以访问的位置上打上“ √”不能访问的打上“×”

类的作用域

作用域	当前类	同一 package	子孙类	其它
Public				
Protected				
Default				
Private				

参考答案：

作用域	当前类	同一 package	子孙类	其它
Public	√	√	√	√
Protected	√	√	√	×
Default	√	√	×	×
Private	√	×	×	×

2.6.abstract class 和 interface 有什么区别？

参考答案：

abstract class 和 interface 的区别如下：

首先，从语法角度来说。abstract class 方法中可以有自己的数据成员，也可以有非 abstract 的成员方法，并赋予方法的默认行为，而在 interface 方式中一般不定义成员数据变量，所有的方法都是 abstract，方法不能拥有默认的行为。

然后，从编程的角度来说。abstract class 在 Java 语言中表示的是一种继承关系，一个类只能使用一次继承关系，而一个类可以实现多个 interface。

最后，从问题域角度来说。abstract class 在 Java 语言中体现了一种继承关系，要想使得继承关系合理，父类和派生类之间必须存在"is a"关系，即父类和派生类在概念本质上应该是相同的。对于 interface 来说则不然，并不要求 interface 的实现者和 interface 定义在概念本质上是一致的，仅仅是实现了 interface 定义的契约而已。

2.7.面向对象的 3 个基本特征？

参考答案：

面向对象的 3 个基本特征为：封装、继承、多态。

2.8.面向对象的特征有哪些方面？简单描述对这些特征的理解？

参考答案：

面向对象的特征主要有以下几个方面：

1) 抽象：抽象就是忽略一个主题中与当前目标无关的那些方面，以便更充分地注意与当前目标有关的方面。抽象并不打算了解全部问题，而只是选择其中的一部分，暂时不用部分细节。抽象包括两个方面：一是抽象的属性,二是抽象的方法。

2) 继承：继承是一种联结类的层次模型，并且允许和鼓励类的重用，它提供了一种明确表述共性的方法。对象的一个新类可以从现有的类中派生，这个过程称为类继承。新类继承了原始类的特性，新类称为原始类的派生类（子类），而原始类称为新类的基类（父类）。派生类可以从它的基类那里继承方法和实例变量，并且类可以修改或增加新的方法使之更适合特殊的需要。

3) 封装：封装是把过程和数据包围起来，对数据的访问只能通过已定义的界面。面向对象设计始于这个基本概念，即现实世界可以被描绘成一系列完全自治、封装的对象,这些对象通过一个受保护的接口访问其他对象。

4) 多态性：多态性是指允许不同类的对象对同一消息作出响应，是类的多种表现方式，比如同名不同参。多态性包括参数化多态性和包含多态性，具体表现在重载和重写。多态性语言具有灵活、抽象、行为共享、代码共享的优势，很好的解决了应用程序函数同名问题。

2.9.一个 subclass 怎样调用 superclass 中的方法（myMethod）和构造函数？

参考答案：

子类 subclass 实例可以直接调用父类 superclass 未被重写的方法；若子类重写父类方法，可以在子类内通过“super.方法名()”的形式调用父类方法，例如：super.myMethod();。

创建子类实例时默认调用父类无参的构造方法，若调用父类有参的构造方法可以在子类构造方法第一行通过“super(参数)”形式调用。

2.10.构造器 Constructor 是否可被 override？

参考答案：

构造器 Constructor 不能被继承，因此不能被重写 Override，但可以被重载 Overload。

2.11.是否可以继承 String 类？

参考答案：

String 类是 final 类，故不可以继承。

2.12.阐述一下 static 关键字的作用？

参考答案：

static 表示“静态”的意思，用来修饰成员变量和成员方法，也可以形成静态代码块。只要这个类被加载，Java 虚拟机就能根据类名在运行时数据区的方法区内找到它们。因此，static 成员可以在它的任何对象创建之前访问，无需引用任何对象。

1） 修饰成员变量。用 static 修饰的成员变量不属于对象的数据结构；static 变量是属于类的变量，通常可以通过类名来引用 static 成员；static 成员变量和类的信息一起存储在方法区，而不是在堆中，一个类的 static 成员变量只有“一份”，无论该类创建了多少对象。

2） 修饰成员方法。static 修饰的方法则不需要针对某些对象进行操作，其运行结果仅仅与输入的参数有关，调用时直接用类名引用。由于 static 在调用时没有具体的对象，因此在 static 方法中不能对非 static 成员（对象成员）进行访问。static 方法的作用在于提供一些“工具方法”和“工厂方法”等。

3） static 块：属于类的代码块，在类加载期间执行的代码块，只执行一次，可以用来在软件中加载静态资源。

2.13.解释 Java 关键字的含义及用法

1) abstract
2) extends
3) final / finally
4) implements
5) import
6) instanceof
7) synchronized
8) throw/throws

参考答案：

1) abstract：抽象，修饰类和方法。含有 abstract 方法的类是 abstract 类，abstract 方法主要是为了让子类继承实现。

2) extends：继承，子类通过继承父类来添加变量或方法，或者覆盖父类的方法；子接口继承父接口来添加方法。

3) final / finally：final 用于定义常量。finally 用来执行一段代码，不管在前面定义的 try 语句中是否有异常或运行时错误发生。

4) implements：在类的声明中是可选的，用来指明当前类实现的接口。

5) import：在源文件的开始部分指明后面将要引用的一个类或整个包。

6) instanceof：判断其指向对象的实际类型。

7) synchronized：修饰方法或方法块。防止多个线程同时访问这个类中的 synchronized 块。

8) throw/throws：throw 允许用户抛出一个 exception 对象或者任何实现 throwable 的对象。 throws 用在方法的声明中来说明哪些异常这个方法是不处理的，而是提交到程序的更高一层。

2.14.静态变量和实例变量的区别？

参考答案：

静态变量也称为类变量，归全类共有，它不依赖于某个对象，可通过类名直接访问，而实例变量必须依存于某一实例，只能通过对象才能访问到它。

2.15.GC 是什么？为什么要有 GC?

参考答案：

GC 是垃圾收集的意思（Gabage Collection），内存处理是编程人员容易出现问题的地方，忘记或者错误的内存回收会导致程序或系统的不稳定甚至崩溃，Java 提供的 GC 功能可以自动监测对象是否超过作用域从而达到自动回收内存的目的。Java 程序员编程的时候不用考虑变量不用时释放内存，Java 虚拟机可以自动判断出并收集到垃圾。

2.16.简述垃圾回收的优点和原理。并例举 2 种回收机制？

参考答案：

Java 语言中一个显著的特点就是引入了垃圾回收机制，使 c++程序员最头疼的内存管理的问题迎刃而解，它使得 Java 程序员在编写程序的时候不再需要考虑内存管理。由于有个垃圾回收机制，Java 中的对象不再有“作用域”的概念，只有对象的引用才有“作用域”。垃圾回收可以有效的防止内存泄露，有效的使用可以使用的内存。

垃圾回收器通常是作为一个单独的低级别的线程运行，不可预知的情况下对内存堆中已经死亡的或者长时间没有使用的对象进行清除和回收，程序员不能实时的调用垃圾回收器对

某个对象或所有对象进行垃圾回收。

列举两种垃圾回收机制如下：

1) 增量收集器

增量收集器把堆栈分为多个域，每次仅从一个域收集垃圾。这会造成较小的应用程序中断。

2) 分代收集器

这种收集器把堆栈分为两个或多个域，用以存放不同寿命的对象。JVM 生成的新对象一般放在其中的某个域中。过一段时间，继续存在的对象将获得使用期并转入更长寿命的域中。分代收集器对不同的域使用不同的算法以优化性能。

2.17.接口是否可继承接口？抽象类是否可实现(implements)接口？抽象类是否可继承实体类(concrete class)？

参考答案：

各问题的参考答案如下：

1) 接口可以继承接口；

2) 抽象类可以实现(implements)接口；

3) 抽象类可继承实体类。

2.18.public class MyString extends String{}有什么错？

参考答案：

String 类是 final 类，不能被继承。

2.19.子类A继承父类 B，A a = new A();则父类 B 构造函数、父类 B 静态代码块、父类 B 非静态代码块、子类A构造函数、子类 A 静态代码块、子类A非静态代码块 执行的先后顺序是？

参考答案：

父类 B 静态代码块->子类 A 静态代码块->父类 B 非静态代码块->父类 B 构造函数->子类 A 非静态代码块->子类 A 构造函数。

2.20.简述 “ 类(class) ” 、 “ 类库 (class library)” 、 “ 包(package) ” 、 “jar 文件 ” 这四个概念间的联系？

参考答案：

1) 类（class）实际上是对某种类型的对象定义变量和方法的原型。它表示对现实生活中一类具有共同特征的事物的抽象。

2) 为了更好地组织类，Java 提供了包机制。包（package）是类的容器，用于分隔类

名空间。

3) 类库 (class library)是用来实现各种功能的类的集合。

4) jar 文件用于发布和使用类库，可被编译器和 JVM 直接使用。

2.21.面向对象的程序在运行时会创建多个对象，这些对象之间通常可以相互 “ 发送消息 ” ，谈谈你对 “ 对象之间发送消息 ” 这句话的理解，并编写几句 Java 示例代码展示对象之间发送消息的具体编程实现方法？

参考答案：

调用方法的行为通常被称为发送消息给对象。代码示例如下：

```
public boolean login(String userName,String pwd){
//实现登录的代码
}
//处理用户请求的代码
public String action(){
  String userName=request.getParameter("userName");
  String pwd=request.getParameter("pwd");
  //调用 login 方法传递用户名，密码信息进入方法
  login(userName,pwd);
   return "success";
}
```

上述代码中 ,在 action 方法中 ,调用 login 方法并传递用户名 ,密码信息进入该方法，即将消息传递到了 login 方法的内部。

2.22.重写 Clazz 类的 equals 方法：

```
public class Clazz{
private int id;
public int getId(){
      return id;
  }
  public void setId(int id){
      this.id = id;
  }
  @Override
  public boolean equals(Object obj) {

  }
}
```

参考答案：

在 Clazz 类中重写 equals 方法的代码如下所示：

```
public class Clazz {
 private int id;
 public int getId() {
     return id;
 }
 public void setId(int id) {
```

```
        this.id = id;
    }
    @Override
    public boolean equals(Object obj) {
        if (this == obj)
            return true;
        if (obj == null)
            return false;
        if (getClass() != obj.getClass())
            return false;
        Clazz other = (Clazz) obj;
        if (id != other.id)
            return false;
        return true;
    }
}
```

2.23.请看下列代码：

Test1 类：

```
package com.yinhai.test;
public class Test1 {
 public static String method() {
     return "hello world!";
 }
 public static void main(String arg[]) {
     System.out.println(method());
 }
}
```

Test2 类：

```
package com.yinhai.test;
public class Test2 {
 public static void main(String arg[]) {
     System.out.println(Test1.method());
 }
}
```

Test3 类：

```
package com.yinhai.test.a;
import com.yinhai.test.Test1;
public class Test3 {
 public static void main(String arg[]) {
     System.out.println(Test1.method());
 }
}
```

Test4 类：

```
package com.yinhai.test.a;
import com.yinhai.test.Test1;
public class Test4 extends Test1 {
 public static void main(String arg[]) {
     System.out.println(method());
 }
}
```

将Test1中的method方法的作用域修饰符public以下表第一列的作用域修饰符替换，请在第二、三、四、五列表格中以“√”和“×”标注在各中情况下类Test1、Test2、Test3、Test4是否能正常输出结果。

作用域

作用域修饰符	Test1	Test2	Test3	Test4
public				
protected				
private				
不写				

参考答案：

作用域修饰符	Test1	Test2	Test3	Test4
public	√	√	√	√
protected	√	√	×	√
private	√	×	×	×
不写	√	√	×	×

2.24.heap 和 stack 有什么区别？

参考答案：

栈(stack)与堆(heap)都是Java用来在内存中存放数据的地方，二者区别在于：

1．栈存放基本类型变量和对象引用，当超过作用域后释放；堆存放new出来的对象和数组。

2．堆可以动态地分配内存大小，生存期也不必事先告诉编译器，Java的垃圾收集器会自动收走这些不再使用的数据。存在栈中的数据大小与生存期必须是确定的，缺乏灵活性。栈的存取速度比堆要快，仅次于直接位于CPU中的寄存器。堆由于要在运行时动态分配内存，存取速度较慢。

3．栈数据可以共享，如字面量3等；堆不可以。

4．栈是一种线形集合，其添加和删除元素的操作应在同一段完成，栈按照后进先出的方式进行处理；堆地址是不连续的，可随机访问。

3. JAVA SE

3.1.int 和 Integer 有什么区别?

参考答案：

Java 语言是一个面向对象的语言，但是 Java 中的基本数据类型却不是面向对象的，这在实际使用时存在很多的不便，为了解决这个不足，在设计类时为每个基本数据类型设计了一个对应的类进行代表，这样八个和基本数据类型对应的类统称为包装类(Wrapper Class)

Java 语言提供两种不同的类型：引用类型和基本数据类型；int 是 Java 语言中的原始数据类型，Integer 是 Java 为 int 提供的包装类。Java 为每一个基本数据类型提供了封装类，其中，基本数据类型包括 byte、short、int、long、float、double、char、boolean 这八个基本数据类型，这八个基本数据类型对应的封装类为 Byte、Short、Integer、Long、Float、Double 、Character、Boolean。

在这八个类名中，除了 Integer 和 Character 类以外，其它六个类的类名和基本数据类型一致，只是类名的第一个字母大写即可。对于包装类说，这些类的用途主要包含两种：

1） 作为和基本数据类型对应的类类型存在，方便涉及到对象的操作；

2） 包含每种基本数据类型的相关属性如最大值、最小值等，以及相关的操作方法。

3.2.请看以下示例代码：

```
String s1 = "Hello";
String s2 = "Hello";
System.out.println(s1 == s2);  // 输出： true
String s3 = new String("Hello");
String s4 = new String("Hello");
System.out.println(s3 == s4);  // 输出： false
```

请解释一下为什么上述代码中 “System.out …” 两句代码输出完全不同的结果 ？

参考答案：

“Hello”是一个字符串常量，它存放在内存的常量池中。在第一行代码执行后创建此对象，而创建 s2 对象的时候首先会到常量池中检查是否存在字符串”Hello”，如果存在，则直接引用已存在的对象。

s3 和 s4 所引用的对象是通过 new 关键字创建的，会在堆中分别创建。这两个引用分别指向不同的两个对象。“== ”只有在两个变量引用指向同一个对象时才返回 true，因此出现如上结果。

3.3.如何将 int 型变量转换成 String 类型变量？如何将 String 类型变量转换成 int 类型变量？

参考答案：

int 型变量转换成 String 类型变量，可以使用 String 类的静态方法 valueOf，请看如下代码示例：

```
//将 int 类型变量 i 转成 String 类型
String s=String.valueOf(i);
```

将 String 类型变量转换成 int 类型变量使用 Integer 类的静态方法 parseInt，请看如下代码示例：

```
//将 String 类型变量 s 转变成 int 类型
int i=Integer.parseInt(s);
```

3.4.加法运算符 “+” 可以施加于原始数值类型（比如 int ）的变量，但我们发现一些对象类型（比如 Integer ）的变量，也支持 “+” 运算，请看如下代码：

```
Integer v1 = 100;
Integer v2 = 200;
System.out.println(v1 + v2 );  // 输出： 300
```

这看上去好象 Integer 类型重载了 “+” 运算符，一些编程语言比如 C++ 可以重载运算符，但 Java 本身并不支持这一特性。依你的理解或猜测， Java 采用什么方法处理两个 Integer 对象直接 “ + ” 的语句？

参考答案：

Java5.0 以后提供了自动装箱和自动拆箱功能，如下代码：

```
Integer v1 = 100;
```

实际上系统为我们执行的代码为：

```
Integer i = new Integer(100);
```

上述代码即实现了基本数据类型的自动装箱功能，然后，执行如下代码：

```
System.out.println(v1 + v2 );
```

实际上系统为我们执行的代码为：

```
System.out.println(v1.intValue() + v2.intValue );
```

上述代码即实现了自动拆箱，然后，以 int 类型计算值的方式计算相加后的结果。

3.5.通过正则表达式判断一个字符串： 首位是字母，后面可以是字母、数字或者下划线，总长度 6-18 位 ？

参考答案：

判断一个字符串首位是字母，后面可以是字母、数字或者下划线，总长度 6-18 位的正则表达式如下：

```
^[a-zA-Z]\w{5,17}$
```

3.6.请写出中国身份证号的正则表达式 ？

参考答案：

中国身份证号的正则表达式如下所示：

```
^\d{15}(\d{2}[0-9xX])?$
```

3.7.Java 中 字符串转换到数字的 API ？

参考答案：

将 String 类型变量 s 转换成 byte 类型、Short 类型、Integer 类型、Long 类型、Float 类型、Double 类型的 API 如下:

```
Byte.parseByte(s);
Short.parseShort(s);
Integer.parseInt(s);
Long.parseLong(s);
Float.parseFloat(s);
Double.parseDouble(s);
```

3.8.Anonymous Inner Class (匿名内部类) 是否可以 extends(继承)其它类？是否可以 implements(实现)interface(接口)?

参考答案：

匿名内部类可以继承其他类或实现其他接口，在 Java 编程中常用此方式。

3.9.内部类可以引用他包含类的成员吗？有没有什么限制？

参考答案：

完全可以。如果不是静态内部类，那没有什么限制！

如果你把静态内部类当作内部类的一种特例，那在这种情况下不可以访问外部类的普通成员变量，而只能访问外部类中的静态成员，例如，下面的代码：

```
class Outer{
 static int x;
 static class Inner{
     void test(){
         syso(x);
     }
 }
```

3.10.内部类的实现方式？

参考答案：

本题以成员内部类为例说明内部类的实现方式，示例代码如下：

```
public class OuterClass {
 private class InterClass {
     public InterClass() {
         System.out.println("InterClass Create");
     }
 }
 public OuterClass() {
     InterClass ic = new InterClass();
     System.out.println("OuterClass Create");
 }
 public static void main(String[] args) {
     OuterClass oc = new OuterClass();
 }
}
```

以上代码的输出结果为:

```
InterClass Create
OuterClass Create
```

3.11.Checked Exception 和 Unchecked Exception 是什么以及它们的区别？

参考答案：

Checked exception: 这类异常都是 Exception 的子类 。

Unchecked exception: Error 和 RuntimeException 及其子类是 unchecked exception.

二者的区别在于,Unchekced Exception 中的 Error 是 Java 自己的错误或者诸如内存耗尽等严重错误，是不可抗拒的，显然没有捕捉的必要，而且也没有办法捕捉。 unchekced Exception 中的 RuntimeException 是你的程序有逻辑错误 ,是程序员应该积极避免其出现的异常。比如 NullPointerException 等，完全是程序员马虎出的错。当遇到这种错误时，Java 将这个错误自动捕捉到 ,比如显示到 concole 里 ,然后继续运行。而 checked exception 如果不捕捉则会导致程序终止，checked exception 是需要强制 catch 的异常，你在调用这个方法的时候，你如果不捕获这个异常，那么编译器就会报错，比如说我们读写文件的时候会捕获 IOException，执行数据库操作会捕获 SQLException 等。

3.12.在 Java 语言中，如何引发异常?对异常处理的两条途径是什么？

参考答案：

在 Java 语言中，可以使用 throw 关键字来引发异常。

对异常处理的两条途径为：

1）使用 throws 抛出异常。

2）使用 try-catch 语句捕获异常。

3.13.异常(Exception)分几种类型？有什么区别？写出几个常见异常(若记不得异常的英文名,写中文名亦可)

参考答案：

异常(Exception)分为两类：运行时异常和非运行时异常。

运行时异常和非运行时异常的区别是运行时异常（RuntimeException）又被称为非检查异常，非运行时异常指的是检查异常。运行时异常是指直接或间接继承 RuntimeException 的异常。非运行时异常是指除了 RuntimeException 以外的其它异常，是你的程序有逻辑错误，程序员应该积极避免其出现的异常。Java 编译器要求方法必须声明抛出可能发生的非运行时异常，但是并不要求必须声明抛出未被捕获的运行时异常。

常见的运行时异常：

1）NullPointerException（空指针异常）：当操作一个空引用时会出现此异常。

2）NumberFormatException（数字格式化异常）：试图将字符串转换成一种数值类型，但该字符串不能转换为适当格式时，抛出该异常。

3）ClassCastException（类型转换异常）：强制类型转换类型不匹配时出现此异常。

4）ArrayIndexOutOfBoundsException（数组下标越界异常）：当使用一个不存在的数组下标时出现此异常。

5）ArithmeticException（数学异常）：当出现异常的运算条件时出现此异常

常见的非运行时异常：

1） SQLException：提供关于数据库访问错的异常。

2） IOException：当发生某种 I/O 异常时，抛出此异常。

3）ClassNotFoundException :当应用程序试图使用 Class 类中的 forName 方法、loadClass 方法时，抛出该异常。

3.14.Java 中的异常处理机制的简单原理？

参考答案：

当 Java 程序违反了 Java 的语义规则时，Java 虚拟机就会将发生的错误表示为一个异常。违反语义规则包括 2 种情况，一种是 Java 类库内置的语义检查，例如：数组下标越界,会引发 IndexOutOfBoundsException;访问 null 的对象时会引发 NullPointerException。另一种情况是 Java 允许程序员扩展这种语义检查，程序员可以创建自己的异常，并自由选

择在何时用 throw 关键字引发异常。

3.15.try {}里有一个 return 语句，那么紧跟在这个 try 后的 finally {}里的 code 会不会被执行，什么时候被执行，在 return 前还是后？

参考答案：

会执行，在 return 前执行。

3.16.当使用多个 catch 语句捕获多个异常时，Java 规定捕获 Exception 的 catch 语句必须排在最后，如下所示：

```
try { ..... }
catch(ClassCastException ex){ ..... }
catch(NumberFormatException ex){ ..... }
catch(Exception ex){ ..... }  // 此句必须放在最后！
```

为什么会有这个限制？谈谈你的理解。

参考答案：

Exception 是异常的父类。如果没有把该语句 catch(Exception ex){ ..... }放在最后面，那么 try 块中产生的异常不可能被 catch(Exception ex){ ..... }之后的 catch 语句捕获，也就是说 catch(Exception ex){ ..... }之后的 catch 语句永远不会执行到。Java 编译器不允许这样的情况，会出现编译错误，因此如果要捕获的异常有继承关系，则子类在前，父类在后。

3.17.Java 语言如何进行异常处理，关键字：throws,throw,try,catch,finally 分别代表什么意义？在 try 块中可以抛出异常吗？

参考答案：

Java 通过面向对象的方法进行异常处理，把各种不同的异常进行分类，并提供了良好的接口。在 Java 中，每个异常都是一个对象，它是 Throwable 类或其它子类的实例。当一个方法出现异常后便抛出一个异常对象，该对象中包含了异常信息，调用这个对象的方法可以捕获到这个异常并进行处理。Java 的异常处理是通过 5 个关键词来实现的：try、catch、throw、throws 和 finally。

try 用来指定一块预防所有“异常”的程序；catch 子句紧跟在 try 块后面，用来指定你想要捕捉的“异常”的类型；

throw 语句用来明确地抛出一个“异常”；

throws 用来标明一个成员方法可能抛出的各种“异常”；

finally 为确保一段代码不管发生什么“异常”都被执行一段代码；

可以在一个成员方法调用的外面写一个 try 语句，在这个成员方法内部写另一个 try 语

句保护其他代码。每当遇到一个 try 语句，“异常”的框架就放到堆栈上面，直到所有的 try 语句都完成。如果下一级的 try 语句没有对某种“异常”进行处理，堆栈就会展开，直到遇到有处理这种“异常”的 try 语句。

3.18.final, finally, finalize 的区别？

参考答案：

Final 是一个修饰符。如果一个类被声明为 final，意味着它不能再派生出新的子类，不能作为父类被继承，因此一个类不能既被声明为 abstract 的，又被声明为 final 的；将变量或方法声明为 final，可以保证它们在使用中不被改变；被声明为 final 的变量必须在声明时给定初值，而在以后的引用中只能读取，不可修改；被声明为 final 的方法也同样只能使用，不能重载。

Finally 用在异常处理时提供 finally 块来执行任何清除操作；如果抛出一个异常，那么相匹配的 catch 子句就会执行，然后控制就会进入 finally 块（如果有的话）。

finalize 是方法名，Java 技术允许使用 finalize() 方法在垃圾收集器将对象从内存中清除出去之前做必要的清理工作。这个方法是由垃圾收集器在确定这个对象没有被引用时对这个对象调用的。它是在 Object 类中定义的，因此所有的类都继承了它。子类覆盖 finalize() 方法以整理系统资源或者执行其他清理工作

3.19.阐述一下 IndexOutOfBoundsException 通常在什么情况下发生？

参考答案：

指示某排序索引（例如对数组、字符串等）超出范围时抛出。例如：当使用一个不存在的数组下标时出现 ArrayIndexOutOfBoundsException。

3.20.阐述一下 `NullPointerException` 通常在什么情况下发生？

参考答案：

通常在操作一个空引用时会产生 NullPointerException，例如：空引用调用属性或方法时。

3.21.阐述一下 ClassCastException 通常在什么情况下发生？

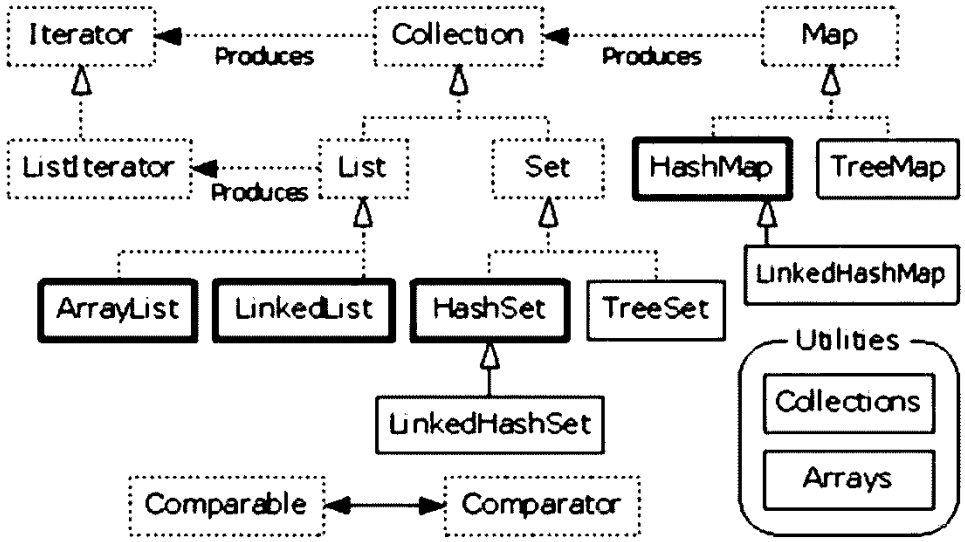
参考答案：

强制类型转换类型不匹配时出现此异常。例如：把某一对象强转其他类型，如果该对象并非该类的实例，就会发生 ClassCastException。

3.22.画出集合的框架图？

参考答案：

集合框架的结构下图所示。



3.23.编写一段程序，用来创建和迭代一个 List？

参考答案：

```
public static void main(String[] args) {
    List<String> names = new ArrayList<String>();
    names.add("Tom");
    names.add("Jerry");
    names.add("Black");
    names.add("Andy");
    names.add("Lee");
    for (int i = 0; i < names.size(); i++) {
        String str=names.get(i);
        System.out.println(str);
    }
}
```

3.24.编写一段程序，用来创建和迭代一个 Set？

参考答案：

```
public static void main(String[] args) {
    Set<String> set = new HashSet<String>();
    set.add("aaa");
    set.add("bbb");
    set.add("ccc");
    set.add("ddd");
    Iterator<String> ite = set.iterator();
    while (ite.hasNext()) {
        String egg = ite.next();
        System.out.println(egg);
    }
}
```

3.25.编写一段程序，用来创建和迭代一个 Map？

参考答案：

```
public static void main(String[] args) {
    Map<String,Integer> map = new HashMap<String,Integer>();
    map.put("zhangsan", 89);
    map.put("lisi", 98);
    map.put("wangwu", 56);

    Set<Entry<String,Integer>> entrySet = map.entrySet();
    for(Entry<String,Integer> entry : entrySet){
        System.out.println(entry.getKey()+":"+entry.getValue());
    }
}
```

3.26.Collection 和 Collections 的区别？

参考答案：

Collection 是 java.util 下的接口，它是各种集合的父接口，继承于它的接口主要有 Set 和 List；Collections 是个 java.util 下的类，是针对集合的帮助类，提供一系列静态方法实现对各种集合的搜索、排序、线程安全化等操作。

3.27.说说 java 集合类：HashMap 是如何设计的？ 是如何解决什么冲突的？

参考答案：

1）HashMap 是基于哈希表的 Map 接口的非同步实现。在 Java 编程语言中，最基本的结构就是两种，一个是数组，另外一个是模拟指针（引用），所有的数据结构都可以用这两个基本结构来构造的，HashMap 也不例外。HashMap 实际上是一个“链表的数组”的数据结构，每个元素存放链表头结点的数组，即数组和链表的结合体。HashMap 底层就是一个数组结构，数组中的每一项又是一个链表。当新建一个 HashMap 的时候，就会初始化一个数组。Entry 就是数组中的元素，每个 Map.Entry 其实就是一个 key-value 对，它持有一个指向下一个元素的引用，这就构成了链表。

2） HashMap 的存储。当我们往 HashMap 中 put 元素的时候，先根据 key 的 hashCode 重新计算 hash 值，根据 hash 值得到这个元素在数组中的位置（即下标），如果数组该位置上已经存放有其他元素了，那么在这个位置上的元素将以链表的形式存放，新加入的放在链头，最先加入的放在链尾。如果数组该位置上没有元素，就直接将该元素放到此数组中的该位置上。

3）HashMap 的读取。从 HashMap 中 get 元素时，首先计算 key 的 hashCode，找到数组中对应位置的某一元素，然后通过 key 的 equals 方法在对应位置的链表中找到需要的元素。

4） HashMap 的 resize（rehash）。当 HashMap 中的元素越来越多的时候，hash 冲突的几率也就越来越高，因为数组的长度是固定的。所以为了提高查询的效率，就要对 HashMap 的数组进行扩容，数组扩容这个操作也会出现在 ArrayList 中，这是一个常用的

操作，而在 HashMap 数组扩容之后，最消耗性能的点就出现了：原数组中的数据必须重新计算其在新数组中的位置，并放进去，这就是 resize。那么 HashMap 什么时候进行扩容呢？当 HashMap 中的元素个数超过数组大小*loadFactor 时，就会进行数组扩容，loadFactor 的默认值为 0.75，这是一个折中的取值。也就是说，默认情况下，数组大小为 16，那么当 HashMap 中元素个数超过 16*0.75=12（这个值就是代码中的 threshold 值，也叫做临界值）的时候，就把数组的大小扩展为 2*16=32，即扩大一倍，然后重新计算每个元素在数组中的位置，而这是一个非常消耗性能的操作，所以如果我们已经预知 HashMap 中元素的个数，那么预设元素的个数能够有效的提高 HashMap 的性能。

3.28.简述 HashTable 原理，并说明它和 HashMap 区别？

参考答案：

Hashtable 原理：Hashtable 是基于哈希表的实现。通过使用 put(Object key,Object value)方法把两个对象进行关联，需要时用 get(Object key)取得与 key 关联的值对象。还可以查询某个对象的索引值等等。这里的 get 方法查找一个对象时与 Vector 中的 get 方法在内部实现时有很大不同，在一个 Hashtable 中查找一个键对象要比在一个 Vector 中快的多。这是因为 Hashtable 使用了一种哈希表的技术，在 Java 每个对象缺省都有一个通过 Object 的 hashCode()方法获得的哈希码，Hashtable 就是利用这个哈希实现快速查找键对象的。

二者都实现了 Map 接口，是将惟一键映射到特定的值上；主要区别在于：

1．HashMap 没有排序，允许一个 null 键和多个 null 值,而 Hashtable 不允许；

2．HashMap 把 Hashtable 的 contains 方法去掉了，改成 containsvalue 和 containsKey,因为 contains 方法容易让人引起误解；

3．Hashtable 继承自 Dictionary 类，HashMap 是 Java1.2 引进的 Map 接口的实现；

4．Hashtable 的方法是 Synchronize 的，而 HashMap 不是，在多个线程访问 Hashtable 时，不需要自己为它的方法实现同步，而 HashMap 就必须为之提供外同步。Hashtable 和 HashMap 采用的 hash/rehash 算法大致一样，所以性能不会有很大的差异。

3.29.HashMap 和 HashSet 有什么关系？

参考答案：

HashSet 底层是采用 HashMap 实现的，请看如下代码：

```
private transient HashMap<E,Object> map;
```

上述代码是 HashSet 类里面定义的一个私有的成员变量。放进 HashSet 中对象,其实是用这个 HashMap 的 key 来存储的。当调用 HashSet 的 add 方法时，实际上是向

HashMap 中增加了一行(key-value 对)，该行的 key 就是向 HashSet 增加的那个对象，该行的 value 就是一个 Object 类型的常量。

3.30.Vector 和 ArrayList 有什么差异？

参考答案：

ArrayList 与 Vector 的区别主要从二方面来说：

1）同步性：Vector 是线程安全的（同步），而 ArrayList 是线程序不安全的；

2）扩容：当需要扩容时,Vector 默认增长一倍，而 ArrayList 却是一半。

3.31.List,Set,Map 是否继承自 Collection 接口？

参考答案：

List,Set 继承自 Collection 接口，而 Map 没有继承自 Collection 接口。

3.32.List 底层是怎么实现的？双向链表和数组的区别？

参考答案：

在 Java 中 List 是一个接口，继承于 Collection 接口，并定义了添加元素，删除元素，取出元素等对集合操作的抽象方法。

双向链表和数组区别为：

1. 数组必须事先定义固定的长度（元素个数），不能适应数据动态地增减的情况。当数据增加时，可能超出原先定义的元素个数；当数据减少时，造成内存浪费；数组可以根据下标直接存取。

2. 链表动态地进行存储分配，可以适应数据动态地增减的情况，且可以方便地插入、删除数据项；而数组中插入、删除数据项时，需要移动其它数据项，非常繁琐。

3.33.说出 ArrayList,Vector, LinkedList 的存储性能和特性？

参考答案：

ArrayList 和 Vector 都是使用数组方式存储数据，此数组元素数大于实际存储的数据以便增加和插入元素，它们都允许直接按序号索引元素，但是插入元素要涉及数组元素移动等内存操作，所以索引数据快而插入数据慢，Vector 由于使用了 synchronized 方法（线程安全），通常性能上较 ArrayList 差，而 LinkedList 使用双向链表实现存储，按序号索引数据需要进行前向或后向遍历，但是插入数据时只需要记录本项的前后项即可，所以插入速度较快。

3.34.请通过一次循环，清除掉一个 ArrayList 中的每个元素？

参考答案：

```
public static void main(String[] args){
 ArrayList<String> list=new ArrayList<String>();
     list.add("a");
     list.add("a");
     list.add("c");
     list.add("c");
     list.add("b");
     list.add("b");
     list.add("a");
     for (int i = list.size()-1; i >=0; i--) {
         list.remove(i);
     }
     System.out.println(list);
}
```

3.35.设计一个程序基于 Map 泛型完成 10 个英文单词的翻译？

参考答案：

```
public static void main(String[] args){
    Map<String, String> map = new HashMap<String, String>();
     map.put("January", "一月");
     map.put("February", "二月");
     map.put("March", "三月");
     map.put("April", "四月");
     map.put("May", "五月");
     map.put("June", "六月");
     map.put("July", "七月");
     map.put("August", "八月");
     map.put("September", "九月");
     map.put("October", "十月");

     String str = "September";
     if (map.containsKey(str)) {
         System.out.println(map.get(str));
     } else {
         System.out.println("词库里没有这个词");
     }
}
```

3.36.Set 里的元素是不能重复的，那么用什么方法来区分重复与否呢？是用==还是 equals()？它们有何区别？

参考答案：

Set 里的元素是不能重复的，使用 equals 方法和 hashCode 方法来区分重复与否。覆盖 equals 方法、hashCode 方法用来判断两个对象是否为同一对象，而”==”判断地址是否相等,用来决定引用值是否指向同一对象。

3.37.编写一个 list 集合，存储通讯录，(同学姓名和电话)并输出通讯录？

参考答案：

```
public class Record implements Comparable<Record>{
 private int id;
 private String name;
 private String phone;

 public Record() {

 }
 public Record(String name,String phone) {
     this.name=name;
     this.phone=phone;
 }

 public int getId() {
     return id;
 }

 public void setId(int id) {
     this.id = id;
 }

 public String getName() {
     return name;
 }

 public void setName(String name) {
     this.name = name;
 }

 public String getPhone() {
     return phone;
 }

 public void setPhone(String phone) {
     this.phone = phone;
 }

 @Override
 public String toString() {

     return name+":"+phone+" ";
 }
 @Override
 public int compareTo(Record o) {
     return        Collator.getInstance(Locale.CHINESE).compare(this.name,
o.getName());
 }

}
public class CommunicationRecord {
 private int size=0;
 private List<Record> list=new ArrayList<Record>();
 public CommunicationRecord() {

 }
 public int getSize() {
     return size;
 }
 public void add(Record r){
     if(r!=null){
         list.add(r);
         Collections.sort(list);
```

```
        }
    }
    @Override
    public String toString() {
        return list.toString();
    }

    public static void main(String[] args) {
        CommunicationRecord c=new CommunicationRecord();
        c.add(new Record("张三","111"));
        c.add(new Record("李四","222"));
        c.add(new Record("安安","444"));
        c.add(new Record("zz","6667"));
        c.add(new Record("abc","6667"));
        c.add(new Record("bbb","6667"));
        c.add(new Record("ccc","6667"));
        c.add(new Record("fff","6667"));
        c.add(new Record("gg","6667"));
        c.add(new Record("t","6667"));
        c.add(new Record("abc","6667"));
        c.add(new Record("李四","222"));
        c.add(new Record("王五","444"));
        c.add(new Record("赵六","222"));
        c.add(new Record("田七","444"));
        System.out.println(c);

    }
}
```

3.38.有两个 List<Integer>，写一个方法要求合并这两个集合，不能有重复的，不能用集合的 sort 等方法，而且要求中间的数字最大，两边逐渐减小例如(m 是集合的 size)（get(0)<get(m-1),get(1)<get(m-2)）？

参考答案：

```
public static void main(String[] args) {
    List<Integer> list1=new ArrayList<Integer>();
    list1.add(4);
    list1.add(4);
    list1.add(5);
    list1.add(1);
    list1.add(7);
    list1.add(9);

    List<Integer> list2=new ArrayList<Integer>();
    list2.add(4);
    list2.add(8);
    list2.add(5);
    list2.add(11);
    list2.add(17);
    list2.add(9);
    list2.add(2);
    list2.add(3);
    list2.add(19);

    TreeSet<Integer> set=new TreeSet<Integer>(list1);
    set.addAll(list2);
    List<Integer> ss=new ArrayList<Integer>(set);
    List<Integer> list=new ArrayList<Integer>();
    for (int i = 0; i < ss.size(); i=i+2) {
```

```
            list.add(ss.get(i));
        }
        for (int i = (ss.size())/2*2-1; i > 0; i=i-2) {
            list.add(ss.get(i));
        }
        System.out.println(list);
    }
```

3.39.将文件夹 a（包括其下所有子文件夹和文件）复制到文件夹 b 下。要求：使用 10 个线程同时进行，每一个线程独立处理一个文件？

参考答案：

```
import java.io.BufferedInputStream;
import java.io.BufferedOutputStream;
import java.io.File;
import java.io.FileInputStream;
import java.io.FileOutputStream;
import java.io.IOException;

public class Q039 {
 MyThread[] threads = new MyThread[10];
 public int n = 0;

 public Q039() {
     for (int i = 0; i < threads.length; i++) {
         threads[i] = new MyThread();
     }
 }

 public static void main(String[] args) throws Exception {
     new Q039().copyDirectiory(new File("d:\\a"), new File("d:\\b"));
 }

 public void copyDirectiory(File sourceDir, File targetDir) throws Exception
{
     if (!targetDir.exists()) {
         targetDir.mkdirs();
     }
     File[] file = sourceDir.listFiles();
     for (int i = 0; i < file.length; i++) {
         if (file[i].isFile()) {
             File targetFile = new File(targetDir.getAbsolutePath()
                     + File.separator + file[i].getName());
             MyThread t = threads[n++ % 10];
             t.set(file[i], targetFile);
             t.start();
             t.join();
             threads[n % 10] = new MyThread();
         }
         if (file[i].isDirectory()) {
             String dir2 = targetDir.getAbsolutePath() + File.separator
                     + file[i].getName();
             copyDirectiory(file[i], new File(dir2));
         }
     }
 }
}

class MyThread extends Thread {
 private File src;
 private File desc;

 public MyThread() {
```

```
    }

    public void set(File src, File desc) {
        this.src = src;
        this.desc = desc;
    }

    @Override
    public void run() {
        try {
            copyFile(src, desc);
        } catch (IOException e) {
            e.printStackTrace();
        }
    }

    public static void copyFile(File src, File desc) throws IOException {
        if (!desc.exists()) {
            desc.createNewFile();
        }
        BufferedInputStream bis = null;
        BufferedOutputStream bos = null;
        try {
            FileInputStream fis = new FileInputStream(src);
            FileOutputStream fos = new FileOutputStream(desc);
            bis = new BufferedInputStream(fis);
            bos = new BufferedOutputStream(fos);
            int b = -1;
            while ((b = bis.read()) != -1) {
                bos.write(b);
            }
        } finally {
            if (bis != null) {
                bis.close();
            }
            if (bos != null) {
                bos.flush();
                bos.close();
            }
        }
    }
}
```

3.40.请说明什么是线程安全？

参考答案：

如果你的代码所在的进程中有多个线程在同时运行，而这些线程可能会同时运行这段代码；如果每次运行结果和单线程运行的结果是一样的，而且其他的变量的值也和预期的是一样的，就是线程安全的。或者说：一个类或者程序所提供的接口对于线程来说是原子操作或者多个线程之间的切换不会导致该接口的执行结果存在二义性，也就是说我们不用考虑同步的问题。

线程安全问题都是由全局变量及静态变量引起的。若每个线程中对全局变量、静态变量只有读操作，而无写操作，一般来说，这个全局变量是线程安全的；若有多个线程同时执行写操作，一般都需要考虑线程同步，否则的话就可能影响线程安全。

3.41.为什么要使用多线程编程？

参考答案：

多线程就象是人体一样，一直在并行的做许多工作，例如，人可以同时呼吸，血液循环，消化食物的。多线程可以将一个程序划分成多个任务，他们彼此独立的工作，以方便有效的使用处理器和用户的时间。

3.42.启动一个线程是用 run()还是 start()？

参考答案：

启动一个线程是调用 start()方法，使线程所代表的虚拟处理机处于可运行状态，这意味着它可以由 JVM 调度并执行。 这并不意味着线程就会立即运行。run()方法可以产生必须退出的标志来停止一个线程。

3.43.sleep() 和 wait() 有什么区别？

参考答案：

sleep 方法 是线程类（Thread）的方法，导致此线程暂停执行指定时间，把执行机会给其它线程，但是监控状态依然保持，到时后会自动恢复。调用 sleep 不会释放对象锁。wait 是 Object 类的方法，对象调用 wait 方法导致本线程放弃对象锁，进入等待此对象的等待锁定池，只有针对此对象发出 notify 方法（或 notifyAll）后本线程才进入对象锁定池准备获得对象锁进入运行状态。

3.44.sleep 和 yield 的区别？

参考答案：

yield 方法只是让出分配给自己的 CPU 时间片，并且会立刻进入 Runnable 状态参与 CPU 时间的竞争，若程序中没有其它线程，那么该线程马上就会开始往下执行；sleep 会进入 Blocked 状态，等待时间结束事件的发生，然后进入 Runnable 状态参与 CPU 时间的竞争。

3.45.当一个线程进入一个对象的一个 synchronized 方法后，其它线程是否可进入此对象的其它方法？

参考答案：

其它线程只能访问该对象的其它非同步方法，同步方法则不能进入。

3.46.请说出你所知道的线程同步的方法？

参考答案：

实现同步的方法为使用 synchronized 关键字，实现方式：

1) 同步方法。可以是静态和非静态方法，不可以是抽象和接口方法。当一个线程调用这个对象的同步方法，则这个对象的其他所有同步方法将被锁定，不能调用，但可以调用非同步方法。非静态同步方法锁定的是方法所属的主体对象自身。静态同步方法锁定的是主体类对应的 Class 类型的对象，所以静态同步方法只跟所属类的其他静态同步方法相互制约。

2）同步块。锁定一个指定的对象，来对同步块中的代码进行同步。

3.47.同步和异步有何异同，在什么情况下分别使用他们？举例说明？

参考答案：

如果数据将在线程间共享，使用同步编程。例如正在写的数据以后可能被另一个线程读到，或者正在读的数据可能已经被另一个线程写过了，那么这些数据就是共享数据，必须进行同步存取。当应用程序在对象上调用了一个需要花费很长时间来执行的方法，并且不希望让程序等待方法的返回时，则使用异步编程，在很多情况下采用异步途径往往更有效率，比如发送短信。

3.48.简述 synchronized 和 java.util.concurrent.locks.Lock 的异同？

参考答案：

相同点：Lock 和 synchronized 都可以实现了线程同步；

不同点： Lock 有比 synchronized 更精确的线程语义和更好的性能。synchronized 会自动释放锁，而 Lock 一定要求程序员手工释放，并且必须在 finally 从句中释放。

3.49.分别用 Java 中多线程的两种方法实现：输出字符串“Hello”，要求每间隔一秒输出一个字母？

参考答案：

```
public class Test {
 public static void main(String[] args) {
     TestThread1 tt1 = new TestThread1();
     tt1.start();
     Thread tt2 = new Thread(new TestThread2());
     tt2.start();
 }
}
class TestThread1 extends Thread {
 public void run() {
     char[] str = {'H','e','l','l','o'};
     try {
         for(int i = 0;i < str.length;i++) {
             System.out.print(str[i]);
             sleep(1000);
         }
```

```
            System.out.println();
        } catch (Exception e) {
            e.printStackTrace();
        }
    }
}
class TestThread2 implements Runnable {
    public void run() {
        char[] str = {'H','e','l','l','o'};
        try {
            for(int i = 0;i < str.length;i++) {
                System.out.print(str[i]);
                Thread.sleep(1000);
            }
            System.out.println();
        } catch (Exception e) {
            e.printStackTrace();
        }
    }
}
```

3.50.线程池是什么？

参考答案：

当一个程序中若创建大量线程，并在任务结束后销毁，会给系统带来过度消耗资源，以及过度切换线程的危险，从而可能导致系统崩溃。为此我们应使用线程池来解决这个问题。

首先创建一些线程，它们的集合称为线程池，当服务器接受到一个客户请求后，就从线程池中取出一个空闲的线程为之服务，服务结束后不关闭该线程，而是将该线程还回到线程池中。在线程池的编程模式下，任务是提交给整个线程池，而不是直接交给某个线程，线程池在拿到任务后，它就在内部找有无空闲的线程，再把任务交给内部某个空闲的线程，一个线程同时只能执行一个任务，但可以同时向一个线程池提交多个任务。

3.51.写一个简单的 socket 实现聊天功能的程序的例子（不要求界面，只实现数据传输）一个主服务端，一个客户端？

参考答案：

```
public class ServerDemo {
    public static void main(String[] args) throws IOException {
        ServerDemo server = new ServerDemo();
        server.start();
    }

    public void start() throws IOException {
        ServerSocket ss = new ServerSocket(8000);
        System.out.println("等待客户的连接...");
        Socket s = ss.accept();
        System.out.println("客户连接成功:" + s.getInetAddress());
        InputStream in = s.getInputStream();
        OutputStream out = s.getOutputStream();
        Reader r = new Reader(out);
        r.start();
        Writer w=new Writer(in);
        w.start();

    }
```

```
}

public class ClientDemo {
 public static void main(String[] args) throws IOException {
     ClientDemo client = new ClientDemo();
     client.open();
 }

 public void open() throws IOException {
     Socket s = new Socket("localhost", 8000);
     InputStream in = s.getInputStream();
     OutputStream out = s.getOutputStream();
     new Reader(out).start();
     new Writer(in).start();
 }

}

class Reader extends Thread {
 OutputStream out;

 public Reader(OutputStream out) {
     this.out = out;
     setDaemon(true);
 }

 public void run() {
     Scanner s = new Scanner(System.in);
     try {
         while (true) {
             String str = s.nextLine();//
             out.write(str.getBytes());//
             out.write('\n');
             out.flush();
         }
     } catch (IOException e) {
         e.printStackTrace();
     }
 }
}

class Writer extends Thread {
 InputStream in;

 public Writer(InputStream in) {
     this.in = in;
 }

 public void run() {
     try {
         int b;
         while ((b = in.read()) != -1) {
             System.out.write(b);
         }
     } catch (Exception e) {
         e.printStackTrace();
     }
 }
}
```

3.52.JAVA SOCKET 编程，从服务器读几个字符，再写入本地控制台？

参考答案：

```
public class ServerDemo {
```

```
    public static void main(String[] args)
      throws IOException{
      ServerSocket ss = new ServerSocket(8000);
      Socket s = ss.accept();
      OutputStream out = s.getOutputStream();
      out.write("从服务器读几个字符!\n".getBytes());
      out.flush();
      out.close();
    }
}

public class ClientDemo {
 public static void main(String[] args) throws IOException {
     Socket s = new Socket("localhost", 8000);
     InputStream in = s.getInputStream();
     int b;
     while ((b = in.read()) != -1) {
         System.out.write(b);
     }
 }
}
```

3.53.什么是 java 序列化，如何实现 java 序列化？

参考答案：

序列化就是一种用来处理对象流的机制，所谓对象流也就是将对象的内容进行流化。可以对流化后的对象进行读写操作，也可将流化后的对象传输于网络之间。序列化是为了解决在对对象流进行读写操作时所引发的问题；序列化的实现：将需要被序列化的类实现 Serializable 接口，该接口没有需实现的方法，implements Serializable 只是为了标注该对象是可被序列化的，然后使用一个输出流(如 FileOutputStream)来构造一个 ObjectOutputStream(对象流)对象，接着，使用 ObjectOutputStream 对象的 writeObject(Object obj)方法就可以将参数为 obj 的对象写出(即保存其状态)，要恢复的话则用输入流。

3.54.Java 中有几种类型的流？JDK 为每种类型的流提供了一些抽象类以供继承，请说出它们分别是哪些类？

参考答案：

Java 中有字节流、字符流。

字节输入流的父类为 InputStream、字节输出流的父类为 OutputStream；字符输入流的父类为 Reader、字符输出流的父类为 Writer。

3.55.字节流和字符流有何区别？

参考答案：

字节流与字符流主要的区别是他们的处理方式。字节流是最基本的,所有的 InputStream 和 OutputStream 的子类用于处理二进制数据，它是按字节来处理的。而字符流是以字符为单位读写数据的。

3.56.文件和目录（IO）操作：

1) 如何列出某个目录下的所有文件？
2) 如何列出某个目录下的所有子目录？
3) 如何判断一个文件或目录是否存在？

参考答案：

1) 列出 D 盘 demo 目录下的所有文件的示例代码如下所示：

```
File file = new File("d:/demo");
File[] files = file.listFiles();
for(int i=0; i<files.length; i++){
   if(files[i].isFile())
      System.out.println(files[i]);
}
```

2) 列出 D 盘 demo 目录下的所有子目录的示例代码如下:

```
File file = new File(""d:/demo");
File[] files = file.listFiles();
for(int i=0; i<files.length; i++){
   if(files[i].isDirectory())
     System.out.println(files[i]);
}
```

3) 创建 File 对象,调用其 exists()方法即可判断文件和目录是否存在，示例代码如下所示:

```
System.out.println(new File("d:/t.txt").exists());
```

3.57.String s=new String(“xyz”);创建了几个 String Object？

参考答案：

如果在执行这行代码之前没有创建过“xyz”对象（也就是说这行代码之前没有 String str=”xyz”或 String str2=new String(“xyz”);类似的代码），则执行这行代码会创建两个 String Object;如果在执行这行代码之前创建过“xyz”对象，则执行这行代码会创建一个 String Object。

3.58.是否可以继承 String 类？

参考答案：

String 类是 final 的，所以不能够被继承。

3.59.说出“==”和 equals()方法在字串变量操作中的不同？

参考答案：

“==“比较的是引用，相当于比较两个字符串是否是同一个对象。

String 类的 equals()比较的是字符串的内容。

3.60.数组有没有 length()这个方法? String 有没有 length()这个方法？

参考答案：

数组没有 length()方法，有 length 的属性；String 有 length()方法。

3.61.String, StringBuffer StringBuilder 的区别？

参考答案：

String 的长度是不可变的；StringBuffer 的长度是可变的，如果对字符串中的内容经常进行操作，特别是内容要修改时，则使用 StringBuffer，如果最后需要 String，那么使用 StringBuffer 的 toString()方法转换成 String 类型的数据；

StringBuffer 是线程安全的；StringBuilder 是从 JDK 5 开始，为 StringBuffer 该类补充了一个单个线程使用的等价类；通常应该优先使用 StringBuilder 类，因为它支持所有相同的操作，但由于它不执行同步，所以速度更快。

3.62.编写程序将由数字及字符组成的字符串中的数字截取出来被按顺序输出，例如：“ABC137GMNQQ2049PN5FFF”输出结果应该为 01234579？

参考答案：

```
public static void main(String[] args) {
    String str = "ABC137GMNQQ2049PN5FFF";
    StringBuilder stringBuilder = new StringBuilder();
    for (int i = 0; i < str.length(); i++) {
        char c = str.charAt(i);
        if (c >= '0' && c <= '9') {
            stringBuilder.append(c);
        }
    }
    System.out.println(stringBuilder);
    char[] ary = stringBuilder.toString().toCharArray();
    Arrays.sort(ary);
    System.out.println(ary);
}
```

3.63.String s = “abcdefghijklmnopqrstuvwxyz”；编写一段程序，实现”mnop”输出？

参考答案：

```
public static void main(String[] args) {
```

```
        String s = "abcdefghijklmnopqrstuvwxyz";
        int a=s.indexOf("mnop");
        int n="mnop".length();
        String str=s.substring(a, a+n);
        System.out.println(str);
    }
```

3.64.在 Java 代码中有一个字符串 String ss = "abc,efgAA,12q,456,zA,KJD";在不使用 split 方法的情况下写一段代码实现按“,”拆分字符的功能，并输出结果？

参考答案：

```
public static void main(String[] args) {
    String ss = "abc,efgAA,12q,456,zA,KJD";
    List<String> list=new ArrayList<String>();
    int n1=0;
    int n2;
    do{
        n2=ss.indexOf(",", n1);
        if(n2==-1){
            n2=ss.length();
            String str=ss.substring(n1, n2);
            list.add(str);
            break;
        }
        String str=ss.substring(n1, n2);
        list.add(str);
        n1=n2+1;
    }while(n2!=-1);
    System.out.println(list);
}
```

3.65.从文件 IN.DAT 中读取一篇英文文章存入到字符串数组 xx 中，以行为单位对行中以空格或标点符号为分隔的所有单词进行倒排。最后把已处理的字符串(应不含标点符号)仍按行重新存入字符串数组 xx 中，最后把结果 xx 输出到文件 OUT．DAT 中？

参考答案：

```
import java.io.BufferedInputStream;
import java.io.BufferedReader;
import java.io.FileInputStream;
import java.io.FileOutputStream;
import java.io.InputStreamReader;
import java.io.OutputStreamWriter;
import java.io.PrintWriter;

public class Q065 {
 public static void main(String[] args) throws Exception {
    BufferedReader in = new BufferedReader(new InputStreamReader(
            new BufferedInputStream(
                new FileInputStream("IN.DAT")), "gbk"));
    FileOutputStream fos = new FileOutputStream("OUT.DAT");
    OutputStreamWriter osw = new OutputStreamWriter(fos, "gbk");
    PrintWriter out = new PrintWriter(osw);
    String str;
    while ((str = in.readLine()) != null) {
        str = str.trim();
        if (str.equals("")) {// 忽略空行
            continue;
        }
```

```
            String[] ary = str.split("\\W\\s*");// 拆分
            ary = f(ary);// 倒排
            str = toString(ary);// 将数组转为字符串
            out.println(str);// 写入OUT.DAT
        }
        in.close();
        out.flush();
        out.close();
    }

    /*
     * 单词倒排
     */
    public static String[] f(String[] ary) {
        int n = ary.length;
        String[] ss = new String[n];
        for (int i = 0; i < n; i++) {
            ss[i] = ary[n - 1 - i];
        }
        return ss;
    }

    /*
     * 将数组转为字符串
     */
    public static String toString(String[] ary) {
        StringBuilder sb = new StringBuilder();
        for (int i = 0; i < ary.length; i++) {
            sb.append(ary[i]);
        }
        return sb.toString();
    }

}
```

3.66.Class.forName(String className) 的功能？

参考答案：

如果没有加载给定字符串对应的类，要加载这个类，然后返回与带有给定字符串名的类或接口相关联的 Class 对象。

3.67.谈谈对反射技术的理解？

参考答案：

Java 中类的反射是一种自我管理机制，通过反射可实现：在运行时判断任意一个对象所属的类；在运行时构造任意一个类的对象；在运行时判断任意一个类所具有的成员变量和方法；在运行时调用任意一个对象的方法。

3.68.如何获得当前日期时间？

参考答案：

以下两种方式都可以获取到当前时间：

```
    Date date=new Date();
    Calendar c=Calendar.getInstance();
```

3.69.如何将日期时间转换成想要的格式？如：2008-04-01 13:39:24

参考答案：

```
public static String getFormatDate(Date date){
    DateFormat fmt=new SimpleDateFormat("yyyy-MM-dd HH:mm:ss");
    String str=fmt.format(date);
    return str;
}
```

3.70.编写程序，计算任意两个日期之间的天数？

参考答案：

```
public static double getDays(Calendar c1,Calendar c2){
    Date date1=c1.getTime();
    Date date2=c2.getTime();
    return getDays(date1,date2);
}
public static double getDays(Date date1,Date date2){
    long time1=date1.getTime();
    long time2=date2.getTime();
    double d=Math.abs((time1-time2)/1000d/60/60/24);
    return d;
}
public static double getDays(String str1,String str2) throws ParseException{
    DateFormat format=new SimpleDateFormat("yyyy-MM-dd");
    Date date1=format.parse(str1);
    Date date2=format.parse(str2);
    return getDays(date1,date2);
}
```

3.71.什么是标识接口(Mark Interface)，它与接口的区别在什么地方，并简单列举你知道的标识接口？

参考答案：

标识接口：标识接口是没有任何方法和属性的接口。标识接口不对实现它的类有任何语义上的要求，它仅仅表明实现它的类属于一个特定的类型。当一个类实现了一个标识接口之后就像是给自己打了个标签。

区别：标识接口没有任何方法和属性；而接口有方法或属性，或者即有方法又有属性。

举例： java.io.Serializable，java.rmi.Remote。

3.72.请使用 Java 语言定义一个具备栈功能的类，实现以下接口：

```
public interface StackInterface{
    void push(Object value);
    Object pop();
    boolean isEmpty();
}
```

参考答案：

```
public class Stack implements StackInterface {
 private int capacity = 100;
 private Object[] items;
 private int top = 0;
 // 不带参数构造器
 public Stack() {
     this(100);
 }
 // 带参数构造器，参数为堆栈大小
 public Stack(int cap) {
     this.capacity = cap;
     items = new Object[cap];
 }
 @Override
 public boolean isEmpty() {
     if(top==0){
         return true;
     }
     return false;
 }
 @Override
 public Object pop() {
     Object temp=items[top];
     items[top] = null;
     top--;
     return temp;
 }
 @Override
 public void push(Object value) {
     top++;
     items[top] = value;
 }
}
```

3.73.编写一个程序，用户输入任何一个字符串之后，反转输出其结果。其运行结果如下所示：

请输入一个字串：我在学习 Java 程序设计
您输入了：计设序程 avaJ 习学在我

参考答案：

```
import java.util.Scanner;

public class Q073 {
 public static void main(String[] args) {
     System.out.print("请输入一个字串：");
     Scanner in = new Scanner(System.in);
     String input = in.nextLine();
     System.out.print("您输入了：");
     System.out.println(reverseString(input));
 }

 public static String reverseString(String str) {
     char[] chars = str.toCharArray();
     StringBuilder sb = new StringBuilder();
     for (int i = chars.length - 1; i >= 0; i--) {
```

```
            sb.append(chars[i]);
        }
        return sb.toString();
    }
}
```

3.74.请用面向对象的思想设计一个计算器程序,请写出程序框架(主要使用的类和方法)?

参考答案:

Operation 类:

```
public class Operation {
 private double numberA = 0;
 private double numberB = 0;

 public double GetResult() {
     double result = 0;
     return result;
 }

 public double getNumberA() {
     return numberA;
 }

 public void setNumberA(double numberA) {
     this.numberA = numberA;
 }

 public double getNumberB() {
     return numberB;
 }

 public void setNumberB(double numberB) {
     this.numberB = numberB;
 }
}
```

OperationAdd 类:

```
public class OperationAdd extends Operation{
 public double GetResult() {
       double result = 0;
          result = getNumberA() + getNumberB();
          return result;
 }
}
```

OperationDiv 类:

```
public class OperationDiv extends Operation{
 public double GetResult() {
       double result = 0;
          result = getNumberA() / getNumberB();
          return result;
 }
}
```

OperationMul 类:

```
public class OperationMul extends Operation{
 public double GetResult() {
       double result = 0;
          result = getNumberA() * getNumberB();
          return result;
 }
}
```

OperationSub 类：

```
public class OperationSub extends Operation{
 public double GetResult() {
       double result = 0;
          result = getNumberA() - getNumberB();
          return result;
 }
}
```

OperationFactory 类：

```
public class OperationFactory {
 public static Operation createOpearate(char operate) {
     Operation oper = null;
     switch (operate) {
     case '+':
         oper = new OperationAdd();
         break;
     case '-':
         oper = new OperationSub();
         break;
     case '*':
         oper = new OperationMul();
         break;
     case '/':
         oper = new OperationDiv();
         break;
     default:
         break;
     }
     return oper;
 }

}
```

Calculator 类：

```
import java.util.Scanner;

public class Calculator {
 public static void main(String[] args) {
     Operation oper;
     oper = OperationFactory.createOpearate('+');
     System.out.println("请输入_numberA:");
     Scanner s = new Scanner(System.in);
     oper.setNumberA(s.nextDouble());
     System.out.println("请输入_numberB:");
     oper.setNumberB(s.nextDouble());
     double result = oper.GetResult();
     System.out.println("结果是：" + result);
 }
```

```
}
```

3.75.写一个函数，2 个参数，1 个字符串，1 个字节数，返回截取的字符串，要求字符串中的中文不能出现乱码：如（“我 ABC”，4）应该截为“我 AB”，输入（“我 ABC 汉 DEF”，6）应该输出为“我 ABC”而不是“我 ABC+汉的半个”。

参考答案：

```
public static String subString(String str, int subBytes) {
    int bytes = 0; // 用来存储字符串的总字节数
    for (int i = 0; i < str.length(); i++) {
        if (bytes == subBytes) {
            return str.substring(0, i);
        }
        char c = str.charAt(i);
        if (c < 256) {
            bytes += 1; // 英文字符的字节数看作 1
        } else {
            bytes += 2; // 中文字符的字节数看作 2
            if (bytes - subBytes == 1) {
                return str.substring(0, i);
            }
        }
    }
    return str;
}
```

3.76.日期和时间：

1) 如何取得年月日、小时分秒？
2) 如何取得从 1970 年到现在的毫秒数？
3) 如何取得某个日期的当月的最后一天？
4) 如何格式化日期？

参考答案：

1) 创建 java.util.Calendar 实例(Calendar.getInstance()),调用其 get()方法传入不同的参数即可获得参数所对应的值，如：calendar.get(Calendar.YEAR);//获得年

2) 以下方法均可获得该毫秒数:

```
Calendar.getInstance().getTimeInMillis();
System.currentTimeMillis();
```

3) 取得某个日期的当月的最后一天的示例代码如下:

```
Calendar time = Calendar.getInstance();
time.getActualMaximum(Calendar.DAY_OF_MONTH));
```

4) 利用 java.text.DateFormat 类中的 format()方法可将日期格式化。

3.77.写一个方法,输入一个文件名和一个字符串,统计这个字符串在这个文件中出现的次数？

参考答案：

```
public int countWords(String file, String find) throws Exception {
    int count = 0;
    Reader in = new FileReader(file);
    int c;
    while ((c = in.read()) != -1) {
        while (c == find.charAt(0)) {
            for (int i = 1; i < find.length(); i++) {
                c = in.read();
                if (c != find.charAt(i))
                    break;
                if (i == find.length() - 1)
                    count++;
            }
        }
    }
    return count;
}
```

3.78.写一个函数，要求输入一个字符串和一个字符的长度，对该字符串进行分隔？

参考答案：

```
public String[] split(String str, int chars) {
    int n = (str.length() + chars - 1) / chars;
    String ret[] = new String[n];
    for (int i = 0; i < n; i++) {
        if (i < n - 1) {
            ret[i] = str.substring(i * chars, (i + 1) * chars);
        } else {
            ret[i] = str.substring(i * chars);
        }
    }
    return ret;
}
```

3.79.Java 类实现序列化的方法？ 在 COLLECTION 框架中，实现比较要实现什么样的接口？

参考答案：

Java 类实现序例化的方式为使要序列化的类实现 java.io.Serializable 接口；Collection 框架中实现比较要实现 Comparable 接口或 Comparator 接口。

3.80.设计 4 个线程， 其中两个线程每次对 j 增加 1， 另两个线程对 j 每次减少 1；写出程序？

参考答案：

```
public class TestThread {
 private int j;
```

```
    private Object obj = new Object();
    public TestThread(int j) {
        this.j = j;
    }
    class Dec extends Thread {
        public void run() {
            synchronized (obj) {
                j--;
            }
        }
    }
    class Inc implements Runnable {
        public void run() {
            synchronized (obj) {
                j++;
            }
        }
    }
    public static void main(String[] args) {
        new TestThread(0).new Dec().start();
        new TestThread(0).new Dec().start();
        new Thread(new TestThread(0).new Inc()).start();
        new Thread(new TestThread(0).new Inc()).start();
    }
}
```

3.81.请写出线程的实现方式？

参考答案：

线程有两种实现方式，分别是继承于 Thread 类和实现 Runnalbe 接口。

方式一：继承于 Thread 类，代码如下所示：

```
class Person1 extends Thread{
  public void run() {//覆盖 Thread 的 run()方法，提供具体的过程
    System.out.println(this.getName()+","+this.getId());
    for(int i=0;i<100; i++){
      System.out.println("做弹弓子打你家玻璃!");
    }
    System.out.println("弹弓子 Over! ");
  }
}
```

方式二：创建一个类实现 Runnable 接口，重写 run 方法以实现了 Runnable 接口的方法，代码如下所示：

```
class Person1  implements  Runnable{
  public void run() {//实现 Runnable 的 run()方法，提供具体的过程
    System.out.println(this.getName()+","+this.getId());
    for(int i=0;i<100; i++){
      System.out.println("做弹弓子打你家玻璃!");
     }
    System.out.println("弹弓子 Over! ");
  }
}
```

上述类的对象可以作为创建 Thread 类对象的构造函数的参数。

3.82.字符串“sh24sdgf909sdf8”,不改变原字符串对象,写一个方法,返回“shsdgfsdf？

参考答案：

```
public String deleteDigital (String str){
    return str.replaceAll("\\d+","");
}
```

3.83.“driver|name|id|”写一个函数，分别获取 driver,name,id 并且输出？

参考答案：

```
public void split(String str) {
    String[] strs = str.split("\\|");
    for (String s : strs) {
        System.out.println(s);
    }
}
```

3.84.剔除下面字符串中的“,”，例如： 字符串“A,B,C,D,E,F”,剔除的返回结果为“ABCDEF”？

参考答案：

```
public class Q084 {
 public static String delete(String str) {
    String result = "";
    for (int i = 0; i < str.length(); i++) {
        String single = String.valueOf(str.charAt(i));
        result += single.replaceAll(",", "");
    }
    return result;
 }
 public static void main(String[] args) {
    String s = "A,B,C,D,E,F";
    String str=delete(s);
    System.out.println(str);
 }
}
```

3.85.编写一个命令行程序，提示输入 2 个字符串，找出 2 串字符串中是否存在相同的长度为 3 的子字符串。比如字符串 “abc12uwks”和字符串“q72bnabc00”,则把"abc"找出？

参考答案：

```
public class Q085 {
 public static void main(String[] args) {
    Scanner s = new Scanner(System.in);
    System.out.println("请输入第一个字符串：");
    String first = s.next();
    System.out.println("请输入第二个字符串");
    String second = s.next();
    for (int i = 0; i < first.length(); i++) {
        String sub1 = "";
```

```
            if (i + 3 < first.length()) {
                sub1 = first.substring(i, i + 3);
            }
            String sub2="";
            for (int j = 0; j < second.length(); j++) {
                if (j+ 3 < second.length()) {
                    sub2 = second.substring(j, j + 3);
                }
                if(sub1.equals(sub2)){
                    System.out.println(sub1);
                    break;
                }
            }
        }
    }
}
```

3.86.现有一个数组 int data_arr[]={12,31,56,23,27,1,43,65,4,99} ,已完成如下代码,请在注释处增加一段代码,实现这样的功能:将 data_arr 数组的内容先写入一个名为”temp.dat”文件中,再重新将该文件的内容读出,能逆序将文件的内容输出至控制台(形如 99,4,65,43,1,27,23,56,31,12)？

```
package a;
import java.io.BufferedInputStream;
import java.io.BufferedOutputStream;
import java.io.BufferedReader;
import java.io.BufferedWriter;
import java.io.DataInputStream;
import java.io.DataOutputStream;
import java.io.FileInputStream;
import java.io.FileOutputStream;
import java.io.IOException;
import java.io.InputStreamReader;
import java.io.OutputStreamWriter;
public class arrayTest {
 public static void main(String[] args) throws IOException{
     try {

         int data arr[]={12,31,56,23,27,1,43,65,4,99};
         //定义输出流

         //定义 BufferedWriter,如果是中文数组,可以加上字符编码

         //定义输入流

         //定义 BufferedReader,如果是中文数组,可以加上字符编码

         //写入操作


         //读取操作


         System.out.println("逆序结果\n"+
                  outContent.substring(0,outContent.length()-1)); //截取
         bw.close();
         br.close();
     } catch (Exception e) {
         e.printStackTrace();
```

```
        }
    }
}
```

参考答案：

```
public class ArrayTest {
    public static void main(String[] args) throws IOException {
        try {

            int data_arr[] = { 12, 31, 56, 23, 27, 1, 43, 65, 4, 99 };
            // 定义输出流
            DataOutputStream douts = new DataOutputStream(
                    new BufferedOutputStream(
                            new FileOutputStream("D:/temp.dat")));
            // 定义 BufferedWriter,如果是中文数组,可以加上字符编码
            BufferedWriter bw = new BufferedWriter(
                    new OutputStreamWriter(douts));
            // 定义输入流
            DataInputStream dinputs = new DataInputStream(
                    new                                    BufferedInputStream(new
FileInputStream("D:/temp.dat")));
            // 定义 BufferedReader,如果是中文数组,可以加上字符编码
            BufferedReader br = new BufferedReader(new InputStreamReader(
                    dinputs));
            // 写入操作
            String inContent = "";
            for (int i = 0; i < data_arr.length; i++) {
                inContent += data_arr[i] + "\n";
            }
            bw.write(inContent);
            bw.flush();
            // 读取操作
            String outContent = "";
            char tag = 0;
            while (tag != (char) -1) {
                outContent = tag + br.readLine() + "," + outContent.trim();
                tag = (char) br.read();
            }
            System.out.println("逆序结果\n"
                    + outContent.substring(0, outContent.length() - 1)); // 截
取
            bw.close();
            br.close();
        } catch (Exception e) {
            e.printStackTrace();
        }
    }
}
```

3.87.现有一个程序需要记录运行中的关键信息，要求在内存中保留最近的 n 条信息供查询。保存的信息不需要做持久化处理。要求提供一个独立的日志类，负责对日志的写入和读取。请列出该类所应具有的方法并写出大致的实现代码？

注：持久化指将数据存入数据库或存为文件。读取日志时只要求能一次性读出现有的所有记录。

参考答案：

```
import java.util.ArrayList;
import java.util.List;

public class LogOpr {
private List<String> list=new ArrayList<String>();
public boolean writerLog(String str){
     return list.add(str);
   }
   public List<String> readLog(){
     return list;
    }
}
```

3.88.数学中，一个复数包容着一个实部（ Real ）和一个虚部（ Imaginary ）。请设计一个 Complex 类，它的实例代表一个复数，并且用户可以这样使用它：

```
 // 创建一个实部为 3 ，虚部为 4 的复数
Complex obj = new Complex(3,4);
//Complex 对象具备按照数学中复数的习惯表示形式输出的能力。
System.out.println(obj);  // 输出： 3+4i
```

参考答案：

```
public class Complex {
 private int real;
 private int imaginary;
 public Complex(){
     this.real=0;
     this.imaginary=0;
 }
 public Complex(int real,int imaginary){
     this.real=real;
     this.imaginary=imaginary;
 }
 public String toString(){
     return ""+real+"+"+imaginary+"i";
 }
 public static void main(String[] args) {
      // 创建一个实部为 3 ，虚部为 4 的复数
     Complex obj = new Complex(3,4);
     //Complex 对象具备按照数学中复数的习惯表示形式输出的能力。
     System.out.println(obj);  // 输出： 3+4i

 }
}
```

3.89.我们可以调用 `Integer.parseInt()` 方法将一个字串转换为 `int` 类型，但当要转换的字符串不是一个有效的数字时（比如 “a123” ），此方法会抛出一个 `NumberFormatException`？

要求编写一个程序，当程序运行时让用户从键盘上输入一个字串，代表考试成绩，然后调用 Integer.parseInt() 方法将其转换为 int 类型，并给出是否通过的提示，控制台交互

的过程如下所示：

```
请输入您的考试成绩： 76
恭喜您通过了考试！
```

由于无法控制用户的输入，因此可能出现以下两种出错情况：

1) 用户输入了一个无法转换为 int 类型的字符串；

2) 用户输入的数字不在 [0,100] 区间内。

请设计一个自定义的异常类 InvalidScoreException ，当出现上述异常情况时抛出此异常对象。要求程序运行时，对用户的错误输入能给以明确的提示。

参考答案：

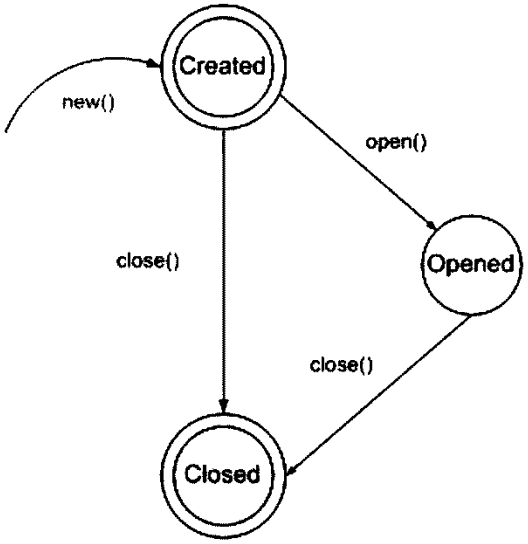
首先，自定义异常处理类 InvalidScoreException，代码如下所示：

```
public class InvalidScoreException extends Exception {
 public InvalidScoreException() {
     super();
 }

 public InvalidScoreException(String msg) {
     super(msg);
 }
}
```

然后，当用户输入了一个无法转换为 int 类型的字符串或用户输入的数字不在 [0,100] 区间内时抛出自定义异常 InvalidScoreException，代码如下所示：

```
public class Q021 {
 public void readScore() throws InvalidScoreException {
     System.out.println("请输入成绩：");
     Scanner s = new Scanner(System.in);
     try {
         int score = Integer.parseInt(s.next());
         if (score < 0 || score > 100) {
             throw new InvalidScoreException("分数要在 1 到 100 之间");
         }
     } catch (NumberFormatException e) {
         throw new InvalidScoreException("分数要求是数字");
     }

 }
}
```

3.90.设计一个类，实现如下图所示的状态机？

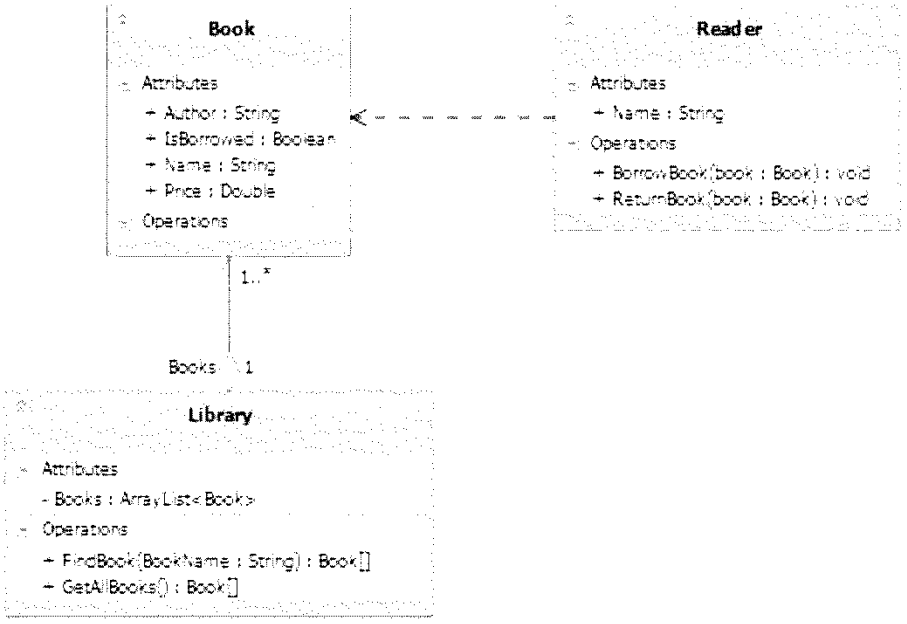


提示： 上图说明，当使用 new 关键字创建此类对象时，对象居于 Created 状态（这是初始状态），这时，调用对象的 open() 方法，对象转换到 Opened 状态，再调用对象的 close() 方法，对象转入 “ Closed ” 状态，这是终止状态。

参考答案：

```
public class Q091 {
 private String status;

 public Q022() {
     this.status = "Created";
 }

 public void open() {
     this.status = "Opened";
 }

 public void close() {
     this.status = "Closed";
 }

 public String getStatus() {
     return status;
 }

 public void setStatus(String status) {
     this.status = status;
 }

}
```

3.91.一个图书管理系统的面向对象设计方案如下图 所示：



Book 代表书，有 “Name （书名）” 、 “Author （作者名）” 、 “Price （单价）” 和 “IsBorrowed （是否被借 出）” 四个属性。

类 Library 代表图书馆，其内部字段 books 用于保存图书馆中所有的书。它的 FindBook() 方法依据书名查找同名的书（可能有多本）。另一个 GetAllBooks() 方法获取馆藏所有书的详细信息。

类 Reader 代表读者，Name 字段代表其姓名，读者可以 “ReturnBook （还书）” 和 “BorrowBook （借书）” 。

请编程完成以下工作：

1． 用 Java 编程实现上述 3 个类。

2．在 main() 方法内书写以下测试代码：

1）创建一个 Library 类的实例 myLittleLibrary ，其中预存有以下 3 本书：

Java 程序设计，张三著， 45 元

Java 核心技术，李四著， 50 元

Java 程序设计，王五著， 38 元

2）显示图书馆中所有图书的信息，输出样例如下：

Java 程序设计，张三著， 45 元，可借

Java 核心技术，李四著， 50 元，可借

Java 程序设计，王五著， 38 元，未还

3）(创建一个 Reader 类的实例 oneBeautifulGirl ，她先在 myLittleLibrary 中查找《 Java 程序设计》

4）oneBeautifulGirl 借了张三著的那一本书。现在显示图书馆中所有图书的信息 .

5）oneBeautifulGirl 把书还了，再次显示图书馆中图书信息。

注意： 在满足题目要求实现功能的前提下，你可以依据自己的考虑修改系统设计方案

（比如给某个类添加或修改类的方法，甚至是添加新的类）。

参考答案：

Book 类代码如下：

```
public class Book {
 private String name;
 private String author;
 private double price;
 private boolean isBorrowed;

 public String getName() {
     return name;
 }

 public void setName(String name) {
     this.name = name;
 }

 public String getAuthor() {
     return author;
 }

 public void setAuthor(String author) {
     this.author = author;
 }

 public double getPrice() {
     return price;
 }

 public void setPrice(double price) {
     this.price = price;
 }

 public boolean isBorrowed() {
     return isBorrowed;
 }

 public void setBorrowed(boolean isBorrowed) {
     this.isBorrowed = isBorrowed;
 }
    public String toString(){
     return name+","+author+","+price+"元,"+(isBorrowed?"未还":"可借");
    }
}
```

Library类代码如下：

```
import java.util.ArrayList;
import java.util.List;

public class Library {
 private static Library lib = null;
 private List<Book> books = new ArrayList<Book>();

 private Library() {}
 public void addBooks(Book book){
     books.add(book);
 }
```

```
    public static Library getLibrary() {
        if (lib == null) {
            lib = new Library();
        }
        return lib;
    }

    public List<Book> findBook(String name) {
        List<Book> findList = new ArrayList<Book>();
        for (Book book : books) {
            if (name != null && name.equals(book.getName())) {
                findList.add(book);
            }
        }
        return findList;
    }

    public List<Book> getAllBooks() {
        return books;
    }
}
```

Reader 类代码如下：

```
import java.util.List;

public class Reader {
    private String name;

    public String getName() {
        return name;
    }

    public void setName(String name) {
        this.name = name;
    }

    public void returnBook(String name, String author) {
        // 设置书为归还状态
        List<Book> books = Library.getLibrary().getAllBooks();
        for (Book book : books) {
            if (name.equals(book.getName()) && author.equals(book.getAuthor()))
{
                book.setBorrowed(false);
                break;
            }
        }
    }

    public void borrowBook(String name, String author) {
        // 设置书为出借状态
        List<Book> books = Library.getLibrary().getAllBooks();
        for (Book book : books) {
            if (name.equals(book.getName()) && author.equals(book.getAuthor()))
{
                book.setBorrowed(true);
                break;
            }
        }
    }
}
```

Test 类代码如下：

```
import java.util.List;

public class Test {
 public static void main(String[] args) {
     Library myLittleLibrary = Library.getLibrary();

     Book b1 = new Book();
     b1.setName("Java 程序设计");
     b1.setAuthor("张三著");
     b1.setPrice(45);
     b1.setBorrowed(false);

     Book b2 = new Book();
     b2.setName("Java 核心技术");
     b2.setAuthor("李四著");
     b2.setPrice(50);
     b2.setBorrowed(false);

     Book b3 = new Book();
     b3.setName("Java 程序设计");
     b3.setAuthor("张五著");
     b3.setPrice(38);
     b3.setBorrowed(false);
     // 初始化图书馆
     myLittleLibrary.addBooks(b1);
     myLittleLibrary.addBooks(b2);
     myLittleLibrary.addBooks(b3);
     // 显示全部图书
     List<Book> books = myLittleLibrary.getAllBooks();
     for (Book book : books) {
         System.out.println(book);
     }
     // 出借
     Reader oneBeautifulGirl = new Reader();
     oneBeautifulGirl.borrowBook("Java 程序设计", "张三著");
     for (Book book : books) {
         System.out.println(book);
     }
     // 归还
     oneBeautifulGirl.returnBook("Java 程序设计", "张三著");
     for (Book book : books) {
         System.out.println(book);
     }
 }
}
```

3.92.读取 txt 文件内容

文件内容大致如下：

00001　张三　　计算机系　　男　...

00002　李四　　外语系　　女　...

读取文件后对内容进行整合，按院系分类输出。格式为：

计算机系

00001　张三　　男　...

外语系

00002 李四 女 …

参考答案：

Student 类代码如下:

```
public class Student {
 private String no;
 private String name;
 private String major;
 private String gender;
 private String other;

 public String getNo() {
     return no;
 }
 public void setNo(String no) {
     this.no = no;
 }
 public String getName() {
     return name;
 }
 public void setName(String name) {
     this.name = name;
 }
 public String getMajor() {
     return major;
 }
 public void setMajor(String major) {
     this.major = major;
 }
 public String getGender() {
     return gender;
 }
 public void setGender(String gender) {
     this.gender = gender;
 }
 public String getOther() {
     return other;
 }
 public void setOther(String other) {
     this.other = other;
 }
}
```

StudentInfos 类代码如下：

```
import java.io.BufferedReader;
import java.io.FileInputStream;
import java.io.IOException;
import java.io.InputStreamReader;
import java.util.ArrayList;
import java.util.Arrays;
import java.util.HashMap;
import java.util.List;
import java.util.Map;
import java.util.Set;

public class StudentInfos {
 private              Map<String,List<Student>>              students=new
HashMap<String,List<Student>>();
 public void readStudents(String fileName) throws IOException{
     BufferedReader   br=new   BufferedReader(new   InputStreamReader(new
FileInputStream(fileName),"UTF-8"));
```

```
        String line="";
        while((line=br.readLine())!=null){
            String[] infos=line.split("\\s+");
            System.out.println(Arrays.toString(infos));
            String no=infos[0];
            String name=infos[1];
            String major=infos[2];
            String gender=infos[3];
            String other=infos[4];

            Student stu=new Student();
            stu.setNo(no);
            stu.setName(name);
            stu.setMajor(major);
            stu.setGender(gender);
            stu.setOther(other);
            List<Student> list=null;
            if(students.containsKey(major)){
                list=students.get(major);
                list.add(stu);
            }else{
                list=new ArrayList<Student>();
                list.add(stu);
                students.put(major, list);
            }
        }
        br.close();
    }
    public void showStudent(){
        Set<String> keys=students.keySet();
        for(String key:keys){
            System.out.println(key);
            List<Student> list=students.get(key);
            for(Student stu:list){
                System.out.println(stu.getNo()+"            "+stu.getName()+"
"+stu.getGender()+" "+stu.getOther());
            }
        }

    }
    }
```

Test 类代码如下：

```
    import java.io.IOException;

    public class Test {
    public static void main(String[] args) {
        StudentInfos infos=new StudentInfos();
        try {
            infos.readStudents("tmp.txt");
            infos.showStudent();
        } catch (IOException e) {
            e.printStackTrace();
        }
    }
    }
```

3.93.采用 Java 多线程技术（wait 和 notify），设计实现一个符合生产者和消费者问题的程序。对一个对象（枪膛）进行操作，其最大容量是 20 颗子弹。生产者线程是一个压入线程，它不断向枪膛中压入子弹；消费者线程是一个射出线程，它不断从枪膛中射出子弹？

参考答案：

StackInterface 接口代码如下：

```
public interface StackInterface
{
public void push(int n);//压入子弹
public int[] pop();//射出子弹
}
```

PushThread 类代码如下：

```
//生产者线程
public class PushThread implements Runnable
{
  private StackInterface s;
  public PushThread(StackInterface s)
  {
    this.s = s;
  }
  public void run()
  {
    int i = 0;

    while(true)
    {
      java.util.Random r = new java.util.Random();
      i = r.nextInt(10);
      s.push(i);
      try {
        Thread.sleep(100);
      }
      catch(InterruptedException e){}
    }
  }
}
```

PopThread 类代码如下：

```
//消费者线程
public class PopThread implements Runnable
{
  private StackInterface s;
  public PopThread(StackInterface s)
  {
    this.s = s;
  }
  public void run()
  {
    while(true)
    {
     System.out.println("->"+ s.pop()[0] + "<-");
```

```
        try {
                Thread.sleep(100);
           }
           catch(InterruptedException e){}
       }
     }
}
```

SafeStack 代码如下：

```
//实现生产和消费的过程
public class SafeStack implements StackInterface {
 private int top = 0;

 private int[] values = new int[20];//表示枪膛对象

 private boolean dataAvailable = false;

 public void push(int n) {
     synchronized (this) {
         while (dataAvailable) // 1
         {
             try {
                 wait();
             } catch (InterruptedException e) {
                 // 忽略 //2
             }
         }
         values[top] = n;
         System.out.println("压入数字" + n + "步骤 1 完成");
         top++;
         dataAvailable = true;
         notifyAll();
         System.out.println("压入数字完成");
     }
 }

 public int[] pop() {
     synchronized (this) {
         while (!dataAvailable) // 3
         {
             try {
                 wait();
             } catch (InterruptedException e) {
                 // 忽略 //4
             }
         }
         System.out.print("弹出");
         top--;
         int[] test = { values[top], top };
         dataAvailable = false;
         // 唤醒正在等待压入数据的线程
         notifyAll();
         return test;
     }
 }
}
```

TestSafeStack 类代码如下：

```
//测试
public class TestSafeStack {
 public static void main(String[] args) {
     SafeStack s = new SafeStack();
     PushThread r1 = new PushThread(s);
     PopThread r2 = new PopThread(s);
     PopThread r3 = new PopThread(s);
     Thread t1 = new Thread(r1);
     Thread t2 = new Thread(r2);
     Thread t3 = new Thread(r3);
     t1.start();
     t2.start();
     t3.start();
 }
}
```

3.94.写一个线程，每隔 10 秒钟标准输出到屏幕上一个 “hello world"。打印 10 次以后退出？

参考答案：

```
public class TimerThread extends Thread{
 public void run(){
     for(int i=0;i<10;i++){
         System.out.println("hello world");
         try {
             sleep(10000);
         } catch (InterruptedException e) {
             e.printStackTrace();
         }
     }
 }
    public static void main(String[] args) {
     TimerThread tt=new TimerThread();
     tt.start();
 }
}
```

3.95.下列两个方法有什么区别？

```
 public synchronized void method1(){}
 public  void method2(){
     synchronized  (obj){}
 }
```

参考答案：

```
public synchronized void method1(){}
```

上述代码锁定的对象是调用这个同步方法的那个对象。

```
public  void method2(){
        synchronized  (obj){}
 }
```

上述代码锁定的对象是 obj 对象。

3.96.如何格式化日期？

参考答案：

使用 SimpleDateFormat 类，可以实现日期的格式化。以下代码中的 nowStr 变量就是格式化后的日期形式，年-月-日 时：分：秒。

```
import java.text.SimpleDateFormat;
import java.util.Date;
public class Q029 {
 public static void main(String[] args) {
     Date now=new Date();
     SimpleDateFormat sdf=new SimpleDateFormat("yyyy-MM-dd hh:mm:ss");
     String nowStr=sdf.format(now);
 }
}
```

3.97.如何取 1970 到现在的毫秒数？

参考答案：

```
import java.util.Date;
public class Q030 {
 public static void main(String[] args) {
     Date dat=new Date();
     long now=dat.getTime();
 }
}
```

3.98.给 SomeInputStream 类的 skip 函数添加 JAVA 注释，需要添加函数本身功能（流指针向后偏移指定长度），作者（答题者姓名），异常，参数，函数返回，函数定义最早出现的版本（x.x）？

```
public abstract class SomeInputStream extends java.io.InputStream{

    @Override
    public long skip(long n) throws java.io.IOException {
            return super.skip(n);
    }
}
```

参考答案：

```
public abstract class SomeInputStream extends java.io.InputStream {
   /**
    * 流指针向后偏移指定长度
    *
    * @author jessica
    * @param  n
    * 指定长度
    * @return 偏移后的位置
    * @throws IOException
    * @version 1.0
    */
@Override
```

```
    public long skip(long n) throws java.io.IOException {
        return super.skip(n);
    }
}
```

3.99.有数组[5,0,-5,2,-4,5,10,3,-5,2,-4,3,4,9,1] ,请写代码输出每个数的频率数(正负数算一个数)，如下面结果：

5 出现 4 次
0 出现 1 次

参考答案：

```
import java.util.HashMap;
import java.util.Map;
import java.util.Set;

public class Q033 {
    public static void main(String[] args) {
        int[] arr={5,0,-5,2,-4,5,10,3,-5,2,-4,3,4,9,1};
        Map<Integer,Integer> map=new HashMap<Integer,Integer>();
        for(Integer i:arr){
            Integer key=Math.abs(i);
            if(map.containsKey(key)){
             map.put(key, map.get(key)+1);
            }else{
             map.put(key, 1);
            }
        }
        Set<Integer> keys=map.keySet();
        for(Integer key:keys){
            System.out.println(key+"出现"+map.get(key)+"次");
        }
    }
}
```

3.100.请写一个 Test 类包含 divide 方法，该方法实现两个整数相除精确返回四舍五入到百分位的数值？

参考答案：

```
public static double divide(double first, double second) {
    return Math.round(first*100 / second) / 100.0;
}
```

3.101.将某网页评论提交给后台的字符串 str 中，如果包含有“\”，”<”，”>”，”=”四种符号的，过滤掉以后防止系统被执行恶意代码。写出你的 java 代码？

```
String str =“<<楼主说的非常对！\\和谐社会，科学上网>=。“
```

参考答案：

```
public class Q038 {
```

```
    // 过滤特殊字符
    public static String StringFilter(String str) throws PatternSyntaxException
{
        String regEx = "[=\\\\<>]";
        Pattern p = Pattern.compile(regEx);
        Matcher m = p.matcher(str);
        return m.replaceAll("").trim();
    }
    public static void main(String[] args) {
        String str ="<<楼主说的非常对！\\和谐社会，科学上网>=。";
        System.out.println(str);
        System.out.println(StringFilter(str));
    }
    }
```

3.102.现在给你一个新一代身份证号，默认为 18 位号码；如果该号码是 15 位的话，请在第 6 位后面加上“19”，并在末尾加上“X”；如果是 18 位，则不变？

参考答案：

```
import java.util.Scanner;

public class Q039 {
 public static String change18(String cardId){
     StringBuilder sb=new StringBuilder(cardId);
     return sb.insert(6, "19").append("X").toString();
 }
 public static void main(String[] args) {
     System.out.println("请输入省份证号：");
     Scanner sc=new Scanner(System.in);
     String idCard=sc.next();
     if(idCard.length()==15){
         idCard=change18(idCard);
     }
     System.out.println(idCard);
 }
 }
```

3.103.编写一个方法：

给你一个字符串数组 String s[]={“A”,”B”,”C”,”D”,”E”,”F”}，请随机生成一个数，比如：生成的数为 2，则输出数组中随机两个元素的组合 B、E 或者 A、C 等；生成数为 3，就为三个随即元素组合？

参考答案：

```
public class Q104 {
    public List<String> group(int count,String[] str) {
    List<String> list = new ArrayList<String>();
    for (int i = 0; i < str.length; i++) {
        String result = "";
        result = str[i];
        for (int j = i + 1; j < str.length; j++) {
            for (int h = 0; h < count-1; h++) {
                if ((j + h) < str.length) {
                    result += str[j + h];
                }
            }
            if (result.length() == count) {
```

```
                    list.add(result);
                }
                result = str[i];
            }
        }
        return list;
    }
    public static void main(String[] args) {
        String[] s = { "A", "B", "C", "D", "E", "F" };
        List<String> list = new Q104().group(3,s);
        System.out.println(list);
    }
}
```

3.104.请编写一个对于 JAVA 例外处理的完整的程序？

参考答案：

本案例以自定义异常为例说明 Java 异常处理的完整过程。

首先新建自定义异常类 NotNumberException，该异常可以在某个字符串不为数字组成的情况下抛出，代码如下：

```
public class NotNumberException extends Exception {
    public NotNumberException() {
        super();
    }
    public NotNumberException(String message, Throwable cause) {
        super(message, cause);
    }

    public NotNumberException(String message) {
        super(message);
    }

    public NotNumberException(Throwable cause) {
        super(cause);
    }
}
```

然后，新建类 ExceptionDemo，在类中添加 parseInt 方法，该方法用于将字符串转换为整数类型，当该字符串不是由数字组成时，使用 throw 抛出上面自定义的异常 NotNumberException，并且在方法声明处，使用 throws 声明抛出该异常，然后在 main 方法中调用 parseInt 方法时，使用 try-catch 处理异常。代码如下：

```
public class ExceptionDemo {
    public static void main(String[] args) {
        // 如果调用了有异常抛出的方法，就必须处理异常。
        try {
            int i = parseInt("32145");
            System.out.println(i);
        } catch (NotNumberException e) {// 异常的处理
            System.out.println(e);
        }
    }

    /** 将字符串转换为整数值 */
    public static int parseInt(String str) throws NotNumberException {// 声明异
```

常的抛出

```
        int sum = 0;
        for (int i = 0; i < str.length(); i++) {
            char c = str.charAt(i);
            if (c < '0' || c > '9')
                throw new NotNumberException("不是数字呀！");// 异常抛出点
            sum = sum * 10 + (c - '0');
        }
        return sum;
    }
}
```

3.105.编写自定义异常，处理输入字符串 abc 抛出异常，其他不抛出？

参考答案：

定义异常 MyException，当输入字符串为 abc 时，抛出该异常。代码如下：

```
public class MyException extends Exception {
    private String errorMsg;

    public MyException(String errorMsg) {
        this.errorMsg = errorMsg;
    }

    public String getErrorMsg() {
        return errorMsg;
    }

    public void setErrorMsg(String errorMsg) {
        this.errorMsg = errorMsg;
    }

    @Override
    public String toString() {
        return errorMsg;
    }
}
```

定义 TestException 类，在 main 方法中接收控制台输入，当输入字符串为 abc 时，抛出异常 MyException。代码如下：

```
import java.util.Scanner;
public class TestException {
    public static void main(String[] args) throws Exception {
        Scanner scan = new Scanner(System.in);
        String str = scan.nextLine();
        if (str.equals("abc")){
            throw new MyException("输入的字符串不正确！");
        }
    }
}
```

第二部分

1. 数据库

1.1.什么是关系型数据库管理系统？

参考答案：

由关系、数据 、数据之间的约束三者所组成的数据模型则称为 RDBMS，即关系型数据库管理系统。

1.2.简述几种主流的数据库及其厂商？

参考答案：

主流的数据库及其厂商如下：

1）Oracle 数据库是著名的 Oracle(甲骨文)公司的数据库产品，Oracle 数据库是世界上第一个商品化的关系型数据库管理系统；Oracle 数据库采用标准 SQL（结构化查询语言），支持多种数据类型，提供面向对象的数据支持，具有第四代语言开发工具，支持 UNIX、WINDOWS、OS/2 等多种平台；Oracle 公司的产品丰富，包括 Oracle 服务器、 Oracle 开发工具和 Oracle 应用软件，其中最著名的就是 Oracle 数据库。

2）DB2 是 IBM 公司的关系型数据库管理系统。DB2 有很多不同的版本， 可以运行在从掌上产品到大型机不同的终端机器上；DB2 Universal Database Personal Edition 和 DB2 Universal Database Workgroup Edition 分别是单用户和多用户系统， 可以运行在 OS/2 和 Windows 上；DB2 是 Oracle 的主要竞争对手。

3）Sybase 是美国 Sybase 公司的关系型数据库系统。Sybase 是较早采用 C/S 技术的数据库厂商，典型的 UNIX 或 Windows NT 平台上客户机/服务器环境下的大型数据库系统；Sybase 通常与 Sybase SQL Anywhere 用于客户机/服务器环境，前者作为服务器数据库，后者为客户机数据库，采用该公司研制的 PowerBuilder 为开发工具，在国内大中型系统中具有广泛的应用；Sybase 公司 2010 年被 SAP 收购。

4）Microsoft SQL Server 是微软的产品，运行在 Windows NT 服务器上。Microsoft SQL Server 的最初版本适用于中小企业，但是应用范围不断扩展，已经触及到大型、跨国企业的数据库管理。

5）MySQL 是开放源码的小型关系型数据库管理系统。广泛应用在中小型网站中，成本低、规模较 Oracle 和 DB2 小；2008 年 1 月 16 日，Sun 收购 MySQL。2009 年 4 月 20 日，SUN 被 Oracle 公司收购 ，所以 MySQL 现在属于 Oracle 公司。

1.3.简述结构化查询语言的分类？

参考答案：

结构化查询语言（SQL）可分为:

1．数据定义语言（DDL）: Data Definition Language

2．操纵语言（DML）: Data Manipulation Language

3．事务控制语言（TCL）: Transaction Control Language

4．数据查询语言（DQL）: Data Query Language

5．数据控制语言（DCL）: Data Control Language

1.4.简述 date 和 timestamp 的区别？

参考答案：

date 的最小单位是秒，timestamp 包含小数位的秒。如果需要秒以下的单位，需要用 timestamp。

1.5.简述索引的原理及创建索引的意义？

参考答案：

索引是对表的一列或多列进行排序的结构。因为绝大多数的搜索方法在搜索排序结构时效率都会大大提高，所以如果表中某一列经常被作为关键字搜索，则建议对此列创建索引。

索引提供指针以指向存储在表中指定列的数据值，根据指定的排序次序排列这些指针。数据库使用索引的方式与使用书的目录很相似：通过搜索索引找到特定的值，然后跟随指针到达包含该值的行。

1.6.简述视图的意义？

参考答案：

视图的意义在于两个方面：

1） 简化复杂查询。如果需要经常执行某项复杂查询，可以基于这个复杂查询建立视图，此后查询此视图即可。

2）限制数据访问。视图本质上就是一条 SELECT 语句，所以当访问视图时，只能访问到所对应的 SELECT 语句中涉及到的列，对基表中的其它列起到安全和保密的作用。

1.7.触发器分为事前触发和事后触发，这两种触发有何区别？语句级触发和行级触发有何区别？

参考答案：

事前触发器运行于触发事件发生之前，而事后触发器运行于触发事件发生之后。通常事前触发器可以获取事件之前和新的字段值。语句级触发器可以在语句执行前或后执行，而行

级触发器所影响的每一行时都会触发一次。

简单而言，事前触发主要是验证一些条件或进行一些准备工作，在数据保存之前触发，而事后触发则是进行收尾工作，保证事务的完整性，在表经过修改以后才触发。行级触发器是对 DML 语句影响的每个行执行一次，如 UPDATE 语句影响多行，就会对每行都激活一次触发器。而语句级触发器是对每个 DML 语句执行一次，如 INSERT 语句在表中即使插入了 100 多行，表上的 INSERT 语句级触发器也只会执行一次。

1.8.sql 语句中 exists 和 in 有何区别？SQL 语句优化有哪些方法？

参考答案：

sql 语句中 exists 和 in 有何区别如下：

exists 是用循环（loop）的方式，由 outer 表的记录数决定循环的次数，对于 exists 影响最大，所以，外表的记录数少，适合用 exists；in 先执行子查询，子查询的返回结果去重之后，再执行主查询，所以，子查询的返回结果越少，越适合用该方式。

SQL 语句的优化方式如下：

1. 尽量避免非操作符的使用。在索引列上使用 NOT 、<> 等非操作符，数据库管理系统是不会使用索引的，可以将查询语句转换为可以使用索引的查询。

2. 避免对查询的列的操作。任何对列的操作都可能导致全表扫描，这里所谓的操作包括数据库函数、计算表达式等，查询时要尽可能将操作移至等式的右边，甚至去掉函数。

3. 避免不必要的类型转换 。需要注意的是，尽量避免潜在的数据类型转换。如将字符型数据与数值型数据比较，会自动将字符进行转换，从而导致全表扫描。

4. 增加查询的范围限制。增加查询的范围限制，避免全范围的搜索。

5. 合理使用 IN 与 EXISTS。 例如：有 A，B 两个表，它们分别使用如如下情况：

1） 当只显示一个表的数据如 A，关系条件只一个如 ID 时，使用 IN 更适合，SQL 语句如下：

```
SELECT * FROM A WHERE id IN (SELECT id FROM B);
```

2) 当只显示一个表的数据如 A ,关系条件不只一个列 ,例如关系条件涉及到的列为 ID，col1 时，使 用 IN 就不方便了，可以使用 EXISTS，SQL 语句如下：

```
SELECT * FROM A WHERE EXISTS (SELECT 1 FROM B WHERE id = A.id and col1 = A.col1);
```

in 与 exists、 not in 与 not exists 的区别如下：

1） in 与 exists 的区别：in 是把外表和内表作 hash 连接，而 exists 是对外表作 loop 循环，每次 loop 循环再对内表进行查询。一直以来认为 exists 比 in 效率高的说法是不准确的。 如果查询的两个表大小相当，那么用 in 和 exists 差别不大。 如果两个表中一个较小，一个是大表，则子查询表大的用 exists，子查询表小的用 in。

2） not in 与 not exists 的区别：如果查询语句使用了 not in 那么内外表都进行全表扫描，没有用到索引；而 not extsts 的子查询依然能用到表上的索引。所以无论哪个表

大，用 not exists 都比 not in 要快。

6. 尽量去掉<> 。尽景去掉<>，避免全表扫描，如果数据是枚举值，且取值范围同定，则修改为 OR 或者 IN a<>0 改为 a>0 or a<0 a<>'' 改为 a>''。

7. 去掉 WHERE 子句中的 IS NULL 和 IS NOT NULL。WHERE 字句中的 IS NULL 和 IS NOT NULL 将不会使用索引而是进行全表搜索，因此需要通过改变查询方式，分情况讨论等办法 ,去掉 WHERE 子句中的 IS NULL 和 IS NOT NULL ,IS NOT NULL 改为 A>0 或者 A>''。

8. 尽量不要使用前导模糊查询。由于前导模糊查询(前面有%的 like 查询)不能利用索引,所以速度会比较慢。

9. SELECT 子句中避免使用'*' 。当你想在 SELECT 子句中列出所有的 COLUMN 时，使用动态 SQL 列引用 '*' 是一个方便的方法，不幸的是，这是一个非常低效的方法。实际上，数据库在解析的过程中， 会将'*' 依次转换成所有的列名， 这个工作是通过查询数据字典完成的，这意味着将耗费更多的时间。

10. 规范所有的 SQL 关键字的书写 比如 ,SELECT、UPDATE 、DELETE 、FROM 等，要么全部大写，要么全部小写，不要大小写混用。

1.9.说说您对数据库事务隔离级别的理解？

参考答案：

事务隔离级别：一个事务对数据库的修改与并行的另一个事务的隔离程度。两个并发事务同时访问数据库表相同的行时，可能存在以下三个问题：

1. 幻读：事务 T1 读取一条指定 where 条件的语句，返回结果集。此时事务 T2 插入一行新记录，恰好满足 T1 的 where 条件。然后 T1 使用相同的条件再次查询，结果集中可以看到 T2 插入的记录，这条新纪录则是幻读记录。

2． 不可重复读取： 事务 T1 读取一行记录，紧接着事务 T2 修改了 T1 刚刚读取的记录，然后 T1 再次查询，发现与第一次读取的记录不同，这称为不可重复读。

3. 脏读：事务 T1 更新了一行记录，还未提交所做的修改，这个 T2 读取了更新后的数据，然后 T1 执行回滚操作，取消刚才的修改，所以 T2 所读取的行就无效，也就是脏数据。

为了处理这些问题，SQL 标准定义了以下几种事务隔离级别：

1） READ UNCOMMITTED 幻读、不可重复读和脏读都允许。

2） READ COMMITTED 允许幻读、不可重复读，不允许脏读

3） REPEATABLE READ 允许幻读，不允许不可重复读和脏读

4） SERIALIZABLE 幻读、不可重复读和脏读都不允许

Oracle 数据库支持 READ COMMITTED 和 SERIALIZABLE 这两种事务隔离级别。所以 Oracle 不支持脏读。SQL 标准所定义的默认事务隔离级别是 SERIALIZABLE ,但是 Oracle 默认使用的是 READ COMMITTED。

1.10.在数据库中条件查询速度很慢的时候,如何优化？

参考答案：

1. 为经常出现在 WHERE 子句中的列创建索引；为经常出现在 ORDER BY、DISTINCT 后面的字段建立索引。如果建立的是复合索引，索引的字段顺序要和这些关键字后面的字段顺序一致；为经常作为表的连接条件的列上创建索引。

2. 减少表之间的关联

3. 优化 sql，尽量让 sql 很快定位数据，不要让 sql 做全表查询，应该走索引，把数据量大的表排在前面。

4. 简化查询字段，没用的字段不要，对返回的结果进行控制，尽量返回少量数据。

1.11.请说明数据库主键、外键的作用？

参考答案：

主键作用：能保证设置主键的列非空且唯一。另外，在定义主键时，如果这列之前没有索引，系统会为其创建唯一性索引

外键作用：能保证设置外键的列取值必须匹配父表中已有的值。通过外键可以与同一张表的列建立引用关系，也可以与不同表的列建立引用关系。

1.12.索引的优点和缺点是什么？

参考答案：

索引的优点如下：

1. 通过创建唯一性索引，可以保证数据库表中每一行数据的唯一性；

2. 可以大大加快数据的检索速度，这也是创建索引的最主要的原因；

3. 可以加速表和表之间的连接，特别是在实现数据的参考完整性方面特别有意义；

4. 在使用分组和排序子句进行数据检索时，同样可以显著减少查询中分组和排序的时间。

5. 通过使用索引，可以在查询的过程中，使用优化隐藏器，提高系统的性能。

索引的缺点如下：

1. 创建索引和维护索引要耗费时间，这种时间随着数据量的增加而增加；

2. 索引需要占物理空间，除了数据表占数据空间之外，每一个索引还要占一定的物理空间，如果要建立聚簇索引，那么需要的空间就会更大；

3. 当对表中的数据进行增加、删除和修改的时候，索引也要动态的维护，这样就降低了数据的维护速度；

4. 如果给不适合创建索引的列创建了索引，不会提高性能。

1.13.主键和索引的区别？

参考答案：

1．主键是为了标识数据库记录唯一性，不允许记录重复，且键值不能为空，主键也是一个特殊索引；

2．数据表中只允许有一个主键，但是可以有多个索引；

3．使用主键数据库会自动创建主索引，也可以在非主键上创建索引，方便查询效率.

4．索引可以提高查询速度，它就相当于字典的目录，可以通过它很快查询到想要的结果，而不需要进行全表扫描.

5．主键也可以由多个字段组成，组成复合主键，同时主键肯定也是唯一索引；

6．唯一索引则表示该索引值唯一，可以由一个或几个字段组成，一个表可以有多个唯一索引。

1.14.使用索引查询一定能提高数据库的性能吗？为什么？

参考答案：

不一定。

如果给不适合创建索引的列创建了索引，不会提高性能。

1.15.什么叫视图？

参考答案：

视图（VIEW）是一个虚拟表，其内容由查询定义。同真实的表一样，视图包含一系列带有名称的列和行数据。但是，视图并不在数据库中以存储的数据值集形式存在。行和列数据来自由定义视图的查询所引用的表，并且在引用视图时动态生成。

视图在数据库中不存储数据值，即不占空间。只在系统表中存储对视图的定义。

视图实际上就是一条 SELECT 语句，类似 windows 中的快捷方式。

1.16.数据库事务是什么，并分别说明 ACID？

参考答案：

事务，是指作为单个逻辑工作单元执行的一系列操作。 事务处理可以确保除非事务性单元内的所有操作都成功完成，否则不会永久更新面向数据的资源。通过将一组相关操作组合为一个要么全部成功要么全部失败的单元，可以简化错误恢复并使应用程序更加可靠。

事务的特性四个特性 ACID，分别表示：

原子性（atomicity）：一个事务或者完全发生、或者完全不发生；

一致性（consistency）：事务把数据库从一个一致状态转变到另一个状态；

隔离性（isolation）：在事务提交之前，其他事务觉察不到事务的影响；

持久性（durability）：一旦事务提交，它是永久的。

1.17.简要叙述数据库 TRUNCATE，DROP,DELETE 之间的区别？

参考答案：

1. truncate 删除表内容释放表空间保留表结构(即 :只删除表内的数据 ,不删除表本身。相当于 delete 语句不写 where 子句一样)，不使用事务处理即和事务无关。truncate 删除表内容，数据不可以回滚。delete 后面可跟 where 子句，truncate 则不可以。truncate 比 delete 的删除数据的速度快 。

2. delete 属于数据操作语言（DML），不能自动提交事务，需 commit 提交。这个操作会放到 rollback segement 中，事务提交之后才生效；如果有相应的 trigger，执行的时候将被触发。

3. drop 属于数据定义语言（DDL）可以自动提交事务；drop 语句将删除表的结构、依赖的约束(constraint)、触发器(trigger)、索引(index)； 删除表数据同时删除表结构；依赖于该表的存储过程/函数将保留,但是变为失效状态。

1.18.存储过程和触发器的区别？

参考答案：

存储过程是 SQL 语句和可选控制流语句的预编译集合 ,以一个名称存储并作为一个单元处理。存储过程存储在数据库内，可由应用程序通过一个调用执行，而且允许用户声明变量、有条件执行以及其它强大的编程功能。存储过程可包含程序流、逻辑以及对数据库的查询。它们可以接受参数、输出参数、返回单个或多结果集以及返回值。可以出于任何使用 SQL 语句的目的来使用存储过程，它具有以下优点：

1.可以在单个存储过程中执行一系列 SQL 语句。

2.可以从自己的存储过程内引用其它存储过程，这可以简化一系列复杂语句。

3.存储过程在创建时即在服务器上进行编译，所以执行起来比单个 SQL 语句快。

触发器是一种特殊类型的存储过程 ,当使用下面的一种或多种数据修改操作在指定表中对数据进行修改时，触发器会生效：UPDATE、INSERT 或 DELETE。触发器可以查询其它表，而且可以包含复杂的 SQL 语句。它们主要用于强制复杂的业务规则或要求。例如，可以控制是否允许基于顾客的当前帐户状态插入定单。触发器还有助于强制引用完整性，以便在添加、更新或删除表中的行时保留表之间已定义的关系。然而，强制引用完整性的最好方法是在相关表中定义主键和外键约束。如果使用数据库关系图，则可以在表之间创建关系以自动创建外键约束。

触发器的优点如下：

1.触发器是自动的：它们在对表的数据作了任何修改（比如手工输入或者应用程序采取的操作）之后即被激活。

2.触发器可以通过数据库中的相关表进行层叠更改。例如，可以在 titles 表的 title_id 列上写入一个删除触发器 ,以使其它表中的各匹配行采取删除操作。该触发器用 title_id 列作为唯一键，在 titleauthor、sales 及 roysched 表中对各匹配行进行定位。

3.触发器可以强制限制，这些限制比用 CHECK 约束所定义的更复杂。与 CHECK 约

束不同的是，触发器可以引用其它表中的列。例如，触发器可以回滚试图对价格低于 10 美元的书（存储在 titles 表中）应用折扣（存储在 discounts 表中）的更新。

1.19.存储过程和函数的区别？

参考答案：

存储过程是用户定义的一系列 sql 语句的集合，涉及特定表或其它对象的任务，用户可以调用存储过程，而函数通常是数据库已定义的方法，它接收参数并返回某种类型的值并且不涉及特定用户表。具体区别如下：

1. 对于存储过程来说可以返回参数，而函数只能返回值或者表对象。
2. 函数必须有返回值，存储过可有可无
3. 存储过程一般是作为一个独立的部分来执行，而函数可以作为查询语句的一个部分来调用。

1.20.数据库中有以下数据 ：

```
ID(pri)(Auto) name pass
 1             aaa  111
 2             bbb  222
 3             ccc  333
```

1. 请用一条 SQL 语句将现有的三条记录复制一下，达到以下的效果 ：

```
ID(pri) (Auto) name pass
1              aaa  111
2              bbb  222
3              ccc  333
4              aaa  111
5              bbb  222
6              ccc  333
```

2. 再用 sql 语句删除重复记录。

参考答案：

在 MySQL 数据库中，创建测试表和测试数据，代码如下所示：

```
create table info(
  id int primary key AUTO_INCREMENT,
  name varchar(20),
  pass varchar(20)
);
insert into info(name,pass)values('aaa','111');
insert into info(name,pass)values('bbb','222');
insert into info(name,pass)values('ccc','333');
```

1. 用一条 SQL 语句将现有的三条记录复制一下，代码如下所示：

```
insert into info(name,pass) select name, pass from info;
```

2. 删除重复记录的 SQL 语句如下所示：

```
create table tmp as select min(id) as col1 from info group by name,pass;
delete from info where id not in (select col1 from tmp);
drop table tmp;
```

1.21.表名：高考信息表

准考证号	科目	成绩
2006001	语文	119
2006001	数学	108
2006002	物理	142
2006001	化学	136
2006001	物理	127
2006002	数学	149
2006002	英语	110
2006002	语文	105
2006001	英语	98
2006002	化学	129

……

给出高考总分在 600 以上的学生准考证号。

参考答案：

在 MySQL 数据库中，创建测试表和测试数据，代码如下所示：

```
create table score(
   num int ,
   name varchar(20),
   score numeric
);
insert into score(num,name,score)values(2006001,'语文',119);
insert into score(num,name,score)values(2006001,'数学',108);
insert into score(num,name,score)values(2006002,'物理',142);
insert into score(num,name,score)values(2006001,'化学',136);
insert into score(num,name,score)values(2006001,'物理',127);
insert into score(num,name,score)values(2006002,'数学',149);
insert into score(num,name,score)values(2006002,'英语',110);
insert into score(num,name,score)values(2006002,'语文',105);
insert into score(num,name,score)values(2006001,'英语',98);
insert into score(num,name,score)values(2006002,'化学',129);
```

查询高考总分在 600 以上的学生准考证号的 SQL 语句如下所示：

```
select num from score group by num having sum(score)>600;
```

1.22.数据库有两张表一个学生表(id ,name ,sex),一个学生成绩表(id,chineses ,English ,math),要求查询学生基本信息以及各科成绩和总成绩，总成绩要求在 200 到 300 之间，学生姓名降序？

参考答案：

在 MySQL 数据库中，创建测试表和测试数据，代码如下所示：

```
create table student
   id int primary key,
   name varchar(20),
   sex  varchar(2)
);
create table grade(
   id int REFERENCES student(id),
   chineses numeric,
   english  numeric,
   math   numeric

);
insert into student(name ,sex) values('张三','男');
insert into student(name ,sex) values('李四','男');
insert into student(name ,sex) values('王五','男');

insert into grade(id,chineses,english,math) values(1,80,80,60);
insert into grade(id,chineses,english,math) values(2,50,50,50);
insert into grade(id,chineses,english,math) values(3,80,80,70);
```

查询学生基本信息以及各科成绩和总成绩，总成绩要求在 200 到 300 之间，学生姓名降序的 SQL 语句如下所示：

```
select name ,chineses,english,math,(chineses+english+math) sumscore
from student join grade
where student.id=grade.id
and (chineses+english+math) between  200 and 300
order by name desc;
```

1.23.现有关系数据库表如下:

学生表(学号 char(6)、姓名、性别、身份证号)；

课程表(课号 char(6)、 名称)；

成绩表(id、学号、课号、分数)。

用 sql 实现如下两道题：

1. 检索姓马的女同学情况(姓名、身份证号)；
2. 检索有一门或一门以上课程成绩大于等于 90 的所有学生信息(学号、姓名)。

参考答案：

在 MySQL 数据库中，创建测试表和测试数据，代码如下所示：

```
create table stu(
  stuNo char(6),
  name varchar(20),
```

```
    gender varchar(4),
    idcard varchar(18)
  );
  insert into stu(stuno,name,gender,idcard)values('100001',' 马 霞 ',' 女
','232301198006013421');
  insert into stu(stuno,name,gender,idcard)values('100002',' 马 立 军 ',' 男
','232301198006013422');
  insert into stu(stuno,name,gender,idcard)values('100003',' 黄 蓉 ',' 女
','232301198006013423');

  create table course(
    cno char(6),
    cname varchar(20)
  );

  insert into course(cno,cname)values('1','语文');
  insert into course(cno,cname)values('2','数学');
  insert into course(cno,cname)values('3','英语');


  create table score(
    id  int primary key AUTO_INCREMENT,
    stuno char(6) REFERENCES stu(stuno),
    cno   char(6)  REFERENCES course(cno),
    score  numeric
  );

  insert into score(stuno,cno,score)values('100001','1',80);
  insert into score(stuno,cno,score)values('100001','2',90);
  insert into score(stuno,cno,score)values('100001','3',95);

  insert into score(stuno,cno,score)values('100002','1',70);
  insert into score(stuno,cno,score)values('100002','2',80);
  insert into score(stuno,cno,score)values('100002','3',85);

  insert into score(stuno,cno,score)values('100003','1',60);
  insert into score(stuno,cno,score)values('100003','2',70);
  insert into score(stuno,cno,score)values('100003','3',80);
```

1. 检索姓马的女同学的姓名、身份证号的 SQL 语句如下所示：

```
select name ,idcard from stu where name like '马%' and gender='女';
```

2. 检索有一门或一门以上课程成绩大于等于 90 的所有学生的学号、姓名，SQL 语句如下所示：

```
select distinct stu.stuno,name from stu
join score
on stu.stuno=score.stuno
join course
on course.cno=score.cno
where score>=90;
```

1.24.有三张表,学生表 Student,课程 Course,学生课程表 SC,学生可以选修多门课程,一门课程可以被多个学生选修,通过 SC 表关联，详细要求如下：

1．写出建表语句；

2．写出 SQL 语句，查询选修了所有选修课程的学生；

3．写出 SQL 语句，查询选修了至少 2 门以上的课程的学生。

参考答案：

1. 在 Oracle 数据库中，创建表和测试数据的 SQL 语句如下所示：

```
create table student (
id number(10) primary key,
name varchar2(20));

create table course (
id number(10) primary key,
name varchar2(20));

create table sc(
sid number(10) references student(id),
cid number(10) references course(id),
grade number(4,2));

insert into student values(1,'feifei');
insert into student values(2,'jingjing');
insert into student values(3,'nannan');
insert into student values(4,'yuanyuan');
insert into student values(5,'jiejie');

insert into course values(1,'corejava');
insert into course values(2,'c++');
insert into course values(3,'jdbc');
insert into course values(4,'hibernate');

insert into sc values(1,1,98);
insert into sc values(2,1,97);
insert into sc values(3,1,94);
insert into sc values(4,1,92);
insert into sc values(5,1,93);
insert into sc values(1,2,94);
insert into sc values(2,2,92);
insert into sc values(3,2,95);
insert into sc values(5,2,97);
insert into sc values(1,3,92);
insert into sc values(2,3,92);
insert into sc values(4,3,91);
insert into sc values(1,4,99);
insert into sc values(3,4,89);
```

2. 查询选修了所有选修课程的学生的 SQL 语句如下所示：

```
select name from student where id in(select sid from sc group by sid having
count(*)=(select count(*) from course));
```

3. 查询选修了至少 2 门以上的课程的学生的 SQL 语句如下所示：

```
select name from student where id in(select sid from sc group by sid having
count(*)>=2);
```

1.25.表 class 和 student 结构如下，请完成后续 SQL 语句？

表 class

属性	类型(长度)	默认值	约束	含义
CLASSNO	数值 (2)	无	主键	班级编号
CNAME	变长字符 (10)	无	非空	班级名称

表 student

属性	类型(长度)	默认值	约束	含义
STUNO	数值 (8)	无	主键	学号
SNAME	变长字符 (12)	无	非空	姓名
SEX	字符 (2)	男	无	性别
BIRTHDAY	字符(8)	无	无	生日
EMAIL	变长字符 (20)	无	唯一	电子邮件
SCORE	数值 (5, 2)	无	检查	成绩
CLASSNO	数值 (2)	无	外键，关联到表 class 的 CLASSNO 主键	班级编号

测试 数据

STUNO	SNAME	SEX	BIRTHDAY	EMAIL	SCORE	CLASSNO
21	tom	男	19790203	tom@163.net	89.50	1
56	jerry	默认值	空	空	空	2

1. 修改表 student 的数据，将所有一班的学生成绩加 10 分。
2. 删除表 student 的数据 将所有 3 班出生日期晚于 1981 年 5 月 12 日的记录删除。
3. 按班级升序排序，成绩降序排序，查询 student 表的所有记录。
4. 查询 student 表中所有三班成绩为空的学生记录。
5. 表 student 与 class 联合查询，要求查询所有学生的学号、姓名、成绩、班级名称。
6. 按班级编号分组统计每个班的人数、最高分、最低分、平均分，按平均分降序排序。
7. 查询一班学生记录中所有成绩高于本班学生平均分的记录。
8. 查询所有学生记录中成绩前十名的学生的学号、姓名、成绩、班级编号。

参考答案：

在 Oracle 数据库中，创建测试表和测试数据，代码如下所示：

```
create table class(
  classno number(2) primary key,
  cname varchar2(10)
);
create table student(
  stuno number(18) primary key,
  sname varchar2(12),
  sex  char(2) default '男',
  birthday char(8),
  email varchar2(20) unique,
  score number(5,2) check( score between 0 and 100),
  classno number(2) references class(classno)
```

```
);

insert into class (classno,cname)values(1,'一班');
insert into class (classno,cname)values(2,'二班');
insert into class (classno,cname)values(3,'三班');

insert into student(stuno,sname,sex,birthday,email,score,classno)values(21,'tom','男','19790203','tom@163.net',89.50,1);
insert into student(stuno,sname,birthday,email,score,classno)values(56,'jerry',null,null,89.50,2);
```

1. 修改表 student 的数据，将所有一班的学生成绩加 10 分，SQL 语句如下所示：

```
update student set score=score+10
where classno=(select classno from class where cname='一班');
```

2. 删除表 student 的数据 将所有 3 班出生日期晚于 1981 年 5 月 12 日的记录删除，SQL 语句如下所示：

```
delete student
where to date(birthday,'yyyymmdd')<to date('19810512','yyyymmdd')
and classno=(select classno from class where cname='三班');
```

3. 按班级升序排序 ,成绩降序排序 ,查询 student 表的所有记录 ,SQL 语句如下所示：

```
select * from student order by classno,score desc;
```

4. 查询 student 表中所有三班成绩为空的学生记录，SQL 语句如下所示：

```
select * from student where classno=(select classno from class where cname='三班') and score is null;
```

5. 表 student 与 class 联合查询，要求查询所有学生的学号、姓名、成绩、班级名称，SQL 语句如下所示：

```
select s.stuno,s.sname,s.score,c.cname from student s join class c on s.classno=c.classno;
```

6. 按班级编号分组统计每个班的人数、最高分、最低分、平均分，按平均分降序排序，代码如下所示：

```
select count(*) "人数",max(score) "最高分",min(score) "最低分",avg(nvl(score,0)) "平均分" from student group by classno order by avg(nvl(score,0));
```

7. 查询一班学生记录中所有成绩高于本班学生平均分的记录，代码如下所示：

```
select * from student
where score>(select avg(nvl(score,0)) from student where classno=(select
classno from class where cname='一班'))
and classno=(select classno from class where cname='一班');
```

8. 查询所有学生记录中成绩前十名的学生的学号、姓名、成绩、班级编号，代码如下所示：

```
select stuno,sname,score,classno from
(select stuno,sname,score,classno from student order by score desc)
where rownum<=10;
```

1.26.有两张表 student 和 score：

Student:学号、姓名、性别、年龄；

Score:学号、语文、数学、英语；

1.查询张三的学号、姓名、性别、语文、数学、英语；

2.查询语文比数学好的同学；

3.查出姓名相同的学生学号。

参考答案：

在 Oracle 数据库中，创建测试表和测试数据，代码如下所示：

```
create table student(
  stuno number(8) primary key,
  sname varchar2(12),
  sex  char(2) default '男',
  age  int
);

create table score(
  stuno number(8),
  chinese number(3),
  math   number(3),
  english  number(3)
);
insert into STUDENT (STUNO, SNAME, SEX, AGE)
values (1, 'tom', '男', 22);
insert into STUDENT (STUNO, SNAME, SEX, AGE)
values (2, 'terry', '男', 21);
insert into STUDENT (STUNO, SNAME, SEX, AGE)
values (3, 'marry', '女', 23);
insert into STUDENT (STUNO, SNAME, SEX, AGE)
values (4, 'marry', '女', 23);
insert into SCORE (STUNO, CHINESE, ENGLISH, MATH)
values (1, 70, 80, 90);
insert into SCORE (STUNO, CHINESE, ENGLISH, MATH)
values (2, 60, 80, 70);
insert into SCORE (STUNO, CHINESE, ENGLISH, MATH)
values (3, 70, 85, 95);
insert into SCORE (STUNO, CHINESE, ENGLISH, MATH)
```

```
values (4, 98, 85, 95);
```

1. 查询张三的学号、姓名、性别、语文、数学、英语，SQL 语句如下所示：

```
select s.stuno,sname,sex,chinese,math,english from student s join score c on s.stuno=c.stuno and sname='张三';
```

2. 查询语文比数学好的同学，SQL 语句如下所示：

```
select s.stuno,sname,sex,chinese,math from student s join score c on s.stuno=c.stuno and chinese>math;
```

3. 查出姓名相同的学生学号，SQL 语句如下所示：

```
select stuno from student where sname in(select sname from student group by sname having count(sname)>1);
```

1.27.对一个用户登录模块，要求每个用户只允许 3 次登录错误，超过则将锁定此帐户？

参考答案：

基于数据库，在数据库中增加两个字段,分别记录次数和第三次的登录时间，当大于 3 的那次登陆时，判定当前时间和记录的第三次的时间，如果在 10 分钟内，则依然不能登录如果超了 10 分钟，则使得次数和时间重置。

锁定后可以通过以下方式解锁，可以给每个账户都设一把密钥，当其中任何一个账户被锁定时，系统会自动以短信+电子邮件的形式生成一个随机的密钥发送到对应的用户，该用户想解锁可以等待时间过时，也可以重新输入密钥解锁。也就是说，只要被锁定了以后，除非用户名+密码+密钥全部通过才能解锁。

1.28.数据脚本

```
create table test1
(
  pici    VARCHAR2(30),
  busicode VARCHAR2(50),
  amt     NUMBER,
  flag    VARCHAR2(1)
);
-- Add comments to the columns comment on column test1.flag is '1 表示成功 2 表示失败';

insert into test1 values('20130201','0201111',10,1);
insert into test1 values('20130201','0201112',5,2);
insert into test1 values('20130201','0201113',10,2);
insert into test1 values('20130201','0201114',5,1);

insert into test1 values('20130202','0202111',10,1);
insert into test1 values('20130202','0202112',20,1);
insert into test1 values('20130202','0202113',20,1);
insert into test1 values('20130202','0202114',20,1);
```

```
insert into test1 values('20130203','0203111',10,2);
insert into test1 values('20130203','0203111',10,2);
insert into test1 values('20130203','0203111',10,2);
```

实现要求：标识位 flag， 1 表示扣款成功、2 表示扣款失败，使用一个 SQL 查询出每天扣款成功笔数、成功金额、失败笔数、失败金额。

参考答案：

查询每天扣款成功笔数、成功金额、失败笔数、失败金额的 SQL 语句如下所示：

```
select to_date(pici,'yyyymmdd') as "日期",
   sum(case when flag=1 then 1 else 0 end) as "成功笔数",
   sum(case when flag=1 then amt else 0 end) as "成绩金额",
   sum(case when flag=2 then 1 else 0 end) as "失败笔数",
   sum(case when flag=2 then amt else 0 end) as "失败金额"
 from test1
 GROUP by to_date(pici,'yyyymmdd');
```

1.29.数据库题

1．创建 sms 表的语句；

2．写出 users 与 sms 左关联的查询语句。

参考答案：

1. 在 Oracle 数据库中，创建 sms 表和 users 表的 SQL 语句如下：

```
create  table sms(
  id number primary key,
     sname  varchar2(20)
);
create table users(
  id number primary key,
  username varchar2(20),
  smsid number references sms(id)
);
```

2. users 与 sms 左关联的查询语句的 SQL 语句如下：

```
select sname,username from users u left join sms s on u.smsid=s.id;
```

1.30.书表(books)

books 表字段及测试数据

book_id	book_name	creatdate	Lastmodifydate	decription
001	三个人的世界	2005-02-02	2005-07-07	NULL

作者表(authors)字段及测试数据

a_id	a_name
01	王纷
02	李尚
03	泰和

部门表(depts)字段及测试数据

d_id	d_name
001	编辑一部
002	编辑二部
003	编辑三部

书和作者关联表(bookmap)字段及测试数据

d_id	a_id
001	01
001	02
001	03

查询出每个部门的所写的总书量，比如：一本书有 3 个人写，如果三个人在不同的部门，则每个部门的总书量则为 1，最后结果如下:

部门 , 书量

编辑一部 , 1

编辑二部 , 1

编辑三部 , 1

参考答案：

在 Oracle 数据库中，创建测试表和测试数据，代码如下所示：

```
create table books (
book_id number(20) primary key,
book_name varchar2(200),
creatdate varchar2(200),
Lastmodifydate varchar2(200),
decription varchar2(200)
);
insert into books (book_id,book_name,creatdate,Lastmodifydate,decription)
values ('001','三个人的世界','2005-02-02','2005-07-07','NULL');

create table authors (
A_id number(20) primary key,
A_name varchar2(200)
);
insert into authors (A_id,A_name) values ('01','王纷');
insert into authors (A_id,A_name) values ('02','李尚');
insert into authors (A_id,A_name) values ('03','泰和');

create table depts (
d_id number(20) primary key,
```

```
d_name varchar2(200)
);
insert into depts (d_id,d_name) values ('001','编辑一部');
insert into depts (d_id,d_name) values ('002','编辑二部');
insert into depts (d_id,d_name) values ('003','编辑三部');

create table bookmap(
  book_id number(20) references books(book_id),
  a_id  number(20) references authors(a_id)

);
insert into bookmap(book id,a id)values(001,01);
insert into bookmap(book id,a id)values(001,02);
insert into bookmap(book_id,a_id)values(001,03);

create table depmap(
  d_id number(20) references depts(d_id),
  a_id number(20) references authors(a_id)
);
insert into depmap(d id,a id)values(001,01);
insert into depmap(d id,a id)values(002,02);
insert into depmap(d_id,a_id)values(003,03);
```

查询出每个部门的所写的总书量的 SQL 语句如下所示：

```
SELECT a.d_name "部门" ,count(c.book_id) "书量"
 FROM depts a, authors b, books c, depmap d, bookmap e
 WHERE a.d id=d.d id
  AND d.a id=b.a id
  AND c.book id=e.book id
  AND e.a id=b.a id
 GROUP BY a.d_name;
```

1.31.两个表情况如下：

表名：wu_plan

ID	plan	model	corp_code	plannum	prixis
1	00001	exx22	nokia	2000	0
2	00002	lc001	sony	3000	0

表名：wu_bom

ID	plan	pact	amount
1	00001	aa1	300
2	00001	aa2	200
3	00002	bb1	500
4	00002	bb2	800
5	00002	bb3	400

查询这两个表中 plan 唯一，每一个 plan 中，amount 最少的，plannum 大于 prixis 的记录。结果为:

ID	plan	model	corp_code	plannum	prixis	pact	amount
1	00001	exx22	nokia	2000	0	aa2	200

2 00002 lc001 sony 3000 0 bb3 400

参考答案：

在 Oracle 数据库中，创建测试表和测试数据，代码如下所示：

```
CREATE TABLE wu_bom (
  id number(20) ,
  plan varchar2(20) ,
  pact varchar2(20) ,
  amount number(11)
) ;

insert  into wu_bom(id,plan,pact,amount) values (1,'00001','aa1',300);
insert  into wu_bom(id,plan,pact,amount) values(2,'00001','aa2',200);
insert  into wu_bom(id,plan,pact,amount) values(3,'00002','bb1',500);
insert  into wu_bom(id,plan,pact,amount) values(4,'00002','bb2',800);
insert  into wu_bom(id,plan,pact,amount) values(5,'00002','bb3',400);

CREATE TABLE wu plan (
  id number(20) ,
  plan varchar2(20),
  model varchar2(20),
  corp_code varchar2(20) ,
  plannum number(11) ,
  prixis number(11)
);

insert     into   wu plan(id,plan,model,corp code,plannum,prixis)   values
(1,'00001','exx22','nokia',2000,0);
insert     into   wu_plan(id,plan,model,corp_code,plannum,prixis)   values
(2,'00002','lc001','sony',3000,0);
```

查询这两个表中 plan 唯一，每一个 plan 中，amount 最少的，plannum 大于 prixis 的记录，SQL 语句如下所示：

```
 select A.*, B.pact,B.minamount as amount from wu_plan A,
  (select plan,max(pact) pact, min(amount) as minamount from wu_bom group by
plan) B where A.plan=B.plan and A.plannum>A.prixis;
```

1.32.数据库方面：

```
create table tbl threat
( pk threat id        int unsigned not null auto increment,
  dt_log_time         datetime,                 //发生时间
  i_severity          int,                      //严重程度
  i_device_id         int,                      //设备 id
  str_tr_type         varchar(64),               //告警类型
  str_tr_name         varchar(256),              //类型名称
  i_work_id           int unsigned,              //工单 id
  dt_complete_time     datetime,                 //完成时间
  i_status    tinyint default 0 comment '0-新分派,1-重新激活 2-完成',//告警状态
  primary key (pk_threat_id)
);

create table tbl_work
( pk_work_id          int not null auto_increment,
```

```
    str_title          varchar(256),              //工单名称
    i_owner_org_id      int unsigned,             //负责人组织 id
    i_owner_id         int,                     //负责人 id
    dt_dispatch_time    datetime,                //派单时间
    dt_finish_time     datetime,                 //完成时间
    i_is_history       tinyint default 0,              //是否为历史工单
    dt_expect_time     datetime,                //期望完成时间
    i_in_time          tinyint(1) default 0,           //及时性
    i_priority         tinyint(1),               //优先级
    primary key (pk_work_id)
);
```

ps: threat 为告警、work 为工单、i_severity 严重程度（0 代表一般、1 代表低危、2 代表中危、3 代表高危）。

1. 请查出当前日期前三天发生的告警的类型名称、告警类型、发生时间、严重程度及派单时间用日期按照降序排列。

2. 请查出当前日期前三天发生的告警的数量最多的告警类型及数量，数量按降序排列的前 5 个。

参考答案：

在 MySQL 数据库中，创建测试表和测试数据，代码如下所示：

```
insert into tbl_threat values(2,'20131028',1,1,'s','s',1,'20131029',0);
insert into tbl_threat values(3,'20131027',1,1,'s','s',1,'20131029',0);
insert into tbl_threat values(4,'20131025',1,1,'s','s',1,'20131029',0);
insert into tbl_threat values(5,'20131025',1,1,'r','r',1,'20131029',0);
insert into tbl_threat values(6,'20131027',1,1,'r','r',1,'20131029',0);
insert into tbl_threat values(7,'20131027',1,1,'r','r',1,'20131029',0);
insert into tbl_threat values(8,'20131027',1,1,'r','r',1,'20131029',0);
insert into tbl_threat values(9,'20131027',1,1,'r','r',1,'20131029',0);
insert into tbl_threat values(10,'20131027',1,1,'y','y',1,'20131029',0);
insert into tbl_work values
(1,'e',1,1,'20131027','20131027',0,'20131027',0,1)
```

1. 查出当前日期前三天发生的告警的类型名称、告警类型、发生时间、严重程度及派单时间用日期按照降序排列，SQL 语句如下所示：

```
select  str_tr_name,str_tr_type,dt_log_time,i_severity,dt_dispatch_time
from tbl_threat  thread join tbl_work work
on  work.pk_work_id =thread.i_work_id
and(  dt_log_time=(CURRENT_DATE-1)  or  dt_log_time=(CURRENT_DATE-2)  or
dt_log_time=(CURRENT_DATE-3))
order by dt_dispatch_time ;
```

2. 查出当前日期前三天发生的告警的数量最多的告警类型及数量，数量按降序排列的前 5 个，如下所示：

```
select * from (select  str tr type,count(str tr type) quantity
from tbl threat
where  dt_log_time=(CURRENT_DATE-1)  or  dt_log_time=(CURRENT_DATE-2)  or
```

```
dt log time=(CURRENT DATE-3)
    group by  str tr type ) s
    order by quantity desc
    limit 5 ;
```

1.33.设有图书管理数据库：

图书(总编号 C(6)、分类号 C(8)、书名 C(16)、作者 C(6)、出版单位 C(20)、单价 N(6,2))。

1. 检索书价在 15 元至 25 元(含 15 元和 25 元)之间的图书的书名、作者、书价和分类号，结果按分类号升序排序。

2. 为图书表建立一个视图。

参考答案：

在 Oracle 数据库中，创建测试表和测试数据，代码如下所示：

```
create table book(
  no  varchar2(6),
  type varchar2(8),
  bookname varchar2(16),
  author varchar2(6),
  company varchar2(20),
  price  number(6,2)
);
```

1. 检索书价在 15 元至 25 元(含 15 元和 25 元)之间的图书的书名、作者、书价和分类号，结果按分类号升序排序，SQL 语句如下所示：

```
select bookname,author,price,type from  book where price between 15 and 25
order by type;
```

2. 为图书表建立一个视图，SQL 语句如下所示：

```
create or replace view v book
as
select * from book;
```

1.34.写一个 oracle 函数 ,输入参数(字符串 str ,整型 len ,字符 c)返回字符串 rstr ?

要求：如果字符串 str 的长度小于 len, 则返回的字符串 rstr 为在字符串 str 前填充字符 c 达到长度为 len 的字符串。 如果字符串 str 的长度大于等于 len ,则返回的字符串 rstr 为 str。

参考答案：

在 Oracle 数据库中创建 get_str 函数的 SQL 语句如下所示：

```
create or replace function get str(str in varchar2, len in int, c in char)
return varchar2 is
   Result varchar2(1000);

   v len int;
   rstr varchar2(1000);
```

```
begin
  rstr := str;
  v_len := length(str);
  if v_len < len then
     while(v_len < len) loop
         rstr:= c|| rstr;
         v len := v len + 1;
       end loop;
  end if;
  Result :=rstr;
  return(Result);
end get_str;
```

1.35.表结构如下：

1. 表名： g_cardapply

字段 （字段名/类型/长度）：

g_applyno varchar 8; //申请单号 (关键字)

g_applydate bigint 8; //申请日期

g_state varchar 2; //申请状态

2. 表名： g_cardapplydetail

字段(字段名/类型/长度):

g_applyno varchar 8; // 申请单号

g_name varchar 30; //申请人姓名

g_idcard varchar 18; // 申请人身份证号

g_state varchar 2; // 申请状态

其中，两个表的关联字段为申请单号。详细要求如下：

1. 查询身份证号码为 440401430103082 的申请日期；
2. 查询同一个身份证号码有两条以上记录的身份证号码及记录个数；
3. 将身份证号码为 440401430103082 的记录在两个表中的申请状态均改为 07；
4. 删除 g_cardapplydetail 表中的所有李姓记录。

参考答案：

在 MySQL 数据库中，创建测试表和测试数据的 SQL 语句如下所示：

```
create table g_cardapply(
  g_applyno varchar(8),
  g_applydate bigint(8),
  g_state    varchar(2)
);

create table g_cardapplydetail(
    g_applyno varchar(8),
    g_name varchar(30),
    g_idcard varchar(18),
    g_state  varchar(2)
);

insert into  g cardapply values(1,'20130909','Y');
insert into g_cardapplydetail values(1,'校长','440401430103082 ','01');
insert into g_cardapplydetail values(2,'校长','440401430103082 ','01');
```

```
insert into g_cardapplydetail values(2,'李长江','440401430103083 ','02');
```

1. 查询身份证号码为 440401430103082 的申请日期，SQL 语句如下所示：

```
select  apply.g applydate
from g cardapply apply join g cardapplydetail detail
on apply.g_applyno = detail.g_applyno and detail.g_idcard='440401430103082';
```

2. 查询同一个身份证号码有两条以上记录的身份证号码及记录个数，SQL 语句如下所示：

```
select g idcard ,count(g idcard) as xcount from g cardapplydetail
Group by g_idcard having count(g_idcard)>=2;
```

3. 将身份证号码为 440401430103082 的记录在两个表中的申请状态均改为 07 ,更新 g_cardapplydetail 表的 SQL 语句如下所示：

```
update g_cardapplydetail set g_state='07' where g_idcard='440401430103082' ;
```

更新 g_cardapply 表的 SQL 语句如下所示：

```
update g cardapply set  g state='07'
where  g_applyno in (select  g_applyno
from  g_cardapplydetail where g_idcard='440401430103082');
```

4. 删除 g_cardapplydetail 表中的所有李姓记录，SQL 语句如下所示：

```
delete from g_cardapplydetail where g_name like '李%';
```

1.36.参见如下表结构回答问题

出版社:
出版社代码 char(2),
出版社名称 varchar2(32)

图书:
图书编号 char(8),
图书名称 varchar2(128),
出版社代码 char(2),
作者代号 char(4),
图书简介 varchar2(128)

作者:
作者代号 char(4),

作者名称 varchar2(10),

性别 char(1),

年龄 number(3),

文学方向 varchar2(64)

获奖名单:

获奖日期 date,

获奖人员 char(4)

详细要求如下：

1. 编写 SQL 语句，找出“作者”库中没有出现在“获奖名单”库中所有作者信息的 SQL 语句，要求使用 not in、not exists 以及外关联三种方法，并说明哪种方法最优。

2. “获奖名单”表，写出 SQL 语句，查询出在上一个月获奖的人员。

参考答案：

在 Oracle 数据库中，创建测试表和测试数据：

```
create table author(
 aid char(4),
 aname char(10),
 sex char(1)
);
insert into author values('1','john','1');
insert into author values('2','karl','0');
insert into author values('3','foul','0');
insert into author values('4','jell','1');

create table award(
  adate date,
  aid char(4)
);
insert into award values(sysdate,'1');
insert into award values((sysdate-1),'4');
```

1.当数据量比较大时 not exists 的方式效率最高。

使用 not in 的 SQL 语句写法如下所示：

```
select * from  author where aid not in ( select aid from award);
```

使用 not exists 的 SQL 语句写法如下所示：

```
select * from author a where not exists (select 1 from award b where
b.aid=a.aid);
```

使用外联的 SQL 语句写法如下所示：

```
select  a.* from  author a left join award b on  a.aid=b.aid where b.aid is
null;
```

2. 查询出在上一个月获奖的人员的 SQL 语句如下所示：

```
select a.aname  from award w join author a on a.aid=w.aid
Where to_char(adate , 'yyyy-mm')=to_char( add_months(sysdate,-1),'yyyy-mm')
```

1.37.某公司信息管理系统的需求分析和部分关系模式的结果描述如下：

1. 公司有多个部门，每个部门有一名负责人、一间办公室、一部电话、多名职员，每个职员最多属于一个部门，负责人也是公司一名职员。

2. 数据库的部分关系模式设计如下：

职员（职员号，职工姓名，月工资，部门号，办公室，电话）

部门（部门号，部门名，负责人代码，任职时间）

请回答下述问题：

1. 根据上述说明，请分别给出”职员”和”部门”关系模式的主键及外键。

2. 请编写 SQL 语句，针对人数大于等于 10 的部门创建视图 D_View（Dept,D_num, D_Avgpay），其中，Dept 为部门号，D_num 为部门人数，D_Avgpay 为平均工资。

3. 目前的“职员”关系模式存在什么问题？在不增加新关系模式的前提下，请给出修改后的“职员”和“部门”关系模式。

参考答案：

1. 职员表的职员号作为主键，部门表的部门号作为主键，职员表的部门号作为外键。

2. 针对人数大于等于 10 的部门创建视图 D_View 的 SQL 语句如下所示：

```
create or replace view D_View
as
select 部门号 as detp,count(职员号) as d_num,avg(月工资) as D_Avgpay from 职员表 group by 部门号 having count(职员号)>=10;
```

3.无法从职员表中查询出职员的领导和任职时间，修改后的关系模型如下：

职员（职员号、职工姓名、月工资、部门号、办公室、电话、负责人代码、任职时间）

部门（部门号、部门名 ）

1.38.请看如下数据库操作要求：

1. 建一个员工信息表 employee, 表中 id （员工代码)、sex (员工性别)、name(姓名）、departmentid (部门代码) 、address (地址)，birthdate (生日)、postcode (邮编) 、salary〔薪水)、workdate (入职日期) 、remark （备注信息)，其中 postcode 、 remark 可以为空， 薪水需为 number 类型，生日、入职日期为 date 类型。以员工代码作为主键。

2. 插入两条记录，id 分别为 0023, 1023，其余信息自己编造。

3. 查询员工总数 、薪水总额。

4. 查询出各部门的最小年龄、最大年龄。

5. 创建入职日期(workdate)索引 employee_idx 。

6. 修改 id 为 0023 的员工的入职日期为 2007-12-31 。

7. 删除 id 为 1023 的员工信息。

8. 使前面所做的修改、删除生效（假定数据库设置不是自动生效）。

9.假定有一表结构和 employee 完全一样的表 employee_bak ,把 employee 表的数据完全导入 employee_bak 表。

10.假设还有一表 duty，其记录为员工的级别 level，也是以员工 id 为主键，根据表 employee 、duty 查询出级别在 10 级以上的所有员工详细信息。

11.不区分部门查询出入职日期最早的 10 位员工信息。

12.删除索引 employee_idx。

参考答案：

1.创建 employee 表的 SQL 语句如下：

```
create table employee(
  id number(10) primary key,
  sex char(2) not null,
  name varchar2(20) not null,
  deptmentid number(4) not null,
  address varchar2(50) not null,
  birthdate date  not null,
  postcode number(6),
  salary number(10,2) not null,
  workdate date not null,
  remark varchar2(100)
);
```

2. 向 employee 表中插入测试数据的 SQL 语句如下所示：

```
insert into employee
(id,sex,name,deptmentid,address,birthdate,postcode,salary,workdate,remark)
values(0023,'女','黄蓉',12,'北京市',to_date('1985-12-1','yyyy-mm-dd'),100000,2345,to_date('2013-6-1','yyyy-mm-dd'),'共产党好');
insert into employee
(id,sex,name,deptmentid,address,birthdate,postcode,salary,workdate,remark)
values(1023,'男','郭靖',12,'北京市',to date('1985-12-1','yyyy-mm-dd'),100000,5345,to date('2013-7-1','yyyy-mm-dd'),'共产党好');
```

3. 查询员工总数 、薪水总额的 SQL 语句如下所示：

```
select count(*),sum(salary) from employee;
```

4. 查询出各部门的最小年龄、最大年龄的 SQL 语句如下所示：

```
select min(birthdate),max(birthdate) from employee;
```

5. 创建入职日期(workdate)索引 employee_idx 的 SQL 语句如下所示：

```
create index employee_idx on employee(workdate);
```

6. 修改 id 为 0023 的员工的入职日期为 2007-12-31 的 SQL 语句如下所示：

```
update employee set workdate=to date('2007-12-31','yyyy-mm-dd') where
id=0023;
```

7. 删除 id 为 1023 的员工信息的 SQL 语句如下所示：

```
delete from employee where id=1023;
```

8. 使前面所做的修改，删除生效，SQL 语句如下所示：

```
commit;
```

9. 把 employee 表的数据完全导入 employee_bak 表，SQL 语句如下所示：

```
create table employee_bak
as
select * from employee;
```

10. 根据表 employee 、duty 查询出级别在 10 级以上的所有员工详细信息，SQL 语句如下所示：

```
select * from employee e join duty d on e.id=d.id and level>10;
```

11. 不区分部门查询出入职日期最早的 10 位员工信息，SQL 语句如下所示：

```
select * from (select * from employee order by workdate
                                                                    )
where rownum<=10;
```

12. 删除索引 employee_idx，SQL 语句如下所示：

```
drop index employee_idx;
```

1.39.有一表格（T_user）有如下数据：

Id name age gender

----

1 张三 20 男

2 李四 22 男

3 张三 20 男

4 王五 21 男

5 王五 20 男

请用 SQL 查询出姓名相同而且年龄也相同的人员姓名。

参考答案：

在 Oracle 数据库中，创建测试表和测试数据，代码如下所示：

```
create table t_user(
  id number,
  name varchar2(20),
  age number(3),
  gender char(2)

);
insert into t_user values(1,'张三',20,'男');
insert into t_user values(2,'李四',22,'男');
insert into t_user values(3,'张三',20,'男');
insert into t_user values(4,'王五',20,'男');
insert into t_user values(5,'王五',21,'男');
```

查询出姓名相同而且年龄也相同的人员姓名，SQL 语句如下所示：

```
select distinct a.name from t user a join t user b  on a.name=b.name and
a.age=b.age and a.id<>b.id;
```

1.40.下面是某班级进行期末考试的相关数据表：

学生信息表(T_Student)

学生 ID	学生姓名
1	小明
2	小刚
3	小红

课程信息表(T_Course)

课程编号	课程名称
A	语文
B	数学
C	英语

考试成绩表(T_Score)

学生 ID	课程编号	分数
1	A	82
1	C	95
2	A	54
2	C	62

1. 按照区间对学生成绩进行优良评级，>=85 对应“优”；>=75 并且<85 对应“良”；>=60 并且<75 对应“及格”，<60 对应“不及格”（输出列：学生姓名，课程名称，成绩级别）

2. 找出本次考试平均分高于 70 的学生姓名（输出列：学生姓名）

参考答案：

在 Oracle 数据库中，创建测试表和测试数据，代码如下所示：

```
create table t_student(
  stu_id number primary key,
  name varchar2(20)
);
insert into t_student  values(1,'小明');
insert into t_student  values(2,'小刚');
insert into t_student  values(3,'小红');
create table t_course(
  course_id varchar2(20) primary key,
  name   varchar2(20)
);
insert into t_course values('A','语文');
insert into t_course values('B','数学');
insert into t_course values('C','英语');
create table t score(
  stu id number references t student(stu id),
  course id varchar2(20) references t course(course id),
  score number(5,2)
);

insert into t score values(1,'A',82);
insert into t score values(1,'C',95);
insert into t score values(2,'A',54);
insert into t_score values(2,'C',62);
```

1. 按照区间对学生成绩进行优良评级，>=85 对应“优”；>=75 并且<85 对应“良”；>=60 并且<75 对应“及格”，<60 对应“不及格”，SQL 语句如下所示：

```
select stu.name,cour.name,
    case when sc.score>=85  then '优'
     when sc.score>=75 and sc.score<85  then '良'
     when sc.score>=60  and sc.score<75 then '及格'
     when sc.score<60  then '不及格' end
    as "成绩级别"
from t_student stu
```

```
join t score sc
on stu.stu id=sc.stu id
join t_course cour
on sc.course_id=cour.course_id ;
```

2. 找出本次考试平均分高于 70 的学生姓名，SQL 语句如下所示：

```
select stu.name
from t_student stu
join t score sc
on stu.stu_id=sc.stu_id
group by stu.name
having avg(sc.score)>70;
```

1.41.分别写出 Oracle 数据库和 MySQL 数据库对 t_employees 表分页的 SQL 语句，要求每页输出 20 条？

参考答案：

1. Oracle 数据库实现的分页，SQL 语句如下所示：

```
SELECT * FROM
(
SELECT A.*, ROWNUM RN
FROM (SELECT * FROM t employees) A
WHERE ROWNUM <= 40
)
WHERE RN >= 21
```

2. MySql 数据库实现的分页，SQL 语句如下所示：

```
select *  from t_employees limit 20,20;
```

1.42.阅读以下说明，回答问题 1 至问题 4：

某宾馆需要建立一个住房管理系统，部分的需求分析结果如下：

1． 一个房间有多个床位，同一房间内的床位具有相同的收费标准；不同房间的床位收费标准可能不同。

2． 每个房间有房间号（如 201、202 等）、收费标准、床位数目等信息。

3． 每位客人有身份证号码、姓名、性别、出生日期和地址等信息。

4． 对每位客人的每次住宿，应该记录其入住日期、退房日期和预付款额信息。

5． 管理系统可查询出客人所住房间号。

根据以上的需求分析结果，设计一种关系模型下图所示所示：

住房管理系统的实体关系图

1. 根据上述说明和实体-关系图，得到该住房管理系统的关系模式如下所示，请在下划线处补充住宿关系？

房间(房间号、收费标准、床位数目)

客人(身份证号、姓名、性别、出生日期、地址)

住宿(_____、_____ 、入住日期、退房日期、预付款额)

2. 请给出问题 1 中住宿关系的主键和外键。

3. 若将上述各关系直接实现为对应的物理表，现需查询在 2005 年 1 月 1 日到 2005 年 12 月 31 日期间，在该宾馆住宿次数大于 5 次的客人身份证号，并且按照入住次数进行降序排列。下面是实现该功能的 SQL 语句，请填补语句中的空缺。

```
SELECT 住宿.身份证号,count(入住日期) FROM 住宿，客人
WHERE 入住日期 >= '20050101' AND 入住日期 <= '20051231'
AND 住宿.身份证号 = 客人.身份证号
GROUP BY ___(1)____
___(2)____ count(入住日期) > 5
____(3)___;
```

4. 为加快 SQL 语句的执行效率，可在相应的表上创建索引。根据问题 3 中的 SQL 语句， 除主键和外键外，还需要在哪个表的哪些属性上创建索引，应该创建什么类型的索引，请说明原因。

参考答案：

题目中，各问题的参考答案如下：

1. 房间号，身份证号。

2. 房间表的房间号做主键，客人表的身份证号做主键，住宿表的房间号和身份证号做外键。

3. (1)处：住宿.身份证号

(2)处：having

(3)处：order by count(住宿.身份证号) desc

4. 在住宿表的入住日期上创建非唯一性索引，用于提高查询效率。

1.43.假如有一个制造商生产若干种产品，并由不同的销售负责销售这些产品，产品和销售商之间是多对多的关系，数据库结构如下：

Table1：制造商制造的产品表

制造商制造的产品表(Table1)

字段名称	字段类型	字段长度	字段含义
ProductID	文本	6	产品编号 (Primary Key)
ProductName	文本	20	产品名称

ProductMeno	文本	50	产品说明

Table2：销售商表

销售商表(Table2)

字段名称	字段类型	字段长度	字段含义
SalesID	文本	4	销售商编号（Primary Key)
SalesName	文本	20	销售商名称
SalesPhone	文本	15	销售商联系电话

Table3：销售商销售的产品表

销售商销售的产品表(Table3)

字段名称	字段类型	字段长度	字段含义
SalesID	文本	4	销售商编号
ProductID	文本	6	产品编码

请用 SQL 语句实现以下功能：

1. 查询所有负责销售编号为 p00001 的产品的销售商的编号，名称和联系电话；
2. 查询各种产品的销售的数量；
3. 查询编号为 s001 的销售商所不经销的产品的编码和名称。

参考答案：

在 Oracle 数据库中，创建表和测试数据的 SQL 语句如下所示：

```
create table table1(
   productid varchar2(6) primary key,
   productname varchar2(20),
   productmemo varchar2(50)

);
insert into table1 values('p00001','apple','香甜可口');
insert into table1 values('p00002','pear','香甜可口');
insert into table1 values('p00003','banana','香甜可口');
create table table2(
  salesid varchar2(4) primary key,
  salesname varchar2(20),
  salesphone varchar2(15)
);
insert into table2 values('s001','marry','12345678908');
insert into table2 values('s002','tom','12345678909');
create table table3(
  salesid varchar2(4) references table2(salesid),
  productid varchar2(6) references table1(productid)
);
insert into table3 values('s001','p00001');
insert into table3 values('s001','p00002');
insert into table3 values('s001','p00003');
insert into table3 values('s002','p00001');
insert into table3 values('s002','p00002');
```

1. 查询所有负责销售编号为 p00001 的产品的销售商的编号，名称和联系电话，SQL

语句如下所示：

```
select t2.salesid,t2.salesid,t2.salesphone
from table1 t1
join table3 t3 on t1.productid=t3.productid
join table2 t2 on t3.salesid=t2.salesid
and t1.productid='p00001';
```

2. 查询各种产品的销售的数量，SQL 语句如下所示：

```
select t3.productid,count(t3.productid)
from table1 t1
join table3 t3 on t1.productid=t3.productid
group by t3.productid;
```

3. 查询编号为 s001 的销售商所不经销的产品的编码和名称，SQL 语句如下所示：

```
select productid,productname  from table1
where productid not in(select t1.productid
from table1 t1
join table3 t3 on t1.productid=t3.productid
join table2 t2 on t3.salesid=t2.salesid
and t2.salesid='s001');
```

1.44.数据库编程

账户表

CustNo	CustName	AccNo	AccBalance	LastModAmt	LastModDate
21890001	Huawuque	100001	1000	+500	20110201
21890001	Huawuque	100004	1500	-100	20110101
21890002	Xiaoyuer	100002	2000	+200	20110301
21890003	Zhangwuji	100003	3000	+1000	20110401
21890004	Zhouxingchi	100005	5000	-300	20101231

注：以客户号和卡号为联合主键。请实现以下功能：

1．现有一个客户新开户，客户号 21890005，客户姓名刘帅，账号 100006，同时该客户往账户中存了 10 元钱，请以此写一条 insert 语句；

2．假设客户 huawuque 今天用银行卡 100004 在 ATM 机上取走了 500 元，写一条 update 语句更新该卡的余额信息

3 .请写一条 select 语句选出客户号一样的账户信息 ,并算出该客户所有卡的积累余额；

参考答案：

在 Oracle 数据库中，创建表和测试数据的 SQL 语句如下所示:

```
create table customer(
  custno number(8),
  custname varchar2(20),
  accno  number(6),
  accbalance  number(10,2),
  lastmodamt  varchar2(20),
```

```
     lastmoddate varchar2(8),
     primary key(custno,accno)
   );
   insert                              into                              customer
values(21890001,'huawuque',100001,1000,'+500','20110201');
   insert                              into                              customer
values(21890001,'huawuque',100004,1500,'-100','20110101');
   insert                              into                              customer
values(21890002,'Xiaoyuer',100002,2000,'+200','20110301');
   insert                              into                              customer
values(21890003,'Zhangwuji',100003,3000,'+1000','20110401');
   insert                              into                              customer
values(21890004,'Zhouxingchi',100005,5000,'-300','20101231');
```

1. 现有一个客户新开户，客户号 21890005，客户姓名刘帅，账号 100006，同时该客户往账户中存了 10 元钱，向数据库插入该客户信息的 SQL 语句如下所示：

```
   insert       into       customer       values(21890005,'       刘       帅
',100006,10,0,to_char(sysdate,'yyyymmdd'));
```

2. 假设客户 huawuque 今天用银行卡 100004 在 ATM 机上取走了 500 元，更新该卡的余额信息的 SQL 语句如下所示：

```
   update                                    customer                    set
accbalance=accbalance-500,lastmodamt='-500',lastmoddate=to char(sysdate,'yyy
ymmdd')
   where custname='huawuque' and accno=100004;
```

3. 选出客户号一样的账户信息，并算出该客户所有卡的积累余额的 SQL 语句如下所示：

```
   select * from customer c join
    (select  custno,sum(accbalance)   from  customer   group  by  custno  having
count(custno)>1) b
    on c.custno=b.custno;
```

2. JDBC

2.1.Java 数据库编程包含哪些类和接口？Java 数据库编程的基本过程是什么？

参考答案：

题目中各问题的答案如下：

1. Java 数据库编程包含 Connection、ResultSet、PreparedStatement、Statement，DriverManager。

2. Java 中访问数据库的步骤如下：

1）注册驱动程序

2）建立连接；

3）创建 Statement；

4）执行 sql 语句；

5）处理结果集（若 sql 语句为查询语句）；

6）关闭连接。

2.2.Statement、PreparedStatement、CallableStatement 的区别？

参考答案：

Statement、PreparedStatement、CallableStatement 的区别有以下几点：

1. Statement 是 PreparedStatement 和 CallableStatement 的父类；

2. Statement 是直接发送 SQL 语句到数据库，事先没有进行预编译。PreparedStatement 会将 SQL 进行预编译，当 SQL 语句要重复执行时，数据库会调用以前预编译好的 SQL 语句，所以 PreparedStatement 在性能方面会更好；

3. PreparedStatement 在执行 SQL 时，对传入的参数可以进行强制的类型转换。以保证数据格式与底层的数据库格式一致；

4. CallableStatement 适用于执行存储过程。

2.3.JAVA 中如何进行事务的处理？

参考答案：

Connection 类中提供了 3 个事务处理方法：

setAutoCommit(Boolean autoCommit):设置是否自动提交事务，默认为自动提交，即为 true，通过设置 false 禁止自动提交事务；

commit():提交事务；

rollback():回滚事务。

2.4.下述程序是一段简单的基于 JDBC 的数据库访问代码，实现了以下功能：从数据库中查询 product 表中的所有记录，然后打印输出到控制台。该代码质量较低，如没有正确处理异常，连接字符串以“魔数”的形式直接存在于代码中等，请用你的思路重新编写程序，完成相同的功能，提高代码质量。

```
public void printPoducts(){
  Connection c=null;
Statements s=null;
ResultSet r=null;
try{
c=DriverManager.getConnection(“jdbc:oracle:thin:@127.0.0.1:1521:sid”,“use
rname”,“password”);
s=c.createStatement();
r=s.executeQuery(“select id,name,price from product”);
System.out.println(“Id\tName\tPrice”);
while(r.next()){
     int x=r.getInt(“id”);
     String y=r.getString(“name”);
     float z=r.getFloat(“price”);
   System.out.println(x+“\t”+y+”\t”+z);
}
}catch(Exception e){
}
}
```

参考答案：

程序改造一：

SQLException 一般情况下无法恢复。通常情况下，数据库访问代码在捕获该异常后，在 catch 块中进行一些必要的处理（例如事务回滚）后，将其直接抛出；或者封装成自定义异常的形式抛出。使用格式如下：

```
try {
       //JDBC 访问
       return ... ;
} catch (SQLException e) {
       //异常处理
       throw new DAOException("访问数据库错误", e);
} finally {
       //关闭连接
}
```

其中 DAOException 是自定义的异常。

程序改造二：

定义常量类 Constant，把连接数据库的信息，定义成常量后再使用。

改造后的程序代码如下所示：

```
class Constant{
public static final String URL="jdbc:oracle:thin:@127.0.0.1:1521:sid";
public static final String USERNAME="username";
public static final String PASSWORD="password";
}
class DAOException extends Exception{
public DAOException(){
    super();
```

```
    }
    public DAOException(String msg){
        super(msg);
    }
}
public class Test {
    public void printPoducts() throws DAOException {
        Connection c = null;
        Statement s = null;
        ResultSet r = null;
        try {
            c = DriverManager.getConnection
(Constant.URL,Constant.USERNAME,Constant.PASSWORD);
            s = c.createStatement();
            r = s.executeQuery("select id,name,price from product");
            System.out.println("Id\tName\tPrice");
            while (r.next()) {
                int x = r.getInt("id");
                String y = r.getString("name");
                float z = r.getFloat("price");
                System.out.println(x + "\t" + y + "\t" + z);
            }
        } catch (SQLException e) {
            throw new DAOException("数据库异常");
        }finally{
            try {
                r.close();
                s.close();
                c.close();
            } catch (SQLException e) {
                e.printStackTrace();
            }
        }
    }
}
```

2.5.请根据以下要求来完成题目：

会议室预定模块。某公司有多个会议室，以房间号区分。如果某部门需要预定会议室，则会提交预定请求（包含预定开始使用时间、预定结束使用，所预定会议室房间号）。

1. 设计一个表，保存会议室预定信息。

2. 要求采用 SQL 语句及 Java 代码段判断 在 2003-3-10 下午 3：00~4:00 3 号会议室是否空闲。

参考答案：

在 Oracle 数据库中创建会议表及插入测试数据的 SQL 语句：

```
CREATE TABLE MEETING(
ID NUMBER PRIMARY KEY ,
ROOM_ID VARCHAR2(10),
ISUSED CHAR,
BEGIN TIMESTAMP,
END TIMESTAMP
);
INSERT INTO MEETING VALUES
(1,'201',1,TO_DATE('2003-03-10 15:00:00','YYYY-MM-DD HH24:MI:SS')
,TO_DATE('2003-03-10 16:00:00','YYYY-MM-DD HH24:MI:SS'));
INSERT INTO MEETING VALUES
(2,'201',1,TO DATE('2003-03-10 17:00:00','YYYY-MM-DD HH24:MI:SS')
,TO_DATE('2003-03-10 22:00:00','YYYY-MM-DD HH24:MI:SS'));
```

使用 Java 代码段判断 在 2003-3-10 下午 3：00~4:00 3 号会议室是否空闲的代码如下所示：

```
package com.tarena;
import java.sql.*;
public class Test {
 public static void main(String[] args) {
     String driverName = "oracle.jdbc.OracleDriver";
     String url = "jdbc:oracle:thin:@127.0.0.1:1521:orcl";
     String username = "scott";
     String pwd = "tiger";
     Connection con = null;
     Statement stmt = null;
     ResultSet rs = null;
     try {
         Class.forName(driverName);
         con = DriverManager.getConnection(url, username, pwd);
         stmt = con.createStatement();
         String sql = "select isUsed from " +
         "meeting " +
         "where  ((begin between to_date('2003-03-10 15:00:00'," +
         "'yyyy-mm-dd hh24:mi:ss') and  to_date('2003-03-10 16:00:00'," +
         "'yyyy-mm-dd hh24:mi:ss'))  " +
         "or(end between to date('2003-03-10 15:00:00'," +
         "'yyyy-mm-dd hh24:mi:ss') and  to_date('2003-03-10 16:00:00'," +
         "'yyyy-mm-dd hh24:mi:ss')))" +
         " and room_id=201";
         if (stmt.execute(sql)) {
             rs = stmt.getResultSet();
         }
         StringBuffer sb = new StringBuffer("isFree:");
         while (rs.next()) {
             sb.append(rs.getInt(1) + "  ");
         }
         System.out.print(sb.toString());
     } catch (Exception e) {
         e.printStackTrace();
     } finally {
         try {
             rs.close();
             stmt.close();
             con.close();
         } catch (Exception e1) {
             e1.printStackTrace();
         }
     }
  }
}
```

2.6.假设有以下数据，请分别用 XML 和 JSON 表述以下数据。请写一个实体类来定义以下数据类型？

表数据(person)

公司	部门	名字	性别	年龄	生日
中国石油	技术部	张三	男	30	1983 - 4 - 1
中国石油	销售部	李四	女	30	1983 - 4 - 2
中国电信	技术部	王五	男	25	1988 - 1 - 2

中国电信	人事部	jack	男	30	1983 - 4 - 3

参考答案：

上述数据使用 JSON 的形式表述如下：

```
{ " persons": [
{ "公司": "中国石油", "部门":"技术部", "名字": "张三", "性别": "男", "年龄": "30", "生日": "1983-4-1" },
{ "公司": "中国石油", "部门":"销售部", "名字": "李四", "性别": "女", "年龄": "30", "生日": "1983-4-2" },
{ "公司": "中国电信", "部门":"技术部", "名字": "王五", "性别": "男", "年龄": "25", "生日": "1988-1-2" },
{ "公司": "中国电信", "部门":"人事部", "名字": "jack", "性别": "男", "年龄": "30", "生日": "1983-4-3" }
]}
```

上述数据使用 XML 表述形式：

```
<?xml version="1.0" encoding="gb2312"?>
<persons>
  <person>
    <公司>中国石油</公司>
    <部门>技术部</部门>
    <名字>张三</名字>
    <性别>男</性别>
    <年龄>30</年龄>
    <生日>1983 - 4 - 1</生日>
  </person>
    <person>
    <公司>中国石油</公司>
    <部门>销售部</部门>
    <名字>李四</名字>
    <性别>女</性别>
    <年龄>30</年龄>
    <生日>1983 - 4 - 2</生日>
  </person>
    <person>
    <公司>中国电信</公司>
    <部门>技术部</部门>
    <名字>王五</名字>
    <性别>男</性别>
    <年龄>25</年龄>
    <生日>1988 - 1 - 2</生日>
  </person>
    <person>
    <公司>中国电信</公司>
    <部门>人事部</部门>
    <名字>jack</名字>
```

```
    <性别>男</性别>
    <年龄>30</年龄>
    <生日>1983-4-3</生日>
  </person>
</persons>
```

使用实体类定义如下：

```
public class Person {
 private String company;
 private String dept;
 private String name;
 private String gender;
 private int age;
 private String birthday;
 public String getCompany() {
     return company;
 }
 public void setCompany(String company) {
     this.company = company;
 }
 public String getDept() {
     return dept;
 }
 public void setDept(String dept) {
     this.dept = dept;
 }
 public String getName() {
     return name;
 }
 public void setName(String name) {
     this.name = name;
 }
 public String getGender() {
     return gender;
 }
 public void setGender(String gender) {
     this.gender = gender;
 }
 public int getAge() {
     return age;
 }
 public void setAge(int age) {
     this.age = age;
 }
 public String getBirthday() {
     return birthday;
 }
 public void setBirthday(String birthday) {
     this.birthday = birthday;
 }
}
```

2.7.使用 Java 语言写一个连接 Oracle 数据库的程序,能够完成修改和查询工作？

参考答案：

使用 JDBC 连接 Oracle 数据库，并完成修改和查询功能的代码如下：

1.创建数据库表 SERVICE_DETAIL，SQL 语句如下所示：

```
CREATE TABLE SERVICE_DETAIL(
```

```
    ID   NUMBER(11)      CONSTRAINT SERVICE DTAIL ID PK PRIMARY KEY,
    SERVICE ID           NUMBER(11),
    HOST                 VARCHAR2(15),
    USER_NAME            VARCHAR2(50),
    PID              NUMBER(11),
    LOGIN TIME           DATE,
    LOGOUT TIME          DATE,
    DURATION             NUMBER(20,9),
    COST                 NUMBER(20,6)
);
```

2.db.properties 文件包含连接数据库的信息。

```
jdbc.driver=oracle.jdbc.OracleDriver
jdbc.url=jdbc:oracle:thin:@127.0.0.1:1521:ORCL
jdbc.user=scott
jdbc.password=tiger
```

3. BaseDAO 类包含打开和关闭连接的方法。

```
import java.sql.Connection;
import java.sql.DriverManager;
import java.sql.SQLException;
import java.util.Properties;

public class BaseDAO {
 private static Properties properties = new Properties();
 private static String driver = properties.getProperty("jdbc.driver");
 private static String url = properties.getProperty("jdbc.url");
 private static String user = properties.getProperty("jdbc.user");
 private static String pwd = properties.getProperty("jdbc.password");

 static {
     try {
         // 加载配置文件

 properties.load(BaseDAO.class.getClassLoader().getResourceAsStream(
                 "com/tarena/dms/daodemo/v2/db.properties"));
         driver = properties.getProperty("jdbc.driver");
         url = properties.getProperty("jdbc.url");
         user = properties.getProperty("jdbc.user");
         pwd = properties.getProperty("jdbc.password");
         Class.forName(driver);
     } catch (Exception e) {
         e.printStackTrace();
         throw new RuntimeException(e);
     }
 }

 protected static Connection openConnection() throws SQLException {
     return DriverManager.getConnection(url, user, pwd);
 }
 protected static void closeConnection(Connection con) {
     if (con != null) {
         try {
             con.close();
         } catch (SQLException e) {
         }
     }
 }
}
```

4. ServiceDetail 类是数据库表和 Java 类的映射。

```
import java.io.Serializable;
import java.math.BigDecimal;
import java.util.Date;

public class ServiceDetail implements Serializable {
 private static final long serialVersionUID = 1L;
 private Integer id;
 private Integer serviceid;
 private String host;
 private String userName;
 private Integer pid;
 private Date loginTime;
 private Date logoutTime;
 private Integer duration;
 private BigDecimal cost;
 public ServiceDetail() {
 }
 public Integer getId() {
     return id;
 }
 public void setId(Integer id) {
     this.id = id;
 }
 public Integer getServiceid() {
     return serviceid;
 }
 public void setServiceid(Integer serviceid) {
     this.serviceid = serviceid;
 }
 public String getHost() {
     return host;
 }
 public void setHost(String host) {
     this.host = host;
 }
 public String getUserName() {
     return userName;
 }
 public void setUserName(String userName) {
     this.userName = userName;
 }
 public Integer getPid() {
     return pid;
 }
 public void setPid(Integer pid) {
     this.pid = pid;
 }
 public Date getLoginTime() {
     return loginTime;
 }
 public void setLoginTime(Date loginTime) {
     this.loginTime = loginTime;
 }
 public Date getLogoutTime() {
     return logoutTime;
 }
 public void setLogoutTime(Date logoutTime) {
     this.logoutTime = logoutTime;
 }
 public Integer getDuration() {
     return duration;
 }
 public void setDuration(Integer duration) {
     this.duration = duration;
```

```
    }
    public BigDecimal getCost() {
        return cost;
    }
    public void setCost(BigDecimal cost) {
        this.cost = cost;
    }
    @Override
    public String toString() {
        return "ServiceDetail [id=" + id + ", serviceid=" + serviceid
                + ", host=" + host + ", userName=" + userName + ", pid=" + pid
                + ", loginTime=" + loginTime + ", logoutTime=" + logoutTime
                + ", duration=" + duration + ", cost=" + cost + "]";
    }
    }
```

5.实现对 ServiceDetail 表的更新和查询。

```
    import java.sql.Connection;
    import java.sql.Date;
    import java.sql.PreparedStatement;
    import java.sql.ResultSet;
    import java.sql.SQLException;
    import java.sql.Timestamp;
    import java.util.ArrayList;
    import java.util.List;

    import com.tarena.dms.daodemo.v3.entity.ServiceDetail;

    public class ServiceDetailDAO extends BaseDAO {
    private static final String UPDATE = "update SERVICE DETAIL set "
            + "SERVICE_ID=?, HOST=?, USER_NAME=?, PID=?, LOGIN_TIME=?, "
            + "LOGOUT_TIME=?, DURATION=?, COST=? where ID=?";
    private static final String FIND_ALL = "select ID, SERVICE_ID, HOST, USER_NAME,
PID, "
            + "LOGIN TIME, LOGOUT TIME, DURATION, COST from SERVICE DETAIL";


    public List<ServiceDetail> findAll() {
        Connection con = null;
        PreparedStatement stmt = null;
        try {
            con = openConnection();
            stmt = con.prepareStatement(FIND_ALL);
            ResultSet rs = stmt.executeQuery();
            List<ServiceDetail> list = new ArrayList<ServiceDetail>();
            while (rs.next()) {
                list.add(toServiceDetail(rs));
            }
            rs.close();
            stmt.close();
            return list;
        } catch (SQLException e) {
            System.out.println("数据库访问异常!");
            throw new RuntimeException(e);
        } finally {
            closeConnection(con);
        }
    }
    public void update(ServiceDetail serviceDetail) {
        Connection con = null;
        PreparedStatement stmt = null;
        try {
            con = openConnection();
            stmt = con.prepareStatement(UPDATE);
```

```
            stmt.setInt(1, serviceDetail.getServiceid());
            stmt.setString(2, serviceDetail.getHost());
            stmt.setString(3, serviceDetail.getUserName());
            stmt.setInt(4, serviceDetail.getPid());
            stmt.setDate(5, new Date(serviceDetail.getLoginTime().getTime()));
            stmt.setDate(6,                                                    new
Date(serviceDetail.getLogoutTime().getTime()));
            stmt.setTimestamp(5,new
Timestamp(serviceDetail.getLoginTime().getTime()));
            stmt.setTimestamp(6,                                               new
Timestamp(serviceDetail.getLogoutTime().getTime()));
            stmt.setInt(7, serviceDetail.getDuration());
            stmt.setBigDecimal(8, serviceDetail.getCost());
            stmt.setInt(9, serviceDetail.getId());

            stmt.executeUpdate();
            stmt.close();
        } catch (SQLException e) {
            System.out.println("数据库访问异常!");
            throw new RuntimeException(e);
        }
    }

    private ServiceDetail toServiceDetail(ResultSet rs) throws SQLException {
        ServiceDetail serviceDetail = new ServiceDetail();
        serviceDetail.setId(rs.getInt("ID"));
        serviceDetail.setServiceid(rs.getInt("SERVICE_ID"));
        serviceDetail.setHost(rs.getString("HOST"));
        serviceDetail.setUserName(rs.getString("USER NAME"));
        serviceDetail.setPid(rs.getInt("PID"));
        serviceDetail.setLoginTime(rs.getTimestamp("LOGIN TIME"));
        serviceDetail.setLogoutTime(rs.getTimestamp("LOGOUT TIME"));
        serviceDetail.setDuration(rs.getInt("DURATION"));
        serviceDetail.setCost(rs.getBigDecimal("COST"));
        return serviceDetail;
    }
    }
```

2.8.以下数据库操作的程序片段如何改进会更好？

```
        try {
            Class.forName(“com.mysql.jdbc.Driver”);
                          Connection                conn                =
DriverManager.getConnection(“jdbc:mysql://localhost/test”, “root”, “123456”);
            Statement stmt = conn.createStatement();
            String sql = “select * from T User where username=’” + name
                + “’ and password=’” + password + “’”;
             ResultSet rs = stmt.executeQuery(sql);
            if (rs.next()) {
                System.out.println(“User Name and Password is correct!”);
            } else {
                 System.out.println(“User  Name  and  Password  pair  is
invalidate”);
            }
       } catch (Exception e) {
                e.printStackTrace();
        }
```

参考答案：

使用 PreparedStatement 防止 SQL 注入，改进的代码如下所示：

```
        try {
            Class.forName("com.mysql.jdbc.Driver");
            Connection                          conn                          =
DriverManager.getConnection("jdbc:mysql://localhost/test", "root", "123456");


            String  sql="select  count(*)  from  t user  where  username=?  and
password=?";
            PreparedStatement ps=conn.prepareStatement(sql);
            ps.setString(1, username);
            ps.setString(2, password);
            ResultSet rs = ps.executeQuery();
            rs.next();
           int count =rs.getInt(1);
            if (count==1) {
                System.out.println("User Name and Password is correct!");
            } else {
                System.out.println("User Name and Password pair is invalidate");
            }
       } catch (Exception e) {
           e.printStackTrace();
       }
```

2.9.写出一段 JDBC 连接本机 MySQL 数据库的代码？

参考答案：

```
Class.forName("com.mysql.jdbc.Driver");
String url="jdbc:mysql://localhost/test;
String user='root';
String password=root;
Connection conn= DriverManager.getConnection(url,user,password);
```

2.10.利用 JAVA　IO 流编写一个 SQL 语句执行工具，功能类似于（oracle sqlplus）

例如：输入 select * from t_user 按回车，可以显示如下结果：

id　　name　　age　　gender

1　　张三　　20　　男

2　　李四　　21　　女

输入 insert into t_user(id,name,age,gender) values(1,’王五’,20,’男’)按回车，控制台显示结果：已增加一条记录。

参考答案：

```
public class Q010 {
 public static void main(String args[]) throws Exception {
     Scanner in = new Scanner(System.in);
     String sql = in.nextLine();
     sql = sql.trim();
     System.out.println(sql);
     Connection con = Q010.getConnection();
     PreparedStatement pst = con.prepareStatement(sql);
     if (sql.startsWith("select")) {
         ResultSet rs = pst.executeQuery();
         System.out.println("Id  Name  age  gender");
         System.out.println("------------------------------");
         while (rs.next()) {
```

```
                System.out.println(rs.getString("id")    +    "          "    +
rs.getString("name") + "   "
                        + rs.getString("age") + "   " + rs.getString("gender"));
            }
        } else if (sql.startsWith("insert")) {
            pst.executeUpdate();
            System.out.println("增加一条数据");
        }
    }

    // 下面的方法是使用 jdbc 连接 oracle 数据库
    public static Connection getConnection() throws Exception {
        Class.forName("oracle.jdbc.OracleDriver");
        Connection con = DriverManager.getConnection(
                "jdbc:oracle:thin:@127.0.0.1:1521:ORCL", "scott", "tiger");
        return con;
    }
   }
```

3. XML

3.1. XML 文档定义有几种形式？有何本质区别？

参考答案：

两种形式 DTD 和 Schema，二者区别如下：

1．Schema 是标准的 XML 文件,而 DTD 则使用自己的特殊语法,因此,只需要知道 XML 的语法规则就可以编写 Schema 了，不需要再学习其它语法规则。

2. Schema 利用命名空间将文件中特殊的节点与 Schema 说明相联系，一个 XML 文件可以有多个对应的 Schema；而一个 XML 文件只能有一个相对应的 DTD 文件。

3. Schema 的内容模型是开放的，可以随意扩充,而 DTD 则无法解读扩充的内容。DTD 只能把文件类型定义为一个字符串，而 XML Schema 却允许把文件类型定义为整数,浮点数,字符串,布尔值或其他各数据类型,而无须重新定义。

3.2. Java 中常用的 XML 解析技术有哪些？区别是什么？

参考答案：

Java 中常用的 XML 解析技术有 DOM、SAX 两种方式，这两种方式的区别如下：

DOM 解析处理大型文件时其性能下降的非常厉害。这个问题是由 DOM 的树结构所造成的，这种结构占用的内存较多，而且 DOM 必须在解析文件之前把整个文档载入内存，适合对 XML 的随机访问。

SAX 解析不同于 DOM ,SAX 是事件驱动型的 XML 解析方法。它顺序读取 XML 文件，不需要一次全部装载整个文件。当遇到像文档开头 ,文档结束 ,或者标签开头与标签结束时，它会触发一个事件，用户通过在其回调事件中写入处理代码来处理 XML 文件，适合对 XML 的顺序访问。

第三部分

1. Web 基础

1.1.请描述一个网页从开始请求到最终显示的完整过程？

参考答案：

一个网页从请求到最终显示的完整过程一般可分为如下 7 个步骤：

1. 在浏览器中输入网址；

2. 发送至 DNS 服务器并获得域名对应的 WEB 服务器的 IP 地址；

3. 与 WEB 服务器建立 TCP 连接；

4. 浏览器向 WEB 服务器的 IP 地址发送相应的 HTTP 请求；

5. WEB 服务器响应请求并返回指定 URL 的数据，或错误信息，如果设定重定向，则重定向到新的 URL 地址。

6. 浏览器下载数据后解析 HTML 源文件，解析的过程中实现对页面的排版，解析完成后在浏览器中显示基础页面。

7. 分析页面中的超链接并显示在当前页面，重复以上过程直至无超链接需要发送，完成全部显示。

1.2.是否对 HTML5 和 CSS3 有关注？如果有请简述？

参考答案：

HTML5 是用于取代 1999 年所制定的 HTML 4.01 和 XHTML 1.0 标准的 HTML 标准版本，现在仍处于发展阶段，但大部分浏览器已经支持某些 HTML5 技术。HTML 5 有两大特点：首先，强化了 Web 网页的表现性能。其次，追加了本地数据库等 Web 应用的功能。广义论及 HTML5 时，实际指的是包括 HTML、CSS 和 JavaScript 在内的一套技术的组合。HTML5 有如下几种特性：

1. 语义特性（Semantic）:HTML5 赋予网页更好的意义和结构。更加丰富的标签将随着对 RDFa、微数据、微格式等方面的支持，构建对程序、对用户都更有价值的数据驱动的 Web。

2. 本地存储特性（OFFLINE & STORAGE）:基于 HTML5 开发的网页 APP 拥有更短的启动时间，更快的联网速度，这些全得益于 HTML5 APP Cache，以及本地存储功能。Indexed DB（html5 本地存储最重要的技术之一）和 API 说明文档。

3. 设备兼容特性（DEVICE ACCESS):从地理位置的 API 文档公开以来，HTML5 为网页应用开发者们提供了更多功能上的优化选择，带来了更多体验功能的优势。HTML5 提供了

前所未有的数据与应用接入开放接口。使外部应用可以直接与浏览器内部的数据直接相连，例如视频影音可直接与麦克风及摄像头相联。

4. 连接特性（CONNECTIVITY）:更有效的连接工作效率，使得基于页面的实时聊天，更快速的网页游戏体验，更优化的在线交流得到了实现。HTML5 拥有更有效的服务器推送技术，Server-Sent Event 和 WebSockets 就是其中的两个特性，这两个特性能够帮助我们实现服务器将数据“推送”到客户端的功能。

5. 网页多媒体特性(MULTIMEDIA):支持网页端的 Audio、Video 等多媒体功能， 与网站自带的 APPS，摄像头，影音功能相得益彰。

6. 三维、图形及特效特性（3D, Graphics & Effects）:基于 SVG、Canvas、WebGL 及 CSS3 的 3D 功能，用户会惊叹于在浏览器中，所呈现的惊人视觉效果。

性能与集成特性（Class: Performance & Integration）:没有用户会永远等待你的 Loading——HTML5 会通过 XMLHttpRequest2 等技术，帮助您的 Web 应用和网站在多样化的环境中更快速的工作。

CSS3 是 CSS 技术的升级版本，CSS3 语言开发是朝着模块化发展的。以前的规范作为一个模块实在是太庞大而且比较复杂，所以，把它分解为一些小的模块，更多新的模块也被加入进来。这些模块包括： 盒子模型、列表模块、超链接方式 、语言模块 、背景和边框 、文字特效 、多栏布局等。CSS3 有如下几个特性：

1． CSS3 圆角表格：圆角表格，对应属性：border-radius。

2． 以往对网页上的文字加特效只能用 filter 这个属性，这次 CSS3 中专门制订了一个加文字特效的属性，而且不止加阴影这种效果。对应属性：font-effect。

3． 丰富了对链接下划线的样式，以往的下划线都是直线，这次可不一样了，有波浪线、点线、虚线等等，更可对下划线的颜色和位置进行任意改变。（还有对应顶线和中横线的样式，效果与下划线类似）对应属性：text-underline-style，text-underline-color，text-underline-mode，text-underline-position。

4． 在文字下点几个点或打个圈以示重点，CSS3 也开始加入了这项功能，这应该在某些特定网页上很有用。对应属性：font-emphasize-style 和 font-emphasize-position。

1.3.简要描述标准 HTML 文档的结构？

参考答案：

HTML 文档的开始需要版本声明,剩下的页面内容需要包含在开始标记 <html> 和结束标记 </html> 之间。在 <html> 元素中,包含两个主要的部分,文件头部分（<head> 元素）和主体部分（<body>元素）。标准 HTML 文档的结构如下所示：

```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN"
http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">
<html xmlns="http://www.w3.org/1999/xhtml">
    <head>
    </head>
    <body>
    </body>
</html>
```

1.4.简要描述行内元素和块级元素的区别？

参考答案：

块级元素的前后都会自动换行，如同存在换行符一样。默认情况下，块级元素会独占一行。例如，<p>、<hn>、 都是块级元素。在显示这些元素中间的文本时，都将从新行中开始显示，其后的内容也将在新行中显示。

行内元素可以和其他行内元素位于同一行，在浏览器中显示时不会换行。例如，<a>、 等。

我们可以这样理解：如果元素是块级的，则总是在新行上显示，好比是书中的一个新段落；而元素如果是行内的，那么只能在当前行中显示，就像是段落中的一个单词。

因此，块级元素常用来构建网页上比较大的结构，用于包含其他块级元素、行内元素和文本；而行内元素一般只能包含其他行内元素和文本。

1.5.锚点的作用是什么？如何创建锚点？

参考答案：

锚点是文档中某行的一个记号，类似于书签，用于链接到文档中的某个位置。当定义了锚点后，我们可以创建直接跳至该锚点（比如页面中某个小节）的链接，这样使用者就无需不停地滚动页面来寻找他们需要的信息了。

在使用 <a> 元素创建锚点时，需要使用 name 属性为其命名，代码如下所示：

```
<a name="anchorname1">锚点一</a>
```

然后就可以创建链接，直接跳转到锚点，代码如下所示：

```
<a href="#anchorname1">回到锚点一</a>
```

1.6.简要描述 CSS 中的定位机制？

参考答案：

CSS 中，除了默认的流定位方式以外，还有如下几种定位机制：浮动定位、相对定位、绝对定位和固定定位。

浮动定位是指将元素排除在普通流之外，并且将它放置在包含框的左边或者右边，但是依旧位于包含框之内。

相对定位将元素相对于它在普通流中的位置进行定位。

绝对定位是指将元素的内容从普通流中完全移除，并且可以使用偏移属性来固定该元素的位置。

固定定位是指将元素的内容固定在页面的某个位置。

1.7.display 属性的作用是什么？

参考答案：

可以使用 display 属性定义建立布局时元素生成的显示框类型。

1. 如果将 display 属性设置为 block，可以让行内元素（比如 <a> 元素）表现得像块级元素一样；

2. 如果将 display 属性设置为 inline，可以让块级元素（比如 <p> 元素）表现得像内联元素一样；

3. 可以通过把 display 属性设置为 none，让生成的元素根本没有框。这样的话，该框及其所有内容就不再显示，不占用文档中的空间。

1.8.在 HTML 页面中定义以下表格：

公司名称	上海屹通
地址	软件园 4 号楼 205 室

参考答案：

```
<body>
    <table border="1" cellpadding="10" cellspacing="0">
        <tr valign="middle" align="left">
            <td width="150px">
                公司名称
            </td>
            <td width="250px">
                上海屹通
            </td>
        </tr>
        <tr valign="middle" align="left">
            <td width="150px">
                地址
            </td>
            <td width="250px">
                软件园 4 号楼 205 室
            </td>
        </tr>
    </table>
</body>
```

1.9.绘制包含用户名、密码输入框、登录、重置按钮的登录框的 HTML，并使单击登录按钮后将相应数据提交到 login.action？

参考答案：

```
<form action="login.action">
    用户名：<input type="text" name="user"><br/>
    密　码：<input type="password" name="pwd"><br/>
    <input type="submit" value="登录"/>
    <input type="reset" value="重置"/>
</form>
```

1.10.PNG 和 GIF 各有什么特点？适用于那些场合？

参考答案：

PNG 特点：PNG 用来存储灰度图像时，灰度图像的深度可多到 16 位，存储彩色图像时 ,彩色图像的深度可多到 48 位 ,并且还可存储多到 16 位的α通道数据。PNG 使用从 LZ77 派生的无损数据压缩算法。一般应用于 JAVA 程序、网页、S60 程序中，因其压缩比高，生成文件容量小。

GIF 特点: GIF 文件的数据，是一种基于 LZW 算法的连续色调的无损压缩格式。其压缩率一般在 50%左右，它不属于任何应用程序。目前几乎所有相关软件都支持它，公共领域有大量的软件在使用 GIF 图像文件。GIF 图像文件的数据是经过压缩的，而且是采用了可变长度等压缩算法。GIF 格式的另一个特点是其在一个 GIF 文件中可以存多幅彩色图像，如果把存于一个文件中的多幅图像数据逐幅读出并显示到屏幕上，就可构成一种最简单的动画。GIF 分为静态 GIF 和动画 GIF 两种，扩展名为.gif，是一种压缩位图格式， 支持透明背景图像，适用于多种操作系统，“体型” 很小，网上很多小动画都是 GIF 格式。

1.11.谈谈 innerHTML　outerHTML　innerText 之间的区别？

参考答案：

innerHTML　outerHTML　innerText 之间的区别如下：

1. innerHTML 设置或获取位于对象起始和结束标签内的 HTML；

2. outerHTML 设置或获取对象及其内容的 HTML 形式 ；

3. innerText 设置或获取位于对象起始和结束标签内的文本 ；

4. innerHTML 与 outerHTML 在设置对象的内容时包含的 HTML 会被解析，而 innerText 与 outerText 则不会。

5. 在设置时，innerHTML 与 innerText 仅设置标签内的文本，而 outerHTML 与 outerText 设置包括标签在内的文本。

6. innerHTML 是符合 W3C 标准的属性，而 innerText 只适用于 IE 浏览器，因此，尽可能地去使用 innerHTML，而少用 innerText，如果要输出不含 HTML 标签的内容，可以使用 innerHTML 取得包含 HTML 标签的内容后，再用正则表达式去除 HTML 标签。

1.12.简述 CSS 样式表的使用方式？

参考答案：

HTML 页面有三种使用 CSS 样式表的方式：

1. 内联样式表：样式规定在单个的元素中，写在元素的 style 属性里；

2. 内部样式表：样式定义在 HTML 页面的头元素中；

3. 外部样式表：将样式定义在一个外部的 CSS 文件中，然后由 HTML 页面引用样式表文件。

1.13.如何理解 CSS 样式表的层叠性？

参考答案：

CSS 使用层叠（Cascade）的原则来考虑继承、层叠次序和优先级等重要特征，从而判断相互冲突的规则中哪个规则应该起作用。

继承性是指，许多 CSS 的样式规则不但影响选择器所定义的元素，而且会被这些元素的后代继承。

层叠性是指，当一个 Web 页面使用多个样式表，多个样式表中的样式可层叠为一个。在多个样式表之间所定义的样式没有冲突的时候，浏览器会显示所有的样式。

优先级是指，当发生样式定义冲突时，浏览器首先会按照不同样式规则的优先级来应用样式。CSS 样式的优先级如下所示（其中数字 3 拥有最高的优先权）：

1. 浏览器缺省设置；
2. 外部样式表（.css 文件）或者内部样式表（位于 <head> 元素内部）；
3. 内联样式（作为某个元素的 style 属性的值）。

同等优先级下，以最后定义的样式为准。

1.14.CSS 选择器中，元素选择器和类选择器的区别是什么？

参考答案：

元素选择器是最常见的 CSS 选择器，即，文档的元素就是最基本的选择器。选择器通常是某个 HTML 元素，比如 <p>、<h1>、、<a>等，甚至可以是 <html> 元素本身。

类选择器用于将样式规则与附带 class 属性的元素匹配，其中该 class 属性的值为类选择器中指定的值。使用类选择器时，首先需要定义样式类，其语法为：

```
.className { }
```

所有能够附带 class 属性的元素都可以使用此样式声明。只需要将 class 属性的值设置为“className”，则可以将类选择器的样式与元素关联。

在实际使用时，如果需要为某种元素定义样式，则往往使用元素选择器；如果要应用样式而不考虑具体设计的元素，最常用的方法就是使用类选择器。

1.15.DIV 设计中如何区别 display:none 和 visibility:hidden？

参考答案：

在 DIV 设计中，使用 display:none 属性后，HTML 元素（对象）的宽度、高度等各种属性值都将“丢失”；而使用 visibility:hidden 属性后，HTML 元素（对象）仅仅是在视觉上看不见（完全透明），而它所占据的空间位置仍然存在，也即是说它仍具有高度、宽度等属性值。

1.16.下面是一个 CSS 样式文件的片段，写出每种定义方式的含义：

```
td {width:100%;}
.td {width:100%}
#td {width:100%}
td input {font-size: 20pt}
```

参考答案：

上述每种定义方式的含义如下：

1. td {width:100%;}：用元素选择器设置 td 元素（表格当中）的宽度为 100%；

2. .td {width:100%}：用类选择器设置 class 属性值为 td 的元素的宽度为 100%；

3. #td {width:100%}：用 id 选择器设置 id 值为 td 的元素的宽度为 100%；

4. td input {font-size: 20pt}：派生（后代）选择器设置 td 元素中的 input 元素的字体大小为 20pt。

1.17.如何居中一个浮动元素？

参考答案：

请看如下代码，通过 css 设置 div 的 left 和 right 属性，来使浮动元素 div 居中：

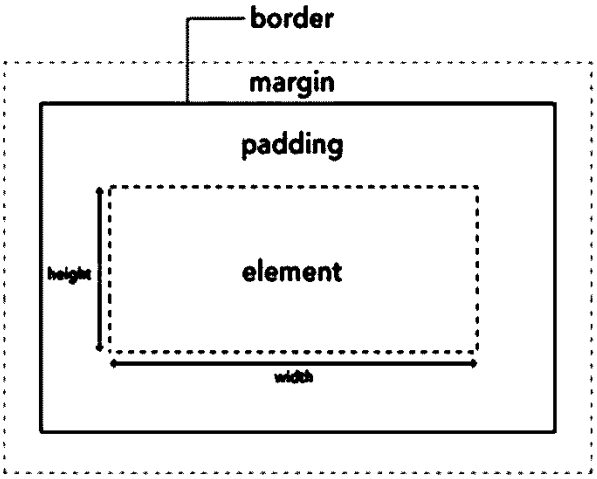
```
<html>
  <head>
    <title>如何居中一个浮动元素</title>
    <meta http-equiv="content-type" content="text/html; charset=UTF-8">
    <style type="text/css">
 div
 {
 width:10px;
 height:10px;
 background-color:green;
 border:1px solid red;
 position:relative;
 left:50%;
 right:50%
 }
    </style>
  </head>
  <body>
    <div style="float:left;"></div>
  </body>
</html>
```

1.18.简述对 CSS 的盒子模型理解？

参考答案：

CSS 盒子模式具备内容(content)、填充(padding)、边框(border)、边界(margin)这些属性。这些属性我们可以把它转移到我们日常生活中的盒子（箱子）上来理解，日常生活中所见的盒子也就是能装东西的一种箱子，也具有这些属性，所以叫它盒子模式。那么内容（CONTENT）就是盒子里装的东西；而填充(PADDING)就是怕盒子里装的东西（贵重的）

损坏而添加的泡沫或者其它抗震的辅料；边框(BORDER)就是盒子本身了；至于边界(MARGIN)则说明盒子摆放的时候的不能全部堆在一起，要留一定空隙保持通风，同时也为了方便取出。在网页设计上，内容常指文字、图片等元素，但是也可以是小盒子（DIV 嵌套），与现实生活中盒子不同的是，现实生活中的东西一般不能大于盒子，否则盒子会被撑坏的，而 CSS 盒子具有弹性，里面的东西大过盒子本身最多把它撑大，但它不会损坏的。填充只有宽度属性，可以理解为生活中盒子里的抗震辅料厚度，而边框有大小和颜色之分，我们又可以理解为生活中所见盒子的厚度以及这个盒子是用什么颜色材料做成的，边界就是该盒子与其它东西要保留多大距离。每个 HTML 标记都可看作一个盒子；每个盒子都有：边界、边框、填充、内容四个属性；每个属性都包括四个部分：上、右、下、左；这四部分可同时设置，也可分别设置；盒子模型的结构如下图所示：



1.19.CSS 选择符有哪些？哪些属性可以继承？优先级算法如何计算？内联和 important 哪个优先级高？

参考答案：

上述各问题的参考答案如下：

1.CSS 有如下几种常用选择符：

1）元素选择器

2）类选择器

3）分类选择器

4）id 选择器

5）选择器分组

6）派生（后代）选择器

7）伪类选择器

2.CSS 中可以继承的属性如下：

1)文本相关属性 :font-family、font-size、font-style、font-variant, font-weight、font、 letter-spacing、line-height、text-align、 text-indent、text-transform、word-spacing；

2)列表相关属性 :list-style-image、list-style-position、list-style-type、list-style；

3）color 属性。

3.CSS 优先级算法如下:

优先级为就近原则，样式定义最近者为准。相同的样式，如果重复定义，以最后一次的定义为准。

4.important 比 内联优先级高

1.20.CSS 中 margin 和 padding 的区别？ link 和@import 的区别是？

参考答案：

1.margin 和 padding 的区别如下：

padding：边框的里面的一层边内补白

margin：边框的外面的一层边外补白

2.将样式定义在单独的.css 的文件里，link 和@import 都可以在 html 页面引入 css 文件。有 link 和@import 两种方式，导入方式如下

link 方式：

```
<link rel="stylesheet" type="text/css" href="aa.css">
```

@import 方式:

```
<style type="text/css">
    @import "aa.css";
</style>
```

link 和@import 两种导入 css 文件的区别：

1） 祖先的差别。link 属于 XHTML 标签，而@import 完全是 CSS 提供的一种方式。link 标签除了可以加载 CSS 外，还可以做很多其它的事情，比如定义 RSS，定义 rel 连接属性等，@import 就只能加载 CSS 了；

2） 加载顺序的差别。当一个页面被加载的时候（就是被浏览者浏览的时候），link 引用的 CSS 会同时被加载，而@import 引用的 CSS 会等到页面全部被下载完再被加载。所以有时候浏览@import 加载 CSS 的页面时开始会没有样式（就是闪烁），网速慢时更为明显；

3）兼容性的差别。由于@import 是 CSS2.1 提出的所以老的浏览器不支持，@import 只有在 IE5 以上的才能识别，而 link 标签无此问题；

4）使用 DOM 控制样式时的差别。当使用 JavaScript 控制 DOM 去改变样式的时候，只能使用 link 标签，因为@import 不是 DOM 可以控制的；

5) @import 可以在 css 中再次引入其它样式表，比如可以创建一个主样式表，在主样式表中再引入其他的样式表。

1.21.请列出至少三种 JS 框架并简要谈谈你的理解？

参考答案：

Prototype 是一个非常优雅的 JS 库，定义了 JS 的面向对象扩展，DOM 操作 API，事件等等，以 prototype 为核心，形成了一个外围的各种各样的 JS 扩展库，是相当有前途的 JS 底层框架，值得推荐。

jQuery 是一款同 prototype 一样优秀 js 开发库类，特别是对 css 和 XPath 的支持，使我们写 js 变得更加方便，并且语法简洁和效率高一直是 jQuery 追求的目标。它注重简洁和高效，js 效果有 yui-ext 的选择，因为 yui-ext 重用了很多 jQuery 的函数。

moo.fx 是一个超级轻量级的 javascript 特效库(7k)能够与 prototype.js 或 mootools 框架一起使用。它非常快、易于使用、跨浏览器、符合标准，提供控制和修改任何 HTML 元素的 CSS 属性，包括颜色。它内置检查器能够防止用户通过多次或疯狂点击来破坏效果。moo.fx 整体采用模块化设计，所以可以在它的基础上开发你需要的任何特效。

1.22.如何动态引入 JS 文件？

参考答案：

将 JavaScript 代码写入一个单独的文件，并保存为后缀为 js 的文件，html 页面的 <head> 中引用外部的 .js 文件，假设 my.js 文件中的代码如下：

```
function method2()
{
   alert("hello word!");
}
```

在 <head> 中添加 <script> 标签，设置 <script> 标签的 “src” 属性，以指定 js 文件的 url，请看如下引入 js 文件的代码：

```
html>
    <head>
        <script language="JavaScript" src="my.js"
            type="text/javascript">
        </script>
    </head>
    <body>
        <form>
              <input type= "button"  value="第二个按钮"
                                        onclick="method2();">
        </form>
    </body>
</html>
```

1.23.Java 和 JavaScript 的区别？

参考答案：

JavaScript 与 Java 是两个公司开发的不同的两个产品。Java 是 SUN 公司推出的新

一代面向对象的程序设计语言，特别适合于 Internet 应用程序开发；而 JavaScript 是 Netscape 公司的产品，其目的是为了扩展 Netscape Navigator 功能，而开发的一种可以嵌入 Web 页面中的基于对象和事件驱动的解释性语言，它的前身是 Live Script；而 Java 的前身是 Oak 语言。下面对两种语言间的异同作如下比较：

1 基于对象和面向对象：Java 是一种真正的面向对象的语言，即使是开发简单的程序，必须设计对象；JavaScript 是种脚本语言，它可以用来制作与网络无关的，与用户交互作用的复杂软件。它是一种基于对象（Object Based）和事件驱动（Event Driver）的编程语言。因而它本身提供了非常丰富的内部对象供设计人员使用；

2．解释和编译：Java 的源代码在执行之前，必须经过编译；JavaScript 是一种解释性编程语言，其源代码不需经过编译，由浏览器解释执行；

3．强类型变量和弱类型变量：Java 采用强类型变量检查，即所有变量在编译之前必须作声明；JavaScript 中变量声明，采用弱类型。即变量在使用前不需作声明，而是解释器在运行时检查其数据类型；

1.24.JavaScript 的优缺点和内置对象？

参考答案：

1．优点：简单易用，与 Java 有类似的语法，可以使用任何文本编辑工具编写，只需要浏览器就可执行程序，并且事先不用编译，逐行执行，无需进行严格的变量声明，而且内置大量现成对象，编写少量程序可以完成目标；

2．缺点：不适合开发大型应用程序；

3 .Javascript 有 11 种内置对象： Array、String、Date、Math、Boolean、Number、Function、Global、Error、RegExp、Object。

1.25.简要描述 JavaScript 的数据类型？

参考答案：

JavaScript 的数据类型可以分为三类：基本类型、特殊类型和复杂类型。

基本类型有 string、number 和 boolean 三种。其中，字符串是使用一对单引号或者一对双引号括起来的任意文本；而数值类型都采用 64 位浮点格式存储，不区分整数和小数；布尔（逻辑）只能有两个值：true 或 false。

特殊类型有 null、undefined 两种。其中，Undefined 这个值表示变量不含有值，即声明了变量但从未赋值；null 在程序中代表“无值”或者“无对象”，因此，可以通过将变量的值设置为 null 来清空变量。

复杂类型指其他对象，如 Array、Date、Object 等。

1.26.简述 arguments 对象的作用？

参考答案：

在函数代码中，使用特殊对象 arguments 可以访问函数的参数。即，开发者在定义函数时，无需明确的为方法声明参数，也可以在方法体中使用 arguments 来访问参数。这是因为， arguments 是一种特殊对象，在函数代码中，表示函数的参数数组。

正因为 arguments 表示参数组成的数组，因此，首先可以使用 arguments.length 检测函数的参数个数，其次，可以通过下标（arguments[index]）来访问某个参数。这样，可以用 arguments 对象判断传递给函数的参数个数并获取参数，从而模拟函数重载。

1.27.列举几个 JavaScript 中常用的全局函数，并描述其作用？

参考答案：

JavaScript 中常用的全局函数，及其作用如下：

1. parseInt：解析一个字符串并返回一个整数；
2. parseFloat：解析一个字符串并返回一个浮点数；
3. isNaN：检查某个值是否是数字，返回 true 或者 false；
4. encodeURI ：把字符串作为 URI 进行编码；
5. decodeURI ：对 encodeURI() 函数编码过的 URI 进行解码；
6. eval：计算某个字符串，以得到结果，或者用于执行其中的 JavaScript 代码。

1.28.简述 window 对象除 document 以外的一些常用子对象，并描述其作用？

参考答案：

window 对象有很多子对象，除了 document 以外，还有如下常用子对象：

- screen 对象：此对象包含有关客户端显示屏幕的信息，常用于获取屏幕的分辨率和色彩；
- history 对象：此对象包含用户（在浏览器窗口中）访问过的 URL；
- location 对象：此对象包含有关当前 URL 的信息，常用于获取和改变当前浏览的网址；
- navigator 对象：此对象包含有关浏览器的信息，常用于获取客户端浏览器和操作系统信息；
- event 对象：任何事件触发后将会产生一个 event 对象，该对象记录事件发生时的鼠标位置、键盘按键状态和触发对象等信息。

1.29.简述三种创建对象的方式？

参考答案：

1. 创建对象的实例：使用 Object 对象，并封装属性和方法；

2. 创建对象的模板：定义构造函数，以创建自定义对象并封装属性和方法；

3. JSON：使用 JSON 的语法方式创建。

1.30.JavaScript 试题：

1. 页面有一个下拉菜单和一个输入框，写一方法使下拉菜单值变动时输入框显示对应的值；

2. 写一方法检查多选框有几个被勾选。

参考答案：

1．使下拉菜单值变动时输入框显示对应的值，代码如下所示：

```
<html>
  <head>
    <title>页面有一个下拉菜单和一个输入框，写一方法使下拉菜单值变动时输入框显示对应的值
</title>
    <meta http-equiv="content-type" content="text/html; charset=UTF-8">
    <script type="text/javascript">
    function f(){
     var selectItems = document.getElementsByName("bank No");
     var a=selectItems[0].value;
     var b=document.getElementById("bbb");
     b.value=a;
    }
    </script>
  </head>
  <body>
    <select id="aaa" size="5" name="bank No" onchange="f();">
     <option value="1" class="ss" >aaa</option>
     <option value="2" class="ss">bbb</option>
     <option value="3" class="ss">ccc</option>
     <option value="4" class="ss">ddd</option>
     <option value="5" class="ss">eee</option>
     <option value="6" class="ss">fff</option>
     <option value="7" class="ss">ggg</option>
     <option value="8" class="ss">hhh</option>
     <option value="9" class="ss">iii</option>
 </select>
 <input type="text" id="bbb"/>
  </body>
</html>
```

2. 检查多选框有几个被勾选，代码如下所示：

```
<html>
  <head>
  <title>写一方法校验多选框有多少被勾选</title>
    <meta http-equiv="content-type" content="text/html; charset=UTF-8">
    <script type="text/javascript" >
     function aa(){
         var checkall = document.getElementsByName("la");
         var n=0;
         for(var i=0;i<checkall.length;i++){
             if(checkall[i].checked==true){
                 n++;
             }
         }
         alert("选中了:"+n+"个");
```

```
        }
      </script>
    </head>

    <body>

     <input type="checkbox" name="la"/>aa
     <input type="checkbox" name="la"/>bb
     <input type="checkbox" name="la"/>cc
     <input type="checkbox" name="la"/>dd
     <input type="button" value="选中" onclick="aa();">
    </body>
  </html>
```

1.31.JavaScript 中的提示框都有哪些，请列举出来？

参考答案：

JavaScript 中的提示框有：警告框、确认框、提示框，代码表示如下：

```
alert("我是警告框");
confirm("确认框");
prompt("提示框","你的密码与你的生日有关");
```

1.32.如何利用 JavaScript 在 html 中添加一个 div？

参考答案：

利用 JavaScript 在 html 中添加一个 div 的代码如下所示：

```
<html>
  <head>
    <meta http-equiv="content-type" content="text/html; charset=UTF-8">
     <script type="text/javascript">
function add(){
var d=document.getElementById("aaa");
d.innerHTML='<div style="border:1px solid red;width:200px;height:200px;" >
添加 div</div>';
}
</script>
 </head>
 <body id="aaa">
   <input type="button" value="添加" onclick="add();"/>
 </body>
</html>
```

1.33.怎样用 JavaScript 提交一个 form？

参考答案：

使用 JavaScript 提交表单的代码如下所示：

```
<html>
  <head>
    <title>怎样用 Javascript 提交一个 form?</title>
    <meta http-equiv="content-type" content="text/html; charset=UTF-8">
```

```
        <script type="text/javascript">
        function save(){
          document.myform.submit();
    }
        </script>
      </head>
      <body>
      <form id="myform" name="myform" action="save.html">
        <input type="text" name="user" id="username" /><br/>
        <input type="password" name="pwd" id="userpwd" /><br/>
        <input type="button" value="提交" onclick="save();"/>
    </form>
      </body>
    </html>
```

1.34.使用 JavaScript 在 html 页面中 id 为 mytable 的 div 中动态生成一个 2 行 2 列的表格？

参考答案：

使用 JavaScript 在 html 页面中 id 为 mytable 的 div 中动态生成一个 2 行 2 列的表格的代码如下所示：

```
    <html>
      <head>
        <meta http-equiv="content-type" content="text/html; charset=UTF-8">
         <script type="text/javascript">
    function add(){
    var d=document.getElementById("aaa");
    d.innerHTML='<table border="1" width="150px" height=120px align="center"
cellspacing="0"><tr><td>a</td><td>b</td></tr><tr><td>c</td><td>d</td></tr></
table>';
    }
    </script>
      </head>
      <body>
        <div            id="aaa"            style="border:1px            solid
red;width:200px;height:200px;" ></div>
        <input type="button" value="添加" onclick="add();"/>
      </body>
    </html>
```

1.35.WEB 页面开发，有"Btn_set”按钮和 "Btn_xsbh”的文本框，用 JS 编程，点击按钮，当文本框内容为空时，为文本框赋值 “Test”，不为空时，让文本框不可再输入？

参考答案：

题目中要求的代码如下所示：

```
<html>
  <head>
    <script type="text/javascript">
    function f(){
     var s=document.getElementById("aaa");
     if(s.value.length==0){
         s.value="Test";
     }else{
         s.disabled=true;
     }
 }
    </script>

  </head>

  <body>
    <input type="button" name="user" value="Btn_set" onclick="f();">
 <input type="text" name="Btn_xsbh" id="aaa" >
  </body>
</html>
```

1.36.输入 N 个数字，用逗号隔开。当你按提交按钮时 出来的数字按照顺序排序（什么顺序都可以）？

参考答案：

输入 N 个数字，用逗号隔开，当你按提交按钮时，出来的数字按照顺序排序，代码如下所示：

```
<html>
 <head>
     <title>Q001.html</title>
     <meta http-equiv="content-type" content="text/html; charset=UTF-8">
     <script type="text/javascript">
//数组排序
function operateArray(t) {
 //拆分为数组
 var array = document.getElementById("txtNumbers").value.split(",");
    array.sort(sortFunc);
 alert(array.toString());
}

function sortFunc(a, b) {
 return a - b;
}
</script>

 </head>

 <body>
     <input type="text" id="txtNumbers" value="12,4,3,123,51" />
     <input type="button" value="数组排序（数值）" onclick="operateArray();" />
     <br>
 </body>
</html>
```

1.37.HTML 中 和的区别。请写出至少 5 种常见的 http 状态码以及代表的意义？

参考答案：

1. 多个 内的元素会换行显示，多个内的元素显示在同一行。

2. 5 种常见的 http 状态码以及代表的意义如下：

1）200 OK:请求已成功，请求所希望的响应头或数据体将随此响应返回。

2）302 Found:请求的资源现在临时从不同的 URI 响应请求。

3）404 Not Found:请求失败，请求所希望得到的资源未被在服务器上发现。

4）500 Internal Server Error:服务器遇到了一个未曾预料的状况，导致了它无法完成对请求的处理。

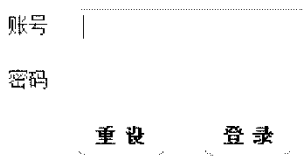
5）400 Bad Request:1)语义有误，当前请求无法被服务器理解。除非进行修改，否则客户端不应该重复提交这个请求。2)请求参数有误。

1.38.用 JavaScript 动态生成一张图片，图下面是字符（图片人物名字）？

参考答案：

```
<html>
  <head>
    <title>Q007.html</title>
    <meta http-equiv="content-type" content="text/html; charset=UTF-8">
<script language="javascript">
    function SelectPic(cc)
    {
      var pIC= document.createElement("img");
      ID=pIC.id="pic"+Math.random()+cc.src+Math.random();//避免 id 相同图片不显示
      pIC.alt=ID; //图片不显示时的提示
      pIC.width=cc.width;
      pIC.height=cc.height;
      pIC.onclick=function()
      { //调用函数
        SelectPic(this)
      }
      document.getElementById("PicC").appendChild(pIC);
      document.getElementById(ID).src=cc.src;
      document.getElementById("picTxt").innerHTML="李大钊";
}
    </script>

</head>
<body>
    <form id="form1" >
        <div>
            <img  src="images/7.jpg"  name="pic3"  onclick="SelectPic(this)" style="height:150px; width:90px" />
            <div id="PicC"></div>
            <div id="picTxt"></div>
        </div>
    </form>
</body>
</html>
```

1.39.请用 HTML 标签写出如下图所示的登录框？



参考答案：

```
<html xmlns="http://www.w3.org/1999/xhtml">
    <head>
        <meta http-equiv="Content-Type" content="text/html; charset=utf-8" />
        <title>达内 - NetCTOSS</title>
    </head>
    <body class="index">
        <div class="login_box">
            <table >
              <tr>
                  <td class="login_info">账号：</td>
                  <td><input       name="info.adminCode"       type="text"
class="width150" /></td>
              </tr>
              <tr>
                  <td class="login_info">密码：</td>
                  <td    ><input    name="info.password"    type="password"
class="width150" /></td>
              </tr>
              <tr>
                  <td  colspan="2">
                      <input type="button" value="重置"/>
                      <input type="button" value="登录"/>
                  </td>
              </tr>
            </table>
        </div>
    </body>
</html>
```

1.40.有下面 HTML 代码，请用 JS 获取 checkbox,radio,select 选中的值？

```
<html>
 <head>
 </head>
<body>
    <form action=" #" >
        <div>爱好：<input type=" checkbox"  name=" hobby"  value=" 篮球" >篮球;
                <input type="  checkbox"   name=" hobby"  value=" 音乐" >音乐;
                    <input type="  checkbox"  name=" hobby"  value=" 跑步" >跑步
        <div>
        <div>
            性别：<input type=" radio"  name=" gender"  value=" 男" >男;
                <input type=" radio"   name="  gender"  value=" 女" >女
        </div>
        <div>
```

```
        年龄：<select>
                <option value="" >请选择</option>
                <option value=" 18" >18</option>
                <option value=" 19" >19</option>
                <option value=" 20" >20</option>
                <option value=" 21" >21</option>
                <option value=" 22" >22</option>
            </select>
    </div>
</form>
</body>
</html>
```

参考答案：

```
<html>
  <head>
    <title>Q010.html</title>
    <meta http-equiv="content-type" content="text/html; charset=UTF-8">
     <script type="text/javascript">
      function foo()
     {
        var chkObj = document.getElementsByName("hobby");
     for(var i=0;i<chkObj.length;i++){
     if(chkObj[i].checked){
       alert("复选框："+chkObj[i].value);
      }
     }

        var radObj = document.getElementsByName("gender");
     for(var i=0;i<radObj.length;i++){
     if(radObj[i].checked){
       alert("单选按钮："+radObj[i].value);
      }
     }

    var selObj=document.getElementsByTagName("select")[0];
    for(var i=0;i<selObj.length;i++){
      if(selObj[i].selected){
         alert("下拉列表框："+selObj[i].value);
      }
    }
    }
     </script>
  </head>
  <body>
     <form action="#">
         <div>
             爱好：
             <input type="checkbox" name="hobby" value="篮球">
             篮球；
             <input type="checkbox"  name="hobby" value="音乐">
             音乐；
             <input type="checkbox" name="hobby" value="跑步">
             跑步
             <div>
                 <div>
```

```
                性别：
                <input type="radio" name="gender" value="男">
                男；
                <input type="radio" name="gender" value="女">
                女
            </div>
            <div>
                年龄：
                <select>
                    <option value="">
                        请选择
                    </option>
                    <option value="18">
                        18
                    </option>
                    <option value="19">
                        19
                    </option>
                    <option value="20">
                        20
                    </option>
                    <option value="21">
                        21
                    </option>
                    <option value="22">
                        22
                    </option>
                </select>
            </div>
        </div>
          <input type="button" value="查看" onclick="foo()">
        </div>
    </form>
  </body>
</html>
```

1.41.用 JS 动态添加表格行，删除表格行？

参考答案：

```
<html>
 <head>
    <title>Q011.html</title>
    <meta http-equiv="content-type" content="text/html; charset=UTF-8">
    <script type="text/javascript">
//为表格添加行
function addRow() {
 //得到表格对象
 var table = document.getElementById("table1");

 //创建新行
 var row = table.insertRow(table.rows.length);

 //为行创建 id 单元格
 var idCell = row.insertCell(0);
 idCell.innerHTML = document.getElementById("txtID").value;

 //为行创建 name 单元格
 var nameCell = row.insertCell(1);
```

```
    nameCell.innerHTML = document.getElementById("txtName").value;

    //为行创建操作按钮的单元格
    var buttonCell = row.insertCell(2);
    var button = document.createElement("input");
    button.type = "button";
    button.value = "删除";
    button.onclick = function() {
        delFunc(this);
    };
    buttonCell.appendChild(button);
}

//删除按钮的单击事件
function delFunc(btnObj) {
    var isDel = confirm("真的要删除吗？");
    if (!isDel)
        return;

    //找到当前行的 ID
    var rowObj = btnObj.parentNode.parentNode;
    var id = rowObj.getElementsByTagName("td")[0].innerHTML;

    //循环行，根据 id 定位需要删除的行，并删除
    var table = document.getElementById("table1");
    for ( var i = 1; i < table.rows.length; i++) {
        if (table.rows[i].cells[0].innerHTML == id) {
            table.deleteRow(i);
            break;
        }
    }

    //提示
    alert("删除 ID 为 " + id + " 的数据。");
}
</script>
</head>
<body>
    ID：
    <input type="text" id="txtID" />
    Name：
    <input type="text" id="txtName" />
    <input type="button" value="增加" onclick="addRow();" />
    <br />
    <br />
    <table id="table1">
        <tr class="header">
            <td>
                产品 ID
            </td>
            <td>
                产品名称
            </td>
            <td></td>
        </tr>
        <tr>
            <td>
                1
            </td>
            <td>
                book1
```

```
            </td>
            <td>
            <input type="button" value="删除" onclick="delFunc(this);" />
            </td>
        </tr>
    </table>
</body>
</html>
```

1.42.写出一个 JavaScript 表单验证 验证 HTML 表单中<input type="text" name="num" id="num">输入项必须为数字？

参考答案：

使用 JavaScript 校验文本框中的输入内容为数字的代码如下所示：

```
<html>
<head>
    <title>Q012.html</title>
    <meta http-equiv="content-type" content="text/html; charset=UTF-8">
    <script type="text/javascript">
function validate() {
var reg = new RegExp("^[0-9]+$");
var obj = document.getElementById("num");
if (!reg.test(obj.value)) {
    alert("请输入数字!");
}
}
</script>
</head>

<body>
    <input type="text"  name="num"  id="num">
    <input type="button" value="验证数字" onclick="validate()" />
    <br>
</body>
</html>
```

2. Servlet 和 JSP

2.1.什么是 B/S 结构？什么是 C/S 结构？

参考答案：

B/S 是 Browser/Server 的缩写，客户机上只要安装一个浏览器(Browser)，如 Netscape Navigator 或 Internet Explorer，服务器安装 Oracle、Sybase、Informix 或 SQL Server 等数据库。在这种结构下，用户界面完全通过 WWW 浏览器实现，一部分事务逻辑在前端实现，但是主要事务逻辑在服务器端实现。浏览器通过 Web Server 同数据库进行数据交互。

C/S 是 Client/Server 的缩写。服务器通常采用高性能的 PC、工作站或小型机，并采用大型数据库系统，如 Oracle、Sybase、Informix 或 SQL Server。客户端需要安装专用的客户端软件。

2.2.J2EE 是什么？

参考答案：

J2EE 是 Sun 公司提出的多层(multi-diered)、分布式(distributed)、基于组件(component-base)的企业级应用模型(enterprise application model)。在这样的一个应用系统中，可按照功能划分为不同的组件，这些组件又可在不同计算机上，并且处于相应的层次(tier)中。所属层次包括客户层(client tier)组件、web 层组件、Business 层组件以及企业信息系统(EIS)层。

2.3.J2EE 是技术是平台还是框架？

参考答案：

J2EE 本身是一个标准，一个为企业分布式应用的开发提供的标准平台；J2EE 也是一个框架，包括 JDBC、JNDI、RMI、JMS、EJB、JTA 等技术。

2.4.请对以下在 J2EE 中常用的名词进行解释(或简单描述)

1) JNDI
2) JMS
3) JTA
4) JAF
5) RMI

参考答案：

1) JNDI : (Java Naming & Directory Interface) Java 命名目录服务。主要提供的

功能是：提供一个目录系统，让其它各地的应用程序在其上面留下自己的索引，从而满足快速查找和定位分布式应用程序的功能；

2）JMS：（Java Message Service）JAVA 消息服务。主要实现各个应用程序之间的通讯，包括点对点和广播；

3）JTA：（ Java Transaction API）JAVA 事务服务。提供各种分布式事务服务，应用程序只需调用其提供的接口即可；

4）JAF：（Java Action FrameWork）JAVA 安全认证框架。提供一些安全控制方面的框架，让开发者通过各种部署和自定义实现自己的个性安全控制策略；

5）RMI:（Remote Method Invocation /internet 对象请求中介协议）他们主要用于通过远程调用服务。例如，远程有一台计算机上运行一个程序，它提供股票分析服务，我们可以在本地计算机上实现对其直接调用。当然这是要通过一定的规范才能在异构的系统之间进行通信。RMI 是 JAVA 特有的。

2.5.在 JavaEE 中，数据连接池的工作机制是什么？

参考答案：

JavaEE 服务器启动时会建立一定数量的池连接，并一直维持不少于此数目的池连接。客户端程序需要连接时，池驱动程序会返回一个未使用的池连接并将其标记为忙。如果当前没有空闲连接，池驱动程序就新建一定数量的连接，新建连接的数量由配置参数决定。当使用的池连接调用完成后，池驱动程序将此连接标记为空闲，其他调用就可以使用这个连接。

2.6.什么是 web 容器， 其作用是？ 尽可能写出你知道的容器名称

参考答案：

web 容器给处于其中的应用程序组件（JSP、SERVLET）提供一个环境，使 JSP、SERVLET 直接跟容器中的环境变量接口交互，不必关注其它系统问题。例如：TOMCAT、WEBLOGIC、WEBSPHERE 等都为 web 容器。

2.7.简述什么是 HTTP 协议？

参考答案：

HTTP 协议是 HyperText Transfer Protocol 的简写，它是由 w3c（万维网联盟）制定的一种应用层协议，用来定义浏览器与 web 服务器之间如何通信以及通信的数据格式。

2.8.Servlet 是什么？谈谈你的理解？

参考答案：

Servlet 是一种服务器端的 Java 应用程序，具有独立于平台和协议的特性，可以生成动态的 Web 页面。 它担当客户请求（Web 浏览器或其他 HTTP 客户程序）与服务器响应

（HTTP 服务器上的数据库或应用程序）的中间层。与传统的从命令行启动的 Java 应用程序不同，Servlet 由 Web 服务器进行加载，该 Web 服务器必须包含支持 Servlet 的 Java 虚拟机。

2.9.请简述 servlet 工作原理？

参考答案：

Servlet 是运行在 Servlet 容器中的，由 Servlet 容器来负责 Servlet 实例的查找、创建以及整个生命周期的管理，Servlet 整个生命周期可以分为四个阶段：类装载及实例创建阶段、实例初始化阶段、服务阶段以及实例销毁阶段。

1）类装载及实例创建阶段：

默认情况下，Servlet 实例是在接受到第一个请求时进行创建并且以后的请求进行复用，如果有 Servlet 实例需要进行一些复杂的操作，需要在初始化时就完成，比如打开文件、初始化网络连接等，可以配置在服务器启动时就创建实例，具体配置方法为在声明 servlet 标签中添加<load-on-startup>1</load-on-startup>标签。

2）初始化 init(ServletConfig config)：

一旦 Servlet 实例被创建，将会调用 Servlet 的 init 方法，同时传入 ServletConfig 实例，传入 Servlet 的相关配置信息，init 方法在整个 Servlet 生命周期中只会调用一次。

3）服务 service()：

为了提高效率，Servlet 规范要求一个 Servlet 实例必须能够同时服务于多个客户端请求，即 service()方法运行在多线程的环境下，Servlet 开发者必须保证该方法的线程安全性。

4）销毁 destory()：

当 Servlet 容器将决定结束某个 Servlet 时，将会调用 destory()方法，在 destory 方法中进行资源释放，一旦 destory 方法被调用，Servlet 容器将不会再发送任何请求给这个实例，若 Servlet 容器需再次使用该 Servlet，需重新再实例化该 Servlet 实例。

2.10.简述如何处理表单提交的中文？

参考答案：

处理表单提交的中文，分为两种情况，一是 post 方式提交表单、一是 get 方式提交表单。

1. 处理 post 方式提交表单时的中文，步骤如下：

- step1：确保表单所在的页面按照指定的字符集打开，在 HTML 中设置如下：

```
<meta http-equiv = "content-type" content = "text/html;charset=utf-8" >
```

另外，在 HTML 中，将表单 form 的提交方式设置为 post。

- step2：在服务器端按照上述设置的编码格式进行解码，代码如下：

```
request.setCharacterEncoding( "utf-8" );
```

该行代码要在第一次使用 request 的时候进行设置。

2. 处理 get 方式提交表单时的中文，步骤如下：

- step1：使用 meta 确保表单所在页面按指定字符集打开，在 HTML 中设置如下：

```
<meta http-equiv = "content-type" content = "text/html;charset=utf-8" >
```

另外，在 HTML 中，将表单 form 的提交方式设置为 get。

- step2：将从表单中获取的信息使用上述设置的字符集 utf-8 进行重新编码。例如：将从表单获取的 username 进行重新编码，代码如下：

```
String username = request.getParameter( "username" );
username = new String(username.getBytes( "iso-8859-1" ), "utf-8" );
```

2.11.简述 GET 和 POST 的区别？

参考答案：

GET 和 POST 的区别如下：

1. 从提交的数据量上来说，get 方式会将请求参数及参数值放在请求资源路径里面，携带的数据大小有限制，不适合提交大量的数据；post 方式会将请求参数及参数值放在实体内容里面，理论上没有限制，适合大量数据的提交。

2． 从安全上来讲，post 方式相对安全(因为请求参数及值存放在实体内容里面，而 get 方式会将请求参数及值显示在浏览器地址栏)。但是要注意，post 方式并没有将数据加密。

2.12.简述什么是重定向？

参考答案：

服务器向浏览器发送一个 302 状态码及一个 Location 消息头(该消息头的值是一个地址，称之为重定向地址)，浏览器收到后会立即向重定向地址发出请求。

2.13.Servlet 是否是线程安全的,如何解决？

参考答案：

Servlet 存在线程安全问题。容器收到请求之后，会启动一个线程来进行相应的处理。

默认情况下，容器只会为某个 Servlet 创建一个实例，如果同时有多个请求同时访问某个 Servlet 则肯定会有多个线程访问同一个 Servlet 实例。如果这些线程要修改 Servlet 实例的某个属性，就有可能发生线程安全问题。

可以使用 synchronized 对代码加锁来解决 Servlet 的安全问题。

2.14.简述转发和重定向有什么区别？

参考答案：

转发和重定向的区别有以下几点：

1. 重定向是浏览器发送请求并收到响应以后再次向一个新地址发请求，转发是服务器收到请求后为了完成响应转到一个新的地址。
2. 重定向中有两次请求，不共享数据，转发只产生一次请求，且在组件间共享数据。
3. 重定向后地址栏地址改变，而转发则不会。
4. 重定向的新地址可以是任意地址，转发到的新地址必须是同一个应用内的某地址。

2.15.简述什么是转发？以及如何实现转发？

参考答案：

转发是一个 Web 组件（Servlet/JSP）将未完成的处理通过容器转交给另外一个 Web 组件继续完成。

可以按照以下三个步骤来实现转发：

1.绑定数据到 request 对象，代码如下：

```
request.setAttribute(String name,Object obj);
```

2.获得转发器，代码如下：

```
RequestDispatcher  rd =
                request.getRequestDispatcher( String uri);
```

3.转发,代码如下：

```
rd.forward ( request , response ) ;
```

2.16.指出/images/123.jpg 与 images/123.jpg 两种写法的区别？

参考答案：

“/images/123.jpg”是绝对地址，根目录下的 images 子目录下的 123.jpg 文件；

“images/123.jpg”是相对地址，位置是相对于当前目录而言，指当前目录下的 images 子目录下的 123.jpg 文件。

2.17.Tomcat 服务器的默认端口是什么？怎样修改 Tomcat 的端口为 8009？

参考答案：

Tomcat 服务器的默认端口是 8080。改 tomcat 端口的方法如下：

首先，在 Tomcat 安装根目录下的子文件夹 conf 中找到文件 server.xml；

然后，将其用记事本程序打开，找到下列配置代码：

```
   <Connector     port="8080"     maxHttpHeaderSize="8192"     maxThreads="150"
minSpareThreads="25"        maxSpareThreads="75"        enableLookups="false"
redirectPort="8443"        acceptCount="100"        connectionTimeout="20000"
disableUploadTimeout="true" />
```

将以上配置代码中的 8080 更改为 8009，即修改了 Tomcat 的端口为 8009。

2.18.以 tomcat 为例，描述一个 jsp 数据请求，到数据返回的过程，该过程中 tomcat 内部机制如何实现？

参考答案：

Tomcat 服务器是一个免费的开放源代码的 Web 应用服务器，属于轻量级应用服务器，在中小型系统和并发访问用户不是很多的场合下被普遍使用，是开发和调试 JSP 程序的首选。Tomcat 处理一个 http 请求的过程，假设来自客户的请求为:

```
http://localhost:8080/wsota/wsota_index.jsp
```

对上述请求的执行步骤如下：

1) 请求被发送到本机端口 8080，被在那里侦听的 Coyote HTTP/1.1 Connector 获得；

2) Connector 把该请求交给它所在的 Service 的 Engine 来处理，并等待来自 Engine 的回应；

3) Engine 获得请求 localhost/wsota/wsota_index.jsp，匹配它所拥有的所有虚拟主机 Host；

4) Engine 匹配到名为 localhost 的 Host（即使匹配不到也把请求交给该 Host 处理，因为该 Host 被定义为该 Engine 的默认主机）；

5) localhost Host 获得请求/wsota/wsota_index.jsp，匹配它所拥有的所有 Context；

6) Host 匹配到路径为/wsota 的 Context（如果匹配不到就把该请求交给路径名为"" 的 Context 去处理）；

7) path="/wsota"的 Context 获得请求/wsota_index.jsp，在它的 mapping table 中寻找对应的 servlet；

8) Context 匹配到 URL PATTERN 为*.jsp 的 servlet，对应于 JspServlet 类；

9) 构造 HttpServletRequest 对象和 HttpServletResponse 对象，作为参数调用 JspServlet 的 doGet 或 doPost 方法；

10) Context 把执行完了之后的 HttpServletResponse 对象返回给 Host；

11) Host 把 HttpServletResponse 对象返回给 Engine；

12) Engine 把 HttpServletResponse 对象返回给 Connector；

13) Connector 把 HttpServletResponse 对象返回给客户浏览器。

2.19.熟悉 Tomcat 吗？你知道在 Tomcat server.xml 文件中都配置了哪些内容？

参考答案：

server.xml 文件描述了如何启动 Tomcat Server。文件格式如下：

```
<Server>
   <Listener />
   <GlobaNamingResources>
   </GlobaNamingResources
   <Service>
      <Connector />
      <Engine>
         <Logger />
         <Realm />
           <host>
              <Logger />
              <Context />
           </host>
      </Engine>
   </Service>
</Server>
```

下面介绍下每个元素：

1. <Server>元素：它代表整个容器，是 Tomcat 实例的顶层元素，由 org.apache.catalina.Server 接口来定义。它可以包含一个或多个<Service>元素，并且它不能做为任何元素的子元素。

2. <Service>元素：一个“Service”是一个或多个共用一个单独“Container”(容器)的“Connectors”组合(因此，应用程序在容器中可见)。通常，这个容器是一个“Engine”（引擎），但这不是必须的。该元素由 org.apache.catalina.Service 接口定义，它包含一个<Engine>元素,以及一个或多个<Connector>，这些Connector元素共享用同一个Engine元素。

3. <Connector>元素：代表与客户程序实际交互的组件，它负责接收客户请求，以及向客户返回响应结果。请看下列配置代码：

```
   <Connector      port="8080"      maxThread="50"      minSpareThreads="25"
maxSpareThread="75"         enableLookups="false"         redirectPort="8443"
acceptCount="100"             debug="0"             connectionTimeout="20000"
disableUploadTimeout="true" />
   <Connection port="8009" enableLookups="false" redirectPort="8443" debug="0"
protocol="AJP/1.3" />
```

上述代码中，第一个 Connector 元素定义了一个 HTTP Connector，它通过 8080 端口接收 HTTP 请求；第二个 Connector 元素定义了一个 JD Connector，它通过 8009 端口接收由其它服务器转发过来的请求。

4. <Engine>元素 :每个 Service 元素只能有一个 Engine 元素。处理在同一个<Service>中所有<Connector>元素接收到的客户请求，由 org.apahce.catalina.Engine 接口定义.

一个“Engine”（引擎）代表处理每个请求的入口点。这个 Tomcat 的标准独立引擎实现分析包含在请求中的 HTTP 头信息，并将请求传送到适当的主机或虚拟主机上。

5. <Host>元素：一个 Engine 元素可以包含多个<Host>元素，每个<Host>的元素定义了一个虚拟主机，它包含了一个或多个 Web 应用。

6. <Context>元素：是使用最频繁的元素，每个<Context>元素代表了运行在虚拟主机上的单个 Web 应用。一个<Host>可以包含多个<Context>元素，每个 web 应用有唯一的一个相对应的 Context。

2.20.CORBA 是什么？用途是什么？

参考答案：

CORBA 标准是公共对象请求代理结构(Common Object Request BrokerArchitecture)，由对象管理组织(Object Management Group，缩写为 OMG)标准化。它的组成是接口定义语言(IDL)，语言绑定(binding:也译为联编)和允许应用程序间互操作的协议。其目的为：用不同的程序设计语言书写在不同的进程中运行，为不同的操作系统开发。

2.21.如何从 form 表单中得取 checkbox 的值？

参考答案：

可在页面把 checkbox 的 name 属性取同一个，value 属性取每个条目的 id，在 Servlet 中使用 HttpServletRequest 的 getParamterValues(“name”)能取到 checkbox 的一组值。

2.22.请画出 Servlet 2.2 以上 Web Application 的基本目录结构？

参考答案：

web 程序的基本结构：

```
web 工程(dir)
    |--页面，图片，CSS，JS
    |--WEB-INF(dir)
            |--classes(dir) .class 文件(可选)
            |--lib（dir）  .jar  （可选）
            |--web.xml     web 工程的信息描述文件
```

Web 工程下可以有页面，图片，样式，脚本以及 WEB-INF 目录，在 WEB-INF 目录下 web.xml 文件是 web 工程的信息描述文件，classes 目录是放置.class 文件的地方，lib 目录是放置.jar 文件的地方。

2.23.Servlet 执行时一般实现哪几个方法？

参考答案：

public void init(ServletConfig config)
public ServletConfig getServletConfig()
public String getServletInfo()
public void service(ServletRequest request,ServletResponse response)
public void destroy()

2.24.说出 Servlet 的生命周期，并说出 Servlet 和 CGI 的区别？

参考答案：

Web 容器加载 Servlet 并将其实例化后，Servlet 生命周期开始，容器运行其 init 方法进行 Servlet 的初始化，请求到达时运行其 service 方法，service 方法自动派遣运行与请求对应的 doXXX 方法（doGet，doPost）等，当服务器决定将实例销毁的时候调用其 destroy 方法。

与 CGI 的区别在于 servlet 处于服务器进程中，它通过多线程方式运行其 service 方法，一个实例可以服务于多个请求，并且其实例一般不会销毁，而 CGI 对每个请求都产生新的进程，服务完成后就销毁，所以效率上低于 servlet。

2.25.请简述 Request,Response 的作用？

参考答案：

Request 对象：提供了当客户端请求一个页面或者传递一个窗体时，客户端提供的全部信息。这包括能指明浏览器和用户的 HTTP 变量，在这个域名下存放在浏览器中的 cookie，任何作为查询字符串而附于 URL 后面的字符串或页面的 < FORM > 段中的 HTML 控件的值。也提供使用 Secure Socket Layer（SSL）或其他加密通信协议的授权访问，及有助于对连接进行管理的属性。

Response 对象：用来访问服务器端所创建的并发回到客户端的响应信息。提供 HTTP 变量，指明服务器和服务器的功能和关于发回浏览器的内容的信息，以及任何将为这个域而存放在浏览器里新的 cookie。它也提供了一系列的方法用来创建输出，例如无处不在的 Response.write 方法。

2.26.请说明 cookie，request，session，application 的作用域和生命周期,举例说明他们用在什么场景？

参考答案：

request 的生命周期是一次请求。可以用于 JSP 表单提交数据；

session 会话可以设置它的时间，tomcat 中的默认时间为 30 分钟，当你关闭浏览器，

结束本次会话。Session 对象的典型应用是用来对用户的身份进行验证，进而进行权限设置；

cookie 存放的载体在客户端的浏览器中，生命周期默认是根据服务器返回的 Set-Cookie 头设置的。有 2 大类：

1）会话 cookie：浏览器一关就没了；

2）有过期时间：超过设定的过期时间才消失。Cookie 能使用户在不键入密码和用户名的情况下进入曾经浏览过的一些站点；

application 生命周期在整个应用程序中。生命周期为应用程序启动到停止。application 对象的最常见的应用是用来统计页面的访问人数或者是记录网站的在线人数。

2.27.描述一下界面的跳转方式有几种？分别是什么？

参考答案：

界面跳转方式有以下 8 种：

1.
2. response.sendRedirect("");
3. response.setHeader("Location","");
4. <jsp:forward page="" />
5. request.getRequestDispatcher("").forward(request, response);
6. 通过表单提交：<form method=" " name=" " action=" " >
7. 通过 JS 方式：
   1)window.location.href="http://www.baidu.com/"；
   2)window.navigate()；
   3)window.location.replace("http://www.baidu.com/")；
   4)self.location='' ；
   5)top.location='' ；
   6)window.history.back(-1)/history.go()；
8. <meta http-equiv="refresh" content="300; url=target.html" > ：在 5 分钟之后正在浏览的页面将会自动变为 target.html 这一页。

2.28.谈谈你对 Servlet 的过滤器的理解？

参考答案：

过滤器是 Servlet2.3 规范中定义的一种小型的、可插入的 Web 组件。用来拦截 Servlet 容器的请求和响应过程，以便查看、提取客户端和服务器之间正在交换的数据。过滤器通常是封装了一些功能的 Web 组件，这些功能很重要，但对于处理客户端请求或发送响应来说不是决定性的。典型的应用包括记录请求和响应的数据、管理会话属性等。

2.29.简述什么是监听器？

参考答案：

Servlet 规范中定义的一种特殊的组件，用来监听 Servlet 容器产生的事件并进行相应的处理。

2.30.描述 Cookie 和 Session 的作用，区别和各自的应用范围。Session 工作原理？

参考答案：

1. Cookie 和 Session 的作用如下:

Cookie 是网站保存在浏览器客户端的信息，也就是说保存在访客的机器里的变量，一般随着 HTTP 头发送到客户端。在 Cookie 生效之后及失效之前，客户每次发出页面请求的时候，都会把 Cookie 一块发送到服务器，只要我们针对它进行相应的处理，就可以改变它的值。

Session 的中文译名叫做“会话”，其本来的含义是指有始有终的一系列动作/消息，比如用户在浏览某个网站时，从进入网站到浏览器关闭所经过的这段时间，也就是用户浏览这个网站所花费的时间。

2. Cookie 和 Session 的区别和各自的应用范围如下：

1）cookie 数据存放在客户的浏览器上，session 数据放在服务器上。

2）cookie 不是很安全，别人可以分析存放在本地的 Cookie 并进行 Cookie 欺骗，考虑到安全应当使用 session。

3）session 会在一定时间内保存在服务器上。当访问增多，会比较占用服务器的资源，考虑到提高服务器的性能，应当使用 Cookie。

4）单个 cookie 保存的数据不能超过 4K，很多浏览器 Cookie 有数量限制。

5）将登录信息等重要信息存放为 Session；其它信息如果需要保留，可以放在 Cookie 中。

3.Session 的工作原理如下：

session 机制是一种服务器端的机制，服务器使用一种类似于散列表的结构（也可能就是使用散列表）来保存信息。当程序需要为某个客户端的请求创建一个 session 时，服务器首先检查这个客户端的请求里是否已包含了一个 session 标识（称为 session id），如果已包含则说明以前已经为此客户端创建过 session，服务器就按照 session id 把这个 session 检索出来使用（检索不到，会新建一个），如果客户端请求不包含 session id，则为此客户端创建一个 session 并且生成一个与此 session 相关联的 session id，session id 的值应该是一个既不会重复，又不容易被找到规律以仿造的字符串，这个 session id 将被在本次响应中返回给客户端保存。保存这个 session id 的方式可以采用 cookie，这样在交互过程中浏览器可以自动的按照规则把这个标识发送给服务器。一般这个 cookie 的名字都是类似于 SEEESIONID。但 cookie 可以被人为的禁止，则必须有其他机制以便在 cookie 被禁止时仍然能够把 session id 传递回服务器。

2.31.session 如何存取？何时被创建？session 何时被删除？如何在关闭浏览器的时候删除 session？

参考答案：

1. 通过 HttpServletRequest 对象的 getSession()来获取 session。
   存储数据方式：session.setAttribute("key",value);
   获取数据方式：session.getAttribute("key");
2. Session 对象在调用 HttpServletRequest.getSession(true)语句时被创建。
3. 删除 session：
   1）调用 HttpSession.invalidate()方法;
   2)距离上一次收到客户端发送的 session id 时间间隔超过了 session 的超时设置。
4. 关闭浏览器的时候删除 session：

在页面中添加 onunload 事件，当关闭浏览器时，执行服务器端删除 session 代码；

优点：退出时，能及时进行处理。缺点：当用户打开多个页面时，关闭任何一个页面都有可能导致用户的退出。

2.32.HttpSession session = request.getSession()与 HttpSession session = request.getSession(true)的区别？getParameter 与 getAttribute 的区别？

参考答案：

1）HttpSession session = request.getSession()与 HttpSession session = request.getSession(true)的区别如下：

HttpSession session = request.getSession(true) 表示当 flag 为 true 时：先查看请求中有没有 SessionId，如果没有 SessionId，服务器创建一个 Session 对象；如果有 SessionId，依据 SessionId 查找对应 Session 对象，找到则返回，找不到则创建一个新的 Session 对象，所以 flag 为 true 时，一定能得到一个 Session 对象；当 flag 为 false 时，没有 SessionId 及有 SessionId 但没有找到 Session 对象，均返回 null；找到则返回。HttpSession session = request.getSession() 等价于 HttpSession session =request.getSession（true）；提供该方法是为了代码书写更方便一些，大部分情况下是不管找没找到都需要返回一个 Session 对象

2）getParameter 与 getAttribute 的区别如下：

request.getParameter()方法是获得客户端传送给服务器的参数值，代表 http 请求数据。由 URL 传入或由 FORM 提交的内容，返回值是 String 类型。

request.getAttribute()方法获得属性值，数据在具有转发关系的 Web 组件之间共享，返回值为 Object 类型。

2.33.在 JSP 页面中，能否获得 Servlet 中 request.setAttribute 中的值，如果能获得，通过何种方式，如果不能获得，请简述为什么？

参考答案：

servlet 通过转发方式跳转到 JSP 页面，JSP 页面就可以获得 request 中属性的值。

例如：servlet 代码如下：

```
request.setAttribute("name","zs");
request.getRequestDispatcher("/XX.jsp").forward(request,response);
```

JSP 代码如下：

```
<%=(String)request.getAttribute("name") %>
```

2.34.简述什么是 Session 超时，如何修改缺省的时间限制？

参考答案：

Session 超时指的是：Web 服务器会将空闲时间过长的 Session 对象删除掉，以节省服务器内存空间资源。web 服务器缺省的超时时间限制：一般是 30 分钟。修改 Session 的缺省时间限制，有如下两种方式：

1. 通过修改 tomcat 中 conf/web.xml 文件的设置，代码如下所示：

```
<session-config>
 <session-timeout>30</session-timeout>
</session-config>
```

2. 通过编程的方式来修改，通过调用 Session 对象的 setMaxInactiveInterval 方法来修改，该方法的声明如下所示：

```
void  setMaxInactiveInterval ( int seconds ){}
```

2.35.JSP 标准提供了三种独立的向 JSP 添加 java 代码的技术,请列举？

参考答案：

JSP 标准提供了三种独立的向 JSP 添加 java 代码的技术，分别是：

1） <% %> JSP 小脚本

2） <%! %> JSP 声明

3） <%= %> JSP 表达式

2.36.JSP 中的 include 有两种形式，分别用在何处？

参考答案：

动态 INCLUDE 用 jsp:include 动作实现 <jsp:include page="head.jsp"/> 。它总

是会检查所含文件中的变化，适合用于包含动态页面，并且可以带参数；

静态 INCLUDE 用 include 伪码实现<%@include file="head.htm" %>。它不会检查所含文件的变化，适用于包含静态页面。

2.37.在 JSP 中：

1. 如何获得当前 Web 应用在文件系统里的绝对路径？
2. 如何获得 Web 应用中某一文件的绝对路径？

参考答案：

1. 获得当前 Web 应用在文件系统里的绝对路径的方式如下：

```
<%= application.getRealPath("/"); %>
```

2. 获取应用中/WebRoot/img 文件夹里的 a.png 文件的绝对路径的方式如下：

```
<%= request.getContextPath()%>/img/a.png
```

2.38.JSP 和 Servlet 区别和联系是什么？

参考答案：

JSP 是 Servlet 技术的扩展，本质上是 Servlet 的简易方式，更强调应用的外表表达。JSP 编译后是"类 servlet"。Servlet 和 JSP 最主要的不同点在于，Servlet 的应用逻辑是在 Java 文件中，并且完全从表示层中的 HTML 里分离开来。而 JSP 的情况是 Java 和 HTML 可以组合成一个扩展名为.jsp 的文件。JSP 侧重于视图，Servlet 主要用于控制逻辑。

2.39.JSP 是怎么转化为 html 的？

参考答案：

JSP 在执行前先被转译成.Java 文件(servlet)，再编译成.class 文件。每个 JSP 实例都有个 jspservice 方法，而这个 jspservice 方法将动态数据解释成以 html 标记的内容，然后再用 JspWriter 对象将一段一段地内容写向服务器，之后刷新 JspWriter 对象和关闭它，最后客户端所得到的就是 html 内容了。

2.40.JSP 的四种范围？

参考答案：

JSP 的四种范围如下：

1. page 是代表与一个页面相关的对象和属性。一个页面由一个编译好的 Java servlet 类表示。这既包括 servlet 又包括被编译成 servlet 的 JSP 页面；

2. request 是代表与 Web 客户机发出的一个请求相关的对象和属性。一个请求可能跨越多个页面，涉及多个 Web 组件；

3. session 是代表与用于某个 Web 客户机的一个用户体验相关的对象和属性。一个 Web 会话可以也经常会跨越多个客户机请求；

4. application 是代表与整个 Web 应用程序相关的对象和属性。这实质上是跨越整个 Web 应用程序，包括多个页面、请求以及会话的一个全局作用域。

2.41.JSP 的内置对象及方法？

参考答案：

JSP 有 9 个内置对象 :request、response、out、session、pageContext、applicaton, config、page、exception。

1． request 表示 javax.servlet.http.HttpServletRequest 对象。用于获取客户端提供的数据，包括头信息、客户端地址、请求方式等。主要方法如下：

1） getCookies():返回客户端所有 Cookies 对象，结果是一个 Cookies 数组。

2） getSession():返回与请求相关的 session。

3） setAttribute(String key,Object obj) 设置属性的属性值。

4） getAttribute(String name) 返回指定属性的属性值。

5） request.getParameter(String name):获得客户端传送给服务器端的参数值。

6） getCharacterEncoding() 返回字符编码方式。

7） getContentType() 得到请求体的 MIME 类型。

2． response 表示 javax.servlet.http.HttpServletResponse 对象，用于对客户端的请求作出动态的响应，向客户端发送数据。主要方法有：

1） addCookie(Cookies c):添加一个 Cookie 对象，用来保存客户端的用户信息。

2） sendRedirect(String location):把响应发送到另一个位置进行处理。

3） sendError(int):向客户端发送错误的信息。

4） getBufferSize():返回缓冲区的大小。

5） getOutputStream():返回客户端的输出流对象。

6） setContentType(String ContentType):设置相应的 MIME 类型。

7） setHeader(String name,String value):设置指定名字的 HTTP 文件头的值，如果已经存在则将覆盖已经存在的旧值。

8） encodeURL():使用 sessionId 封装 URL。如果没有必要封装 URL，返回原值。

3． out 对象是 javax. servlet.jsp.JspWriter 的一个实例，用来向客户端输出各种数据。主要方法如下：

1） print()/println():根据参数类型输出各种类型的数据。

2） flush():输出缓冲区的数据。

3） close():关闭输出流。

4） clear():清除缓冲区里的数据，但不会把数据输出到客户端。

5） getBufferSize():获得缓冲区的大小。

6） clearBuffer():清除缓冲区里的数据，并把数据输出到客户端。

7） getRemaining():获得缓冲区中没有被占用的空间的大小。

8） isAutoFlush():返回布尔值，如果 AutoFlush 为真，返回 true；反之，返回 false。

4．pageContext 表示一个 javax.servlet.jsp.PageContext 对象。它是用于方便存取所有范围的名字空间的对象，并且包装了通用的 servlet 相关功能的方法。主要方法如下：

1） getOut() 返回当前客户端响应被使用的 JspWriter 流(out)。

2） getSession() 返回当前页中的 HttpSession 对象(session)。

3） getPage() 返回当前页的 Object 对象(page)。

4） getRequest() 返回当前页的 ServletRequest 对象(request)。

5） getResponse() 返回当前页的 ServletResponse 对象(response)。

6） getException() 返回当前页的 Exception 对象(exception)。

7） getServletConfig() 返回当前页的 ServletConfig 对象(config)。

8） getServletContext() 返回当前页的 ServletContext 对象(application)。

9） setAttribute(String name,Object attribute) 设置属性及属性值。

10） getAttribute(String name,int scope) 在指定范围内取属性的值。

11） removeAttribute(String name) 删除某属性。

12） release() 释放 pageContext 所占用的资源。

13） forward(String relativeUrlPath) 使当前页面重导到另一页面。

14） include(String relativeUrlPath) 在当前位置包含另一文件。

5．session 表示一个请求的 javax.servlet.http.HttpSession 对象。Session 用来保存用户的会话信息和会话状态。主要方法如下：

1） setAttribute(String name,Object obj):设置指定名字 name 的属性值 value，并存储在 session 对象中。

2） getAttribute(String name):获取与指定名字 name 相联系的属性。

3） invalidate() 取消 SESSION，使 SESSION 不可用。

4） getId() 返回 SESSION 创建时 JSP 引擎为它设的惟一 ID 号。

5） getCreationTime() 返回 SESSION 创建时间。

6） getLastAccessedTime() 返回此 SESSION 里客户端最近一次请求时间。

7） getMaxInactiveInterval() 返回两次请求间隔多长时间此 SESSION 被取消(ms)。

8） getValueNames() 返回一个包含此 SESSION 中所有可用属性的数组。

9） removeValue(String name) 删除 SESSION 中指定的属性。

6． application 表示一个 javax.servlet.ServletContext 对象。用于用户间的数据共

享，可以存放全局变量。主要方法如下：

1） getAttribute(String name) 返回给定名的属性值 。

2） getAttributeNames() 返回所有可用属性名的枚举 。

3） setAttribute(String name,Object obj) 设定属性的属性值 。

4） removeAttribute(String name) 删除一属性及其属性值 。

5） getServerInfo() 返回 JSP(SERVLET)引擎名及版本号 。

6） getRealPath(String path) 返回一虚拟路径的真实路径。

7） getContext(String uripath) 返回指定 WebApplication 的 application 对象 。

8） getResource(String path) 返回指定资源(文件及目录)的 URL 路径。

9）getRequestDispatcher(String uripath) 返回指定资源的 RequestDispatcher 对象。

7 .config 表示一个 javax.servlet.ServletConfig 对象。该对象用于存取 servlet 实例的初始化参数。主要方法如下：

1） getServletContext() 返回含有服务器相关信息的 ServletContext 对象 。

2） getInitParameter(String name) 返回初始化参数的值 。

3） getInitParameterNames() 返回 Servlet 初始化所需所有参数的枚举。

8. page 表示从该页面产生的一个 servlet 实例，即 this。

9 . exception 用于处理 JSP 页面发生的错误和异常。主要方法有：

1） getMessage() 返回描述异常的消息。

2） toString() 返回关于异常的简短描述消息。

3） printStackTrace() 显示异常及其栈轨迹。

4） fillInStackTrace() 重写异常的执行栈轨迹。

2.42.JSP 的常用指令？

参考答案：

JSP 指令包括 page、include 以及 taglib。page 指令是针对当前页面的指令；include 指令用来指定如何包含另外一个文件；taglib 指令用来定义和访问自定义标记库。

2.43.简述 EL 表达式的作用？

参考答案：

EL 表达式的作用可分为以下三类：

1. 访问 Bean 的属性。
2. 输出简单的运算结果。
3. 获取请求参数值。

2.44.JSP 标签的作用？如何定义？

参考答案：

1. JSP 标签的作用如下：

1）分离 JSP 页面的内容和逻辑；

2）业务逻辑开发者可以创建自定义标签；

3）封装业务逻辑；

4）可重用并且易维护；

5）易于手工修改、易于工具维护；

6）提供简洁的语法。

2. JSP 标签的定义:

1）编写标签处理器；

2）编写 tld 文件；

3）将标签处理器和 tld 文件放到同一个包里面；

4）把 jsp 页面和标签库配置部署在一起。

2.45.jsp 页面中，<% %>、<%！%>、<%= %>、<%-- --%>有什么区别？

参考答案：

<% %> 内部可直接嵌入 java 代码。

<%! %> 内部可以声明变量和方法，它们只当前 JSP 页面有效。

<%= %> 将变量或表达式显示在页面上。

<%-- --%> JSP 注释，其内部标记的字符会在 JSP 编译时被忽略掉。

2.46.写出熟悉的 JSTL 标签？

参考答案：

常用的标签：<c:out>、<c:remove>、<c:catch>、<c:set>

迭代标签：<c:foreach>

条件标签：<c:if>、<c:when>、<c:choose>、<c:otherwise>

URL 标签：<c:import>、<c:redirect>、<c:url>

2.47.JAVA B/S 模式,使用 Map 做数据存储,来实现增加、删除、修改、查询等功能？

参考答案：

ShoppingCartItem 类：

```
public class ShoppingCartItem implements java.io.Serializable{
 private String id;//唯一表示一条选购商品数据
 private String name;//商品名称
 private double price;//商品价格
```

```
    private double quantity;//商品数量
    public String getId() {
        return id;
    }
    public void setId(String id) {
        this.id = id;
    }
    public String getName() {
        return name;
    }
    public void setName(String name) {
        this.name = name;
    }
    public double getPrice() {
        return price;
    }
    public void setPrice(double price) {
        this.price = price;
    }
    public double getQuantity() {
        return quantity;
    }
    public void setQuantity(double quantity) {
        this.quantity = quantity;
    }
   }
```

ShoppingCart 类：

```
   import java.util.ArrayList;
   import java.util.HashMap;
   import java.util.Iterator;
   import java.util.List;
   import java.util.Map;

   import com.jessica.shop.shop.bean.ShoppingCartItem;

   public class ShoppingCart {
    protected       Map<String,ShoppingCartItem>       items       =       new
HashMap<String,ShoppingCartItem>();

    // 添加
    public void addItem(String id, String name, double price, double quantity)
{
        if (items.containsKey(id)) // 存在的话数量相加
        {
            ShoppingCartItem tempSC = (ShoppingCartItem) items.get(id);
            // 取出已经存在的这个商品
            tempSC.setQuantity(quantity + tempSC.getQuantity());
        } else {
            ShoppingCartItem sc = new ShoppingCartItem();
            sc.setId(id);
            sc.setName(name);
            sc.setPrice(price);
            sc.setQuantity(quantity);

            items.put(id, sc); // 存放到哈希中，模拟一个购物车
        }
    }

    // 更新
    public void editCart(String id, double quantity){
        ShoppingCartItem tempSC = (ShoppingCartItem) items.get(id);
```

```
        tempSC.setQuantity(quantity);
    }

    // 删除
    public void delCart(String id){
        items.remove(id);
    }

    // 获取所有
    public List getAllCarts(){
        ShoppingCartItem sc = null;
        List<ShoppingCartItem> list = new ArrayList<ShoppingCartItem>();
        Iterator it = items.keySet().iterator();
        while (it.hasNext()) {
            String key = (String) it.next();
            sc = (ShoppingCartItem) items.get(key);
            list.add(sc);
        }
        return list;
    }

    // 获取总数量
    public float getTotalQuantity(){
        ShoppingCartItem sc = null;
        Iterator it = items.keySet().iterator();
        float totalQuantity = 0;
        while (it.hasNext()) {
            String key = (String) it.next();
            sc = (ShoppingCartItem) items.get(key);
            totalQuantity += sc.getQuantity();
        }
        return totalQuantity;
    }

    // 获取总金额
    public float getTotalPrice(){
        ShoppingCartItem sc = null;
        Iterator it = items.keySet().iterator();
        float totalPrice = 0;
        while (it.hasNext()) {
            String key = (String) it.next();
            sc = (ShoppingCartItem) items.get(key);
            totalPrice += sc.getPrice() * sc.getQuantity();
        }
        return totalPrice;
    }

    }
```

2.48.请在以下 a.jsp 和 b.jsp 中补充代码，使得在一次会话中 b.jsp 能获取到 a.jsp 中 p1 变量的值？

a.jsp：

```
<%
String p1="124";

%>
```

b.jsp：

```
<%

%>
```

参考答案：

a.jsp：

```
<%
  String p1="124";
  session.setAttribute("p1",p1);
%>
```

b.jsp：

```
<%
 String p1=(String)session.getAttribute("p1");
%>
```

3. Ajax 和 JQuery

3.1. 简述对 Ajax 的理解？

参考答案：

AJAX 是 Asynchronous JavaScript and Xml 异步的 JavaScript 和 Xml。它一种用来改善用户体验的技术 ,其实质是 ,使用 XMLHttpRequest 对象异步地向服务器发请求。服务器返回部分数据 ,而不是一个完整的页面 ,以页面无刷新的效果更改页面中的局部内容。

3.2. 请说明 Ajax 同步模式与异步模式的区别，并根据您的开发经验，列出 Ajax 的应用场景？

参考答案：

1． ajax 设置同异步语句是：ajax.open("Post",url,true);其中第 3 个参数是设同步或者异步，true 表示为异步。Ajax 同步模式与异步模式的区别如下：

同步：提交请求->等待服务器处理->处理完毕返回。这个期间客户端浏览器不能做任何事。

异步: 请求通过事件触发->服务器处理（这时浏览器仍然可以做其他事情）->处理完毕

在同步的情况下，js 会等待请求返回，获取 status。不需要 onreadystatechange 事件处理函数。而异步则需要 onreadystatechange 事件处理 ,且值为 4 再处理下面的内容。如果想获得返回值必须用同步，因为异步无法得到返回值。

2． ajax 主要功能就是提供与服务器的异步交互，比如需要输入一个用户名，在输送完毕之后，没有确认提交，ajax 可以通过异步提交来实现仅仅将输入的用户名发送到服务器数据库进行检测，然后回复是否重复。而同步提交就像注册完毕，将信息存储到数据库后提示注册成功。

3.3. 什么是 JSON，在什么情况下使用？

参考答案：

JSON （JavaScript Object Notation）是一种轻量级的数据交换格式。易于人阅读和编写，同时也易于机器解析和生成。JSON 采用完全独立于语言的文本格式。

JSON 最常见的用法之一，是从 web 服务器上读取 JSON 数据，将 JSON 数据转换为 JavaScript 对象，然后在网页中使用该数据。

3.4. JQuery 选取页面元素 ID 为 deptid 的写法？

参考答案：

```
$("#deptid")
```

3.5. 说出 jQuery 中常见的几种函数以及他们的含义是什么？

参考答案：

jQuery 中常见的函数如下：

1） get() 取得所有匹配的 DOM 元素集合。

2） get(index) 取得其中一个匹配的元素。index 表示取得第几个匹配的元素。

3） append(content) 向每个匹配的元素内部追加内容。

4） after(content) 在每个匹配的元素之后插入内容。

5） html()/html(val) 取得或设置匹配元素的 html 内容。

6） find(expr) 搜索所有与指定表达式匹配的元素。

7） bind(type, [data], fn) 为每个匹配元素的特定事件绑定事件处理函数。

8） empty() 删除匹配的元素集合中所有的子节点。

9） hover(over, out) 一个模仿悬停事件（鼠标移动到一个对象上面及移出这个对象）的方法。

10）attr(name) 取得第一个匹配元素的属性值。

11）addClass(class)和 removeClass(class) 为指定的元素添加或移除样式。

12）css(name) 访问第一个匹配元素的样式属性。

13）ajax([options]) 通过 HTTP 请求加载远程数据。

14）get(url, [data], [callback], [type]) 通过远程 HTTP GET 请求载入信息。

15）post(url, [data], [callback], [type]) 通过远程 HTTP POST 请求载入信息。

16）load(url, [data], [callback]) 载入远程 HTML 文件代码并插入至 DOM 中。

3.6. 用 JSON、AJAX 读取指定目录下的文件，获取文件对象以及文件的大小、长度？

参考答案：

Q006Servlet 代码如下：

```
import java.io.File;
import java.io.IOException;
import java.io.PrintWriter;

import javax.servlet.ServletException;
import javax.servlet.http.HttpServlet;
import javax.servlet.http.HttpServletRequest;
import javax.servlet.http.HttpServletResponse;

public class Q006Servlet extends HttpServlet {
 public void service(HttpServletRequest request, HttpServletResponse
response)
        throws ServletException, IOException {
```

```
        response.setContentType("text/html;charset=utf-8");
        PrintWriter out = response.getWriter();
        File f = new File("tmp.txt");
        long fileSize = f.length();
        out.println("{'fileSize':" + fileSize + "}");
        out.close();
    }
    }
```

Q006.jsp 页面代码如下：

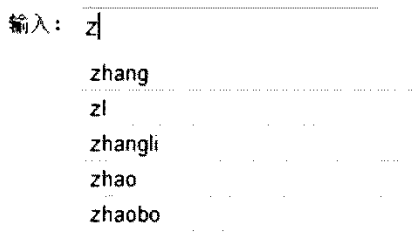
```
    <%@ page  contentType="text/html; charset=utf-8"%>
       <%
    String path = request.getContextPath();
    String                                basePath                                =
request.getScheme()+"://"+request.getServerName()+":"+request.getServerPort(
)+path+"/";
    %>
    <html>
    <head>
     <title>Insert title here</title>
     <script type="text/javascript">
         function getXmlHttpRequest(){
             var xhr = null;
             if((typeof XMLHttpRequest)!='undefined'){
                     xhr = new XMLHttpRequest();
                     }else {
                     xhr = new ActiveXObject('Microsoft.XMLHttp');
                  }
                 return xhr;
         }

         function getFileSize(){
             var xhr = getXmlHttpRequest();
             xhr.open('post','<%=basePath%>servlet/Q001Servlet',true);
             xhr.setRequestHeader("Content-Type",
                 "application/x-www-form-urlencoded");
             xhr.onreadystatechange=function(){

                 if(xhr.readyState == 4){

                     var txt = xhr.responseText;
                      alert(txt);
                      var dataObj=eval("("+txt+")");//转换为 json 对象
                     var saleInfo = '文件的大小为：'+dataObj.fileSize;

                     document.getElementById('d1').innerHTML = saleInfo;
                 }
             };
             xhr.send(null);
         }
     </script>
    </head>
    <body style="font-size:30px;" onload="getFileSize()">
     <div id="d1"></div>
    </body>
    </html>
```

web.xml 配置代码如下：

```
     <servlet>
       <servlet-name>Q006Servlet</servlet-name>
       <servlet-class>com.tarena.ajax.Q006Servlet</servlet-class>
```

```
  </servlet>

  <servlet-mapping>
    <servlet-name>Q006Servlet</servlet-name>
    <url-pattern>/servlet/Q006Servlet</url-pattern>
  </servlet-mapping>
```

3.7. Ajax 实现自动输入，如下图所示：



参考答案：

1.编写 Q007.jsp 页面，代码如下所示：

```
<%@ page language="java" import="java.util.*" pageEncoding="UTF-8"
contentType="text/html; charset=UTF-8"%>
<%
String path = request.getContextPath();
String basePath =
request.getScheme()+"://"+request.getServerName()+":"+request.getServerPort(
)+path+"/";
%>

<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN">
<html>
  <head>
    <base href="<%=basePath%>">

    <title>My JSP 'autoCompleteInput.jsp' starting page</title>

  <meta http-equiv="pragma" content="no-cache">
  <meta http-equiv="cache-control" content="no-cache">
  <meta http-equiv="expires" content="0">
  <meta http-equiv="keywords" content="keyword1,keyword2,keyword3">
  <meta http-equiv="description" content="This is my page">
  <!--
  <link rel="stylesheet" type="text/css" href="styles.css">
  -->
    <script type="text/javascript">
       var xmlHttp;

  function createXMLHttpRequest(){
      if (window.XMLHttpRequest)
        {// code for IE7+, Firefox, Chrome, Opera, Safari
               xmlHttp=new XMLHttpRequest();
            }else
           if (window.ActiveXObject)
           {// code for IE6, IE5
               xmlHttp=new ActiveXObject("Microsoft.XMLHTTP");
            }
      if(xmlHttp == null){
```

```
                alert("不能创建 XMLHttpRequest 对象");
            }
    }

    function makeRequest(){
        inputField = document.getElementById("searchField");
        str= inputField.value;
        if(str!=""){
            createXMLHttpRequest();
            xmlHttp.onreadystatechange= handleStateChange;
            var queryString="pre="+encodeURIComponent(str);
            xmlHttp.open("POST", "<%=basePath%>servlet/Q002Servlet", true);
            xmlHttp.setRequestHeader("Content-Type",
"application/x-www-form-urlencoded;");
            xmlHttp.send(queryString);
            /*
            var url= "AutoCompleteServlet?pre="+escape(str);
            xmlHttp.open("GET", url, true);
            xmlHttp.send(null);
            */
        }else{
            clearSuggestion();
        }
    }

    function handleStateChange(){
        if(xmlHttp.readyState==4){
            if(xmlHttp.status==200){
                displayNames();
            }else {
                 alert("Not able to retrieve description : " +xmlHttp.status+"
-- "+ xmlHttp.statusText);
                 }
        }
    }
    function displayNames(){
        var str = document.getElementById("searchField").value;
        document.getElementById("searchField").className="";
        var popUp= document.getElementById("popups");
        var results = xmlHttp.responseXML.getElementsByTagName("name");
        if(results.length>0){
            clearSuggestion();
            var suggest= null;
            for(var i=0;i<results.length;i++){
                suggest= document.createElement("div");

    suggest.appendChild(document.createTextNode(results[i].firstChild.nodeVal
ue));
                suggest.onclick= makeChoice;

    suggest.onmouseover="javascript:this.style.backgroundColor='orange';"
                suggest.className= "suggestions";
                popUp.appendChild(suggest);
            }
            if(popUp.childNodes.length==0)
               document.getElementById("searchField").className="error";
        }
    }
    //
    function makeChoice(evt){
        var thisDiv = (evt) ? evt.target : window.event.srcElement;
        document.getElementById("searchField").value = thisDiv.innerHTML;
        document.getElementById("popups").innerHTML="";
    }
    function clearSuggestion(){
        var models= document.getElementById("popups");
        while(models.childNodes.length>0){
```

```
            models.removeChild(models.childNodes[0]);
        }
    }
    </script>
    <style type="text/css">
          body,#searchField{
              font:1.1em arial , helvetica, sans-serif;
          }
          .suggestions{
              width:auto;
              background-color: #FFE3FF;
              padding: 2px 6px;
              border-bottom: 1px dotted #24f;
              cursor: hand;
          }
          .suggestions:hover{
              background-color: #C8C8FC;
          }
          #popups{
              position: absolute;
              width:200px;
          }
          #searchField.error{
              background-color: #ffc;
          }

    </style>
   </head>

   <body>
    <br>
    <br>
  <br><br>
  <center>
  <form action="#">
  <table >
  <tr>
  <td>输入：
  </td>
  <td><input type="text" id="searchField"  onkeyup="makeRequest()">
  </td>
  </tr>
  <tr>
  <td>
  </td>
  <td>
  <div id="popups" >
  </div>
  </td>
  </tr>
  </table>
  </form>
  </center>
   </body>
 </html>
```

2.编写 Q007Servlet，代码如下所示：

```
import java.io.IOException;
import java.io.PrintWriter;

import javax.servlet.ServletException;
import javax.servlet.http.HttpServlet;
import javax.servlet.http.HttpServletRequest;
import javax.servlet.http.HttpServletResponse;
```

```
public class Q007Servlet extends HttpServlet {

    private String[] info = { "zhang", "zl", "zhangli", "zhao", "zhaobo",
            "\u7528\u6237", "\u7528\u6237\u540D", "章", "章力好啊" };

    public Q002Servlet() {
        super();
    }

    public void doGet(HttpServletRequest request, HttpServletResponse response)
            throws ServletException, IOException {

        doPost(request, response);
    }

    public   void   doPost(HttpServletRequest   request,   HttpServletResponse
response)
            throws ServletException, IOException {
        response.setHeader("Cache-Control", "no-cache");
        response.setCharacterEncoding("UTF-8");
        response.setContentType("text/xml");

        String prefix = request.getParameter("pre");
        PrintWriter out = response.getWriter();
        out.println("<response>");
        System.out.println("pre=  " + prefix); //

        for (int i = 0; i < info.length; i++) {
            if (info[i].toLowerCase().indexOf(prefix.toLowerCase()) == 0) {
                out.println("<name>" + info[i] + "</name>");
                System.out.println("<name>" + info[i] + "</name>");//
            }
        }
        out.println("</response>");
        out.flush();
        out.close();
    }
   }
```

3.配置 web.xml，代码如下所示：

```
 <servlet-mapping>
   <servlet-name>Q007Servlet</servlet-name>
   <url-pattern>/servlet/Q007Servlet</url-pattern>
 </servlet-mapping>
 <servlet-mapping>
   <servlet-name>Q007Servlet</servlet-name>
  <url-pattern>/servlet/Q007Servlet</url-pattern>
```

第四部分

1. Spring

1.1.描述 Spring 框架的作用和优点?

参考答案：

Spring 框架的作用和优点如下：

1. Spring 是一个开源的轻量级的应用开发框架，其目的是用于简化企业级应用程序开发，减少侵入；

2. Spring 提供的 IoC 和 AOP 应用，可以将组件的耦合度降至最低，即解耦，便于系统日后的维护和升级；

3. Spring 为系统提供了一个整体的解决方案，开发者可以利用它本身提供的功能外，也可以与第三方框架和技术整合应用，可以自由选择采用哪种技术进行开发。

1.2.如何控制 Bean 对象的作用域，默认作用域是什么？

参考答案：

1. 可以通过<bean>定义的 scope 属性指定 Bean 对象的作用域或者使用注解@Scope 指定 Bean 对象的作用域。

2. 默认 Bean 对象的作用域为 singleton。

1.3.描述下列注解标记的作用？

@Component、@Repository 、@Service、@Scope、@Autowired、@Inject、@Value

参考答案：

1. @Component 为通用注解。
2. @Repository 为持久层组件注解。
3. @Service 为业务层组件注解。
4. @Scope 为 Bean 的作用域注解。
5. @Autowired、@Inject 为指定 Bean 之间依赖关系的注解。
6. @Value 为注入 Spring 表达式值的注解。

1.4.描述 Spring Web MVC 的工作流程？

参考答案：

Spring Web MVC 的工作流程如下：

1. 浏览器发出 spring mvc 请求，请求交给前端控制器 DispatcherServlet 处理。

2. 控制器通过 HandlerMapping 维护的请求和 Controller 映射信息，找到相应的 Controller 组件处理请求。

3. 执行 Controller 组件约定方法处理请求，在约定方法中可以调用 Service 和 DAO 等组件完成数据库操作。约定方法可以返回一个 ModelAndView 对象，封装了模型数据和视图名称信息。

4. 控制器接收 ModelAndView 之后，调用 ViewResolver 组件，定位 View 的 JSP 并传递 Model 信息，生成响应界面结果。

1.5.Spring 有什么缺点？

参考答案：

Spring 有什么缺点如下：

1. jsp 中要写很多代码；
2. 控制器过于灵活，缺少一个公用控制器；
3. 不支持分布式部署。

1.6.Spring 中的 IOC 和 AOP 是什么含义，它们在项目中起到什么作用，并举例说明？

参考答案：

IOC：控制反转，是一种设计模式。一层含义是控制权的转移：由传统的在程序中控制依赖转移到由容器来控制；第二层是依赖注入：将相互依赖的对象分离，在 Spring 配置文件中描述他们的依赖关系。他们的依赖关系只在使用的时候才建立。

AOP：面向切面，是一种编程思想，OOP 的延续。将系统中非核心的业务提取出来，进行单独处理。

Spring 的 AOP 和 IOC 在项目中都是为了解决系统代码耦合度过高的问题。使代码重用度高、易于维护。比如事务、日志和安全等。

1.7.简述 Spring 事务有几种管理方法，写出一种配置方式？

参考答案：

Spring 事务有两种方式：

1. 编程式事务：(代码中嵌入)
2. 声明式事务：(注解，XML)

注解方式配置事务的方式如下：

首先，需要在 applicationContext.xml 中添加启用配置，代码如下所示：

```
<!-- 定义事务管理器 -->
 <bean id="txManager"
class="org.springframework.orm.hibernate3.HibernateTransactionManager">
    <property name="sessionFactory" ref="sessionFactory" />
 </bean>
<tx:annotation=driven transaction-manager="txManager"/>
```

然后，使用@Transactional 注解，代码如下所示：

```
@Transactional
public class DefaultFooService implements FooService {
  // @Transactional
  public void insertFoo(Foo foo){...}
  public void updateFoo(Foo foo){...}
}
```

@Transactional 注解标记可以用在类定义和方法定义前，方法的事务设置将优先于类级别注解的事务设置。

1.8.简单解释一下事务拦截器的实现原理？

参考答案：

spring 中的事务管理是通过 AOP 代理来实现的，对被代理对象的每个方法进行拦截，在方法执行前启动事务，方法执行完后根据是否有异常和异常的种类进行提交或回滚。

1.9.请说明 Spring 事务管理中的传播行为和隔离等级？

参考答案：

Spring 事务的传播行为如下：

1. PROPAGATION_REQUIRED -- 支持当前事务，如果当前没有事务，就新建一个事务。这是最常见的选择。

2. PROPAGATION_SUPPORTS -- 支持当前事务，如果当前没有事务，就以非事务方式执行。

3. PROPAGATION_MANDATORY -- 支持当前事务，如果当前没有事务，就抛出异常。

4. PROPAGATION_REQUIRES_NEW -- 新建事务，如果当前存在事务，把当前事务挂起。

5. PROPAGATION_NOT_SUPPORTED--以非事务方式执行操作，如果当前存在事务，就把当前事务挂起。

6. PROPAGATION_NEVER -- 以非事务方式执行，如果当前存在事务，则抛出异常。

7. PROPAGATION_NESTED -- 如果当前存在事务，则在嵌套事务内执行。如果当前没有事务，则进行与 PROPAGATION_REQUIRED 类似的操作。

Spring 事务的隔离级别如下：

数据库系统提供了四种事务隔离级别供用户选择。不同的隔离级别采用不同的锁类型来实现，在四种隔离级别中，Serializable 的隔离级别最高，ReadUncommited 的隔离级别最低。大多数据库默认的隔离级别为 ReadCommited，如 SqlServer，当然也有少部分数据库默认的隔离级别为 RepeatableRead，如 Mysql。

1. ReadUncommited：读未提交数据(会出现脏读,不可重复读和幻读)。

2. ReadCommited：读已提交数据(会出现不可重复读和幻读)

3. RepeatableRead：可重复读(会出现幻读)

4. Serializable：串行化

脏读：一个事务读取到另一事务未提交的更新新据。

不可重复读：在同一事务中，多次读取同一数据返回的结果有所不同。换句话说就是，后续读取可以读到另一事务已提交的更新数据。相反，“可重复读”在同一事务中多次读取数据时，能够保证所读数据一样，也就是，后续读取不能读到另一事务已提交的更新数据。

幻读：一个事务读取到另一事务已提交的 insert 数据。

1.10.Spring 的 Bean 有哪些作用域？

参考答案：

Spring 的 Bean 有以下五种作用域：

1. singleton：SpringIoc 容器只会创建该 Bean 的唯一实例；

2. prototype: 每次请求都创建一个实例；

3. request: 每次 HTTP 请求都会产生一个新的 bean。需要注意的是，该作用域仅在基于 Web 的 Spring ApplicationContext 情形下有效，以下的 session 和 global Session 也是如此；

4. session：每次会话创建一个实例；

5. global session：全局的 HttpSession 中，容器会返回该 bean 的同一个实例。

1.11.Spring 提倡面向接口编程,请讲一下你对它的理解,它有什么好处？

参考答案：

在一个面向对象的系统中，系统的各种功能是由许许多多的不同对象协作完成的。在这种情况下，各个对象内部是如何实现自己的，对系统设计人员来讲就不那么重要了；而各个对象之间的协作关系则成为系统设计的关键。小到不同类之间的通信，大到各模块之间的交互，在系统设计之初都是要着重考虑的，这也是系统设计的主要工作内容。面向接口编程就是指按照这种思想来编程。

1.12.Spring 和 web 应用整合？

参考答案：

在 Web 应用程序中，要对 Spring 的 IoC 容器（WebApplicationContext）进行初始

化，可以通过配置 ContextLoadListener 监听器实现。具体配置如下：

1. 在 web.xml 中通过应用上下文初始化参数来指定 Spring 的配置文件的路径，配置代码如下：

```
<context-param>
  <param-name>contextConfigLocation</param-name>
  <param-value>classpath:applicationContext-*.xml</param-value>
</context-param>
```

2. 在 web.xml 中配置 ContextLoaderListener 监听器，代码如下所示：

```
<listener>
<listener-class>
org.springframework.web.context.ContextLoaderListener
</listener-class>
</listener>
```

经过以上配置后，在 Servlet 或 JSP 中，直接使用 Spring 提供的 WebApplicationContextUtils 工具类可以获取 Spring 的 WebApplicationContext 容器，然后就可以从该容器中获取你想要的 Bean 了。

1.13.说说你对用 SSH 框架进行开发的理解？

参考答案：

SSH 框架指的是 Struts、Spring、Hibernate。其中 ,Struts 主要用于流程控制 ;Spring 的控制反转能起到解耦合的作用；Hibernate 主要用于数据持久化。

1.14.三大框架中高内聚、低耦合是哪个框架实现的？怎么实现的？

参考答案：

低耦合是通过 spring 框架的 IOC 和 AOP 实现的。

将基于实现类的耦合变成基于接口的耦合，可以避免硬编码所造成的过度程序耦合，而接下来需要解决的问题就是，如何确定该接口的实现类。IoC 控制反转，就是将某一接口的具体实现类的控制从调用类中移除，转交给第三方，即 Spring 容器。

在业务系统里除了要实现业务功能之外，还要实现如权限拦截、性能监控、事务管理等非业务功能。通常的作法是非业务的代码穿插在业务代码中，从而导致了业务组件与非业务组件的耦合。AOP 面向切面编程，就是将这些分散在各个业务逻辑代码中的非业务代码，通过横向切割的方式抽取到一个独立的模块中，从而实现业务组件与非业务组件的解耦。

2. MyBatis

2.1.什么是 MyBatis？

参考答案：

MyBatis 最早源自 Apache 基金会的一个开源项目 iBatis， 2010 年这个项目由 Apache software foundation 迁移到了 google code，并且改名为 MyBatis；MyBatis 是支持普通 SQL 查询，存储过程和高级映射的优秀持久层框架。MyBatis 封装了几乎所有的 JDBC 代码和参数的手工设置以及结果集的检索；MyBatis 使用简单的 XML 或注解做配置和定义映射关系，将 Java 的 POJOs（Plain Old Java Objects）映射成数据库中的记录。

2.2.简述 MyBatis 的体系结构？

参考答案：

MyBatis 体系结构主要由以下几个关键部分:

1. 加载配置

配置有两种形式，一种是 XML 配置文件，另一种是 Java 代码的注解。MyBatis 将 SQL 的配置信息加载成为一个个的 MappedStatement 对象（包括了传入参数映射配置、执行的 SQL 语句、结果映射配置），并将其存储在内存中

2. SQL 解析

当 API 接口层接收到调用请求时，会接收到传入 SQL 的 ID 和传入对象(可以是 Map、JavaBean 或者基本数据类型），Mybatis 会根据 SQL 的 ID 找到对应的 MappedStatement，然后根据传入参数对象对 MappedStatement 进行解析，解析后可以得到最终要执行的 SQL 语句和参数。

3. SQL 执行

将最终得到的 SQL 和参数拿到数据库进行执行，得到操作数据库的结果。

4. 结果映射

将操作数据库的结果按照映射的配置进行转换，可以转换成 HashMap、JavaBean 或者基本数据类型，并将最终结果返回。

2.3.列举 MyBatis 的常用 API 及方法？

参考答案：

在使用 MyBatis 框架时，主要涉及以下几个 API:

1.SqlSessionFactoryBuilder 该对象负责根据 MyBatis 配置文件 SqlMapConfig.xml 构建 SqlSessionFactory 实例。

2.SqlSessionFactory 每一个 MyBatis 的应用程序都以一个 SqlSessionFactory 对象为核心。该对象负责创建 SqlSession 对象实例。

3.SqlSession 该对象包含了所有执行 SQL 操作的方法，用于执行已映射的 SQL 语句。

2.4.对于 Hibernate 和 MyBatis 的区别与利弊，谈谈你的看法？

参考答案：

hibernate 与 MyBatis 的对比：

1. MyBatis 非常简单易学，Hibernate 相对较复杂，门槛较高；

2. 二者都是比较优秀的开源产品；

3. 当系统属于二次开发，无法对数据库结构做到控制和修改，那 MyBatis 的灵活性将比 hibernate 更适合；

4. 系统数据处理量巨大，性能要求极为苛刻，这往往意味着我们必须通过经过高度优化的 sql 语句（或存储过程）才能达到系统性能设计指标。在这种情况下 MyBatis 会有更好的可控性和表现；

5. MyBatis 需要手写 sql 语句 ,也可以生成一部分 ,Hibernate 则基本上可以自动生成，偶尔会写一些 hql。同样的需求，MyBatis 的工作量比 Hibernate 要大很多。类似的，如果涉及到数据库字段的修改 ,Hibernate 修改的地方很少 ,而 MyBatis 要把那些 sql mapping 的地方一一修改；

6. MyBatis 以数据库字段一一对应映射得到的 po 和 Hibernte 这种对象化映射得到的 po 是截然不同的，本质区别在于这种 po 是扁平化的，不像 hibernate 映射的 po 是可以表达立体的对象继承，聚合等等关系的，这将会直接影响到你的整个软件系统的设计思路；

7. Hibernate 现在已经是主流 OR Mapping 框架，从文档的丰富性，产品的完善性，版本的开发速度都要强于 MyBatis。

3. Struts

3.1.为什么要用 struts2？

参考答案：

主流的开发技术，大多数公司在使用，Struts 是基于 MVC 模式开发的，MVC 结构是一个优秀的设计思想，可以提高程序结构的灵活性，便于日后的维护和扩展。

3.1. 简述 Struts 的发展历史？

参考答案：

最早出现的 Struts1 是一个非常著名的框架，它实现了 MVC 模式。Struts1 简单小巧，其中最成熟的版本是 Struts1.2。

之后出现了 WebWork 框架，其实现技术比 Struts1 先进，但影响力不如 Struts1。

在框架技术不断发展过程中，有人在 WebWork 核心 XWork 的基础上包装了 Struts1（算是两种框架的整合），由此，结合了 Struts1 的影响力和 WebWork 的先进技术，Struts 2 诞生了。

所以说，Struts2 不是 Struts1 的升级，它更像是 WebWork 的升级版本。

3.2. 请简述 Struts2 与 Struts1 的区别和联系？

参考答案：

Struts2 与 Struts1 差别巨大，不能理解为 Struts1 的升级版；Struts2 以 Xwork 为核心，可以理解为 WebWork 的升级版。

3.3. Struts2 如何实现 MVC，与 Spring MVC 有什么不同？

参考答案：

Struts2 采用 filter 充当前端控制器来处理请求，filter 会根据 struts.xml 的配置，将请求分发给不同的业务控制器 Action，再由 Action 处理具体的业务逻辑。Action 处理完业务之后，filter 会根据其返回的字符串，从 struts.xml 中找到对应的 result，最终由 result 将请求转发给页面。

这个实现的思路与 Spring MVC 基本一致。其差异如下：

1） Spring 采用 Servlet 充当前端控制器，分发请求。

2） Spring 采用 RequestMapping 配置请求与业务控制器的关系。

3） Spring 采用 Controller 充当业务控制器。

4） Spring 采用 ViewResolver 将请求转发给页面。

3.4. 在 Struts2 中页面如何向 Action 传参？

参考答案：

Struts2 中常用的传参方式有 2 种，分别是基本属性注入和域模型注入。

其中，基本属性注入是将表单中的数据，分别传给 Action 中声明的基本属性，要求这些属性有 set 方法，并且名称与表单中框体的 name 值一致。

域模型注入是将表单中的数据，传给 Action 中的一个实体对象，要求这个对象具有 set 方法，并且表单中框体的 name 中的表达式要有如下格式“对象.属性”。

3.5. 什么是 OGNL？

参考答案：

Object Graphics Navigation Language，对象图导航语言，属于表达式语言的一种，与 EL 表达式类似；Ognl 技术是 struts2 中的核心知识，它封装于 ognl.jar 中；Ognl.jar 工具包提供一个引擎，该引擎可以按照提供的 ognl 表达式访问对象数据和方法。

3.6. OGNL 工具的构成？

参考答案：

OGNL 工具由三部分构成：

1. OGNL 引擎，负责解析执行 OGNL 表达式

2. Root 存储区，负责存储一个 Object 类型的对象，该存储区数据访问时，OGNL 表达式格式为“属性”

3. Context 存储区，负责存储一个 Map 类型的对象，该存储区数据访问时，OGNL 表示式格式为“#key”

3.7. OGNL 表达式有哪些用法，你熟悉其中哪几种？

参考答案：

OGNL 有 8 种用法，其中常用的有 2 种，不太常用的有 6 种。

我熟悉这 2 种常用的用法，其作用是：

1） 可以给基本属性注入值

2） 可以给实体对象注入值

另外 6 种用法也有所了解，作用是：

1） 可以访问数组或集合

2） 可以访问 Map

3） 可以在访问时进行一些基本的运算

4） 可以在访问时调用返回对象的方法

5） 可以直接创建一个临时的集合

6） 可以直接创建一个临时的 Map

3.8. 请简述 struts2 的 Action 的工作原理？

参考答案：

struts2 的 Action 的工作原理如下：

1. 当客户端发出请求，请求到达控制器；

2. 控制器根据请求创建一个 ValueStack 对象，每个请求创建一个 Action 对象，Action 对象存入到 ValueStack 对象的 root 栈顶。将 ValueStack 对象存入到 request 中。存储的 key 为”struts.valueStack”；

3. 控制器调用 Action 对象接收请求参数，执行业务方法处理；

4. 控制器根据 Action 返回值调用 result 视图组件处理；

5. 请求处理完成后，将 ValueStack 对象和 Action 对象销毁。

3.9. 请列出常用的至少五种 struts2 的 Result 组件，并说明它们的作用？

参考答案：

1. dispatcher (默认) 以请求转发方式调用一个 JSP，生成响应视图；

2. redirect 以重定向方式调用一个 JSP，生成响应视图；

3. redirectAction 以重定向方式调用一个 action；

4. chain 以请求转发方式调用一个 action；

5. stream 以字节流方式响应，将 Action 中指定的一个 InputStream 类型属性输出. 将 Action 中的 InputStream 属性以字节流方式输出；

6. json 以 json 字符串方式响应，将 Action 中指定的属性,拼成一个 json 字符串输出。

3.10.请简述 Struts2 中各组件的作用及调用顺序？

参考答案：

Struts2 有 6 大核心组件，分别为前端控制器 filter、业务控制器 Action、值对象 ValueStack、拦截器 Interceptor、输出组件 Result、Struts2 标签。

它们的调用顺序是，请求提交给前端控制器 filter，它会根据 struts.xml 中的配置找到对应的业务控制器 Action，然后实例化值对象 ValueStack 并实例化 Action 放于 ValueStack 的栈顶，再调用 Action 的业务方法，在调用过程中会被拦截器组件所拦截。最终根据 Action 方法的返回值，filter 从 struts.xml 中可以找到对应的 Result，于是使用这个 Result 向页面输出内容。

当然，最常见的输出实际上是将请求转发给一个 JSP。那么在容器解析 JSP 生成 HTML 的过程中， Struts2 标签将被解析。此时标签中的 OGNL 表达式会发送给 ValueStack 取值，返回的结果被写入最终生成的 HTML。

3.11.简述拦截器的作用？

参考答案：

拦截器适合封装一些共通处理,便于重复利用.例如请求参数给 Action 属性,日志的记录,权限检查，事务处理等.拦截器是通过配置方式调用,因此使用方法比较灵活,便于维护和扩展。

3.12.Struts2 中有哪些 UI 标签，请简述其作用？

参考答案：

Struts2 中有如下 UI 标签：

1. 表单标签，可以生成 form 元素。

2. 文本框、密码框、文本域标签，这些标签很相似，都是生成一个框体，然后在框体中显示出默认的内容，差异仅仅是长相不同。

3. 布尔框，可以生成一个 checkbox，要求用户进行确认选择。这个标签可以做默认的勾选，但是要求 OGNL 表达式访问的值是布尔类型的。

4. 单选框标签，可以生成一组单选框，并根据 OGNL 取值自动勾选上一个单选框。

5. 复选框标签，可以生成一组复选框，并根据 OGNL 取值自动勾选上几个复选框。

6. 下拉列表标签，可以生成一个下拉列表，包含一组 option，并根据 OGNL 取值自动勾选上一个 option。

注意，上述单选框、复选框、下拉列表标签，在使用上都有 2 种初始化选项的方式，一种是静态的方式，可以直接在标签上写出固定的范围，并根据此范围初始化选项。也可以在标签上通过 OGNL 表达式访问 Action 中的一个集合，并根据此集合来初始化选项。

3.13.Struts2 中的拦截器有什么用，与 Spring 中的 AOP 有什么区别和联系？

参考答案：

Struts2 中的拦截器可以批量扩展 Action，处理一组 Action 的通用业务逻辑。

Struts2 中的拦截器实际上就是采用 AOP 思想实现的，只是它只能处理 Action 的通用逻辑，无法处理其他组件的通用逻辑。而 Spring 的 AOP 实现更为灵活，可以实现任意一批组件的通用业务逻辑。

3.14.Action 默认引用哪个拦截器，如果没有默认引用，会导致什么问题？

参考答案：

Action 默认引用一个拦截器栈，叫做 defaultStack。这个默认引用不能丢掉，因为里面包含了 Struts2 框架所必须依赖的一些拦截器，如果丢掉会导致项目报错。因此我们在开发时，如果自己引用了自定义拦截器，要注意不丢掉默认的拦截器栈。

3.15.struts2 写 Action 的时候，需不需要继承什么类？

参考答案：

可以继承自 ActionSupport 类，也可以不继承。

3.16.Struts Action 是不是线程安全？如果不是，有什么方式可以保证 Action 的线程安全？如果是，请说明原因（Struts1 和 Struts2）？

参考答案：

Struts 1 的 Action 不是线程安全的。Struts 1 在第一次请求某个 Action 时，会创建这个 Action 实例。之后再请求该 Action 实例时，就用之前创建好的这个 Action 处理，即它是单例模式。在 Struts 1 的方法调用模式用到的参数一般都是局部变量（包括 request、response 等），局部变量是线程安全的，因此不用考虑线程安全问题。如果在 Action 类中使用了实例变量，就会存在线程安全问题。所以我们用 Struts 1 开发时尽量不要使用实例变量。

Struts2 Action 对象为每一个请求产生一个实例，因此没有线程安全问题。不过在 Spring + Struts2 的应用中，由 Spring 来管理 Struts2 的 Action，而 IoC 容器管理的 bean 默认是单实例的（scope="singleton"），加上 Struts 2 的 Action 就像一个 POJO 一样，定义了很多的类变量，这就有线程安全问题了。解决此问题最简单的办则是不使用单例模式（设置 scope="prototype"），配置如下：

```
<bean id="testAction" class="com.webapp.action.TestAction"
 scope="prototype">
```

3.17.struts2 中的拦截器和 servlet 中的过滤器有什么区别？

参考答案：

struts2 中的拦截器和 servlet 中的过滤器的区别如下：

1. 拦截器是基于 Java 反射机制的，而过滤器是基于函数回调的；
2. 过滤器依赖于 servlet 容器，而拦截器不依赖于 servlet 容器；
3. 拦截器只能对 Action 请求起作用，而过滤器则可以对几乎所有请求起作用；
4. 拦截器可以访问 Action 上下文、值栈里的对象，而过滤器不能；
5. 在 Action 的生命周期中，拦截器可以多次调用，而过滤器只能在容器初始化时被调用一次。

4. Hibernate

4.1. hibernate 有什么缺点？

参考答案：

1. HQL 最终还是要转换为 JDBC，效率降低；
2. 使用数据库特性的语句，将很难调优。Hibernate 对 JDBC 封装的过于厉害，所以就失去了对 SQL 的控制；
3. 不适合系统中复杂的关联查询，包括多表、复杂查询、大量数据的查询处理都不好；
4. 数据量大时，缓存机制不好使，效率明显降低；
5. 对于大批量数据的修改、删除，不适合用 Hibernate，这也是 ORM 框架的弱点；
6. 对象化限制我们的查询。例如，一个持久性类不能映射到多个表；
7. Hibernate 带来方便的同时，也使得程序错误排查变得非常困难；
8. Hibernate 是完善的 ORM 框架，要想 Hibernate 工作好，数据库设计必须好。数据库结构的变更，需要修改 hbm 和 bean，Hibernate 自适应能力为 0；
9. setter 方法的参数需要自己转换类型。

4.2. 如何优化 Hibernate？

参考答案：

1. 使用双向一对多关联，不使用单向一对多；
2. 灵活使用单向一对多关联；
3. 不用一对一，用多对一取代；
4. 配置对象缓存，不使用集合缓存；
5. 一对多集合使用 Bag,多对多集合使用 Set；
6. 继承类使用显式多态；
7. 表字段要少，表关联不要怕多，有二级缓存撑腰。

4.3. Hibernate 的原理及为什么要用它？

参考答案：

Hibernate 的原理如下：

1. 读取并解析配置文件；
2. 读取并解析映射信息，创建 SessionFactory；
3. 打开 Sesssion；
4. 创建事务 Transation；
5. 持久化操作；
6. 提交事务；

7. 关闭 Session；

为什么要用 Hibernate：

1. 对 JDBC 访问数据库的代码做了封装，大大简化了数据访问层繁琐的重复性代码。

2. Hibernate 是一个基于 JDBC 的主流持久化框架，是一个优秀的 ORM 实现。他很大程度的简化 DAO 层的编码工作

3. hibernate 使用 Java 反射机制，而不是字节码增强程序来实现透明性。

4. hibernate 的性能非常好，因为它是个轻量级框架。映射的灵活性很出色。它支持各种关系数据库，从一对一到多对多的各种复杂关系。

4.4. Hibernate 是如何延迟加载？

参考答案：

1. Hibernate2 延迟加载实现：a)实体对象 b)集合（Collection）；

2. Hibernate3 提供了属性的延迟加载功能；

当 Hibernate 在查询数据的时候，数据并没有存放在内存中，当程序真正对数据操作时，对象才存放在内存中，就实现了延迟加载，他节省了服务器的内存开销，从而提高了服务器的性能。

4.5. Hibernate 的核心配置文件是什么及其作用？

参考答案：

Hibernate 的核心配置文件是 hibernate.cfg.xml，主要用于配置数据库连接和 Hibernate 运行时所需的各种参数。

4.6. 说出几个与 Spring 同类型的开源框架，说出几个与 Hibernate 同类型的开源框架,说出几个与 Struts 同类型的开源框架？

参考答案：

1. 与 Spring 同类型的开源框架： EJB3.0、picoContainer；

2. 与 Hibernate 同类型的开源框架：MyBatis，JDO，JPA；

3. 几个与 struts 同类型的开源框架：webwork，tapestry，JSF；

4.7. 简述 Hibernate 与 JDBC 相比的优缺点？

参考答案：

Hibernate 与 JDBC 相比较的主要优点为：

1. 面向对象

Hibernate 可以让开发人员以面向对象的思想来操作数据库。JDBC 只能通过 SQL 语句将元数据传送给数据库，进行数据操作。而 Hibernate 可以在底层对元数据和对象进行

转化，使得开发者只用面向对象的方式来存取数据即可。

2. 移植性

Hibernate 使用 XML 或 JPA 的配置以及数据库方言等等的机制，使得 Hibernate 具有更好的移植性，对于不同的数据库，开发者只需要使用相同的数据操作即可，无需关心数据库之间的差异。而直接使用 JDBC 就不得不考虑数据库差异的问题。

3. 封装性

Hibernate 提供了大量的封装（这也是它最大的缺点），很多数据操作以及关联关系等都被封装的很好，开发者不需写大量的 sql 语句，这就极大的提高了开发者的开发效率。

4. 缓存机制

Hibernate 提供了缓存机制（session 缓存，二级缓存，查询缓存），对于那些改动不大且经常使用的数据，可以将它们放到缓存中，不必在每次使用时都去查询数据库，缓存机制对提升性能大有裨益。

Hibernate 与 JDBC 相比较的主要缺点为：

1. 对 Hibernate 而言，它对 JDBC 封装过于厉害，所以就失去了对 SQL 的控制（当然 hibernate 也可以使用 native sql 即使用 createSQLQuery 等方法来调用与数据库相关的 sql，但这样一来也就影响了 Hibernate 的可移植性），使得 Hibernate 在很多地方不够灵活，难于优化，尤其对于一些复杂的关联查询时，Hibernate 提供的功能远不及直接使用 JDBC 方便性能更高。

2. Hibernate 没有提供专门的批处理机制，如果要批量更新或插入数据时，还需要显示的 flush，clear 之类的操作，性能不如 JDBC。

3. 相对于 JDBC，Hibernate 更消耗内存，因为它每次的数据库操作都要做数据和对象的转换/封装，查询出一条数据就要创建一个或多个对象，这样太消耗内存了。

4. Hibernate 提供了很多好处，但这些好处本身就是陷阱（如 proxy 陷阱等），开发者如果不注意就会调入陷阱而不知，这样就可能会出现一些无法排查的异常情况，比如程序表面上看着毫无错误，可就是达不到预期的效果，而且并无异常抛出，断点排查也不一定能找到症结所在，这将是非常令人抓狂的一件事！

4.8. 请例举 Hibernate 的主键生成方式？

参考答案：

1. sequence 是采用序列方式生成主键，适用于 Oracle 数据库。

2. identity 是采用数据库自增长机制生成主键，适用于 Oracle 之外的其他数据库。

3. native 是根据当前配置的数据库方言，自动选择 sequence 或者 identity。

4. increment 不是采用数据库自身的机制来生成主键，而是 Hibernate 提供的一种生成主键的方式。它会获取当前表中主键的最大值，然后加 1 作为新的主键。这种方式在并发量高时存在问题，可能会生成重复的主键，因此不推荐使用。

5. assigned 是 Hibernate 不负责生成主键，需要程序员自己处理主键的生成。

6. uuid/hilo 是采用 uuid 或 hilo 算法生成一个主键值，这个主键值是一个不规则的长

数字。这种方式生成的主键可以保证不重复，但是没有规律，因此不能按主键排序。

4.9. hibernate 中 inverse 属性及其含义？

参考答案：

inverse 是指的关联关系的控制方向 ,inverse=false 的 side(side 其实是指 inverse =false 所位于的 class 元素）端有责任维护关系，而 inverse = true 端无须维护这些关系。

4.10.对 hibernate 的延迟加载如何理解，在实际应用中，延迟加载与 session 关闭的矛盾是如何处理的？

参考答案：

延迟加载是指在使用某些 Hibernate 方法查询数据时，Hibernate 返回的只是一个空对象(除 id 外属性都为 null)，并没有真正查询数据库。而在使用这个对象时才会触发查询数据库，并将查询到的数据注入到这个空对象中。这种将查询时机推迟到对象访问时的机制称之为延迟加载。

采用具有延迟加载机制的操作，需要避免 session 提前关闭，避免在使用对象之前关闭 session。可以采用以下 2 种方案解决此问题：

1） 采用非延迟加载的查询方法，如 query.get()、session.list()等。

2） 在使用对象之后再关闭 session。

在项目中，DAO 只是负责查询出数据，而使用数据的时机是在 JSP 解析的过程中，因此要避免在 DAO 中关闭 session，或者说要在视图层保持 session 的开启。项目中解决这个问题的手段称之为 Open session in view，即在视图层保持 session 的开启。在不同的技术框架下，实现 Open session in view 的手段不同：

1）在 Servlet 中使用过滤器实现。

2) 在 Struts2 中使用拦截器实现。

3) 在 Spring 中使用 AOP 实现。

4.11.简要描述对对象关系映射（ object-relational mapping,简称 ORM）的理解。并说明经典实现框架？

参考答案：

对象关系映射（Object—Relational Mapping，简称 ORM）是一种为了解决面向对象与面向关系数据库存在的互不匹配的现象的技术；简单的说，ORM 是通过使用描述对象和数据库之间映射的元数据，将 java 程序中的对象自动持久化到关系数据库中；本质上就是将数据从一种形式转换到另外一种形式。

经典的 ORM 实现框架有 Hibernate 和 Mybatis。

4.12.Hibernate 有哪 5 个核心接口？

参考答案：

Hibernate 有以下五个核心接口：

1. Configuration 负责加载主配置文件信息，同时也加载映射关系文件信息。
2. SessionFactory 负责创建 Session 对象。
3. Session 数据库连接会话，负责执行增删改操作。
4. Transaction 负责事务控制。
5. Query 负责执行特殊查询。

4.13.关于 hibernate:

1. 在 Hibernate 中，在配置文件呈现一对多，多对多的标签是什么？
2. Hibernate 是如何处理事务的？

参考答案：

1. 一对多的标签为<one-to-many> ；多对多的标签为<many-to-many>；

2. Hibernate 的事务实际上是底层的 JDBC Transaction 的封装或者是 JTATransaction 的封装；默认情况下使用 JDBCTransaction。

4.14.简单说说 session.load()和 session.get()的区别，哪种方法可以使用二级缓存？

参考答案：

session.load()和 session.get()的区别为：

1. get 不支持延迟加载，而 load 支持。换句话说，get 方法一定获取实际的对象，而 load 有可能返回代理对象。

2. 如果未能发现符合条件的记录， get 方法返回 null，而 load 方法会抛出一个 ObjectNotFoundException。

get 方法和 load 方法都可以使用二级缓存。

4.15.简述 update 和 saveOrUpdate 方法的区别？

参考答案：

update 方法更新数据时，如果不存在该条数据的主键则会报错； saveOrUpdate 方法保存或更新，如果不存在主键则执行插入。

4.16.JDBC，Hibernate 分页怎样实现？

参考答案：

1. Hibernate 实现分页的方式如下：

```
Query query = session.createQuery("from Student");
query.setFirstResult(firstResult);//设置每页开始的记录号
query.setMaxResults(resultNumber);//设置每页显示的记录数
Collection students = query.list();
```

上述代码中的集合 students 中为某页要获取的数据。

2. JDBC 实现分页的方式如下：

根据不同的数据库采用不同的 sql 分页语句，例如: Oracle 中的 sql 语句为:

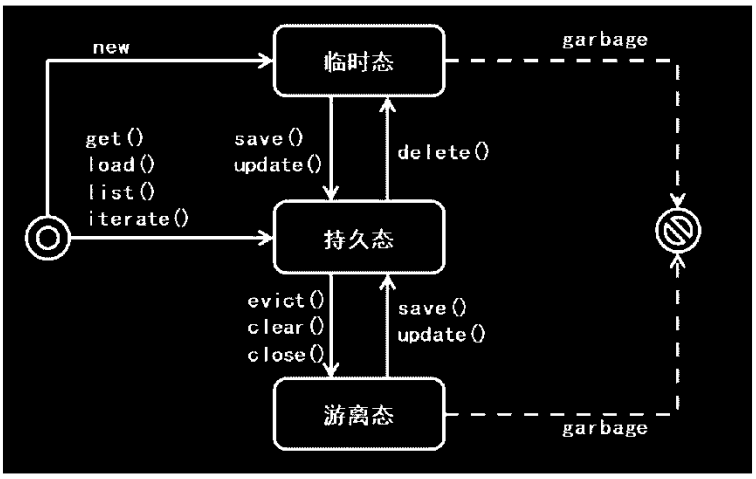
```
SELECT * FROM (SELECT a.*, rownum r FROM TB_STUDENT) WHERE r between 2 and 10;
```

上述查询语句，查询了从记录号 2 到记录号 10 之间的所有记录。将上述 SQL 语句使用 JDBC 进行执行，即可得到分页后数据。

4.17.Hibernate 中对象有哪几种状态，有什么规则？

参考答案：

Hibernate 中对象可以看做有 3 种状态，分别是临时态、持久态、游离态，这些状态可以相互转换，转换规则如下图：



具体各个状态的特征为:

1. 临时态

1) 临时态的对象可以被垃圾回收；

2) 临时态的对象未进行过持久化，未与 session 关联。

2. 持久态

1) 持久态对象垃圾回收器不能回收；

2 ） 持久态的对象进行了持久化，与 session 相关联，即持久态对象存在于 session 缓存中，由 session 负责管理。

3)持久态对象的数据可以自动更新到数据库中，时机是在调用 session.flush()时执行。而提交事务时会调用 session.flush()，因此提交事务时也会触发同步，可以理解为

ts.commit=session.flush()+commit。

3. 游离态

1）游离态的对象可以被垃圾回收；

2）游离态的对象进行过持久化，但已与 session 解除了关联。

4.18.Hibernate 中哪些查询方法具有延迟加载机制

参考答案：

1. session.load()；
2. query.iterate()；
3. 关联映射中对关联属性的加载。

4.19.Hibernate 中有几种类型的关联映射？

参考答案：

Hibernate 中的关联映射有如下几种：

1）一对一关联

2） 一对多关联

3） 多对一关联

3） 多对多关联

4.20.Hibernate 有哪几种查询方式？

参考答案：

Hibernate 有如下几种查询方式

1） 使用 API 查询，如 get、load

2） 使用 HQL 查询

3） 使用 SQL 查询

4） 使用 Criteria 查询

4.21.请简述 Hibernate 一级缓存和二级缓存的区别和联系？

参考答案：

一级缓存是 Session 级别的缓存，由 Session 负责管理，因此一级缓存是 Session 独享的，即每个 Session 只能访问自己的一级缓存区。

二级缓存是 SessionFactory 级别的缓存，由 SessionFactory 负责管理，因此二级缓存是 Session 间共享的，即不同的 Session 都可以访问二级缓存区。

一级缓存和二级缓存相同的地方是，他们缓存的都是对象数据。