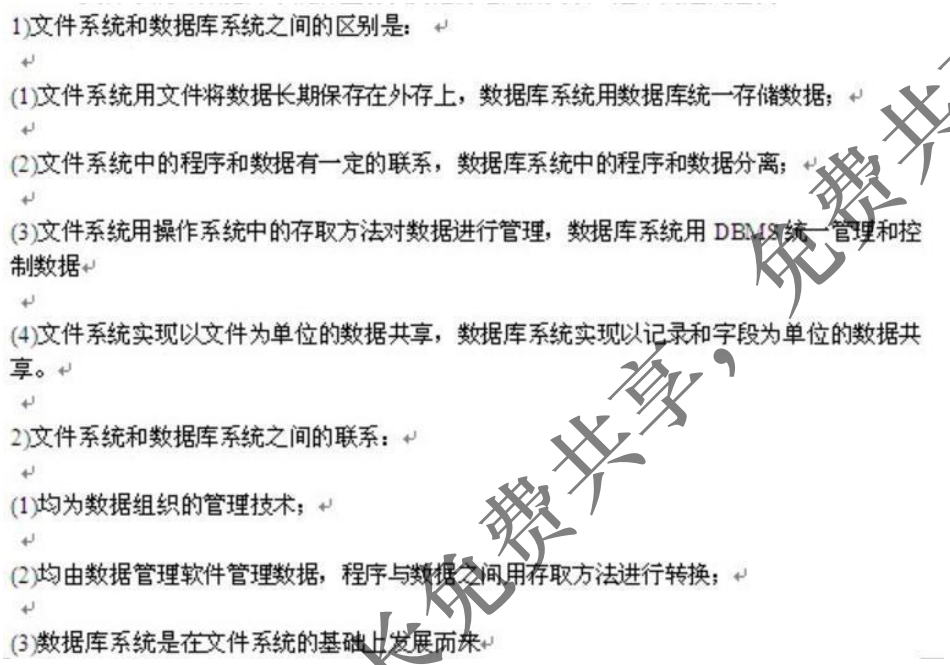
数据库

1. 数据库系统和文件系统相比有哪些优势

   1) 提高了数据的共享性，使多个用户能够同时访问数据库中的数据。

   2) 提高了数据的一致性和完整性。

   3) 提供数据与应用程序的独立性。



1)文件系统和数据库系统之间的区别是：

(1)文件系统用文件将数据长期保存在外存上，数据库系统用数据库统一存储数据；

(2)文件系统中的程序和数据有一定的联系，数据库系统中的程序和数据分离；

(3)文件系统用操作系统中的存取方法对数据进行管理，数据库系统用 DBMS 统一管理和控制数据

(4)文件系统实现以文件为单位的数据共享，数据库系统实现以记录和字段为单位的数据共享。

2)文件系统和数据库系统之间的联系：

(1)均为数据组织的管理技术；

(2)均由数据管理软件管理数据，程序与数据之间用存取方法进行转换；

(3)数据库系统是在文件系统的基础上发展而来

2. 数据库设计过程（6 个步骤）

   1) 需求分析

   2) 概念结构设计

   3) 逻辑结构设计

   4) 物理结构设计

   5) 数据库实施

   6) 数据库的运行和维护

3. 你知道的云计算知识

   云计算是通过网络提供可伸缩的廉价分布式计算能力
   云计算是一种商业计算模型。它将计算任务分布在大量计算机构成的资源池上，使各种应用系统能够更具需要获取计算力、存储空间和信息服务
   提供资源的网络被称作云。云中的资源在使用者看来是可以无限扩展的，并且可以随时获取，按需使用，随时扩展，按使用付费，就像煤气 水电一样，取用方便费用低廉

4. 普适计算

   通过在日常环境中广泛部署微小的计算设备，人们能够在任何时间和任何地点获取并处理信息，计算将最终和环境融为一体。这就是普适计算。

5. UML

统一建模语言（UML，UnifiedModelingLanguage）是面向对象软件的标准化建模语言。UML 因其简单、统一的特点，而且能表达软件设计中的动态和静态信息，目前已成为可视化建模语言的工业标准。**五种类图定义：**

1.用例图：从用户角度描述系统功能，并指各功能的操作者。

2.静态图：包括类图，包图，对象图。

类图：描述系统中类的静态结构

包图：是包和类组成的，表示包与包之间的关系，包图描述系统的分层结构

对象图：是类图的实例

3.行为图：描述系统动态模型和对象组成的交换关系。包括状态图和活动图

活动图：描述了业务实现用例的工作流程

状态图：是描述状态到状态控制流，常用于动态特性建模

4.交互图：描述对象之间的交互关系

顺序图：对象之间的动态合作关系，强调对象发送消息的顺序，同时显示对象之间的交互

合作图：描述对象之间的协助关系

5.实现图：

配置图：定义系统中软硬件的物理体系结构

6. 完整性约束

关系完整性约束是为保证数据库中数据的正确性和相容性，对关系模型提出的某种约束条件或规则。

实体完整性实体完整性是指一个关系中所有主属性（即主码的属性）不能取空值。

参照完整性 参照的完整性要求关系中不允许引用不存在的实体。又称引用完整性。

用户定义完整性 用户定义某个具体数据库所涉及的数据必须满足的约束条件，是由具体应用环境来决定的。例如，约定学生成绩的数据必须小于或等于 100。

7. 数据库事务 ACID 四大属性，事务的概念？

事务（Transaction），一般是指要做的或所做的事情。在计算机术语中是指访问并可能更新数据库中各种数据项的一个程序执行单元(unit)。

事务是恢复和并发控制的基本单位。

事务应该具有 4 个属性：原子性、一致性、隔离性、持久性。这四个属性通常称为 ACID 特性。

原子性（atomicity）。一个事务是一个不可分割的工作单位，事务中包括的诸操作要么都做，要么都不做。

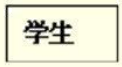
一致性（consistency）。事务必须是使数据库从一个一致性状态变到另一个一致性状态。一致性与原子性是密切相关的。

隔离性（isolation）。一个事务的执行不能被其他事务干扰。即一个事务内部的操作及使用的数据对并发的其他事务是隔离的，并发执行的各个事务之间不能互相干扰。

持久性（durability）。持久性也称永久性（permanence），指一个事务一旦提交，它对数据库中数据的改变就应该是永久性的。接下来的其他操作或故障不应该对其有任何影响。

8. ER 模型

在 E-R 模型中，主要有实体、属性、联系三个部分组成。

	含义	表示方式
实体	考虑问题的对象（学生）	
属性	实体或联系的某一特性（姓名）	
联系	实体之间的联系（学习）	

9. 存储过程（Stored Procedure）是在大型数据库系统中，一组为了完成特定功能的 SQL 语句集，存储在数据库中，经过第一次编译后再次调用不需要再次编译，用户通过指定存储过程的名字并给出参数（如果该存储过程带有参数）来执行它。存储过程是数据库中的一个重要对象。
相当于数据库中的函数（@）

分表：针对每个时间周期产生大量的数据，可以考虑采用一定的策略将数据存到多个数据表中

分库：将系统按照模块相关的的特征分布到不同的数据中，以提高系统整体的负载能力

数据库笛卡尔问题：使用叉乘的方法

11. 范式

范式是符合某一种级别的关系模式的集合。关系数据库中的关系必须满足一定的要求，满足不同程度要求的为不同范式。

I第一范式（1NF）：
数据库表中的字段都是单一属性的，不可再分。这个单一属性由基本类型构成，包括整型、实数、字符型、逻辑型、日期型等。
定义：如果一个关系模式 R 的所有属性都是不可分的基本数据项，则 R 属于 1NF。
例如，下面的数据库表是符合第一范式的：
字段 1 字段 2 字段 3 字段 4
而这样的数据库表是不符合第一范式的：
字段 1 字段 2 字段 3 字段 4 字段 3.1 字段 3.2

很显然，在当前的任何关系数据库管理系统（DBMS）中，谁也不可能做出不符合第一范式的数据库，因为这些 DBMS 不允许你把数据库表的一列再分成二列或多列。因此，想在现有的 DBMS 中设计出不符合第一范式的数据库都是不可能的。

II 第二范式（2NF）：

数据库表中不存在非关键字段对任一候选关键字段的部分函数依赖（部分函数依赖指的是存在组合关键字中的某些字段决定非关键字段的情况），也即所有非关键字段都完全依赖于任意一组候选关键字。

假定选课关系表为 SelectCourse(学号，姓名，年龄，课程名称，成绩，学分)，关键字为组合关键字(学号，课程名称)，因为存在如下决定关系：

(学号，课程名称) → (姓名，年龄，成绩，学分)

这个数据库表不满足第二范式，因为存在如下决定关系：

(课程名称) → (学分)

(学号) → (姓名，年龄)

即存在组合关键字中的字段决定非关键字的情况。

由于不符合 2NF，这个选课关系表会存在如下问题：

(1) 数据冗余：

同一门课程由 n 个学生选修，"学分"就重复 n-1 次；同一个学生选修了 m 门课程，姓名和年龄就重复了 m-1 次。

(2) 更新异常：

若调整了某门课程的学分，数据表中所有行的"学分"值都要更新，否则会出现同一门课程学分不同的情况。

(3) 插入异常：

假设要开设一门新的课程，暂时还没有人选修。这样，由于还没有"学号"关键字，课程名称和学分也无法记录入数据库。

(4) 删除异常：

假设一批学生已经完成课程的选修，这些选修记录就应该从数据库表中删除。但是，与此同时，课程名称和学分信息也被删除了。很显然，这也会导致插入异常。

把选课关系表 SelectCourse 改为如下三个表：

学生：Student(学号,姓名，年龄)；

课程：Course(课程名称，学分)；

选课关系：SelectCourse(学号，课程名称，成绩)。

这样的数据库表是符合第二范式的， 消除了数据冗余、更新异常、插入异常和删除异常。

另外，所有单关键字的数据库表都符合第二范式，因为不可能存在组合关键字。

III 第三范式（3NF）：

在第二范式的基础上，数据表中如果不存在非关键字段对任一候选关键字段的传递函数依赖则符合第三范式。所谓传递函数依赖，指的是如果存在"A → B → C"的决定关系，则 C 传递函数依赖于 A。因此，满足第三范式的数据库表应该不存在如下依赖关系：

关键字段 → 非关键字段 x → 非关键字段 y

假定学生关系表为 Student(学号，姓名，年龄，所在学院，学院地点，学院电话)，关键字为单一关键字"学号"，因为存在如下决定关系：

(学号) → (姓名，年龄，所在学院，学院地点，学院电话)

这个数据库是符合 2NF 的，但是不符合 3NF，因为存在如下决定关系：

(学号) → (所在学院) → (学院地点，学院电话)

即存在非关键字段"学院地点"、"学院电话"对关键字段"学号"的传递函数依赖。

它也会存在数据冗余、更新异常、插入异常和删除异常的情况，读者可自行分析得知。

把学生关系表分为如下两个表：
学生：(学号，姓名,年龄，所在学院)；
学院：(学院，地点，电话)。
这样的数据库表是符合第三范式的，消除了数据冗余、更新异常、插入异常和删除异常。

IV鲍依斯-科得范式（BCNF）：
在第三范式的基础上，数据库表中如果不存在任何字段对任一候选关键字段的传递函数依赖则符合第三范式。
假设仓库管理关系表为 StorehouseManage(仓库 ID,存储物品 ID,管理员 ID,数量)，且有一个管理员只在一个仓库工作，一个仓库可以存储多种物品。这个数据库表中存在如下决定关系：
(仓库 ID,存储物品 ID) →(管理员 ID,数量)
(管理员 ID,存储物品 ID) → (仓库 ID,数量)
所以(仓库 ID,存储物品 ID)和(管理员 ID,存储物品 ID)都是 StorehouseManage 的候选关键字，表中的唯一非关键字段为数量，它是符合第三范式的。但是，由于存在如下决定关系：
(仓库 ID) → (管理员 ID)
(管理员 ID) → (仓库 ID)
即存在关键字段决定关键字段的情况，所以其不符合 BCNF 范式。它会出现如下异常情况：
(1) 删除异常：
当仓库被清空后，所有"存储物品 ID"和"数量"信息被删除的同时，"仓库 ID"和"管理员 ID"信息也被删除了。
(2) 插入异常：
当仓库没有存储任何物品时，无法给仓库分配管理员。
(3) 更新异常：
如果仓库换了管理员，则表中所有行的管理员 ID 都要修改。
把仓库管理关系表分解为二个关系表：
仓库管理：StorehouseManage(仓库 ID,管理员 ID)；
仓库：Storehouse(仓库 ID,存储物品 ID,数量)。
这样的数据库表是符合 BCNF 范式的，消除了删除异常、插入异常和更新异常。

12. 三级模式

外模式

外模式又称子模式或用户模式，对应于用户级。它是某个或某几个用户所看到的数据库的数据视图，是与某一应用有关的数据的逻辑表示。外模式是从模式导出的一个子集，包含模式中允许特定用户使用的那部分数据。用户可以通过外模式描述语言来描述、定义对应于用户的数据记录(外模式)，也可以利用数据操纵语言(Data Manipulation Language，DML)对这些数据记录进行操作。外模式反映了数据库系统的用户观。

概念模式

概念模式又称模式或逻辑模式，对应于概念级。它是由数据库设计者综合所有用户的数据，按照统一的观点构造的全局逻辑结构，是对数据库中全部数据的逻辑结构和特征的总体描述，是所有用户的公共数据视图(全局视图)。它是由数据库管理系统提供的数据模式描述语言(Data Description Language，DDL)来描述、定义的。概念模式反映了数据库系统的整体观。

内模式

内模式又称存储模式，对应于物理级。它是数据库中全体数据的内部表示或底层描述，是数据库最低一级的逻辑描述，它描述了数据在存储介质上的存储方式和物理结构，对应着实际存储在外存储介质上的数据库。内模式由内模式描述语言来描述、定义的。内模式反映了数据库系统的存储观。

在一个数据库系统中，只有唯一的数据库， 因而作为定义 、描述数据库存储结构的内模式和定义、描述数据库逻辑结构的模式，也是唯一的，但建立在数据库系统之上的应用则是非常广泛、多样的，所以对应的外模式不是唯一的，也不可能是唯一的。

13. 自主存储控制和强制存储控制区别

自主访问控制：

由客体的属主对自己的客体进行管理，由属主自己决定是否将自己的客体访问权或部分访问权授予其他主体，这种控制方式是自主的。也就是说，在自主访问控制下，用户可以按自己的意愿，有选择地与其他用户共享他的文件。

自主访问控制是保护系统资源不被非法访问的一种有效手段。但是这种控制是自主的，即它是以保护用户的个人资源的安全为目标并以个人的意志为转移的。

自主访问控制是一种比较宽松的访问控制,一个主题的访问权限具有传递性。其强调的是自主，自己来决定访问策略，其安全风险也来自自主。

强制访问控制：

用户的权限和客体的安全属性都是系统管理员人为设置，或由操作系统自动地按照严格的安全策略与规则进行设置，用户和他们的进程不能修改这些属性。

其强调是强制，由系统来决定。

14. 公共秘钥和传统秘钥的区别和优势

公共密钥密码术利用两个密钥取代常规的一个密钥：一个公共密钥被用来加密数据，而另一个私人密钥被用来解密数据。这两个密钥在数字上相关，但是即便使用许多计算机协同运算，要想从公共密钥中运算出对应的私人密钥也是不现实的。

发送和接收数据的双方，使用相同的或对称的密钥对明文进行加密解密运算的加密方法。

传统密码的缺点：

(1)收发双方持有相同密钥，密钥分配困难。KE=KD

(2)不能方便的实现数字签名，应用不方便。(数字签名概念下面有)

与传统加密方法相比，公共密钥加密系统具有以下三个比较突出的优点：

①用户可以把用于加密的密钥，公开地分发给任何用户。谁都可以使用这把公共的加密密钥与该用户秘密通信。除了持有解密密钥的收件方用户外，没有人能够解开密文。这样，传统加密方法中令人头痛的、代价沉重的密钥分发问题就转变为一个性质完全不同的公共密钥分发问题。

②公共密钥加密系统允许用户事先把公共密钥发表或刊登出来。例如，用户可以把它和电话号码、产品说明等一起刊登出来，让任何人都可以查找并使用。这使得公共密钥应用的范围不再局限于信息加密，还可以应用于身份鉴别、权限区分等各种领域。例如，大家熟知的各种应用软件，如 Windows 95/98 等系统安装时需要的产品序列号，其实就是公共密钥，它通常印在产品授权书的封面或封底上，供安装时鉴别用户的授权身份。

③公共密钥加密不仅改进了传统加密方法，而且还提供了传统加密方法所不具备的应用，即数字签名的公开鉴定系统。

数字签名（又称公钥数字签名）是只有信息的发送者才能产生的别人无法伪造的一段数字串，这段数字串同时也是对信息的发送者发送信息真实性的一个有效证明。

它是一种类似写在纸上 的普通的物理签名，但是使用了公钥加密领域的技术来实现的，用于鉴别数字信息的方法。一套数字签名通常定义两种互补的运算，一个用于签名，另一个用于验证。数字签名是非对称密钥加密技术与数字摘要技术的应用。

15. orm 是什么

对象-关系映射（OBJECT/RELATIONALMAPPING，简称 ORM），是随着面向对象的软件开发方法发展而产生的。用来把对象模型表示的对象映射到基于 S Q L 的关系模型数据库结构中去。这样，我们在具体的操作实体对象的时候，就不需要再去和复杂的 SQ L 语句打交道，只需简单的操作实体对象的属性和方法 [2] 。O R M 技术是在对象和关系之间提供了一条桥梁，前台的对象型数据和数据库中的关系型的数据通过这个桥梁来相互转化 。

16. Sql 优化

1、在表中建立索引，优先考虑 where、group by 使用到的字段。

2、尽量避免使用 select *，返回无用的字段会降低查询效率。如下：

SELECT * FROM t

优化方式：使用具体的字段代替*，只返回使用到的字段。

3、尽量避免使用 in 和 not in，会导致数据库引擎放弃索引进行全表扫描。如下：

SELECT * FROM t WHERE id IN (2,3)

SELECT * FROM t1 WHERE username IN (SELECT username FROM t2)

优化方式：如果是连续数值，可以用 between 代替。如下：

SELECT * FROM t WHERE id BETWEEN 2 AND 3

如果是子查询，可以用 exists 代替。如下：

SELECT * FROM t1 WHERE EXISTS (SELECT * FROM t2 WHERE t1.username = t2.username)

4、尽量避免使用 or，会导致数据库引擎放弃索引进行全表扫描。如下：

SELECT * FROM t WHERE id = 1 OR id = 3

优化方式：可以用 union 代替 or。如下：

SELECT * FROM t WHERE id = 1
UNION
SELECT * FROM t WHERE id = 3

（PS：如果 or 两边的字段是同一个，如例子中这样。貌似两种方式效率差不多，即使 union 扫描的是索引，or 扫描的是全表）

5、尽量避免在字段开头模糊查询，会导致数据库引擎放弃索引进行全表扫描。如下：

SELECT * FROM t WHERE username LIKE '%li%'

优化方式：尽量在字段后面使用模糊查询。如下：

SELECT * FROM t WHERE username LIKE 'li%'

6、尽量避免进行 null 值的判断，会导致数据库引擎放弃索引进行全表扫描。如下：

SELECT * FROM t WHERE score IS NULL

优化方式：可以给字段添加默认值 0，对 0 值进行判断。如下：

SELECT * FROM t WHERE score = 0

7、尽量避免在 where 条件中等号的左侧进行表达式、函数操作，会导致数据库引擎放弃索引进行全表扫描。如下：

SELECT * FROM t2 WHERE score/10 = 9

SELECT * FROM t2 WHERE SUBSTR(username,1,2) = 'li'

优化方式：可以将表达式、函数操作移动到等号右侧。如下：

SELECT * FROM t2 WHERE score = 10*9

SELECT * FROM t2 WHERE username LIKE 'li%'

8、当数据量大时，避免使用 where 1=1 的条件。通常为了方便拼装查询条件，我们会默认使用该条件，数据库引擎会放弃索引进行全表扫描。如下：

SELECT * FROM t WHERE 1=1

优化方式：用代码拼装 sql 时进行判断，没 where 加 where，有 where 加 and。

关系代数是一种抽象的查询语言，用对关系的运算来表达查询，作为研究关系数据语言的数学工具。关系代数的运算对象是关系，运算结果亦为关系。关系代数用到的运算符包括四类：集合运算符、专门的关系运算符、算术比较符和逻辑运算符比较运算符和逻辑运算符是用来辅助专门的关系运算符进行操作的，所以按照运算符的不同，主要将关系代数分为传统的集合运算和专门的关系运算两类。有并、交、差、除、笛卡尔积等运算。

触发器的作用：触发器是一中特殊的存储过程，主要是通过事件来触发而被执行的。它可以强化约束，来维护数据的完整性和一致性，可以跟踪数据库内的操作从而不允许未经许可的更新和变化。可以联级运算。如，某表上的触发器上包含对另一个表的数据操作，而该操作又会导致该表触发器被触发。
例如：创建一个触发器，当插入或更新成绩列时，检查插入的数据是否在合法范围内

什么是存储过程？用什么来调用？
存储过程是一个预编译的 SQL 语句，优点是允许模块化的设计，就是说只需创建一次，以后在该程序中就可以调用多次。如果某次操作需要执行多次 SQL，使用存储过程比单纯 SQL 语句执行要快。 调用： 1）可以用一个命令对象来调用存储过程。 2）可以供外部程序调用，比如：java 程序。

存储过程优缺点：优点： 1）存储过程是预编译过的，执行效率高。 2）存储过程的代码直接存放于数据库中，通过存储过程名直接调用，减少网络通讯。 3）安全性高，执行存储过程需要有一定权限的用户。 4）存储过程可以重复使用，可减少数据库开发人员的工作量。 缺点：移植性差

存储过程和函数的区别

存储过程	函数
用于在数据库中完成特定的操作或者任务（如插入、删除等）	用于特定的数据（如选择）
程序头部声明用 procedure	程序头部声明用 function
程序头部声明时不需描述返回类型	程序头部声明时要描述返回类型，而且 PL/SQL 块中至少要包括一个有效的 return 语句
可以使用 in/out/in out 三种模式的参数	可以使用 in/out/in out 三种模式的参数
可作为一个独立的 PL/SQL 语句来执行	不能独立执行，必须作为表达式的一部分调用
可以通过 out/in out 返回零个或多个值	通过 return 语句返回一个值，且改值要与声明部分一致，也可以是通过 out 类型的参数带出的变量
SQL 语句(DML 或 SELECT)中不可调用存储过程	SQL 语句(DML 或 SELECT)中可以调用函数

索引的作用以及优缺点：索引就一种特殊的查询表，数据库的搜索可以利用它加速对数据的检索。它很类似与现实生活中书的目录，不需要查询整本书内容就可以找到想要的数据。索引可以是唯一的，创建索引允许指定单个列或者是多个列。缺点是它减慢了数据录入的速度，同时也增加了数据库的尺寸大小。

什么字段适合建索引：唯一、不为空、经常被查询的字段

索引类型
逻辑上：单行索、 多行索引、唯一索引、 非唯一索引、函数索引、域索引
物理上：分区索引、非分区索引
B-tree ： 正常型 B 树、反转型 B 树 Bitmap 位图索引

锁：在所以的 DBMS 中，锁是实现事务的关键，锁可以保证事务的完整性和并发性。与现实生活中锁一样，它可以使某些数据的拥有者，在某段时间内不能使用某些数据或数据结构。当然锁还分级别的。

视图：是一种虚拟的表，具有和物理表相同的功能。可以对视图进行增，改，查，操作，试图通常是有一个表或者多个表的行或列的子集。对视图的修改会影响基本表。它使得我们获取数据更容易，相比多表查询。

视图优缺点：优点： 1）对数据库的访问，因为视图可以有选择性的选取数据库里的一部分。 2 ）用户通过简单的查询可以从复杂查询中得到结果。 3 ）维护数据的独立性，试图可从多个表检索数据。 4 ）对于相同的数据可产生不同的视图。 缺点： 性能：查询视图时，必须把视图的查询转化成对基本表的查询，如果这个视图是由一个复杂的多表查询所定义，那么就无法更改数据

游标：是对查询出来的结果集作为一个单元来有效的处理。游标可以定在该单元中的特定行，从结果集的当前行检索一行或多行。可以对结果集当前行做修改。一般不使用游标，但是需要逐条处理数据的时候，游标显得十分重要。

列举几种表连接方式,有什么区别？

内连接、自连接、外连接（左、右、全）、交叉连接 内连接：只有两个元素表相匹配的才能在结果集中显示。 外连接： 左外连接:左边为驱动表，驱动表的数据全部显示，匹配表的不匹配的不会显示。 右外连接:右边为驱动表，驱动表的数据全部显示，匹配表的不匹配的不会显示。 全外连接：连接的表中不匹配的数据全部会显示出来， 交叉连接： 笛卡尔效应，显示的结果是链接表数的乘积。

表和视图的关系

视图其实就是一条查询 sql 语句，用于显示一个或多个表或其他视图中的相关数据。 表就是关系数据库中实际存储数据用的。

主键和外键的区别？

主键在本表中是唯一的、不可唯空的，外键可以重复可以唯空；外键和另一张表的主键关联，不能创建对应表中不存在的外键。

union 和 union all 有什么不同？

UNION 在进行表链接后会筛选掉重复的记录，所以在表链接后会对所产生的结果集进行排序运算，删除重复的记录再返回结果。实际大部分应用中是不会产生重复的记录，最常见的是过程表与历史表 UNION。 UNION ALL 只是简单的将两个结果合并后就返回。这样，如果返回的两个结果集中有重复的数据，那么返回的结果集就会包含重复的数据了。 从效率上说，UNION ALL 要比 UNION 快很多，所以，如果可以确认合并的两个结果集中不包含重复的数据的话，那么就使用 UNION ALL。

Varchar2 和 varchar 有什么区别？

Char 的长度是固定的，而 varchar2 的长度是可以变化的，比如，存储字符串“abc”对于 char(20)，表示你存储的字符将占 20 个字节，包含 17 个空，而同样的 varchar2（20）只占了 3 个字节，20 只是最大值，当你存储的字符小于 20 时，按实际长度存储。 char 的效率要被 varchar2 的效率高。 目前 varchar 是 varchar2 的同义词，工业标准的 varchar 类型可以存储空字符串，但是 oracle 不能这样做，尽管它保留以后这样做的权利。Oracle 自己开发了一个数据类型 varchar2,这个类型不是一个标准的 varchar，他将在数据库中 varchar 列可以存储空字符串的特性改为存储 null 值，如果你想有向后兼容的能力，oracle 建议使用 varchar2 而不是 varchar

Oracle 和 Mysql 的区别？

1）库函数不同。 2）Oracle 是用表空间来管理的，Mysql 不是。 3）显示当前所有的表、用户、改变连接用户、显示当前连接用户、执行外部脚本的语句的不同。 4）分页查询时候时候，mysql 用 limit oracle 用 rownum

Oracle 语句有多少类型

Oracle 语句分三类：DDL、DML、DCL。 DDL（Data Definition Language）数据定义语言，包括： Create 语句：可以创建数据库和数据库的一些对象。 Drop 语句：可以删除数据表、索引、触发程序、条件约束以及数据表的权限等。 Alter 语句：修改数据表定义及属性。 Truncate 语句：删除表中的所有记录,包括所有空间分配的记录被删除。 DML（Data Manipulation Language）数据操控语言，包括： Insert 语句：向数据表张插入一条记录。 Delete 语句：删除数据表中的一条或多条记录，也可以删除数据表中的所有记录，但是它的操作对象仍是记录。 Update 语句：用于修改已存在表中的记录的内容。 DCL（Data Control Language）数据库控制语言，包括： Grant 语句：允许对象的创建者给某用户或某组或所有用户（PUBLIC）某些特定的权限。 Revoke 语句：可以废除某用户或某组或所有用户访问权限

order by 与 group by 的区别

order by 排序查询、asc 升序、desc 降序 group by 分组查询、having 只能用于 group by 子句、作用于组内，having 条件子句可以直接跟函数表达式。使用 group by 子句的查询语句需要使用聚合函数

commit 在哪里会运用

oracle 的 commit 就是 DML 语句提交数据（这里是释放锁不是锁表），在未提交前你前面的操作更新的都是内存，没有更新到物理文件中。执行 commit 从用户角度讲就是更新到物理文件了，事实上 commit 时还没有写 date file，而是记录了 redo log file，要从内存写到 data 物理文件，需要触发检查点，由 DBWR 这个后台进程来写，这里内容有点多的，如果不深究的话你就理解成 commit 即为从内存更新到物理文件。

行转列、列换行怎么转

1）使用 decode 函数 2）使用 case when 语句

什么是 PL/SQL?

PL/SQL 是一种程序语言，叫做过程化 SQL 语言（Procedural Language/SQL）。PL/SQL 是 Oracle 数据库对 SQL 语句的扩展。在普通 SQL 语句的使用上增加了编程语言的特点，所以 PL/SQL 把数据操作和查询语句组织在 PL/SQL 代码的过程性单元中，通过逻辑判断、循环等操作实现复杂的功能或者计算。PL/SQL 只有 Oracle 数据库有。 MySQL 目前不支持 PL/SQL 的。

序列的作用

Oracle 使用序列来生成唯一编号，用来处理一个表中自增字段。 Oracle 序列是原子对象，并且是一致的。也就是说，一旦您访问一个序列号，Oracle 将在处理下一个请求之前自动递增下一个编号，从而确保不会出现重复值。

oracle 基本数据类型

1）字符串类型 char、nchar、varchar、varchar2、nvarchar2 2）数字类型 number、integer 3）浮点类型 binary_float、binary_double、float 4）日期类型 date、 timestamp 5）LOB 类型 blob、clob、nclob、bfile

truncate 与 delete 区别

TRUNCATE TABLE 在功能上与不带 WHERE 子句的 DELETE 语句相同：二者均删除表中的全部行。但 TRUNCATE TABLE 比 DELETE 速度快，且使用的系统和事务日志资源少。

DELETE 语句每次删除一行，并在事务日志中为所删除的每行记录一项。
TRUNCATE TABLE 通过释放存储表数据所用的数据页来删除数据，并且只在事务日志中记录页的释放。 TRUNCATE,DELETE,DROP 放在一起比较：
TRUNCATE TABLE ：删除内容、释放空间但不删除定义。
DELETE TABLE: 删除内容不删除定义，不释放空间。
DROP TABLE ：删除内容和定义，释放空间。

oracle 怎么去除去重：使用 distinct 关键字
数据库存储满了是否还可以进行其他操作：可以删除和修改，不能添加

数据独立性是指应用程序和数据结构之间相互独立, 互不影响。在三层模式体系结构中数据独立性是指数据库系统在某一层次模式上的改变不会使它的上一层模式也发生改变的能力。数据独立性包括数据逻辑独立性和数据物理独立性。

物理独立性是指用户的应用程序与存储在磁盘上的数据库中数据是相互独立的，即，数据在磁盘上怎样存储由 DBMS 管理，用户程序不需要了解，应用程序要处理的只是数据的逻辑结构，这样当数据的物理存储改变了，应用程序不用改变。 [3]

逻辑独立性是指用户的应用程序与数据库的逻辑结构是相互独立的，即，当数据的逻辑结构改变时，用户程序也可以不变。

数据模型是数据特征的抽象。数据（Data）是描述事物的符号记录，模型（Model)是现实世界的抽象。按不同的应用层次分成三种类型，分别是概念数据模型、逻辑数据模型、物理数据模型。

概念模型

概念模型（Conceptual Data Model），是一种面向用户、面向客观世界的模型，主要用来描述世界的概念化结构，它是数据库的设计人员在设计的初始阶段，摆脱计算机系统及 DBMS 的具体技术问题，集中精力分析数据以及数据之间的联系等，与具体的数据管理系统（Database Management System，简称 DBMS）无关。概念数据模型必须换成逻辑数据模型，才能在 DBMS 中实现。

概念模型用于信息世界的建模，一方面应该具有较强的语义表达能力，能够方便直接表达应用中的各种语义知识，另一方面它还应该简单、清晰、易于用户理解。

在概念数据模型中最常用的是 E-R 模型、扩充的 E-R 模型、面向对象模型及谓词模型。较为有名的是 E-R 模型。

逻辑模型

逻辑模型（Logical Data Model），是一种面向数据库系统的模型，是具体的 DBMS 所支持的数据模型，如网状数据模型(Network Data Model)、层次数据模型(Hierarchical Data Model)等等。此模型既要面向用户，又要面向系统，主要用于数据库管理系统（DBMS）的实现。

物理模型

物理模型（Physical Data Model），是一种面向计算机物理表示的模型，描述了数据在储存介质上的组织结构，它不但与具体的 DBMS 有关，而且还与操作系统和硬件有关。每一种逻辑数据模型在实现时都有其对应的物理数据模型。DBMS 为了保证其独立性与可移植性，大部分物理数据模型的实现工作由系统自动完成，而设计者只设计索引、聚集等特殊结构。

having 和 where 的联系与区别？

having 和 where 都是用来筛选用的

having 是筛选组 而 where 是筛选记录

他们有各自的区别

1》当分组筛选的时候 用 having

2》其它情况用 where

用 having 就一定要和 group by 连用，

用 group by 不一有 having （它只是一个筛选条件用的）

规范化理论正是用来改造关系模式，通过分解关系模式来消除其中不合适的数据依赖，以解决插入异常、删除异常、更新异常和数据冗余问题。

规范化理论是数据库逻辑设计的指南和工具，具体表现在一下三个方面：

1，在数据分析阶段，用数据依赖的概念分析和表示各项数据项之间的关系。

2，在设计概念结构阶段，用规范化理论消除初步 ER 图冗余的联系。

3，有 ER 图像数据模型转化阶段，用模式分解的概念和方法指导设计。

函数依赖是唯一确定的关系，例如在一个表 授课（课程号，课程名，课程学分，教师号，教师姓名，参考书号，参考书名)。当然这个表的主键是（课程号，教师号，参考书号），课程名和课程学分函数依赖课程号，也就是说课程号唯一确定名字和学分；同理，教师姓名函数依赖教师号；参考书名函数依赖参考书号。但是，教师号可能多值依赖课程号，因为给定一个（课程号，参考书号）的组合，可能有对应多个教师号。这是因为多个老师可以使用相同或不同的参考书上同一门课。 简单点讲，函数就是唯一确定的关系；多值依赖却不能唯一确定。

从已知的一些函数依赖，可以推导出另外一些函数依赖，这就需要一系列推理规则，这些规则常被称作“**Armstrong 公理**”。

数据库设计(Database Design)是指对于一个给定的应用环境，构造最优的数据库模式，建立数据库及其应用系统，使之能够有效地存储数据，满足各种用户的应用需求（信息要求和处理要求）。在数据库领域内，常常把使用数据库的各类系统统称为数据库应用系统

DBMS 常用存取方式：第一种是索引方法，主要时 B+树索引方法；第二种是聚簇（clustering）方法；第三种是 hash 方法。
计算机系统对并发事务中并发操作的调度是随机的，而不同的调度可能会产生不同的结果。在计算机中，多个事务的并发执行是正确的，当且仅当其结果与按某一次序串行地执行它们时的结果相同，我们称这种调度策略为可串行化（Serializable）调度。

共享锁【S 锁】
又称读锁，若事务 T 对数据对象 A 加上 S 锁，则事务 T 可以读 A 但不能修改 A，其他事务只能再对 A 加 S 锁，而不能加 X 锁，直到 T 释放 A 上的 S 锁。这保证了其他事务可以读 A，但在 T 释放 A 上的 S 锁之前不能对 A 做任何修改。

排他锁【X 锁】
又称写锁。若事务 T 对数据对象 A 加上 X 锁，事务 T 可以读 A 也可以修改 A，其他事务不能再对 A 加任何锁，直到 T 释放 A 上的锁。这保证了其他事务在 T 释放 A 上的锁之前不能再读取和修改 A。

关系数据库，是建立在关系数据库模型基础上的数据库，借助于集合代数等概念和方法来处理数据库中的数据，同时也是一个被组织成一组拥有正式描述性的表格，该形式的表格作用的实质是装载着数据项的特殊收集体，这些表格中的数据能以许多不同的方式被存取或重新召集而不需要重新组织数据库表格。关系数据库的定义造成元数据的一张表格或造成表格、列、范围和约束的正式描述。每个表格（有时被称为一个关系）包含用列表示的一个或更多的数据种类。 每行包含一个唯一的数据实体，这些数据是被列定义的种类。当创造一个关系数据库的时候，你能定义数据列的可能值的范围和可能应用于那个数据值的进一步约束。而 SQL 语言是标准用户和应用程序到关系数据库的接口。其优势是容易扩充，且在最初的数据库创造之后，一个新的数据种类能被添加而不需要修改所有的现有应用软件。主流的关系数据库有 oracle、db2、sqlserver、sybase、mysql 等。 [1]
操作
◇查询:选择、投影、连接、除、并、交、差
◇数据更新:插入（insert）、删除(delete)、修改(update)

操作系统和计组

16. 进程和线程的区别

1 调度性：线程在 OS 中作为调度和分派的基本单位，进程只作为资源拥有的基本单位。
2 并发性：进程可以并发执行，一个进程的多个线程也可并发执行。
3 拥有资源：进程始终是拥有资源的基本单位，线程只拥有运行时必不可少的资源，本身基本不拥有系统资源，但可以访问隶属进程的资源。
4 系统开销：操作系统在创建、撤消和切换进程时付出的开销显著大于线程。

17. 物理地址 逻辑地址

物理地址：加载到内存地址寄存器中的地址，是内存单元的真正地址。
逻辑地址：CPU 所生成的地址。逻辑地址是内部和编程使用的、并不唯一。

18. 单片机和微机存储的不同

微型计算机：微处理器(CPU)、存储器、I/O 接口电路由总线有机地连接在一起的整体，称为微型计算机。

单片机：将微处理器(CPU)、存储器、I/O 接口电路和相应实时控制器件集成在一块芯片上，称为单片微型计算机，简称单片机。

两者的主要区别：

(1) 通用微机的 CPU 主要面向数据处理，其发展主要围绕数据处理功能、计算速度和精度的进一步提高。单片机主要面向控制，控制中的数据类型及数据处理相对简单，所以单片机的数据处理功能比通用微机相对要弱一些，计算速度和精度也相对要低一些。

(2) 通用微机中存储器组织结构主要针对增大存储容量和 CPU 对数据的存取速度。单片机中存储器的组织结构比较简单，存储器芯片直接挂接在单片机的总线上，CPU 对存储器的读写按直接物理地址来寻址存储器单元，存储器的寻址空间一般都为 64 KB。

(3) 通用微机中 I/O 接口主要考虑标准外设(如 CRT、标准键盘、鼠标、打印机、硬盘、光盘等)。单片机的 I/O 接口实际上是向用户提供的与外设连接的物理界面。用户对外设的连接要设计具体的接口电路，需有熟练的接口电路设计技术。

19. 计算机操作系统的作用和组成？

操作系统的作用：操作系统位于底层硬件和用户之间，是两者沟通的桥梁。用户可以通过操作系统的用户界面，输入命令。操作系统则对命令进行解释，驱动硬件设备，实现用户要求。以现代标准而言，一个标准的 PC 操作系统应该提供如下功能：

1. 处理机管理功能（即管理 cpu）
2. 存储器管理功能 （管理内存）
3. 设备管理功能 （管理其它外设，比如打印机）
4. 文件管理功能 （管理我们的文件）
5. 作业管理 （如何完成用户交给的任务）

操作系统的组成：进程管理、存储器管理、设备管理、文件管理、用户接口

20. 有没有一进制存

在有 11111

21. 计算机组成原理和微机原理的关系

微机原理是是对《计算机组成》的具体实现。一般会选 x86 计算机来说，这时一般不会再详细讲工作原理了（因为大多数原理都在《计算机组成》中讲了），而是直接讲述 cpu 的具本结构是什么，具体引脚的作用，各种总线多少多少，各种控制寄存器的各个位有什么意义， I/O 的具体交接口（ISA pci）……因为是具体的东西，所以有好多东东要记的。

计算机组成原理主要是介绍计算机的基本硬件及原理。重在各个部分的连接。相对宽泛一些。

微机原理接口技术比计算机组成原理要具体些，也比计算机组成要好学，一般是以 8086 为例，介绍 8086CPU 的结构，其中最最重要的就是汇编语言和芯片；掌握了汇编语言这几块芯片的编程基本上就差不多了。

22. 作业调度和进程调度的区别：

进程调度是真正让某个就绪状态的进程到处理机上运行，而作业调度只是使作业具有了竞争处理机的机会。

进程调度（又称微观调度、低级调度、短程调度）：是按照某种调度算法从就绪状态的进程中选择一个进程到处理机上运行。负责进程调度功能的内核程序称为进程调度程序。

作业调度（又称高级调度、宏观调度、长程调度）：是按某种调度算法从后备作业队列中选择作业装入内存运行；另外当该作业执行完毕后，还负责回收系统资源。完成作业调度功能的程序称为作业调度程序。

22. 指令和数据放一起存储的，计算机是怎么区别指令和算法的？

1) 通过不同的时间段来区分指令和数据，即在取指令阶段（或取指微程序）取出的为指令，在执行指令阶段（或相应微程序）取出的即为数据。
2) 通过地址来源区分，由 PC 提供存储单元地址的取出的是指令，由指令地址码部分提供存储单元地址的取出的是操作数。

进程的生存周期：创建 就绪 执行 阻塞 结束

进程间如何通信：共享存储、消息传递、管道通信

互斥锁的原理机制：信号量 pv 操作、标注等待与释放。

进程管理的状态切换：就绪到运行 运行到就绪 运行到阻塞 阻塞到就绪

操作系统的特征：并发 共享 虚拟 异步

进程的特征：并发性、异步性、独立性、结构性、动态性

进程组织包括：PCB、数据段、程序段

什么是进程？
进程是一个具有独立功能的程序关于某个数据集合的一次运行活动。
当一个程序开始运行时，它就是一个进程，进程包括运行中的程序和程序所使用到的内存和系统资源。
而一个进程又是由多个线程所组成的。

什么是线程？
线程是程序中的一个执行流，每个线程都有自己的专有寄存器(栈指针、程序计数器等)，但代码区是共享的，
即不同的线程可以执行同样的函数。

什么是单线程？
单线程在程序执行时，所走的程序路径按照连续顺序排下来，前面的必须处理好，后面的才会执行。单线程对于多线程来说的好处：系统稳定、扩展性极强、软件丰富。多用于点对点的服务。

什么是多线程？
多线程是指程序中包含多个执行流，即在一个程序中可以同时运行多个不同的线程来执行不同的任务，
也就是说允许单个程序创建多个并行执行的线程来完成各自的任务。
举例：文字处理器应用程序在处理文档的同时，可以检查拼写（作为单独的任务）

多线程的好处：
可以提高 CPU 的利用率。在多线程程序中，一个线程必须等待的时候，CPU 可以运行其它的线程而不是等待，
这样就大大提高了程序的效率。

多线程的不利方面：
线程也是程序，所以线程需要占用内存，线程越多占用内存也越多；
多线程需要协调和管理，所以需要 CPU 时间跟踪线程；
线程之间对共享资源的访问会相互影响，必须解决竞用共享资源的问题；
线程太多会导致控制太复杂，最终可能造成很多 Bug；

线程相对于进程的优点：
1、开销小
2、资源共享性好。

线程相对于进程的缺点：
1、共享资源需要耗费一定的锁资源，同步相对复杂。
2、一个线程崩溃可能导致整个进程崩溃，这个当然是自己的应用程序有问题

计算机组成指的是系统结构的逻辑实现，包括机器内的数据流和控制流的组成及逻辑设计等。主要分为五部分：控制器、运算器、存储器、输入设备、输出设备

76.操作系统

※OS 作用：1 OS 作为用户与计算机硬件系统之间的接口；2 OS 作为计算机系统资源的管理者；3 OS 实现了对计算机资源的抽象。

※OS 特征：并发性、共享性、虚拟性和异步性四个基本特征；最基本的特征是并发性

计算机网络

23. TCP/IP 协议可以说成 3 层、4 层、5 层，怎么理解

传输控制协议/网际协议，是 Internet 最基本的协议，由网络层的 IP 协议和传输层的 TCP 协议组成。TCP/IP 定义了电子设备接入互联网和数据传输。俗而言：TCP 负责发现传输的问题，一有问题就发出信号，要求重新传输，直到所有数据安全正确地传输到目的地。而 IP 是给因特网的每一台电脑规定一个地址。

OSI 7层模型	TCP/IP 4层	5层	主要协议
应用层	应用层	应用层	http,ftp,dns, Dhcp,Vpn，私有协议等
表示层			
会话层			
传输层	传输层	传输层	Tcp，udp
网络层	网络层	网络层	Icmp
数据链路层	链路层	链路层	ARP
物理层		物理层	以太网,WIFI,4G

TCP/IP 协议族常用协议：Telnet 提供远程登录功能、FTP 远程文件传输协议、SMTP 简单邮政传输协议、UDP 用户数据包协议

每一层的作用如下：

一、物理层（Physical Layer）

OSI 模型的最低层或第一层，规定了激活、维持、关闭通信端点之间的机械特性、电气特性、功能特性以及过程特性，为上层协议提供了一个传输数据的物理媒体。
在这一层，协议数据单元为比特（bit）。
在物理层的互联设备包括：集线器（Hub）、中继器（Repeater）等。
相关接口或协议：RJ45 就是水晶头、CLOCK、IEEE802.3

二、数据链路层（Datalink Layer）
OSI 模型的第二层，它控制网络层与物理层之间的通信，其主要功能是在不可靠的物理介质上提供可靠的传输。该层的作用包括：物理地址寻址、数据的成帧、流量控制、数据的检错、重发等。
在这一层，协议数据单元为帧（frame）。
在数据链路层的互联设备包括：网桥（Bridge）、交换机（Switch）等。
协议：PPP 适用于点到点数据链路、VLAN 虚拟局域网、MAC、CSMA/CD 适用于广播信道

三、网络层也叫 IP 层（Network Layer）
OSI 模型的第三层，其主要功能是将网络地址翻译成对应的物理地址，并决定如何将数据从发送方路由到接收方。该层的作用包括：对子网间的数据包进行路由选择，实现拥塞控制、网际互连等功能。
在这一层，协议数据单元为数据包（packet）。
在网络层的互联设备包括：路由器（Router）等。
协议：网际协议 IP、因特网控制信息协议 ICMP、ARP、RARP、开放最短路径优先协议 OSPF、路由信息协议 RIP、内部网关路由协议 IGRP、 Internet 组管理协议 IGMP

四、传输层（Transport Layer）
OSI 模型中最重要的一层，是第一个端到端，即主机到主机的层次。其主要功能是负责将上层数据分段并提供端到端的、可靠的或不可靠的传输。此外，传输层还要处理端到端的差错控制和流量控制问题。
在这一层，协议数据单元为数据段（segment）。
传输层协议的代表包括：TCP、UDP、SPX 等。

五、会话层（Session Layer）
OSI 模型的第五层，管理主机之间的会话进程，即负责建立、管理、终止进程之间的会话。其主要功能是建立通信链接，保持会话过程通信链接的畅通，利用在数据中插入校验点来同步两个结点之间的对话，决定通信是否被中断以及通信中断时决定从何处重新发送。
应用：网络文件系统 NFS、SQL、网上基本输入输出系统 NETBIOS、远程过程调用 RPC

六、表示层（Presentation Layer）
OSI 模型的第六层，应用程序和网络之间的翻译官，负责对上层数据或信息进行变换以保证一个主机应用层信息可以被另一个主机的应用程序理解。表示层的数据转换包括数据的解密和加密、压缩、格式转换等。
应用：JPEG、MPEG、ASII

七、应用层（Application Layer）
OSI 模型的第七层，负责为操作系统或网络应用程序提供访问网络服务的接口。术语“应用层”并不是指运行在网络上的某个特别应用程序，应用层提供的服务包括文件传输、文件管理以及电子邮件的信息处理。
在应用层的互联设备包括：网关（Gateway）等。
文件传输协议 FTP（File Transfer Protocol），端口号为 21；
超文本传输协议 HTTP（HypertextTransfer Protocol），端口号为 80；
简单网络管理协议 SNMP（SimpleNetwork Management Protocol）
域名服务协议 DNS（Domain Name Service）
网络文件系统 NFS（Network File System）
等……

OSI 参考模型和网络设备的对应关系：传输层以上是网关（地址类型端口），网络层是路由器和交换机（地址类型：网络地址），链路层是网桥（地址类型 MAC 地址），物理层是中继器和集线器，调制解调器

数据链路层三个问题：封装成帧，透明传输，差错检测

互联网组管理协议（IGMP，Internet Group Management Protocol）是因特网协议家族中的一个组播协议，TCP/IP 协议族的一个子协议，用于 IP 主机向任一个直接相邻的路由器报告他们的组员情况。

数据包不分片最大 1500 字节其中包括首部 20 字节

网络畅通条件：数据包有去有回

静态路由和动态路由的区别
两者的区别在于，静态路由是指由网络管理员手工配置的路由信息；动态路由是指路由器能够自动地建立自己的路由表，并且能够根据实际实际情况的变化适时地进行调整。
1、静态路由是指由网络管理员手工配置的路由信息。当网络的拓扑结构或链路的状态发生变化时，网络管理员需要手工去修改路由表中相关的静态路由信息。
2、动态路由是指路由器能够自动地建立自己的路由表，并且能够根据实际实际情况的变化适时地进行调整。

IP 地址的分类

A 类地址：以 0 开头， 第一个字节范围：0~127（1.0.0.0 - 126.255.255.255）；

B 类地址：以 10 开头， 第一个字节范围：128~191（128.0.0.0 - 191.255.255.255）；

C 类地址：以 110 开头， 第一个字节范围：192~223（192.0.0.0 - 223.255.255.255）；
10.0.0.0—10.255.255.255， 172.16.0.0—172.31.255.255， 192.168.0.0—192.168.255.255 。（Internet 上保留地址用于内部）

IP 地址与子网掩码相与得到主机号

IP 地址组成：网络号+地址类型号+主机号

域名和 IP 地址的关系：IP 地址与域名是一对多的关系。一个 IP 地址可以对应多个域名，但一个域名只有一个 IP 地址

ARP 是地址解析协议，简单语言解释一下工作原理。

1：首先，每个主机都会在自己的 ARP 缓冲区中建立一个 ARP 列表，以表示 IP 地址和 MAC 地址之间的对应关系。

2：当源主机要发送数据时，首先检查 ARP 列表中是否有对应 IP 地址的目的主机的 MAC 地址，如果有，则直接发送数据，如果没有，就向本网段的所有主机发送 ARP 数据包，该数据包包括的内容有：**源主机 IP 地址，源主机 MAC 地址，目的主机的 IP 地址**。

3：当本网络的所有主机收到该 ARP 数据包时，首先检查数据包中的 IP 地址是否是自己的 IP 地址，如果不是，则忽略该数据包，如果是，则首先从数据包中取出源主机的 IP 和 MAC 地址写入到 ARP 列表中，如果已经存在，则覆盖，然后将自己的 MAC 地址写入 ARP 响应包中，告诉源主机自己是它想要找的 MAC 地址。

4：源主机收到 ARP 响应包后。将目的主机的 IP 和 MAC 地址写入 ARP 列表，并利用此信息发送数据。如果源主机一直没有收到 ARP 响应数据包，表示 ARP 查询失败。广播发送 ARP 请求，单播发送 ARP 响应。

ICMP 协议： 因特网控制报文协议。它是 TCP/IP 协议族的一个子协议，用于在 IP 主机、路由器之间传递控制消息。测试网络是否畅通 （ping 一下）

HTTP 协议： 超文本传输协议，是一个属于应用层的面向对象的协议，由于其简捷、快速的方式，适用于分布式超媒体信息系统。用于实现 WWW 服务。

NAT 网络地址转换协议：网络地址转换属接入广域网(WAN)技术，是一种将私有（保留）地址转化为合法 IP 地址的转换技术，

DHCP 动态主机配置协议：一个局域网的网络协议，使用 UDP 协议工作，用途：给内部网络或网络服务供应商自动分配 IP 地址，给用户或者内部网络管理员作为对所有计算机作中央管理的手段。
域名系统(Domain Name System，DNS)：用于实现网络设备名字到 IP 地址映射的网络服务。
文件传输协议(File Transfer Protocol，FTP)：用于实现交互式文件传输功能。
简单邮件传送协议(Simple Mail Transfer Protocol, SMTP)：用于实现电子邮箱传送功能。
简单网络管理协议(simple Network Management Protocol，SNMP)：用于管理与监视网络设备。
远程登录协议(Telnet)：用于实现远程登录功能。

24. TCP 和 UDP 的区别？

TCP 提供面向连接的、可靠的数据流传输，而 UDP 提供的是非面向连接的、不可靠的数据流传输。

TCP 传输单位称为 TCP 报文段，UDP 传输单位称为用户数据报。
TCP 需要将传输的文件分段，建立会话，流量控制，例如 QQ 传文件。UDP 一个数据包就能完成通信不需要分段，不需要流量控制，例如 QQ 聊天。

TCP 注重数据安全性，UDP 数据传输快，因为不需要连接等待，少了许多操作，但是其安全性却一般。

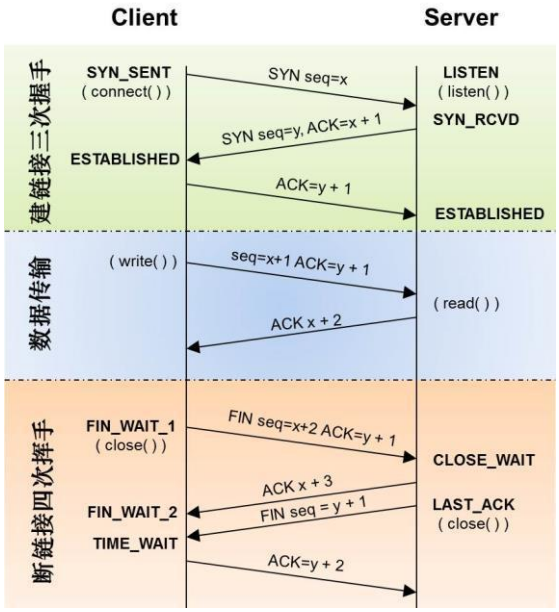
26. 广域网和局域网的主流技术

广域网（WAN，Wide Area Network）也称远程网（long haul network）。通常跨接很大的物理范围，所覆盖的范围从几十公里到几千公里，它能连接多个城市或国家，或横跨几个洲并能提供远距离通信，形成国际性的远程网络。广域网的通信子网主要使用分组交换技术。广域网的通信子网可以利用公用分组交换网、卫星通信网和无线分组交换网，它将分布在不同地区的局域网或计算机系统互连起来，达到资源共享的目的。如因特网（Internet）是世界范围内最大的广域网。

局域网（Local Area Network，LAN）是指在某一区域内由多台计算机互联成的计算机组。一般是方圆几千米以内。局域网可以实现文件管理、应用软件共享、打印机共享、工作组内的日程安排、电子邮件和传真通信服务等功能。局域网是封闭型的，可以由办公室内的两台计算机组成，也可以由一个公司内的上千台计算机组成。

27. 网络 红蓝军打仗通信问题 2

三次握手



第一次握手：客户端发送 syn 包(syn=x)到服务器，并进入 SYN_SEND 状态，等待服务器确认；

第二次握手：服务器收到 syn 包，必须确认客户的 SYN（ack=x+1），同时自己也发送一个 SYN 包（syn=y），即 SYN+ACK 包，此时服务器进入 SYN_RECV 状态；

第三次握手：客户端收到服务器的 SYN + ACK 包，向服务器发送确认包 ACK(ack=y+1)，此包发送完毕，客户端和服务器进入 ESTABLISHED 状态，完成三次握手。

握手过程中传送的包里不包含数据，三次握手完毕后，客户端与服务器才正式开始传送数据。理想状态下，TCP 连接一旦建立，在通信双方中的任何一方主动关闭连接之前，TCP 连接都将被一直保持下去。

三次握手的必要性：
假如只进行一次握手，客户端发送连接请求后，没有收到服务端的应答，是没法判断连接是否成功的。

假如只进行两次握手，客户端发送连接请求后，会等待服务器端的应答。但是会出现的问题是，假如客户端的 SYN 迟迟没有到达服务器端，此时客户端超时后，会重新发送一次连接，假如重发的这次服务器端收到了，且应答客户端了，连接建立了。但是建立后，第一个 SYN 也到达服务端了，这时服务端会认为这是一个新连接，会再给客户端发送一个 ACK，这个 ACK 当然会被客户端丢弃。但是此时服务器端已经为这个连接分配资源了，而且服务器端会一直维持着这个资源，会造成浪费

三次握手，两次握手的问题在于服务器端不知道 SYN 的有效性，所以如果是三次握手，服务器端会等待客户端的第三次握手，如果第三次握手迟迟不来，服务器端就会释放相关资源。但是有人会问，假如第三次握手没有到达服务器端呢？但是这时客户端认为连接已经建立了。但是其实这种情况下，只要客户端向服务器端写数据，就会收到服务器端的 RST 应答，这时客户端就能知道出现问题了。

四次握手
与建立连接的“三次握手”类似，断开一个 TCP 连接则需要“四次握手”。

第一次挥手：主动关闭方发送一个 FIN，用来关闭主动方到被动关闭方的数据传送，也就是主动关闭方告诉被动关闭方：我已经不 会再给你发数据了(当然，在 fin 包之前发送出去的数据，如果没有收到对应的 ack 确认报文，主动关闭方依然会重发这些数据)，但是，此时主动关闭方还可 以接受数据。

第二次挥手：被动关闭方收到 FIN 包后，发送一个 ACK 给对方，确认序号为收到序号+1（与 SYN 相同，一个 FIN 占用一个序号）。

第三次挥手：被动关闭方发送一个 FIN，用来关闭被动关闭方到主动关闭方的数据传送，也就是告诉主动关闭方，我的数据也发送完了，不会再给你发数据了。

第四次挥手：主动关闭方收到 FIN 后，发送一个 ACK 给被动关闭方，确认序号为收到序号+1，至此，完成四次挥手。

四次挥手必要性
其实是客户端和服务端的两次挥手，也就是客户端和服务端分别释放连接的过程。可以看到，客户端在发送完最后一次确认之后，还要等待 2MSL 的时间。主要有两个原因，一个是为了让服务端能够按照正常步骤进入 CLOSED 状态，二是为了防止已经失效的请求连接报文出现在下次连接中。
解释：

1）、由于**客户端**最后一个 ACK 可能会丢失，这样**服务端**就无法正常进入 CLOSED 状态。于是**服务端**会重传请求释放的报文，而此时**客户端**如果已经关闭了，那就收不到**服务端**的重传请求，就会导致**服务端**不能正常释放。而如果**客户端**还在等待时间内，就会收到**服务端**的重传，然后进行应答，这样**服务端**就可以进入 CLOSED 状态了。

2）、在这 2MSL（报文最大生存时间）等待时间里面，本次连接的所有的报文都已经从网络中消失，从而不会出现在下次连接中。

停止等待协议：“停止等待”就是每发送完一个分组就停止发送，等待对方的确认。在收到确认再发送下一个分组。
优点：简单，好实现。缺点：效率太低，信道利用率太低。

连续 ARQ（自动重传请求）协议
连续 ARQ 协议规定，发送方每收到一个确认，就把发送窗口向前滑动一个分组的位置。如果原来已经发送了前 5 个分组，那么现在就可以发送窗口内的第 6 个分组了。

接收方一般都是采用累积确认的方式，也就是说，接收方不必对收到的分组逐个发送确认，而是在收到几个分组后，对按序到达的最后一个分组发送确认，这就表示：到这个分组分组为止的所有分组都已正确收到了。

累计确认优点：容易实现，即使确认丢失也不必重传。缺点：不能向发送方反映出接收方已经正确收到所有的分组的信息。

TCP 还有滑动窗口技术和流量控制

红蓝军对抗

两军问题可以这样描述：一支白军被围困在一个山谷中，山谷的两侧是蓝军。困在山谷中的白军人数多余山谷两侧的任意一支蓝军，而少于两支蓝军的总和。若一支蓝军对白军单独发起进攻，则必败无疑；但若两支蓝军同时发起进攻，则可取胜。两只蓝军希望同时发起进攻，这样他们就要传递消息，以便确定发起进攻的具体时间。假设他们只能派遣士兵穿越白军所在的山谷（唯一的通信信道）来传递消息，那么在穿越山谷是，士兵有可能被俘虏，从而造成消息的丢失。现在的问题是：如何通信，以便蓝军必胜。下面我们进行设计。

假设一支蓝军指挥官发出消息：“我建议在明天佛晓发起进攻，请确认。”如果消息到达了另一支蓝军，其指挥官同意这一建议，并且他的回信也安全的送到，那么能否进攻呢？不能。这是一个两步握手协议，因为该指挥官无法知道他的回信是否安全送到了，所以，他不能发起进攻，改进协议，将两步握手协议改为三步握手协议，这样，最初提出建议的指挥官必须确认对该建议的应答信息。假如信息没有丢失，并收到确认消息，则他必须将收到的确认信息告诉对方，从而完成三步握手协议。然而，这样他就无法知道消息是否被对方收到，因此，他不能发起进攻。那么现在采用四步握手协议会如何呢？结果仍是于事无补。

结论是：不存在使蓝军必胜的通信约定（协议）。

该结论可以用反证法证明，证明如下：假如存在某种协议，那么协议中最后一条信息要么是必要的，要么不是。如果不是，可以删除它，知道剩下的每条信息都是至关重要的。若最后一条消息没有安全到达目的地，则会怎样呢？刚才说过每条信息都是必要的，因此，若它丢了，则进攻不会如期进行。由于最后发出信息的指挥官永远无法确定该信息是否安全到达，所以他不会冒险发动攻击。同样，另一只蓝军也明白这个道理，所以也不会发动进攻。

28. IPV4 和 IPV6

目前的全球因特网所采用的协议族是 TCP/IP 协议族。IP 是 TCP/IP 协议族中网络层的协议，是 TCP/IP 协议族的核心协议。目前 IP 协议的版本号是 4(简称为 IPv4)，发展至今已经使用了 30 多年。
IPv4 的地址位数为 32 位，也就是最多有 2 的 32 次方的电脑可以联到 Internet 上。

近十年来由于互联网的蓬勃发展，IP 位址的需求量愈来愈大，使得 IP 位址的发放愈趋严格，各项资料显示全球 IPv4 位址可能在 2005 至 2008 年间全部发完。什么是 IPv6?

IPv6 是下一版本的互联网协议，也可以说是下一代互联网的协议，它的提出最初是因为随着互联网的迅速发展，IPv4 定义的有限地址空间将被耗尽，地址空间的不足必将妨碍互联网的进一步发展。为了扩大地址空间，拟通过 IPv6 重新定义地址空间。IPv6 采用 128 位地址长度，几乎可以不受限制地提供地址。按保守方法估算 IPv6 实际可分配的地址，整个地球的每平方米面积上仍可分配 1000 多个地址。在 IPv6 的设计过程中除了一劳永逸地解决了地址短缺问题以外，还考虑了在 IPv4 中解决不好的其它问题，主要有端到端 IP 连接、服务质量（QoS）、安全性、多播、移动性、即插即用等。

IPv6 静态路由分为三种：
配置直连 IPv6 静态路由（就是配置出接口）
配置递归 IPv6 静态路由（就是配置下一跳出口）
配置默认 IPv6 静态路由（就是配置零路由后面可以使下一跳或者出接口）

IPv6 地址类型可分为单播地址，组播地址和任播地址。
单播地址：可作为原地址和目的地址。单点发送。
组播地址和任播地址，目的地址可有多个。任播地址：标识网络中的一组主机，只可作为 IPv6 分组的目的地址，但当向一个任播地址发送 IP 分组时，只有该任播地址标识的任播组的某个成员收到该 IP 分组。组播地址是组播地址标识的多播组每个成员都会收到一个该 IP 分组的一个副本。

CS 即 Client/Server(客户机/服务器)结构。C/S 结构在技术上很成熟，它的主要特点是交互性强、具有安全的存取模式、网络通信量低、响应速度快、利于处理大量数据。但是该结构的程序是针对性开发，变更不够灵活，维护和管理的难度较大。通常只局限于小型局域网，不利于扩展。并且，由于该结构的每台客户机都需要安装相应的客户端程序，分布功能弱且兼容性差，不能实现快速部署安装和配置，因此缺少通用性，具有较大的局限性。要求具有一定专业水准的技术人员去完成。
BS 即 Browser/Server(浏览器/服务器)结构，就是只安装维护一个服务器(Server)，而客户端采用浏览器(Browse)运行软件。B/S 结构应用程序相对于传统的 C/S 结构应用程序是一个非常大的进步。B/S 结构的主要特点是分布性强、维护方便、开发简单且共享性强、总体拥有成本低。但数据安全性问题、对服务器要求过高、数据传输速度慢、软件的个性化特点明显降低，这些缺点是有目共睹的，难以实现传统模式下的特殊功能要求。例如：通过浏览器进行大量的数据输入或进行报表的应答、专用性打印输出都比较困难和不便。此外，实现复杂的应用构造有较大的困难。

29. 了解交换机、路由器、网关的概念
1）交换机

在计算机网络系统中，交换机是针对共享工作模式的弱点而推出的。交换机拥有一条高带宽的背部总线和内部交换矩阵。交换机的所有的端口都挂接在这条背部总线上，当控制电路收到数据包以后，处理端口会查找内存中的地址对照表以确定目的 MAC（网卡的硬件地址）的 NIC（网卡）挂接在哪个端口上，通过内部 交换矩阵迅速将数据包传送到目的端口。目的 MAC 若不存在，交换机才广播到所有的端口，接收端口回应后交换机会“学习”新的地址，并把它添加入内部地址表 中。

交换机工作于 OSI 参考模型的第二层，即数据链路层。交换机内部的 CPU 会在每个端口成功连接时，通过 ARP 协议学习它的 MAC 地址，保存成一张 ARP 表。在今后的通讯中，发往该 MAC 地址的数据包将仅送往其对应的端口，而不是所有的端口。因此，交换机可用于划分数据链路层广播，即冲突域；但它不 能划分网络层广播，即广播域。

交换机被广泛应用于二层网络交换，俗称“二层交换机”。

交换机的种类有：二层交换机、三层交换机、四层交换机、七层交换机分别工作在 OSI 七层模型中的第二层、第三层、第四层盒第七层，并因此而得名。

2）路由器

路由器（**Router**）是一种计算机网络设备，提供了路由与转送两种重要机制，可以决定数据包从来源端到目的端所经过 的路由路径（host 到 host 之间的传输路径），这个过程称为路由；将路由器输入端的数据包移送至适当的路由器输出端(在路由器内部进行)，这称为转送。路由工作在 OSI 模型的第三层——即网络层，例如网际协议。

路由器的一个作用是连通不同的网络，另一个作用是选择信息传送的线路。 路由器与交换器的差别，路由器是属于 OSI 第三层的产品，交换器是 OSI 第二层的产品(这里特指二层交换机)。

3）**网关**(Gateway)又称网间连接器、协议转换器。网关在网络层以上实现网络互连，是最复杂的网络互连设备，仅用于两个高层协议不同的网络互连。网关既可以用于广域网互连，也可以用于局域网互连。 网关是一种充当转换重任的计算机系统或设备。使用在不同的通信协议、数据格式或语言，甚至体系结构完全不同的两种系统之间，网关是一个翻译器。

30. 简述网卡的功能

网卡（Network Interface Card，简称 NIC），也称网络适配器，是电脑与局域网相互连接的设备。

网卡的功能主要有两个：一是将电脑的数据封装为帧，并通过网线（对无线网络来说就是电磁波）将数据发送到网络上去；二是接收网络上其它设备传过来的帧，并将帧重新组合成数据，发送到所在的电脑中。网卡能接收所有在网络上传输的信号，但正常情况下只接受发送到该电脑的帧和广播帧，将其余的帧丢弃。然后，传送到系统 CPU 做进一步处理。当电脑发送数据时，网卡等待合适的时间将分组插入到数据流中。接收系统通知电脑消息是否完整地到达，如果出现问题，将要求对方重新发送

WiFi 用的协议：802.1x 协议起源于 802.11 协议，后者是标准的无线局域网协议，802.1x 协议的主要目的是为了解决无线局域网用户的接入认证问题。

以太网的类型：同轴电缆以太网、光纤以太网、全双工以太网。进化：快速以太网、千兆以太网、万兆以太网
以太网设备：集线器、网桥、交换机

10M 以太网应遵循 5-4-3 规则。既在一个 10M 网络中，一共可以分为 5 个网段，其中用 4 个中继器连接，允许其中 3 个网段有设备，其他 2 个网段只是传输距离的延长。在 10BSE-

T 网络中，只允许级连 4 个 HUB。可以使用可堆叠的集线器，可堆叠集线器如 DE1824，可以堆叠 8 个，在逻辑上这 8 个 HUB 栈可以看作一个 HUB。或者可以使用交换机和 HUB 连接。避免了 5-4-3 规则的限制，而增加端口数。

MAC 地址类型：单播、广播、多播

开放式最短路径优先 Ospf 和路由信息协议 Rip 的区别

一、适用范围不同。

RIP 适用于中小网络，比较简单。没有系统内外、系统分区，边界等概念，用到不是分类的路由。
OSPF 适用于较大规模网络。它把自治系统分成若干个区域，通过系列内外路由的不同处理，区域内和区域间路由的不同处理方法，减少网络数据量大传输。

二、运行有区别。

RIP 运行时，首先向外发送请求报文，其他运行 RIP 的路由器收到请求后，马上把自己的路由表发送过去，在没收到请求时，会将路由删除，并广播自己新的路由表。
OSPF 要求每个路由器周期性的发送链路状态信息，使得区域内所有路由器最终都能形成一个跟踪网络链路状态的链路状态数据库。利用链路状态数据库，每一个路由器都可以以自己为“根”，建立一个最短路径优先树，用来描述以自己出发，到达每个目的网络所需的开销。

三、使用情况不同。

OSPF 占用的实际链路带宽比 RIP 少；OSPF 使用的 CPU 时间比 RIP 少；OSPF 适用的内存比 RIP 大；RIP 在网络上达到平衡用的时间比 OSPF 多。

网络协议的三个重要因素以及作用

(1) 语义。语义是解释控制信息每个部分的意义。它规定了需要发出何种控制信息，以及完成的动作与做出什么样的响应。
(2) 语法。语法是用户数据与控制信息的结构与格式，以及数据出现的顺序。
(3) 时序。时序是对事件发生顺序的详细说明。（也可称为“同步”）。
人们形象地把这三个要素描述为：语义表示要做什么，语法表示要怎么做，时序表示做的顺序。

传输介质

1、双绞线：屏蔽双绞线 STP (Shielded Twisted Pair)
无屏蔽双绞线 UTP (Unshielded Twisted Pair)
特点：容易受到外部高频电磁波的干扰，误码率高，但因为其价格便宜，且安装方便，既适于点到点连接，又可用于多点连接，故仍被广泛应用。
优缺点：成本低，密度高，节省空间，安装容易，高速率，抗干扰性一般，连接距离较短。
2、同轴电缆：50 W 同轴电缆　75 W 同轴电缆
特点：高带宽（高达 300～400Hz）、低误码率、性能价格比高，所以用在 LAN 中
优缺点：抗干扰性好，接入复杂

3、光缆

特点：直径小、重量轻；传输频带宽、通信容量大；抗雷电和电磁干扰性能好，无串音干扰，保密性好，误码率低。但光电接口的价格较昂贵。光纤被广泛用于电信系统铺设主干线。
优缺点：通信容量大，传输损耗小，抗干扰性好，保密性好，体积小重量轻，需要专用设备连接。
4、无线传输：短波通信/微波/卫星通信。
特点：频率高，频带范围宽，通信信道的容量大；信号所受工业干扰较小，传输质量高，通信比较稳定；不受地理环境的影响，建设投资少。
1、短波通信
优缺点：通信质量较差，速率低。
2、微波通信又分地面微波接力通信和卫星通信
地面微波接力通信
优缺点：信道容量大，传输质量高，投资少，相邻站点间直视，易受天气影响，保密性差。
卫星通信
优缺点：通信距离远，通信容量大，传播时延大 270ms。

香农定理指出，如果信息源的信息速率 R 小于或者等于信道容量 C，那么，在理论上存在一种方法可使信息源的输出能够以任意小的差错概率通过信道传输。
奈奎斯特采样定理解释了采样率和所测信号频率之间的关系。 阐述了采样率 fs 必须大于被测信号最高频率分量的两倍。 该频率通常被称为奈奎斯特频率 fN。

无线局域网上发送数据帧后要对方必须放回确认帧，而以太网就不需要对方发回确认帧
以太网相当于有线的局域网

子网掩码作用
一是屏蔽部分 IP 地址，区分网络标识和主机标识，解释 IP 地址是在局域网上还是在远程网络上；其次将一个大的 IP 网络划分为几个小的子网络。

子网划分的作用或好处：1.减少广播域。 2.有效利用和规划 IP。 3.安全考虑，方便管理

主机号不能全 0 或全 1

超网(supernetting)是与子网类似的概念--IP 地址根据子网掩码被分为独立的网络地址和主机地址。但是，与子网把大网络分成若干小网络相反，它是把一些小网络组合成一个大网络--超网。

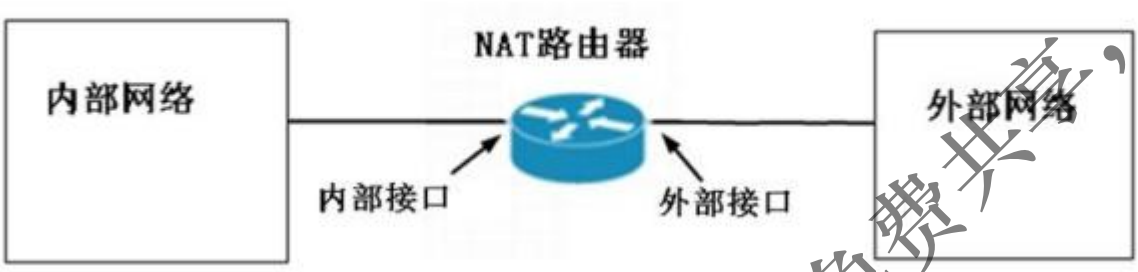
IP 地址决定终点，MAC 地址决定下一跳

vpn 的作用：
1）个人远程 VPN 访问到公司的内部网络；例如一般为企业内部员工出差，休假或者特殊情况下远离办公室的时候，又需要访问到公司的内部网络获取资源时。就可以通过 VPN 拨号到公司内部，
2）企业内部网络之间的通讯；例如：各大超市之间业务结算等
3）互联网公司多 IDC 机房之间的互联。

NAT 是网络地址转换，比方说有一万用户一千个 IP，每个用户一个 IP 肯定不够，但同时上线的用户可能只有 500，于是分给用户私网 IP，等他上线时 NAT 设备进行私网 IP 和公网 IP 的映射转化。

PAT 是端口地址转换，比方说有一万个用户而只有一百个 IP，同时上线的用户也许好几百个，这样 NAT 的方法也就满足不了了，于是只好把用户的私网 IP+端口映射到一个公网 IP+端口上，当然其他的用户也可以映射到相同的 IP 上，最后通过端口号来区分。

1、PAT 是 NAT 的一种，NAT 包括静态 NAT 和动态 NAT 还有就是 PAT。
2、静态 NAT 是一对一，一个内网地址对一个公网地址，动态 NAT 是一对多，可重复利用，PAT 是接口地址转换。



点到点网络的子网掩码最好是 252，点到点表明只有两台计算机故子网掩码为 255.255.255.252 二进制表示为[11111111.11111111.11111111.11111100]

透明传输是指不管所传数据是什么样的比特组合，都应当能够在链路上传送。当所传数据中的比特组合恰巧与某一个控制信息完全一样时，就必须采取适当的措施，使接收方不会将这样的数据误认为是某种控制信息。这样才能保证数据链路层的传输是透明的。

比特填充法具体地说，发送端的数据链路层遇到数据比特流中出现 5 个连续“1”的时候，它就自动在输出比特流中插入一个“0”接收端遇到 5 个输入比特为“1”，且后面紧接的是“0”时，自动将其删除。

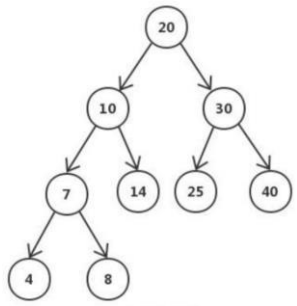
Internet 与 Intranet 区别
Internet 中文翻译叫因特网。因特网（Internet），又叫做国际互联网。它是由那些使用公用语言互相通信的计算机连接而成的全球网络。一旦你连接到它的任何一个节点上，就意味着您的计算机已经连入 Internet 网上了。因特网（Internet），是一个全球性的巨大的计算机网络体系，它把全球数百万个计算机网络，数亿台计算机主机连接起来，包含了无穷无尽的信息资源，向全世界提供信息服务。它是一组全球信息资源的总汇。

Intranet 称为企业内部网，或称内部网、内联网、内网，是一个使用与因特网同样技术的计算机网络，它通常建立在一个企业或组织的内部并为其成员提供信息的共享和交流等服务，例如万维网，文件传输，电子邮件等。

数据结构与算法

31. 什么是平衡二叉树

平衡二叉搜索树（Self-balancing binary search tree）又被称为 AVL 树（有别于 AVL 算法），且具有以下性质：它是一 棵空树或它的左右两个子树的高度差的绝对值不超过 1，并且左右两个子树都是一棵平衡二叉树，同时，平衡二叉树必定是二叉搜索树，反之则不一定。



平衡二叉树

33. 面向对象的对象是什么？对象和对象之间怎么作用？

面向对象就是：把数据及对数据的操作方法放在一起，作为一个相互依存的整体——对象。对同类对象抽象出其共性，形成类。类中的大多数数据，只能用本类的方法进行处理。类通过一个简单的外部接口与外界发生关系，对象与对象之间通过消息进行通信。程序流程由用户在使用中决定。

对象即为人对各种具体物体抽象后的一个概念，人们每天都要接触各种各样的对象，如手机就是一个对象。

静态地调用对方的接口(函数)

34. 通常设计一个好的算法应考虑一下目前：

正确性、可读性、健壮性、效率和低存储量需求

35. C 语言问题 局部变量能不能和全局变量重名

能，局部会屏蔽全局。要用全局变量，需要用“：：”

时间复杂度是指执行算法所需的计算工作量

空间复杂度是指执行这个算法所需的内存空间

动态规划程序设计是对解最优化问题的一种途径、一种方法，而不是一种特殊算法。不像搜索或数值计算那样，具有一个标准数学表达式和明确清晰的解题方法。动态规划程序设计往往是针对一种最优化问题，由于各种问题的性质不同，确定最优解得条件也互不相相同，因而动态规划的设计方法针对不同的问题，有特色的解题方法，而不存在一种万能的动态规划算法，可以解决各类最优化问题。例如二分查找树、背包问题、最短路径问题。

34. 数组 链表 的优劣

数组:

优点：使用方便，查询效率 比链表高，内存为一连续的区域缺点：大小固定，不适合动态存储，不方便动态添加

链表：

优点：可动态添加删除 大小可变

缺点：只能通过顺次指针访问，查询效率低

35. P 和 NP 问题

P 问题-多项式时间可以解决的问题
NP 问题-多项式时间可以验证的问题

NPC 问题-首先，它得是一个 NP 问题；然后，所有的 NP 问题都可以约化到它。 NP-Hard 问题-所有的 NP 问题都可以约化到它，不一定是 NP 问题

36. 栈是怎么作用的

(1)保存现场/上下文

(2)传递参数:汇编代码调用 c 函数时，需传递参数

(3)保存临时变量:包括函数的非静态局部变量以及编译器自动生成的其他临时变量。

37. Heap 和 stack 的概念和区别

1.heap 是堆，stack 是栈。2.stack 的空间由操作系统自动分配和释放，heap 的空间是手动申请和释放的，heap 常用 new 关键字来分配。3.stack 空间有限，heap 的空间是很大的自由区。在 Java 中，若只是声明一个对象，则先在栈内存中为其分配地址空间，若再 new 一下，实例化它，则在堆内存中为其分配地址。4.举例：数据类型 变量名；这样定义的东西在栈区。如：Object a =null; 只在栈内存中分配空间 new 数据类型();或者 malloc(长度); 这样定义的东西就在堆区如：Object b =new Object(); 则在堆内存中分配空间

C++中：

栈（stack）

由编译器自动分配释放管理。局部变量及每次函数调用时返回地址、以及调用者的环境信息（例如某些机器寄存器）都存放在栈中。新被调用的函数在栈上为其自动和临时变量分配存储空间。

堆（heap）

需要由程序员分配释放管理，若程序员不释放，程序结束时可能由 OS 回收。通常在堆中进行动态存储分配。

39. 算法的确定性是什么？与确定性相对应的算法是哪个？2

算法中每一条指令必须有确切的含义，读者理解时不会产生二义性。即对于相同的输入只能得出相同的输出。

相对应的是随机算法

40. 算法的定义？更严格的定义？

算法(algorithm)是对特定问题求解步骤的一种描述，它是指令的有限序列，其中每一条指令表示一个或多个操作。此外，一个算法还具有下列 5 个重要特性：

1) 有穷性

一个算法必须总是（对任何合法的输入值）在执行有穷步之后结束，且每一步都可在有穷时间内完成。

2) 确定性

算法中每一条指令必须有确切的含义，读者理解时不会产生二义性。即对于相同的输入只能得出相同的输出。

3) 可行性

一个算法是可行的，即算法中描述的操作都是通过已经实现的基本运算执行有限次来实现的。

4) 输入

一个算法有零个或多个的输入，这些输入取自于某个特定的对象的集合。

5) 输出

一个算法有一个或多个的输出，这些输出是同输入有着某种特定关系的量。

41. 有哪些排序算法，有什么优劣 3

快速排序是冒泡排序的改进版，是目前已知的最快的排序方法

优点：极快，数据移动少；

缺点：不稳定

缩小增量排序

由希尔在 1959 年提出，又称希尔排序(shell 排序)。

优点：快，数据移动少；

缺点：不稳定，d 的取值是多少，应取多少个不同的值，都无法确切知道，只能凭经验来取。

插入排序

优点：稳定，快；

缺点：比较次数不一定，比较次数越少，插入点后的数据移动越多，特别是当数据总量 k 庞大的时候，但用链表可以解决 这个问题。

选择排序

每一趟从待排序的数据元素中选出最小（或最大）的一个元素，顺序放在已排好序的数列的最后，直到全部待排序的数 据元素排完。优点：移动数据的次数已知（n-1 次）；

缺点：比较次数多。

冒泡排序

优点：稳定；

缺点：慢，每次只能移动相邻两个数据

排序方式	时间复杂度			空间复杂度	稳定性	复杂性
	平均情况	最坏情况	最好情况			
插入排序	$O(n^2)$	$O(n^2)$	$O(n)$	$O(1)$	稳定	简单
希尔排序	$O(n^{1.3})$			$O(1)$	不稳定	较复杂
冒泡排序	$O(n^2)$	$O(n^2)$	$O(n)$	$O(1)$	稳定	简单
快速排序	$O(n\log_2 n)$	$O(n^2)$	$O(n\log_2 n)$	$O(\log_2 n)$	不稳定	较复杂
选择排序	$O(n^2)$	$O(n^2)$	$O(n^2)$	$O(1)$	不稳定	简单
堆排序	$O(n\log_2 n)$	$O(n\log_2 n)$	$O(n\log_2 n)$	$O(1)$	不稳定	较复杂
归并排序	$O(n\log_2 n)$	$O(n\log_2 n)$	$O(n\log_2 n)$	$O(n)$	稳定	较复杂
基数排序	$O(d(n+r))$	$O(d(n+r))$	$O(d(n+r))$	$O(r)$	稳定	较复杂

42. 对全国人民的年龄进行排序应该用什么算法？2

基数排序

43. 一段代码 填空

44. c 语言程序题 2

45. 问程序输出结果

char a[]="a//gd/sjr/a";

问

strlen(a) 12

sizeof(a) 12*1=12 字节长度乘以字符串长度

strlen(a+3) 9*1=9

strlen 计算字符串的长度，以\0'为字符串结束标记。

sizeof 计算的则是分配的数组所占的内存空间的大小，不受里面存储的内容影响

char 占一个字节,int 占 4 个字节, double 占 8 个字节

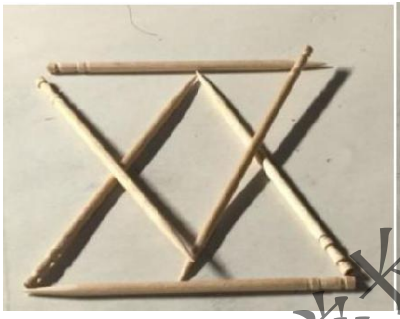
46. 小球重量 2

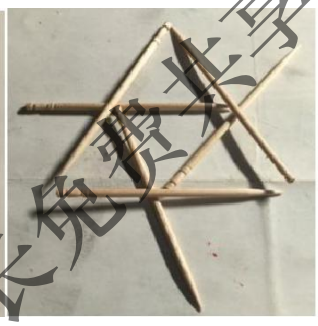
每次待定的 n 个球中取[(n+2)/3]个球放在天平的左边[(n+2)/3]个球放在右边

九个球，一个特别重，另外八个一样重，天平称几次找出重的球？

答：两次 第一次两边各三个 第二次判断再称

47. 6 根火柴排 4 个正三角形





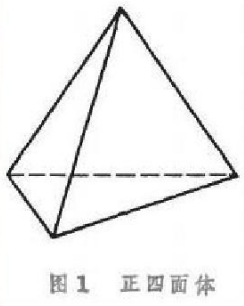


图 1 正四面体

48. 不递归的方法实现广度优先和深度优先

1.广度优先

（1）顶点 v 入队列。

（2）当队列非空时则继续执行，否则算法结束。

（3）出队列取得队头顶点 v；访问顶点 v 并标记顶点 v 已被访问。（4）查找顶点 v 的第一个邻接顶点 col。

（5）若 v 的邻接顶点 col 未被访问过的，则 col 入队列。

（6）继续查找顶点 v 的另一个新的邻接顶点 col，转到步骤（5）。

直到顶点 v 的所有未被访问过的邻接点处理完。转到步骤（2）。

2.深度优先

（1）栈 S 初始化；visited[n]=0；

（2）访问顶点 v；visited[v]=1；顶点 v 入栈 S

（3）while(栈 S 非空)

x=栈 S 的顶元素(不出栈)；

if(存在并找到未被访问的 x 的邻接点 w)

访问 w；visited[w]=1；

w 进栈;

else

x 出栈；

49. 迷宫算法
即深度优先搜索算法，迷宫的搜索者从起点开始将一只手扶着墙面前行，总能保证不会迷失并且找到迷宫中存在的出口。如果墙是相互连接的，那么他们可以被转换成一个回路或者环。那么摸着墙走就可以被看作是从一个圆环的起点进行至其终点。

---

typedef vs #define

#define 是 *C* 指令，用于为各种数据类型定义别名，与 **typedef** 类似，但是它们有以下几点不同：

- **typedef** 仅限于为类型定义符号名称，**#define** 不仅可以为类型定义别名，也能为数值定义别名，比如您可以定义 *1* 为 *ONE*。
- **typedef** 是由编译器执行解释的，**#define** 语句是由预编译器进行处理的。

---

getchar() & putchar() 函数

int getchar(void) 函数从屏幕读取下一个可用的字符，并把它返回为一个整数。这个函数在同一个时间内只会读取一个单一的字符。您可以在循环内使用这个方法，以便从屏幕上读取多个字符。
int putchar(int c) 函数把字符输出到屏幕上，并返回相同的字符。这个函数在同一个时间内只会输出一个单一的字符。您可以在循环内使用这个方法，以便在屏幕上输出多个字符。

---

gets() & puts() 函数

char *gets(char *s) 函数从 **stdin** 读取一行到 **s** 所指向的缓冲区，直到一个终止符或 *EOF*。
int puts(const char *s) 函数把字符串 *s* 和一个尾随的换行符写入到 **stdout**。

全局变量和局部变量的区别如下：
1. 作用域不同：全局变量的作用域为整个程序，而局部变量的作用域为当前函数或循环等
2. 内存存储方式不同：全局变量存储在全局数据区中，局部变量存储在栈区
3. 生命期不同：全局变量的生命期和主程序一样，随程序的销毁而销毁，局部变量在函数内部或循环内部，随函数的退出或循环退出就不存在了
4. 使用方式不同：全局变量在声明后程序的各个部分都可以用到，但是局部变量只能在局部使用。函数内部会优先使用局部变量再使用全局变量。

软件工程

50. 什么是软件工程？它目标和内容是什么？

软件工程 (Software Engineering) 是一门研究和应用如何以系统性的、规范化的、可定量的过程化方法去开发和维护软件，以及如何把经过时间考验而证明正确的管理技术和当前能够得到的最好的技术方法结合起来的学科。

软件工程目标：付出较低开发成本；达到要求的功能；取得较好的性能；开发的软件易于移植；只需较低的维护费用；能按时完成开发任务，及时交付使用；开发的软件可靠性高。

软件工程内容：包括开发技术和开发管理两个方面。

51. 什么是软件概要设计？该阶段的基本任务是什么？

把一个软件需求转换为软件表示时，首先设计出软件总的体系结构。称为概要设计或结构设计。

基本任务：

(1)设计软件系统结构

(2)进行数据结构及数据库的设计

(3)编写概要设计的文档

(4)评审

52. 软件维护有哪些内容？

（1）校正性维护。在软件交付使用后，一些隐含的错误在某些特定的使用环境下会暴露出来。为了识别和纠正错误，修改软件性能上的缺陷，应进行确定和修改错误的过程，这个过程就称为校正性维护。

（2）适应性维护。为了使应用软件适应计算机硬件、软件环境及数据环境的不断发生的变化而修改软件的过程称为适应性维护。

（3）完善性维护。为增加软件功能、增强软件性能、提高软件运行效率而进行的维护活动称为完善性维护。

（4）预防性维护。为了提高软件的可维护性和可靠性而对软件进行的修改称为预防性维护。

53. 什么是需求分析？需求分析阶段的基本任务是什么？

需求分析：开发人员准确地理解用户的要求，进行细致的调查分析，将用户非形式的需求陈述转化为完整的需求定义，再由需求定义转换到相应的需求规格说明的过程。

基本任务：

(1)问题识别

(2)分析与综合，导出软件的逻辑模型

(3)编写文档

软件需求包括三个不同的层次：业务需求、用户需求和功能需求（也包括非功能需求）。

1．业务需求（business requirement）反映了组织机构或客户对系统、产品高层次的目标要求，它们在项目视图与范围文档中予以说明。

2．用户需求(user requirement) 文档描述了用户使用产品必须要完成的任务，这在使用实例（use case）文档或方案脚本说明中予以说明。

3. 功能需求(functional requirement)定义了开发人员必须实现的软件功能，使得用户能完成他们的任务，从而满足了业务需求。

54. 单元测试有哪些内容?

单元测试主要针对模块的以下五个基本特征进行测试：

(1)模块接口

(2)局部数据结构

(3)重要的执行路径

(4)错误处理

(5)边界条件

55. 软件设计的基本原理包括哪些内容？

(1)模块化：模块是软件的组成部分，是具有独立功能且可命名的一段程序，所有模块组成整体，可以满足问题的要求。
(2)抽象：认识复杂现象过程所使用权的工具，只考虑事物本质的共同特性，忽略细节和其它因素。通过抽象确定组成软件的过程实体。
(3)信息隐蔽：将模块实现自身功能的细节与数据"隐蔽"起来。模块间仅交换为完成系统功能所必须的信息。
(4)模块独立性：每个模块只完成系统要求的独立的子功能。

56. 详细设计有哪几种描述方法？

详细设计的描述方法有图形、表格和语言，其中图形常用结构化程序流程图、盒图和PAD(问题分析图)为描述工具，语言常用过程设计语言(PDL)来作为工具。

57. 单元测试中采用什么方法?

测试的方法是为被测试模块编写驱动模块和桩模块来实现被测试单元的可运行。通过驱动模块来模拟被测试模块的上级调用模块，以上级模块调用被测模块的格式驱动被测模块，接收被测模块的测试结构并输出。桩模块则用来代替被测试模块所调用的模块。它的作用是返回被测模块所需的信息。

58. 什么是软件配置管理?什么是基线?

软件配置管理，简称 SCM(Software Configuration Management)，是指一组管理整个软件生存期各阶段中变更的活动。软件配置管理技术可以使软件变更所产生的错误达到最小并最有效地提高生产率。
基线：是软件生存期中各开发阶段的一个特定点，它的作用是把开发各阶段工作的划分更加明确化，使本来连续的工作在这些点上断开，以便于检查与肯定阶段成果。

59. 简述文档在软件工程中的作用？

文档在软件工程中的作用如下：
(1)提高软件开发过程的能见度
(2)提高开发效率
(3)作为开发人员阶段工作成果和结束标志
(4)记录开发过程的有关信息便于使用与维护；
(5)提供软件运行、维护和培训有关资料；
(6)便于用户了解软件功能、性能。

60. 软件工程三要素之间的关系？

软件质量是软件工程的生命线，软件工程以质量保证为基础。
质量管理促进了过程的改进，创造了许多行之有效的软件开发方法和工具。
软件工程采用层次化的方法，每个层次都包括**过程、方法、工具三要素。**
方法支撑过程和工具，过程和工具又促进方法学的研究。

61. 请简述可行性研究有哪些步骤？

复查系统规模和目标；研究当前的系统；导出新系统的高层逻辑模型；重新定义问题；导出和评价可供选择的解决方案；推荐可行的行动方案；草拟开发计划；书写文档并提交审查。

62. 什么是软件可维护性？常见的软件维护活动有哪几类？

软件可维护性的定义：软件能够被理解、校正、适应及增强功能的容易程度；
常见的软件维护活动有：改正型、适应型、完善型、预防型。

63. CMMI 全称是什么?分为哪几个级别？

CMMI 全称是软件能力成熟度模型；
从无序到有序的进化分成 5 个级别，分别为：

1 初始级；2 可重复级；3 已定义级；4 已管理级；5 优化级。

64. 什么是数据字典?简述数据字典与数据流图的关系。

数据字典是关于数据的信息的集合，对数据流程图中的各个元素做完整的定义与说明，是数据流程图的补充工具；

数据字典与数据流图的关系：数据流图和数据字典共同构成系统的逻辑模型，没有数据字典数据流图就不严格，然而没有数据流图数据字典也难于发挥作用。 数据流图和对数据流图中每个元素的精确定义放在一起，才能共同构成系统的规格说明。

65. 软件生命周期划分为哪几个阶段？

软件生命周期分为三个时期八个阶段：

软件定义：问题定义、可行性研究；

软件开发：需求分析、概要设计、详细设计、编码、测试；

软件运行：软件维护

66. 简述三种面向对象模型的主要功能？

①对象模型：表示了静态的结构化的系统数据性质，描绘了系统的静态结构，从客观世界实体的对象关系角度来描绘对象。

②动态模型：该模型描述了系统的控制结构，它表示了瞬间的、行为化的系统控制性质，它关心的是系统的控制及操作的执行顺序，它从对象的事件和状态的角度出发，表现了对象的交互行为。

③功能模型：表示变化的系统“功能”性质，它指明系统应该“做什么”，因此功能模型更直接的反映了用户对目标系统的要求。

67. 详细设计的基本任务是什么？

详细设计的基本任务包括：为每个模块进行详细的算法设计；为模块内的数据结构进行设计；对数据库进行物理设计；其他设计；编写详细设计说明书；评审。

68. 简述提高可维护性的方法。

(1)建立明确的软件质量目标；

(2)利用先进的软件开发技术和工具；

(3)建立明确的质量保证工作；

(4)选择可维护的程序设计语言；

(5)改进程序文档。

69. 调试的目的是什么?调试有哪些技术手段?

调试则是在进行了成功的测试之后才开始的工作。调试的目的是确定错误的原因和位置，并改正错误，因此调试也称为纠错(Debug)。调试的技术手段有简单的调试方法、归纳法、演绎法和回溯法等。

70. 软件危机？有什么表现？

软件危机是指落后的软件生产方式无法满足迅速增长的计算机软件需求，从而导致软件开发与维护过程中出现一系列严重问题的现象。

软件危机的形成

1.硬件生产率大幅提高 生产能力过剩。

2. 软件生产随规模增大复杂度增大

3. 软件生产率很低

伴随计算机的普及，整个社会对计算机应用的需求越来越大。

但软件的生产却还沿用"手工作坊"的生产方式，人工编程生产。生产效率仅提高了几倍。生产能力极其低下。

4. 硬、软件供需失衡

社会大量需求，生产成本高，生产过程控制复杂，生产效率低等等因素构成软件生产的恶性循环。由此产生"软件危机"。

5. 矛盾引发"软件危机"

软件危机是指在计算机软件的开发和维护过程中所遇到的一系列严重问题。

为了研究、解决软件危机，诞生了一门新兴学科--软件工程学。它把软件作为工程对象，从技术措施和组织管理两个方面来研究、解决软件危机

软件危机的具体体现

1. 软件开发成本难以预测
2. 软件开发成本难以控制

投资一再追加，令人难于置信。

3. 用户对产品功能难以满足

开发人员和用户之间很难沟通、矛盾很难统一。

4. 软件产品质量无法保证

系统中的错误难以消除。软件是逻辑产品，质量问题很难以统一的标准度量，因而造成质量控制困难。

5. 软件产品难以维护

软件产品本质上是开发人员的代码化的逻辑思维活动，他人难以替代。除非是开发者本人，否则很难及时检测、排除系统故障。为使系统适应新的硬件环境，或根据用户的需要在原系统中增加一些新的功能，又有可能增加系统中的错误。

6. 软件缺少适当的文档资料

文档资料是软件必不可少的重要组成部分。

软件开发模型：是指软件开发全部过程、活动、和任务的结构框架。软件开发包括需求、设计、编码和测试等阶段，有时也包括维护阶段。

一：瀑布开发模型

瀑布开发模型也称生命周期法，是生命周期法中最常用的模型，它把开发周期分为六个阶段：软件计划，需求分析，软件设计，程序编码，软件测试，软件运行与维护阶段。瀑布开发模型适用于大型软件开发过程中人员的组织与管理，适用于大型软件的开发工具和开发方法，提高了大型软件的开发效率和质量，可以快速的生成软件，但也有一定的弊端，如果在测试阶段发现错误，会返回重做，有事可能会返回到更前面的工作。

（1）因为瀑布开发模型呈线性，所以在软件未测试完成之前，未与用户见面，可能会导致软件的偏差性，增加风险

（2）前面的错误没有发现，可能到软件的后期会造成错误的扩散，进而可能会导致整个软件项目开发失败

（3）在软件需求阶段，完全确定用户的所有需求是比较困难的，甚至可以说是不可能的

二：快速原型模型

原型是指模拟某种产品的原始模型，在其他产业中经常使用。软件开发中的原型是软件的一个早期可运行的版本，它反映了最终系统的重要特性。

快速原型模型又称原型模型，它是增量模型的另一种形式；它是在开发真实系统之前，构造一个原型，在该原型的基础上，逐渐完成整个系统的开发工作。快速原型模型的第一步是建造一个快速原型，实现客户或未来的用户与系统的交互，用户或客户对原型进行评价，进一步细化待开发软件的需求。通过逐步调整原型使其满足客户的要求，开发人员可以确定客户的真正需求是什么；第二步则在第一步的基础上开发客户满意的软件产品。

优点：克服瀑布模型的缺点，减少由于软件需求不明确带来的开发风险。

这种模型适合预先不能确切定义需求的软件系统的开发。

缺点：所选用的开发技术和工具不一定符合主流的发展；快速建立起来的系统结构加上连续的修改可能会导致产品质量低下。

使用这个模型的前提是要有一个展示性的产品原型，因此在一定程度上可能会限制开发人员的创新。

三：演化模型
演化模型又称变换模型，是在快速开发一个原型的基础上，是对在用户调用原型的过程中所反馈的建议和意见进行修改，对原有模型的改进版本，重复这一过程直到完成最终的软件产品

四：螺旋模型
螺旋模型结合了变换模型和瀑布开发模型，并且增加了风险分析，并且在原型的基础上上，由里到外，每一轮都增加制定计划，风险分析，客户评价等要求，直至完成最终的软件产品
优点：大型软件开发有较好的风险控制
缺点：需要风险评估经验，契约开发通常需要事先指定过程模型和发布产品，普及不如瀑布模型和快速模型

五：喷泉模型
喷泉模型对生命周期和软件复用等多种活动提供了集成支持，主要**支持面向对象**的开发。“喷泉”一词本身就体现了迭代和无间隙。在软件的某个部分重复的进行修改，相关功能在每次迭代中加入系统中。无间隙是指在开发过程中对分析，设计和编码等没有确定的界限
优点：无缝，可同步开发，提高开发效率，节省开发时间，适应面向对象软件
缺点：可能随时附加各种信息、需求与资料，需要严格管理文档，审核难度加大

六：V 模型
V 模型是基于软件测试的软件开发模型，对每一次的测试进行改进，形成新版本，最后形成最终的软件产品
V 模型规定了一些测试级别
（1）**单元测试** 主要针对设计代码过程中存在的错误的测试 比如：输入信息与输出结果的匹配，边界值的问题
（2）**集成测试** 主要针对软件设计过程的测试 比如：软件与用户的接口问题，软件单元与程序各部分之间的接口
（3）**系统测试** 在概要设计阶段，主要针对系统的完整性，运行情况 比如：系统在运行情况中是否达到了预期的效果
（4）**验收测试** 主要由业务专家或者用户进行验收，确认产品符合用户的真正需求

七：增量模型
增量模型是融合了瀑布开发模型和原型迭代，它一开始只开发一个“核心产品”，即可以实现主要功能的产品，然后在一次次迭代中，增加新的量，每一次发布都可形成一个可操作的新版本，形成的最终软件，拆卸以后可得到最初开发的核心产品，它引入了一个新的概念，增量包。

增量模型的**优点**是人员分配灵活，在实现最初产品的时候所需人员较少，在增量增多的时候可合理的增加人员。尤其是在最初的时候设计一个核心产品，实现基本的功能，可对用户实现一个定心的作用。**缺点**是在增量包内容有交叉的时候，要对全局做系统的分析。主要用于将功能细化，需求经常变化的软件开发。

比较几种模型的优缺点？
瀑布模型：优点是简单，使用广泛，适合需求明确的项目；缺点是实际上需求在变动，用户在一次交付时才能知道软件性能。原型模型：优点是建立快速模型挖掘用户需求，适合难定需求的项目，减少项目风险。缺点是没有瀑布模型那样简单。时间盒模型：优点是固定框架可短时间内开发大量功能项目，可灵活重组构成新系统。缺点是人力资源为代价，缩短交付时间。

51. 设计模式懂嘛，简单举个例子？

答：设计模式（Design pattern）是一套被反复使用、多数人知晓的、经过分类编目的、代码设计经验的总结。

适配器模式(Adapter Pattern)：将一个接口转换成客户希望的另一个接口，使接口不兼容的那些类可以一起工作，其别名为包装器(Wrapper)。适配器模式既可以作为类结构型模式，也可以作为对象结构型模式。

在适配器模式中，我们通过增加一个新的适配器类来解决接口不兼容的问题，使得原本没有任何关系的类可以协同工作。

根据适配器类与适配者类的关系不同，适配器模式可分为对象适配器和类适配器两种，在对象适配器模式中，适配器与适配者之间是关联关系；在类适配器模式中，适配器与适配者之间是继承（或实现）关系。

例如，讲中文的人同讲英文的人对话时需要一个翻译，用直流电的笔记本电脑接交流电源时需要一个电源适配器，用计算机访问照相机的 SD 内存卡时需要一个读卡器等。

52. 设计模式

设计模式（Design Pattern）是一套被反复使用、多数人知晓的、经过分类的、代码设计经验的总结。

使用设计模式的目的：为了代码可重用性、让代码更容易被他人理解、保证代码可靠性。设计模式使代码编写真正工程化；设计模式是软件工程的基石脉络，如同大厦的结构一样。设计模式分为三种类型，共 23 种。

创建型模式：单例模式、抽象工厂模式、建造者模式、工厂模式、原型模式。

结构型模式：适配器模式、桥接模式、装饰模式、组合模式、外观模式、享元模式、代理模式。

行为型模式：模版方法模式、命令模式、迭代器模式、观察者模式、中介者模式、备忘录模式、解释器模式（Interpreter 模式）、状态模式、策略模式、职责链模式(责任链模式)、访问者模式。

53. 体系结构问题

计算机体系结构通常包含的系统元素有：计算机软件、计算机硬件、人员、数据库、文档和过程。其中，软件是程序、数据库和相关文档的集合，用于实际所需要的的逻辑方法、过程或控制；硬件是提供计算能力的电子设备和提供外部世界功能的电子器械设备（例如传感器、马达、水泵等）；人员是硬件和软件的用户操作者；数据库是通过软件访问的大型的、有组织的信息集合；文档是描述系统使用方法的手册、表格、图形和其他描述性信息；过程是一系列步骤，它们定义每个系统元素的特定使用方法或系统驻留的过程性语句

54. 开发中常用的软件

eclipse vs

55. 介绍结构优化设计

结构优化设计 (optimum structural design)在给定约束条件下，按某种目标(如重量最轻、成本最低、刚度最大等)求出最好的设计方案，曾称为结构最佳设计或结构最优设计，相对于“结构分析”而言，又称“结构综合”；如以结构的重量最小为目标，则称为最小重量设计。

52. α测试和β测试

α、β、λ常用来表示软件测试过程中的三个阶段，α是第一阶段，一般只供内部测试使用；β是第二个阶段，已经消除了软件中大部分的不完善之处，但仍有可能还存在缺陷和漏洞，一般只提供给特定的用户群来测试使用；λ是第三个阶段，此时产品已经相当成熟，只需在个别地方再做进一步的优化处理即可上市发行。

α测试是由一个用户**在开发环境下进行的测试，**也可以是开发机构内部的用户在模拟实际操作环境下进行的测试。软件在一个自然设置状态下使用。开发者坐在用户旁边，随时记下错误情况和使用中的问题。这是在受控制的环境下进行的测试，α测试的目的是评价软件产品的 FLURPS(即功能、局域化、可使用性、可靠性、性能和支持)。尤其注重产品的界面和特色。α测试人员是除开产品开发人员之外首先见到产品的人，他们提出的功能和修改意见是特别有价值的。α测试可以从软件产品编码结束之时开始，或在模块（子系统）测试完成之后开始，也可以在确认测试过程中产品达到一定的稳定和可靠程序之后再开始。有关的手册（草稿）等应事先准备好。

β测试是由软件的多个用户在一个或多个用户的**实际使用环境下进行的测试**。这些用户是与公司签定了支持产品预发行合同的外部客户，他们要求使用该产品，并愿意返回有关错位错误信息给开发者。与α测试不同的是，开发者通常不在测试现场。因而，β测试是在开发者无法控制的环境下进行的软件现场应用。在β测试中，由用户记下遇到的所有问题，包括真实的以及主观认定的，定期向开发者报告，开发者在综合用户的报告之后，做出修改，最将软件产品交付给全体用户使用。β测试主要衡量产品的 FLURPS。着重于产品的支持性，包括文档、客户培训和支持产品生产能力。只有当α测试达到一定的可靠程度时，才能开始β测试。由于它处在整个测试的最后阶段，不能指望这时发现主要问题。同时，产品的所有手册文本也应该在此阶段完全定稿。

由于β测试的主要目标是测试可支持性，所以β测试应尽可能由主持产品发行的人员来管理。

54. 完全测试有可能吗

测试的覆盖率几乎不可能达到 100%，也就是说，软件测试不能穷举所有的测试用例，不能将程序中所有的路径都测试一遍，因为对于多数软件系统，由于其复杂性和规模，测试用例数或程序路径数会是一个非常大的数据。不能完成 100%的测试，也就不可能将所有的缺陷发现出来，因此测试总是存在风险的。如果有充足的时间不断地进行测试，总是可以找到更多的缺陷。

57. 项目计划 软件计划是什么

软件项目计划（Software Project Planning）是一个软件项目进入系统实施的启动阶段，主要进行的工作包括：确定详细的项目实施范围、定义递交的工作成果、评估实施过程中主要的风险、制定项目实施的时间计划、成本和预算计划、人力资源计划等。

58. 软件的生命周期

主要有需求分析、软件设计、程序编码、软件测试、运行维护。
软件的产生直到报废的生命周期，周期内有问题定义、可行性分析、总体描述、系统设计、编码、调试和测试、验收和运行、维护升级到废弃等阶段，这种按时间分程的思想方法是软件工程中的一种思想原则，即按部就班、逐步推进，每个阶段都要有定义、工作、审查、形成文档以供交流或备查，以提高软件质量。

模块化程序设计是指在进行程序设计时将一个大程序按照功能划分为若干小程序模块，每个小程序模块完成一个确定的功能，并在这些模块之间建立必要的联系，通过模块的互相协作完成整个功能的程序设计方法。

59. 黑盒测试和白盒测试

一、黑盒测试

软件的**黑盒测试**意味着测试要在软件的接口处进行。这种方法是把测试对象看做一个黑盒子,测试人员完全不考虑程序内部的逻辑结构和内部特性，只依据程序的需求规格说明书,检查程序的功能是否符合它的功能说明。因此黑盒测试又叫功能测试。

(1)黑盒测试的**优点**：适用于功能测试、可用性测试及可接受性测试;对照说明书测试程序功能;可测试长的、复杂的程序的工作逻辑，易被理解。
(2)黑盒测试的**缺点**：不可能进行完全的、毫无遗漏的输入测试，有一些软件 Bug 或人为设置的故障通过黑盒测试是无法检测出来的。正是因为黑盒测试的测试数据来自规格说明书，这一方法的主要缺点是它依赖于规格说明书的正确性。实际上，人们并不能保证规格说明书完全正确。如在规格说明书中规定了多余的功能，或是漏掉了某些功能，这对于黑盒测试来说是完全无能为力的。

测试方法：等价类划分法、边界值分析法、错误推测法、因果图法、判定表驱动法、正交试验设计法、功能图法、场景法等。

二、白盒测试

软件的**白盒测试**是对软件的过程性细节做细致的检查。这种方法是把测试对象看做一个打开的盒子，它允许测试人员利用程序内部的逻辑结构及有关信息，设计或选择测试用例，对程序的所有逻辑路径进行测试，通过在不同点检查程序状态，确定实际状态是否与预期的状态一致。因此白盒测试又称为结构测试。白盒测试主要是想对程序模块进行检查。
白盒测试法的覆盖标准有逻辑覆盖、循环覆盖和基本路径测试。其中逻辑覆盖包括语句覆盖、判定覆盖、条件覆盖、判定/条件覆盖、条件组合覆盖和路径覆盖。六种覆盖标准发现错误的能力呈由弱到强的变化：
1.语句覆盖每条语句至少执行一次。
2.判定覆盖每个判定的每个分支至少执行一次。
3.条件覆盖每个判定的每个条件应取到各种可能的值。
4.判定/条件覆盖同时满足判定覆盖条件覆盖。
5.条件组合覆盖每个判定中各条件的每一种组合至少出现一次。
6.路径覆盖使程序中每一条可能的路径至少执行一次。
优点
(1)迫使测试人员去仔细思考软件的实现。
(2)可以检测代码中的每条分支和路径
(3)揭示隐藏在代码中的错误。
(4)对代码的测试比较彻底。
(5)让软件最优化。
缺点
(1)昂贵。
(2)无法检测代码中遗漏的路径和数据敏感性错误。
(3)不验证规格的正确性。

60. 主要软件开发方法

一、结构化设计方法

结构化设计，亦称 SD(Structured Design)，是一种面向数据流的设计方法，采用自顶向下、逐层分解的方法，建立系统的处理流程。

结构化设计的目的：使程序的结构尽可能反映要解决的问题的结构。
结构化设计的任务：把需求分析得到的数据流图 DFD 等变换为系统结构图（SC）。

基本思想是:根据结构化分析（SA）方法中的数据流图建立一个良好的模块结构图(例如SC 图或软件层次方框图)；运用模块化的设计原理控制系统的复杂性，即设计出模块相对独立的,模块结构图深度,宽度都适当的，单入口单出口的，单一功能的模块结构的软件结构图或软件层次方框图。此方法提供了描述软件系统的工具，提出了评价模块结构图质量的标准，即模块之间的联系越松散越好，而模块内各成分之间的联系越紧凑越好。

模块独立性强必须做到高内聚低耦合。
● 耦合：模块之间联系的紧密程度，耦合度越高模块的独立性越差。耦合度从低到高的次序为：非直接耦合、数据耦合、标记耦合、控制耦合、外部耦合、公共耦合、内容耦合。
● 内聚是指内部各元素之间联系的紧密程度，内聚度越低模块的独立性越差。内聚度从低到高依次是：偶然内聚、逻辑内聚、瞬时内聚、过程内聚、通信内聚、顺序内聚、功能内聚。

结构化分析(Structured Analysis，简称 SA 法)是面向数据流的需求分析方法，是 70 年代由 Yourdon,Constaintine 及 DeMarco 等人提出和发展，并得到广泛的应用。
结构化分析方法的基本思想是“分解”和“抽象”。
分解：是指对于一个复杂的系统，为了将复杂性降低到可以掌握的程度，可以把大问题分解成若干小问题，然后分别解决。
抽象：分解可以分层进行，即先考虑问题最本质的属性，暂把细节略去，以后再逐层添加细节，直至涉及到最详细的内容，这种用最本质的属性表示一个自系统的方法就是“抽象”。

二、Jackson 方法
Jackson 方法是一种面向数据结构的开发方法。
JSP(JacksonStructure Programming)方法是以数据结构为驱动的，适合于小规模的项目。JSP 方法首先描述问题的输入/输出数据结构，分析其对应性，然后推出相应的程序结构，从而给问题的软件过程描述。
JSD 方法是 JSP 方法的扩展，是一个完整的系统开发方法。首先建立现实世界的模型，再确定系统的功能需求，对需求的描述特别强调操作之间的时序性。它是以事件作为驱动的，是一种基于进程的开发方法，所以适用于时序特别较强的系统，包括数据处理系统和一些实时控制系统。

三、原型方法
原型方法比较适合于用户需求不清、需求经常变化的情况。当系统规模不是很大也不太复杂时，采用该方法比较好。

四、面向对象方法
面向对象方法正是以对象作为最基本的元素，它也是分析问题、解决问题的核心。面向对象方法包括面向对象分析、面向对象设计和面向对象实现。
UML 是面向对象的标准建模语言，通过统一的语义和符号表示，使各种方法的建模过程和表示统一起来，现已成为面向对象建模的工业标准。

61. 数据流程图（DFD）
数据流程图是在系统分析员在系统设计阶段，对实际构建的系统分析综合后，提取逻辑模型的一个过程，它更关注于过程内数据的处理，而把具体处理数据的物理过程，物理分布忽略。实际上，最初始的数据流程图标准图元只有四个！实体，过程，数据流，数据的存储。
第一步，画子系统的输入输出
第二步，画子系统的内部

第三步，画加工的内部
第四步，画子加工的分解图
第五步，对数据流图和加工编号

图3-2 飞机机票预订系统

62. 软件体系结构定义

软件体系结构是具有一定形式的结构化元素，即构件的集合，包括处理构件、数据构件和连接构件。处理构件负责对数据进行加工，数据构件是被加工的信息，连接构件把体系结构的不同部分组合连接起来。

编译原理和离散数学

63. 确定性有限机 非确定性的

64. 永真式和永假式

是永真式,永真式一定是可满 足式,可满足式未必是永真式。

设 A 为任一命题公式，若 A 在它的各种赋值下取值均为假，则称 A 是永假式。永假式又称为矛盾式

61. 四色定理的内容？解决方法？

四色定理又称四色猜想、四色问题，是世界三大数学猜想之一。四色定理是一个著名的数学定理，通俗的说法是：每个平面地图都可以只用四种颜色来染色，而且没有两个邻接的区域颜色相同。

1976 年借助电子计算机证明了四色问题，问题也终于成为定理，这是第一个借助计算机证明的定理。四色定理的本质就是在平面或者球面无法构造五个或者五个以上两两相连的区域。

62. 离散里面的图怎么去掉环

不知道

63. 如果月亮是方的，那 2+2=4 正确吗

正确

64. 离散中闭包原理

子函数可以使用父函数中的局部变量，这种行为就叫做闭包。离散数学中，一个关系 R 的闭包是指加上最小数目的有序偶而形成的具有自反性、对称性或传递性的新的有序偶集，此集就是关系 R 的闭包。

65. 图形中的矢量和标量

概念区别：标量只用数字表示量的大小，矢量还要用方向表示
运算法则区别：矢量用平行四边形法则或者三角形法则
正负号区别：矢量表示方向的正反向
表达式区别：比如力的表达式要考虑矢量方向，正负号加进表达式中

66. 优化

优化的目的是为了产生更高效的代码。
优化遵循的原则：等价原则，有效原则，合算原则
常用的优化技术：删除公共子表达式，复写传播，删除无用代码，以下三种涉及循环：代码外提，强度削弱，删除归纳变量

局部优化

基本块入口语句：1、程序的第一个语句 2、能由条件语句或者无条件语句转到的语句 3、紧跟在条件转移语句后面的语句

循环优化

活跃变量，指 A 的值要从 p 开始的某通路上被引用

综合能力

67. 大学写过多少行代码

几个项目 balabal

68. 大学项目，经验，心得 3

69. 微博 微信的区别

微信上，用户之间是对话关系，微信普通用户之间，需要互加好友，这构成了对等关系。而微博普通用户之间则不需要互加好友，双方的关系并非对等，而是多向度错落，一对多。

微信是私密空间内的闭环交流，而微博是开放的扩散传播。一个向内，一个向外；一个私密，一个公开；一个注重交流，一个注重传播。

微信用户主要是双方同时在线聊天，而微博则是差时浏览信息，用户各自发布自己的微博，粉丝查看信息并非同步，而是刷新查看所关注对象此前发布的信息。这种同时与差时也决定了微信与微博的功能与内容之差。

67. 安卓和 Ios 的区别 2

1、两者运行机制不同：IOS 采用的是沙盒运行机制，安卓采用的是虚拟机运行机制。

2、两者后台制度不同：IOS 中任何第三方程序都不能在后台运行；安卓中任何程序都能在后台运行，直到没有内存才会关闭。

3、IOS 中用于 UI 指令权限最高，安卓中数据处理指令权限最高。

iphone 沙盒机制解释：每个 iOS 应用都被限制在“沙盒”中，沙盒相当于一个加了仅主人可见权限的文件夹，及时在应用程序安装过程中，系统为每个单独的应用程序生成它的主目录和一些关键的子目录。苹果对沙盒有几条限制:

1. 应用程序在自己的沙盒中运作，但是不能访问任何其他应用程序的沙盒；
2. 应用之间不能共享数据，沙盒里的文件不能被复制到其他应用程序的文件夹中，也不能把其他应用文件夹复制到沙盒中；
3. 苹果禁止任何读写沙盒以外的文件，禁止应用程序将内容写到沙盒以外的文件夹中；
4. 沙盒目录里有三个文件夹：

Documents——存储应用程序的数据文件，存储用户数据或其他定期备份的信息；
Library 下有两个文件夹，Caches 存储应用程序再次启动所需的信息，Preferences 包含应用程序的偏好设置文件，不可在这更改偏好设置；
temp 存放临时文件即应用程序再次启动不需要的文件。

安卓虚拟机机制解释：android 本身不是为触摸屏打造的，所以所有的应用都是运行在一个虚拟的环境中，由底层传输数据到虚拟机中，再由虚拟机传递给用户 UI，任何程序都就可以轻松访问其他程序文件。

Ios 假后台是因为在后台挂起几分钟后进程就会被杀掉

private -只有当前的类才能访问该字段或方法。protected -仅此类的当前类和子类（有时还包括相同包装的类）将有权访问该字段或方法。public -任何类都可以引用该字段或调用该方法。假定这些关键字被用作类定义中字段或方法声明的一部分。

68. 毕业论文的知识点 参考文献
69. 分布式计算机系统是由多台计算机组成的系统，系统中各机无主次之分，各机通过通信交换信息，资源为所有用户所共享，若干台计算机可协作完成一个共同的任务。

70. new 和 malloc 的区别？

1、malloc 与 free 是 C++/C 语言的标准库函数，new/delete 是 C++的运算符。它们都可用于申请动态内存和释放内存。

2、对于非内部数据类型的对象而言，光用 malloc/free 无法满足动态对象的要求。对象在创建的同时要自动执行构造函数，对象在消亡之前要自动执行析构函数。

3、由于 malloc/free 是库函数而不是运算符，不在编译器控制权限之内，不能够把执行构造函数和析构函数的任务强加于 malloc/free。因此 C++语言需要一个能完成动态内存分配和初始化工作的运算符 new，以一个能完成清理与释放内存工作的运算符 delete。注意 new/delete 不是库函数。

4、C++程序经常要调用 C 函数，而 C 程序只能用 malloc/free 管理动态内存。

5、new 可以认为是 malloc 加构造函数的执行。new 出来的指针是直接带类型信息的。而 malloc 返回的都是 void 指针。

71. 解释 C++中静态函数和静态变量？

(1)类静态数据成员在编译时创建并初始化；在该类的任何对象建立之前就存在，不属于任何对象，而非静态类成员变量则是属于对象所有的。类静态数据成员只有一个拷贝，为所有此类的对象所共享。

(2)类静态成员函数属于整个类，不属于某个对象，由该类所有对象共享。

1、static 成员变量实现了同类对象间信息共享。

2、static 成员类外存储，求类大小，并不包含在内。

3、static 成员是命名空间属于类的全局变量，存储在 data 区的 rw 段。

4、static 成员只能类外初始化。

5、可以通过类名访问（无对象生成时亦可），也可以通过对象访问。

1、静态成员函数的意义，不在于信息共享，数据沟通，而在于管理静态数据成员，完成对静态数据成员的封装。

2、静态成员函数只能访问静态数据成员。原因：非静态成员函数，在调用时 this 指针时被当作参数传进。而静态成员函数属于类，而不属于对象，没有 this 指针。

66.STL 库用过吗？常见的 STL 容器有哪些？算法用过哪几个？

答：STL 包括两部分内容：容器和算法。（重要的还有融合这二者的迭代器）

容器，即存放数据的地方。比如 array 等。

在 STL 中，容器分为两类：序列式容器和关联式容器。

序列式容器，其中的元素不一定有序，但都可以被排序。如：vector、list、deque、stack、queue、heap、priority_queue、slist；

关联式容器，内部结构基本上是一颗平衡二叉树。所谓关联，指每个元素都有一个键值和一个实值，元素按照一定的规则存放。如：RB-tree、set、map、multiset、multimap、hashtable、ash_set、hash_map、hash_multiset、hash_multimap。

下面各选取一个作为说明。

vector：它是一个动态分配存储空间的容器。区别于 c++中的 array，array 分配的空间是静态的，分配之后不能被改变，而 vector 会自动重分配（扩展）空间。

set：其内部元素会根据元素的键值自动被排序。区别于 map，它的键值就是实值，而 map 可以同时拥有不同的键值和实值。

算法，如排序，复制……以及个容器特定的算法。这点不用过多介绍，主要看下面迭代器的内容。

迭代器是 STL 的精髓，我们这样描述它：迭代器提供了一种方法，使它能够按照顺序访问某个容器所含的各个元素，但无需暴露该容器的内部结构。它将容器和算法分开，好让这二者独立设计。

动态链接和虚拟存储的关系？

古典密码编码方法归根结底主要有两种，即置换和代换。

把明文中的字母重新排列，字母本身不变，但其位置改变了，这样编成的密码称为**置换**密码。最简单的置换密码是把明文中的字母顺序倒过来，然后截成固定长度的字母组作为密文。

代换密码则是将明文中的字符替代成其他字符。

分类是数据挖掘的一种非常重要的方法。分类的概念是在已有数据的基础上学会一个分类函数或构造出一个分类模型（即我们通常所说的分类器(Classifier)）。该函数或模型能够把数据库中的数据纪录映射到给定类别中的某一个，从而可以应用于数据预测。总之，分类器是数据挖掘中对样本进行分类的方法的统称，包含决策树、逻辑回归、朴素贝叶斯、神经网络等算法。

1.决策树分类器

提供一个属性集合，决策树通过在属性集的基础上作出一系列的决策，将数据分类。这个过程类似于通过一个植物的特征来辨认植物。可以应用这样的分类器来判定某人的信用程度，比如，一个决策树可能会断定“一个有家、拥有一辆价值在 1.5 万到 2.3 万美元之间的轿车、有两个孩子的人”拥有良好的信用。决策树生成器从一个“训练集”中生成决策树。SGI 公司的数据挖掘工具 MineSet 所提供的可视化工具使用树图来显示决策树分类器的结构，在图中，每一个决策用树的一个节点来表示。图形化的表示方法可以帮助用户理解分类算法，提供对数据的有价值的观察视角。生成的分类器可用于对数据的分类。

2. 选择树分类器

选择树分类器使用与决策树分类器相似的技术对数据进行分类。与决策树不同的是，选择树中包含特殊的选择节点，选择节点有多个分支。比如，在一棵用于区分汽车产地的选择树中的一个选择节点可以选择马力、汽缸数目或汽车重量等作为信息属性。在决策树中，一个节点一次最多可以选取一个属性作为考虑对象。在选择树中进行分类时，可以综合考虑多种情况。选择树通常比决策树更准确，但是也大得多。选择树生成器使用与决策树生成器生成决策树同样的算法从训练集中生成选择树。MineSet 的可视化工具使用选择树图来显示选择树。树图可以帮助用户理解分类器，发现哪个属性在决定标签属性值时更重要。同样可以用于对数据进行分类。

3. 证据分类器

证据分类器通过检查在给定一个属性的基础上某个特定的结果发生的可能性来对数据进行分类。比如，它可能作出判断，一个拥有一辆价值在 1.5 万到 2.3 万美元之间的轿车的人有 70 %的可能是信用良好的，而有 30 %的可能是信用很差。分类器在一个简单的概率模型的基础上，使用最大的概率值来对数据进行分类预测。与决策树分类器类似，生成器从训练集中生成证据分类器。MineSet 的可视化工具使用证据图来显示分类器，证据图由一系列描述不同的概率值的饼图组成。证据图可以帮助用户理解分类算法，提供对数据的深入洞察，帮助用户回答像“如果... 怎么样" 一类的问题。同样可以用于对数据进行分类。

67.const 知道吗？解释其作用。

答：

1.const 修饰类的成员变量，成员常量不能被修改。

2.const 修饰函数承诺在本函数内部不会修改类内的数据成员，不会调用其它非 const 成员函数。

3.如果 const 构成函数重载，const 对象只能调用 const 函数，非 const 对象优先调用非 const 函数。

4.const 函数只能调用 const 函数。非 const 函数可以调用 const 函数。

5.类体外定义的 const 成员函数，在定义和声明处都需要 const 修饰符。

68.解释下封装、继承和多态？（面向对象三个特征）

答：

一、封装：

封装是实现面向对象程序设计的第一步，封装就是将数据或函数等集合在一个个的单元中（我们称之为类）。

封装的意义在于保护或者防止代码（数据）被我们无意中破坏。

二、继承：

继承主要实现重用代码，节省开发时间。

子类可以继承父类的一些东西。

三、多态

多态：同一操作作用于不同的对象，可以有不同的解释，产生不同的执行结果。在运行时，可以通过指向基类的指针，来调用实现派生类中的方法。

多态的实现方式：分为方法的重载和重写,重载是指方法的名和返回类型相同,但是参数不同,它是一种编译时多态.
重写是指子类继承父类的方法,但是在子类自己类体里边又写了一个方法, 跟父类中的方法名,返回类型,参数列表都完全一样,但是方法体不同,也就是说它有自己的实现方式.这就是重写，它是一种运行时多态.

69.指针和引用的区别？

答：

1. 指针是一个变量，只不过这个变量存储的是一个地址，指向内存的一个存储单元；而引用仅是个别名；
2. 引用使用时无需解引用(*)，指针需要解引用；
3. 引用只能在定义时被初始化一次，之后不可变；指针可变；
4. 引用没有 const，指针有 const；
5. 引用不能为空，指针可以为空；
6. “sizeof 引用”得到的是所指向的变量(对象)的大小，而“sizeof 指针”得到的是指针本身的大小；
7. 指针和引用的自增(++)运算意义不一样；
8. 指针可以有多级，但是引用只能是一级（int **p；合法 而 int &&a 是不合法的）
9. 从内存分配上看：程序为指针变量分配内存区域，而引用不需要分配内存区域。

70.什么是内存泄漏？面对内存泄漏和指针越界，你有哪些方法？你通常采用哪些方法来避免和减少这类错误？

用动态存储分配函数动态开辟的空间，在使用完毕后未释放，结果导致一直占据该内存单元即为内存泄露。

使用的时候要记得指针的长度。

malloc 的时候得确定在那里 free.

对指针赋值的时候应该注意被赋值指针需要不需要释放.

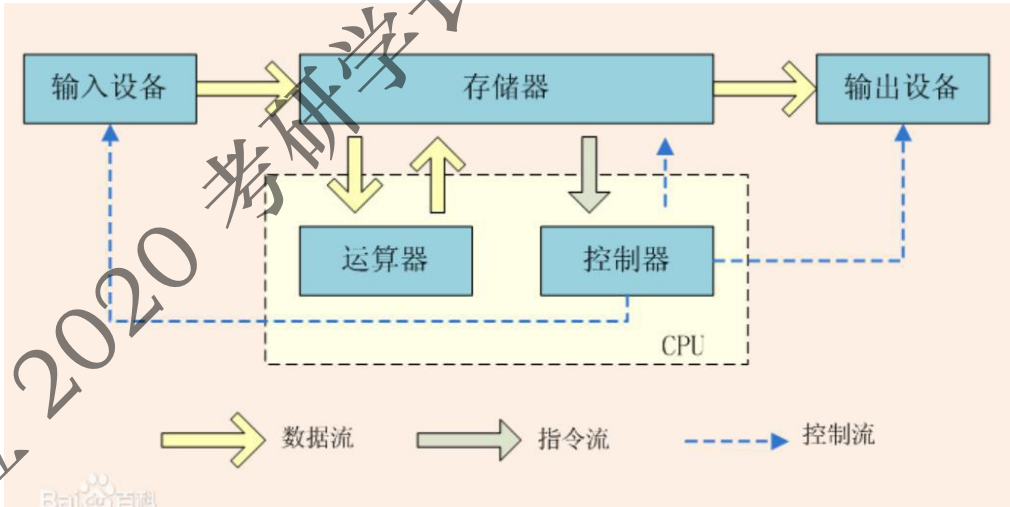
动态分配内存的指针最好不要再次赋值.

74. 摩尔定律是由英特尔（Intel）创始人之一戈登·摩尔（Gordon Moore）提出来的，其内容为：当价格不变时，集成电路上可容纳的元器件的数目，约每隔 18-24 个月便会增加一倍，性能也将提升一倍。换言之，每一美元所能买到的电脑性能，将每隔 18-24 个月翻一倍以上。这一定律揭示了信息技术进步的速度。

尽管这种趋势已经持续了超过半个世纪，摩尔定律仍应该被认为是观测或推测，而不是一个物理或自然法。预计定律将持续到至少 2015 年或 2020 年[1]。然而，2010 年国际半导体技术发展路线图的更新增长已经放缓在 2013 年年底，之后的时间里晶体管数量密度预计只会每三年翻一番。[1]

75.冯诺依曼体系结构

采用存储程序方式，指令和数据不加区别混合存储在同一个存储器中，数据和程序在内存中是没有区别的,它们都是内存中的数据,当 EIP 指针指向哪 CPU 就加载那段内存中的数据,如果是不正确的指令格式,CPU 就会发生错误中断. 在现在 CPU 的保护模式中,每个内存段都有其描述符,这个描述符记录着这个内存段的访问权限(可读,可写,可执行).这就变相的指定了哪些内存中存储的是指令哪些是数据)



MVC 和三层架构的区别

MVC：是一种框架设计方式（Model View Controller），是模型(model)－视图(view)－控制器(controller)的缩写。提高了程序的可维护性、可移植性、可扩展性与可重用性，降低了程序的开发难度。

1.模型(model)它是应用程序的主体部分，主要包括业务逻辑模块和数据模块。模型与数据格式无关，这样一个模型能为多个视图提供数据。由于应用于模型的代码只需写一 次就可以被多个视图重用，所以减少了代码的重复性。

2.视图(view) 用户与之交互的界面、在 web 中视图一般由 jsp,html 组成。

3.控制器(controller)接收来自界面的请求 并交给模型进行处理 在这个过程中控制器不做任何处理只是起到了一个连接的作用。

MVC 优点：
1.降低代码耦合性。在 MVC 模式中，三个层各施其职，所以如果一旦哪一层的需求发生了变化，就只需要更改相应的层中的代码而不会影响到其他层中的代码。
2.有利于分工合作。在 MVC 模式中，由于按层把系统分开，那么就能更好的实现开发中的分工。网页设计人员可进行开发视图层中的 JSP，而对业务熟悉的人员可开发业务层，而其他开发人员可开发控制层。
3.有利于组件的重用。如控制层可独立成一个能用的组件，表示层也可做成通用的操作界面。可以为一个模型在运行时同时建立和使用多个视图。
MVC 缺点：
1.增加了系统结构和实现的复杂性。对于简单的界面，严格遵循 MVC，使模型、视图与控制器分离，会增加结构的复杂性，并可能产生过多的更新操作，降低运行效率。
2.视图层展示依赖与模型层，视图层需要很少的字段，但是模型层全部提供，性能上有一定影响

注意：mvc 由于市场的需求，演变出三层框架

三层框架：按照高内聚低耦合的思想，形成一种标准的分层方式。分为三层：界面层（UserInterface layer）、业务逻辑层（BusinessLogic Layer）、数据访问层（Dataaccess layer）。
界面层：主要对用户的请求接受，以及数据的返回，为客户端提供应用程序的访问。
业务逻辑层：主要负责对数据层的操作。也就是说把一些数据层的操作进行组合。
数据访问层：也称为是持久层其功能主要是负责数据库的访问，可以访问数据库系统、二进制文件、文本文档或是 XML 文档。完成 CRUD 的操作，数据交互及落地。

形式化方法内容

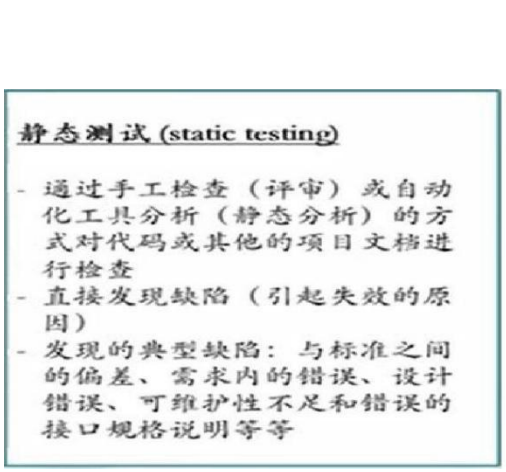
形式化方法的本质是基于数学的方法来描述目标软件系统属性的一种技术。不同的形式化方法的数学基础是不同的，有的以集合论和一阶谓词演算为基础（如 Z 和 VDM），有的则以时态逻辑为基础。形式化方法需要形式化规约说明语言的支持。
这样的形式化方法提供了一个框架，可以在框架中以系统的而不是特别的方式刻划、开发和验证系统。如果一个方法有良好的数学基础，那么它就是形式化的，典型地以形式化规约语言给出。这个基础提供一系列精确定义的概念，如：一致性和完整性，以及定义规范的实现和正确性。形式化方法模型的主要活动是生成计算机软件形式化的数学规格说明。形式化方法使软件开发人员可以应用严格的数学符号来说明、开发和验证基于计算机的系统。这种方法的一个变型是净室软件工程（cleanroom software engineering），这一软件工程方法目前已应用于一些软件开发机构。

数据和信息之间是相互联系的。数据是反映客观事物属性的记录，是信息的具体表现形式。数据经过加工处理之后，就成为信息；而信息需要经过数字化转变成数据才能存储和传输。
从信息论的观点来看，描述信源的数据是信息和数据冗余之和，即：数据=信息+数据冗余。
数据是数据采集时提供的，信息是从采集的数据中获取的有用信息。

编译执行过程？
词法分析，语法分析，语义分析和中间代码生成，优化，目标代码生成
对于 c 语言：源程序文件，预编译处理（cpp），编译程序本身，优化程序，汇编程序，链接程序，可执行文件

缓存模块举例：hibernate spring3 有缓存模块

81.软件测试
软件质量表现：1.用户需求　2.开发规范　3.隐含要求
软件质量：　　是软件产品的特性可以满足用户的功能、性能需求 的能力 。
软件可靠性：　　在特定环境下，在给定时间内无故障运行的概率
软件错误：　　软件没有实现其最终用户合理预期的功能要求



代码审查 （Code Review）三部曲：

代码互查（Peer Review）

代码走查 （Walk Through）

正式会议审查 （Inspection）

多任务：指在同一时间内执行多个任务（任务可以是函数或者方法），例如: 现在电脑安装的操作系统都是多任务操作系统，可以同时运行着多个软件。

多任务的执行方式：并发和并行

（1）并发：在一段时间内交替去执行任务。

并行：多个任务真正意义上一起执行。

多任务的实现方式：多进程与多线程

多进程可以完成多任务，每个进程就好比一家独立的公司，每个公司都各自在运营，每个进程也各自在运行，执行各自的任务。

互斥锁: 对共享数据进行锁定，保证同一时刻只能有一个线程去操作。

互斥锁是多个线程一起去抢，抢到锁的线程先执行，没有抢到锁的线程需要等待，等互斥锁使用完释放后，其它等待的线程再去抢这个锁。

面向服务（SOA）是一个组件模型，它将应用程序的不同功能单元（称为服务）进行拆分，并通过这些服务之间定义良好的接口和协议联系起来。接口是采用中立的方式进行定义的，它应该独立于实现服务的硬件平台、操作系统和编程语言。这使得构件在各种各样的系统中的服务可以以一种统一和通用的方式进行交互。

面向对象（OO）面向对象是相对于面向过程来讲的，面向对象方法，把相关的数据和方法组织为一个整体来看待，从更高的层次来进行系统建模，更贴近事物的自然运行模式。

面向过程(POP)是一种以过程为中心的编程思想。这些都是以什么正在发生为主要目标进行编程，不同于面向对象的是谁在受影响。与面向对象明显的不同就是封装、继承、类。就是分析出解决问题所需要的步骤，然后用函数把这些步骤一步一步实现，使用的时候一个一个依次调用就可以了。

比如拿学生早上起来这件事说明面向过程，粗略的可以将过程拟为：

(1)起床　　(2)穿衣　　(3)洗脸刷牙　　(4)去学校

而这 4 步就是一步一步地完成，它的顺序很重要，你只需要一个一个地实现就行了。而如果是用面向对象的方法的话，可能就只抽象出一个学生的类，它包括这四个方法，但是具体的顺序就不一定按照原来的顺序。

面向切面编程（AOP），可以说是 OOP（Object Oriented Programming，面向对象编程）的补充和完善。OOP 引入封装、继承、多态等概念来建立一种对象层次结构，用于模拟公共行为的一个集合。不过 OOP 允许开发者定义纵向的关系，但并不适合定义横向的关系，例如日志功能。日志代码往往横向地散布在所有对象层次中，而与它对应的对象的核心功

能毫无关系对于其他类型的代码，如安全性、异常处理和透明的持续性也都是如此，这种散布在各处的无关的代码被称为横切（cross cutting），在 OOP 设计中，它导致了大量代码的重复，而不利于各个模块的重用。

AOP 技术恰恰相反，它利用一种称为"横切"的技术，剖解开封装的对象内部，并将那些影响了多个类的公共行为封装到一个可重用模块，并将其命名为"Aspect"，即切面。所谓"切面"，简单说就是那些与业务无关，却为业务模块所共同调用的逻辑或责任封装起来，便于减少系统的重复代码，降低模块之间的耦合度，并有利于未来的可操作性和可维护性。

使用"横切"技术，AOP 把软件系统分为两个部分：核心关注点和横切关注点。业务处理的主要流程是核心关注点，与之关系不大的部分是横切关注点。横切关注点的一个特点是，他们经常发生在核心关注点的多处，而各处基本相似，比如权限认证、日志、事物。AOP 的作用在于分离系统中的各种关注点，将核心关注点和横切关注点分离开来。

1. 大数据、云计算和物联网的区别

大数据侧重于海量数据的存储、处理与分析，从海量数据中发现价值，服务于生产和生活;云计算本质上旨在整合和优化各种 IT 资源，并通过网络以服务的方式廉价提供给用户;物联网的发展目标是实现物物相连，应用创新是物联网发展的核心。

2. 大数据、云计算和物联网的联系

从整体上看，大数据、云计算和物联网这三者是相辅相成的。大数据根植于云计算，大数据分析的很多技术都来自于云计算，云计算的分布式和数据存储和管理系统(包括分布式文件系统和分布式数据库系统)提供了海量数据的存储和管理能力，分布式并行处理框架 MapReduce 提供了海量数据分析能力，没有这些云计算技术作为支撑，大数据分析就无从谈起。反之，大数据为云计算提供了“用武之地”，没有大数据这个“练兵场”，云计算技术再先进，也不能发挥它的应用价值。

物联网的传感器源源不断产生的大量数据，构成了大数据的重要来源，没有物联网的飞速发展，就不会带来数据产生方式的变革，即由人工产生阶段向自动产生阶段，大数据时代也不会这么快就到来。同时，物联网需要借助于云计算和大数据技术、实现物联网大数据的存储、分析和处理。

线性代数中特征值的现实应用

互联网，Google 的 PageRank，就是 对 www 链接关系的修正邻接矩阵的，主要特征向量的投影分量，给出了页面平分。也就是搜索排名，凭什么我靠前你靠后。

人像识别，我们把图像 A 看成矩阵，进一步看成线性变换矩阵，把这个训练图像的特征矩阵求出来(假设取了 n 个能量最大的特征向量)。用 A 乘以这个 n 个特征向量，得到一个 n 维矢量 a，也就是 A 在特征空间的投影。

还有聚类分析，信号处理等等。

图片为什么有的越放大越模糊，有的图片放大 500 倍也不失真？

主要看你的图片是哪种：有的是位图 有的是矢量图，矢量图不失真

矢量图:矢量图也叫面向对象绘图，是用数学方式描述的曲线及曲线围成的色块制作的图形，它们是在计算机内部中表示成一系列的数值而不是像素点，这些值决定了图形如何在屏幕上。用户所作的每一个图形，打印的每一个字母都是一个对象，每个对象都决定其外形的路径，一个对象与别的对象相互隔离，因此，可以自由地改变对象的位置、形状、大小和颜色。同时，由于这种保存图形信息的办法与分辨率无关，因此无论放大或缩小多少，都有一样平滑的边缘，一样的视觉细节和清晰度。矢量图形尤其适用于标志设计、图案设计、文字设计、版式设计等，它所生成文件也比位图文件要小一点。基于矢量绘画的软件有 CorelDRAW、Illustrator、Freehand 等。

位图:位图也叫像素图，它由像素或点的网格组成，与矢量图形相比，位图的图像更容易模拟照片的真实效果。其工作方式就像是用画笔在画布上作画一样。如果将这类图形放大到一定的程度，就会发现它是由一个个小方格组成的，这些小方格被称为像素点。一个像素点是图像中最小的图像元素。一幅位图图像包括的像素可以达到百万个，因此，位图的大

小和质量取决于图像中像素点的多少，通常说来，每平方英寸的面积上所含像素点越多，颜色之间的混合也越平滑，同时文件也越大。基于位图的软件有 Photoshop、Painter 等

监督学习（supervised learning）

从给定的训练数据集中学习出一个函数（模型参数），当新的数据到来时，可以根据这个函数预测结果。监督学习的训练集要求包括输入输出，也可以说是特征和目标。训练集中的目标是由人标注的。监督学习就是最常见的分类（注意和聚类区分）问题，通过已有的训练样本（即已知数据及其对应的输出）去训练得到一个最优模型（这个模型属于某个函数的集合，最优表示某个评价准则下是最佳的），再利用这个模型将所有的输入映射为相应的输出，对输出进行简单的判断从而实现分类的目的。也就具有了对未知数据分类的能力。监督学习的目标往往是让计算机去学习我们已经创建好的分类系统（模型）。监督学习是训练神经网络和决策树的常见技术。这两种技术高度依赖事先确定的分类系统给出的信息，对于神经网络，分类系统利用信息判断网络的错误，然后不断调整网络参数。对于决策树，分类系统用它来判断哪些属性提供了最多的信息。

常见的有监督学习算法：回归分析和统计分类。最典型的算法是 KNN 和 SVM。

无监督学习（unsupervised learning）

输入数据没有被标记，也没有确定的结果。样本数据类别未知，需要根据样本间的相似性对样本集进行分类（聚类，clustering）试图使类内差距最小化，类间差距最大化。通俗点将就是实际应用中，不少情况下无法预先知道样本的标签，也就是说没有训练样本对应的类别，因而只能从原先没有样本标签的样本集开始学习分类器设计。

非监督学习目标不是告诉计算机怎么做，而是让它（计算机）自己去学习怎样做事情。非监督学习有两种思路。第一种思路是在指导 Agent 时不为其指定明确分类，而是在成功时，采用某种形式的激励制度。需要注意的是，这类训练通常会置于决策问题的框架里，因为它的目标不是为了产生一个分类系统，而是做出最大回报的决定，这种思路很好的概括了现实世界，agent 可以对正确的行为做出激励，而对错误行为做出惩罚。

无监督学习的方法分为两大类：

(1) 一类为基于概率密度函数估计的直接方法：指设法找到各类别在特征空间的分布参数，再进行分类。

(2) 另一类是称为基于样本间相似性度量的简洁聚类方法：其原理是设法定出不同类别的核心或初始内核，然后依据样本与核心之间的相似性度量将样本聚集成不同的类别。

利用聚类结果，可以提取数据集中隐藏信息，对未来数据进行分类和预测。应用于数据挖掘，模式识别，图像处理等。

PCA 和很多 deep learning 算法都属于无监督学习。

两者的不同点

1. 有监督学习方法必须要有训练集与测试样本。在训练集中找规律，而对测试样本使用这种规律。而非监督学习没有训练集，只有一组数据，在该组数据集内寻找规律。

2. 有监督学习的方法就是识别事物，识别的结果表现在给待识别数据加上了标签。因此训练样本集必须由带标签的样本组成。而非监督学习方法只有要分析的数据集的本身，预先没有什么标签。如果发现数据集呈现某种聚集性，则可按自然的聚集性分类，但不予以某种预先分类标签对上号为目的。

3. 非监督学习方法在寻找数据集中的规律性，这种规律性并不一定要达到划分数据集的目的，也就是说不一定要“分类”。

这一点是比有监督学习方法的用途要广。 譬如分析一堆数据的主分量，或分析数据集有什么特点都可以归于非监督学习方法的范畴。

4. 用非监督学习方法分析数据集的主分量与用 K-L 变换计算数据集的主分量又有区别。后者从方法上讲不是学习方法。因此用 K-L 变换找主分量不属于无监督学习方法，即方法上不是。而通过学习逐渐找到规律性这体现了学习方法这一点。在人工神经元网络中寻找主分量的方法属于无监督学习方法。

何时采用哪种方法

简单的方法就是从定义入手，有训练样本则考虑采用监督学习方法；无训练样本，则一定不能用监督学习方法。但是，现实问题中，即使没有训练样本，我们也能够凭借自己的双眼，从待分类的数据中，人工标注一些样本，并把它们作为训练样本，这样的话，可以把条件改善，用监督学习方法来做。对于不同的场景，正负样本的分布如果会存在偏移（可能大的偏移，可能比较小），这样的话，监督学习的效果可能就不如用非监督学习了。

请求转发和重定向

1）Forward（请求转向）：服务器程序内部请求转向，这个特性允许前一个程序用于处理请求，而后一个程序用来返回响应。

2）Redirect（重定向）：服务端发送给客户端一个重定向的临时响应头，这个响应头包含重定向之后的 URL，客户端用新的 URL 重新向服务器发送一个新的请求。

3、本质区别

1）请求转发发生在服务器端，由服务器（比如 servlet）控制。

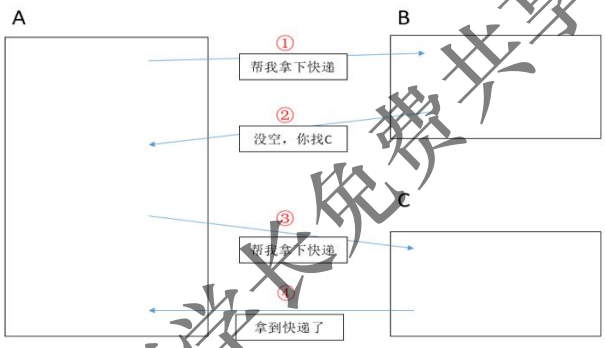
2）重定向发生在客户端，由客户（通常是浏览器）控制。

4、请求和响应次数

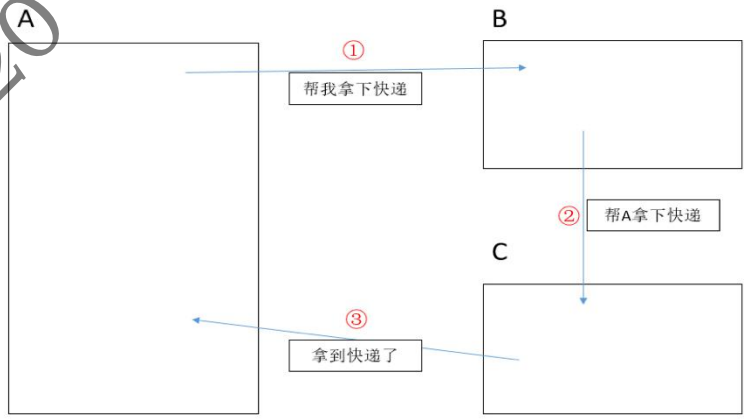
1）请求转发过程在同一个请求当中完成，只会返回一个响应。

2）重定向过程则发生在两个不同的请求中，会返回两个不同响应。

重定向



请求转发



POST 和 GET 都是向服务器提交数据，并且都会从服务器获取数据。

区别：

1、传送方式：get 通过地址栏传输，post 通过报文传输。

2、传送长度：get 参数有长度限制（受限于 url 长度），而 post 无限制
3、GET 和 POST 还有一个重大区别，简单的说：
GET 产生一个 TCP 数据包；POST 产生两个 TCP 数据包
长的说：
对于 GET 方式的请求，浏览器会把 http header 和 data 一并发送出去，服务器响应 200（返回数据）；
而对于 POST，浏览器先发送 header，服务器响应 100 continue，浏览器再发送 data，服 0 务器响应 200 ok（返回数据）。
也就是说，GET 只需要汽车跑一趟就把货送到了，而 POST 得跑两趟，第一趟，先去和服务器打个招呼“嗨，我等下要送一批货来，你们打开门迎接我”，然后再回头把货送过去。

对象名	功能	类型	作用域
request	向客户端请求数据	javax.servlet.ServletRequest	Request
response	封装了 jsp 产生的响应,然后被发送到客户端以响应客户的请求	javax.servlet.SrvletResponse	Page
pageContext	为 JSP 页面包装页面的上下文。管理对属于 JSP 中特殊可见部分中已经命名对象的该问	javax.servlet.jsp.PageContext	Page
session	用来保存每个用户的信息,以便跟踪每个用户的操作状态	javax.servlet.http.HttpSession	Session
application	应用程序对象	javax.servlet.ServletContext	Application
out	向客户端输出数据	javax.servlet.jsp.JspWriter	Page
config	表示 Servlet 的配置,当一个 Servlet 初始化时,容器把某些信息通过此对象传递给这个 Servlet	javax.servlet.ServletConfig	Page
page	Jsp 实现类的实例,它是 jsp 本身,通过这个可以对它进行访问	javax.lang.Object	Page
exception	反映运行的异常	javax.lang.Throwable	Page

一、cookie：
在网站中，http 请求是无状态的。也就是说即使第一次和服务器连接后并且登录成功后，第二次请求服务器依然不能知道当前请求是哪个用户。cookie 的出现就是为了解决这个问题，第一次登录后服务器返回一些数据（cookie）给浏览器，然后浏览器保存在本地，当该用户发送第二次请求的时候，就会自动的把上次请求存储的 cookie 数据自动的携带给服务器，服务器通过浏览器携带的数据就能判断当前用户是哪个了。cookie 存储的数据量有限，不同的浏览器有不同的存储大小，但一般不超过 4KB。因此使用 cookie 只能存储一些小量的数据。
二、session:

session 和 cookie 的作用有点类似，都是为了存储用户相关的信息。不同的是，cookie 是存储在本地浏览器，而 session 存储在服务器。存储在服务器的数据会更加的安全，不容易被窃取。但存储在服务器也有一定的弊端，就是会占用服务器的资源，但现在服务器已经发展至今，一些 session 信息还是绰绰有余的。

cookie 和 session 的区别

1、存储位置不同

cookie 的数据信息存放在客户端浏览器上。

session 的数据信息存放在服务器上。

2、存储容量不同

单个 cookie 保存的数据<=4KB，一个站点最多保存 20 个 Cookie。

对于 session 来说并没有上限，但出于对服务器端的性能考虑，session 内不要存放过多的东西，并且设置 session 删除机制。

3、存储方式不同

cookie 中只能保管 ASCII 字符串，并需要通过编码方式存储为 Unicode 字符或者二进制数据。

session 中能够存储任何类型的数据，包括且不限于 string，integer，list，map 等。

4、隐私策略不同

cookie 对客户端是可见的，别有用心的人可以分析存放在本地的 cookie 并进行 cookie 欺骗，所以它是不安全的。

session 存储在服务器上，对客户端是透明的，不存在敏感信息泄漏的风险。

5、有效期上不同

开发可以通过设置 cookie 的属性，达到使 cookie 长期有效的效果。

session 依赖于名为 JSESSIONID 的 cookie，而 cookie JSESSIONID 的过期时间默认为-1，只需关闭窗口该 session 就会失效，因而 session 不能达到长期有效的效果。

6、服务器压力不同

cookie 保管在客户端，不占用服务器资源。对于并发用户十分多的网站，cookie 是很好的选择。

session 是保管在服务器端的，每个用户都会产生一个 session。假如并发访问的用户十分多，会产生十分多的 session，耗费大量的内存。

pageContext 对象的范围只适用于当前页面范围，即超过这个页面就不能够使用了。所以使用 pageContext 对象向其它页面传递参数是不可能的。

request 对象的范围是指在一 JSP 网页发出请求到另一个 JSP 网页之间，随后这个属性就失效。

session 的作用范围为一段用户持续和服务器所连接的时间，但与服务器断线后，这个属性就无效。比如断网或者关闭浏览器。

application 的范围在服务器一开始执行服务，到服务器关闭为止。它的范围最大，生存周期最长。

PreparedStatement 和 Statement 的区别

1.Statement 主要用于执行静态的 SQL 语句，内容固定不变。（对数据库只执行一次性存取）

2.PreparedStatement 对象不仅包含了 SQL 语句，而且大多数情况下这个语句已经被预编译过，因而当其执行时，只需 DBMS 运行 SQL 语句，而不必先编译。当你需要执行 Statement 对象多次的时候，PreparedStatement 对象将会大大降低运行时间，当然也加快了访问数据库的速度。

3.PrepareStatement 这种转换也给你带来很大的便利，不必重复 SQL 语句的句法，而只需更改其中变量的值，便可重新执行 SQL 语句。选择 PreparedStatement 对象与否，在于相同句法的 SQL 语句是否执行了多次，而且两次之间的差别仅仅是变量的不同。如果仅仅执行了一次的话，它应该和普通的对象毫无差异，体现不出它预编译的优越性。
4.除了缓冲的问题之外，使用 PreparedStatement 对象,那就是安全性（预防 SQL 注入攻击 ）。

SQL 注入即是指 web 应用程序对用户输入数据的合法性没有判断或过滤不严，攻击者可以在 web 应用程序中事先定义好的查询语句的结尾上添加额外的 SQL 语句，在管理员不知情的情况下实现非法操作，以此来实现欺骗数据库服务器执行非授权的任意查询，从而进一步得到相应的数据信息。

JavaBean 就是使用 Java 语言开发的一个可重用的组件（遵循一定规则的普通 Java 类）
jsp 中使用 javabean 的好处如下：
1.提高代码的可复用性：对于通用的事务处理逻辑，数据库操作等都可以封装在 JavaBean 中，通过调用 JavaBean 的属性和方法可快速进行程序设计。
2.程序易于开发维护：实现逻辑的封装，使事务处理和显示互不干扰。
3.支持分布式运用：多用 JavaBean，尽量减少 java 代码和 html 的混编。

Servlet 是运行在 Web 服务器或应用服务器上的程序，它是作为来自 Web 浏览器或其他 HTTP 客户端的请求和 HTTP 服务器上的数据库或应用程序之间的中间层。
使用 Servlet，您可以收集来自网页表单的用户输入，呈现来自数据库或者其他源的记录，还可以动态创建网页。

jsp 和 servlet 的区别有以下几点 ：
Jsp 是 Servlet 的一种简化，使用 Jsp 只需要完成程序员需要输出到客户端的内容，Jsp 中的 Java 脚本如何镶嵌到一个类中，由 Jsp 容器完成。
1、jsp 经编译后就变成了 Servlet。
2、jsp 更擅长表现于页面显示,servlet 更擅长于逻辑控制。
3、Servlet 中没有内置对象，Jsp 中的内置对象都是必须通过 HttpServletResponse 对象以及 HttpServlet 对象得到。
4、而 Servlet 则是个完整的 Java 类，这个类的 Service 方法用于生成对客户端的响应。

Servlet 生命周期可被定义为从创建直到毁灭的整个过程。以下是 Servlet 遵循的过程：
- Servlet 通过调用 init () 方法进行初始化。
- Servlet 调用 service() 方法来处理客户端的请求。
- Servlet 通过调用 destroy() 方法终止（结束）。
- 最后，Servlet 是由 JVM 的垃圾回收器进行垃圾回收的

(1)**分类**。分类是找出数据库中的一组数据对象的共同特点并按照分类模式将其划分为不同的类，其目的是通过分类模型，将数据库中的数据项映射到某个给定的类别中。可以应用到涉及到应用分类、趋势预测中，如淘宝商铺将用户在一段时间内的购买情况划分成不同的类，根据情况向用户推荐关联类的商品，从而增加商铺 的销售量。

(2)**回归分析**。回归分析反映了数据库中数据的属性值的特性，通过函数表达数据映射的关系来发现属性值之间的依赖关系。它可以应用到对数据序列的预测及相 关关系的研究中去。在市场营销中，回归分析可以被应用到各个方面。如通过对本季度销售的回归分析，对下一季度的销售趋势作出预测并做出针对性的营销改变。

(3)**聚类**。聚类类似于分类，但与分类的目的不同，是针对数据的相似性和差异性将一组数据分为几个类别。属于同一类别的数据间的相似性很大，但不同类别之间数据的相似性很小，跨类的数据关联性很低。

(4)**关联规则**。关联规则是隐藏在数据项之间的关联或相互关系，即可以根据一个数据项的出现推导出其他数据项的出现。关联规则的挖掘过程主要包括两个阶 段：第一阶段为从海量原始数据中找出所有的高频项目组;第二极端为从这些高频项目组产生关联规则。关联规则挖掘技术已经被广泛应用于金融行业企业中用以预测客户的需求，各银行在自己的 ATM 机上通过捆绑客户可能感兴趣的信息供用户了解并获取相应信息来改善自身的营销。

过滤器：依赖于 servlet 容器。在实现上基于函数回调，可以对几乎所有请求进行过滤，但是缺点是一个过滤器实例只能在容器初始化时调用一次。使用过滤器的目的是用来做一些过滤操作，获取我们想要获取的数据，比如：在过滤器中修改字符编码；在过滤器中修改 HttpServletRequest 的一些参数，包括：过滤低俗文字、危险字符等
拦截器：依赖于 web 框架，在 SpringMVC 中就是依赖于 SpringMVC 框架。在实现上基于 Java 的反射机制，属于面向切面编程（AOP）的一种运用。由于拦截器是基于 web 框架的调用，因此可以使用 Spring 的依赖注入（DI）进行一些业务操作，同时一个拦截器实例在一个 controller 生命周期之内可以多次调用。但是缺点是只能对 controller 请求进行拦截，对其他的一些比如直接访问静态资源的请求则没办法进行拦截处理。
web 监听器是一种 Servlet 中的特殊的类，它们能帮助开发者监听 web 中的特定事件，比如 ServletContext,HttpSession,ServletRequest 的创建和销毁；变量的创建、销毁和修改等。可以在某些动作前后增加处理,实现监控。

Session 的钝化与活化
(一)钝化
当服务器正常关闭时,还存活着的 session(在设置时间内没有销毁) 会随着服务器的关闭被以文件(“SESSIONS.ser”)的形式存储在 tomcat 的 work 目录下,这个过程叫做 Session 的钝化。
(二)活化
当服务器再次正常开启时,服务器会找到之前的“SESSIONS.ser” 文件，从中恢复之前保存起来的 Session 对象，这个过程叫做 Session 的活化。

XMLHttpRequest 是 AJAX 的基础。XMLHttpRequest 用于在后台与服务器交换数据。这意味着可以在不重新加载整个网页的情况下，对网页的某部分进行更新。

过拟合：指为了得到一致假设而使假设变得过度严格。