概念问题

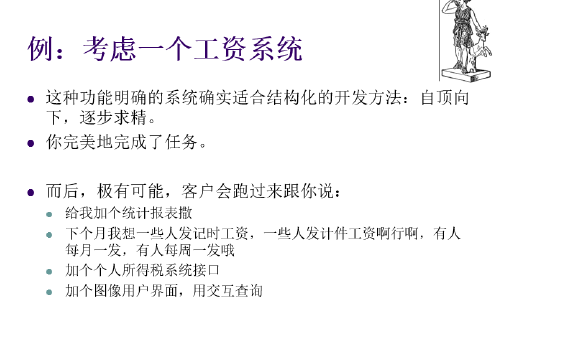
C++/数据结构

1、

简述你对“面向对象”和“面向过程”编程思想的认识与思考

用就可以了。

面向过程



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

面向对象是把构成问题事务分解成各个对象，建立对象的目的不是为了完成一个步骤，而是为了描叙某个事物在整个解决问题的步骤中的行为。

例如五子棋，面向过程的设计思路就是首先分析问题的步骤：1、开始游戏，2、黑子先走，3、绘制画面，4、判断输赢，5、轮到白子，6、绘制画面，7、判断输赢，8、返回步骤2，9、输出最后结果。把上面每个步骤用分别的函数来实现，问题就解决了。

而面向对象的设计则是从另外的思路来解决问题。整个五子棋可以分为 1、黑白双方，这两方的行为是一模一样的，2、棋盘系统，负责绘制画面，3、规则系统，负责判定诸如犯规、输赢等。第一类对象（玩家对象）负责接受用户输入，并告知第二类对象（棋盘对象）棋子布局的变化，棋盘对象接收到了棋子的i变化就要负责在屏幕上面显示出这种变化，同时利用第三类对象（规则系统）来对棋局进行判定。

可以明显地看出，面向对象是以功能来划分问题，而不是步骤。同样是绘制棋局，这样的行为在面向过程的设计中分散在了总多步骤中，很可能出现不同的绘制版本，因为通常设计人员会考虑到实际情况进行各种各样的简化。而面向对象的设计中，绘图只可能在棋盘对象中出现，从而保证了绘图的统一。

功能上的统一保证了面向对象设计的可扩展性。比如我要加入悔棋的功能，如果要改动面向过程的设计，那么从输入到判断到显示这一连串的步骤都要改动，甚至步骤之间的循序都要进行大规模调整。如果是面向对象的话，只用改动棋盘对象就行了，棋盘系统保存了黑白双方的棋谱，简单回溯就可以了，而显示和规则判断则不用顾及，同时整个对对象功能的调用顺序都没有变化，改动只是局部的。

再比如我要把这个五子棋游戏改为围棋游戏，如果你是面向过程设计，那么五子棋的规则就分布在了你的程序的每一个角落，要改动还不如重写。但是如果你当初就是面向对象的设计，那么你只用改动规则对象就可以了，五子棋和围棋的区别不就是规则吗？（当然棋盘大小好像也不一样，但是你会觉得这是一个难题吗？直接在棋盘对象中进行一番小改动就可以了。）而下棋的大致步骤从面向对象的角度来看没有任何变化。

当然，要达到改动只是局部的需要设计的人有足够的经验，使用对象不能保证你的程序就是面向对象，初学者或者很蹩脚的程序员很可能以面向对象之虚而行面向过程之实，这样设计出来的所谓面向对象的程序很难有良好的可移植性和可扩展性。

2、 ADT是什么？简述你对“数据抽象”和“信息隐藏”的认识

抽象数据类型(Abstract Data Type 简称ADT)是指一个数学模型以及定义在此数学模型上的一组操作。抽象数据类型需要通过固有数据类型（高级编程语言中已实现的数据类型）来实现。抽象数据类型是与表示无关的数据类型，是一个数据模型及定义在该模型上的一组运算。对一个抽象数据类型进行定义时，必须给出它的名字及各运算的运算符名，即函数名，并且规定这些函数的参数性质。一旦定义了一个抽象数据类型及具体实现，程序设计中就可以像使用基本数据类型那样，十分方便地使用抽象数据类型。

抽象数据类型通过类(class)实现

λ 程序设计语言对抽象数据类型的支持是指允许用户自定义具有如下特征的数据类型：

1. 模块封装：The representation of, and operations on, objects of the type are defined in a single

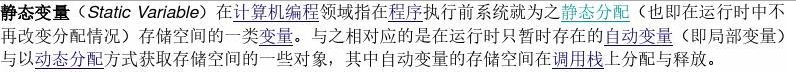
syntactic unit

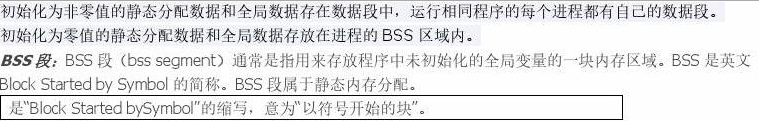
2. 信息隐蔽：The representation of objects of the type is hidden from the program units that use these

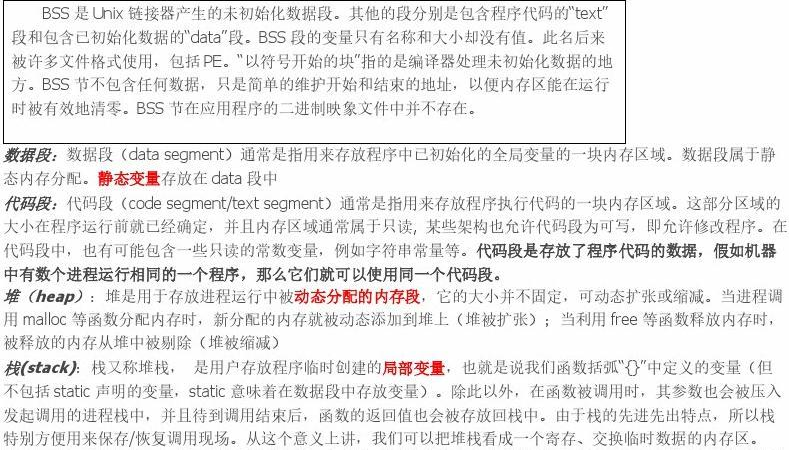
objects, so the only operations possible are those provided in the type's definition

3、const和static有什么作用？

const是一个C和C++语言的关键字，它限定一个变量不允许被改变，即只读。使用const在一定程度上可以提高程序的安全性和可靠性，也便于实现对此进行优化（如把只读对象放入ROM中）。const作为类型限定符，是类型的一部分。







4、友元关系的利与弊

如果将一个函数或一个类声明为另一个类的友元，那么它就可以直接存取这个类对象中的各种数据，而不必在意这些数据的封装级别，即无论是private的，protected的，还是public的，有钱同使，有难同当。

5、C++多态的实现

1. 用virtual关键字申明的函数叫做虚函数，虚函数肯定是类的成员函数。

2. 存在虚函数的类都有一个一维的虚函数表叫做虚表。类的对象有一个指向虚表开始的虚指针。虚表是和类对应的，虚表指针是和对象对应的。

3. 多态性是一个接口多种实现，是面向对象的核心。分为类的多态性和函数的多态性。

4. 多态用虚函数来实现，结合动态绑定。

5. 纯虚函数是虚函数再加上= 0。

6. 抽象类是指包括至少一个纯虚函数的类。

构造函数顺序：

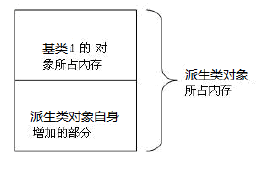
基类构造函数◊派生类构造函数

前面输出的结果是因为编译器在编译的时候，就已经确定了对象调用的函数的地址，要解决这个问题就要使用迟绑定（late binding）技术。当编译器使用迟绑定时，就会在运行时再去确定对象的类型以及正确的调用函数。而要让编译器采用迟绑定，就要在基类中声明函数时使用virtual关键字（注意，这是必须的，很多学员就是因为没有使用虚函数而写出很多错误的例子），这样的函数我们称为虚函数。一旦某个函数在基类中声明为virtual，那么在所有的派生类中该函数都是virtual，而不需要再显式地声明为virtual。

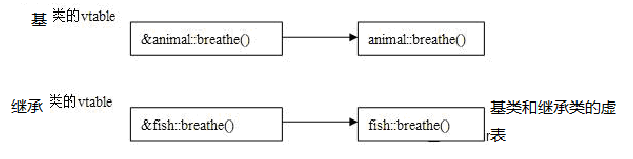
前面输出的结果是因为编译器在编译的时候，就已经确定了对象调用的函数的地址，要解决这个问题就要使用迟绑定（late binding）技术。当编译器使用迟绑定时，就会在运行时再去确定对象的类型以及正确的调用函数。而要让编译器采用迟绑定，就要在基类中声明函数时使用virtual关键字（注意，这是必须的，很多学员就是因为没有使用虚函数而写出很多错误的例子），这样的函数我们称为虚函数。一旦某个函数在基类中声明为virtual，那么在所有的派生类中该函数都是virtual，而不需要再显式地声明为virtual。

编译器在编译的时候，发现基类中有虚函数，此时编译器会为每个包含虚函数的类创建一个虚表（即vtable），该表是一个一维数组，在这个数组中存放每个虚函数的地址。

那么如何定位虚表呢？编译器另外还为每个类的对象提供了一个虚表指针（即vptr），这个指针指向了对象所属类的虚表。在程序运行时，根据对象的类型去初始化vptr



，从而让



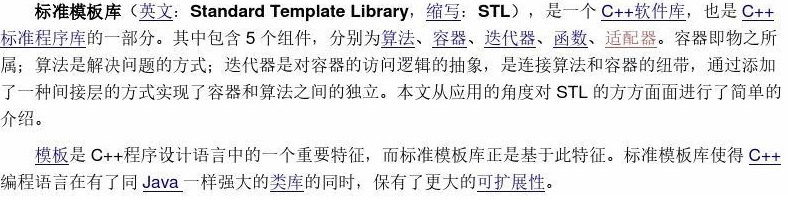
vptr正确的指向所属类的虚表，从而在调用虚函数时，就能够找到正确的函数。对于例1-2的程序，由于pAn实际指向的对象类型是fish，因此vptr指向的fish类的vtable，当调用pAn->breathe()时，根据虚表中的函数地址找到的就是fish类的breathe()函数。

那么虚表指针在什么时候，或者说在什么地方初始化呢？

答案是在构造函数中进行虚表的创建和虚表指针的初始化。还记得构造函数的调用顺序吗，在构造子类对象时，要先调用父类的构造函数，此时编译器只“看到了”父类，并不知道后面是否后还有继承者，它初始化父类对象的虚表指针，该虚表指针指向父类的虚表。当执行子类的构造函数时，子类对象的虚表指针被初始化，指向自身的虚表。对于例2-2的程序来说，当fish类的fh对象构造完毕后，其内部的虚表指针也就被初始化为指向fish类的虚表。在类型转换后，调用pAn->breathe()，由于pAn实际指向的是fish类的对象，该对象内部的虚表指针指向的是fish类的虚表，因此最终调用的是fish类的breathe()函数。

要注意：对于虚函数调用来说，每一个对象内部都有一个虚表指针，该虚表指针被初始化为本类的虚表。所以在程序中，不管你的对象类型如何转换，但该对象内部的虚表指针是固定的，所以呢，才能实现动态的对象函数调用，这就是C++多态性实现的原理。

6、STL是什么？组成部分和核心作用



标准模板库于1994年2月年正式成为ANSI/ISO C++的一部份，它的出现，促使C++程序员的思维方式更朝向泛型编程（generic program）发展。

7、阐述C++在什么情况下必须进行运算符重载。

实现两个自定义对象的加法操作时,C++本身提供的运算符无法满足要求.

8、为什么说“继承是C++面向对象的一个主要特征之一”，请做一下简要说明。

避免重复造轮子

9、 请说明函数模板(Function Template)和函数模板实例化(function-template specification)的区别和联系。

函数模板实例化

在函数模板为每个类型时首先调用中，编译器创建一个实例化。 每个实例化是为该类型的该模板化功能的版本。 在中，此函数为类型时，使用此实例化将调用。 如果您有几个相同的实例化，即使在不同的模块，因此，只有该实例化的一个副本在可执行文件将结果。

函数参数将所有参数的函数模板允许和参数，对该参数不依赖于模板参数的位置。

函数模板可以通过声明与特定类型的模板显式实例化作为参数。

C++中提供了函数模板，实际上是建立一个通用函数，其函数类型和形参类型不具体指定，用一个虚拟的类型来代表，这个通用函数就成为函数模板。使用模板的好处就是对于那些函数体相同的函数都可以用这个模板来代替，而不必去定义每个具体的函数去实现。下面通过一个简单的具体例子（比较两个数的大小）来说明：

#include <iostream>

using namespace std;

template<class T> //模板声明，T为类型参数

T Max(T a, T b) //定义一个通用函数，用T作虚拟的类型名

{

if (a>b)

{

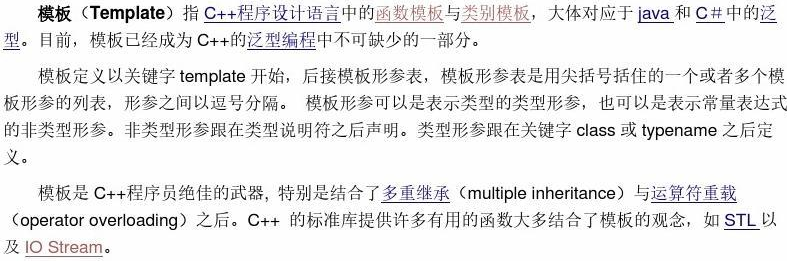
return a;

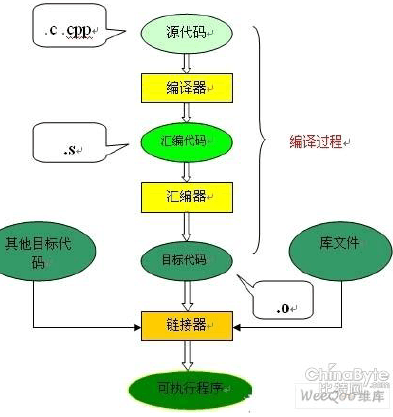
}

else

return b;

}





模板实例化（template instantiation ）是指在编译或链接时生成函数模板或类模板的具体实例源代码。ISO C++定义了两种模板实例化方法：隐式实例化（当使用实例化的模板时自动地在当前代码单元之前插入模板的实例化代码）、显式实例化（直接声明模板实例化）。

10、编译和链接的过程

源文件的编译过程包含两个主要阶段，而它们之间的转换是自动的。第一个阶段是预处理阶段，在正式的编译阶段之前进行。预处理阶段将根据已放置在文件中的预处理指令来修改源文件的内容。#include指令就是一个预处理指令，它把头文件的内容添加到.cpp文件中还有其他许多预处理指令

这个在编译之前修改源文件的方式提供了很大的灵活性，以适应不同的计算机和操作系统环境的限制。一个环境需要的代码跟另一个环境所需的代码可能有所不同，因为可用的硬件或操作系统是不同的。在许多情况下，可以把用于不同环境的代码放在同一个文件中，再在预处理阶段修改代码，使之适应当前的环境。

预处理器显示为一个独立的操作，但一般不能独立于编译器来执行这个操作。调用编译器会自动执行预处理过程，之后才编译代码。

编译器为给定源文件输出的是机器码，执行这个过程需要较长时间。在对象文件之间并没有建立任何连接。对应于某个源文件的对象文件包含在其他源文件中定义的函数引用或其他指定项的引用，而这些函数或项仍没有被解析。同样，也没有建立同库函数的链接。实际上，这些函数的代码并不是文件的一部分。这些工作是由链接程序(有时称为链接编辑器)完成的

链接程序把所有对象文件中的机器码组合在一起，并解析它们之间的交叉引用。它还集成了对象模块所使用的库函数的代码。这是链接程序的一种简化表示，因为这里假定在可执行模块中，模块之间的所有链接都是静态建立的。实际上有些链接是动态的，即这些链接是在程序执行时建立的。

链接程序静态地建立函数之间的链接，即在程序执行之前建立组成程序的源文件中所包含的函数链接。动态建立的函数之间的链接(在程序执行过程中建立的链接)将函数编译并链接起来，创建另一种可执行模块—— 动态链接库或共享库。动态链接库中的函数链接是在程序调用函数时才建立的，在程序调用之前，该链接是不存在的。

动态链接库有几个重要的优点。一个主要的优点是动态链接库中的函数可以在几个并行执行的程序之间共享，这将节省相同函数占用的内存空间。另一个优点是动态链接库在调用其中的函数之前是不会加载到内存中的。也就是说，如果不使用给定动态链接库中的函数，该动态链接库就不会占用内存空间

11、解释“优先级队列”这一抽象数据类型及实现方法

如果我们给每个元素都分配一个数字来标记其优先级，不妨设较小的数字具有较高的优先级，这样我们就可以在一个集合中访问优先级最高的元素并对其进行查找和删除操作了。这样，我们就引入了优先级队列 这种数据结构。

缺省情况下，优先级队列利用一个最大堆完成 函数列表：

empty() 如果优先队列为空，则返回真

pop() 删除第一个元素

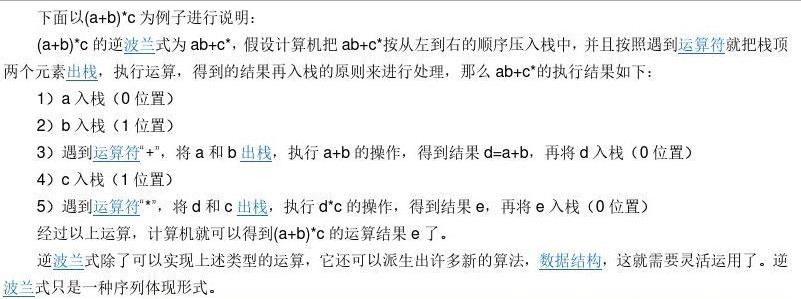
push() 加入一个元素

size() 返回优先队列中拥有的元素的个数

top() 返回优先队列中有最高优先级的元素

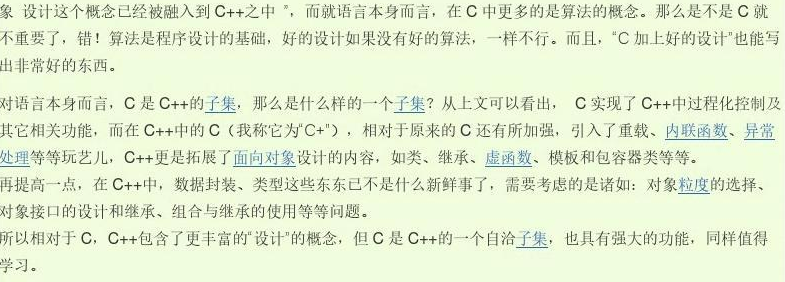
用途就不用多说了吧，例如Huffman编码、分支限界、A\*启发式都需要用到优先队列存放信息。

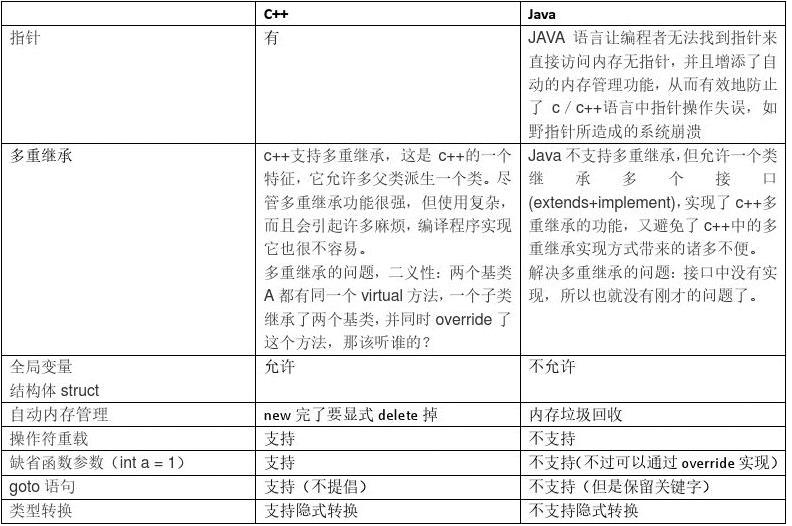
12、逆波兰式用什么数据结构算法的效率比较高，为什么



13、C和C++，C++和Java的区别

C是一个结构化语言，如谭老爷子所说：它的重点在于算法和数据结构。C程序的设计首要考虑的是如何通过一个过程，对输入（或环境条件）进行运算处理得到输出（或实现过程（事务）控制），而对于C++，首要考虑的是如何构造一个对象模型，让这个模型能够契合与之对应的问题域，这样就可以通过获取对象的状态信息得到输出或实现过程（事务）控制。

所以C与C++的最大区别在于它们的用于解决问题的思想方法不一样。之所以说C++比C更先进，是因为面向对



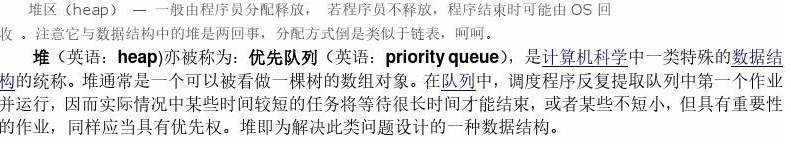
14、什么是预处理

程序设计中的预处理(Preprocess)，程序设计领域，预处理是在程序源代码被编译之前，由预处理器（Preprocessor）对程序源代码进行的处理。这个过程并不对程序的源代码进行解析，但它把源代码分割或处理成为特定的符号用来支持宏调用。

预处理器的主要作用就是把通过预处理的内建功能对一个资源进行等价替换，最常见 的预处理有：文件包含，条件编译、布局控制和宏替换4种。

15、堆和栈的区别

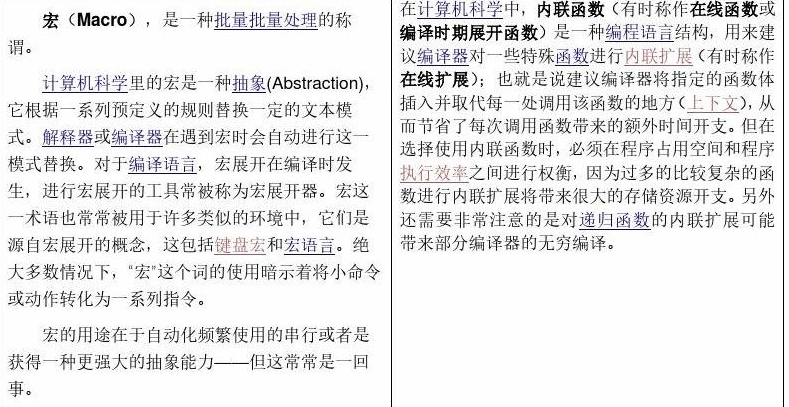
栈区（stack）— 由编译器自动分配释放 ，存放函数的参数值，局部变量的值等。其 操作方式类似于数据结构中的栈。



堆栈（英文：stack），也可直接称栈。台湾作堆叠，在计算机科学中，是一种特殊的串行形式的数据结构，它的特殊之处在于只能允许在链结串行或阵列的一端（称为堆栈顶端指标，英文为top）进行加入资料（push）和输出资料（pop）的运算。另外堆栈也可以用一维阵列或连结串行的形式来完成。堆栈的另外一个相对的操作方式称为伫列。

由于堆栈数据结构只允许在一端进行操作，因而按照后进先出（LIFO, Last In First Out）的原理运作。

堆栈数据结构使用两种基本操作：推入（push）和弹出（pop）：



17、简述在面向对象程序设计中，引入继承和封装的主要作用 继承：代码重用 封装：代码安全

18、简述C语言中指针及其作用(指向某个地址,加快访问速度)

19、Java语言的多线程机制(thread,runnable)

20、简述四种常见的数据逻辑结构

① 集合中任何两个数据元素之间都没有逻辑关系，组织形式松散。

C:\Users\Administrator\AppData\Roaming\Tencent\Users\949611520\QQ\WinTemp\RichOle\L5RMR]23}NZK$ITP4X]NV(H.png

C:\Users\Administrator\AppData\Roaming\Tencent\Users\949611520\QQ\WinTemp\RichOle\_K_%I21RH%@SMRU)(TIHT40.png

④ 图状结构 图状结构中的结点按逻辑关系互相缠绕，任何两个结点都可以邻接

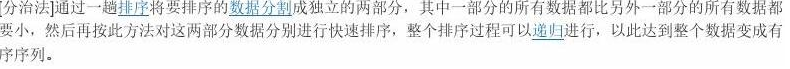
21、简述在一棵二叉排序树中查找一特定元素x的算法过程

二叉排序树（Binary Sort Tree）又称二叉查找树。 它或者是一棵空树；

或者是具有下列性质的二叉树：

（1）若左子树不空，则左子树上所有结点的值均小于它的根结点的值；

（2）若右子树不空，则右子树上所有结点的值均大于它的根结点的值；

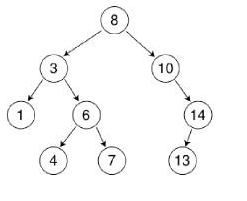


（3）左、右子树也分别为二叉排序树；

在最坏的情况是，两子数列拥有大各为 1 和 n-1，且调用树（call tree）变成为一个 n 个嵌套（nested）调用的线性连串（chain）。第 i 次调用作了O(n-i)的工作量，且递归关系式为： 这与插入排序和选择排序有相同的关系式，以及它被解为T(n) = O(n2)。

[http://retype.wenku.bdimg.com/retype/zoom/d592cb4c33687e21af45a9ed?pn=10&x=0&y=857&raww=893&rawh=55&aimh=13&o=png_6_0_0_0_0_0_0_892.979_1262.879&type=pic&fr=color&md5sum=3dbab4582a9bcbd236810bf12ca7f61c&sign=0c26fa3592&png=88908-104835&jpg=650119-675118](http://retype.wenku.bdimg.com/retype/zoom/d592cb4c33687e21af45a9ed?pn=10&x=0&y=857&raww=893&rawh=55&o=png_6_0_0_0_0_0_0_892.979_1262.879&type=pic&fr=color&md5sum=3dbab4582a9bcbd236810bf12ca7f61c&sign=0c26fa3592&png=88908-104835&jpg=650119-675118)

它的最坏情况是很恐怖的，需要



空间，远比数列本身还多。

23、简述在一带权有向图中寻找关键路径的基本思想

关键路径：关键路径是指网络终端元素的元素的序列，该序列具有最长的总工期并决定了整个项目的最短完成时间。在AOE网中，从始点到终点具有最大路径长度（该路径上的各个活动所持续的时间之和）的路径为关键路径 关键活动：关键路径上的活动称为关键活动。

只有所有关键活动提前完成，整个工程才能提前完成。

最早可能开始时间Ve[i]：从原点到顶点Vi最长路径的长度（之前所有事情做完了才可能开始）； 最迟允许开始时间Vl[i]：保证汇点Vn-1在Ve[n-1]时刻完成的前提下（整体工期不拖），事件Vi最迟允许的开始时间。Vl[i]=min{Vl[k]-dur(<Vi,Vk>)}

关键活动：松弛时间(slack time)Al[j]-Ae[k]==0的节点。

24、类作用域和文件作用域的区别是什么

文件作用域也称“全局作用域”。 •

•

• 定义在所有函数之外的标识符，具有文件作用域，作用域为从定义处到整个源文件结束。 文件中定义的全局变量和函数都具有文件作用域。 如果某个文件中说明了具有文件作用域的标识符，该文件又被另一个文件包含，则该标识符的作用域延伸到新的文件中。如cin和cout是在头文件iostream.h中说明的具有文件作用域的标识符，它们的作用域也延伸到嵌入iostream.h的文件中。

操作系统

1. 进程和线程的区别及联系，操作系统的程序栈…

线程和进程的区别：

1、 线程是进程的一部分，所以线程有的时候被称为是轻权进程或者轻量级进程。

2、 一个没有线程的进程是可以被看作单线程的，如果一个进程内拥有多个进程，进程的执行过程不是一条线（线程）的，而是多条线（线程）共同完成的。

3、 系统在运行的时候会为每个进程分配不同的内存区域，但是不会为线程分配内存（线程所使用的资源是它所属的进程的资源），线程组只能共享资源。那就是说，出了CPU之外（线程在运行的时候要占用CPU资源），计算机内部的软硬件资源的分配与线程无关，线程只能共享它所属进程的资源。

4、 与进程的控制表PCB相似，线程也有自己的控制表TCB，但是TCB中所保存的线程状态比PCB表中少多了（上下文保存一下就行）。

5、 进程是系统所有资源分配时候的一个基本单位，拥有一个完整的虚拟空间地址，并不依赖线程而独立存在。

进程与程序的区别：

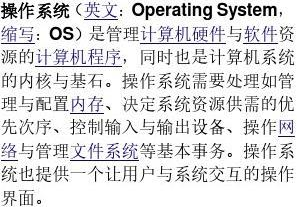
程序是一组指令的集合，它是静态的实体，没有执行的含义。而进程是一个动态的实体，有自己的生命周期。一般说来，一个进程肯定与一个程序相对应，并且只有一个，但是一个程序可以有多个进程，或者一个进程都没有也可以只有一个进程。除此之外，进程还有并发性和交往性。简单地说，进程是程序的一部分，程序运行的时候会产生进程。总结：线程是进程的一部分，进程是程序的一部分。

前一句说的不太准确，线程也有自己的资源，比如栈，私有数据等等。说他使用而不拥有资源指的是使用的是进程的打开文件句柄，进程的全局数据，进程的地址空间等等,这些都属于进程，而不属于线程，进程内个线程共享。

进程切换比线程切换开销大是因为进程切换时要切页表，而且往往伴随着页调度，因为进程的数据段代码段要换出去，以便把将要执行的进程的内容换进来。本来进程的内容就是线程的超集。而且线程只需要保存线程的上下文（相关寄存器状态和栈的信息）就好了，动作很小。

2. OS里面进程的“三态”“五态”“七态”是什么

3. 什么是操作系统

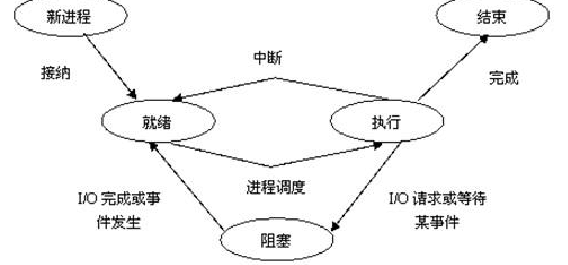


4. 死锁的条件，检测死锁的可能方法及其基本思想

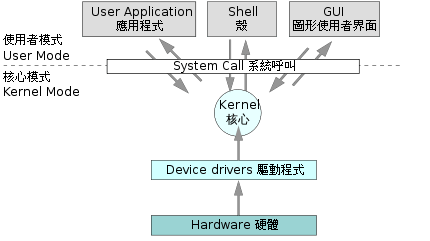
A deadlock is a situation in which two or more competing actions are each waiting for the other to finish, and thus neither ever does.

1. （互斥）: At least two resources must be non-shareable.[1] Only one process can use the resource at any given instant of time.

2. Hold and Wait or Resource Holding: A process is currently holding at least one resource and requesting



additional resources which are being held by other processes.



3. No （禁止抢占）: The operating system must not de-allocate resources once they have been

allocated; they must be released by the holding process voluntarily.

4. A process must be waiting for a resource which is being held by another process,

which in turn is waiting for the first process to release the resource. In general, there is a of waiting

processes, P = {P1, P2, ..., PN}, such that P1 is waiting for a resource held by P2, P2 is waiting for a resource held by P3 and so on until PN is waiting for a resource held by P1.[1][7]

These four conditions are known as the Coffman conditions from their first description in a 1971 article

by [7] Unfulfillment of any of these conditions is enough to preclude a deadlock from occurring.

Ensure that the system will never enter a deadlock state.

Prevention: Ensure one of the four conditions fails.

Avoidance: The OS needs more information so that it can determine if the current request can be satisfied or

delayed.

死锁检测：

1. Resource-Allocation Graph(资源分配图)

2. Detection Algorithm(算法检测)

5. 用户态和内核态

内核态：cpu可以访问内存的所有数据，包括外围设备，例如硬盘，网卡，cpu也可以将自己从一个程序切换到另一个程序。

用户态：只能受限的访问内存，且不允许访问外围设备，占用cpu的能力被剥夺，cpu资源可以被其他程序获取。

当程序运行在3级特权级上时，就可以称之为运行在用户态，因为这是最低特权级，是普通的用户进程运行的特权级，大部分用户直接面对的程序都是运行在用户态；反之，当程序运行在3级特权级下时，就可以称之为运行在内核态。 用户态切换到内核态的3种方式

a. 系统调用

这是用户态进程主动要求切换到内核态的一种方式，用户态进程通过系统调用申请使用操作系统提供的服务程序完成工作，比如前例中fork()实际上就是执行了一个创建新进程的系统调用。而系统调用的机制其核心还是使用了操作系统为用户特别开放的一个中断来实现，例如Linux的int 80h中断。

b. 异常

当CPU在执行运行在用户态下的程序时，发生了某些事先不可知的异常，这时会触发由当前运行进程切换到处理此异常的内核相关程序中，也就转到了内核态，比如缺页异常。

c. 外围设备的中断

当外围设备完成用户请求的操作后，会向CPU发出相应的中断信号，这时CPU会暂停执行下一条即将要执行的指令转而去执行与中断信号对应的处理程序，如果先前执行的指令是用户态下的程序，那么这个转换的过程自然也就发生了由用户态到内核态的切换。比如硬盘读写操作完成，系统会切换到硬盘读写的中断处理程序中执行后续操作等。

这3种方式是系统在运行时由用户态转到内核态的最主要方式，其中系统调用可以认为是用户进程主动发起的，异常和外围设备中断则是被动的。

6. 面包店算法

该算法的基本思想源于顾客在面包店中购买面包时的排队原理. 顾客在进入面包店前, 首先抓一个号, 然后按照号码由小到大的次序依次进入面包店购买面包. 这里, 面包店发放的号码是由小到大的, 但是两个或两个以上的顾客却有可能得到相同的号码(使所抓号码不同需要互斥), 如果多个顾客抓到相同的号码, 则规定按照顾客名字的字典次序进行排序, 这里假定顾客是没有重名的. 在计算机系统中, 顾客就相当于进程, 每个进程有一个唯一的标识, 我们用P的下面加一个下标来表示. 例如: 对于 Pi和Pj, 如果有i<j, 则先为Pi服务, 即Pi先进入临界区

7. 系统调用和库函数的区别

(1)调用形式不同。过程（函数）使用一般调用指令，其转向地址是固定不变的，包含在跳转语句中；但系统调用中不包含处理程序入口，而仅仅提供功能号，按功能号调用。

(2)被调用代码的位置不同。过程（函数）调用是一种静态调用，调用者和被调用代码在同一程序内，经过连接编辑后作为目标代码的一部份。当过程（函数）升级或修改时，必须重新编译连结。而系统调用是一种动态调用，系统调用的处理代码在调用程序之外（在操作系统中），这样一来，系统调用处理代码升级或修改时，与调用程序无关。而且，调用程序的长度也大大缩短，减少了调用程序占用的存储空间。

(3)提供方式不同。过程（函数）往往由编译系统提供，不同编译系统提供的过程（函数）可以不同；系统调用由操作系统提供，

一旦操作系统设计好，系统调用的功能、种类与数量便固定不变了。

(4)调用的实现不同。程序使用一般机器指令（跳转指令）来调用过程（函数），是在用户态运行的；程序执行系统调用，是通过中断机构来实现，需要从用户态转变到核心态，在管理状态执行，因此，安全性好。

8. 经典进程同步问题是什么，同步思想？

生产者-消费者问题是著名的进程同步问题，它描述一组生产者进程向一组消费者进程提供消息。它们共享一个有界缓冲池，生产者向其中投放消息，消费者从中取得消息。生产者-消费者问题是许多相互合作进程的一种抽象。假定缓冲池中有n个缓冲区，每个缓冲区存放一个消息。由于缓冲池是临界资源，它只允许一个生产者投入消息，或者一个消费者从中取出消息。生产者之间、生产者与消费者之间、消费者之间都必须互斥地使用缓冲池。所以必须设置互斥信号量mutex，它代表缓冲池资源，它的数值为1。 读者-写者问题

问题描述：一个数据集（如文件）被几个并行进程所共享，有些进程只要求读数据集内容，称为读者，而另一些进程则要求修改数据集内容，称为写者，几个读者可以同时读数据集，而不需要互斥，但一个写者不能和其他进程（不管是写者或读者）同时访问这些数据集，它们之间必须互斥。

哲学家进餐问题

该问题描述如下：有五个哲学家，他们的生活方式是交替地进行思考和进餐。哲学家们公用一张圆桌，周围放有五把椅子，每人坐一把。在圆桌上有五个碗和五根筷子，当一个哲学家思考时，他不与其他人交谈，饥饿时便试图取用其左、右最靠近他的筷子，但他可能一根都拿不到。只有在他拿到两根筷子时，方能进餐，进餐完后，放下筷子又继续思考。

9. 文件管理及组织，FAT

FAT = File Allocation Table

10. 设备驱动程序是否属于OS，他的作用是什么

不是，驱动程序是另外安装的软件，是操作系统控制并且和硬件之间通讯的桥梁（程序）

11. 程序和任务的区别

任务是最抽象的，是一个一般性的术语，指由软件完成的一个活动。一个任务既可以是一个进程,也可以是一个线程。简而言之,它指的是一系列共同达到某一目的的操作。例如,读取数据并将数据放入内存中。这个任务可以作为一个进程来实现,也可以作为一个线程（或作为一个中断任务）来实现。 进程常常被定义为程序的执行。可以把一个进程看成是一个独立的程序,在内存中有其完备的数据空间和代码空间。一个进程所拥有的数据和变量只属于它自己。 线程则是某一进程中一路单独运行的程序。也就是说，线程存在于进程之中。一个进程由一个或多个线程构成，各线程共享相同的代码和全局数据，但各有其自己的堆栈。由于堆栈是每个线程一个，所以局部变量对每一线程来说是私有的。由于所有线程共享同样的代码和全局数据，它们比进程更紧密，比单独的进程间更趋向于相互作用,线程间的相互作用更容易些,因为它们本身就有某些供通信用的共享内存：进程的全局数据进程的全局数据进程的全局数据进程的全局数据

进程：资源分配、调度运行的基本单位。

线程：进程中执行运算的最小单位，执行处理机调度的基本单位。

12. 数据库安全性和操作系统安全性的关系

安全性问题不是数据库系统所独有的,所有计算机系统都有这个问题.只是在数据库系统中大量数据集中存放,而且为许多最终用户直接共享,从而使安全性问题更为突出.系统安全保护措施是否有效是数据库系统的主要指标之一.数据库的安全性和计算机系统的安全性,包括操作系统,网络系统的安全性是紧密联系,相互支持的.

13. 中断处理的过程

请求中断→响应中断→关闭中断→保留断点→中断源识别→保护现场→中断服务子程序→恢复现场→中断返回 在实际运行中，一旦设备通过某引脚N向8259A发出中断指令，后者便向8086A的INTR引脚发送中断信号。8086A通过INTA引脚通知8259A中断有效（这个过程实际上还包括对此8259A的选址），后者即通过地址总线将对应引脚N的中断类型码（已预先存好，见上节）发送给CPU。CPU得到中断类型码后，先进行现场保护，主要包括：

1. 状态寄存器FLAGS压栈（同时堆栈寄存器SP-2）；

2. 关闭中断（将FLAGS寄存器的IF位置零）；

3. 将当前代码段寄存器CS和程序计数器IP压栈（同时堆栈寄存器SP-4）。 现场保护完成后，CPU开始按照前述的两步骤翻译中断程序入口地址。在得到中断处理程序地址之后但调用中断处理程序之前，CPU会再检查一下NMI引脚是否有信号，以防在刚才的处理过程中忽略了可能的NMI中断。NMI的优先级始终高于INTR。

中断处理程序虽然是由程序员编写，但须循一定规范。作为例程，中断处理程序应该先将各寄存器信息（除了IP和CS，此二寄存器现已指向当前中断程序）压入堆栈予以保存，这样才能在中断处理程序内部使用这些寄存器。在程序结束时，应该按与压栈保护时相反的顺序弹出各寄存器的值。中断程序的最后一句始终是IRET指令，这条指令将栈顶6个字节分别弹出并存入IP、CS和FLAGS寄存器，完成了现场的还原。

当然，如果是操作系统的中断处理程序，则未必——通常不会——还原中断前的状态。这样的中断处理程序通常会在调用完寄存器保存例程后，调用进程调度程序（多由高级语言编写），并决定下一个运行的进程。随后将此进程的寄存器信息（上次中断时保存下来的）存入寄存器并返回。在中断程序结束之后，主程序也发生了改变。

14. 计算机系统怎样实现存储保护

1. 防止地址越界（对进程所产生的地址必须加以检查，发生越界时产生中断，由操作系统进行相应处理）

2. 防止操作越权（对属于自己区域的信息，可读可写：对公共区域中允许共享的信息或获得授权可使用的信息，可读而不可修改；对未授权使用的信息，不可读，不可写）

15. 调度的基本准则(Scheduling Criteria)

There are many criteria for comparing different scheduling algorithms. Here are five common ones: CPU utilization(CPU利用率)

Throughput（吞吐量）

Turnaround time（周转时间）

Waiting time（等待时间）

Response time（响应时间）

16. 多线程是否真正能提高效率(不能提高效率,只能减少等待时间)

17. 磁盘调度算法有哪几种

FIFO：先来先服务算法；

SSTF： 最短寻道时间算法；

SCAN：电梯调度算法;

CSCAN： 循环扫描算法

FSCAN：分步电梯调度算法(分两个队列)

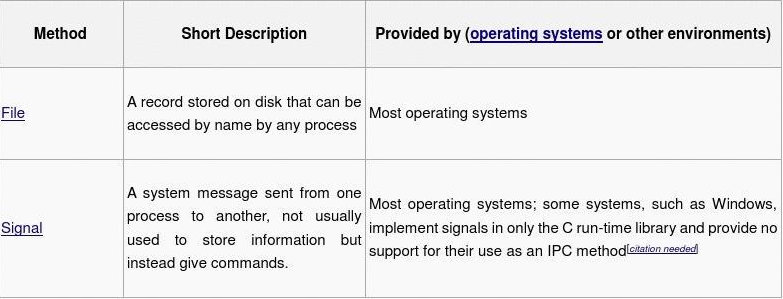
18. RAID工作原理

RAID(Redundant Array of Independent Disks)通过条带化存储和奇偶校验两个措施来实现其冗余和容错的目标。条带化存储意味着可以一次写入一个数据块的方式将文件写入多个磁盘。条带化存储技术将数据分开写入多个驱动器，从而提高数据传输速率并缩短磁盘处理总时间。这种系统非常适用于交易处理、但可靠性却很差，因为系统的可靠性等于最差的单个驱动器的可靠性。

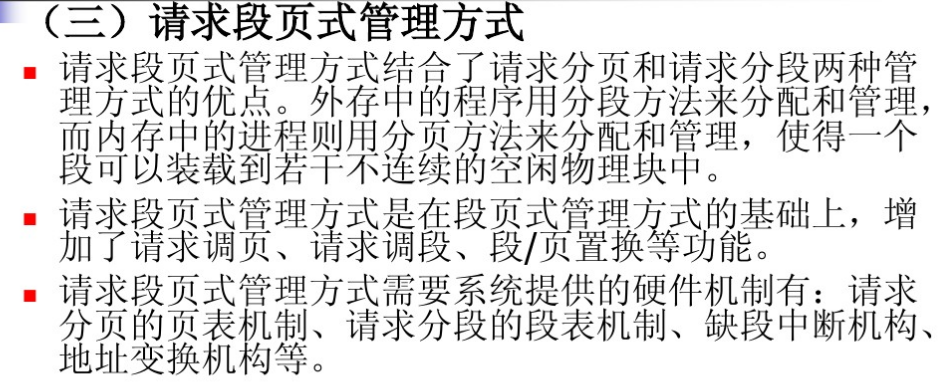
奇偶校验通过在传输后对所有数据进行冗余校验可以确保数据的有效性。利用奇偶校验，当RAID系统的一个磁盘发生故障时，其它磁盘能够重建该故障磁盘。在这两种情况中，这些功能对于操作系统都是透明的。由磁盘阵列控制器（DAC）进行条带化存储和奇偶校验控制。

19. Windows中段最长多少字节(64字节)

20. 同步、互斥



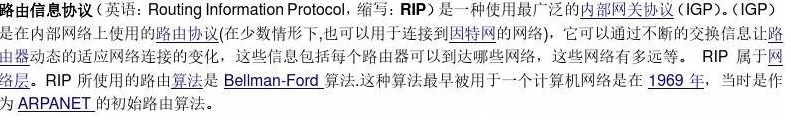
22. 简述请求段页式虚拟内存管理基本思想

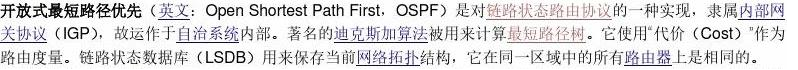


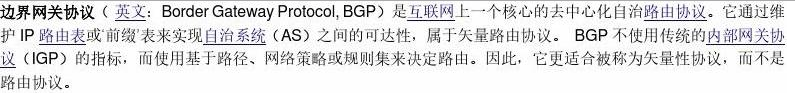
网络

1. 路由协议

常见路由协议有：RIP路由信息协议(Route Information Protocol)，OSPF开放式最短路径优先(Open Shortest Path First)，BGP边界网关协议(Border Gateway Protocol)等







2. CSMA/CD, 指数回退

载波侦听多路访问／冲突检测（英语：Carrier Sense Multiple Access with Collision

Detection，CSMA/CD）

此方案要求设备在发送帧的同时要对信道进行侦听，以确定是否发生冲突，若在发送数据过程中检测到冲突，则进行如下冲突处理操作：

1. 发送特殊阻塞信息并立即停止发送数据：特殊阻塞信息是连续几个字节的全1信号，此举意在强化冲突，以使得其它设备能尽快检测到冲突发生。

2. 在固定时间（一开始是 1 contention period times）内等待随机的时间，再次发送。

3. 若依旧碰撞，则采用进行发送。即十次之内停止前一次“固定时间”的两

倍时间内随机再发送，十次后则停止前一次“固定时间”内随机再发送。尝试16次之后仍然失败则放弃传送。

载波侦听多路访问／冲突避免（英语：Carrier Sense Multiple Access with Collision Avoidance，CSMA/CA）

此种方案采用主动避免碰撞而非被动侦测的方式来解决冲突问题。可以满足那些不易准确侦测是否有冲突发生的需求，如无线网域。

CSMA/CA协议主要使用两种方法来避免碰撞： 1. 设备欲发送讯框(Frame)，且讯框听到通道空闲时，维持一段时间后，再等待一段随机的时间依然空闲时，才提交数据。由于各个设备的等待时间是分别随机产生的，因此很大可能有所区别，由此可以减少冲突的可能性。

2. RTS-CTS三向握手（handshake）：设备欲发送讯框前，先发送一个很小的RTS

（Request to Send）讯框给目标端，等待目标端回应CTS（Clear to Send）帧后，才开始传送。此方式可以确保接下来传送数据时，不会发生冲突。同时由于RTS帧与CTS帧都很小，让传送的无效开销变小。

3. 解释FTP、HTTP的全称及其原理。FTP、HTTP文件传输的异同

FTP：File Transfer Protocol

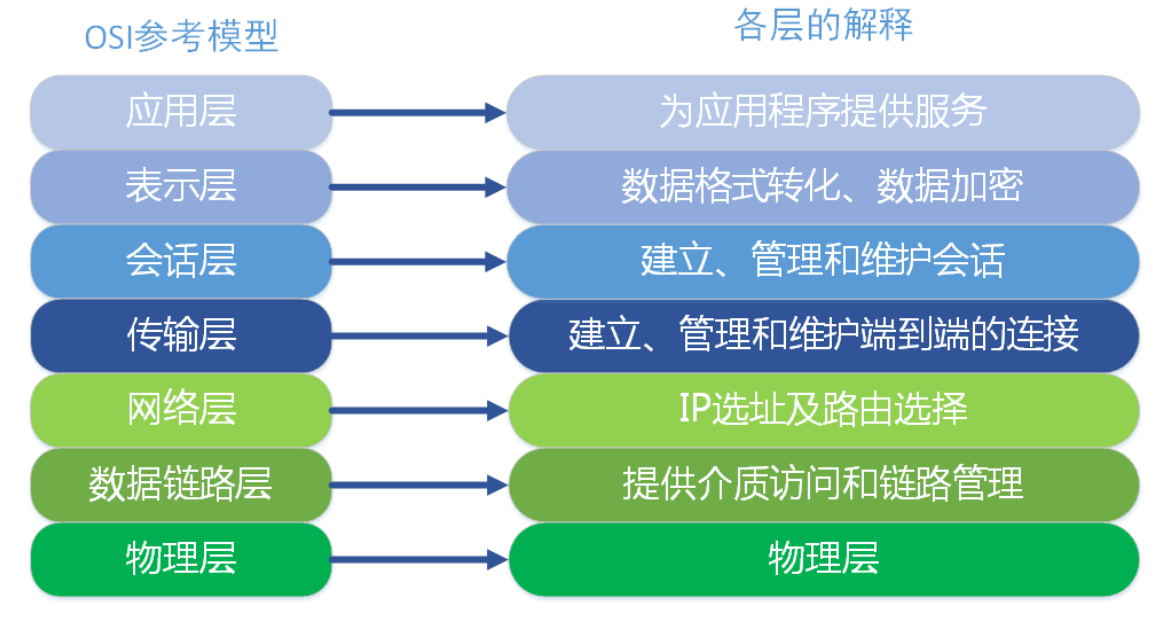
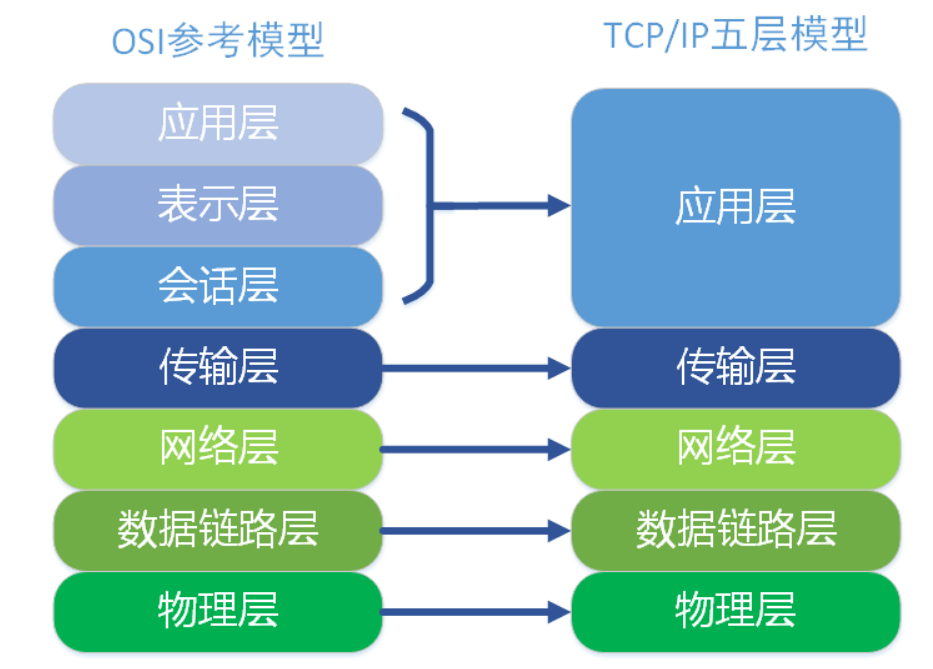
HTTP: Hyper Text Transfer Protocol

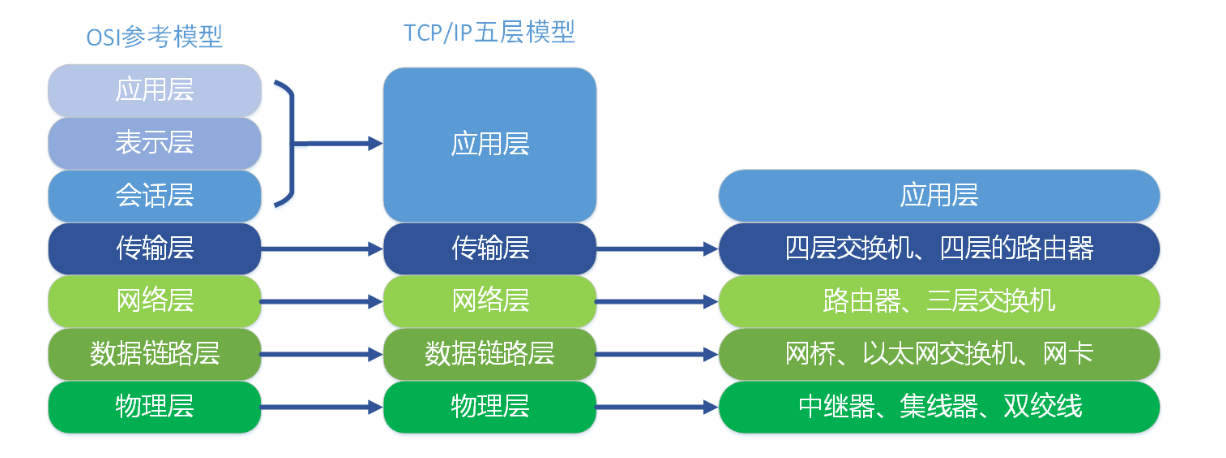
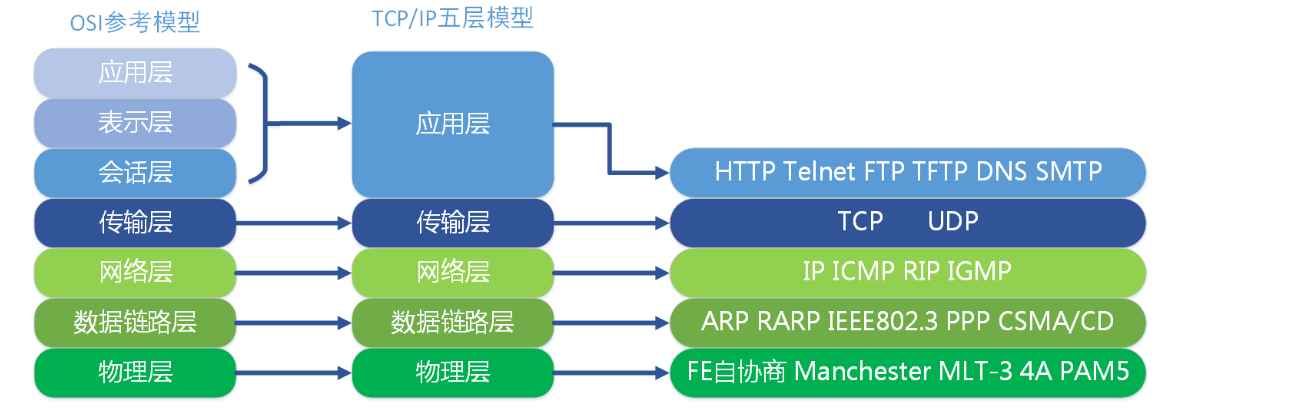
都是应用层协议。

FTP多用于局域网文件传输,HTTP多用于互联网html页面传输

FTP实现比较复杂

4. 七层协议的名称

5. 电子邮件发送到接收的过程，协议

电子邮件的工作过程遵循客户-服务器模式。每份电子邮件 的发送都要涉及到发送方与接收方，发送方构成客户端，而接收方构成服务器，服务器含有众多用户的电子信箱。发送方通过邮件客户程序，将编辑好的电子邮件向邮局服务器（SMTP服务器）发送。邮局服务器识别接收者的地址，并向管理该地址的邮件服务器（POP3服务器）发送消息。邮件服务器识将消息存放在接收者的电子信箱内，并告知接收者有新邮件到来。接收 者通过邮件客户程序连接到服务器后，就会看到服务器的通知，进而打开自己的电子信箱来查 收邮件。

发送：SMTP

接收：POP, IMAP

6. P2P技术，具体的实现机制

实现P2P需要一个中转服务器。也就是需要一个第三方。（一会儿我们来说为什么需要一个第三方）

7. 网络中的握手问题



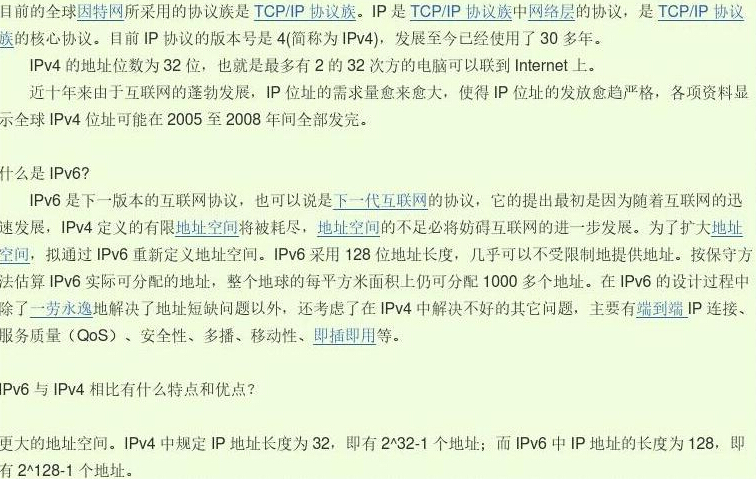
8. 无线传感网络(WSN)及其应用

无线传感网络（Wireless sensor network），或译无线感知网络，是由许多在空间中分布的自动装置组成的一种无线通讯计算机网络，这些装置使用传感器协作地监控不同位置的物理或环境状况（比如温度、声音、振动、压力、运动或污染物）。无线传感器网络的发展最初起源于战场监测等军事应用。而现今无线传感器网络被应用于很多民用领域，如环境与生态监测、健康监护、家居自动化以及交通控制等。

无线传感网络主要包括三个方面：感应，通讯，计算（硬件，软件，算法）。其中的关键技术主要有无线数据库技术，比如使用在无线传感器网络的查询，和用于和其他传感器通讯的网络技术，特别是多次跳跃路由协议。

9. 当前网络新技术及其应用,Web2.0

Web 2.0，指的是一个利用Web的平台，由用户主导而生成的内容互联网产品模式，为了区别传统由网站雇员主导生成的内容而定义为web2.0



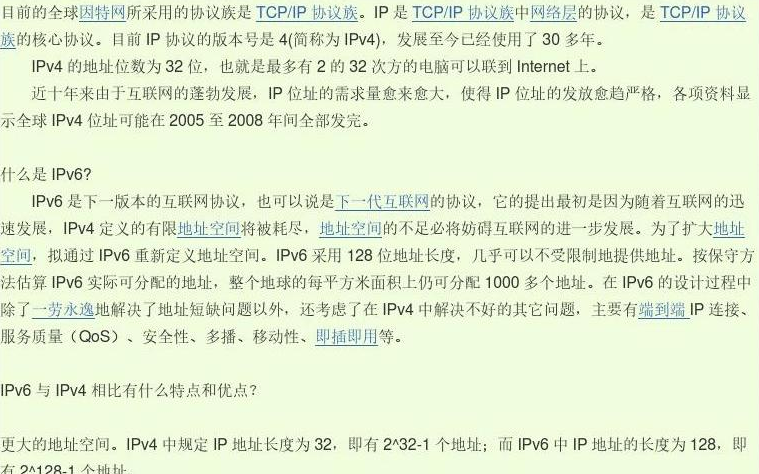
8. 无线传感网络(WSN)及其应用

无线传感网络（Wireless sensor network），或译无线感知网络，是由许多在空间中分布的自动装置组成的一种无线通讯计算机网络，这些装置使用传感器协作地监控不同位置的物理或环境状况（比如温度、声音、振动、压力、运动或污染物）。无线传感器网络的发展最初起源于战场监测等军事应用。而现今无线传感器网络被应用于很多民用领域，如环境与生态监测、健康监护、家居自动化以及交通控制等。

无线传感网络主要包括三个方面：感应，通讯，计算（硬件，软件，算法）。其中的关键技术主要有无线数据库技术，比如使用在无线传感器网络的查询，和用于和其他传感器通讯的网络技术，特别是多次跳跃路由协议。

9. 当前网络新技术及其应用,Web2.0

Web 2.0，指的是一个利用Web的平台，由用户主导而生成的内容互联网产品模式，为了区别传统由网站雇员主导生成的内容而定义为web2.0



11. TCP拥塞控制(congestion control)与流量控制(flow control)的功能和区别

流量控制解决的问题：通信双方速率不匹配。

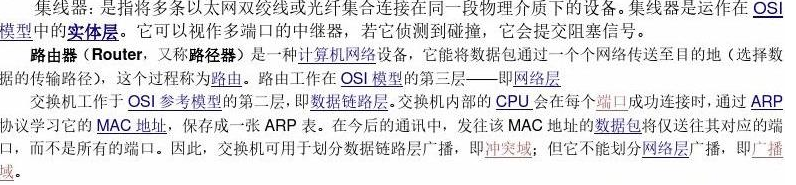
方法：Sliding window，控制滑动窗口大小，自适应速率。

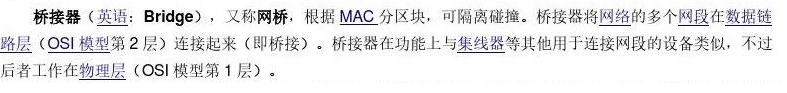
拥塞控制解决的问题：双方速率都允许的情况下，如何保证途中不出问题. 慢启动( slow-start )、拥塞避免( congestion avoidance )、快重传( fast retransmit )和快恢复( fast recovery )

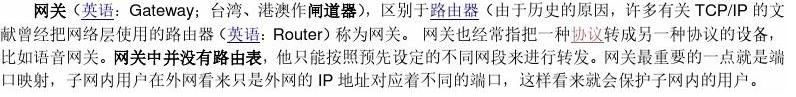
12. PPP协议

点对点协议（英语：Point-to-Point Protocol，PPP）工作在数据链路层（以OSI参考模型的观点）。它通常用在两节点间创建直接的连接，并可以提供连接认证、传输加密（使用ECP，RFC 1968）以及压缩。

13. 集线器、路由器和交换机有什么区别。中继器、集线器、交换机、网桥、网关、路由器的功能







14. P2P(对等网络)网络编程的特点

分布式,共享资源

15. DNS的递归查询和迭代查询

（1）递归查询(逐层)

递归查询是一种DNS 服务器的查询模式，在该模式下DNS 服务器接收到客户机请求，必须使用一个准确的查询结果回复客户机。如果DNS 服务器本地没有存储查询DNS 信息，那么该服务器会询问其他服务器，并将返回的查询结果提交给客户机。

（2）迭代查询(平级)

DNS 服务器另外一种查询方式为迭代查询，DNS 服务器会向客户机提供其他能够解析查询请求的DNS 服务器地址，当客户机发送查询请求时，DNS 服务器并不直接回复查询结果，而是告诉客户机另一台DNS 服务器地址，客户机再向这台DNS 服务器提交请求，依次循环直到返回查询的结果为止。

16. ARP协议、ARP攻击

地址解析协议（Address Resolution Protocol），其基本功能为通过目标设备的IP地址，查询目标设备的MAC地址，以保证通信的顺利进行。它是网络层必不可少的协议，不过已不再适用，并被邻居发现协议（NDP）所替代。

在以太网协议中规定，同一局域网中的一台主机要和另一台主机进行直接通信，必须要知道目标主机的MAC地址。而在TCP/IP

[http://retype.wenku.bdimg.com/retype/zoom/d592cb4c33687e21af45a9ed?pn=22&x=0&y=258&raww=787&rawh=47&aimh=13&o=jpg_6_0_______&type=pic&fr=color&md5sum=3dbab4582a9bcbd236810bf12ca7f61c&sign=0c26fa3592&png=158331-158856&jpg=1267390-1348460](http://retype.wenku.bdimg.com/retype/zoom/d592cb4c33687e21af45a9ed?pn=22&x=0&y=258&raww=787&rawh=47&o=jpg_6_0_______&type=pic&fr=color&md5sum=3dbab4582a9bcbd236810bf12ca7f61c&sign=0c26fa3592&png=158331-158856&jpg=1267390-1348460)

协议中，网络层和传输层只关心目标主机的IP地址。这就导致在以太网中使用IP协议时，数据链路层的以太网协议接到上层IP协议提供的数据中，只包含目的主机的IP地址。于是需要一种方法，根据目的主机的IP地址，获得其MAC地址。这就是ARP协议要做的事情。所谓地址解析（address resolution）就是主机在发送帧前将目标IP地址转换成目标MAC地址的过程。

ARP欺骗的运作原理是由攻击者发送假的ARP数据包到网络上，尤其是送到网关上。其目的是要让送至特定的IP地址的流量被错误送到攻击者所取代的地方。因此攻击者可将这些流量另行转送到真正的闸道（被动式数据包嗅探，passive sniffing）或是篡改后再转送（中间人攻击，man-in-the-middle attack）。

17. 计算机网络的接入类型有哪些

局域网、城域网、广域网和互联网四种

19. 电路交换和报文交换的区别

报文交换（英语：Message switching），又称存储转发交换，是数据交换的三种方式之一，报文整个地发送，一次一跳。报文交换是分组交换的前身，是由由列奥纳德·克莱因饶克于1961年提出的。

报文交换的主要特点是：存储接受到的报文，判断其目标地址以选择路由，最后，在下一跳路由空闲时，将数据转发给下一跳路由。报文交换系统现今都由分组交换或电路交换网络所承载。

电路交换（Circuit Switching）是相对于封包交换(或称分组交换)的一个概念。电路交换要求必须首先在通信双方之间建立连接通道。在连接建立成功之后，双方的通信活动才能开始。通信双方需要传递的信息都是通过已经建立好的连接来进行传递的，而且这个连接也将一直被维持到双方的通信结束。在某次通信活动的整个过程中，这个连接将始终占用着连接建立开始时，通信系统分配给它的资源（通道、带宽、时隙、码字等等），这也体现了电路交换区别于分组交换的本质特征。

20.

计组/接口

1. Cache的两种更新策略

一般有两种回写策略：写回（Write back）和写通（Write through）。

1. write back

只有在一个cache行被选中替换回主存时，如果cache 行的数据是修改过的（dirty），才将它写回主存。

这种策略，要在Cache中设置一个脏位（dirty bit），用来表示缓存中的cache 行是否被修改过。

如果 一个内存块在加载到Cache后未被修改过，Cache直接把该cache行设置为无效。不需要把数据写回主存，这样可以有效降低从Cache到主存的写次数。

2.write through

写通是指，每当Cache收到写数据（store）指令时，若写命中，则CPU会同时将数据写到Cache和主存。

2. 计算机中小数点是如何表示的

定点、浮点表示(科学记数法)

3. 计算机中如何表示数据，知识

二进制

4. Cache的原理及思想，评价标准，改进方案，计算机软硬件中其他用到这个思想的方法

缓存之所以有效，主要是因为程序运行时对内存的访问呈现局部性（Locality）特征。这种局部性既包括空间局部性（Spatial Locality），也包括时间局部性（Temporal Locality）。有效利用这种局部性，缓存可以达到极高的命中率。

评估缓存的性能通常使用平均内存访问时间（Average Memory Access Time, AMAT）这一指标。

5. 分页、分段、段页式的特点，为什么要引入?(段包含页)

分页（英语：Paging），是一种操作系统里存储器管理的一种技术，可以使电脑的主存可以使用存储在辅助存储器中的数据。操作系统会将辅助存储器中的数据分区成固定大小的区块，称为“页”（pages）。当不需要时，将分页由主存移到辅助存储器；当需要时，再将数据取回，加载主存中。相对于分段，分页允许存储器存储于不连续的区块以维持文件系统的整齐。在分页实现前，分段会使文件于系统中连续，这使它容易造成空间杂乱且产生许多碎片。

[http://retype.wenku.bdimg.com/retype/zoom/d592cb4c33687e21af45a9ed?pn=23&x=0&y=0&raww=792&rawh=137&aimh=38&o=jpg_6_0_______&type=pic&fr=color&md5sum=3dbab4582a9bcbd236810bf12ca7f61c&sign=0c26fa3592&png=158857-159272&jpg=1348461-1369133](http://retype.wenku.bdimg.com/retype/zoom/d592cb4c33687e21af45a9ed?pn=23&x=0&y=0&raww=792&rawh=137&o=jpg_6_0_______&type=pic&fr=color&md5sum=3dbab4582a9bcbd236810bf12ca7f61c&sign=0c26fa3592&png=158857-159272&jpg=1348461-1369133)

段 页 式 存 储 器 兼 顾 了 程 序 和 数 据 的 逻 辑 结 构 以 及 存 储 器 的 物 理 结 构 两 方 面 的 特 点， 兼 收 页 式 和 段 式 虚 拟 存 储 器 两 者 的 长 处， 并 具 有 如 下 特 点：

面 向 用 户 的 程 序 地 址 空 间 采 用 段 式 分 隔。

内 存 分 成 位 置 固 定 、 长 度 相 等 的 若 干 页 面。

将 每 一 段 的 线 性 地 址 空 间 划 分 为 页， 页 长 与 内 存 页 面 相 等。

6. 中断的作用(及时响应应用的请求)

中断的典型应用包括系统时钟、磁盘输入输出操作、断电信号以及软件自陷等。

7. DMA(直接存取访问)优先级为什么比CPU高

DMA是有专用数据通路的,DMA传送过程中cpu可以继续工作,不会阻塞 可以响应外部中断,独享总线的DMA会在传输时占用总线,这时候CPU将总线让给DMA,这时候DMA优先级比CPU高.请求型DMA会在占用总线是向CPU发起请求,CPU优先级比DMA高

8. 虚拟内存容量由什么决定

寻址长度（地址总线数量）

虚拟存储区的容量与物理主存大小无关，而受限于计算机的地址结构和可用磁盘容量。

9. 根据内存大小算地址总线数量

10. 8086指令地址的计算方式

基址寄存器BX, BP

变址寄存器 SI, DI

BX,SI,DI默认DS段

BP默认SS段

立即数寻址

直接寻址： MOV AX, [2000H] MOV AX, ES:[2000H]

寄存器寻址 MOV AX, BX

寄存器间接寻址 MOV AX, [BX]

寄存器相对寻址 MOV AX, [SI+06H]

基址变址寻址 MOV AX, [BS+SI]

基址变址相对寻址 MOV AX, [BX+DI+6]

11. 冯.诺伊曼机以什么部件为中心

计算机五大组成部件:运算器、控制器、存储器、输入设备和输出设备.

以运算器为中心，I/O设备与存储器间的数据传送都要经过运算器

12. CISC/RISC的特点

（1） 指令系统：RISC 设计者把主要精力放在那些经常使用的指令上，尽量使它们具有简单高效的特色。对不常用的功能，常通过组合指令来完成。因此，在RISC 机器上实现特殊功能时，效率可能较低。但可以利用流水技术和超标量技术加以改进和弥补。而CISC 计算机的指令系统比较丰富，有专用指令来完成特定的功能。因此，处理特殊任务效率较高。

（2） 存储器操作：RISC 对存储器操作有限制，使控制简单化；而CISC 机器的存储器操作指令多，操作直接。

（3） 程序：RISC 汇编语言程序一般需要较大的内存空间，实现特殊功能时程序复杂，不易设计；而CISC 程序编程相对简单，科学计算及复杂操作的程序社设计相对容易，效率较高。

（4） 中断：RISC 机器在一条指令执行的适当地方可以响应中断；而CISC 机器是在一条指令执行结束后响应中断。

（5） CPU：RISC CPU 包含有较少的单元电路，因而面积小、功耗低；而CISC CPU 包含有丰富的电路单元，因而功能强、面积大、功耗大。

（6） 设计周期：RISC 微处理器结构简单，布局紧凑，设计周期短，且易于采用最新技术；CISC 微处理器结构复杂，设计周期长。

（7） 用户使用：RISC 微处理器结构简单，指令规整，性能容易把握，易学易用；CISC微处理器结构复杂，功能强大，实现特殊功能容易。

（8） 应用范围：由于RISC 指令系统的确定与特定的应用领域有关，故RISC 机器更适合于专用机；而CISC 机器则更适合于通用机。

15. 一级缓存存取速度快是为什么

静态随机存取存储器 (SRAM)不用刷新，直接与CPU数据总线相连

16. RS-232接口

18. 主存-Cache结构有什么作用

19. 简述虚拟存储器的基本概念

虚拟存储器是指具有请求调入功能和置换功能，能从逻辑上对内存容量加以扩充的一种存储器系统。其逻辑容量由内存容量和外存容量之和来决定，其运行速度接近于内存速度，而每位的成本却又接近于外存。可见，虚拟存储技术是一种性能非常优越的存储器管理技术，故被广泛地应用于大、中、小型和微型机器中。

[http://retype.wenku.bdimg.com/retype/zoom/d592cb4c33687e21af45a9ed?pn=25&x=0&y=0&raww=787&rawh=140&aimh=39&o=jpg_6_0_______&type=pic&fr=color&md5sum=3dbab4582a9bcbd236810bf12ca7f61c&sign=0c26fa3592&png=159925-160464&jpg=1446495-1474007](http://retype.wenku.bdimg.com/retype/zoom/d592cb4c33687e21af45a9ed?pn=25&x=0&y=0&raww=787&rawh=140&o=jpg_6_0_______&type=pic&fr=color&md5sum=3dbab4582a9bcbd236810bf12ca7f61c&sign=0c26fa3592&png=159925-160464&jpg=1446495-1474007)

21.

其他

（广）什么是计算机，计算，语法，语义，语用

计算机（computer）俗称电脑，是一种用于高速计算的电子计算机器，可以进行数值计算，又可以进行逻辑计算，还具有存储记忆功能。

语义：机器对自然语言的理解

语用：如何运用和理解语言

（离散）群中Lagrange定理及其证明

根本不懂

（离散）解释元素，树，图。并各举一例

（软工）软件架构的理解

软件架构是有关软件整体结构与组件的抽象描述，用于指导大型软件系统各个方面的设计。

软件架构师定义和设计软件的模块化，模块之间的交互，用户界面风格，对外接口方法，创新的设计特性，以及高层事物的对象操作、逻辑和流程。

软件架构师与客户商谈概念上的事情，与经理商谈广泛的设计问题，与软件工程师商谈创新的结构特性，与程序员商谈实现技巧，外观和风格。

软件架构是一个系统的草图。软件架构描述的对象是直接构成系统的抽象组件。各个组件之间的连接则明确和相对细致地描述组件之间的通讯。在实现阶段，这些抽象组件被细化为实际的组件，比如具体某个类或者对象。在面向对象领域中，组件之间的连接通常用接口来实现。

（广）什么是平台无关性

就是说（java）写的程序不用修改就可以在不同的软硬件平台上运行

比如你写了一个程序，在windows下可以执行

那你直接拷到Linux下也可以执行，只要都装了JRE

（数据库）查询优化有哪些

基于关系代数等价变换规则的优化方法：代数优化

物理优化：基于规则的启发式优化，基于代价估算的优化，两者结合的优化方法

（离散）握手定理

握手定理：有n个人握手，每人握手x次，握手总次数为S，必有S= nx/2。

顶点度数之和为边数之和的两倍。

推论 任何图(无向的或有向的)中，奇度顶点的个数是偶数

（数据库）事务的四个特点（ACID）

ACID，是指数据库管理系统（DBMS）在写入/异动资料的过程中，为保证事务(transaction)是正确可靠的，所必须具备的四个特性：原子性（Atomicity，或称不可分割性）、一致性（Consistency）、隔离性（Isolation，又称独立性）、持久性（Durability）。 • 原子性：一个事务(transaction)中的所有操作，要么全部完成，要么全部不完成，不会结束在中间某个环

节。事务在执行过程中发生错误，会被回滚（Rollback）到事务开始前的状态，就像这个事务从来没有执行过一样。

• 一致性：在事务开始之前和事务结束以后，数据库的完整性没有被破坏。这表示写入的资料必须完全符

合所有的默认规则，这包含资料的精确度、串联性以及后续数据库可以自发性地完成预定的工作。 隔离性：当两个或者多个事务并发访问（此处访问指查询和修改的操作）数据库的同一数据时所表现出的相互关系。事务隔离分为不同级别，包括读未提交(Read uncommitted)、读提交（read

committed）、可重复读（repeatable read）和串行化（Serializable）。 •

• 持久性：在事务完成以后，该事务对数据库所作的更改便持久地保存在数据库之中，并且是完全的。

（数据库）基本的关系操作有哪些

关系模型中常用的关系操作包括查询操作和插入、删除、修改操作两大部分。

关系的查询表达能力很强，是关系操作中最主要的部分。查询操作可以分为： 选择、投影、连接、除、并、差、交、笛卡尔积等。

其中，选择、投影、并、差、笛卡尔积是五种基本操作。

（数据库）数据库故障的种类

1、 事务内部的故障2、系统故障3.介质故障4.计算机病毒

（数据库）SQL的主键约束和唯一约束有什么区别

主键不能为空

而唯一可以为空

相同的就是 都不允许重复

（数据库）2PL协议

1) 两段锁协议是指所有事务必须分两个阶段对数据项加锁和解锁。

1)在对任何数据进行读、写操作之前，首先要申请并获得对该数据的封锁；

2)在释放一个封锁之后，事务不再申请和获得任何其他封锁。

“两段”的含义是，事务分为两个阶段：第一阶段是获得封锁，也称为扩展阶段。在这阶段，事务可以申请获得任何数据项上的任何类型的锁，但是不能释放任何锁。第二阶段是释放封锁，也称为收缩阶段。在这阶段，事务释放已经获得的锁，但是不能再申请任何锁。

（数据库）关系完整性包括哪三个方面

1. 实体完整性2.参照完整性3.用户定义的完整性

（数据库）数据库恢复策略有哪几种

1. 数据转储（数据冗余）2.日志文件、

数据字典通常包含5个部分

1.数据项2.数据结构3.数据流4.数据存储5.处理过程

（数据库）视图（View）的作用及优点

（数据库）数据库三要素

数据模型的三要素： 一般而言，数据模型是一组严格定义的概念的集合。这些概念精确地描述了系统的静态特征（数据结构）、动态特征（数据操作）和完整性约束条件，这就是数据模型的三要素。

（数据库）索引技术、日志文件

目的：提供多种存储路径，加快查找速度。

建立索引需要考虑的问题：1.没有查询、统计的需要则不建2.数据增删改频繁，系统会花费许多时间来维护索引，从而降低了查询效率

（数据库）完整性与安全性的区别

完整性和安全性是两个不同的概念。前者是为了防止数据库中存在不符合语义的数据，防止错误信息的输入和输出造成的无效操作和错误结果，而后者是防止数据库被恶意的破坏和非法的存取。当然，完整性和安全性是密切相关的。特别是从系统实现的方法来看，某一种机制常常既可以用于安全保护亦可用于完整性保证。

（离散）无向图中某条边出现在所有生成树中的充要条件

这条边是割边（删去这条边后，连通分支会增加）

（编译）LR(0)

（离散）一棵树有1个度为4的结点，1个度为3的结点，1个度为2的结点，其余全为叶子结点，问一共有多少个叶子？

5个

（编译/离散）闭包运算

数学中，对一个集合的成员进行某种运算，生成的仍然是这个集合的成员，则该集合被称为在某个运算下闭合。 例如，实数在减法下闭合，但自然数不行: 自然数 3 和 7 的减法 3 − 7 的结果不是自然数。

（离散）17条边的简单图最少多少个顶点

n阶完全图的边数n(n-1)/2，完全图的度是最大的

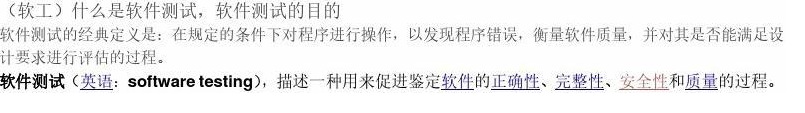
n>=7

（软工）白盒测试、黑盒测试

黑盒测试也称功能测试或数据驱动测试，它是在已知产品所应具有的功能，通过测试来检测每个功能是否都能正常使用，在测试时，把程序看作一个不能打开的黑盆子，在完全不考虑程序内部结构和内部特性的情况下，测试者在程序接口进行测试，它只检查程序功能是否按照需求规格说明书的规定正常使用，程序是否能适当地接收输入数锯而产生正确的输出信息，并且保持外部信息（如数据库或文件）的完整性。

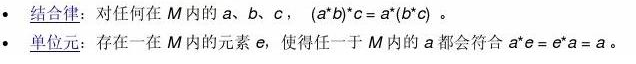
白盒测试也称结构测试或逻辑驱动测试，它是知道产品内部工作过程，可通过测试来检测产品内部动作是否按照规格说明书的规定正常进行，按照程序内部的结构测试程序，检验程序中的每条通路是否都有能按预定要求正确工作，而不顾它的功能，白盒测试的主要方法有逻辑驱动、基路测试等，主要用于软件验证。

灰盒测试，确实是介于二者之间的，可以这样理解，灰盒测试关注输出对于输入的正确性，同时也关注内部表现，但这种关注不象白盒那样详细、完整，只是通过一些表征性的现象、事件、标志来判断内部的运行状态，有时候输出是正确的，但内部其实已经错误了，这种情况非常多，如果每次都通过白盒测试来操作，效率会很低，因此需要采取这样的一种灰盒的方法。

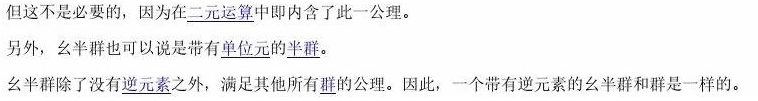


C:\Users\Administrator\AppData\Roaming\Tencent\Users\949611520\QQ\WinTemp\RichOle\ER5ZB_S384L]EDQNYX(W_{P.png

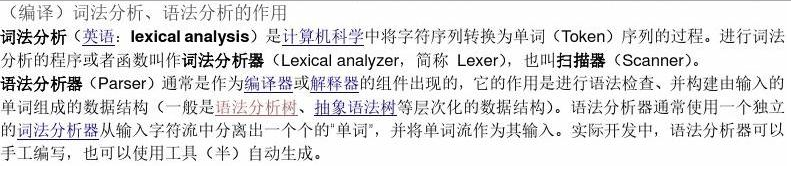
（离散）独异点的\*运算表包涵什么



C:\Users\Administrator\AppData\Roaming\Tencent\Users\949611520\QQ\WinTemp\RichOle\54`7S(@D5X1CY3OL8NWGHJH.png



通常也会多加上另一个公理：



（离散）什么是联结词完备集

设S是一个联结词集合，如果任何n(n≥1)元真值函数都可以由仅含S中的联结词构成的公式表示，则称S是联

结词完备集

S={┐,∧,∨}是联结词完备集

（软工）什么是UML

统一建模语言（UML，英语：Unified Modeling Language）是非专利的第三代建模和规约语言。UML是一种开放的方法，用于说明、可视化、构建和编写一个正在开发的、面向对象的、软件密集系统的制品的开放方法。UML展现了一系列最佳工程实践，这些最佳实践在对大规模，复杂系统进行建模方面，特别是在软件架构层次已经被验证有效。

（算法）阐述对一个数据集进行等价类划分的基本思想和过程

并查集应用:数据结构书p267

（离散）简述欧拉回路的基本含义及一个应用。另，哈密顿回路

对于一个给定的连通图，怎样判断是否存在着一个恰好包含了所有的边，并且没有重复的路径？这就是一笔画问题。用图论的术语来说，就是判断这个图是否是一个能够遍历完所有的边而没有重复。这样的图现称为欧拉图。这时遍历的路径称作欧拉路径（一个环或者一条链），如果路径闭合（一个圈），则称为欧拉回路

一笔画问题

哈密顿图（英语：Hamiltonian path，或Traceable path）是一个无向图，由天文学家哈密顿提出，由指定的起点前往指定的终点，途中经过所有其他节点且只经过一次。在图论中是指含有哈密顿回路的图，闭合的哈密顿路径称作哈密顿回路（Hamiltonian cycle），含有图中所有顶的路径称作哈密顿路径。

第29/33页

（？）简述对一个集合可能的表示方法（至少描述两种）

列举法，描述法..

（离散）简述逻辑命题与逻辑谓词的关系

？

（编译）简述用于定义语言句法的文法类型及其主要特征

乔姆斯基把方法分成四种类型，即0型、1型、2型和3型。 0型，无限制文法。无限制文法是对文法的产生式左右两侧都没有限制的。

1型，上下文相关文法。此文法对应于线性有界自动机。它是在0型文法的基础上每一个α→β,都有|β|>=|α|。这里的|β|表示的是β的长度。另外允许Sε

2型，上下文无关文法。它对应于下推自动机。2型文法是在1型文法的基础上,

再满足：每一个

α→β

都有α是非终结符。如A->Ba,符合2型文法要求。

3型，正规文法。它对应于有限状态自动机。它是在2型文法的基础上满足:A→α|αB（右线性）或A→α|Bα（左线性）。

（编译）简述预测分析法LL(1)的基本思想

自顶向下句法分析以文法开始符号为根，试图构造以输入符号串为树叶的推导树，每次都以最左边的那个非终结符为子树根，选择适当的产生式，往下生长子树。

预测分析表进行处理。

（编译）简述LR(1)分析法的基本思想

“向前看”，减少移近-规约冲突。

（编译）简述在程序流图中寻找循环的基本思想

注意Back Edge

第30/33页

（离散）举例偏序关系、全序关系，说明区别

偏序只对部分元素成立关系R，全序对集合中任意两个元素都有关系R。

例如：

集合的包含关系就是半序，也就是偏序，因为两个集合可以互不包含；

而实数中的大小关系是全序，两个实数必有一个大于等于另一个；

又如：复数中的大小就是半序，虚数不能比较大小。

知识面

（广）图灵机

图灵机，又称确定型图灵机，是英国数学家阿兰·图灵于1936年提出的一种抽象计算模型，其更抽象的意义为一种数学逻辑机，可以看作等价于任何有限逻辑数学过程的终极强大逻辑机器。

（离散）罗素悖论

（广）ACM全称，是哪国的

计算机协会（Association of Computing Machinery, 简称

ACM

）

是一个世界性的计算机从业员专业组织，创立于1947年，是世界上第一个科学性及教育性计算机学会。

（广）IEEE全称

电气电子工程师学会（英语：Institute of Electrical and Electronics Engineers，简称为IEEE，英文读作“i

第31/33页

triple e”[ai trɪpl i:]）是一个建立于1963年1月1日的国际性电子技术与电子工程师协会，亦是世界上最大的专业技术组织之一，拥有来自175个国家的36万会员。

除设立于美国纽约市的总部以外，亦在全球150多个国家拥有分会，并且还有35个专业学会及2个联合会。其每年均会发表多种杂志、学报、书籍，亦举办至少300次的专业会议。

（广）图灵奖及获得者

图灵奖（Turing Award，又译杜林奖），是计算机协会（ACM）于1966年设立的，又叫“A.M. 图灵奖”，专门奖励那些对计算机事业作出重要贡献的个人。其名称取自世界计算机科学的先驱、英国科学家，英国曼彻斯特大学教授艾伦·图灵（Professor Alan Turing），这个奖设立目的之一是纪念这位现代电脑、计算机奠基者。获奖者必须是在计算机领域具有持久而重大的先进性的技术贡献。大多数获奖者是计算机科学家。

（广）图灵测试

图灵测试（英语：Turing test，又译图灵试验）是图灵提出的一个关于判断机器是否能够思考的著名试验，测试某机器是否能表现出与人等价或无法区分的智能。

如果一个人（代号C）使用测试对象皆理解的语言去询问两个他不能看见的对象任意一串问题。对象为：一个是正常思维的人（代号B）、一个是机器（代号A）。如果经过若干询问以后，C不能得出实质的区别来分辨A与B的不同，则此机器A通过图灵测试。

（广）实时系统的分类

实时系统（Real-time operating system,RTOS）的正确性不仅依赖系统计算的逻辑结果，还依赖于产生这个结果的时间。实时系统能够在指定或者确定的时间内完成系统功能和外部或内部、同步或异步时间做出响应的系统。因此实时系统应该在事先定义的时间范围内识别和处理离散事件的能力；系统能够处理和储存控制系统所需要的大量数据。 强实时

强实时系统（Hard Real-Time）：在航空航天、军事、核工业等一些关键领域中，应用时间需求应能够得到完全满足，否则就造成如飞机失事等重大地安全事故，造成重大地生命财产损失和生态破坏。因此，在这类系统的设计和实现过程中，应采用各种分析、模拟及形式化验证方法对系统进行严格的检验，以保证在各种情况下应用的时间需求和功能需求都能够得到满足。

弱实时

弱实时系统（Soft Real-Time）：某些应用虽然提出了时间需求，但实时任务偶尔违反这种需求对系统的运行以及环境不会造成严重影响，如视频点播（Video-On-Demand，VOD）系统、信息采集与检索系统就是典型的弱实时系统。在VOD系统中，系统只需保证绝大多数情况下视频数据能够及时传输给用户即可，偶尔的数据传输延迟对用户不会造成很大影响，也不会造成像飞机失事一样严重的后果。

（广）并行技术有哪些

并行计算（英语：parallel computing）一般是指许多指令得以同时进行的计算模式。在同时进行的前提下，可以将计算的过程分解成小部份，之后以并发方式来加以解决。

相对于串行计算，并行计算可以划分成时间并行和空间并行。时间并行即流水线技术，空间并行使用多个处理器执行并发计算，当前研究的主要是空间的并行问题。以程序和算法设计人员的角度看，并行计算又可分为数据并行和任务并行。数据并行把大的任务化解成若干个相同的子任务，处理起来比任务并行简单。

空间上的并行导致两类并行机的产生，按照麦克·弗莱因（Michael Flynn）的说法分为单指令流多数据流（SIMD）和多指令流多数据流（MIMD），而常用的串行机也称为单指令流单数据流（SISD）。MIMD类的机器又可分为常见的五类：并行矢量处理机（PVP）、对称多处理机（SMP）、大规模并行处理机（MPP）、工作站机群（COW）、分布式共享存储处理机（DSM）。

（广）什么是系列机

系列机指基本指令系统相同、基本体系结构相同的一系列不同型号的计算机。系列机的概念就是指先设计好一种系统结构，而后就按这种系统结构设计它的系统软件，按器件状况和硬件技术研究这种结构的各种实现方法。并按照

第32/33页

速度、价格等不同要求，分别提供不同速度、不同配置的各档机器。系列机必须保证用户看到的机器属性一致。

（广）多核处理器存在的问题

目前在多核技术的开发中需要解决3个重要问题：一是多核之间的竞争关系如何协调，二是多核的负载均衡如何实 现，三是对如何实现对多核中内存、cache等的管理。要解决这些问题，就需要软件开发和硬件开发在早期阶段进行合作，对某些功能进行定义。

（广）什么是IP电话

IP电话（简称VoIP，源自英语Voice over Internet Protocol；又名宽带电话或网络电话）是一种透过互联网或其他使用IP技术的网络，来实现新型的电话通讯。过去IP电话主要应用在大型公司的内联网内，技术人员可以复用同一个网络提供数据及语音服务，除了简化管理，更可提高生产力。随着互联网日渐普及，以及跨境通讯数量大幅飙升，IP电话亦被应用在长途电话业务上。由于世界各主要大城市的通信公司竞争日剧，以及各国电信相关法令松绑，IP电话也开始应用于固网通信，其低通话成本、低建设成本、易扩充性及日渐优良化的通话质量等主要特点，被目前国际电信企业看成是传统电信业务的有力竞争者。