软件设计师

# 计算机系统知识

## 不同进制数值之间的转化

常用的进制：二进制，十进制，十六进制，

之间的转换没有转化公式时 我们借助十进制进行转换

## 数的表示

对于计算机数的表示只有一种表

示，就是二进制的表示方式，

不过对于同一个数的二进制我们也是有不同的二进制可以表示的，

各种数在计算机内的表示形式我们称之为机器数，

机器数是由1/0 组成的，就是我们常说的二进制的数字

原码：最高位是符号位，就是最高位仅仅是表示数的正还是负，不参与数的计算表达

符号位：0 正1 负

优点：表示和数的真值对应关系简单，相互转化比较容易，据说进行乘除法时方便，但是加减法联方便

注：零有两个编码[+0]=00000000 [-0]=11111111

反码：和源码一致的是最高位是符号位 ，0正 1 负 ，

和原码不同的是当最高位是1时，剩余的数进行取反才是这个数对应的二进制值，而是0时则不用，

注：零有两个编码[+0]=00000000 [-0]=10000000

补码：和源码一致的是最高位是符号位 ，0正 1 负 。其他的不同的是最高位是0时，其余车几位就是这个数的而进制数，为1 时着需要进行再次的操作，其他的数按位取反，同反码，之后在将末尾加1，才是这个数的二进制数值。

移码：这个一般是用来表示浮点数的

定点数/浮点数

### 校验码

奇偶校验，海明码，循环冗余校验

常见的三种奇偶校验：水平校验，垂直校验，水平垂直校验

## 数的运算

数的四则运算

## 计算机的组成

基本组成：运算器，控制器，存储器，输入设备，输出设备，以及其它

运算器和控制器等部件被集成到一起就是是统称的CPU

输入输出合称外设。

CPU，存储器，外设，总线。····

### 基本硬件系统：

运算器/控制器/存储器/输入/输出。

运算器+控制器=CPU----被集成

运算器：算术逻辑单元+累加寄存器+数据缓冲寄存器+状态条件寄存器

控制器：指令寄存器+程序计数器+地址寄存器+指令译码器+

### 存储系统：

高速缓存（Cache），内存（主存储器），外存（辅助存储器）

CPU不可以直接读取外存中的数据和程序，，想获得外存的数据必须将内容加载进内存才可以

不同分类标准有不同叫法：

按位置：内存，外存

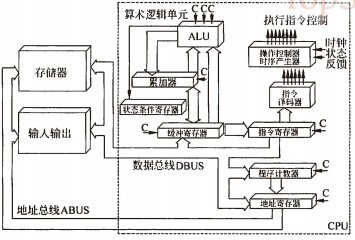
按材料：磁存，半导体，光存、

按工作方式：读写，只读

按访问方式：按地址访问，按内容访问（相联）

按寻址方式：随机，顺序，直接

### CPU的功能



是计算机的核心部件，负责程序指令的获取和对指令的译码并进行执行

其主要的功能体现就是控制，

程序控制，

操作控制

时间控制，

数据处理，

#### CPU和外设直接数据的传送方式

直接程序控制方式

无条件传送：无条件的与CUP进行交换数据

程序查询方式：在使用之前先查询设备的状态，状态满足后进行数据传输，

优点是易于理解，缺点是降低了CPU的效率

中断方式

中断处理方法

中断优先级控制

直接存储器存取方式

输入输出处理机

## 总线结构

广义区分：内部总线（和芯片一级链接），外部总线（又名通信总线），系统总线（和插板一级链接）

系统总线：ISA （工业标准），EISA，PCI（当下最流行）

外部总线：RS-232-C：串行/SCSI：并行/USB：即插即用/IEEE-1394：串行

## 指令系统

指令体系：

堆栈、累加器、寄存器

复杂指令集系统计算（CISC）

精简指令集计算机（RISC）

指令控制方式：

流水，顺序，重叠 三种

顺序方式：

控制简单，执行速度慢，机器各部件利用率低

重叠方式：

流水方式；

流水是在重叠的基础上加入了并发的情况

指令吞吐率/流水建立时间

率：单位时间内流水线处理机流出的结果数，对于指令而言就是单位时间指令的执行数量，如果没个子流程的所用的时间不一样，则按照耗时最久的那个的倒数计算。

P=1/嘴唇子流程时间

建立时间：从开始多次时间到达了最大吞吐率

# 程序语言

低级语言：给计算机看的 只有01 组成

高级语言：给人看的，人能够通过这个类解读认识的。

源程序：由高级语言书写而成

## 程序设计的基本组成

常量，变量，全局量，局部量，数据类型

数据类型：基本类型 int， char，float，double，Boolean

运算部分：

逻辑运算，关系运算，算术运算，还有不常用的位运算

控制部分：

顺序结构，判断结构 ，选择结构，循环结构

函数部分：

函数定义，函数声明，函数使用，

### 编译程序的基本原理：

词法分析

语法分析

语义分析

中间代码生成

代码优化

目标代码生成

符号表管理

出错管理

### 有限自动机

这个东西是一种识别装置的抽象概念，它可以正确的识别正规集。

确定的有限自动机（DFA）：

不确定的有限自动机（NFA）：

知识点：

Python，js，Tjs都是 弱语言

### 中间代码优化和目标代码生成

优化就是对程序仅等价的转换，使得变化后的代码更好的，更有效的转为目标代码。

所谓的等价就是不改变程序的最终的执行结果。

所谓的有效是指，程序在断的时间内可以完成运行，同时占用的空间是比较少的，

**谁来优化**

优化的这个动作可以出现在各个阶段的，而不是在一个规定的阶段，也是不是某个人的职责，而是开发者的自觉，尽力是自己的代码活的时间更长。

# 操作系统知识

作用：

通过资源管理界面提高计算机的效率

改善人机界面，向用户提供友好的工作环境

基本特征：

并发性，共享性，虚拟性，不确定性

## 操作系统分类：

批处理操作系统，分时操作系统，实时操作系统，网络操作系统，分布式操作系统，微型计算机操作西戎，嵌入式操作系统

批处理系统又分为 单到批处理和多道批处理

分时操作系统：

特点：多路新，独立性，交互性和及时性

## 死锁

产生的原因:

资源竞争以及进程推进顺序非法

产生的必要条件：

互斥条件

请求保持条件

不可剥夺条件

环路条件

死锁的处理：

预防

避免

检测

解除

## 设备管理

按数据组织分：块设备，字符设备

按资源分配分：独占，共享，虚拟

按数据传输分：低速，中速，高速

按可否交互分：可以交互，不可以交互

按输入输出分：人机通信，机机通信

### 设备管理使用的技术：

#### 通道技术

目的使数数据传输独立于CPU，CPU仅仅是发出命令和接受命令就可以。Cpu想通道发出I/O命令，通道接到命令后开始工作，在完成命令后在发送中断请求的命令到CPU即可。

这样可以使CPU从复杂的I/O操作中脱离出来。

根据信息交换方式的不同，将通道分为：字节多通路/数组选择通道/数组多路通道

通道很贵 导致计算机系统中的通道数量是有限的

#### DMA技术

直接主存存取

这样的是原理是数据的主存和I/O设备直接直接成块传递，这样就不再需要CPU的任何干涉，仅仅是需要CPU过程开始和结束时发送指令。这样在数据的传输过程中，CPU就可以进行其他的操作了。

#### 缓冲技

缓冲技术可以提高外设的利用率，尽可能的使外设处于忙碌的状态

缓冲又可以分为硬件缓冲和软件缓冲。

引入缓冲的原因的：

缓和CPU和I/O设备之间的速度不匹配的矛盾

减少对Cpu的中断频率，放宽对中断响应时间的限制

提高CPU和I/O设备之间的并行性

#### SPOOLing 技术：

用一类物理设备模拟另一类物理设备的技术，是独占使用的设备变成多台虚拟设备的一种技术，也是一种速度匹配的技术、

## 磁盘驱动管理

先来先服务（FCFS）

最短寻道时间优先（SSTF）

扫描算法（SCAN）

单向扫描算法（CSCAN）

### 旋转调度算法

同磁道不同编号、不同磁道不同编号、不同磁道相同编号

## 文件管理

固态硬盘

存储介质：闪存（FLASH芯片）、DRAM

## 系统的安全性与可靠性

安全性：系统级，用户级，目录级，文件级

可靠性：系统抵抗性，预防各种物理性破坏，人为性破坏

### 加密技术

常见的对称加密技术

数据加密技术（DES）

三重（3DES或者TDEA）

RC-5

国际数据加密算法（IDEA）

高级加密标准（ASE）

常见的非对称加密技术

非对称的加密技术（），在进行加密和解密时，使用的是不同的一套密码，这里的一套是分公钥和私钥两种。

加密时，使用私钥时，解密就只可以使用对应的公钥才可以。反之亦然！

一种加密标准RSA

# 软件工程基础知识

## 计算机软件

* 系统软件
* 应用软件
* 工程/科学软件
* 嵌入式软件
* 产品线软件
* Web应用
* 人工智能软件
* 开发软件
* 网络资源
* 开源软件

## 软件的生命周期：

可行性分析与项目开发计划

需求分析

概要设计

详细设计

编码

测试

维护

## 软件过程模型

瀑布模型/增量模型/演化模型/喷泉模型/基于构件的开发模型和形式化方法模型、

瀑布模型的优点：

容易理解，管理成本低，

瀑布模型的缺点：

客户必须能够完整的正确的清晰的表示他们等要求。对项目的的风险控制能力较弱，从而导致项目常常延期，开发费用超出预期。

## 统一过程模型

起始阶段，精华阶段，构件阶段，移交阶段

## 测试方法

静态方法和动态方法

静态测试是指被测试的程序不在机器上运行，而是采用人工检查的方法和计算机静态分析的为辅助的方法对程序进行测试

动态测试也是我们经常进行的测试，一般使用白盒和黑盒（功能测试）测试

黑合测试又称作功能测试，

黑盒测试的目的是：

确定软件的功能是不是有错误和遗漏

确定界面是不是有误，输入的值是不是正确的接受，同时输出是不是对的

判断数据结构和外部的数据库访问是不是正确

软件的性能是不是已经是满足要求了

是不是有初始化和终止性的错误

黑盒的测试的技术：等价类划分，边界值分析，错误推测，因果图

白盒测试又称为结构测试，

使用的技术一般是逻辑覆盖，循环覆盖，基本路径测试

## 软件项目管理

软件管理涉及的范围：

简称4P，即人(Person) 产品(Product) 过程(Procedure) 项目(Project)

# 系统开发与并行

每一款软件都有其生命周期，就如同一个人一样，有出生到工作生活知道死亡。

软件的生命周期包括：孕育，诞生，成长，成熟，衰亡

## 开发的几种模型

瀑布模型：

给出了了软件生存周期的各阶段的固定顺序，只有在上一个阶段完成后才可以进入下一个阶段。

优点：可以保证软件的质量。

缺点：缺乏灵活性，无法通过开发活动来澄清本来就不明确的需求，有可能导致需求开发完成了才发现不并不是客户需要的。

演化模型：

在接到一个需求后，迅速分析，构建出软件的初始可运行版本，，通常认识这是原型。然后客户根据这个原型在提出意见和对原型的改进措施，之后在构建出改进后的版本再给客户进行提已经和修改，之后再改，一直循环直到完成。

该模型使用于对于软件缺乏准确认识的情况。

螺旋模型：

结合演化模型和瀑布模型

喷泉模型：

该模型具有迭代和无间隙特性

## 软件开发方法

结构化方法，jackson 方法，原型化法，面向对象开发方法

## 系统分析

### 数据流图（DFD），数据字典，加工逻辑的描述

顶层图只有一张，

0层图也是只有一张，

数据流图的组成部分：

外部实体：数据的发源地或者数据归宿地

加工:对数据进行处理/转化

数据存储：数据保存

数据流：标识数据走向

数据流图的设计原则：

数据守恒原则，

数据加工守恒原则，

对于每一个加工必须有进有出，

外部实体之间没有数据流

外部实体和数据存储之间没有数据流

数据存储之间没有数据流

父图子图之间的平衡原则

数据流与加工有关，且必须经过加工

### 数据流图分析技巧

把题目看清

利用数据流图的数据平衡原则

子图是父图中的一部分的加工处理，他们的输入输出应当是保持一致的。在上一级的数据图中有几个数据流么他的子图也一定有相同数量的数据流，而且他们数输出方向是一致的。

残缺的数据流图补全

这时的补充一般是补充外部实体、补全缺失的数据流和数据存储

### 试题解答技巧

数据流图部分的核心考核重点是：补充外部实体、补充缺失的数据流、找出错误的数据流、补充加工处理

两个技巧：

紧紧抓紧试题的系统说明部分，数据流图和系统说明有着严格的对应关系，系统说明的每一句话都可以在数据流图中找到对应的部分，

掌握数据平衡原则，平衡分为子图和父图之间的数据平衡还包括每张数据图的数据输入和输出之间的平衡。

### 系统设计基本原理

1抽象

2模块化

3信心隐蔽

4模块独立

模块化设计方法（SD）

## 系统实施

### 系统测试和调测

目前在程序运营之期尽最大的努力降低程序的bug出现率。

测试包括：软件测试，硬件测试，网络测试

软件测试策略：

单元测试/组装测试/确认测试/系统测试

黑盒测试/白盒测试

黑盒：也称作性能测试。白盒又称作结构测试

调测：

测试出来哦问题，进行定位分析并处理问题。

系统可维护性的评价包括：可理解性，可测试性，可修改性，移植工具

单元测检查的主要内容：边界测试，错误处理测试，路径测试，局部数据结构测试，模块接口测试。

软件需求中对软件产品的响应时间，吞吐量，价格等属性的要求都是属于非功能需求。

#### 题的出题类型：

数据流分析题，补全题

# 结构化开发方法

结构化开发方法包括：结构化分析，结构化设计，结构化程序设计构成。 这是一种面向数据流开发的方法

模块独立：

融合，

无直接耦合，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

偶然内聚，逻辑内聚，时间内聚，过程内聚，通信内聚，顺序内聚，功能内聚

## 数据流图 DFD

包含的元素：数据流、加工、数据存储、外部实体

数据流:有一组固定成分组成的数据，表示数据的流向。

数据流不是控制流，数据流之间的交互都是数据流可以通过数据流进行交互的。

加工：描述的是书数据流流入流出的之前的转换

加工这里不能是黑洞和灰洞

黑洞：有数据的输入没有数据的输出

灰洞：有足够的数据的输出没有足够的数据的输入

数据存储：顾名思义是存放数据的地方

# 面向对象技术

一切事物皆对象

什么样系统是一个好的系统呢？

使用者（外部）

易学易用，界面友好，正确使用时能够快速响应并给出正确的结果，效率高

开发者（内部）

易于修改和扩充，易于理解，易于测试和重用，易于与其他系统兼容和管理。

4个概念：

对象、消息、类、继承、多态、动态绑定。

面向对象设计语言：C++，Java，VisualC++ 。。。

## 面向对象设计原则：

单一职责原则

一个类应该仅仅有一个引起其变化的原因，一个类只做一种类型责任

开放—封闭原则

软件实体应该是可以扩展张的，即开放的，也有是不可修改的即是封闭的。

里氏替换原则

子类型必须可以替换他们的基类型。

依赖倒置原则

抽象不应该依赖细节，细节应该依赖抽象。

接口分离原则

不应该强迫客户依赖它们不用的方法，

以下的是五大原则之外的几个：

重用发布等价原则

重用粒度就是发布粒度

共同封闭原则

共同重用原则

无环依赖原则

稳定依赖原则

稳定抽象原则

## 面向对象程序设计（OOP）

面向对象程序设计（Object-Oriented Programming,OOP）

面向对象程序设计语言（Object-Oriented Programming Language ，OOPL）

类

同一个类的不同实例有着相同的数据结构

同一个类的不同实例可以用不同的值，婴儿可以具有不同的状态

实例的初始状态（初值）可以在实例化时确定

继承和类的层次结构

对象、消息传递和方法

对象是类的实例

对象自身引用

对象自身引用是OOPL中的一种特有的结构

重置

重置和覆盖是子类重新定义父类方法的一种实现，根本是一种动态的绑定机制的支持，

类属类

无实例的类

## UML

统一建模语言

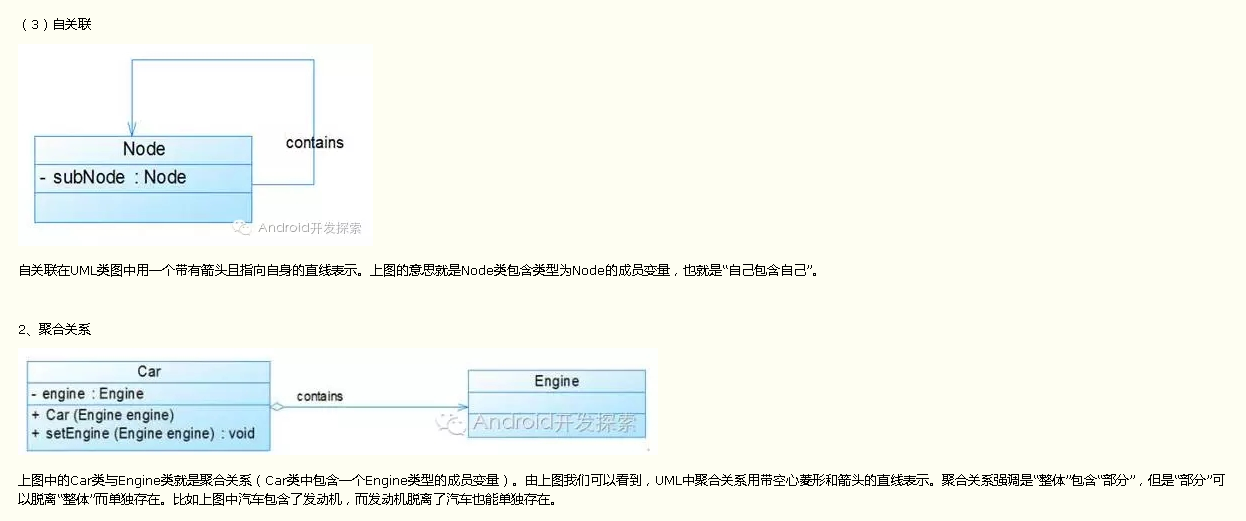
UML中有4种事务：结构事务，行为事务，分组事务，注释事务

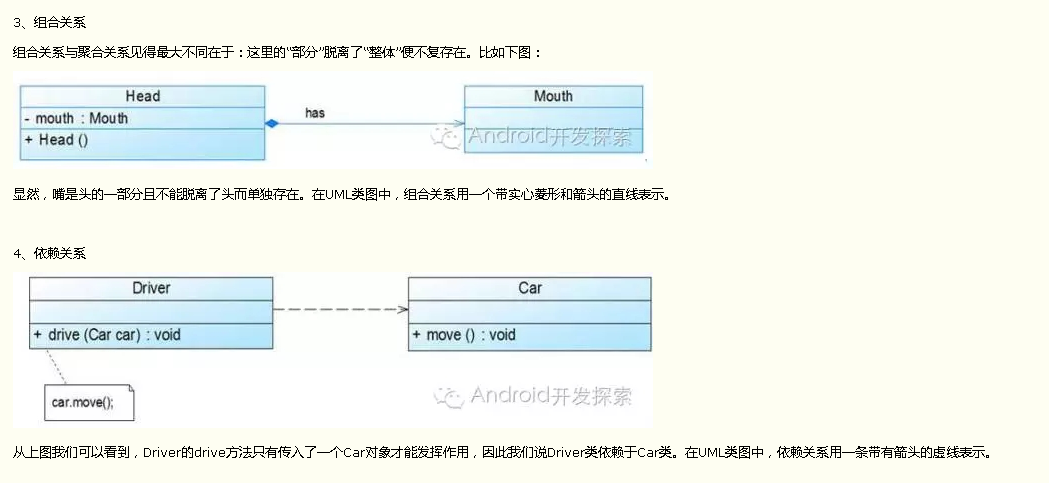
4中关系：依赖，关联，泛化和实现

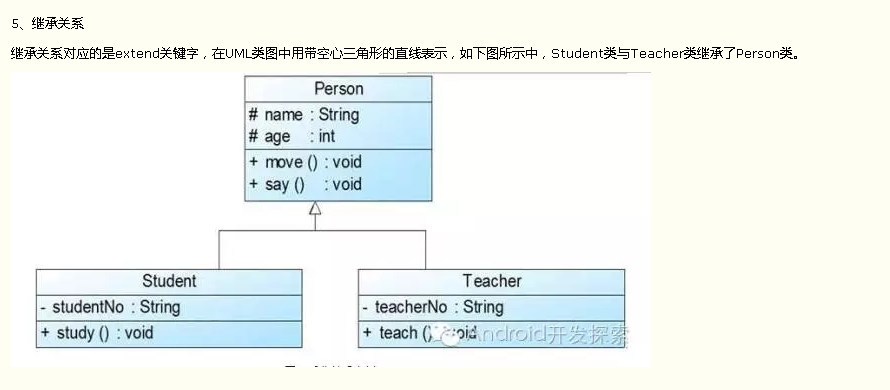
13种图：类图、对象图、用例图、序列图、通信图、状态图、活动图、构件图、组合结构图、部署图、包图、交互概览图、计时图，

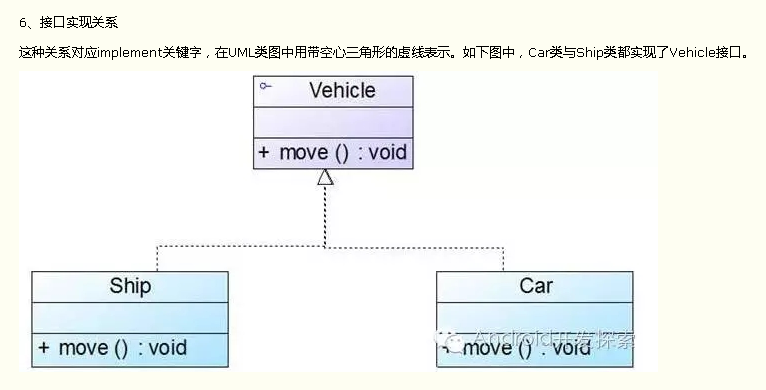
其中黄色的称为交互图。











### UML建模中常见的关系

#### 一、泛化关系

1.概念

泛化关系是指一个类（子类、子接口）继承另外一个类（父类，父接口）的功能，并可以增强它自己新功能的能力，这也被称作继承，另外，泛化还可以用来定义下列情况：当一个变量被声明承载某个给定的类的值时，可以使用类的实例作为值，例如一个变量被声明为拥有借贷，那么一个抵押对象就是一个合法的值。

2.表示方法



注：从子指向父的箭头，其中箭头为空的三角形

#### 二、实现

1.概念

相信大家对这个关系并不陌生，就是指一个类实现一个或多个接口，java中的关键字是implements 这里值得再次说明的是，类在实现接口的时候必须实现接口中的所有方法

2.表示方法



注：从类指向接口，箭头为空三角形

#### 三、依赖（弱依赖关系）

1.概念

表示类与类之间的连接，即一个类依赖于另一个类的定义，是单向的。例如有A、B两个类，A类使用了B类，其中B的变化会影响A，这就是一个依赖

2.表示方式



#### 四、关联（强依赖关系）

1.概念

表示类与类之间的连接，使一个类知道另一个类的属性和方法。有单向关联（单箭头）和双向关联（双箭头，不建议使用）。

2.表示方式



另外，关联关系的两端可以有一个基数，表示这个关联的类可以有几个实例

0..1：表示可以有0个或1个实例

0..\*：表示对实例的数目没有限制

1   ：表示只能有一个实例

1..\*：表示至少有一个实例

#### 五：聚合

1. 概念

关联关系的特例，是强的关联关系，表示整体和个体的关系（has a）。 注：在java语法方面，关联和聚合没有明显的区别，但是关联关系中的类出于同一个层次，而聚合关系明显在两个不同的层次（例如family和child）

2.表示方法



注：从整体指向个体，空的菱形

#### 六.组合

1.概念

关联关系的特例，表示的是一个contain-a的关系，是一种强聚合关系，整体和部分不可分，整体的生命周期结束，部分的生命周期也结束。与聚合的唯一区别就是部分不脱离整体。

2.表示方法



注：从整体指向个体，实心的菱形 总结： 关联关系的强弱排序：依赖-->关联--->聚合---->组合（由弱到强）

--------------------- 本文来自 丑八怪28 的CSDN 博客 ，全文地址请点击：https://blog.csdn.net/m0\_37630602/article/details/70207458?utm\_source=copy

## 设计模式

设计模式要素：

模式名称、问题、解决方案、效果、

设计模式

Abstract Factory 抽象工厂

Buileder 生成器

Factory Method 工厂方法

Prototype 原型

Singleton 单例

结构性设计模式

Adapter （适配器）

意图

将一个类的接口转化为另一个客户的接口。使得由于接口不一致导致不能一起工作的接口一起工作

实用性

想使用一个已经存在的类，而他的接口不符合要求

想创建一个可以复用的类，该类可以和其他的不相干的类或不可见的类进行一起工作

Bridge桥接

意图

将抽象部分和实现部分分离，使他们可以独立的变化

试用性

不希望在抽象和实现部分之间有个固定的绑定关系

类抽象和实现都应该使用子类的方式加以扩展

Composite组合

意图

将对象组合成树形结构以表示“部分—整体”的层次结构

实用性

想表示对象的部分-整体层次结构

希望用户忽略组合对像和单个对象的不同，用户将统一地使用组合结构中的所有对象

Decorator 装饰

意图

动态的给一个对象添加一个额外的职责。就增加功能而言，使用这个比使用生成子类的方法更合理

实用性

在不影响其他对象的情况下，以动态、透明的方式给单个对象添加职责

处理那些可以撤销的职责

当不能使用生成子类的方式进行扩展时。

Proxy代理

意图：

为其他的对象提供一种代理以控制这个对象的访问

适用点：

# 网络基础知识

计算机网络：

按通信距离分：局域网，城域网，广域网

按信息交换方式分：电路交换网，分组交换网，综合交换网

按拓扑类型分：星形网，环形网，树形网，总线型

按通信材质：双绞线，同轴，光线，卫星

按传输速度：高 中 低

使用范围：共用，私用

国际化标准组织（ISO） 提出开放系统互联参考模型（OSI）

IP地址：

ABCDE 五类：

A：1.0.0.1——126.255.255.254 126个网络 可容纳224-2个主机

B：128.0.0.1——191.255.255.254 1634个网络 65534个主机

C：192.0.1.1——223.255.255.254

D：224.0.0.1——239.255.255.254

E：以1111开头

# 数据库技术

与数据库系统有关的人员：

系统分析人员和数据库设计人员，应用程序员，最终的用户，数据库管理人员

数据库管理系统：（DBMS）

数据定义语言：（DDL）

数据库操作语言（DML）

## 数据模型

概念数据模型和基本数据模型

三要素：数据结构、数据操作、数据的约束条件

E-R模型：实体-联系模型

## 关系数据库SQL

SQl的基本组成：

数据定义语言，数据操作语言，事务控制，嵌入式SQL和动态SQL，完整性，权限管理

### 创建表：

简单建表

CREATE TABLE data\_dict (ID INTEGER NOT NULL PRIMARY KEY,data\_Code VARCHAR2(20) NOT NULL,data\_code\_name VARCHAR2(20) ,data\_data\_value varchar2(30),ext1 VARCHAR2(100) );

执行完 再查看数据对应的建表 sql

*-- Create table*

create table DATA\_DICT

(

ID INTEGER not null,

DATA\_CODE VARCHAR2(20) not null,

DATA\_CODE\_NAME VARCHAR2(20),

DATA\_DATA\_VALUE VARCHAR2(30),

EXT1 VARCHAR2(100)

)

tablespace USERS

pctfree 10

initrans 1

maxtrans 255;

*-- Create/Recreate primary, unique and foreign key constraints*

alter table DATA\_DICT

add primary key (ID)

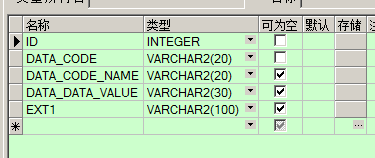
using index

tablespace USERS

pctfree 10

initrans 2

maxtrans 255;



Sql模板：

Create table table\_name (

列名 数据类型 约束条件，

列名 数据类型 约束条件，

列名 数据类型 约束条件，

。。。。

)

### 修改和删除表：

Alert talbe table\_name add 列名 数据类型 约束条件

Modify列名 数据类型 约束条件

Drop

Drop table table\_name;

### 创建删除索引

Create 索引类型 index 索引名 on 表名 （列名）；

Drop index 索引名字

### 创建删除视图

Create view 视图名 （列表名） aS select 查询的sql [with check option]

Drop view 视图名

### Sql查询

基本结构

简单查询

连接查询，

子查询，联合查询，聚合查询，分组查询

### 出题点

案例分析题，

## 数据结构

## 线性结构

线性结构是最简单的一种基本的数据结构，主要用于对客观世界中的具有单一前驱和后继的数据关系进行描述。线性结构的特点是数据元素之前呈现一种线性关系，即元素“一个接一个排列”。

非空线性表的特征：

存在唯一的一个第一个的元素

存在唯一的一个最后一个的元素

除了第一个元素外每个元素都有一个前驱

除了最后一个元素外每个元素都有一个后继

线性表的存储结构

顺序存储

优点可以随机存取表中的数据，缺点是添加和删除操作需要移动大量的元素

链式存储

优点是添加和删除操作不需要移动大量的元素

栈和队列

排序：简单排序，希尔排序，快速排序，堆排序，归并排序，基数排序，内部排序，外部排序

栈：先进后出

队列：先进先出

## 树：

节点分类：根节点，父节点，子节点

在树的内部又有内部节点的说法

## 查找方法

顺序查找，分块查找，折半查找，

## 简单排序：

直接插入排序：

最简单的一种排序方法

冒泡排序

简单选择排序

## 希尔排序

## 堆排序

## 归并排序

## 基数排序

## 快速排序

内部排序小结

|  |  |  |  |
| --- | --- | --- | --- |
| 排序方法 | 时间复杂度 | 辅助空间 | 稳定性 |
| 直接插入 | O（n2） | O(1) | 稳定 |
| 简单选择 | O（n2） | O(1) | 不稳定 |
| 冒泡排序 | O（n2） | O(1) | 稳定 |
| 希尔排序 | O（n1.3） | O(1) | 不稳定 |
| 快速排序 | O(nlogn) | O(logn) | 不稳定 |
| 堆排序 | O(nlogn) | O(1) | 不稳定 |
| 归并排序 | O(nlogn) | O(n) | 稳定 |
| 基数排序 | O(d(n+rd)) | O(rd) | 稳定 |

# 算法设计和分析

## 算法

5个基本的特征：有穷性，确定性，可行性，输入，输出

算法设计

算法分析

算法表示

时间复杂性

渐进符号

递归式

分治法

动态规划法

最优子结构和高度的重复性

贪心法

回溯法

# 软件系统分析设计

DFD

对问题进行结构化分析实施的步骤：

确定系统边界，画出系统环境图

自顶而下画出各层数据流图

定义数据字典

定义加工说明

将图、字典以及加工组成分析模型

结构化分析

这个动作的最终结果会得到系统的数据流图、数据字典和加工说明。

# 计算机硬件基础

## 计算机体系结构：

单指令流单数据流（SISD）

控制部分：一个 处理器：一个 主存模块：一个

单指令流多数据流（SIMD）

控制部分：一个 处理器：多个 主存模块：多个

多指令流单数据流（MISD）（被证明不可能实现）

控制部分：多个 处理器：多个 主存模块：多个

多指令流多数据流（MIMD）

控制部分：多个 处理器：多个 主存模块：多个

## 总线

STD总线：

是一种规模小，面向工业控制的8位系统总线，支持多处理器系统。

PCI总线（外部组件互连），主要将显卡、声卡、网卡、硬盘控制器等外围设备直接挂在CPU总线上。

Centronic总线：主要是将打印机等外设相连接