在输入输出控制方法中，采用（1）可以使得设备与主存间的数据块传送无需 CPU 干预。

(1)A.程序控制输入输出　B.中断　C.DMA　D.总线控制

**【答案】C**

**【解析】本题考查 CPU 中相关寄存器的基础知识。**

计算机中主机与外设间进行数据传输的输入输出控制方法有程序控制方式、中断方式、DMA 等。

在程序控制方式下，由 CPU 执行程序控制数据的输入输出过程。

在中断方式下，外设准备好输入数据或接收数据时向 CPU 发出中断请求信号，若 CPU 决定响应该请求,则暂停正在执行的任务,转而执行中断服务程序进行数据的输入输出处理，之后再回去执行原来被中断的任务。

在 DMA 方式下，CPU 只需向 DMA 控制器下达指令，让 DMA 控制器来处理数据的传送，数据传送完毕再把信息反馈给 CPU,这样就很大程度上减轻了 CPU 的负担，可以大大节省系统资源。

若某计算机采用 8 位整数补码表示数据，则运算（2)将产生溢出。

(2)A.-127+1　B.-127-1　C.127+1　D.127-1

**【答案】C**

**【解析】本题考查计算机中的数据表示和运算基础知识。**

采用 8 位补码表示整型数据时,可表示的数据范围为-128～127,因此进行 127+1 运算会产生溢出。

若内存容量为 4GB，字长为 32,则（3)。

(3) A.地址总线和数据总线的宽度都为 32

B.地址总线的宽度为 30,数据总线的宽度为 32

C.地址总线的宽度为 30,数据总线的宽度为 8

D.地址总线的宽度为 32,数据总线的宽度为 8

**【答案】A**

**【解析】本题考查计算机系统的总线基础知识。**

内存容量为 4GB，即内存单元的地址宽度为 32 位。字长为 32 位即要求数据总线的宽度为 32 位，因此地址总线和数据总线的宽度都为 32。

地址总线的宽度就是处理机寻址范围，若地址总线为 n 位，则可寻址空间为 2 的 n 次方字节。所以本题的可寻址空间为：4*1024*1024*1024*位，所以地址总线宽度为 32

设用 2KX4 位的存储器芯片组成 16KX8 位的存储器（地址单元为 0000H~3FFFH，每个芯片的地址空间连续），则地址单元 0B1FH 所在芯片的最小地址编号为（4）。

(4)A.0000H　　B.0800H　　C.2000H　　D.2800

**【答案】B**

**【解析】本题考查计算机系统中存储部件的基础知识。**

由 2KX4 位的存储器芯片组成容量为 16KX8 位的存储器时，共需要 16 片（16KX8/（2KX4））。用 2 个存储器芯片组成 2KX8 的存储空间（每个芯片的地址空间连续），16KX8 位的存储空间共分为 8 段，即 0000H~07FFH，0800H~0FFFH ，1000H~17FFH，1800H~1FFFH，2000H~27FFH，2800H~2FFFH，3000H~37FFH，3800H~3FFFH。显然，地址单元 0B1FH 所在芯片的起始地址为 0800H。

编写汇编语言程序时，下列寄存器中程序员可访问的是（5）。

(5)A.程序计数器（PC）　　B.指令寄存器（IR）

C.存储器数据寄存器（MDR）　　D.存储器地址寄存器（MAR）

**【答案】A**

**【解析】本题考查 CPU 中相关寄存器的基础知识。**

指令寄存器（IR)用于暂存从内存取出的、正在运行的指令，这是由系统使用的寄存器，程序员不能访问。

存储器数据寄存器（MDR)和存储器地址寄存器（MAR)用于对内存单元访问时的数据和地址暂存，也是由系统使用的，程序员不能访问。

程序计数器（PC)用于存储指令的地址，CPU 根据该寄存器的内容从内存读取待执行的指令，程序员可以访问该寄存器。

正常情况下，操作系统对保存有大量有用数据的硬盘进行（6)操作时，不会清除有用数据。

(6) A.磁盘分区和格式化　　B.磁盘格式化和碎片整理

C.磁盘清理和碎片整理　　D.磁盘分区和磁盘清理

**【答案】C**

**【解析】本题考查计算机系统的基础知识。**

磁盘格式化是指把一张空白的盘划分成一个个小区域并编号，以供计算机储存和读取数据。格式化是一种纯物理操作，是在磁盘的所有数据区上写零的操作过程，同时对硬盘介质做一致性检测，并且标记出不可读和坏的扇区。由于大部分硬盘在出厂时已经格式化过，所以只有在硬盘介质产生错误时才需要进行格式化。

磁盘分区就是将磁盘划分成一块块的存储区域。在传统的磁盘管理中，将一个硬盘分为两大类分区：主分区和扩展分区。主分区是能够安装操作系统、能够进行计算机启动的分区，这样的分区可以直接格式化，然后安装系统，直接存放文件。

磁盘里的文件都是按存储时间先后来排列的，理论上文件之间都是紧凑排列而没有空隙的。但是，用户常常会对文件进行修改，而且新增加的内容并不是直接加到原文件的位置的，而是放在磁盘存储空间的最末尾，系统会在这两段之间加上联系标识。当有多个文件被修改后，磁盘里就会有很多不连续的文件。一旦文件被删除，所占用的不连 续空间就会空着，并不会被自动填满，而且，新保存的文件也不会放在这些地方，这些 空着的磁盘空间，就被称作“磁盘碎片”。因此，硬盘的每个分区里都会有碎片。碎片太多，其他的不连续文件相应也多，系统在执行文件操作时就会因反复寻找联系标识，工作效率大大降低，直接的反映就是感觉慢。

磁盘清理将删除计算机上所有不需要的文件（这些文件由用户或系统进行确认）。

磁盘碎片整理，就是通过系统软件或者专业的磁盘碎片整理软件对电脑磁盘在长期使用过程中产生的碎片和凌乱文件重新整理，释放出更多的磁盘空间，可提高电脑的整体性能和运行速度。

如果使用大量的连接请求攻击计算机，使得所有可用的系统资源都被消耗殆尽，最终计算机无法再处理合法用户的请求，这种手段属于 (7)攻击。

(7)A.拒绝服务　　B.口令入侵　　C.网络监听　　D.IP 欺骗

**【答案】A**

**【解析】本题考查网络安全中网络攻击的基础知识。**

网络攻击的主要手段包括口令入侵、放置特洛伊木马程序、拒绝服务（DoS)攻击、端口扫描、网络监听、欺骗攻击和电子邮件攻击等。

口令入侵是指使用某些合法用户的账号和口令登录到目的主机，然后再实施攻击活动。

特洛伊木马（Trojans)程序常被伪装成工具程序或游戏，一旦用户打开了带有特洛伊木

马程序的邮件附件或从网上直接下载，或执行了这些程序之后，当用户连接到互联网上时，这个程序就会向黑客通知用户的 IP 地址及被预先设定的端口。

拒绝服务（DoS)攻击目的是使计算机或网络无法提供正常的服务。最常见的拒绝服务攻击有网络带宽攻击和连通性攻击。带宽攻击指以极大的通信量冲击网络，使得所有可用网络资源都被消耗殆尽，最后导致合法的用户请求无法通过。连通性攻击是指用大量的连接请求冲击计算机，使得所有可用的操作系统资源都被消耗殆尽，最终计算机无法再处理合法用户的请求。

端口扫描就是利用 Socket 编程与目标主机的某些端口建立 TCP 连接、进行传输协议的验证等，从而侦知目标主机的扫描端口是否处于激活状态、主机提供了哪些服务、提供的服务中是否含有某些缺陷等。

网络监听是主机的一种工作模式，在这种模式下，主机可以接收到本网段在同一条物理通道上传输的所有信息。使用网络监听工具可轻而易举地截取包括口令和账号在内的信息资料。

欺骗攻击是攻击者创造一个易于误解的上下文环境，以诱使受攻击者进入并且做出缺乏安全考虑的决策。IP 欺骗是欺骗攻击的一种，IP 欺骗实现的过程是：使得被信任的主机丧失工作能力，同时采样目标主机发出的 TCP 序列号，猜测出它的数据序列号。然后，伪装成被信任的主机，同时建立起与目标主机基于地址验证的应用连接。如果成功，黑客可以使用一种简单的命令放置一个系统后门，以进行非授权操作。

ARP 攻击造成网络无法跨网段通信的原因是__(8)__。

(8) A.发送大量 ARP 报文造成网络拥塞

B.伪造网关 ARP 报文使得数据包无法发送到网关

C.ARP 攻击破坏了网络的物理连通性

D.ARP 攻击破坏了网关设备

**【答案】B**

**【解析】本题考查网络攻击中 ARP 攻击的原理。**

ARP 攻击（ARP 欺骗）是欺骗攻击的一种，通过伪造 IP 地址和 MAC 地址，能够在网络中产生大量的 ARP 通信量使网络阻塞，如果伪造网关的 IP 地址和 MAC 地址对，则所有发往网关的 IP 包将因为 MAC 地址错误而无法到达网关(ARP 攻击一般会将 MAC 地址改为发起 ARP 攻击的主机地址)，造成无法跨网段通信。

处理 ARP 攻击的方法为首先断开 ARP 攻击主机的网络连接，然后用“arp-d”命令清除受攻击影响的 ARP 缓存。

下列选项中，防范网络监听最有效的方法是(9).

(9)A.安装防火墙　　B.采用无线网络传输　C.数据加密　　D.漏洞扫描

**【答案】C**

**【解析】本题考查网络攻击中网络监听的基础知识。**

网络监听是主机的一种工作模式，在这种模式下，主机可以接收到本网段在同一条物理通道上传输的所有信息。使用网络监听工具可轻而易举地截取包括口令和账号在内的信息资料。采用数据加密的方式保护包括口令和账号在内的信息资料，使得即使网络监听获取密文后也无法解密成明文，是对付网络监听的有效手段。

软件商标权的权利人是指（10)。

(10)A.软件商标设计人　　B.软件商标制作人

C.软件商标使用人　　D.软件注册商标所有人

**【答案】D**

**【解析】本题考查知识产权方面的基础知识，涉及软件商标权主体资格的相关概念。**

在我国，商标权是指注册商标专用权，只有依法进行商标注册后，商标注册人才能取得商标权，其商标才能得到法律的保护。商标权不包括商标设计人的权利，主要注重商标所有人的权利，即注册商标所有人具有其商标的专用权。商标设计人的发表权、署名权等人身权在商标的使用中没有反映，所以不受商标法保护。商标设计人可以通过其他法律来保护属于自己的权利，如可以将商标设计图案作为美术作品通过著作权法来保护；与产品外观关系密切的商标图案还可以申请外观设计专利通过专利法加以保护。软件商标制作人、软件商标使用人均未涉及软件注册商标，所以均不能成为软件商标权的权利人。

利用（11)可以对软件的技术信息、经营信息提供保护。

(11)A.著作权　　B.专利权　　C.商业秘密权　　D.商标权

**【答案】C**

**【解析】本题考查知识产权方面的基础知识，涉及软件商业秘密权的相关概念。**

著作权从软件作品性的角度保护其表现形式，源代码（程序)、目标代码（程序)、 软

件文档是计算机软件的基本表达方式（表现形式），受著作权保护；专利权从软件功能 性的角度保护软件的思想内涵，即软件的技术构思、程序的逻辑和算法等的思想内涵，当计算机软件同硬件设备是一个整体，涉及计算机程序的发明专利，可以申请方法专利，取得专利权保护。商标权是为商业化的软件从商品、商誉的角度为软件提供保护，利用商标权可以禁止他人使用相同或者近似的商标、生产（制作）或销售假冒软件产品。商标权受保护的力度大于其他知识产权，对软件的侵权行为更容易受到行政查处。而商业秘密权是商业秘密的合法控制人采取了保密措施，依法对其经营信息和技术信息享有的专有使用权，我国《反不正当竞争法》中对商业秘密的定义为“不为公众所知悉、能为权利人带来经济利益、具有实用性并经权利人采取保密措施.的技术信息和经营信息”。软件技术秘密是指软件中适用的技术情报、数据或知识等，包括程序、设计方法、技术方案、功能规划、开发情况、测试结果及使用方法的文字资料和图表，如程序设计说明书、 流程图、用户手册等。软件经营秘密指具有软件秘密性质的经营管理方法以及与经营管理方法密切相关的信息和情报，其中包括管理方法、经营方法、产销策略、客户情报（客户名单、客户需求），以及对软件市场的分析、预测报告和未来的发展规划、招投标中的标底及标书内容等。

李某在某软件公司兼职，为完成该公司交给的工作，做出了一项涉及计算机程序的发明。李某认为该发明是自己利用业余时间完成的，可以个人名义申请专利。关于此项发明的专利申请权应归属 (12)。

(12)A.李某　　B.李某所在单位

C.李某兼职的软件公司　　D.李某和软件公司约定的一方

**【答案】C**

**【解析】本题考查知识产权方面的基础知识，涉及软件发明专利申请权归属的相关概念。**

根据《专利法》第六条第 1 款规定，执行本单位的任务所完成的发明创造是职务发明创造。职务发明创造申请专利的权利属于单位，申请被批准后，该单位为专利权人。专利法实施细则》第十一条对“执行本单位的任务所完成的发明创造”作出了解释。执行本单位的任务所完成的发明创造是指：(1)在本职工作中作出的发明创造：(2)履行本单位交付的本职工作之外的任务所作出的发明创造；(3)退职、退休或者调动工作后 一年内所作出的、与其在原单位承担的本职工作或原单位分配的任务有关的发明创造。 李某是为完成其兼职软件公司交给的工作而作出的该项发明，属于职务发明。专利申请权应归属软件公司。

《专利法》第六条第 3 款规定：“利用本单位的物质技术条件所完成的发明创造，单 位

与发明人或者设计人订有合同，对申请专利的权利和专利权的归属作出约定的，从其约定。”在事先有约定的情况下，按照约定确定权属。如果单位和发明人没有对权属问题作出约定或约定不明的，该发明创造仍视为职务发明创造，专利申请权仍然属于单位。本题未涉及合同约定，故 D 项不正确，

一幅彩色图像（RGB),分辨率为 256X512,每一种颜色用 8b 表示，则该彩色图像的数据量为（13） b。

(13)A.256X512X8　　B.256X512X3X8　　C.256X512X3/8　　D.256X512X3

**【答案】B**

**【解析】本题考查多媒体方面的基础知识，涉及彩色图像数据量计算。**

图像的分辨率越高，图像深度越深，则数字化后的图像效果越逼真，图像数据量也越大。其图像数据量可用下面的公式估算：

图像数据量=图像的总像素 X 图像深度（b)

其中图像的总像素为图像的水平方向像素乘以垂直方向像素数。例如，一幅 640X480 的 256 色图像，其图像文件大小约为 640X480X8≈300KB。

10000 张分辨率为 1024X768 的真彩（32 位）图片刻录到 DVD 光盘上，假设每张光盘可以存放 4GB 的信息，则需要（14)张光盘。

(14)A.7　　B.8　　C.70　　D.71

**【答案】B**

**【解析】本题考查多媒体方面的基础知识。涉及图片存储光盘数量的计算。**

图像数据量的计算方式如下：

图像数据量=图像的总像素 X 图像深度（b),

需用光盘数量的计算方式如下：

光盘数量=图像的总像素 X 图像深度/4GB(张）

$$\frac{10000 \times 1024 \times 768 \times 32}{8 \times 4 \times 2^{30}} = 7.3$$

某项目组拟开发一个大规模系统，且具备了相关领域及类似规模系统的开发经验。下列过程模型中，(15)最适合开发此项目。

(15)A.原型模型　　B.瀑布模型　　C.V 模型　　D.螺旋模型

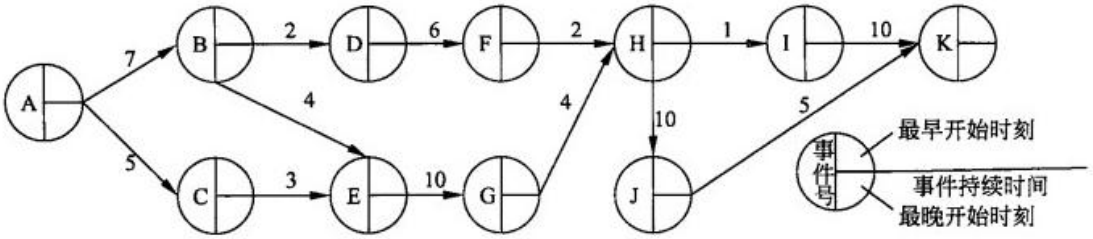
**【答案】**B

**【解析】本题考查软件开发生命周期模型的基本知识。**

常见的软件生存周期模型有瀑布模型、演化模型、螺旋模型、喷泉模型等。瀑布模型是将软件生存周期各个活动规定为依线性顺序连接的若干阶段的模型，适合于软件需求很明确的软件项目。V 模型是瀑布模型的一种演变模型，将测试和分析与设计关联进行，加强分析与设计的验证。原型模型是一种演化模型，通过快速构建可运行的原型系统，然后根据运行过程中获取的用户反馈进行改进。演化模型特别适用于对软件需求缺乏准确认识的情况。螺旋模型将瀑布模型和演化模型结合起来，加入了两种模型均忽略的风险分析。

本题中项目组具备了所开发系统的相关领域及类似规模系统的开发经验，即需求明确，瀑布模型最适合开发此项目。

使用 PERT 图进行进度安排，不能清晰地描述（16），但可以给出哪些任务完成后才能开始另一些任务。下面的 PERT 图所示工程从 A 到 K 的关键路径是（17）（图中省略了任务的开始和结束时刻）。




(16)A.每个任务从何时开始　　B.每个任务到何时结束

C.各任务之间的并行情况　　D.各任务之间的依赖关系

(17)A.ABEGHIK　　B.ABEGHJK　　C.ACEGHIK　　D.ACEGHJK

**【答案】**C　B

**【解析】本题考查软件项目管理的基础知识。**

软件项目计划的一个重要内容是安排进度，常用的方法有 Gantt 图和 PERT 图。Gantt 图用水平条状图描述，它以日历为基准描述项目任务，可以清楚地表示任务的持续时间和任务之间的并行，但是不能清晰地描述各个任务之间的依赖关系。PERT 图是一种网络模型，描述一个项目的各任务之间的关系。可以明确表达任务之间的依赖关系，即哪些任务完成后才能开始另一些任务，以及如期完成整个工程的关键路径，但是不能清晰地描述各个任务之间

的并行关系。

图中任务流 ABEGHIK 的持续时间是 36, ABEGHJK 的持续时间是 40, ACEGHIK 的持续时间是 33, ACEGHJK 的持续时间为 37。所以项目关键路径长度为 40。

敏捷开发方法 XP 是一种轻量级、高效、低风险、柔性、可预测的、科学的软件开发方法，其特性包含在 12 个最佳实践中。系统的设计要能够尽可能早交付，属于(18)最佳实践。

(18)A.隐喻　　B.重构　　C.小型发布　　D.持续集成

**【答案】**C

**【解析】本题考查软件开发过程管理的基本知识。**

敏捷开发方法 XP 是一种轻量级、高效、低风险、柔性、可预测的、科学的软件开发方法，其特性包含在 12 个最佳实践中。

（1）计划游戏：快速制定计划、随着细节的不断变化而完善；

（2）小型发布：系统的设计要能够尽可能早地交付；

（3）隐喻：找到合适的比喻传达信息；

（4）简单设计：只处理当前的需求使设计保持简单；

（5）测试先行：先写测试代码再编写程序；

（6）重构：重新审视需求和设计，重新明确地描述它们，以符合新的和现有的需求；

（7）结队编程；

（8）集体代码所有制；

（9）持续集成：可以按日甚至按小时为客户提供可运行的版本；

（10）每周工作 40 个小时；

（11）现场客户；

（12）编码标准。

在软件开发过程中进行风险分析时，(19)活动的目的是辅助项目组建立处理风险的策略，有效的策略应考虑风险避免、风险监控、风险管理及意外事件计划。

(19)A.风险识别　　B.风险预测　　C.风险评估　　D.风险控制

**【答案】**D

**【解析】本题考查软件开发风险分析的基本知识。**

风险分析实际上是 4 个不同的活动：风险识别、风险预测、风险评估和风险控制。 风

险识别是试图系统化地确定对项目计划（估算、进度、资源分配）的威胁。风险预测又称为风险估算，它从两个方面评估一个风险：风险发生的可能性或概率；以及如果风险发生时所产生的后果。风险评估根据风险及其发生的概率和产生的影响预测是否影响 参考水平值。风险控制的目的是辅助项目组建立处理风险的策略，有效的策略应考虑风 险避免、风险监控、风险管理及意外事件计划。

以下关于变量和常量的叙述中，错误的是（20）。

(20)A.变量的取值在程序运行过程中可以改变，常量则不行

B.变量具有类型属性，常量则没有

C.变量具有对应的存储单元，常量则没有

D.可以对变量赋值，不能对常量赋值

**【答案】B**

**【解析】本题考查程序设计语言的基础知识。**

变量是计算机内存单元的抽象，在程序中表示数据，具有名称、类型、值、地址、作用域、存储类别等属性，其值在运行过程中由指令进行修改。常量也用于在程序中表示数据，但常量在程序运行过程中不能修改，常量也具有类型，如整型常量、浮点型常量、字符串常量等，也称为字面量或文字。

编译程序分析源程序的阶段依次是（21）。

(21)A.词法分析、语法分析、语义分析　　B.语法分析、词法分析、语义分析

C.语义分析、语法分析、词法分析　　D.语义分析、词法分析、语法分析

**【答案】A**

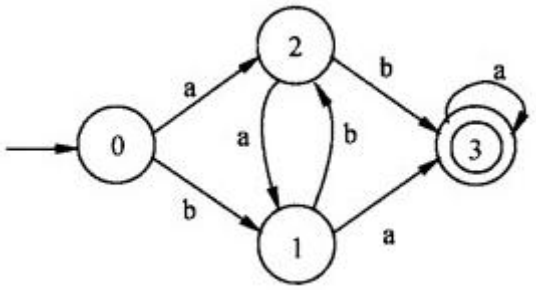
**【解析】本题考查程序语言翻译的基础知识。**

编译程序是一种将高级语言程序翻译成目标程序的系统软件，它对源程序的翻译过程分为词法分析、语法分析、语义分析、中间代码生成、代码优化和目标代码生成，以及符号表管理和出错处理。

源程序可以被看成是一个字符串。词法分析是编译过程的第一阶段，其任务是对源程序从前到后（从左到右）逐个字符地扫描，从中识别出一个个的“单词”符号。语法分析的任务是在词法分析的基础上，根据语言的语法规则将单词符号序列分解成各类语法单位，如“表达式”、“语句”、“程序”等。语义分析阶段主要检查源程序是否包含语义错误，并收集类型

信息供后面的代码生成阶段使用。只有语法和语义都正确的源程序才能被翻译成正确的目标代码。

下图所示的有限自动机中，0 是初始状态，3 是终止状态，该自动机可以识别（22）。




(22)A. abab　　B. aaaa　　C. bbbb　　D. abba

**【答案】B**

**【解析】**

本题考查程序语言翻译的基础知识。

有限自动机可识别一个字符串的含义是，从有限自动机的初态出发，存在一条到达终态的路径，其上的标记可构成该字符串。若从初态到终态不存在能构成指定字符串的路径，则称该字符串不能被该自动机识别。

对于“abab”，其识别路径为状态 0→状态 2→状态 3→状态 3，虽然到达终态，但是没有识别出最后的字符“b”。

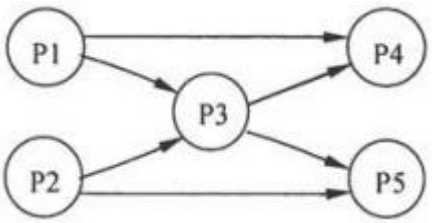
对于“bbbb”，其识别路径为状态 0→状态 1→状态 2→状态 3，虽然到达终态，但是没有识别出最后的字符“b”。

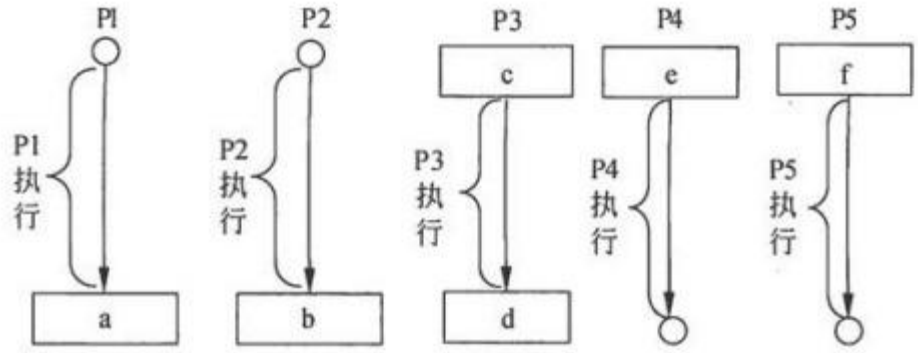
对于“abba”，其识别路径为状态 0→状态 2→状态 3，虽然到达终态，但是没有识别出“ba”。

对于“aaaa”，其识别路径为状态 0→状态 2→状态 1→状态 3→状态 3，存在从初态到终态的路径标记形成“aaaa”，所以可识别。

进程 P1、P2、P3、P4 和 P5 的前趋图如下：

若用 PV 操作控制进程 P1～P5 并发执行的过程，则需要设置 6 个信号量 S1、S2、S3. S4. S5 和 S6，且信号量 S1～S6 的初值都等于零。下图中 a 和 b 处应分别填写（23）；c 和 d 处应分别填写（24），e 和 f 处应分别填写（25）。






(23)A.P(S1) P(S2)和 P(S3)P(S4)　　B.P(S1) V(S2)和 P(S2) V(S1)

C.V(S1) V(S2)和 V(S3) V(S4)　　D.P(S1) P(S2)和 V(S1) V(S2)

(24)A.P(S1) P(S2)和 V(S3) V(S4)　　B.P(S1) P(S3)和 V(S5) V(S6)

C.V(S1) V(S2)和 P(S3) P(S4)　　D.P(S1) V(S3)和 P(S2) V(S4)

(25)A.P(S3) P(S4)和 V(S5)V(S6)　　B.V(S5) V(S6)和 P(S5) P(S6)

C.P(S2) P(S5)和 P(S4) P(S6)　　D.P(S4) V(S5)和 P(S5) V(S6)

**【答案】C B C**

**【解析】本题考查 PV 操作方面的基本知识。**

试题（23)的正确答案是 C，因为 P1 是 P3 和 P4 的前驱，当 P1 执行完成后，应通知 P3 和 P4，故应采用 V(S1) V(S2)操作分别通知 P3 和 P4;同理，P2 是 P3 和 P5 的前驱，当 P2 执行完后，应通知 P3 和 P5,故应采用 V(S3)V(S4)操作分别通知 P3 和 P5。

试题（24)的正确答案是 B,因为 P3 是 P1 和 P2 的后继，当 P3 执行前应测试 P1 和 P2 是否执行完，故应采用 P(S1) P(S3)操作分别测试 P1 和 P2 是否执行完；又因为 P3 是 P4 和 P5 的前驱，当 P3 执行完应通知 P4 和 P5,故应采用 V(S5)V(S6)操作分别通知 P4 和 P5。

试题（25)的正确答案是 C,因为 P4 是 P1 和 P3 的后继，当 P4 执行前应测试 P1 和 P3 是否执行完，故应采用 P(S2) P(S5)操作分别测试 P1 和 P3 是否执行完；又因为 P5 是 P2 和 P3 的前驱的后继，当 P5 执行前应测试 P2 和 P3 是否执行完,故应采用 P(S4) P(S6) 操作分别测试 P2 和 P3 是否执行完。

某磁盘磁头从一个磁道移至另一个磁道需要 10ms。文件在磁盘上非连续存放，逻辑上相邻数据块的平均移动距离为 10 个磁道，每块的旋转延迟时间及传输时间分别为 100ms 和 2ms，则读取一个 100 块的文件需要（26） ms 时间。

(26)A.10200　　B.11000　　C.11200　　D.20200

**【答案】D**

**【解析】本题考查操作系统中设备管理的基本知识。**

访问一个数据块的时间应为寻道时间加旋转延迟时间及传输时间。根据题意，每块的旋转延迟时间及传输时间共需 102ms，磁头从一个磁道移至另一个磁道需要 10ms，但逻辑上相邻数据块的平均距离为 10 个磁道，即读完一个数据块到下一个数据块寻道时间需要 100ms。通过上述分析，本题访问一个数据块的时间应为 202ms，而读取一个 100 块的文件共需要 20200ms，因此，本题的正确答案为 D。

某文件系统采用多级索引结构，若磁盘块的大小为 512B，每个块号需占 3B，那么根索引采用一级索引时的文件最大长度为（27）KB：采用二级索引时的文件最大长度为（28）KB。

(27)A.85　　B.170　　C.512　　D.1024

(28)A.512　　B.1024　　C.14450　　D.28900

**【答案】A　C**

**【解析】本题考查操作系统中文件管理的基本知识。**

根据题意，磁盘块的大小为 512B，每个块号需占 3B，因此一个磁盘物理块可存放 512/3=170 个块号。

根索引采用一级索引时的文件最大长度为：

170X512/1024=87040/1024=85KB

根索引采用二级索引时的文件最大长度为：

170X170X512/1024=28900X512/1024=14450KB

冗余技术通常分为 4 类，其中（29)按照工作方法可以分为静态、动态和混合冗余。

(29)A.时间冗余　　B.信息冗余　　C.结构冗余　　D.冗余附件技术

**【答案】C**

**【解析】**

冗余是指对于实现系统规定功能是多余的那部分资源，包括硬件、软件、信息和时间。通常冗余技术分为 4 类：(1)结构冗余，按其工作方法可以分为静态、动态和混合冗余；(2)信息冗余，指的是为了检测或纠正信息在运算或传输中的错误另外加的一部分信息；(3)时间冗余，是指以重复执行指令或程序来消除瞬时错误带来的影响；(4) 冗余附件技术，是指为实现上述冗余技术所需的资源和技术。

以下关于过程改进的叙述中，错误的是（30）。

(30)A.过程能力成熟度模型基于这样的理念：改进过程将改进产品，尤其是软件产品

B.软件过程改进框架包括评估、计划、改进和监控 4 个部分

C.软件过程改进不是一次性的，需要反复进行

D.在评估后要把发现的问题转化为软件过程改进计划

**【答案】**B

**【解析】**

软件成熟度模型 CMM 是对软件组织进化阶段的描述，该模型在解决软件过程存在问题方面取得了很大的成功，因此在软件界产生了巨大影响，促使软件界重视并认真对待过程改进工作。过程能力成熟度模型基于这样的理念：改进过程将改进产品，尤其是软件产品。软件组织为提高自身的过程能力，把不够成熟的过程提升到较成熟的过程涉及 4 个方面，这 4 个方面构成了软件过程改进的框架，即过程改进基础设施、过程改进线路图、软件过程评估方法和软件过程改进计划。在进行评估后需要把发现的问题转化为软件过程改进计划。而过程改进通常不可能是一次性的，需要反复进行。每一次改进要经历 4 个步骤：评估、计划、改进和监控。

软件复杂性度量的参数不包括（31）。

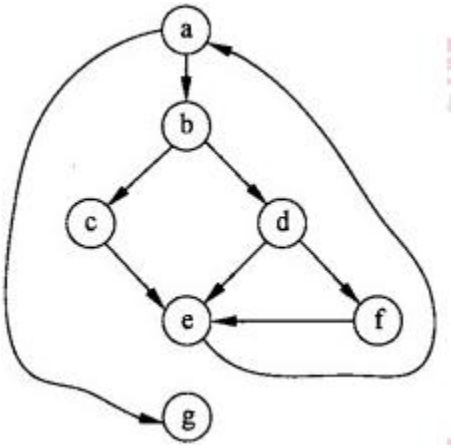
(31)A.软件的规模　　B.开发小组的规模　　C.软件的难度　　D.软件的结构

**【答案】**B

**【解析】**

软件复杂性度量是软件度量的一个重要分支。软件复杂性度量的参数有很多，主要包括：(1)规模，即指令数或者源程序行数；(2)难度，通常由程序中出现的操作数所决定的量来表示；(3)结构，通常用与程序结构有关的度量来表示；(4)智能度，即算法的难易程度。

根据 McCabe 度量法，以下程序图的复杂性度量值为（32）。




(32)A.4　　B.5　　C.6　　D.7

**【答案】A**

**【解析】**

软件复杂性度量是软件度量的一个重要分支，而其主要表现在程序的复杂性。其中，McCabe 度量法是一种基于程序控制流的复杂性度量方法，该方法认为程序的复杂性很大程度上取决于控制的复杂性。首先根据程序画出程序图，然后基于图论用图的环路数来度量程序复杂性，即 V(G) = m-n+ 2p，其中 m，n 和 p 分别表示图 G 中弧的个数、顶点的个数和强连通分量数。根据上述公式可得，上图的复杂性为 9-7+2=4。

软件系统的可维护性评价指标不包括（33）。

(33)A.可理解性　　B.可测试性　　C.可扩展性　　D.可修改性

**【答案】C**

**【解析】**

软件的可维护性是指维护人员理解、改正、改动和改进这个软件的难易程度，是软件开发阶段各个时期的关键目标。软件系统的可维护性评价指标包括可理解性、可测试性、可修改性、可靠性、可移植性、可使用性和效率。

以下关于软件系统文档的叙述中，错误的是（34）。

(34)A.软件系统文档既包括有一定格式要求的规范文档，又包括系统建设过程中的各种来往文件、会议纪要、会计单据等资料形成的不规范文档

B.软件系统文档可以提高软件开发的可见度

C.软件系统文档不能提高软件开发效率

D.软件系统文档便于用户理解软件的功能、性能等各项指标

**【答案】**C

**【解析】**

软件系统文档是系统建设过程的“痕迹”，是系统维护人员的指南，是开发人员与用户交流的工具。软件系统文档不仅包括应用软件开发过程中产生的文档，还包括硬件采购和网络设计中形成的文档；不仅包括有一定格式要求的规范文档，还包括系统建设过程中的各种来往文件、会议纪要、会计单据等资料形成的不规范文档。软件系统文档可以提高软件开发的可见度，提高软件开发效率，且便于用户理解软件的功能、性能等各项指标。

以下关于软件测试的叙述中，正确的是（35）。

(35)A.软件测试不仅能表明软件中存在错误，也能说明软件中不存在错误

B.软件测试活动应从编码阶段开始

C.一个成功的测试能发现至今未发现的错误

D.在一个被测程序段中，若已发现的错误越多，则残存的错误数越少

**【答案】**C

**【解析】**

软件测试是软件开发过程中一个独立而且非常重要的阶段，它是为了发现错误而执行程序的过程。因此一个成功的测试应该能发现至今未发现的错误。而且需要特别指出的是软件测试不能表明软件中不存在错误，它只能说明软件中存在错误。另外，由于问题的复杂性、软件本身的复杂性和抽象性、软件开发各个阶段工作的多样性、参加开发各种人员之间的配合关系等因素，使得开发的每个环节都可能产生错误，因此软件测试应该贯穿到软件开发的各个阶段中，且需要尽早地和不断地进行。经验表明，测试中存在一种集群现象，即在被测程序段中，若发现的错误数目越多，则残存的错误数目也较多。

不属于黑盒测试技术的是（36）。

(36)A.错误猜测　　B.逻辑覆盖　　C.边界值分析　　D.等价类划分

**【答案】**B

**【解析】**

黑盒测试也称为功能测试，在完全不考虑软件的内部结构和特性的情况下来测试软件的外部特性。常用的黑盒测试技术包括等价类划分、边界值分析、错误猜测和因果图的报告。

白盒测试也称为结构测试，根据程序的内部结构和逻辑来设计测试用例，对程序的执行路径和过程进行测试，检查是否满足设计的需要。常用的白盒测试技术包括逻辑覆盖和基本路径测试。

开-闭原则（Open-Closed Principle，OCP)是面向对象的可复用设计的基石。开-闭原则是指一个软件实体应当对（37)开放，对(38)关闭:里氏代换原则（Liskov Substitution Principle, LSP)是指任何（39)可以出现的地方，(40) 一定可以出现。 依赖倒转原则（Dependence Inversion Principle, DIP)就是要依赖于（41),而不依赖于（42),或者说要针对接口编程，不要针对实现编程。

(37)A.修改　　B.扩展　　C.分析　　D.设计

(38)A.修改　　B.扩展　　C.分析　　D.设计

(39)A.变量　　B.常量　　C.基类对象　　D.子类对象

(40)A.变量　　B.常量　　C.基类对象　　D.子类对象

(41)A.程序设计语言　B.建模语言　　C.实现　　D.抽象

(42)A.程序设计语言　B.建模语言　　C.实现　　D.抽象

**【答案】**B　A　C　D　D　C

**【解析】本题考查面向对象设计的原则。**

开-闭原则(Open-Closed Principle)是面向对象的可复用设计(Object Oriented Design, OOD)的基石。开-闭原则是指一个软件实体应当对扩展开放，对修改关闭，即在设计 一个模块的时候，应当使这个模块可以在不被修改的前提下被扩展。满足开-闭原则的系统可以通过扩展已有的软件系统，提供新的能力和行为，以满足对软件的新需求，使软件系统有一定的适应性和灵活性;因为已有的软件模块,特别是最重要的抽象层模块不能再修改，这就使变化中的软件系统有一定的稳定性和延续性；满足开-闭原则的系统具备更好的可复用性与可维护性。

在面向对象编程中，通过抽象类及接口，规定了具体类的特征作为抽象层，相对稳定，从而满足“对修改关闭”的要求；而从抽象类导出的具体类可以改变系统的行为， 从而满足对扩展开放。

里氏代换原则（Liskov Substitution Principle，LSP)是指一个软件实体如果使用的是一个基类的话，那么一定适用于其子类，而且软件系统觉察不出基类对象和子类对象的区别，也就是说，在软件系统中把基类都替换成它的子类，程序的行为没有变化。但需要注意

的是，里氏代换原则中仅仅指出了用子类的对象去代替基类的对象，而反过来的代换则是不成立的。例如，如果一个软件模块中使用的是一个子类对象，那么使用父类 对象去代换子类对象则可能产生错误。用一句简单的话概括：任何基类对象可以出现的地方，子类对象一定可以代替基类对象。

依赖倒转原则（Dependence Inversion Principle, DIP)就是要依赖于抽象，而不依赖于实现，或者说要针对接口编程，不要针对实现编程。系统中进行设计和实现的时候应 当使用接口和抽象类进行变量类型声明、参数类型声明、方法返回类型说明，以及数据 类型的转换等，而不要用具体类进行上述操作。要保证做到这一点，一个具体类应当只 实现接口和抽象类中声明过的方法，而不要给出多余的方法。

传统的过程性系统的设计办法倾向于使高层次的模块依赖于低层次的模块，抽象层次依赖于具体层次。依赖倒转原则就是把这个不良的依赖关系倒转过来。面向对象设计 的重要原则是创建抽象层次，并且从该抽象层次导出具体层次，具体层次给出不同的实现。继承关系就是一种从抽象化到具体化的导出。抽象层包含的应该是应用系统的业务 逻辑和宏观的、对整个系统来说重要的战略性决定，而具体层次含有的是一些次要的与 实现有关的算法和逻辑，以及战术性的决定，带有一定的偶然性选择。从复用的角度来说，高层抽象的模块是应当复用的，而且是复用的重点，因为它含有一个应用系统最重 要的宏观业务逻辑，是较为稳定的部分。而在传统的过程性设计中，复用则侧重于具体层次模块的复用。

使用依赖倒转原则时建议不依赖于具体类，即程序中所有的依赖关系都应该终止于抽象类或者接口。尽量做到：任何变量都不应该持有一个指向具体类的指针或者引用； 任何类都不应该从具体类派生；任何方法都不应该覆写它的任何基类中的已经实现的方法。

(43)是一种很强的“拥有”关系，“部分”和“整体”的生命周期通常一样。整体对象完全支配其组成部分；包括它们的创建和销毁等；(44)同样表示“拥有” 关系，但有时候“部分”对象可以在不同的“整体”对象之间共享，并且“部分”对象的生命周期也可以与“整体”对象不同，甚至“部分”对象可以脱离“整体”对象而单独存在。上述两种关系都是 (45)关系的特殊种类。

(43)A.聚合　　B.组合　　C.继承　　D.关联

(44)A.聚合　　B.组合　　C.继承　　D.关联

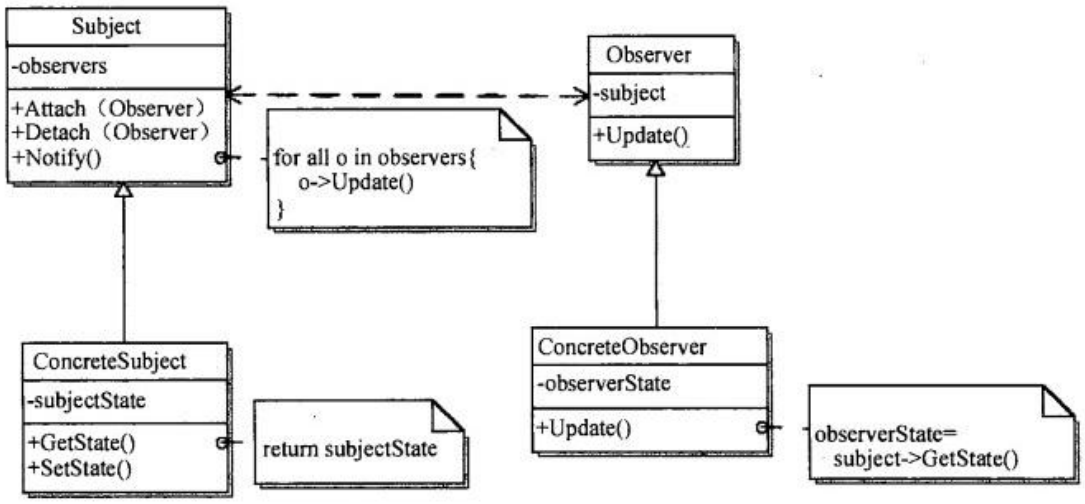
(45)A.聚合　　B.组合　　C.继承　　D.关联

**【答案】**B　A　D

**【解析】本题考查组合和聚合的基本概念。**

组合（Composition)和聚合（Aggregation)都是关联（Association)的特殊种类。 组合是一种很强的“拥有”关系，部分和整体的生命周期通常一样。组合成的新对象完全支配其组成部分，包括它们的创建和湮灭等。一个组合关系的成分对象是不能被另一个组合构成的对象共享的。聚合同样表示“拥有”关系，但其程度不如组合强，有时候“部分”对象可以在不同的“整体”对象之间共享，并且“部分”对象的生命周期也可以与“整体”对象不同，甚至“部分”对象可以脱离“整体”对象而单独存在。一般而言，组合是值的合成（Aggregation by Value)，而聚合是引用的合成（Aggregation by Reference )。

下面的 UML 类图描绘的是 (46) 设计模式。关于该设计模式的叙述中，错误的是 (47)。




(46)A.桥接　　B.策略　　C.抽象工厂　　D.观察者

(47)A.该设计模式中的 Observer 需要维护至少一个 Subject 对象

B.该设计模式中的 ConcreteObserver 可以绕过 Subject 及其子类的封装

C.该设计模式中一个 Subject 对象需要维护多个 Observer 对象

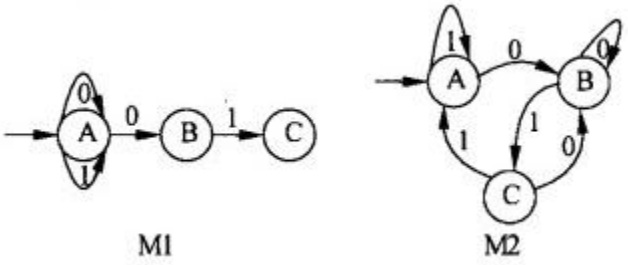
D.该设计模式中 Subject 需要通知 Observer 对象其自身的状态变化

**【答案】D　B**

**【解析】本题考查面向对象设计中的设计模式。**

题中的类图是观察者设计模式，在该设计模式中的 Subject 和 Observer 分别表示抽象的被观察者和观察者。通常，一个观察者（Observer)观察一个被观察者（Subject)，而一个被观察者可以被多个观察者关注。当 Subject 的状态发生变化时，Subject 将通知所有的 Observer，告知状态已经发生了变化，而 Observer 收到通知后，将查询 Subject 的状态。

下图所示为两个有限自动机 M1 和 M2（A 是初态、C 是终态），(48)。




(48)A.M1 和 M2 都是确定的有限自动机

B.M1 和 M2 都是不确定的有限自动机

C.M1 是确定的有限自动机，M2 是不确定的有限自动机

D.M1 是不确定的有限自动机，M2 是确定的有限自动机

**【答案】**D

**【解析】**

本题考查程序语言翻译的基础知识。

有限自动机是一种识别装置的抽象概念，它能准确地识别正规集。有限自动机分为两类：确定的有限自动机和不确定的有限自动机。它们都可以用状态转换图和状态转换矩阵表示。

一个确定的有限自动机是个五元组：（$S$，$\Sigma$，$f$，$s0$，$Z$），其中：$S$ 是一个有限集合，它的每个元素称为一个状态；$\Sigma$ 是一个有穷字母表，它的每个元素称为一个输入字符；$f$ 是从 $S\times\Sigma\rightarrow S$ 上的单值部分映像，$f(A, a)=Q$ 表示当前状态为 $A$、输入为 $a$ 时，将转换到下一状态 $Q$。我们称 $Q$ 为 $A$ 的一个后继状态；$s0\in S$，是唯一的一个开始状态；$Z$ 是非空的终止状态集合，$Z\subseteq S$。

一个不确定的有限自动机也是一个五元组，它与确定有限自动机的区别是：$f$ 是从 $S\times\Sigma\rightarrow 2S$ 上的映像。对于 $S$ 中的一个给定状态及输入符号，返回一个状态的集合。即当前状态的后继状态不一定是唯一确定的；有向弧上的标记可以是 $\varepsilon$。

以下关于可视化程序设计的叙述中，错误的是（49）。

(49)A.可视化程序设计使开发应用程序无需编写程序代码

B.可视化程序设计基于面向对象的思想，引入了控件和事件驱动

C.在可视化程序设计中，构造应用程序界面就像搭积木

D.在可视化程序设计中，采用解释方式可随时查看程序的运行效果

**【答案】**A

**【解析】本题考查程序设计的基础知识。**

可视化程序设计是以“所见即所得”的编程思想为原则，力图实现编程工作的可视化，即随时可以看到结果，程序与结果的调整同步。

与传统的编程方式相比，“可视化程序设计”仅通过直观的操作方式即可完成界面的设计工作。

可视化程序设计语言的特点主要表现在两个方面：一是基于面向对象的思想，引入了控件的概念和事件驱动；二是程序开发过程一般遵循以下步骤，即先进行界面的绘制工作，再基于事件编写程序代码，以响应鼠标、键盘的各种动作。

可视化程序设计最大的优点是设计人员可以不用编写或只需编写很少的程序代码，就能完成应用程序的设计，这样就能极大地提高设计人员的工作效率。

以下关于汇编语言的叙述中，错误的是（50）。

(50)A.汇编语言源程序中的指令语句将被翻译成机器代码

B.汇编程序先将源程序中的伪指令翻译成机器代码，然后再翻译指令语句

C.汇编程序以汇编语言源程序为输入，以机器语言表示的目标程序为输出

D.汇编语言的指令语句必须具有操作码字段，可以没有操作数字段

**【答案】**B

**【解析】本题考查程序设计语言的基础知识。**

汇编语言源程序中的每一条指令语句在源程序汇编时都要产生可供计算机执行的指令代码（即目标代码）。

伪指令语句用于指示汇编程序如何汇编源程序，常用于为汇编程序提供以下信息：该源程序如何分段，有哪些逻辑段在程序段中，哪些是当前段，它们分别由哪个段寄存器指向；定义了哪些数据，存储单元是如何分配的等。伪指令语句除定义的具体数据要生成目标代码外，其他均没有对应的目标代码。伪指令语句的这些命令功能是由汇编程 序在汇编源程序时，通过执行一段程序来完成的，而不是在运行目标程序时实现的。

目前主要有两种不同标准的汇编语言指令格式：Windows 下的汇编语言基本上都遵循 Intel 风格的语法,如 MASM、NASM，而 Unix/Linux 下的汇编语言基本上都遵循 AT&T 风格的语法。

汇编语言语句的通用格式如下：

[名称[:]] 指令码 [第一操作数][,第二操作数] ;注释

汇编语言指令码的操作数的个数可以是0、1、2个；当操作数的个数为2的时候，语句还有两种不同的格式。

Windows下Intel风格的汇编语言语句格式为：

[名称[:]] 指令码 目的操作数DST，源操作数SRC ;注释

Unix/Linux下AT&T风格的汇编语言语句格式为：

[名称[:]] 指令码 源操作数SRC，目的操作数DST ;注释

汇编语言语句格式中的“名称”并不是所有语句都必需的。如果语句中带有“名称”，则大多数情况下“名称”都表示的是内存中某一存储单元的地址，也就是其后面各项在内存中存放的第一个存储单元的地址。

在某企业的营销管理系统设计阶段，属性“员工”在考勤管理子系统中被称为“员工”，而在档案管理子系统中被称为“职工”，这类冲突称为(51)冲突。

(51)A.语义　　B.结构　　C.属性　　D.命名

**【答案】D**

**【解析】本题考查数据库概念结构设计中的基础知识。**

根据局部应用设计好各局部E-R图之后，就可以对各个局部E-R图进行合并。合并的目的在于解决分E-R图中相互间存在的冲突，消除分E-R图之间存在的信息冗余，使之成为能够被全系统所有用户共同理解和接受的统一的、精炼的全局概念模型。分E-R图之间的冲突主要分为结构冲突、属性冲突和命名冲突三类。

选项A显然是不正确的。

选项B不正确。因为结构冲突是指同一实体在不同的分E-R图中有不同的属性，同一对象在某一分E-R图中被抽象为实体而在另一分E-R图中又被抽象为属性，需要统一。

选项C不正确，因为属性冲突是指同一属性可能会存在于不同的分E-R图，由于设计人员不同或是出发点不同，对属性的类型、取值范围、数据单位等可能会不一致，这些属性对应的数据将来只能以一种形式在计算机中存储，这就需要在设计阶段进行统一。

选项D正确，因为命名冲突是指相同意义的属性在不同的分E-R图上有着不同的命名，或是名称相同的属性在不同的分E-R图中代表着不同的意义，这些也要进行统一。

设有学生实体Students（学号，姓名，性别，年龄，家庭住址，家庭成员，关系，联系电话），其中“家庭住址”记录了邮编、省、市、街道信息；“家庭成员，关系，联系电话”

分别记录了学生亲属的姓名、与学生的关系以及联系电话。

学生实体 Students 中的“家庭住址”是一个 (52)属性:为使数据库模式设计更合理，对于关系模式 Students (53),

(52)A.简单　　B.多值　　C.复合　　D.派生

(53)A.可以不作任何处理，因为该关系模式达到了 3NF

B.只允许记录一个亲属的姓名、与学生的关系以及联系电话的信息

C.需要对关系模式 Students 增加若干组家庭成员、关系及联系电话字段

D.应该将家庭成员、关系及联系电话加上学生号，设计成为一个独立的实体

**【答案】C　D**

**【解析】本题考查关系运算和 E-R 图的基本概念。**

试题（52)的正确答案为 C。简单属性是原子的、不可再分的。复合属性可以细分为更小的部分（即划分为别的属性)。有时用户希望访问整个属性，有时希望访问属性 的某个成分，那么在模式设计时可采用复合属性。本题学生实体集 Students 的“家庭住址”可以进一步分为邮编、省、市、街道。

在大多数情况下，定义的属性对于一个特定的实体都只有单独的一个值。例如，对于一个特定的学生，只对应一个学生号、学生姓名，这样的属性叫做单值属性。但是， 在某些特定情况下，一个属性可能对应一组值。例如，学生可能有 0 个、1 个或多个亲属，那么学生的亲属的姓名可能有多个。这样的属性称为多值属性。为了将数据库模式 设计得更合理，试题（53)应该将家庭成员、关系及联系电话加上学生号设计成为一个独立的实体。

设有关系模式 R（课程，教师，学生，成绩，时间，教室)，其中函数依赖集 F 如下：

关系模式的一个主键是 (54)，R 规范化程度最高达到 (55)。若将关系模式 R 分解为 3 个关系模式及 R1 (课程，教师)、R2 (学生，课程，成绩)、R3 (学生， 时间，教室，课程),. 其中 R2 的规范化程度最高达到 (56)。

F={课程→→教师，（学生，课程)→成绩，(时间，教室)→课程，
(时间，教师)→教室，(时间，学生)→教室}

(54)A.(学生，课程）　B.(时间，教室）　C.(时间，教师）　D.(时间，学生）

(55)A.INF　　B.2NF　　C.3NF　　D.BCNF

(56)A.2NF　　B.3NF　　C.BCNF　　D.4NF

**【答案】D　B　C**

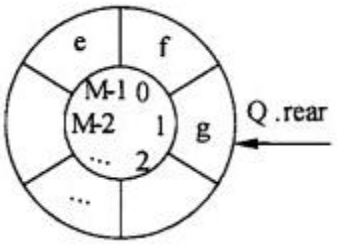
**【解析】本题主要考查关系模式规范化方面的相关知识。**

试题（54)的正确答案为D。因为根据函数依赖集F可知（时间，学生）可以决定关系R中的全部属性，故关系模式R的一个主键是（时间，学生)。

试题（55)的正确答案为B。因为根据函数依赖集F可知，R中的每个非主属性完全函数依赖于（时间，学生)，所以及是2NF。

试题（56)的正确答案为C。因为R2(学生，课程，成绩）的主键为（学生，课程)，而R2的每个属性都不传递依赖于R2的任何键，所以R2是BCNF。

设循环队列Q的定义中有rear和len两个域变量,其中rear表示队尾元素的指针,len表示队列的长度，如下图所示（队列长度为3,队头元素为e)。设队列的存储空间容量为M，则队头元素的指针为（57)。




(57)A.(Q.rear+Q.len-1)　　B.(Q.rear+Q.len-1+M)%M

C.(Q.rear-Q.len+1)　　D.(Q.rear-Q.len+1+M)%M

**【答案】D**

**【解析】本题考查数据结构的基础知识。**

从题目中的图可以推导出，队头元素的指针为(Q.rear-Q.len+1+M)%M。

下面关于哈夫曼树的叙述中，正确的是（58)。

(58)A.哈夫曼树一定是完全二叉树

B.哈夫曼树一定是平衡二叉树

C.哈夫曼树中权值最小的两个节点互为兄弟节点

D.哈夫曼树中左孩子节点小于父节点、右孩子节点大于父节点

**【答案】C**

**【解析】本题考查数据结构的基础知识。.**

树的带权路径长度为树中所有叶子节点的带权路径长度之和。哈夫曼树是指权值为 w1，

w2,…，wn 的 n 个叶子节点的二叉树中带权路径长度最小的二叉树。

构造最优二叉树的哈夫曼算法如下：

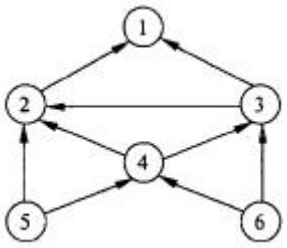
(1)根据给定的 n 个权值{w1, w2,…，wn},构成 n 棵二叉树的集合 F= {T1， T2,…，Tn}，其中每棵二叉树 Ti 中只有一个带权为 wi 的根节点，其左右子树均空。

在 F 中选取两棵权值最小的二叉树作为左、右子树构造一棵新的二叉树，置新 构造二叉树的根节点的权值为其左、右子树根节点的权值之和。

从 F 中删除这两棵树，同时将新得到的二叉树加入到 F 中。

重复（2)、(3)，直到 F 中只含一棵树时为止。这棵树便是最优二叉树（哈夫曼树)。 从以上叙述可知，哈夫曼树中权值最小的两个节点互为兄弟节点。

(59)是右图的合法拓扑序列。




(59)A.654321　　B.123456　　C.563421　　D.564213

**【答案】A**

**【解析】本题考查数据结构的基础知识。**

拓扑排序是将 AOV 网中所有顶点排成一个线性序列的过程，并且该序列满足：若在 AOV 网中从顶点 vi 到 vj 有一条路径，则在该线性序列中，顶点 vi 必然在顶点 vj 之前。

对 AOV 网进行拓扑排序的方法如下：

(1)在 AOV 网中选择一个入度为零（没有前驱）的顶点且输出它；

(2)从网中删除该顶点及与该顶点有关的所有边；

(3)重复上述两步，直至网中不存在入度为零的顶点为止。

本题中只有序列“6 5 43 2 1”可由上述过程导出。

对有向图进行拓扑排序的结果会有两种情况：一种是所有顶点已输出，此时整个拓扑排序完成，说明网中不存在回路；另一种是尚有未输出的顶点，剩余的顶点均有前驱顶点，表明网中存在回路。

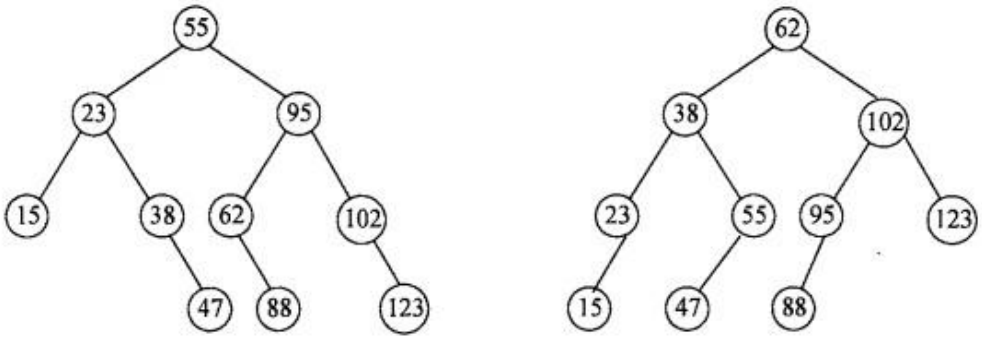
某一维数组中依次存放了数据元素 15，23，38，47，55，62，88，95，102，123，采用折半（二分）法查找元素 95 时，依次与（60)进行了比较。

(60)A.62，88，95　　B.62，95　　C.55，88，95　　D.55，95

**【答案】D**

**【解析】本题考查数据结构的基础知识。**

对序列 15，23，38，47，55，62，88，95，102，123 进行二分查找的过程可用以下二叉树之一描述，其中，左图描述的是除 2 以后向下取整时的判定过程，右图则对应除 2 以后向上取整时的判定过程




从上图可知，二分法查找 95 时，参与比较的元素依次为 55、95，或者 62、102、95。

已知一棵度为 3 的树（一个节点的度是指其子树的数目，树的度是指该树中所有节点的度的最大值）中有 5 个度为 1 的节点，4 个度为 2 的节点，2 个度为 3 的节点，那么，该树中的叶子节点数目为（61)。

(61)A.10　　B.9　　C.8　　D.7

**【答案】B**

**【解析】本题考查数据结构的基础知识。**

设树中的节点总数为 n、分支数目为 m，那么 n=5+4+2+叶子节点数，m=5X1+4X2+2X3。

在树中，节点总数等于分支数目加上 1，即 n=m+1。

因此，叶子节点数=5X1+4X2+2X3+1-5-4-2=9

某算法的时间复杂度可用递归式 $T(n)=\begin{cases} O(1), & n=1 \\ 2T(n/2)+n\lg n, & n>1 \end{cases}$ 表示，若用 $\Theta$ 表示该算法的渐进时间复杂度的紧致界，则正确的是（62）。

(62)A. $\Theta(n\lg^2 n)$ B. $\Theta(n\lg n)$ C. $\Theta(n^2)$ D. $\Theta(n^3)$

【答案】A

【解析】

该题可以用主方法来求解，对该递归式，$a$=1，$b$=2，$n^{\log_b a}=n^{\log_2 2}=n$ 而 $f(n)$=$n$lg$n$，属于第二种情况，因此，其时间复杂度为 $\Theta(n\lg^2 n)$。该题还可以用递归树求解。

用动态规划策略求解矩阵连乘问题 $M_1\times M_2\times M_3\times M_4$，其中 $M_1(20*5)$、$M_2(5\times35)$、$M_3(35\times4)$和 $M_4(4\times25)$，则最优的计算次序为__（63）__。

(63)A. $((M_1\times M_2)\times M_3)\times M_4$ B. $(M_1\times M_2)\times(M_3\times M_4)$

C. $(M_1\times(M_2\times M_3))\times M_4$ D. $M_1\times(M_2\times(M_3\times M_4))$

【答案】C

【解析】

矩阵连乘问题指的是确定 $n$ 个矩阵相乘的次序，即给这 $n$ 个相乘的矩阵加括号，使得按照该顺序进行计算时所需要的标量乘法的次数最少。用 $m$[1，$n$]来表示 $n$ 个矩阵 $M_1\times M_2\times\ldots\times M_n$ 相乘所需要的最小标量乘法的次数，则可以用递归式

$$m[i,j]=\begin{cases}0 & \text{若}i=j\\ \min\limits_{i\le k<j}\{m[i,k]+m[k+1,j]+p_{i-1}p_kp_j\} & \text{若}i\neq j\end{cases}$$

其中 $p_{i-1}p_kp_j$ 分别表示第 $i$ 个矩阵的行数、第 $k$ 个矩阵的列数和第 $j$ 个矩阵的列数来表示该问题的最优子结构。

根据上述递归式，自底向上求解得到 $m$[1,4] = 3100，对应的最优的加括号方式为 $(M_1\times(M_2\times M_3))\times M_4$。

下面 C 程序段中 count++语句执行的次数为__（64）__。

```
for(int i = 1;i <= 11;i *= 2)
for(int j = 1; j <= i;j++)
    count++;
```

(64)A. 15 B. 16 C. 31 D. 32

【答案】A

【解析】该题考查算法分析的基础知识，以及对算法中循环结构的掌握。

分析算法时间复杂度并不是确定算法运行的具体时间的长短，而是执行某个（某些）操作的次数。该题要求计算 count++语句执行的次数，根据上述 C 程序段可知，i=1 时执行 1 次，i=2 时执行 2 次，i=4 时执行 4 次，i=8 时执行 8 次，总共执行次数为 1+2+4+8=15 次。

(65)不能保证求得 0-1 背包问题的最优解。

(65)A.分支限界法　　B.贪心算法　　C.回溯法　　D.动态规划策略

【答案】B

【解析】

0-1 背包问题是一个经典的最优化问题，问题描述为：有 $n$ 个物品，第 $i$ 个物品价值为 $v_i$，重量为 $w_i$，其中 $v_i$ 和 $w_i$ 均为非负数，背包的容量为 $W$，$W$ 为非负数。现需要考虑如何选择装入背包的物品，使装入背包的物品总价值最大。该问题可以形式化地描述如下：

目标函数为 $\max\sum_{i=1}^{n}v_ix_i$，约束条件为 $\sum_{i=1}^{n}w_ix_i\leqslant W$，$x_i\in\{0,1\}$

0-1 背包问题具有最优子结构。考虑重量至多为 $W$ 的背包中的物品的最大价值，若把第 $j$ 个物品从背包中去掉，则剩下的背包中的物品的价值应该是 $n$–1 项物品（除了第 $j$ 项）背包容量为 $W$–$w_j$ 的子问题的最大价值。该问题可以通过动态规划算法来求得最优解。同时可以通过回溯法和分支限界法进行系统的搜索得到最优解。但是由于问题不具有贪心选择性质，即通过局部最优选择不能得到全局最优解，因此用贪心算法求解不能保证得到最优解。

公钥体系中，私钥用于（66），公钥用于（67）。

(66)A.解密和签名　　B.加密和签名　　C.解密和认证　　D.加密和认证

(67)A.解密和签名　　`B.加密和签名　　C.解密和认证　　D.加密和认证

【答案】A　D

【解析】本题考查公钥体系的概念和应用

1976 年斯坦福大学的 Diffie 和 Heilman 提出了使用不同的密钥进行加密和解密的公钥加密算法。设 P 为明文，C 为密文，E 为公钥控制的加密算法，D 为私钥控制的解密算法，这些参数满足下列 3 个条件：

（1）D (E (P)) =P

（2）不能由 E 导出 D

（3）选择明文攻击（选择任意明文-密文对以确定未知的密钥）不能破解 E

加密时计算 C=E (P)，解密时计算 P=D (C)。加密和解密是互逆的。用公钥加密、私钥解密，可实现保密通信；用私钥加密、公钥解密，可实现数字签名。

HTTP 协议中，用于读取一个网页的操作方法为（68）。

(68)A.READ　　B.GET　　C.HEAD　　D.POST

【答案】B

【解析】本题考查考生对 HTTP 命令的掌握程度。

GET 是 HTTP 协议提供的少数操作方法中的一种，其含义是读一个网页。HEAD 命令用于读取网页头信息。POST 命令用于把消息加到指定的网页上。没有 READ 这一命令。

帧中继作为一种远程接入方式有许多优点，下面的选项中错误的是（69）。

(69)A.帧中继比 X.25 的通信开销少，传输速度更快

B.帧中继与 DDN 相比，能以更灵活的方式支持突发式通信

C.帧中继比异步传输模式能提供更高的数据速率

D.租用帧中继虚电路比租用 DDN 专线的费用低

【答案】C

【解析】本题考查数据交换网的基础知识。

帧中继（Frame Relay，FR)是为克服 X.25 交换网的缺陷、提高传输性能而发展起来的高速分组交换技术。帧中继网络不进行差错和流量控制，并且通过流水方式进行交换，所以比 X.25 网络的通信开销更少，传输速度更快。

帧中继提供面向连接的虚电路服务，因而比 DDN 专线更能提高通信线路利用率， 用户负担的通信费用也更低廉。在帧中继网中，用户的信息速率可以在一定的范围内变化，从而既可以适应流式业务，又可以适应突发式业务，这使得帧中继成为远程传输的理想形式。

HTML 文档中<table>标记的 align 属性用于定义（70）。

(70)A.对齐方式　　B.背景颜色　　C.边线粗细　　D.单元格边距

【答案】A

【解析】

本题考查 HTML 文档中<table>记常用的属性定义。Align 用于定义文本的对齐方式。

People are indulging in an illusion whenever they find themselves explaining at a cocktail（鸡尾酒）party, say, that they are “in computers,”or “in telecommunications,”or “in electronic firnds transfer”. The implication is that they are part of the high-tech world. Just between us, they usually aren’t. The researchers who made fimdamental breakthroughs in those areas are in a high-tech

business. The rest of us are (71) of their work. We use computers and other new technology components to develop our products or to organize our affairs. Because we go about this work in teams and projects and other tightly knit working groups (紧密联系在一起的工作小组), we are mostly in the human communication business. Our successes stem from good human interactions by all participants in the effort, and our failures stem from poor human interactions.

The main reason we tend to focus on the (72) rather than the human side of the work is not because it's more (73) , but because itfs easier to do. Getting the new disk drive installed is positively trivial compared to figuring out why Horace is in a blue funk (恐惧)or why Susan is dissatisfied with the company after only a few months. Human interactions are complicated and never very crisp (干脆的，干净利落的) and clean in their effects, but they matter more than any other aspect of the work.

If you find yourself concentrating on the (74) rather than the (75) , you're like the vaudeville character (杂耍人物) who loses his keys on a dark street and looks for them on the adjacent street because,as he explains, "The light is better there!".

| | | | |
|---|---|---|---|
| (71)A.creators | B.innovators | C.appliers | D.inventors |
| (72)A.technical | B.classical | C.social | D.societal |
| (73)A.trivial | B.crucial | C.minor | D.insignificant |
| (74)A.technology | B.sociology | C.physiology | D.astronomy |
| (75)A.technology | B.sociology | C.physiology | D.astronomy |

**【答案】**C A B A B

**【解析】**

无论何时当人们发现自己在鸡尾酒会上向别人解释，比方说他们"在计算机领域" 或"在远程通信领域"或"在电子基金转账领域"工作时，他们都会沉浸在高科技的幻觉中，这就暗示他们是高科技王国里的一分子。在我们看来，他们一般都不是。在这些领域中，只有那些有根本性突破的研究人员是在做高科技业务,我们所有其他局外人只是他们工作成果的应用者。我们用计算机和其他新技术组件来开发产品或者组织我们的事务，因为是以团队和项目以及其他紧密结合的工作小组的形式来从事这项工作的,主要在从事人类交流的业务。

我们的成功源自良好的、与所有此项工作的参与者之间的人际交往，同样我们的失败原因也是由于糟糕的人际交往。

我们倾向于集中精力做技术方面，而不是人际关系方面工作的主要原因，不是因为它更重要，而是因为它更容易做。与弄清楚贺瑞斯为什么恐惧不安，或者苏珊为什么在公司只工作了几个月就对公司不满意之类的事情相比，安装一个新的磁盘驱动器肯定是微不足道的。人际交往是很复杂的，并且就效果而言从来都不会是很明晰和清楚的，但是它们比工作的任何其他方面更重要。

如果你发现自己关注的是技术而不是社会方面的问题，你就相当于在一条黑暗的街上丢失了钥匙，却到邻近的另一条街上去寻找。因为“这条街上的灯比那条街上的灯要亮一些”。

**试题一**

某时装邮购提供商拟开发订单处理系统，用于处理客户通过电话、传真、邮件或 Web 站点所下订单。其主要功能如下：

（1）增加客户记录。将新客户信息添加到客户文件，并分配一个客户号以备后续使用。

（2）查询商品信息。接收客户提交的商品信息请求，从商品文件中查询商品的价格和可订购数量等商品信息，返回给客户。

（3）增加订单记录。根据客户的订购请求及该客户记录的相关信息，产生订单并添加到订单文件中。

（4）产生配货单。根据订单记录产生配货单，并将配货单发送给仓库进行备货；备好货后，发送备货就绪通知。如果现货不足，则需向供应商订货。

（5）准备发货单。从订单文件中获取订单记录，从客户文件中获取客户记录，并产生发货单。

（6）发货。当收到仓库发送的备货就绪通知后，根据发货单给客户发货；产生装运单并发送给客户。

（7）创建客户账单。根据订单文件中的订单记录和客户文件中的客户记录，产生并发送客户账单，同时更新商品文件中的商品数量和订单文件中的订单状态。

（8）产生应收账户。根据客户记录和订单文件中的订单信息，产生并发送给财务部门应收账户报表。

现采用结构化方法对订单处理系统进行分析与设计，获得如图 1-1 所示的顶层数据流图和图 1-2 所示的 0 层数据流图。

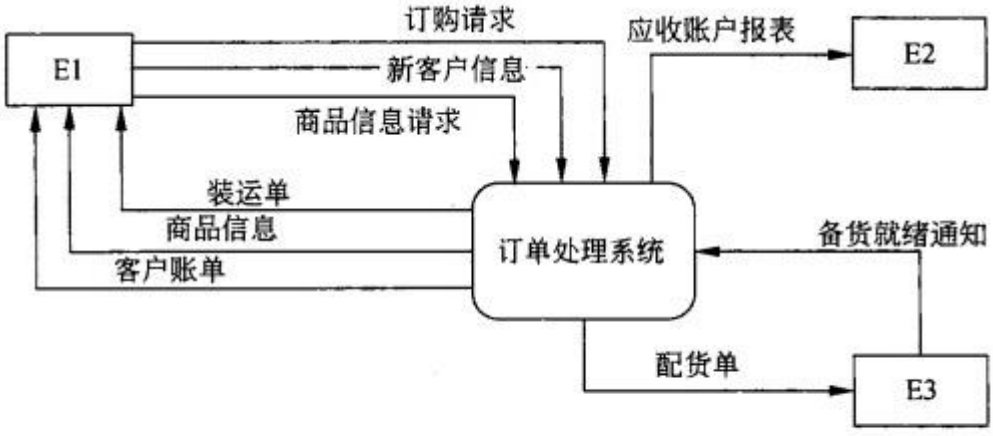



图 1-1　顶层数据流图

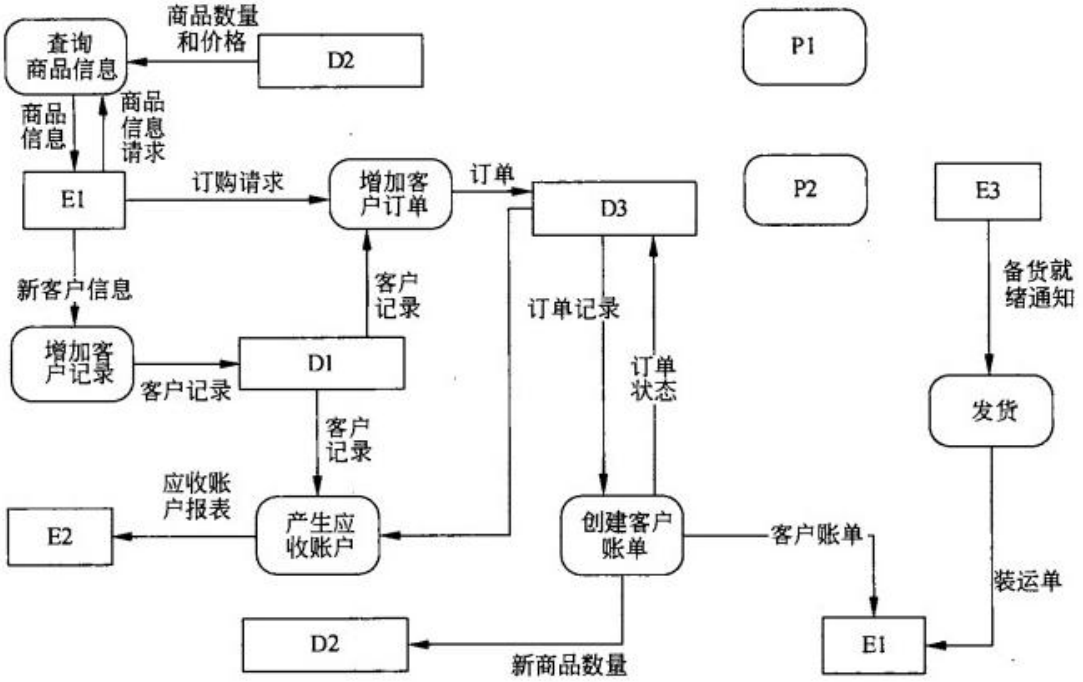



图 1-2　0 层数据流图

【问题 1】

使用说明中的词语，给出图 1-1 中的实体 E1～E3 的名称。

E1：客户 E2:财务部门 E3:仓库

本问题考查顶层 DFD。顶层 DFD 一般用来确定系统边界，将待开发系统看作一个加工，因此图中只有唯一的一个处理和一些外部实体，以及这两者之间的输入输出数据流。题目要求根据描述确定图中的外部实体。根据题目中的描述，并结合已经在顶层数据流图中给出的数据流进行分析。从题目的说明中可以看出：客户提交商品信息请求、订购请求等；将配货单发送给仓库、仓库向系统发送备货就绪通知；发送给财务部门应收账户报表。由此可知该订单系统有客户、仓库和财务部门三个外部实体。对应图 1-1 中数据流和实体的对应关系，可知 E1 为客户，E2 为财务部门，E3 为仓库。本题中需注意说明（4)中向供应商订货是系统外部的行为，因此，供应商并非本系统的外部实体。

【问题 2】

使用说明中的词语，给出图 1-2 中的数据存储 D1～D3 的名称。

D1：客户文件 D2:商品文件 D3：订单文件

本问题考查 0 层 DFD 中数据存储的确定。根据说明中的以下描述：将新客户信息添加到客户文件；从商品文件中查询商品的价格和可订购数量等商品信息；产生订单并添加到订单文件中，得出数据存储为客户文件、商品文件以及订单文件，再根据图 1-2 中 D1 的输入和输出数据流均为客户记录，D2 的输入数据流为从处理“创建客户账单”来的新商品数量，输出数据流为到处理“查询商品信息”的商品数量和价格，D3 的输入数据流为从处理“增加客户订单”来的订单，可知，D1 为客户文件，D2 为商品文件，D3 为订单文件。

**【问题 3】**

（1）给出图 1-2 中处理（加工）P1 和 P2 的名称及其相应的输入输出流。

（2）除加工 P1 和 P2 的输入输出流外，图 1-2 还缺失了 1 条数据流，请给出其起点和终点。

| 起　点 | 终　点 |
| --- | --- |
| | |

注：名称使用说明中的词汇，起点和终点均使用图 1-2 中的符号或词汇。

(1)处理（加工）名称，数据流。

P1: 产生配货单　　P2: 准备发货单

| 数据流名称 | 起　点 | 终　点 |
| --- | --- | --- |
| 订单记录 | D3 或 订单文件 | P1 或 产生配货单 |
| 配货单 | P1 或 产生配货单 | E3 或 仓库 |
| 订单记录 | D3 或 订单文件 | P2 或 准备发货单 |
| 客户记录 | D1 或 客户文件 | P2 或 准备发货单 |
| 发货单 | P2 或 准备发货单 | 发货 |

上表中各行次序无关，但每条数据流的名称、起点、终点必须相对应。
P1 和 P2 可互换，即 P1 为“准备发货单”、P2 为“产生配货单”。

| 起　点 | 终　点 |
| --- | --- |
| D1 或 客户文件 | 创建客户账单 |

本问题考查 0 层 DFD 中缺失的处理和数据流。从说明中的描述功能和图 1-2,可知 产生配货单和准备发货单没有在图 1-2 中，即缺少两个处理：产生配货单和准备发货单。 根据说明(4)中的描述：根据订单记录产生配货单，并将配货单发送给仓库进行备货； 备好货后，发送备货就绪通知。可知，产生配货单的输入流为订单记录，该输入流的起 点为订单文件

（D3)，输出流为配货单，其终点为仓库（E3)。根据说明（5)中的描述：从订单文件中获取订单记录，从客户文件中获取客户记录，并产生发货单。可知，准备发货单的输入流为订单记录和客户记录，订单记录的起点为订单文件，客户记录的起点为客户文件；输出流为发货单。再根据说明(6)中处理发货的描述：根据发货单给客户发货，发货单的终点为处理发货。产生配货单和准备发货单分别对应 P1 和 P2（或 P2 和 P1)。

P1 和 P2 及其输入输出流均识别出来之后，再对照说明和图 1-2，以找出缺少的另外一条数据流。对照说明（7)中的描述：根据订单文件中的订单记录和客户文件中的客户记录，产生并发送客户账单。因此，创建客户账单缺少一条输入流：客户记录，其起点为客户文件（D1)。

**试题二**

某公司拟开发一套小区物业收费管理系统。初步的需求分析结果如下：

（1）业主信息主要包括：业主编号、姓名、房号、房屋面积、工作单位、联系电话等。房号可唯一标识一条业主信息，且一个房号仅对应一套房屋；一个业主可以有一套或多套的房屋。

（2）部门信息主要包括：部门号、部门名称、部门负责人、部门电话等。一个员工只能属于一个部门，一个部门只有一位负责人。

（3）员工信息主要包括：员工号、姓名、出生年月、性别、住址、联系电话、所在部门号、职务和密码等。根据职务不同，员工可以有不同的权限：职务为“经理”的员工具有更改（添加、删除和修改）员工表中本部门员工信息的操作权限；职务为“收费”的员工只具有收费的操作权限。

（4）收费信息包括：房号、业主编号、收费日期、收费类型、数量、收费金额、员工号等。收费类型包括物业费、卫生费、水费和电费，并按月收取，收费标准如表 2-1 所示。其中：物业费=房屋面积（平方米）X 每平方米单价，卫生费=套房数量（套）X 每套房单价，水费=用水数量（吨）X 每吨水单价，电费=用电数量（度）X 每度电单价。

（5）收费完毕应为业主生成收费单，收费单示例如表 2-2 所示。

**表 2-1　收费标准**

| 收费类型 | 单位 | 单价 |
|---|---|---|
| 物业费 | 平方米 | 1.00 |
| 卫生费 | 套 | 10.00 |
| 水　费 | 吨 | 0.70 |
| 电　费 | 度 | 0.80 |

**表 2-2　收费单示例**

房号：A1608　　业主姓名：李斌

| 序号 | 收费类型 | 数量 | 金额 |
|---|---|---|---|
| 1 | 物业费 | 98.6 | 98.60 |
| 2 | 卫生费 | 1 | 10.00 |
| 3 | 水　费 | 6 | 4.20 |
| 4 | 电　费 | 102 | 81.60 |
| 合计 | 壹佰玖拾肆元肆角整 | | 194.40 |

收费日期：2010-9-2　　员工号：001

**【概念模型设计】**

根据需求阶段收集的信息，设计的实体联系图（不完整）如图 2-1 所示。图 2-1 中收费员和经理是员工的子实体。

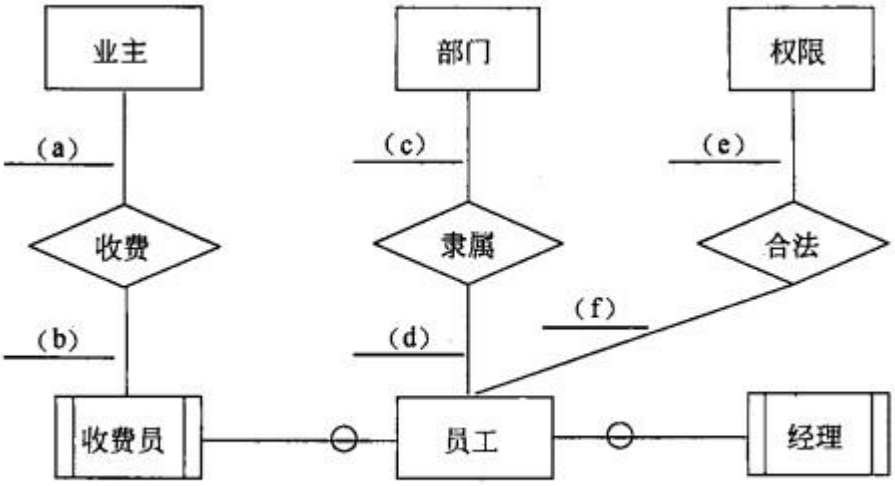



图 2-1　实体联系图

**【逻辑结构设计】**

根据概念模型设计阶段完成的实体联系图，得出如下关系模式（不完整）：

业主（　　(1)　　，姓名，房屋面积，工作单位，联系电话）
员工（　　(2)　　，姓名，出生年月，性别，住址，联系电话，职务，密码）
部门（　　(3)　　，部门名称，部门电话）
权限（职务，操作权限）
收费标准（　　(4)　　）
收费信息（　　(5)　　，收费类型，收费金额，员工号）

**【问题 1】**

根据图 2-1,将逻辑结构设计阶段生成的关系模式中的空（1)～（5)补充完整，然后给出各关系模式的主键和外键。

（1）业主编号，房号
　　主键：房号　　外键：无
（2）员工号，所在部门号
　　主键：员工号　　外键：所在部门号
（3）部门号，部门负责人
　　主键：部门号　　外键：部门负责人
（4）收费类型，单位，单价
　　主键：收费类型　　外键：无
（5）房号，业主编号，收费日期
　　主键：房号，业主编号，收费日期　外键：房号，员工号

根据题意，业主关系中信息主要包括：业主编号、姓名、房号、房屋面积、工作单位、联系

电话等，因此，空（1)应填写“业主编号，房号”。又因为房号可唯一标识一 条业主信息，所以以“房号”为主键。完整的关系模式如下：

业主（业主编号，房号，姓名，房屋面积，工作单位，联系电话）

根据题意，员工信息主要包括：员工号、姓名、出生年月、性别、住址、联系电话、 所在部门号、职务和密码等，因此，空（2)应填写“员工号，所在部门号”。又因为员工号可唯一标识一条员工信息，所以“员工号”为主键。根据题意，一个员工只能属于一个部门，“所在部门号”应参照部门关系的“部门号”，因此，“所在部门号”为外键。 完整的关系模式如下：

员工（员工号，所在部门号，姓名，出生年月，性别，住址，联系电话，职务，密码）

部门信息主要包括：部门号、部门名称、部门负责人、部门电话等，因此，部门关 系的空（3)应填写“部门号，部门负责人”，显然该关系的主键为“部门号”。又因为部门关系的“部门负责人”应参照员工关系的“员工号”，因此，“部门负责人”为外键。

根据题意分析收费标准关系的空（4)应填写“收费类型，单位，单价”，这样收费 信息关系可以根据收费类型（如水费、电费或物业费）去收费标准关系中查出单价来计算收费金额。显然收费标准关系的主键为“收费类型”。

收费信息的空（5)应填写“房号，业主编号，收费日期”，由于“房号，业主编号， 收费日期”能唯一确定该关系的每一个元组，故“房号，业主编号，收费日期”为关系的主键。又由于房号、员工号分别为业主和员工关系的主键，故“房号，员工号”为收费信息关系的外键。完整的关系模式如下：

收费信息（房号，业主编号，收费日期，收费类型，收费金额，员工号）

**【问题 2】**

填写图 2-1 中（a)～（f)处联系的类型（注：一方用 1 表示，多方用 m 或 n 或* 表示)，并补充完整图 2-1 中的实体、联系和联系的类型。

(a) n，或 m，或*
(b) n，或 m，或*
(c) 1
(d) n，或 m，或*
(e) 1
(f) n，或 m，或*

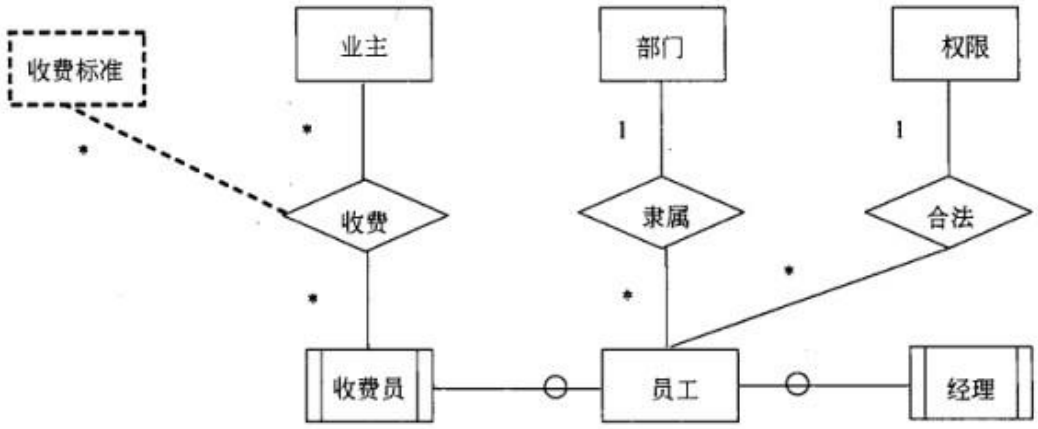



图 2-1　补充完整的实体联系图

根据题意，一个员工可以为多个业主收费，同样一个业主也可以有多个员工为其收费，因此业主和收费员之间的收费联系为多对多。故空（a)应填写*，空（b)应填写*。

因为一个员工只能属于一个部门，所以部门与员工之间的隶属联系是一对多的。故空（c）应填写 1，空（d)应填写*。

根据题意，职务不同员工可以有不同的权限，所以权限和员工之间的合法联系是一对多。

又由于收费员收费时必需根据收费类型（如水费、电费或物业费）到收费标准关系中查出单价来计算收费金额，所以需要增加一个收费标准关系，以及收费标准到收费联系的连线。

**【问题 2】**

业主关系属于第几范式？请说明存在的问题。

业主关系属于第 2 范式。

问题是当某业主有多套住房时，属性“业主编号，姓名，房屋面积，工作单位，联系电话”等信息在业主关系表中重复存储，存在数据冗余。

由业主关系可知：房号一业主编号，业主编号一姓名，房号一姓名，所以存在传递依赖房号一姓名。故业主关系属于第 2 范式。业主关系存在的问题是当某业主有多套住房时，属性“业主编号，姓名，房屋面积，工作单位，联系电话”等信息在业主关系表中重复存储，存在数据冗余。

**试题三**

某网上药店允许顾客凭借医生开具的处方，通过网络在该药店购买处方上的药品。 该网上药店的基本功能描述如下：

（1）注册。顾客在买药之前，必须先在网上药店注册。注册过程中需填写顾客资料以及付款方式（信用卡或者支付宝账户）。此外顾客必须与药店签订一份授权协议书，授 权药店可以向其医生确认处方的真伪。

（2）登录。已经注册的顾客可以登录到网上药房购买药品。如果是没有注册的顾客， 系统将拒绝其登录。

（3）录入及提交处方。登录成功后，顾客按照“处方录入界面”显示的信息，填写开具处方的医生的信息以及处方上的药品信息。填写完成后，提交该处方。

（4）验证处方。对于已经提交的处方（系统将其状态设置为“处方已提交”），其验证过程为：

①核实医生信息。如果医生信息不正确，该处方的状态被设置为“医生信息无效”， 并取消这个处方的购买请求；如果医生信息是正确的，系统给该医生发送处方确认请求， 并将处方状态修改为“审核中”。

②如果医生回复处方无效，系统取消处方，并将处方状态设置为“无效处方”。如果医生没有在 7 天内给出确认答复，系统也会取消处方，并将处方状态设置为“无法 审核”。

③如果医生在 7 天内给出了确认答复，该处方的状态被修改为“准许付款”。系统取消所有未通过验证的处方，并自动发送一封电子邮件给顾客，通知顾客处方 被取消以及取消的原因。

（5）对于通过验证的处方，系统自动计算药品的价格并邮寄药品给已经付款的顾客。该网上药店采用面向对象方法开发，使用 UML 进行建模。系统的类图如图 3-1 所示。

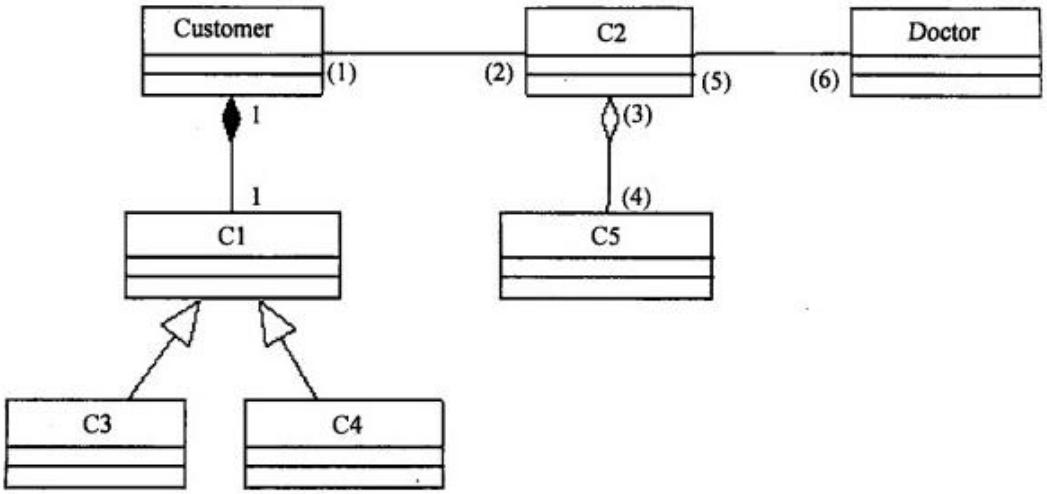



图 3-1　类图

【问题 1】

根据说明中的描述，给出图 3-1 中缺少的 C1～C5 所对应的类名以及（1）～（6）处所对应的多重度。

C1：付款方式　　C2：处方　　C3：信用卡　　C4：支付宝账户
C5：处方上的药品（或药品）　　（C3，C4 可以互换）
（1）1　　（2）0..*　　（3）1
（4）1..*　　（5）0..*　　（6）1

本问题考查 UML 的类图。类图展现了一组对象、接口、协作和它们之间的关系。在面向对象系统的建模中，最常用的模型之一就是类图。

类图用于对系统的静态设计视图建模。这种视图主要支持系统的功能需求，即系统要提供给用户的服务。但对系统的静态设计视图建模时，通常有三种使用方式：

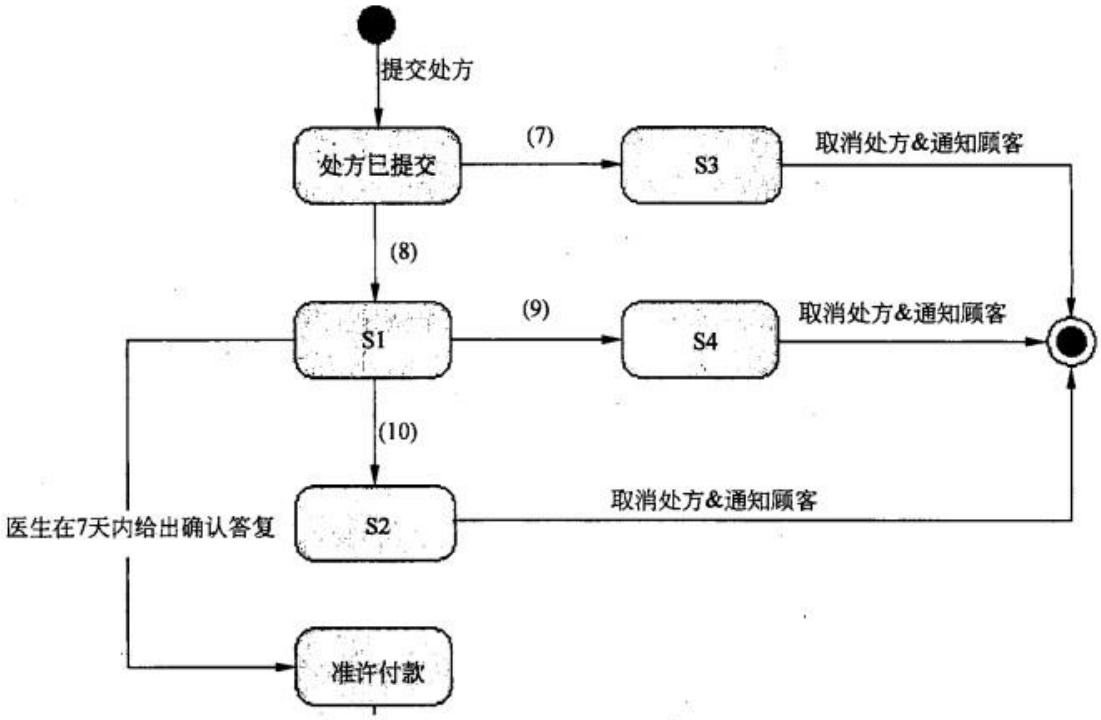



图 3-2　状态图

（1）对系统的词汇建模

对系统的词汇建模涉及做出这样的决定：哪些抽象是考虑中的系统的一部分，哪些抽象处于系统边界之外。用类图详细描述这些抽象和它们的职责。

（2）对简单的协作建模

协作是一些共同工作的类、接口和其他元素的群体，该群体提供的一些合作行为强于所有这些元素的行为之和。例如当对分布式系统的事务语义建模时，不能仅仅盯着一 个单独的类来推断要发生什么，而要有相互协作的一组类来实现这些语义。用类图对这组类以及它们之间的关系进行可视化和详述。

（3）对逻辑数据库模式建模

将模式看作数据库的概念设计的蓝图。在很多领域中，要在关系数据库或面向对象数据库中存储永久信息。可以用类图对这些数据库的模式建模。

本题主要使用类图对系统词汇进行建模。题目中已经给出了类图的基本框架及部分的类，要求考生将类图中其余的类补充完整。在解答这类题目时，需要细心阅读说明中 的文字，并记录和整理其中出现的名词。这些名词将来有可能成为类。其次应特别关注类图中出现的特殊关联关系，如继承关系、聚集/组装关系等。

在本题中，首先考查类图中的 Customer、C2 和 Doctor 这三个类。由说明可知，在网上购药

时，顾客与医生之间不会直接发生交互，而是通过顾客持有的“处方”而发生关联。由此可以确定 C2 对应的类应该是“处方”。

C2 与 C5 之间是聚集关系，其中 C2 表示整体类，C5 表示部分类。由于已经确定了 C2 表示的是“处方”类，那么 C5 表示就应该是处方所包含的内容。处方中包含的是药品，所以 C5 对应的类应该是“处方上的药品，’。

下面来分析类图中的继承关系。继承关系表示类之间的“一般/特殊”关系。C1 表 示一般类，C3 和 C4 是 C1 的两个具体类；并且这三个类与 Customer 之间具有组装关系。 那么在说明中出现的所有名词词汇中，具有明显的一般/特殊关系的就是“付款方式”、“信用卡”和“支付宝账户”。“信用卡”和“支付宝账户”是具体的付款形式，当顾客付款的时候选择二者中的一个。而且每一次付款都与一个特定的顾客（即类 Customer 的一个实例）相关，没有顾客就不会发生付款行为。所以 C1 对应的类应该是“付款方式”、C3 和 C4 分别对应的是类“信用卡”、“支付宝账户”。

多重度表示一个类的实例与多少个另一个类的实例发生关联。因此，在确定多重度时需要关注说明中关于类之间关系的描述。

首先来看 C2 和 C5,这两个类之间是聚集关系。前面已经确定了 C2 和 C5 分别对应类“处方”和“处方上的药品”。一张处方上应包含 1 种或多种药品。这样很容易确定出

(3)和（4)的多重度应分别为 1 以及 1..*。

“处方”和“医生”之间的关系如下：一名医生可以开多张处方，也可以不开处方， 所以(5)处的多重度应该为 0..*;而一张处方必定是由一名医生开具的，所以（6)处的多重度应该为 1。

“顾客”与“处方”之间的关系如下：一个顾客可以持有多张处方来买药，也可以没有处方，这样就不会发生购买行为。所以（2)处的多重度应该为 0..*。而每张处方一定属于一名顾客，所以（1)处的多重度应该为 1。

**【问题 2】**

图 3-2 给出了“处方”的部分状态图。根据说明中的描述，给出图 3-2 中缺少的 S1～S4 所对应的状态名以及（7)～（10)处所对应的迁移（transition)名。

S1：审核中　S2：无法审核　　S3：医生信息无效　　S4：无效处方

（7）医生信息不正确　　（8）医生信息正确

（9）医生回复处方无效　　（10）医生没有在 7 天内给出确认答复

或者：

S2：无效处方　　S4：无法审核

（9）医生没有在 7 天内给出确认答复　　（10）医生回复处方无效

S1、S3、（7）、（8）同上

状态图关注系统的动态视图，它注重描述可能的状态序列，以及在特定状态下对象对外部离散事件的响应动作。

本题考查的是类“处方”的对象的状态变化。关于网上药店对“处方”的处理流程，在说明的(4)验证处方中，给出了详细的描述。对该描述进行分析之后，可以用下面的表来说明“处方”在整个验证流程中所经历的状态。

| 处 方 状 态 | 产生该状态的原因 | 验 证 结 果 |
|---|---|---|
| 医生信息无效 | 医生信息不正确 | 不通过 |
| 审核中 | 医生信息正确 | ---------- |
| 无效处方 | 医生回复处方无效 | 不通过 |
| 无法审核 | 医生没有在 7 天内给出答复 | 不通过 |
| 准许付款 | 医生在 7 天内给出确认答复 | 通过 |

下一步工作就是把上表中的信息与题中的状态图对应起来。

由说明可知，处方提交后的第一步操作就是核实医生信息，而这个操作会产生两种结果：医生信息正确，或者不正确。医生信息不正确会使处方的状态变更为“医生信息无效”，并导致购买行为被取消，即表中的第一行。对于这种情况，“处方”的状态变更 轨迹为：处方已提交→医生信息无效→结束。而在状态图中与这条轨迹匹配的状态序列 就是：处方已提交→S3→结束。由此可以确定，S3 对应的就是状态“医生信息无效”， 而（7)对应的迁移就是“医生信息不正确”。

相应地，就可以判断出（8)应该代表的是核实医生信息的另一种结果，因此（8）对应的迁移应该是“医生信息正确”。由上表可知，医生信息正确时，处方状态会变更为“审核中”，这样 S1 对应的状态就是“审核中”。

但处方在状态“审核中”时，实际上会有三个后续状态：一个是图中已经给出的“准许付款”，另外两个是“无效处方”和“无法审核”。而产生这两个状态的原因分别是“医 生回复处方无效”和“医生没有在 7 天内给出答复”。由此得出，(9)对应“医生回复处方无效”，S4 对应状态“无效处方”；(10)对应“医生没有在 7 天内给出答复”，S2 对应“无法审核”。

如果 S2 为状态“无效处方”，那么（10)就对应着“医生回复处方无效”；S4 对应状态“无

法审核”，那么（9)就对应着“医生没有在 7 天内给出答复”。.

【问题 3】

图 3-1 中的符号“  ”和“  ”在 UML 中分别表示类和对象之间的哪两种关系？两者之间的区别是什么？

 表示组合（composition），  表示聚合（aggregation）。

在组合关系中，整体对象与部分对象具有同一的生存周期。当整体对象不存在时，部分对象也不存在。（1 分）

而在聚合关系中，对整体对象与部分对象没有这样的要求。

**试题四**

堆数据结构定义如下：

对于 n 个元素的关键字序列 a1，a2，…，an}，当且仅当满足下列关系时称其为堆。

$$\begin{cases} a_i \leqslant a_{2i} \\ a_i \leqslant a_{2i+1} \end{cases} \quad 或 \quad \begin{cases} a_i \geqslant a_{2i} \\ a_i \geqslant a_{2i+1} \end{cases} \quad 其中，\ i=1,2,\cdots,\left\lfloor n/2 \right\rfloor$$

在一个堆中，若堆顶元素为最大元素，则称为大顶堆；若堆顶元素为最小元素，则 称为小顶堆。堆常用完全二叉树表示，图 4-1 是一个大顶堆的例子。

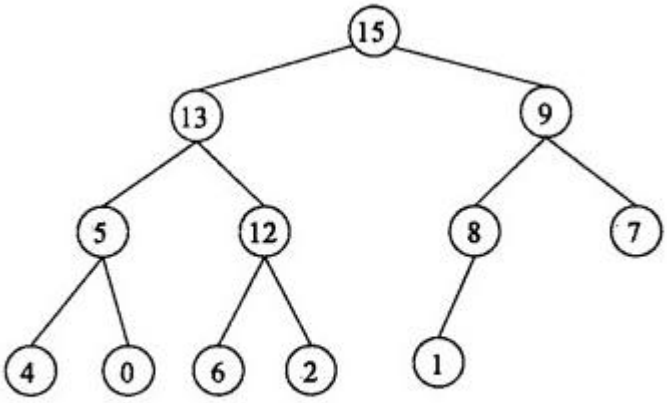



图 4-1　大顶堆示例

堆数据结构常用于优先队列中，以维护由一组元素构成的集合。对应于两类堆结构， 优先队列也有最大优先队列和最小优先队列，其中最大优先队列采用大顶堆，最小优先队列采用小顶堆。以下考虑最大优先队列。

假设现已建好大顶堆 A，且已经实现了调整堆的函数 heapify(A，n，index)。

下面将 C 代码中需要完善的三个函数说明如下：

(1)heapMaximum(A)：返回大顶堆 A 中的最大元素。

(2)heapExtractMax(A)：去掉并返回大顶堆 A 的最大元素，将最后一个元素“提前” 到堆顶位置，并将剩余元素调整成大顶堆。

(3)maxHeapInsert(A, key)：把元素 key 插入到大顶堆 A 的最后位置，再将 A 调整成大顶堆。

优先队列采用顺序存储方式，其存储结构定义如下:

```
#define PARENT(i)  i/2
typedef struct array{
    int *int_array;  //优先队列的存储空间首地址
    int array_size;  //优先队列的长度
    int capacity;    //优先队列存储空间的容量
} ARRAY;
```

【C 代码】

(1) 函数 heapMaximum

```
int heapMaximum(ARRAY *A){  return ___(1)___;  }
```

(2) 函数 heapExtractMax

```
int heapExtractMax(ARRAY *A){
    int max;
    max = A->int_array[0];
    ___(2)___;
    A->array_size --;
    heapify(A,A->array_size,0); //将剩余元素调整成大顶堆
    return max;
}
```

(3) 函数 maxHeapInsert

```
int maxHeapInsert(ARRAY *A,int key){
    int i,*p;
    if (A->array_size == A->capacity)  {  //存储空间的容量不够时扩充空间
        p = (int*)realloc(A->int_array, A->capacity *2 * sizeof(int));
        if (!p) return -1;
        A->int_array = p;
        A->capacity = 2 * A->capacity;
    }
    A->array_size ++;
```

```
    heapify(A,A->array_size,0); //将剩余元素调整成大顶堆
    return max;
}
```

(3) 函数 maxHeapInsert

```
int maxHeapInsert(ARRAY *A,int key){
    int i,*p;
    if (A->array_size == A->capacity)   {  //存储空间的容量不够时扩充空间
        p = (int*)realloc(A->int_array, A->capacity *2 * sizeof(int));
        if (!p) return -1;
        A->int_array = p;
        A->capacity = 2 * A->capacity;
    }
    A->array_size ++;
    i = ___(3)___;
    while (i > 0 &&___(4)___){
        A->int_array[i] = A->int_array[PARENT(i)];
        i = PARENT(i);
    }
    ___(5)___;
return 0;
}
```

【问题 1】

根据以上说明和 C 代码，填充 C 代码中的空（1)～（5)。

(1)A->int_array[0]

(2)A->int_array[0] = A->int_array[A->array_size-1 ]

(3)A->array_size-1

(4)A->int_array[PARENT(i)] <key

(5)A->int_array[i] = key

据题干说明，函数 heapMaximum 返回大顶堆 A 的最大元素，即堆顶元素，因此空（1)处应填 A->int_array[0]。

函数 heapExtractMax(A)取出大顶堆 A 的最大元素，将最后一个元素“提前”到堆顶位置，并将剩余元素调整成大顶堆。因此在将堆顶元素赋给 max 后，应该将堆的最后 一个元素移到堆顶位置，即空（2)处应填 A->int_array[0] = A->int_array[A->array_size-1]。

函数 maxHeapInsert(A，key)把元素 key.插入到大顶堆 A 的最后位置，再将 A 调整成 大顶堆。该函数前面的代码行考虑的是当存储空间不够时扩展存储空间。而后面是根据 该函数

的定义实现的问题求解的算法表示，A->array_size ++;表示为堆的规模增加 1，i 表示堆的最后一个元素的下标，即新插入的元素的下标，应该为 A->array_size-1。while 循环是自下而上调整堆，当还没有到堆顶位置，且新插入的元素大于其父亲元素，即 A->int_array[PARENT(i)] <key 时，i 变为其父亲元素的下标。直到 i 到达堆顶位置，说明新插入的元素为最大值，或者 i 的父亲元素大于新插入的元素，说明新插入的元素在 i 处，因此空（5)处填 A->int_array[i] = key。

**【问题 2】**

根据以上 C 代码，函数 heapMaximum、heapExtractMax 和 maxHeapInsert 的时间复杂度的紧致上界分别为 （6）、（7）和（8)(用 O 符号表示)。

(6)O（1)

(7)O（lgn)

(8)O（lgn)

本问题考查算法的时间复杂度。

根据上述 C 代码，函数 heapMaximum 返回数组 A 的第 1 个元素，因此为常数时间即 O（1)。

函数 heapExtractMax 首先将数组 A 的第 1 个元素的值放到变量 max 中，然后将最后一个元素提到堆顶，最后再进行堆的调整，因此该时间复杂度实际上是调整堆的时间复杂度，即 O(lgn)。

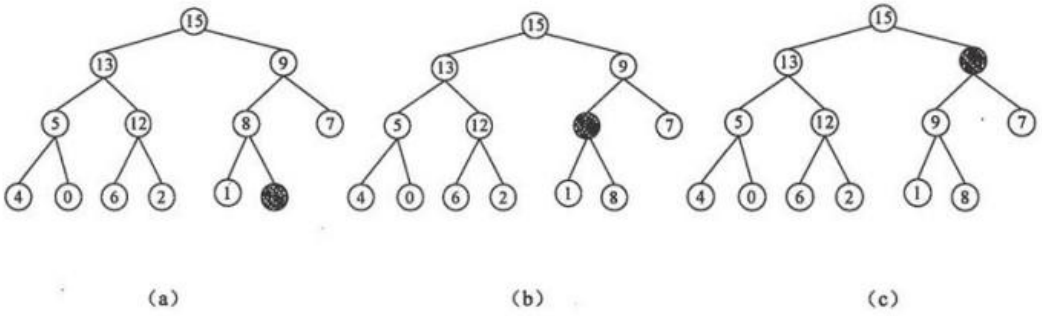
函数 maxHeapInsert 将一个元素 key 插入到堆 A 中，具体的过程为先将堆的规模增加 1，然后将元素插入到堆的最后一个位置，最后自下而上调整该元素，其时间复杂度 为堆（二叉树）的高度，即 O(lgn)。

**【问题 3】**

若将元素 10 插入到堆 A =<15, 13, 9, 5, 12, 8, 7,4, 0, 6,2, 1>中，调用 maxHeapInsert 函数进行操作，则新插入的元素在堆 A 中第 （9）个位置(从 1 开始)。

(9) 3

将元素 10 插入到堆 A =〈15, 13,9,5, 12,8,7,4,0,6,2, 1〉中，根据 maxHeapInsert 函数进行操作，则过程如下图（a)～（c)所示。




(a) (b) (c)

新插入的元素 10 在堆 A 中处于第 3 个位置，15 和 13 分别处于第 1 和第 2 个位置。

**试题五**

某公司的组织结构图如图 5-1 所示，现采用组合（Composition)设计模式来构造该公司的组织结构，得到如图 5-2 所示的类图。

其中 Company 为抽象类，定义了在组织结构图上添加（Add)和删除（Delete)分公司/办事处或者部门的方法接口。类 ConcreteCompany 表示具体的分公司或者办事处，分公司或办事处下可以设置不同的部门。类 HRDepartment 和 FinanceDepartment 分别表示人力资源部和财务部。

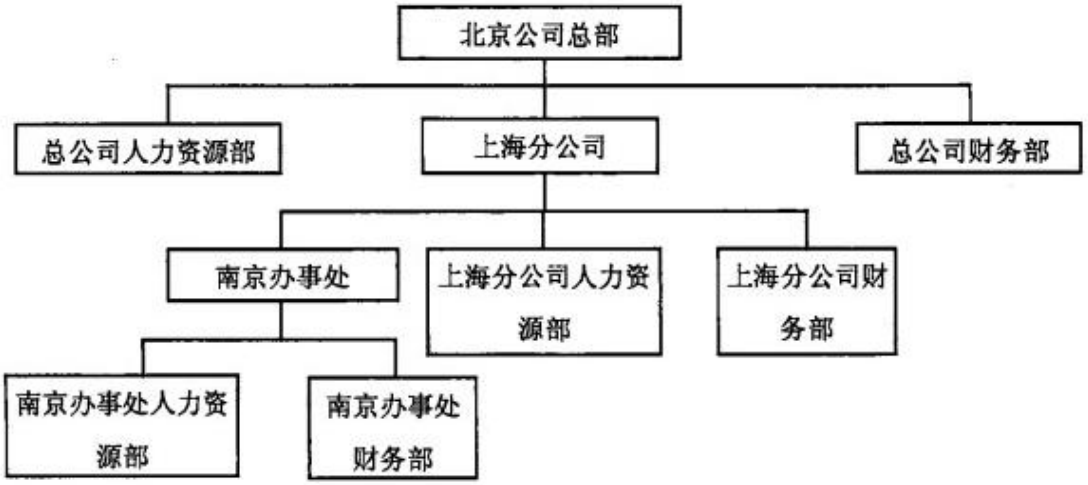



图 5-1　组织结构图

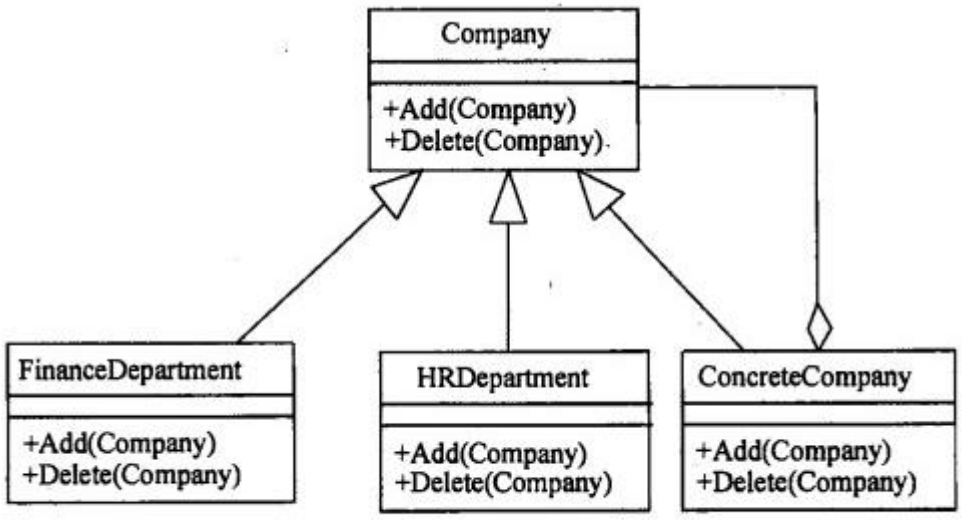



图 5-2　类图

【问题 1】

【C++代码】

```
#include <iostream>
#include <list>
#include <string>
using namespace std;

class Company {                    // 抽象类
    protected:
       string name;
    public:
       Company(string name) { __(1)__ = name; }
       __(2)__;                    // 增加子公司、办事处或部门
       __(3)__;                    // 删除子公司、办事处或部门

};

class ConcreteCompany : public Company {
    private:
       list< __(4)__ > children;                    // 存储子公司、办事处或部门
    public:
       ConcreteCompany(string name) : Company(name) { }
       void Add(Company* c) { __(5)__.push_back(c); }
       void Delete(Company* c) { __(6)__.remove(c); }
};

class HRDepartment : public Company {
    public:
       HRDepartment(string name) : Company(name) {}      // 其他代码省略
};

class FinanceDepartment : public Company {
    public:
       FinanceDepartment(string name) : Company(name) {}  // 其他代码省略
};

void main() {
  ConcreteCompany *root = new ConcreteCompany("北京总公司");
```

```
    root->Add(new HRDepartment("总公司人力资源部"));
    root->Add(new FinanceDepartment("总公司财务部"));

    ConcreteCompany *comp = new ConcreteCompany("上海分公司");
    comp->Add(new HRDepartment("上海分公司人力资源部"));
    comp->Add(new FinanceDepartment("上海分公司财务部"));
    ___(7)___;

    ConcreteCompany *comp1 = new ConcreteCompany("南京办事处");
    comp1->Add(new HRDepartment("南京办事处人力资源部"));
    comp1->Add(new FinanceDepartment("南京办事处财务部"));
    ___(8)___;     //其他代码省略
}
```

（1）this->name

（2）virtual void Add(Company* c) = 0

（3）virtual void Delete(Company* c) = 0

（4）Company*

（5）children

（6）children

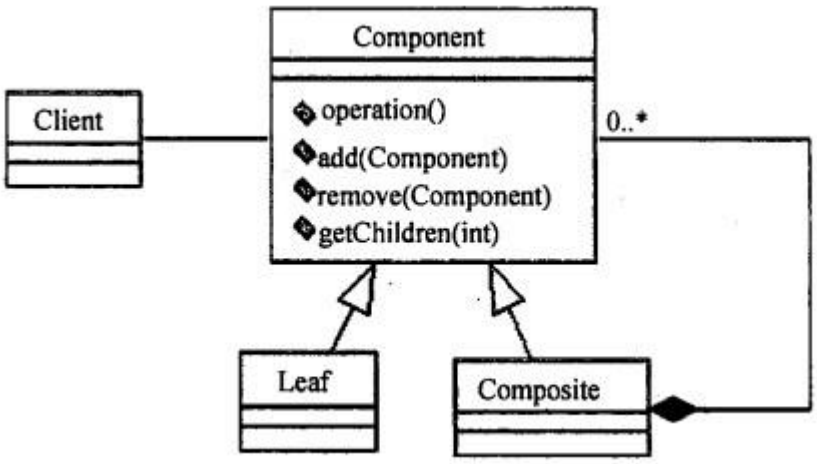
（7）root->Add(comp)

（8）comp->Add(comp1)

Composite 模式将对象组合成树形结构以表示“整体-部分”的层次结构，其中的组合对象使得用户可以组合基元对象以及其他的组合对象，从而形成任意复杂的结构。

Composite 模式使得用户对单个对象和组合对象的使用具有一致性。

Composite 模式的结构如下图所示。



其中：

•类 Component 为组合中的对象声明接口，在适当的情况下，实现所有类共有接口的缺省行为，声明一个接口用于访问和管理 Component 的子部件；

•类 Leaf 在组合中表示叶节点对象，叶节点没有子节点；并在组合中定义图元对象的行为；

•类 Composite 定义有子部件的那些部件的行为，存储子部件，并在 Component 接口中实现与子部件有关的操作；

•类 Client 通过 Component 接口操纵组合部件的对象。

下列情况可以使用 Composite 模式：

(1)表示对象的整体-部分层次结构；

(2)希望用户忽略组合对象与单个对象的不同，用户将统一地使用组合结构中的所有对象。

图 5-2 中的 Company 对应的就是上图中的类 Component， ConcreteCompany 对应的是类 Composite;而上图中的 FinanceDepartment 和 HRDepartment 扮演的就是类 Leaf 的角色。

由于类 Company 的作用是为其子类提供统一的操作接口，所以将其定义为抽象类。 在 C#中，抽象类的定义是：至少包含一个纯虚拟函数的类。而纯虚拟函数是没有函数体的虚拟函数，其作用是为子类提供统一接口。若要使用纯虚拟函数，必须在子类中对其进行重置。定义纯虚拟函数的语法为：

virtual <返回值><函数名>(<参数列表>)=0;

空(1)～(3)考查的是如何定义抽象类 Company。Company 提供了两个方法接口 Add 和 Delete，即该类中应包含两个纯虚拟函数。如何确定 Add.和 Delete 的函数原型呢？ 这要借助于 Company 的子类。因为子类重置父类定义的虚拟函数时，不能改变其接口定义。所以从 ConcreteCompany 中的 Add 和 Delete 方法就能够确定出空(2)和(3)处应 分别填入 “virtual void Add(Company* c) = 0” 和 “virtual void Delete(Company* c) = 0”。 空(1)考察的是在构造函数中如何给数据成员赋初值。当构造函数的参数与类的数据成员同名时，可以借助 this 指针来进行区别，因此空(1)处应填入 this->name。

空(4)～(6)考查对模式中 Composite 节点的定义。由图 5-2 可知，ConcreteCompany 与 Company 之间是聚集关系，即 ConcreteCompany 的实例中包含多个 Company 的子类 的实例。为了表示这种聚集关系，使用了 C++标准类库中的类模板 list。C++的类模板必须在实例化之后才能使用。实例化类模板时，要给出类型实参。由于 children 表示的是 类 Company 的子类的实例集合，所以空(4)处应填入 Company*。空(5)和(6)处 分别使用了 list 中提供的方法来实现添加和删除子公司、办事处或部门。children 是 list 的实例，所以空(5)

和（6)处都应填入 children。

空（7)和（8)考查的是组合模式的使用。由图 5-1 可知，组织结构图的根目录是 “北京总公司”，“上海分公司”应该插入在根目录之下。所以空（7)处应填入 root->Add(Comp)。而“南京办事处”是以“上海分公司”为根的子树中的节点，应插入 在“上海分公司”这个节点的下面。对象 comp 表示的是以“上海分公司”为根的子树的根节点，所以空（8)处应该填入 comp->Add(comp1)。

**试题六**

某公司的组织结构图如图 6-1 所示，现采用组合（Composition)设计模式来设计，得到如图 6-2 所示的类图。

其中 Company 为抽象类，定义了在组织结构图上添加（Add)和删除（Delete)分公司/办事处或者部门的方法接口。类 ConcreteCompany 表示具体的分公司或者办事处，分公司或办事处下可以设置不同的部门。类 HRDepartment 和 FinanceDepartment 分别表示人力资源部和财务部。

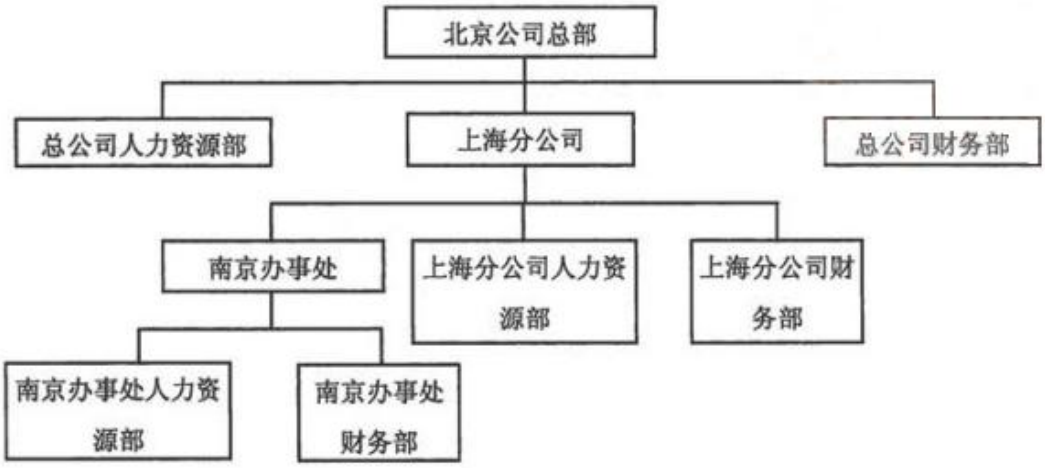



图 6-1 组织结构图

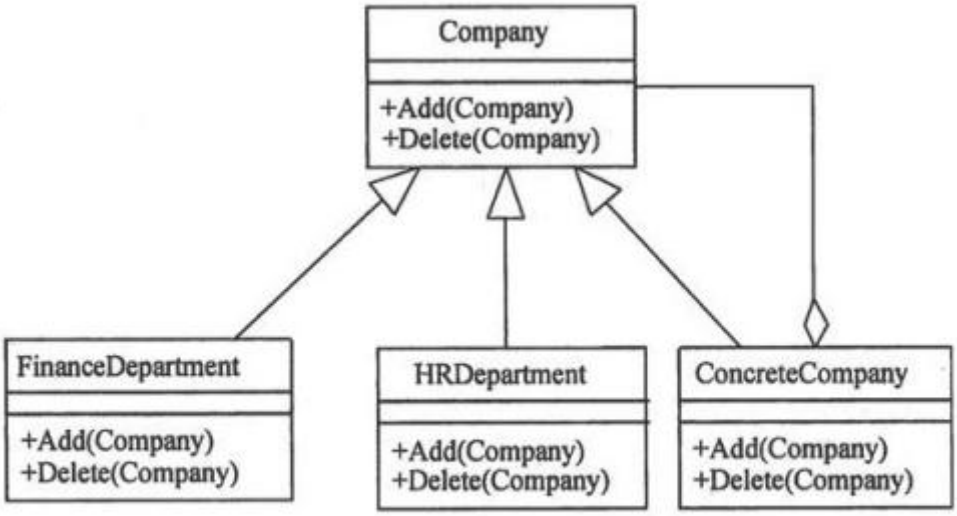



图 6-2 类图

【问题 1】

【Java 代码】

```
import java.util.*;
__(1)__ Company {
   protected String  name;
   public Company(String name) { __(2)__ = name; }
   public abstract void Add(Company c);        // 增加子公司、办事处或部门
   public abstract void Delete(Company c);     // 删除子公司、办事处或部门
}

class ConcreteCompany extends Company {
   private List< __(3)__ > children = new ArrayList< __(4)__ >();
// 存储子公司、办事处或部门
   public ConcreteCompany(String name) { super(name); }
   public void Add(Company c) { __(5)__.add(c);  }

  public void Delete(Company c) { __(6)__.remove(c);  }
}

class HRDepartment extends Company {
   public HRDepartment(String name)  { super(name); }
   // 其他代码省略
}

class FinanceDepartment extends Company {
   public FinanceDepartment(String name)  { super(name); }
   // 其他代码省略
}

public class Test  {
  public static void main(String[] args) {
    ConcreteCompany root = new ConcreteCompany("北京总公司");
    root.Add(new HRDepartment("总公司人力资源部"));
    root.Add(new FinanceDepartment("总公司财务部"));

    ConcreteCompany comp = new ConcreteCompany("上海分公司");
    comp.Add(new HRDepartment("上海分公司人力资源部"));
    comp.Add(new FinanceDepartment("上海分公司财务部"));
    __(7)__;
```

```
        ConcreteCompany comp1 = new ConcreteCompany("南京办事处");
        comp1.Add(new HRDepartment("南京办事处人力资源部"));
        comp1.Add(new FinanceDepartment("南京办事处财务部"));
        ____(8)____;        // 其他代码省略
    }
}
```

（1）abstract class

（2）this.name

（3）Company

（4）Company

（5）children

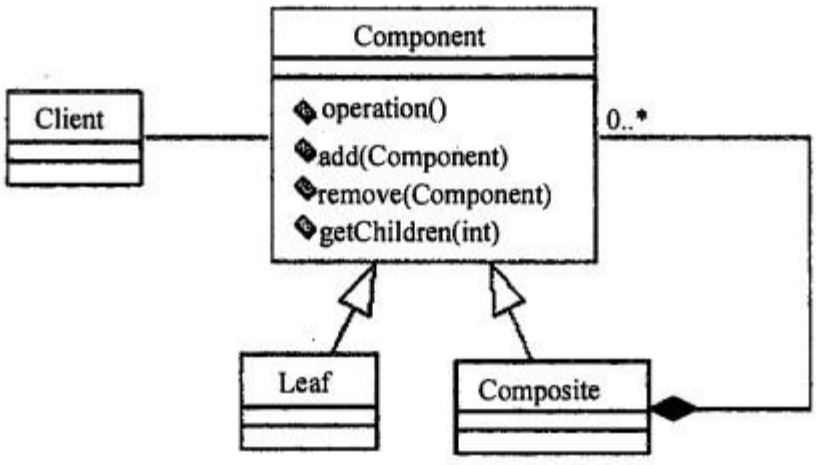
（6）children

（7）root.Add(comp)

（8）comp.Add(comp1)

Composite 模式将对象组合成树形结构以表示“整体-部分”的层次结构，其中的组合对象使得你可以组合基元对象以及其他的组合对象，从而形成任意复杂的结构。 Composite 模式使得用户对单个对象和组合对象的使用具有一致性。

Composite 模式的结构如下图所示。




其中：

•类 Component 为组合中的对象声明接口，在适当的情况下，实现所有类共有接口的缺省行为，声明一个接口用于访问和管理 Component 的子部件；

•类 Leaf 在组合中表示叶节点对象，叶节点没有子节点；并在组合中定义图元对象的行为；

•类 Composite 定义有子部件的那些部件的行为，存储子部件，并在 Component 接口中实现

与子部件有关的操作；

•类 Client 通过 Component 接口操纵组合部件的对象。

下列情况可以使用 Composite 模式：

(1)表示对象的整体-部分层次结构；

(2)希望用户忽略组合对象与单个对象的不同，用户将统一地使用组合结构中的所有对象。

图 6-2 中的 Company 对应的就是上图中的类 Component, ConcreteCompany 对应的是类 Composite;而上图中的 FinanceDepartment 和 HRDepartment 扮演的就是类 Leaf 的角色。

由于类 Company 的作用是为其子类提供统一的操作接口，所以将其定义为抽象类。空（1）～（2)考查的是如何定义抽象类 Company。在 Java 中，可以通过在类名之前加 abstract 关键字来定义抽象类，因此空（1)处应填入 abstract class。空（2)考查的是在构造函数中如何给数据成员赋初值。当构造函数的参数与类的数据成员同名时，可以借助 this 指针来进行区别，因此空（2)处应填入 this.name。

空（3)～（6)考查对模式中 Composite 节点的定义。由图 5-2 可知，ConcreteCompany 与 Company 之间是聚集关系，即 ConcreteCompany 的实例中包含多个 Company 的子类 的实例。为了表示这种聚集关系，使用了 Java 包中的类模板 List。类模板必须在实例化之后才能使用。实例化类模板时，要给出类型实参。由于 children 表示的是类 Company 的子类的实例集合，所以空（3)和（4)处都应填入 Company。空（5)和（6)处分别 使用了 List 中提供的方法来实现添加和删除子公司、办事处或部门。children 是 list 的实例，所以空(5)和(6)处都应填入 children。

空（7)和（8)考查的是组合模式的实用。由图 6-1 可知，组织结构图的根目录是 “北京总公司”,“上海分公司”应该插入在根目录之下。所以空(7)处应填入 root.Add(comp)0 而“南京办事处”是以“上海分公司”为根的子树中的节点，应插入在“上海分公司”这个节点的下面。对象 comp 表示的是以“上海分公司”为根的子树的根节点，所以空（8)处应该填入 comp.Add(comp1)。