在 CPU 中用于跟踪指令地址的寄存器是 (1).

(1)A.地址寄存器（MAR)　　B.数据寄存器（MDR)

C.程序计数器（PC)　　D.指令寄存器（IR)

【答案】C

【解析】本题考查寄存器的基本知识。

CPU 中通常设置一些寄存器，用于暂时存储程序运行过程中的相关信息。其中，通用寄存器常用于暂存运算器需要的数据或运算结果，地址寄存器和数据寄存器用于访问内存时的地址和数据暂存，指令寄存器用于暂存正在执行的指令，程序计数器中存放待执行的指令的地址。

指令系统中采用不同寻址方式的目的是 (2).

(2)A.提高从内存获取数据的速度　　B.提高从外存获取数据的速度

C.降低操作码的译码难度　　D.扩大寻址空间并提高编程灵活性

【答案】D

【解析】本题考查指令系统的基本概念。

寻址方式是指寻找操作数或操作数地址的方式。指令系统中采用不同寻址方式的目的是为了在效率和方便性上找一个平衡。立即寻址和寄存器寻址在效率上是最快的， 但是寄存器数目少，不可能将操作数都存入其中等待使用，立即寻址的使用场合也非常有限，这样就需要将数据保存在内存中，然后使用直接寻址、寄存器间接寻址、寄存器相对寻址、基址加变址寻址、相对基址及变址寻址等寻址方式将内存中的数据移入寄存器中。

在计算机系统中采用总线结构，便于实现系统的积木化构造，同时可以 (3).

(3)A.提高数据传输速度　　B.提高数据传输量

C.减少信息传输线的数量　　D.减少指令系统的复杂性

【答案】C

【解析】本题考查计算机系统的基础知识。

总线是连接计算机有关部件的一组信号线，是计算机中用来传送信息代码的公共通道。采用总线结构主要有以下优点：简化系统结构，便于系统设计制造；大大减少了连线数目，便于布线，减小体积，提高系统的可靠性；便于接口设计，所有与总线连接的设备均采用类似的接口；便于系统的扩充、更新与灵活配置，易于实现系统的模块化； 便于设备的软件

设计，所有接口的软件就是对不同的口地址进行操作；便于故障诊断和维修，同时也降低了成本。

原码表示法和补码表示法是计算机中用于表示数据的两种编码方法，在计算机系统中常采用补码来表示和运算数据，原因是采用补码可以 (4)。

(4)A.保证运算过程与手工运算方法保持一致　　B.简化计算机运算部件的设计

C.提高数据的运算速度　　D.提高数据的运算精度

【答案】B

【解析】本题考查数据表示的基础知识。

使用补码表示数据时，可以将符号位和其他位统一处理，减法也可按加法来处理，从而简化运算部件的设计。

计算机中的浮点数由三部分组成：符号位 S，指数部分 E（称为阶码）和尾数部分 M。在总长度固定的情况下，增加 E 的位数、减少 M 的位数可以 (5)。

(5)A.扩大可表示的数的范围同时降低精度　　B.扩大可表示的数的范围同时提高精度

C.减小可表示的数的范围同时降低精度　　D.减小可表示的数的范围同时提高精度

【答案】A

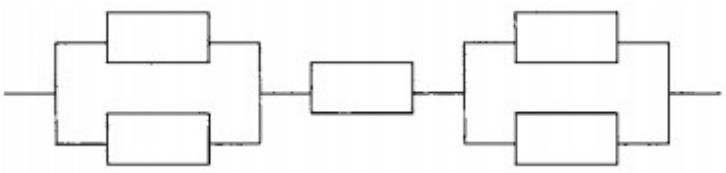
【解析】本题考查数据表示的基础知识。

浮点数在计算机中用以近似表示任意某个实数，一个浮点数 a 可如下表示：

a = MXbE

其中，尾数部分 M 的位数越多，数的精度越高，指数部分 E 的位数越多，能表示的数值越大。因此在总长度固定的情况下，增加 E 的位数、减少 M 的位数可以扩大可表示的数的范围同时降低精度。

某计算机系统由下图所示的部件构成，假定每个部件的千小时可靠度都为 R，则该系统的千小时可靠度为 (6).



(6)A.R + 2R/4　　B.R + R2/4　　C.R(1-(1-R)2)　　D.R(1-(1-R)2)2

【答案】D

【解析】本题考查系统可靠性方面的基础知识。

由子系统构成串联系统时，其中任何一个子系统失效就足以使系统失效，其可靠度等于各子系统可靠度的乘积;构成并联系统时，只要有一个子系统正常工作，系统就能正常工作。设每个子系统的可靠性分别以 R1，R2，…，RN 表示，则整个系统用串联方式构造时的可靠度为 R=R1×R2…×…RN，整个系统用串联方式构造时的可靠度为 R=1-(1-R1)(1-R2)…(1-RN)

因此，本系统的可靠度为 R(1-(1-R)2)2

用户 A 从 CA 获得用户 B 的数字证书，并利用 （7） 验证数字证书的真实性。

(7)A.B 的公钥　　B.B 的私钥　　C.CA 的公钥　　D.CA 的私钥

【答案】C

【解析】本题考查数字证书和 CA 证书授权（Certificate Authority)中心的相关知识点。

数字证书是由权威机构 CA 证书授权（CertificateAuthority)中心发行的，能提供在 Internet 上进行身份验证的一种权威性电子文档，人们可以在互联网交往中用它来 证明自己的身份和识别对方的身份。

数字证书包含版本、序列号、签名算法标识符、签发人姓名、有效期、主体名、主体公钥信息等并附有 CA 的签名，用户 A 获取用户 B 的数字证书后通过验证 CA 的签名

来确认数字证书的有效性。验证 CA 的签名时使用的是 CA 的公钥。

宏病毒一般感染以 （8） 为扩展名的文件。

(8)A.EXE　　B.COM　　C.DOC　　D.DLL

【答案】C

【解析】本题考查计算机病毒的基础知识。

病毒文件名称一般分为三部分，第一部分表示病毒的类型，如 Worm 表示蠕虫病毒，Trojan 表示特洛伊木马，Backdoor 表示后门病毒，Macro 表示宏病毒等。

宏病毒感染的对象是使用某些程序创建的文本文档、数据库、电子表格等文件。

在 IE 浏览器中，安全级别最高的区域设置是 (9)。

(9)A. Internet　　B. 本地 Intranet　　C. 可信站点　　D. 受限站点

【答案】D

【解析】本题考查 IE 浏览器不同区域安全等级的基础知识。

在 IE 浏览器中，安全等级从可信站点、本地 Intranet、Internet 到受限站点默认情况下依次为低、中低、中、高，逐步提升，如下图所示。

下列关于软件著作权中翻译权的叙述正确的是：翻译权是指 （10） 的权利。

(10)A. 将原软件从一种自然语言文字转换成另一种自然语言文字

B. 将原软件从一种程序设计语言转换成另一种程序设计语言

C. 软件著作权人对其软件享有的以其它各种语言文字形式再表现

D. 对软件的操作界面或者程序中涉及的语言文字翻译成另一种语言文字

【答案】B

【解析】

软件著作权中翻译权是指以不同于原软件作品的一种程序语言转换该作品原使用的程序语言，而重现软件作品内容的创作的产品权利。简单地说，也就是指将原软件从一种程序语言转换成另一种程序语言的权利。

某软件公司研发的财务软件产品在行业中技术领先，具有很强的市场竞争优势。为确保其软件产品的技术领先及市场竞争优势，公司采取相应的保密措施，以防止软件技术秘密的

外泄。并且，还为该软件产品冠以“用友”商标，但未进行商标注册。此情况下，公司仅享有该软件产品的(11)。

(11)A.软件著作权和专利权　　B.商业秘密权和专利权

C.软件著作权和商业秘密权　　D.软件著作权和商标权

【答案】C

【解析】

由于是软件公司研发的财务软件产品，因此，软件公司享有该软件产品的软件著作权。又由于商业秘密的构成条件是：商业秘密必须具有未公开性，即不为公众所知悉； 商业秘密必须具有实用性，即能为权利人带来经济效益；商业秘密必须具有保密性，即采取了保密措施。

综上所述，公司仅享有该软件产品的软件著作权和商业秘密权。

以下编码方法中，(12)属于熵编码。

(12)A.哈夫曼编码　　B.小波变换编码　　C.线性预测编码　　D.行程编码

【答案】A

【解析】

在计算机信息处理中，“哈夫曼编码”是一种一致性编码法（又称“熵编码法”），用于数据的无损耗压缩。这一术语是指使用一张特殊的编码表将源字符（例如某文件中的一个符号）进行编码。这张编码表的特殊之处在于，它是根据每一个源字符出现的估算概率而建立起来的。出现概率高的字符使用较短的编码，出现概率低的则使用较长的编码，这便使编码之后的字符串的平均期望长度降低，从而到无损压缩数据的目的。

CIF 视频格式的图像分辨率为(13)。

(13)A.352X240　　B.352X288　　C.640X480　　D.320X240

【答案】B

【解析】

CIF 是常用的标准化图像格式（Common Intermediate Format）。在 H.323 协议簇中， 规定了视频采集设备的标准采集分辨率，CIF=352X288 像素。

由 ISO 制定的 MPEG 系列标准中，(14)是多媒体内容描述接口标准。

(14)A.MPEG-1　　B.MPEG-2　　C.MPEG-4　　D.MPEG-7

【答案】D

【解析】

由 ISO 制定的 MPEG 系列标准中,MPEG-7 称为“多媒体内容描述接口”(multimedia content description interface)。该标准是建立对多媒体内容的描述标准，满足包括静止图像、图形、3D 模型、音频、话音、视频以及以上元素组合在一起的合成多媒体信息的应用领域的要求，并兼顾标准的通用性和扩展性的要求。

包含 8 个成员的开发小组的沟通路径最多有 (15)条。

(15)A.28　　B.32　　C.56　　D.64

【答案】A

【解析】本题考查项目管理及工具技术。

软件开发小组的沟通路径受到小组组织形式和规模的影响。若任意小组成员之间均可能有沟通路径，则可用完全连通图来对开发小组的沟通路径建模，最多的沟通路径为完全连通图的边数，即 n 个成员的开发小组的沟通路径是 n(n-1)/2,因此 8 个成员的开发小组的沟通路径有 28 条。

模块 A 直接访问模块 B 的内部数据，则模块 A 和模块 B 的耦合类型为 (16).

(16)A.数据耦合　　B.标记耦合　　C.公共耦合　　D.内容耦合

【答案】D

【解析】本题考查软件的分析与设计方法。

模块独立性是创建良好设计的一个重要原则,一般采用模块间的耦合和模块的内聚两个准则来进行度量。耦合是模块之间的相对独立性的度量，模块之间的连接越紧密， 联系越多，耦合性就越高，而其模块独立性就越弱。一般来说，模块之间的耦合有 7 种类型，根据耦合性从低到高为非直接耦合、数据耦合、标记耦合、控制耦合、外部耦合、公共耦合和内容耦合。如果一个模块访问另一个模块时，彼此之间是通过数据参数（不是控制参数、公共数据结构或外部变量）来交换输入、输出信息的，则称这种耦合为数据耦合；如果一组模块通过数据结构本身传递,则称这种耦合为标记耦合;若一组模块都访问同一个公共数据环境，则它们之间的耦合就称为公共耦合；若一个模块直接访问另一个模块的内部数据、一个模块不通过正常入口转到另一个模块内部、两个模块有一部分程序代码重叠或者一个模块有多个入口，上述几个情形之一发生则说明两个模块之间就发生了内容耦合。

下列关于风险的叙述不正确的是：风险是指 (17).

(17)A.可能发生的事件

B.一定会发生的事件

C.会带来损失的事件

D.可能对其进行干预，以减少损失的事件

【答案】B

【解析】本题考查风险分析和风险控制技术。

风险是一种具有负面后果的、人们不希望发生的事件。通常认为风险具有以下特点： 风险是可能发生的事件，其发生的可能性用风险概率来描述；风险是会给项目带来损失的事件；可能对风险进行干预，以期减少损失。针对每一种风险，应弄清可能减少造成损失或避免损失的程度。对风险加以控制，采取一些有效的措施来降低风险或是消除风险。

下列关于项目估算方法的叙述不正确的是 （18）。

(18)A.专家判断方法受到专家经验和主观性影响

B.启发式方法（如 COCOMO 模型）的参数难以确定

C.机器学习方法难以描述训练数据的特征和确定其相似性

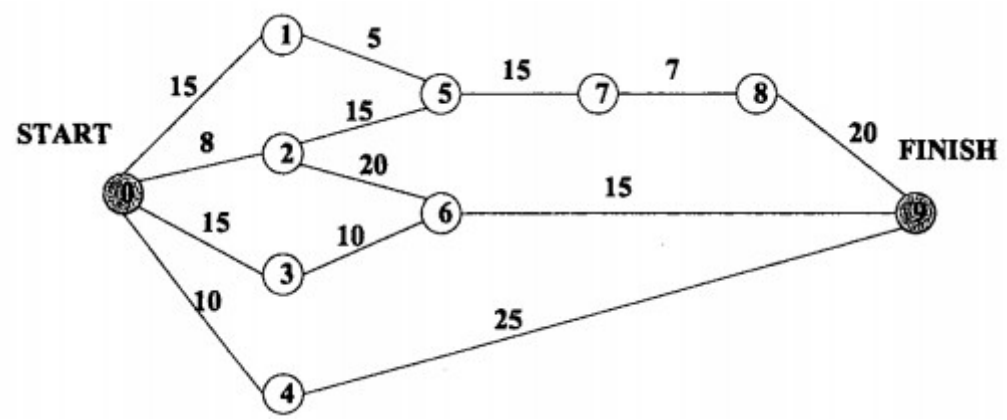
D.结合上述三种方法可以得到精确的估算结果

【答案】D

【解析】本题考查项目管理及工具技术。

项目估算是项目计划和管理的一个至关重要的方面。成本超出某个限度可能导致客户取消项目，而过低的成本估算可能会迫使开发小组投入大量的时间却没有相应的经济回报。目前常用的项目估算方法有专家判断方法，该方法受到专家经验和主观性等方面的影响；算法方法，根据某个计算模型来估算项目开发成本，如启发式方法 COCOMO 模型，但这些模型中的参数难以确定；机器学习方法，如根据过去的项目开发数据，建立分类模型，预测新项目的开发成本，但这类方法难以定义训练数据的特征以及定义数据对象之间的相似性。即使结合多种方法，上述问题仍然存在，因此并不能得到精确地估算结果。

下图是一个软件项目的活动图，其中顶点表示项目里程碑，边表示包含的活动，边上的权重表示活动的持续时间，则里程碑 (19)在关键路径上。



(19)A.1　　B.2　　C.3　　D.4

【答案】B

【解析】本题考查项目管理及工具技术。

根据关键路径法，计算出关键路径为0—2—5—7—8—9,关键路径长度为65。因此里程碑2在关键路径上，而里程碑1、3和4不在关键路径上。

算术表达式采用逆波兰式表示时不用括号，可以利用 （20）进行求值。与逆波兰式ab-cd+*对应的中缀表达式是（21）。

(20)A.数组　　B.栈　　C.队列　　D.散列表

(21)A.a-b+c*d　　B.(a-b)*c+d　　C.(a-b)*(c+d)　　D.a-b*c+d

【答案】B　C

【解析】本题考查程序语言的基础知识。

逆波兰式（reverse polish notation，也叫后缀表达式）是将运算符写在操作数之后的表达式表示方法。对逆波兰式进行求值的方法是：从左至右扫描表达式，遇到操作数则压栈，遇到运算符号则从栈中弹出操作数进行运算，然后将运算结果压入栈中，重复该过程直到表达式结束，最后的结果为栈顶元素。由于控制上比较简单，所以逆波兰式更便于计算。

表达式“a-b+c*d”的后缀式为“ab-cd*+”。

表达式“(a-b)* c+d”的后缀式为“ab-c*d+”。

表达式“(a-b)* (c+d)”的后缀式为“ab-cd+*”。

表达式“a-b*c+d”的后缀式为“abc*-d+”。

若一种程序设计语言规定其程序中的数据必须具有类型，则有利于（22).

①在翻译程序的过程中为数据合理分配存储单元

②对参与表达式计算的数据对象进行检查

③定义和应用动态数据结构

④规定数据对象的取值范围及能够进行的运算

⑤对数据进行强制类型转换

(22)A.①②③　　B.①②④　　C.②④⑤　　D.③④⑤

【答案】B

【解析】本题考查程序语言的基础知识。

程序中的数据具有类型属性时，就可以规定数据对象的取值范围及能够进行的运算，在运算前便于进行类型检查，也更有利于为数据合理分配存储单元。

某文件管理系统在磁盘上建立了位示图（bitmap)，记录磁盘的使用情况。若系统的字长为32位，磁盘上的物理块依次编号为0、1、2、…，那么4096号物理块的使用情况在位示图中的第（23)个字中描述:若磁盘的容量为200GB，物理块的大小为1MB，那么位示图的大小为（24)个字。

(23)A.129　　B.257　　C.513　　D.1025

(24)A.600　　B.1200　　C.3200　　D.6400

【答案】A　D

【解析】本题考查操作系统文件管理方面的基础知识。

根据题意，系统中字长为32位，可记录32个物理块的使用情况，这样0～31号物理块的使用情况在位示图中的第1个字中描述，32～63号物理块的使用情况在位示图中的第2个字中描述，……，4064～4095号物理块的使用情况在位示图中的第128个字中描述，4096～4127号物理块的使用情况在位示图中的第129个字中描述。

根据题意，若磁盘的容量为200GB，物理块的大小为1MB，那么该磁盘就有204800个物理块（即200X1024)，位示图的大小为204800/32=6400个字。

系统中有R类资源m个，现有n个进程互斥使用。若每个进程对R资源的最大需求为vv，那么当m、n、vv分别取下表中的值时，对于表中的①～⑥种情况，(25)可能会发生死锁。若将这些情况的m分别加上（26)，则系统不会发生死锁。

	①	②	③	④	⑤	⑥
m	3	3	5	5	6	6
n	2	3	2	3	3	4
w	2	2	3	3	3	2

(25)A.①②⑤　　B.③④⑤　　C.②④⑤　　D.②④⑥

(26)A.1、1 和 1　　B.1、1 和 2　　C.1、1 和 3　　D.1、2 和 1

【答案】C　D

【解析】本题考查操作系统进程管理方面的基础知识。

试题(25)的正确答案是分析如下：

情况①不会发生死锁：已知系统资源 R 的数目等于 3,进程数等于 2,每个进程对 R 资源的最大需求为 2。若系统为 2 个进程各分配 1 个资源，系统可供分配的剩余资源数等于 1，则可以保证 1 个进程得到所需资源运行完毕。当该进程释放资源后又能保证另一个进程运行完毕，故系统不会发生死锁。

情况②会发生死锁：已知系统资源 R 的数目等于 3,进程数等于 3,每个进程对 R 资源的最大需求为 2。若系统为 3 个进程各分配 1 个资源，系统可供分配的剩余资源数等于 0,则无法保证进程得到所需资源运行完毕，故系统会发生死锁。

情况③不会发生死锁：已知系统资源 R 的数目等于 5,进程数等于 2,每个进程对 R 资源的最大需求为 3。若系统为两个进程各分配两个资源，系统可供分配的剩余资源数 等于 1，则可以保证 1 个进程得到所需资源运行完毕。当该进程释放资源后又能保证另 一个进程运行完毕，故系统不会发生死锁。

情况④会发生死锁：已知系统资源 R 的数目等于 5,进程数等于 3,每个进程对 R 资源的最大需求为 3。若系统为 3 个进程分别分配 2、2 和 1 个资源，系统可供分配的剩余资源数等于 0,则无法保证进程得到所需资源运行完毕，故系统会发生死锁。

情况⑤会发生死锁：已知系统资源 R 的数目等于 6,进程数等于 3,每个进程对 R 资源的最大需求为 3。若系统为 3 个进程各分配 2 个资源，系统可供分配的剩余资源数等于 0,则无法保证进程得到所需资源运行完毕，故系统会发生死锁。

情况⑥不会发生死锁：已知系统资源 R 的数目等于 6,进程数等于 4,每个进程对 R 资源的最大需求为 2。若系统为 4 个进程各分配 1 个资源，系统可供分配的剩余资源数等于 2，则可以保证 2 个进程得到所需资源运行完毕。当该进程释放资源后又能保证剩余 2 个进程运行完毕，故系统不会发生死锁。

某系统采用请求页式存储管理方案，假设某进程有 6 个页面，系统给该进程分配了 4 个存储块，其页面变换表如下表所示，表中的状态位等于 1/0 分别表示页面在内存/不在内存。当该进程访问的页面 2 不在内存时，应该淘汰表中页号为 (27)的页面。假定页面大小为 4K，逻辑地址为十六进制 3C18H，该地址经过变换后的页帧号为 (28)。

页　号	页 帧 号	状 态 位	访 问 位	修 改 位
0	5	1	1	1
1	—	0	0	0
2	—	0	0	0
3	2	1	1	0
4	8	1	1	1
5	12	1	0	0

(27)A.0　　B.3　　C.4　　D.5

(28)A.2　　B.5　　C.8　　D.12

【答案】D　A

【解析】本题考查操作系统存储管理方面的基础知识。

在请求页式存储管理方案中，当访问的页面不在内存时需要置换页面，置换页面的原则如下表，即最先置换访问位和修改位为 00 的页，其次是访问位和修改位为 01 的页， 然后是访问位和修改位为 10 的页，最后才置换访问位和修改位为 11 的页。因此本题当该进程访问的页面 2 不在内存时，应该淘汰表中页号为 5 的页面。

置 换 顺 序	访 问 位	修 改 位
1	0	0
2	0	1
3	1	0
4	1	1

由于 3C18H = 3000 + 0C18,因此该地址对应的页号为 3,根据页面变换表，经变换后的页帧号为 2。

为了有效地捕获系统需求，应采用 (29).

(29)A.瀑布模型　　B.V 模型　　C.原型模型　　D.螺旋模型

【答案】C

【解析】本题考查软件过程模型。

软件过程是软件生命周期中的一系列相关活动，即用于开发和维护软件及相关产品的一系列活动。软件过程模型可以帮助开发团队理解开发过程，形成对开发中的活动、资源和约束的共同理解，可以根据具体情况对一个过程进行裁剪等。瀑布模型从一种非常高层的角度描述了软件开发过程中进行的活动，并且提出了要求开发人员经过的事件序列。该模型适用于项目开始时需求已确定的情况。V 模型是瀑布模型的变种，它说明测试活动是如何与分析和设计相联系的。原型模型允许开发人员快速地构造整个系统或系统的一部分以理解或澄清问题。原型的用途是获知用户的真正需求，因此原型模型可以有效地引发系统需求。螺旋模型把开发活动和风险管理结合起来，以将风险减到最小并控制风险。

关于过程改进，以下叙述中不正确的是（30）。

(30)A.软件质量依赖于软件开发过程的质量，其中个人因素占主导作用

B.要使过程改进有效，需要制定过程改进目标

C.要使过程改进有效，需要进行培训

D.CMMI 成熟度模型是一种过程改进模型，仅支持阶段性过程改进而不支持连续性过程改进

【答案】D

【解析】本题考查软件过程改进。

软件开发过程极大地影响所生成的产品质量，因此改进过程将改进软件产品的质量。这也是进行过程改进的前提和理念。软件质量依赖于软件开发过程的质量，其中，人的因素是主导的，开发技术、过程质量、成本时间和进度也是影响因素。另外，要使得过程改进有效，需要制定过程改进的目标，还需要对开发人员进行培训。CMMI 是 SEI 将已有的几个 CMM 模型结合在一起，使之构成“集成模型”，即成熟度模型，该模型支持阶段性过程改进和连续性过程改进。

软件产品的可靠性并不取决于（31）。

(31)A.潜在错误的数量　　B.潜在错误的位置

C.软件产品的使用方式　　D.软件产品的开发方式

【答案】D

【解析】本题考查软件质量管理。

软件可靠性指的是一个系统对于给定的时间间隔内、在给定条件下无失效运作的概率。

根据定义，软件可靠性与软件的潜在错误的数量、位置有关，与软件产品的使用方式有关，而软件产品的开发方式不决定软件产品的可靠性。

软件（32）是指一个系统在给定时间间隔内和给定条件下无失效运行的概率。

(32)A.可靠性　　B.可用性　　C.可维护性　　D.可伸缩性

【答案】A

【解析】本题考查软件质量管理。

软件可靠性指的是一个系统对于给定的时间间隔内、在给定条件下无失效运作的概率。软件可用性使之在给定的时间点上，一个软件系统能够按照规格说明正确运行的概率。软件可维护性是在给定的使用条件下，在规定的时间间隔内，使用规定的过程和资源完成维护活动的概率。

高质量的文档所应具有的特性中，不包括（33）。

(33)A.针对性，文档编制应考虑读者对象群

B.精确性，文档的行文应该十分确切，不能出现多义性的描述

C.完整性，任何文档都应当是完整的、独立的，应该自成体系

D.无重复性，同一软件系统的几个文档之间应该没有相同的内容，若确实存在相同内容，则可以用“见**文档**节”的方式引用

【答案】D

【解析】本题考查文档与软件维护。

文档是指某种数据媒体和其中所记录的数据。在软件开发过程中，有大量的信息要记录和使用，因此文档具有重要的作用，如可以提高软件开发过程的能见度、提高开发效率、作为开发人员在一定阶段的工作成果和结束标志、记录开发过程中的有关信息、提高对软件运行维护和培训的有关信息、便于用户了解软件功能和性能等各项指标。

高质量的文档应该体现在几个方面：针对性，文档编制应考虑读者。按不同的类型、不同层次的读者，决定怎样适应他们的需要；精确性，文档的行文应该十分确切，不能出现多义性的描述。同一项目几个文档的内容应该是协调一致，没有矛盾的；清晰性， 文档编写应力求简明，如有可能，配以适当的图表，以增强其清晰性；完整性，任何文 档都应当是完整的、独立的，应该自成体系；灵活性，各个不同软件项目，其规模和复 杂程度有着许多实际差别，不能一律看待；可追溯性，由于各开发阶段编制的文档与各个阶段完成的工作

有密切的关系，前后两个阶段生成的文档，随着开发工作的逐步延伸， 具有一定的继承关系，在一个项目各开发阶段之间提供的文档必定存在着可追溯的关系。

在软件维护阶段，为软件的运行增加监控设施属于 (34) 维护。

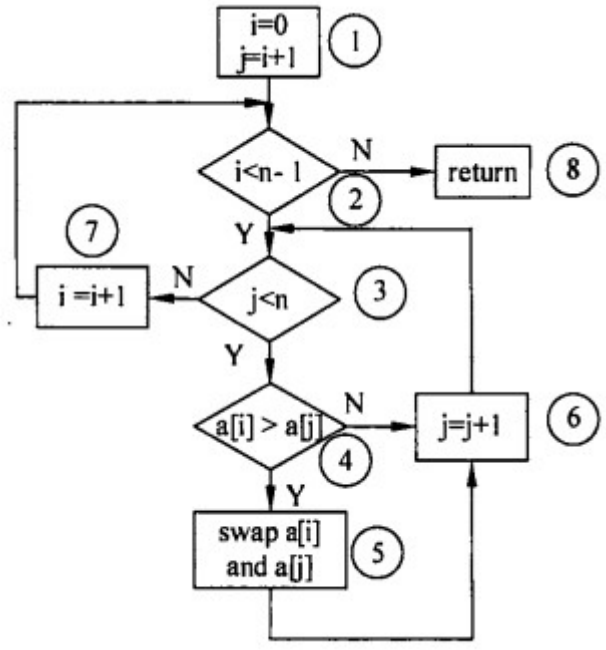
(34)A.改正性　　B.适应性　　C.完善性　　D.预防性

【答案】C

【解析】本题考查软件维护技术。

在软件开发完成交付用户使用后，就进入软件运行/维护阶段。软件维护活动根据其内容可以分为 4 种类型：改正性维护，为了识别和纠正软件错误、改正软件性能上的缺陷、排除实施中的误使用，应进行的诊断和改正错误的过程；适应性维护，由于信息技术飞速发展，软件运行的外部环境或数据环境可能会发生变化，为了使软件适应这种变 化，而修改软件的过程；完善性维护，在软件使用过程中，用户往往会对软件提出新的 功能与性能要求，为了满足这些要求，需要修改或再开发软件，以扩充软件功能、增强软件性能、改进加工效率、提高软件的可维护性而进行的维护活动；预防性维护是为了提高软件的可维护性和可靠性等，为以后进一步改进软件打下良好基础而进行的维护工作。

下图所示的逻辑流，最少需要 (35) 个测试用例可实现语句覆盖。



(35)A.1　　B.2　　C.3　　D.5

【答案】A

【解析】本题考查软件测试技术。

语句覆盖是一种白盒测试技术，指的是设计若干测试用例，运行被测程序，使得每一个可执行语句至少执行一次。题中的逻辑流的输入是一个数组 a，只要存在某个 a[i]>a[j] 的情况，则该测试用例下可以覆盖所有的可执行语句，因此至少需要 1 个测试用例即可。

在改正当前故障的同时可能会引入新的故障，这时需要进行 (36)。

(36)A.功能测试　　B.性能测试　　C.回归测试　　D.验收测试

【答案】C

【解析】本题考查软件测试技术。

功能测试检查软件是否能实现需求中指定的那些功能。性能测试是测试软件的安全性、精确性、速度和可靠性。回归测试用于识别在改正当前故障的同时可能会引入新的故障。验收测试是客户对系统进行测试以验证软件系统是否符合他们对需求的理解。

面向对象分析的第一步是 (37)。

(37)A.定义服务　　B.确定附加的系统约束

C.确定问题域　　D.定义类和对象

【答案】C

【解析】本题考查面向对象分析的基本知识。

面向对象分析的目的是为了获得对应用问题的理解，确定系统的功能、性能要求。面向对象分析包含 5 个活动：认定对象、组织对象、描述对象间的相互作用、定义对象的操作和定义对象的内部信息。而分析阶段最重要的是理解问题域的概念，其结果将影响整个工作。经验表明，从应用定义域概念标识对象是非常合理的。因此，面向对象分析的第一步就是确定问题域。

下列关于一个类的静态成员的描述中，不正确的是 (38)。

(38)A.类的静态方法只能访问该类的静态数据成员

B.静态数据成员可被该类的所有方法访问

C.该类的对象共享其静态数据成员的值

D.该类的静态数据成员的值不可修改

【答案】D

【解析】本题考查面向对象开发中静态成员的基本知识。

面向对象开发方法中，静态成员的含义是所修饰的成员是属于类的，而不是属于某对象的。静态数据成员对该类只有一份，该类的所有对象共享静态数据成员，可被该类的所有方法访问，其值可以修改，但是不论是通过对象还是类对静态数据成员值的修改， 都会反应到整个类。类的静态方法只能访问该类的静态数据成员。

UML 的设计视图包含了类、接口和协作，其中，设计视图的静态方面由 (39)和 (40)表现:动态方面由交互图、(41)表现。

(39)A.类图　　B.状态图　　C.活动图　　D.序列图

(40)A.交互图　　B.对象图　　C.通信图　　D.定时图

(41)A.状态图和类图　B.类图和活动图　　C.对象图和状态图　D.状态图和活动图

【答案】A　B　D

【解析】本题考查统一建模语言（UML）的基本知识。

通常是用一组视图反映系统的各个方面，以完整地描述系统，每个视图代表系统描述中的一个抽象，显示系统中的一个特定的方面。UML2.0 中提供了多种图形，从静态和动态两个方面表现系统视图。

类图展现了一组对象、接口、协作和它们之间的关系。对象图展现了一组对象以及其之间的关系，描述了在类图中所建立的事物的实例的静态快照。序列图是场景的图形化表示，描述了以时间顺序组织的对象之间的交互活动。通信图和序列图同构，强调收发消息的对象的结构组织。状态图展现了一个状态机，由状态、转换、事件和活动组成， 它关注系统的动态视图，强调对象行为的事件顺序。活动图是一种特殊的状态图，展现了在系统内从一个活动到另一个活动的流程，它专注于系统的动态视图。序列图、通信图、交互图和定时图均被称为交互图，它们用于对系统的动态方面进行建模。

UML 中关联的多重度是指 （42）。

(42)A.一个类中被另一个类调用的方法个数

B.一个类的某个方法被另一个类调用的次数

C.一个类的实例能够与另一个类的多少个实例相关联

D.两个类所具有的相同的方法和属性

【答案】C

【解析】本题考查面向对象开发的基本知识。

进行面向对象设计时，类图中可以展现类之间的关联关系，还可以在类图中图示关联中的数量关系，即多重度。表示数量关系时，用多重度说明数量或数量范围，表示有多少个实例（对象）能被连接起来，即一个类的实例能够与另一个类的多少个实例相关联。

在面向对象软件开发过程中，采用设计模式（43），

(43)A.以复用成功的设计

B.以保证程序的运行速度达到最优值

C.以减少设计过程创建的类的个数

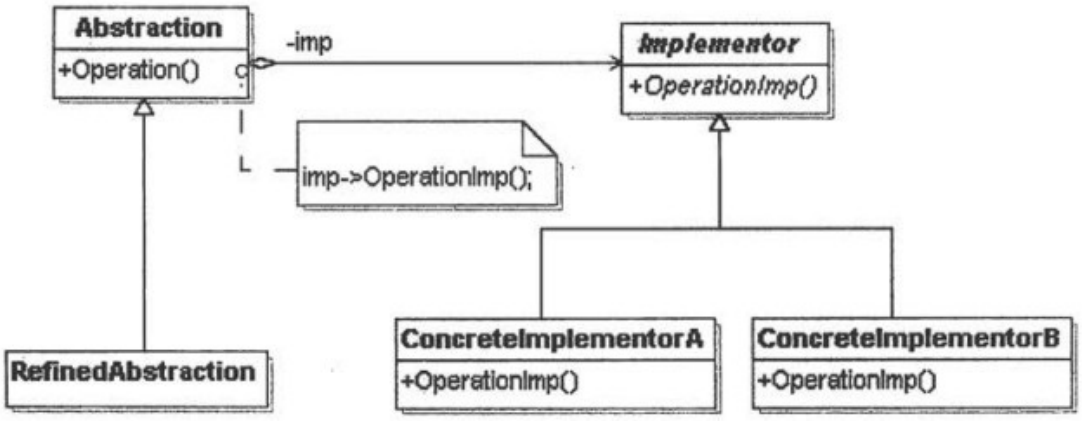
D.允许在非面向对象程序设计语言中使用面向对象的概念

【答案】A

【解析】本题考查设计模式的基本知识。

每一个设计模式描述了一个在我们周围不断重复发生的问题，以及该问题的解决方案的核心。这样，就能重复地使用该方案而不必做重复劳动。设计模式的核心在于提供了相关问题的解决方案。因此，面向对象软件开发过程中，采用设计模式的主要目的就是复用成功的设计。

设计模式（44）将抽象部分与其实现部分相分离，使它们都可以独立地变化。下图为该设计模式的类图，其中，（45）用于定义实现部分的接口。



(44)A.Bridge（桥接）B.Composite（组合） C.Facade（外观）D.Singleton（单例）

(45)A.Abstraction B.ConcretelxnplementorA

C.ConcretelmplementorB　　D.Implementor

【答案】A　D

【解析】

Bridge（桥接）模式将对象的抽象和其实现分离，从而可以独立地改变它们，抽象类定义对该抽象的接口，如上图中 Implementor，而具体的子类则用不同方式加以实现，如 ConcretelmplementorA 和 ConcretelmplementorB。Composite（组合）模式是结构型对象模式的一个实例。它描述了如何构造一个类层次式结构，这一结构由两种类型的对象所对应的类构成，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。Facade（外观）模式则描述了如何用单个对象表示整个子系统。Singleton（单例）模式保证一个类只产生唯一的一个实例。

以下关于 Singleton（单例）模式的描述中，正确的是 (46)。

(46)A.它描述了只有一个方法的类的集合

B.它描述了只有一个属性的类的集合

C.它能够保证一个类的方法只能被一个唯一的类调用

D.它能够保证一个类只产生唯一的一个实例

【答案】D

【解析】本题考查设计模式的基本知识。

例如，通常用户可以对应用系统进行配置，并将配置信息保存在配置文件中，应用系统启动时首先加载配置文件，而这一配置信息在内存中仅有一份。为了保证这一配置实例只有一份，采用 Singleton（单例）模式，以保证一个类只产生唯一的一个实例。

(47)将一个类的接口转换成客户希望的另外一个接口，使得原本由于接口不兼容而不能一起工作的那些类可以一起工作。

(47)A.Adapter（适配器）模式　　B.Command（命令）模式

C.Singleton（单例）模式　　D.Strategy（策略）模式

【答案】A

【解析】本题考查设计模式的基本知识。

Adapter 模式是将类的接口转换成客户希望的另外一个接口，使得原本由于接口不兼容而不能一起工作的那些类可以一起工作。Command 模式将请求封装在对象中，这样它就可作

为参数来传递，也可以被存储在历史列表里，或者以其他方式使用。Singleton（单例）模式保证一个类只产生唯一的一个实例。策略模式（Strategy)定义一系列的算法，把它们一个个封装起来,并使它们可以相互替换,这一模式使得算法可以独立于使用它的客户而变化。

以下关于高级程序设计语言翻译的叙述中，正确的是（48）.

(48)A.可以先进行语法分析，再进行词法分析

B.在语法分析阶段可以发现程序中的所有错误

C.语义分析阶段的工作与目标机器的体系结构密切相关

D.目标代码生成阶段的工作与目标机器的体系结构密切相关

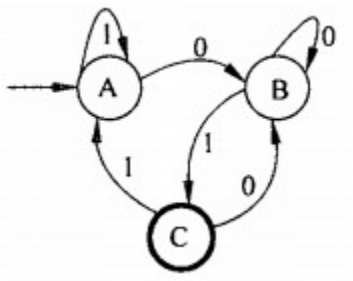
【答案】D

【解析】本题考查程序语言处理的基础知识。

将高级语言程序翻译为机器语言程序的过程中，需要依次进行词法分析、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等阶段，其中，中间代码生成和代码优化可以省略。程序中的错误分为语法错误和语义错误，语法分析阶段不能发现语义错误。

语义分析阶段主要处理语法正确的语言结构的含义信息,可以与目标机器的体系结构无关。目标代码生成阶段的工作与目标机器的体系结构是密切相关的。

下图所示为一个有限自动机(其中，A 是初态、C 是终态)，该自动机可识别(49)。



(49)A.0000　　B.1111　　C.0101　　D.1010

【答案】C

【解析】本题考查程序语言处理的基础知识。

从有限自动机的初态到终态的路径上的标记形成其可识别的字符串。

对于题中的自动机，0000 的识别路径为 A—B—B—B—B，不能到达终态 C，所以 0000 不能被该自动机识别；1111 的识别路径为 A—A—A—A—A，不能到达终态 C，所以 1111 也不能被该自动机识别；1010 的识别路径为 A—A—B—C—B,结束状态不是终态 C，所以 1010

不能被该自动机识别；0101 的识别路径为 A—B—C—B—C，存在从初态到终态的识别路径，所以 0101 可以被该自动机识别。

传值与传地址是函数调用时常采用的信息传递方式，(50)。

(50)A.在传值方式下，是将形参的值传给实参

B.在传值方式下，形参可以是任意形式的表达式

C.在传地址方式下，是将实参的地址传给形参

D.在传地址方式下，实参可以是任意形式的表达式

【答案】C

【解析】本题考查程序语言的基础知识。

一个函数被调用时，可能需要接受从外部传入的数据信息，传值调用与引用调用（传地址）是函数调用时常采用的信息传递方式。传值调用是将实参的值传给被调用函数的形参，引用调用的实质是将实参的地址传给被调用函数的形参。

某医院数据库的部分关系模式为：科室（科室号，科室名，负责人，电话)、病患 (病历号，姓名，住址，联系电话）和职工（职工号，职工姓名，科室号，住址，联系电 话)。假设每个科室有一位负责人和一部电话,每个科室有若干名职工,一名职工只属于一个科室；一个医生可以为多个病患看病；一个病患可以由多个医生多次诊治。

科室与职工的所属联系类型为 (51)，病患与医生的就诊联系类型为 (52)。对于就诊联系最合理的设计是 (53),就诊关系的主键是 (54)。

(51)A.1:1　　B.1:n　　C.n:1　　D.n:m

(52)A.1:1　　B.1:n　　C.n:1　　D.n:m

(53)A.就诊（病历号，职工号，就诊情况）

B.就诊（病历号，职工姓名，就诊情况）

C.就诊（病历号，职工号，就诊时间，就诊情况）

D.就诊（病历号，职工姓名，就诊时间，就诊情况）

(54)A.病历号，职工号　　B.病历号，职工号，就诊时间

C.病历号，职工姓名　　D.病历号，职工姓名，就诊时间

【答案】B　D　C　B

【解析】本题考查数据库基本概念、数据库设计的基础知识。

试题（51)、(52)考查数据库联系类型方面的基本概念。根据题意，“每个科室有若干名职工，一名职工只属于一个科室”，因此科室和职工的所属联系类型是 1:n，由“一个医生可以为多个病患看病；一个病患可以由多个医生多次诊治”，得知病患和医生的就 诊联系类型是 n:m。

试题（53)、(54)考查数据库设计方面的基础知识。就诊联系是多对多联系，对于多对多联系只能转换成一个独立的关系模式，关系模式的名称取联系的名称，关系模式的属性取该联系所关联的两个多方实体的码及联系的属性，关系的码是多方实体的码构成的属性组。另外，由于病患会找多个医生为其诊治，因此就诊关系模式设计时需要加上就诊时间，以便唯一区分就诊关系中的每一个元组,即就诊关系模式的主键为(病历号,职工号,就诊时间)。

给定关系模式 R<U，F>，U = {A,B,C},F={AB→C,C→B}。关系 R(55)且分别有_。

(55)A.只有 1 个候选关键字 AC　　B.只有 1 个候选关键字 AB

C.有 2 个候选关键字 AC 和 BC　　D.有 2 个候选关键字 AC 和 AB

(56)A.1 个非主属性和 2 个主属性　　B.2 个非主属性和 1 个主属性

C.0 个非主属性和 3 个主属性　　D.3 个非主属性和 0 个主属性

【答案】D　C

【解析】本题考查关系数据库规范化理论方面的基础知识。

试题（55)的正确答案是 D。根据函数依赖定义，可知 AC→U，AB→U，所以 AC 和 AB 为候选关键字。

(56) 根据主属性的定义，"包含在任何一个候选码中的属性叫做主属性（Prime attribute),否则叫做非主属性（Nonprime attribute)",所以，关系中的 3 个属性都是主属性。

设下三角矩阵(上三角部分的元素值都为 0)A[0..n,0..n]如下所示，将该三角矩阵的所有非零元素（即行下标不小于列下标的元素）按行优先压缩存储在容量足够大的数组 M[ ]中（下标从 1 开始)，则元素 a[i,j](0≤i≤n,j≤i)存储在数组 M 的 (57)中。

$$\begin{bmatrix} A_{0,0} & & & & & \\ A_{1,0} & A_{1,1} & & & 0 & \\ \vdots & & \ddots & & & \\ A_{7,0} & A_{7,1} & A_{7,2} & \cdots & A_{7,7} & \\ A_{8,0} & A_{8,1} & A_{8,2} & A_{8,3} & \cdots & A_{8,8} \end{bmatrix}$$

(57)A. $M\left[\frac{i(i+1)}{2}+j+1\right]$ B. $M\left[\frac{i(i+1)}{2}+j\right]$

C. $M\left[\frac{i(i-1)}{2}+j\right]$ D. $M\left[\frac{i(i-1)}{2}+j+1\right]$

【答案】A

【解析】本题考查数组存储的基础知识。

按行方式存储时，元素 A[i，j]之前的元素个数为(1+2+…+i+j)，由于数组 M 的下标从 1 开始，因此，存储 A[i，j]的是 M[1+2+…+i+j+1]，即 $M\left[\frac{i(i+1)}{2}+j+1\right]$

对 n 个元素的有序表 A[1..n]进行顺序查找，其成功查找的平均查找长度(即在查找表中找到指定关键码的元素时，所进行比较的表中元素个数的期望值)为 (58)。

(58)A.n B.(n+ 1)/2 C.log2 n D.n2

【答案】B

【解析】本题考查顺序查找方法。

假设从前往后找，则所找元素为第 1 个元素时，与表中的 1 个元素作了比较，所找元素为第 2 个元素时，与表中的 2 个元素作了比较，……，所找元素为第 n 个元素时， 与表中的 n 个元素作了比较，因此，平均查找长度等于(1+2+…+n)/n。

在 (59)中,任意一个结点的左、右子树的高度之差的绝对值不超过 1。

(59)A.完全二叉树 B.二叉排序树 C.线索二叉树 D.最优二叉树

【答案】A

【解析】本题考查二叉树的基本概念。

在平衡二叉树中，任意一个结点的左、右子树的高度之差的绝对值不超过 1。

虽然在结构上都符合二叉树的定义，但完全二叉树、线索二叉树、二叉排序树与最优二叉树的应用场合和概念都不同。

线索二叉树与二叉树的遍历运算相关，是一种存储结构。

二叉排序树的结构与给定的初始关键码序列相关。

最优二叉树（即哈夫曼树）是一类带权路径长度最短的二叉树，由给定的一个权值序列构造。

线索二叉树、二叉排序树和最优二叉树在结构上都不要求是平衡二叉树。

在完全二叉树中，去掉最后一层后就是满二叉树，而且最后一层上的叶子结点必须从该层的最左边开始排列，满足任意一个结点的左、右子树的高度之差的绝对值不超过 1 的条件，因此在形态上是一个平衡的二叉树。

设一个包含 N 个顶点、E 条边的简单无向图采用邻接矩阵存储结构(矩阵元素 A[i][j]等于 1/0 分别表示顶点 i 与顶点 j 之间有/无边),则该矩阵中的非零元素数目为 （60） 。

(60)A.N　　B.E　　C.2E　　D.N+E

【答案】C

【解析】本题考查数据结构的基础知识。

无向图的邻接矩阵是一个对称矩阵，每条边会表示两次，因此矩阵中的非零元素数目为 2E。

对于关键字序列（26,25,72,38,8,18,59)，采用散列函数 H(Key)=Key mod 13 构造散列表（哈希表)。若采用线性探测的开放定址法解决冲突（顺序地探查可用存储单元)， 则关键字 59 所在散列表中的地址为 (61)。

(61)A.6　　B.7　　C.8　　D.9

【答案】D

【解析】本题考查散列表的基本概念。

对于关键字序列（26,25,72,38,8,18,59)和散列函数 H(Key)=Key mod 13,采用线性探测的开放定址法解决冲突构造的散列表如下表所示：

0	1	2	3	4	5	6	7	8	9	10	11	12
26	38				18		72	8	59			25

要在 8X8 的棋盘上摆放 8 个“皇后”，要求“皇后”之间不能发生冲突，即任何两 个“皇后”不能在同一行、同一列和相同的对角线上，则一般采用 (62)来实现。

(62)A.分治法　　B.动态规划法　　C.贪心法　　D.回溯法

【答案】D

【解析】本题考查算法设计技术。

N-皇后问题是一个经典的计算问题，该问题基于一些约束条件来求问题的可行解。该问题不易划分为子问题求解，因此分治法不适用：由于不是要求最优解，因此不具备最优子结构性质，也不宜用动态规划法和贪心法求解。而系统搜索法--回溯法可以有效地求解该问题。

分治算法设计技术 (63)。

(63)A.一般由三个步骤组成：问题划分、递归求解、合并解

B.一定是用递归技术来实现

C.将问题划分为 k 个规模相等的子问题

D.划分代价很小而合并代价很大

【答案】A

【解析】本题考查算法设计技术。

分治方法是一种重要的算法设计技术(设计策略)，该策略将原问题划分成 n 个规模较小而结构与原问题相似的子问题；递归地解决这些子问题；然后再合并其结果，最终得到原问题的解。分治算法往往用递归技术来实现，但并非必须。分治算法最理想的情况是划分为 k 个规模相等的子问题，但很多时候往往不能均匀地划分子问题。分治算法的代价在划分子问题和合并子问题的解上，根据不同的问题，划分的代价和合并的代价有所不同。例如归并排序中，主要的计算代价在合并解上，而在快速排序中，主要的计算代价在划分子问题上。

某算法的时间复杂度可用递归式= $T(n)=\begin{cases}\Theta(1) & ,n=1\\ 6T(n/5)+n & ,n>1\end{cases}$ 表示，若用表示，则正确的是（64）。

(64)A. $\Theta(n^{\log_5^6})$　　B. $\Theta(n^2)$　　C. $\Theta(n)$　　D. $\Theta(n^{\log_6^5})$

【答案】A

【解析】

本题考查算法分析技术。

用主定理可以很容易算出该递归式。主定理给出了求解形如$T(n)=aT(n/b)+f(n)$的递归式的一般方法。比较$n^{\log_b^a}$和$f(n)$中n的最高次幂的关系，考虑三种情况：若存在某常数$\varepsilon>0$，有$f(n)=O(n^{\log_b^a-\varepsilon})$，则$T(n)=\Theta(n^{\log_b^a})$；若$f(n)=O(n^{\log_b^a}\lg^k n)$，则$T(n)=\Theta(n^{\log_b^a}\lg^{k+1} n)$；若$f(n)=O(n^{\log_b^a+\varepsilon})$，则$T(n)=\Theta(f(n))$。题中，$a$=6，$b$=5，属于第一种情况，因此有$T(n)=\Theta(n^{\log_b^a})=\Theta(n^{\log_5^6})$。

用插入排序和归并排序算法对数组<3，1，4，1，5，9，6，5>进行从小到大排序，则分别需要 进行 (65)次数组元素之间的比较。

(65)A.12,14　　B.10,14　　C.12,16　　D.10,16

【答案】A

【解析】本题考查排序算法。

插入排序算法的基本思想是将待排序数组分为两个部分，已排好序部分和未排序部分。其主要步骤为：开始时，第一个元素在已排好序部分中，其余元素在未排序部分。 然后依次从未排序部分中取出第一个元素，从后向前与排好序部分的元素进行比较并将其插入到已排好序部分的正确位置。直到所有元素排好序。

归并排序的基本思想是将待排序数组划分为子问题，对子问题求解，然后合并解。其主要步骤为：将数组分为两个相同规模的子数组，分别包含前 n/2 个元素和后 n/2 个 元素；递归地排序这两个子数组；合并排好序的两个子数组，依次比较两个排好序的子数组的元素，得到整个数组的排好序的序列。

根据上述算法思想和算法步骤，可以得到题中实例的比较次数分别为 12 和 14。

ARP 协议属于 (66)协议，它的作用是 (67)。

(66)A.物理层　　B.数据链路层　　C.网络层　　D.传输层

(67)A.实现 MAC 地址与主机名之间的映射　　B.实现 IP 地址与 MAC 地址之间的变换

C.实现 IP 地址与端口号之间的映射　　D.实现应用进程与物理地址之间的变换

【答案】C　B

【解析】

ARP 是网络层协议，它的作用是实现 IP 地址与 MAC 地址之间的变换。IP 地址是分配给主机的逻辑地址，在互联网中表示唯一的主机。另外，每个主机还有一个物理地址，通常用网卡地址（MAC 地址）来表示主机的物理地址。

物理地址和逻辑地址的区别可以从两个角度看：从网络互连的角度看，逻辑地址在整个互连网络中有效，而物理地址只是在子网内部有效；从网络协议分层的角度看，逻辑地址由 Internet 层使用，而物理地址由子网访问子层（具体地说就是数据链路层）使用。

由于有两种主机地址，因而需要一种映像关系把这两种地址对应起来。在 Internet 中用地址分解协议（Address Resolution Protocol，ARP)来实现逻辑地址到物理地址映像。 ARP 分组的格式如下图所示。

<table>
<tr><td colspan="2">硬件类型</td><td>协议类型</td></tr>
<tr><td>硬件地址长度</td><td>协议地址长度</td><td>操作类型</td></tr>
<tr><td colspan="3">发送结点硬件地址</td></tr>
<tr><td colspan="3">发送结点协议地址</td></tr>
<tr><td colspan="3">目标结点硬件地址</td></tr>
<tr><td colspan="3">目标结点协议地址</td></tr>
</table>

各字段的含义解释如下：

•硬件类型：网络接口硬件的类型，对以太网此值为 1。

•协议类型：发送方使用的协议，0800H 表示 IP 协议。

•硬件地址长度：对以太网，地址长度为 6 字节。

•协议地址长度：对 IP 协议，地址长度为 4 字节。

•操作类型：1 一 ARP 请求，2—ARP 响应，3—RARP 请求，4—RARP 响应。

通常 Internet 应用程序把要发送的报文交给 IP 协议，IP 当然知道接收方的逻辑地址（否则就不能通信了），但不一定知道接收方的物理地址。在把 IP 分组向下传送给本地数据链路实体之前可以用两种方法得到目标物理地址：

①查本地内存中的 ARP 地址映像表，其逻辑结构如下表所示。可以看出这是 IP 地址和以太网地址的对照表。

②如果在 ARP 表中查不到，就广播一个 ARP 请求分组，这种分组经过路由器进一步转发，可

以到达所有连网的主机。它的含义是“如果你的 IP 地址是这个分组中的目标结点协议地址，请回答你的物理地址是什么”。收到该分组的主机一方面可以用分组中的 两个源地址更新自己的 ARP 地址映像表，另一方面用自己的 IP 地址与目标结点协议地址字段比较，若相符则发回一个 ARP 响应分组，向发送方报告自己的硬件地址，若不相符则不予回答。

IP 地址	以太网地址
130.130.87.1	08 00 39 00 29 D4
129.129.52.3	08 00 5A 21 17 22
192.192.30.5	08 00 10 99 A1 44

下面关于集线器与交换机的描述中，错误的是 (68)。

(68)A.交换机是一种多端口网桥.　　B.交换机的各个端口形成一个广播域

C.集线器的所有端口组成一个冲突域　　D.集线器可以起到自动寻址的作用

【答案】D

【解析】

集线器是一种物理层设备，它的作用是从一个端口接收信息，并向其他端口广播出去。集线器不解释所传送信息的含义，也不能识别任何协议数据单元。集线器的各个端口构成一个冲突域，即只能有一个端口发送数据，如果有两个以上端口同时发送，就冲突了。网桥是数据链路层设备，能识别数据链路层协议数据单元，并根据数据链路层地址进行数据转发。交换机是一种多端口网桥，任何一对端口之间都能进行数据转发。交换机的各个端口构成一个广播域，但不是冲突域，即可以有多个端口同时发送数据而不会出现冲突。

“三网合一”的三网是指 （69）。

(69)A.电信网、广播电视网、互联网　　B.物联网、广播电视网、电信网

C.物联网、广播电视网、互联网　　D.物联网、电信网、互联网

【答案】A

【解析】

“三网合一”是将电信网、广播电视网以及互联网进行整合，实现业务互联互通的一种网络解决方案。

要使 4 个连续的 C 类网络汇聚成一个超网，则子网掩码应该为（70）。

(70)A.255.240.0.0　B.255.255.0.0　C.255.255.252.0　D.255.255.255.252

【答案】C

【解析】

把 4 个 C 类网络汇聚成一个超网地址，使用的网络掩码为 255.255.252.0。

Ravi, like many project (71) , had studied the waterfall model of softwaredevelopment as the primary software life-cycle (72) . He was all set to use it for an upcoming project, his first assignment. However, Ravi found that the waterfall model couldnot be used because the customer wanted the software delivered in stages, something that implied that the system had to be delivered and built in (73) and not as (74) .The situation in many other projects is not very different. The real world rarely presents a problem in which a standard process, or the process used in a previous project, is the best choice. To be the most suitable, an existing process must be (75) to the new problem.A development process, even after tailoring, generally cannot handle change requests. To accommodate change requests without losing control of the project, you must supplement the development process with a requirement change management process.

(71)A.customers　B.managers　C.users　D.administrators

(72)A.activity　B.procedure　C.process　D.progress

(73)A.parts　B.modules　C.software　D.a whole

(74)A.parts　B.modules　C.software　D.a whole

(75)A.modified　B.used　C.suited　D.tailored

【答案】B　C　A　D　D

【解析】本题考查英语的基本知识。

和许多项目经理一样,Ravi 研究了作为主要软件开发生命周期过程的瀑布模型，但是，他发现瀑布模型不能满足要求，原因是客户希望软件分阶段提交。也就说明系统必须按照部分构建和交付系统，而不是作为一个整体进行。这种情况在很多其他项目中也类似。现实世界中,很难有一种标准的过程或在前期的项目中使用的过程作为目前项目的最佳选择。因此，为了达到最佳的适应性，需要针对新的问题，对已有开发过程进行裁剪（针对新的问题，做

适应性修改）。但是，即使经过裁剪，一个开发过程也很难应对变更的需求。因此，为了适应变化的需求而不失去对项目的控制，必须用需求变更管理过程对开发过程进行补充。

试题一

某医院欲开发病人监控系统。该系统通过各种设备监控病人的生命特征，并在生命特征异常时向医生和护理人员报替。该系统的主要功能如下：

(1) 本地监控：定期获取病人的生命特征，如体温、血压、心率等数据。

(2) 格式化生命特征：对病人的各项重要生命特征数据进行格式化，然后存入曰志文件并检查生命特征。

(3) 检查生命特征：将格式化后的生命特征与生命特征范围文件中预设的正常范围进行比较。如果超出了预设范围，系统就发送一条警告信息给医生和护理人员。

(4) 维护生命特征范围：医生在必要时（如，新的研究结果出现时）添加或更新生命特征值的正常范围。

(5) 提取报告：在医生或护理人员请求病人生命特征报告时，从日志文件中获取病人生命特征生成特征报告，并返回给请求者。

(6) 生成病历：根据日志文件中的生命特征，医生对病人的病情进行描述，形成病历存入病历文件。

(7) 查询病历：根据医生的病历查询请求，查询病历文件，给医生返回病历报告。

(8) 生成治疗意见：根据日志文件中的生命特征和病历，医生给出治疗意见，如处方等，并存入治疗意见文件。

(9) 查询治疗意见：医生和护理人员查询治疗意见，据此对病人进行治疗。

现采用结构化方法对病人监控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。

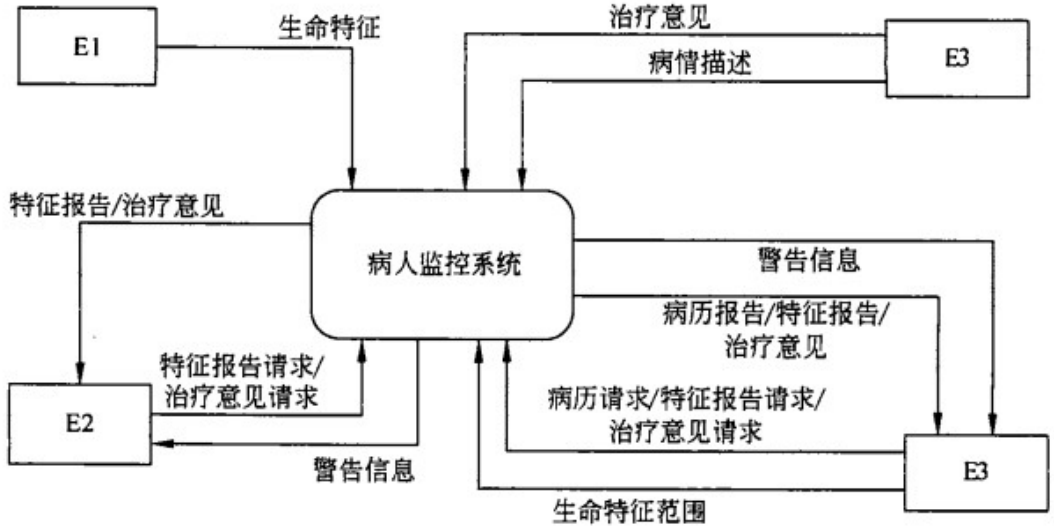


图 1-1　顶层数据流图

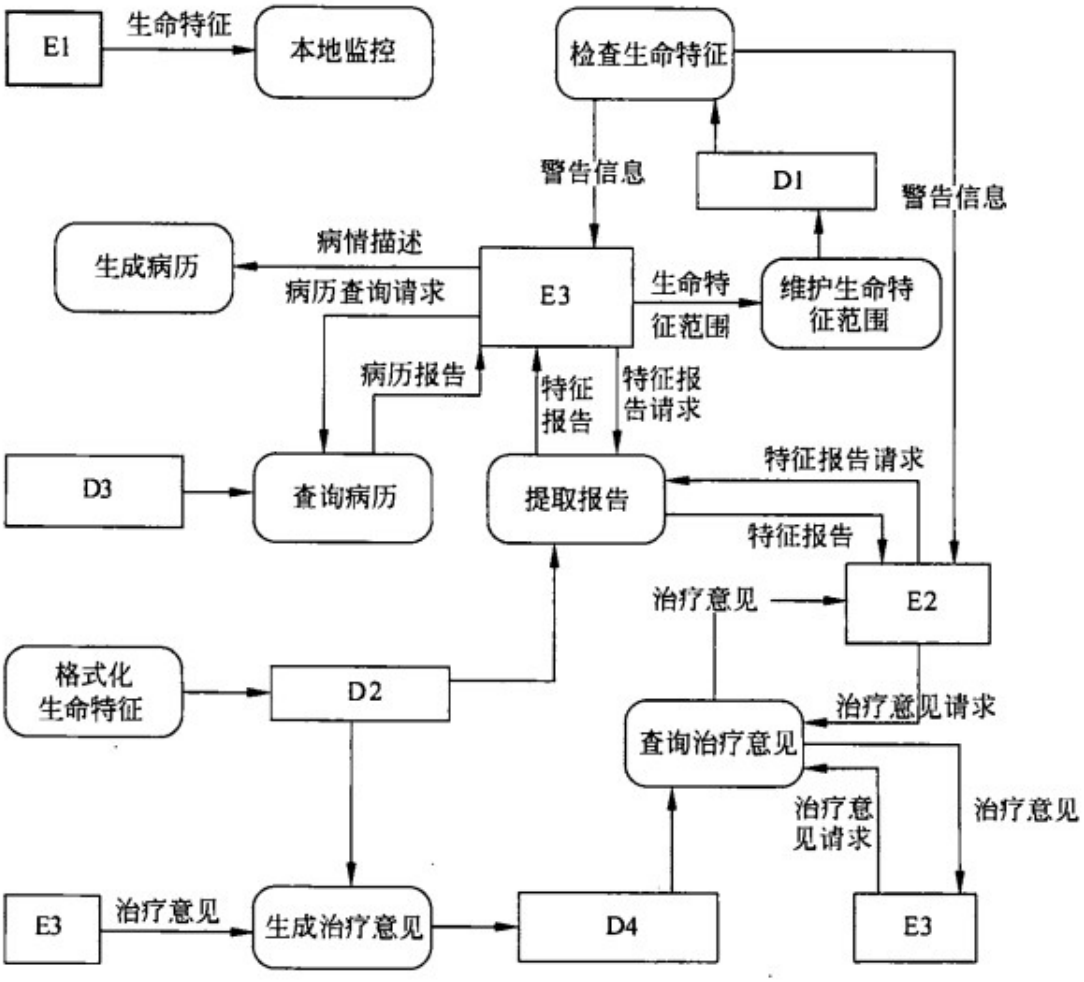


图 1-2　0 层数据流图

【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。

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。题目要求根据描述来确定图中的外部实体。分析题目中的描述，并结合已经在顶层数据流图中给出的数据流进行分析。从中可以看出，与系统的交互者包括病人、医生和护理人员。其中，本地监控定期获取病人的生命特征，病人是生命特征数据来源，医生和护理人员提出相关请求，并得到相关报告结果，如请求病人生命特征报告，并获得相关报告。医生还需要在必要时添加或更新生命特征范围。对应图 1-1 中数据流和实体的对应关系，可知 E1 为病人，E2 为护理人员，E3 为医生。

【问题 2】

使用说明中的词语，给出图 1-2 中的数据存储 D1～D4 的名称。

本问题考查 0 层 DFD 中数据存储的确定。根据说明中描述：(2)格式化生命特征： 对病人的各项重要生命特征数据进行格式化，然后存入日志文件并检查生命特征；(4) 维护生命特征范围：医生在必要时（如新的研究结果出现时）添加或更新生命特征值的正常范围；(6)生成病历：根据日志文件中的生命特征，医生对病人的病情进行描述，形成病历存入病历文件；(8)生成治疗意见：根据日志文件中的生命特征和病历，医生给出治疗意见，如处方等，并存入治疗意见文件。因此，D1 为生命特征范围文件，D2 为日志文件，D3 为病例文件，D4 为治疗意见文件。

【问题 3】

图 1-2 中缺失了 4 条数据流，使用说明、图 1-1 和图 1-2 中的术语，给出数据流的名称及其起点和终点。

数据流名称	起　点	终　点
重要生命特征	本地监控	格式化生命特征
格式化后的生命特征	格式化生命特征	检查生命特征
病例	生成病历	D3 或 病历（文件）
生命特征	D2 或 日志（文件）	生成病例

【问题 4】

说明实体 E1 和 E3 之间可否有数据流，并解释其原因。

本问题考查绘制 DFD 时的注意事项。在 DFD 中，每条数据流的起点和终点之一必须是加工(处理)。本题中，医生和护理人员根据查询到的治疗意见对病人进行治疗属于系统之外的行为，

所以两个实体之间不可以有数据流。

试题二

某服装销售公司拟开发一套服装采购管理系统，以方便对服装采购和库存进行管理。

【需求分析】

(1) 采购系统需要维护服装信息及服装在仓库中的存放情况。服装信息主要包括： 服装编码、服装描述、服装类型、销售价格、尺码和面料，其中，服装类型为销售分类，服装按销售分类编码。仓库信息包括：仓库编码、仓库位置、仓库容量和库管员。系统记录库管员的库管员编码、姓名和级别。一个库管员可以管理多个仓库，每个仓库有一名库管员。一个仓库中可以存放多类服装，一类服装可能存放在多个仓库中。

(2) 当库管员发现有一类或者多类服装缺货时，需要生成采购订单。一个采购订单可以包含多类服装。每类服装可由多个不同的供应商供应，但具有相同的服装编码。采购订单主要记录订单编码、订货日期和应到货日期，并详细记录所采购的每类服装的数量、采购价格和对应的多个供应商。

(3) 系统需记录每类服装的各个供应商信息和供应商生产服装的情况。供应商信息包括：供应商编码、供应商名称、地址、企业法人和联系电话。一个供应商可以供应多类服装，一类服装可由多个供应商供应。库管员根据入库时的服装质量情况，设定每个供应商所供应的每类服装的服装质量等级，作为后续采购服装时，选择供应商的参考标准。 .

【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。

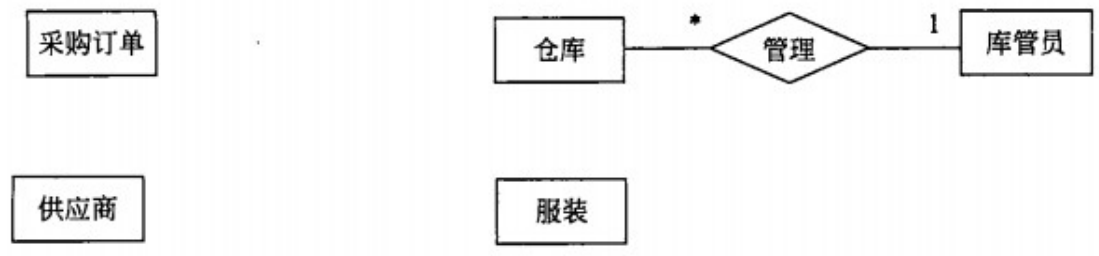


图 2-1 实体联系图

【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）：

库管员（库管员编码，姓名，级别）

仓库信息（(1)，仓库位置，仓库容量）

服装（服装编码，服装描述，服装类型，尺码，面料，销售价格）

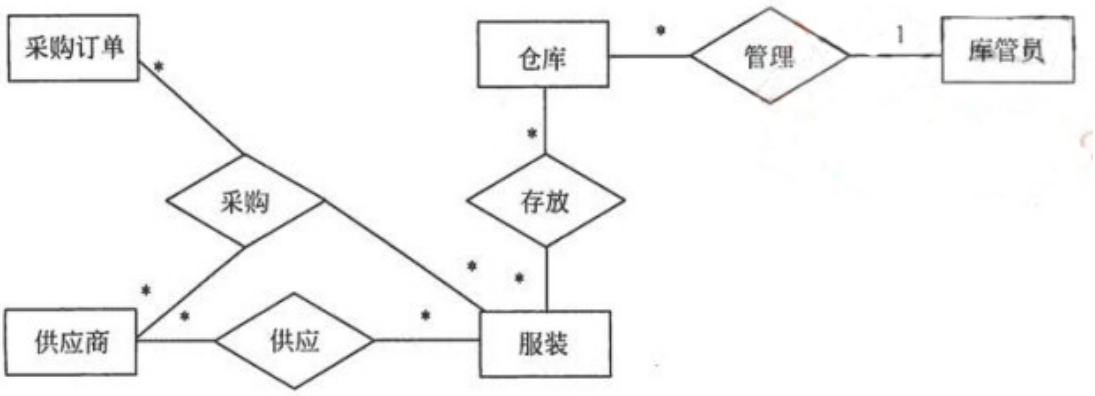
供应商(供应商编码，供应商名称，地址，联系电话，企业法人）

供应情况（(2)，服装质量等级）

采购订单（(3）采购订单明细（4））

【问题 1】

根据需求分析的描述，补充图 2-1 中的联系和联系的类型。



本问题考查数据库的概念结构设计，题目要求补充完整实体联系图中的联系和联系的类型。

根据题目的需求描述可知，一个库管员可以管理多个仓库，每个仓库有一名库管员。所以，仓库实体和库管员实体之间存在“管理”联系，联系的类型为多对一（*:1)。

根据题目的需求描述可知，一个仓库中可以存放多类服装，一类服装可能存放在多个仓库中。所以，仓库实体和服装实体之间存在“存放”联系，联系的类型为多对多（*:*)。

根据题目的需求描述可知，一个采购订单可以包含多类服装，每类服装可由多个不同的供应商供应。所以，采购订单实体与服装实体和供应商实体三者之间存在“采购” 联系，三者之间联系的类型为多对多对多（*:*:*)。

根据题目的需求描述可知，一个供应商可以供应多类服装，一类服装可由多个供应商供应。所以，供应商实体和服装实体之间存在“供应”联系，联系的类型为多对多（*:*)。

【问题 2】

根据补充完整的图 2-1，将逻辑结构设计阶段生成的关系模式中的空（1)～（4）补充完整，并给出其主键（用下划线指出)。

(1) 仓库编码，库管员编码

(2) 供应商编码，服装编码

(3) 订单编码，订货日期，应到货日期

(4) 订单编码，服装编码，供应商编码，数量，采购价格

本问题考查数据库的逻辑结构设计，题目要求补充完整各关系模式，并给出各关系模式的主键。

根据实体联系图和需求描述，系统记录库管员的库编码、姓名和级别。所以，对于“库管员”关系模式，需补充属性“库管员编码”。

根据实体联系图和需求描述，仓库信息主要包括：仓库编码、仓库位置、仓库容量和库管员。对于“仓库信息”关系模式，由于仓库实体与库管员实体有多对一联系，需记录对应的库管员，并且需补充属性——仓库编码。因此，“仓库信息，，关系模式，需补充属性“仓库编码”和“库管员编码”。

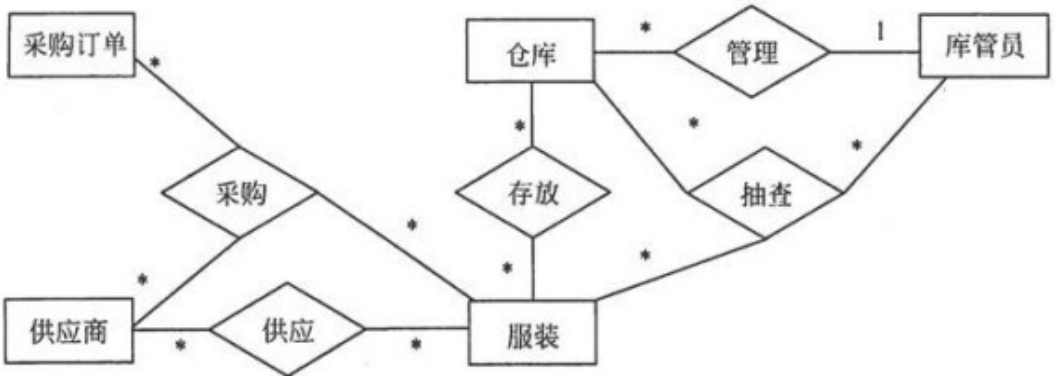
根据实体联系图和需求描述，供应商信息包括：供应商编码、供应商名称、地址、企业法人和联系电话。所以，对于“供应商”关系模式，需补充属性“供应商编码”。

根据实体联系图和需求描述，“供应情况”关系模式，需记录供应商和服装的多对多联系，即一个供应商可以供应多类服装，一类服装可由多个供应商供应。所以，对于“供应商”关系模式，需补充属性“供应商编码”和“服装编码”。

根据实体联系图和需求描述，采购订单主要记录订单编码、订货日期和应到货日期。所以，对于“采购订单”关系模式需补充属性：订单编码、订货日期和应到货日期。由于采购订单还需详细记录所采购的每类服装的数量、采购价格和对应的多个供应商。因此，“采购订单明细”关系模式，需记录采购订单实体与服装实体和供应商实体三者之间 存在的多对多对多联系。对于“采购订单明细”关系模式，需补充属性：订单编码、服装编码、供应商编码、数量和采购价格。

【问题 3】

如果库管员定期需要轮流对所有仓库中的服装质量进行抽查，对每个仓库中的每一类被抽查服装需要记录一条检查结果，并且需要记录抽查的时间和负责抽查的库管员。 请根据该要求，对图 2-1 进行修改，画出修改后的实体间联系和联系的类型。



本问题考查的是数据库的概念结构设计，根据新增的需求增加实体联系图中的实体的联系和联系的类型。

根据问题描述，多个库管员需对每个仓库中的每一类被抽查服装记录一条抽查结果。则在库管员实体与仓库实体和服装实体三者之间存在“抽查”联系，联系的类型是多对多对多(*:*:*)。

试题三

一个简单的图形编辑器提供给用户的基本操作包括：创建图形、创建元素、选择元素以及删除图形。图形编辑器的组成及其基本功能描述如下：

(1) 图形由文本元素和图元元素构成，图元元素包括线条、矩形和椭圆。

(2) 图形显示在工作空间中，一次只能显示一张图形（即当前图形，current)。

(3) 编辑器提供了两种操作图形的工具：选择工具和创建工具。对图形进行操作时，一次只能使用一种工具（即当前活动工具，active)。

①创建工具用于创建文本元素和图元元素。

②对于显示在工作空间中的图形，使用选择工具能够选定其中所包含的元素，可以选择一个元素，也可以同时选择多个元素。被选择的元素称为当前选中元素（selected).

③每种元素都具有对应的控制点。拖拽选定元素的控制点，可以移动元素或者调整元素的大小。

现采用面向对象方法开发该图形编辑器，使用 UML 进行建模。构建出的用例图和类图分别如图 3-1 和图 3-2 所示。

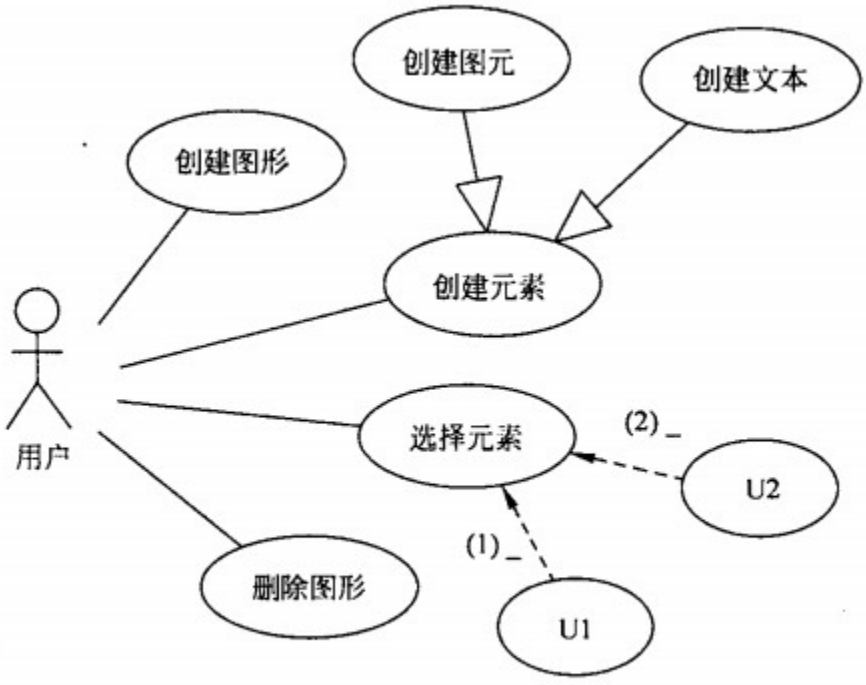


图 3-1　用例图

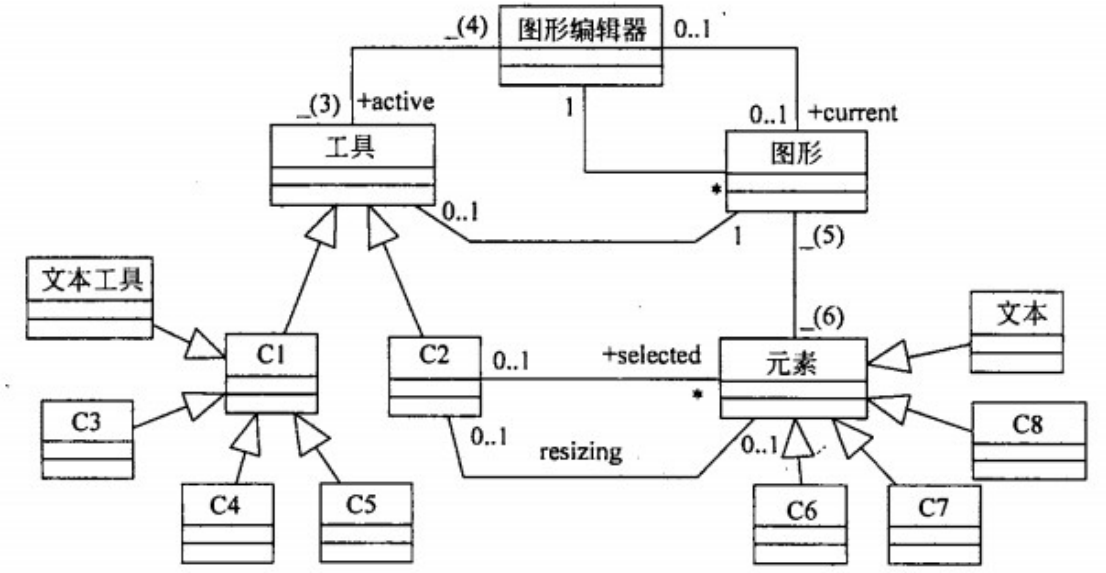


图 3-2 类图

【问题 1】

根据说明中的描述，给出图 3-1 中 U1 和 U2 所对应的用例，以及（1)和（2)处所对应的关系。

本问题主要考查用例之间的关系。在 UML 中，用例之间有 3 种关系：包含（include)、 概括（generalize）和扩展（extend)。

如果多个用例中都含有相同的事件流，那么可以将其抽取出来放在一个单独的用例中，其他用例都可以通过包含(include)这个用例来使用其中的事件流。包含关系可以避免在多个用例的描述中重复拷贝相同的事件流。

概括关系是指子用例（child use case)继承父用例（parent use case)的行为，而子用例本身还可以增加新的行为或重置父类的某些行为。这种关系与面向对象程序设计中的“继承”很类似。

一个用例（基础用例，base use case)中加入一些新的动作后则构成了另外一个用例（扩展用例，extending use case)，那么这两个用例之间的关系就是扩展关系。扩展关系与概括关系有相似之处，但是比概括关系更为严格。基础用例必须声明特定的扩展点，而扩展用例只能在这些扩展点上添加新行为。

由说明可知，图形编辑器的基本操作为创建图形、创建元素、选择元素和删除图形。对照图 3-1，可知这些最终都被确定为用例。除此之外，用例“创建图元”、“创建文本” 与用例“创建元素”之间是概括关系，即能创建的元素分别是图元和文本。图 3-1 中缺少了两个用例，而这两个用例都是与“选择元素”相关的。因此需要仔细阅读说明中关于“选择元素”的描

述，其中最关键的一句描述为“拖拽选定元素的控制点，可以移动元素或者调整元素的大小”。这句话中出现了两个动词短语“移动元素”、“调整元素大小”，这两个动作都是要先选择对应元素之后，才能实施的。因此，可以推出，U1 和 U2 应对应“移动元素”和“调整元素大小”。

下一步就是确定“移动元素”、“调整元素大小”与“选择元素”之间的关系。由说明可知，必须先选择元素才能通过拖拽控制点来对元素进行相应的操作。因此，“移动元素”和“调整元素大小”是对“选择元素”的扩展，因此这三个用例之间应该是扩展关系。（1）和（2）处应填写 extend。

【问题 2】

根据说明中的描述，给出图 3-2 中缺少的 C1 至 C8 所对应的类名以及（3)至（6）处所对应的多重度。

本问题考查类图，考点是类层次结构及多重度。图 3-2 中有两个非常明显的继承结构，需要考生将其填充完整。这两个继承结构的最顶层父类分别是“工具”和“元素”，这就需要仔细阅读说明中与这两个词汇相关的描述。说明中第一次出现“工具”这个词，是在句子“编辑器提供了两种操作图形的工具:选择工具和创建工具”。这是典型的一般/特殊关系的描述，由此可以推断出，C1 和 C2 应该对应“选择工具”和“创建工具”， 到底是怎样的对应关系，还需要进一步的细节信息。说明中的①给出“创建工具用于创建文本元素和图元元素”，而 C1 的一个子类就是“文本工具”，所以可以确定 C1 是“创建工具”，C2 是“选择工具”。那么 C3～C5 应该就是与创建图元元素相关的工具了，而图元分为三类：线条、矩形和椭圆。所以 C3～C5 分别对应“线条工具”、“矩形工具” 和“椭圆工具”。

现在图 3-2 中左边的继承结构已经填充完整了。右边的继承结构就可以对应地填写出来了，C6～C8 分别对应的是类“线条”、“矩形”和“椭圆”。

确定多重度时，需要在说明寻找关联两端的类相关的描述。“对图形进行操作时，一次只能使用一种工具（即当前活动工具，active)”，即在图形编辑器中一次只能使用一个工具，而任何一个工具只属于这个图形编辑器。所以（3)处应填 0..1，(4)处应填 1。

一个图形可以包含多个元素，对于一个图形中的特定元素来说，只能属于这个图形。所以(5)处应填 1，(6)处应填 1..*或*。

【问题 3】

图 3-2 中的类图设计采用了桥接（Bridge)设计模式，请说明该模式的内涵。

桥接模式将抽象部分与它的实现部分分离，使它们都可以独立地变化，对一个抽象的实现部分的修改应该对使用它的程序不产生影响。

本问题考查桥接模式，该模式将抽象部分与其实现部分分离，使它们都可以独立地变化。

在以下情况中可以使用 Bridge 模式：

(1) 不希望在抽象以及抽象的实现部分之间有一个固定的绑定关系。例如这种情况可能是因为，在程序运行时刻可以选择或切换实现部分。

(2) 类的抽象以及它的实现都应该可以通过生成子类的方法加以扩充，使用 Bridge 模式可以对不同的抽象接口和实现部分进行组合，并分别对它们进行扩充。

(3) 对一个抽象的实现部分的修改应该对用户不产生影响，即客户的代码不必重新编译。

试题四

某应用中需要对 100000 个整数元素进行排序，每个元素的取值在 0～5 之间。排序算法的基本思想是：对每一个元素 x，确定小于等于 x 的元素个数（记为 m），将 x 放在输出元素序列的第 m 个位置。对于元素值重复的情况，依次放入第 m-1、m-2、…个位置。例如，如果元素值小于等于 4 的元素个数有 10 个，其中元素值等于 4 的元素个数有 3 个，则 4 应该在输出元素序列的第 10 个位置、第 9 个位置和第 8 个位置上。

算法具体的步骤为：

步骤 1:统计每个元素值的个数。

步骤 2:统计小于等于每个元素值的个数。

步骤 3:将输入元素序列中的每个元素放入有序的输出元素序列。

【C 代码】

下面是该排序算法的 C 语言实现。

(1) 常量和变量说明

R：常量，定义元素取值范围中的取值个数，如上述应用中 R 值应取 6。

i：循环变量。

n：待排序元素个数。

a：输入数组，长度为 n。

b：输出数组，长度为 m

c：辅助数组，长度为 R,其中每个元素表示小于等于下标所对应的元素值的个数。

(2) 函数 sort

```
1 void sort(int n, int a[], int b[]) {
2    int c[R], i;
3    for(i = 0; i < ___(1)___; i++) {
4       c[i] = 0;
5    }
6    for(i = 0; i < n; i++) {
7        c[a[i]] = ___(2)___;
```

```
8     }
9     for(i = 1; i < R; i++) {
10        c[i] = ____(3)____;
11    }
12    for(i = 0; i < n; i++) {
13        b[c[a[i]] - 1] = ____(4)____;
14        c[a[i]] = c[a[i]] - 1;
15    }
16 }
```

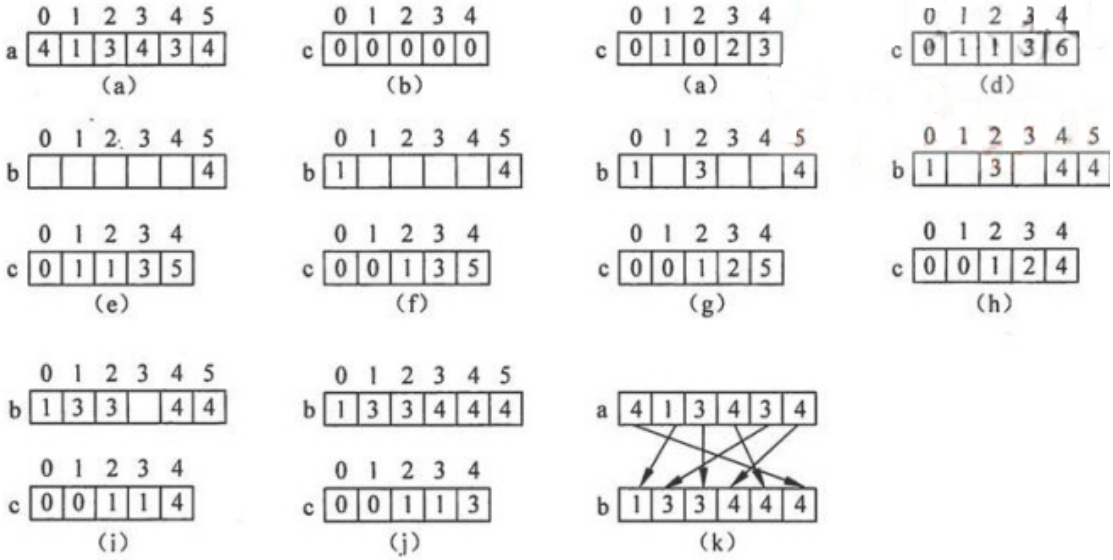
【问题 1】

根据说明和 C 代码，填充 C 代码中的空缺(1)～(4)。

根据题中说明，第 3 到第 5 行代码进行 c 数组的初始化，c 数组的长度为 R，在 C 语言中，下标从 0 开始，因此空格（1)中填写 R。第 6 到第 8 行检查 a 数组的每一个元素。如果元素的值为 i，则增加 c[i]的值。因此 c[a[i]] = c[a[i]] + 1，空格(2)填写 c[a[i]]+1。 完成第 6 行到第 8 行的代码后，c[i]中就存放了等于 i 的元素的个数。第 9 到第 11 行， 通过在数组 c 中记录计数和，c[i] = c[i - 1] + c[i],可以确定对每一个 i=0，1，…，R-1,有多 少个元素是小于或等于 i 的。因此空格(3)填写 c[i - 1] + c[i]o 第 12 行到第 15 行把数组 a 中每个元素 a[i]放在输出数组 b 中与其相应的最终位置上，b[c[a[i]] - 1] =a[i]，因 此空格(4)填写 a[i]。由于可能存在相同元素,因此每次将一个值 a[i]放入数组 b 中时，都要减小 c[i]的值。下面以一个例子来说明排序过程。

设 a= {4，1，3,4,3,4}，R = 5,即待排序的元素值在{0，1，2,3,4}中，其排序过程如下图所示。

图中(a)为输入数组 a,(b)为初始化后的 c 数组,(c)为统计数组 a 中每个元素 后的 c 数组，(d)为计数和，即统计小于等于 i 的元素个数后的 c 数组。(e)到(j)是 将 a 数组中的元素依次放到 b 数组的过程，(k)是数组 a 和数组 b 的元素对应关系。



【问题 2】

根据 C 代码，函数的时间复杂度和空间复杂度分别为(5)和(6)(用 O 符号表示)。

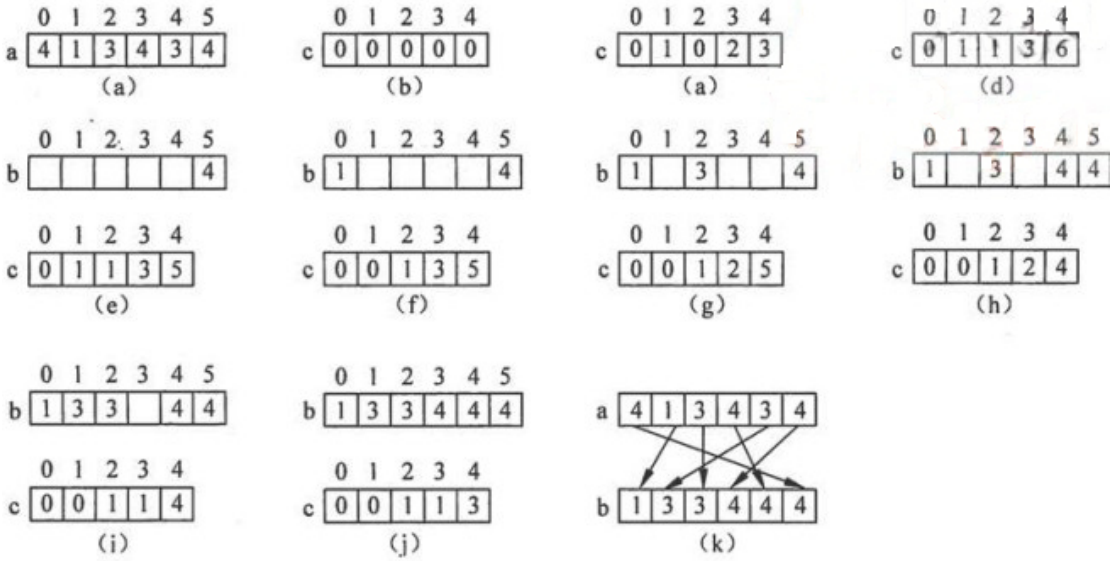
(5) $\Theta(n+R)$ 或者 $\Theta(n)$ 或 n 或线性

(6) $\Theta(n+R)$ 或者 $\Theta(n)$ 或 n 或线性

根据上述 C 代码，第 3 到第 5 行代码的 for 循环所花时间为 $\Theta(R)$。第 6 到第 8 行的 for 循环所花时间为 $\Theta(n)$。第 9 到第 11 行的 for 循环所花时间为 $\Theta(R)$。第 12 到第 15 行 for 循环所花时间为 $\Theta(n)$。因此整个算法的时间复杂度为 $\Theta(n+R)$。若 R 远小于 n 或者 $R=\Theta(n)$ 时，时间复杂度可以表示为 $\Theta(n)$。

【问题 3】

根据以上 C 代码，分析该排序算法是否稳定。若稳定，请简要说明（不超过 100 字）；若不稳定，请修改其中代码使其稳定（给出要修改的行号和修改后的代码）。



从图(k)可以看出，算法不稳定。算法不稳定的原因在于将数组 a 中元素放到数组 b 中时，

是从数组 a 的第一个元素开始，依次取出元素放到数组 b 中。这样，相.同的两个元素值，在数组 a 中的相对位置和在数组 b 中的相对位置正好相反。若从数组 a 的最后一个元素开始，依次向前取元素放到 b 数组中，可以保持相同元素的相对位置。因此将第 12 行的代码 for(i = 0; i< n;i++)改为 for(i = n-1;i>=0;i--),则排序算法是稳定的。

试题五

某饭店在不同的时段提供多种不同的餐饮，其菜单的结构图如图 5-1 所示。

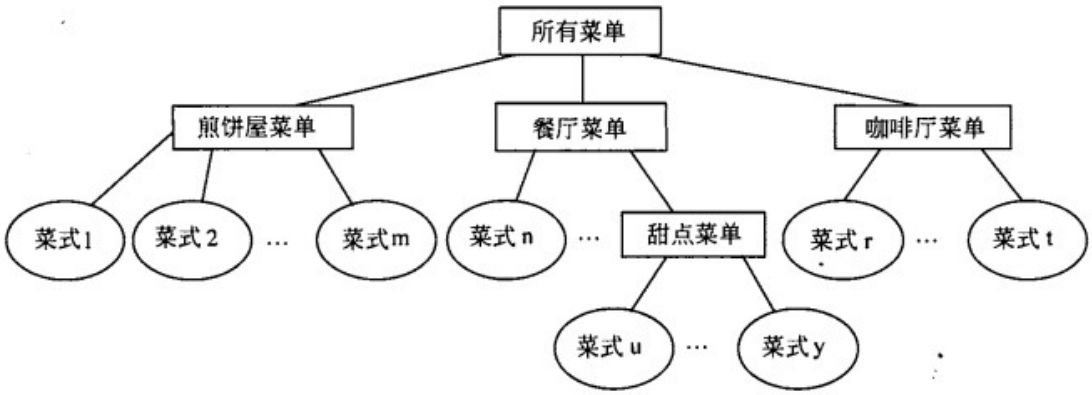


图 5-1　菜单结构图

现在采用组合(Composition)模式来构造该饭店的菜单，使得饭店可以方便地在其中增加新的餐饮形式，得到如图5-2所示的类图。其中MermCompcment为抽象类，定义了添加(add)新菜单和打印饭店所有菜单信息(print)的方法接口。类 Menu 表示饭店提供的每种餐饮形式的菜单，如煎饼屋菜单、咖啡屋菜单等。每种菜单中都可以添加子菜单，例如图 5-1 中的甜点菜单。类 Menultem 表示菜单中的菜式。

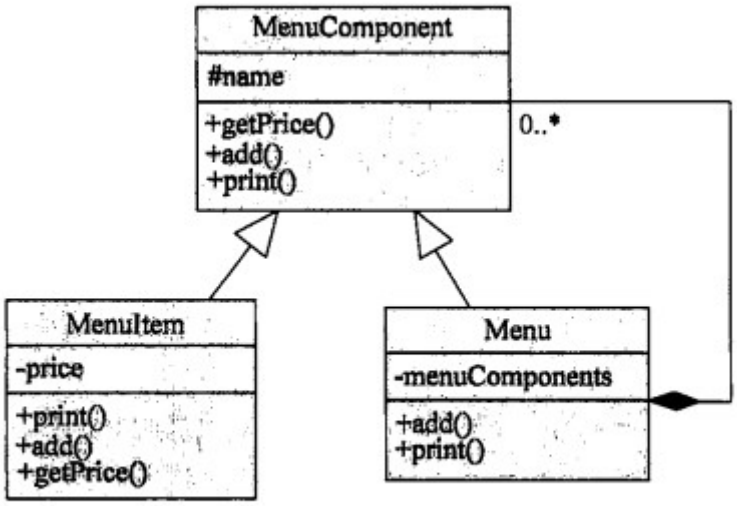


图 5-2　类图

【C++代码】

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

【问题 1】

```
protected:  string name;
public:
   MenuComponent(string name) {  this->name = name; }
   string getName() { return name; }
   __(1)__;                              // 添加新菜单
   virtual void print() = 0;             // 打印菜单信息
};
class MenuItem : public MenuComponent {
private:  double price;
public:
   MenuItem(string name, double price) : MenuComponent(name) { this->price
   = price; }
   double getPrice() { return price; }
   void add(MenuComponent* menuComponent) { return ; }  // 添加新菜单
   void print() { cout << "    " << getName() << ", " << getPrice() << endl; }
};
class Menu : public MenuComponent {
```

```
private:   list<__(2)__>  menuComponents;
public:
   Menu(string name): MenuComponent(name){}
   void add(MenuComponent* menuComponent)                    // 添加新菜单
   {     __(3)__;    }
   void print() {
     cout << "\n" << getName() << "\n----------------------------" << endl;
     std::list<MenuComponent *>::iterator iter;
     for(iter = menuComponents.begin(); iter != menuComponents.end(); iter++)
          __(4)__->print();
   }
};
void main() {
  MenuComponent* allMenus = new Menu("ALL MENUS");
  MenuComponent* dinerMenu = new Menu("DINER MENU");
  …… // 创建更多的 Menu 对象，此处代码省略
  allMenus->add(dinerMenu);              // 将 dinerMenu 添加到餐厅菜单中
  …… // 为餐厅增加更多的菜单，此处代码省略
  __(5)__->print();                      // 打印饭店所有菜单的信息
}
```

(1) virtual void add(MenuComponent* menuComponent) = 0

(2) MenuComponent *

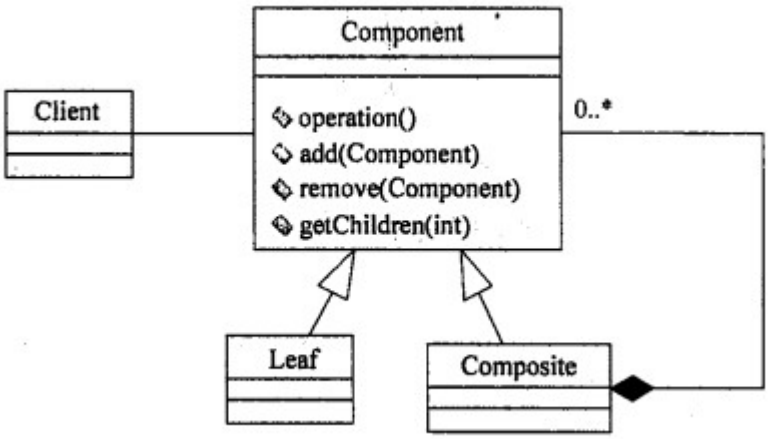
(3) menuComponents.push_back(menuComponent)

(4) (*iter)

(5) allMenus

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。

试题五将组合模式应用到饭店菜单的构造中。图 5-2 中的类 MenuComponent 对应上图中的 Component，Menultem 对应 Leaf，Menu 对应 Composite。在实现时，通常都会把 Component 定义为抽象类。

在 C++中，抽象类是指至少包含一个纯虚拟函数的类。类 MenuComponent 中已经包含了一个纯虚拟函数 print，所以 MenuComponent 已经是一个抽象类了。(1)处根据注释，这里应该定义功能为“添加新菜单”的成员函数。在子类 Menultem 和 Menu 中可以看到，都有 add 成员函数，说明子类中重置了父类中的成员函数。所以(1)处的成员函数也应该定义为纯虚拟函数，即 virtual void add(MenuComponent* menuComponent) = 0。

由图 5-2 可以看出，Menu 中包含了 MenuComponent 的对象集合。程序中用 C++标准模板库中的 list 来实现这个聚集关系。因此(2)处应填入 MenuComponent *。由于使用了 list，就可以利用 list 中提供的各种方法了。list 中用于添加元素的方法是 push_back，所以（3）处应填入 menuComponents.push_back(menuComponent)»

(4) 处出现在方法 print 中，其功能是打印出所有菜单的信息。这里使用了 list 中的迭代器类 iterator,遍历每个子菜单,并调用子菜单中定义的 print 方法打印该子菜单的 信息。

(4)处应填入*iter。为了能够在 main 中打印出所有的菜单信息，必须使用表示菜单结构中最顶层菜单的对象来调用 print，因此(5)处应填入 allMenus。

试题六

某饭店在不同的时段提供多种不同的餐饮，其菜单的结构图如图 6-1 所示。

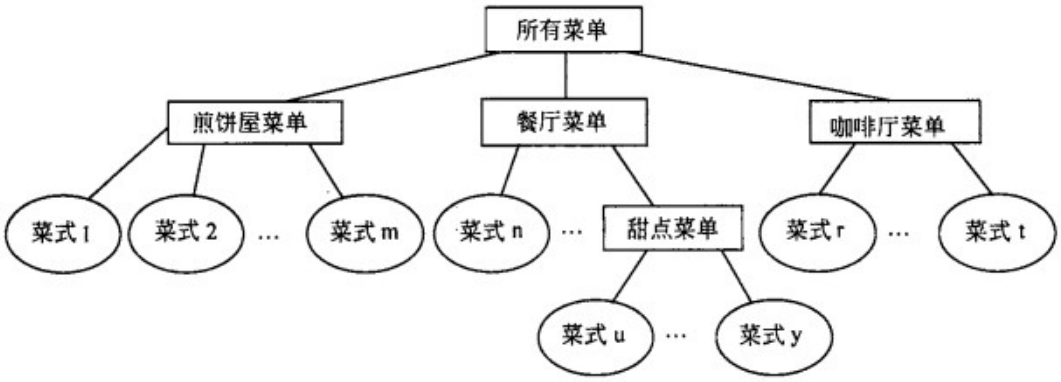


图 6-1　菜单结构图

现在采用组合（Composition)模式来构造该饭店的菜单，使得饭店可以方便地在其中增加新的餐饮形式，得到如图6-2所示的类图。其中MenuComponent为抽象类，定义了添加(add)新菜单和打印饭店所有菜单信息（print)的方法接口。类 Menu 表示饭店提供的每种餐饮形式的菜单，如煎饼屋菜单、咖啡屋菜单等。每种菜单中都可以添加子菜单，例如图 6-1 中的甜点菜单。类 Menultem 表示菜单中的菜式。

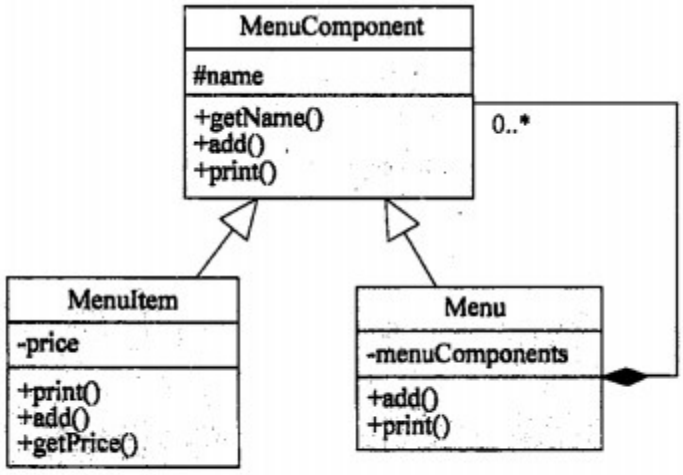


图 6-2　类图

【Java 代码】

```
import java.util.*;
__(1)__ MenuComponent {
   protected String name;
   __(2)__;  //添加新菜单
   public abstract void print();                          //打印菜单信息
   public String getName() { return name; }
}
class MenuItem extends MenuComponent {
   private double price;
   public MenuItem(String name, double price) {
       this.name = name;   this.price = price;
   }
   public double getPrice()  { return price; }
   public void add(MenuComponent menuComponent){ return ; } // 添加新菜单
   public void print() {
     System.out.print(" " + getName());
     System.out.println("," + getPrice());
   }
}
class Menu extends MenuComponent {
   private List<MenuComponent> menuComponents = new ArrayList<Menu-
   Component>();
   public Menu(String name) { this.name = name; }
   public void add(MenuComponent menuComponent) {            //添加新菜单
     menuComponents.__(3)__;
   }
```

```
    public void print() {
      System.out.print("\n" + getName());
      System.out.println("," + "----------------------");
      Iterator iterator = menuComponents.iterator();
      while(iterator.hasNext()) {
         MenuComponent menuComponent = (MenuComponent)iterator.next();
         __(4)__;
      }
    }
 }
 class MenuTestDrive {
    public static void main(String args[]) {
       MenuComponent  allMenus = new Menu("ALL MENUS");
 MenuComponent  dinerMenu = new Menu("DINER MENU");
       …… // 创建更多的 Menu 对象，此处代码省略
       allMenus.add(dinerMenu);    // 将 dinerMenu 添加到餐厅菜单中
       …… // 为餐厅增加更多的菜单，此处代码省略
       __(5)__;    // 打印饭店所有菜单的信息
    }
 }
```

(1) abstract class 或 public abstract class

(2) public abstract void add(MenuComponent menuComponent)

或 abstract void add(MenuComponent menuComponent)

或 protected abstract void add(MenuComponent menuComponent) ,

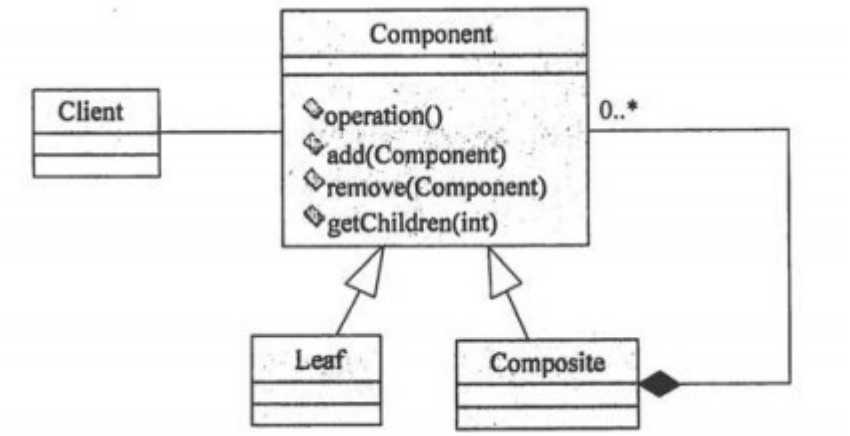
(3) add(menuComponent)

(4) menuComponent.print()

(5) allMenus.print()

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。

试题六将组合模式应用到饭店菜单的构造中。图 6-2 中的类 MenuComponent 对应上图中的 Component，Menultem 对应 Leaf，Menu 对应 Composite。在实现时，通常都会把 Component 定义为抽象类。

在 Java 中，用 abstract 关键字限定的类即为抽象类，所以（1）处应填入 abstract class。

(2)处根据注释，这里应该定义功能为“添加新菜单”的成员函数。在子类 Menultem 和 Menu 中可以看到，都有 add 成员函数，说明子类中重置了父类中的成员函数。所以(2) 处应填入 public abstract void add(MenuComponent menuComponent)。

由图 6-2 可以看出，Menu 中包含了 MenuComponent 的对象集合。程序中用 Java 中的 list 来实现这个聚集关系，这样就可以利用 list 中提供的各种方法了。list 中用于添加元素的方法是 add,所以（3）处应填入 add(menuComponent)。

(4) 处出现在方法 print 中，其功能是打印出所有菜单的信息。这里使用了 list 中的迭代器类 iterator,遍历每个子菜单，并调用子菜单中定义的 print 方法打印该子菜单的信息。（4）处应填入 menuComponent.print()。

为了能够在 main 中打印出所有的菜单信息，必须使用表示菜单结构中最顶层菜单的对象来调用 print，因此（5）处应填入 allMenus.print()。