

目 录

第一章 2013 年上半年软件设计师试题..... 2

    1.1 上午试题 ..... 2

    1.2 下午试题 ..... 9

第二章 2013 年下半年软件设计师试题..... 16

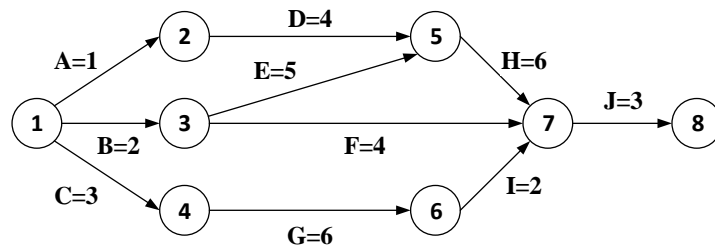
    2.1 上午试题 ..... 16

    2.2 下午试题 ..... 22

## 第一章 2013 年上半年软件设计师试题

### 1.1 上午试题

- 常用的虚拟存储器由 (1) 两级存储器组成。  
(1) A. 主存—辅存                      B. 主存—网盘                      C. Cache—主存                      D. Cache—硬盘
- 中断向量可提供 (2)。  
(2) A. FO 设备的端口地址              B. 所传送数据的起始地址              C. 中断服务程序的入口地址              D. 主程序的断点地址
- 为了便于实现多级中断嵌套, 使用 (3) 来保护断点和现场最有效  
(3) A. ROM                                  B. 中断向量表                                  C. 通用寄存器                                  D. 堆栈
- DMA 工作方式下, 在 (4) 之间建立了直接的数据通路。  
(4) A. CPU 与外设                                  B. CPU 与主存                                  C. 主存与外设                                  D. 外设与外设
- 地址编号从 S0000H 到 BFFFFH 且按字节编址的内存容量为 (5) KB, 若用 16Kx4bit 的存储器芯片构成该内存, 共需 (6) 片。  
(5) A. 128    B. 256    C. 512    D. 1024  
(6) A. 8    B. 16    C. 32    D. 64
- 利用报文摘要算法生成报文摘要的目的是 (7)。  
(7) A. 验证通信对方的身份, 防止假冒                      B. 对传输数据进行加密, 防止数据被窃听  
C. 防止发送方否认发送过的数据                      D. 防止发送的报文被篡改
- 防火墙通常分为内网、外网和 DMZ 三个区域, 按照受保护程度, 从高到低正确的排列次序为 (8)。  
(8) A. 内网、外网和 DMZ                      B. 外网、内网和 DMZ                      C. DMZ、内网和外网                      D. 内网、DMZ 和外网
- 近年来, 在我国出现的各类病毒中, (9) 病毒通过木马形式感染智能手机。  
(9) A. 欢乐时光                                  B. 熊猫烧香                                  C. X 卧底                                  D. CIH
- 王某是一名软件设计师, 按公司规定编写软件文档, 并上交公司存档。这些软件文档属于职务作品, 且 (10)。  
(10) A. 其著作权由公司享有                      B. 其著作权由软件设计师享有  
C. 除其署名权以外, 著作权的其他权利由软件设计师享有                      D. 其著作权由公司和软件设计师共同享有
- 甲经销商擅自复制并销售乙公司开发的 OA 软件光盘已构成侵权。丙企业在未知的情形下从甲经销商处购入 10 张并已安装使用。在丙企业知道了所使用的软件为侵权复制品的情形下, 以下说法正确的是 (11)。  
(11) A. 丙企业的使用行为侵权, 须承担赔偿责任  
B. 丙企业的使用行为不侵权, 可以继续使用这 10 张软件光盘  
C. 丙企业的使用行为侵权, 支付合理费用后可以继续使用这 10 张软件光盘  
D. 企业的使用行为不侵权, 不需承担任何法律责任
- 声音信号数字化过程中首先要进行 (12)。  
(12) A. 解码    B. D/A 转换    C. 编码    D. A/D 转换
- 以下关于 dpi 的叙述中, 正确的是 (13)。  
(13) A. 每英寸的 bit 数                      B. 存储每个像素所用的位数                      C. 每英寸像素点                      D. 显示屏上能够显示出的像素数目
- 媒体可以分为感觉媒体、表示媒体、表现媒体、存储媒体、传输媒体, (14) 属于表现媒体。  
(14) A. 打印机    B. 硬盘    C. 光缆    D. 图像
- 统一过程模型是一种“用例和风险驱动, 以架构为中心, 迭代并且增量”的开发过程, 定义了不同阶段及其制品, 其中精化阶段关注 (15)。  
(15) A. 项目的初创活动                                  B. 需求分析和架构演进  
C. 系统的构建, 产生实现模型                                  D. 软件提交方面的工作, 产生软件增量
- 在进行进度安排时, PERT 图不能清晰地描述 (16), 但可以给出哪些任务完成后才能开始另一些任务。某项目 X 包含任务 A、B、……、J, 其 PERT 图如下图所示 (A=1 表示任务 A 的持续时间是 1 天), 则项目 X 的关键路径是 (17)。



- (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. A-D-H-J      B. B-E-H-J      C. B-F-J      D. C-G-I-J

● “软件产品必须能够在 3 秒内对用户请求作出响应”属于软件需求中的 (18)。

- (18) A. 功能需求      B. 非功能需求      C. 设计约束      D. 逻辑需求

● 某项目为了修正一个错误而进行了修改。错误修正后，还需要进行 (19) 以发现这一修正是否引起原本正确运行的代码出错。

- (19) A. 单元测试      B. 接受测试      C. 安装测试      D. 回归测试

● 以下关于解释程序和编译程序的叙述中，正确的是 (20)。

- (20) A. 编译程序和解释程序都生成源程序的目标程序      B. 编译程序和解释程序都不生成源程序的目标程序  
 C. 编译程序生成源程序的目标程序，解释程序则不然      D. 编译程序不生成源程序的目标程序，而解释程序反之

● 以下关于传值调用与引用调用的叙述中，正确的是 (21)。

- ①在传值调用方式下，可以实现形参和实参间双向传递数据的效果  
 ②在传值调用方式下，实参可以是变量，也可以是常量和表达式  
 ③在引用调用方式下，可以实现形参和实参间双向传递数据的效果  
 ④在引用调用方式下，实参可以是变量，也可以是常量和表达式

- (21) A. ①③      B. ①④      C. ②③      D. ②④

● 在对高级语言源程序进行编译的过程中，为源程序中变量所分配的存储单元的地址属于 (22)。

- (22) A. 逻辑地址      B. 物理地址      C. 接口地址      D. 线性地址

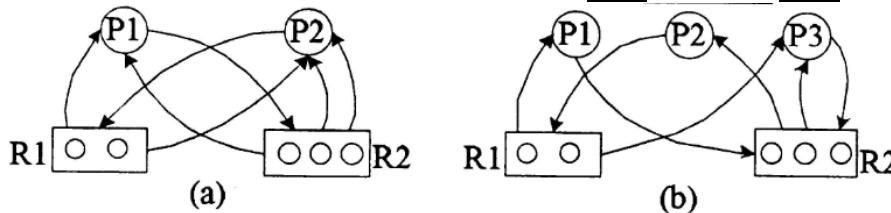
● 假设某分时系统采用简单时间片轮转法，当系统中的用户数为  $n$ 、时间片为  $q$  时，系统对每个用户的响应时间  $T =$  (23)。

- (23) A.  $n$       B.  $q$       C.  $n \times q$       D.  $n+q$

● 在支持多线程的操作系统中，假设进程 P 创建了若干个线程，那么 (24) 是不能被这些线程共享的。

- (24) A. 该进程的代码段      B. 该进程中打开的文件      C. 该进程的全局变量      D. 该进程中某线程的栈指针

● 进程资源图如图 (a) 和 (b) 所示，其中：图 (a) 中 (25)；图 (b) 中 (26)。



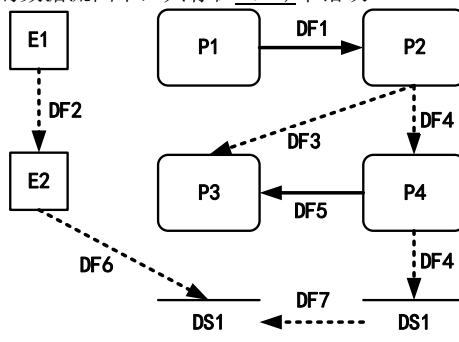
- (25) A. P1 是非阻塞节点，P2 是阻塞节点，所以该图不可以化简、是死锁的  
 B. P1、P2 都是阻塞节点，所以该图不可以化简、是死锁的  
 C. P1、P2 都是非阻塞节点，所以该图可以化简、是非死锁的  
 D. P1 是阻塞节点、P2 是非阻塞节点，所以该图不可以化简、是死锁的  
 (26) A. P1、P2、P3 都是非阻塞节点，该图可以化简、是非死锁的  
 B. P1、P2、P3 都是阻塞节点，该图不可以化简、是死锁的  
 C. P1 是阻塞节点，P1、P3 是非阻塞节点，该图可以化简、是非死锁的  
 D. P1、P2 是非阻塞节点，P3 是阻塞节点，该图不可以化简、是死锁的

● 假设内存管理采用可变式分区分配方案，系统中有五个进程 P1~P5，且某一时刻内存使用情况如下图所示（图中空白处表示未使用分区）。此时，若 P5 进程运行完并释其占有的空间，则释放后系统的空闲区数应 (27)；造成这种情况的原因是 (28)。

分区号	进程
0	P1
1	P2
2	
3	P4
4	P3
5	
6	P5
7	

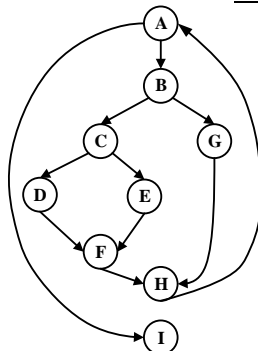
- (27) A. 保持不变 B. 减 1 C. 加 1 D. 置零  
 (28) A. 无上邻空闲区, 也无下邻空闲区 B. 有上邻空闲区, 但无下邻空闲区  
 C. 有下邻空闲区, 但无上邻空闲区 D. 有上邻空闲区, 也有下邻空闲区

● 在如下所示的数据流图中, 共存在 (29) 个错误。



- (29) A. 4 B. 6 C. 8 D. 9

● 软件的复杂性主要体现在程序的复杂性。(30)是度量软件复杂性的一个主要参数。若采用 McCabe 度量法计算环路复杂性, 则对于下图所示的程序图, 其环路复杂度为 (31)。



- (30) A. 代码行数 B. 常量的数量 C. 变量的数量 D. 调用的库函数的数量  
 (31) A. 2 B. 3 C. 4 D. 5

● (32) 不属于软件设计质量评审。

- (32) A. 功能与模块之间的对应关系 B. 软件规格说明是否符合用户的要求  
 C. 软件是否具有可测试性 D. 软件是否具有有良好的可靠性

● 在软件维护中, 由于企业的外部市场环境和管理需求的变化而导致的维护工作, 属于 (33) 维护。

- (33) A. 正确性 B. 适应性 C. 完善性 D. 预防性

● 在对软件系统进行评价时, 需要从信息系统的组成部分、评价对象和经济学角度出发进行综合考虑以建立起一套指标体系理论架构。从信息系统评价对象出发, 对于用户方来说, 他们所关心的是 (34)。

- (34) A. 用户需求和运行质量 B. 系统外部环境 C. 系统内部结构 D. 系统质量和技术水平

● 在设计测试用例时, 应遵循 (35) 原则。

- (35) A. 仅确定测试用例的输入数据, 无需考虑输出结果  
 B. 只需检验程序是否执行了应有的功能, 不需要考虑程序是否做了多余的功能  
 C. 不仅要设计有效合理的输入, 也要包含不合理、失效的输入

D. 测试用例应设计得尽可能复杂

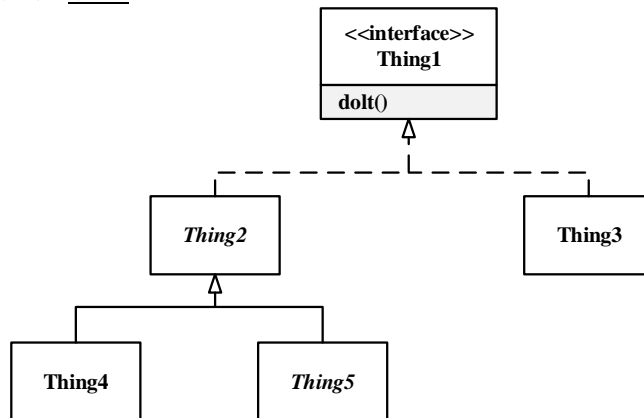
● 单元测试中，检查模块接口时，不需要考虑 (36)。

- (36) A. 测试模块的输入参数和形式参数在个数、属性、单位上是否一致  
 B. 全局变量在各模块中的定义和用法是否一致  
 C. 输入是否改变了形式参数  
 D. 输入参数是否使用了尚未赋值或者尚未初始化的变量

● 在多态的几种不同形式中，(37) 多态是一种特定的多态，指同一个名字在不同上下文中可代表不同的含义。

- (37) A. 参数 B. 包含 C. 过载 D. 强制

● 继承是父类和子类之间共享数据和方法的机制。以下关于继承的叙述中，不正确的是 (38)。有关下图中 doIt() 方法的叙述中，正确的是 (39)。

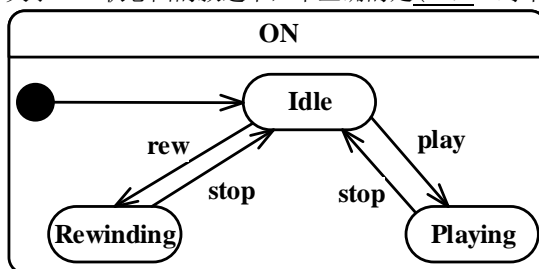


- (38) A. 一个父类可以有多个子类，这些子类都是父类的特例  
 B. 父类描述了这些子类的公共属性和操作  
 C. 子类可以继承它的父类（或祖先类）中的属性和操作而不必自己定义  
 D. 子类中可以定义自己的新操作而不能定义和父类同名的操作
- (39) A. doIt() 必须由 Thing3 实现，同时可能由 Thing4 实现  
 B. doIt() 必须由 Thing3 实现  
 C. doIt() 必须由 Thing2、Thing3、Thing4 和 Thing5 实现  
 D. doIt() 已经由 Thing1 实现，因此无需其它类实现

● 以下关于 UML 部署图的叙述中，正确的是 (40)。

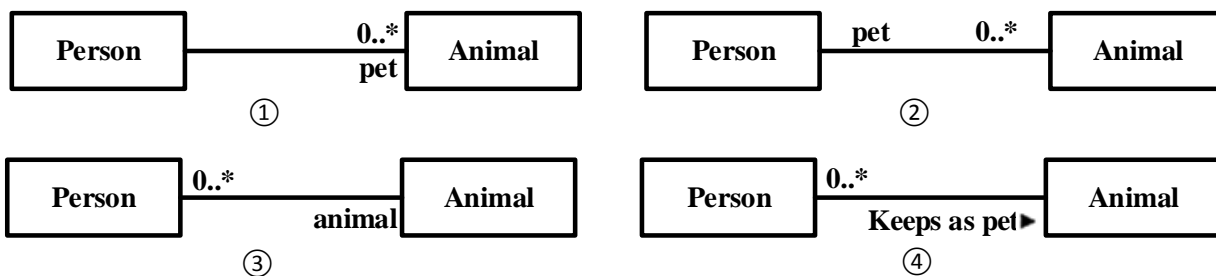
- (40) A. 因为一条消息总是有某种响应，所以部署组件之间的依赖是双向的  
 B. 部署组件之间的依赖关系类似于包依赖  
 C. 部署图不用于描述代码的物理模块  
 D. 部署图不用于描述系统在不同计算机系统的物理分布

● 以下关于 UML 状态图的叙述中，不正确的是 (41)。对下图的描述正确的是 (42)。



- (41) A. 用于描述一个对象在多个用例中的行为  
 B. 用于某些具有多个状态的对象而不是系统中大多数或全部对象  
 C. 用于描述多个对象之间的交互  
 D. 可以用于用户界面或控制对象
- (42) A. ON 是一个并发状态  
 B. 因为此状态图中没有终止 (final) 状态，所以此图是无效的  
 C. play、stop 和 rew 是动作  
 D. ON 是超状态

● 描述一些人 (Person) 将动物 (Animal) 养为宠物 (Pet) 的是图 (43)。



- (43) A. ① B. ② C. ③ D. ④

● (44) 设计模式能使一个对象的状态发生改变时通知所有依赖它的监听者。(45) 设计模式限制类的实例对象只能有一个。适配器 (Adapter) 设计模式可以用于 (46)。用于为一个对象添加更多功能而不使用子类的是 (47) 设计模式。

- (44) A. 责任链 (Chain of Responsibility) B. 命令 (Command)  
C. 抽象工厂 (Abstract Factory) D. 观察者 (Observer)  
(45) A. 原型 (Prototype) B. 工厂方法 (Factory Method)  
C. 单例 (Singleton) D. 生成器 (Builder)  
(46) A. 将已有类的接口转换成和目标接口兼容 B. 改进系统性能  
C. 将客户端代码数据转换成目标接口期望的合适的格式 D. 使所有接口不兼容类可以一起工作  
(47) A. 桥接 (Bridge) B. 适配器 (Adapter) C. 组合 (Composite) D. 装饰器 (Decorator)

● 以下关于语言  $L = \{a^n b^n | n \geq 1\}$  的叙述中, 正确的是 (48)。

- (48) A. 可用正规式 “ $aa^*bb^*$ ” 描述, 但不能通过有限自动机识别  
B. 可用正规式 “ $a^*b^*$ ” 表示, 也可用有限自动机识别  
C. 不能用正规式表示, 但可用有限自动机识别  
D. 不能用正规式表示, 也不能通过有限自动机识别

● 编译过程中, 对高级语言程序语句的翻译主要考虑声明语句和可执行语句。对声明语句, 主要是将所需要的信息正确地填入合理组织的 (49) 中; 对可执行语句, 则是 (50)。

- (49) A. 符号表 B. 栈 C. 队列 D. 树  
(50) A. 翻译成机器代码并加以执行 B. 转换成语法树 C. 翻译成中间代码或目标代码 D. 转换成有限自动机

● 采用顺序表和单链表存储长度为  $n$  的线性序列, 根据序号查找元素, 其时间复杂度分别为 (51)。

- (51) A.  $O(1)$ 、 $O(1)$  B.  $O(1)$ 、 $O(n)$  C.  $O(n)$ 、 $O(1)$  D.  $O(n)$ 、 $O(n)$

● 设元素序列  $a$ 、 $b$ 、 $c$ 、 $d$ 、 $e$ 、 $f$  经过初始为空的栈  $S$  后, 得到出栈序列  $c$   $e$   $d$   $f$   $b$   $a$ , 则栈  $S$  的最小容量为 (52)。

- (52) A. 3 B. 4 C. 5 D. 6

● 输出受限的双端队列是指元素可以从队列的两端输入、但只能从队列的一端输出, 如下图所示。若有  $e_1$ 、 $e_2$ 、 $e_3$ 、 $e_4$  依次进入输出受限的双端队列, 则得不到输出序列 (53)。



- (53) A.  $e_4$ 、 $e_3$ 、 $e_2$ 、 $e_1$  B.  $e_4$ 、 $e_2$ 、 $e_1$ 、 $e_3$  C.  $e_4$ 、 $e_3$ 、 $e_1$ 、 $e_2$  D.  $e_4$ 、 $e_2$ 、 $e_3$ 、 $e_1$

● 在数据库系统中, 视图是一个 (54)。

- (54) A. 真实存在的表, 并保存了待查询的数据 B. 真实存在的表, 只有部分数据来源于基本表  
C. 虚拟表, 查询时只能从一个基本表中导出 D. 虚拟表, 查询时可以从一个或者多个基本表或视图中导出

● 给定关系模式  $R(U, F)$ , 其中: 属性集  $U = \{A, B, C, D, E, G\}$ , 函数依赖集  $F = \{A \rightarrow B, A \rightarrow C, C \rightarrow D, AE \rightarrow G\}$ 。若将  $R$  分解为如下两个子模式 (55), 则分解后的关系模式保持函数依赖。

- (55) A.  $R_1(A, B, C)$  和  $R_2(D, E, G)$  B.  $R_1(A, B, C, D)$  和  $R_2(A, E, G)$   
C.  $R_1(B, C, D)$  和  $R_2(A, E, G)$  D.  $R_1(B, C, D, E)$  和  $R_2(A, E, G)$

● 假设学生  $Students$  和教师  $Teachers$  关系模式如下所示:

$Students$  (学号, 姓名, 性别, 类别, 身份证号)

$Teachers$  (教师号, 姓名, 性别, 身份证号, 工资)

其中, 学生关系中的类别分为 “本科生” 和 “研究生” 两类。

a. 查询在读研究生的教师的平均工资、最高与最低工资之间差值的 SQL 语句如下:

```
SELECT (56)
FROM Students, TCachers
WHERE (57);
```

- (56) A.  $AVG(工资)$  AS 平均工资,  $MAX(工资) - MIN(工资)$  AS 差值  
B. 平均工资 AS  $AVG(工资)$ , 差值 AS  $MAX(工资) - MIN(工资)$

- C. AVG (工资) ANY 平均工资, MAX (工资) - MIN (工资) ANY 差值  
 D. 平均工资 ANY AVG (工资), 差值 ANY MAX (工资) - MIN (工资)
- (57) A. Students. 身份证号=Teachers. 身份证号  
 B. students. 类别='研究生'  
 C. Students. 身份证号=Teachers. 身份证号 AND Students. 类别='研究生'  
 D. Studentis. 身份证号=Teachers. 身份证号 OR Students. 类别='研究生'
- b. 查询既是女教师, 又是研究生且工资大于等于 3500 元的身份证号和姓名的 SQL 语句如下:

```
(SELECT 身份证号, 姓名
FROM Students
WHERE (58) )
INTERSECT
(SELECT 身份证号, 姓名
FROM Teachers
WHERE (59) )
```

- (58) A. 工资>=3500 B. 工资>='3500'  
 C. 性别=女 AND 类别=研究生 D. 性别='女', AND 类别='研究生'  
 (59) A. 工资>=3500 B. 工资>='3500',  
 C. 性别=女 AND 类别=研究生 D. 性别='女', AND 类别='研究生'

● 考虑下述背包问题的实例。有 5 件物品, 背包容量为 100, 每件物品的价值和重量如下表所示, 并已经按照物品的单位重量价值从大到小排好序。根据物品单位重量价值大优先的策略装入背包中, 则采用了 (60) 设计策略。考虑 0/1 背包问题 (每件物品或者全部装入背包或者不装入背包) 和部分背包问题 (物品可以部分装入背包), 求解该实例得到的最大价值分别为 (61)。

物品编号	价值	重量
1	50	5
2	200	25
3	180	30
4	225	45
5	200	50

- (60) A. 分治 B. 贪心 C. 动态规划 D. 回溯  
 (61) A. 605 和 630 B. 605 和 605 C. 430 和 630 D. 630 和 430

● 给定  $n$  个整数构成的数组  $\{a_1, a_2, \dots, a_n\}$  和整数  $x$ , 判断  $A$  中是否存在两个元素  $a_i$  和  $a_j$ , 使得  $a_i + a_j = x$ 。为了求解该问题, 首先用归并排序算法对数组  $A$  进行从小到大排序; 然后判断是否存在  $a_i + a_j = x$ , 具体的方法如下列伪代码所示。则求解该问题时排序算法应用了 (62) 算法设计策略, 整个算法的时间复杂度为 (63)。

```
...
i=1; j=n
while i<j
    if  $a_i + a_j = x$  return true
    else if  $a_i + a_j > x$ 
        j--;
    else
        i++;
return false;
...
```

- (62) A. 分治 B. 贪心 C. 动态规划 D. 回溯  
 (63) A.  $O(n)$  B.  $O(n \lg n)$  C.  $O(n^2)$  D.  $O(n \lg^2 n)$

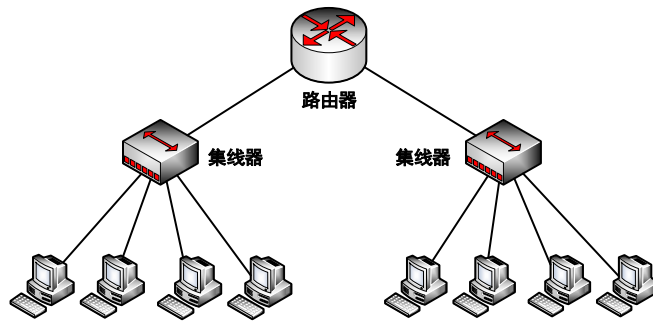
● 一个高度为  $h$  的满二叉树的结点总数为  $2^h - 1$ , 从根结点开始, 自上而下、同层次结点从左至右, 对结点按照顺序依次编号, 即根结点编号为 1, 其左、右孩子结点编号分别为 2 和 3, 再下一层从左到右的编号为 4、5、6、7, 依此类推。那么, 在一棵满二叉树中, 对于编号为  $m$  和  $n$  的两个结点, 若  $n = 2m + 1$ , 则 (64) 结点。

- (64) A.  $m$  是  $n$  的左孩子 B.  $m$  是  $n$  的右孩子 C.  $n$  是  $m$  的左孩子 D.  $n$  是  $m$  的右孩子

● 以下关于哈希 (Hash, 散列) 查找的叙述中, 正确的是 (65)。

- (65) A. 哈希函数应尽可能复杂些, 以消除冲突  
 B. 构造哈希函数时应尽量使关键字的所有组成部分都能起作用  
 C. 进行哈希查找时, 不再需要与查找表中的元素进行比较  
 D. 在哈希表中只能添加元素不能删除元素

● 网络配置如下图所示，其中使用了一台路由器、一台交换机和一台集线器，对于这种配置，下面的论断中正确的是 (66)。



- (66) A. 2 个广播域和 2 个冲突域  
B. 1 个广播域和 2 个冲突域  
C. 2 个广播域和 5 个冲突域  
D. 1 个广播域和 8 个冲突域

● 把网络 117.15.32.0/23 划分为 117.15.32.0/27，则得到的子网是 (67) 个。每个子网中可使用的主机地址是 (68) 个。

- (67) A. 4  
(68) A. 30  
B. 8  
B. 31  
C. 16  
C. 32  
D. 32  
D. 34

● 通常工作在 UDP 协议之上的应用是 (69)。

- (69) A. 浏览网页  
B. Telnet 远程登录  
C. VoIP  
D. 发送邮件

● 随着网站知名度不断提高，网站访问量逐渐上升，网站负荷越来越重，针对此问题，一方面可通过升级网站服务器的软硬件，另一方面可以通过集群技术，如 DNS 负载均衡技术来解决。在 Windows 的 DNS 服务器中通过 (70) 操作可以确保域名解析并实现负载均衡。

- (70) A. 启用循环，启动转发器指向每个 Web 服务器  
B. 禁止循环，启动转发器指向每个 Web 服务器  
C. 禁止循环，添加每个 Web 服务器的主机记录  
D. 启用循环，添加每个 Web 服务器的主机记录

● So it is today. Schedule disaster, functional misfits, and system bugs all arise because the Left hand doesn't know what the right hand is doing. As work (71), the several teams slowly change the functions, sizes, and speeds of their own programs, and they explicitly or implicitly (72) their assumptions about the inputs available and the uses to be made of the outputs.

For example, the implementer of a program-overlaying function may run into problems and reduce speed relying on statistics that show how (73) this function will arise in application programs. Meanwhile, back at the ranch, his neighbor may be designing a major part of the supervisor so that it critically depends upon the speed of this function. This change in speed itself becomes a major specification change, and it needs to be proclaimed abroad and weighed from a system point of view.

How, then, shall teams (74) with one another? In as many ways as possible.

. Informally. Good telephone service and a clear definition of intergroup dependencies will encourage the hundreds of calls upon which common interpretation of written documents depends.

. Meetings. Regular project meetings, with one team after another giving technical briefings, are (75). Hundreds of minor misunderstandings get smoked out this way.

• Workbook. A formal project workbook must be started at the beginning.

- (71) A. starts  
(72) A. change  
(73) A. frequently  
(74) A. work  
(75) A. worthless  
B. proceeds  
B. proceed  
B. usually  
B. program  
B. valueless  
C. stops  
C. smooth  
C. commonly  
C. communicate  
C. useless  
D. speeds  
D. hide  
D. rarely  
D. talk  
D. invaluable

## 1.2 下午试题

### 试题一（共 15 分）

阅读下列说明和图，回答问题 1 至问题 3，将解答填入答题纸的对应栏内。

#### 【说明】

某慈善机构欲开发一个募捐系统，以跟踪记录为事业或项目向目标群体进行募捐而组织的集体性活动。该系统的主要功能如下所述。

(1) 管理志愿者。根据募捐任务给志愿者发送加入邀请、邀请跟进、工作任务；管理志愿者提供的邀请响应、志愿者信息、工作时长、工作结果等。

(2) 确定募捐需求和收集所募捐赠（资金及物品）。根据需求提出募捐任务、活动请求和捐赠请求，获取所募集的资金和物品。

(3) 组织募捐活动。根据活动请求，**确定活动时间范围**。根据活动时间，搜索**场馆**，即：向场馆发送场馆可用性请求，获得场馆可用性。然后根据活动时间和地点推广募捐活动，根据相应的活动信息举办活动，从募款机构获取资金并向其发放赠品。获取和处理捐赠，根据捐赠请求，提供所募集的捐赠；处理与捐赠人之间的交互，即：

录入捐赠人信息，处理后存入捐赠人信息表；从捐赠人信息表中查询捐赠人信息，向捐赠人发送募捐请求，并将已联系的捐赠人存入已联系的捐赠人表。根据捐赠请求进行募集，募得捐赠后，将捐赠记录存入捐赠表；对捐赠记录进行处理后，存入已处理捐赠表，向捐赠人发送致谢函。根据已联系的捐赠人和捐赠记录进行跟进，将捐赠跟进情况发送给捐赠人。

现采用结构化方法对募捐系统进行分析与设计，获得如图 1-1、1-2 和 1-3 所示分层数据流图。

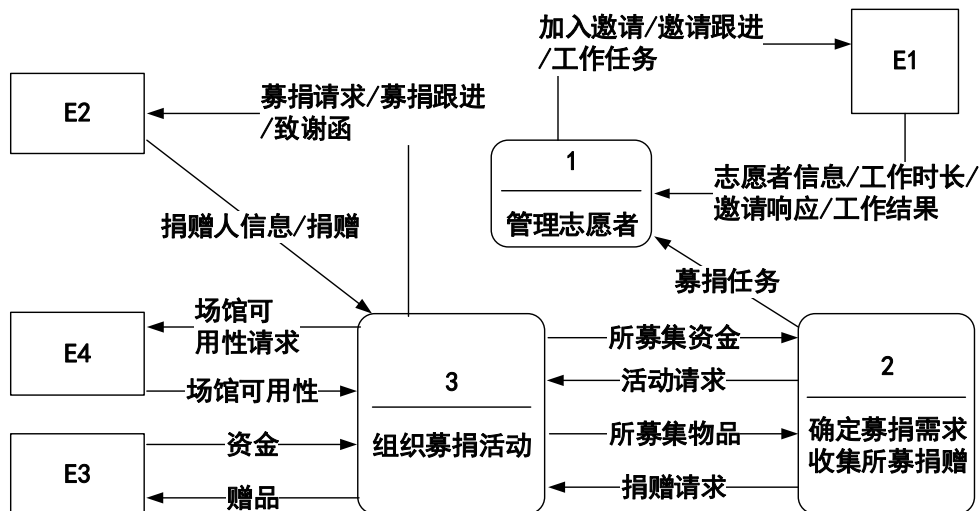


图 1-1 0 层数据流图

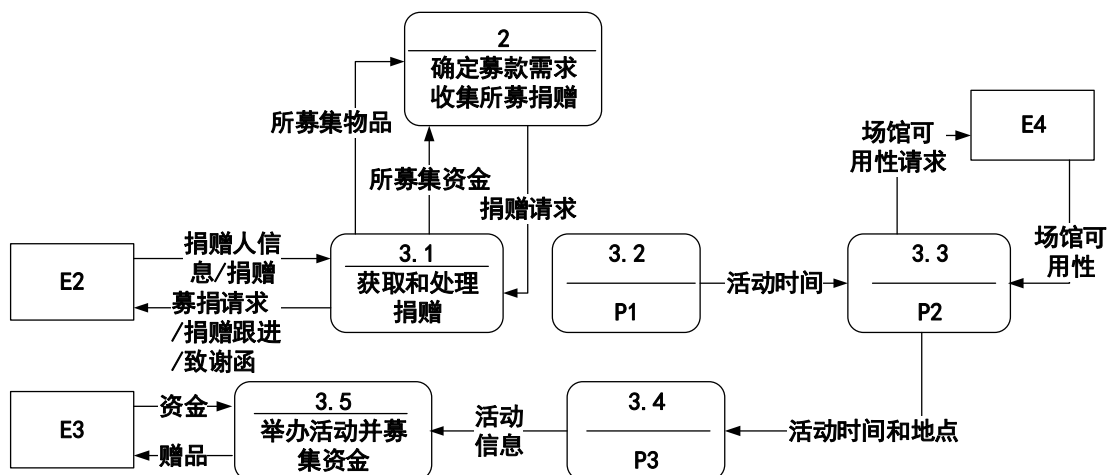


图 1-2 1 层数据流图

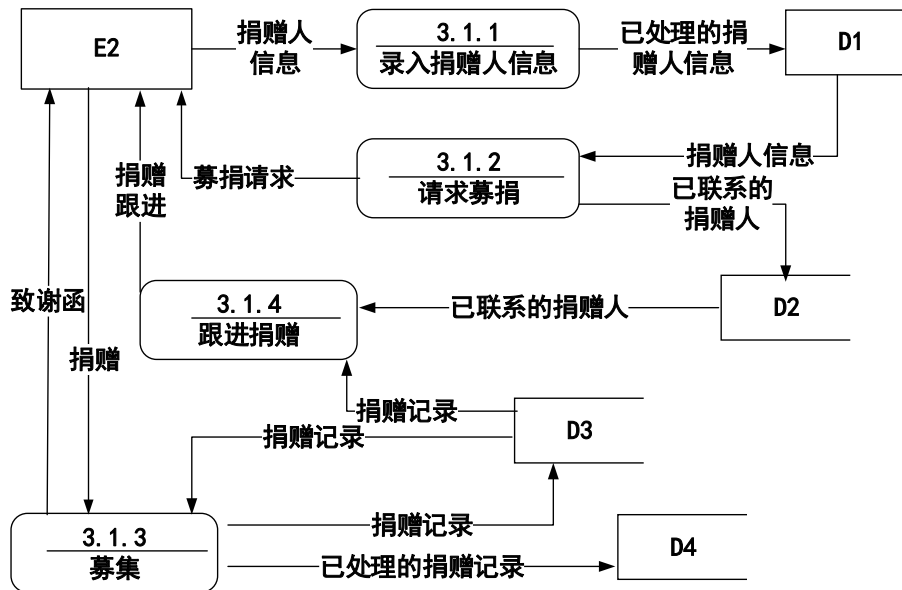


图 1-3 2 层数据流图

**【问题 1】（4 分）**

使用说明中的词语，给出图 1-1 中的实体 E1~E4 的名称。

**【问题 2】（7 分）**

在建模 DFD 时，需要对有些复杂加工（处理）进行进一步精化，图 1-2 为图 1-1 中处理 3 的进一步细化的 1 层数据流图，图 1-3 为图 1-2 中 3.1 进一步细化的 2 层数据流图。补全图 1-2 中加工 P1、P2 和 P3 的名称和图 1-2 与图 1-3 中缺少的数据流。

**【问题 3】（4 分）**

使用说明中的词语，给出图 1-3 中的数据存储 D1~D4 的名称。

**试题二（共 15 分）**

阅读下列说明，回答问题 1 至 3 问题，将解答填入答题纸的对应栏内。

**【说明】**

某电视台拟开发一套信息管理系统，以方便对全台的员工、栏目、广告和演播厅等进行管理。

**【需求分析】**

(1) 系统需要维护全台员工的详细信息、栏目信息、广告信息和演播厅信息等。员工的信息主要包括：工号、姓名、性别、出生日期、电话、住址等。栏目信息主要包括：栏目名称、播出时间、时长等。广告信息主要包括：广告编号、价格等。演播厅信息包括：房间号、房间面积等。

(2) 电视台根据调度单来协调各档栏目、演播厅和场务。一销售档栏目只会占用一个演播厅，但会使用多名场务来进行演出协调。演播厅和场务可以被多个栏目循环使用。

(3) 电视台根据栏目来插播广告。每档栏目可以插播多条广告，每条广告也可以在多档栏目插播。

(4) 一档栏目可以有多个主持人，但一名主持人只能主持一档栏目。

(5) 一名编辑人员可以编辑多条广告，一条广告只能由一名编辑人员编辑。

**【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）：

演播厅（房间号，房间面积）

栏目（栏目名称，播出时间，时长）

广告 (广告编号, 销售价格,     (1)    )  
 员工 (工号, 姓名, 性别, 出生日期, 电话, 住址)  
 主持人 (主持人工号,     (2)    )  
 插播单 (    (3)    , 播出时间)  
 调度单 (            (4)            )

**【问题 1】 (7 分)**

补充图 2-1 中的联系和联系的类型。

**【问题 2】** (5 分)

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)~(4) 补充完整, 并用下划线指出 (1)~(4) 所在关系模式的主键。

**【问题 3】 (3 分)**

现需要记录广告商信息，增加广告商实体。一个广告商可以提供多条广告，一条广告只由一个广告商提供。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。

### 试题三（共 15 分）

阅读下列说明和图，回答问题 1 至问题 3，将解答填入答题纸的对应栏内。

**【说明】**

某城市拟开发一个基于 Web 的城市黄页，公开发布该城市重要的组织或机构（以下统称为客户）的基本信息，方便城市生活。该系统的主要功能描述如下：

(1) 搜索信息：任何使用 Internet 的网络用户都可以搜索发布在城市黄页中的信息，例如客户的名称、地址、联系电话等。

(2) 认证: 客户若想在城市黄页上发布信息, 需通过系统的认证。认证成功后, 该客户成为系统授权用户。

(3) 更新信息: 授权用户登录系统之后, 可以更改自己在城市黄页中的相关信息, 例如变更联系电话等。

(4) 删除客户：对于拒绝继续在城市黄页上发布信息的客户，由系统管理员删除该客户的相关信息。

系统采用面向对象方法进行开发，在开发过程中认定出如表 3-1 所示的类。系统的用例图和类图分别如图 3-1 和 3-2 所示。

表 3-1 类列表

类名	说明
InternetClient	网络用户
CustomerList	客户集，维护城市黄页上的所有客户信息
Customer	客户信息，记录单个客户的信息
RegisteredClient	授权用户
Administrator	系统管理员

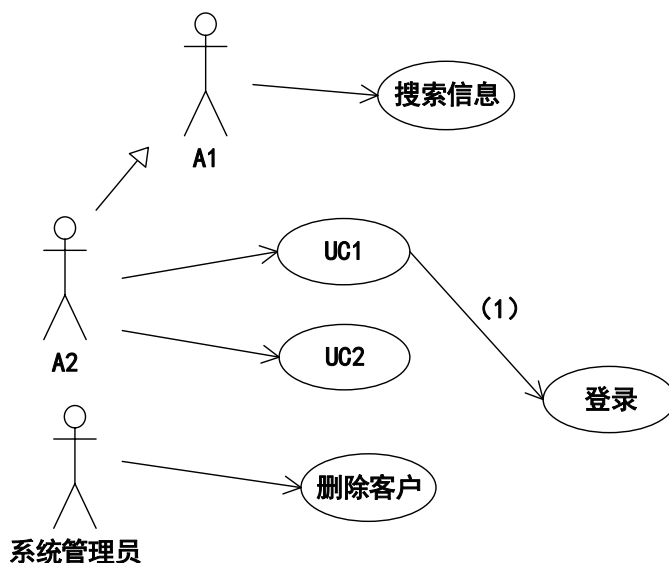


图 3-1 系统用例图

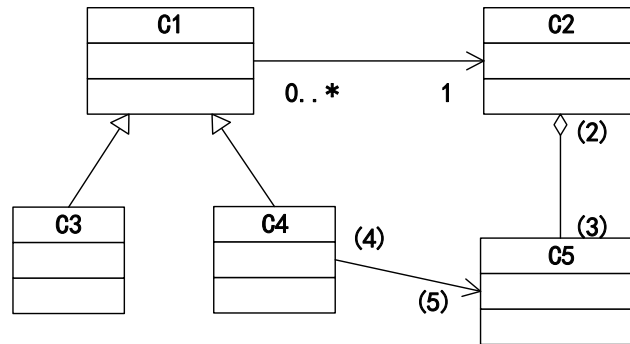


图 3-2 系统类图

**【问题 1】(5 分)**

根据说明中的描述，给出图 3-1 中 A1 和 A2 处所对应的参与者、UC1 和 UC2 处所对应的用例以及 (1) 处的关系。

**【问题 2】(7 分)**

根据说明中的描述，给出图 3-2 中 C1~C5 所对应的类名（表 3-1 中给出的类名）和 (2)~(5) 处所对应的多重度。

**【问题 3】(3 分)**

认定类是面向对象分析中非常关键的一个步骤。一般首先从问题域中得到候选类集合，再根据相应的原则从该集合中删除不作为类的，剩余的就是从问题域中认定出来的类。简要说明选择候选类的原则，以及对候选类集合进行删除的原则。

**试题四（共 15 分）**

阅读下列说明和 C 代码，回答问题 1 至问题 3，将解答写在答题纸的对应栏内。

**【说明】**

设有  $m$  台完全相同的机器运行  $n$  个独立的任务，运行任务  $i$  所需要的时间为  $t_i$ ，要求确定一个调度方案，使得完成所有任务所需要的时间最短。

假设任务已经按照其运行时间从大到小排序。算法基于最长运行时间作业优先的策略：按顺序先把每个任务分配到一台机器上，然后将剩余的任务依次放入最先空闲的机器。

**【C 代码】**

下面是算法的 C 语言实现。

(1) 常量和变量说明

$m$ : 机器数

$n$ : 任务数

$t[]$ : 输入数组，长度为  $n$ ，其中每个元素表示任务的运行时间，下标从 0 开始

$s[][]$ : 二维数组，长度为  $m \times n$ ，下标从 0 开始，其中元素  $s[i][j]$  表示机器  $i$  运行的任务  $j$  的编号

$d[]$ : 数组，长度为  $m$ ，其中元素  $d[i]$  表示机器  $i$  的运行时间，下标从 0 开始

$count[]$ : 数组，长度为  $m$ ，下标从 0 开始，其中元素  $count[i]$  表示机器  $i$  运行的任务数

$i$ : 循环变量

$j$ : 循环变量

$k$ : 临时变量

$max$ : 完成所有任务的时间

$min$ : 临时变量

(2) 函数 schedule

```

void schedule() {
    inti, j, k, max=0;
    for(i=0; i<m; i++) {
        d[i]=0;
        for(j=0; j<n; j++) {
            s[i][j]=0;
        }
    }
    for(i=0; i<m; i++) {           //分配前 m 个任务
        s[i][0]=i;
        (1);
        count[i]=1;
    }
    for ( (1); 1<n; i++) {         //分配后 n-m 个任务
        int min=d[0];
        k=0;
    }
}
    
```

```

    for(j=1; j<m; j++) {          //确定空闲机器
        if(min > d[j]) {
            min=d[j];
            k=j;                  //机器 k 空闲
        }
    }
    (3) ;
    count[k]=count[k]+1;
    d[k]=d[k] + t[i];
}
for(i=0; i<m; i++) {            //确定完成所有任务需要的时间
    if ( (4) ) {
        max=d[i];
    }
}
}

```

**【问题 1】(8 分)**

根据说明和 C 代码，填充 C 代码中的空 (1)~(4)。

**【问题 2】(2 分)**

根据说明和 C 代码，该问题采用了 (5) 算法设计策略，时间复杂度为 (6) (用 O 符号表示)。

**【问题 3】(5 分)**

考虑实例 m=3 (编号 0~2), n=7 (编号 0~6)，各任务的运行时间为 {16, 14, 6, 5, 4, 3, 2}。则在机器 0、1 和 2 上运行的任务分别为 (7)、(8) 和 (9) (给出任务编号)。从任务开始运行到完成所需要的时间为 (10)。

从下列的 2 道试题（试题 5 和试题 6）中任选 1 道解答。  
如果解答的试题数超过 1 道，则题号小的 1 道解答有效。

**试题五（共 15 分）**

阅读下列说明和 C++ 代码，将应填入 (n) 处的字句写在答题纸的对应栏内。

**【说明】**

现要求实现一个能够自动生成求职简历的程序。简历的基本内容包括求职者的姓名、性别、年龄及工作经历等。希望每份简历中的工作经历有所不同，并尽量减少程序中的重复代码。

现采用原型 (Prototype) 模式来实现上述要求，得到如图 5-1 所示的类图。

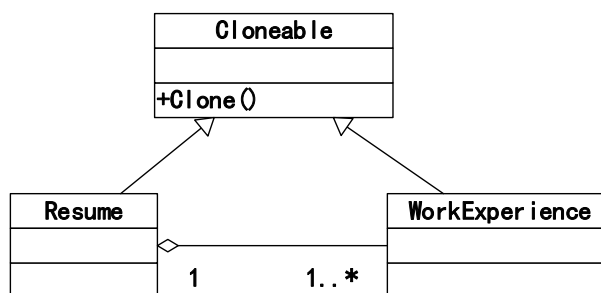


图 5-1 类图

**【C++代码】**

```

#include<string>
using namespace std;
class Cloneable{
public:
    (1) ;
};
class WorkExperience : public Cloneable{          //工作经历
private:
    string workDate;
    string company;

```

```

public:
    Cloneable * Clone() {
        _____ (2) _____;
        obj ->workDate=this ->workDate;
        obj ->company=this ->company;
        return obj;
    }
    //其余代码省略
};
class Resume : public Cloneable{    //简历
Private:
    string name; string sex; string age;
public:
    Resume(string name){    /*实现略*/    }
    void SetPersonalInfo(string sex,string age) {    /*实现略*/    }
    void SetWorkExperience(string workDate,string company) {    /*实现略*/    }
    Cloneable * Clone() {
        _____ (3) _____;
        obj ->name=this ->name;
        obj ->sex=this ->sex;
        obj ->age=this ->age;
        retrun obj;
    }
};
int main() {
    Resume * a=new Resume(“张三”);
    a ->SetPersonalInfo(“男”,“29”);
    a ->SetworkExPerience(“1998~2000”,“XXX 公司”);
    Resume * b=_____ (4) _____;
    b ->SetWorkExperience(“2001 - 2006”,“YYY 公司”);
    return 0;
}

```

#### 试题六 (共 15 分)

阅读下列说明和 Java 代码，将应填入\_\_\_\_\_ (n) \_\_\_\_\_处的字句写在答题纸的对应栏内。

##### 【说明】

现要求实现一个能够自动生成求职简历的程序。简历的基本内容包括求职者的姓名、性别、年龄及工作经历等。希望每份简历中的工作经历有所不同，并尽量减少程序中的重复代码。

现采用原型 (Prototype) 模式来实现上述要求，得到如图 6-1 所示的类图。

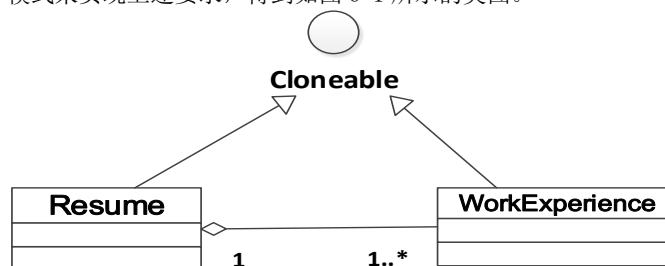


图 6-1 类图

##### 【Java 代码】

```

class WorkExperience _____ (1) _____ Cloneable{    //工作经历
    private String workDate;
    private String comPany;
    public Object Clone() {
        _____ (2) _____;
        obj.workDate=this. workDate;
        obj.comPany=this.comPany;
        return obj;
    }
}

```

```
class Resume _____ (3) _____ Cloneable {    //简历
    private String name;
    private String sex;
    private String age;
    private WorkExperience work;
    public Resume(String name) {
        this.name=name;        work=new WorkExperience ();
    }
    private Resume(WorkExperience work) {
        this.work=_____ (4) _____;
    }
    public void SetPersonalInfo (string sex,string age) {    /*实现略*/ }
    public void SetWorkExperience(string workDate,string comany) {    /*实现略*/ }
    public object Clone() {
        Resume obj=_____ (5) _____;
        //其余代码省略
        return obj;
    }
}

class WorkResume{
    public static void main(String[] arg) {
        Resume=new Resume( “张三” );
        a.SetPersonalInfo( “男”, “29” );
        a.SetWorkExperience( “1998~2000”, “XXX 公司” );

        Resume b=_____ (6) _____;
        b.SetWorkExperience( “2001~2006”, “YYY 公司” );
    }
}
```

## 第二章 2013 年下半年软件设计师试题

### 2.1 上午试题

●在程序执行过程中, Cache 与主存的地址映像由 (1) 。

- (1) A. 硬件自动完成      B. 程序员调度      C. 操作系统管理      D. 程序员与操作系统协议完成

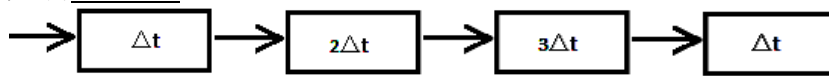
●指令寄存器的位数取决于 (2) 。

- (2) A. 存储器的容量      B. 指令字长      C. 数据总线的宽度      D. 地址总线的宽度

●若计算机存储数据采用的是双符号位 (00 表示正号, 11 表示负号), 两个符号相同的数相加时, 如果运算结果的两个符号位经 (3) 运算得 1, 才可断定这两个数相加的结果产生了溢出。

- (3) A. 逻辑与      B. 逻辑或      C. 逻辑同或      D. 逻辑异或

●若指令流水线由 4 段组成, 各段所需要的时间如下图所示。连续输入 8 条指令时的吞吐率 (单位时间内流水线完成的任务数或输出的结果数) 为 (4) 。



- (4) A.  $8/56 \Delta t$       B.  $8/32 \Delta t$       C.  $8/28 \Delta t$       D.  $8/24 \Delta t$

● (5) 不是 RISC 的特点。

- (5) A. 指令种类丰富      B. 高校的流水线操作      C. 寻址方式较少      D. 硬布线控制

●若某计算机字长为 32 位, 内存容量为 2GB, 按字编址, 则可寻址范围为 (6) 。

- (6) A. 1024M      B. 1GB      C. 512M      D. 2GB

●下列网络攻击行为中, 属于 DoS 攻击的是 (7) 。

- (7) A. 特洛伊木马攻击      B. SYN Flooding 攻击      C. 端口欺骗攻击      D. IP 欺骗攻击

●PKI 体制中, 保证数字证书不被篡改的方法是 (8) 。

- (8) A. 用 CA 的私钥对数字证书签名      B. 用 CA 的公钥对数字证书签名  
C. 用证书主人的私钥对数字证书签名      D. 用证书主人的公钥对数字证书签名

●下面算法中, 不属于公开密钥加密算法的是 (9) 。

- (9) A. ECC      B. DSA      C. RSA      D. DES

●矢量图是常用的图形图像表示形式, (10) 是描述矢量图的基本组成单位

- (10) A. 像素      B. 像素点      C. 图元      D. 二进制位

●视频信息是连续的图像序列, (11) 是构成视频信息的基本单元。

- (11) A. 帧      B. 场      C. 幅      D. 像素

●以下多媒体像素编辑软件中, (12) 主要用于动画编辑和处理。

- (12) A. WPS      B. Xara3D      C. Photoshop      D. Cool Edit Pro

●为说明某一问题, 在学术论文中需要引用某些资料, 以下叙述中错误的是 (13) 。

- (13) A. 既可引用发表的作品, 也可引用未发表的作品      B. 只能限于介绍、评论作品  
C. 只要不构成自己作品的主要部分, 可适当引用资料      D. 不必征得原作者的同意, 不需要向他支付报酬

●以下作品中, 不适用或不受著作权法保护的是 (14) 。

- (14) A. 某教师在课堂上的讲课      B. 某作家的作品《红河谷》  
C. 最高人民法院组织编写的《行政诉讼案例选编》      D. 国务院颁布的《计算机软件保护条例》

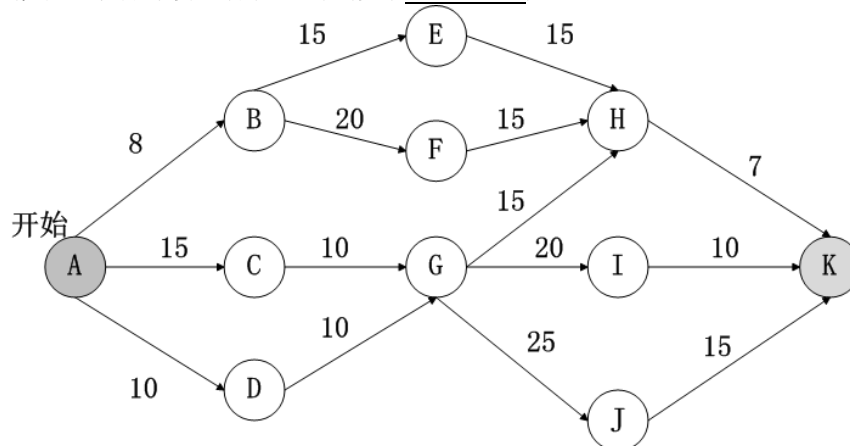
●以下关于数据流图中基本加工的叙述, 不正确的是 (15) 。

- (15) A. 对每一个基本加工, 必须有一个加工规格说明  
B. 加工规格说明必须描述把输入数据流变换为输出数据流的加工规则  
C. 加工规格说明必须描述实现加工的具体流程  
D. 决策表可以用来表示加工规格说明

●在规划分模块时, 一个模块的作用范围应该在其控制范围之内, 若发现其作用范围不在其控制范围内, 则 (16) 不是适当的处理方法。

- (16) A. 将判定所在模块合并到父模块中, 使判定处于较高层次      B. 将受判定影响的模块下移到控制范围内  
C. 将判定上移到层次较高的位置      D. 将父模块下移, 使该判定处于较高层次

●下图是一个软件项目的活动图，其中顶点表示项目里程碑，连接顶点的边表示包含的活动，则里程碑 (17) 在关键路径上。若在实际项目进展中，活动 AD 在活动 AC 开始 3 天后才开始，而完成活动 DG 过程中，由于临时时间发生，实际需要 15 天才能完成，则完成该项目的最短时间比原计划多了 (18) 天。



- (17) A. B                      B. C                      C. D                      D. I  
 (18) A. 8                      B. 3                      C. 5                      D. 6

●针对“关键职员在项目没完成时就跳槽”的风险，最不合格的风险管理策略是 (19)。

- (19) A. 对每一个关键性的技术人员，要培养后备人员                      B. 建立项目组，以使大家都了解有关开发活动的信息  
 C. 临时招聘具有相关能力的新职员                      D. 对所有工作组织细致的评审

●程序运行过程中常使用参数在函数（过程）间传递信息，引用调用传递的是实参的 (20)。

- (20) A. 地址                      B. 类型                      C. 名称                      D. 值

●已知文法  $G: S \rightarrow A0|B1, A \rightarrow S1|1, B \rightarrow S0|0$ ，其中 S 是开始符号，从 S 出发可以推导出 (21)。

- (21) A. 所有由 0 构成的字符串                      B. 所有由 1 构成的字符串  
 C. 某些 0 和 1 个数相等的字符串                      D. 所有 0 和 1 个数不同的字符串

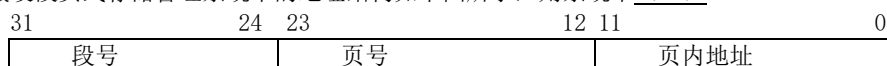
●算数表达式  $a+(b-c)*d$  的后缀式是 (22)（一、+、\* 表示算数的减加乘运算，运算符的优先级和结合性遵循惯例）。

- (22) A.  $bc-d*a+$                       B.  $abc-d*+$                       C.  $ab+c-d*$                       D.  $abcd-*+$

●假设系统采用 PV 操作系统实现进程同步与互斥，若有 n 个进程共享一台扫描仪，那么当信号量 S 的值为 -3 时，表示系统中有 (23) 个进程等待使用扫描仪。

- (23) A. 0                      B. n-3                      C. 3                      D. n

●假设段页式存储管理系统中的地址结构如下图所示，则系统中 (24)。



- (24) A. 页的大小为 4K，每个段的大小均为 4096 个页，最多可有 256 个段。  
 B. 页的大小为 4K，每个段最大只许有 4096 个页，最多可有 256 个段。  
 C. 页的大小为 8K，每个段的大小均为 2048 个页，最多可有 128 个段。  
 D. 页的大小为 8K，每个段最大只许有 2048 个页，最多可有 128 个段。

●某文件管理系统采用位示图 (bitmap) 记录磁盘的使用情况，如果系统的字长为 32 位，磁盘物理块的大小为 4MB，物理块一次编号为：0、1、2、...，位示图字依次编号为：0、1、2、...，那么 16385 号物理块的使用情况在位示图中的第 (25) 个字中描述；如果磁盘的容量为 1000GB，那么位示图需要 (26) 个字来表示。

- (25) A. -128                      B. 256                      C. 512                      D. 1024  
 (26) A. 1200                      B. 3200                      C. 6400                      D. 80100

●假设系统中有三类互斥资源 R1、R2 和 R3，可用资源分别为 10、5 和 3，在  $T_0$  时刻系统中有 P1、P2、P3、P4 和 P5 五个进程，这些进程对资源的最大需求和已分配资源数如下表所示，此时系统剩余的可用资源数分别为 (27)。如果进程按 (28) 序列执行，那么系统状态是安全的。

进程 \ 资源	最大需求量			已分配资源数		
	R1	R2	R3	R1	R2	R3
P1	5	3	1	1	1	1
P2	3	2	0	2	1	0
P3	6	1	1	3	1	0
P4	3	3	2	1	1	1
P5	2	1	1	1	1	0

- (27) A. 1、1 和 0                      B. 1、1 和 1                      C. 2、1 和 0                      D. 2、0 和 1  
 (28) A. P1—P2—P4—P5—P3                      B. P5—P2—P4—P3—P1                      C. P4—P2—P1—P5—P3                      D. P5—P1—P4—P2—P3

● (29) 开发过程模型最不适用于开发初期对软件需求缺乏准确全面认识的情况。

- (29) A. 瀑布 B. 演化 C. 螺旋 D. 增量

● (30) 不是增量式开发的优势。

- (30) A. 软件可以快速的交付  
B. 早起的增量作为原型，从而可以加强对系统后续开发需要的理解  
C. 具有最高优先级的功能首先交付，随着后续的增量不断加入，这就使得更重要的功能得到更多的测试  
D. 很容易将客户需求划分为多个增量

● 在对程序质量进行评审，模块结构式一个重要的评审项，评审内容中不包含 (31)。

- (31) A. 数据结构 B. 数据流结构 C. 控制流结构 D. 模块结构与功能结构之间的对应关系

● SEI 能力成熟度模型 (SEI CMM) 把软件开发企业分为 5 个成熟度级别，其中 (32) 重点关注产品和过程质量。

- (32) A. 级别 2: 重复级 B. 级别 3: 确定级 C. 级别 4: 管理级 D. 级别 5: 优化级

● 系统可维护性的评价指标不包括 (33)。

- (33) A. 可理解性 B. 可测试性 C. 可移植性 D. 可修改性

● 逆向工程从源代码或目标代码中提取设计信息，通常在原软件声明周期的 (34) 阶段进行。

- (34) A. 需求分析 B. 软件设计 C. 软件实现 D. 软件维护

● 一个程序根据输入的年份和月份计算该年中该月的天数，输入参数包括年份（正整数）、月份（用 1~12 表示），若用等价类划分测试方法测试，则 (35) 不是一个进行核实的测试用例（分号后表示测试的输出）。

- (35) A. (2013, 1; 31) B. (0, 1; ' 错误' ) C. (0, 13; ' 错误' ) D. (2000, -1; ' 错误' )

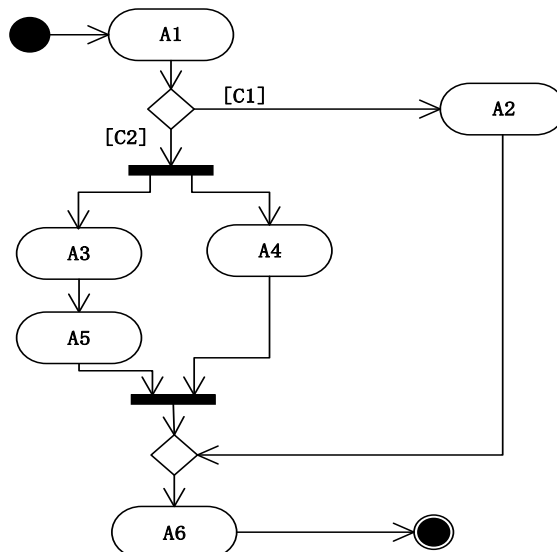
● (36) 不是单元测试主要检查的内容。

- (36) A. 模块接口 B. 局部数据结构 C. 全部数据结构 D. 重要的执行路径

● 在领域类模型中不包含 (37)。

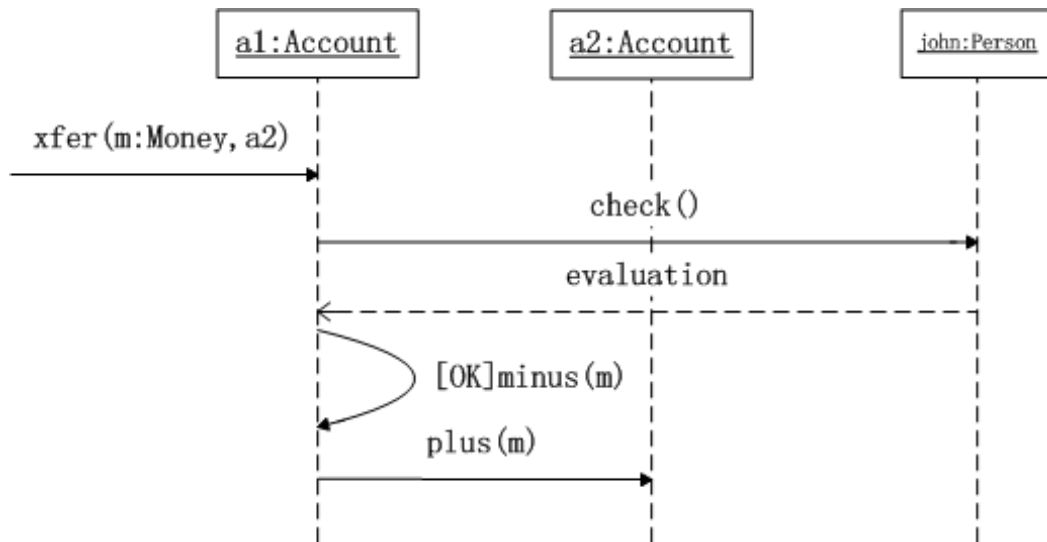
- (37) A. 属性 B. 操作 C. 关联 D. 领域对象

● 在执行如下所示的 UML 活动图时，能同时运行的最大线程数为 (38)。



- (38) A. 4 B. 3 C. 2 D. -1

● 下图所示的 UML 序列图中，(39) 表示返回消息，Account 类应该实现的方法有 (40)。



- (39) A. xfer                      B. check                      C. evaluation                      D. minus  
 (40) A. xfer0                      B. xfer()、plus()和minus()  
       C. check()、plus()和minus()                      D. xfer()、evaluation()、plus()和minus()

●在面向对象技术中，(41)定义了父类和子类之间的关系，子类中以更具体的方式实现从父类继承下来的方法称为(42)，不同类的对象通过(43)相互通信。

- (41) A. 覆盖                      B. 继承                      C. 消息                      D. 多态  
 (42) A. 覆盖                      B. 继承                      C. 消息                      D. 多态  
 (43) A. 覆盖                      B. 继承                      C. 消息                      D. 多态

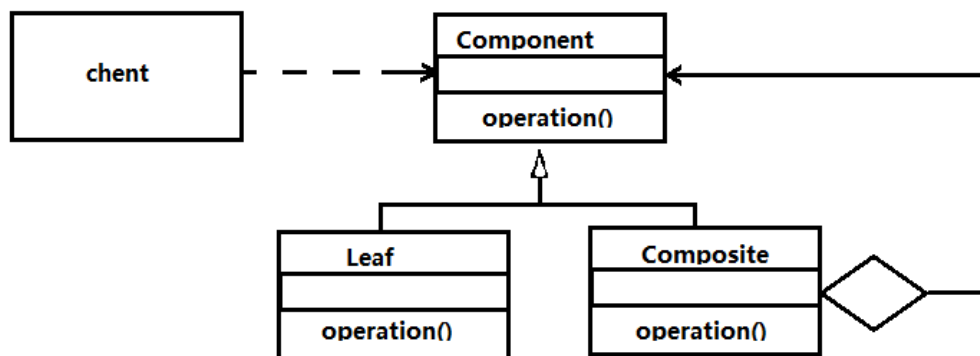
●(44)设计模式定义一系列算法，把他们一个个封装起来，并且使他们可相互通信，这一模式使得算法可独立于它的客户而变化

- (44) A. 策略 (Strategy)    B. 抽象工厂 (Abstract Factory)    C. 状态 (State)    D. 观察者 (Visitor)

●在发布-订阅 (Publish Subscribe) 消息模型中，订阅者订阅一个主题后，当该主题有新消息到达时，所有订阅者都会受到通知，(45)设计模式最合适这一模型。

- (45) A. 适配器 (Adapter)    B. 通知 (Notifier)    C. 状态 (State)    D. 观察者 (Observer)

●下图所示为(46)设计模式，适用于(47)。



- (46) A. 组件 (Component)    B. 适配器 (Adapter)    C. 组合 (Composite)    D. 装饰器 (Decorator)

(47) A. 表示对象的部分-整体层次结构

- B. 不希望在抽象和它的实现部分之间有一个固定的绑定关系  
 C. 在不影响其他对象的情况下，以动态、透明的方式给单个对象添加职责  
 D. 使所有接口不兼容类可以一起工作

●将高级语言程序翻译为机器语言的过程中，常引入中间代码，其好处是(48)。

- (48) A. 有利于反编译处理    B. 有利于进行与机器无关的优化    C. 尽早发现语法错误    D. 可以简化语法和语义分析

●对高级语言程序进行编译的过程中，有穷自动机 (NFA 或 DFA) 是进行(49)的适当工具。

- (49) A. 词法分析    B. 语法分析    C. 语义分析    D. 出错处理

●弱类型语言 (动态类型语言) 是指不需要进行变量/对象类型声明的语言。(50)属于弱类型语言。

- (50) A. Java    B. C / C++    C. Python    D. C#

●若有关系 R (A, B, C, D, E) 和 S (B, C, F, G)，则 R 和 S 自然连接运算后的属性列有 (51) 个，与表达式  $\Pi_{1,3,6,7}(\sigma_{3<6}(R \bowtie S))$  等价的 SQL 语句如下：

SELECT (52) FROM (53) WHERE (54) ;

- (51) A. 5                      B. 6                      C. 7                      D. 9  
 (52) A. A, R, C, F, G                      B. A, C, S, B, S, F                      C. A, C, S, B, S, C                      D. R, A, R, C, S, B, S, C  
 (53) A. R                      B. S                      C. RS                      D. R, S  
 (54) A. R.B=S.B AND R.C=S.C AND R.C<S.B                      B. R.B=S.B AND R.C=S.C AND R.C<S.F  
       C. R.B=S.B OR R.C=S.C OR R.C<S.B                      D. R.B=S.B OR R.C=S.C OR R.C<S.F

●分布式数据库系统中，(55) 是指用户无需知道数据存放的物理位置。

- (55) A. 分片透明                      B. 复制透明                      C. 逻辑透明                      D. 位置透明

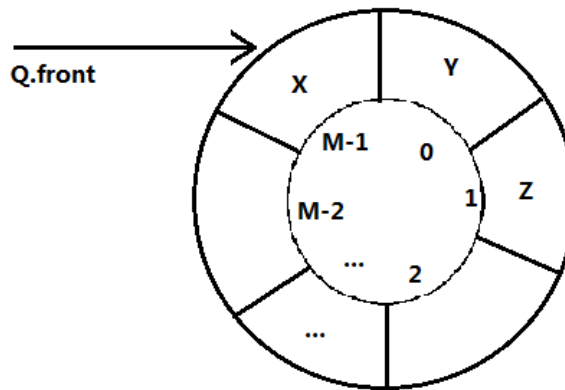
●计算机系统的软硬件故障可能会造成数据库中的数据被破坏，为了防止这一问题，通常需要 (56)，以便发生故障时恢复数据库。

- (56) A. 定期安装 DBMS 和应用程序                      B. 定期安装应用程序，并将数据库做镜像  
       C. 定期安装 DBMS，并将数据库作备份                      D. 定期将数据库作备份，在进行事务处理时，需要将数据更新写入日志文件

●以下关于线性表存储结构的叙述，正确的是 (57)

- (57) A. 线性表采用顺序存储结构时，访问表中任意一个指定序号元素的时间复杂度为常量级  
       B. 线性表采用顺序存储结构时，在表中任意位置插入新元素的运算时间复杂度为常量级  
       C. 线性表采用链式存储结构时，访问表中任意一个指定序号元素的时间复杂度为常量级  
       D. 线性表采用链式存储结构时，在表中任意位置插入新元素的运算时间复杂度为常量级

●设循环队列的定义中有 front 和 size 两个域变量，其中 front 表示队头元素的指针，size 表示队列的长度，如下图所示 (队列长度为 3，队头元素为 X，队尾元素为 Z)，设队列的存储空间容量为 M，则队尾元素的指针为 (58)



- (58) A. (Q.front+Q.size-1)                      B. (Q.front+ Q.size-1+M) %M                      C. (Q.front-Q.size)                      D. (Q.front-Q.seze+M) %M

●在一个有时间图 G 的拓扑序列中，定点  $V_i$  排列在  $V_j$  之前，说明图 G 中 (59)

- (59) A. 一定存在弧  $(V_i, V_j)$                       B. 一定存在弧  $(V_j, V_i)$   
       C. 可能存在  $V_i$  到  $V_j$  的路径，而不可能存在  $V_j$  到  $V_i$  的路径                      D. 可能存在  $V_j$  到  $V_i$  的路径，而不可能存在  $V_i$  到  $V_j$  的路径

●以下关于哈夫曼树的叙述，正确的是 (60)

- (60) A. 哈夫曼树一定是满二叉树，其每层结点数都达到最大值  
       B. 哈夫曼树一定是平衡二叉树，其每个结点左右子树的高度差为-1, 0, 1  
       C. 哈夫曼树中左孩子结点的权值小于父节点、右孩子结点的权值大于父节点  
       D. 哈夫曼树中叶子结点的权值越小则距离树根越远、叶子结点的权值越大则距离树根越近

●某哈希表 (散列表) 的长度为 N，设散列函数为  $H(\text{Key}) \bmod p$ ，采用线性探测法解决冲突。以下关于 P 值的叙述中，正确的是 (61)

- (61) A. P 的值一般为不大于 n 且最接近 n 的质数                      B. P 的值一般为不大于 n 的任意数  
       C. P 的值必须小于 n 的合数                      D. P 的值必须等于 n

●对 n 个基本有序的整数进行排序，若采用插入排序算法，则时间和空间复杂度为 (62)；若采用快速排序算法，则时间和空间复杂度分别为 (63)。

- (62) A.  $O(n^2)$  和  $O(n)$                       B.  $O(n)$  和  $O(n)$                       C.  $O(n^2)$  和  $O(1)$                       D.  $O(n)$  和  $O(1)$   
 (63) A.  $O(n^2)$  和  $O(n)$                       B.  $O(n \lg n)$  和  $O(n)$                       C.  $O(n^2)$  和  $O(1)$                       D.  $O(n \lg n)$  和  $O(1)$

●在求解某问题时，经过分析发现该问题具有最优子结构性质，求解过程中子问题被重复求解，则采用 (64) 算法设计策略，以深度优先的方法是搜索解空间，则采用 (65) 算法设计策略。

- (64) A. 分治                      B. 动态规则                      C. 贪心                      D. 回溯

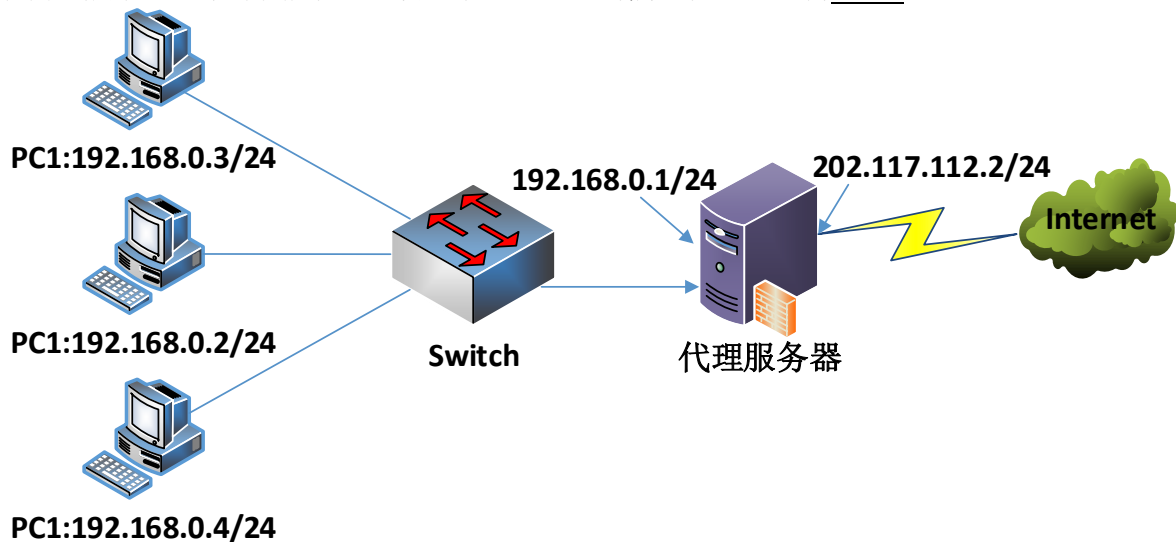
(65) A. 动态规则

B. 贪心

C. 回溯

D. 分治限界

●某单位的局域网配置如下图所示, PC2 发送到 Internet 上的报文的原 IP 地址为 (66)。



(66) A. 192.168.0.2

B. 192.168.0.1

C. 202.117.112.1

D. 202.117.112.2

●在 IPv4 向 IPv6 的过渡期间, 如果要使得两个 IPv6 结点可以通过现有的 IPv4 网络通信, 则应该使用 (67); 如果要使得纯 IPv6 结点可以与纯 IPv4 结点进行通信, 则需要使用 (68)。

(67) A. 堆栈技术

B. 双协议栈技术

C. 隧道技术

D. 翻译技术

(68) A. 堆栈技术

B. 双协议栈技术

C. 隧道技术

D. 翻译技术

●POP3 协议采用 (69) 模式进行通信, 当客户机需要服务时, 客户端软件与 POP3 服务器建立 (70) 连接。

(69) A. Browser/Server

B. Client/Server

C. Peer to Peer

D. Peer to Server

(70) A. TCP

B. UDP

C. PHP

D. IP

●There is nothing in this world constant, but inconstancy. —SWIFT

Project after project designs a set of algorithms and then plunges into construction of customer deliverable software on a schedule that demands delivery of the first thing built.

In most projects, the first system built is (71) usable. It may be too slow, too big, awkward to use, or all three. There is no (72) but to start again, smarting but smarter and build a redesigned version in which these problems are solved. The discard and (73) may be done in one lump, or it may be done piece by piece. But all large-system experience shows that it will be done. Where a new system concept or new technology is used, one has to build a system to throw away, for even the best planning is not so omniscient (全知的) as to get it right the first time.

The management question, therefore, is not whether to build a pilot system and throw it away. You will do that. The only question is whether to plan in advance to build a (74) or to promise to deliver the throwaway to customers. Seen this way, the answer is much clearer. Delivering that throwaway to customers buys time, but it does so only at the (75) of agony (极大痛苦) for the user, distraction for the builders while they do the redesign, and a bad reputation for the product that the best redesign will find hard to live down.

Hence plan to throw one away, you will, anyhow.

(71) A. almost

B. often

C. usually

D. barely

(72) A. alternative

B. need

C. possibility

D. solution

(73) A. design

B. redesign

C. plan

D. build

(74) A. throwaway

B. system

C. software

D. product

(75) A. worth

B. value

C. cost

D. invaluable

## 2.2 下午试题

### 试题一（共 15 分）

阅读以下说明和图，根据要求回答问题 1~问题 4。

#### 【说明】

某大学欲开发一个基于 Web 的课程注册系统，该系统的主要功能如下：

#### 1. 验证输入信息

（1）检查学生信息：检查学生输入的所有注册所需信息。如果信息不合法，返回学生信息不合法提示；如果合法，输出合法学生信息。

（2）检查学位考试结果：检查学生提供的学位考试结果。如果不合法，返回学位考试结果不合法提示；如果合法，检查该学生注册资格。

（3）检查学生注册资格：根据合法学生信息和合法学位考试结果，检查该学生对欲选课程的注册资格。如果无资格，返回无注册资格提示；如果有注册资格，则输出注册学生信息（包含选课学生标识）和欲注册课程信息。

#### 2. 处理注册申请

（1）存储注册信息：将注册学生信息记录在学生库。

（2）存储所注册课程：将选课学生标识与欲注册课程进行关联，然后存入课程库。

（3）发送注册通知：从学生库中读取注册学生信息，从课程库中读取所注册课程信息，给学生发送接受提示；给教务人员发送注册课程信息和已注册学生信息。

现采用结构化方法对课程注册系统进行分析设计，获得如图 1-1 所示的 0 层数据流图和图 1-2 所示的 1 层数据流图。

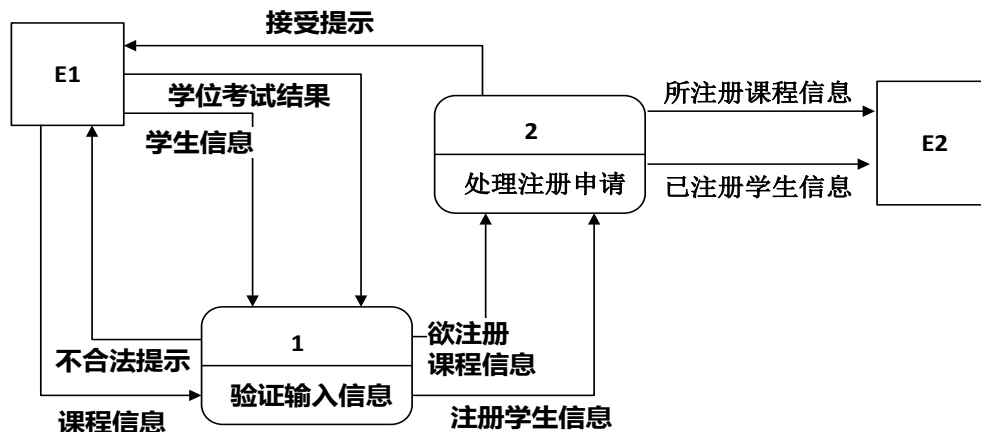


图 1-1 某课程注册系统 0 层数据流图

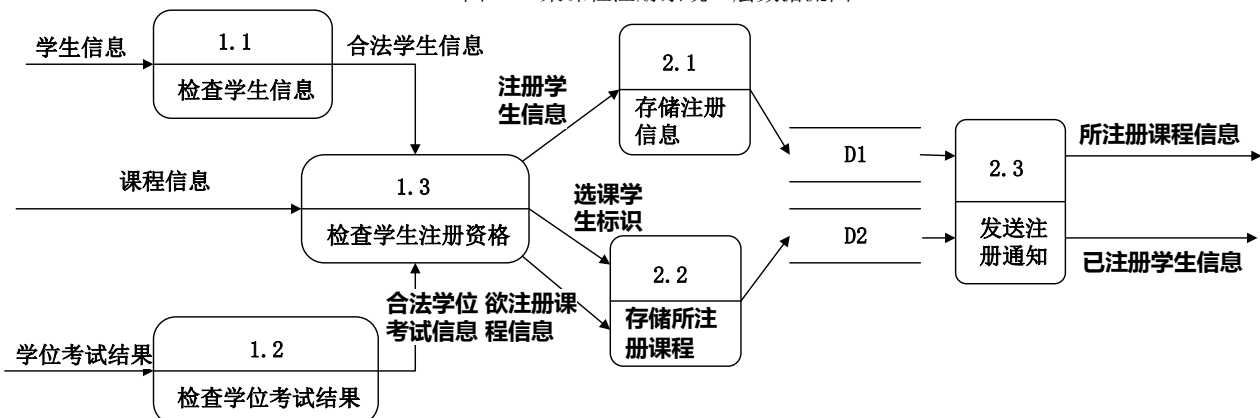


图 1-2 某课程注册系统 1 层数据流图

#### 【问题 1】（2 分）

使用说明中的词语，给出图 1-1 中的实体 E1 和 E2 的名称。

#### 【问题 2】（2 分）

使用说明中的语句，给出图 1-2 中的数据存储 D1 和 D2 的名称。

#### 【问题 3】（8 分）

根据说明和图中术语。补充图 1-2 中缺失的数据流及其起点和终点。

#### 【问题 4】（3 分）

根据补充完整的图 1-1 和图 1-2，说明上层的哪些数据流是由下层的哪些数据流组合而成。

**试题二（共 15 分）**

阅读以下说明，根据要求回答问题 1~问题 3。

**【说明】**

某快递公司为了方便管理公司物品运送的各项业务活动，需要构建一个物品运送信息管理系统。

**【需求分析结果】**

(1) 快递公司有多分公司，分公司信息包括分公司编号、名称、经理、办公电话和地址。每个分公司可以有多名员工处理分公司的日常业务，每名员工只能在一个分公司工作。每个分公司有一名经理负责管理分公司的业务和员工，系统需要记录每个经理的任职时间。

(2) 员工信息包括员工号、姓名、岗位、薪资、手机和家庭地址。其中，员工号唯一标识员工信息的每一个元组。岗位包括经理、调度员、业务员等。业务员根据客户提交的快件申请单进行快递受理事宜，一个业务员可以受理多个客户的快件申请，一个快件申请只能由一个业务员受理。调度员根据已受理的申请单安排快件的承运事宜，例如：执行承运的业务员、运达时间等。一个业务员可以执行调度员安排的多个快件的承运业务。

(3) 客户信息包括客户号、单位名称、通信地址、所属省份、联系人、联系电话、银行账号。其中，客户号唯一标识客户信息的每一个元组。当客户要寄快件时，先要提交快件申请单，申请号由系统自动生成。快件申请信息包括申请号、客户号、发件人、发件人电话、快件名称、运费、发出地、收件人、收件人电话、收件地址。其中，一个申请号对应唯一的一个快件申请，一个客户可以提交多个快件申请，但一个快件申请由唯一的一个客户提交。

**【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（图 2-1）和关系模式（不完整）如下：

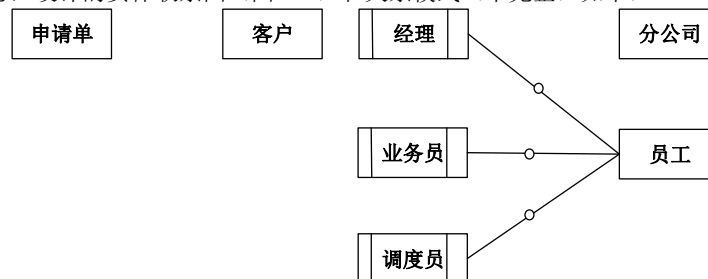


图 2-1 某物品运送信息管理系统实体联系图（不完整）

**【关系模式设计】**

分公司（分公司编号，名称，经理，办公电话，地址）

员工（员工号，姓名，(a)，岗位，薪资，手机号，家庭地址）

客户（客户号，单位名称，通信地址，所属省份，联系人，联系电话，银行账号）

申请单（(b)，发件人，发件人电话，发件人地址，快件名称，运费，收件人，收件人电话，收件地址，受理标识，业务员）

安排承运（(c)，实际完成时间，调度员）

**【问题 1】（5 分）**

根据问题描述，补充 5 个联系，完善图 2-1 的实体联系图。联系名可用联系 1、联系 2、联系 3、联系 4 和联系 5 代替，联系的类型分为 1:1、1:n 和 m:n（或 1:1、1:\* 和 \*:1）。

**【问题 2】（6 分）**

(1) 根据实体联系图，将关系模式中的空 (a) ~ (c) 补充完整；

(2) 给出员工，申请单和安排承运关系模式的主键和外键。

**【问题 3】（4 分）**

(1) 客户关系的通信地址可以进一步分为邮编、省、市、街道，那么该属性是否属于简单属性，为什么？请用 100 字以内的文字说明。

(2) 假设分公司需要增设一位经理的职位，那么分公司与经理之间的联系类型应修改为 (d)，分公司的主键应修改为 (e)。

**试题三（共 15 分）**

阅读以下说明和图，根据要求回答问题 1~问题 3。

**【说明】**

某航空公司会员积分系统(CFrequentFlyer)的主要功能描述如下：

乘客只要办理该航空公司的会员卡，即可成为普卡会员(CBasic)。随着飞行里程数的累计，可以从普通会员升级到银卡会员(CSilver)或金卡会员(CGold)。非会员(CNonMember)不能累计里程数。

每年年末，系统根据会员在本年度积累的里程数对下一年会员等级进行调整。

普卡会员在一年内累积的里程数若满 25,000 英里但不足 50,000 英里，则自动升级为银卡会员；若累积的里程数在 50,000 英里以上，则自动升级为金卡会员。银卡会员在一年内累积的里程数若在 50,000 英里以上，则自动升级为金卡会员。

若一年内没有达到对应级别要求的里程数，则自动降低会员等级。金卡会员一年内累积的里程数若不足 25,000 英里，则自动降级为普卡会员；若累积的里程数达到 25,000 英里，但是不足 50,000 英里，则自动降级为银卡会员。银卡会员一年内累积的里程数若不足 25,000 英里，则自动降级为普卡会员。

采用面向对象方法对会员积分系统进行分析设计，得到如图 3-1 所示的状态图和图 3-2 所示的类图。

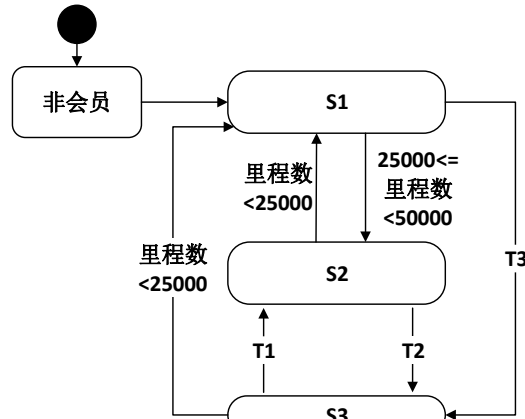


图 3-1 某会员积分系统图

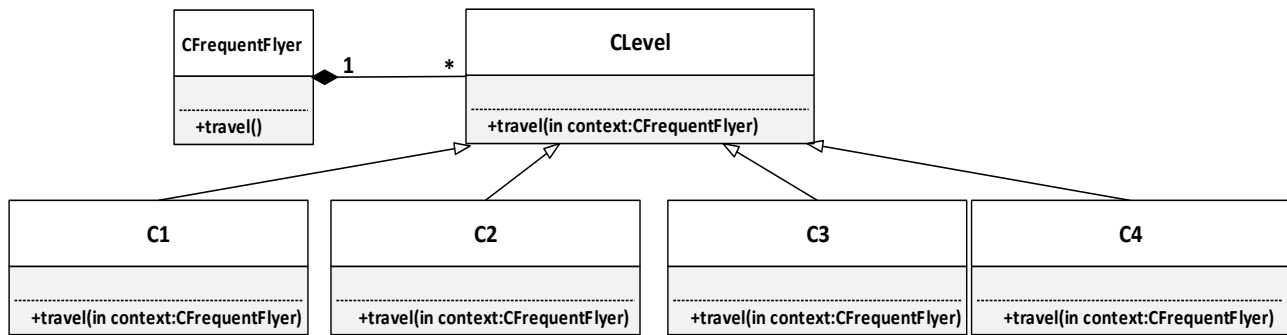


图 3-2 某会员积分系统类图

**【问题 1】（6 分）**

根据说明中的描述，给出图 3-1 中 S1~S3 处所对应的状态以及 T1~T3 处所对应的迁移的名称。

**【问题 2】（4 分）**

根据说明中的描述，给出图 3-2 中 C1~C4 所对应的类名（类名使用说明中给出的英文词汇）。

**【问题 3】（5 分）**

图 3-2 中所示的类图中使用了哪种设计模式？在这种设计模式下，类 CFrequentFlyer 必须具有的属性是什么？C1~C4 中的 travel 方法应具有什么功能？

**试题四（共 15 分）**

阅读以下说明和 C 代码，根据要求回答问题 1~问题 3。

**【说明】**

某工程计算中要完成多个矩阵相乘（链乘）的计算。

两个矩阵相乘要求第一个矩阵的列数等于第二个矩阵的行数，计算量主要由进行乘法运算的次数决定。采用标准的矩阵相乘算法，计算  $A_{m \times i} * B_{i \times p}$ ，需要  $m * i * p$  次乘法运算。

矩阵相乘满足结合律，多个矩阵相乘，不同的计算顺序会产生不同的计算量。以矩阵  $A_{10 \times 100}$ ， $A_{200 \times 5}$ ， $A_{5 \times 50}$  三个矩阵相乘为例，若按  $(A1 * A2) * A3$  计算，则需要进行  $10 * 100 * 5 + 10 * 5 * 50 = 7500$  次乘法运算；若按  $A1 * (A2 * A3)$  计算，则需要进行  $100 * 5 * 50 + 10 * 100 * 50 = 75000$  次乘法运算。可见不同的计算顺序对计算量有很大的影响。

矩阵链乘问题可描述为：给定  $n$  个矩阵  $\langle A1, A2, \dots, An \rangle$ ，矩阵  $A_i$  的维数为  $P_{i-1} \times P_i$ ，其中  $i=1, 2, \dots, n$ 。确定一种乘法顺序，使得这  $n$  个矩阵相乘时进行乘法的运算次数最少。

由于可能的计算顺序数量非常庞大，对较大的  $n$ ，用蛮力法确定计算顺序是不实际的。经过对问题进行分析，发现矩阵链乘问题具有最优子结构，即若  $A1 * A2 * \dots * An$  的一个最优计算顺序从第  $k$  个矩阵处断开，即分为  $A1 * A2 * \dots * Ak$  和  $A_{k+1} * A_{k+2} * \dots * An$  两个子问题，则该最优解应该包含  $A1 * A2 * \dots * Ak$  的一个最优计算顺序和  $A_{k+1} * A_{k+2} * \dots * An$  的一个最优计算顺序。据此构造递归式，

$$\text{cost}[i][j] = \begin{cases} 0 & \text{if } i=j \\ \min_{i \leq k < j} \{ \text{cost}[i][k] + \text{cost}[k+1][j] + p_i * p_{k+1} * p_{j+1} \} & \text{if } i < j \end{cases}$$

其中， $\text{cost}[i][j]$  表示  $A_{i+1} * A_{i+2} * \dots * A_{j+1}$  的最优计算的计算代价。最终需要求解  $\text{cost}[0][n-1]$ 。

**【C 代码】**

算法实现采用自底向上的计算过程。首先计算两个矩阵相乘的计算量，然后依次计算 3 个矩阵、4 个矩阵、……n 个矩阵相乘的最小计算量及最优计算顺序。下面是该算法的 C 语言实现。

(1) 主要变量说明

n: 矩阵数

seq[]: 矩阵维数序列

cost[][]: 二维数组，长度为 n\*n，其中元素 cost[i][j] 表示  $A_{i+1} * A_{i+2} * \dots * A_{j+1}$  的最优计算的计算代价

trace[][]: 二维数组，长度为 n\*n，其中元素 trace[i][j] 表示  $A_{i+1} * A_{i+2} * \dots * A_{j+1}$  的最优计算对应的划分位置，即 k

(2) 函数 cmm

```
#define N 100
```

```
int cost[N][N];
```

```
int trace[N][N];
```

```
int cmm(int n, int seq[]) {
```

```
    int tempCost;
```

```
    int tempTrace;
```

```
    int i, j, k, p;
```

```
    int temp;
```

```
    for(i=0; i<n; i++) { cost[i][i]=0; }
```

```
    for(p=1; p<n; p++) {
```

```
        for(i=0; (1); i++) {
```

```
            (2);
```

```
            tempCost=-1;
```

```
            for(k=i; k<j; k++) {
```

```
                temp=(3);
```

```
                if(tempCost==-1 || tempCost>temp) {
```

```
                    tempCost=temp;
```

```
                    (4);
```

```
                }
```

```
            }
```

```
            cost[i][j]=tempCost;
```

```
            trace[i][j]=tempTrace;
```

```
        }
```

```
    }
```

```
    return cost[0][n-1];
```

```
}
```

### 【问题 1】(8 分)

根据以上说明和 C 代码，填充 C 代码中的空 (1) ~ (4)。

### 【问题 2】(4 分)

根据以上说明和 C 代码，该问题采用了 (5) 算法设计策略，时间复杂度为 (6) (用 O 符号表示)。

### 【问题 3】(3 分)

考虑实例 n=6，各个矩阵的维数：A1 为 5\*10，A2 为 10\*3，A3 为 3\*12，A4 为 12\*5，A5 为 5\*50，A6 为 50\*6，即维数序列为 5, 10, 3, 12, 5, 50, 6。则根据上述 C 代码得到的一个最优计算顺序为 (7) (用加括号方式表示计算顺序)，所需要的乘法运算次数为 (8)。

从下列的两道试题(试题 5 和试题 6)中任选 1 道解答。如果解答的试题数超过 1 道，则题号小的 1 道解答有效。

### 试题五(共 15 分)

阅读以下说明和 C++ 代码，将应该填入 (n) 处的字句写在对应栏内。

#### 【说明】

欲开发一个绘图软件，要求使用不同的绘图程序绘制不同的图形。以绘制直线和圆形为例，对应的绘图程序如表 5-1 所示。

	DP1	DP2
绘制直线	draw_a_line(x1, y1, x2, y2)	drawline(x1, x2, y2)
绘制圆	draw_a_circle(x, y, r)	drawcircle(x, y, r)

表 5-1 不同的绘图程序

该绘图软件的扩展性要求，将不断扩充新的图形和新的绘图程序。为了避免出现类爆炸的情况，先采取桥接 (Bridge) 模式来实现上述要求，得到如图 5-1 所示的类图。

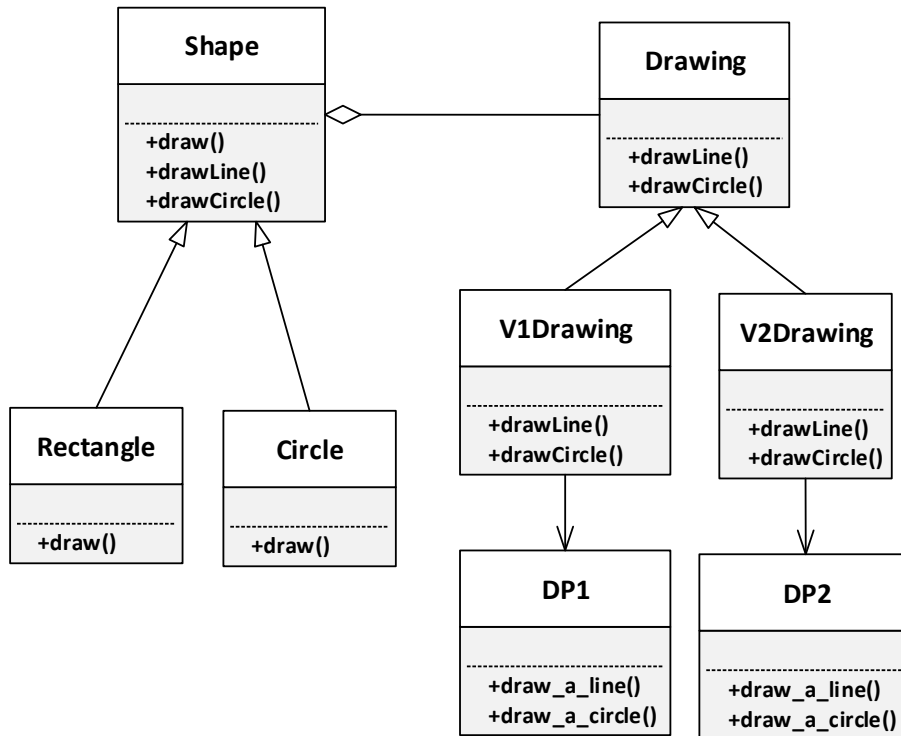


图 5-1 某绘图软件类图

【C++代码】

```

class DP1{
public:
    static void draw_aline(double x1,double y1,double x2,double y2) { /*代码省略*/ }
    static void draw_a_circle(double x,double y,double r) { /*代码省略*/ }
};
class DP2{
public:
    static void drawline(double x1,double x2,double y1,double y2) { /*代码省略*/ }
    static void drawcircle(double x,double y,double r) { /*代码省略*/ }
};
class Drawing{
public:
    (1) _____;
    (2) _____;
};
class V1Drawing:public Drawing{
public:
    void drawLine(double x1,double y1,double x2,double y2) { /*代码省略*/ }
    void drawCircle(double x,double y,double r) { (3) _____; }
};
class V2Drawing:public Drawing{
public:
    void drawLine(double x1,double y1,double x2,double y2) { /*代码省略*/ }
    void drawCircle(double x,double y,double r) { (4) _____; }
};
class shape{
public:
    (5) _____;
    Shape(Drawing *dp) { _dp=dp; }
    void drawLine(double x1,double y1,double x2,double y2) {
        _dp->drawLine(x1,y1,x2,y2);
    }
    void drawCircle(double x,double y,double r) { _dp->drawCircle(x,y,r); }
private: Drawing *_dp;
};
class Rectangle:public Shape{

```

```

public:
    void draw() { /*代码省略*/ }
    //其余代码省略
};

class Circle:public Shape{
private:double _x,_y,_r;
public:
    Circle(Drawing *_dp,double x,double y,double r):__(6)_{ _x=x;_y=y;_r=r;
};
    void draw(){drawCircle(_x,_y,_r); }
};
  
```

### 试题六（共 15 分）

阅读以下说明和 Java 代码，将应填入\_\_(n)\_\_\_处的字句写在对应栏内。

#### 【说明】

欲开发一个绘图软件，要求不同的绘图程序绘制不同的图形。以绘制直线和圆形为例，对应的绘图程序如表 6-1 所示。

	DP1	DP2
绘制直线	draw_a_line(x1,y1,x2,y2)	drawline(x1,x2,y2)
绘制圆	draw_a_circle(x,y,r)	drawcircle(x,y,r)

表 6-1 不同的绘图程序

该绘图软件的扩展性要求，将不断扩充新的图形和新的绘图程序。为了避免出现类爆炸的情况，先采取桥接（Bridge）模式来实现上述要求，得到如图 6-1 所示的类图。

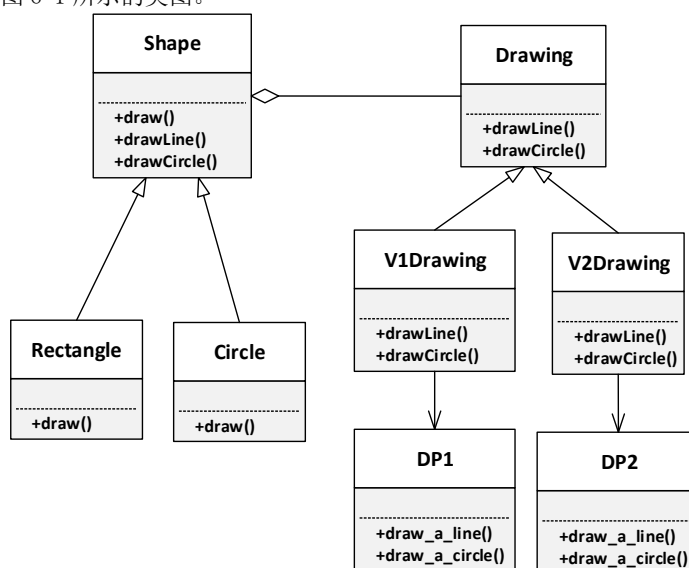


图 6-1 某绘图软件类图

#### 【Java 代码】

```

(1) Drawing{
    (2) ;
    (3) ;
}

class DP1{
    static public void draw_aline (double x1, double y1, double x2, double y2)
    static public void draw_a_circle(double x,double y,double r) { /*代码省略*/ }
};

class DP2{
public:
    static public void drawline(double x1,double x2,double y1,double y2) { /*代码省略*/ }
    static public void drawcircle(double x,double y,double r) { /*代码省略*/ }
};

class V1Drawing implements Drawing{
    public void drawLine(double x1,double y1,double x2,double y2) { /*代码省略*/ }
  
```

```
        public void drawCircle(double x,double y,double r) { ____ (4) ____; }
    };
    class V2Drawing implements Drawing{
    public:
        public void drawLine(double x1,double y1,double x2,double y2) { /*代码省略*/ }
        public void drawCircle(double x,double y,double r) { ____ (5) ____; }
    };
    abstract class shape{
        private Drawing _dp;
        ____ (6) ____;
        Shape(Drawing *dp) { _dp=dp }
        public void drawLine(double x1,double y1,double x2,double y2) {
            _dp->drawLine(x1,y1,x2,y2); }
        public void drawCircle(double x,double y,double r) { _dp->drawCircle(x,y,r); }
    };

    class Rectangle extends Shape{
        private double _x1,_x2,_y1,_y2;
        public Rectangle(Drawing dp,double x1,double y1,double x2,double y2)
            { /*代码省略*/ }
        public void draw() { /*代码省略*/ }
    };

    class Circle extends Shape{
        private:double _x,_y,_r;
        private double _x,_y,_r;
        public Circle(Drawing dp,double x,double y,double r) { /*代码省略*/ }
        public void draw() { drawCircle(_x,_y,_r); }
    };
};
```