全国计算机技术与软件专业技术资格（水平）考试

中级 软件设计师 2017 年 上半年 上午试卷 综合知识

（考试时间 150 分钟）

1. 在答题卡的指定位置上正确写入你的姓名和准考证号，并用正规 2B 铅笔在你写入的准考证号下填涂准考证号。

2. 本试卷的试题中共有 75 个空格，需要全部解答，每个空格 1 分，满分 75 分。

3. 每个空格对应一个序号，有 A 、 B 、 C 、 D 四个选项，请选择一个最恰当的选项作为解答，在答题卡相应序号下填涂该选项。

4. 解答前务必阅读例题和答题卡上的例题填涂样式及填涂注意事项。解答时用正规 2B 铅笔正确填涂选项，如需修改，请用橡皮擦干净，否则会导致不能正确评分。

试题一 CPU 执行算术运算或者逻辑运算时，常将源操作数和结果暂存在( )中。

A. 程序计数器（PC） B. 累加器（AC） C. 指令寄存器（IR） D. 地址寄存器（AR）

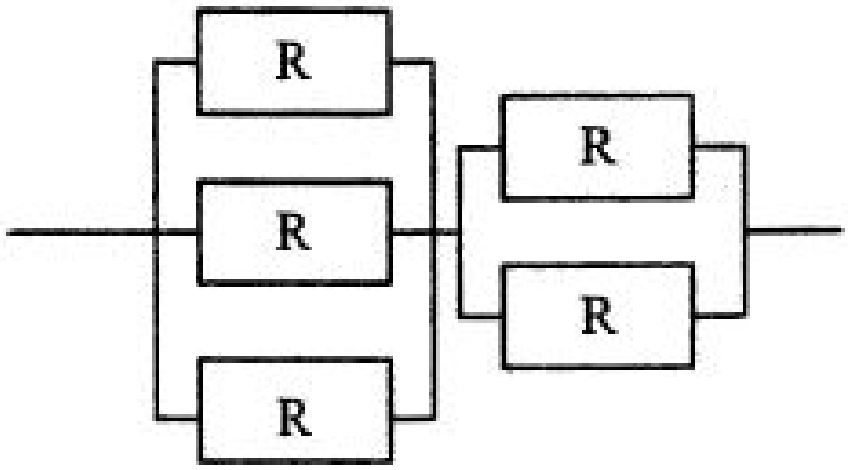
试题二 要判断字长为 16 位的整数 a 的低四位是否全为 0，则( )

A. 将 a 与 0x000F 进行"逻辑与"运算，然后判断运算结果是否等于 0
B. 将 a 与 0x000F 进行"逻辑或"运算，然后判断运算结果是否等于 F
C. 将 a 与 0x000F 进行"逻辑异或"运算，然后判断运算结果是否等于 0
D. 将 a 与 0x000F 进行"逻辑与"运算，然后判断运算结果是否等于 F

试题三 计算机系统中常用的输入/输出控制方式有无条件传送、中断、程序查询和 DMA 方式等。当采用( )方式时，不需要 CPU 执行程序指令来传送数据。

A. 中断 B. 程序查询 C. 无条件传送 D. DMA

试题四 某系统由下图所示的冗余部件构成。若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( )。



A．$(1-R^3)(1-R^2)$　　B．$(1-(1-R)^3)(1-(1-R)^2)$
C．$(1-R^3)+(1-R^2)$　　D．$(1-(1-R)^3)+(1-(1-R)^2)$

试题五 已知数据信息为 16 位，最少应附加( )位校验位，才能实现海明码纠错。
A．3　　B．4　　C．5　　D．6

试题六 以下关于 Cache（高速缓冲存储器)的叙述中，不正确的是( )
A．Cache 的设置扩大了主存的容量　　B．Cache 的内容是主存部分内容的拷贝
C．Cache 的命中率并不随其容量增大线性地提高　　D．Cache 位于主存与 CPU 之间

试题七 HTTPS 使用( )协议对报文进行封装。
A．SSH　　B．SSL　　C．SHA-1　　D．SET

试题八 以下加密算法中适合对大量的明文消息进行加密传输的是( )。
A．RSA　　B．SHA-1　　C．MD5　　D．RC5

试题九 假定用户 A 、 B 分别在 I1 和 I2 两个 CA 处取得了各自的证书，下面( )是 A 、 B 互信的必要条件。
A．A 、 B 互换私钥　　B．A 、 B 互换公钥　　C．I1 、 I2 互换私钥　　D．I1 、 I2 互换公钥

试题一十 甲软件公司受乙企业委托安排公司软件设计师开发了信息系统管理软件，由于在委托开发合同中未对软件著作权归属作出明确的约定，所以该信息系统管理软件的著作权由( )享有。

A. 甲 B. 乙 C. 甲与乙共同 D. 软件设计师

试题一十一 根据我国商标法，下列商品中必须使用注册商标的是( )。

A. 医疗仪器 B. 墙壁涂料 C. 无糖食品 D. 烟草制品

试题一十二 甲、乙两人在同一天就同样的发明创造提交了专利申请，专利局将分别向各申请人通报有关情况，并提出多种可能采用的解决办法。下列说法中，不可能采用( )。

A. 甲、乙作为共同申请人 B. 甲或乙一方放弃权利并从另一方得到适当的补偿

C. 甲、乙都不授予专利权 D. 甲、乙都授予专利权

试题一十三 数字语音的采样频率定义为 8kHz，这是因为( )。

A. 语音信号定义的频率最高值为 4kHz B. 语音信号定义的频率最高值为 8kHz

C. 数字语音转输线路的带宽只有 8kHz D. 一般声卡的采样频率最高为每秒 8k 次

试题一十四 使用图像扫描仪以 300DPI 的分辨率扫描一幅 3×4 英寸的图片，可以得到( )像素的数字图像。

A. 300×300 B. 300×400 C. 900×4 D. 900×1200

试题一十五 (第 1 空)在采用结构化开发方法进行软件开发时，设计阶段接口设计主要依据需求分析阶段的( )。接口设计的任务主要是( )。

A. 数据流图 B. E-R 图 C. 状态-迁移图 D. 加工规格说明

试题一十六 (第 2 空)在采用结构化开发方法进行软件开发时，设计阶段接口设计主要依据需求分析阶段的( )。接口设计的任务主要是( )。

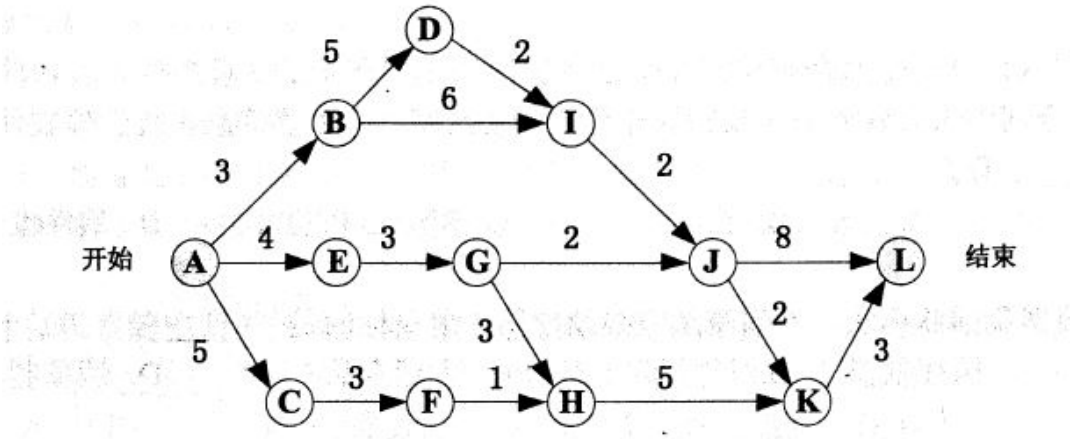
A. 定义软件的主要结构元素及其之间的关系

B. 确定软件涉及的文件系统的结构及数据库的表结构

C. 描述软件与外部环境之间的交互关系，软件内模块之间的调用关系

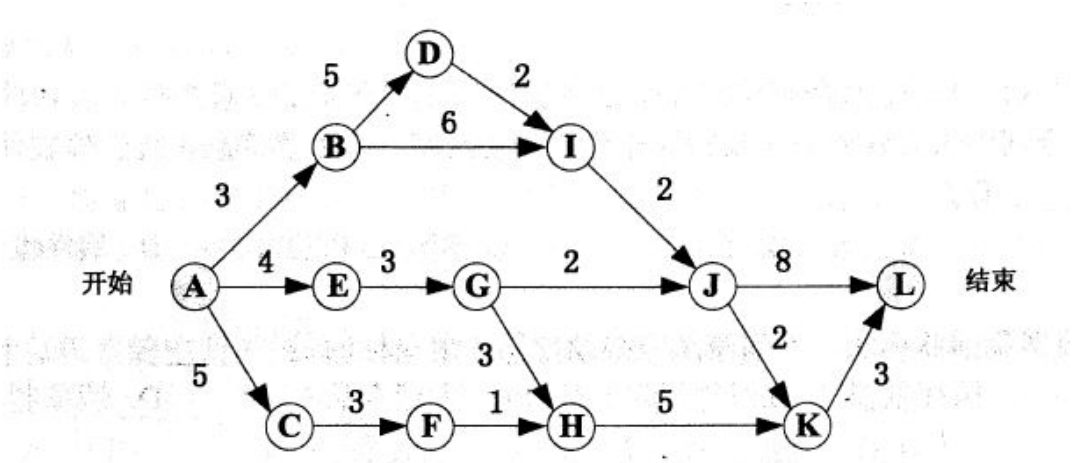
D. 确定软件各个模块内部的算法和数据结构

试题一十七 (第 1 空)某软件项目的活动图如下图所示，其中顶点表示项目里程碑，连接顶点的边表示包含的活动，边上的数字表示活动的持续时间(天)，则完成该项目的最长时间为( )天。活动 BD 和 HK 最早可以从第( )天开始。(活动 AB 、 AE 和 AC 最早从第 1 天开始)



A．17　　B．18　　C．19　　D．20

试题一十八 (第 2 空)某软件项目的活动图如下图所示，其中顶点表示项目里程碑，连接顶点的边表示包含的活动，边上的数字表示活动的持续时间(天)，则完成该项目的最长时间为( )天。活动 BD 和 HK 最早可以从第( )天开始。(活动 AB 、 AE 和 AC 最早从第 1 天开始)



A．3 和 10　　B．4 和 11　　C．3 和 9　　D．4 和 10

试题一十九 在进行软件开发时，采用无主程序员的开发小组，成员之间相互平等;而主程序员负责制的开发小组，由一个主程序员和若干成员组成，成员之间没有沟通。在一个由8名开发人员构成的小组中，无主程序员组和主程序员组的沟通路径分别是( )。

A．32和8　　B．32和7　　C．28和8　　D．28和7

试题二十 在高级语言源程序中，常需要用户定义的标识符为程序中的对象命名，常见的命名对象有( )

①关键字(或保留字)②变量③函数④数据类型⑤注释

A. ①②③　　B. ②③④　　C. ①③⑤　　D. ②④⑤

试题二十一 在仅由字符a、b构成的所有字符串中，其中以b结尾的字符串集合可用正则表达式为( )。

A．(b|ab)*b　　B．(ab*)*b　　C．a*b*b　　D．(a|b)*b

试题二十二 在以阶段划分的编译过程中，判断程序语句的形式是否正确属于( ) 阶段的工作。

A. 词法分析　　B. 语法分析　　C. 语义分析　　D. 代码生成

试题二十三 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。若计算机系统的字长为32 位，磁盘的容量为300GB ，物理块的大小为4MB ，那么位示图的大小需要( )个字。

A．1200　　B．2400　　C．6400　　D．9600

试题二十四 某系统中有3个并发进程竞争资源R，每个进程都需要5个R，那么至少有( )个R，才能保证系统不会发生死锁。

A．12　　B．13　　C．14　　D．15

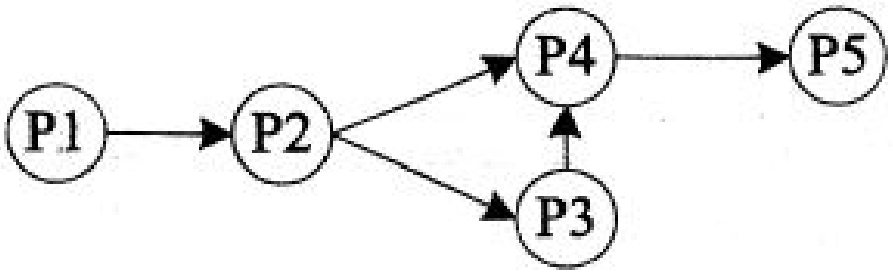
试题二十五 某计算机系统页面大小为4K ，进程的页面变换表如下所示。若进程的逻辑地址为2D16H 。该地址经过变换后，其物理地址应为( )

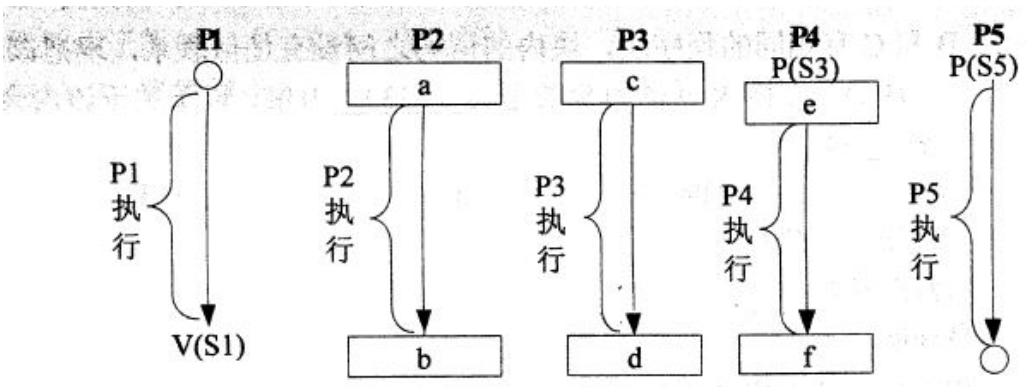
页号	物理块号
0	1
1	3
2	4
3	6

A．2048H　　B．4096H　　C．4D16H　　D．6D16H

试题二十六　(第 1 空)进程 P1 、 P2 、 P3 、 P4 和 P5 的前趋图如下所示:

若用 PV 操作控制进程 P1 、 P2 、 P3 、 P4 和 P5 并发执行的过程，需要设置 5 个信号量 S1 、 S2 、 S3 、 S4 和 S5 ，且信号量 S1~S5 的初值都等于零。如下的进程执行图中 a 和 b 处应分别填写(　)； c 和 d 处应分别填写(　)； e 和 f 处应分别填写(　) 。

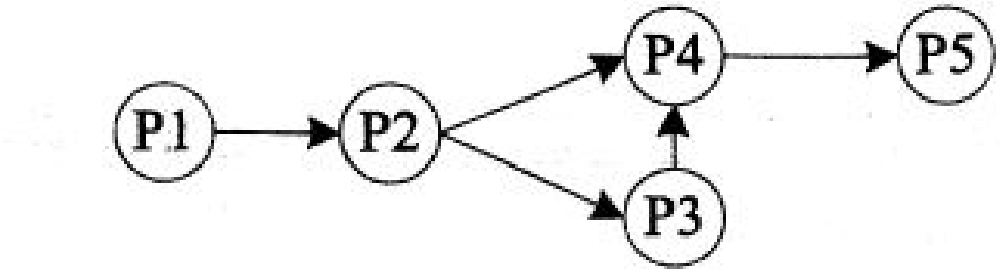


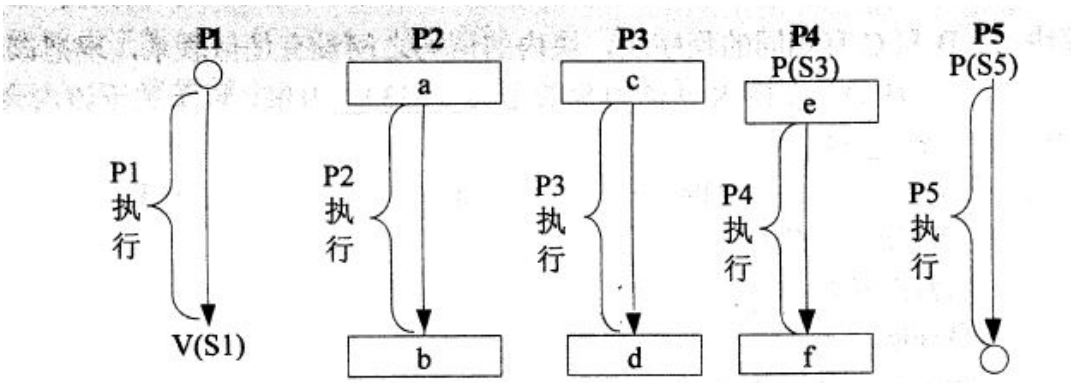


A．V(S1)和 P(S2)V(S3)　　B．P(S1)和 V(S2)V(S3)
C．V(S1)和 V(S2)V(S3)　　D．P(S1)和 P(S2)V(S3)

试题二十七 （第 2 空）进程 P1 、 P2 、 P3 、 P4 和 P5 的前趋图如下所示：

若用 PV 操作控制进程 P1 、 P2 、 P3 、 P4 和 P5 并发执行的过程，需要设置 5 个信号量 S1 、 S2 、 S3 、 S4 和 S5 ，且信号量 S1~S5 的初值都等于零。如下的进程执行图中 a 和 b 处应分别填写( )； c 和 d 处应分别填写( )； e 和 f 处应分别填写( ) 。

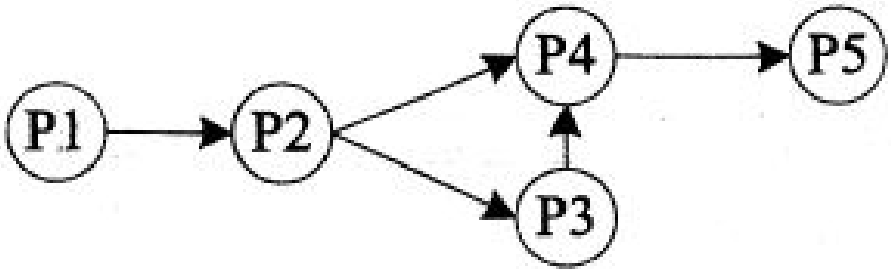


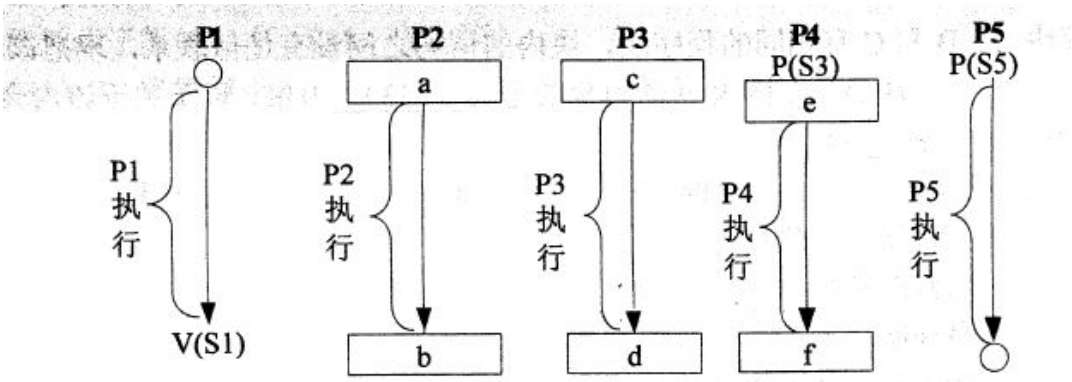


A．P(S2)和 P(S4)　　B．V(S2)和 P(S4)　　C．P(S2)和 V(S4)　　D．V(S2)和 V(S4)

试题二十八 （第 3 空）进程 P1 、 P2 、 P3 、 P4 和 P5 的前趋图如下所示：

若用 PV 操作控制进程 P1 、 P2 、 P3 、 P4 和 P5 并发执行的过程，需要设置 5 个信号量 S1 、 S2 、 S3 、 S4 和 S5 ，且信号量 S1~S5 的初值都等于零。如下的进程执行图中 a 和 b 处应分别填写( )； c 和 d 处应分别填写( )； e 和 f 处应分别填写( ) 。





A．P(S4)和 V(S5)　　B．V(S5)和 P(S4)　　C．V(S4)和 P(S5)　　D．V(S4)和 V(S5)

试题二十九　以下关于螺旋模型的叙述中，不正确的是(　)。

A. 它是风险驱动的，要求开发人员必须具有丰富的风险评估知识和经验
B. 它可以降低过多测试或测试不足带来的风险
C. 它包含维护周期，因此维护和开发之间没有本质区别
D. 它不适用于大型软件开发

试题三十　以下关于极限编程(XP) 中结对编程的叙述中，不正确的是(　)。
A. 支持共同代码拥有和共同对系统负责　　B. 承担了非正式的代码审查过程
C. 代码质量更高　　D. 编码速度更快

试题三十一　以下关于 C/S（客户机/服务器)体系结构的优点的叙述中，不正确的是(　)。

A. 允许合理地划分三层的功能，使之在逻辑上保持相对独立性
B. 允许各层灵活地选用平台和软件
C. 各层可以选择不同的开发语言进行并行开发
D. 系统安装、修改和维护均只在服务器端进行

试题三十二　在设计软件的模块结构时，(　)不能改进设计质量。
A. 尽量减少高扇出结构　　B. 模块的大小适中　　C. 将具有相似功能的模块合并　　D. 完善模块的功能

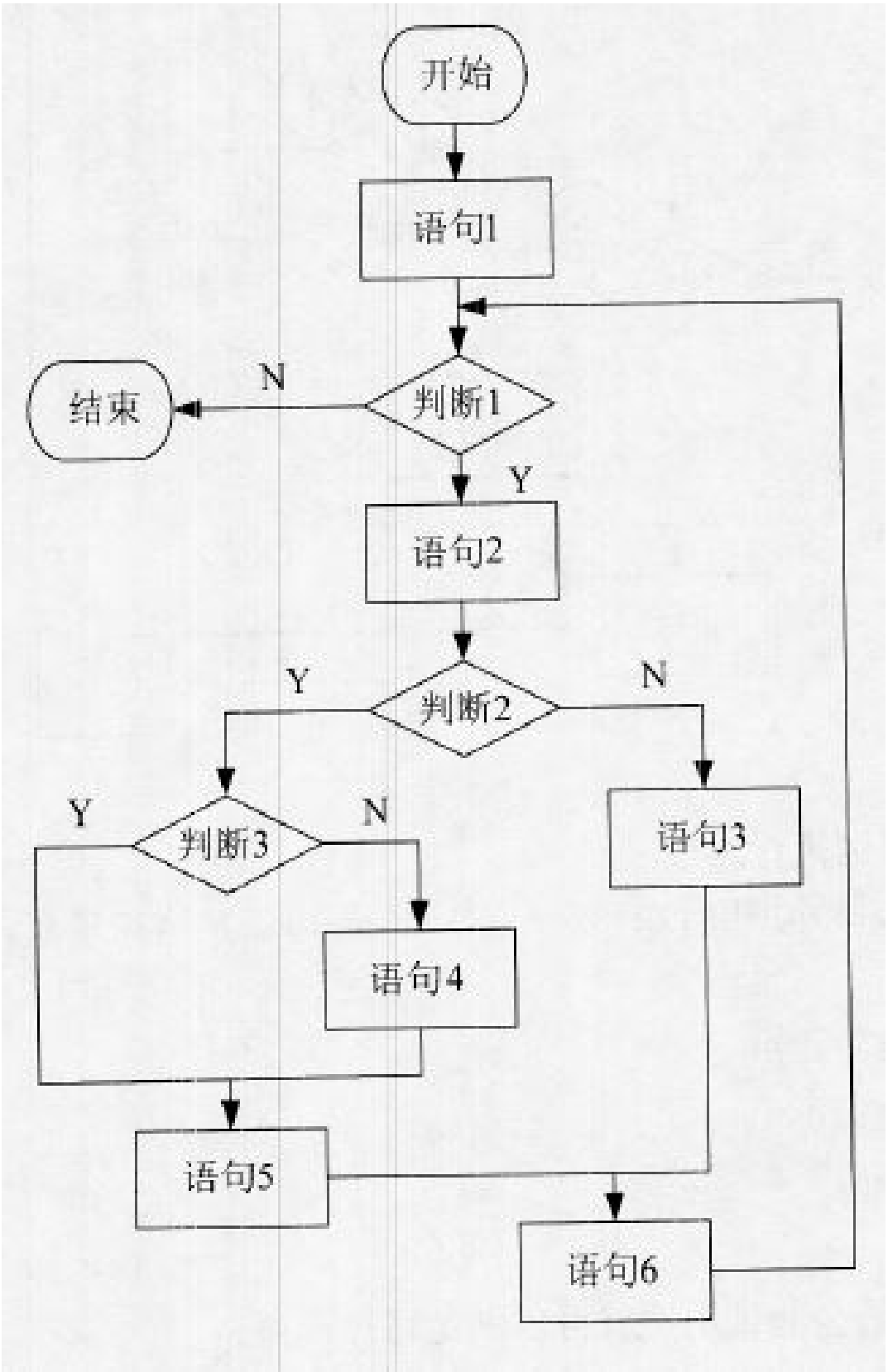
试题三十三 （第 1 空）模块 A 、 B 和 C 有相同的程序块，块内的语句之间没有任何联系，现把该程序块取出来，形成新的模块 D ，则模块 D 的内聚类型为( )内聚。以下关于该内聚类型的叙述中，不正确的是( )。

A. 巧合　　B. 逻辑　　C. 时间　　D. 过程

试题三十四 （第 2 空）模块 A 、 B 和 C 有相同的程序块，块内的语句之间没有任何联系，现把该程序块取出来，形成新的模块 D ，则模块 D 的内聚类型为( )内聚。以下关于该内聚类型的叙述中，不正确的是( )。

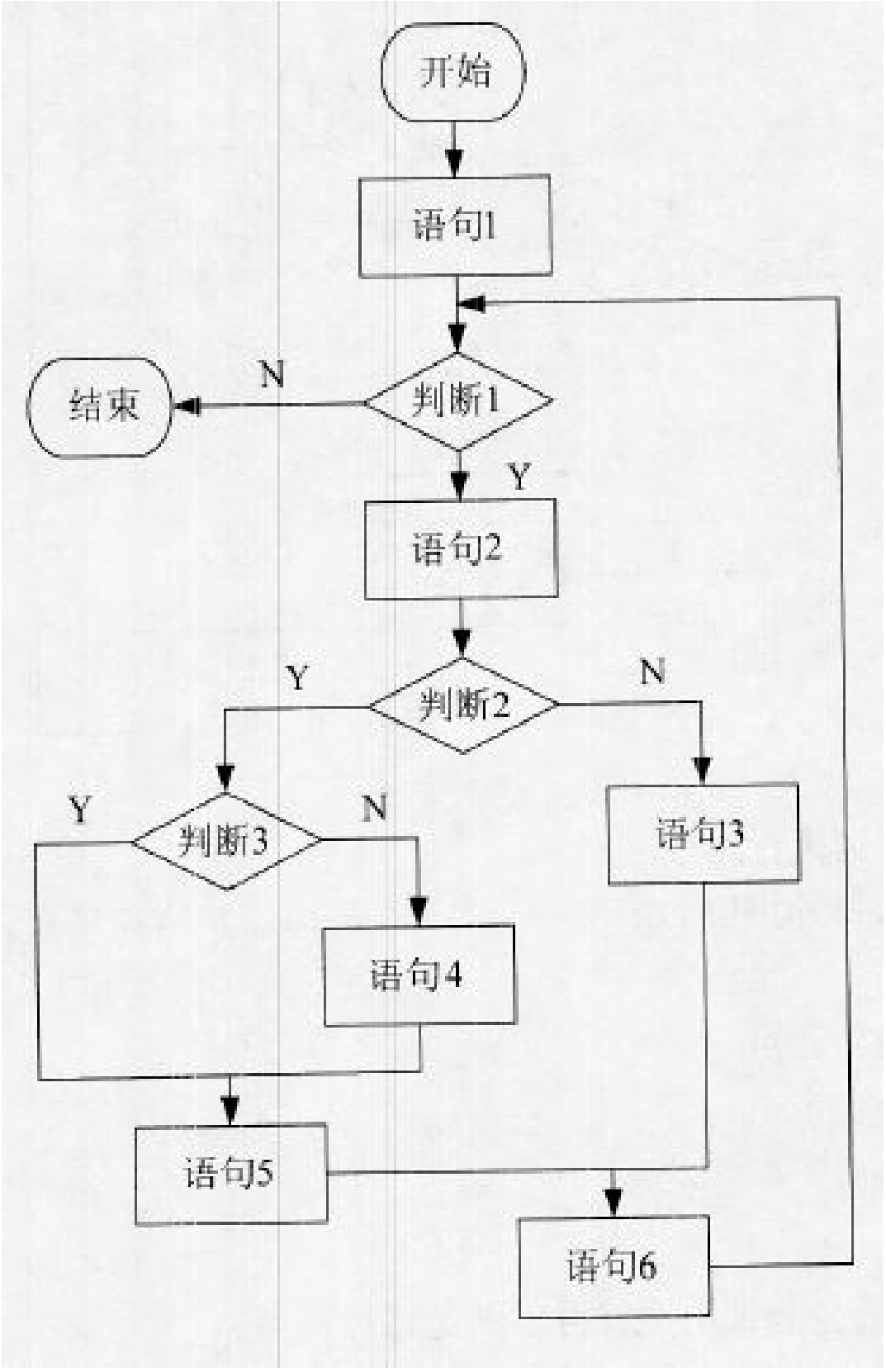
A. 具有最低的内聚性　　B. 不易修改和维护　　C. 不易理解　　D. 不影响模块间的耦合关系

试题三十五 （第 1 空）对下图所示的程序流程图进行语句覆盖测试和路径覆盖测试，至少需要( )个测试用例。采用 McCabe 度量法计算其环路复杂度为( )。



A．2和3　　B．2和4　　C．2和5　　D．2和6

试题三十六 （第 2 空）对下图所示的程序流程图进行语句覆盖测试和路径覆盖测试，至少需要( )个测试用例。采用 McCabe 度量法计算其环路复杂度为( )。



A．1　　B．2　　C．3　　D．4

试题三十七 （第 1 空）在面向对象方法中，两个及以上的类作为一个类的父类时，称为( )，使用它可能造成子类中存在( )的成员。

A. 多重继承　　B. 多态　　C. 封装　　D. 层次继承

试题三十八 （第 2 空）在面向对象方法中，两个及以上的类作为一个类的父类时，称为( )，使用它可能造成子类中存在( )的成员。

A. 动态　　B. 私有　　C. 公共　　D. 二义性

试题三十九 采用面向对象方法进行软件开发，在分析阶段，架构师主要关注系统的( )。

A. 技术　　B. 部署　　C. 实现　　D. 行为

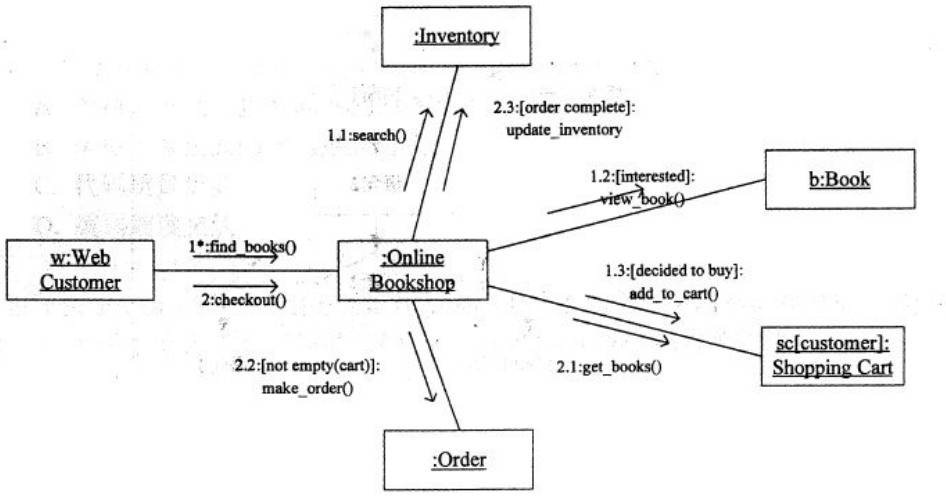
试题四十 在面向对象方法中，多态指的是( )。

A. 客户类无需知道所调用方法的特定子类的实现　　B. 对象动态地修改类

C. 一个对象对应多张数据库表　　D. 子类只能够覆盖父类中非抽象的方法

试题四十一 （第 1 空）

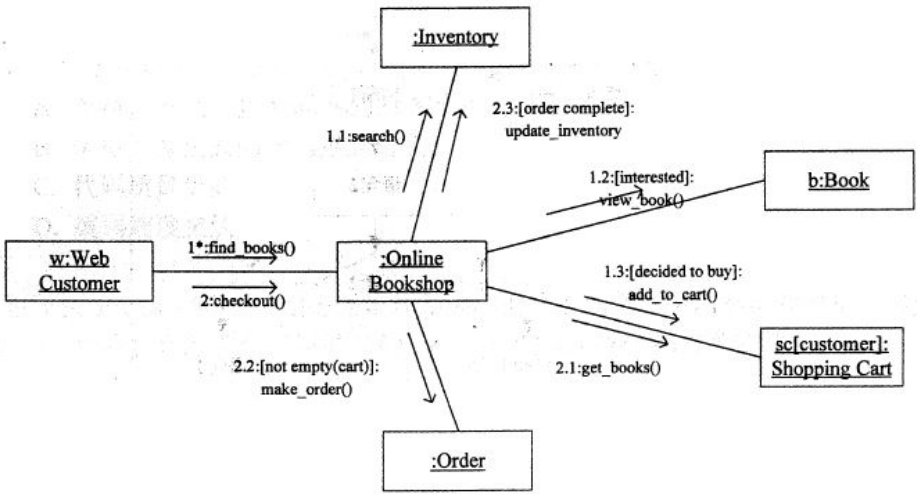
● 以下 UML 图是__（41）__，图中 :Order 和 b:Book 表示__（42）__，1*:find_books() 和 1.1:search() 表示__（43）__。



A. 序列图　　B. 状态图　　C. 通信图　　D. 活动图

试题四十二 （第 2 空）

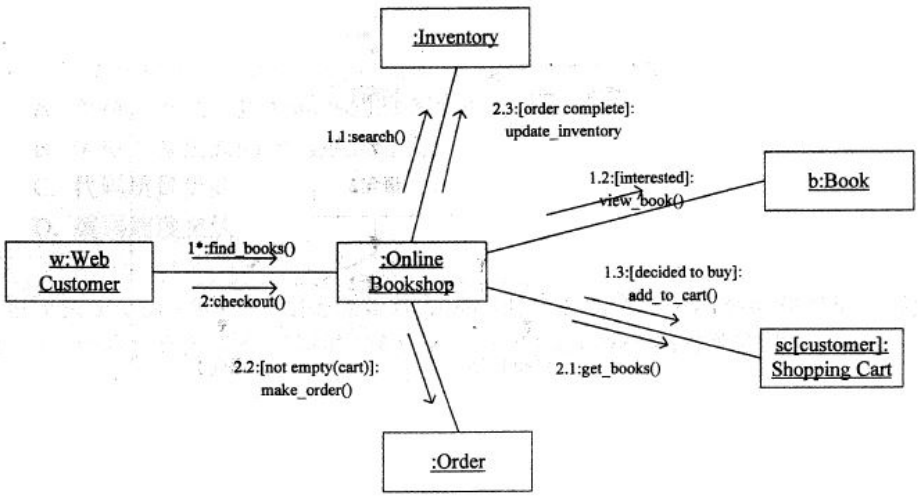
● 以下 UML 图是__（41）__，图中 :Order 和 b:Book 表示__（42）__，1*:find_books() 和 1.1:search() 表示__（43）__。



A. 类　　B. 对象　　C. 流名称　　D. 消息

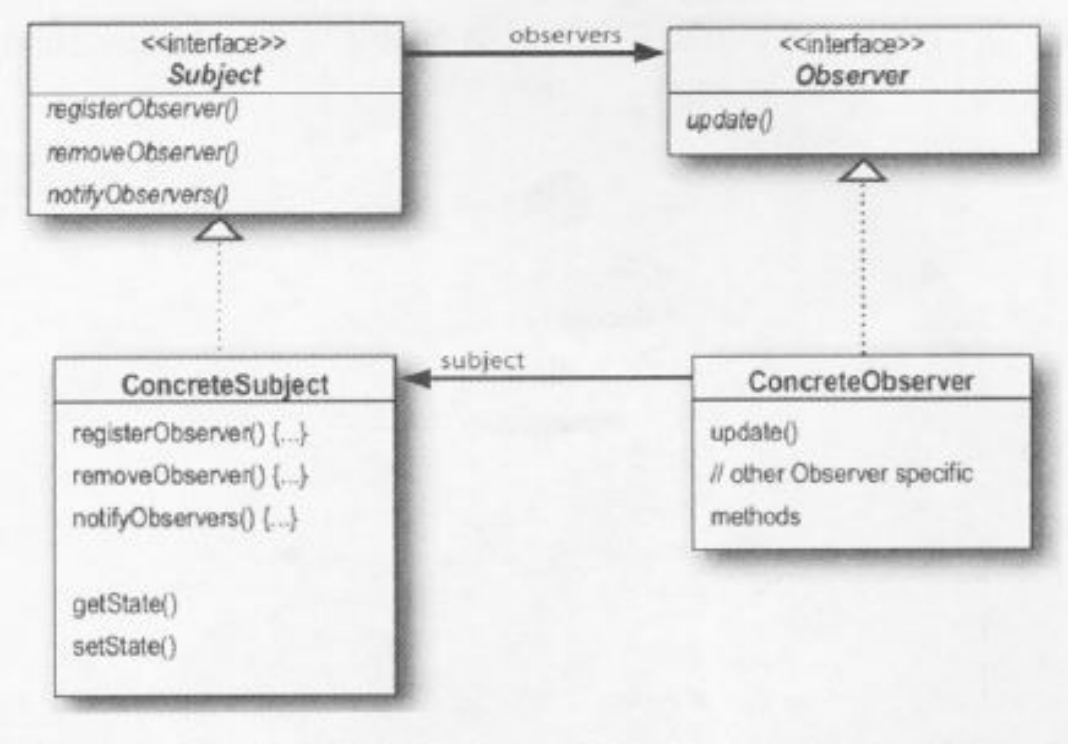
试题四十三　（第 3 空）

● 以下 UML 图是__（41）__，图中 :Order 和 b:Book 表示__（42）__，1*:find_books() 和 1.1:search() 表示__（43）__。



A. 类　　B. 对象　　C. 流名称　　D. 消息

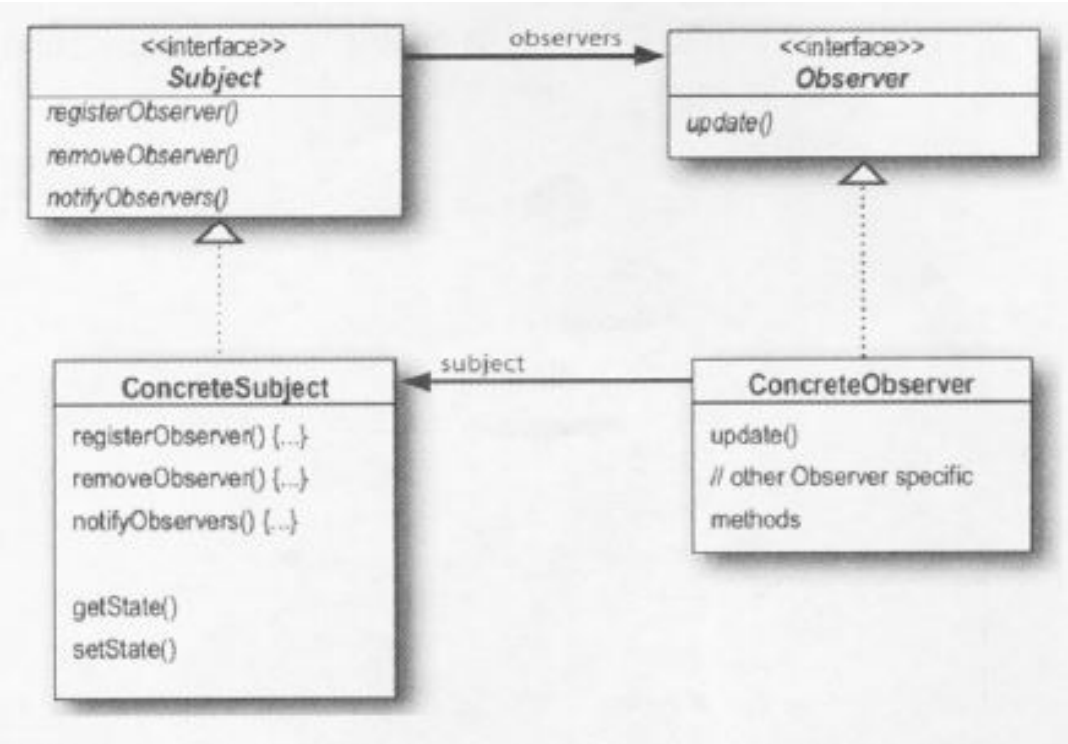
试题四十四　(第 1 空)下图所示为观察者(Observer)模式的抽象示意图，其中(44)知道其观察者，可以有任何多个观察者观察同一个目标；提供注册和删除观察者对象的接口。此模式体现的最主要的特征是(45)。



A. Subject　　B. Observer

C. ConcreteSubject　　D. ConcreteObserver

试题四十五　(第 2 空)下图所示为观察者(Observer)模式的抽象示意图，其中(44)知道其观察者，可以有任何多个观察者观察同一个目标；提供注册和删除观察者对象的接口。此模式体现的最主要的特征是(45)。



A. 类应该对扩展开放，对修改关闭　　B. 使所要交互的对象尽量松耦合
C. 组合优先于继承使用　　D. 仅与直接关联类交互

试题四十六　(第 1 空)装饰器（Decorator）模式用于（);外观（Facade）模式用于( )。

①将一个对象加以包装以给客户提供其希望的另外一个接口

②将一个对象加以包装以提供一些额外的行为

③将一个对象加以包装以控制对这个对象的访问

④将一系列对象加以包装以简化其接口

A. ①　　B. ②　　C. ③　　D. ④

试题四十七　(第 2 空)装饰器（Decorator）模式用于（);外观（Facade）模式用于( )。

①将一个对象加以包装以给客户提供其希望的另外一个接口

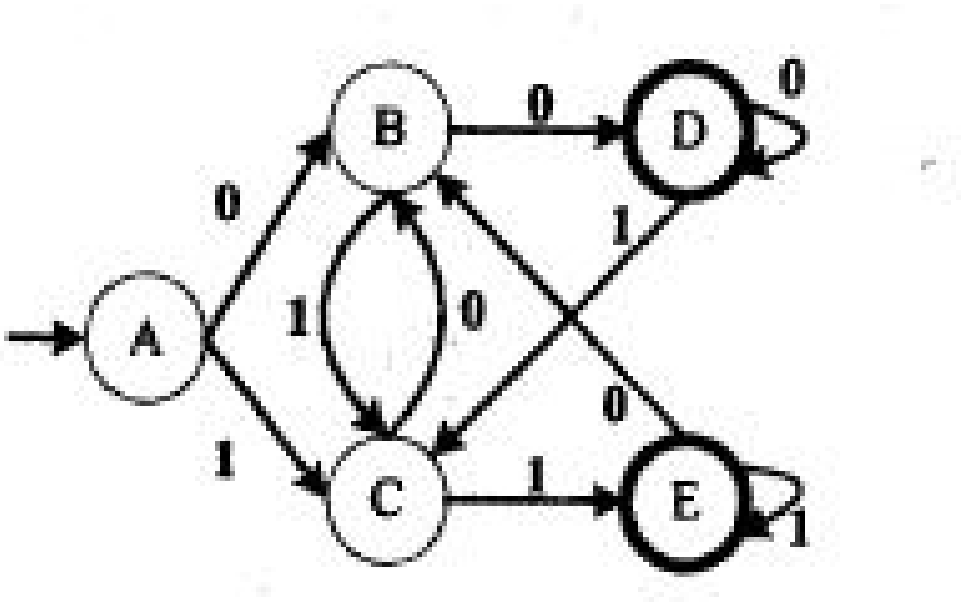
②将一个对象加以包装以提供一些额外的行为

③将一个对象加以包装以控制对这个对象的访问

④将一系列对象加以包装以简化其接口

A. ①　　B. ②　　C. ③　　D. ④

试题四十八　某确定的有限自动机（DFA）的状态转换图如下图所示（A 是初态，D、E 是终态)，则该 DFA 能识别（)



A．00110　　B．10101　　C．11100　　D．11001

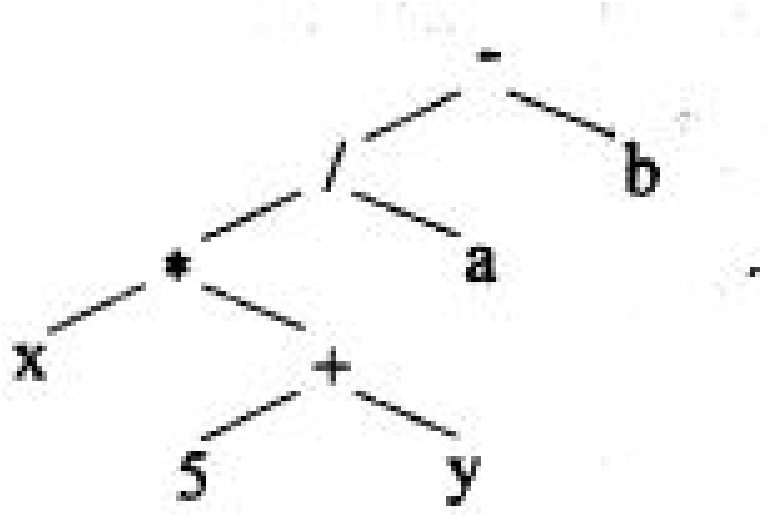
试题四十九 函数 main()、 f()的定义如下所示，调用函数们 f()时，第一个参数采用传值(callbyvalue) 方式，第二个参数采用传引用(callbyreference) 方式， main()函数中"print(x)〞执行后输出的值为( )。

```
main( )
int x = 5;
f(x+1, x);
print(x);
```

```
f(int x, int &a)
x = x*x -1;
a = x + a;
return;
```

A．11　　B．40　　C．45　　D．70

试题五十 下图为一个表达式的语法树，该表达式的后缀形式为 ( )



A．x5y+*a/b-　　B．x5yab*+/-　　C．-/*x+5yab　　D．x5*y+a/b-

试题五十一 (第 1 空)若事务 T_1 对数据 D_1 加了共享锁，事务 T_2、 T_3 分别对数据 D_2、 D_3 加了排它锁，则事务 T_1 对数据( ) ；事务 T_2 对数据( )。

A．D_2、 D_3 加排它锁都成功　　B．D_2、 D_3 加共享锁都成功

C．D_2 加共享锁成功， D_3 加排它锁失败　　D．D_2、 D_3 加排它锁和共享锁都失败

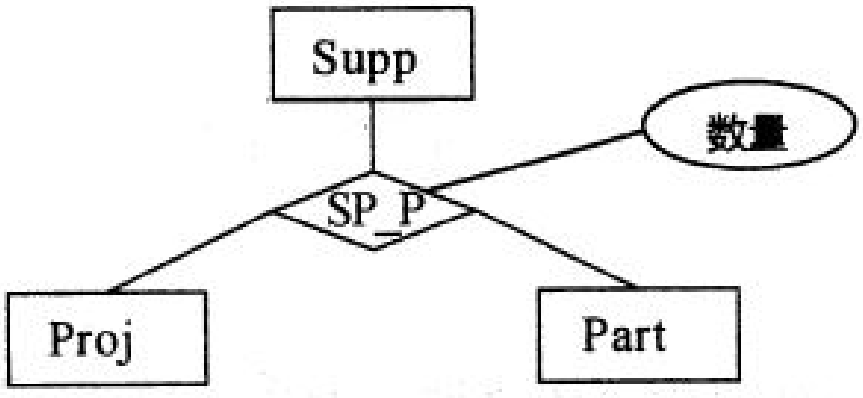
试题五十二 (第 2 空)若事务 T_1 对数据 D_1 加了共享锁，事务 T_2、 T_3 分别对数据 D_2、 D_3 加了排它锁，则事务 T_1 对数据( ) ；事务 T_2 对数据( )。

A．D_1、D_3加共享锁都失败　　B．D_1、D_3加共享锁都成功
C．D_1加共享锁成功，D_3加排它锁失败　　D．D_1加排它锁成功，D_3加共享锁失败

试题五十三　假设关系 R,U= {A_1,A_2, A_3}，F = {$A_1A_3 \rightarrow A_2$,$A_1A_2 \rightarrow A_3$}，则关系 R 的各候选关键字中必定含有属性(　)。
A．A_1　　B．A_2　　C．A_3　　D．A_2A_3

试题五十四　(第 1 空)在某企业的工程项目管理系统的数据库中供应商关系 Supp、项目关系 Proj 和零件关系 Part 的 E-R 模型和关系模式如下:

Supp(供应商号,供应商名,地址,电话)
Proj(项目号,项目名,负责人,电话)
Part(零件号,零件名)
其中，每个供应商可以为多个项目供应多种零件，每个项目可由多个供应商供应多种零件。
SP_P 需要生成一个独立的关系模式，其联系类型为(　)
给定关系模式 SP_P(供应商号,项目号,零件号,数量)查询至少供应了 3 个项目(包含 3 项)的供应商，输出其供应商号和供应零件数量的总和，并按供应商号降序排列。
SELECT 供应商号， SUM(数量) FROM(　)
GROUPBY 供应商号
(　)
ORDERBY 供应商号 DESC;



A．*:*:*　　B．1:*:*　　C．1:1:*　　D．1:1:1

试题五十五　(第 2 空)在某企业的工程项目管理系统的数据库中供应商关系 Supp、项目关系 Proj 和零件关系 Part 的 E-R 模型和关系模式如下:

Supp(供应商号,供应商名,地址,电话)

Proj(项目号,项目名,负责人,电话)

Part(零件号,零件名)

其中，每个供应商可以为多个项目供应多种零件，每个项目可由多个供应商供应多种零件。

SP_P 需要生成一个独立的关系模式，其联系类型为( )

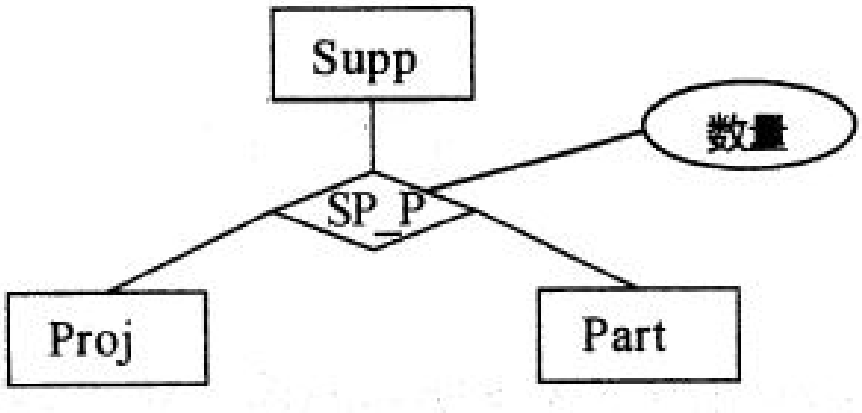
给定关系模式 SP_P(供应商号,项目号,零件号,数量)查询至少供应了 3 个项目(包含 3 项)的供应商，输出其供应商号和供应零件数量的总和，并按供应商号降序排列。

SELECT 供应商号， SUM(数量) FROM( )

GROUPBY 供应商号

( )

ORDERBY 供应商号 DESC;



A. Supp　　B. Proj　　C. Part　　D. SP_P

试题五十六　(第 3 空)在某企业的工程项目管理系统的数据库中供应商关系 Supp、项目关系 Proj 和零件关系 Part 的 E-R 模型和关系模式如下:

Supp(供应商号,供应商名,地址,电话)

Proj(项目号,项目名,负责人,电话)

Part(零件号,零件名)

其中，每个供应商可以为多个项目供应多种零件，每个项目可由多个供应商供应多种零件。

SP_P 需要生成一个独立的关系模式，其联系类型为( )

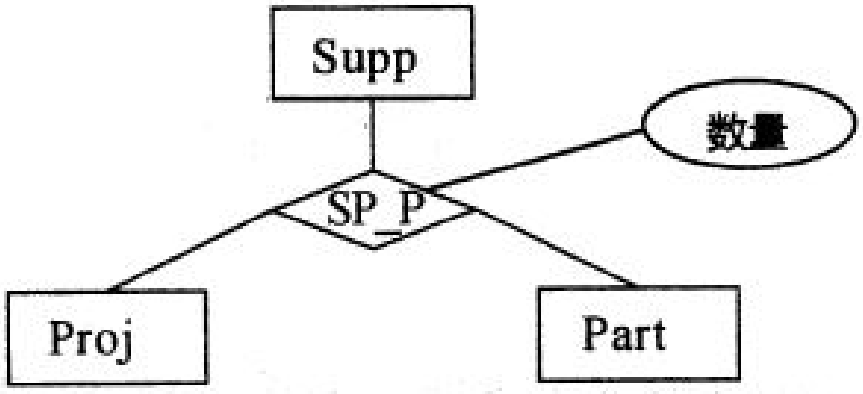
给定关系模式 SP_P(供应商号,项目号,零件号,数量)查询至少供应了 3 个项目(包含 3 项)的供应商，输出其供应商号和供应零件数量的总和，并按供应商号降序排列。

SELECT 供应商号， SUM(数量) FROM( )

GROUPBY 供应商号

( )

ORDERBY 供应商号 DESC;



A. HAVINGCOUNT(项目号)>2
B. WHERECOUNT(项目号)>2
C. HAVINGCOUNT(DISTINCT(项目号))>2
D. WHERECOUNT(DISTINCT(项目号))>3

试题五十七 以下关于字符串的叙述中，正确的是( )。
A. 包含任意个空格字符的字符串称为空串 B. 字符串不是线性数据结构
C. 字符串的长度是指串中所含字符的个数 D. 字符串的长度是指串中所含非空格字符的个数

试题五十八 已知栈 S 初始为空，用 I 表示入栈、 O 表示出栈，若入栈序列为 $a_1a_2a_3a_4a_5$，则通过栈 S 得到出栈序列 $a_2a_4a_5a_3a_1$ 的合法操作序列( )。
A. IIOIIOIOOO B. IOIOIOIOIO C. IOOIIOIOIO D. IIOOIOIOOO

试题五十九 某二叉树的先序遍历序列为 ABCDEF ，中序遍历序列为 BADCFE ，则该二叉树的高度(即层数)为( )。
A. 3 B. 4 C. 5 D. 6

试题六十 对于 n 个元素的关键字序列$\{k_1,k_2,\ldots k_n\}$，当且仅当满足关系 $k_i \leqslant k_{2i}$ 且 $ki \leqslant k_{2i+1}\{i=1.2...[n/2]\}$ 时称其为小根堆(小顶堆)。以下序列中，( )不是小根堆。
A. 16,25,40,55,30,50,45 B. 16,40,25,50,45,30,55
C. 16,25,39.,41,45,43,50 D. 16,40,25,53,39,55,45

试题六十一 在 12 个互异元素构成的有序数组 a[1..12] 中进行二分查找(即折半查找，向下取整)，若待查找的元素正好等于 a[9]，则在此过程中，依次与数组中的( )比较后，查找成功结束。

A．a[6]、a[7]、a[8]、a[9]　　B．a[6]、a[9]

C．a[6]、a[7]、a[9]　　D．a[6]、a[8]、a[9]

试题六十二 (第 1 空)某汽车加工工厂有两条装配线 L1 和 L2，每条装配线的工位数均为 n(S_{ij}， i=1 或 2， j= 1， 2，...， n)，两条装配线对应的工位完成同样的加工工作，但是所需要的时间可能不同(a_{ij}， i=1 或 2， j=1， 2，...， n)。汽车底盘开始到进入两条装配线的时间（e_1， e_2）以及装配后到结束的时间(x_1x_2)也可能不相同。从一个工位加工后流到下一个工位需要迁移时间(t_{ij}， i=1 或 2， j=2，...n)。现在要以最快的时间完成一辆汽车的装配，求最优的装配路线。

分析该问题，发现问题具有最优子结构。以 L1 为例，除了第一个工位之外，经过第 j 个工位的最短时间包含了经过 L1 的第 j-1 个工位的最短时间或者经过 L2 的第 j-1 个工位的最短时间，如式( )。装配后到结束的最短时间包含离开 L1 的最短时间或者离开 L2 的最短时间如式( )。

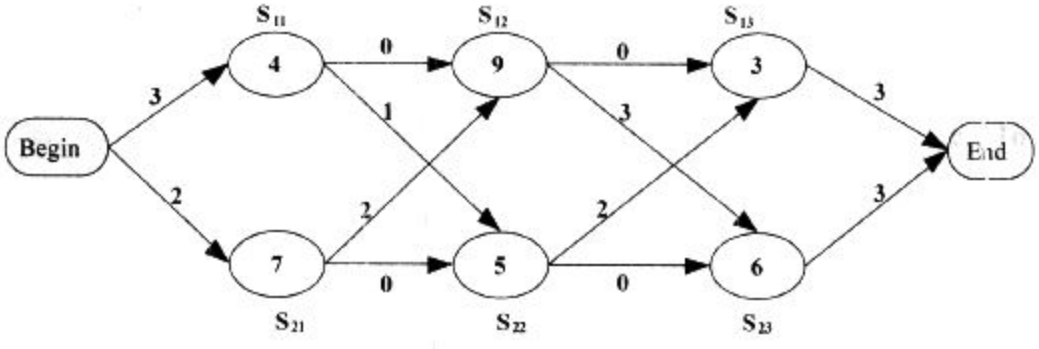
由于在求解经过 L1 和 L2 的第 j 个工位的最短时间均包含了经过 L1 的第 j-1 个工位的最短时间或者经过 L2 的第 j-1 个工位的最短时间，该问题具有重复子问题的性质，故采用迭代方法求解。

该问题采用的算法设计策略是( )，算法的时间复杂度为( )

以下是一个装配调度实例，其最短的装配时间为( )，装配路线为( )

$$f_{1,j}=\begin{cases} e_1+a_{1,j} & \text{若} j=1 \\ \min\left(f_{1,j-1}+a_{1,j}+t_{1,j-1},f_{2,j-1}+a_{1,j}+t_{2,j-1}\right) & \text{其他} \end{cases} \quad (1)$$

$$f_{min}=\min\left(f_{1,n}+x_1,f_{2,n}+x_2\right) \quad (2)$$



A. 分治　　B. 动态规划　　C. 贪心　　D. 回溯

试题六十三　(第 2 空)某汽车加工工厂有两条装配线 L1 和 L2，每条装配线的工位数均为 n(S_{ij}， i=1 或 2， j= 1， 2，...， n)，两条装配线对应的工位完成同样的加工工作，但是所需要的时间可能不同(a_{ij}， i=1 或 2， j=1， 2，...， n)。汽车底盘开始到进入两条装配线的时间（e_1， e_2）以及装配后到结束的时间(x_1x_2)也可能不相同。从一个工位加工后流到下一个工位需要迁移时间(t_{ij}， i=1 或 2， j=2，...n)。现在要以最快的时间完成一辆汽车的装配，求最优的装配路线。

分析该问题，发现问题具有最优子结构。以 L1 为例，除了第一个工位之外，经过第 j 个工位的最短时间包含了经过 L1 的第 j-1 个工位的最短时间或者经过 L2 的第 j-1 个工位的最短时间，如式(　)。装配后到结束的最短时间包含离开 L1 的最短时间或者离开 L2 的最短时间如式(　)。

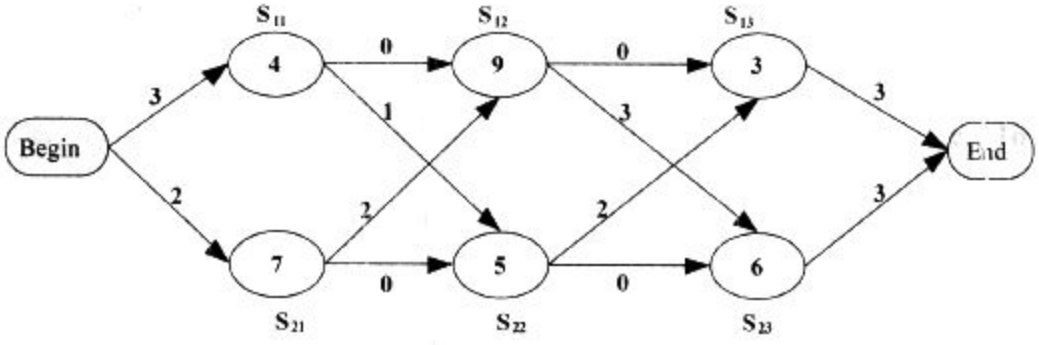
由于在求解经过 L1 和 L2 的第 j 个工位的最短时间均包含了经过 L1 的第 j-1 个工位的最短时间或者经过 L2 的第 j-1 个工位的最短时间，该问题具有重复子问题的性质，故采用迭代方法求解。

该问题采用的算法设计策略是(　)，算法的时间复杂度为(　)

以下是一个装配调度实例，其最短的装配时间为(　)，装配路线为(　)

$$f_{1,j}=\begin{cases} e_1+a_{1,j} & 若j=1 \\ \min\left(f_{1,j-1}+a_{1,j}+t_{1,j-1},f_{2,j-1}+a_{1,j}+t_{2,j-1}\right) & 其他 \end{cases} \quad (1)$$

$$f_{min}=\min\left(f_{1,n}+x_1,f_{2,n}+x_2\right) \quad (2)$$



A. θ(lgn)　　B. θ(n)　　C. θ(n^2)　　D. θ(nlgn)

试题六十四　(第 3 空)某汽车加工工厂有两条装配线 L1 和 L2，每条装配线的工位数均为 n(S_{ij}， i=1 或 2， j= 1， 2，...， n)，两条装配线对应的工位完成同样的加工工作，

但是所需要的时间可能不同(a_{ij}， i=1 或 2， j=1， 2，...， n)。汽车底盘开始到进入两条装配线的时间（e_1， e_2）以及装配后到结束的时间(x_1x_2)也可能不相同。从一个工位加工后流到下一个工位需要迁移时间(t_{ij}， i=1 或 2， j=2，...n)。现在要以最快的时间完成一辆汽车的装配，求最优的装配路线。

分析该问题，发现问题具有最优子结构。以 L1 为例，除了第一个工位之外，经过第 j 个工位的最短时间包含了经过 L1 的第 j-1 个工位的最短时间或者经过 L2 的第 j-1 个工位的最短时间，如式( )。装配后到结束的最短时间包含离开 L1 的最短时间或者离开 L2 的最短时间如式( )。

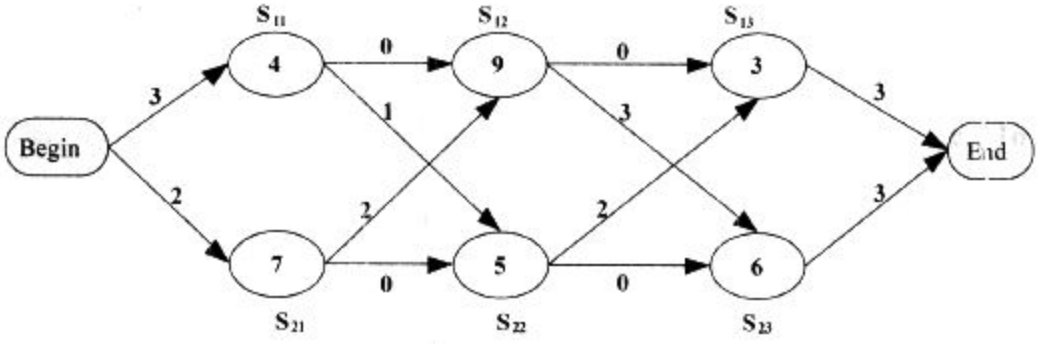
由于在求解经过 L1 和 L2 的第 j 个工位的最短时间均包含了经过 L1 的第 j-1 个工位的最短时间或者经过 L2 的第 j-1 个工位的最短时间，该问题具有重复子问题的性质，故采用迭代方法求解。

该问题采用的算法设计策略是( )，算法的时间复杂度为( )

以下是一个装配调度实例，其最短的装配时间为( )，装配路线为( )

$$f_{1,j}=\begin{cases} e_1+a_{1,j} & \text{若} j=1 \\ \min\left(f_{1,j-1}+a_{1,j}+t_{1,j-1}, f_{2,j-1}+a_{1,j}+t_{2,j-1}\right) & \text{其他} \end{cases} \quad (1)$$

$$f_{min}=\min\left(f_{1,n}+x_1, f_{2,n}+x_2\right) \quad (2)$$



A．21　　B．23　　C．20　　D．26

试题六十五　（第 4 空）某汽车加工工厂有两条装配线 L1 和 L2，每条装配线的工位数均为 n(S_{ij}， i=1 或 2， j= 1， 2，...， n)，两条装配线对应的工位完成同样的加工工作，但是所需要的时间可能不同(a_{ij}， i=1 或 2， j=1， 2，...， n)。汽车底盘开始到进入两条装配线的时间（e_1， e_2）以及装配后到结束的时间(x_1x_2)也可能不相同。从一个工位加工后流到下一个工位需要迁移时间(t_{ij}， i=1 或 2， j=2，...n)。现在要以最快的时间完成一辆汽车的装配，求最优的装配路线。

分析该问题，发现问题具有最优子结构。以 L1 为例，除了第一个工位之外，经过第 j 个工位的最短时间包含了经过 L1 的第 j-1 个工位的最短时间或者经过 L2 的第 j-1 个工位的最短时间，如式( )。装配后到结束的最短时间包含离开 L1 的最短时间或者离开 L2 的最短时间如式( )。

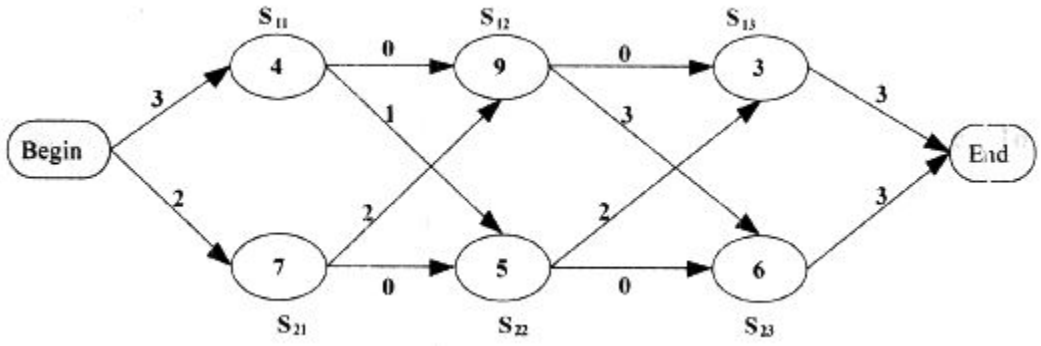
由于在求解经过 L1 和 L2 的第 j 个工位的最短时间均包含了经过 L1 的第 j-1 个工位的最短时间或者经过 L2 的第 j-1 个工位的最短时间，该问题具有重复子问题的性质，故采用迭代方法求解。

该问题采用的算法设计策略是( )，算法的时间复杂度为( )

以下是一个装配调度实例，其最短的装配时间为( )，装配路线为( )

$$f_{1,j}=\begin{cases} e_1+a_{1,j} & \text{若} j=1 \\ \min\left(f_{1,j-1}+a_{1,j}+t_{1,j-1},f_{2,j-1}+a_{1,j}+t_{2,j-1}\right) & \text{其他} \end{cases} \quad (1)$$

$$f_{min}=\min\left(f_{1,n}+x_1,f_{2,n}+x_2\right) \quad (2)$$



A. S11→S12→S13　　B. S11→S22→S13　　C. S21→S12→S23　　D. S21→S22→S23

试题六十六　在浏览器地址栏输入一个正确的网址后，本地主机将首先在( )查询该网址对应的 IP 地址。

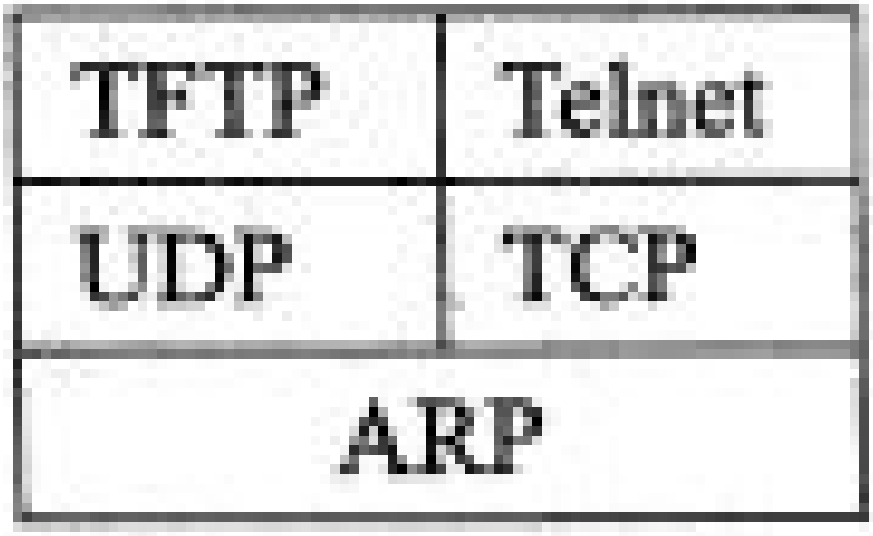
A. 本地 DNS 缓存　　B. 本机 hosts 文件　　C. 本地 DNS 服务器　　D. 根域名服务器

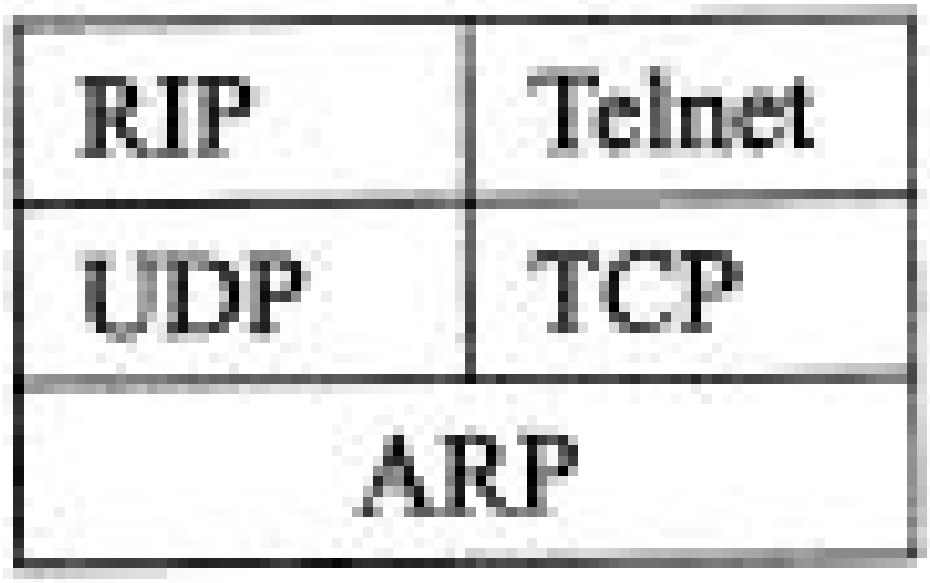
试题六十七　下面关于 Linux 目录的描述中，正确的是( )。

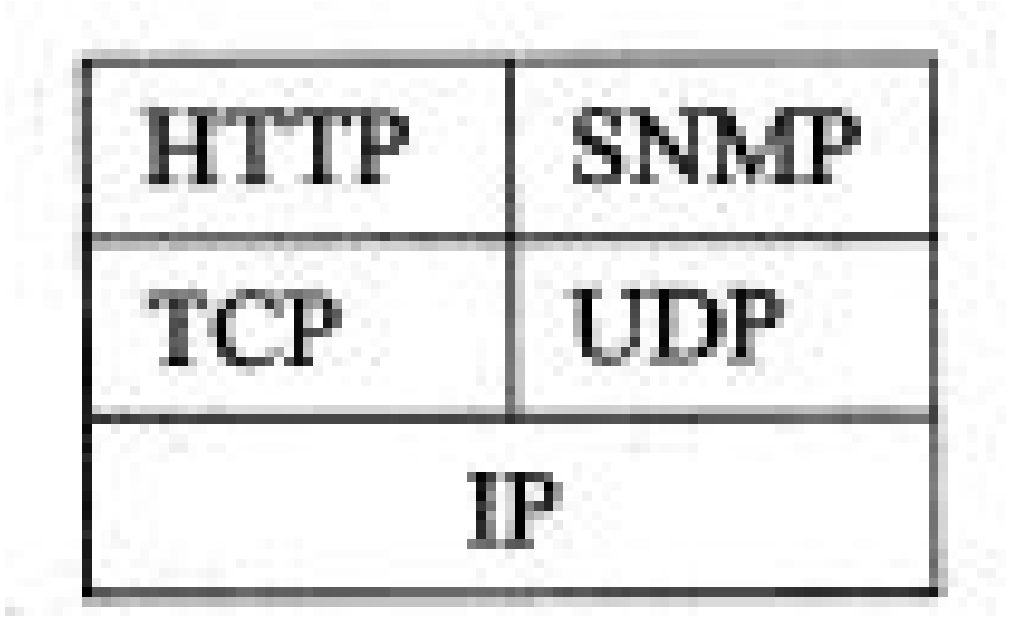
A. Linux 只有一个根目录，用 ″ /root ″表示
B. Linux 中有多个根目录，用″/″加相应目录名称表示
C. Linux 中只有一个根目录，用″/″表示
D. Linux 中有多个根目录，用相应目录名称表示

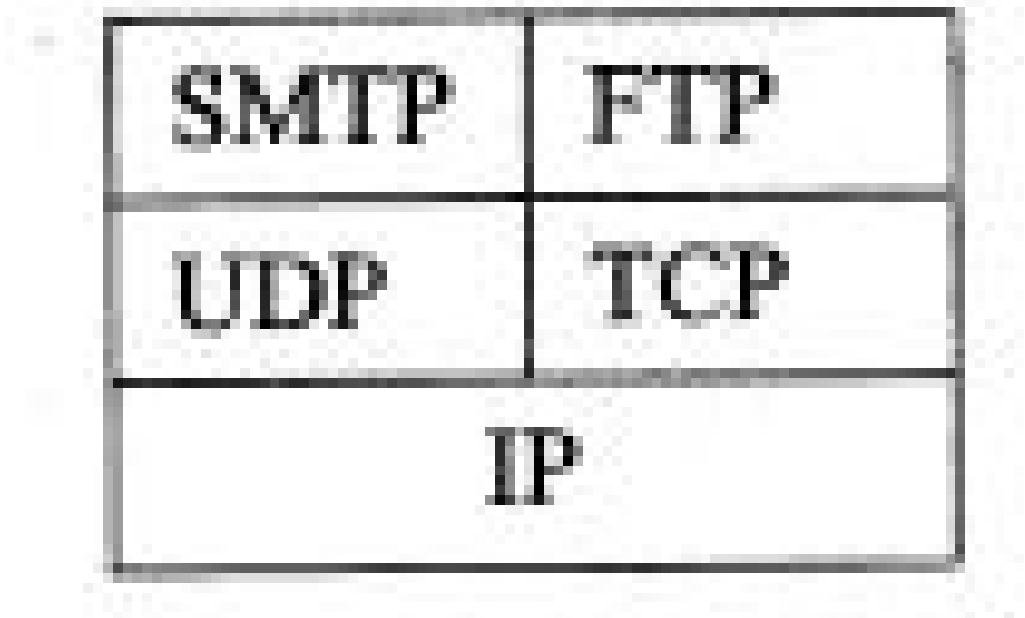
试题六十八 以下关于 TCP/IP 协议栈中协议和层次的对应关系正确的是( )。

A. B. C. D.









试题六十九 在异步通信中，每个字符包含 1 位起始位、 7 位数据位和 2 位终止位，若每秒钟传送 500 个字符，则有效数据速率为( )。

A．500b/s B．700b/s C．3500b/s D．5000b/s

试题七十 以下路由策略中，依据网络信息经常更新路由的是( )。

A. 静态路由 B. 洪泛式 C. 随机路由 D. 自适应路由

试题七十一 (第1空)Thebeautyofsoftwareisinitsfunction， initsinternalstructure ， andinthewayinwhichitiscreatedbyateam. Toauser， aprogramwithjusttheright featurespresentedthroughanintuitiveand( )interfaceisbeautiful.Toasoftware designer， aninternalstructurethatispartitionedinasimpleandintuitivemanner， andthatminimizesinternalcouplingisbeautiful.Todevelopersandmanagers ， a motivatedteamofdevelopersmakingsignificantprogresseveryweek， andproducing defect-freecode， isbeautiful.Thereisbeautyonalltheselevels.

ourworldneedssoftware--lotsofsoftware. Fiftyyearsagosoftwarewassomethingthatran inafewbigandexpensivemachines. Thirtyyearsagoitwassomethingthatraninmost companiesandindustrialsettings. Nowthereissoftwarerunninginourcellphones， watches， appliances， automobiles， toys， andtools. Andneedfornewandbetter softwarenever( ).Asourcivilizationgrowsandexpands， asdevelopingnationsbuild theirinfrastructures， asdevelopednationsstrivetoachieveevergreaterefficiencies ， theneedformoreandmoreSoftware( )toincrease. Itwouldbeagreatshameif， inall thatsoftware， therewasnobeauty.

Weknowthatsoftwarecanbeugly. Weknowthatitcanbehardtouse， unreliable ， and carelesslystructured. Weknowthattherearesoftwaresystemswhosetangledandcareless internalstructuresmakethemexpensiveanddifficulttochange. Weknowthatthereare softwaresystemsthatpresenttheirfeaturesthroughanawkwardandcumbersomeinterface. Weknowthattherearesoftwaresystemsthatcrashandmisbehave. Theseare( )systems. Unfortunately， asaprofession， softwaredeveloperstendtocreatemoreuglysystems thanbeautifulones.

Thereisasecretthatthebestsoftwaredevelopersknow. Beautyischeaperthanugliness. Beautyisfasterthanugliness. Abeautifulsoftwaresystemcanbebuiltandmaintainedin lesstime， andforlessmoney ,thananuglyone. Novicesoftwaredevelopersdon't. understandthis. Theythinkthattheyhavetodoeverythingfastandquick. Theythinkthat beautyis( ) .No! Bydoingthingsfastandquick， theymakemessesthatmakethesoftware stiff， andhardtounderstand， Beautifulsystemseflexibleandeasytounderstand. Buildingthemandmaintainingthemisajoy. Itisuglinessthatisimpractical.Ugliness willslowyoudownandmakeyoursoftwareexpensiveandbrittle. Beautifulsystemscostthe leastbuildandmaintain， andaredeliveredsoonest.

A. Simple　B. Hard　C. Complex　D. duplicated

试题七十二 （第 2 空）Thebeautyofsoftwareisinitsfunction， initsinternalstructure ， andinthewayinwhichitiscreatedbyateam. Toauser， aprogramwithjusttheright featurespresentedthroughanintuitiveand( )interfaceisbeautiful.Toasoftware designer， aninternalstructurethatispartitionedinasimpleandintuitivemanner， andthatminimizesinternalcouplingisbeautiful.Todevelopersandmanagers ， a motivatedteamofdevelopersmakingsignificantprogresseveryweek， andproducing defect-freecode， isbeautiful.Thereisbeautyonalltheselevels.

ourworldneedssoftware--lotsofsoftware. Fiftyyearsagosoftwarewassomethingthatran inafewbigandexpensivemachines. Thirtyyearsagoitwassomethingthatraninmost companiesandindustrialsettings. Nowthereissoftwarerunninginourcellphones， watches， appliances， automobiles， toys， andtools. Andneedfornewandbetter softwarenever( ).Asourcivilizationgrowsandexpands， asdevelopingnationsbuild theirinfrastructures， asdevelopednationsstrivetoachieveevergreaterefficiencies ， theneedformoreandmoreSoftware( )toincrease. Itwouldbeagreatshameif， inall thatsoftware， therewasnobeauty.

Weknowthatsoftwarecanbeugly. Weknowthatitcanbehardtouse， unreliable ， and carelesslystructured. Weknowthattherearesoftwaresystemswhosetangledandcareless internalstructuresmakethemexpensiveanddifficulttochange. Weknowthatthereare softwaresystemsthatpresenttheirfeaturesthroughanawkwardandcumbersomeinterface. Weknowthattherearesoftwaresystemsthatcrashandmisbehave. Theseare( )systems. Unfortunately， asaprofession， softwaredeveloperstendtocreatemoreuglysystems thanbeautifulones.

Thereisasecretthatthebestsoftwaredevelopersknow. Beautyischeaperthanugliness. Beautyisfasterthanugliness. Abeautifulsoftwaresystemcanbebuiltandmaintainedin lesstime， andforlessmoney ,thananuglyone. Novicesoftwaredevelopersdon't. understandthis. Theythinkthattheyhavetodoeverythingfastandquick. Theythinkthat beautyis( ) .No! Bydoingthingsfastandquick， theymakemessesthatmakethesoftware stiff， andhardtounderstand， Beautifulsystemseflexibleandeasytounderstand. Buildingthemandmaintainingthemisajoy. Itisuglinessthatisimpractical.Ugliness willslowyoudownandmakeyoursoftwareexpensiveandbrittle. Beautifulsystemscostthe leastbuildandmaintain， andaredeliveredsoonest.

A. happens　B. exists　C. stops　D. starts

试题七十三 （第 3 空）Thebeautyofsoftwareisinitsfunction， initsinternalstructure ， andinthewayinwhichitiscreatedbyateam. Toauser， aprogramwithjusttheright featurespresentedthroughanintuitiveand( )interfaceisbeautiful.Toasoftware designer， aninternalstructurethatispartitionedinasimpleandintuitivemanner， andthatminimizesinternalcouplingisbeautiful.Todevelopersandmanagers ， a motivatedteamofdevelopersmakingsignificantprogresseveryweek， andproducing defect-freecode， isbeautiful.Thereisbeautyonalltheselevels.

ourworldneedssoftware--lotsofsoftware. Fiftyyearsagosoftwarewassomethingthatran inafewbigandexpensivemachines. Thirtyyearsagoitwassomethingthatraninmost companiesandindustrialsettings. Nowthereissoftwarerunninginourcellphones， watches， appliances， automobiles， toys， andtools. Andneedfornewandbetter softwarenever( ).Asourcivilizationgrowsandexpands， asdevelopingnationsbuild theirinfrastructures， asdevelopednationsstrivetoachieveevergreaterefficiencies ， theneedformoreandmoreSoftware( )toincrease. Itwouldbeagreatshameif， inall thatsoftware， therewasnobeauty.

Weknowthatsoftwarecanbeugly. Weknowthatitcanbehardtouse， unreliable ， and carelesslystructured. Weknowthattherearesoftwaresystemswhosetangledandcareless internalstructuresmakethemexpensiveanddifficulttochange. Weknowthatthereare softwaresystemsthatpresenttheirfeaturesthroughanawkwardandcumbersomeinterface. Weknowthattherearesoftwaresystemsthatcrashandmisbehave. Theseare( )systems. Unfortunately， asaprofession， softwaredeveloperstendtocreatemoreuglysystems thanbeautifulones.

Thereisasecretthatthebestsoftwaredevelopersknow. Beautyischeaperthanugliness. Beautyisfasterthanugliness. Abeautifulsoftwaresystemcanbebuiltandmaintainedin lesstime， andforlessmoney ,thananuglyone. Novicesoftwaredevelopersdon't. understandthis. Theythinkthattheyhavetodoeverythingfastandquick. Theythinkthat beautyis( ) .No! Bydoingthingsfastandquick， theymakemessesthatmakethesoftware stiff， andhardtounderstand， Beautifulsystemseflexibleandeasytounderstand. Buildingthemandmaintainingthemisajoy. Itisuglinessthatisimpractical.Ugliness willslowyoudownandmakeyoursoftwareexpensiveandbrittle. Beautifulsystemscostthe leastbuildandmaintain， andaredeliveredsoonest.

A. starts　B. continues　C. appears　D. stops

试题七十四 (第4空)Thebeautyofsoftwareisinitsfunction， initsinternalstructure ， andinthewayinwhichitiscreatedbyateam. Toauser， aprogramwithjusttheright featurespresentedthroughanintuitiveand( )interfaceisbeautiful.Toasoftware designer， aninternalstructurethatispartitionedinasimpleandintuitivemanner， andthatminimizesinternalcouplingisbeautiful.Todevelopersandmanagers ， a motivatedteamofdevelopersmakingsignificantprogresseveryweek， andproducing defect-freecode， isbeautiful.Thereisbeautyonalltheselevels.

ourworldneedssoftware--lotsofsoftware. Fiftyyearsagosoftwarewassomethingthatran inafewbigandexpensivemachines. Thirtyyearsagoitwassomethingthatraninmost companiesandindustrialsettings. Nowthereissoftwarerunninginourcellphones， watches， appliances， automobiles， toys， andtools. Andneedfornewandbetter softwarenever( ).Asourcivilizationgrowsandexpands， asdevelopingnationsbuild theirinfrastructures， asdevelopednationsstrivetoachieveevergreaterefficiencies ， theneedformoreandmoreSoftware( )toincrease. Itwouldbeagreatshameif， inall thatsoftware， therewasnobeauty.

Weknowthatsoftwarecanbeugly. Weknowthatitcanbehardtouse， unreliable ， and carelesslystructured. Weknowthattherearesoftwaresystemswhosetangledandcareless internalstructuresmakethemexpensiveanddifficulttochange. Weknowthatthereare softwaresystemsthatpresenttheirfeaturesthroughanawkwardandcumbersomeinterface. Weknowthattherearesoftwaresystemsthatcrashandmisbehave. Theseare( )systems. Unfortunately， asaprofession， softwaredeveloperstendtocreatemoreuglysystems thanbeautifulones.

Thereisasecretthatthebestsoftwaredevelopersknow. Beautyischeaperthanugliness. Beautyisfasterthanugliness. Abeautifulsoftwaresystemcanbebuiltandmaintainedin lesstime， andforlessmoney ,thananuglyone. Novicesoftwaredevelopersdon't. understandthis. Theythinkthattheyhavetodoeverythingfastandquick. Theythinkthat beautyis( ) .No! Bydoingthingsfastandquick， theymakemessesthatmakethesoftware stiff， andhardtounderstand， Beautifulsystemseflexibleandeasytounderstand. Buildingthemandmaintainingthemisajoy. Itisuglinessthatisimpractical.Ugliness willslowyoudownandmakeyoursoftwareexpensiveandbrittle. Beautifulsystemscostthe leastbuildandmaintain， andaredeliveredsoonest.

A. practical　　B. useful　　C. beautiful　　D. ugly

试题七十五 (第5空)Thebeautyofsoftwareisinitsfunction， initsinternalstructure ， andinthewayinwhichitiscreatedbyateam. Toauser， aprogramwithjusttheright featurespresentedthroughanintuitiveand( )interfaceisbeautiful.Toasoftware designer， aninternalstructurethatispartitionedinasimpleandintuitivemanner， andthatminimizesinternalcouplingisbeautiful.Todevelopersandmanagers ， a motivatedteamofdevelopersmakingsignificantprogresseveryweek， andproducing defect-freecode， isbeautiful.Thereisbeautyonalltheselevels.

ourworldneedssoftware--lotsofsoftware. Fiftyyearsagosoftwarewassomethingthatran inafewbigandexpensivemachines. Thirtyyearsagoitwassomethingthatraninmost companiesandindustrialsettings. Nowthereissoftwarerunninginourcellphones， watches， appliances， automobiles， toys， andtools. Andneedfornewandbetter softwarenever( ).Asourcivilizationgrowsandexpands， asdevelopingnationsbuild theirinfrastructures， asdevelopednationsstrivetoachieveevergreaterefficiencies ， theneedformoreandmoreSoftware( )toincrease. Itwouldbeagreatshameif， inall thatsoftware， therewasnobeauty.

Weknowthatsoftwarecanbeugly. Weknowthatitcanbehardtouse， unreliable ， and carelesslystructured. Weknowthattherearesoftwaresystemswhosetangledandcareless internalstructuresmakethemexpensiveanddifficulttochange. Weknowthatthereare softwaresystemsthatpresenttheirfeaturesthroughanawkwardandcumbersomeinterface. Weknowthatthereareasoftwaresystemsthatcrashandmisbehave. Theseare( )systems. Unfortunately， asaprofession， softwaredeveloperstendtocreatemoreuglysystems thanbeautifulones.

Thereisasecretthatthebestsoftwaredevelopersknow. Beautyischeaperthanugliness. Beautyisfasterthanugliness. Abeautifulsoftwaresystemcanbebuiltandmaintainedin lesstime， andforlessmoney ,thananuglyone. Novicesoftwaredevelopersdon't. understandthis. Theythinkthattheyhavetodoeverythingfastandquick. Theythinkthat beautyis( ) .No! Bydoingthingsfastandquick， theymakemessesthatmakethesoftware stiff， andhardtounderstand， Beautifulsystemseflexibleandeasytounderstand. Buildingthemandmaintainingthemisajoy. Itisuglinessthatisimpractical.Ugliness willslowyoudownandmakeyoursoftwareexpensiveandbrittle. Beautifulsystemscostthe leastbuildandmaintain， andaredeliveredsoonest.

A. impractical　B. perfect　C. time-wasting　D. practical

试题一 答案： B 解析：

本题考查计算机系统基础知识。

CPU 中常设置多个寄存器，其中，程序计数器的作用是保存待读取指令在内存中的地址，累加器是算逻运算单元中用来暂存源操作数和计算结果的寄存器，指令寄存器暂存从内存读取的指令，地址寄存器暂存要访问的内存单元的地址。

试题二 答案： A 解析：

本题考查计算机系统基础知识。

在位级表示中，将 x 与 y 进行“逻辑与”“逻辑或”和“逻辑异或”的结果如下所示。

将整数 a 与 0x000F4 进行“逻辑与”运算，则运算结果中高 12 位都为 0, 而位则完全是 a 的低 4 位，所以“逻辑与”运算的结果为 0, 则说明 a 的低 4 位为 0 。

将整数 a 与 0x000F4 进行“逻辑或”运算，则运算结果中高 12 位都保留的是高 12 位，而低 4 位则全为 1 ，所以“逻辑或”运算的结果不能判定 a 的低 4 位是否为 1 。

将整数 a 与 0xFFF0 进行“逻辑异或”运算，则运算结果中高 12 位是将 a 的高 12 取反，而低 4 位则保留了 a 的低 4 位, 所以“逻辑异或”运算的结果不能判定 a 的低 4 位是否为 0 ，因为高 12 位中可能有 0 有 1 。

将整数 a 与 0xFFF0 进行“逻辑或”运算，则运算结果中高 12 位全是 1 ，而低 4 位则保留了 a 的低 4 位，所以“逻辑或”运算的结果不能判定 a 的低 4 位是否为 0, 因为高 12 位全是 1 。

x	y	逻辑与	逻辑或	逻辑异或
0	0	0	0	0
0	1	0	1	1
1	0	0	1	1
1	1	1	1	0

试题三 答案： D 解析：

本题考查计算机系统基础知识。

中断方式、程序查询方式和无条件传逵方式都是通过 CPU 执行程序指令来传送数据的，DMA 方式下是由 DMA 控制器直接控制数据的传送过程， CPU 需要让出对总线的控制权，并不需要 CPU 执行程序指令来传送数据。

试题四 答案： B 解析：

本题考查计算机系统基础知识。

可靠度为 R_1 和 R_2 的两个部件并联后的可靠度为(1 一(1-R_1)(1 - R_2))，这两个部件串联后的可靠度为 R_1R_2，因此图中所示系统的节的(1-(1-R)sup3)(1-(1-R)$^{2)}$。

试题五 答案： C 解析：

本题考查计算机系统基础知识。

设数据位是 n 位，校验位是 k 位，则海明码中 n 和 k 必须满足以下关系： $2^k-1 \geqslant n+k$ 。

若 n=16，则 k 为 5 时可满足 $2^5 \geqslant 16+5$ 。

海明码的编码规则如下。

设 k 个校验位为 P_k， P_{k-1}，.…， P_1， n 个数据位为 D_{n-1}， D_{n-2}，…， D_1， D_0，对应的海明码为 H_{n+k}， H_{n+k-1}，…， H_1，那么：

①P_i 在海明码的第 2^{i-1} 位置，即 $H_j=P_i$，且 $j=2^{i-1}$；数据位则依序从低到高占据海明码中剩下的位置。

②海明码中的任一位都是由若干个校验位来校验的。其对应关系如下：被校验的海明位的下标等于所有参与校验该位|的校验位的下标之和，而校验位则由自身校验。

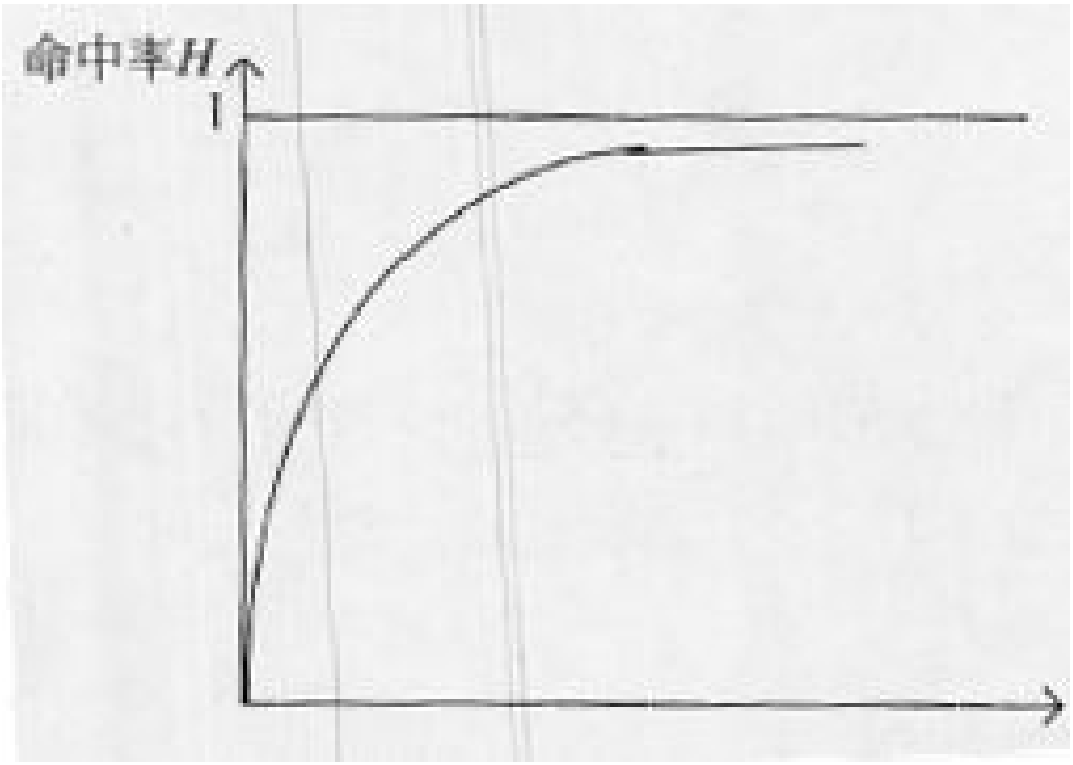
试题六 答案： A 解析：

本题考查计算机系统基础知识。

高速缓存(Cache)是随着 CPU 与主存之间的性能差距不断增大而引入的，其速度比主存快得多，所存储的内容是 CPU 近期可能会需要的信息，是主存内容的副本，因此 CPU 需要访问数据和读取指令时要一访问 Cache, 若命中则直接访问，若不命中再去访问主存。

评价 Cache 性能的关键指标是 Cache 的命中率，影响命中率的因素有其容量、替换算法、其组织方式等。 Cache 的命中率随容量的增大而提高，其关系如下图所示。

Cache 的设置不以扩大主存容量为目的，事实上也并没有扩大主存的容量。



试题七 答案： B 解析：

本题考查 HTTPS 基础知识。

HTTPS (HyperTextTransferProtocoloverSecureSocketLayer)是以安全为目标的 HTTP 通道，即使用 SSL 加密算法的 HTTP 。

试题八 答案： D 解析：

本题考查加密算法的基本知识。

根据题意，要求选出适合对大量明文进行加密传输的加密算法。备选项中的 4 种加密算法均能够对明文进行加密。

RSA 是一种非对称加密算法，由于加密和解密的密钥不同，因此便于密钥管理和分发，同时在用户或者机构之间进行身份认证方面有较好的应用；

SHA-1 是一种安全散列算法，常用宁对接收到的明文输入产生固定长度的输出，来确保明文在传输过程中不会被篡改；

MD5 是一种使用最为广泛的报文摘要算法；

RC5 是一种用于对明文进行加密的算法，在加密速度和强度上，均较为合适，适用于大量明文进行加密并传输。

试题九 答案： D 解析：

本题考查证书认证的基本知识。

用户可在一定的认证机构(CA)处取得各自能够认证自身身份的数字证书，与该用户在同一机构取得的数字证书可通过相互的公钥认证彼此的身份；当两个用户所使用的证书来自于不同的认证机构时，用户双方要相互确定对方的身份之前，首先需要确定彼此的证书颁发机构的可信度。即两个 CA 之间的身份认证，需交换两个 CA 的公钥以确定 CA 的合法性，然后再进行用户的身份认证。

试题一十 答案： A 解析：

本题考查知识产权相关知识。

依照《计算机软件保护条例》的相关规定，计算机软件著作权的归属可以分为以下情况。

①独立开发

这种开发是最普遍的情况。此时，软件著作权当然属于软件开发者，即实际组织开发、直接进行开发，并对开发完成的软件承担责任的法人或者其他组织；或者依照自己具有的条件独立完成软件开发，并对软件承担责任的自然人。

②合作开发

由两个以上的自然人、法人或者其他组织合作开发的软件，一般是合作开发者签定书面合同约定软著作权归属。如果没有书面合同或者合同并未明确约定软件著作权的归属，合作开发的软件如果可以分割使用的，开发者对各自开发的部分可以单独享有著作权；但是行使著作权时，不得扩展到合作开发的软件整体的著作权。如果合作开发的软件不能分割使用，其著作权由各合作开发者共同享有，通过协商一致行使；不能协商一 致，又无正当理由的，任何一方不得阻止他方行使除转让权以外的其他权利，但是所提收益应当合理分配给所有合作开发。

③委托开发

接受他人委托开发的软件，一般也是由委托人与受托人签订书面合同约定该软件著作权的归属；如无书面合同或者合同未作明确约定的，则著作权人由受托人享有。

④国家机关下达任务开发

由国家机关下达任务开发的软件，一般是由国家机关与接受任务的法人或者其他组织依照项目任务书或者合同规定来确定著作权的归属与行使。这里需要注意的是，国家机关下达任务开发，接受任务的人不能是自然人，只能是法人或者其他组织。但如果项目任务书或者合同中未作明确规定的，软件著作权由接受任务的法人或者其他组织享有。

⑤职务开发

自然人在法人或者其他组织中任职期间所开发的软件有下列情形之一的，该软件著作权由

该法人或者其他组织享有。(一)针对本职工作中明确指定的开发目标所开的软件；(二)开发的软件是从事本职工作活动所预见的结果或者自然的结果；(三)主要使用了法人或者其他组织的资金、专用设备、未公开的专门信息等物质技术条件所开发并由法人或者其他组织承担责任的软件。但该法人或者其他组织可以对开发软件的自然人进行奖励。

⑥继承和转让

软件著作权是可以继承的。软件寒作权是属于自然人的，该自然人死亡后，在软件著作权的保护期内，软件著作权法的雄承人可以依照继承法的有关规定，继承除署名权以外的其他软件著作权权利，包括人身权利和财产权利。软件著作权属于法人或者其他组织的，法人或者其他组织变更、终止其著作权在条例规定的保护期内由承受其权利义务的法人或者其他组织享有；没有承受其权利义务的法人或者其他组织的，由国家享有。

试题一十一 答案： D 解析：

本题考查法律法规知识。

我国商标法第六条规定：“国家规定必须使用注册商标的商品，必须申请商标注册，未经核准注册的，不得在市场销售。”

目前根据我国法律法规的规定必须使用注册商标的是烟草类商品。

试题一十二 答案： D 解析：

本题考查知识产权知识。

专利权是一种具有财产权属性的独占权以及由其衍生出来和相应处理权。专利权人的权利包括独占实施权、转让权、实施许可权、放弃权和标记权等。专利权人对其拥有的专利权享有独占或排他的权利，未经其许可或者出现法律规定的特殊情况，任何人不得使用，否则即构成侵权。这是，利权(知识产权)最重要的法律特点之一。

试题一十三 答案： A 解析：

本题考查多媒体基础知识。

语音信号频率范围是 300Hz 3.4kHz ，也就是不超过 4kHz ，按照奈奎斯特定律，要保持话音抽样以后再恢复时不失真，最低抽样频率是 2 倍的最高频率，即 8kHz 就可以保证信号能够正确恢复，因此将，字语音的采样频率定义为 8kHz 。

试题一十四 答案： D 解析：

本题考查多媒体基础知识。

300*3*300*4=900*1200

试题一十五 答案： A 解析：

本题考查结构化分析与设计的相关知识。

结构化分析的输出是结构化设计的输入，设计活动依据分析结果进行。接口设计是描述软件与外部环境之间的交互关系，软件内模块之间的调用关系，而这些关系的依据主要是分析阶段的数据流图。

试题一十六 答案： C 解析：

本题考查结构化分析与设计的相关知识。

结构化分析的输出是结构化设计的输入，设计活动依据分析结果进行。接口设计是描述软件与外部环境之间的交互关系，软件内模块之间的调用关系，而这些关系的依据主要是分析阶段的数据流图。

试题一十七 答案： D 解析：

本题考查软件项目管理的基础知识。

活动图是描述一个项目中各个工作任务相互依赖关系的一种模型，项目的很多重要特性可以通过分析活动图得到，如估算项目完成时间，计算关键路径和关键活动等。

根据上图计算出关键路径为 A-B-D-I-J-L，其长度为 20 。

活动弧 BD 对应的活动的最早开始时间为第 4 天。活动弧 HK 对应活动的最早开始时间为第 11 天。

试题一十八 答案： B 解析：

本题考查软件项目管理的基础知识。

活动图是描述一个项目中各个工作任务相互依赖关系的一种模型，项目的很多重要特性可以通过分析活动图得到，如估算项目完成时间，计算关键路径和关键活动等。

根据上图计算出关键路径为 A-B-D-I-J-L，其长度为 20 。

活动弧 BD 对应的活动的最早开始时间为第 4 天。活动弧 HK 对应活动的最早开始时间为第 11 天。

试题一十九 答案： D 解析：

本题考查项目管理中人员管理的相关知识。

无主程序员组的开发小组，每两十开发人员之间都有沟通路径，因此， 8 人组成的开发小组沟通路径为完全连通无向图的边数，即

m = n(n-1)/2, 其中 n 和 m 分另表示图的顶点数和边数。当 n=8 时， m==28 。

主程序员组中，除了主程序员外的每个开发人员只能和主程序员沟通，因此 8 人组成的开发小组的沟通路径 8-1=7 。

试题二十 答案： B 解析：

本题考查程序语言基础知识。

在源程序中，可由用户(程序员)为变量、函数和数据类型等命名。

试题二十一 答案： D 解析：

本题考查程序语言基础知识。

(b|ab)*b 表示的字符串集合为{b， bb，abb，bbb，abab，bbbb，abbb，babb,... }，除了以 b 结尾，还要求每个 a 后面至少有 1 个 b 。

(ab*)*b 表示的字符串集合为{b， ab，abb，aab，abbb，aaab，abab，...}, 除了以 b 结尾，还要求以 a 开头(除了仅有 1 个 b 的情形)。

a*b*b 表示的字符串集合为{b， ab，bb，abb，aab，bbb，abbb，aabb，aaab，bbbb，...}, 除了以 b 结尾，还要求若干个 a 之后连接若干个 b， b 只能出现在 a 之后。

(a|b)*b 表示的字符串集合为{b， ab，bb，aab，abb，bab，bbb，aaab，aabb，aba)，abbb，baab，babb，bbab，... }

试题二十二 答案： B 解析：

本题考查程序语言基础知识。

程序语言中的词(符号)的构成规则可由正规式描述，词法分析的基本任务就是识别出源程序中的每个词。

语法分析是分析语句及程序的结构是否符合语言定义的规范，对于语法正确的语句，语义分析是判断语句的含义是否正确，因此判断语句的形式是否正确是语法分析阶段的工作。

试题二十三 答案： B 解析：

本题考查操作系统文件管理基知识。

根据题意，计算机系统中的字长为 32 位，每位可以表示一个物通块的“使用”还是“未用”，一个字可记录 32 个物理块的使用情况。又因为磁盘的容量为 300GB，物理块的大小为 4MB，那么该磁盘有 300*1024/4=76800 个物理块，位示图的大小为 76800/32=2400 个字。

试题二十四 答案： B **解析：**

本题考查操作系统进程管理基础知识。

在有限的资源下，要保证系统不发生死锁，则可以按这种逻辑来分析。首先给每个进程分配所需资源数减 1 个资源，然后系统还有 1 个资源，则不可能发生死锁。即： 3*4+1=13 个。

试题二十五 答案： C **解析：**

根据题意，页面大小为 4K，逻辑地址 2D16H 所在页号为 2, 页内地址为 D16H，查页表后可知物理块号为 4, 该地址 i 过变换后，其物理地址应为物理块号 4 拼接上页内地址 D16H，即十六进制 4D16H。

试题二十六 答案： B **解析：**

根据前驱图， P1 进程运行完需要利用 V (S1)操作通知 P2 进程， P2 进程需要等待 P1 进程的通知，故需要利用 P (S1)操作测试 P1 进程是否运行完，所以空 a 应填 “P (S1)”。又由于 P2 进程运行完需要分别通知 P3 、 P4 进程，所以空 b 应填两个 V 操 作，故 b 应填写 V (S2) V (S3)。

根据前驱图， P3 进程需要等待 P2 进程的通知，需要执行 1 个 P 操作，而 P3 进程运行结束需要利用 1 个 V 操作通知 P4 进程，故空 d 应为 1 个 V 操作。

根据前驱图， P4 进程执行需要等待 P2 、 P3 进程的通知，题中 P4 已执行了 P (S3)，故还需要执行 1 个 P (S4)操作；而 P4 进程执行完需要利用一个 V 操作通知 P5 进程，从进程执行图中看出， P5 进程执行 P (S5)操作等待 P4 进程唤醒，故 P4 进程应该执行 V (S5)„

试题二十七 答案： C **解析：**

根据前驱图， P1 进程运行完需要利用 V (S1)操作通知 P2 进程， P2 进程需要等待 P1 进程的通知，故需要利用 P (S1)操作测试 P1 进程是否运行完，所以空 a 应填 “P (S1)”。又由于 P2 进程运行完需要分别通知 P3 、 P4 进程，所以空 b 应填两个 V 操 作，故 b 应填写 V (S2) V (S3)。

根据前驱图， P3 进程需要等待 P2 进程的通知，需要执行 1 个 P 操作，而 P3 进程运行结束需要利用 1 个 V 操作通知 P4 进程，故空 d 应为 1 个 V 操作。

根据前驱图， P4 进程执行需要等待 P2 、 P3 进程的通知，题中 P4 已执行了 P (S3)，故还需要执行 1 个 P (S4)操作；而 P4 进程执行完需要利用一个 V 操作通知 P5 进程，从进程执行图中看出， P5 进程执行 P (S5)操作等待 P4 进程唤醒，故 P4 进程应该执行 V (S5)„

试题二十八 答案： A 解析：

根据前驱图， P1 进程运行完需要利用 V (S1)操作通知 P2 进程， P2 进程需要等待 P1 进程的通知，故需要利用 P (S1)操作测试 P1 进程是否运行完，所以空 a 应填 “P (S1)”。又由于 P2 进程运行完需要分别通知 P3 、 P4 进程，所以空 b 应填两个 V 操 作，故 b 应填写 V (S2) V (S3)。

根据前驱图， P3 进程需要等待 P2 进程的通知，需要执行 1 个 P 操作，而 P3 进程运行结束需要利用 1 个 V 操作通知 P4 进程，故空 d 应为 1 个 V 操作。

根据前驱图， P4 进程执行需要等待 P2 、 P3 进程的通知，题中 P4 已执行了 P (S3)，故还需要执行 1 个 P (S4)操作；而 P4 进程执行完需要利用一个 V 操作通知 P5 进程，从进程执行图中看出， P5 进程执行 P (S5)操作等待 P4 进程唤醒，故 P4 进程应该执行 V (S5)„

试题二十九 答案： D 解析：

本题考查软件过程模型的相关知识。

瀑布模型、原型化模型、增量或迭代的阶段化开发、螺旋模型等都是典型的软件过程模型，要求考生理解这些模型的优缺点以及适用的场合。螺旋模型是一个风险驱动的过程模型，因此要求开放人员必须具有丰富的风险评估知识和经验，否则因为忽视或过于重视风险造成问题。在对测试风险评估后，可以降低过多测试或测试不足带来的风险。而且，螺旋模型是一个迭代的模型，维护阶段是其中的一个迭代。螺旋模型适用于大规模的软件项目开发。

试题三十 答案： D 解析：

本题考查敏捷开发中极限编程(XP)的相关知识。

敏捷开发方法是一个强调灵活性和快速开发的一种开发方法，有多种具体的方法，其中极限编程是敏捷方法中最普遍的一种方法。极限编程包含 12 个实践操作。其中，集体所有权表示任何开发人员可以对系统任何部分进行改变，结对编程实际上存在一个非正式的代码审查过程，可以获得更高的代码质量。据统计，结对编程的编码速度与传统的单人编程相当。

试题三十一 答案： D 解析：

本题考查软件体系结构的相关知识。

三层 C/S 体系结构由逻辑上相互分离的表示层、业务层和数据层构成。其中表示层向客户提供数据，业务层实施业务相数据规则，数据层定义数据访问标准。该体系结构具有许多优点，如逻辑上相对独立，不同层可以用不同的平台、软件和开发语言，而系统的安装、修改和维护在各层都可能进行。

试题三十二 答案： C 解析：

本题考查软件设计的相关知识。

在软件设计中，人们总结了一些启发式原则，根据这些原则进行设计，可以设计出较高质量的软件系统。其中，模块的扇入扇出适中，模块大小适中以及完善模块功能都可以改进设计质量。而将相似功能的模块合并可能会降低模块内聚和提高模块之间的耦合，因此并不能改进设计质量。

试题三十三 答案： A 解析：

本题考查计算机软件设计的相关知识。

模块的内聚是一个模块内部各个元素彼此结合的紧密程度的度量。

①巧合内聚：指一个模块内的各处理元素之间没有任何联系。

②逻辑内聚：指模块内执行考干个逻辑上相似的功能，通过参数确定该模块完成哪一个功能。

③时间内聚：把需要同时执行的动作组合在一起形成的模块。

④过程内聚：指一个模块完成多个任务，这些任务必须按指定的过程执行。

由于模块 D 内的语句之间没有任何联系，因此该模块的内聚类型为巧合内聚。

该内聚类型具有最低的内聚性，是最不好的一种内聚类型。具有该类内聚类型的模块具有不易修改、不易理解和不易维护等特点，同时会影响到模块间的耦合关系。

试题三十四 答案： D 解析：

本题考查计算机软件设计的相关知识。

模块的内聚是一个模块内部各个元素彼此结合的紧密程度的度量。

①巧合内聚：指一个模块内的各处理元素之间没有任何联系。

②逻辑内聚：指模块内执行考干个逻辑上相似的功能，通过参数确定该模块完成哪一个功能。

③时间内聚：把需要同时执行的动作组合在一起形成的模块。

④过程内聚：指一个模块完成多个任务，这些任务必须按指定的过程执行。

由于模块 D 内的语句之间没有任何联系，因此该模块的内聚类型为巧合内聚。

该内聚类型具有最低的内聚性，是最不好的一种内聚类型。具有该类内聚类型的模块具有不易修改、不易理解和不易维护等特点，同时会影响到模块间的耦合关系。

试题三十五 答案： B 解析：

本题考查软件测试的相关知识。

要求考生能够熟练掌握典型的白盒测试和黑盒测试方法。语句覆盖和路径覆盖是两种具体的白盒测试方法，语句覆盖是指设计若干测试用例，覆盖程序中的所有语句，而路径覆盖是指设计若干个测试用例，覆盖程序中的所有路径。该流程图中：

设计两个测试用例执行路径：

①②③④⑤⑥⑦⑧⑩ 和①②③④⑤⑨⑩ 即可满足语句覆盖。

c 流程图中有：

①②③④⑤⑥⑦⑧⑩

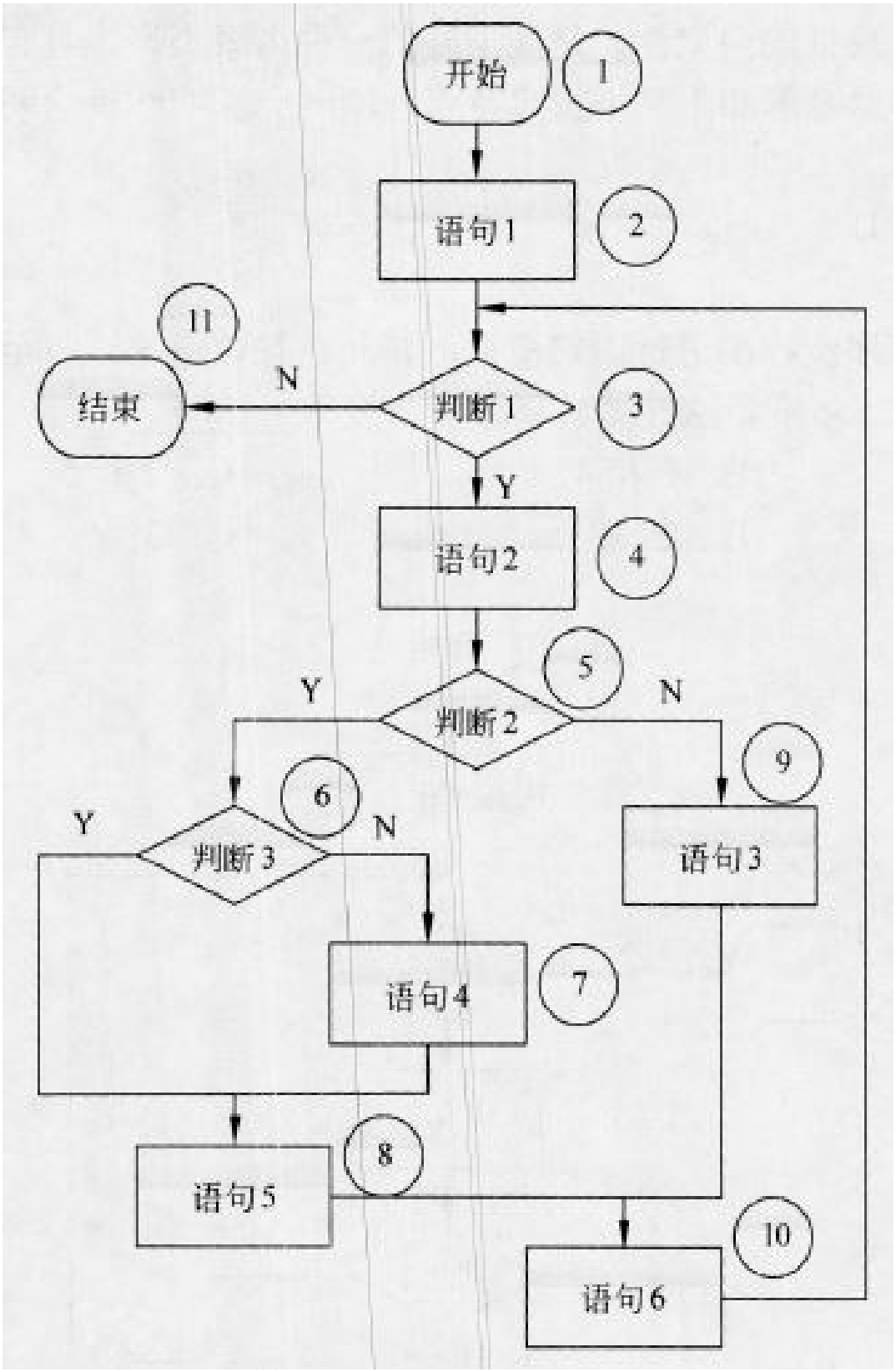
①②③④⑤⑥⑧⑩

①②③④⑤⑨⑩

①②③

4 条路径，因此 4 个最少测试用例就可以满足路径覆盖。

McCabe 度量法是一种基于程序控制流的复杂性度量方法，环路复杂性为 V(G) =m-n+2, 图中 m=13， n=11，V(G)=13-11+2=4 。


开始
1
语句1
2
11
结束
N
判断1
3
Y
语句2
4
5
Y
判断2
N
9
6
Y
判断3
N
语句3
语句4
7
8
语句5
10
语句6

试题三十六 答案： D 解析：

本题考查软件测试的相关知识。

要求考生能够熟练掌握典型的白盒测试和黑盒测试方法。语句覆盖和路径覆盖是两种具体的白盒测试方法，语句覆盖是指设计若干测试用例，覆盖程序中的所有语句，而路径覆盖是指设计若干个测试用例，覆盖程序中的所有路径。该流程图中：

设计两个测试用例执行路径：

①②③④⑤⑥⑦⑧⑩ 和①②③④⑤⑨⑩ 即可满足语句覆盖。

c 流程图中有：

①②③④⑤⑥⑦⑧⑩

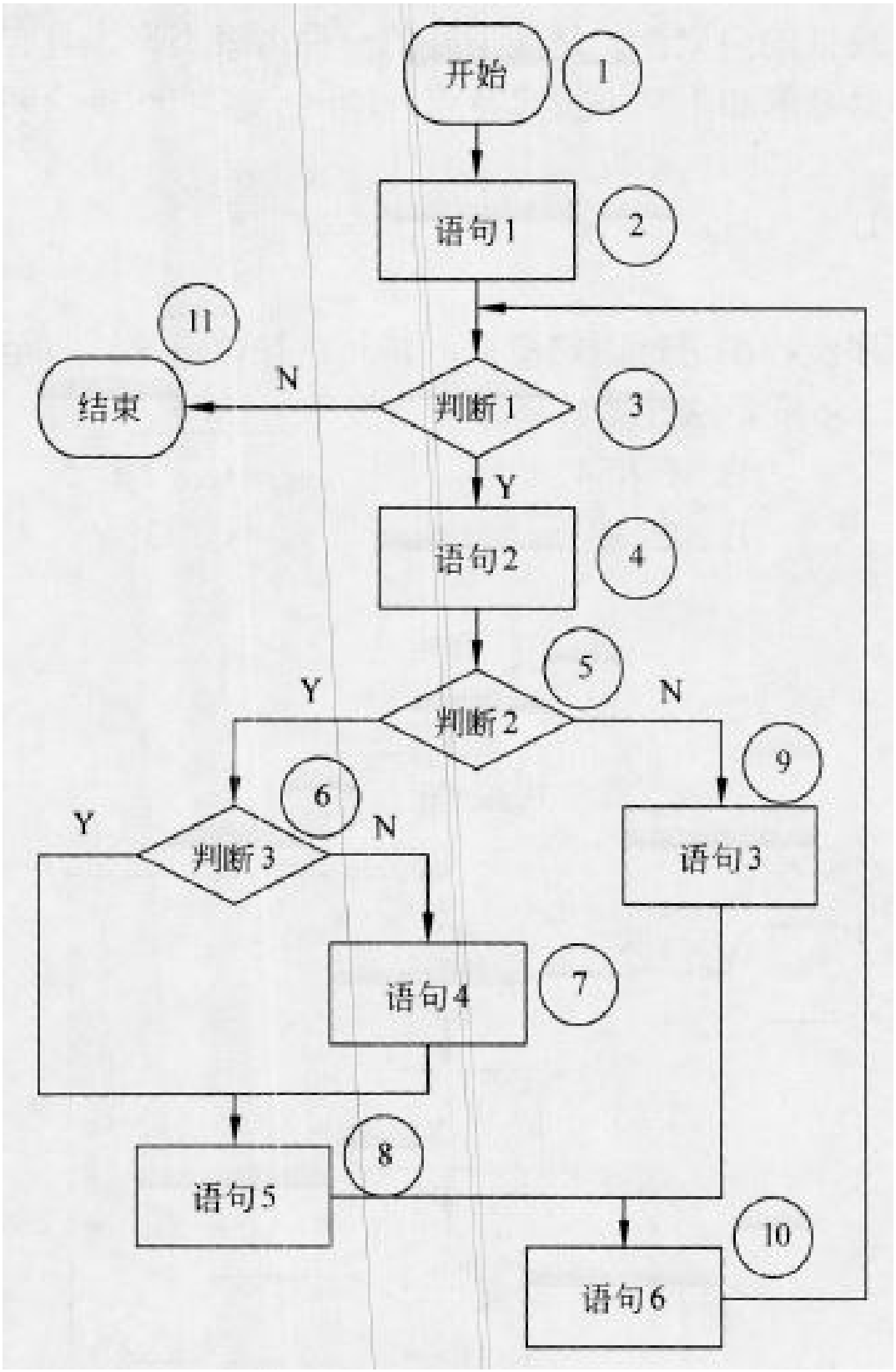
①②③④⑤⑥⑧⑩

①②③④⑤⑨⑩

①②③

4 条路径，因此 4 个最少测试用例就可以满足路径覆盖。

McCabe 度量法是一种基于程序控制流的复杂性度量方法，环路复杂性为 V(G) =m-n+2, 图中 m=13， n=11，V(G)=13-11+2=4 。

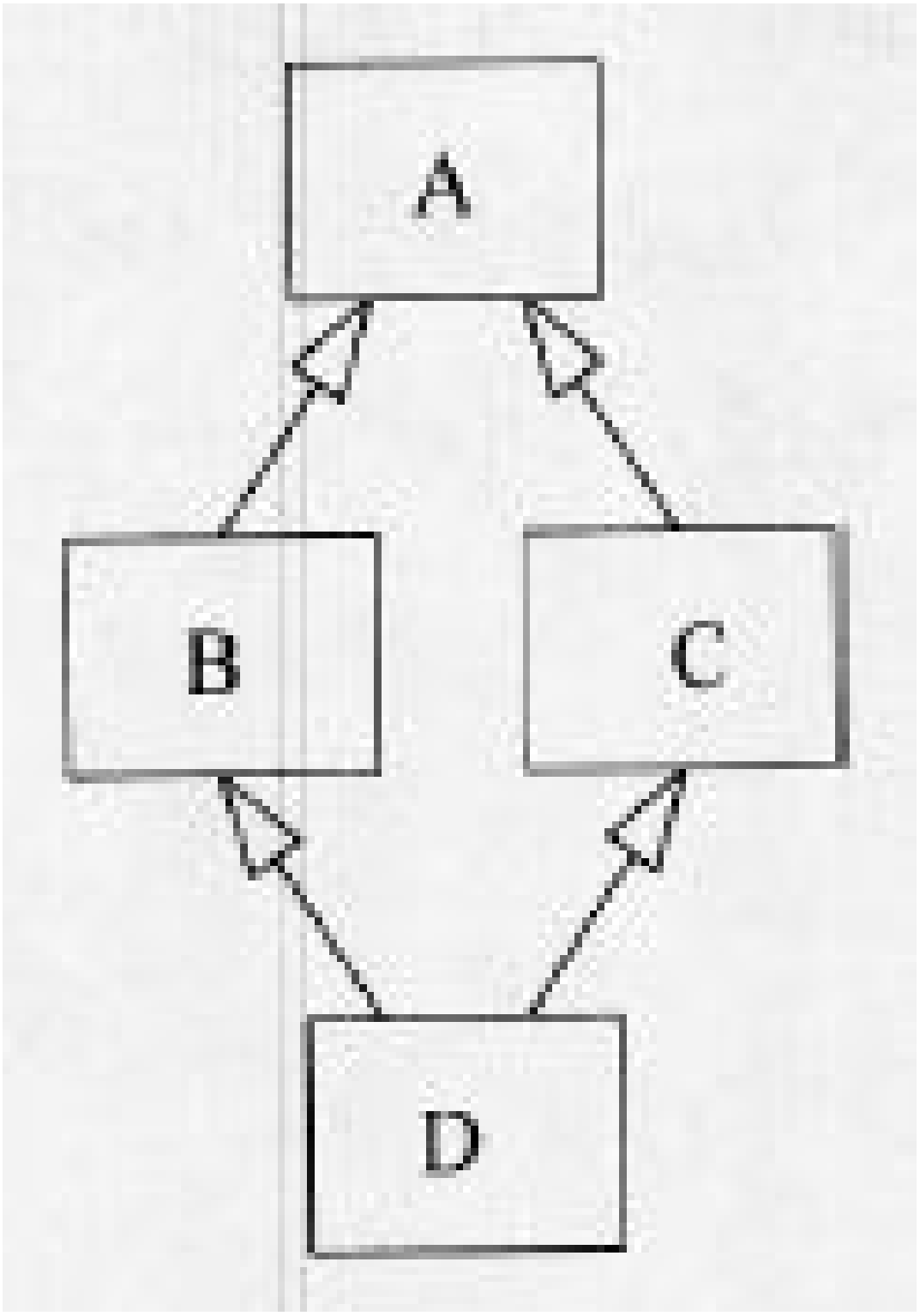

开始
1
语句1
2
11
结束
N
判断1
3
Y
语句2
4
5
Y
判断2
N
9
6
Y
判断3
N
语句3
语句4
7
8
语句5
10
语句6

试题三十七 答案： A 解析：

本题考查面向对象的基本知识。

在面向对象方法中，对象是基本的运行时实体，一组大体上相似的对象定义为一个类。有些类之间存在一般和特殊关在定义和实现一个类的时候，可以在已经存在的类的基础上来进行，把已经存在的类所定义的属性和行为作为自己的内容，并加入新的内容，这种机制就是超类(父类)如子类之间共享数据和方法的机制(即继承)。在继承的支持下，不同对象在收到同一消息是可以产生不同的结果，这就是多态。

定义一个类时继承多于一个超称为多重继承。使用多重继承时，多个超类中可能定义有相同名称而不同含义的成员，就可能会造成子类中存在二义性的成员。例如下图所示为典型的“钻石问题”，类 B 和 C 都继承 A，类 D 继承 B 和 C。若 A 中有一个方法， B 和 C 进行覆盖，而 D 没有进行覆盖，那么 D 应该继承 B 中的方法版本还是 C 中的方法版本就无法确定，从而产生二叉性。



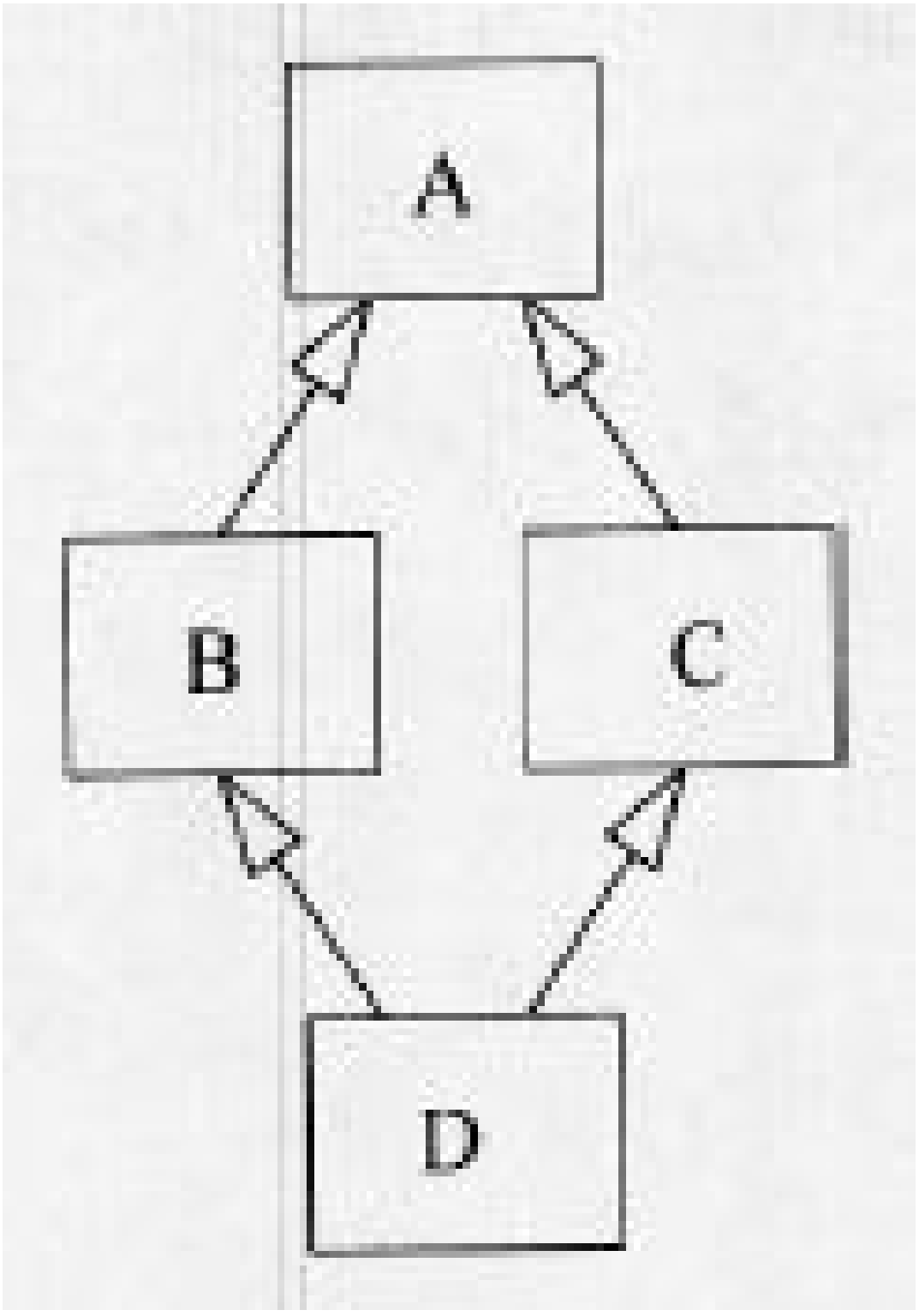
试题三十八 答案：　D 解析：

本题考查面向对象的基本知识。

在面向对象方法中，对象是基本的运行时实体，一组大体上相似的对象定义为一个类。有些类之间存在一般和特殊关在定义和实现一个类的时候，可以在已经存在的类的基础上来

进行，把已经存在的类所定义的属性和行为作为自己的内容，并加入新的内容，这种机制就是超类(父类)如子类之间共享数据和方法的机制(即继承)。在继承的支持下，不同对象在收到同一消息是可以产生不同的结果，这就是多态。

定义一个类时继承多于一个超称为多重继承。使用多重继承时，多个超类中可能定义有相同名称而不同含义的成员，就可能会造成子类中存在二义性的成员。例如下图所示为典型的“钻石问题”，类 B 和 C 都继承 A，类 D 继承 B 和 C 。若 A 中有一个方法， B 和 C 进行覆盖，而 D 没有进行覆盖，那么 D 应该继承 B 中的方法版本还是 C 中的方法版本就无法确定，从而产生二叉性。



试题三十九 答案：　D 解析：

本题考查面向对象技术的基本知识。

采用面向对象技术进行软件开发时，需要进行面向对象分析(OOA)、面向对象设计(OOD)、面向对象实现和面向对象测试几个阶段。分析阶段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，在此阶段主要关注系统的行为，明确系统需要提供什么服务。在设计阶段，采用面向对象技术将 OOA 所创建的分析模型转化为设计模型，其目标是定义系统构造蓝图。在实现阶段(面向对象程序设计)，系统实现人员选用一种面向对象程序设计语言，采用对象、类及其相关概念进行程序设计，即实现系统。

试题四十 答案： A 解析：

本题考查面向对象的基本知识。

多态的实现受到继承的支持，利用类的继承的层次关系，把具有通用功能的消息存放在高层次，而不同的实现这一功能的行为放在较低层次。当一个客户类对象发送通用消息请求服务时，它无需知道所调用方法的特定子类的实现，是根据接收对象的具体情况将请求的操作与实现的方法进行连接，即动态绑定，以实现在这些低层次上生成的对象给通用消息以不同的响应。

试题四十一 答案： C 解析：

本题考查统一建模语言(UML)的基本知识。

UML2.0 及后续版本提供了 13 种图，部分图用于刻画系统的静态方面，如类图、对象图等；部分图刻画系统的动态方两，如序列图、状态图、通信图和活动图等。

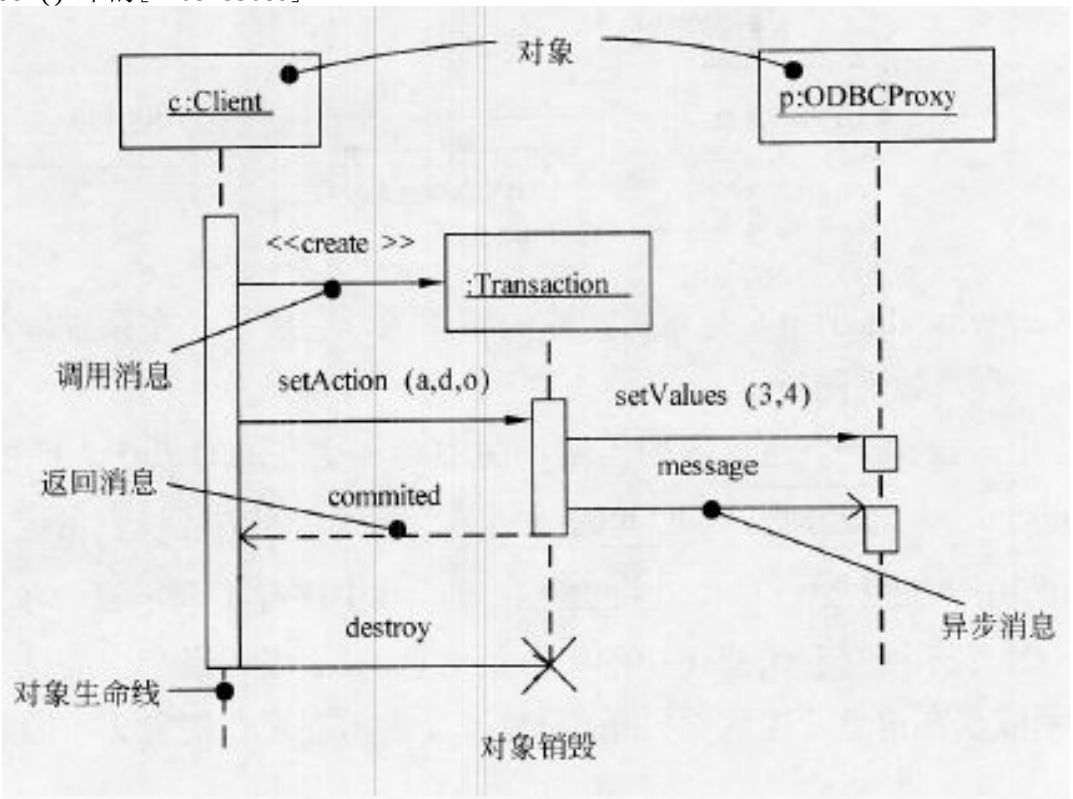
序列图(sequencediagram) 场景(scenario)的图形化表示，描述了以时间顺序组织的对象之间的交互活动。如下图所示，参加交互的对象放在图的上方，沿水平方向排列。然后，把这些对象发送和接收的消息沿垂直方向按时间顺序从上到下放置。

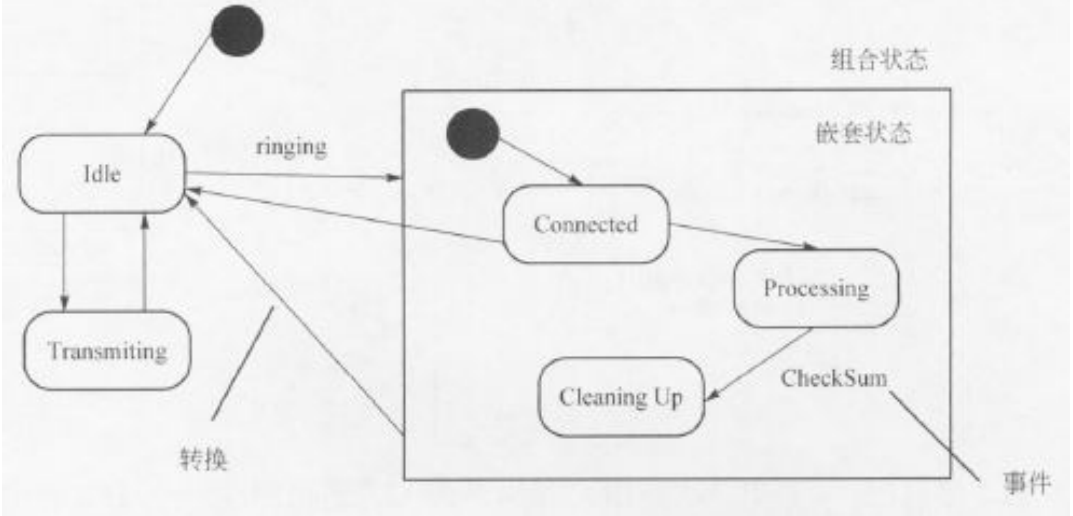
状态图(statediagram)展现了一个状态机，它由状态、转换、事件和活动组成，对于接口、类和协作的行为建模尤为重要，强调对象行为的事件顺序。如下图所示，状态图通常包括简单状态和组合状态、转换(事件和动作)。

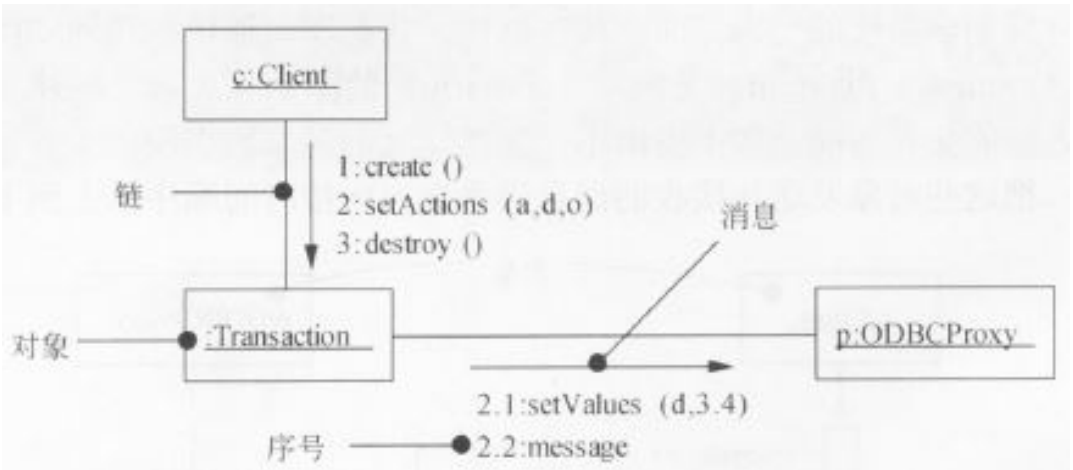
通信图(communicationdiagram)强调收发消息的对象的结构组织，强调参加交互的对象的组织，在早期的版本中也被称作协作图。如下图所示，参与交互的对象作为图的顶点，连接这些对象的链表示为图的弧，用对象发送和接收的消息来修饰这些链。通信图有路径，消息有序号，如图中的 1 、 2 、 2.1 等。

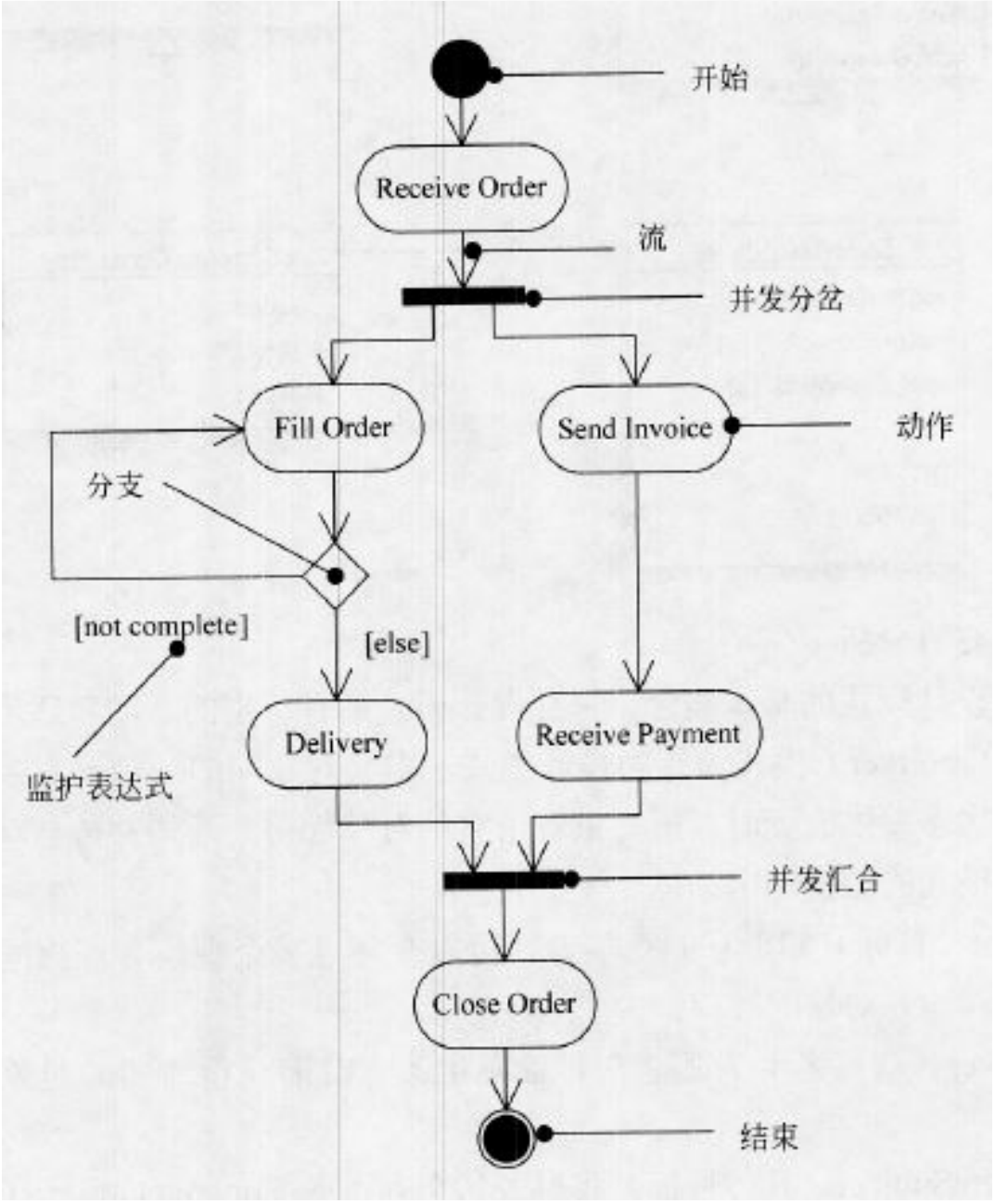
活动图(activitydiagram)是一种特殊的状态图，它展现了在系统内从一个活动到另一个活动的流程，如下图所示。

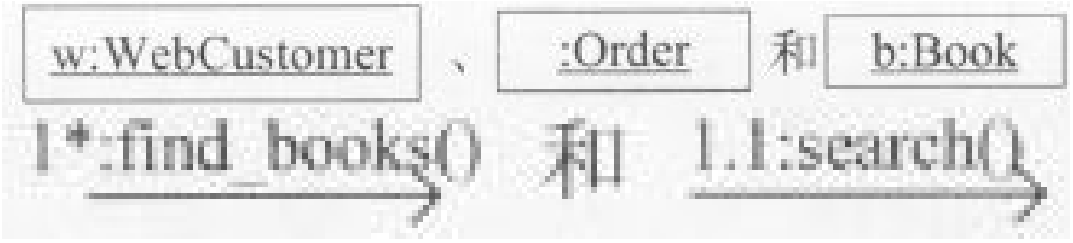
本题叙述中，给出一个通信图的实例，图中参与交互的对象作为图的顶点，包括等，对象之间的连线表示链，并用消息来进行修饰，如图中等，其中序号 1 表示第一条消息， 2 表示第二条消息， 1.1 表示嵌套在消息 1 中的第一条消息， 1.2 表示嵌套在消息 1 中的第二条消息， 2.1 表示嵌套在消息 2 中的第一条消息， 2.2 表示嵌套在消息 2 中的第二条消息，依此类推。在一条消息中，可以有监护条件，如题图中消息 1.2:[interested]:view_book() 中的[interested]。






c:Client
链
1:create ()
2:setActions (a,d,o)
3:destroy ()
消息
对象
:Transaction
p:ODBCProxy
2.1:setValues (d,3.4)
序号
2.2:message


开始
Receive Order
流
并发分岔
Fill Order
Send Invoice
动作
分支
[not complete]
[else]
监护表达式
Delivery
Receive Payment
并发汇合
Close Order
结束



试题四十二 答案： B 解析：

从图示可以了解到，题目中的图是通信图。通信图描述的是对象和对象之间的关系，即一个类操作的实现。简而言之就是，对象和对象之间的调用关系，体现的是一种组织关系。该图明显表达的是对象与对象之间的关系。其中如果一个框中的名称中带有“:”号，说明这表示的是一个对象，“:”号前的部分是对象名，“:”号后面的部分是类名。而对象之间连线上面的箭头所标识的是对象之间通信的消息。

试题四十三 答案： D 解析：

从图示可以了解到，题目中的图是通信图。通信图描述的是对象和对象之间的关系，即一个类操作的实现。简而言之就是，对象和对象之间的调用关系，体现的是一种组织关系。该图明显表达的是对象与对象之间的关系。其中如果一个框中的名称中带有“:”号，说明这表示的是一个对象，“:”号前的部分是对象名，“:”号后面的部分是类名。而对象之间连线上面的箭头所标识的是对象之间通信的消息。

试题四十四 答案： A 解析：

本题考查设计模式的基本概念。每种设计模式都有特定的意图和适用情况。

观察者(Observer)模式为行为设计模式，定义对象间的一种一对多的依赖关系，当一个对象的状态发生改变时，所有依赖于它的对象都得到通知并被自动更新。在题图所示的观察者模式的结构图中：

①Subject（目标)知道它的观察者，可以有任意多个观察者观察同一个目标；提供注册和删除观察者对象的接口；

②Observer（观察者)为那些在目标发生改变时需获得通知的对象定义一个更新接口；

③ConcreteSubject（具体目标)将有关状态存入各 ConcreteObserver 对象；当它的状态发生改变时，向其各个观察者发出通知；

④ConcreteObserver（具体观察者)维护一个指向 ConcreteSubject 对象的引用；存储有关状态，这些状态应与目标的状态保持一致；实现 Observer 的更新接口，以使自身状态与目标的状态保持一致。

该模式的特点适用于以下情况：

①当一个抽象模型有两个方面，其中一个方面依赖于另一个方面，将这两者封装在独立的对象中以使它们可以各自独立地改变和复用；

②当对一个对象的改变需要同时改变其他对象，而不知道具体有多少对象有待改变时；

③当一个对象必须通知其他对象，而它又不能假定其他对象是谁，即不希望这些对象是紧耦合的。

试题四十五 答案： B 解析：

本题考查设计模式的基本概念。每种设计模式都有特定的意图和适用情况。

观察者(Observer)模式为行为设计模式，定义对象间的一种一对多的依赖关系，当一个对象的状态发生改变时，所有依赖于它的对象都得到通知并被自动更新。在题图所示的观察者模式的结构图中：

①Subject（目标)知道它的观察者，可以有任意多个观察者观察同一个目标；提供注册和删除观察者对象的接口；

②Observer（观察者)为那些在目标发生改变时需获得通知的对象定义一个更新接口；

③ConcreteSubject（具体目标)将有关状态存入各 ConcreteObserver 对象；当它的状态发生改变时，向其各个观察者发出通知；

④ConcreteObserver（具体观察者)维护一个指向 ConcreteSubject 对象的引用；存储有关状态，这些状态应与目标的状态保持一致；实现 Observer 的更新接口，以使自身状态与目标的状态保持一致。

该模式的特点适用于以下情况：

①当一个抽象模型有两个方面，其中一个方面依赖于另一个方面，将这两者封装在独立的对象中以使它们可以各自独立地改变和复用；

②当对一个对象的改变需要同时改变其他对象，而不知道具体有多少对象有待改变时；

③当一个对象必须通知其他对象，而它又不能假定其他对象是谁，即不希望这些对象是紧耦合的。

试题四十六 答案： B 解析：

本题考查设计模式的基本概念。

装饰器(Decorator)模式和外观(Facade)模式均为对象结构型设计模式。装饰器模式动态地给一个对象添加一些额外的职责。就增加功能而言，此模式比生成子类更加灵活。外观模式为子系统中的一组接口提供一个统一的界面，此模式定义了一个高层接口，这个接口使得这一子系统更加容易使用。这两种模式适用于不同的情况。

装饰器模式适用于以下几种情况：

①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；

②处理那些可以撤销的职责；

③当不能采用生成子类的方式进行扩充时。一种情况是，可能有大量独立的扩展，为支持每一种组合将产生大量的子类，使得子类数目呈爆炸性增长。另一种情况可能是，由于类定义被隐藏，或类定义不能用于生成子类。

外观模式适用于以下几种情况：

①要为一个复杂子系统提供一个简单接口时，子系统往往因为不断演化而变得越来越复杂，Facade 可以提供一个简单的默认视图，这一视图对大多数用户来说已经足够，而那些需要更多的可定制性的用户可以越过 Facade 层；

②客户程序与抽象类的实现知分之间存在着很大的依赖性。引入 Facade 将这个子系统与客户以及其他的子系统分离，可以提高子系统的独立性和可移植性；

③当需要构建一个层次结构子系统时，使用 Facade 模式定义子系统中每层的入口点。如果子系统之间是相互依赖的，则可以让它们仅通过 Facade 进行通信，从而简化了它们之间的依赖关系。

本题所给 4 种情况中，②将一个对象加以包装以提供一些额外的行为符合装饰器的动态、透明地给单个对象添加职责的适用性；④将一系列对象加以包装以简化其接口符合外观模式的主要特点。

试题四十七 答案： D **解析：**

本题考查设计模式的基本概念。

装饰器(Decorator)模式和外观(Facade)模式均为对象结构型设计模式。装饰器模式动态地给一个对象添加一些额外的职责。就增加功能而言，此模式比生成子类更加灵活。外观模式为子系统中的一组接口提供一个统一的界面，此模式定义了一个高层接口，这个接口使得这一子系统更加容易使用。这两种模式适用于不同的情况。

装饰器模式适用于以下几种情况：

①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；

②处理那些可以撤销的职责；

③当不能采用生成子类的方式进行扩充时。一种情况是，可能有大量独立的扩展，为支持每一种组合将产生大量的子类，使得子类数目呈爆炸性增长。另一种情况可能是，由于类定义被隐藏，或类定义不能用于生成子类。

外观模式适用于以下几种情况：

①要为一个复杂子系统提供一个简单接口时，子系统往往因为不断演化而变得越来越复杂，

Facade 可以提供一个简单的默认视图，这一视图对大多数用户来说已经足够，而那些需要更多的可定制性的用户可以越过 Facade 层；

②客户程序与抽象类的实现知分之间存在着很大的依赖性。引入 Facade 将这个子系统与客户以及其他的子系统分离，可以提高子系统的独立性和可移植性；

③当需要构建一个层次结构子系统时，使用 Facade 模式定义子系统中每层的入口点。如果子系统之间是相互依赖的，则可以让它们仅通过 Facade 进行通信，从而简化了它们之间的依赖关系。

本题所给 4 种情况中，②将一个对象加以包装以提供一些额外的行为符合装饰器的动态、透明地给单个对象添加职责的适用性；④将一系列对象加以包装以简化其接口符合外观模式的主要特点。

试题四十八 答案： C **解析：**

本题考查程序语言基础知识。

在自动机中，从初态到达终态的一条路径上所标记的字符形成的串正好与要识别的串相同，则称该自动机能识别此字符串。

对于 00110，从状态 A 出发，其识别路径为 A→B→D→C→E→B，字符串结束时，自动机所在状态为 B，而不是终态 D 或 E,所以该 DFA 不能识别 00110。

对于 10101，从状态 A 出发，其识别路径为 A→C→B→C→B→C，字符串结束时，自动机所在状态为 C，而不是终态 D 或 E,所以该 DFA 不能识别 10101。

对于 11100，从状态 A 出发，其识别路径为 A→C→E→E→B→D，字符串结束时，自动机所在状态为终态 D，所以该 DFA 可以识别 11100。

对于 11001，从状态 A 出发，其识别路径为 A→C→E→B→D→C，字符串结束时，自动机所在状态为 C，而不是终态 D 或 E，所以该 DFA 不能识别 11001。

试题四十九 答案： B **解析：**

本题考查程序语言基础知识。

若实现函数调用时，是将实参的值传递给对应的形参，则称之为传值调用。这种方式下，形参不能向实参传递信息。引用调用的本质是将实参的地址传给形参，函数中对形参的访问和修改实际上就是针对 i 应实参变量所作的访问和改变。

根据题目说明，调用 f 时，第一个参数是传值方式，第二个参数是引用方式，因此在 main 函数中，先将其局部变量 i 的值加 1 后(即 6)传递给 f 的第一个参数 x，而将其 main 中 x 的地址传给 f 的第二个参数 a，因此在 f 中对 a 的修改等同于是对 main 中 x 的修改。

在 f 中， x 的初始值为 6, 经过“x=x*x-1”运算后修改为 35 ，经过“a=x+a”运算后将 a 的值改为 40 。这里需要注意的是， f 中的 x 与 main 中的 x 是两个不同且相互独立的变量。

试题五十 答案： A 解析：

本题考查程序语言基础知识。

对题中的二叉树进行后序遍历即可得该二叉树所表示表达式的后缀式，为“x5y+*a/b-”。

试题五十一 答案： D 解析：

本题考查数据库并发控制方面的基础知识。

在多用户共享的系统中，许多用户可能同时对同一数据进行操作，可能带来数据不一致问题。为了解决这类问题，数据库系统必须控制事务的并发执行，保证数据库处于一致的状态，在并发控制中引入两种锁：排他锁(ExclusiveLocks, 简称 X 锁)和共享锁(ShareLocks, 简称 S 锁)。

排他锁又称为写锁，用于对数据进行写操作时进行锁定。如果事务 T 对数据 A 加上 X 锁后，就只允许事务 T 读取和修改数据 A ，其他事务对数据 A 不能再加任何锁，从而也不能读取和修改数据 A ，直到事务 T 释放 A 上的锁。

共享锁又称为读锁，用于对数据进行读操作时进行锁定。如果事务 T 对数据 A 加上了 S 锁后，事务 T 就只能读数据 A 但不可以修改，其他事务可以再对数据 A 加 S 锁 来读取，只要数据 A 上有 S 锁，任何事务都只能再对其加 S 锁(读取)而不能加 X 锁 (修改)。

试题五十二 答案： C 解析：

本题考查数据库并发控制方面的基础知识。

在多用户共享的系统中，许多用户可能同时对同一数据进行操作，可能带来数据不一致问题。为了解决这类问题，数据库系统必须控制事务的并发执行，保证数据库处于一致的状态，在并发控制中引入两种锁：排他锁(ExclusiveLocks, 简称 X 锁)和共享锁(ShareLocks, 简称 S 锁)。

排他锁又称为写锁，用于对数据进行写操作时进行锁定。如果事务 T 对数据 A 加上 X 锁后，就只允许事务 T 读取和修改数据 A ，其他事务对数据 A 不能再加任何锁，从而也不能读取和修改数据 A ，直到事务 T 释放 A 上的锁。

共享锁又称为读锁，用于对数据进行读操作时进行锁定。如果事务 T 对数据 A 加上了 S 锁后，事务 T 就只能读数据 A 但不可以修改，其他事务可以再对数据 A 加 S 锁 来读取，只要数据 A 上有 S 锁，任何事务都只能再对其加 S 锁(读取)而不能加 X 锁 (修改)。

试题五十三 答案： A **解析：**

本题考查关系数据库基本知识。

在关系数据库中，候选关键字可以决定全属性。由于属性 A_1 只出现在函数依赖的左部，所以必为候选关键字的成员。本题 $(A_1A_3)_F^+ = U$，$(A_1A_2)_F^+ = U$，所以 A_1A_3 和 A_1A_2 均为候选关键字，且含有属性 A_1。而选项 D 求属性的闭包不能包含全属性 $(A_2A_3)_F^+ \neq U$，故 A_2A_3 不是候选关键字。

试题五十四 答案： A **解析：**

本题考查 SQL 语言基础知识。

试根据题意“一个供应商可以为多个项目供应多种零件，每个项目可由多个供应商供应多种零件。”可知， SP_P 的联系类型为多对多对多(*:*:*)，其 ER 模型如下图所示。而多对多对多的联系必须生成一个独立的关系模式，该模式是由多端的码即“供应商号”“项目号”“零件号”以及 SP_P 联系的属性“数量”构成。

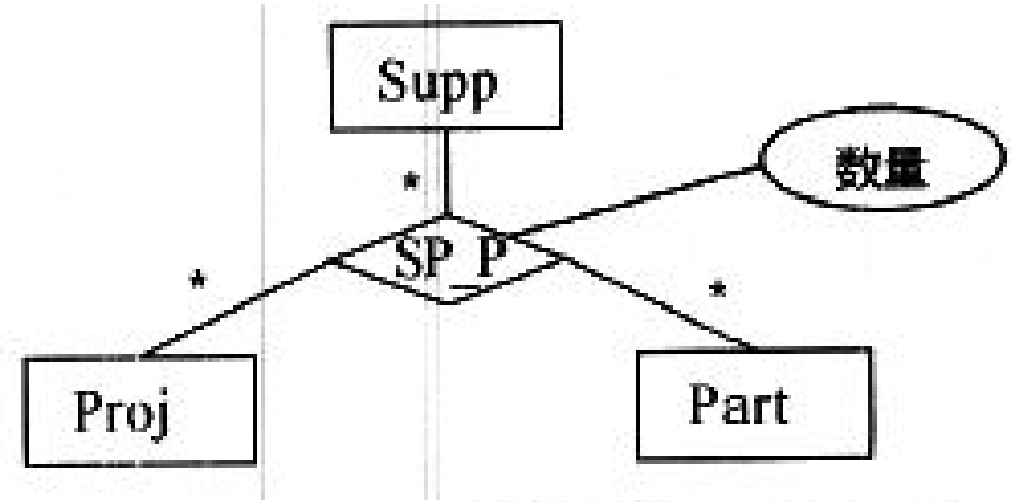
SELECT 语句格式如下：

根据题意，从关系模式 SP_P（供应商号，项目号，零件号，数量)查询至少供应了 3 个项目(包含 3 项)的供应商，输出其供应商号和供应零件数量的总和。

根据题意，需查询至少供应了 3 个项目(包含 3 项)的供应商，故应该按照供应商号分组，而且应该加上条件项目号的统计数目。一个供应商可能为同一个项目供应了多种零件，因此，在统计工程项目数的时候需要加上 DISTINCT,以避免重复统计导致错误的结果。

假如按供应商号=’s1’分组，结果如表 1 。

从表 1 我们可以看出，如果不加 DISTINCT,统计结果为 7,而加了 DISTINCT,统计结果是 3 。




```
SELECT [ALL|DISTINCT]<目标列表达式>[,<目标列表达式>]…
    FROM <表名或视图名>[,<表名或视图名>]
    [WHERE <条件表达式>]
    [GROUP BY <列名 1>[HAVING<条件表达式>]]
    [ORDER BY <列名 2>[ASC|DESC]…]
```

表 1 按供应商号='S1'分组

供应商号	零件号	项目号	数量
S1	P1	J1	200
S1	P3	J1	400
S1	P3	J2	200
S1	P5	J2	100
S1	P1	J3	200
S1	P6	J3	300
S1	P3	J3	200

试题五十五 答案： D **解析：**

后面 2 个空考查的是 SQL 语言，目前需要查询的是零件数量总和，很明显在题目的多个关系中只有 SP_P 有这个属性。所以查询只能 FROMSP_P 。接下来分析如何能把至少供应了 3 个项目的供应商找出来，此时需要写查询条件。查询条件 Where 与 Having 的区别要弄清楚， Where 是针对单条记录的判断条件，而 Having 是针对分组之后的判断条件，此处应选 Having ，同时，由于考虑到项目号可能重复，所以需要加 Distinct 关键字以便去掉重复。

试题五十六 答案： C **解析：**

后面 2 个空考查的是 SQL 语言，目前需要查询的是零件数量总和，很明显在题目的多个关系中只有 SP_P 有这个属性。所以查询只能 FROMSP_P 。接下来分析如何能把至少供应了 3

个项目的供应商找出来，此时需要写查询条件。查询条件 Where 与 Having 的区别要弄清楚， Where 是针对单条记录的判断条件，而 Having 是针对分组之后的判断条件，此处应选 Having，同时，由于考虑到项目号可能重复，所以需要加 Distinct 关键字以便去掉重复。

试题五十七 答案： C 解析：

本题考查数据结构基础知识。

字符串是由字符构成的序列，属于线性数据结构。包含任意个空格字符的字符串称为空白串。字符串中的字符取自特定的字符集合(常见的是 ASCII 字符集)，其长度是指包含的字符个数。

试题五十八 答案： A 解析：

IIOIIOI000 出栈序列为： a_2 a_4 a_5 a_3 a_1

IOIOIOIOI0 出栈序列为： a_1 a_2 a_3 a_4 a_5

I00IIOIOI0 无合法出栈序列，因为入栈 1 个元素，出栈 2 个元素，会产生错误。

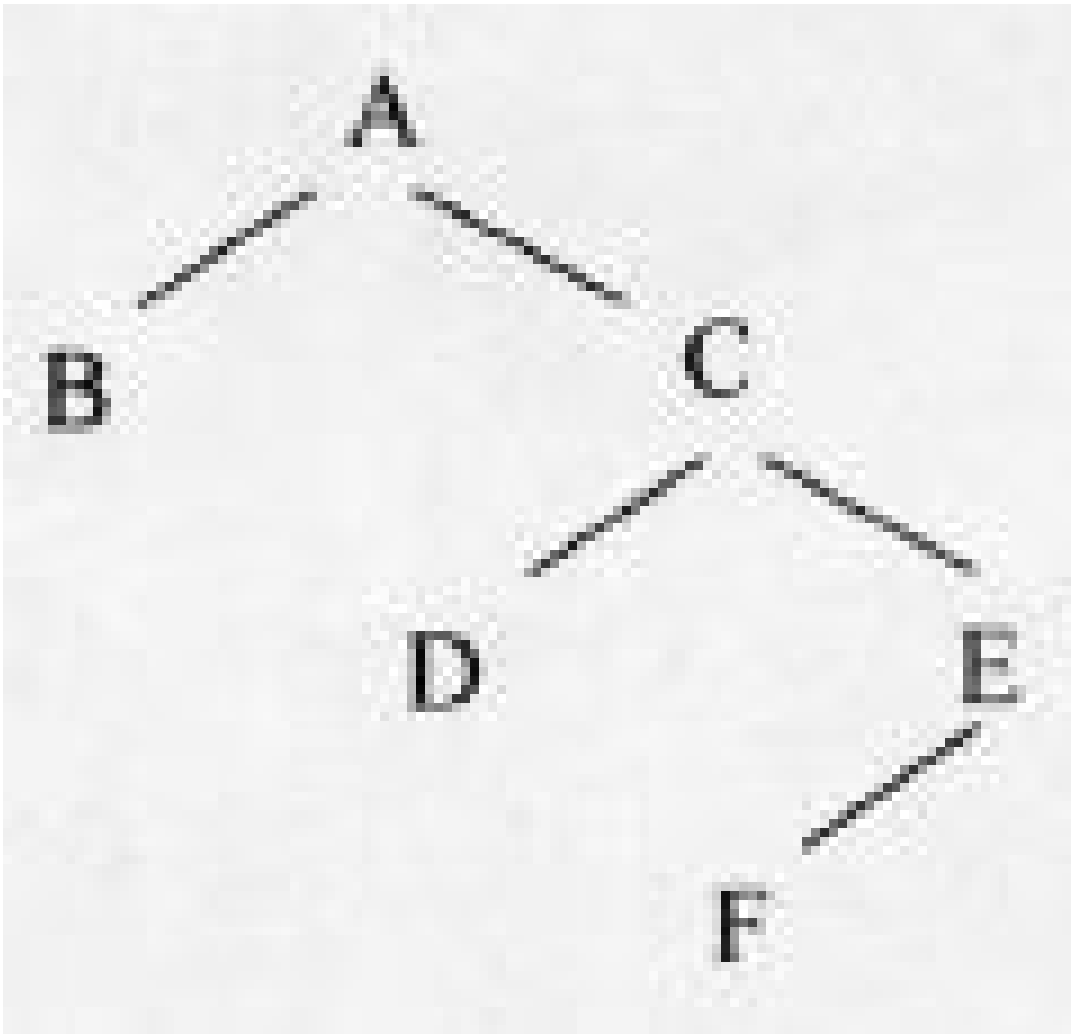
II00IOI000 无合法出栈序列，操作序列中 4 次入栈 6 次出栈也是会产生错误的。

试题五十九 答案： B 解析：

本题考查数据结构基础知识。

对于一个非空的二叉树，其先序遍历序列和中序遍历序列都是唯一确定的。先序遍历是首先访问根结点，其次是先序遍历左子树，最后再先序遍历右子树，因此，先序序列中的第一个元素是根结点。中序遍历是首先中序遍历左子树，然后访问根结点，最后中序遍历右子树，因此，在已知根结点的情况下，可将左子树和右子树的结点区分开。

本题中，根据先序遍历序列，可知树根结点是 A，然后从中序序列得知左子树中只有一个结点(B)，依此类推，可推得该二叉树如下图所示，其高度为 4 。



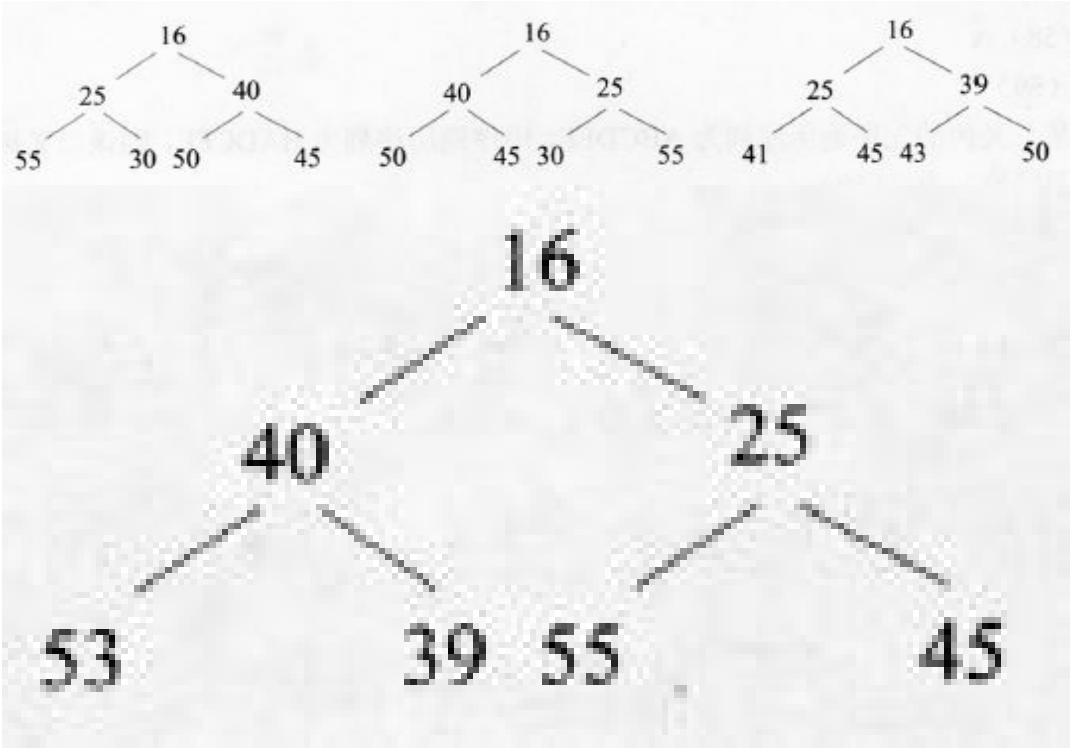
试题六十 答案： D 解析：

本题考查数据结构基础知识。

若将序列中的元素以完全二叉树的方式呈现，则 k_1 与, k_{2i} 、 k_{2i+1} 正好形成父结点、左孩子和右孩子的关系，很容易判断条件 ki≤k2i 且 ki≤k2i+1 是否成立。

题中选项 A 、 B 和 C 的序列如下图所示，树中每个非叶子结点都不大于其左孩子结点和右孩子结点，因此都是小根堆。

选项 D 中序列对应的完全二叉树如下图所示，其中 40 大于其右孩子 39, 因此不是小根堆。

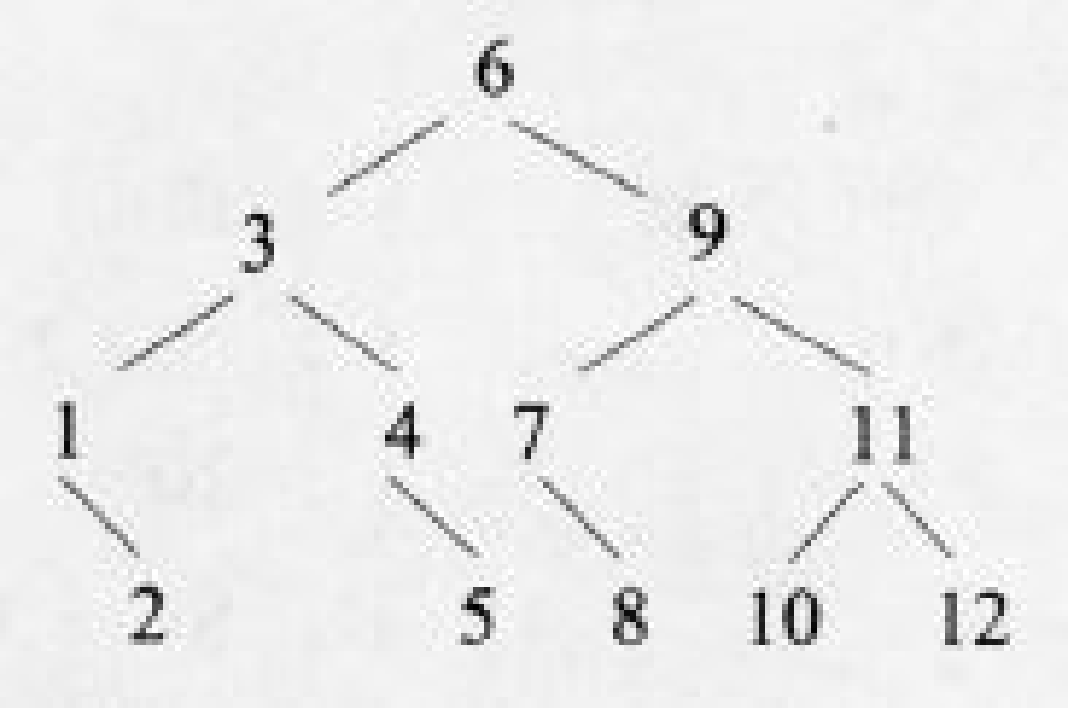


试题六十一 答案： B **解析：**

本题考查数据结构基础知识。

在 12 个元素构成的有序表中进行二分查找的过程可用折半查找判定树表示，如下图所示(数字表示元素的序号)。

若要查找的元素等于 a[9]，则依次与 a[6]、 a[9]进行了比较。



试题六十二 答案： B **解析：**

本题考查算法设计与分析技术的相关知识。

要求考生掌握几个基本的算法设计技术，包括分治法、动态规划、贪心法、回溯法和分支限界法等，并熟悉其典型的应用实例。

根据题干描述，装配线调度问题是一个最优化问题，需要求最短的装配路线。在求解最优解的过程中，利用问题的最优子结构性质(即问题的最优解包含其子问题的最优解)，且分析出问题具有重复子问题的性这是采用动态规划策略求解问题的两个基本要素。

分析题干描述的求解该问题的思路，迭代地求解题干中给的递归式，即一重 n 的循环，因此时间复杂度为 θ (n)。

根据递归式计算 f 数组和 fmin，并构造数组 1 和 fmin 记录到达某个工位的前一个工位来自哪条装配线以及最终从哪条装配线离开，得到下表。

根据上表，得到最短的装配路线的长度为 21 。最短路线从后往前推，首先 1min=1，表示汽车装配完后是从第一条装配线出来的，因此会经过 S_{13}, 然后再查 113=2, 表示 S_{13} 的前一个工位是 S_{22}，再查 S_{22}=1, 表示 S_{22} 的前一个工位是 S_{11}, 于是得到最短路线为 S_{11}-S_{22}-S_{13}。

	S_{11}	S_{12}	S_{13}		fmin
f1j	7	16	18	21	21
f2j	9	13	19	22	
	S_{21}	S_{22}	S_{23}		

	S_{12}	S_{13}	lmin
l1j	1	2	1
l2j	1	2	
	S_{22}	S_{23}	

试题六十三 答案： B 解析：

本题考查算法基础。

题目看似是非常复杂的，涉及到复杂的公式，以及算法逻辑，但如果我们先从后面两个空来分析，问题就简单得多。

求最短装配时间与装配路线，其实是一个求最短路径的过程。此时我们可以把从起点到各个结点的最短路径逐步求出。经过分析得出最短装配路线为： S_{11}->S_{22}->S_{13}，长度为 21 。

解决了一个实际问题后，再来看所谓的迭代公式，其做法与之前手动求最短路径一致，算法是用一个数组将起点到各个结点的最短路径逐个求出，用已求出的最短路径来分析后面的最短路径，所以这符合动态规划法的特征，算法策略应是动态规划法。而算法的复杂度为 θ (n)，因为用一个单重循环就可以解决这个问题。

试题六十四 答案： A 解析：

本题考查算法基础。

题目看似是非常复杂的，涉及到复杂的公式，以及算法逻辑，但如果我们先从后面两个空来分析，问题就简单得多。

求最短装配时间与装配路线，其实是一个求最短路径的过程。此时我们可以把从起点到各个结点的最短路径逐步求出。经过分析得出最短装配路线为： S_{11}->S_{22}->S_{13}，长度为 21 。

解决了一个实际问题后，再来看所谓的迭代公式，其做法与之前手动求最短路径一致，算法是用一个数组将起点到各个结点的最短路径逐个求出，用已求出的最短路径来分析后面的最短路径，所以这符合动态规划法的特征，算法策略应是动态规划法。而算法的复杂度为 θ (n)，因为用一个单重循环就可以解决这个问题。

试题六十五 答案： B 解析：

本题考查算法基础。

题目看似是非常复杂的，涉及到复杂的公式，以及算法逻辑，但如果我们先从后面两个空来分析，问题就简单得多。

求最短装配时间与装配路线，其实是一个求最短路径的过程。此时我们可以把从起点到各个结点的最短路径逐步求出。经过分析得出最短装配路线为： S_{11}->S_{22}->S_{13}，长度为 21 。

解决了一个实际问题后，再来看所谓的迭代公式，其做法与之前手动求最短路径一致，算法是用一个数组将起点到各个结点的最短路径逐个求出，用已求出的最短路径来分析后面的最短路径，所以这符合动态规划法的特征，算法策略应是动态规划法。而算法的复杂度为 θ (n)，因为用一个单重循环就可以解决这个问题。

试题六十六 答案： B 解析：

本题考查 DNS 的基本知识。

DNS 域名查询的次序是：本地的 hosts 文件一>本地 DNS 缓存一>本地 DNS 服务器一>根域名服务器。

试题六十七 答案： C 解析：

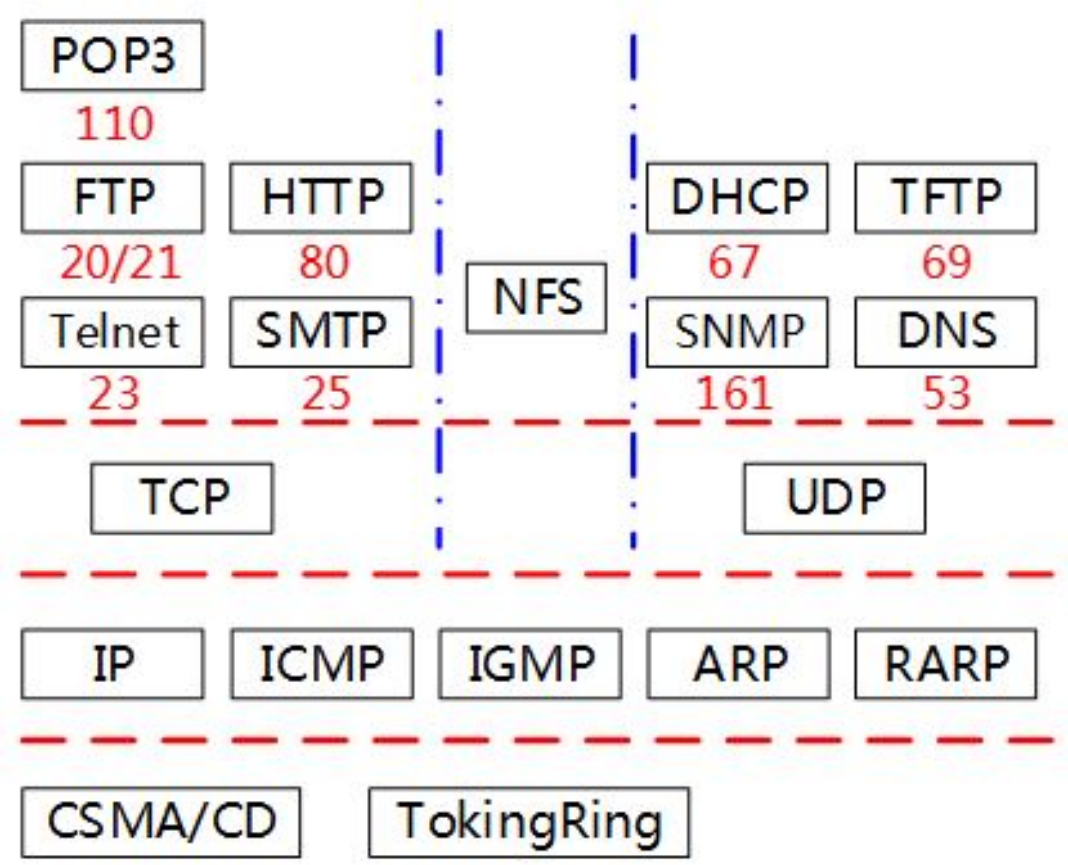
本题考查 Linux 操作系统基础知识。

在 Linux 操作系统中，只有一个根目录，根目录使用“/”来表示。根目录是一个非常重要的目录，其他的文件目录均由根目录衍生而来。

试题六十八 答案： C 解析：

本题考查 TCP/IP 协议栈中的协议与层次关系。

选项 A 、 B 错在第 3 层应为 IP 协议；选项 D 错在 SMTP 采用的传输层协议为 TCP 。



试题六十九 答案： C 解析：

本题考查异步传输协议基础知识。

根据题目中的数据，每秒传送 500 字符，每字符 7 比特，故有效速率为 3500b/s 。

试题七十 答案： D 解析：

本题考查路由策略基础知识。

静态路由是固定路由，从不更新除非拓扑结构发生变化；洪泛式将路由信息发送到连接的所有路由器，不利用网络信息；随机路由是洪泛式的简化；自适应路由依据网络信息进行代价计算，依据最小代价实时更新路由。

试题七十一 答案： A 解析：

软件之美在于它的功能、内部结构以及团队创建它的过程。对用户而言，通过直观、简单的界面呈现出恰当特性的程序就是美的。对软件设计者而言，被简单、直观地分割，并具有最小内部耦合的内部结构就是美的。对开发人员和管理者而言，每周都会取得重大进展，并且生产出无缺陷代码的具有活力的团队就是美的。美存在于所有这些层次之中。

人们需要软件—需要许多软件。 50 年前，软件还只是运行在少量大型、昂贵的机器之上。30 年前，软件可以运行在太多数公司和工业环境之中。现在，移动电话、手表、电器、汽车、玩具以及工具中都运行有软件，并且对更新、更好的软件的需求永远不会停止。随着人类文明的发展和壮大，随着发展中国家不断构建基础设施，随着发达国家努力追求更高的效率，对越来越多的软件的需求不断增加。如果在所有这些软件之中，都没有美存在，这将会是一个很大的遗憾。

我们知道软件可能会是丑陋的。我们知道软件可能会难以使用、不可靠并且是粗制滥造的；我们知道有一些软件系统，界混乱、粗糙的内部结构使得对它们的更改既昂贵又困难；我们还见过那些通过笨拙、编以使用的界面展现其特性的软件系统；我们同样也见过那些易崩溃且行为不当的软件系统。这些都是丑陋的系统。糟糕的是，作为一种职业，软件开发人员所创建出来的美的东西却往往少于丑的东西。

最好的软件开发人员都知道一个秘密：美的东西比丑的东西创建起来更廉价，也更快捷。构建、维护一个美的软件系统所花费的时间、金钱都要少于丑的系统。软件开发新手往往不理解这一点。他们认为做每件事情都必须要快，他们认为美是不实用的。错！由于事情做得过快，他们造成的混乱致使软件僵化，难以理解。美的系统是灵活、易于理解的，构建、维护它们就是一种快乐。丑陋的系统才是不实用的。丑陋会降低你的开发速度，使你的软件昂贵而又脆弱。构建、维护美的系统所花费的代价最少，交付起来也最快。

simple(简单)hard(困难) complex(复杂)duplicated(被复制)

试题七十二 答案： C 解析：

软件之美在于它的功能、内部结构以及团队创建它的过程。对用户而言，通过直观、简单的界面呈现出恰当特性的程序就是美的。对软件设计者而言，被简单、直观地分割，并具有最小内部耦合的内部结构就是美的。对开发人员和管理者而言，每周都会取得重大进展，并且生产出无缺陷代码的具有活力的团队就是美的。美存在于所有这些层次之中。

人们需要软件—需要许多软件。 50 年前，软件还只是运行在少量大型、昂贵的机器之上。30 年前，软件可以运行在太多数公司和工业环境之中。现在，移动电话、手表、电器、汽车、玩具以及工具中都运行有软件，并且对更新、更好的软件的需求永远不会停止。随着人类文明的发展和壮大，随着发展中国家不断构建基础设施，随着发达国家努力追求更高的效率，对越来越多的软件的需求不断增加。如果在所有这些软件之中，都没有美存在，这将会是一个很大的遗憾。

我们知道软件可能会是丑陋的。我们知道软件可能会难以使用、不可靠并且是粗制滥造的；我们知道有一些软件系统，界混乱、粗糙的内部结构使得对它们的更改既昂贵又困难；我们还见过那些通过笨拙、编以使用的界面展现其特性的软件系统；我们同样也见过那些易崩溃且行为不当的软件系统。这些都是丑陋的系统。糟糕的是，作为一种职业，软件开发人员所创建出来的美的东西却往往少于丑的东西。

最好的软件开发人员都知道一个秘密：美的东西比丑的东西创建起来更廉价，也更快捷。构建、维护一个美的软件系统所花费的时间、金钱都要少于丑的系统。软件开发新手往往不理解这一点。他们认为做每件事情都必须要快，他们认为美是不实用的。错！由于事情做得过快，他们造成的混乱致使软件僵化，难以理解。美的系统是灵活、易于理解的，构建、维护它们就是一种快乐。丑陋的系统才是不实用的。丑陋会降低你的开发速度，使你的软件昂贵而又脆弱。构建、维护美的系统所花费的代价最少，交付起来也最快。

happens(发生) exists(存在) stops(停止) starts(开始)

试题七十三 答案： B 解析：

软件之美在于它的功能、内部结构以及团队创建它的过程。对用户而言，通过直观、简单的界面呈现出恰当特性的程序就是美的。对软件设计者而言，被简单、直观地分割，并具有最小内部耦合的内部结构就是美的。对开发人员和管理者而言，每周都会取得重大进展，并且生产出无缺陷代码的具有活力的团队就是美的。美存在于所有这些层次之中。

人们需要软件—需要许多软件。 50 年前，软件还只是运行在少量大型、昂贵的机器之上。30 年前，软件可以运行在太多数公司和工业环境之中。现在，移动电话、手表、电器、汽车、玩具以及工具中都运行有软件，并且对更新、更好的软件的需求永远不会停止。随着人类文明的发展和壮大，随着发展中国家不断构建基础设施，随着发达国家努力追求更高的效率，对越来越多的软件的需求不断增加。如果在所有这些软件之中，都没有美存在，这将会是一个很大的遗憾。

我们知道软件可能会是丑陋的。我们知道软件可能会难以使用、不可靠并且是粗制滥造的；我们知道有一些软件系统，界混乱、粗糙的内部结构使得对它们的更改既昂贵又困难；我们还见过那些通过笨拙、编以使用的界面展现其特性的软件系统；我们同样也见过那些易崩溃且行为不当的软件系统。这些都是丑陋的系统。糟糕的是，作为一种职业，软件开发人员所创建出来的美的东西却往往少于丑的东西。

最好的软件开发人员都知道一个秘密：美的东西比丑的东西创建起来更廉价，也更快捷。构建、维护一个美的软件系统所花费的时间、金钱都要少于丑的系统。软件开发新手往往不理解这一点。他们认为做每件事情都必须要快，他们认为美是不实用的。错！由于事情做得过快，他们造成的混乱致使软件僵化，难以理解。美的系统是灵活、易于理解的，构建、维护它们就是一种快乐。丑陋的系统才是不实用的。丑陋会降低你的开发速度，使你的软件昂贵而又脆弱。构建、维护美的系统所花费的代价最少，交付起来也最快。

starts（开始） continues(持续） appears(出现)stops(停止)

试题七十四 答案： D **解析：**

软件之美在于它的功能、内部结构以及团队创建它的过程。对用户而言，通过直观、简单的界面呈现出恰当特性的程序就是美的。对软件设计者而言，被简单、直观地分割，并具有最小内部耦合的内部结构就是美的。对开发人员和管理者而言，每周都会取得重大进展，并且生产出无缺陷代码的具有活力的团队就是美的。美存在于所有这些层次之中。

人们需要软件—需要许多软件。 50 年前，软件还只是运行在少量大型、昂贵的机器之上。30 年前，软件可以运行在太多数公司和工业环境之中。现在，移动电话、手表、电器、汽车、玩具以及工具中都运行有软件，并且对更新、更好的软件的需求永远不会停止。随着人类文明的发展和壮大，随着发展中国家不断构建基础设施，随着发达国家努力追求更高的效率，对越来越多的软件的需求不断增加。如果在所有这些软件之中，都没有美存在，这将会是一个很大的遗憾。

我们知道软件可能会是丑陋的。我们知道软件可能会难以使用、不可靠并且是粗制滥造的；我们知道有一些软件系统，界混乱、粗糙的内部结构使得对它们的更改既昂贵又困

难；我们还见过那些通过笨拙、编以使用的界面展现其特性的软件系统；我们同样也见过那些易崩溃且行为不当的软件系统。这些都是丑陋的系统。糟糕的是，作为一种职业，软件开发人员所创建出来的美的东西却往往少于丑的东西。

最好的软件开发人员都知道一个秘密：美的东西比丑的东西创建起来更廉价，也更快捷。构建、维护一个美的软件系统所花费的时间、金钱都要少于丑的系统。软件开发新手往往不理解这一点。他们认为做每件事情都必须要快，他们认为美是不实用的。错！由于事情做得过快，他们造成的混乱致使软件僵化，难以理解。美的系统是灵活、易于理解的，构建、维护它们就是一种快乐。丑陋的系统才是不实用的。丑陋会降低你的开发速度，使你的软件昂贵而又脆弱。构建、维护美的系统所花费的代价最少，交付起来也最快。

practical （实用的）useful（有用的） beautiful（美丽的）ugly（丑陋的）

试题七十五 答案： A 解析：

软件之美在于它的功能、内部结构以及团队创建它的过程。对用户而言，通过直观、简单的界面呈现出恰当特性的程序就是美的。对软件设计者而言，被简单、直观地分割，并具有最小内部耦合的内部结构就是美的。对开发人员和管理者而言，每周都会取得重大进展，并且生产出无缺陷代码的具有活力的团队就是美的。美存在于所有这些层次之中。

人们需要软件—需要许多软件。 50 年前，软件还只是运行在少量大型、昂贵的机器之上。30 年前，软件可以运行在太多数公司和工业环境之中。现在，移动电话、手表、电器、汽车、玩具以及工具中都运行有软件，并且对更新、更好的软件的需求永远不会停止。随着人类文明的发展和壮大，随着发展中国家不断构建基础设施，随着发达国家努力追求更高的效率，对越来越多的软件的需求不断增加。如果在所有这些软件之中，都没有美存在，这将会是一个很大的遗憾。

我们知道软件可能会是丑陋的。我们知道软件可能会难以使用、不可靠并且是粗制滥造的；我们知道有一些软件系统，界混乱、粗糙的内部结构使得对它们的更改既昂贵又困难；我们还见过那些通过笨拙、编以使用的界面展现其特性的软件系统；我们同样也见过那些易崩溃且行为不当的软件系统。这些都是丑陋的系统。糟糕的是，作为一种职业，软件开发人员所创建出来的美的东西却往往少于丑的东西。

最好的软件开发人员都知道一个秘密：美的东西比丑的东西创建起来更廉价，也更快捷。构建、维护一个美的软件系统所花费的时间、金钱都要少于丑的系统。软件开发新手往往不理解这一点。他们认为做每件事情都必须要快，他们认为美是不实用的。错！由于事情做得过快，他们造成的混乱致使软件僵化，难以理解。美的系统是灵活、易于理解的，构建、维护它们就是一种快乐。丑陋的系统才是不实用的。丑陋会降低你的开发速度，使你

的软件昂贵而又脆弱。构建、维护美的系统所花费的代价最少，交付起来也最快。

impractical （不实用的)perfect(完美的)time-wasting(浪费时间） practical(实用的)