中级软件设计师下午试题模拟63

试题一

阅读下列说明和数据流图，回答问题1至问题3。

[说明]

图书管理系统旨在用计算机对图书进行管理，包括图书的购入、借阅、归还以及注销。管理人员可以查询某位读者、某种图书的借阅情况，还可以对当前图书借阅情况进行一些统计，给出统计表格，以便掌握图书的流通情况。

系统要实现以下四方面的功能：购入新书、读者借书、读者还书以及图书注销。

1购入新书：需要为该书编制图书卡片，包括分类目录号、图书流水号(要保证每本书都有唯一的流水号，即使同类图书也是如此)、书名、作者、内容摘要、价格和购书日期等信息，写入图书目录文件中。

2读者借书：填写借书单，包括读者号、欲借图书分类目录号，系统首先检查该读者号是否有效，若无效，则拒绝借书，否则进一步检查该读者所借图书是否超过最大限制数，若已达到最大借阅数，则拒绝借书，否则读者可以借出该书，登记图书分类目录号、图书流水号、读者号和借阅日期等，写回到借书文件中去。

3读者还书：根据图书流水号，从借书占文件中读出和该图书相关的借阅记录，表明还书日期，再写回借书文件中；如果图书逾期未还，则处以相应罚款。

4图书注销：将一些过时或无保留价值的图书注销，从图书文件中删除相关记录。

5流通查询：管理员可以对图书流通情况进行查询，包括某位读者、某种图书和全局图书，给出流通情况统计表。

以下是经分析得到的数据流图及部分数据字典，有些地方有待填充，假定顶层数据流图是正确的。图1是顶层数据流图，图2是第0层数据流图，图3是第1层数据流图。

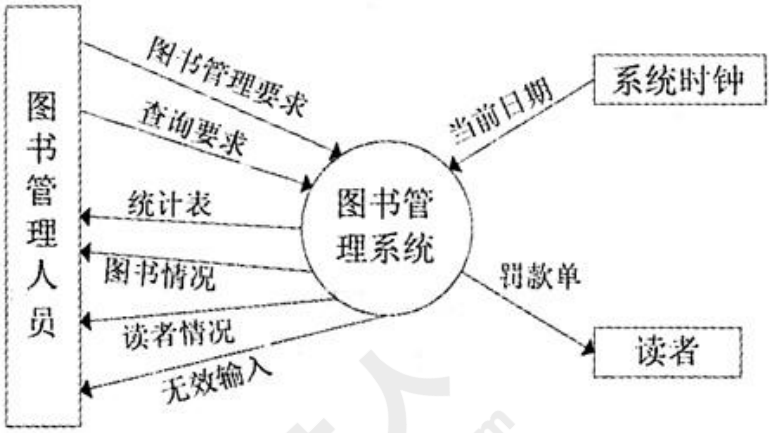


图1

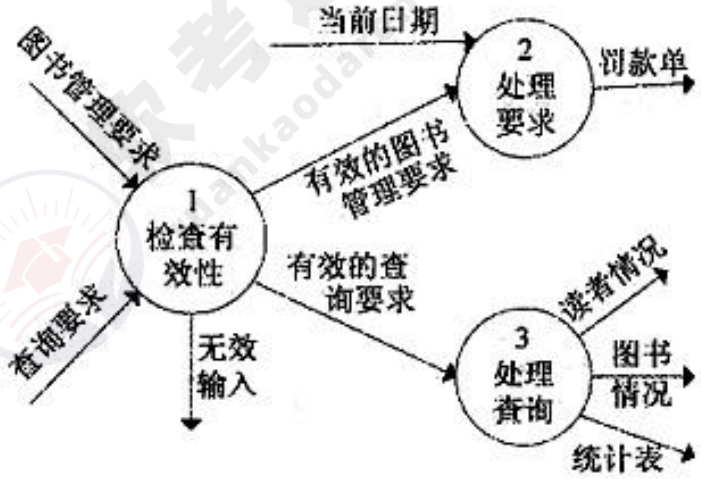


图2

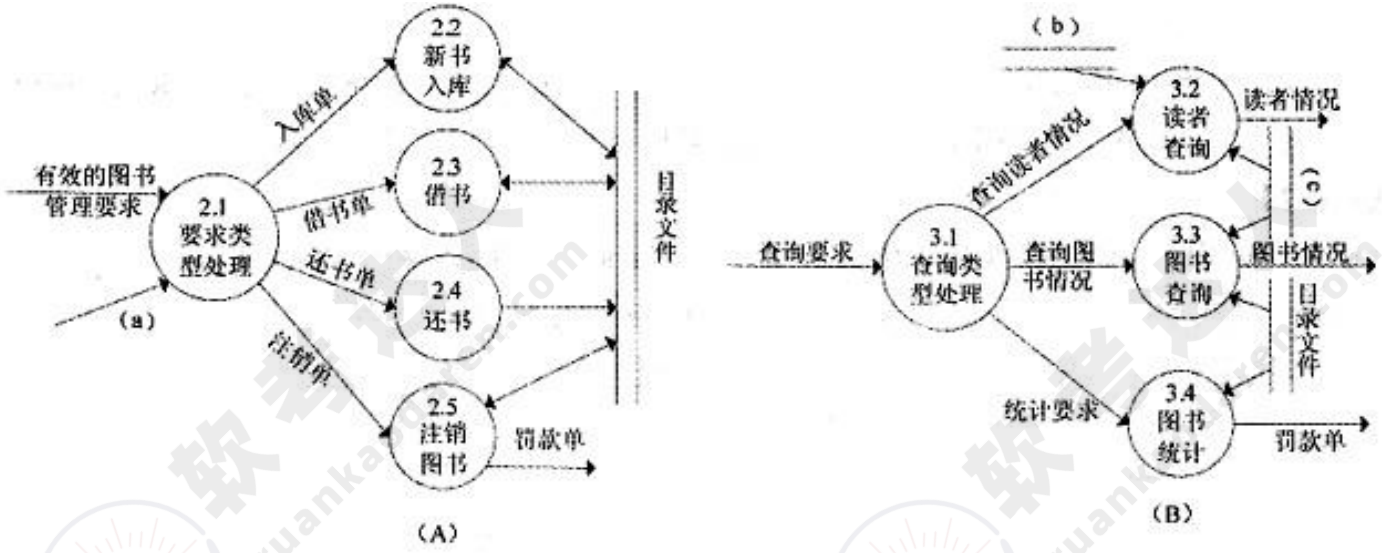


图3

[数据字典]

1数据流条目

图书管理要求=[入库单|借书单|还书单|注销单]

入库单=分类目录号+数量+书名+作者+内容摘要+价格+购书日期

借书单=读者号+__(d)__+借阅日期

还书单=__(e)__+还书日期

2文件说明

文件名：目录文件组成：{分类目录号+书名+作者+内容摘要+价格+入库日期+总数+库存数+(f)}

1、根据题意，指出数据流图中缺失的数据流(a)的名称，并指出该数据流的起点。

2、将下述文件正确填充在数据流图(b)、(c)处：读者文件、借书文件。

3、根据题意，补充数据字典中(d)、(e)、(f)处的空缺。

试题二

阅读下列说明和E-R图，回答问题1至问题3。

[说明]

有个关于运动会的管理系统，在该系统中，委员会为每一个参赛的运动员赋以一个唯一的编号"运动员号"，同时记录姓名、性别、年龄和队名，姓名和队名必须填写。

一个运动员属于且只属于一个队，一个运动员可以参赛多个项目。运动员参加比赛取得一个成绩，相应有一个积分：第一名积分6分，第二名积分4分，第三名积分2分，其他的没有积分。一个队的总积分是该队的所有队员的积分之和。

下图是该系统的E-R图。图中的实体和属性同时给出了中英文两种名字，回答问题时只需写出英文名即可。

4、根据E-R图中给出的词汇，按照"有关模式名(属性，属性，...)"的格式，将此E-R图转换为3个关系模式，指出每个关系模式中的主码和外码，其中模式名根据需要取实体名或联系名。

5、创建Athlete表时，ANo使用CHAR(6)并且唯一，AName使用CHAR(20)，ASex使用CHAR(1)，ATeam使用CHAR(20)。请在下列用于创建表Athelete的SQL语句空缺处填入正确的内容。

```
CREATE TABLE Athlete(ANo CHAR(6) NOT NULL,
AName CHAR(20),
ASex CHAR(1),
ATeam CHAR(20) NOT NULL,
_______);
```

6、假定Games表存储参赛情况，如下的SQL语句是委员会用于查询"队名为'China'的各个运动员各自夺取得的总积分"的不完整语句，请在空缺处填入正确的内容。

```
SELECT _______
FROM Games
WHERE ANo _______
(SELECT ANo
FROM _______
WHERE ATeam = "China")
GROUP BY ANo;
```

试题三

阅读下列说明和图，回答问题1至问题3。

[说明]

某大型旅店为了便于管理，欲开发一个客房管理系统。希望实现客房预定、入住登记、帐务结算、退房，以及将服务项目记入客人帐单。

旅客包括散客和团体，散客预定或入住时需要提供姓名、性别、身份证和联系电话，团体则提供团体名称、负责人的姓名、性别、身份证和联系电话，以及团体人数。对于散客，还要提供换房。

旅店还提供了很多服务项目，比如早餐。对每一个入住客人，服务列表记录了住宿期间的各项服务，包括服务类型、日期、数量等。当然，客人也可以不要任何服务。

旅店的客房有一个唯一的房间号，分为不同的类别，不同的房间床位数和价格不同。

为了有效的管理，需要记录每天的客房状态。客房的状态有：空闲、占用、已预定和维修。

- 客人入住后，客房处于占用状态；
- 客人退房后，客房处于空闲状态；
- 客人预定后，客房处于已预定状态；
- 预定客人入住后，客房处于占用状态；
- 预定客人取消预定后客房处于空闲状态；
- 需要维修时客房处于维修状态；
- 维修完成后客房处于空闲状态。

该系统采用面向对象方法开发，系统中的类以及类之间的关系用UML类图表示，图1是该系统的类图的一部分，图2描述了客房状态的转变情况。

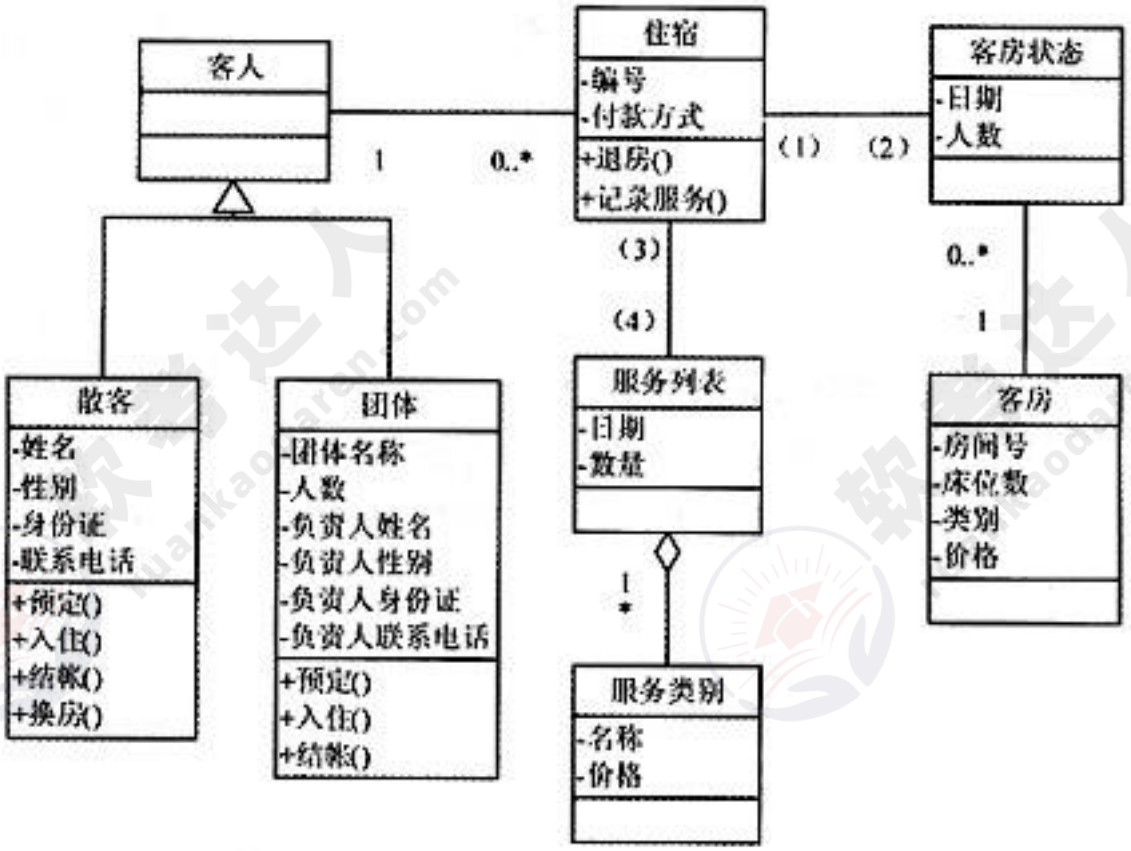


图1

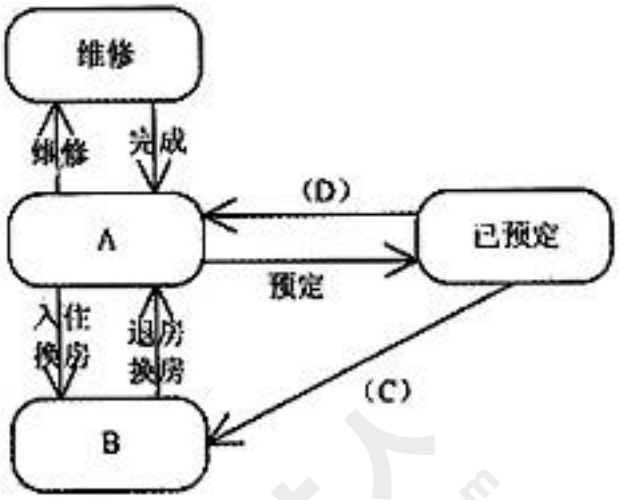


图2

7、请用图1的属性和方法的名称给出客人类的属性和方法。(注意：团体类中的负责人姓名等与散客的对应属性含义相同，不必区分)

8、在UML中，重复度(Multiplicity)定义了某个类的一个实例可以与另一个类的多少个实例相关联。通常把它写成一个表示取值范围的表达式或者一个具体的值。例如图1中的类客人和住宿，客人端的"1"表示：一个住宿类的实例只能与一个1个客人类的实例相关联；住宿类端的"0..*"表示：一个住宿类的实例可以与0个或多个客人类的实例相关。请指出图1中__(1)__到__(4)__的重复度分别为多少？

9、根据题意，请指出图2中状态A、B分别是什么状态，事件C、D分别是什么事件。

试题四

阅读下列说明和图，回答问题1到问题3。

[说明]

目前大多数操作系统都采用虚拟存储技术，这样可在较小的可用内存中执行较大的用户程序，可在内存中容纳更多程序并发执行。

引入虚拟存储技术，其基本思想是利用大容量的外存来扩充内存，产生一个比有限的实际空间大得多、逻辑的虚拟内存空间，以便能够有效地支持多道程序系统的实现和大型程序运行的需要，从而增强系统的处理能力。

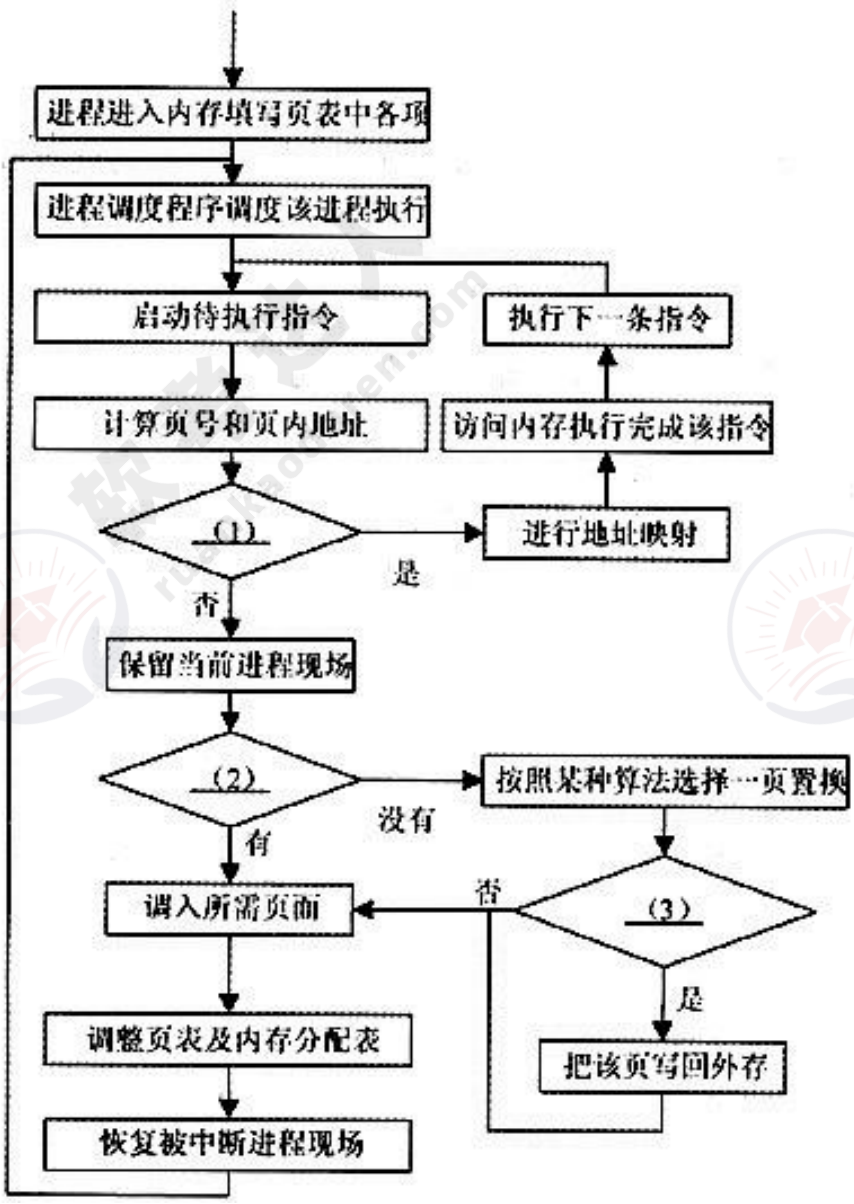
虚拟存储技术主要分为虚拟页式存储管理和虚拟段式存储管理。

虚拟页式存储管理中，在进程开始运行之前，不是装入全部页面，而是装入一个或零个页面之后根据进程运行的需要，动态装入其他页面；当内存空间已满，而又需要装入新的页面时，则根据某种算法淘汰某个页面，以便装入新的页面。在简单页式存储管理的基础上，增加请求调页和页面置换功能。

使用虚拟页式存储管理时需要在页表中增加以下内容：页号、驻留号、内存块号、外存地址、访问位、修改位。其中，驻留位，又称中断位，表示该页是在内存还是在外存；访问位表示该页在内存期间是否被访问过；修改位表示该页在内存中是否被修改过。访问位和修改位可以用来决定置换哪个页面，具体由页面置换算法决定。

10、执行指令时，计算页号与页内地址，判断"该页在内存吗"，若在，则进行地址映射过程；若不在内存，则产生缺页中断。当发生缺页中断时，保存当前进程现场，判断"有空闲页面吗"，如有，直接调入所需的页面。若没有，按照某种算法选择一页置换，判断"该页被修改过吗"，如果被修改过，就必须把它写回磁盘以便更新该页在磁盘上的副本：如果该页没有被修改过，那么它在磁盘上的副本已经是最新的了，则不需要写回，调入的所需的页面直接覆盖被淘汰的页。调整页表及内存分配表，恢复被中断进程现场。

补充缺页中断处理流程图中的判断(1)～(3)。



11、发生缺页时，通常需要进行页面置换，页面置换算法的优劣将会影响虚拟存储系统的性能。常用的页面置换算法有理想页面置换算法(OPT: Optimal)、先进先出页面置换算法(FIFO: First-In First-Out)以及最近最少使用页面置换算法(LRU: Least Recently Used)。

某程序在内存中分配3页，初始为空，页面走向为4、3、2、1、4、3、5、4、3、2、1、5。给出采用先进先出(FIFO)、最近最少使用(LRU)和理想(OPT)页面置换算法所得到的内存中的页面变化序列。

注：缺页标记栏，用○表示没有缺页，用×表示发生了缺页。

OPT	4	3	2	1	4	3	5	4	3	2	1	5
页1												
页2												
页3												
缺页标记												

FIFO	4	3	2	1	4	3	5	4	3	2	1	5
页1												
页2												
页3												

缺页标记												

LRU	4	3	2	1	4	3	5	4	3	2	1	5
页1												
页2												
页3												
缺页标记												

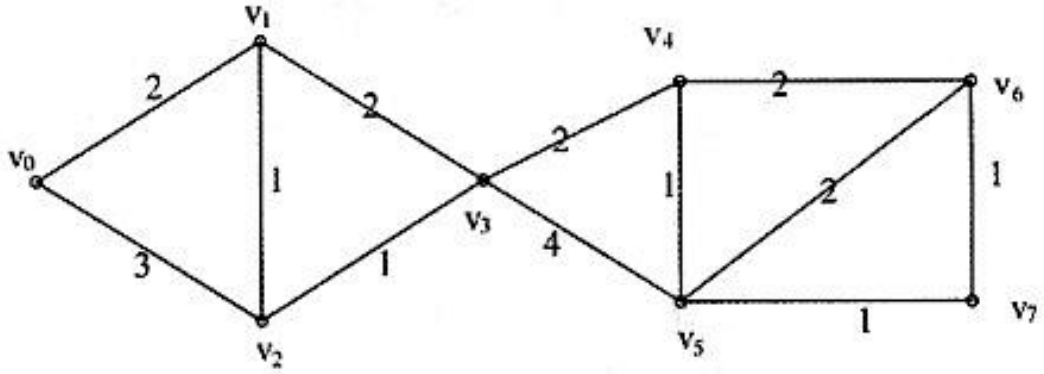
12、简述Belady异常。

试题五

阅读下列函数说明和C代码，将应填入画横线处的字句写在对应栏内。

13、[说明]

若要在N个城市之间建立通信网络，只需要N-1条线路即可。如何以最低的经济代价建设这个网络，是一个网的最小生成树的问题。现要在8个城市间建立通信网络，其间拓扑结构如图所示，边表示城市间通信线路，边上标示的是建立该线路的代价。



无向图用邻接矩阵存储，元素的值为对应的权值。考虑到邻接矩阵是对称的且对角线上元素均为0，故压缩存储，只存储上三角元素(不包括对角线)。

现用Prim算法生成网络的最小生成树。由网络G=(V,E.构造最小生成树T=(U, TE.的Prim算法的基本思想是：首先从集合V中任取一顶点放入集合U中，然后把所有一个顶点在集合U里、另一个顶点在集合V-U里的边中，找出权值最小的边(u, v)，将边加入TE，并将顶点v加入集合U，重复上述操作直U=V为止。

函数中使用的预定义符号如下：

```
#define MAX 32768 /*无穷大权，表示顶点间不连通*/
#define MAXVEX 30 /*图中顶点数目的最大值*/
typedef struct{
int StartVex, StopVex;                          /*边的起点和终点*/
float weight;                                    /*边的权*/
}Edge;
typedef struct{
char vexs [MAXVEX];                              /*顶点信息*/
float arcs [MAXVEX* (MAXVEX-1) /2];    /*邻接矩阵信息，压缩存储*/
int n;                                           /*图的顶点个数*/
}Graph;
[函数]
void PrimMST(Graph *pGraph, Edge mst [])
{
```

```
int i, j, k, min, vx, vy;
float weight, minWeight;
Edge edge;
for(i = 0;i < pGraph->n-1; i++) {
mst[i].StartVex = 0;
mst[i].StopVex = i+1;
mst[i].weight = pGraph->arcs[i];
}
for(i = 0; i <_____; i++) {/*共n-1条边*/
minWeight = (float)MAX;
min = i;
/*从所有边(vx, vy)中选出最短的边*/
for(j = i; j < pGraph->n-1; j++) {
if(mst[j].weight < minWeight) {
minWeight =_____;
min = j;
}
}
/*mst[min]最短的边(vx, vy), 将mst[min]加入最小生成树*/
edge = mst[min];
mst[min] = mst[i];
met[i] = edge;
vx = _____; /*vx为刚加入最小生成树的顶点下标*/
/*调整mst[i+1]到mst[n-1]*/
for(j = i+1; j < pGraph->n-1; j++) {
vy = mst[j].StopVex;
if _____ {/*计算(vx, vy)对应的边在压缩矩阵中的下标*/
k = pGraph->n*vy-vy*(vy+1)/2+vx-vy-1;
}else{
k = pGraph->n*vx-vx* (vx+1)/2+vy-vx-1;
}
weight = _____;
if(weight < mst[j].weight) {
mst[j].weight = weight;
mst[j].StartVex = vx;
}
}
}
}
```

试题六

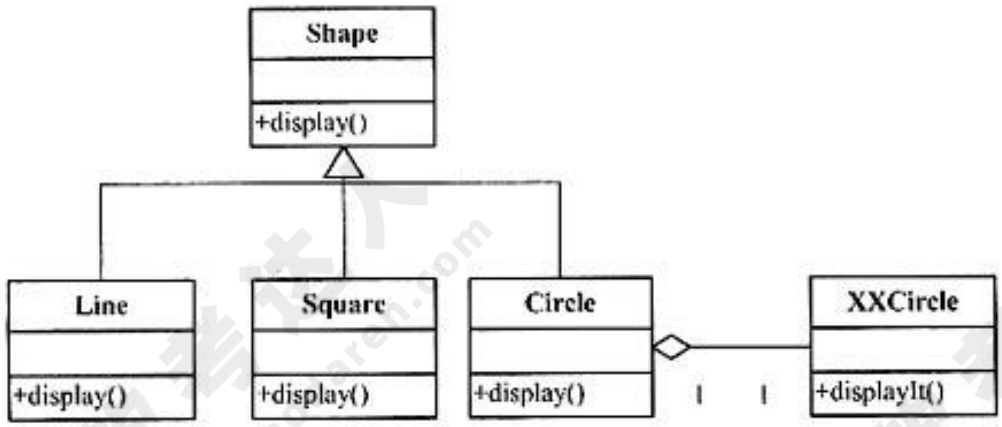
阅读以下说明和C++代码，将应填入画横线处的字句写在对应栏内。

14、[说明]

现有一个显示系统，要显示的图形有线Line、矩形Square，抽象出一个Shape类(接口)，有方法显示display()。

需要新增图形Circle，又已知有类XXCircle实现了所需要实现的功能：显示displayIt()。为了继承自Shape以提供统一接品，又不希望从头开发代码，希望使用XXCircle。这样将XXCircle作为Circle的一个属性，即Circle的对象包含一个XXCircle对象。当一个Circle对象被实例化时，它必须实例化一个相应的XXCircle对象；当Circle对象收到的做任何事的请求都将转发给这个XXCircle对象。通过这种称为Adapter模式，Circle对象就可以通过“让XXCircle做实际工作”

来表现自己的行为了。图显示了各个类间的关系。以下是C++语言实现，能够正确编译通过。



[C++代码]

```
class Shape{
public:
______ void display() = 0;
};
class Line : public Shape{//省略具体实现
};
class Square : public Shape{//省略具体实现
};
class XXCircle{
public:
void displayIt() {
//省略具体实现
}
//省略其余方法和属性
};
class Circle : public Shape{
private:
XXCircle *pxc;
public:
Circle();
void display();
};
Circle::Circle() {
pxc = ______;
}
void Circle::display()
{
pxc-> _____;
}
class Factory{
public:
_____ getShapeInstance(int type) { //生成特定实例
switch(type) {
case 1 : return new Square;
case 2 : return new Line;
case 3 : return new Circle;
default : return NULL;
}
}
};
```

```
void main(int argc, char *argv[]) {
if(argc !=2) {
cout<<"error parameters  ! "<<end1;
return;
}
int type=atoi(argv[1]);
Factory factory;
Shape *s = factory._____;
if(s==NULL) {
cout<<"Error get the instance!"<<end1;
return;
}
s->display( );
delete s;
return;
}
```

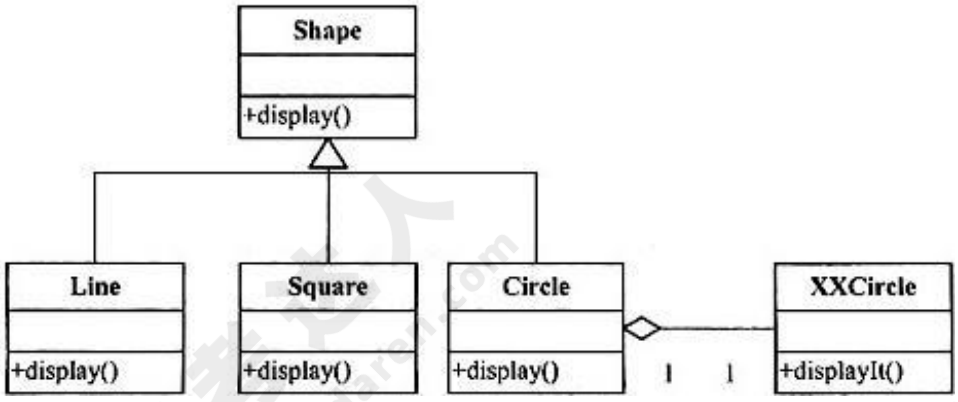
试题七

阅读以下函数说明和*Java*代码，将应填入画横线处的字句写在对应栏内。

15、[说明]

现有一个显示系统，要显示的图形有线Line、矩形Square，抽象出一个Shape类(接口)，有方法显示display()。

需要新增图形Circle，又已知有类XXCircle实现了所需要实现的功能：显示displayIt()。为了继承自Shape以提供统一接口，又不希望从头开发代码，希望使用XXCircle。这样将XXCircle作为Circle的一个属性，即Circle的对象包含一个XXCircle对象。当一个Circle对象被实例化时，它必须实例化一个相应的XXCircle对象；当Circle对象收到的做任何事的请求都将转发给这个XXCircle对象。通过这种称为Adapter模式，Circle对象就可以通过“让XXCircle做实际工作”来表现自己的行为了。图显示了各个类间的关系。以下是JAVA语言实现，能够正确编译通过。



[Java代码]

```
//Shape.java文件
public interface Shape {
public _____ void display();
//XXCircle.java文件
public class xxCircle {
public void displayIt() {
//省略具体实现
}
}
//circle.java文件
public class Circle _____ Shape {
```

```
private XXCircle pcx = _______;
public void display() {
pcx.displayIt();
}
}
//mactory.java文件
public class Factory {
public _______ getShapeInstance(int type) {
switch(type) {
case 1 : return new Line();
case 2 : return new Square();
case 3 : return new Circle();
default : return null;
}
}
}
//Main.java文件
public class Main {
public static void main(String[] args)
int type=1;
Factory factory = new Factory();
Shape s;
s = factory._______;
if(s==null) {
System.out.println("Error get the instance!");
return;
}
s.display();
return;
}
}
```

答案：

试题一

1、(a)名称：当前日期，起点：系统时钟

加工2的输入数据流有"当前日期"和"有效的图书管理要求"。根据平衡原则，加工2.1的输入数据流(a)应为"当前日期"，其起点自然是"系统时钟"。

2、(b)读者文件　(c)借书文件

加工3.2"读者查询"需要"借书文件"及"读者文件"，而加工3.3"图书查询"与"读者文件"不相关，因此b处应填"读者文件"，c处应填"借书文件"。

3、(d)分类目录号　(e)图书流水号　(f)｛图书流水号｝

根据说明"填写借书单，包括读者号、欲借图书分类目录号"可得，借书单应包括"读者号"和"图书分类号"。故d应填"分类目录号"。

还书时是根据"图书流水号"的，因此还书单应包括"图书流水号"。故e应填图书流水号。

"目录文件"存储图书的情况，包括分类目录号、图书流水号、书名、作者、内容摘要、价格，可见"目录文件"需要存储图书流水号，而且"一种"书有多个副本，因此f应填｛图书流水号｝。注意，大括号｛｝表示多个数据项。

试题二

4、Athlete(ANo, AName, ASex, Age, ATeam)，主键为ANo。

Item(INo, IName, ITime, IPlace)，主键为INo。

Games(ANo, INo, Score, Credit)，主键为(ANo, INo)。

E-R模型向关系模型的转换应遵循如下原则：

①每个实体类型转换成一个关系模式；

②一个1:1的联系(一对一联系)可转换为一个关系模式，或与任意一端的关系模式合并。若独立转换为一个关系模式，那么两端关系的码及其联系的属性为该关系的属性；若与一端合并，那么将另一端的码及属性的属性合并到该端。

③一个1:n的联系(一对多联系)可转换为一个关系模式，或与n端的关系模式合并。若独立转换为一个关系模式，那么两端关系的码及其联系的属性为该关系的属性，而n端的码为关系的码。

④一个n:m的联系(多对多联系)可转换为一个关系模式，两端关系的码及其联系的属性为该关系的属性，而关系的码为两端实体的码的组合。

⑤三个或三个以上多对多的联系可转换为一个关系模式，诸关系的码及联系的属性为关系的属性，而关系的码为各实体的码的组合。

⑥具有相同码的关系可以合并。

根据上述规则，可得如下关系模式：

Athlete(ANo，AName，ASex，Age，ATeam)，主键为ANo。

Item(INo，IName，ITime，IPlace)，主键为INo。

Games(ANo，INo，Score，Credit)，主键为(ANo，INo)。

5、PRIMARY KEY ANo

Athlete表中ANo是主键，创建表时需要说明主键，故空应填PRIMARY KEY ANo。

6、SUM(Credit)

IN

Athlete

SELECT [ALLIDISTINCT]＜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＞1]

[ORDER BY＜列名2＞[ASC|DESC]…]

子句顺序为SELECT、FROM、WHERE、GROUP BY、HAVING、ORDER BY，但SELECT和FROM是必须的，HAVING子句只能与GROUP BY搭配起来使用。SELECT子句对应的是关系代数中的投影运算，用来列出查询结果中的属性，其输出可以是列名、表达式、集函数(AVG、COUNT、MAX、MIN、SUM)，DISTINCT选项可以保证查询的结果集中不存在重复元组；FROM子句对应的是关系代数中的笛卡儿积，它列出的是表达式求值过程中须扫描的关系；WHERE子句对应的是关系代数中的选择谓词。

试题三

7、属性：姓名、性别、身份证、联系电话

方法：预定、入住、结帐

"客人"类是"散客"类和"团体"类的泛化，具有二者的公共属性和公共方法。比对二者属性及方法得，"客人"类属性有：姓名、性别、身份证、联系电话；方法有：预定、入住、结账。

8、(1)0..1 (2)1..* (3)1 (4)0..1

散客入住时只改变一个客房状态，而团体入住时则有可能改变多个客房状态；客房状态改变不一定是住宿导致的，客房维修同样改变客房状态。因此(1)处应填0..1，(2)处应填1..*。

客人可以有多项服务，但只需用一张"服务列表"，当然也可能不需要任何服务；而一张服务列表必然属于而且只输于一个住宿。因此(3)处应填1，(4)处应填0..1。

9、A：空闲 B：占用 C：入住 D：取消预定

"维修"完成后客房处于"空闲"状态，故状态A为"空闲"；客人入住后，客房由"空闲"转为"占用"，故状态B为"占用"；状态"已预订"经"入住"转为"占用"，故C为"入住"；状态"已预订"经"取消"转为"空闲"，故D为"取消"。

试题四

10、(1)该页在内存吗?

(2)有空闲页面吗?

(3)该页被修改过吗?

根据缺页中断处理的说明，易于判断：(1)该页在内存吗?(2)有空闲页面吗?(3)该页被修改过吗?

11、OPT7次

OPT	4	3	2	1	4	3	5	4	3	2	1	5
页1	4	3	2	1	1	1	5	5	5	2	1	1
页2		4	3	3	3	3	3	3	3	5	5	5
页3			4	4	4	4	4	4	4	4	4	4
缺页标记	×	×	×	×	o	o	×	o	o	×	×	o

FIFO9次

FIFO	4	3	2	1	4	3	5	4	3	2	1	5
页1	4	3	2	1	4	3	5	5	5	2	1	1
页2		4	3	2	1	4	3	3	3	5	2	2
页3			4	3	2	1	4	4	4	3	35	5
缺页标记	×	×	×	×	×	×	×	o	o	×	×	o

LRU10次

LRU	4	3	2	1	4	3	5	4	3	2	1	5
页1	4	3	2	1	4	3	5	4	3	2	1	5
页2		4	3	2	1	4	3	5	4	3	2	1
页3			4	3	2	1	4	3	5	4	3	2
缺页标记	×	×	×	×	×	×	×	o	o	×	×	×

常用页面置换算法有：

(a)理想页面置换算法(OPT: Optimal)。选择淘汰不再使用或最远的将来才使用的页。

(b)先进先出页面置换算法(FIFO: First-In First-Out)。选择淘汰主存驻留时间最长的页。

(c)最近最少使用页面置换算法(LRU: Least Recently Used)。选择淘汰离当前时刻最近的一段时间使用得最少的页。

(d)随机算法(Rand)。随机地选择淘汰的页。

(e)最近未使用页面置换算法(NFU: Not Recently Used)。

12、一般来讲，在内存中的物理页面数越多，程序的缺页次数应该越少，但令人吃惊的是，实际情况并不是这样。使用FIFO算法时，有时会出现分配的页面数增多，缺页率反而提高的异常现象。

试题五

13、pGraph->n-1

mst[j].weight

mst[i].StopVex

vy<vx

pGraph->arcs[k]

试题六

14、virtual

new XXCircle

displayIt()

Shape*

getShapeInstance(type)

试题七

15、abstract

implements

```
new XXCircle()
Shape
getShapeInstance(type)
```