

[真题][答案] 2015年下半年 软件设计师 下午试卷 案例

试题数量：6 道

以下所有试题由 软考在线 免费智能真题库 提供

难度系数 错误率 由 软考在线 用户做题大数据支撑 最后更新日期：2016-09-05

本文档最后更新日期：2016-09-05

难度系数 错误率 题目 文档 更新频率：每天

软考在线 <http://www.rkpass.cn> -- 最专业的一站式软考复习平台

全网独家 免费智能真题库 定制学习计划

专业致力于全国计算机技术与软件专业资格(水平)考试

第1题 2015年下半年 下午试卷 案例 软件设计师

【说明】

某慕课教育平台欲添加在线作业批改系统，以实现高效的作业提交与批改，并进行统计。学生和讲师的基本信息已经初始化为数据库中的学生表和讲师表。系统的主要功能如下：

- (1)提交作业。验证学生标识后，学生将电子作业通过在线的方式提交，并进行存储。系统给学生发送通知表明提交成功，通知中包含唯一编号；并通知讲师有作业提交。
- (2)下载未批改作业。验证讲师标识后，讲师从系统中下载学生提交的作业。下载的作业将显示在屏幕上。
- (3)批改作业。讲师按格式为每个题目进行批改打分，并进行整体评价。
- (4)上传批改后的作业。将批改后的作业（包括分数和评价）返回给系统，进行存储。
- (5)记录分数和评价。将批改后的作业的分数和评价记录在学生信息中，并通知学生作业已批改口
- (6)获取已批改作业。根据学生标识，给学生查看批改后的作业，包括提交的作业、分数和评价。
- (7)作业抽检。根据教务人员标识抽取批改后的作业样本，给出抽检意见，然后形成抽检报告给讲师。

。

现采用结构化方法对在线作业批改系统进行分析与设计，获得如图1-1所示的上下文数据流图和图1-

2所示的0层数据流图。

问题1.1 使用说明中的词语，给出图1-1中的实体E1 ~ E3的名称。

参考答案：E1：学生 E2：讲师 E3：教务人员

问题1.2 使用说明中的词语，给出图1-2中的数据存储D1 ~ D4的名称。

参考答案：D1：作业 D2：学生表 D3：讲师表 D4：批改后的作业

问题1.3 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。

参考答案：数据流名称：通知 起点：提交作业 终点：E1

数据流名称：抽检报告 起点：作业抽检 终点：E2

数据流名称：分数和评价 起点：记录分数和评价 终点：D2

数据流名称：通知 起点：记录分数和评价 终点：E1

问题1.4 若发送给学生和讲师的通知是通过第三方Email系统进行的，则需要对图1-1和图1-2进行哪些修改？用100字以内文字加以说明。

参考答案：增加外部实体“第三方Email系统”，将原来的两条“通知”数据流合并为一条“通知”数据流，终点为“第三方Email系统”。

第2题 2015年下半年 下午试卷 案例 软件设计师

【说明】

某企业拟构建一个高效、低成本、符合企业实际发展需要的办公自动化系统。工程师小李主要承担该系统的公告管理和消息管理模块的研发工作。公告管理模块的主要功能包括添加、修改、删除和查看公告。消息管理模块的主要功能是消息群发。

小李根据前期调研和需求分析进行了概念模型设计，具体情况分述如下：

【需求分析结果】

(1)该企业设有研发部、财务部、销售部等多个部门，每个部门只有一名部门经理，有多名员工，每名员工只属于一个部门，部门信息包括：部门号、名称、部门经理和电话，其中部门号唯一确定部门关系的每一个元组。

(2)员工信息包括：员工号、姓名、岗位、电话和密码。员工号唯一确定员工关系的每一个元组；岗位主要有经理、部门经理、管理员等，不同岗位具有不同的权限。一名员工只对应一个岗位，但一个岗位可对应多名员工。

(3)消息信息包括：编号、内容、消息类型、接收人、接收时间、发送时间和发送人。其中（编号，接收人）唯一标识消息关系中的每一个元组。一条消息可以发送给多个接收人，一个接收人可以接收多条消息。

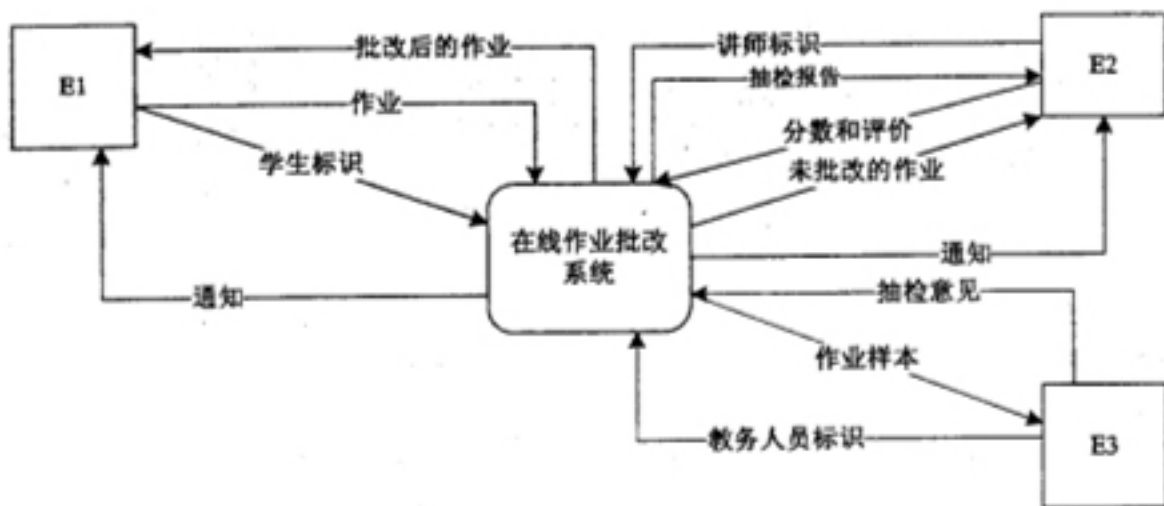


图 1-1 上下文数据流图

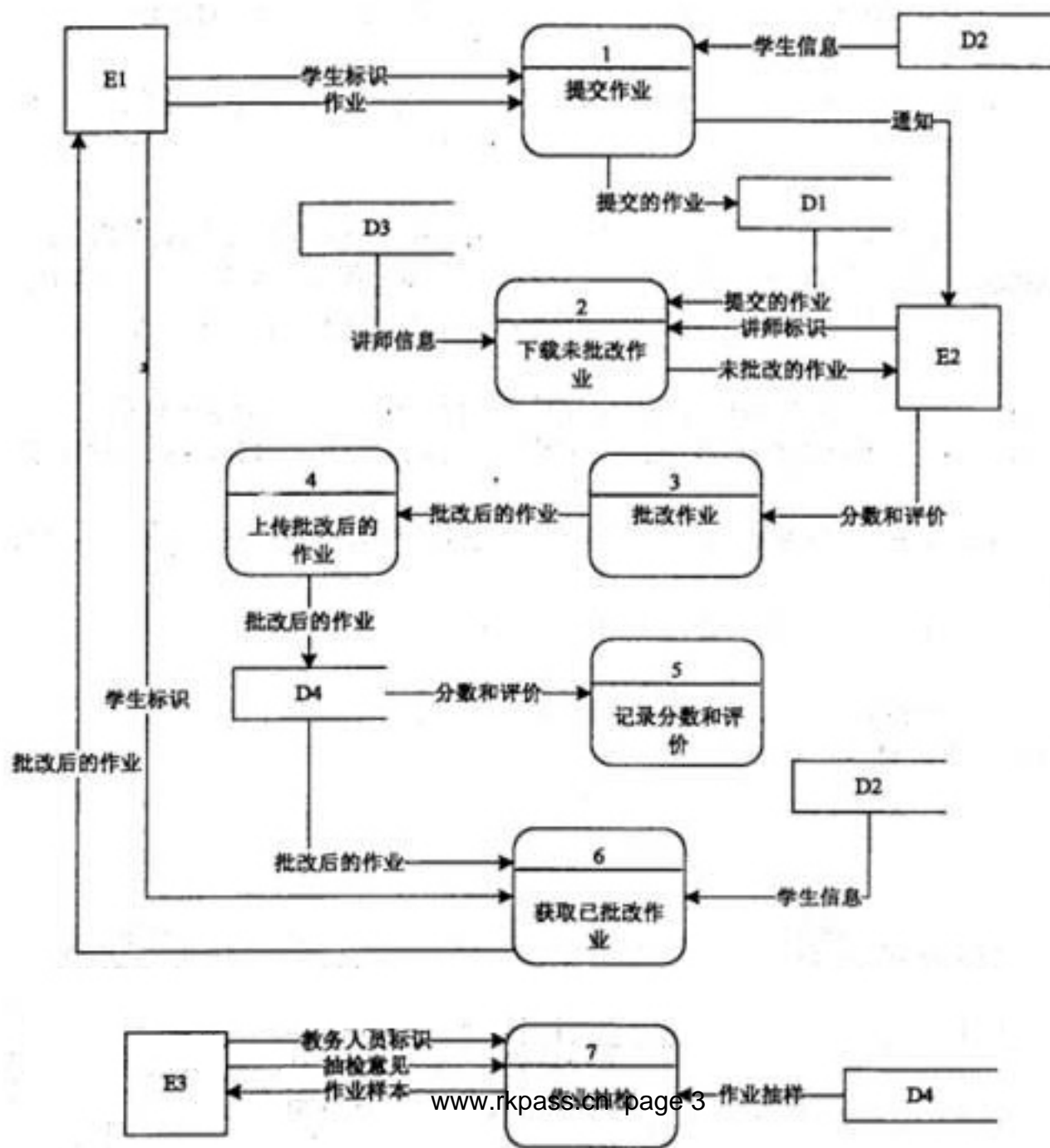


图 1-2 数据流图

每二个元组。一份公告对应一个发布部门，但一个部门可以发布多份公告；一份公告可以被多名员工阅读，一名员工可以阅读多份公告。

【概念模型设计】

根据需求分析阶段收集的信息，设计的实体联系图（不完整）如图2-1所示：

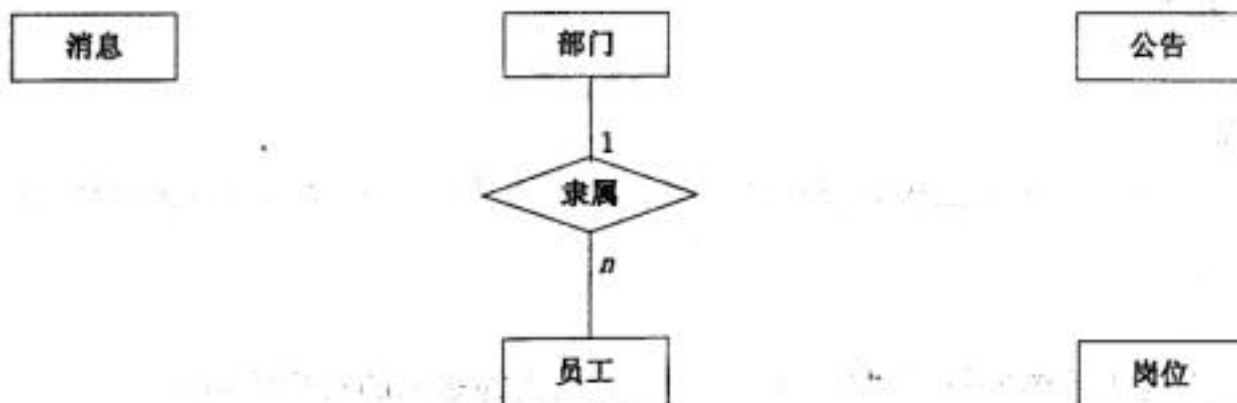


图 2-1 实体联系图

【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）：

部门 (a)，部门经理，电话)

员工 (员工号,姓名,岗位号,部门号，电话，密码)

岗位 (岗位号，名称，权限)

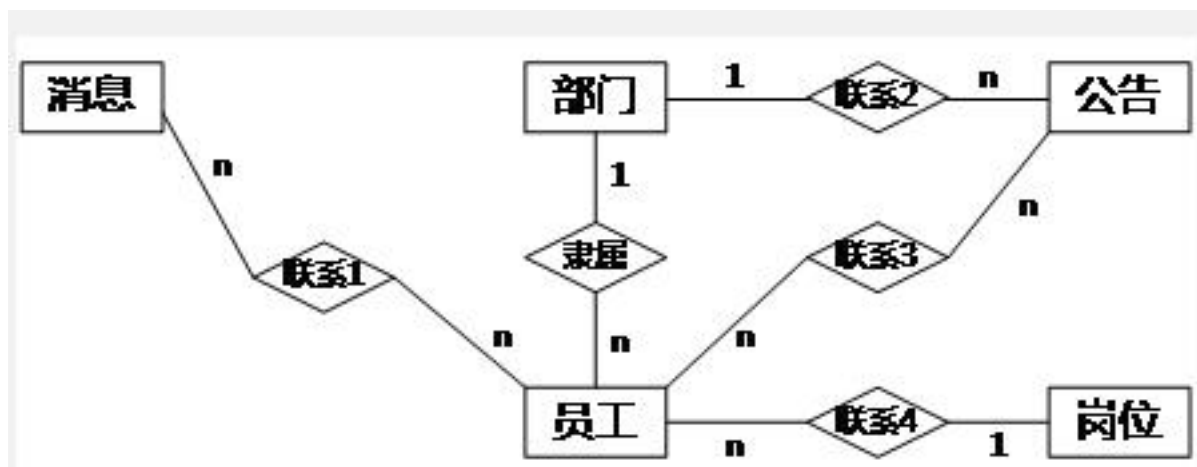
消息 (b)，消息类型，接收时间，发送时间，发送人)

公告 (c)，名称，内容，发布部门，发布时间)

阅读公告 (d)，阅读时间)

问题2.1 根据问题描述，补充四个联系，完善图2-1所示的实体联系图。联系名可用联系1、联系2、联系3和联系4代替，联系的类型分为 1:1、1:n和m:n (或1:1、1:*和*:*)。

参考答案：



问题2.2 (1)根据实体联系图，将关系模式中的空(a) ~ (d)补充完整。

(2)给出“消息”和“阅读公告”关系模式的主键与外键。

参考答案：(a) 部门号，名称

(b) 编号，内容，接收人，

(c) 编号，标题

(d) 员工号，消息编号

消息 主键：(编号，接收人) 外键：接收人，发送人

阅读公告 主键：(员工号，公告编号) 外键：员工号，公告编号

问题2.3 消息和公告关系中都有“编号”属性，请问它是属于命名冲突吗？用100字以内文字说明原因。

参考答案：不属于命名冲突。

命名冲突是在合并ER模型时提出的概念，合并ER模型时之所以产生冲突，是因为对于同样的对象，不同的局部ER模型有着不同的定义，在本题中，本就是不同对象的属性，所以不存在冲突的说法。

第3题 2015年下半年 下午试卷 案例 软件设计师

【说明】

某出版社拟开发一个在线销售各种学术出版物的网上商店(ACShop)，其主要的功能需求描述如下：

(1)ACShop在线销售的学术出版物包括论文、学术报告或讲座资料等。

(2)ACShop的客户分为两种：未注册客户和注册客户。

(3)未注册客户可以浏览或检索出版物，将出版物添加到购物车中。未注册客户进行注册操作之后，成为ACShop注册客户。

(4)注册客户登录之后，可将待购买的出版物添加到购物车中，并进行结账操作。结账操作的具体流程描述如下：

从预先填写的地址列表选择一个作为本次交易的收货地址。如果没有地址信息，则可以添加新地址。

选择付款方式。ACShop支持信用卡付款和银行转账两种方式。注册客户可以从预先填写的信用卡或银行账号中选择一个付款。若没有付款方式信息，则可以添加新付款方式。

确认提交购物车中待购买的出版物后，ACShop会自动生成与之相对应的订单。

(5)管理员负责维护在线销售的出版物目录，包括添加新出版物或者更新在售出版物信息等操作。

现采用面向对象方法分析并设计该网上商店ACShop，得到如图3-1所示的用例图和图3-2所示的类图。

问题3.1 据说明中的描述，给出图3-1中(1)~(4)所对应的用例名。

参考答案：(1) 添加出版物到购物车

(2) 结账

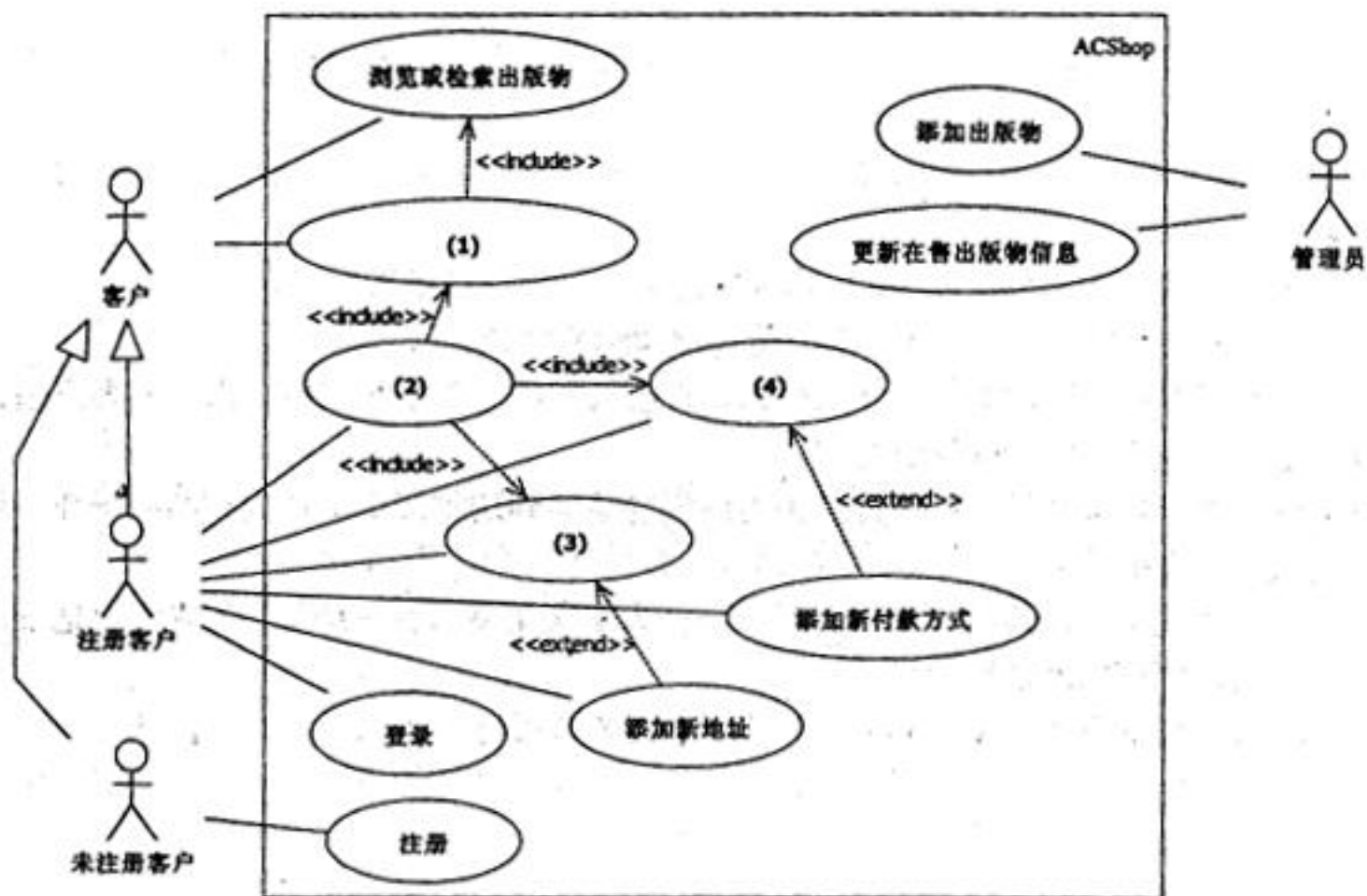
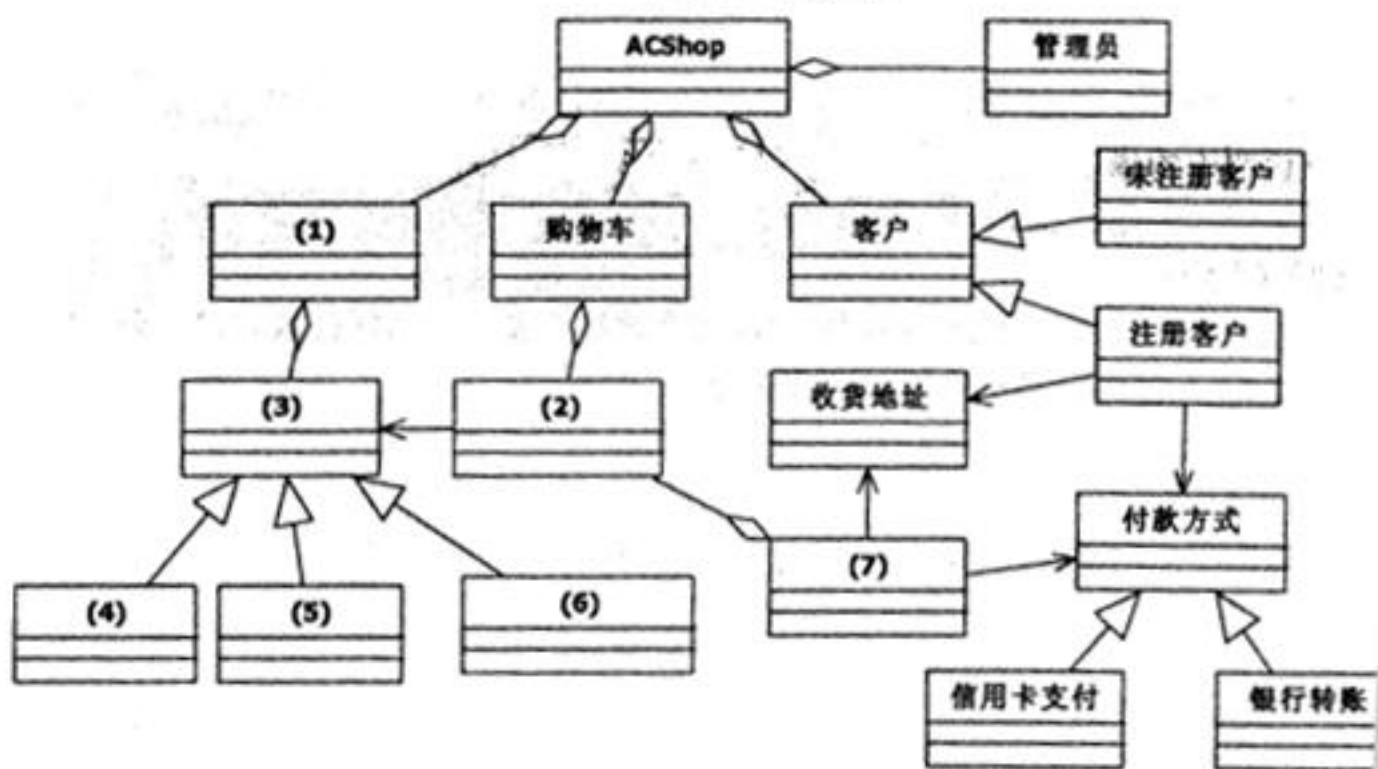


图 3-1 用例图



(3) 选择收货地址

(4) 选择付款方式

问题3.2 根据说明中的描述，分别说明用例“添加新地址”和“添加新付款方式”会在何种情况下由图3-1中的用例(3)和(4)扩展而来？

参考答案：当选择收货地址时，没有地址信息，则使用扩展用例“添加新地址”来完成新地址的添加。

当选择付款方式时，没有付款方式信息，则使用扩展用例“添加新付款方式”来完成新付款方式的添加。

问题3.3 根据说明中的描述，给出图3-2中(1)~(7)所对应的类名。

参考答案：(1)出版物目录

(2) 待购买的出版物

(3) 学术出版物

(4) - (6) 论文、学术报告、讲座资料

(7) 订单

第4题 2015年下半年 下午试卷 案例 软件设计师

【说明】

计算两个字符串x和y的最长公共子串（Longest Common Substring）。

假设字符串x和字符串y的长度分别为m和n，用数组c的元素c[i][j]记录x中前i个字符和y中前j个字符的最长公共子串的长度。

c[i][j]满足最优子结构，其递归定义为：

$$c[i][j] = \begin{cases} c[i-1][j-1] + 1 & \text{若 } i > 0 \text{ 且 } j > 0 \text{ 且 } x[i] = y[j] \\ 0 & \text{其它} \end{cases}$$

计算所有c[i][j](0 ≤ i ≤ m, 0 ≤ j ≤ n)的值，值最大的c[i][j]即为字符串x和y的最长公共子串的长度。根据该长度即i和j，确定一个最长公共子串。

问题4.1 根据以上说明和C代码，填充C代码中的空(1)~(4)。

参考答案：(1)x[i-1] == y[j-1]

(2)max=c[i][j]

(3)c[i][j]=0

(4)i=maxi-max

问题4.2 根据题干说明和以上C代码，算法采用了（5）设计策略。

【C代码】

(1) 常量和变量说明

x, y: 长度分别为m和n的字符串

c[i][j]: 记录x中前i个字符和y中前j个字符的最长公共子串的长度

max: x和y的最长公共子串的长度

maxi, maxj: 分别表示x和y的某个最长公共子串的最后一个字符在x和y中的位置 (序号)

(2) C程序

```
#include <stdio.h>
#include <string.h>
int c[50][50];
int maxi;
int maxj;
int lcs(char *x, int m, char *y, int n)    {
    int i, j;
    int max= 0;
    maxi= 0;
    maxj = 0;
    for ( i=0; i<=m ; i++)                c[i][0] = 0;
    for (i =1; i<= n; i++)                  c[i][0]=0;
    for (i =1; i<= m; i++)    {
        for (j=1; j<= n; j++)    {
            if (    (1)    )    {
                c[i][j] = c[i -1][j -1] + 1;
                if(max<c[i][j]) {
                    (2)    ;
                    maxi = i;
                    maxj =j;
                }
            }
        }
    }
    else    (3)    ;
    }
    return max;
}

void printLCS(int max, char *x) {
    int i= 0;
    if (max == 0)    return;
    for (    (4)    ; i < maxi; i++)
        printf("%c",x[i]);
}

void main() {
    char* x= "ABCADAB";
    char*y= "BDCABA";
    int max= 0;
    int m = strlen(x);
    int n = strlen(y);
    max=lcs(x,m,y,n);
    printLCS(max , x);
}
```

分析时间复杂度为 (6) (用O符号表示)

参考答案：(5)动态规划法
(6) $O(m*n)$

问题4.3 根据题干说明和以上C代码，输入字符串 $x = \text{"ABCADAB"}$ ， $y = \text{"BDCABA"}$ ，则输出为（7）。

参考答案：(7) AB

第5题 2015年下半年 下午试卷 案例 软件设计师

【说明】

某大型购物中心欲开发一套收银软件，要求其能够支持购物中心在不同时期推出的各种促销活动，如打折、返利（例如，满300返100）等等。现采用策略(Strategy)模式实现该要求，得到如图5-1所示的类图。

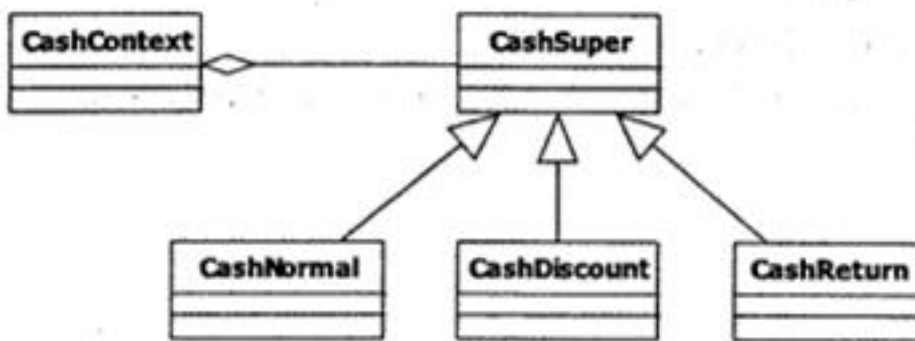


图 5-1 策略模式类图

问题5.1

参考答案：(1) `virtual double acceptCash(double money)=0`
(2) `cs = new CashNormal()`
(3) `cs = new CashReturn(300,100)`
(4) `cs = new CashDiscount(0.8)`
(5) `return cs->acceptCash(money)`

第6题 2015年下半年 下午试卷 案例 软件设计师

【C++代码】

```
#include <iostream>
using namespace std;
enum TYPE{NORMAL, CASH_DISCOUNT, CASH_RETURN};
class CashSuper{
public:
    (1);
};
class CashNormal : public CashSuper {    //正常收费子类
public:
    double acceptCash(double money) {    return money;    }
};
class CashDiscount : public CashSuper {
private:
    double moneyDiscount;    // 折扣率
public:
    CashDiscount(double discount) {    moneyDiscount= discount;    }
    double acceptCash(double money) {    return money * moneyDiscount;    }
};
class CashRetum : public CashSuper {    // 满额返利
private:
    double moneyCondition;    // 满额数额
    double moneyReturn;    // 返利数额
public:
    CashRetum(double motieyCondition, double moneyReturn) {
        this->moneyCondition=moneyCondition;
        this->moneyReturn=moneyReturn;
    }
    double acceptCash(double money) {
        double result = money;
        if(money>=moneyCondition)
            result=money-(int)(money/moneyCondition ) * moneyReturn;
        return result ;
    }
};
class CashContext {
private:
    CashSuper *cs;
public:
    CashContext(int type) {
        switch(type) {
            case NORMAL:    //正常收费
                (2) ;
                break;
            case CASH_RETURN:    //满300返100
                (3) ;
                break;
            case CASH_DISCOUNT:    //打八折
                (4) ;
                break;
        }
    }
    double GetResult(double money) {
        (5);
    }
};
//此外略去main()函数
```

【说明】

某大型购物中心欲开发一套收银软件，要求其能够支持购物中心在不同时期推出的各种促销活动，如打折、返利（例如，满300返100）等等。现采用策略(Strategy)模式实现该要求，得到如图6-1所示的类图。

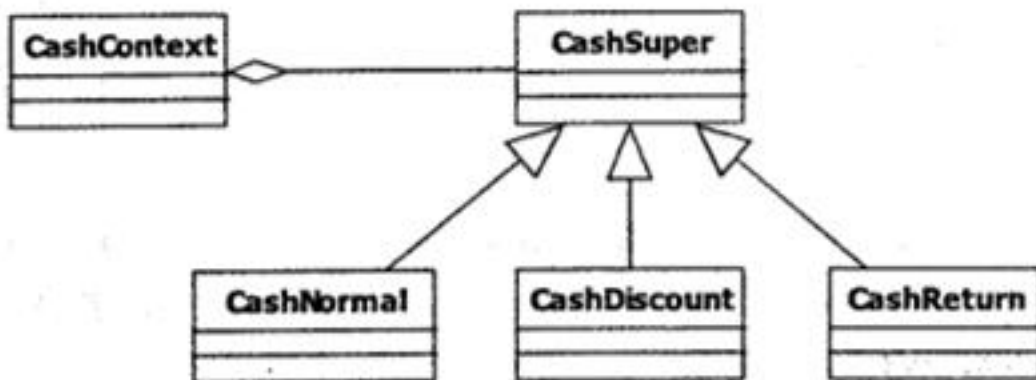


图 6-1 策略模式类图

问题6.1

参考答案：(1) doubleacceptCash(double money)

(2) cs=new CashNormal()

(3) cs=new CashReturn(300,100)

(4) cs=new CashDiscount(0.8)

(5) return cs.acceptCash(money)

【Java代码】

```
import java.util.*;
enum TYPE { NORMAL, CASH_DISCOUNT, CASH_RETURN};
interface CashSuper {
    public    (1)    ;
}
class CashNormal implements CashSuper{    // 正常收费子类
    public double accptCash(double money){
        return money;
    }
}
class CashDiscount implements CashSuper {
    private double moneyDiscount;                // 折扣率
    public CashDiscount(double moneyDiscount) {
        this.moneyDiscount = moneyDiscount;
    }
    public double acceptCash(double money) {
        return money* moneyDiscount;
    }
}
class CashReturn implements CashSuper {          // 满额返利
    private double moneyCondition;
    private double moneyReturn;
    public CashReturn(double moneyCondition, double moneyReturn) {
        this.moneyCondition =moneyCondition;    // 满额数额
        this.moneyReturn =moneyReturn;          // 返利数额
    }
    public double acceptCash(double money) {
        double result = money;
        if(money >= moneyCondition )
            result=money-Math.floor(money/moneyCondition ) * moneyReturn;
        return result;
    }
}
class CashContext_{
    private CashSuper cs;
    private TYPE    t;
    public CashContext(TYPE t)  {
        switch(t){
            case NORMAL:        // 正常收费
                (2)    ;
                break;
            case CASH_DISCOUNT:    // 满300返100
                (3)    ;
                break;
            case CASH_RETURN:      // 打8折
                (4)    ;
                break;
        }
    }
}
```