|  |
| --- |
|  |
| 软件设计师下午试卷 |
| 2009年~2021年 |
|  |
|  |
|  |

|  |
| --- |
| 计算机技术与软件专业技术资格（水平）考试 |



**全国计算机技术与软件专业技术资格（水平）考试**

# 2016年上半年 软件设计师 下午试卷

（考试时间14:00～16:30共150分钟）

|  |
| --- |
| **请按下述要求正确填写答题卡** |

1. 在答题纸的指定位置填写你所在的省、自治区、直辖市、计划单列市的名称。

2. 在答题纸的指定位置填写准考证号、出生年月日和姓名。

3. 答题纸上除填写上述内容外只能写解答。

4. 本试卷共6道题，试题一至试题四是必答题，试题五至试题六选答1道。每题15分，满分75分。

5. 解答时字迹务必清楚，字迹不清时，将不评分。

6. 仿照下面的例题，将解答写在答题纸的对应栏内。

例题

2016年上半年全国计算机技术与软件专业技术资格（水平）考试日期是（1）月（2）日。

因为正确的解答是“5月21日”，故在答题纸的对应栏内写上“5”和“21”（参看下表）。

|  |  |
| --- | --- |
| 例题 | 解答栏 |
| （1） | 5 |
| （2） | 21 |

**试题一（共15分）**

阅读下列说明和数据流图，回答问题1至问题4，将解答填入答题纸的对应栏内。

【说明】

某会议中心提供举办会议的场地设施和各种设备，供公司与各类组织机构租用。场地包括一个大型报告厅、一个小型报告厅以及诸多会议室。这些报告厅和会议室可提供的设备有投影仪、白板、视频播放/回放设备、计算机等。为了加强管理，该中心欲开发一会议预订系统，系统的主要功能如下。

（1）检查可用性。客户提交预订请求后，检查预订表，判定所申请的场地是否在申请日期内可用；如果不可用，返回不可用信息。

（2）临时预订。会议中心管理员收到客户预定请求的通知之后，提交确认。系统生成新临时预订存入预订表，并对新客户创建一条客户信息记录加以保存。根据客户记录给客户发送临时预订确认信息和支付定金要求。

（3）分配设施与设备。根据临时预订或变更预定的设备和设施需求，分配所需设备（均能满足用户要求）和设施，更新相应的表和预订表。

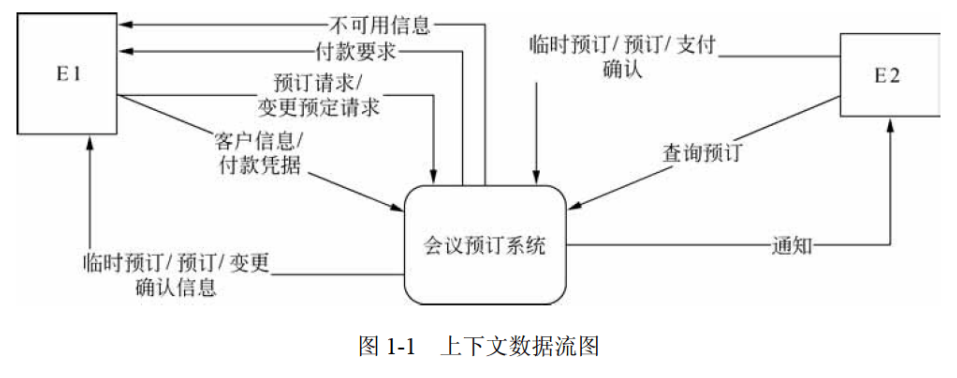
（4）确认预订。管理员收到客户支付定金的通知后，检查确认，更新预订表，根据客户记录给客户发送预订确认信息。

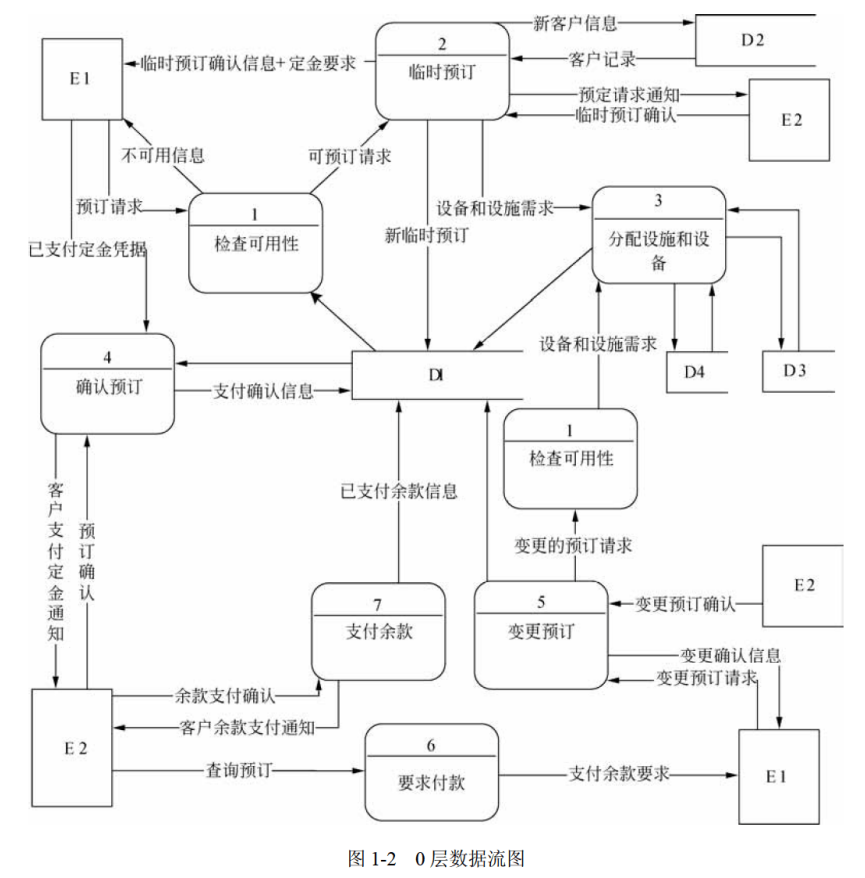
（5）变更预订。客户还可以在支付余款前提交变更预订请求，对变更的预订请求检查可用性，如果可用，分配设施和设备；如果不可用，返回不可用信息。管理员确认变更后，根据客户记录给客户发送确认信息。

（6）要求付款。管理员从预订表中查询距预订的会议时间两周内的预定，根据客户记录给满足条件的客户发送支付余款要求。

（7）支付余款。管理员收到客户余款支付的通知后，检查确认，更新预订表中的已支付余款信息。

现采用结构化方法对会议预定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（不完整）。





【问题1】（2分）

使用说明中的词语，给出图1-1中的实体E1〜E2的名称。

【问题2】（4分）

使用说明中的词语，给出图1-2中的数据存储D1〜D4的名称。

【问题3】（6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）

如果发送给客户的确认信息是通过Email系统向客户信息中的电子邮件地址进行发送的，那么需要对图1-1和1-2进行哪些修改？用150字以内文字加以说明。

**试题二（共15分）**

阅读下列说明，回答问题1至问题3，将解答填入答题纸的对应栏内。

【说明】

某销售公司当前的销售业务为商城实体店销售。现该公司拟开展网络销售业务，需要开发一个信息化管理系统。请根据公司现有业务及需求完成该系统的数据库设计。

【需求分析结果】

（1）记录公司所有员工的信息。员工信息包括工号、身份证号、姓名、性别、出生日期和电话，并只登记一部电话。

（2）记录所有商品的信息。商品信息包括商品名称、生产厂家、销售价格和商品介绍。系统内部用商品条码唯一区别每种商品。

（3）记录所有顾客的信息。顾客信息包括顾客姓名、身份证号、登录名、登录密码和电话号码。一位顾客只能提供一个电话号码。系统自动生成唯一的顾客编号。

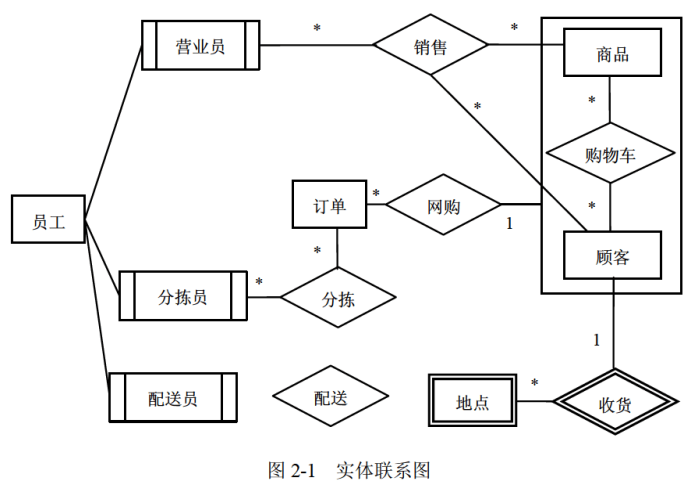
（4）顾客登录系统之后，在网上商城购买商品。顾客可将选购的商品置入虚拟的购物车内，购物车可长期存放顾客选购的所有商品。顾客可在购物车内选择商品、修改商品数量后生成网购订单。订单生成后，由顾客选择系统提供的备选第三方支付平台进行电子支付，支付成功后系统需要记录唯一的支付凭证编号，然后由商城根据订单进行线下配送。

（5）所有的配送商品均由仓库统一出库。为方便顾客，允许每位顾客在系统中提供多组收货地址、收货人及联系电话。一份订单所含的多个商品可能由多名分拣员根据商品所在仓库信息从仓库中进行分拣操作，分拣后的商品交由配送员根据配送单上的收货地址进行配送。

（6）新设计的系统要求记录实体店的每笔销售信息，包括营业员、顾客、所售商品及其数量。

【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。



【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）：

员工（工号，身份证号，姓名，性别，出生日期，电话）

商品（商品条码，商品名称，生产厂家，销售价格，商品介绍， （a） ）

顾客（顾客编号，姓名，身份证号，登录名，登录密码，电话）

收货地点（收货ID，顾客编号，收货地址，收货人，联系电话）

购物车（顾客编号，商品条码，商品数量）

订单（订单ID，顾客编号，商品条码，商品数量， （b） ）

分检（分拣ID，分拣员工号， （c） ，分拣时间）

配送（配送ID，分拣ID，配送员工号，收货ID，配送时间，签收时间，签收快照）

销售（销售ID，营业员工号，顾客编号，商品条码，商品数量）

【问题1】（4分）

补充图2-1中的“配送”联系所关联的对象及联系类型。

【问题2】（6分）

补充逻辑结构设计中的（a）、（b）和（c）三处空缺。

【问题3】（5分）

对于实体店销售，若要增加送货上门服务，由营业员在系统中下订单，与网购的订单进行后续的统一管理。请根据该需求，对图2-1进行补充，并修改订单关系模式。

**试题三（共15分）**

阅读下列说明和UML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】

某软件公司欲设计实现一个虚拟世界仿真系统。系统中的虚拟世界用于模拟现实世界中的不同环境（由用户设置并创建），用户通过操作仿真系统中的1〜2个机器人来探索虚拟世界。机器人维护着两个变量b1和b2，用来保存从虚拟世界中读取的字符。

该系统的主要功能描述如下：

（1）机器人探索虚拟世界（Run Robots）。用户使用编辑器（Editor）编写文件以设置想要模拟的环境，将文件导入系统（Load File）从而在仿真系统中建立虚拟世界（Setup World）。机器人在虚拟世界中的行为也在文件中进行定义，建立机器人的探索行为程序（Setup Program）。机器人在虚拟世界中探索时（Run Program），有2种运行模式：

①自动控制（Run）：事先编排好机器人的动作序列（指令（Instruction）），执行指令，使机器人可以连续动作。若干条指令构成机器人的指令集（Instruction Set）。

②单步控制（Step）：自动控制方式的一种特殊形式，只执行指定指令中的一个动作。

（2）手动控制机器人（Manipulate Robots）。选定1个机器人后（Select Robot），可以采用手动方式控制它。手动控制有4种方式：

①Move：机器人朝着正前方移动一个交叉点。

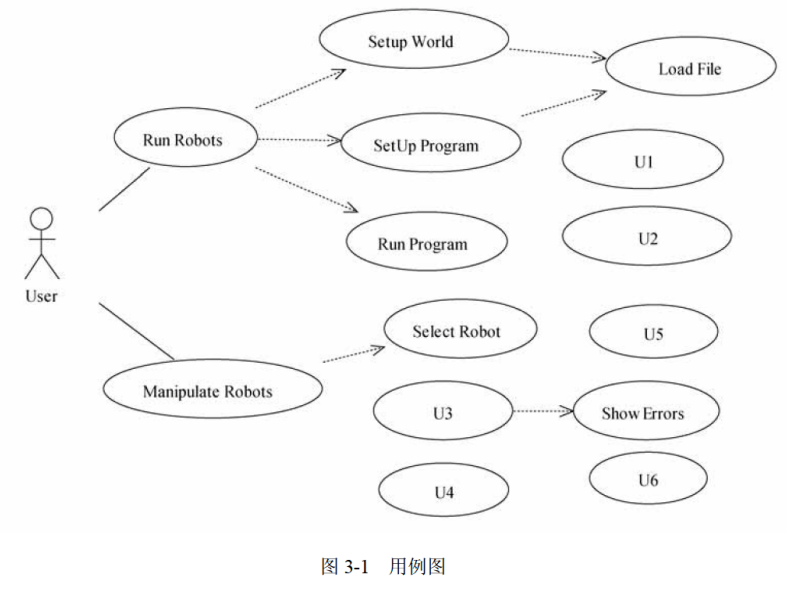
②Left：机器人原地沿逆时针方向旋转90度。

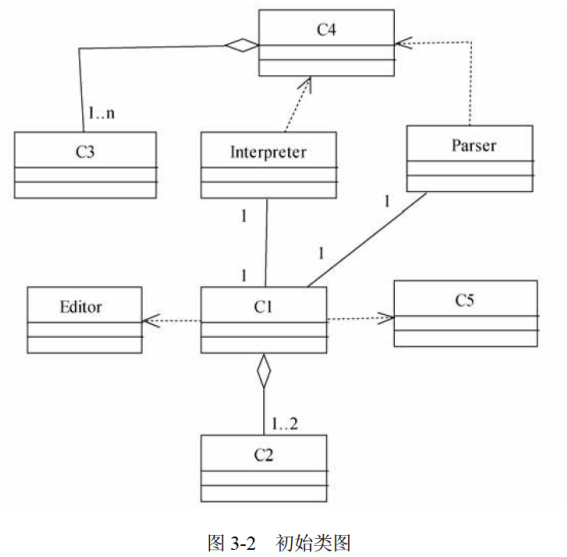
③Read：机器人读取其所在位置的字符，并将这个字符的值赋给b1；如果这个位置上没有字符，则不改变b1的当前值。

④Write：将b1中的字符写入机器人当前所在的位置，如果这个位置上已经有字符，该字符的值将会被b1的值替代。如果这时b1没有值，即在执行Write动作之前没有执行过任何Read动作，那么需要提示用户相应的错误信息（Show Errors）。

手动控制与单步控制的区别在于，单步控制时执行的是指令中的动作，只有一种控制方式，即执行下一个动作；而手动控制时有4种动作。

现采用面向对象方法设计并实现该仿真系统，得到如图3-1所示的用例图和图3-2所示的初始类图。图3-2中的类“Interpreter”和“Parser”用于解析描述虚拟世界的文件以及机器人行为文件中的指令集。





【问题1】（6分）

根据说明中的描述，给出图3-1中U1〜U6所对应的用例名。

【问题2】（4分）

图3-1中用例U1〜U6分别与哪个（哪些）用例之间有关系，是何种关系？

【问题3】（5分）

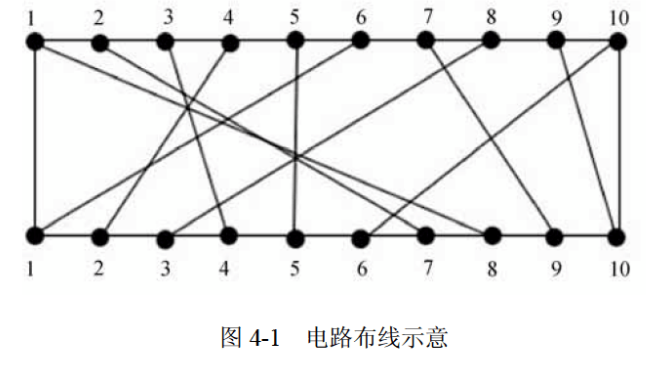
根据说明中的描述，给出图3-2中C1〜C5所对应的类名。

**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，将解答填入答题纸的对应栏内。

【说明】

在一块电路板的上下两端分别有个接线柱。根据电路设计，用(i,(i))表示将上端接线柱i与下端接线柱(i)相连，称其为该电路板上的第i条连线。如图4-1所示的(i)排列为{8, 7, 4, 2, 5, 1, 9, 3, 10, 6}。对于任何1≤i<j≤，第i条连线和第j条连线相交的充要条件是(i)>(j)。

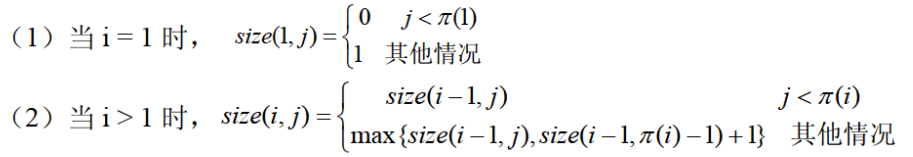


在制作电路板时，要求将这条连线分布到若干绝缘层上，在同一层上的连线不相交。现在要确定将哪些连线安排在一层上，使得该层上有尽可能多的连线，即确定连线集Nets={(i,(i))，1≤i≤n}的最大不相交子集。

【分析问题】

记N(i,j)={t|(t,(t))∈Nets,t≤i,(t)≤j}。N(i,j)的最大不相交子集为MNS(i,j)，size(i,j)=|MNS(i,j)|。

经分析，该问题具有最优子结构性质。对规模为n的电路布线问题，可以构造如下递归式：



【C代码】

下面是算法的C语言实现。

（1）变量说明

size[i][j]：上下端分别有i个和j个接线柱的电路板的第一层最大不相交连接数

pi[i]：π(i)，下标从1开始

（2）C程序

#include <stdlib.h>

#include <stdio.h>

#define N 10 /\* 问题规模 \*/

int m = 0; /\* 记录最大连接集合中的接线柱 \*/

/\* 求最大不相交连接数 \*/

void maxNum(int pi[], int size[N + 1][N + 1], int n) {

int i, j;

for (j = 0;j < pi[1]; j ++ ) size[1][j] = 0; /\* 当j < π(1)时 \*/

for (j = pi[1]; j <= n; j ++ ) (1) ; /\* 当j >= π(1)时 \*/

for (i = 2; i < n; i ++) {

for (j = 0; j < pi[i]; j ++ ) (2) ; /\* 当j < pi[i]时 \*/

for (j = pi[i]; j <= n; j ++ ) { /\* 当j >= c[i]时,考虑两种情况 \*/

size[i][j] = size[i-1][j] >= size[i - 1][pi[i] - 1] + 1

? size[i-1][j] : size[i - 1][pi[i] - 1] + 1;

}

}

/\* 最大连接数 \*/

size[n][n] = size[n - 1][n] >= size[n - 1][pi[n] - 1] + 1

? size[n - 1][n] : size[n-1][pi[n]-1]+1;

}

/\* 构造最大不相交连接集合，net[i]表示最大不相交子集中第i条连线的上端接线柱的序号 \*/

void constructSet(int pi[], int size[N + 1][N + 1], int n, int net[n]) {

int i, j = n;

m = 0;

for (i = n; i > 1; i -- ) { /\* 从后往前 \*/

/\* (i, pi[i])是最大不相交子集的一条连线 \*/

if (size[i][j] != size[i - 1][j]) {

(3) ; /\* 将i记录到数组net中，连接线数自增1 \*/

j = pi[i] - 1; /\* 更新扩展连线柱区间 \*/

}

}

if (j >= pi[1]) net[m ++ ] = 1; /\* 当i = 1时 \*/

}

【问题1】（6分）

根据以上说明和C代码，填充C代码中的空（1）〜（3）。

【问题2】（6分）

据题干说明和以上C代码，算法采用了 （4） 算法设计策略。

函数maxNum和constructSet的时间复杂度分别为 （5） 和 （6） （用O表示）。

【问题3】（3分）

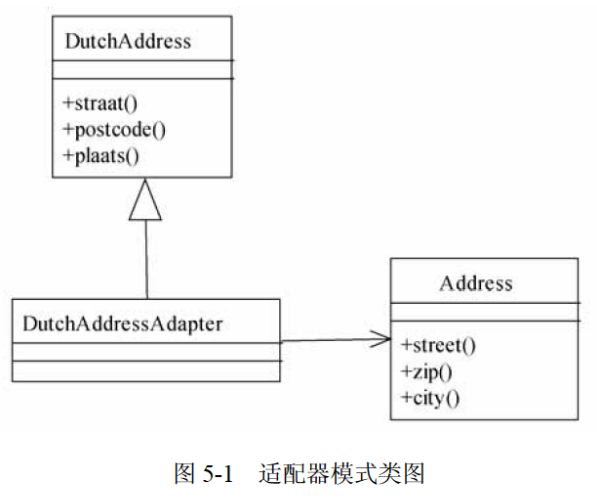
若连接排列为{8, 7, 4, 2, 5, 1, 9, 3, 10, 6}，即如图4-1所示，则最大不相交连接数为 （7） ，包含的连线为 （8） （用(i,(i))的形式给出）。

**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】

某软件系统中，已设计并实现了用于显示地址信息的类Address（如图5-1所示），现要求提供基于Dutch语言的地址信息显示接口。为了实现该要求并考虑到以后可能还会出现新的语言的接口，决定采用适配器（Adapter）模式实现该要求，得到如图5-1所示的类图。



【C++代码】

#include <iostream>

using namespace std;

class Address {

public :

void street() { /\* 实现代码省略 \*/ }

void zip() { /\* 实现代码省略 \*/ }

void city() { /\* 实现代码省略 \*/ }

// 其他成员省略

};

class DutchAddress {

public :

virtual void straat() = 0;

virtual void postcode() = 0;

virtual void plaats() = 0;

// 其他成员省略

};

class DutchAddressAdapter : public DutchAddress {

private :

(1) ;

public :

DutchAddressAdapter(Address \*addr) {

address = addr;

}

void straat() {

(2) ;

}

void postcode() {

(3) ;

}

void plaats() {

(4) ;

}

// 其他成员省略

};

void test Dutch(DutchAddress \*addr) {

addr->straat();

addr->postcode();

addr->plaats();

}

int main() {

Address \*addr = new Address();

(5) ;

cout << "\n The DutchAddress\n" << endl;

testDutch(addrAdapter);

return 0;

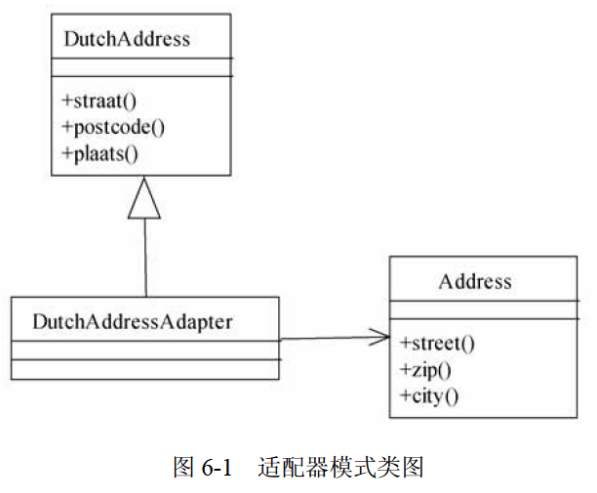
}

**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】

某软件系统中，已设计并实现了用于显示地址信息的类Address（如图6-1所示），现要求提供基于Dutch语言的地址信息显示接口。为了实现该要求并考虑到以后可能还会出现新的语言的接口，决定采用适配器（Adapter）模式实现该要求，得到如图6-1所示的类图。



【Java代码】

import java.util.\*;

class Address {

public void street() { // 实现代码省略 }

public void zip() { // 实现代码省略 }

public void city() { // 实现代码省略 }

// 其他成员省略

}

class DutchAddress {

public void straat() { // 实现代码省略 }

public void postcode() { // 实现代码省略 }

public void plaats() { // 实现代码省略 }

// 其他成员省略

}

class DutchAddressAdapter extends DutchAddress {

private (1) ;

public DutchAddressAdapter (Address addr) {

address = addr;

}

public void straat() {

(2) ;

}

public void postcode() {

(3) ;

}

public void plaats() {

(4) ;

}

// 其他成员省略

}

class Test {

public static void main(String[] args) {

Address addr = new Address();

(5) ;

System.out.println("\n The DutchAddress\n");

testDutch(addrAdapter);

}

static void testDutch(DutchAddress addr) {

addr.straat();

addr.postcode();

addr.plaats();

}

}

**全国计算机技术与软件专业技术资格（水平）考试**

# 2016年下半年 软件设计师 下午试卷

（考试时间14:00～16:30共150分钟）

|  |
| --- |
| **请按下述要求正确填写答题卡** |

1. 在答题纸的指定位置填写你所在的省、自治区、直辖市、计划单列市的名称。

2. 在答题纸的指定位置填写准考证号、出生年月日和姓名。

3. 答题纸上除填写上述内容外只能写解答。

4. 本试卷共6道题，试题一至试题四是必答题，试题五至试题六选答1道。每题15分，满分75分。

5. 解答时字迹务必清楚，字迹不清时，将不评分。

6. 仿照下面的例题，将解答写在答题纸的对应栏内。

例题

2016 年下半年全国计算机技术与软件专业技术资格（水平）考试日期是（1）月（2）日。

因为正确的解答是“11月12日”，故在答题纸的对应栏内写上“11”和“12”（参看下表）。

|  |  |
| --- | --- |
| 例题 | 解答栏 |
| （1） | 11 |
| （2） | 12 |

**试题一（共15分）**

阅读下列说明和数据流图，回答问题1至问题4，将解答填入答题纸的对应栏内。

【说明】

某证券交易所为了方便提供证券交易服务，欲开发一证券交易平台，该平台的主要功能如下：

（1）开户。根据客户服务助理提交的开户信息，进行开户，并将客户信息存入客户记录中，账户信息（余额等）存入账户记录中。

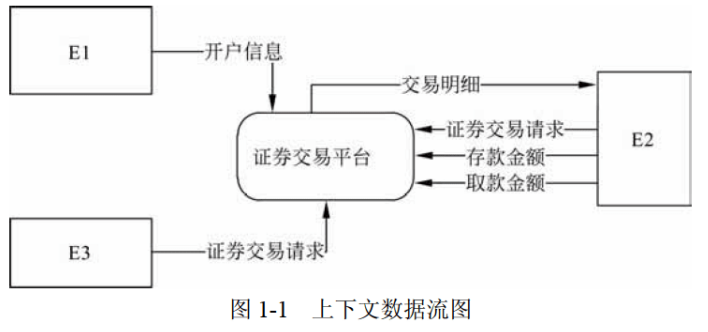
（2）存款。客户可以向其账户中存款，根据存款金额修改账户余额。

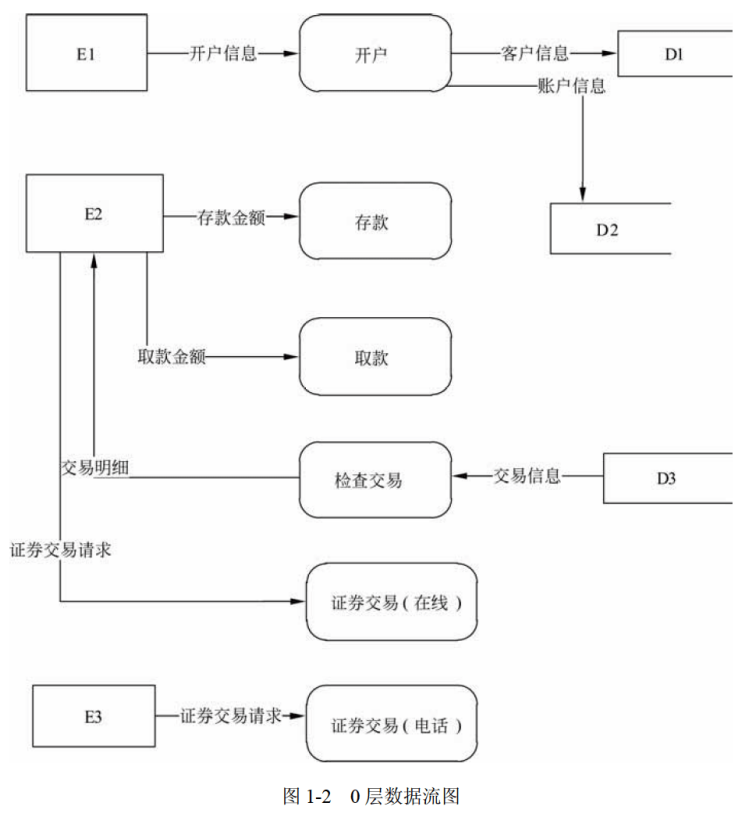
（3）取款。客户可以从其账户中取款，根据取款金额修改账户余额。

（4）证券交易。客户和经纪人均可以进行证券交易（客户通过在线方式，经纪人通过电话），将交易信息存入交易记录中。

（5）检查交易：平台从交易记录中读取交易信息，将交易明细返回给客户。

现采用结构化方法对该证券交易平台进行分析与设计，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】（3分）

使用说明中的词语，给出图1-1中的实体E1〜E3的名称。

【问题2】（3分）

使用说明中的词语，给出图1-2中的数据存储D1〜D3的名称。

【问题3】（4分）

根据说明和图中的术语，补充图1-2中缺失的数据流及其起点和终点。

【问题4】（5分）

实际的证券交易通常是在证券交易中心完成的，因此，该平台的“证券交易”功能需将交易信息传递给证券交易中心。针对这个功能需求，需要对图1-1和图1-2进行哪些修改，请用200字以内的文字加以说明。

**试题二（共15分）**

阅读下列说明，回答问题1至问题3，将解答填入答题纸的对应栏内。

【说明】

某宾馆为了有效地管理客房资源，满足不同客户需求，拟构建一套宾馆信息管理系统，以方便宾馆管理及客房预订等业务活动。

【需求分析结果】

该系统的部分功能及初步需求分析的结果如下：

（1）宾馆有多个部门，部门信息包括部门号、部门名称、电话、经理。每个部门可以有多名员工，每名员工只属于一个部门；每个部门只有一名经理，负责管理本部门。

（2）员工信息包括员工号、姓名、岗位、电话、工资，其中，员工号唯一标识员工关系中的一个元组，岗位有经理、业务员。

（3）客房信息包括客房号（如1301、1302等）、客房类型、收费标准、入住状态（已入住／未入住），其中客房号唯一标识客房关系中的一个元组，不同客房类型具有不同的收费标准。

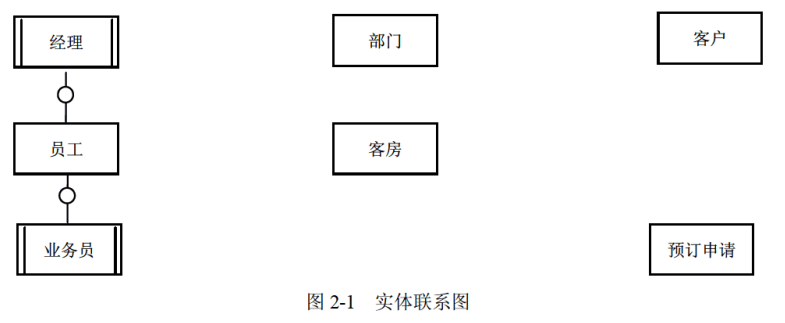
（4）客户信息包括客户号、单位名称、联系人、联系电话、联系地址，其中客户号唯一标识客户关系中的一个元组。

（5）客户预订客房时，需要填写预订申请。预订申请信息包括申请号、客户号、入住时间、入住天数、客房类型、客房数量，其中，一个申请号唯一标识预订申请中的一个元组；一位客户可以有多个预订申请，但一个预订申请对应唯一的一位客户。

（6）当客户入住时，业务员根据客户的预订申请负责安排入住客房事宜。安排信息包括客房号、姓名、性别、身份证号、入住时间、天数、电话，其中客房号、身份证号和入住时间唯一标识一次安排。一名业务员可以安排多个预订申请，一个预订申请只由一名业务员安排，而且可安排多间同类型的客房。

【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。



【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（部门号，部门名称，经理，电话）

员工（员工号， （a） ，姓名，岗位，电话，工资）

客户（ （b） ，联系人，联系电话，联系地址）

客房（客房号，客房类型，收费标准，入住状态）

预订申请（ （c） ，入住时间，天数，客房类型，客房数量）

安排（申请号，客房号，姓名，性别， （d） ，天数，电话，业务员）

【问题1】（4分）

根据问题描述，补充四个联系，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和联系4代替，联系的类型为1:1、1:n和m:n（或1:1，和1:\*和\*:\*）。

【问题2】（8分）

（1）根据题意，将关系模式中的空（a）〜（d）补充完整，并填入答题纸对应的位置上。

（2）给出“预订申请”和“安排”关系模式的主键和外键。

【问题3】（3分）

【关系模式设计】中的“客房”关系模式是否存在规范性问题，请用100字以内文字解释你的观点（若存在问题，应说明如何修改“客房”关系模式）。

**试题三（共15分）**

阅读下列说明和UML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】

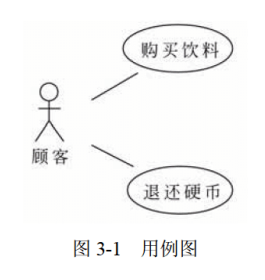
某种出售罐装饮料的自动售货机（Vending Machine）的工作过程描述如下：

（1）顾客选择所需购买的饮料及数量。

（2）顾客从投币口向自动售货机中投入硬币（该自动售货机只接收硬币）。硬币器收集投入的硬币并计算其对应的价值。如果所投入的硬币足够购买所需数量的这种饮料且饮料数量足够，则推出饮料，计算找零，顾客取走饮料和找回的硬币；如果投入的硬币不够或者所选购的饮料数量不足，则提示用户继续投入硬币或重新选择饮料及数量。

（3）一次购买结束之后，将硬币器中的硬币移走（清空硬币器），等待下一次交易。

自动售货机还设有一个退币按钮，用于退还顾客所投入的硬币。已经成功购买饮料的钱是不会被退回的。



现采用面向对象方法分析和设计该自动售货机的软件系统，得到如图3-1所示的用例图，其中，用例“购买饮料”的用例规约描述如下。

参与者：顾客。

主要事件流：

1．顾客选择需要购买的饮料和数量，投入硬币；

2．自动售货机检查顾客是否投入足够的硬币；

3．自动售货机检查饮料储存仓中所选购的饮料是否足够；

4．自动售货机推出饮料；

5．自动售货机返回找零。

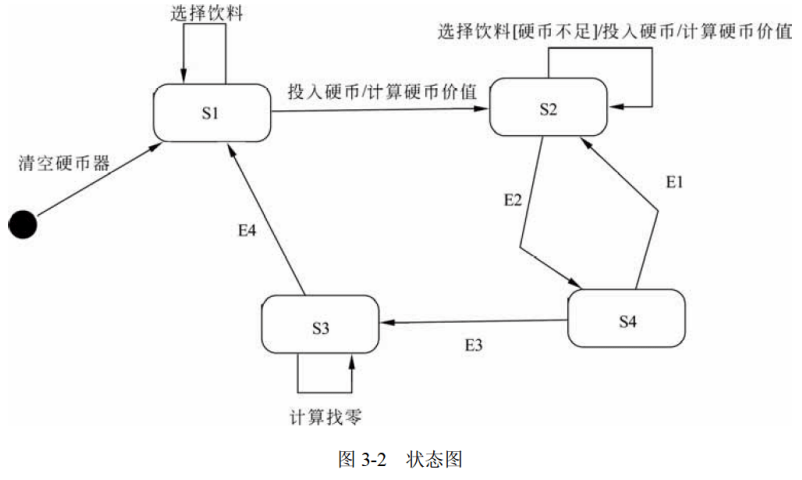
备选事件流：

2a．若投入的硬币不足，则给出提示并退回到1；

3a．若所选购的饮料数量不足，则给出提示并退回到1。

根据用例“购买饮料”得到自动售货机的4个状态：“空闲”状态、“准备服务”状态、“可购买”状态以及“饮料出售”状态，对应的状态图如图3-2所示。

所设计的类图如图3-3所示。





【问题1】（6分）

根据说明中的描述，使用说明中的术语，给出图3-2中的S1〜S4所对应的状态名。

【问题2】（4分）

根据说明中的描述，使用说明中的术语，给出图3-2中的E1〜E4所对应的事件名称。

【问题3】（5分）

根据说明中的描述，使用说明中的术语，给出图3-3中C1〜C5所对应的类名。

**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，将解答填入答题纸的对应栏内。

【说明】

模式匹配是指给定主串t和子串s，在主串t中寻找子串s的过程，其中s称为模式。如果匹配成功，返回s在t中的位置，否则返回-1。

KMP算法用next数组对匹配过程进行了优化。KMP算法的伪代码描述如下：

1．在串t和串s中，分别设比较的起始下标i=j=0。

2．如果串t和串s都还有字符，则循环执行下列操作：

（1）如果j=-1或者t[i]=s[j]，则将i和j分别加1，继续比较t和s的下一个字符；

（2）否则，将j向右滑动到next[j]的位置，即j=next[j]。

3．如果s中所有字符均已比较完毕，则返回匹配的起始位置（从1开始）；否则返回-1。其中，next数组根据子串s求解。求解next数组的代码已由get\_next函数给出。

【C代码】

（1）常量和变量说明

t，s：长度为lt和ls的字符串

next：next数组，长度为ls

（2）C程序

#include <stdio.h>

#include <stdlib.h>

#include <string.h>

/\* 求next[]的值 \*/

void get\_next(int \*next, char \*s, int ls) {

int i = 0, j = -1;

next[0] = -1; /\* 初始化next[0] \*/

while(i < ls) { /\* 还有字符 \*/

if (j == -1 || s[i] == s[j]) { /\* 匹配 \*/

j ++ ;

i ++ ;

if (s[i] == s[j])

next[i] = next[j];

else

next[i] = j;

}

else

j = next[j];

}

}

int kmp(int \*next, char \*t, char \*s, int lt, int ls) {

int i = 0, j = 0;

while (i < lt && (1) ) {

if (j == -1 || (2) ) {

i ++ ;

j ++ ;

} else

(3) ;

}

if (j >= ls)

return (4) ;

else

return -1;

}

【问题1】（8分）

根据题干说明，填充C代码中的空（1）〜（4）

【问题2】（2分）

根据题干说明和C代码，分析出KMP算法的时间复杂度为 （5） （主串和子串的长度分别为lt和ls，用O符号表示）。

【问题3】（5分）

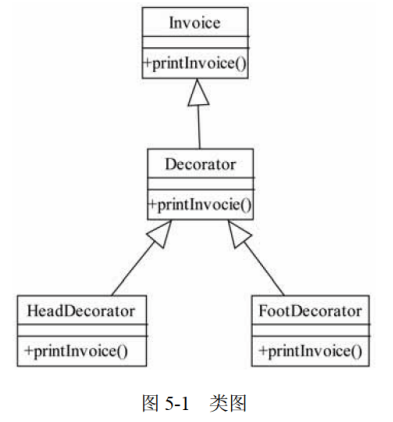
根据C代码，字符串“BBABBCAC”的next数组元素值为 （6） （直接写素值，之间用逗号隔开）。若主串为“AABBCBBABBCACCD”，子串为“BBABBCAC”，则函数kmp的返回值是 （7） 。

**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】

某发票（lnvoice）由抬头（Head）部分、正文部分和脚注（Foot）部分构成。现采用装饰（Decorator）模式实现打印发票的功能，得到如图5-1所示的类图。



【C++代码】

#include <iostream>

using namespace std;

class Invoice {

public :

(1) {

cout << "This is the content of the invoice!" << endl;

}

};

class Decorator : public Invoice {

Invoice \*ticket;

public :

Decorator(Invoice \*t) { ticket = t; }

void printInvoice() {

if (ticket != NULL)

(2) ;

}

};

class HeadDecorator : public Decorator {

public :

HeadDecorator(Invoice \*t) : Decorator(t) { }

void printInvoice() {

cout << "This is the header of the invoice!" << endl;

(3) ;

}

};

class FootDecorator : public Decorator {

public :

FootDecorator(Invoice \*t) : Decorator(t) { }

void printInvoice() {

cout << "This is the footnote of the invoice!" << endl;

(4) ;

}

};

int main(void) {

Invoice t;

FootDecorator f(&t);

HeadDecorator h(&f);

h.printInvoice();

cout << "--------------------" << endl;

HeadDecorator b( (5) );

b.printInvoice();

return 0;

}

程序的输出结果为：

This is the header of the invoice!

This is the content of the invoice!

This is the footnote of the invoice!

--------------------

This is the header of the invoice!

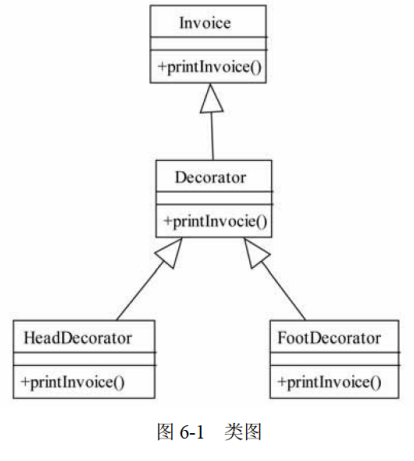
This is the footnote of the invoice!

**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】

某发票（lnvoice）由抬头（Head）部分、正文部分和脚注（Foot）部分构成。现采用装饰（Decorator）模式实现打印发票的功能，得到如图6-1所示的类图。



【java代码】

class invoice {

public void printInvoice() {

System.out.println("This is the content of the invoice!");

}

}

class Decorator extends Invoice {

protected Invoice ticket;

public Decorator(Invoice t) {

ticket = t;

}

public void printInvoice() {

if (ticket != null)

(1) ;

}

}

class HeadDecorator extends Decorator {

public HeadDecorator(Invoice t) {

super(t);

}

public void printInvoice () {

System.out.println("This is the header of the invoice!");

(2) ;

}

}

class FootDecorator extends Decorator {

public FootDecorator(Invoice t) {

super(t);

}

public void printInvoice() {

(3) ;

System.out.println("This is the footnote of the invoice!");

}

}

class Test {

public static void main(String[] args) {

Invoice t = new Invoice();

Invoice ticket;

ticket = (4) ;

ticket.printInvoice();

System.out.println("--------------------");

ticket = (5) ;

ticket.printInvoice();

}

}

程序的输出结果为：

This is the header of the invoice!

This is the content of the invoice!

This is the footnote of the invoice!

--------------------

This is the header of the invoice!

This is the footnote of the invoice!

**全国计算机技术与软件专业技术资格（水平）考试**

# 2017年上半年 软件设计师 下午试卷

（考试时间 14 : 00～16 : 30 共 150 分钟）

|  |
| --- |
| **请按下述要求正确填写答题卡** |

1. 在答题纸的指定位置填写你所在的省、自治区、直辖市、计划单列市的名称。

2. 在答题纸的指定位置填写准考证号、出生年月日和姓名。

3. 答题纸上除填写上述内容外只能写解答。

4. 本试卷共6道题，试题一至试题四是必答题，试题五至试题六选答1道。每题15分，满分75分。

5. 解答时字迹务必清楚，字迹不清时，将不评分。

6. 仿照下面的例题，将解答写在答题纸的对应栏内。

例题

2017年上半年全国计算机技术与软件专业技术资格（水平）考试日期是（1）月（2）日。

因为正确的解答是“5月20日”，故在答题纸的对应栏内写上“5”和“20”（参看下表）。

|  |  |
| --- | --- |
| 例题 | 解答栏 |
| （1） | 5 |
| （2） | 20 |

**试题一（共15分）**

阅读下列说明和数据流图，回答问题1至问题4，将解答填入答题纸的对应栏内。

【说明】

某医疗器械公司作为复杂医疗产品的集成商，必须保持高质量部件的及时供应。为了实现这一目标，该公司欲开发一采购系统。系统的主要功能如下：

1．检查库存水平。采购部门每天检查部件库存量，当特定部件的库存量降至其订货点时，返回低存量部件及库存量。

2．下达采购订单。采购部门针对低存量部件及库存量提交采购请求，向其供应商（通过供应商文件访问供应商数据）下达采购订单，并存储于采购订单文件中。

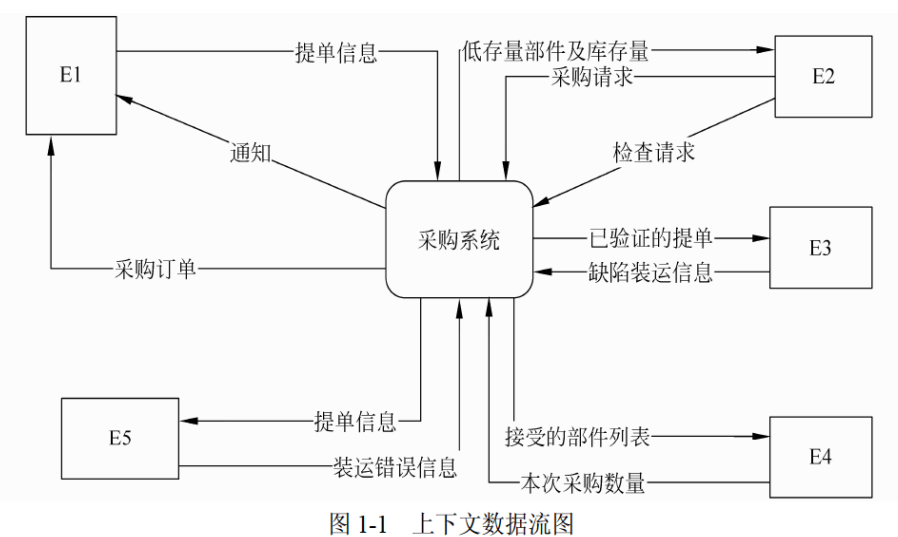
3．交运部件。当供应商提交提单并交运部件时，运输和接收（S/R）部门通过执行以下三步过程接收货物：

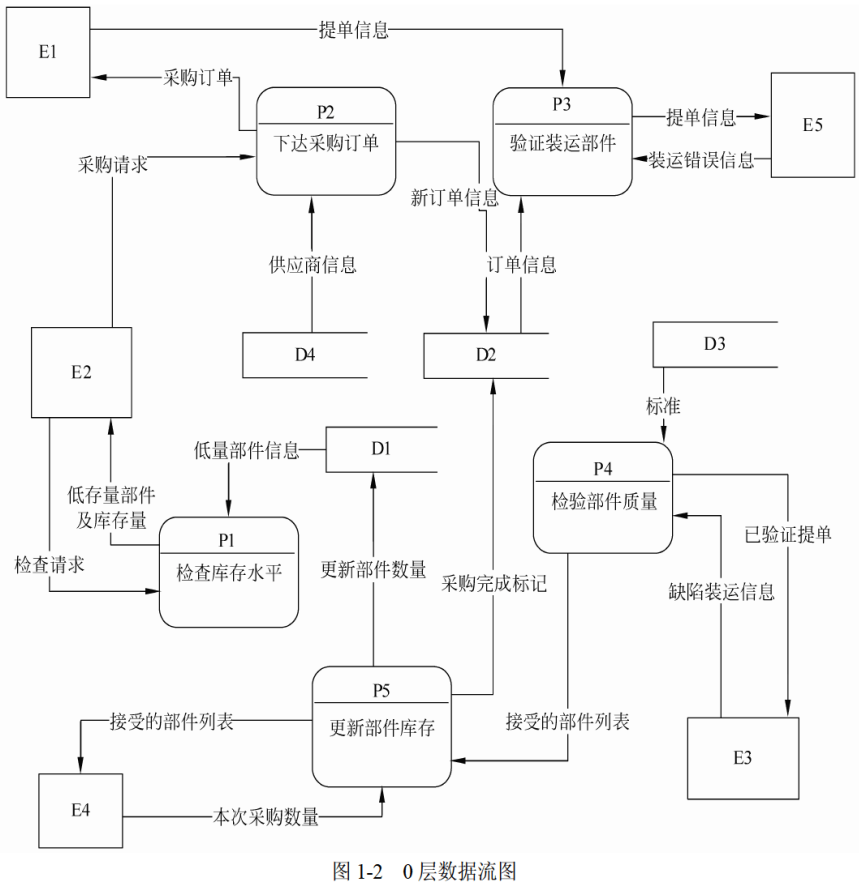
（1）验证装运部件。通过访问采购订单并将其与提单进行比较来验证装运的部件，并将提单信息发给S/R职员。如果收货部件项目出现在采购订单和提单上，则已验证的提单和收货部件项目将被送去检验。否则，将S/R职员提交的装运错误信息生成装运错误通知发送给供应商。

（2）检验部件质量。通过访问质量标准来检查装运部件的质量，并将已验证的提单发给检验员。如果部件满足所有质量标准，则将其添加到接受的部件列表用于更新部件库存。如果部件未通过检查，则将检验员创建的缺陷装运信息生成缺陷装运通知发送给供应商。

（3）更新部件库存。库管员根据收到的接受的部件列表添加本次采购数量，与原有库存量累加来更新库存部件中的库存量。标记订单采购完成。

现采用结构化方法对该采购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】（5分）

使用说明中的词语，给出图1-1中的实体E1〜E5的名称。

【问题2】（4分）

使用说明中的词语，给出图1-2中的数据存储D1〜D4的名称。

【问题3】（4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】（2分）

用200字以内文字，说明建模图1-1和图1-2时如何保持数据流图平衡。

**试题二（共15分）**

阅读下列说明，回答问题1至问题3，将解答填入答题纸的对应栏内。

【说明】

某房屋租赁公司拟开发一个管理系统用于管理其持有的房屋、租客及员工信息。请根据下述需求描述完成系统的数据库设计。

【需求分析结果】

1．公司拥有多幢公寓楼，每幢公寓楼有唯一的楼编号和地址。每幢公寓楼中有多套公寓，每套公寓在楼内有唯一的编号（不同公寓楼内的公寓号可相同）。系统需记录每套公寓的卧室数和卫生间数。

2．员工和租客在系统中有唯一的编号（员工编号和租客编号）。

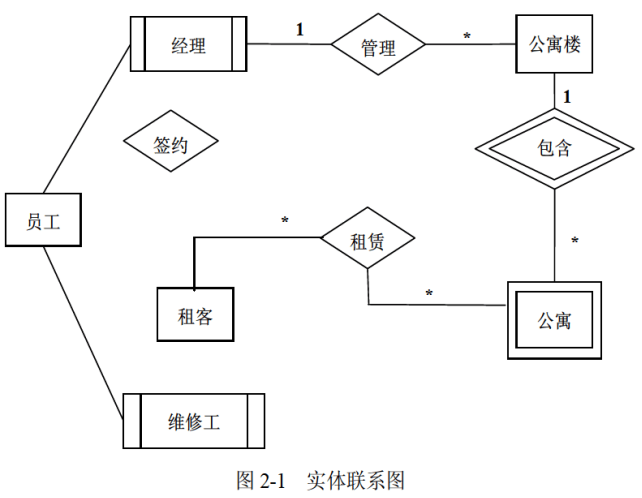
3．对于每个租客，系统需记录姓名、多个联系电话、一个银行账号（方便自动扣房租）、一个紧急联系人的姓名及联系电话。

4．系统需记录每个员工的姓名、一个联系电话和月工资。员工类别可以是经理或维修工，也可兼任。每个经理可以管理多幢公寓楼。每幢公寓楼必须由一个经理管理。系统需记录每个维修工的业务技能，如：水暖维修、电工、木工等。

5．租客租赁公寓必须和公司签订租赁合同。一份租赁合同通常由一个或多个租客（合租）与该公寓楼的经理签订，一个租客也可租赁多套公寓。合同内容应包含签订日期、开始时间、租期、押金和月租金。

【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。



【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）：

联系电话（电话号码，租客编号）

租客（租客编号，姓名，银行账号，联系人姓名，联系人电话）

员工（员工编号，姓名，联系电话，类别，月工资， （a） ）

公寓楼（ （b） ，地址，经理编号）

公寓（楼编号，公寓号，卧室数，卫生间数）

合同（合同编号，租客编号，楼编号，公寓号，经理编号，签订日期，起始日期，租期， （c） ，押金）

【问题1】（4.5分）

补充图2-1中的“签约”联系所关联的实体及联系类型。

【问题2】（4.5分）

补充逻辑结构设计中的（a）、（b）、（c）三处空缺。

【问题3】（6分）

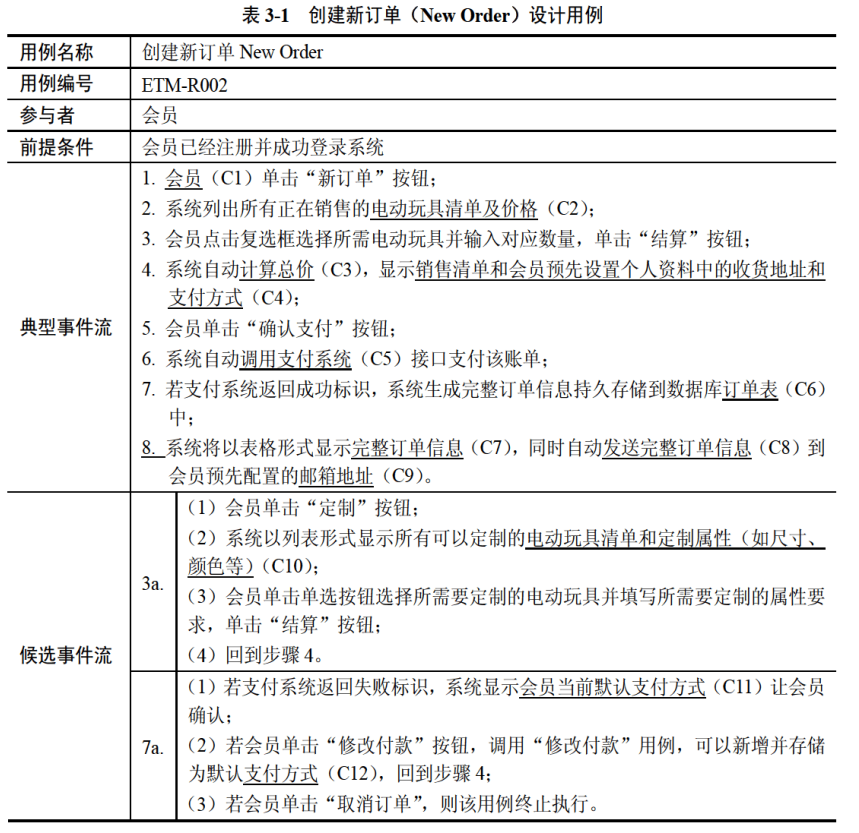
在租期内，公寓内设施如出现问题，租客可在系统中进行故障登记，填写故障描述，每项故障由系统自动生成唯一的故障编号，由公司派维修工进行故障维修，系统需记录每次维修的维修日期和维修内容。请根据此需求，对图2-1进行补充，并将所补充的ER图内容转换为一个关系模式，请给出该关系模式。

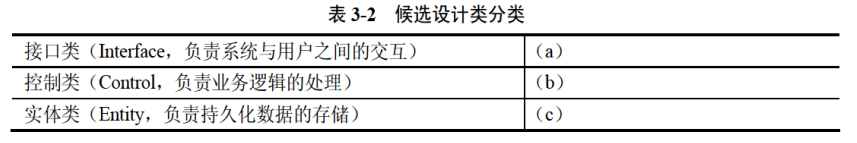
**试题三（共15分）**

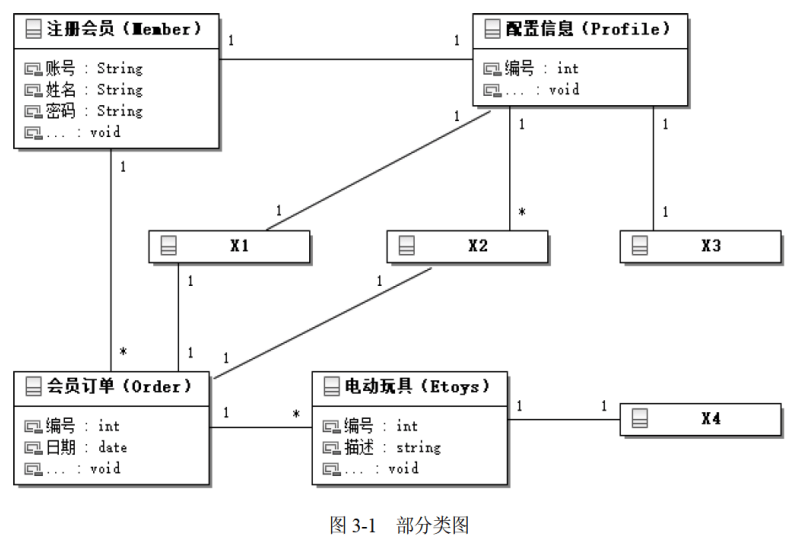
阅读下列说明和UML图，回答问题1至问题3，将解答填入答题纸的对应栏内。

【说明】

某玩具公司正在开发一套电动玩具在线销售系统，用于向注册会员提供端对端的玩具定制和销售服务。在系统设计阶段，“创建新订单（New Order）”的设计用例详细描述如表3-1所示，候选设计类分类如表3-2所示，并根据该用例设计出部分类图如图3-1所示。



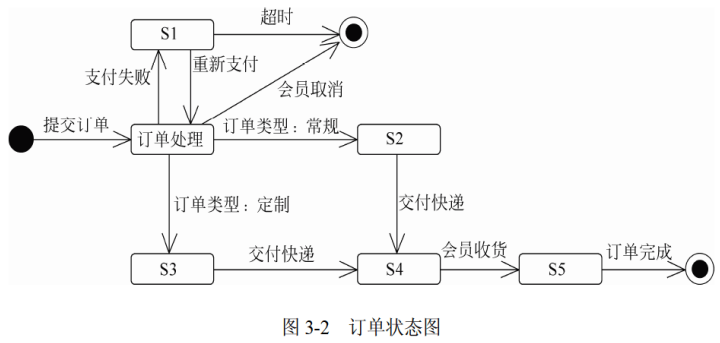




在订单处理的过程中，会员可以点击“取消订单”取消该订单。如果支付失败，该订单将被标记为挂起状态，可后续重新支付，如果挂起超时30分钟未支付，系统将自动取消该订单。订单支付成功后，系统判断订单类型：

（1）对于常规订单，标记为备货状态，订单信息发送到货运部，完成打包后交付快递发货；

（2）对于定制订单，会自动进入定制状态，定制完成后交付快递发货。会员在系统中点击“收货”按钮变为收货状态，结束整个订单的处理流程。根据订单处理过程所设计的状态图如图3-2所示。



【问题1】（6分）

根据表3-1中所标记的候选设计类，请按照其类别将编号C1〜C12分别填入表3-2中的（a）、（b）和（c）处。

【问题2】（4分）

根据创建新订单的用例描述，请给出图3-1中X1〜X4处对应类的名称。

【问题3】（5分）

根据订单处理过程的描述，在图3-2中S1〜S5处分别填入对应的状态名称。

**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，将解答填入答题纸的对应栏内。

【说明】

假币问题：有n枚硬币，其中有一枚是假币，已知假币的重量较轻。现只有一个天平，要求用尽量少的比较次数找出这枚假币。

【分析问题】

将n枚硬币分成相等的两部分：

（1）当n为偶数时，将前后两部分，即1…n/2和n/2+1…n，放在天平的两端，较轻的一端里有假币，继续在较轻的这部分硬币中用同样的方法找出假币；

（2）当n为奇数时，将前后两部分，即1…(n-1)/2和(n+1)/2+1…n，放在天平的两端，较轻的一端里有假币，继续在较轻的这部分硬币中用同样的方法找出假币；若两端重量相等，则中间的硬币，即第(n+1)/2枚硬币是假币。

【C代码】

下面是算法的C语言实现，其中：

coins[]：硬币数组

first, last：当前考虑的硬币数组中的第一个和最后一个下标

#include <stdio.h>

int getCounterfeitCoin(int coins[], int first, int last) {

int firstSum = 0, lastSum = 0;

int i;

if (first == last - 1) { /\* 只剩两枚硬币 \*/

if (coins[first] < coins[last])

return first;

return last;

}

if ((last - first + 1) % 2 == 0) { /\* 偶数枚硬币 \*/

for (i = first; i < (1) ; i ++ ) {

firstSum += coins[i];

}

for (i = first + (last - first) / 2 + 1; i < last + 1; i ++ ) {

lastSum += coins[i];

}

if ( (2) ) {

return getCounterfeitCoin(coins, first,

first + (last - first) / 2);

} else {

return getCounterfeitCoin(coins,

first + (last - first) / 2 + 1, last);

}

} else { /\*奇数枚硬币\*/

for (i = first; i < first + (last - first) / 2; i ++ ) {

firstSum += coins[i];

}

for (i = first + (last - first) / 2 + 1; i < last + 1; i ++ ) {

lastSum += coins[i];

}

if (firstSum < lastSum) {

return getCounterfeitCoin(coins, first,

first + (last - first) / 2 - 1);

} else if (firstSum > lastSum) {

return getCounterfeitCoin(coins,

first + (last - first) / 2 + 1, last);

} else {

return (3) ;

}

}

}

【问题1】（6分）

根据题干说明，填充C代码中的空（1）〜（3）。

【问题2】（6分）

根据题干说明和C代码，算法采用了 （4） 设计策略。

函数getCounterfeitCoin的时间复杂度为 （5） （用O表示）。

【问题3】（3分）

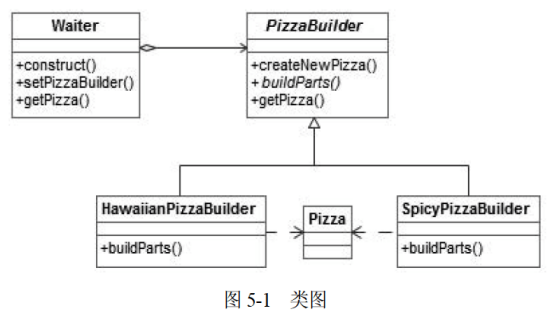
若输入的硬币数为30，则最少的比较次数为 （6） ，最多的比较次数为 （7） 。

**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】

某快餐厅主要制作并出售儿童套餐，一般包括主餐（各类比萨）、饮料和玩具，其餐品种类可能不同，但其制作过程相同。前台服务员（Waiter）调度厨师制作套餐。现采用生成器（Builder）模式实现制作过程，得到如图5-1所示的类图。



【C++代码】

#include <iostream>

#include <string>

using namespace std;

class Pizza {

private : string parts;

public :

void setParts(string parts) { this->parts = parts; }

string getParts() { return parts; }

};

class PizzaBuilder {

protected : Pizza\* pizza;

public :

Pizza\* getPizza() { return pizza; }

void createNewPizza() { pizza = new Pizza(); }

(1) ;

};

class HawaiianPizzaBuilder : public PizzaBuilder {

public :

void buildParts() { pizza->setParts("cross + mild + ham&pineapple"); }

};

class SpicyPizzaBuilder : public PizzaBuilder {

public :

void buildParts() {

pizza->setParts("pan baked + hot + pepperoni&salami");

}

};

class Waiter {

private :

PizzaBuilder\* pizzaBuilder;

public :

void setPizzaBuilder(PizzaBuilder\* pizzaBuilder) { /\* 设置构建器 \*/

(2) ;

}

Pizza\* getPizza() { return pizzaBuilder->getPizza(); }

void construct() {

pizzaBuilder->createNewPizza();

(3) ;

}

};

int main() {

Waiter\* waiter = new Waiter();

PizzaBuilder\* hawaiian\_pizzabuilder = new HawaiianPizzaBuilder();

(4) ;

(5) ;

cout << "pizza：" << waiter->getPizza()->getParts() << endl;

}

程序的输出结果为：

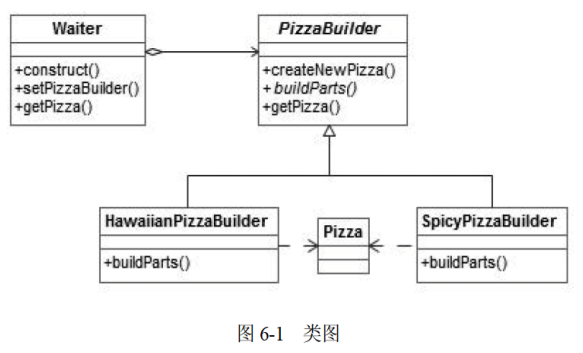
pizza：cross + mild + ham\*pineapple

**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】

某快餐厅主要制作并出售儿童套餐，一般包括主餐（各类比萨）、饮料和玩具，其餐品种类可能不同，但其制作过程相同。前台服务员（Waiter）调度厨师制作套餐。现采用生成器（Builder）模式实现制作过程，得到如图6-1所示的类图。



【Java代码】

class Pizza {

private String parts;

public void setParts(String parts) { this.parts = parts; }

public String toString() { return this.parts; }

}

abstract class PizzaBuilder {

protected Pizza pizza;

public Pizza getPizza() { return pizza; }

public void createNewPizza() { pizza = new Pizza(); }

public (1) ;

}

class HawaiianPizzaBuilder extends PizzaBuilder {

public void buildParts(){

pizza.setParts("cross + mild + ham&pineapp1e");

}

}

class SpicyPizzaBuilder extends PizzaBuilder {

public void buildParts() {

pizza.setParts("panbaked + hot + pepperoni&salami");

}

}

class Waiter {

private PizzaBuilder pizzaBuilder;

public void setPizzaBuilder(PizzaBuilder pizzaBuilder) {/\*设置构建器\*/

(2) ;

}

public Pizza getPizza() { return pizzaBuilder.getPizza(); }

public void construct() { /\* 构建 \*/

pizzaBuilder.createNewPizza();

(3) ;

}

}

class FastFoodOrdering {

public static viod mainSting[]args) {

Waiter waiter = new Waiter();

PizzaBuilder hawaiian\_pizzabuilder = new HawaiianPizzaBuilder();

(4) ;

(5) ;

System.out.println("pizza：" + waiter.getPizza());

}

}

程序的输出结果为：

pizza：cross + mild + ham&pineapple

**全国计算机技术与软件专业技术资格（水平）考试**

# 2017年下半年 软件设计师 下午试卷

（考试时间14:00～16:30共150分钟）

|  |
| --- |
| **请按下述要求正确填写答题卡** |

1. 在答题纸的指定位置填写你所在的省、自治区、直辖市、计划单列市的名称。

2. 在答题纸的指定位置填写准考证号、出生年月日和姓名。

3. 答题纸上除填写上述内容外只能写解答。

4. 本试卷共6道题，试题一至试题四是必答题，试题五至试题六选答1道。每题15分，满分75分。

5. 解答时字迹务必清楚，字迹不清时，将不评分。

6. 仿照下面的例题，将解答写在答题纸的对应栏内。

例题

2017年下半年全国计算机技术与软件专业技术资格（水平）考试日期是（1）月（2）日。

因为正确的解答是“11月11日”，故在答题纸的对应栏内写上“11”和“11”（参看下表）。

|  |  |
| --- | --- |
| 例题 | 解答栏 |
| （1） | 11 |
| （2） | 11 |

**试题一（共15分）**

阅读下列说明和数据流图，回答问题1至问题4，将解答填入答题纸的对应栏内。

【说明】

某公司拟开发一个共享单车系统，采用北斗定位系统进行单车定位，提供针对用户的APP以及微信小程序、基于Web的管理与监控系统。该共享单车系统的主要功能如下。

（1）用户注册登录。用户在APP端输入手机号并获取验证码后进行注册，将用户信息进行存储。用户登录后显示用户所在位置周围的单车。

（2）使用单车。

①扫码/手动开锁。通过扫描二维码或手动输入编码获取开锁密码，系统发送开锁指令进行开锁，系统修改单车状态，新建单车行程。

②骑行单车。单车定时上传位置，更新行程。

③锁车结账。用户停止使用或手动锁车并结束行程后，系统根据已设置好的计费规则及使用时间自动结算，更新本次骑行的费用并显示给用户，用户确认支付后，记录行程的支付状态。系统还将重置单车的开锁密码和单车状态。

（3）辅助管理。

①查询。用户可以查看行程列表和行程详细信息。

②报修。用户上报所在位置或单车位置以及单车故障信息并进行记录。

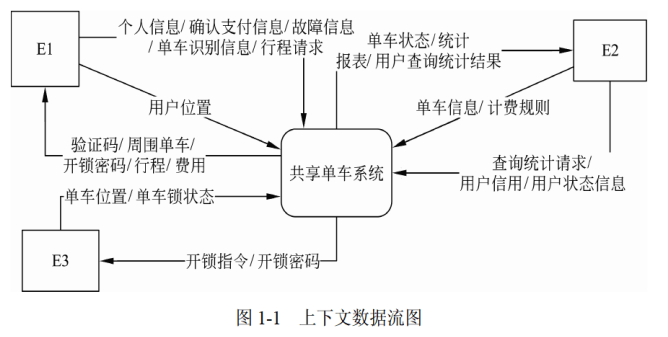
（4）管理与监控。

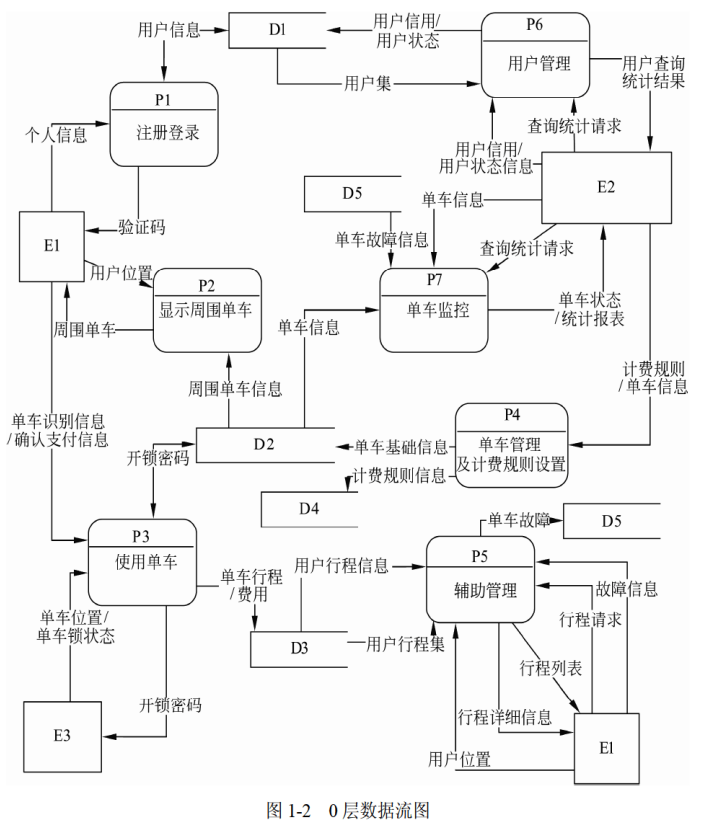
①单车管理及计费规则设置。商家对单车基础信息、状态等进行管理，对计费规则进行设置并存储。

②单车监控。对单车、故障、行程等进行查询统计。

③用户管理。管理用户信用与状态信息，对用户进行查询统计。

现采用结构化方法对共享单车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】（3分）

使用说明中的词语，给出图1-1中的实体E1〜E3的名称。

【问题2】（5分）

使用说明中的词语，给出图1-2中的数据存储D1〜D5的名称。

【问题3】（5分）

根据说明和图中术语及符号，补充图1-2中缺失的数据流及其起点和终点。

【问题4】（2分）

根据说明中术语，说明“使用单车”可以分解为哪些子加工？

**试题二（共15分）**

阅读下列说明，回答问题1至问题4，将解答填入答题纸的对应栏内。

【说明】

M公司为了便于开展和管理各项业务活动，提高公司的知名度和影响力，拟构建一个基于网络的会议策划系统。

【需求分析结果】

该系统的部分功能及初步需求分析的结果如下：

（1）M公司旗下有业务部、策划部和其他部门。部门信息包括部门号、部门名、主管、联系电话和邮箱号；每个部门只有一名主管，只负责管理本部门的工作，且主管参照员工关系的员工号；一个部门有多名员工，每名员工属于且仅属于一个部门。

（2）员工信息包括员工号、姓名、职位、联系方式和薪资。职位包括主管、业务员、策划员等。业务员负责受理用户申请，设置受理标志。一名业务员可以受理多个用户申请，但一个用户申请只能由一名业务员受理。

（3）用户信息包括用户号、用户名、银行账号、电话、联系地址。用户号唯一标识用户信息中的每一个元组。

（4）用户申请信息包括申请号、用户号、会议日期、天数、参会人数、地点、预算和受理标志。申请号唯一标识用户申请信息中的每一个元组，且一个用户可以提交多个申请，但一个用户申请只对应一个用户号。

（5）策划部主管为已受理的用户申请制定会议策划任务。策划任务包括申请号、任务明细和要求完成时间。申请号唯一标识策划任务的每一个元组。一个策划任务只对应一个己受理的用户申请，但一个策划任务可由多名策划员参与执行，且一名策划员可以参与执行多项策划任务。

【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。



【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（部门号，部门名，部门主管，联系电话，邮箱号）

员工（员工号，姓名， （a） ，联系方式，薪资）

用户（用户名， （b） ，电话，联系地址）

用户申请（申请号，用户号，会议日期，天数，参会人数，地点，受理标志， （c） ）

策划任务（申请号，任务明细， （d） ）

执行（申请号，策划员，实际完成时间，用户评价）

【问题1】（5分）

根据问题描述，补充五个联系，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表示，联系的类型为1:1、1:n和m:n（或1:1、1:\*和\*:\*）。

【问题2】（4分）

根据题意，将关系模式中的空（a）〜（d）补充完整，并填入答题纸对应的位置上。

【问题3】（4分）

给出“用户申请”和“策划任务”关系模式的主键和外键。

【问题4】（2分）

请问“执行”关系模式的主键为全码的说法正确吗？为什么？

**试题三（共15分）**

阅读下列说明和UML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】

某大学拟开发一个用于管理学术出版物（Publication）的数字图书馆系统，用户可以从该系统查询或下载已发表的学术出版物。系统的主要功能如下：

1．登录系统。系统的用户（User）仅限于该大学的学生（Student）、教师（Faculty）和其他工作人员（Staff）。在访问系统之前，用户必须使用其校园账户和密码登录系统。

2．查询某位作者（Author）的所有出版物。系统中保存了会议文章（ConfPaper）、期刊文章（JournalArticle）和校内技术报告（TechReport）等学术出版物的信息，如题目、作者以及出版年份等。除此之外，系统还存储了不同类型出版物的一些特有信息：

（1）对于会议文章，系统还记录了会议名称、召开时间以及召开地点；

（2）对于期刊文章，系统还记录了期刊名称、出版月份、期号以及主办单位；

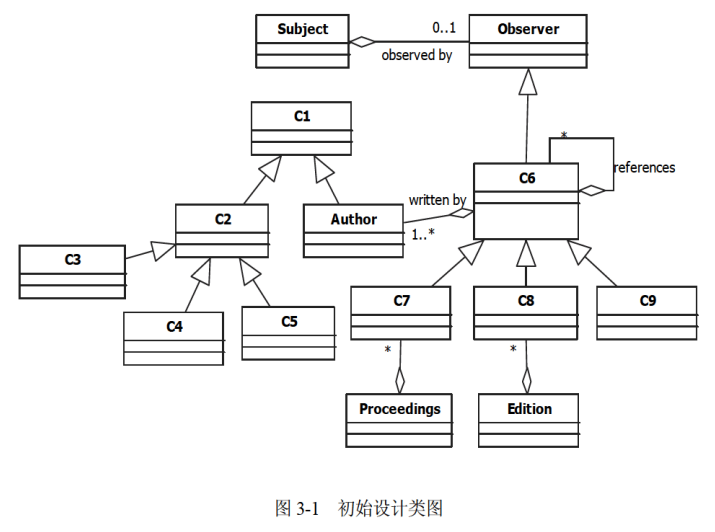
（3）对于校内技术报告，系统记录了由学校分配的唯一ID。

3．查询指定会议集（Proceedings）或某个期刊特定期（Edition）的所有文章。会议集包含了发表在该会议（在某个特定时间段、特定地点召开）上的所有文章。期刊的每一期在特定时间发行，其中包含若干篇文章。

4．下载出版物。系统记录每个出版物被下载的次数。

5．查询引用了某篇出版物的所有出版物。在学术出版物中引用他人或早期的文献作为相关工作或背景资料是很常见的现象。用户也可以在系统中为某篇出版物注册引用通知，若有新的出版物引用了该出版物，系统将发送电子邮件通知该用户。

现在采用面向对象方法对该系统进行开发，得到系统的初始设计类图如图3-1所示。



【问题1】（9分）

根据说明中的描述，给出图3-1中C1〜C9所对应的类名。

【问题2】（4分）

根据说明中的描述，给出图3-1中类C6〜C9的属性。

【问题3】（2分）

图3-1中包含了哪种设计模式？实现的是该系统的哪个功能？

**试题四（共15分）**

阅读下列说明和C代码，回答问题1至问题2，将解答填入答题纸的对应栏内。

【说明】

一个无向连通图G点上的哈密尔顿（Hamilton）回路是指从图G上的某个顶点出发，经过图上所有其他顶点一次且仅一次，最后回到该顶点的路径。一种求解无向图上哈密尔顿回路算法的基本思想如下：

假设图G存在一个从顶点出发的哈密尔顿回路————————...——1——。算法从顶点出发，访问该顶点的一个未被访问的邻接顶点，接着从顶点出发，访问一个未被访问的邻接顶点，对顶点，重复进行以下操作：访问的一个未被访问的邻接接点；若的所有邻接顶点均已被访问，则返回到顶点，考虑的下一个未被访问的邻接顶点，仍记为；直到找到一条哈密尔顿回路或者找不到哈密尔顿回路，算法结束。

【C代码】

下面时算法的C语言实现。

（1）常量和变量说明

n：图G中的顶点数

c[][]：图G的邻接矩阵

k：统计变量，当前已经访问的顶点数为k + 1

x[k]：第k个访问的顶点编号，从0开始

visited[x[k]]：第k个顶点的访问标志，0表示未访问，1表示已访问

（2）C程序

#include <stdio.h>

#include <stdlib.h>

#define MAX 100

void Hamilton(int n, int x[MAX], int c[MAX][MAX]) {

int i;

int visited[MAX];

int k;

/\* 初始化x数组和visited数组 \*/

for (i = 0; i < n; i ++ ) {

x[i] = 0;

visited[i] = 0;

}

/\* 访问起始顶点 \*/

k = 0;

(1) ;

x[0] = 0;

k = k + 1;

/\* 访问其他顶点 \*/

while (k >= 0) {

x[k] = x[k] + 1;

while (x[k] < n) {

/\* 邻接顶点x[k]未被访问过 \*/

if ( (2) && c[x[k - 1]][x[k]] == 1) {

break;

} else {

x[k] = x[k] + 1;

}

}

if (x[k] < n && k == n - 1 && (3) ) { /\* 找到一条哈密尔顿回路 \*/

for (k = 0; k < n; k ++ ) {

printf("%d--", x[k]); /\* 输出哈密尔顿回路 \*/

}

printf("%d\n", x[0]);

return;

/\* 设置当前顶点的访问标志，继续下一个顶点 \*/

} else if (x[k] < n && k < n - 1) {

(4) ;

k = k + 1;

} else { /\* 没有未被访问过的邻接顶点，回退到上一个顶点 \*/

x[k] = 0;

visited[x[k]] = 0;

(5) ;

}

}

}

【问题1】（10分）

根据题干说明。填充C代码中的空（1）~（5）。

【问题2】（5分）

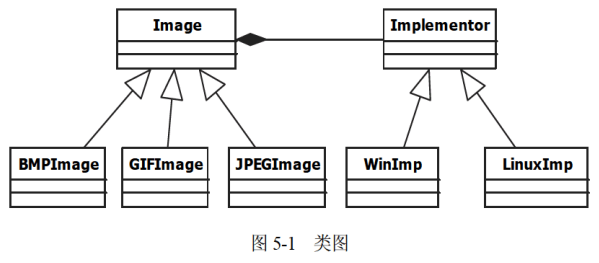
根据题干说明和C代码，算法采用的设计策略为（6），该方法在遍历图的顶点时，采用的是（7）方法（深度优先或广度优先）。

**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】

某图像预览程序要求能够查看BMP、JPEG和GIF三种格式的文件，且能够在Windows和Linux两种操作系统上运行。程序需具有较好的扩展性以支持新的文件格式和操作系统。为满足上述需求并减少所需生成的子类数目，现采用桥接（Bridge）模式进行设计，得到如图5-1所示的类图。



【C++代码】

#include <iostream>

#include <string>

using namespace std;

class Matrix { // 各种格式的文件最终都被转化为像素矩阵

// 此处代码省略

};

class Implementor {

public :

(1) ; // 显示像素矩阵m

};

class WinImp : public Implementor {

public :

void doPaint(Matrix m) { /\* 调用Windows系统的绘制函数绘制像素矩阵 \*/ }

};

class LinuxImp : public Implementor {

public :

void doPaint(Matrix m) { /\* 调用Linux系统的绘制函数绘制像素矩阵 \*/ }

};

class Image {

public :

void setImp(Implementor \*imp) { this->imp = imp; }

virtual void parseFile(string fileName) = 0;

protected:

Implementor \*imp;

};

class BMPImage : public Image {

// 此处代码省略

};

class GIFImage : public Image {

public :

void parseFile(string fileName) {

// 此处解析GIF文件并获得一个像素矩阵对象m

(2) ; // 显示像素矩阵m

}

};

class JPEGImage : public Image {

// 此处代码省略

};

int main() {

// 在Linux操作系统上查看demo.gif图像文件

Image \*image = (3) ;

Implementor \*imageImp = (4) ;

(5) ;

image->parseFile("demo.gif");

return 0;

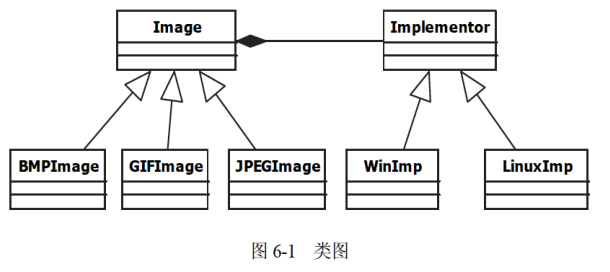
}

**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】

某图像预览程序要求能够查看BMP、JPEG和GIF三种格式的文件，且能够在Windows和Linux两种操作系统上运行。程序需具有较好的扩展性以支持新的文件格式和操作系统。为满足上述需求并减少所需生成的子类数目，现采用桥接（Bridge）模式进行设计，得到如图6-1所示的类图。



【Java代码】

import java.util.\*;

class Matrix { // 各种格式的文件最终都被转化为像素矩阵

// 此处代码省略

}

abstract class Implementor {

public (1) ; // 显示像素矩阵m

}

class WinImp extends Implementor {

public void doPaint(Matrix m) { // 调用 Windows 系统的绘制函数绘制像素矩阵

// 此处代码省略

}

}

class LinuxImp extends Implementor {

public void doPaint(Matrix m) { // 调用 Linux 系统的绘制函数绘制像素矩阵

// 此处代码省略

}

}

abstract class Image {

public void setImp(Implementor imp) { this.imp = imp; }

public abstract void parseFile(String fileName);

protected Implementor imp;

}

class BMPImage extends Image {

// 此处代码省略

}

class GIFImage extends Image {

public void parseFile(String fileName) {

// 此处解析BMP文件并获得一个像素矩阵对象m

(2) ; // 显示像素矩阵m

}

}

class JPEGImage extends Image {

// 此处代码省略

}

class Main {

public static void main(String[]args) {

// 在Linux操作系统上查看demo.gif图像文件

Image image = (3) ;

Implementor imageImp = (4) ;

(5) ;

Image.parseFile("demo.gif");

}

}

**全国计算机技术与软件专业技术资格（水平）考试**

# 2018年上半年 软件设计师 下午试卷

（考试时间14:00～16:30共150分钟）

|  |
| --- |
| **请按下述要求正确填写答题卡** |

1. 在答题纸的指定位置填写你所在的省、自治区、直辖市、计划单列市的名称。

2. 在答题纸的指定位置填写准考证号、出生年月日和姓名。

3. 答题纸上除填写上述内容外只能写解答。

4. 本试卷共6道题，试题一至试题四是必答题，试题五至试题六选答1道。每题15分，满分75分。

5. 解答时字迹务必清楚，字迹不清时，将不评分。

6. 仿照下面的例题，将解答写在答题纸的对应栏内。

例题

2018年上半年全国计算机技术与软件专业技术资格（水平）考试日期是（1）月（2）日。

因为正确的解答是“5月26日”，故在答题纸的对应栏内写上“5”和“26”（参看下表）。

|  |  |
| --- | --- |
| 例题 | 解答栏 |
| （1） | 5 |
| （2） | 26 |

**试题一（共15分）**

阅读下列说明和数据流图，回答问题1至问题4，将解答填入答题纸的对应栏内。

【说明】

某医疗护理机构为老年人或有护理需求者提供专业护理，现欲开发一基于Web的医疗管理系统，以改善医疗护理效率。该系统的主要功能如下：

（1）通用信息查询。客户提交通用信息查询请求，查询通用信息表，返回查询结果。

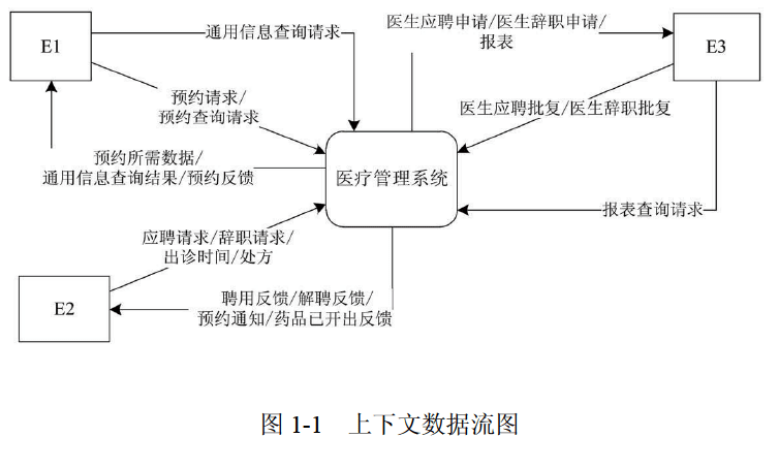
（2）医生聘用。医生提出应聘／辞职申请，交由主管进行聘用/解聘审批，更新医生表，并给医生反馈聘用/解聘结果；删除解聘医生的出诊安排。

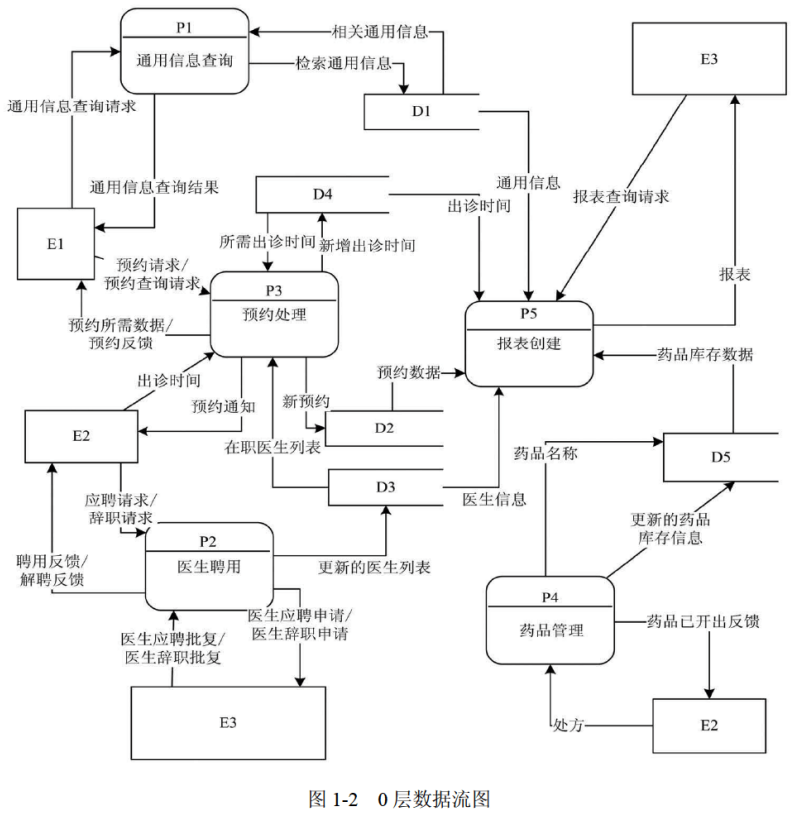
（3）预约处理。医生安排出诊时间，存入医生出诊时间表；根据客户提交的预约查询请求，查询在职医生及其出诊时间等预约所需数据并返回；创建预约，提交预约请求，在预约表中新增预约记录，更新所约医生出诊时间并给医生发送预约通知；给客户反馈预约结果。

（4）药品管理。医生提交处方，根据药品名称从药品数据中查询相关药品库存信息，开出药品，更新对应药品的库存以及预约表中的治疗信息；给医生发送“药品已开出”反馈。

（5）报表创建。根据主管提交的报表查询请求（报表类型和时间段），从预约数据、通用信息、药品库存数据、医生以及医生出诊时间中进行查询，生成报表返回给主管。

现采用结构化方法对医疗管理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】（3分）

使用说明中的词语，给出图1-1中的实体E1〜E3的名称。

【问题2】（5分）

使用说明中的词语，给出图1-2中的数据存储D1〜D5的名称。

【问题3】（4分）

使用说明和图中术语，补充图1-2中缺失的数据流及其起点和终点。

【问题4】（3分）

使用说明中的词语，说明“预约处理”可以分解为哪些子加工，并说明建模图1-1和图1-2是如何保持数据流图平衡。

**试题二（共15分）**

阅读下列说明，回答问题1至问题3，将解答填入答题纸的对应栏内。

【说明】

某海外代购公司为扩展公司业务，需要开发一个信息化管理系统。请根据公司现有业务及需求完成该系统的数据库设计。

【需求分析结果】

（1）记录公司员工信息。员工信息包括工号、身份证号、姓名、性别和一个手机号，工号唯一标识每位员工，员工分为代购员和配送员。

（2）记录采购的商品信息。商品信息包括商品名称、所在超市名称、采购价格、销售价格和商品介绍，系统内部用商品条码唯一标识每种商品。一种商品只在一家超市代购。

（3）记录顾客信息。顾客信息包括顾客真实姓名、身份证号（清关缴税用）、一个手机号和一个收货地址，系统自动生成唯一的顾客编号。

（4）记录托运公司信息。托运公司信息包括托运公司名称、电话和地址，系统自动生成唯一的托运公司编号。

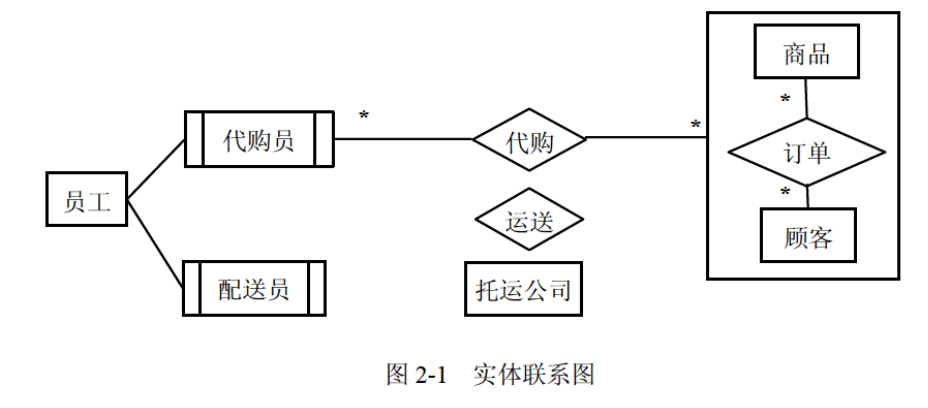
（5）顾客登录系统之后，可以下订单购买商品。订单支付成功后，系统记录唯一的支付凭证编号，顾客需要在订单里指定运送方式：空运或海运。

（6）代购员根据顾客的订单在超市采购对应商品，一份订单所含的多个商品可能由多名代购员从不同超市采购。

（7）采购完的商品交由配送员根据顾客订单组合装箱，然后交给托运公司运送。托运公司按顾客订单核对商品名称和数量，然后按顾客的地址进行运送。

【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。



【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）：

员工（工号，身份证号，姓名，性别，手机号）

商品（条码，商品名称，所在超市名称，采购价格，销售价格，商品介绍）

顾客（编号，姓名，身份证号，手机号，收货地址）

托运公司（托运公司编号，托运公司名称，电话，地址）

订单（订单ID， （a） ，商品数量，运送方式，支付凭证编号）

代购（代购ID，代购员工号， （b） ）

运送（运送ID，配送员工号，托运公司编号，订单ID，发运时间）

【问题1】（3分）

根据问题描述，补充图2-1的实体联系图。

【问题2】（6分）

补充逻辑结构设计结果中的（a）、（b）两处空缺。

【问题3】（6分）

为方便顾客，允许顾客在系统中保存多组收货地址。请根据此需求，增加“顾客地址”弱实体，对图2-1进行补充，并修改“运送”关系模式。

**试题三（共15分）**

阅读下列说明和UML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ETC（Electronic Toll Collection，不停车收费）系统在高速公路沿线的特定位置上设置一个横跨道路上空的龙门架（Toll gantry），龙门架下包括6条车道（Traffic lanes），每条车道上安装有雷达传感器（Radar sensor）、无线传输器（Radio transceiver）和数码相机（Digital Camera）等用于不停车收费的设备，以完成正常行驶速度下的收费工作。该系统的基本工作过程如下：

（1）每辆汽车上安装有车载器，驾驶员（Driver）将一张具有唯一识别码的磁卡插入车载器中。磁卡中还包含有驾驶员账户的当前信用记录。

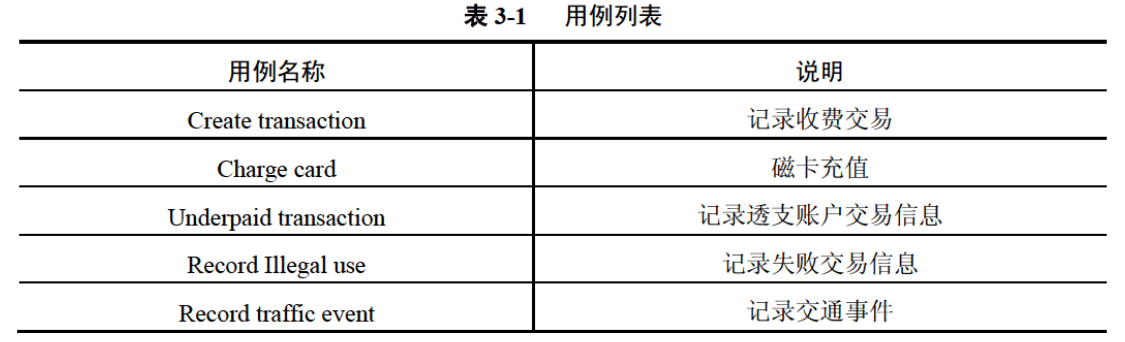
（2）当汽车通过某条车道时，不停车收费设备识别车载器内的特有编码，判断车型，将收集到的相关信息发送到该路段所属的区域系统（Regional center）中，计算通行费用，创建收费交易（Transaction），从驾驶员的专用账户中扣除通行费用。如果驾驶员账户透支，则记录透支账户交易信息。区域系统再将交易后的账户信息发送到维护驾驶员账户信息的中心系统（Central system）。

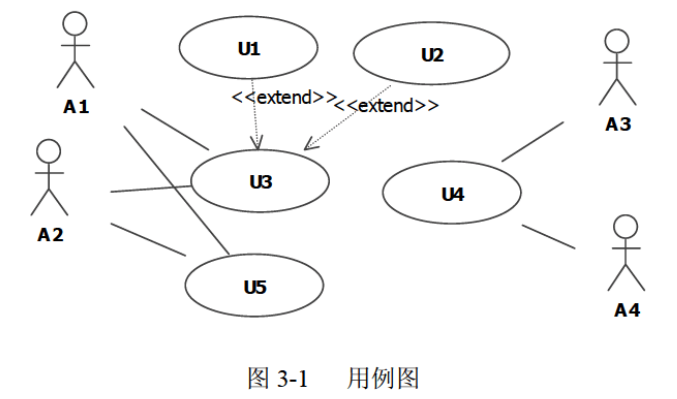
（3）车载器中的磁卡可以使用邮局的付款机进行充值。充值信息会传送至中心系统，以更新驾驶员账户的余额。

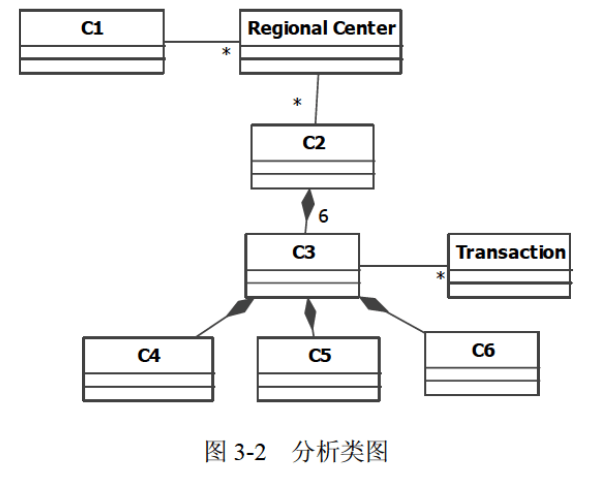
（4）当没有安装车载器或者车载器发生故障的车辆通过车道时，车道上的数码相机将对车辆进行拍照，并将车辆照片及拍摄时间发送到区域系统，记录失败的交易信息；并将该交易信息发送到中心系统。

（5）区域系统会获取不停车收费设备所记录的交通事件（Traffic events）；交通广播电台（Traffic advice center）根据这些交通事件进行路况分析并播报路况。

现采用面向对象方法对上述系统进行分析与设计，得到如表3-1所示的用例列表以及如图3-1所示的用例图和图3-2所示的分析类图。







【问题1】（4分）

根据说明中的描述，给出图3-1中A1〜A4所对应的参与者名称。

【问题2】（5分）

根据说明中的描述及表3-1，给出图3-1中U1〜U5所对应的用例名称。

【问题3】（6分）

根据说明中的描述，给出图3-2中C1〜C6所对应的类名。

**试题四（共15分）**

阅读下列说明和C代码，回答问题1至问题2，将解答填入答题纸的对应栏内。

【说明】

某公司购买长钢条，将其切割后进行出售。切割钢条的成本可以忽略不计，钢条的长度为整英寸。已知价格表，其中表示长度为英寸的钢条的价格。现要求解使销售收益最大的切割方案。

求解此切割方案的算法基本思想如下：

假设长钢条的长度为英寸，最佳切割方案的最左边切割段长度为英寸，则继续求解剩余长度为英寸钢条的最佳切割方案。考虑所有可能的，得到的最大收益对应的切割方案即为最佳切割方案。的递归定义如下：

对此递归式，给出自顶向下和自底向上两种实现方式。

【C代码】

常量和变量说明

n：长钢条的长度

p[]：价格数组

#define LEN 100

int Top\_Down\_Cut\_Rod(int p[], int n) { /\* 自顶向下 \*/

int r = 0;

int i;

if (n == 0) {

return 0;

}

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

int tmp = p[i] + Top\_Down\_Cut\_Rod(p, n - i);

r = (r >= tmp) ? r : tmp;

}

return r;

}

int Bottom\_Up\_Cut\_Rod(int p[], int n) { /\* 自底向上 \*/

int r[LEN] = {0};

int temp = 0;

int i, j;

for (j = 1; j <= n; j ++ ) {

temp = 0;

for (i = 1; (2) ; i ++ ) {

temp = (3) ;

}

(4) ;

}

return r[n];

}

【问题1】（8分）

根据说明，填充C代码中的空（1）〜（4）。

【问题2】（7分）

根据说明和C代码，算法采用的设计策略为 （5） 。

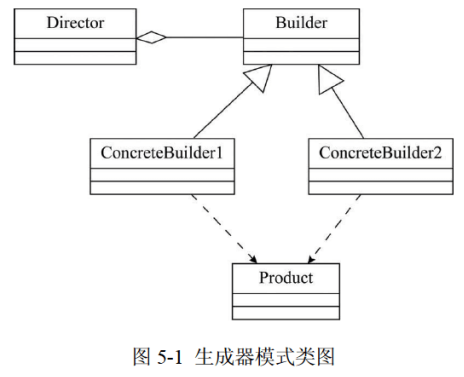
求解时，自顶向下方法的时间复杂度为 （6） ;自底向上方法的时间复杂度为 （7） （用O表示）。

**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】

生成器（Builder）模式的意图是将一个复杂对象的构建与它的表示分离，使得同样的构建过程可以创建不同的表示。图5-1所示为其类图。



【C++代码】

#include <iostream>

#include <string>

using namespace std;

class Product {

private :

string partA, partB;

public :

Product() { }

void setPartA(const string& s) { partA = s; }

void setPartB(const string& s) { partB = s; }

// 其余代码省略

};

class Builder {

public :

(1) ;

virtual void buildPartB() = 0;

(2) ;

};

class ConcreteBuilder1 : public Builder {

private :

Product\* product;

public :

ConcreteBuilder1() { product = new Product(); }

void buildPartA() { (3) ("Component A"); }

void buildPartB() { (4) ("Component B"); }

Product\* getResult() { return product; }

// 其余代码省略

};

class ConcreteBuilder2 : public Builder {

/\* 代码省略 \*/

};

class Director {

private:

Builder\* builder;

public :

Director(Builder\* pBuilder) { builder = pBuilder; }

void construct() {

(5) ;

// 其余代码省略

}

// 其余代码省略

};

int main() {

Director\* director1 = new Director(new ConcreteBuilder1());

director1->construct();

delete director1;

return 0;

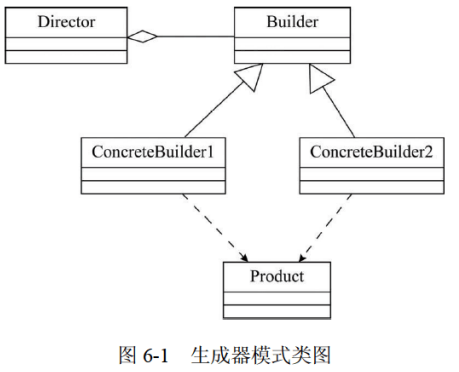
}

**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】

生成器（Builder）模式的意图是将一个复杂对象的构建与它的表示分离，使得同样的构建过程可以创建不同的表示。图6-1所示为其类图。



【Java代码】

import java.util.\*;

class Product {

private String partA;

private String partB;

public Product() {}

public void setPartA(String s) { partA = s; }

public void setPartB(String s) { partB = s; }

}

interface Builder {

public (1) ;

public void buildPartB();

public (2) ;

}

class ConcreteBuilder1 implements Builder {

private Product product;

public ConcreteBuilder1() { product = new Product(); }

public void buildPartA() { (3) ("Component A"); }

public void buildPartB() { (4) ("Component B"); }

public Product getResult() { return product;}

}

class ConcreteBuilder2 implements Builder {

// 代码省略

}

class Director {

private Builder builder;

public Director(Builder builder) { this.builder = builder; }

public void construct() {

(5) ;

// 代码省略

}

}

class Test {

public static void main(String[] args) {

Director director1 = new Director(new ConcreteBuilder1());

director1.construct();

}

}

**全国计算机技术与软件专业技术资格（水平）考试**

# 2018年下半年 软件设计师 下午试卷

（考试时间14:00～16:30共150分钟）

|  |
| --- |
| **请按下述要求正确填写答题卡** |

1. 在答题纸的指定位置填写你所在的省、自治区、直辖市、计划单列市的名称。

2. 在答题纸的指定位置填写准考证号、出生年月日和姓名。

3. 答题纸上除填写上述内容外只能写解答。

4. 本试卷共6道题，试题一至试题四是必答题，试题五至试题六选答1道。每题15分，满分75分。

5. 解答时字迹务必清楚，字迹不清时，将不评分。

6. 仿照下面的例题，将解答写在答题纸的对应栏内。

例题

2018年下半年全国计算机技术与软件专业技术资格（水平）考试日期是（1）月（2）日。

因为正确的解答是“11月10日”，故在答题纸的对应栏内写上“11”和“10”（参看下表）。

|  |  |
| --- | --- |
| 例题 | 解答栏 |
| （1） | 11 |
| （2） | 10 |

**试题一（共15分）**

阅读下列说明和数据流图，回答问题1至问题4，将解答填入答题纸的对应栏内。

【说明】

某房产中介连锁企业欲开发一个基于Web的房屋中介信息系统，以有效管理房源和客户，提高成交率。该系统的主要功能是：

（1）房源采集与管理。系统自动采集外部网站的潜在房源信息，保存为潜在房源。由经纪人联系确认的潜在房源变为房源，并添加出售/出租房源的客户。由经纪人或客户登记的出售/出租房源，系统将其保存为房源。房源信息包括基本情况、配套设施、交易类型、委托方式、业主等。经纪人可以对房源进行更新等管理操作。

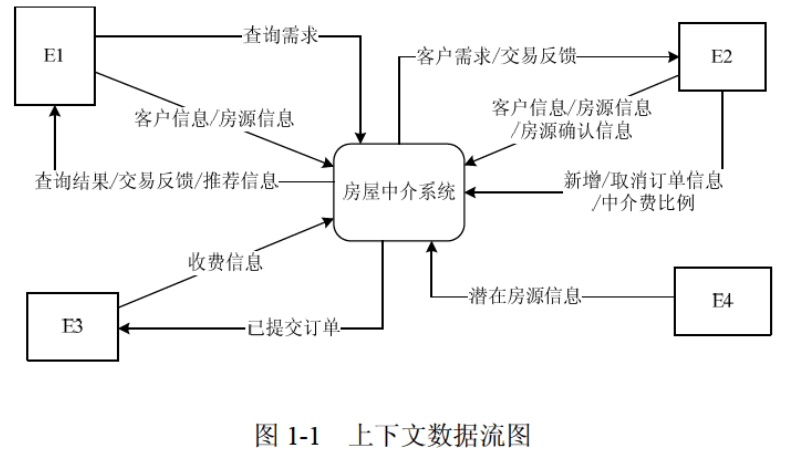
（2）客户管理。求租/求购客户进行注册、更新，推送客户需求给经纪人，或由经纪人对求租/求购客户进行登记、更新。客户信息包括身份证号、姓名、手机号、需求情况、委托方式等。

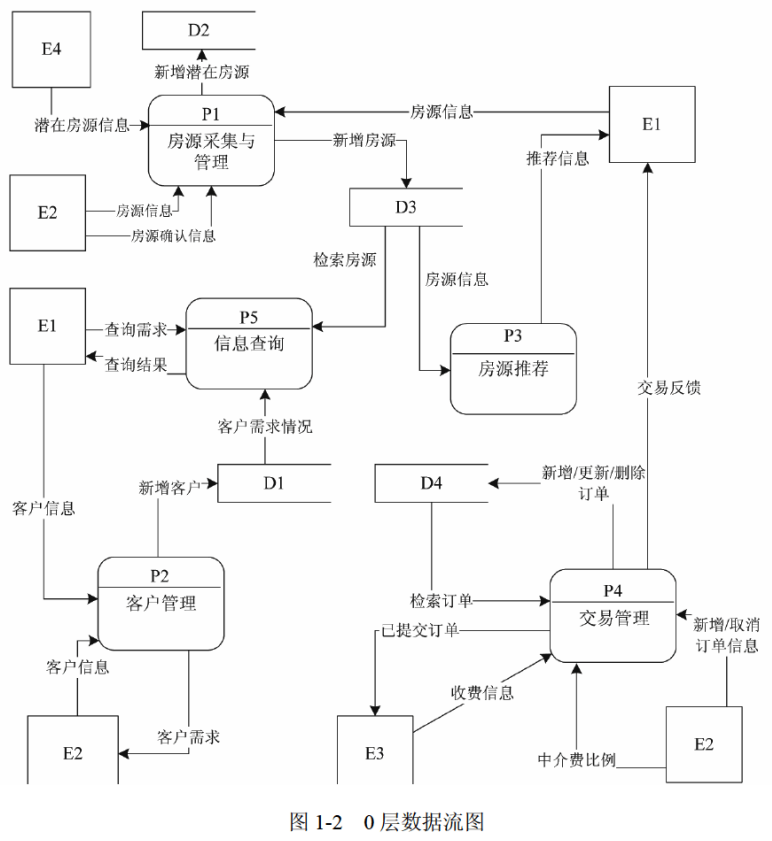
（3）房源推荐。根据客户的需求情况（求购/求租需求情况以及出售/出租房源信息），向已登录的客户推荐房源。

（4）交易管理。经纪人对租售客户双方进行交易信息管理，包括订单提交和取消，设置收取中介费比例。财务人员收取中介费之后，表示该订单已完成，系统更新订单状态和房源状态，向客户和经纪人发送交易反馈。

（5）信息查询。客户根据自身查询需求查询房屋供需信息。

现采用结构化方法对房屋中介信息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】（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】（4分）

使用说明中的词语，给出图1-2中的数据存储D1〜D4的名称。

【问题3】（3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】（4分）

根据说明中术语，给出图1-1中数据流“客户信息”“房源信息”的组成。

**试题二（共15分）**

阅读下列说明，回答问题1至问题4，将解答填入答题纸的对应栏内。

【说明】

某集团公司拥有多个分公司，为了方便集团公司对分公司各项业务活动进行有效管理，集团公司决定构建一个信息系统以满足公司的业务管理需求。

【需求分析结果】

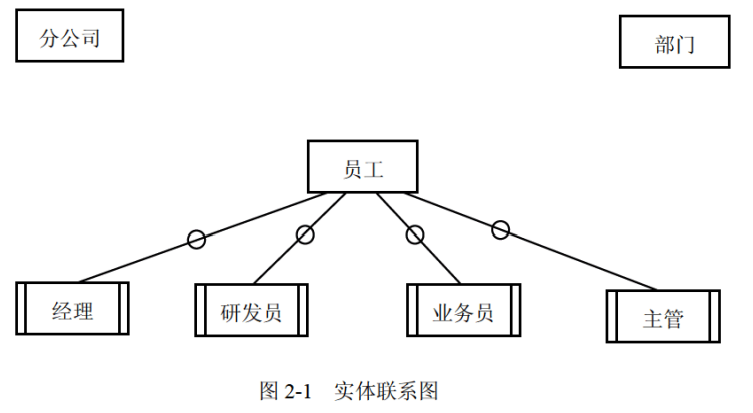
（1）分公司关系需要记录的信息包括分公司编号、名称、经理、联系地址和电话。分公司编号唯一标识分公司信息中的每一个元组。每个分公司只有一名经理，负责该分公司的管理工作。每个分公司设立仅为本分公司服务的多个业务部门，如研发部、财务部、采购部、销售部等。

（2）部门关系需要记录的信息包括部门号、部门名称、主管号、电话和分公司编号。部门号唯一标识部门信息中的每一个元组。每个部门只有一名主管，负责部门的管理工作。每个部门有多名员工，每名员工只能隶属于一个部门。

（3）员工关系需要记录的信息包括员工号、姓名、隶属部门、岗位、电话和基本工资。其中，员工号唯一标识员工信息中的每一个元组。岗位包括：经理、主管、研发员、业务员等。

【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。



【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）：

分公司（分公司编号，名称， （a） ，联系地址，电话）

部门（部门号，部门名称， （b） ，电话）

员工（员工号，姓名， （c） ，电话，基本工资）

【问题1】（4分）

根据问题描述，补充4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和联系4代替，联系的类型为1:1、1:n和m:n（或1:1、1:\*和\*:\*）。

【问题2】（5分）

根据题意，将关系模式中的空（a）〜（c）补充完整。

【问题3】（4分）

给出“部门”和“员工”关系模式的主键和外键。

【问题4】（2分）

假设集团公司要求系统能记录部门历任主管的任职时间和任职年限，那么是否需要在数据库设计时增设一个实体？为什么？

**试题三（共15分）**

阅读下列说明和UML图，回答问题1至问题3，将解答填入答题纸的对应栏内。

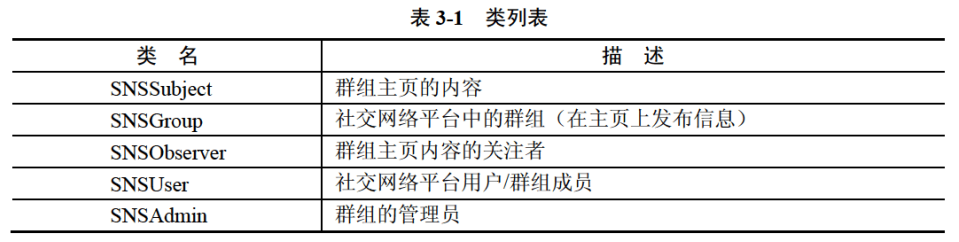
【说明】

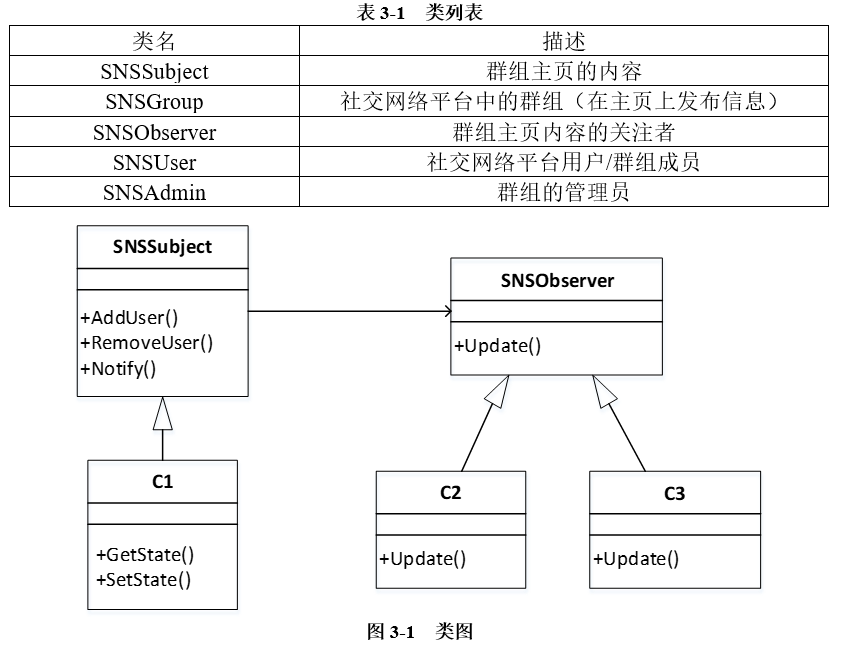
（1）社交网络平台（SNS）的主要功能之一是建立在线群组，群组中的成员之间可以互相分享或挖掘兴趣和活动。每个群组包含标题、管理员以及成员列表等信息。

（2）社交网络平台的用户可以自行选择加入某个群组。每个群组拥有一个主页，群组内的所有成员都可以查看主页上的内容。如果在群组的主页上发布或更新了信息，群组中的成员会自动接收到发布或更新后的信息。

（3）用户可以加入一个群组也可以退出这个群组。用户退出群组后，不会再接收到该群组发布或更新的任何信息。

现采用面向对象方法对上述需求进行分析与设计，得到如表3-1所示的类列表和如图3-1所示的类图。





【问题1】（6分）

根据说明中的描述，给出图3-1中C1〜C3所对应的类名。

【问题2】（6分）

图3-1中采用了哪一种设计模式？说明该模式的意图及其适用场合。

【问题3】（3分）

现在对上述社交网络平台提出了新的需求：一个群体可以作为另外一个群体中的成员，例如群体A加入群体B。那么，群体A中的所有成员就自动成为群体B中的成员。

若要实现这个新需求，需要对图3-1进行哪些修改? （以文字方式描述）

**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，将解答填入答题纸的对应栏内。

【说明】

给定一个字符序列B = b1b2….bn，其中bi∈{A, C, G, U}。B上的二级结构是一组字符对集合S={(bi, bj)},其中i, j∈{1, 2, …., n}，并满足以下四个条件：

（1）S中的每对字符是(A, U), (U, A), (C, G)和(G, C)四种组合之一；

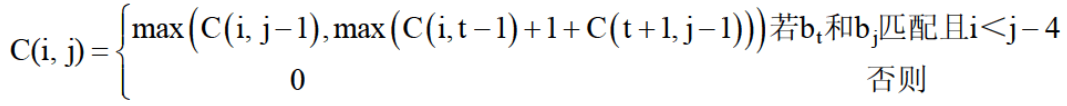
（2）S中的每对字符之间至少有四个字符将其隔开，即i < j – 4；

（3）S中每一个字符（记为bk）的配对存在两种情况：bk不参与任何配对；bk和字符bt，其中t < k – 4；

（4）（不交叉原则）若(bi, bj)和(bk, bt)是S中的两个字符对，且 i < k，则 i < k < j < 1 不成立。

B的具有最大可能字符对数的二级结构S被称为最优配对方案，求解最优配对方案中的字符对数的方法如下：

假设用C(i, j)表示字符序列bibi+1...bj的最优配对方案（即二级结构S）中的字符对数，则C(i ,j)可以递归定义为：



下面代码是算法的C语言实现，其中

n：字符序列长度

B[]：字符序列

C[][]：最优配对数量数组

【C代码】

#include <stdio.h>

#include <stdlib.h>

#define LEN 100

/\* 判断两个字符是否配对 \*/

int isMatch(char a, char b) {

if ((a == 'A' && b == 'U') || (a == 'U' && b == 'A'))

return 1;

if ((a == 'C' && b == 'G') || (a == 'G' && b == 'C'))

return 1;

return 0;

}

/\* 求最大配对数 \*/

int RNA\_2(char B[LEN], int n) {

int i, j, k, t;

int max;

int C[LEN][LEN] = {0};

for (k = 5; k <= n - 1; k ++ ) {

for (i = 1; i <= n - k; i ++ ) {

j = i + k;

(1) ;

for ( (2) ; t <= j - 4; t ++ ) {

if ( (3) && max < C[i][t - 1] + 1 + C[t + 1][j - 1])

max = C[i][t - 1] + 1 + C[t + 1][j - 1];

}

C[i][j] = max;

printf("c[%d][%d] = %d--", i, j, C[i][j]);

}

}

return (4) ;

}

【问题1】（8分）

根据题干说明，填充C代码中的空（1）〜（4）。

【问题2】（4分）

根据题干说明和C代码，算法采用的设计策略为 （5） 。

算法的时间复杂度为 （6） ,（用O表示）。

【问题3】（3分〉

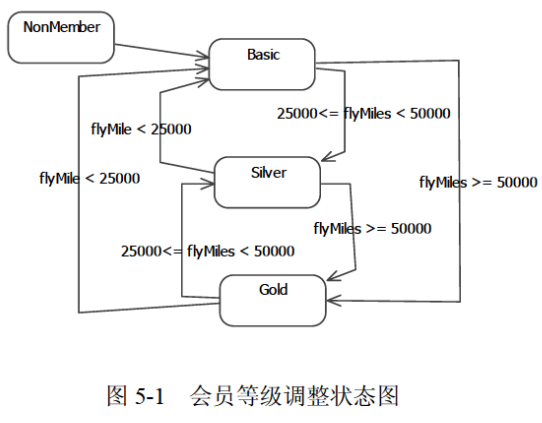
给定字符序列ACCGGUAGU，根据上述算法求得最大字符对数为 （7） 。

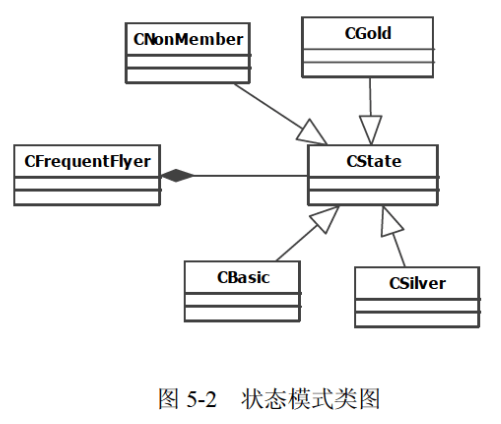
**试题五（15分）**

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】

某航空公司的会员积分系统将其会员划分为：普卡(Basic)、银卡(Silver)和金卡(Gold)三个等级。非会员(NonMember)可以申请成为普卡会员。会员的等级根据其一年内累积的里程数进行调整。描述会员等级调整的状态图如图5-1所示。现采用状态(State)模式实现上述场景，得到如图5-2所示的类图。





【C++代码】

#include <iostream>

using namespace std;

class CFrequentFlyer; clss CBasic; class CSilver; class CGold; class CNoCustomer;

// 提前引用

class CState {

private : int flyMiles; // 里程数

public :

(1) ; // 根据累积里程数调整会员等级

};

class CFrequentFlyer {

friend class CBasic; friend class CSilver; friend class CGold;

private :

CState \*state; CState \*nocustomer; CState \*basic; CState \*silver; Cstate \*gold;

double flyMiles;

public :

CFrequentFlyer() { flyMiles = 0; setState(nocustomer); }

void setState(CState \*state) { this->state = state; }

void travel(int miles) {

double bonusMiles = state->travel(miles, this);

flyMiles = flyMiles + bonusMiles;

}

};

class CNoCustomer : public CState { // 非会员

public :

double travel(int miles, CFrequentFlyer\* context) { // 不累积里程数

cout << "Your travel will not account for points\n";

return miles;

}

};

class CBasic : public CState { // 普卡会员

public :

double travel(int miles, CFrequentFlyer\* context) {

if (context->flyMiles >= 25000 && context->flyMiles < 50000)

(2) ;

if (context->flyMiles >= 50000) (3) ;

return miles + 0.5 \* miles; // 累积里程数

}

};

class CGold : public CState { // 金卡会员

public :

double travel(int miles, CFrequentFlyer\* context) {

if (context->flyMiles >= 25000 && context->flyMiles < 50000)

(4) ;

if (context->flyMiles < 25000) (5) ;

return miles + 0.5 \* miles; // 累积里程数

}

};

class CSilver : public CState { // 银卡会员

public :

double travel(int miles, CFrequentFlyer\* context) {

if (context->flyMiles < 25000)

context->setState(context->basic);

if (context->flyMiles >= 50000)

context->setState(context->gold);

return (miles + 0.25 \* miles);

}

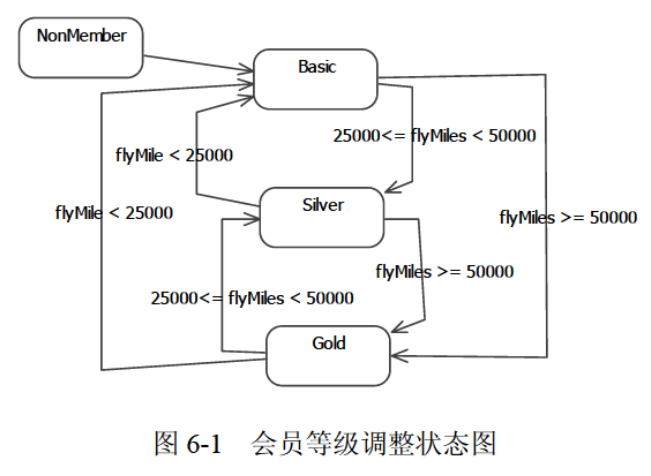
};

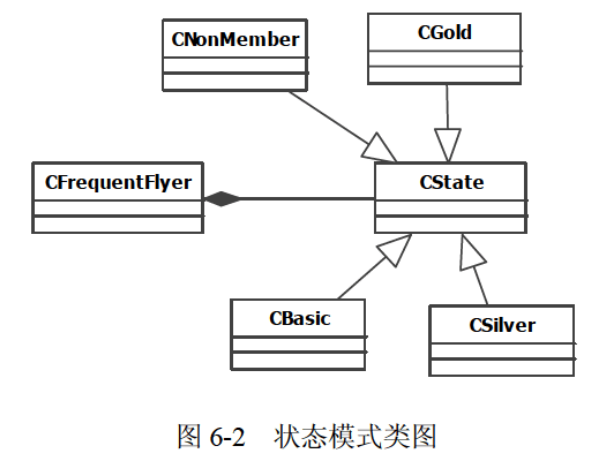
**试题六（共15分）**

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】

某航空公司的会员积分系统将其会员划分为：普卡(Basic)、银卡(Silver)和金卡(Gold)三个等级。非会员(NonMember)可以申请成为普卡会员。会员的等级根据其一年内累积的里程数进行调整。描述会员等级调整的状态图如图6-1所示。现采用状态(State)模式实现上述场景，得到如图6-2所示的类图。





【Java代码】

import java.util.\*;

abstract class CState {

public int flyMiles; // 里程数

public (1) ; // 根据累积里程数调整会员等级

}

class CNoCustomer extends CState { // 非会员

public double travel(int miles, CFrequentFlyer context) {

System.out.println("Your travel will not account for points");

return miles; // 不累积里程数

}

}

class CBasic extends CState { // 普卡会员

public double travel(int miles, CFrequentFlyer context) {

if (context.flyMiles >= 25000 && context.flyMiles < 50000)

(2) ;

if (context.flyMiles >= 50000)

(3) ;

return miles;

}

}

class CGold extends CState { // 金卡会员

public double travel(int miles, CFrequentFlyer context) {

if (context.flyMiles >= 25000 && context.flyMiles < 50000)

(4) ;

if (context.flyMiles < 25000);

(5) ;

return miles + 0.5 \* miles; // 累积里程数

}

}

class CSilver extends CState { // 银卡会员

public double travel(int miles, CFrequentFlyer context) {

if (context.flyMiles <= 25000)

context.setState(new CBasic());

if (context.flyMiles >= 50000)

context.setState(new CGold());

return (miles + 0.25 \* miles); // 累积里程数

}

}

class CFrequentFlyer {

CState state;

double flyMiles;

public CFrequentFlyer() {

state = new CNoCustomer();

flyMiles = 0;

setState(state);

}

public void setState(CState state) { this.state = state; }

public void travel(int miles) {

double bonusMiles = state.travel(miles, this);

flyMiles = flyMiles + bonusMiles;

}

}

**全国计算机技术与软件专业技术资格（水平）考试**

# 2019年上半年 软件设计师 下午试卷

（考试时间14:00～16:30共150分钟）

|  |
| --- |
| **请按下述要求正确填写答题卡** |

1. 在答题纸的指定位置填写你所在的省、自治区、直辖市、计划单列市的名称。

2. 在答题纸的指定位置填写准考证号、出生年月日和姓名。

3. 答题纸上除填写上述内容外只能写解答。

4. 本试卷共6道题，试题一至试题四是必答题，试题五至试题六选答1道。每题15分，满分75分。

5. 解答时字迹务必清楚，字迹不清时，将不评分。

6. 仿照下面的例题，将解答写在答题纸的对应栏内。

例题

2019 年上半年全国计算机技术与软件专业技术资格（水平）考试日期是（1）月（2）日。

因为正确的解答是“5月25日”，故在答题纸的对应栏内写上“5”和“25”（参看下表）。

|  |  |
| --- | --- |
| 例题 | 解答栏 |
| （1） | 5 |
| （2） | 25 |

**试题一（共15分）**

阅读下列说明和数据流图，回答问题1至问题4，将解答填入答题纸的对应栏内。

【说明】

某学校欲开发一学生跟踪系统，以便更自动化、更全面地对学生在校情况（到课情况和健康状态等）进行管理和追踪，使家长能及时了解子女的到课情况和健康状态，并在有健康问题时及时与医护机构对接。该系统的主要功能是：

（1）采集学生状态。通过学生卡传感器，采集学生心率、体温（摄氏度）等健康指标及其所在位置等信息并记录。每张学生卡有唯一的标识（ID）与一个学生对应。

（2）健康状态告警。在学生健康状态出问题时，系统向班主任、家长和医护机构健康服务系统发出健康状态警告，由医护机构健康服务系统通知相关医生进行处理。

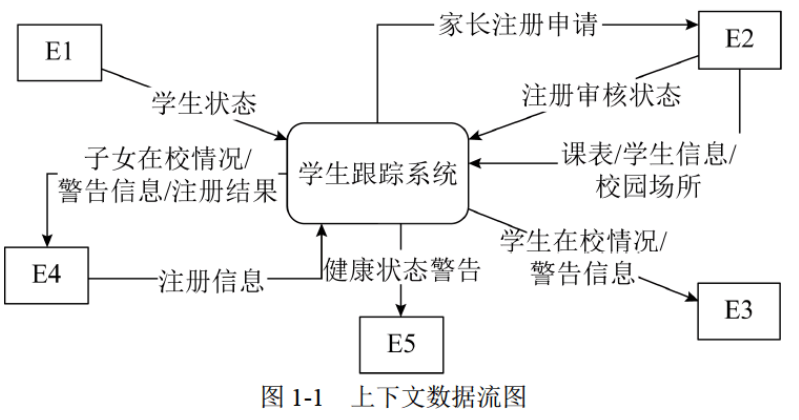
（3）到课检查。综合比对学生状态、课表以及所处校园场所之间的信息对学生到课情况进行判定。对旷课学生，向其家长和班主任发送旷课警告。

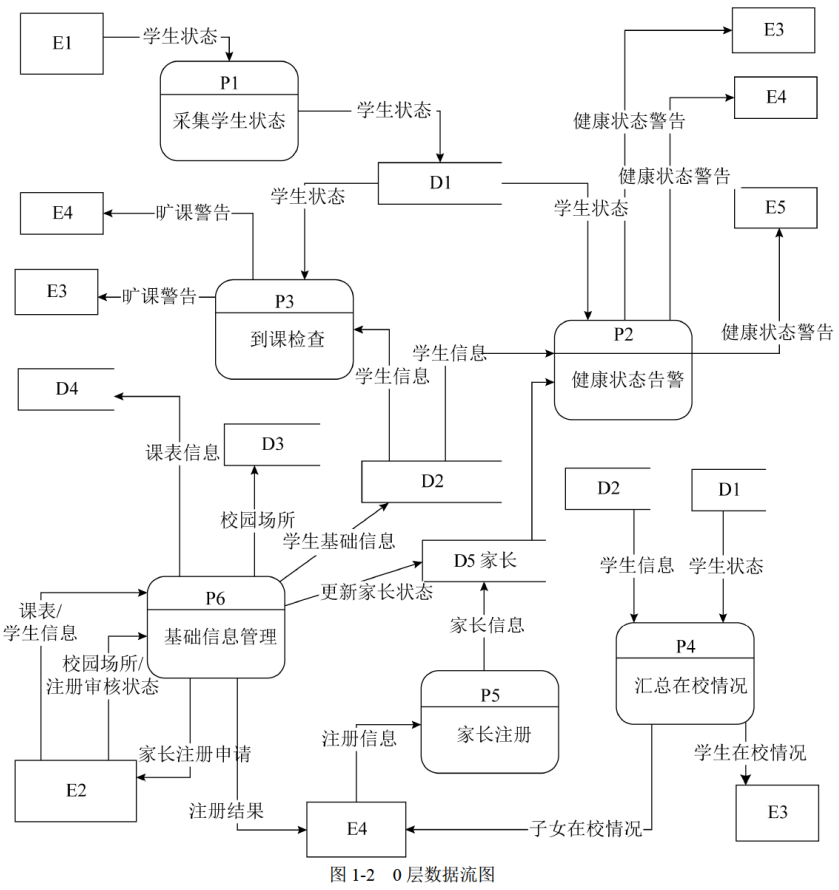
（4）汇总在校情况。定期汇总在校情况，并将报告发送给家长和班主任。

（5）家长注册。家长注册使用该系统，指定自己子女，存入家长信息，待审核。

（6）基础信息管理。学校管理人员对学生及其所用学生卡和班主任、课表（班级、上课时间及场所等）、校园场所（名称和所在位置区域）等基础信息进行管理；对家长注册申请进行审核，更新家长状态，将家长ID加入学生信息记录中使家长与其子女进行关联，向家长发送注册结果。一个学生至少有一个家长，可以有多个家长。课表信息包括班级、班主任、时间和位置等。

现采用结构化方法对学生跟踪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】（5分）

使用说明中的词语，给出图1-1中的实体E1〜E5的名称。

【问题2】（4分）

使用说明中的词语，给出图1-2中的数据存储D1〜D4的名称。

【问题3】（3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（三条即可）。

【问题4】（3分）

根据说明中的术语，说明图1-1中数据流“学生状态”和“学生信息”的组成。

**试题二（共15分）**

阅读下列说明，回答问题1至问题3，将解答填入答题纸的对应栏内。

【说明】

某创业孵化基地管理若干孵化公司和创业公司，为规范管理创业项目投资业务，需要开发一个信息系统。请根据下述需求描述完成该系统的数据库设计。

【需求分析结果】

（1）记录孵化公司和创业公司的信息。孵化公司信息包括公司代码、公司名称、法人代表名称、注册地址和一个电话；创业公司信息包括公司代码、公司名称和一个电话。孵化公司和创业公司的公司代码编码不同。

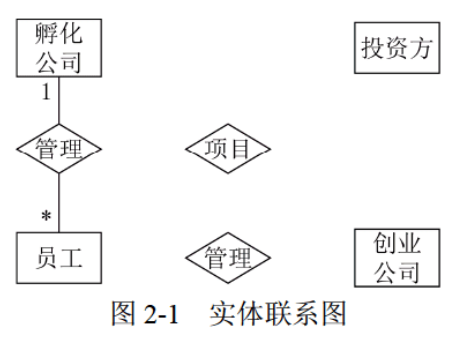
（2）统一管理孵化公司和创业公司的员工。员工信息包括工号、身份证号、姓名、性别、所属公司代码和一个手机号，工号唯一标识每位员工。

（3）记录投资方信息。投资方信息包括投资方编号、投资方名称和一个电话。

（4）投资方和创业公司之间依靠孵化公司牵线建立创业项目合作关系，具体实施由孵化公司的一位员工负责协调投资方和创业公司的一个创业项目。一个创业项目只属于一个创业公司，但可以接受若干投资方的投资。创业项目信息包括项目编号、创业公司代码、投资方编号和孵化公司员工工号。

【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。



【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）：

孵化公司（公司代码，公司名称，法人代表名称，注册地址，电话）

创业公司（公司代码，公司名称，电话）

员工（工号，身份证号，姓名，性别， （a） ，手机号）

投资方（投资方编号、投资方名称，电话）

项目（项目编号，创业公司代码， （b） ，孵化公司员工工号）

【问题1】（5分）

根据问题描述，补充图2-1的实体联系图。

【问题2】（4分）

补充逻辑结构设计结果中的（a）、（b）两处空缺及完整性约束关系。

【问题3】（6分）

若创业项目的信息还需要包括投资额和投资时间，那么：

（1）是否需要增加新的实体来存储投资额和投资时间？

（2）如果增加新的实体，请给出新实体的关系模式，并对图2-1进行补充。如果不需要增加新的实体，请将“投资额”和“投资时间”两个属性补充连线到图2-1合适的对象上，并对变化的关系模式进行修改。

**试题三（共15分）**

阅读下列说明和UML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】

某图书公司欲开发一个基于Web的书籍销售系统，为顾客（Customer）提供在线购买书籍（Books）的功能，同时对公司书籍的库存及销售情况进行管理。系统的主要功能描述如下：

（1）首次使用系统时，顾客需要在系统中注册（Register detail）。顾客填写注册信息表要求的信息，包括姓名（name）、收货地址（address）、电子邮箱（email）等，系统将为其生成一个注册码。

（2）注册成功的顾客可以登录系统在线购买书籍（Buy books）。购买时可以浏览书籍信息，包括书名（title）、作者（author）、内容简介（introduction）等。如果某种书籍的库存量为0，那么顾客无法查询到该书籍的信息。顾客选择所需购买的书籍及购买数量（quantities），若购买数量超过库存量，提示库存不足；若购买数量小于库存量，系统将显示验证界面，要求顾客输入注册码。注册码验证正确后，自动生成订单（Order），否则，提示验证码错误。如果顾客需要，可以选择打印订单（Print order）。

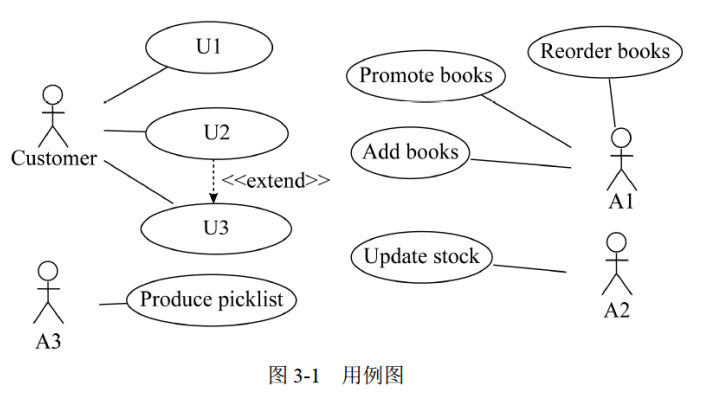
（3）派送人员（Dispatcher）每天早晨从系统中获取当日的派送列表信息（Produce picklist），按照收货地址派送顾客订购的书籍。

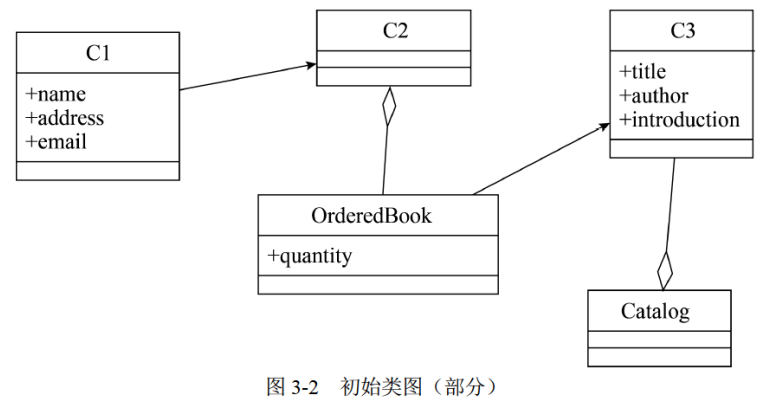
（4）用于销售的书籍由公司的采购人员（Buyer）进行采购（Reorder books）。采购人员每天从系统中获取库存量低于再次订购量的书籍信息，对这些书籍进行再次购买，以保证充足的库存量。新书籍到货时，采购人员向在线销售目录（Catalog）中添加新的书籍信息（Add books）。

（5）采购人员根据书籍的销售情况，对销量较低的书籍设置折扣或促销活动（Promote books）。

（6）当新书籍到货时，仓库管理员（Warehouseman）接收书籍，更新库存（Update stock）。

现采用面向对象方法开发书籍销售系统，得到如图3-1所示的用例图和图3-2所示的初始类图（部分）。





【问题1】（6分）

根据说明中的描述，给出图3-1中A1〜A3所对应的参与者名称和U1〜U3处所对应的用例名称。

【问题2】（6分）

根据说明中的描述，给出图3-1中用例U3的用例描述。（用例描述中必须包括基本事件流和所有的备选事件流）。

【问题3】（3分）

根据说明中的描述，给出图3-2中C1〜C3所对应的类名。

**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，将解答填入答题纸的对应栏内。

【说明】

皇后问题描述为：在一个的棋盘上摆放个皇后，要求任意两个皇后不能冲突，即任意两个皇后不在同一行、同一列或者同一斜线上。

算法的基本思想如下：

将第个皇后摆放在第行，从1开始，每个皇后都从第1列开始尝试。尝试时判断在该列摆放皇后是否与前面的皇后有冲突，如果没有冲突，则在该列摆放皇后，并考虑摆放下一个皇后；如果有冲突，则考虑下一列。如果该行没有合适的位置，回溯到上一个皇后，考虑在原来位置的下一个位置上继续尝试摆放皇后……直到找到所有合理摆放方案。

【C代码】

下面是算法的C语言实现。

（1）常量和变量说明

n：皇后数，棋盘规模为n×n

queen[]：皇后的摆放位置数组，queen[i]表示第i个皇后的位置，

（2）C程序

#include <stdio.h>

#define n 4

int queen[n + 1];

void Show() { /\* 输出所有皇后摆放方案 \*/

int i;

printf("(");

for (i = 1; i <= n; i ++ ) {

printf(" %d", queen[i]);

}

printf(")\n");

}

int Place(int j) { /\* 检查当前列能否放置皇后，不能放返回0，能放返回1 \*/

int i;

for (i = 1; i < j; i ++ ) { /\*检查与已摆放的皇后是否在同一列或者同一斜线上\*/

if ( (1) || abs(queen[i] - queen[j]) == (j - i)) {

return 0;

}

}

return (2) ;

}

void Nqueen(int j) {

int i;

for (i = 1; i <= n; i ++ ) {

queen[j] = i;

if ( (3) ) {

if (j == n) { /\* 如果所有皇后都摆放好，则输出当前摆放方案 \*/

Show();

} else { /\* 否则继续摆放下一个皇后 \*/

(4) ;

}

}

}

}

int main() {

Nqueen(1);

return 0;

}

【问题1】（8分）

根据题干说明，填充C代码中的空（1）~（4）。

【问题2】（3分）

根据题干说明和C代码，算法采用的设计策略为 （5） 。

【问题3】（4分）

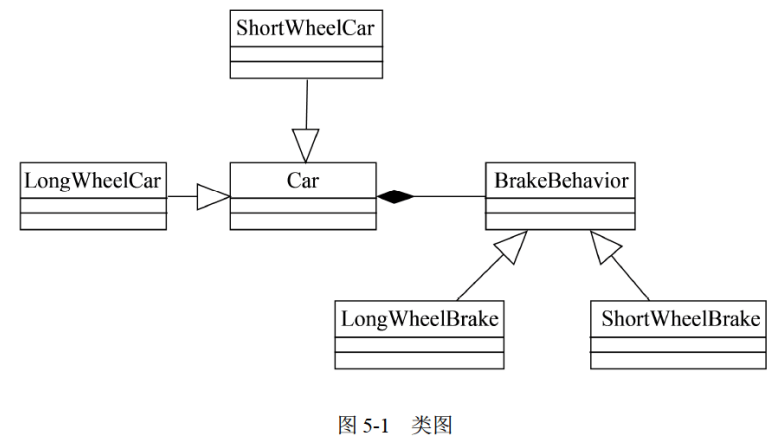
当n = 4时，有 （6） 种摆放方式，分别为 （7） 。

**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】

某软件公司欲开发一款汽车竞速类游戏，需要模拟长轮胎和短轮胎急刹车时在路面上留下的不同痕迹，并考虑后续能模拟更多种轮胎急刹车时的痕迹。现采用策略（Strategy）设计模式来实现该需求，所设计的类图如图5-1所示。



【C++代码】

#include <iostream>

using namespace std;

class BrakeBehavior {

public :

(1) ;

/\* 其余代码省略 \*/

};

class LongWheelBrake : public BrakeBehavior {

public :

void stop() { cout << "模拟长轮胎刹车痕迹！" << endl; }

/\* 其余代码省略 \*/

};

class ShortWheelBrake : public BrakeBehavior {

public :

void stop() { cout << "模拟短轮胎刹车痕迹！" << endl;

/\* 其余代码省略 \*/

};

class Car {

protected :

(2) wheel;

public :

void break() { (3) ; }

/\* 其余代码省略 \*/

};

class ShortWheelCar : public Car {

public :

ShortWheelCar(BrakeBehavior\* behavior) {

(4) ;

}

/\* 其余代码省略 \*/

};

int main() {

BrakeBehavior\* break = new ShortWheelBrake();

ShortWheelCar car1(brake);

car1. (5) ;

return 0;

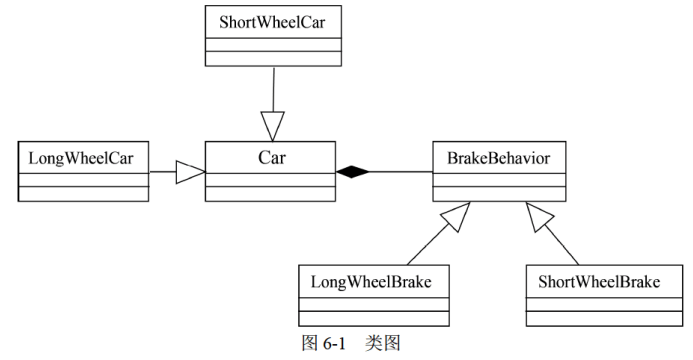
}

**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】

某软件公司欲开发一款汽车竞速类游戏，需要模拟长轮胎和短轮胎急刹车时在路面上留下的不同痕迹，并考虑后续能模拟更多种轮胎急刹车时的痕迹。现采用策略（Strategy）设计模式来实现该需求，所设计的类图如图6-1所示。



【Java代码】

import java.util.\*;

interface BrakeBehavior {

public (1) ;

/\* 其余代码省略 \*/

}

class LongWheelBrake implements BrakeBehavior {

public void stop() { System.out.println("模拟长轮胎刹车痕迹！"); }

/\* 其余代码省略 \*/

}

class ShortWheelBrake implements BrakeBehavior {

public void stop() { System.out.println("模拟短轮胎刹车痕迹！"); }

/\* 其余代码省略 \*/

}

abstract class Car {

protected (2) wheel;

public void brake() { (3) ; }

/\* 其余代码省略 \*/

}

class ShortWheelCar extends Car {

public ShortWheelCar(BrakeBehavior behavior) {

(4) ;

}

/\* 其余代码省略 \*/

}

class StrategyTest {

public static void main(String[] args) {

BrakeBehavior brake = new ShortWheelBrake();

ShortWheelCar car1 = new ShortWheelCar(brake);

car1. (5) ;

}

}

**全国计算机技术与软件专业技术资格（水平）考试**

# 2019年下半年 软件设计师 下午试卷

（考试时间14:00～16:30共150分钟）

|  |
| --- |
| **请按下述要求正确填写答题卡** |

1. 在答题纸的指定位置填写你所在的省、自治区、直辖市、计划单列市的名称。

2. 在答题纸的指定位置填写准考证号、出生年月日和姓名。

3. 答题纸上除填写上述内容外只能写解答。

4. 本试卷共6道题，试题一至试题四是必答题，试题五至试题六选答1道。每题15分，满分75分。

5. 解答时字迹务必清楚，字迹不清时，将不评分。

6. 仿照下面的例题，将解答写在答题纸的对应栏内。

例题

2019 年下半年全国计算机技术与软件专业技术资格（水平）考试日期是（1）月（2）日。

因为正确的解答是“11月9日”，故在答题纸的对应栏内写上“11”和“9”（参看下表）。

|  |  |
| --- | --- |
| 例题 | 解答栏 |
| （1） | 11 |
| （2） | 9 |

**试题一（共15分）**

阅读下列说明和数据流图，回答问题1至问题4，将解答填入答题纸的对应栏内。

【说明】

某公司欲开发一款二手车物流系统，以有效提升物流成交效率。该系统的主要功能是：

（1）订单管理：系统抓取线索，将车辆交易系统的交易信息抓取为线索。帮买顾问看到有买车线索后，会打电话询问买家是否需要物流，若需要，帮买顾问就将这个线索发起为订单并在系统中存储，然后系统帮助买家寻找物流商进行承运。

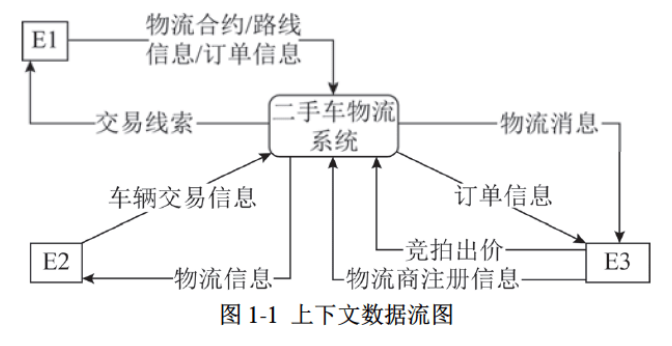
（2）路线管理：帮买顾问对物流商的路线进行管理，存储的路线信息包括路线类型、物流商、起止地点。路线分为三种，即固定路线、包车路线、竞拍体系，其中固定路线和包车路线是合约制。包车路线的发车时间由公司自行管理，是订单的首选途径。

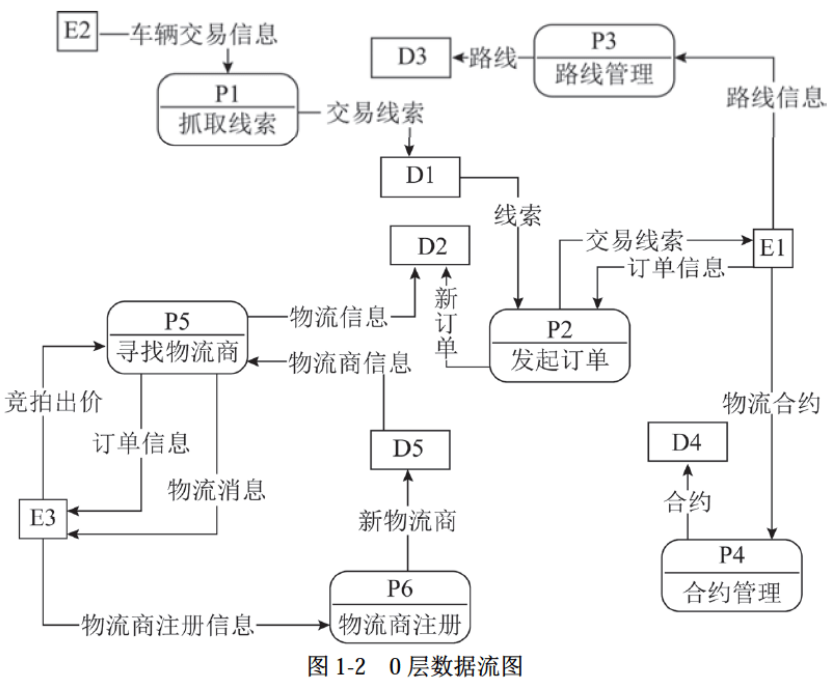
（3）合约管理：帮买顾问根据公司与物流商确定的合约，对合约内容进行设置，合约信息包括物流商信息、路线起止城市、价格、有效期等。

（4）寻找物流商：系统根据订单的类型（保卖车、全国购和普通二手车）、起止城市、需要的服务模式（买家接、送到买家等）进行自动派发或以竞拍体系方式选择合适的物流商。即:有新订单时，若为保卖车或全国购，则直接分配到竞拍体系中；否则，若符合固定路线和/或包车路线，系统自动分配给合约物流商，若不符合固定路线和包车路线，系统将订单信息分配到竞拍体系中。竞拍体系接收到订单后，将订单信息推送给有相关路线的物流商，物流商对订单进行竞拍出价，最优报价的物流商中标。最后，给承运的物流商发送物流消息，更新订单的物流信息，给车辆交易系统发送物流信息。

（5）物流商注册：物流商账号的注册开通。

现采用结构化方法对二手车物流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】（3分）

使用说明中的词语，给出图1-1中的实体E1〜E3的名称。

【问题2】（5分）

使用说明中的词语，给出图1-2中的数据存储D1〜D5的名称。

【问题3】（4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）

根据说明，采用结构化语言对“P5：寻找物流商”的加工逻辑进行描述。

**试题二（共15分）**

阅读下列说明，回答问题1至问题4，将解答填入答题纸的对应栏内。

【说明】

某公司拟开发一套新入职员工的技能培训管理系统，以便使新员工快速胜任新岗位。该系统的部分功能及初步需求分析的结果如下所述：

【需求分析结果】

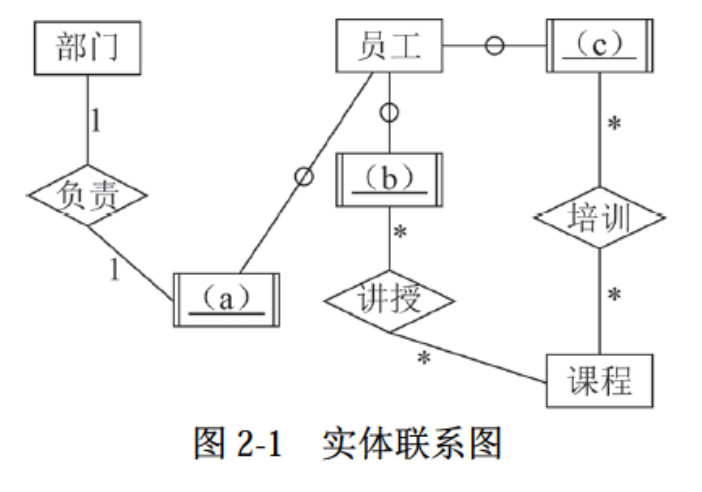
（1）部门信息包括部门号、名称、部门负责人、电话等，部门号唯一标识部门关系中的每一个元组，一个部门有多个员工，但一名员工只属于一个部门；每个部门只有一名负责人，负责部门工作。

（2）员工信息包括员工号、姓名、部门号、岗位、基本工资、电话、家庭住址等，其中员工号是唯一标示员工关系中的每一个元组；岗位有新入职员工、培训师、部门负责人等；不同的岗位设置不同的基本工资。新入职员工要选择多门课程进行培训，并通过考试取得课程的成绩。一名培训师可以讲授多门课程，一门课程可以由多名培训师讲授。

（3）课程信息包括课程号、课程名称、学时等，其中课程号唯一标识课程关系的每一个元组。

【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。



【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（部门号，部门名，部门负责人，电话）

员工（员工号，姓名，部门号， （d） ，电话，家庭住址）

课程（ （e） ，课程名称，学时）

讲授（课程号，培训师，培训地点）

培训（课程号， （f） ）

【问题1】（5分）

（1）补充图2-1中的空（a）〜（c）。

（2）图2-1中是否存在缺失联系，若存在，则说明所缺失的联系和联系类型。

【问题2】（3分）

根据题意，将关系模式中的空（d）〜（f）补充完整。

【问题3】（5分）

（1）员工关系模式的主键为 （g） ，外键为 （h） ；

（2）讲授关系模式的主键为 （i） ，外键为 （j） 。

【问题4】（2分）

员工关系是否存在传递依赖？用100字以内的文字说明理由。

**试题三（共15分）**

阅读下列说明和UML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】

某牙科诊所拟开发一套信息系统，用于管理病人的基本信息和就诊信息。诊所工作人员包括：医护人员（DentalStaff）、接待员（Receptionist）和办公人员（OfficeStaff）等。系统主要功能需求描述如下：

1．记录病人基本信息（Maintain patient info）。初次就诊的病人，由接待员将病人基本信息录入系统。病人基本信息包括病人姓名、身份证号、出生日期、性别、首次就诊时间和最后一次就诊时间等。每位病人与其医保信息（MedicalInsurance）关联。

2．记录就诊信息（Record office visit info）。病人在诊所的每一次就诊，由接待员将就诊信息（Office Visit）录入系统。就诊信息包括就诊时间、就诊费用、支付代码、病人支付费用和医保支付费用等。

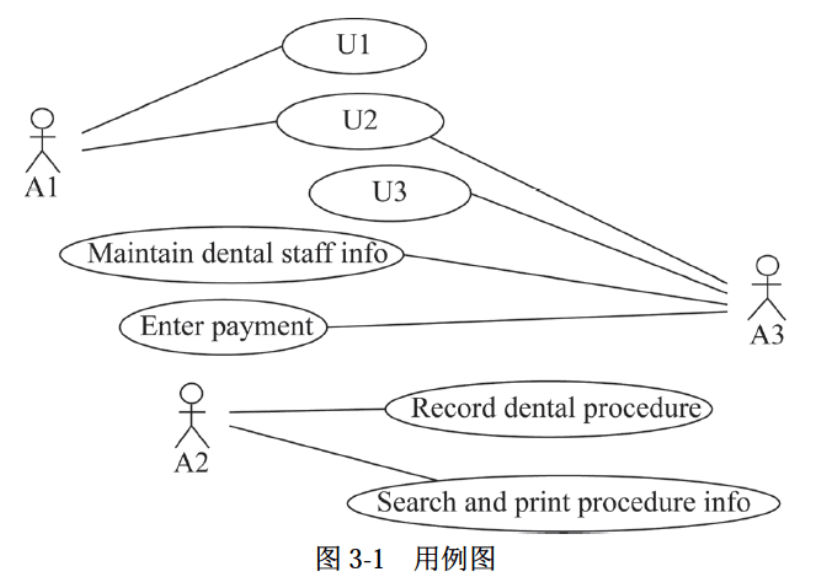
3．记录治疗信息（Record dental procedure）。病人在就诊时，可能需要接受多项治疗，每项治疗（Procedure）可能由多位医护人员为其服务。治疗信息包括：治疗项目名称、治疗项目描述、治疗的牙齿和费用等。治疗信息由每位参与治疗的医护人员分别向系统中录入。

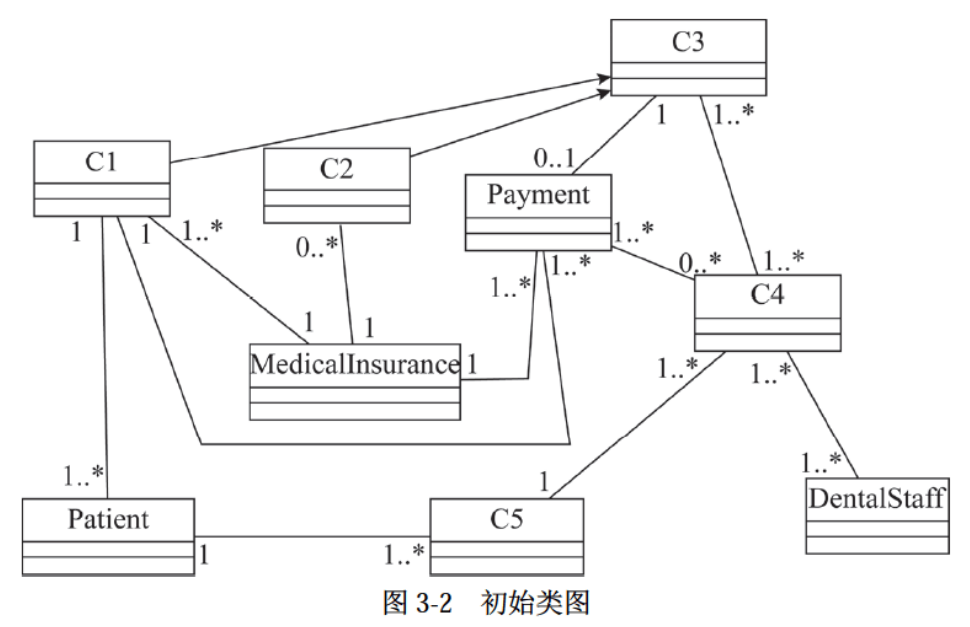
4．打印发票（Print invoices）。发票（Invoice）由办公人员打印。发票分为两种：给医保机构的发票（InsuranceInvoice）和给病人的发票（PatientInvoice）。两种发票内容相同，只是支付的费用不同。当收到治疗费用后，办公人员在系统中更新支付状态（Enterpayment）。

5．记录医护人员信息（Maintain dental staff info）。办公人员将医护人员信息录入系统。医护人员信息包括姓名、职位、身份证号、家庭住址和联系电话等。

6.医护人员可以查询并打印其参与的治疗项目相关信息（Search and print procedure info）。

现采用面向对象方法开发该系统，得到如图3-1所示的用例图和3-2所示的初始类图。





【问题1】（6分）

根据说明中的描述，给出图3-1中A1〜A3所对应的参与者名称和U1〜U3所对应的用例名称。

【问题2】（5分）

根据说明中的描述，给出图3-2中C1〜C5所对应的类名。

【问题3】（4分）

根据说明中的描述，给出图3-2中类C4、C5、Patient和DentalStaff的必要属性。

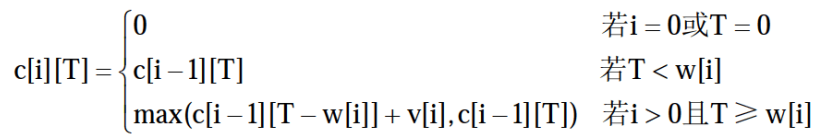
**试题四（共15分）**

阅读下列说明和C代码，回答问题1至问题3，将解答填入答题纸的对应栏内。

【说明】

0-1背包问题定义为：给定i个物品的价值v[1…i]、小重量w[1…i]和背包容量T，每个物品装到背包里或者不装到背包里。求最优的装包方案，使得所得到的价值最大。

0-1背包问题具有最优子结构性质。定义c[i][T]为最优装包方案所获得的最大价值，则可得到如下所示的递归式。



【C代码】

下面是算法的C语言实现。

（1）常量和变量说明

T：背包容量

v[]：价值数组

w[]：重量数组

c[][]：c[i][j]表示前i个物品在背包容量为j的情况下最优装包方案所能获得的最大价值

（2）C程序

#include <stdio.h>

#include <math.h>

#define N 6

#define maxT 1000

int c[N][maxT] = {0};

int Memoized\_Knapsack(int v[N], int w[N], int T) {

int i;

int j;

for (i = 0; i < N; i ++ ) {

for (j = 0; j <= T; j ++ ) {

c[i][j] = -1;

}

}

return Calculate\_Max\_Value(v, w, N - 1, T);

}

int Calculate\_Max\_Value(int v[N], int w[N], int i, int j) {

int temp = 0;

if (c[i][j] != -1) {

return (1) ;

}

if (i == 0 || j == 0) {

c[i][j] == 0;

} else {

c[i][j] = Calculate\_Max\_Value(v, w, i - 1, j);

if ( (2) ) {

temp = (3) ;

if (c[i][j] < temp) {

(4) ;

}

}

}

return c[i][j];

}

【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】（4分）

根据说明和C代码，算法采用了 （5） 设计策略。在求解过程中，采用了 （6）（自底向上或者自顶向下）的方式。

【问题3】（3分）

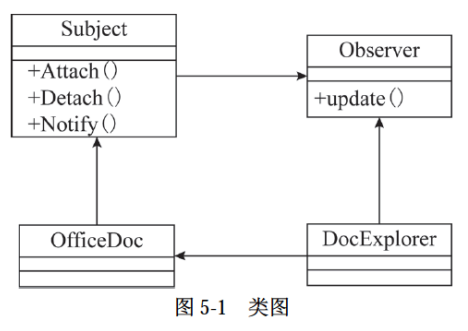
若5项物品的价值数组和重量数组分别为v[] = {0, 1, 6, 18, 22, 28}和w[] = {0, 1, 2, 5, 6, 7}背包容量为T = 11，则获得的最大价值为 （7） 。

**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】

某文件管理系统中定义了类OfficeDoc和DocExplorer。当类OfficeDoc发生变化时，类DocExplorer的所有对象都要更新其自身的状态。现采用观察者（Observer）设计模式来实现该需求，所设计的类图如图5-1所示。



【C++代码】

#include <iostream>

#include <vector>

#include <string>

using namespace std;

class Observer {

public :

(1) ;

};

class Subject {

protected :

vector< (2) > myObs;

public :

virtual void Attach(Observer \*obs) { myObs.push\_back(obs); }

virtual void Detach(Observer \*obs) {

for (vector<Observer\*>::iterator iter = myObs.begin();

iter != myObs.end(); iter ++ ) {

if (\*iter == obs) { myObs.erase(iter); return; }

}

}

virtual void Notify() {

for (vector<Observer\*>::iterator iter = myObs.begin();

iter != myObs.end(); iter ++ ) {

(3) ;

}

}

virtual int getStatus() = 0;

virtual void setStatus(int status) = 0;

};

class OfficeDoc : public Subject {

private :

string mySubjectName;

int m\_status;

public :

OfficeDoc(string name) : mySubjectName(name), m\_status(0) {}

void setStatus(int status) { m\_status = status; }

int getStatus() { return m\_status; }

};

class DocExplorer : public Observer {

private :

string myObsName;

public :

DocExplorer(string name, (4) sub) : myObsName(name) {

sub-> (5) ;

}

void update() { cout << "update observer:" << myObsName << endl; }

};

int main() {

Subject \*subjectA = new OfficeDoc("subject A");

Observer \*observerA = new DocExplorer("observerA", subjectA);

subjectA->setStatus(1); subjectA->Notify();

return 0;

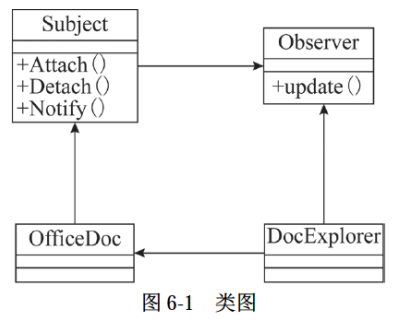
}

**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】

某文件管理系统中定义了类OfficeDoc和DocExplorer。当类OfficeDoc发生变化时，类DocExplorer的所有对象都要更新其自身的状态。现采用观察者（Observer）设计模式来实现该需求，所设计的类图如图6-1所示。



【Java代码】

import java.util.\*;

interface Observer {

public (1) ;

}

interface Subject {

public void Attach(Observer obs);

public void Detach(Observer obs);

public void Notify();

public void setStatus(int status);

public int getStatus();

}

class OfficeDoc implements Subject {

private List< (2) > myObs;

private String mySubjectName;

private int m\_status;

public OfficeDoc(String name) {

mySubjectName = name;

this.myObs = new ArrayList<Observer>();

m\_status = 0;

}

public void Attach(Observer obs) { this.myObs.add(obs); }

public void Detach(Observer obs) { this.myObs.remove(obs); }

public void Notify() {

for (Observer obs : this.myObs) { (3) ; }

}

public void setStatus(int status) {

m\_status = status;

System.out.println("SetStatus subject[" + mySubjectName + "]status:" + status);

}

public int getStatus() { return m\_status; }

}

class DocExplorer implements Observer {

private String myObsName;

public DocExplorer(String name, (4) sub) {

myObsName = name;

sub. (5) ;

}

public void update() {

System.out.println("update observer[" + myObsName + "]");

}

}

class ObserverTest {

public static void main(String[] args) {

Subject subjectA = new OfficeDoc("subject A");

Observer observerA = new DocExplorer("observer A", subjectA);

subjectA.setStatus(1);

subjectA.Notify();

}

}

**全国计算机技术与软件专业技术资格（水平）考试**

# 2020年下半年 软件设计师 下午试卷

（考试时间14:00～16:30共150分钟）

|  |
| --- |
| **请按下述要求正确填写答题卡** |

1. 在答题纸的指定位置填写你所在的省、自治区、直辖市、计划单列市的名称。

2. 在答题纸的指定位置填写准考证号、出生年月日和姓名。

3. 答题纸上除填写上述内容外只能写解答。

4. 本试卷共6题，试题一至试题四是必答题，试题五至试题六选答1道。每题15分，满分75分。

5. 解答时字迹务必清楚，字迹不清时，将不评分。

6. 仿照下面的例题，将解答写在答题纸的对应栏内。

例题

2020年下半年全国计算机技术与软件专业技术资格（水平）考试日期是（1）月（2）日。

因为正确的解答是“11月7日”，故在答题纸的对应栏内写上“11”和“7”（参看下表）。

|  |  |
| --- | --- |
| 例题 | 解答栏 |
| （1） | 11 |
| （2） | 7 |

**试题一（共15分）**

阅读下列说明和数据流图，回答问题1至问题4，将解答填入答题纸的对应栏内。

【说明】

某工厂制造企业开发了智能检测系统以有效提升检测效率，节约人力资源，该系统的主要功能包括：

（1）基础信息管理。管理员对检测标准和监控规则等基础信息设置。

（2）检测模型部署。管理员对常用机器学习方法建立检测模型分布。

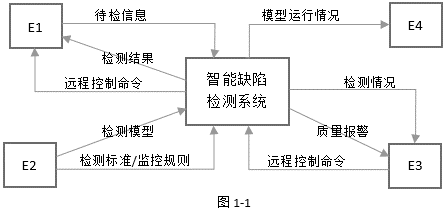
（3）图像采集。实时将检测多样的产品待检测建分存储，包括产品结构，生产时间，图像信号和产品图像。

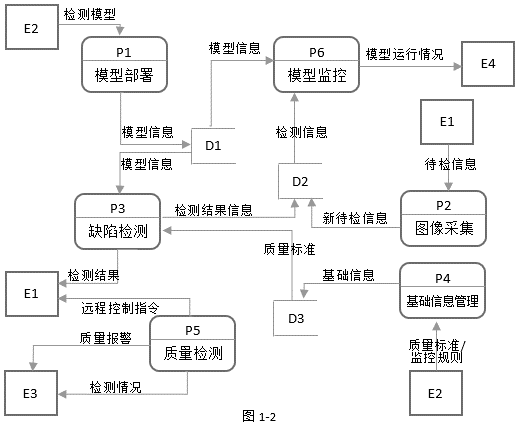
（4）缺陷检测。根据检测模型和检测质量标准对图像采集所收到的产品检测信息中所有图像进行检测或所有图像检测合格。若一个产品出现一张图像检测不合格，就表示该产品不合格，对不合格产品，其检测结果包括，产品型号和不合格类型。

（5）质量检测。根据监控规则对产品质量进行监控，将检测情况展示给检测业务员，若不满足条件，向检测业务员发送质量报警，检测是质量发起远程控制部分，向检测设备发送控制指令进行处理。

（6）模型监控。在系统中部署的模型、产品的检测信息结合基础信 息进行监测分析，将模型运行情况发给监控人员。

现采用结构化方法对智能检测系统，进行分析与设计，获得如图1-1的上下文数据流图和图1-2的数据流图。





【问题1】（4分）

使用说明中的语句给出图1-1中的实体E1〜E4的名称。

【问题2】（3分）

使用说明中的语句给出图1-2中的数据存储D1〜D3的名称。

【问题3】（5分）

根据说明和图中术语，补齐图1-2中缺失的数据及起点和终点。

【问题4】（3分）

根据说明，采用结构化语言对缺陷检测的加工逻辑进行描述。

**试题二（共15分）**

阅读下列说明，回答问题1至问题4，将解答填入答题纸的对应栏内。

【说明】

M集团拥有多个分公司，为了方便集团公司对各个分公司职员进行有效管理，集团公司决定构建一个信息平台以满足公司各项业 务管理需求。

【需求分析结果】

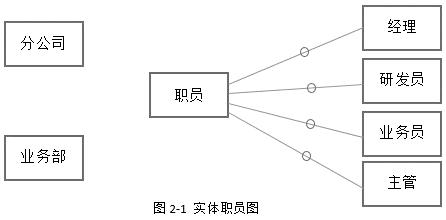
（1）分公司关系模式需要记录的信息包括分公司编号、名称、经理号、联系地址和电话。分公司编号唯一标记分公司关系模式中的每一个元组，每个分公司各有一名经理，负责分公司的管理工作，每个分公司设立仅为本分公司服务的多个业务部，业务部包括：研发部、财务部、采购部、交易部等。

（2）业务部关系模式需要记录的信息包括业务部编号、名称、地址、电话和分公司编号。业务部编号唯一标记业务部关系模式中的每 一个元组，每个业务部各有一名主管负责业务部的管理工作，每个业务部有多名职员，每个职员只能来源于一个业务部。

（3）职员关系模式需要记录的信息包括职员号、姓名、所属业务部编号、岗位、电话、家庭成员姓名和成员关系。其中岗位包括：经理、主管、研发员、业务员等。

【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。



【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）：

分公司（分公司编号，名称， （a） ，联系地址）

业务部（业务部编号，名称， （b） ，电话），

职员（职员号，姓名，岗位， （c） ，电话，家庭成员姓名，成员关系）

【问题1】（4分）

根据问题描述，补充4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和联系4代替，联系的类型为1:1、1:n和m:n（或1:1、1:\*和\*:\*）

【问题2】（3分）

根据题意将以上关系模式中的空（a）〜（c）的属性补充完整，并填入对应位置。

【问题3】（4分）

（1）分析分公司关系模式的主键和外键

（2）分析业务部关系模式的主键和外键

【问题4】（4分）

在职员关系模式中，假设每个职员有多名家属成员，那么职员关系模式存在什么问题？应如何解决？

**试题三（共15分）**

阅读下列说明和UML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】

某房产公司欲开发一个房产信息管理系统，其主要功能描述如下：

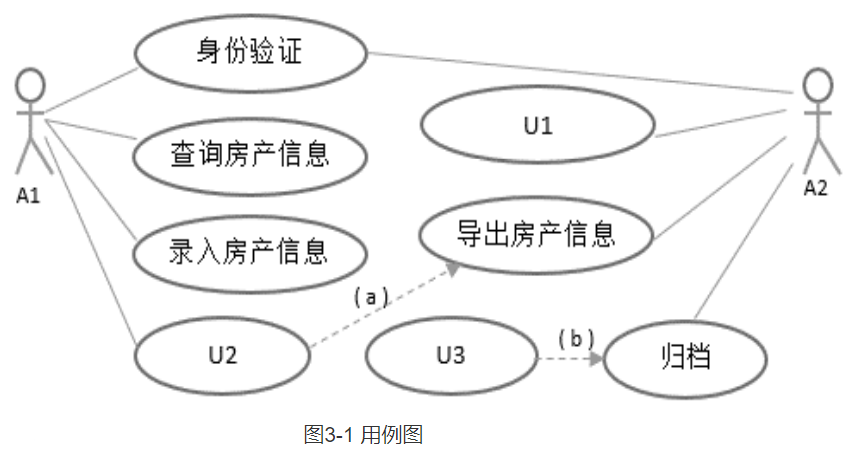
（1）公司销售的房产（Property）分为住宅（House）和公寓（Cando）两类。针对每套房产，系统存储房产证明、地址、建造年份、建筑面积，销售报价、房产照片以及销售状态（在售，售出，停售）等信息。对于住宅，还需存储楼层、公摊面积、是否有地下室等信息；对于公寓，还需存储是否有阳台等信息。

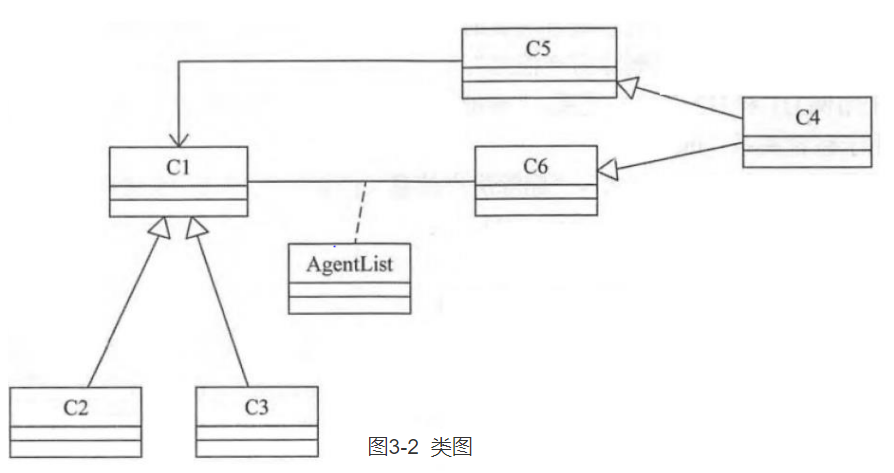
（2）公司雇佣了多名房产经纪（Agent），负责销售房产，系统中需要存储房产经纪的基本信息，包括：姓名、家庭住址、联系电话、受雇的起止时间等。一套房产同一时段仅由一名房产经纪负责销售，系统中会记录房产经纪负责每套房产的起始时间和终止时间。

（3）系统用户（User）包括房产经纪和系统管理员（Manager），用户需经过系统身份验证之后才能登录系统。房产经纪登录系统之后，可以录入负责销售的房产信息，也可以查询所负责的房产信息。房产经纪可以修改其负责的房产信息，但需要经过系统管理员的审批授权。

（4）系统管理员可以从系统中导出所有房产的信息列表，系统管理员定期将售出和停售的房产信息进行归档，若公司确定不再销售某套房产，系统管理员将该房产信息从系统中删除。

现采用面向对象方法开发该系统，得到如图3-1所示的用例图和图3-2所示的初始类图。





【问题1】（7分）

（1）根据说明中描述，分别给出图3-1中A1到A2所对应的参与者名称以及U1到U3所对应的用例名称。

（2）根据说明中描述，分别给图3-1中（a）和（b）用例之间的关系。

【问题2】（6分）

根据说明中描述，分别给图3-2中C1〜C6所对应的类名称。

【问题3】（2分）

图3-2中AgentList是一个英文名称，用来进一步阐述C1和C6之间的关系，根据说明中的描述，给出AgentList的主要属性。

**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，将解答填入答题纸的对应栏内。

【说明】

希尔排序算法又称最小增量排序算法，其基本思想是：

步骤1：构造一个步长序列delta1, delta2, ...., deltak, 其中delta1 = n / 2，后面的每个delta是前一个的1 / 2，deltak = 1；

步骤2：根据步长序列进行k趟排序；

步骤3：对第i趟排序，根据对应的步长delta，将等步长位置元素分，对同一组内元素在原位置上进行直接插入排序。

【C代码】

下面的算法用C语言实现

（1）常量和变量说明

Data：待排序数组data，长度为n，待排序数据在data[0], data[1], data[2]..., data[n - 1]中。

N：数组data中的元素个数

delta：步长数组

（2）C程序

#include <stdlib.h>

void ShellSort(int data[], int n) {

int\* delta, k, i, t, dk, j;

k = n;

delta = (int\*) malloc(sizeof(int) \* (n / 2));

i = 0;

do {

(1) ;

delta[i ++ ] = k;

} while (2) ;

i = 0;

while ( (3) ) {

for (k = delta[i]; k < n; ++ k )

if (data[k] < data[k - dk]) {

t = data[k];

for (j = k - dk; j >= 0 && t < data[j]; j -= dk)

data[j + dk] = data[j];

(4) ;

}

++ 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】（4分）

根据说明和C代码，该算法的时间复杂度 （5） （填写小于、等于或大于）。

该算法是否稳定 （6） （是或否）。

【问题3】（3分）

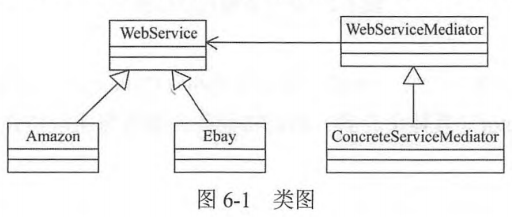
对数组(15, 9, 7, 8, 20, -1, 4)用希尔排序方法进行排序，经过第一趟排后得到的数组为 （7） 。

**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】

在线支付是电子商务的一个重要环节，不同的电子商务平台提供了不同的支付接口。现在需要整合不同电子商务平台的支付接口，使得客户在不同平台上购物时，不需要关心具体的支付接口。拟采用中介者（Mediator）设计模式来实现该需求，所设计的类图如图6-1所示。



【Java代码】

import java.util.\*;

interface WebServiceMediator {

public (1) ;

public void SetAmazon(WebService amazon);

public void SetEbay(WebService ebay);

}

abstract class WebService {

protected (2) mediator;

public abstract void SetMediator(WebServiceMediator mediator);

public (3) ;

public abstract void search(double money);

}

class ConcreteServiceMediator implements WebServiceMediator {

private WebService amazon;

private WebService ebay;

public ConcreteServiceMediator() {

amazon = null;

ebay = null;

}

public void SetAmazon(WebService amazon) {

this.amazon = amazon;

}

public void SetEbay(WebService ebay) {

this.ebay = ebay;

}

public void buy(double money, WebService service) {

if (service == amazon)

amazon.search(money);

else

ebay.search(money);

}

}

class Amazon extends WebService {

public void SetMediator(WebServiceMediator mediator) {

this.mediator = mediator;

}

public void buyService(double money) {

(4) ;

}

public void search(double money) {

System.out.println("Amazon receive：" + money);

}

}

class Ebay extends WebService {

public void SetMediator(WebServiceMediator mediator) {

this.mediator = mediator;

}

public void buyService(double money) {

(5) ;

}

public void search(double money) {

System.out.println("Ebay receive：" + money);

}

}

**全国计算机技术与软件专业技术资格（水平）考试**

# 2021年上半年 软件设计师 下午试卷

（考试时间14:00～16:30共150分钟）

|  |
| --- |
| **请按下述要求正确填写答题卡** |

1. 在答题纸的指定位置填写你所在的省、自治区、直辖市、计划单列市的名称。

2. 在答题纸的指定位置填写准考证号、出生年月日和姓名。

3. 答题纸上除填写上述内容外只能写解答。

4. 本试卷共6道题，试题一至试题四是必答题，试题五至试题六选答1道。每题15分，满分75分。

5. 解答时字迹务必清楚，字迹不清时，将不评分。

6. 仿照下面的例题，将解答写在答题纸的对应栏内。

例题

2021年上半年全国计算机技术与软件专业技术资格（水平）考试日期是（1）月（2）日。

因为正确的解答是“5月29日”，故在答题纸的对应栏内写上“5”和“29”（参看下表）。

|  |  |
| --- | --- |
| 例题 | 解答栏 |
| （1） | 5 |
| （2） | 29 |

**试题一（共15分）**

阅读下列说明和数据流图，回答问题1至问题4，将解答填入答题纸的对应栏内。

【说明】

某停车场运营方为了降低运营成本，减员增效，提供良好的停车体验，欲开发无人值守停车系统，该系统的主要功能是：

（1）信息维护。管理人员对车位（总数、空余车位数等）计费规则等基础信息进行设置。

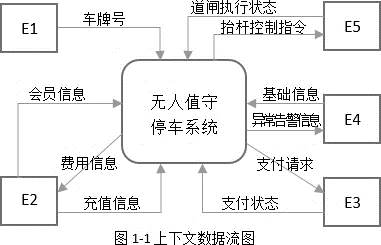
（2）会员注册。车主提供手机号、车牌号等信息进行注册，提交充值信息（等级、绑定并授权支付系统进行充值或交费的支付账号）不同级别和充值额度享受不同停车折扣点。

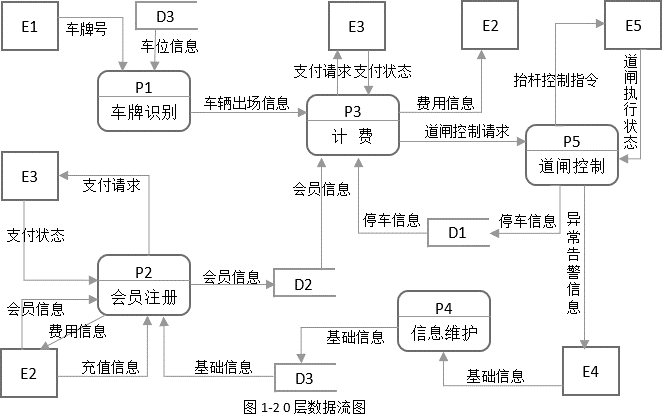
（3）车牌识别。当车辆进入停车场时，若有（空余车位数大于1），自动识别车牌号后进行道闸控制，当车主开车离开停车场时，识别车牌号，计费成功后，请求道闸控制。

（4）计费。更新车辆离场时间，根据计费规则计算出停车费用，若车主是会员，提示停车费用：若储存余额够本次停车费用，自动扣费，更新余额，若储值余额不足，自动使用授权缴费账号请求支付系统进行支付，获取支付状态。若非会员临时停车，提示停车费用，车主通过扫描费用信息中的支付码调用支付系统自助交费，获取支付状态。

（5）道闸控制。根据道闸控制请求向道闸控制系统发送放行指令和接收道闸执行状态。若道闸执行状态为正常放行时，对入场车辆，将车牌号及其入场时间信息存入停车记录，修改空余车位数；对出场车辆更新停车状态，修改空余车位数。当因道闸重置系统出现问题（断网断电或是故障为抬杠等情况），而无法在规定的时间内接收到其返回的执行状态正常放行时，系统向管理人员发送异常告警信息，之后管理人员安排故障排查处理，确保车辆有序出入停车场。

现采用结构化方法对无人值守停车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】（5分）

使用说明中的词语，给出图1-1中的实体E1〜E5的名称。

【问题2】（3分）

使用说明中的词语，给出图1-2中的数据存储D1〜D3的名称。

【问题3】（4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）

根据说明，采用结构化语言对“道闸控制”的加工逻辑进行描述。

**试题二（共15分）**

阅读下列说明，回答问题1至问题3，将解答填入答题纸的对应栏内。

【说明】

某社区蔬菜团购网站，为规范商品收发流程，便于查询客户订单情况，需要开发个信息系统。请根据下述需求描述完成该系统的数据库设计。

【需求分析结果】

（1）记录蔬菜供应商的信息，包括供应商编号、地址和一个电话。

（2）记录社区团购点的信息，包括团购点编号、地址和一个电话。

（3）记录客户信息，包括客户姓名和一个电话。客户可以在不同的社区团购点下订单，不直接与蔬菜供应商发生联系。

（4）记录客户订单信息，包括订单编号、团购点编号、客户电话、订单内容和日期。

【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。

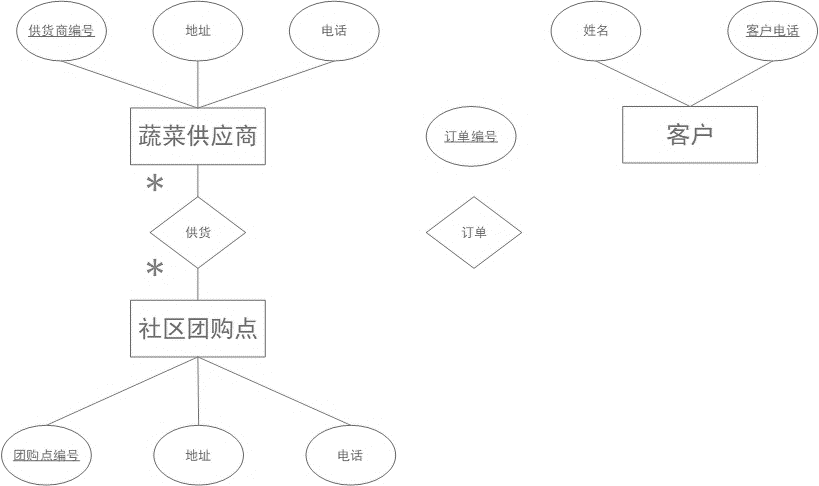


图2-1 实体联系图

【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）：

蔬菜供货商（供货商编号，地址，电话）

社区团购点（团购点编号，地址，电话）

供货（供货商编号， （a） ）

客户（姓名，客户电话）

订单（订单编号，团购点编号，订单内容，日期， （b） ）

【问题1】（6分）

根据问题描述，补充图2-1的实体联系图。

【问题2】（4分）

补充逻辑结构设计结果中的（a）、（b）两处空缺及完整性约束关系。

【问题3】（5分）

若社区蔬菜团购网站还兼有 代收快递的业务，请增加新的“快递”实体，并给出客户实体和快递实体之间的“收取’联系，对图2-1进行补充。“快递”关系模式包括快递编号、客户电话和日期。

**试题三（共15分）**

阅读下列说明和UML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】

某中医医院拟开发一套线上抓药APP，允许患者凭借该医院医生开具的处方线上抓药，并提供免费送药上门服务。该系统的主要功能描述如下：

（1）注册：患者扫描医院提供的二维码进行注册，注册过程中，患者需提供其病历号，系统根据病历号自动获取患者基本信息。

（2）登录：已注册的患者可以登录系统进行线上抓药，未册的患者系统拒绝其登陆。

（3）确认处方：患者登录后，可以查看医生开具的所有处方。患者选择需要抓药的处方和数量（需要抓几副药），同时说明是否需要煎制。选择取药方式：自行到店取药或者送药上门,若选择送药上门，患者需要提供提供收货人姓名、联系方式和收货地址。系统自动计算本次抓药的费用，患者可以使用微信或支付宝等支付方式支付费用。支付成功之后，处方被发送给药师进行药品配制。

（4）处理处方：药师根据处方配置好药品。若患者要求煎制，药师对配置好的药品进行煎制。煎制完成，药师将该处方设置为已完成。若患者选择的是自行取药，取药后确认已取药。

（5）药品派送：处方完成后，对于选择送药上门的患者，系统将给快递人员发送药品配送信息，等待快递人员取药；并给患者发送收获验证码。

（6）送药上门：快递人员将配制好的药品送到患者指定的收获地址。患者收货时，向快递人员出示收获验证码，快递人员使用该验证码确认药品已送到。

现采用面向对象分析与设计方法开发上述系统，得到如图3-1所示的用例图以及图3-2所示的类图。

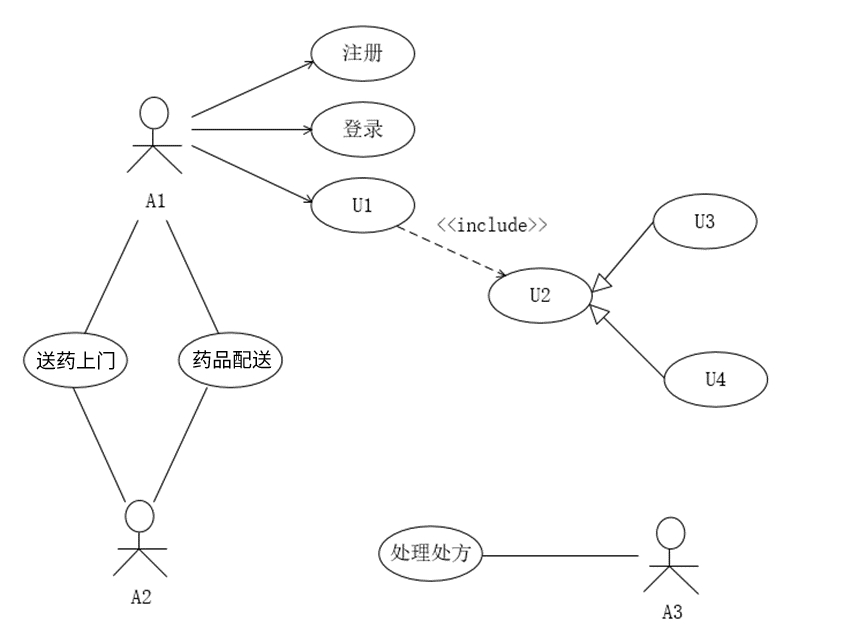


图3-1 用例图

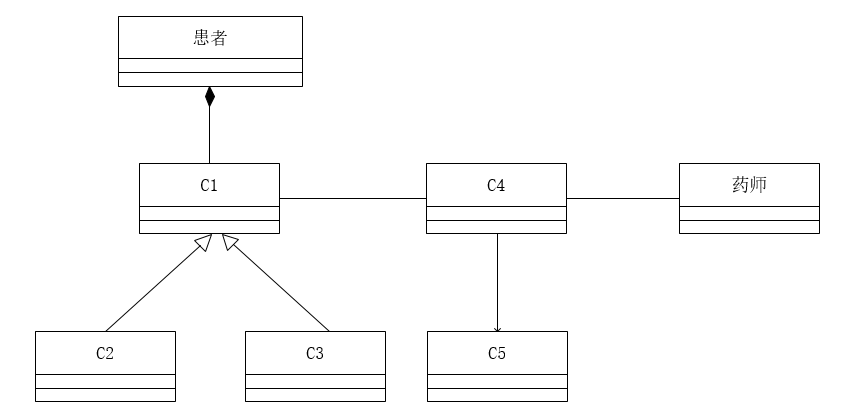


图3-2 类图

【问题1】（7分）

根据说明中的描述，给出图3-1中A1〜A3所对应的参与者名称和U1〜U4处所对应的用例名称。

【问题2】（5分）

根据说明中的描述，给出图3-2中C1〜C5所对应的类名。

【问题3】（3分）

简要解释用例之间的include、extend和generalize关系的内涵。

**试题四（共15分）**

阅读下列说明和C代码，回答问题1至问题2，将解答填入答题纸的对应栏内。

【说明】

凸多边形是指多边形的任意两点的连线均落在多边形的边界或内部。相邻的点连线落在多边形边界上，称为边；不相邻的点连线落在多边形内部，称为弦。假设任意两点连线上均有权重，凸多边形最优三角剖分问题定义为：求将凸多边形划分为不相交的三角形集合，且各三角形权重之和最小的剖分方案。每个三角形的权重为三条边权重之和。

假设N个点的凸多边形点编号为V1, V2, ...... , VN,若在Vk处将原凸多边形划分为一个三角形V1VkVN，两个子多边形V1, V2, ... , Vk和Vk, Vk+1, ... , VN得到一个最优的剖分方案，则该最优剖分方案应该包含这两个子凸边形的最优剖分方案。用m[i][j]表示Vi-1, Vi, ... ,Vj构成的凸多边形的最优剖分方案的权重，S[i][j]记录剖分该凸多边形的k值。

则：

其中：W(Vi-1VkVj) = Wi-1,k + Wk,j + Wj,i-1为三角形Vi-1VkVj的权重，Wi-1,k, Wk,j, Wj,i-1分别为该三角形三条边的权重。求解凸多边形的最优剖分方案，即求解最小剖分的权重及对应的三角形集。

【C代码】

#include<stdio.h>

#define N 6 // 凸多边形规模

int m[N + 1][N + 1]; // m[i][j]表示多边形Vi-1到Vj最优三角剖分的权值

int S[N + 1][N + 1]; // S[i][j]记录多边形Vi-1到Vj最优三角剖分的k值

int W[N + 1][N + 1]; // 凸多边形的权重矩阵，在main函数中输入

/\* 三角形的权重a, b, c, 三角形的顶点下标 \*/

int get\_ triangle\_weight(int a, int b, int c) {

return W[a][b] + W[b][c] + W[c][a];

}

/\* 求解最优值 \*/

void triangle\_partition() {

int i, r, k, j;

int temp;

/\* 初始化 \*/

for (i = 1; i <= N; i ++ ) {

m[i][i] = 0;

}

/\* 自底向上计算m, S \*/

for (r = 2; (1) ; r ++ ) { /\* r为子问题规模 \*/

for (i = 1; k <= N – r + 1; i ++ ) {

(2) ;

m[i][j] = m[i][j] + m[i + 1][j] +

get\_triangle\_weight(i - 1, i, j); /\* k = j \*/

S[i][j] = i;

for (k = j + 1; k < j; k ++ ) { /\*计算m[i][j]的最小代价 \*/

temp = m[i][k] + m[k + 1][j] +

ge\_triangle\_ weight(i - 1, k, j);

if ( (3) ) { /\* 判断是否最小值 \*/

m[i][j] = temp;

S[i][j] = k;

}

}

}

}

}

/\* 输出剖分的三角形i, j：凸多边形的起始点下标 \*/

void print\_triangle(int i, nt j) {

if (i == j) return;

print\_triangle(i, S[i][j]);

print\_triangle( (4) );

printf("V%d--V%d--V%d\n", i - 1, S[i][j], j);

}

【问题1】（8分）

根据题干说明，填充C代码中的空（1）〜（4）。

【问题2】（7分）

根据题干说明和C代码，该算法采用的设计策略为 （5） 。

算法的时间复杂度为 （6） ，空间复杂度为 （7） （用O表示）

**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】

层叠菜单是窗口风格的软件系统中经常采用的一种系统功能组织方式。层叠菜单（如图6-1示例）中包含的可能是一个菜单项（直接对应某个功能），也可能是一个子菜单，现在采用组合（composite）设计模式实现层叠菜单，得到如图6-2所示的类图

层叠菜单（如图6-1示例）暂缺

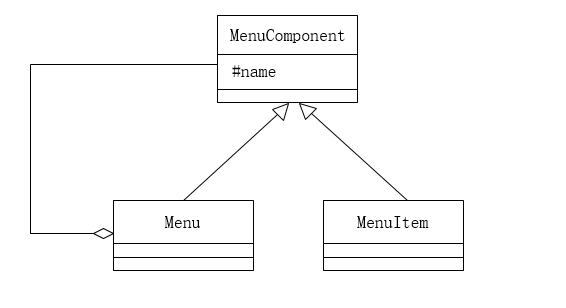


图6-2 类图

【Java代码】

import java.util.\*;

abstract class MenuComponent { // 构成层叠菜单的元素

(1) String name; // 菜单项或子菜单名称

public void printName() { System.out.println(name); }

public (2) ;

public abstract boolean removeMenuElement(MenuComponent element);

public (3) ;

}

class MenuItem extends MenuComponent {

public MenuItem(String name) { this.name=name; }

public boolean addMenuElement(MenuComponent element) { return false; }

public boolean removeMenuElement(MenuComponent element) {

return false;

}

public List<MenuComponent> getElement(){ return null; }

}

class Menu extends MemuComponent {

private (4) ;

public Menu(String name) {

this.name = name;

this.elementList = new ArrayList<MenuComponent>;

}

public boolean addMenuElement(MenuComponent element) {

return elementList.add(element);

}

public boolean removeMenuElement(MenuComponent element) {

return elementList.remove(element);

}

public List<MenuComponent> getElement() { return elementList; }

}

class CompositeTest {

public static void main(String[] args) {

MenuComponent mainMenu = new Menu("Insert");

MenuComponent subMenu = new Menu("Chart");

MenuComponent element = new MenuItem("On This Sheet");

(5) ;

subMenu.addMenuElement(element);

printMenus(mainMenu);

}

private static void printMenus(MenuComponent ifile) {

ifile.printName();

List<MenuComponent> children = ifile.getElement();

if (children == null) return;

for(MenuComponent element; children) {

printMenus(element);

}

}

}

**全国计算机技术与软件专业技术资格（水平）考试**

# 2021年下半年 软件设计师 下午试卷

（考试时间14:00～16:30共150分钟）

|  |
| --- |
| **请按下述要求正确填写答题卡** |

1. 在答题纸的指定位置填写你所在的省、自治区、直辖市、计划单列市的名称。

2. 在答题纸的指定位置填写准考证号、出生年月日和姓名。

3. 答题纸上除填写上述内容外只能写解答。

4. 本试卷共6道题，试题一至试题四是必答题，试题五至试题六选答1道。每题15分，满分75分。

5. 解答时字迹务必清楚，字迹不清时，将不评分。

6. 仿照下面的例题，将解答写在答题纸的对应栏内。

例题

2021年下半年全国计算机技术与软件专业技术资格（水平）考试日期是（1）月（2）日。

因为正确的解答是“11月6日”，故在答题纸的对应栏内写上“11”和“6”（参看下表）。

|  |  |
| --- | --- |
| 例题 | 解答栏 |
| （1） | 11 |
| （2） | 6 |

**试题一（共15分）**

阅读下列说明和数据流图，回答问题1至问题4，将解答填入答题纸的对应栏内。

【说明】

某现代农业种植基地为进一步提升农作物种植过程的智能化，欲开发智慧农业平台，集管理和销售于一体，该平台的主要功能有:

（1）信息维护。农业专家对农作物、环境等监测数据的监控处理规则进行维护。

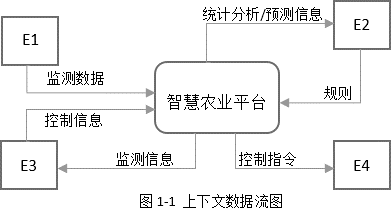
（2）数据采集。获取传感器上传的农作物长势、土壤墒情、气候等连续监测数据，解析后将监测信息进行数据处理、可视化和存储等操作。

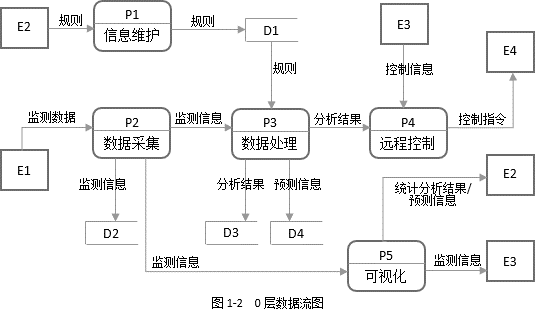
（3）数据处理。对实时监测信息根据监控处理规则进行监测分析，将分析结果进行可视化并进行存储、远程控制，对历史监测信息进行综合统计和预测，将预测信息进行可视化和存储。

（4）远程控制。根据监控处理规则对分析结果进行判定，依据判定结果自动对控制器进行远程控制。平台也可以根据农业人员提供的控制信息对控制器进行远程控制。

（5）可视化。实时向农业人员展示监测信息：实时给农业专家展示统计分析结果和预测信息或根据农业专家请求进行展示。

现采用结构化方法对智慧农业平台进行分析与设计，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】（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】（4分）

使用说明中的词语，给出图1-2中的数据存储D1〜D4的名称。

【问题3】（4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）

根据说明，“数据处理”可以分解为哪些子加工？进一步进行分解时，需要注意哪三种常见的错误？

**试题二（共15分）**

阅读下列说明，回答问题1至问题4，将解答填入答题纸的对应栏内。

【说明】

某汽车维修公司为了便于管理车辆的维修情况，拟开发一套汽车维修管理系统，请根据下述需求描述完成该系统的数据库设计。

【需求分析结果】

（1）客户信息包括：客户号、客户名、客户性质、折扣率、联系人、联系电话。客户性质有个人或单位。客户号唯一标识客户关系中的每一个元组。

（2）车辆信息包括：车牌号、车型、颜色和车辆类别。一个客户至少有一辆车，一辆车只属于一个客户。

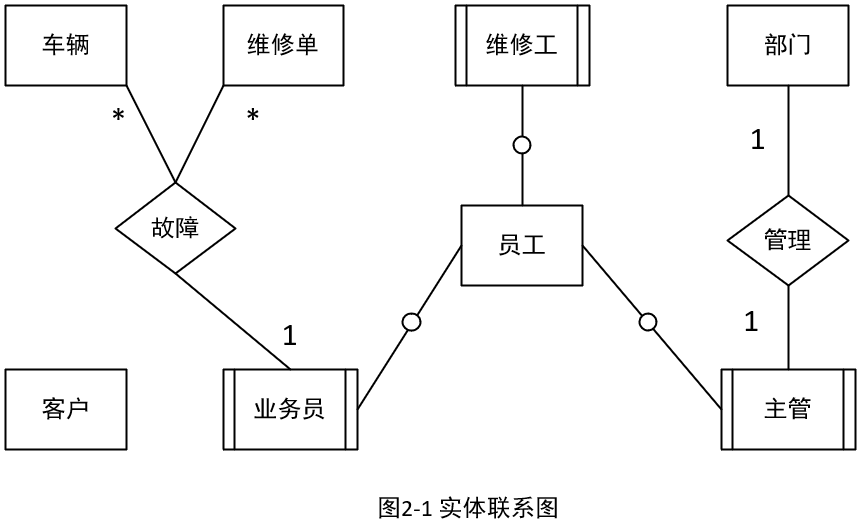
（3）员工信息包括：员工号、员工名、岗位、电话、家庭住址。其中，员工号唯一标识员工关系中的每一个元组。岗位有业务员、维修工、主管。业务员根据车辆的故障情况填写维修单。

（4）部门信息包括：部门号、名称、主管和电话。其中，部门号唯一确定部门关系的每一个元组。每个部门只有一名主管，但每个部门有多名员工，每名员工只属于一个部门。

（5）维修单信息包括：维修单号、车牌号、维修内容、工时。其中，维修单号唯一标识维修单关系中的每一个元组。一个维修工可以接多张维修单，但一张维修单只对应一个维修工。

【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。



【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）：

客户（客户号，客户名， （a） ，折扣率，联系人，联系电话）

车辆（车牌号， （b） ，车型，颜色，车辆类别）

员工（员工号，员工名，岗位， （c） ，电话，家庭住址）

部门（部门号，名称，主管，电话）

维修单（维修单号， （d） ，维修内容，工时）

【问题1】（6分）

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

【问题2】（4分）

根据题意，将关系模式中的空（a）〜（d）的属性补充完整，并填入答题纸对应的位置上。

【问题3】（2分）

分别给出车辆关系和维修单关系的主键和外键。

【问题4】（3分）

如果一张维修单涉及多项维修内容，需要多个维修工来处理，那么哪个联系类型会发生何种变化？你认为应该如何解决这一问题？

**试题三（共15分）**

阅读下列说明和UML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】

某游戏公司欲开发一款吃金币游戏。游戏的背景为一种回廊式迷宫（Maze），在迷宫的不同位置上设置有墙。迷宫中有两种类型的机器人（Robos）：小精灵（PacMan）和幽灵（Ghost）。游戏的目的就是控制小精灵在迷宫中游走，吞吃迷宫路径上的金币，且不能被幽灵抓到。幽灵在迷宫中游走，并会吃掉遇到的小精灵。机器人游走时，以单位距离的倍数计算游走路径的长度。当迷宫中至少存在一个小精灵和一个幽灵时，游戏开始。

机器人上有两种传感器，使机器人具有一定的感知能力。这两种传感器分别是：

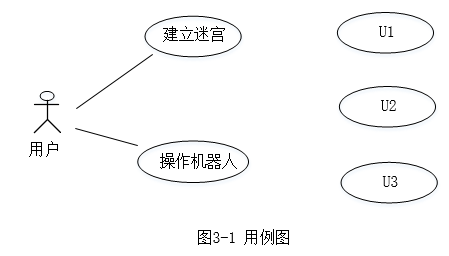
（1）前向传感器（FrontSensor）。探测在机器人当前位置的左边、右边和前方是否有墙（机器人遇到墙时，必须改变游走方向）。机器人根据前向传感器的探测结果，决定朝哪个方向运动。

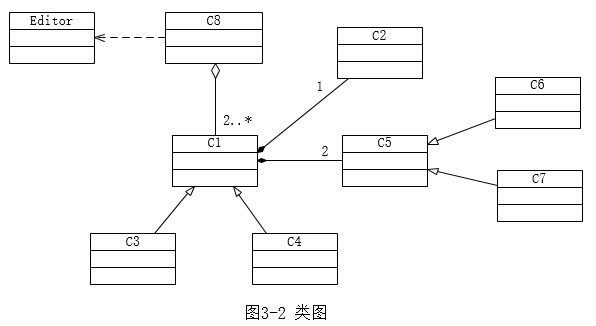
（2）近距离传感器（ProxiSensor）。探测在机器人的视线范围内（正前方）是否存在隐藏的金币或幽灵。近距离传感器并不报告探测到的对象是否正在移动以及朝哪个方向移动。但是如果近距离传感器的连续两次探测结果表明被探测对象处于不同的位置，则可以推到出该对象在移动。

另外，每个机器人都设置有一个计时器（Timer），用于支持执行预先定义好的定时事件。机器人的动作包括：原地向左或向右旋转90°、向前或向后移动。

建立迷宫：用户可以使用编辑器（Editor）编写迷宫文件，建立用户自定义的迷宫。将迷宫文件导入游戏系统建立用户自定义的迷宫。

现采用面向对象分析与设计方法开发该游戏，得到如图3-1所示的用例图以及图3-2所示的初始类图。





【问题1】（3分）

根据说明中的描述，给出图3-1中U1〜U3所对应的用例名。

【问题2】（4分）

图3-1中用例U1〜U3分别与哪个（哪些）用例之间有关系，是何种关系？

【问题3】（8分）

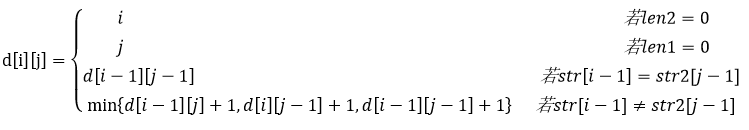
根据说明中的描述，给出图3-2中C1〜C8所对应的类名。

**试题四（共15分）**

阅读下列说明和C代码，回答问题1至问题2，将解答填入答题纸的对应栏内。

【说明】

生物学上通常采用编辑距离来定义两个物种DNA序列的相似性，从而刻画物种之间的进化关系。具体来说，编辑距离是指将一个字符串变换为另一个字符所需要的最小操作次数。操作有三种，分别为：插入一个字符、删除一个字符以及将一个字符修改为另一个字符。用字符数组str1和str2分别表示长度为len1和len2的字符串，定义二维数组d记录求解编辑距离的子问题最优解，则该二维数组可以递归定义为：



【C代码】

C代码

下面是算法的C语言实现。

（1）常量和变量说明

A，B：两个字符数组

d[][]：二维数组

i，j：循环变量

temp：临时变量

（2）C程序

#include <stdio.h>

#define N 100

char A[N] = "CTGA";

char B[N] = "ACGCTA";

int d[N][N];

int min(int a, int b) {

return a < b ? a : b;

}

int editdistance(char \*str1, int len1, char \*str2, int len2) {

int i, j;

int diff;

int temp;

for (i = 0; i <= len1; i ++ ) {

d[i][0] = i;

}

for (j = 0; j <= len2; j ++ ) {

(1) ;

}

for (i = 1; i <= len1; i ++ ) {

for (j = 1; j <= len2; j ++ ) {

if ( (2) ) {

d[i][j] = d[i - 1][j - 1];

} else {

temp = min(d[i - 1][j] + 1, d[i][j - 1] + 1);

d[i][j] = min(temp, (3) );

}

}

}

return (4) ;

}

【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】（4分）

根据说明和C代码，算法采用了 （5） 设计策略，时间复杂度为 （6） （用O符合表示，两个字符串的长度分别用m和n表示）。

【问题3】（3分）

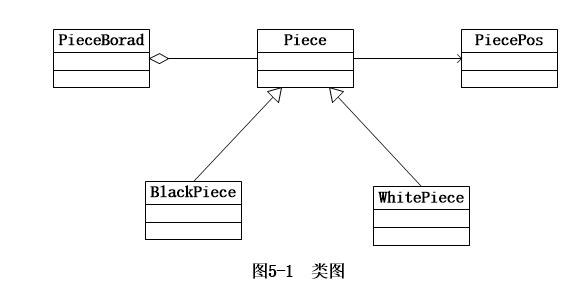
已知两个字符串A=“CTGA”和B=“ACGCTA”，根据说明和C代码，可得出这两个字符串的编辑距离为 （7） 。

**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】

享元（FlyWeight）模式主要用于减少创建对象的数量，以降低内存占用，提高性能。先要开发一个网络围棋程序，允许多个玩家联机下棋。由于只有一台服务器，为节省内存空间，采用享元模式实现该程序，得到如图5-1所示的类图。



【C++代码】

#include <iostream>

#include <vector>

using namespace std;

enum PieceColor {BLACK, WHITE}; // 棋子颜色

class PiecePos { // 棋子位置

private :

int x;

int y;

public :

PiecePos(int a, int b); x(a), y(b) {}

int getX() { return x; }

int getY() { return y; }

};

class Piece { // 棋子定义

protected :

PieceColor m\_color; // 颜色

PiecePos m\_pos; // 位置

public :

Piece(PieceColor color, PiecePos pos) : m\_color(color), m\_pos(pos) {}

(1) ;

};

class BlackPiece : public Piece {

public :

BlackPiece(PieceColor color, PiecePos pos) : Piece(color, pos) {}

void Draw() { cout << "draw a black piece" << endl; }

};

class WhitePiece : public Piece {

public :

WhitePiece(PieceColor color, PiecePos pos) : Piece(color, pos) {}

void Draw() { cout << "draw a white piece" << endl;

};

class PieceBoard {

private :

vector< (2) > m\_vecPiece; // 棋盘上已有的棋子

string m\_blackName; // 黑方名称

string m\_whiteName; // 白方名称

public :

PieceBoard(string black, string white) :

m\_blackName(black), m\_whiteName(white) {}

// 一步棋，在棋盘上放一颗棋子

void SetPiece(PieceColor color, PiecePos pos) {

(3) piece = NULL;

if (color == BLACK) { // 放黑子

piece = new BlackPiece(color, pos); // 获取一颗黑子

cout << m\_blackName << "在位置（" << pos.getX() << ","

<< pos.getY() << "）";

(4) ;

} else { // 放白子

Piece = new WhitePiece(color, pos); // 获取一颗白子

cout << m\_whiteName << "在位置（" << pos.getX() << ","

<< pos.getY() << "）";

(5) ;

}

m\_vecPiece.push\_back(piece);

}

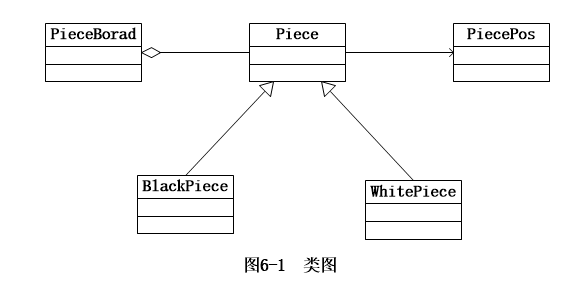
};

**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】

享元（FlyWeight）模式主要用于减少创建对象的数量，以降低内存占用，提高性能。先要开发一个网络围棋程序，允许多个玩家联机下棋。由于只有一台服务器，为节省内存空间，采用享元模式实现该程序，得到如图6-1所示的类图。



【Java代码】

import java.util.\*;

enum PieceColor { BLACK, WHITE } // 棋子颜色

class PiecePos { // 棋子位置

private int x;

private int y;

public PiecePos(int a, int b) { x = a; y = b; }

public int getX() { return x; }

public int getY() { return y; }

}

abstract class Piece { // 棋子定义

protected PieceColor m\_color; // 颜色

protected PiecePos m\_pos; // 位置

public Piece(PieceColor color, PiecePos pos) {

m\_color = color;

m\_pos = pos;

}

(1) ;

}

class BlackPiece extends Piece {

public BlackPiece(PieceColor color, PiecePos pos) {

super(color, pos);

}

public void draw() { System.out.println("draw a blackpiece"); }

}

class WhitePiece extends Piece {

public WhitePiece(PieceColor color, PiecePos pos) {

super(color, pos);

}

public void draw() { System.out.println("white a blackpiece"); }

}

class PieceBoard { // 棋盘上已有的棋子

private static final ArrayList<(2)> m\_arrayPiece = new ArrayList

private String m\_blackName; // 黑方名称

private String m\_whiteName; // 白方名称

public PieceBoard(String black, String white) {

m\_blackName = black;

m\_whiteName = white;

}

// 一步棋，在棋盘上放一颗棋子

public void SetPiece(PieceColor color, PiecePos pos) {

(3) piece = null;

if (color == PieceColor.BLACK) { // 放黑子

piece = new BlackPiece(color, pos); // 获取一颗黑子

System.out.println(m\_blackName + "在位置(" + pos.getX() + ","

+ pos.getY() + ")");

(4) ;

} else { // 放白子

piece = new WhitePiece(color, pos); // 获取一颗白子

System.out.println(m\_whiteName + "在位置(" + pos.getX() + ","

+ pos.getY() + ")");

(5) ;

}

m\_arrayPiece.add(piece);

}

}

**全国计算机技术与软件专业技术资格（水平）考试**

# 2022年上半年 软件设计师 下午试卷

（考试时间14:00～16:30共150分钟）

|  |
| --- |
| **请按下述要求正确填写答题卡** |

1. 在答题纸的指定位置填写你所在的省、自治区、直辖市、计划单列市的名称。

2. 在答题纸的指定位置填写准考证号、出生年月日和姓名。

3. 答题纸上除填写上述内容外只能写解答。

4. 本试卷共6道题，试题一至试题四是必答题，试题五至试题六选答1道。每题15分，满分75分。

5. 解答时字迹务必清楚，字迹不清时，将不评分。

6. 仿照下面的例题，将解答写在答题纸的对应栏内。

例题

2022年上半年全国计算机技术与软件专业技术资格（水平）考试日期是（1）月（2）日。

因为正确的解答是“5月28日”，故在答题纸的对应栏内写上“5”和“28”（参看下表）。

|  |  |
| --- | --- |
| 例题 | 解答栏 |
| （1） | 5 |
| （2） | 28 |

**试题一（共15分）**

阅读下列说明和数据流图，回答问题1至问题4，将解答填入答题纸的对应栏内。

【说明】

某公司欲开发一款外卖订餐系统，集多家外卖平台和商户为一体，为用户提供在线浏览餐品、订餐和配送等服务。该系统的主要功能是：

（1）入驻管理。用户注册；商户申请入驻，设置按时间段接单数量阅值等。系统存储商户/用户信息。

（2）餐品管理。商户对餐品的基本信息和优惠信息进行发布、修改、删除。系统存储相关信息。

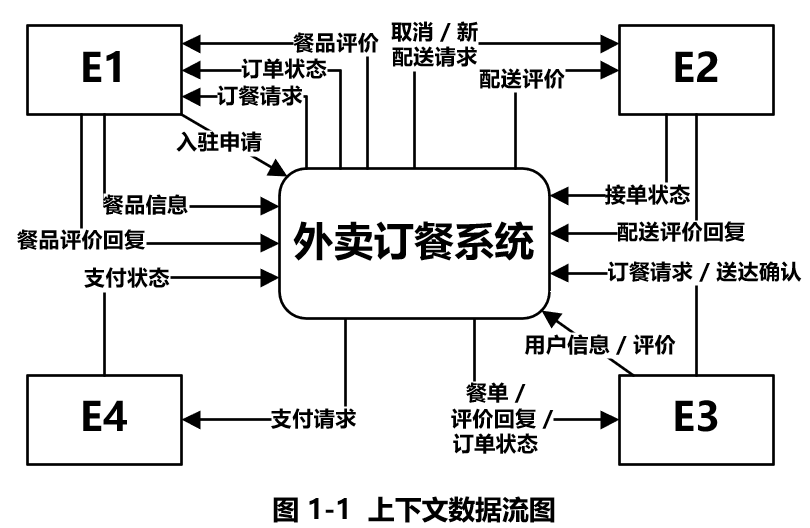
（3）订餐。用户浏览商户餐单，选择餐品及数量后提交订餐请求。系统存储订餐订单。

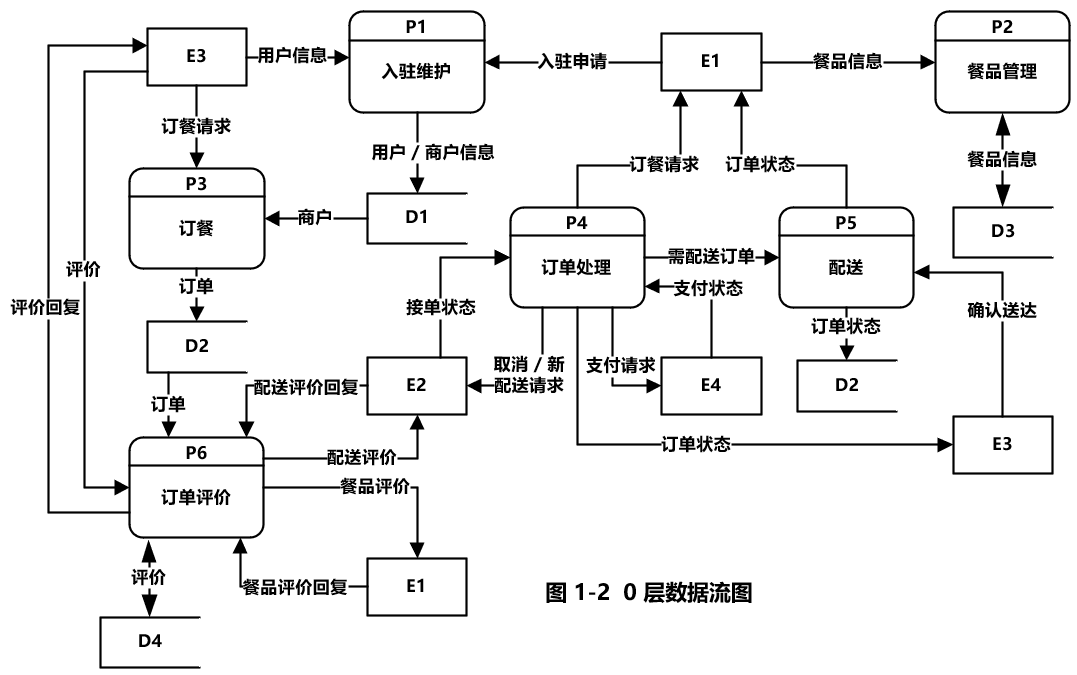
（4）订单处理。收到订餐请求后，向外卖平台请求配送。外卖平台接到请求后发布配送单，由平台骑手接单，外卖平台根据是否有骑手接单返回接单状态。若外卖平台接单成功，系统给支付系统发送支付请求，接收支付状态。支付成功，更新订单状态为已接单，向商户发送订餐请求并由商户打印订单，给用户发送订单状态；若支付失败，更新订单状态为下单失败，向外卖平台请求取消配送，向用户发送下单失败。若系统接到外卖平台返回接单失败或超时未返回接单状态，则更新订单状态为下单失败，向用户发送下单失败。

（5）配送。商户备餐后，由骑手取餐配送给用户。送达后由用户扫描骑手出示的订单上的配送码后确认送达，订单状态更改为已送达，并发送给商户。

（6）订单评价。用户可以对订单餐品、骑手配送服务进行评价，推送给对应的商户、所在外卖平台，商户和外卖平台对用户的评价进行回复。系统存储评价。

现采用结构化方法对外卖订餐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】（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】（4分）

使用说明中的词语，给出图1-2中的数据存储D1〜D4的名称。

【问题3】（4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）

根据说明，采用结构化语言对“订单处理”的加工逻辑进行描述。

**试题二（共15分）**

阅读下列说明，回答问题1至问题4，将解答填入答题纸的对应栏内。

【说明】

为了提高接种工作，提高效率，并为了抗击疫情提供疫苗接种数据支撑，需要开发一个信息系统，下述需求完成该系统的数据库设计。

【需求分析结果】

（1）记录疫苗供应商的信息，包括供应商名称，地址和一个电话。

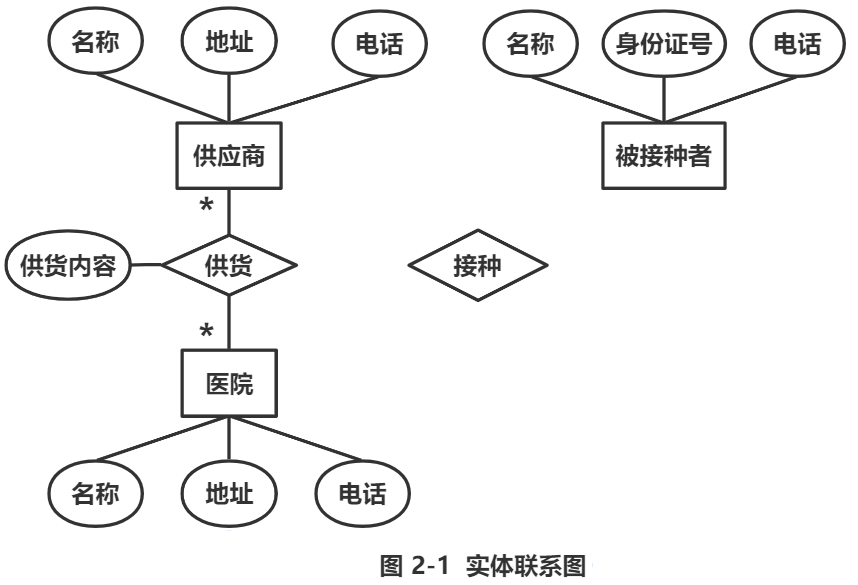
（2）记录接种医院的信息，包括医院名称、地址和一个电话。

（3）记录接种者个人信息，包括姓名、身份证号和一个电话。

（4）记录接种者疫苗接种信息，包括接种医院信息，被接种者信息，疫苗供应商名称和接种日期，为了提高免疫力，接种者可能需要进行多次疫苗接种，（每天最多接种一次，每次都可以在全市任意一家医院进行疫苗接种）。

【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。



【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）：

供应商（供应商名称，地址，电话）

医院（医院名称，地址，电话）

供货（供货商名称， （a） ，供货内容）

被接种者（姓名，身份证号，电话）

接种（接种者身份证号， （b） ，医院名称、供应商名称）

【问题1】（4分）

根据问题描述，补充图2-1的实体联系图（不增加新的实体）。

【问题2】（4分）

根据题意，补充逻辑结构设计结果中的（a），（b）两处空缺，并标注主键和外键完整性约束。

【问题3】（7分）

若医院还兼有核酸检测的业务，检测时可能需要进行多次核酸检测（每天最多检测一次），但每次都可以在全市任意一家医院进行检测。

请在图2-1中增加“被检测者”实体及相应的属性。医院与被检测者之间的“检测”联系及必要的属性，并给出新增加的关系模式。

“被检测者”实体包括姓名、身份证号、地址和一个电话。

“检测”联系包括检测日期和检测结果等。

**试题三（共15分）**

阅读下列说明和UML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】

某公司的人事部门拥有一个地址簿（AddressBook）管理系统（AddressBookSystem），用于管理公司所有员工的地址记录（PersonAddress）。员工的地址记录包括：姓名、住址、城市、省份、邮政编码以及联系电话等信息。

管理员可以完成对地址簿中地址记录的管理操作，包括：

（1）管理地址记录。根据公司的人员变动情况，对地址记录进行添加、修改、删除等操作。

（2）排序。按照员工姓氏的字典顺序或邮政编码对地址簿中的所有记录进行排序。

（3）打印地址记录。以邮件标签的格式打印一个地址单独的地址簿。

系统会对地址记录进行管理，为便于管理，管理员在系统中为公司的不同部门建立员工的地址簿的操作，包括：

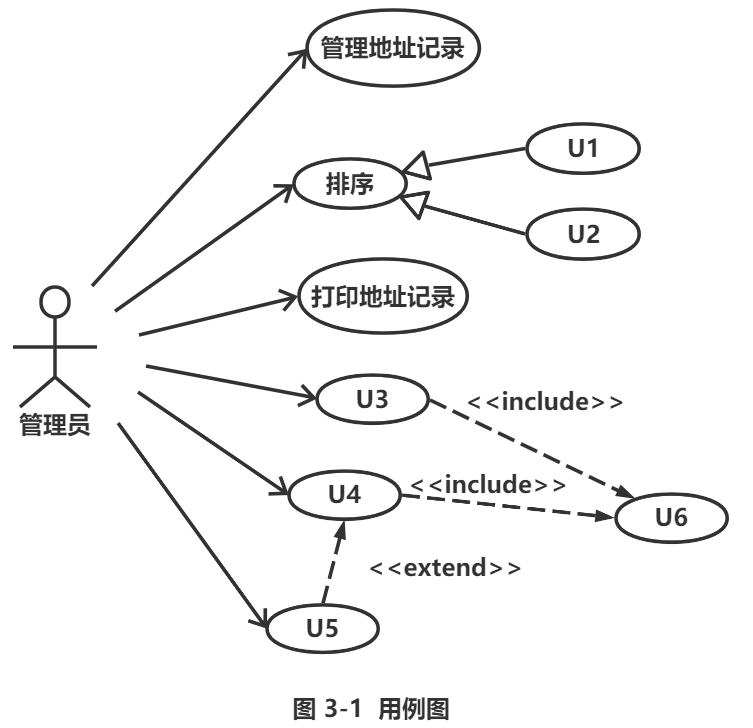
（1）创建地址簿。新建一个地址簿并保存。

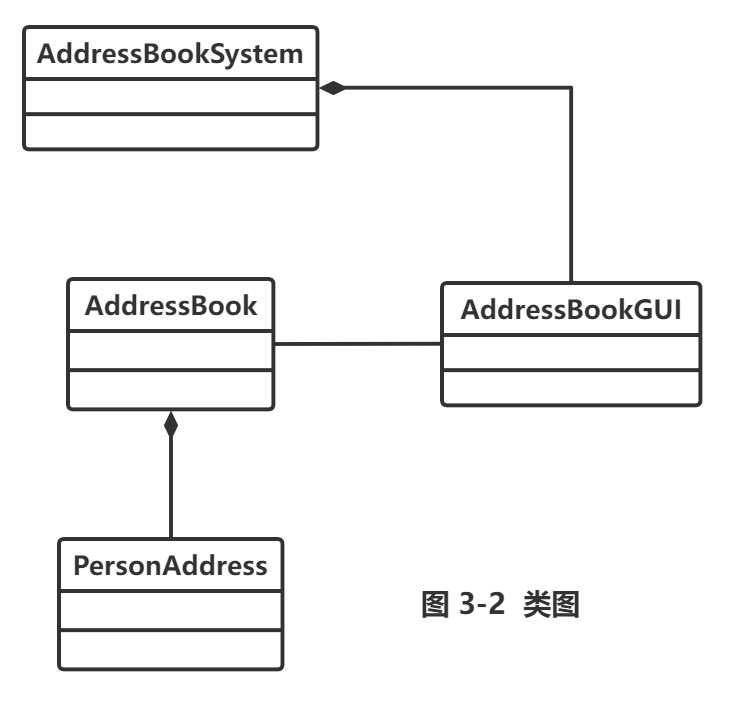
（2）打开地址簿。打开一个已有的地址簿。

（3）修改地址簿。对打开的地址簿进行修改并保存。

系统将提供一个GUI（图形用户界面）实现对地址簿的各种操作。

现采用面向对象方法分析并设计该地址簿管理系统，得到如图3-1所示的用例图以及图3-2所示的类图。





【问题1】（6分）

根据说明中的描述，给出图3-1中U1〜U6所对应的用例名。

【问题2】（5分）

根据说明中的描述，给出图3-2中类AddressBook的主要属性和方法以及类PersonAddress的主要属性（可以使用说明中的文字）

【问题3】（4分）

根据说明中的描述以及图3-1所示的用例图，请简要说明include和extend关系的含义是什么？

**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，将解答填入答题纸的对应栏内。

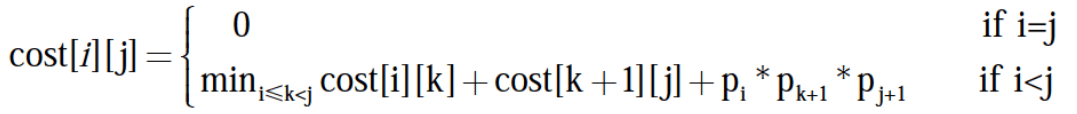
【说明】

某工程计算中要完成多个矩阵相乘（链乘）的计算任务，对矩阵相乘进行以下说明。

（1）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。假设采用标准的矩阵相乘算法，计算，需要次乘法运算，即时间复杂度为

（2）矩阵相乘满足结合律，多个矩阵相乘时不同的计算顺序会产生不同的计算量。以矩阵，，三个矩阵相乘为例，若按计算，则需要进行次乘法运算，若按计算，则需要进行次乘法运算。

矩阵连乘问题可描述为：给定个矩阵，对较大的，可能计算的顺序数量非常庞大，用蛮力法确定计算顺序是不实际的。经过对问题进行分析，发现矩阵连乘问题具有最优子结构，即若的一个最优计算顺序从第个矩阵处断开，即分为和两个子问题，则该最优解应该包含的一个最优计算顺序和的一个最优计算顺序。据此构造递归式：



其中，cost[i][j]表示的最优计算的代价。最终需要求解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、…、个矩阵相乘的最小计算量及最优计算顺序。下面是算法的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，其中元素cost[i][j]表示的最优计算的计算代价

trace[][]：二维数组，长度为，其中元素trace[i][j]表示的最优计算对应的划分位置，即

（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; i < n - p; i ++ ) {

(1) ;

tempCost = -1;

for (k = i; (2) ; k ++ ) {

temp = (3) ;

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

tempCost = temp;

tempTrace = k;

}

}

cost[i][j] = tempCost;

(4) ;

}

}

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 = 4，各个矩阵的维数：A1为15 \* 5，A2为5 \* 10，A3为10 \* 20，A4为20 \* 25，即维数序列为15, 5, 10, 20, 25。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） 。

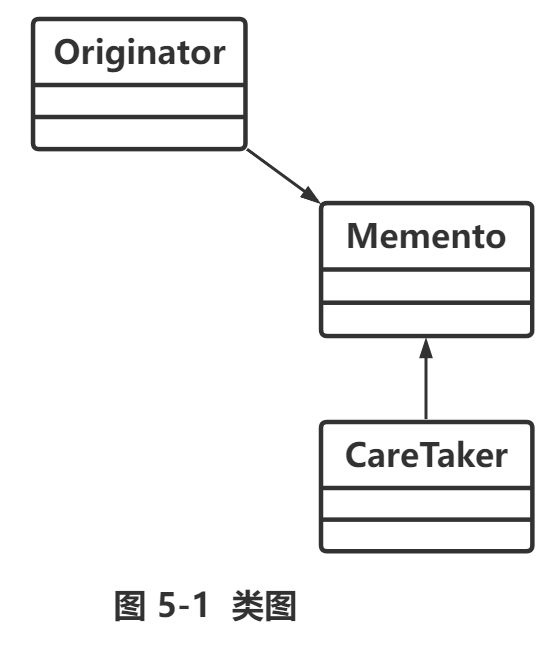
**试题五（共15分）**

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】

在软件系统中，通常都会给用户提供取消、不确定或者错误的操作，允许将系统回复到原先的状态。现使用备忘录（Memento）模式实现该要求，得到如图5-1所示的类图。

Memento包含了要被恢复的状态。Originator创建并在Memento中存储状态。CareTaker负责从Memento中恢复状态。



【C++代码】

#include <iostream>

#include <string>

#include <vector>

using namespace std;

class Memento {

private:

string state;

public:

Memento(string state) { this->state = state; }

string getState() { return state; }

};

class Originator {

private:

string state;

public:

void setState(string state) { this->state = state; }

string getState() { return state; }

Memento saveStateToMemento() { return (1) ; }

void getStateFromMemento(Memento Memento) { state = (2) ; }

};

class CareTaker {

private:

vector<Memento> mementoList;

public:

void (3) { mementoList.push\_back(state); }

(4) { return mementoList[index]; }

};

int main() {

Originator \*originator = new Originator();

CareTaker \*careTaker = new CareTaker();

originator->setState("State #1");

originator->setState("State #2");

careTaker->add( (5) );

originator->setState("State #3");

careTaker->add( (6) );

originator->setState("State #4");

cout << "Current State：" + originator->getState() << endl;

originator->getStateFromMemento(careTaker->get(0));

cout << "First saved State：" + originator->getState() << endl;

originator->getStateFromMemento(careTaker->get(1));

cout << "Second saved State：" + originator->getState() << endl;

}

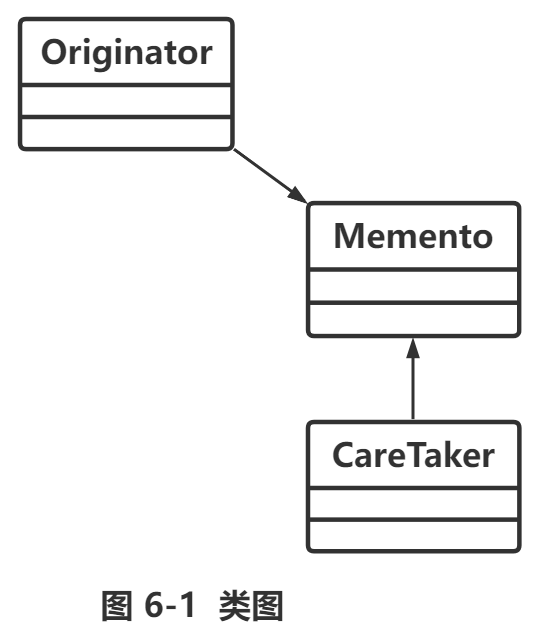
**试题六（共15分）**

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】

在软件系统中，通常都会给用户提供取消、不确定或者错误的操作，允许将系统回复到原先的状态。现使用备忘录（Memento）模式实现该要求，得到如图6-1所示的类图。

Memento包含了要被恢复的状态。Originator创建并在Memento中存储状态。CareTaker负责从Memento中恢复状态。



【Java代码】

import java.util.\*;

class Memento {

private String state;

public Memento(String state) {

this.state = state;

}

public String getState() {

return state;

}

}

class Originator {

private String state;

public void setState(String state) {

this.state = state;

}

public String getState() {

return state;

}

public Memento saveStateToMemento() {

return (1) ;

}

public void getStateFromMemento(Memento Memento) {

state = (2) ;

}

}

class CareTaker {

private List<Memento> mementoList = new ArrayList<Memento>();

public (3) {

mementoList.add(state);

}

public (4) {

return mementoList.get(index);

}

}

class MementoPaneDemos {

public static void main(String[] args) {

Originator originator = new Originator();

CareTaker careTaker = new CareTaker();

originator.setState("State #1");

originator.setState("State #2");

careTaker.add( (5) );

originator.setState("State #3");

careTaker.add( (6) );

originator.setState("State #4");

System.out.println("Current State：" + originator.getState());

originator.getStateFromMemento(careTaker.get(0));

System.out.println("First saved State：" + originator.getState());

originator.getStateFromMemento(careTaker.get(1));

System.out.println("Second saved State：" + originator.getState());

}

}