**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。

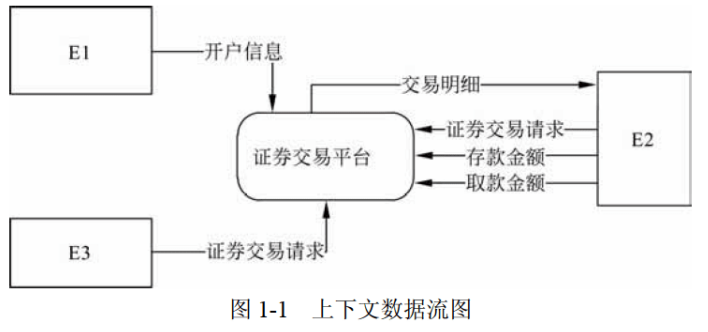
（2）存款。客户可以向其账户中存款，根据存款金额修改账户余额。

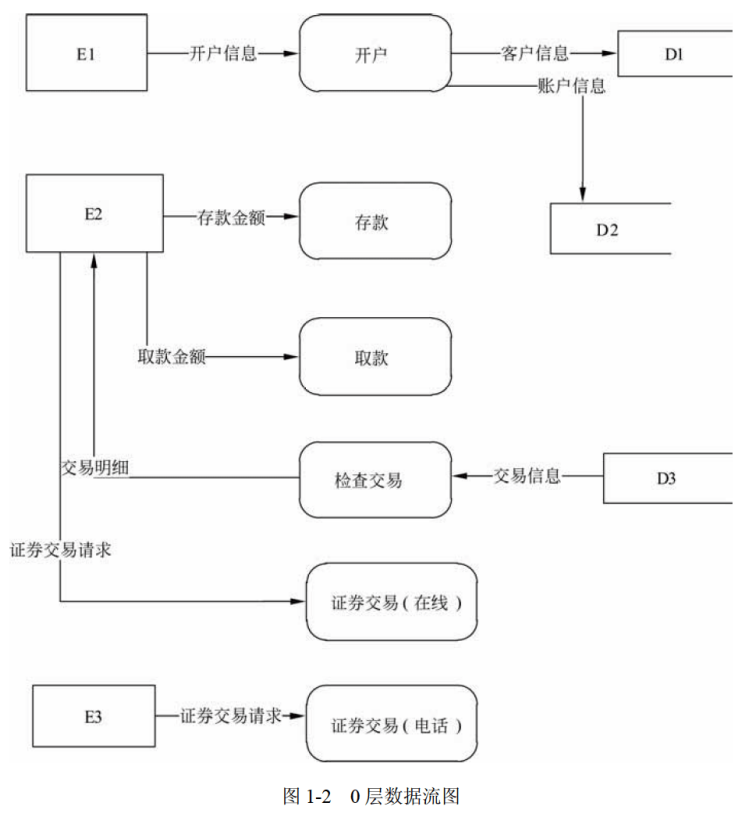
（3）取款。客户可以从其账户中取款，根据取款金额修改账户余额。

（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。

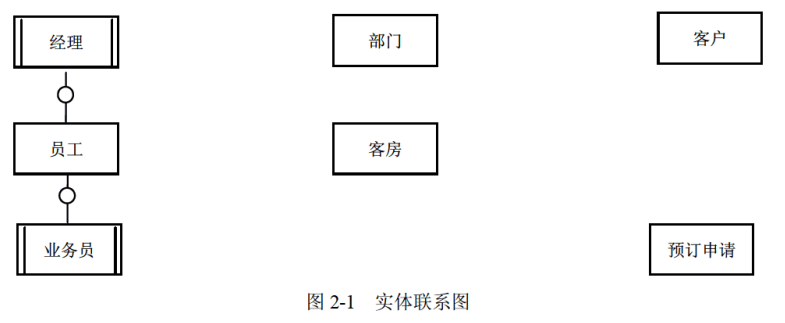
（4）客户信息包括客户号、单位名称、联系人、联系电话、联系地址，其中客户号唯一标识客户关系中的一个元组。

（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】

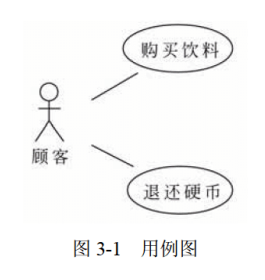
某种出售罐装饮料的自动售货机（Vending Machine）的工作过程描述如下：

（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。

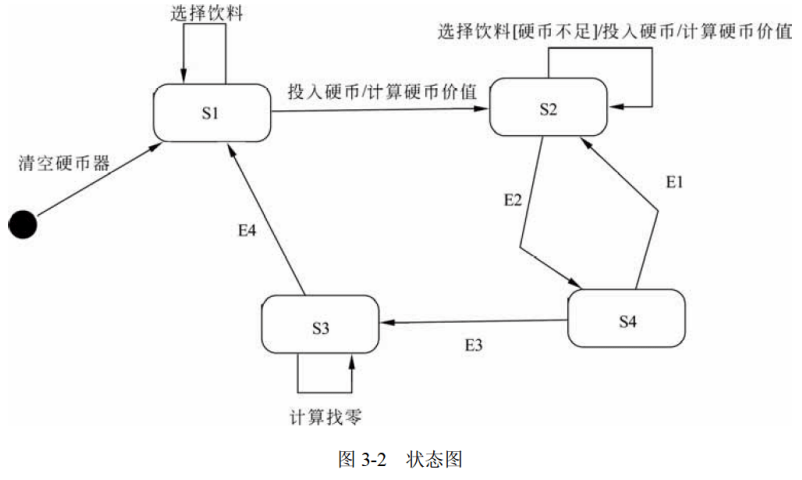
备选事件流：

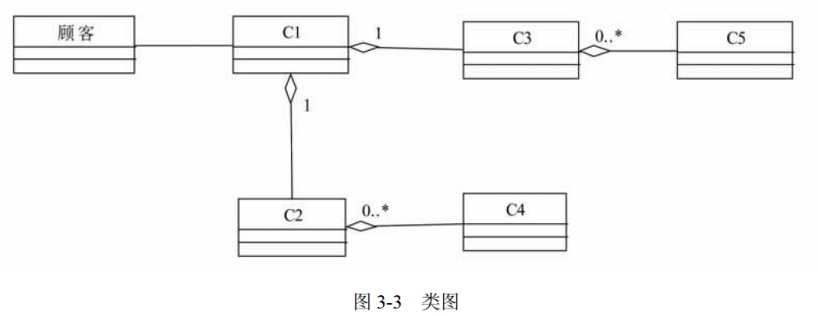
2a．若投入的硬币不足，则给出提示并退回到1；

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）

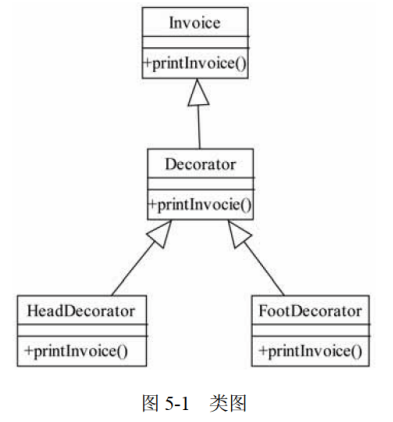
根据C代码，字符串“BBABBCAC”的next数组元素值为 （6） （直接写素值，之间用逗号隔开）。若主串为“AABBCBBABBCACCD”，子串为“BBABBCAC”，则函数kmp的返回值是 （7） 。

**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!

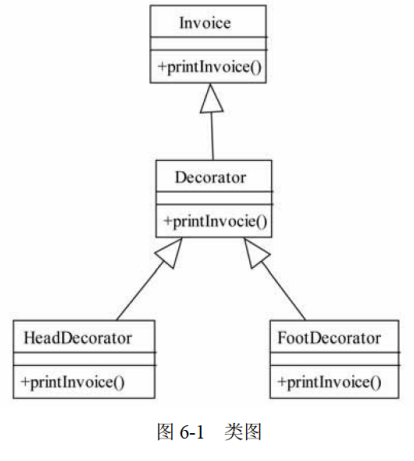
This is the footnote of the invoice!

**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!