

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纸

- 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 4 日”，故在答题纸的对应栏内写上“11”和“4”（参看下表）。

例题	解答栏
（1）	11
（2）	4

试题一至试题四是必答题

试题一（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。

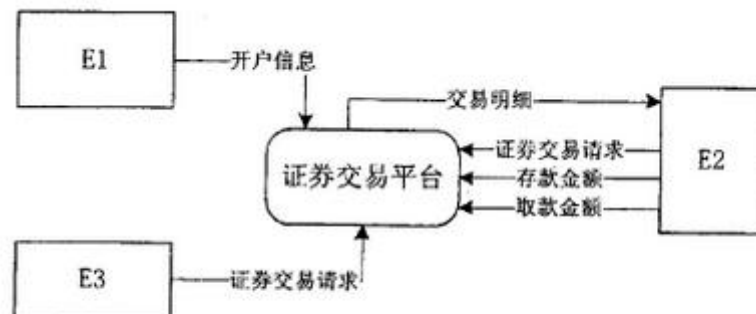


图 1-1 上下文数据流图

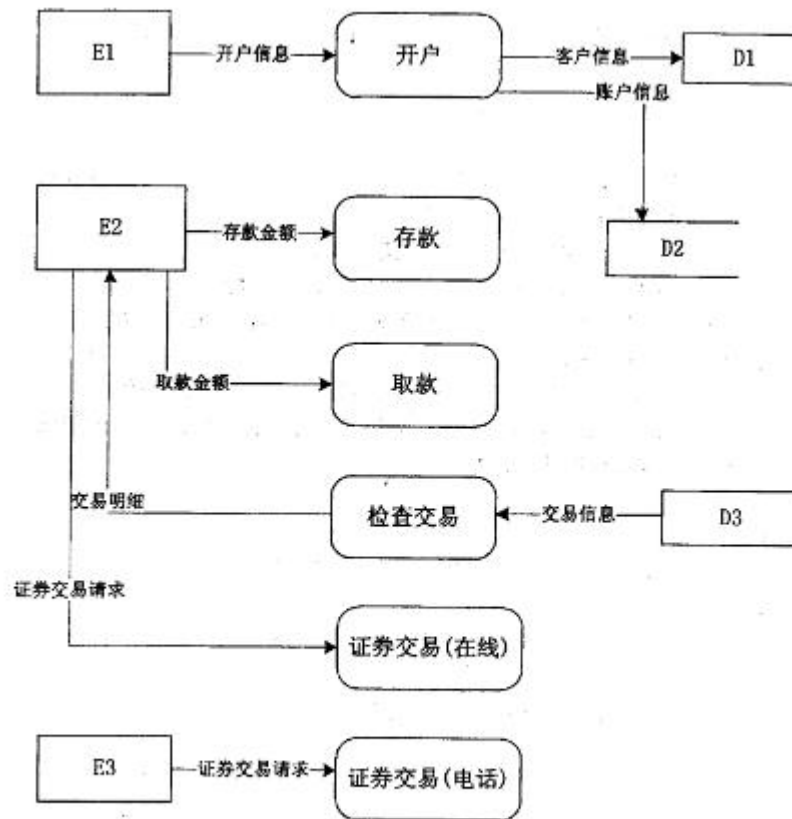


图 1-2 0 层数据流图

【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）

【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 multiple 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，设计的实体联系图如图 2-1 所示。



图 2-1 实体联系图

【关系模式设计】

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）

【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。

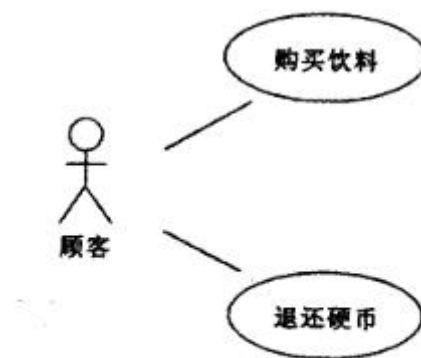


图 3-1 用例图

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。

各选事件流：

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。

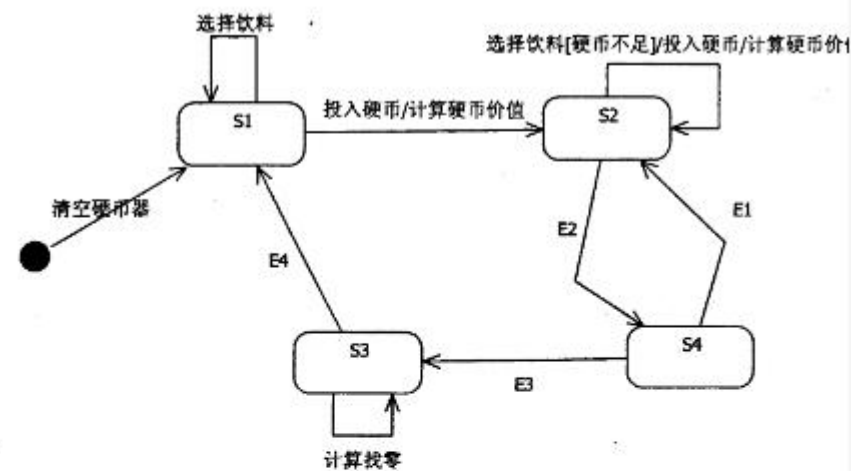


图 3-2 状态图

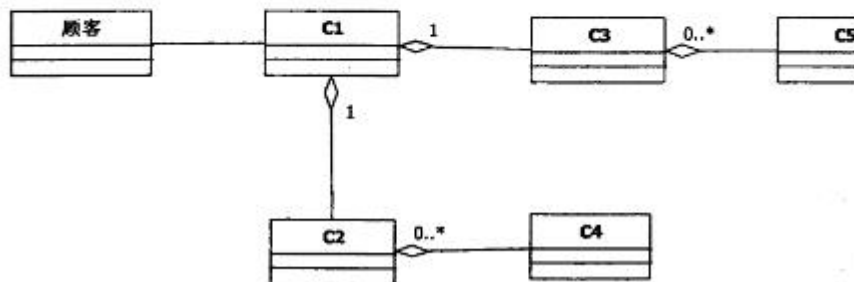


图 3-3 类图

【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

【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，将解答写在答题纸的对应栏内。

【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 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 l_t 和 l_s ，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)。

从下列的 2 道试题（试题五至试题六）中任选 1 道解答。
如果解答的试题数超过 1 道，则题号小的 1 道解答有效。

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（Invoice）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。

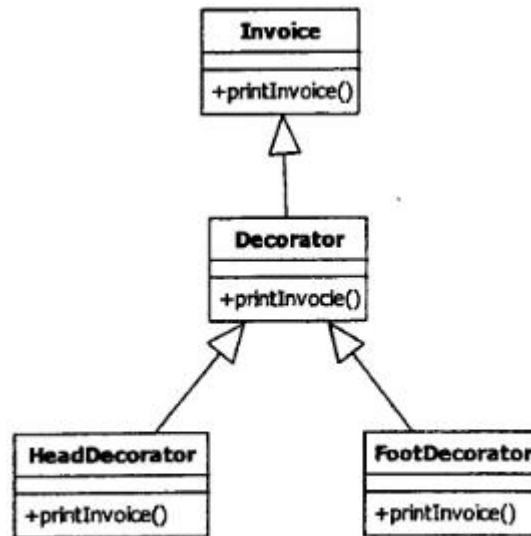


图 5-1 类图

【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(){
            (4) ;
            cout<< "This is the footnote of the invoice!"<< endl;
        }
};

```

```

int main(void) {
    Invoice t;
    FootDecorator f(&t);
    HeadDecorator h(&f);
    h.printInvoice();
    cout<<"-----"<<endl;
    FootDecorator a(NULL) ;
    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（Invoice）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。

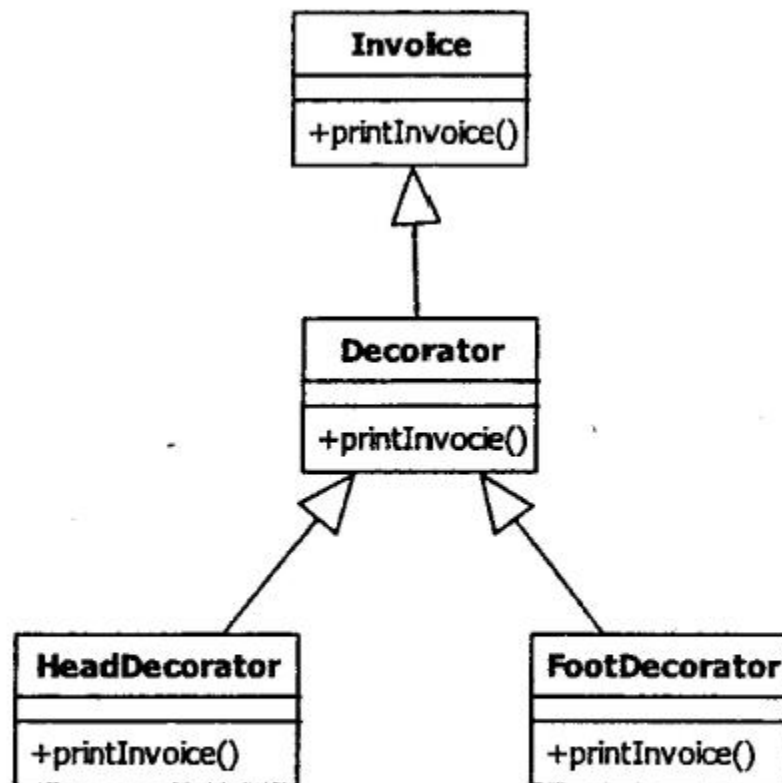


图 6-1 类图

【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!