全国计算机技术与软件专业技术资格考试
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纸

1. 本试卷共六道题，其中，试题（一）～试题（四）为必答题，试题（五）～试题（六）为选答题，满分 75 分。
2. 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。
3. 在答题纸的指定位置填写准考证号、出生年月日和姓名。
4. 答题纸上除填写上述内容外只能写解答。
5. 解答时字迹务必清楚，字迹不清时，将不评分。

例题

2019 年下半年全国计算机技术与软件专业技术资格考试日期是__（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（15 分）

阅读下列说明和图，回答问题 1 至问题 4，将解答填入答题纸的对应栏内。

【说明】

某公司打算开发一款二手车物流系统，以有效提升物流分发效率，该系统的主要功能是：

（1）订单管理。系统抓取线索，将车辆交易系统中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，并在存储系统中存储，然后系统帮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：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。

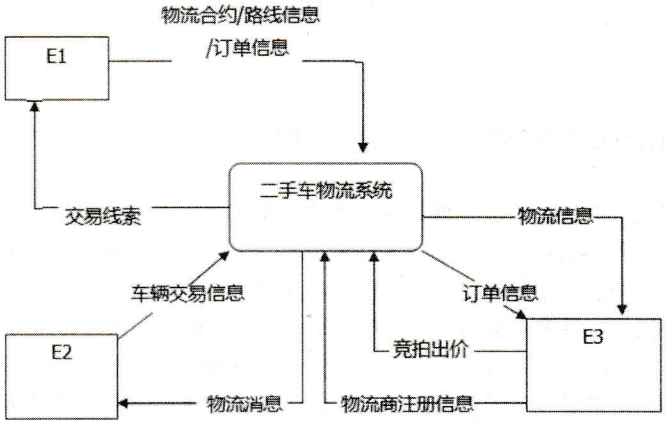

图 1-1　上下文数据流图

【问题 1】（3 分）

使用说明中的词语，给出图 1-1 中的实体 E1-E3 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。

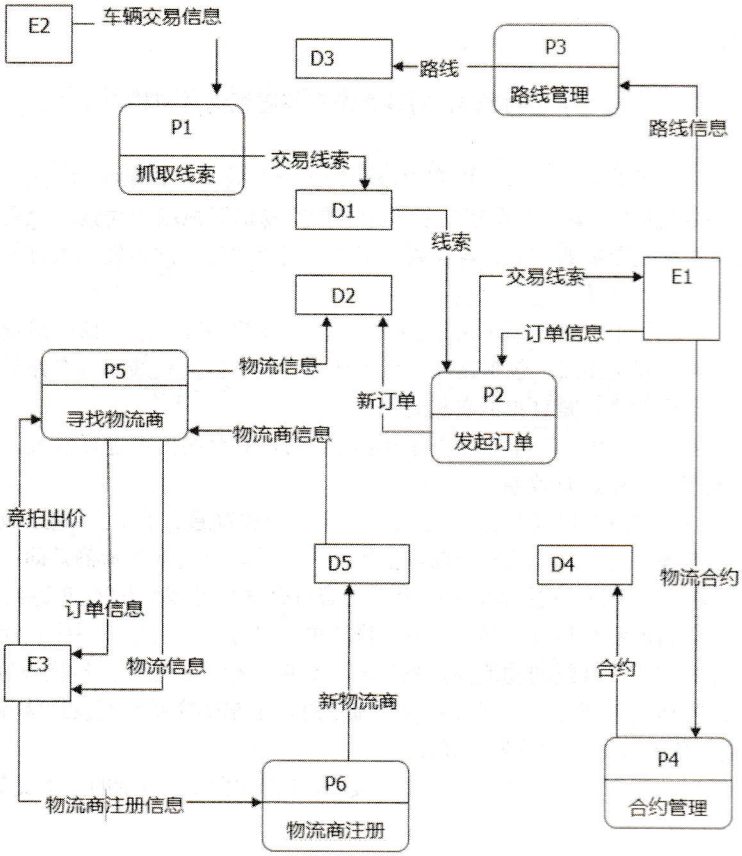

图 1-2　0 层数据流图

试题二（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】

公司拟开发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。

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。

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识员工关系中的每一个元组；根据培训师、部门负责人等不同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成绩；一名培训师可以讲授多门课程，一门课程可由多名培训师讲授。

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中的每一个元组。

【关系模式设计】

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））

根据需求阶段收集的信息，设计的实体-关系图如图 2-1 所示。

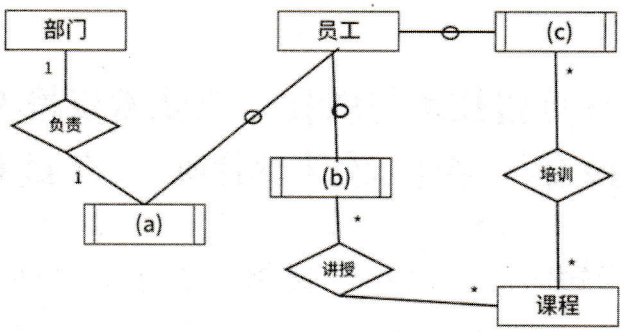

图 2-1　实体-关系图

【问题 1】（4 分）

（1）补充图 2-1 中的空（a）至空（c）。

（2）图 2-1 中是否存在缺失关系，若存在，则说明所缺失的关系和关系类型 。

【问题 2】（5 分）

根据题意，将关系模式设计中的空（d）至空（f）补充完整。

【问题 3】（4 分）

员工关系模式的主键为（g），外键为（h），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。

问题三（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】

某牙科诊所拟开发一套信息系统，用于管理病人基本信息和就诊信息。诊所工作人员包括医护人员（Dental staff）、接待人员（Receptionist）和办公人员（Office staff）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（Medical insurance info）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 info）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（ Insurance Invoice）和给病人的发票（Patient Invoice），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（Enter payment）。

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。

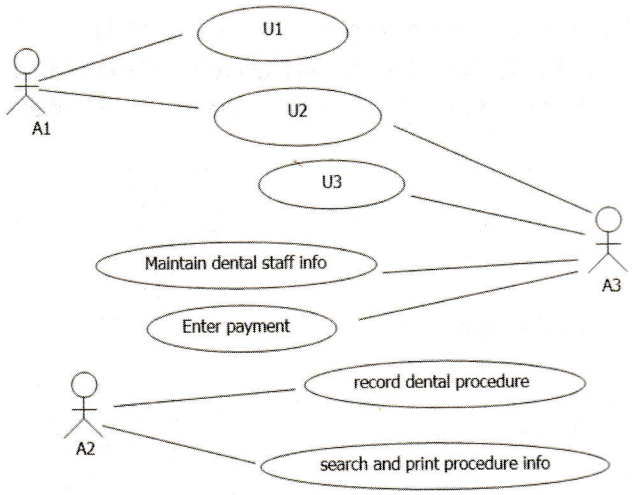

图 3-1　用例图

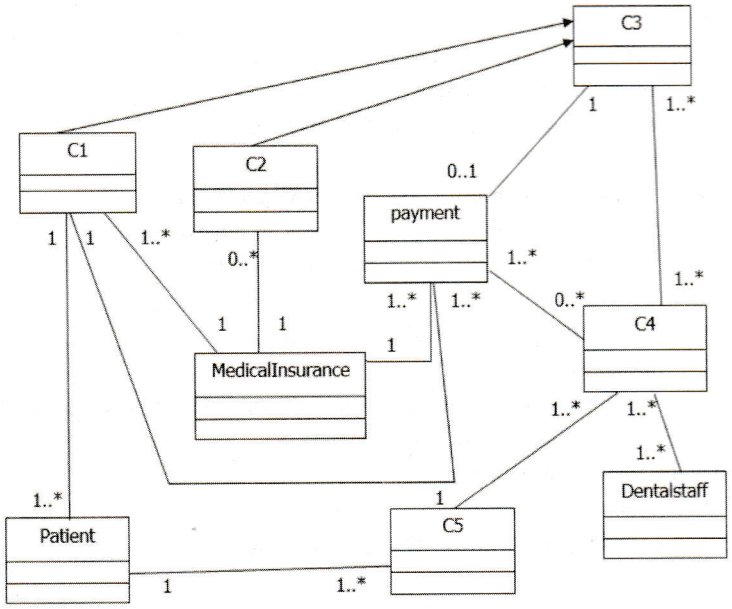

图 3-2　类图

试题四（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】

0-1 背包问题定义为：给定 i 个物品的价值 $v[1...i]$、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 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。

$$c[i][T]=\begin{cases}0 & \text{若}i=0\text{或}T=0\\ c[i-1][T] & \text{若}T<w[i]\\ \max\left(c[i-1][T-w[i]]+v[i],c[i-1][T]\right) & \text{若}i>0\text{且}T\geq w[i]\end{cases}$$

【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 Memorized_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-l, 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（15 分）

阅读下列说明和 C++代码，将应填入（n）处的语句写在答题纸对应栏内。

【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。

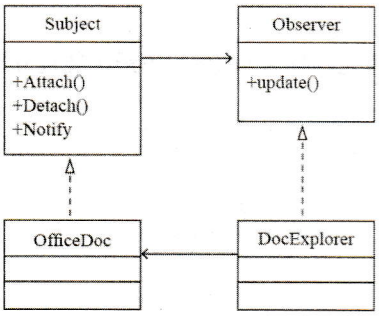

图 5-1　类图

【C++代码】

```
#include<iostream>
const OBS_MAXNUM=20;
__（1）__;
Class DocExplorer {
    Public:
        DocExplorer(__（2）__ *doc );
        __（3）__;
        Void update (OfficeDoc *doc) =0;}
class OfficeDoc {
private:
DocExplorer *myObs [OBS_MAXNUM];
Int index;
public:
    OfficeDoc () {
                Index=0;                }
    void attach (DocExplorer *o) {
            if(index>=OBS_MAXNUM||o==NUMLL) return;
            for (int loop=0; loop<index; loop++) {
                                        if(myObs[loop]==0) return;
                                        myObs[index]=o; index++;}}
    void detach (DocExplorer *o) {
            if (o==NULL) return;
            for (int loop=0; loop<index; loop++) {
                    if(loop<=index-2) myObs[loop]=myObs[index-1];
                    myObs [index -1] = NULL;
                    index--;
                    break;}}
private:
    void notifyObs () { for (int loop=0; loop<index; loop++) {
                                        myObs[loop]->__（4）__;}}}
```

DocExplorer::DocExplorer(OfficeDoc *doc){doc->__（5）__;}

试题六（15 分）

阅读下列说明和 Java 代码，将应填入（n）处的语句写在答题纸对应栏内。

【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。

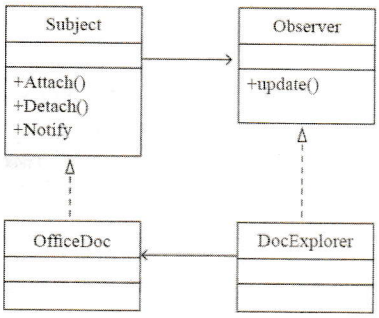

图 6-1　类图

【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) {
                System.out.println("Hello World!");
                Subject subjectA=new OfficeDoc ("subject A");
                Observer observerA=new DocExplorer ("observer A", subjectA);
                subjectA.setStatus (1);
                subjectA.Notify ();}}
```