

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。

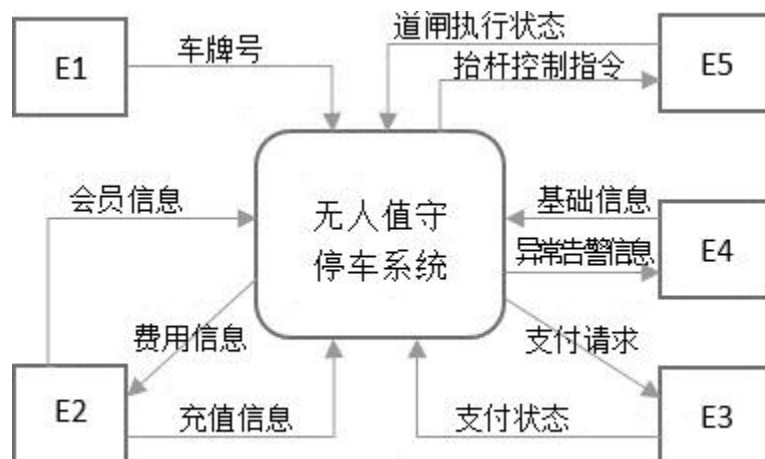


图 1-1 上下文数据流图

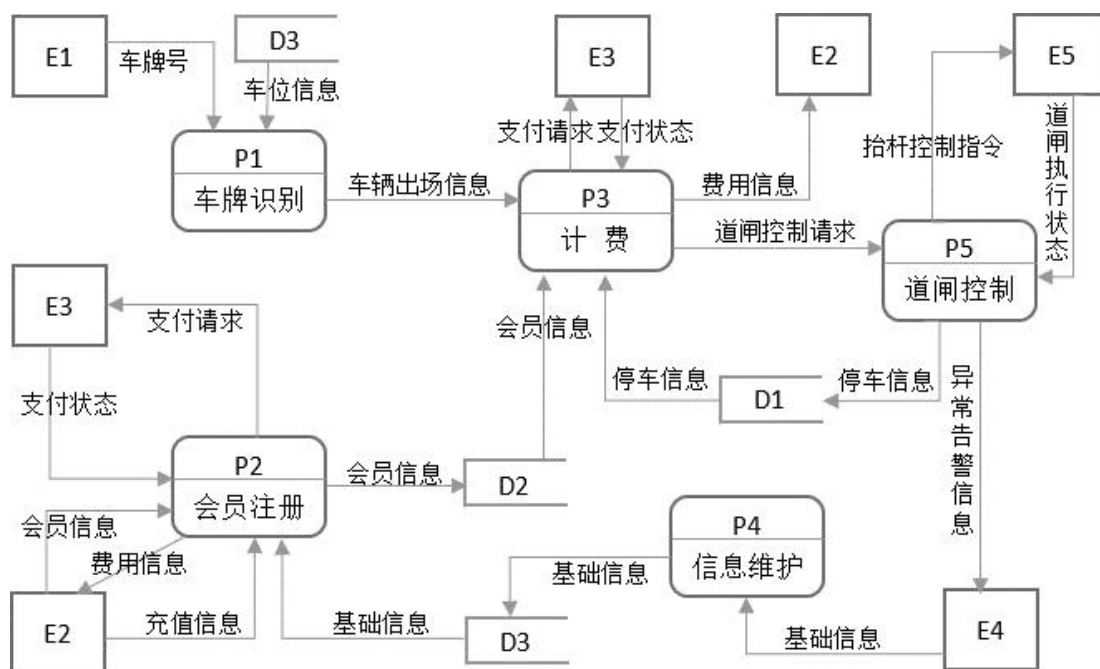


图 1-2 0层数据流图

【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。

E1：车辆 E2：车主 E3：支付系统 E4：管理人员 E5：道闸控制系统

【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。

D1：停车记录表 D2：会员信息表 D3：基础信息表

【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。

数据流名称	起点	终点
道闸控制请求	P1	P5
计费规则	D3	P3
更新余额	P3	D2
修改空余车位数	P5	D3

【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。

收到道闸控制请求

IF 道闸执行状态为正常放行时

THEN

IF 入场车辆

THEN 将车牌号及其入场时间信息存入车停记录，

修改空余车位数

ELSE

更新停车状态，修改空余车位数

ENDIF

ELSE

向管理人员发送异常告警信息

ENDIF

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。

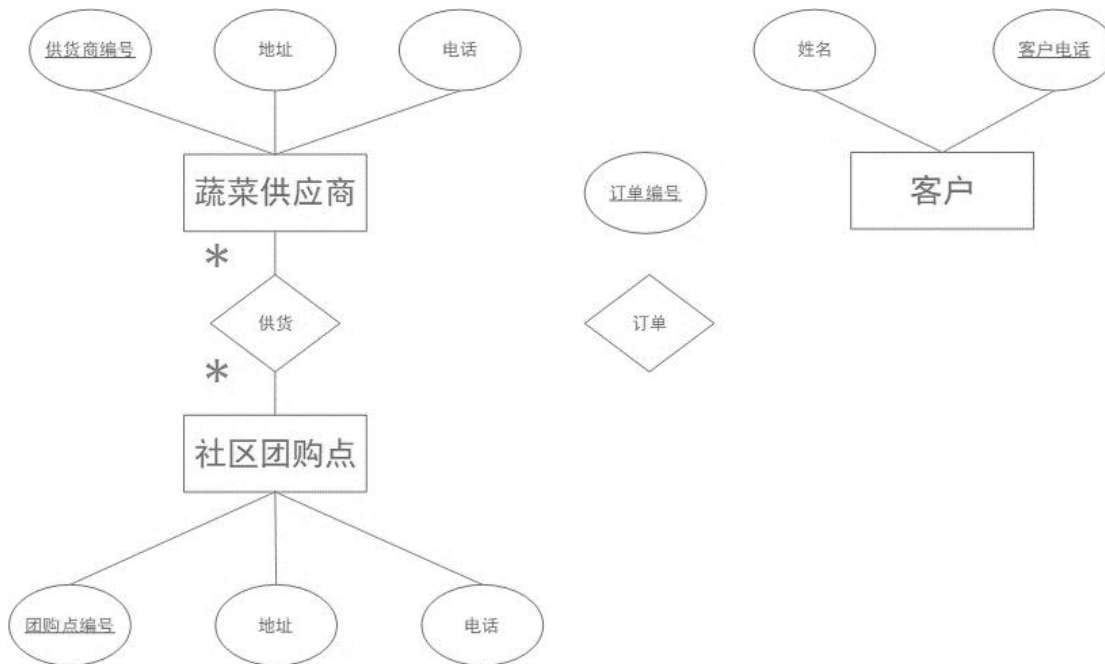


图 2-1 实体联系图

【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。

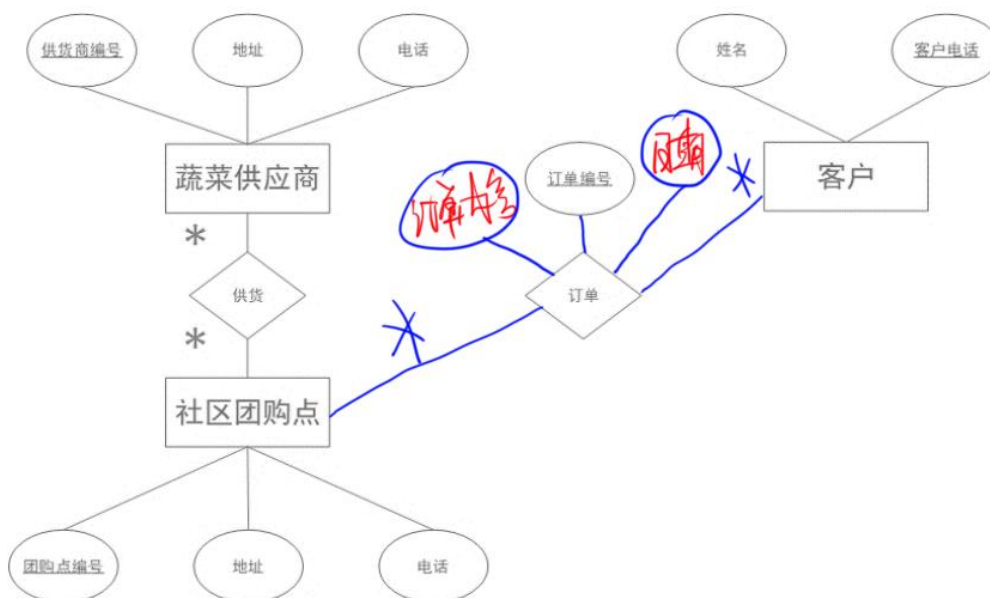


图 2-1 实体联系图

【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。

（a）：团购点编号 （b）：客户电话

“供货”关系的主键：（供货商编号，团购点编号）
外键：供货商编号，团购点编号

“订单”关系的主键：订单编号
外键：团购点编号，客户电话

【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。

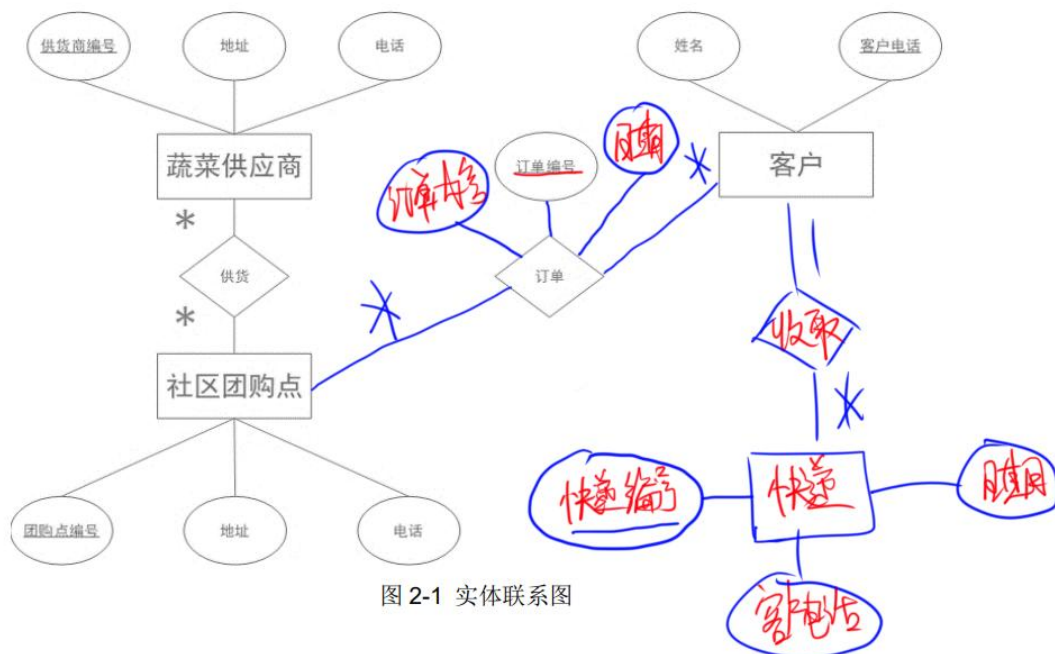


图 2-1 实体联系图

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；并给患者发送收货验证码。

（6）送药上门：快递人员将配制好的药品送到患者指定的收货地址。患者收货时，向快递人员出示收货验证码，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。

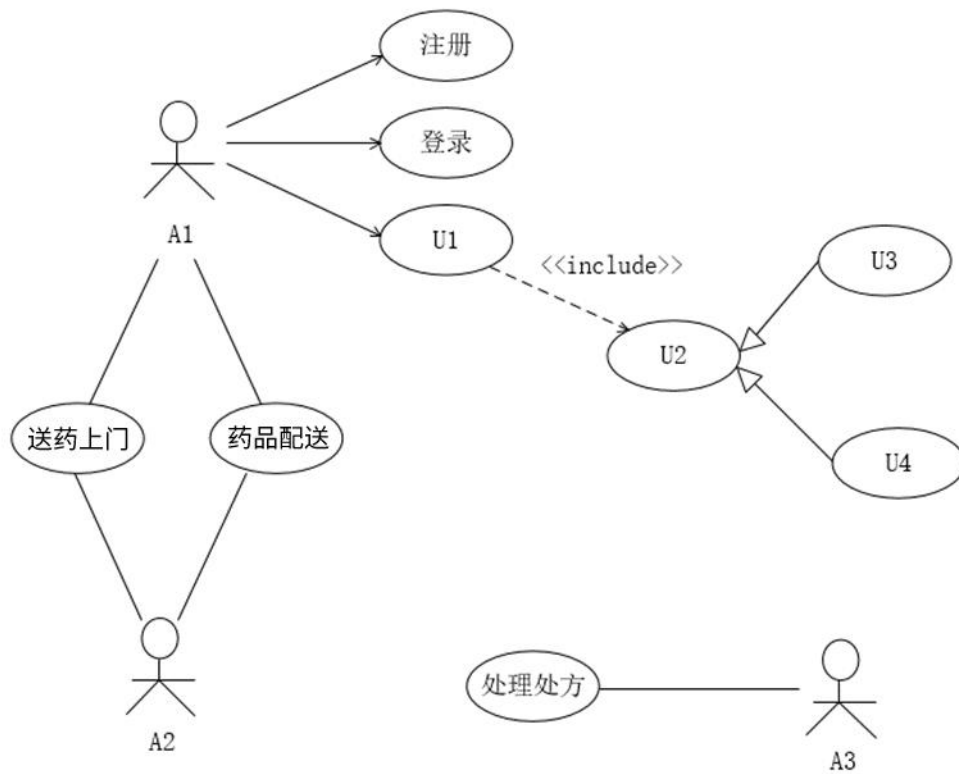


图 3-1 用例图

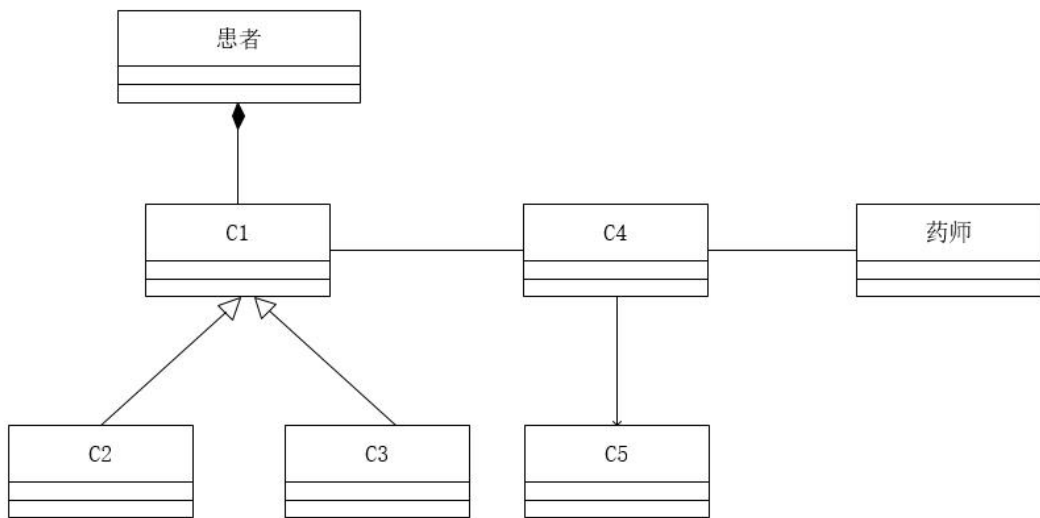


图 3-2 类图

【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 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。

A1: 患者 A2: 快递人员 A3: 药师
U1: 确认处方 U2: 支付方式
U3: 微信 U4: 支付宝 U3和U4可以互换

【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 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。

C1: 支付方式
C2: 微信 C3: 支付宝 C2和C3可以互换

C4: 处方 C5: 药品

【问题 3】（3 分）

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

【问题 3】（3 分）

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

包含**include**: 一个用例包含另一个用例，当基本用例执行时，被包含用例一定会执行。包含关系由基本用例指向被包含用例，关系是一条带箭头的虚线，虚线上包含<<include>>

扩展**extend**: 当一个用例执行时，可能会出现特殊情况 and 可选情况，这个时候就会执行扩展用例。扩展关系由扩展用例指向基本用例，关系是一条带箭头的虚线，虚线上包含<<extend>>

泛化**generalize**: 父用例泛化子用例，子用例继承父用例的所有属性和行为，并且父用例可以出现的地方，子用例都可以将其替换。

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 V_1, V_2, \dots, V_N ，若在 V_k 处将原凸多边形划分为一个三角形 $V_1V_kV_N$ ，两个子多边形 V_1, V_2, \dots, V_k 和 V_k, V_{k+1}, \dots, V_N 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多边形的最优剖分方案。用 $m[i][j]$ 表示 V_{i-1}, V_i, \dots, V_j 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：

$$m[i][j] = \begin{cases} 0, & i \geq j \\ \min_{i \leq k < j} \{m[i][k] + m[k+1][j] + W(V_{i-1}V_kV_j)\}, & i < j \end{cases}$$

其中： $W(V_{i-1}V_kV_j) = W_{i-1,k} + W_{k,j} + W_{j,i-1}$ 为三角形 $V_{i-1}V_kV_j$ 的权重， $W_{i-1,k}, W_{k,j}, W_{j,i-1}$ 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, int j) {
    if (i == j) return;
    print_triangle(i, S[i][j]);
    print_triangle(__ (4) __);
    printf("%d--%d--%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)。

```

(1) i <= N
(2) j = i + r - 1
(3) temp < m[i][j]
(4) S[i][j] + 1, j

```

【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)

```

(5) 动态规划
(6) O (n3)
(7) O (n2)

```

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**

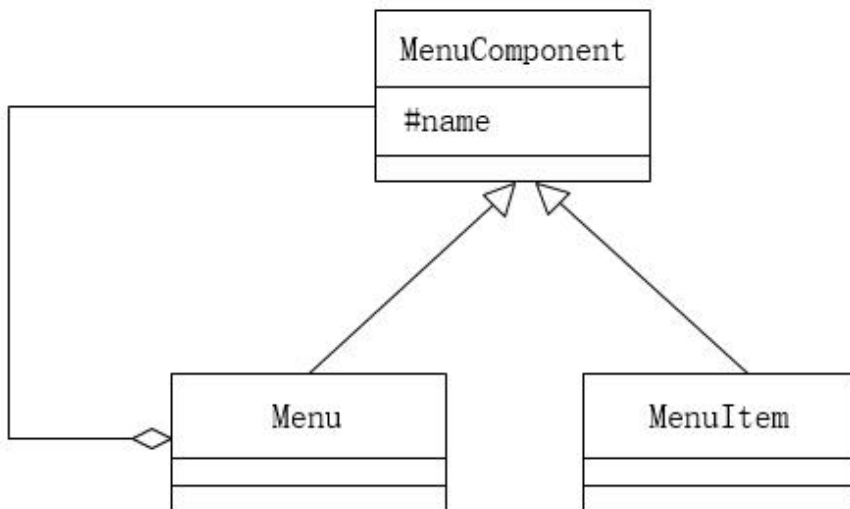


图 6-2 类图

【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 MenuComponent {
    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);
        }
    }
}
```

(1): protected

(2): abstract boolean addMenuElement(MenuComponent element)

(3): abstract List<MenuComponent> getElement()

(4): List<MenuComponent> elementsList

(5): mainMenu.addMenuElement(subMenu)

【软考达人】

软考资料免费获取

- 1、最新软考题库
- 2、软考备考资料
- 3、考前压轴题
- 4、免费督考群



微信扫一扫，立马获取



最新免费题库



备考资料+督考群

PC版题库：ruankaodaren.com