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。

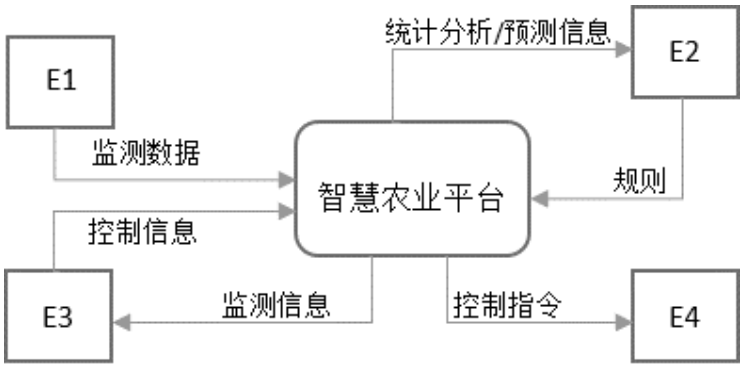

图 1-1 上下文数据流图

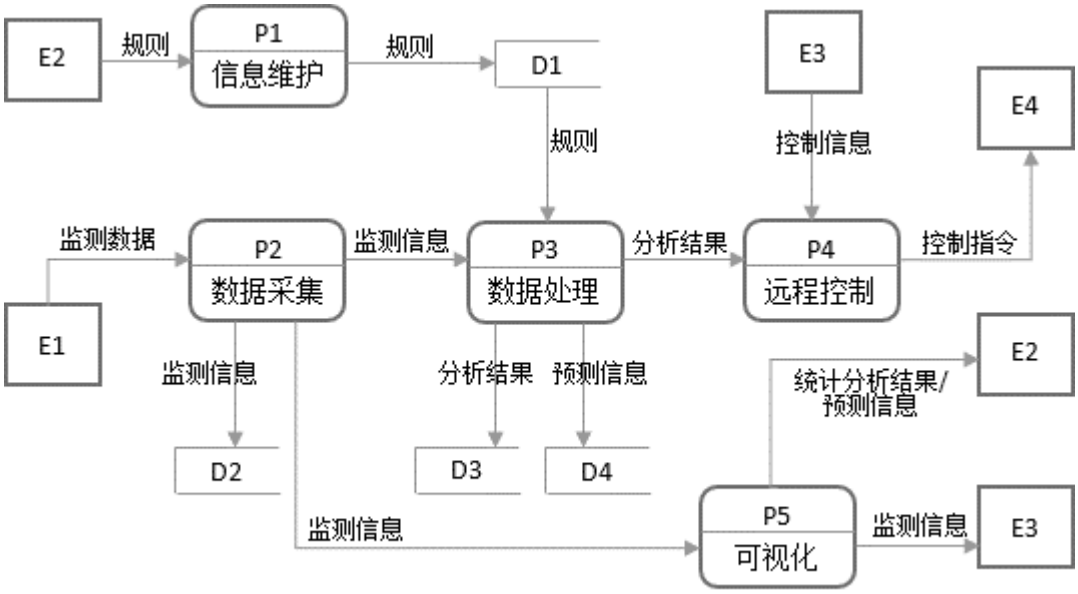

图 1-2　0 层数据流图

【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。

E1：传感器　　E2：农业专家　　E3：农业人员　　E4：控制器

【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。

D1：监控处理规则表　　D2：监测信息表　　D3：分析结果表　　D4：预测信息表

【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
分析结果	P3	P5
历史监测信息	D2	P3
预测信息	P3	P5
监控处理规则	D1	P4
请求	E2	P5

【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？

“数据处理”可以分解为：监测分析实时监测信息；分析结果可视化、存储、远程控制；综合统计和预测历史监测信息；预测信息可视化、存储；

需要注意：

1、加工有输入但是没有输出

2、加工有输出但是没有输入

3、加工的输入不足以产生输出

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。

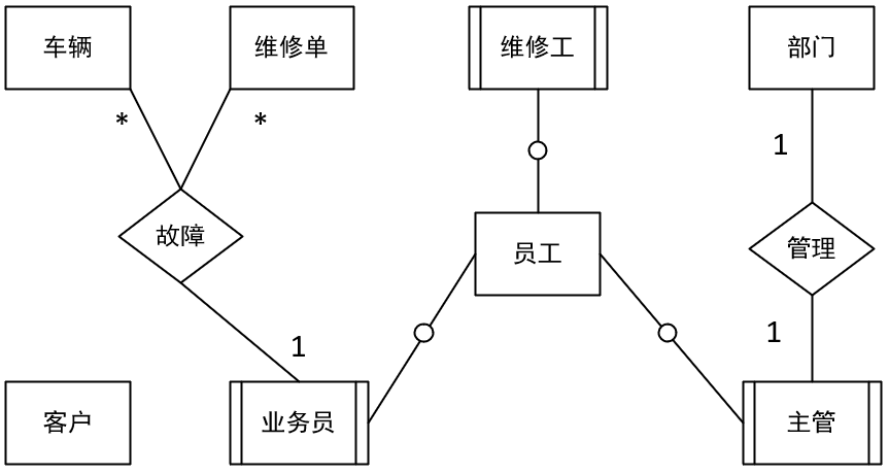

图2-1 实体联系图

【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）：

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*:*）

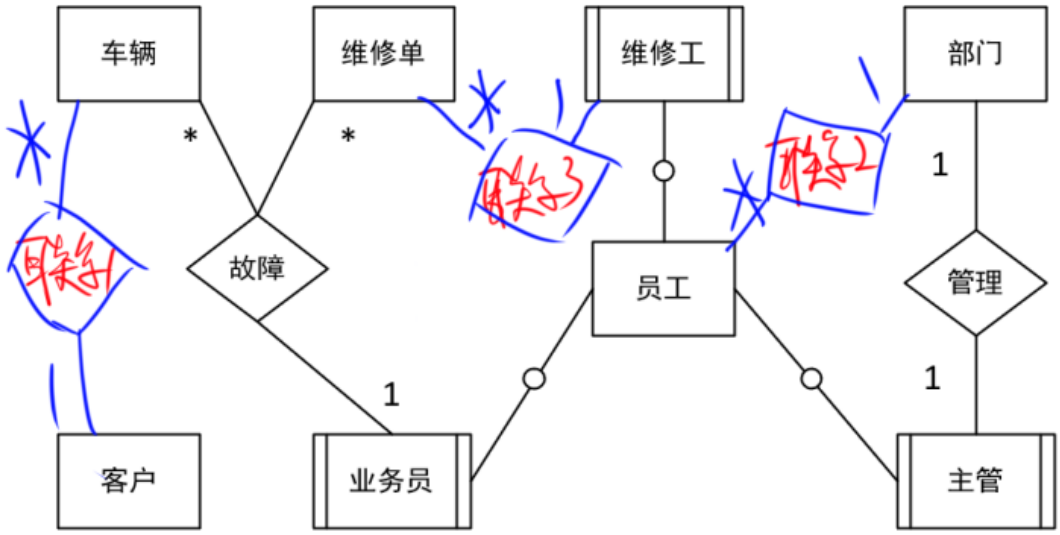

图2-1 实体联系图

【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。

（a）：客户性质　　（b）：客户号

（c）：部门号　　（d）：车牌号，维修工

【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。

“车辆”关系的主键：车牌号　　外键：客户号

“维修单”关系的主键：维修单号
外键：车牌号，维修工

【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？

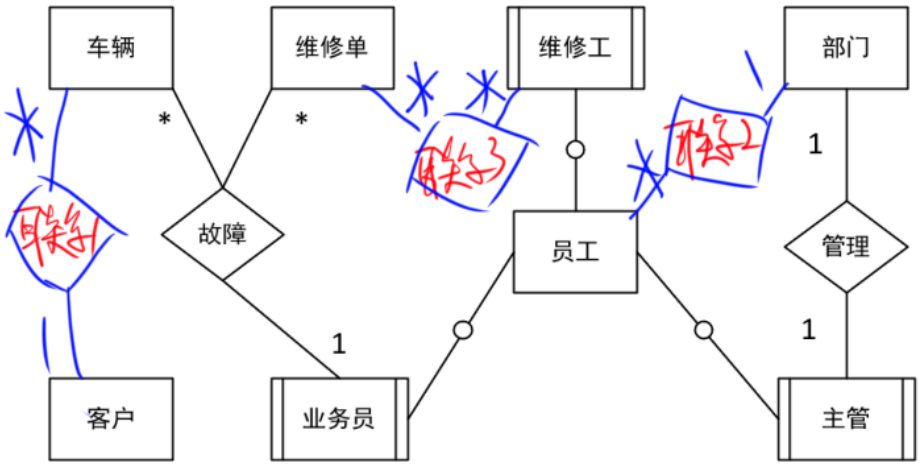

图2-1 实体联系图

这个联系类型会从 * ：1变成 * ：*
应该先将维修单关系中的维修工删除掉，然后将
这个联系转换成一个独立的关系模式，关系模式如下：

维修（维修单号，维修工，维修内容）

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。

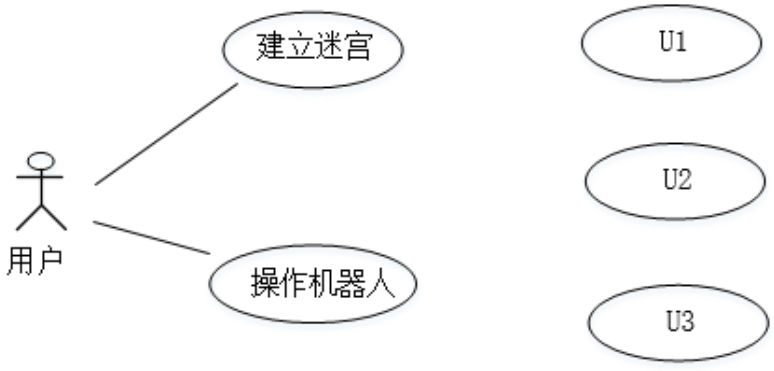

图3-1 用例图

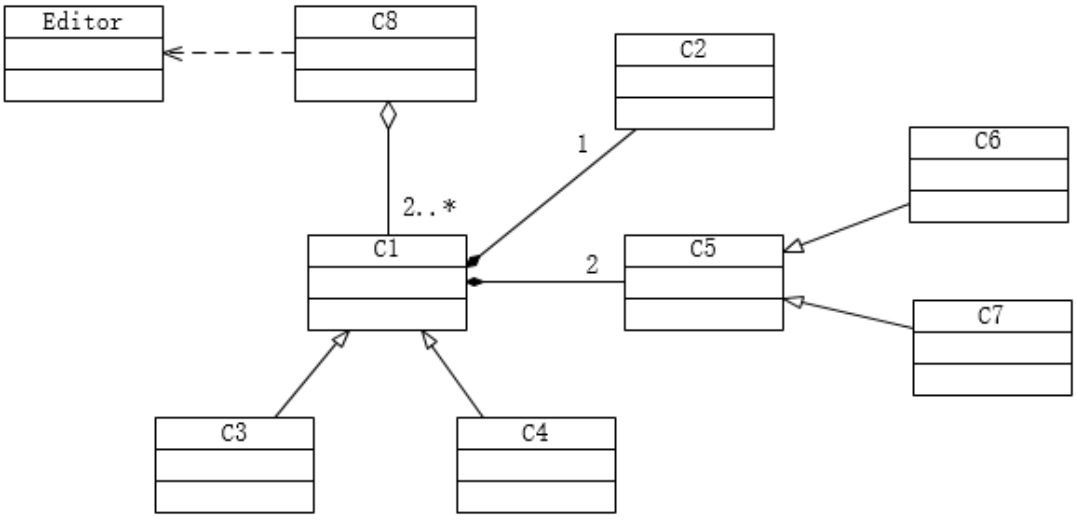

图3-2 类图

【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。

U1：编写迷宫文件　　U2：导入迷宫文件　　U3：设置计时器

【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？

U1、U2和建立迷宫是泛化关系，建立迷宫用例泛化U1、U2用例

U3和操作机器人是包含关系，操作机器人用例包含U3用例

【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。

C1：机器人（Robos）　　C2：计时器（Timer）
C3：小精灵（PacMan）　　C4：幽灵（Ghost）　　C3和C4可以互换
C5：传感器（Sensor）
C6：前向传感器（FrontSensor）
C7：近距离传感器（ProxiSensor）　　C6和C7可以互换
C8：迷宫（Maze）

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：

$$d[i][j]=\begin{cases} i & 若len2=0 \\ j & 若len1=0 \\ d[i-1][j-1] & 若str[i-1]=str2[j-1] \\ \min\{d[i-1][j]+1, d[i][j-1]+1, d[i-1][j-1]+1\} & 若str[i-1]\neq str2[j-1] \end{cases}$$

【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）。

（1） d[0][j] = j

（2） str1[i - 1] == str2[j - 1]

（3） d[i - 1][j - 1] + 1

（4） d[len1][len2]

【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）。

（5） 动态规划法

（6） O(mn) 或 O(m * n)

【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）___。

（7） 4

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。

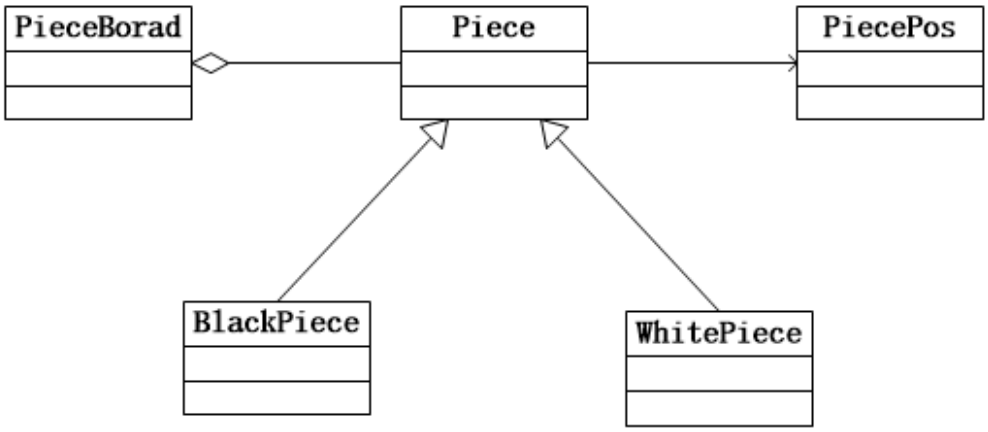

图6-1　类图

【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);
    }
}
```

(1)：public abstract void draw()

(2)：Piece

(3)：Piece

(4)：piece.draw()

(5)：piece.draw()