全国计算机技术与软件专业技术资格（水平）考试

2023 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. 在答题卡的指定位置上正确填写你的姓名和准考证号，并粘贴考生条形码。
2. 本试卷共 6 道题，试题一至试题四是必答题，试题五至试题六任选 1 道作答。每题 15 分，满分 75 分。
3. 解答时字迹务必清楚，字迹不清时，将不评分。
4. 仿照下面的例题，将解答写在答题纸的对应栏内。

（例题）

2023 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。

因为上半年考试的日期是“5 月 27 日”，故在答题纸的对应栏内写上“5”和“27”。（参看下表）

例题	解答栏
（1）	5
（2）	27

试题一至试题四为必答题

试题一（共 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】

随着农业领域科学种植的发展，需要对农业基地及农事进行信息化管理，为租户和农户等人员提供种植相关服务。现欲开发农事管理服务平台，其主要功能是：

（1）人员管理。平台管理员管理租户；租户管理农户并为其分配负责的地块，租户和农户以人员类型区分。

（2）基地管理。租户填写基地名称、地域等描述信息，在显示的地图上绘制地块。

（3）种植管理。租户设定作物及其从种植到采收的整个农事过程，包括农事活动及其实施计划，农户根据相应农事过程提醒进行农事活动并记录。系统会在设定时间向农户进行农事提醒，对逾期未实施活动向租户发出逾期警告。

（4）投入品管理。租户统一维护化肥，杀虫剂等投入品信息。农户在农事活动中设定投入品的实际消耗。

（5）信息服务。用户按查询条件发起农事信息请求，对相关地块农事活动实施情况（如与农事过程比对）等农事信息进行筛选、对比和统计等处理，并将响应信息进行展示。系统也给其他第三方软件提供 APP 接口，通过接口访问的方式，提供账号，密码和查询条件发起农事信息请求，返回特定格式的农事信息，无查询条件时默返回账号下所有信息，多查询条件时返回满足全部条件的信息。

现采用结构化方法对农事管理服务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。

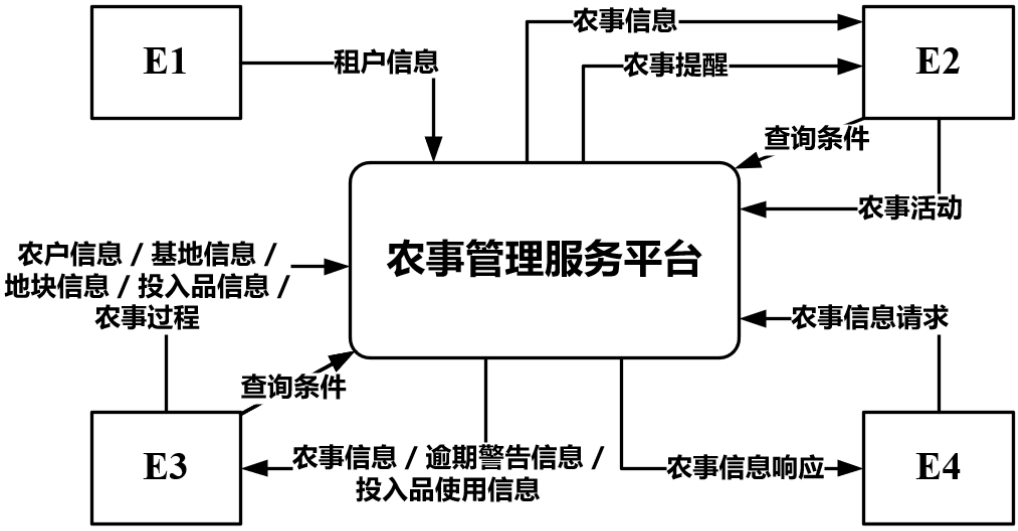


图1-1 上下文数据流图

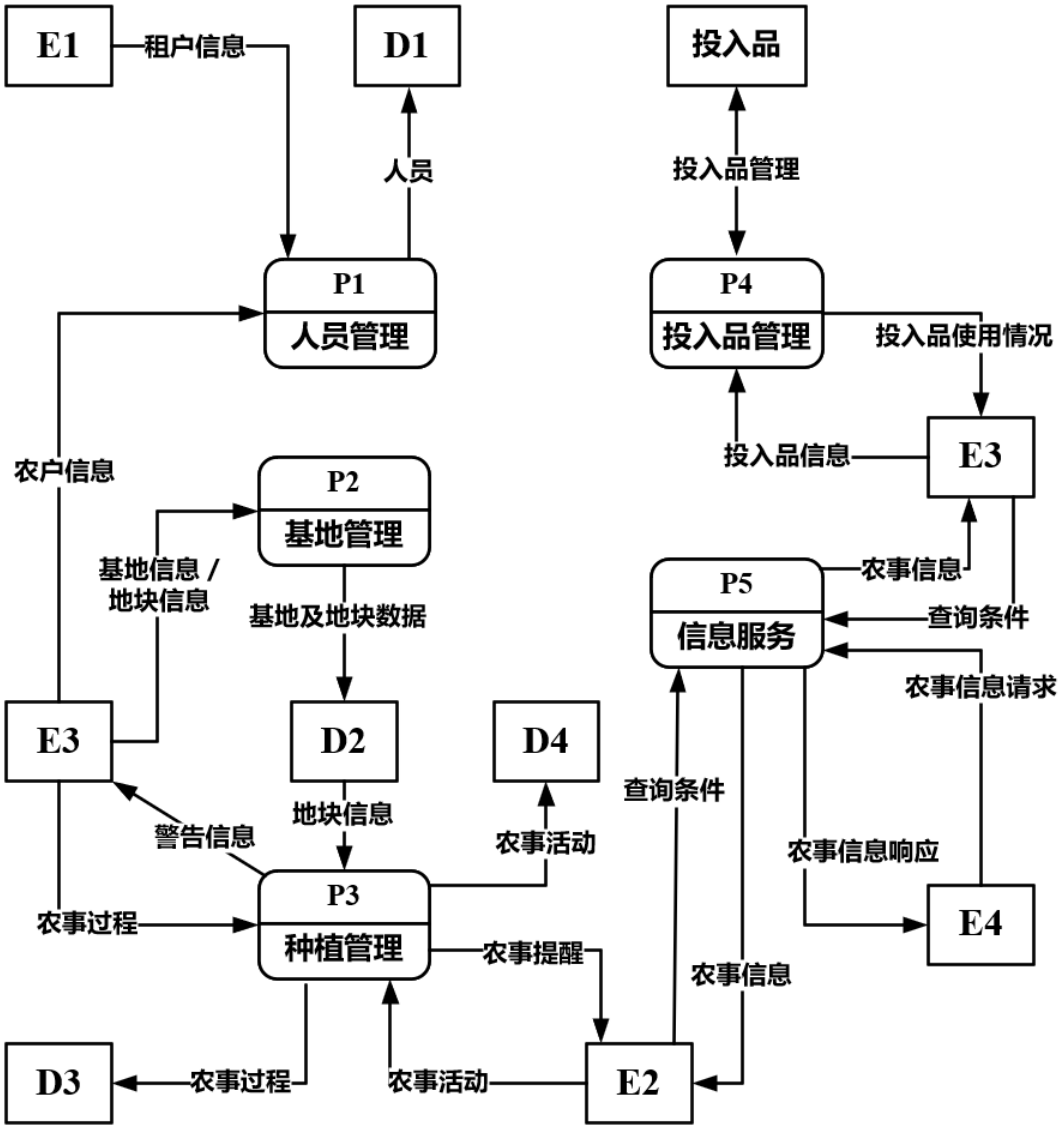


图1-2 0 层数据流图

【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。

【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。

【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，给出“农事信息请求”数据流的组成。

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】

某新能源汽车公司为了提升效率。需开发一个汽车零件采购系统。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）记录供应商的信息，包括供应商的名称，地址和一个电话。

（2）记录零件的信息，包括零件的编码，名称和价格。

（3）记录车型信息，包括车型的编号，名称和规格。

（4）记录零件采购信息，某个车型的某种零件可以从多家供应商采购，某种零件也可以被多个车型采用，某家供应商也可以供应多种零件，还包括采购数量和采购日期。

【概念结构设计】

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。

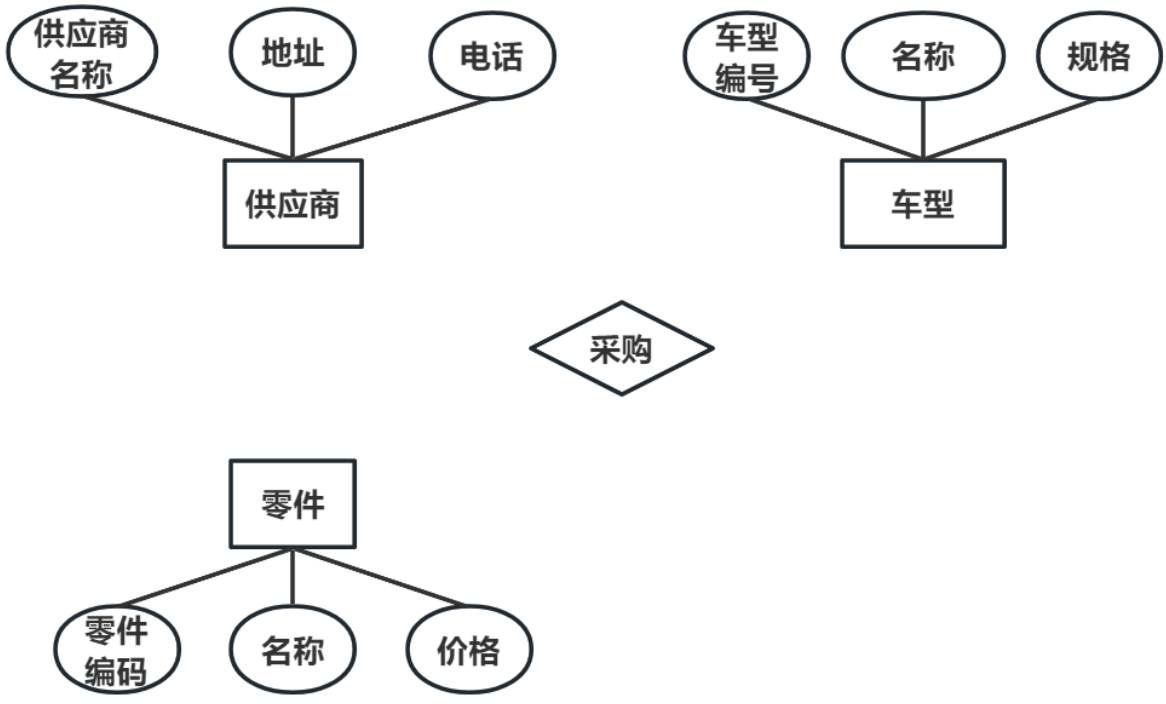


图 2-1 实体联系图

【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）：

供应商（供应商名称，地址，电话）

零件（零件编码，名称，价格）

车型（车型编号，名称，规格）

采购（车型编号，供应商名称，（a），（b），采购日期）

【问题 1】（5 分）

根据需求描述，补充图 2-1 的实体联系图（不增加新的实体）。

【问题 2】（3 分）

补充逻辑结构设计结果中的（a），（b）两处空缺，并标注主键和外键完整性约束。

【问题 3】（7 分）

该汽车公司现新增如下需求：记录车型在全国门店的销售情况，门店信息包括门店的编号，地址和电话，销售包括销售数量和销售日期等

对原有设计进行以下修改以实现该需求：

（1）在图 2-1 中体现门店信息及其车型销售情况，并标明新增的实体和联系及其主要属性。

（2）给出新增加的关系模式，并标注主键和外键完整性约束。

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】

某高校图书馆购买了若干学术资源的镜像数据库（MinorDBMino）资源，现要求开发一套数字图书馆（Digitallibrary）系统，面向校内用户（User）提供学术资源（Resoure）浏览，检索和下载服务。系统的主要功能描述如下：

（1）系统中存储了每个镜像数据库的基本信息，包括：数据库名称，访问地址，数据库属性以及数据库简介等信息，用户进入某个镜像数据库后，可以浏览检索以及下载其中的学术资源。

（2）学术资源包括会议论文（ConferencePaper）、期刑论文（JounalArticle）以及学位论文（Thesis）等；系统中存储了每个学术资源的题名、作者、发表时间、来源（哪个镜像数据库）、被引次数、下载次数等信息。对于会议论文，还需记录会议名称，召开时间以及召开地点；同一次会议的论文被收录在会议集（Proceeding）中。对于期刊论文，还需记录期刊名称，出版月份，期号以及主办单位；同一期号的论文被收录在一本期刊（Edition）中。对于学位论文，记录了学位类别（博士/硕士），毕业学校，专业及指导教师。会议集包含发表在该会议（在某个特定时间段，特定地点召开）上的所有文章。期刊的每一期在特定时间发行，其中包含若干篇文章。

（3）系统用户（User）包括在校学生（Student），教师（Teacher）以及其他在职人员（Staff）。用户使用学校的统一身份认证登录系统后，使用系统提供的各项服务。

（4）系统提供多种资源检索的方式，主要包括：按照资源的题名检索（SearchByTitle），按照作者名称检索（SearchByAuthor），按照来源检索（SearchBySource）等。

（5）用户可以下载资源，系统记录每个资源被下载的次数。

现采用面向对象分析与设计方法开发该系统，得到如图 3-1 所示的用例图以及图 3-2 所示的类图。

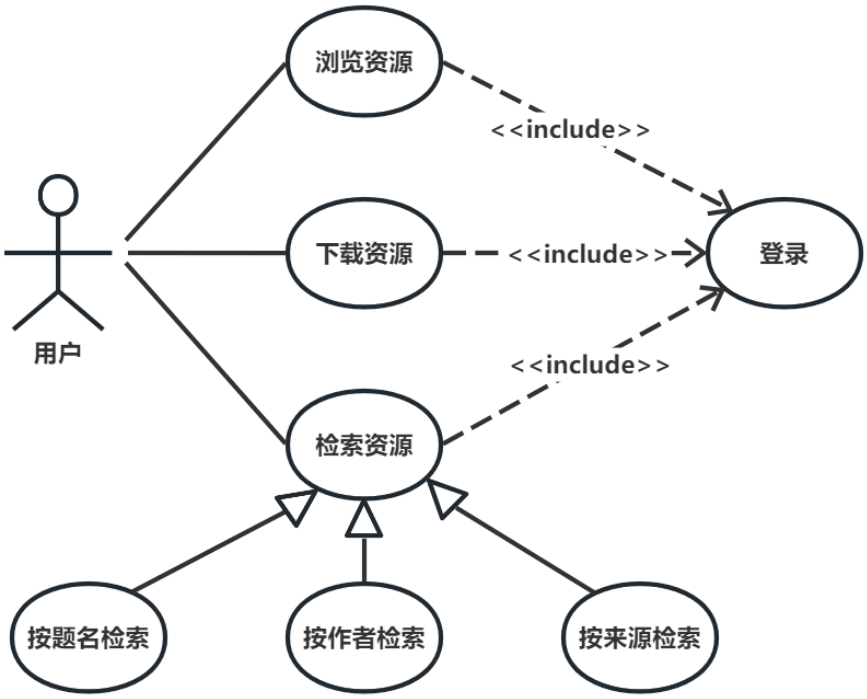


图 3-1 用例图

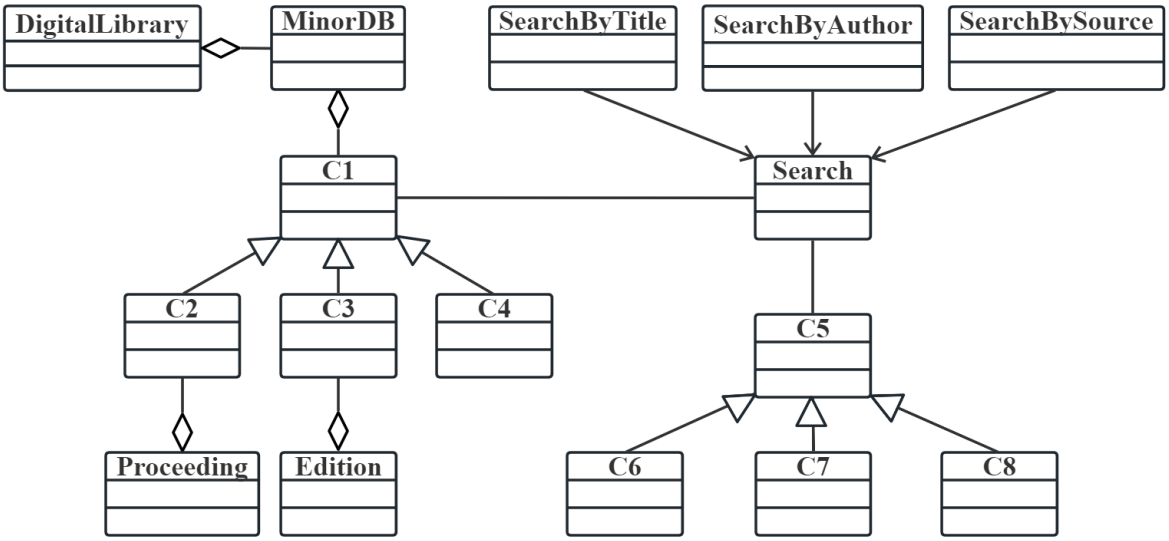


图 3-2 类图

【问题 1】（8 分）

根据说明中的描述，给出图 3-2 中 C1~C8 对应的类名。

【问题 2】（4 分）

根据说明中的描述，给出图 3-2 的类 C1~C4 的关键属性。

【问题 3】（3 分）

在该系统的开发过程中遇到了新的要求：用户能够在系统中对其所关注的数字资源注册他引通知，若该资源的他引次数发生变化，系候可以及时通知该用户，为了实现这个新的要求，可以在图 3-2 所示的类图中增加哪种设计模式？用 150 字以内文字解释选择该模式的原因。

试题四（共 15 分）

题目缺失

【C 代码】

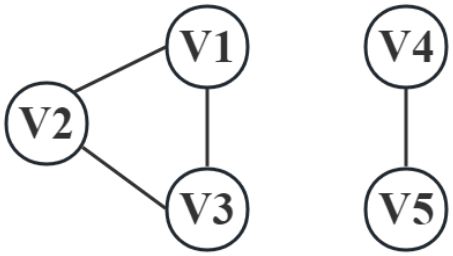
代码缺失

【问题 1】（10 分）

根据以上说明和 C 代码，填充 C 代码中的空（1）~（5）。

【问题 2】（5 分）

画出下图所示中无向图的邻接表。



从下列的 2 道试题（试题五至试题六）中任选 1 道解答。

请在答题纸上的指定位置将所选试题的题号框涂黑。

如多涂或者未涂题号框，则对题号最小的一道试题进行评分。

试题五 ☐　　**试题六** ☐

试题五（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】

在某系统中，类 Interval 代表由下界（lower bound）和上界（upper bound）定义的区间。要求采用不同的格式显示区间范围。如：

[lower bound,upper bound];[lower bound...upper bound];[lower bound-upper bound]等。

现采用策略（Strategy）模式实现该要求，得到如图 5-1 所示的类图。

图 5-1 缺失

【Java 代码】

```
enum TYPE {
    COMMA,
    DOTS,
    LINE
}

interface PrintStrategy {
    public __（1）__;
}
```

```
class Interval {
    private double lower;
    private double upper;

    public Interval(double lower, double upper) {
        this.lower = lower;
        this.upper = upper;
    }

    public double getLower() { return lower; }

    public double getUpper() { return upper; }

    public void printIntervals(PrintStrategy ptr) {
        （2）;
    }
}

class PrintIntervalsComma implements PrintStrategy {
    public void doPrint(Interval val) {
        System.out.println("[" + val.getLower() + "," + val.getUpper() + "]");
    }
}

class PrintIntervalsDots implements PrintStrategy {
    public void doPrint(Interval val) {
        System.out.println("[" + val.getLower() + "..." + val.getUpper() + "]");
    }
}
```

```
class PrintIntervalsLine implements PrintStrategy {
    public void doPrint(Interval val) {
        System.out.println("[" + val.getLower() + "-" + val.getUpper() + "]");
    }
}

public class Main {
    public static PrintStrategy getStrategy(TYPE type) {
        PrintStrategy st = null;
        switch (type) {
            case COMMA:
                （3）;
                break;
            case DOTS:
                （4）;
                break;
            case LINE:
                （5）;
                break;
        }
        return st;
    }

    public static void main(String[] args) {
        Interval a = new Interval(1.7, 2.1);
        a.printIntervals(getStrategy(TYPE.COMMA));
        a.printIntervals(getStrategy(TYPE.DOTS));
        a.printIntervals(getStrategy(TYPE.LINE));
    }
}
```