

## 2010年上半年 软件设计师 下午试卷 案例 [真题]

溯源编码：04521201908150258  
文档生成日期：2019年08月21日

PC+微信+纸质，立体化学习场景，陪伴你每时每刻。  
软考在线 <http://www.rkpass.cn>

.....

以下所有试题由 软考在线 免费智能真题库 提供  
软考在线 -- 最专业的一站式软考复习平台  
全网独家 免费智能真题库 定制学习计划  
专业致力于全国计算机技术与软件专业资格(水平)考试

使用说明：

溯源编码：

在软考在线PC版，“文档溯源”功能中，输入文档溯源编码，即可获知本文档是否为最新文档。

软考在线每天都会完善试题内容质量，更新试题统计数据。同时定期更新文档。

“文档溯源”功能位置：首页->复习资料->试题文档->文档溯源

二维码：

微信扫一扫，直达更多延伸内容。

打印：

文档已排好版，直接打印即可(A4纸)。

第1题 2010上

某大型企业的数据中心为了集中管理、控制用户对数据的访问并支持大量的连接需求，欲构建数据管理中间件，其主要功能如下：

(1) 数据管理员可通过中间件进行用户管理、操作管理和权限管理。用户管理维护用户信息，用户信息（用户名、密码）存储在用户表中；操作管理维护数据实体的标准操作及其所属的后端数据库信息，标准操作和后端数据库信息存放在操作表中；权限管理维护权限表，该表存储用户可执行的操作信息。

(2) 中间件验证前端应用提供的用户信息。若验证不通过，返回非法用户信息；若验证通过，中间件将等待前端应用提交操作请求。

(3) 前端应用提交操作请求后，中间件先对请求进行格式检查。如果格式不正确，返回格式错误信息；如果格式正确，则进行权限验证（验证用户是否有权执行请求的操作），若用户无权执行该操作，则返回权限不足信息，否则进行连接管理。

(4) 连接管理连接相应的后台数据库并提交操作，连接管理先检查是否存在空闲的数据库连接，如果不存在，新建连接；如果存在，则重用连接。

(5) 后端数据库执行操作并将结果传给中间件，中间件对收到的操作结果进行处理后，将其返回给前端应用。

现采用结构化方法对系统进行分析与设计，获得如图1-1所示的顶层数据流图和图1-2所示的0层数据流图。



本题

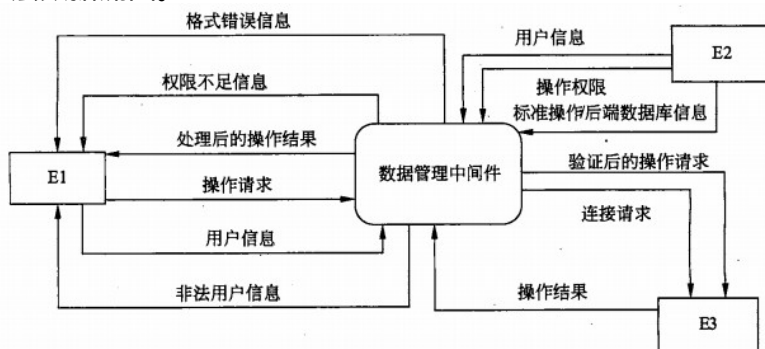


图 1-1 顶层数据流图

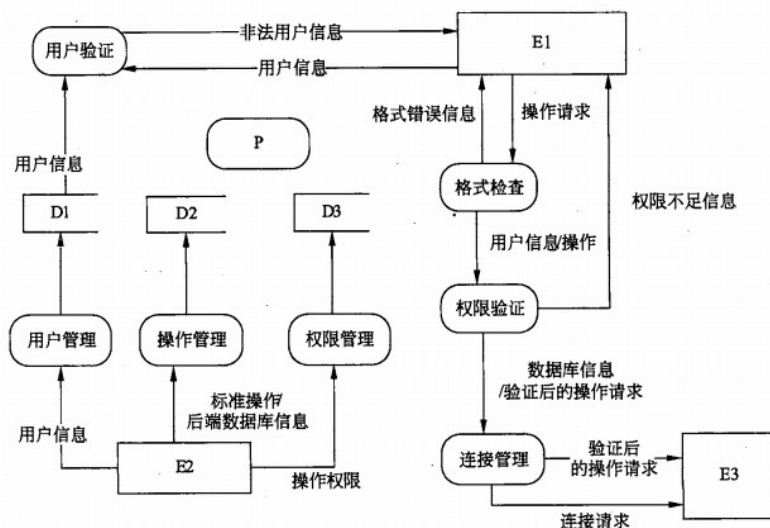


图 1-2 0层数据流图

问题1.1 使用说明中的词语，给出图1-1中的实体E1、E3的名称。

问题1.2 使用说明中的词语，给出图1-2中的数据存储D1&#12316;D3的名称。

问题1.3 给出图1-2中加工P的名称及其输入、输出流。

	名 称	起 点	终 点
输入流			P
输出流		P	

除加工P的输入与输出流外，图1-2还缺失了两条数据流，请给出这两条数据流的起点和终点。

起 点	终 点

注：名称使用说明中的词汇，起点和终点均使用图 1-2 中的符号或词汇。

问题1.4 在绘制数据流图时，需要注意加工的绘制。请给出三种在绘制加工的输入、输出时可能出现的错误。

第2题 2010上

某学校拟发一套实验管理系统，对各课程的实验安排情况进行管理。

【需求分析】

一个实验室可进行多种类型不同的实验。由于实验室和实验员资源有限，需根据学生人数分批次安排实验室和实验员。一门课程可以为多个班级开设，每个班级每学期可以开设多门课程。一门课程的一种实验可以根据人数、实验室的可容纳人数和实验类型，分批次开设在多个实验室的不同时间段。一个实验室的一次实验可以分配多个实验员负责辅导实验，实验员给出学生的每次实验成绩。

(1)课程信息包括：课程编号、课程名称、实验学时、授课学期和开课的班级等信息；实验信息记录该课程的实验进度信息，包括：实验名、实验类型、学时、安排周次等信息，如表2-1所示。

表 2-1 课程及实验信息

课程编号	15054037	课程名称	数字电视原理	实验学时	12
班级	电 0501,信 0501,计 0501	授课院系	机械与电气工程	授课学期	第三学期
序号	实验名	实验类	难度	学时	安排周次
1505403701	音视频 AD-DA 实验	验证性	1	2	3
1505403702	音频编码实验	验证性	2	2	5
1505403703	视频编码实验	演示性	0.5	1	9

(2)以课程为单位制定实验安排计划信息，包括：实验地点，实验时间、实验员等信息，实验计划如表2-2所示。

表 2-2 实验安排计划

课程编号	15054037	课程名称	数字电视原理	安排学期	2009 年秋	总人数	220
实验编号	实验名	实验员	实验时间	地点	批次号	人数	
1505403701	音视频 AD-DA 实验	盛×，陈×	第 3 周周四晚上	实验三楼 310	1	60	
1505403701	音视频 AD-DA 实验	盛×，陈×	第 3 周周四晚上	实验三楼 310	2	60	
1505403701	音视频 AD-DA 实验	吴×，刘×	第 3 周周五晚上	实验三楼 311	3	60	
1505403701	音视频 AD-DA 实验	吴×	第 3 周周五晚上	实验三楼 311	4	40	
1505403702	音频编码实验	盛×，刘×	第 5 周一一下午	实验四楼 410	1	70	

(3)由实验员给出每个学生每次实验的成绩，包括：实验名、学号、姓名、班级、实验成绩等信息，实验成绩如表2-3所示。

表 2-3 实验成绩

实验员： 盛×

实验名	音视频 AD-DA 实验	课程名	数字电视原理
学号	姓名	班级	实验成绩
030501001	陈民	信 0501	87
030501002	刘志	信 0501	78
040501001	张勤	计 0501	86



本题

(4)学生的实验课程总成绩根据每次实验的成绩以及每次实验的难度来计算。

【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。

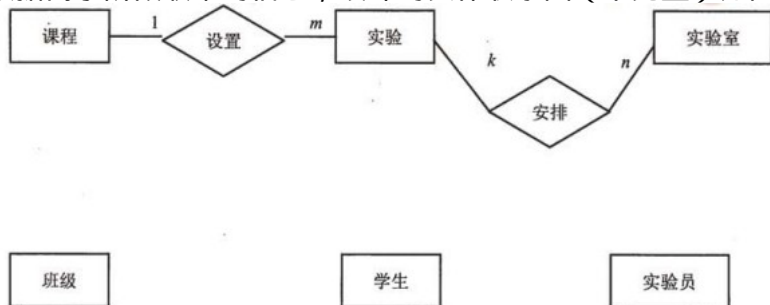


图 2-1 实体联系图

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）：

课程（课程编号，课程名称，授课院系，实验学时）

班级（班级号，专业，所属系）

开课情况（          (1)          ，授课学期）

实验（          (2)          ，实验类型，难度，学时，安排周次）

实验计划（          (3)          ，实验时间，人数）

实验员（          (4)          ，级别）

实验室（实验室编号，地点，开放时间，可容纳人数，实验类型）

学生（          (5)          ，姓名，年龄，性别）

实验成绩（          (6)          ，实验成绩，评分实验员）

问题2.1 补充图2-1中的联系和联系的类型。

问题2.2 根据图2-1，将逻辑结构设计阶段生成的关系模式中的空（1）~（6）补充完整并用下划线指出这六个关系模式的主键。

问题2.3 如果需要记录课程的授课教师，新增加“授课教师”实体。请对图2-1进行修改，画出修改后的实体间联系和联系的类型。

第3题 2010上

某运输公司决定为新的售票机开发车票销售的控制软件。图3-1给出了售票机的面板示意图以及相关的控制部件。

售票机相关部件的作用如下所述：

- (1)目的地键盘用来输入行程目的地的代码（例如，200表示总站）。
- (2)乘客可以通过车票键盘选择车票种类（单程票、多次往返票和座席种类）。
- (3)继续/取消键盘上的取消按钮用于取消购票过程，继续按钮允许乘客连续购买多张票。
- (4)显示屏显示所有的系统输出和用户提示信息。
- (5)插卡口接受MCard（现金卡），硬币口和纸币槽接受现金。
- (6)打印机用于输出车票。



本题

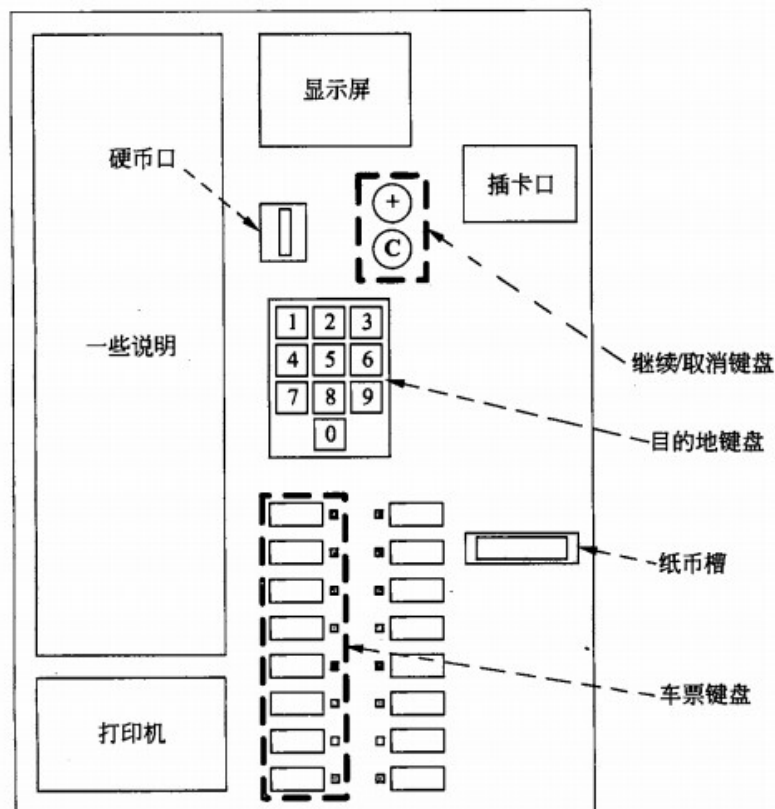


图 3-1 售票机面板示意图

假设乘客总是支付恰好需要的金额而无需找零，售票机的维护工作（取回现金、放入空白车票等）由服务技术人员完成。

系统采用面向对象方法开发/使用UML进行建模。系统的顶层用例图和类图分别如图3-2和图3-3所示。

系统采用面向对象方法开发/使用UML进行建模。系统的顶层用例图和类图分别如图3-2和图3-3所示。

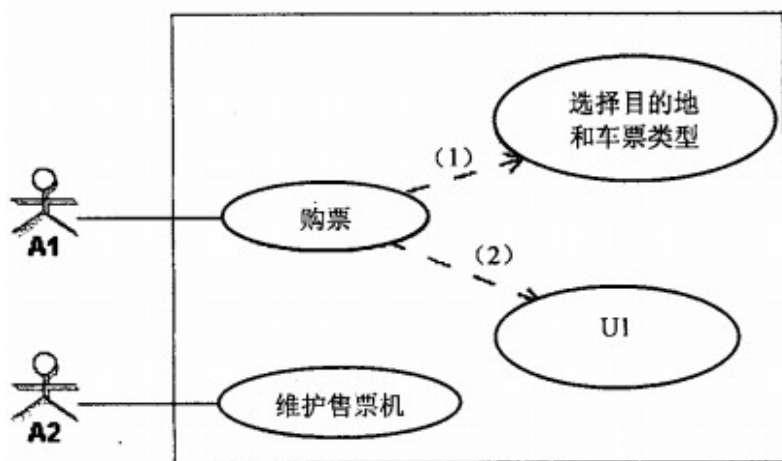


图 3-2 顶层用例图

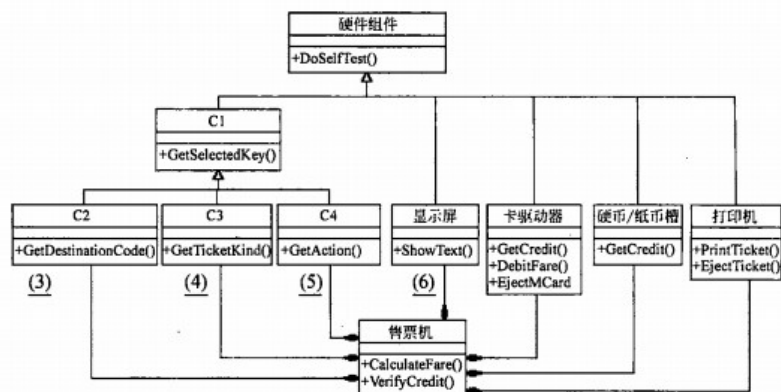


图 3-3 类图

问题3.1 根据说明中的描述，给出图3-2中A1和A2所对应的参与者，U1所对应的用例，以及(1)、(2)处所对应的关系。

问题3.2 根据说明中的描述，给出图3-3中缺少的C1和C4所对应的类名以及 (3)和(6)处所对应的多重度。

问题3.3 图3-3中的类图设计采用了中介者 (Mediator)设计模式，请说明该模式的内涵。

#### 第4题 2010上

对有向图进行拓扑排序的方法是：

- (1) 初始时拓扑序列为空；
- (2) 任意选择一个入度为0的顶点，将其放入拓扑序列中，同时从图中删除该顶点 以及从该顶点出发的弧；
- (3) 重复 (2)，直到不存在入度为0的顶点为止（若所有顶点都进入拓扑序列则完成拓扑排序，否则由于有向图中存在回路无法完成拓扑排序）。

函数int\* TopSort(LinkedDigraph G)的功能是对有向图G中的顶点进行拓扑排序，返回拓扑序列中的顶点编号序列，若不能完成拓扑排序，则返回空指针。其中，图G中的顶点从1开始依次编号，顶点序列为v1, v2,&hellip;, vn,图G采用邻接表表示，其数据类型定义如下：

```
#define MAXVNUM 50          /* 最大顶点数 */
typedef struct ArcNode{     /* 表结点类型 */
    int adjvex;             /* 邻接顶点编号 */
    struct ArcNode *nextarc; /* 指示下一个邻接顶点 */
}ArcNode;
typedef struct AdjList{     /* 头结点类型 */
    char vdata;             /* 顶点的数据信息 */
    ArcNode *firstarc;      /* 指向邻接表的第一个表结点 */
}AdjList;
typedef struct LinkedDigraph{ /* 图的类型 */
    int n;                  /* 图中顶点个数 */
    AdjList Vhead[MAXVNUM]; /* 所有顶点的头结点数组 */
}LinkedDigraph;
```

例如，某有向图G如图4-1所示，其邻接表如图4-2所示。



本题

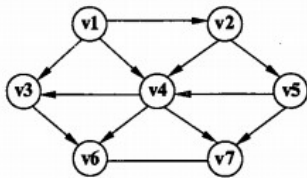


图 4-1 有向图 G

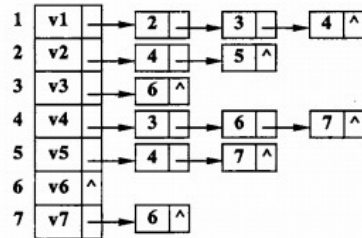


图 4-2 有向图 G 的邻接表示意图

函数TopSort中用到了队列结构（Queue的定义省略），实现队列基本操作的函数原型如下表所示：

函数原型	说明
void InitQueue(Queue *Q)	初始化队列（构造一个空队列）
bool IsEmpty(Queue Q)	判断队列是否为空，若是则返回 true，否则返回 false
void EnQueue(Queue *Q, int e)	元素入队列
void DeQueue(Queue *Q, int *p)	元素出队列

### 【C代码】

```

int *TopSort(LinkedDigraph G) {
    ArcNode *p;          /* 临时指针，指示表结点 */
    Queue Q;              /* 临时队列，保存入度为 0 的顶点编号 */
    int k = 0;             /* 临时变量，用作数组元素的下标 */
    int j = 0, w = 0;      /* 临时变量，用作顶点编号 */
    int *topOrder, *inDegree;
    topOrder = (int *)malloc((G.n+1) * sizeof(int));
    /* 存储拓扑序列中的顶点编号 */
    inDegree = (int *)malloc((G.n+1) * sizeof(int));
    /* 存储图 G 中各顶点的入度 */
    if (!inDegree || !topOrder) return NULL;

    (1);                  /* 构造一个空队列 */

    for ( j = 1; j <= G.n; j++ ) { /* 初始化 */
        topOrder[j] = 0;    inDegree[j] = 0;
    }

    for ( j = 1; j <= G.n; j++ ) /* 求图 G 中各顶点的入度 */
        for ( p = G.Vhead[j].firstarc; p; p = p->nextarc )
            inDegree[p->adjvex] += 1;

    for ( j = 1; j <= G.n; j++ ) /* 将图 G 中入度为 0 的顶点保存在队列中 */
        if ( 0 == inDegree[j] )    EnQueue(&Q, j);

    while (!IsEmpty(Q)) {
        (2);                  /* 队头顶点出队列并用 w 保存该顶点的编号 */
        topOrder[k++] = w;
        /* 将顶点 w 的所有邻接顶点的入度减 1（模拟删除顶点 w 及从该顶点出发的弧的操作） */
        for(p = G.Vhead[w].firstarc; p; p = p->nextarc) {
            (3) -= 1;
            if (0 == (4))    EnQueue(&Q, p->adjvex);
        } /* for */
    } /* while */

    free(inDegree);

    if ( (5) )
        return NULL;
    return topOrder;
} /*TopSort*/

```

问题4.1 根据以上说明和C代码，填充C代码中的空（1）&#12316;（5）。



问题4.2 对于图4-1所示的有向图G,写出函数TopSort执行后得到的拓扑序列。若将函数 TopSort中的队列改为栈 , 写出函数TopSort执行后得到的拓扑序列。

问题4.3 设某有向无环图的顶点个数为n、弧数为e , 那么用邻接表存储该图时 , 实现上述拓扑排序算法的函数TopSort的时间复杂度是 ( 6 )。

若有向图采用邻接矩阵表示 ( 例如 , 图4-1所示有向图的邻接矩阵如图4-3所示 ) , 且将函数TopSort中有关邻接表的操作修改为针对邻接矩阵的操作 , 那么对于有n个顶点、e条弧的有向无环图 , 实现上述拓扑排序算法的时间复杂度是 ( 7 )。

	v1	v2	v3	v4	v5	v6	v7
v1	0	1	1	1	0	0	0
v2	0	0	0	1	1	0	0
v3	0	0	0	0	0	1	0
v4	0	0	1	0	0	1	1
v5	0	0	0	1	0	0	1
v6	0	0	0	0	0	0	0
v7	0	0	0	0	0	1	0

图 4-3 有向图 G 的邻接矩阵

第5题 2010上

某软件公司现欲开发一款飞机飞行模拟系统，该系统主要模拟不同种类飞机的飞行特征与起飞特征。需要模拟的飞机种类及其特征如表5-1所示。

表 5-1		
飞机种类	起飞特征	飞行特征
直升机 (Helicopter)	垂直起飞 (VerticalTakeOff)	亚音速飞行 (SubSonicFly)
客机 (AirPlane)	长距离起飞 (LongDistanceTakeOff)	亚音速飞行 (SubSonicFly)
歼击机 (Fighter)	长距离起飞 (LongDistanceTakeOff)	超音速飞行 (SuperSonicFly)
鹞式战斗机 (Harrier)	垂直起飞 (VerticalTakeOff)	超音速飞行 (SuperSonicFly)



本题

为支持将来模拟更多种类的飞机，采用策略设计模式 ( Strategy)设计的类图如图5-1所示。 .. 图5-1中，AirCraft为抽象类，描述了抽象的飞机，而类Helicopter、AirPlane、Fighter 和 Harrier分别描述具体的飞机种类，方法flyO和takeOff() 分别表示不同飞机都具有飞行特征和起飞特征；类FlyBehavior与TakeOffleavior为抽象类，分别用于表示抽象的飞行行为与起飞行为；类SubSonicFly与SuperSonicFly分别描述亚音速飞行和超音速飞行的行为；类 VerticalTakeOff与LongDistanceTakeOff分别描述垂直起飞与长距离起飞的行为。

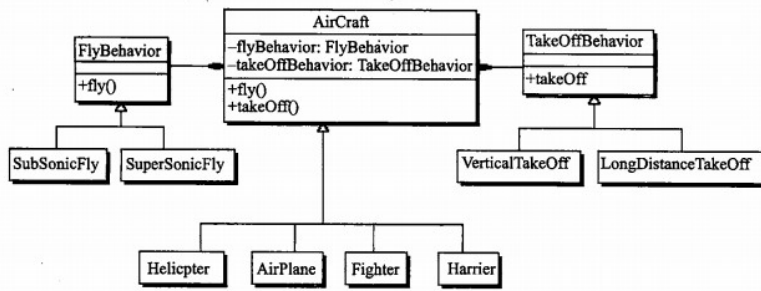


图 5-1 类图

## 问题5.1

### 【C++代码】

```

#include<iostream>
using namespace std;
class FlyBehavior {
public: virtual void fly() = 0;
};
class SubSonicFly:public FlyBehavior{
public: void fly(){ cout << "亚音速飞行 !" << endl; }
};
class SuperSonicFly:public FlyBehavior{
public: void fly(){ cout << "超音速飞行 !" << endl; }
};

class TakeOffBehavior {
public: virtual void takeOff() = 0;
};
class VerticalTakeOff:public TakeOffBehavior{
public: void takeOff(){ cout << "垂直起飞 !" << endl; }
};
class LongDistanceTakeOff:public TakeOffBehavior {
public: void takeOff () { cout << "长距离起飞 !" << endl; }
};

class Aircraft{
protected:
    (1);

```

```

    (2);
public:
    void fly(){ (3); }
    void takeOff() { (4); };
};
class Helicopter: public Aircraft {
public:
    Helicopter () {
        flyBehavior = new (5);
        takeOffBehavior = new (6);
    }
    (7) {
        if(!flyBehavior) delete flyBehavior;
        if(!takeOffBehavior) delete takeOffBehavior;
    }
};
//其他代码省略

```

第6题 2010上

某软件公司现欲开发一款飞机飞行模拟系统，该系统主要模拟不同种类飞机的飞行特征与起飞特征。需要模拟的飞机种类及其特征如表6-1所示。

表 6-1

飞机种类	起飞特征	飞行特征
直升机 (Helicopter)	垂直起飞 (VerticalTakeOff)	亚音速飞行 (SubSonicFly)
客机 (AirPlane)	长距离起飞 (LongDistanceTakeOff)	亚音速飞行 (SubSonicFly)
歼击机 (Fighter)	长距离起飞 (LongDistanceTakeOff)	超音速飞行 (SuperSonicFly)
鹞式战斗机 (Harrier)	垂直起飞 (VerticalTakeOff)	超音速飞行 (SuperSonicFly)



本题

为支持将来模拟更多种类的飞机，采用策略设计模式 ( Strategy)设计的类图如图6-1所示。图6-1中，AirCRAFT为抽象类，描述了抽象的飞机，而类Helicopter、AirPlane、Fighter和Harrier分别描述具体的飞机种类，方法fly()和takeOff()分别表示不同飞机都具有飞行特征和起飞特征；类FlyBehavior与TakeOffBehavior为抽象类，分别用于表示抽象的飞行行为与起飞行为；类SubSonicFly与SuperSonicFly分别描述亚音速飞行和超音速飞行的行为；类VerticalTakeOff与LongDistanceTakeOff分别描述垂直起飞与长距离起飞的行为。

问题6.1

【Java 代码】

```

interface FlyBehavior {
    public void fly();
};
class SubSonicFly implements FlyBehavior{
    public void fly(){ System.out.println("亚音速飞行！"); }
};
class SuperSonicFly implements FlyBehavior{
    public void fly(){ System.out.println("超音速飞行！"); }
};
    
```

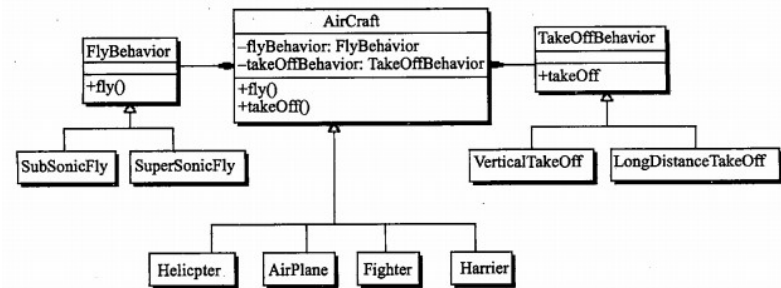


图 6-1 类图

```

interface TakeOffBehavior {
    public void takeOff();
};

class VerticalTakeOff implements TakeOffBehavior {
    public void takeOff () { System.out.println("垂直起飞！"); }
};

class LongDistanceTakeOff implements TakeOffBehavior {
    public void takeOff () { System.out.println("长距离起飞！"); }
};
    
```

```
abstract class AirCraft {  
    protected ____ (1) ____;  
    protected ____ (2) ____;  
    public void fly() { ____ (3) ____; }  
    public void takeOff() { ____ (4) ____; }  
};  
class Helicopter ____ (5) ____ AirCraft {  
    public Helicopter () {  
        flyBehavior = new ____ (6) ____;  
        takeOffBehavior = new ____ (7) ____;  
    }  
};  
//其他代码省略
```