2012 年上半年软件设计师考试下午真题（参考答案）

● 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】

某学校开发图书管理系统，以记录图书馆藏图书及其借出和归还情况，提供给借阅者借阅图书功能，提供给图书馆管理员管理和定期更新图书表功能。主要功能的具体描述如下：

（1）处理借阅。借阅者要借阅图书时，系统必须对其身份（借阅者 ID）进行检查。通过与教务处维护的学生数据库、人事处维护的职工数据库中的数据进行比对，以验证借阅者 ID 是否合法，若合法，则检查借阅者在逾期未还图书表中是否有逾期未还图书，以及罚金表中的罚金是否超过限额。如果没有逾期未还图书并且罚金未超过限额，则允许借阅图书，更新图书表，并将借阅的图书存入借出图书表，借阅者归还所借图书时，先由图书馆管理员检查图书是否缺失或损坏，若是，则对借阅者处以相应罚金并存入罚金表；然后，检查所还图书是否逾期，若是，执行“处理逾期”操作；最后，更新图书表，删除借出图书表中的相应记录。

（2）维护图书。图书馆管理员查询图书信息；在新进图书时录入图书信息，存入图书表；在图书丢失或损坏严重时，从图书表中删除该图书记录。

（3）处理逾期。系统在每周一统计逾期未还图书，逾期未还的图书按规则计算罚金，并记入罚金表，并给有逾期未还图书的借阅者发送提醒消息。借阅者在借阅和归还图书时，若罚金超过限额，管理员收取罚金，并更新罚金表中的罚金额度。

现采用结构化方法对该图书管理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。

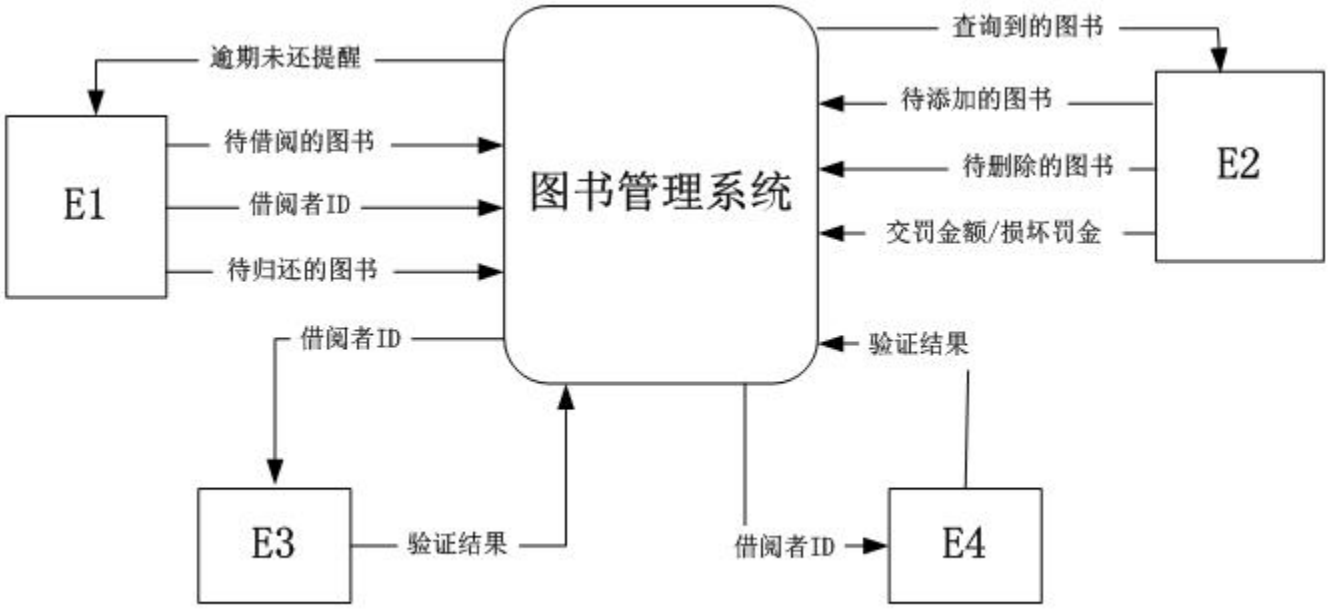

图 1-1 顶层数据流图

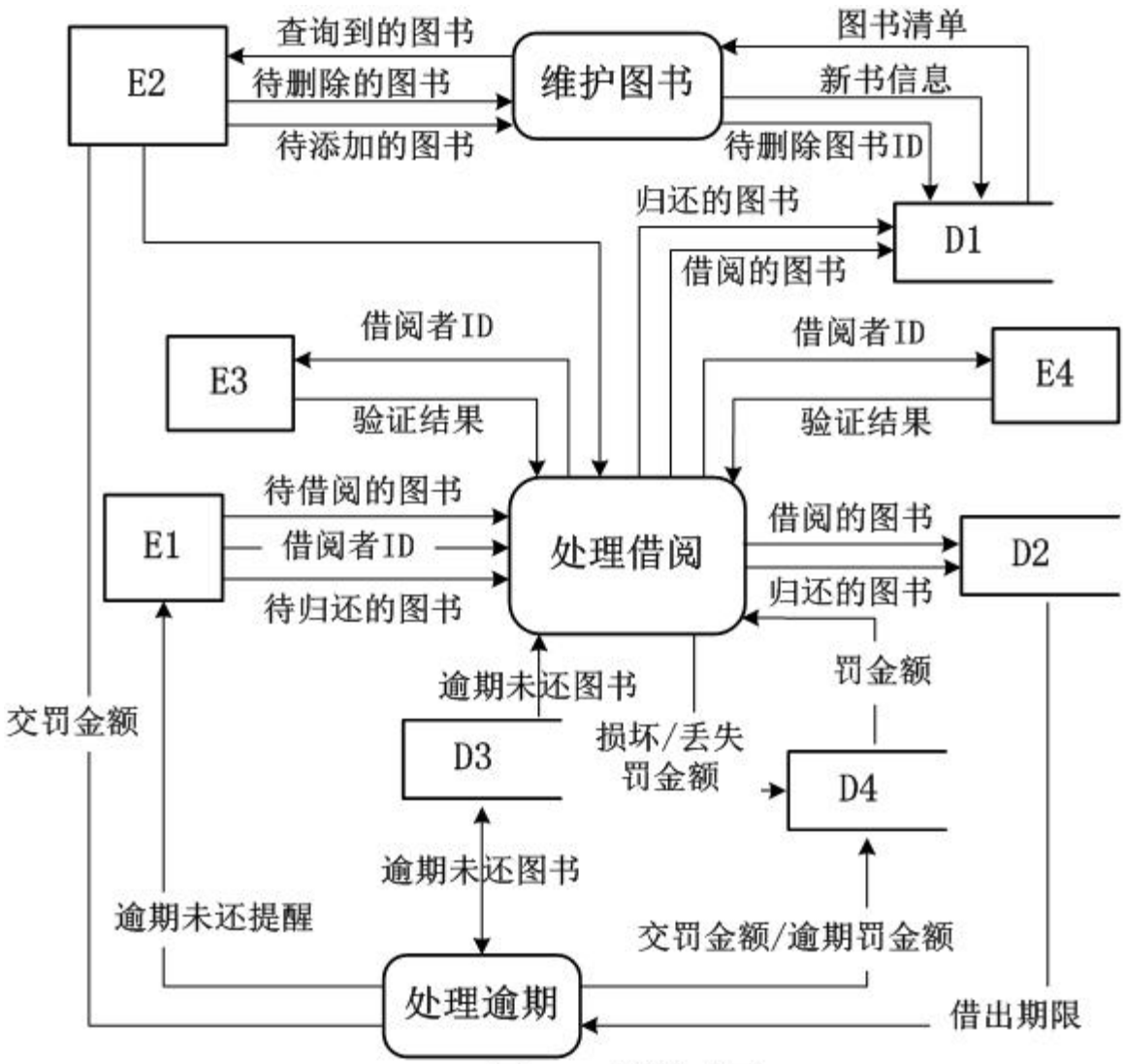

图 1-2　0 层数据流图

【问题 1】

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】

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】

在 DFD 建模时，需要对有些复杂加工（处理）进行进一步精化，绘制下层数据流图。针对图 1-2 中的加工“处理借阅”，在 1 层数据流图中应分解为哪些加工？（使用说明中的术语）

【问题 4】

说明【问题 3】中绘制 1 层数据流图时要注意的问题。

● 阅读下列说明和图，回答问题 1 至问题 3，将解答填入答题纸的对应栏内。

【说明】

某医院拟开发一套住院病人信息管理系统，以方便对住院病人、医生、护士和手术等信息进行

管理。

【需求分析】

（1）系统登记每个病人的住院信息，包括：病案号、病人的姓名、性别、地址、身份证号、电话号码、入院时间及病床等信息，每个病床有唯一所属的病区及病房，如表 2-1 所示。其中病案号唯一标识病人本次住院的信息。

表 2-1 住院登记表

病案号	071002286	姓名	张三	性别	男
登记证号	0102106701001234	入院时间	2011-05-03	病床号	0523401
病房	0524室	病房类型	三人间	所属病区	0520区

（2）在一个病人的一次住院期间，由一名医生对该病人的病情进行诊断，并填写一份诊断书，如表 2-2 所示。对于需要进行一次或多次手术的病人，系统记录手术名称、手术室、手术日期、手术时间、主刀医生及多名协助医生，每名医生在手术中的责任不同，如表 2-3 所示，其中手术室包含手术室号、楼层、地点和类型等信息。

表 2-2 诊断书

病案号	071002286	姓名	张三	性别	男	医生	李某某
诊断							

表 2-3 手术安排表

手术名称	某某手术	病案号	071002286	姓名	张三	性别	男
手术室	032501	手术日期	2011-05-15	手术时间	8:30-10:30	主刀医生	李**
协助医生	医**（协助）,曾**（协助）,刘**（协助）,霍**（麻醉）						

（3）护士分为两类：病床护士和手术室护士。每个病床护士负责护理一个病区内的所有病人，每个病区由多名护士负责护理。手术室护士负责手术室的护理工作。每个手术室护士负责多个手术室，每个手术室由多名护士负责，每个护士在手术室中有不同的责任，并由系统记录其责任。

【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。

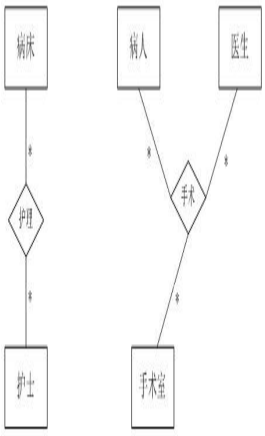

图2-1 实体联系图

【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）：

病床（病床号，病房，病房类型，所属病区）

护士（护士编号，姓名，类型，性别，级别）

病房护士（　（1）　）

手术室（手术室号，楼层，地点，类型）

手术室护士（　（2）　）

病人（　（3），姓名，性别，地址，身份证号，电话号码，入院时间）

医生（医生编号，姓名，性别，职称，所属科室）

诊断书（　（4），诊断，诊断时间）

手术安排（病案号，手术室号，手术时间，手术名称）

手术医生安排（　（5），医生责任）

【问题 1】（6 分）

补充图 2-1 中的联系和联系的类型。

【问题 2】（5 分）

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）～（5）补充完整，并用下划线指出主键。

【问题 3】（4 分）

如果系统还需要记录医生给病人的用药情况，即记录医生给病人所开处方中药品的名称、用量、价格、药品的生产厂家等信息。请根据该要求，对图 2-1 进行修改，画出补充后的实体、实体间联系和联系的类型。

● 阅读下列说明和图，回答问题 1 至问题 3，将解答填入答题纸的对应栏内。

【说明】

某网上购物平台的主要功能如下：

（1）创建订单。顾客（ Customer）在线创建订单（Order），主要操作是向订单中添加项目、从订单中删除项目。订单中应列出所订购的商品（Product）及其数量（quantities）。

（2）提交订单。订单通过网络来提交。在提交订单时，顾客需要提供其姓名（name）、收货地址（address）、以及付款方式（form of payment）（预付卡、信用卡或者现金）。为了制定送货计划以及安排送货车辆，系统必须确定订单量（volume）。除此之外，还必须记录每种商品的名称（Name）、造价（cost price）、售价（sale price）以及单件商品的包装体积（cubic volume）。

（3）处理订单。订单处理人员接收来自系统的订单；根据订单内容，安排配货，制定送货计

划。在送货计划中不仅要指明发货日期（delivery date），还要记录每个订单的限时发送要求（Delivery Time Window）。

（4）派单。订单处理人员将已配好货的订单转交给派送人员。

（5）送货／收货。派送人员将货物送到顾客指定的收货地址。当顾客收货时，需要在运货单（delivery slip）上签收。签收后的运货单最终需交还给订单处理人员。

（6）收货确认。当订单处理人员收到签收过的运货单后，会和顾客进行一次再确认。

现采用面向对象方法开发上述系统，得到如图 3-1 所示的用例图和图 3-2 所示的类图。

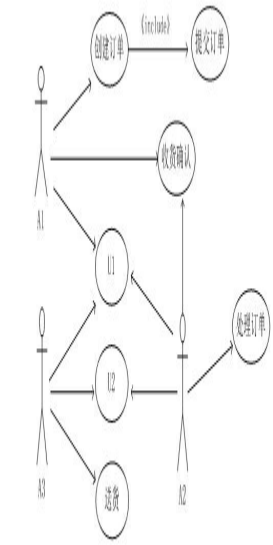

图 3-1 用例图

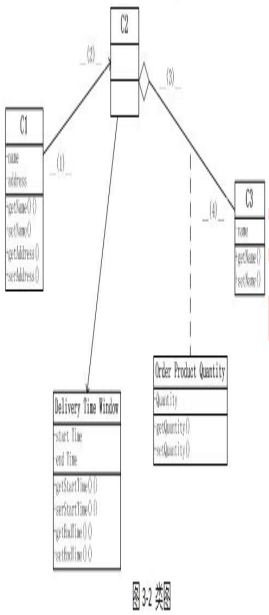

图 3-2 类图

【问题 1】

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～U2 处所对应的用例名称。

【问题 2】

根据说明中的描述，给出图 3-2 中 C1～C3 所对应的类名以及（1）～（4）处所对应的多重度（类名使用说明中给出的英文词汇）。

【问题 3】

根据说明中的描述，将类 C2 和 C3 的属性补充完整（属性名使用说明中给出的英文词汇）。

● 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】

用两台处理机 A 和 B 处理 n 个作业。设 A 和 B 处理第 i 个作业的时间分别为 ai 和 bi。由于各个作业的特点和机器性能的关系，对某些作业，在 A 上处理时间长，而对某些作业在 B 上处理时间长。一台处理机在某个时刻只能处理一个作业，而且作业处理是不可中断的，每个作业只能被处理一次。现要找出一个最优调度方案，使得 n 个作业被这两台处理机处理完毕的时间（所有作业被处理的时间之和）最少。

算法步骤：

（1）确定候选解上界为 R 短的单台处理机处理所有作业的完成时间 m，

$$m = \min\left(\sum_{i=1}^{n} a_i, \sum_{i=1}^{n} b_i\right)$$

（2）用 p（x，y，k）=1 表示前 k 个作业可以在 A 用时不超过 x 且在 B 用时不超过 y 时间 内处理完成，则 p（x，y，k）=p（x-ak，y，k-1）‖p（x，y-bk，k-1）（‖表示逻辑或操作）。

（3）得到最短处理时间为 min（max（x，y））。

【C 代码】

下面是该算法的 C 语言实现。

（1）常量和变量说明

n: 作业数

m: 候选解上界

a: 数组，长度为 n，记录 n 个作业在 A 上的处理时间，下标从 0 开始

b: 数组，长度为 n，记录 n 个作业在 B 上的处理时间，下标从 0 开始

k: 循环变量

p: 三维数组，长度为（m+1）*（m+1）*（n+1）

temp: 临时变量

max: 最短处理时间

（2）C 代码

```
#include<stdio.h>
int n， m;
int a[60]， b[60]， p[100][100][60];
void read__(9)__{ /*输入 n、a、b，求出 m，代码略*/}
void schedule__(10)__{ /*求解过程*/
   int x，y，k;
for（x=0；x<=m；x++）{
 for(y=0；y<m；y++）{
 （1）
    for（k=1；k<n；k++）
 p[x][y][k]=0；
}
```

```
}
for（k=1；k<n；k++）{
 for（x=0；x<=m；x++）{
for（y=0；y<=m；y++）{
if（x - a[k-1]>=0）　（2）　；
 if（ （3） ）p[x][y][k]=(p[x][y][k] ||p[x][y-b[k-1]][k-1]);
    }
 }
 }
}
void write__(11)__{  /*确定最优解并输出*/
int x，y，temp，max=m;
   for（x=0；x<=m；x++）{
    for（y=0;y<=m;y++）{
     if( （4） ）{
 temp=（5）；
 if（temp< max）max = temp;
    }
   }
  }
 printf(“\n%d\n”，max），
}
void main__(12)__{read__(13)__;schedule__(14)__;write__(15)__;}
```

【问题 1】（9 分）

根据以上说明和 C 代码，填充 C 代码中的空（1）～（5）。

【问题 2】（2 分）

根据以上 C 代码，算法的时间复杂度为（6）（用 O 符号表示）。

【问题 3】（4 分）

考虑 6 个作业的实例，各个作业在两台处理机上的处理时间如表 4-1 所示。该实例的最优解为（7），最优解的值（即最短处理时间）为（8）。最优解用（x1，x2，x3，x4，x5，x6）表示，其中若第 i 个作业在 A 上处理，则 xi=l，否则 xi=2。如（1，1，1，1，2，2）表示作业 1，2，3 和 4 在 A 上处理，作业 5 和 6 在 B 上处理

表4-1

	作业 1	作业 2	作业 3	作业 4	作业 5	作业 6
处理机 A	2	5	7	10	5	2
处理机 B	3	8	4	11	3	4

● 阅读下列说明和 C++代码，将应填入（n）处的字句写在答题纸的对应栏内。

【说明】

某咖啡店当卖咖啡时，可以根据顾客的要求在其中加入各种配料，咖啡店会根据所加入的配料来计算费用。咖啡店所供应的咖啡及配料的种类和价格如下表所示。

咖啡	价格/杯	配料	价格/份
蒸馏咖啡（Espresso）	25	摩卡（Mocha）	10
深度烘焙咖啡（DarkRoast）	20	奶泡（Whip）	8

现采用装饰器（Decorator）模式来实现计算费用的功能，得到如图 5-1 所示的类图

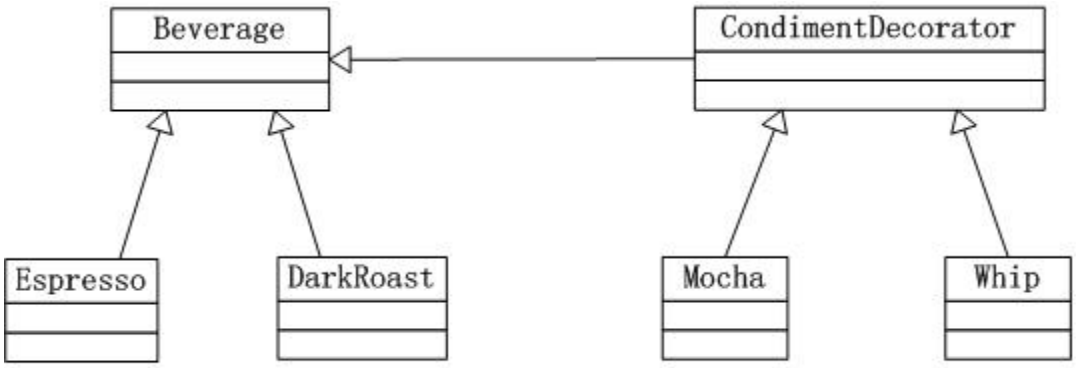

图 5-1 类图

【C++代码】

```
#include <iostream>
#include <string>
using namespace std;
const int ESPRESSO_PRICE = 25;
const int DRAKROAST_PRICE = 20;
const int MOCHA_PRICE = 10;
const int WHIP_PRICE = 8;
class Beverage { //饮料
   （1） ： string description;
public：
   （2） ( ){ return description; }
   （3） ;
};
class CondimentDecorator : public Beverage {  //配料
protected:
    （4） ;
};
class Espresso : public Beverage { / / 蒸馏咖啡
public:
Espresso ( ) {description="Espresso"; }
int cost ( ){return ESPRESSO_PRICE; }
};
class DarkRoast : public Beverage { //深度烘焙咖啡
public:
   DarkRoast( ){ description = "DardRoast"; }
   int cost( ){ return DRAKROAST_PRICE; }
 };
class Mocha : public CondimentDecorator {  / / 摩卡
public:
   Mocha（Beverage*beverage）{ this->beverage=beverage; }
   string getDescription( ){ return beverage->getDescription( )+"， Mocha"; }
   int cost( ){ return MOCHA_PRICE+beverage->cost( ); }
 };
class Whip :public CondimentDecorator {  //奶泡
public:
   Whip（Beverage*beverage） { this->beverage=beverage; }
   string getDescription( ) {return beverage->getDescription( )+"， Whip"; }
```

```
    int cost( ) { return WHIP_PRICE+beverage->cost( ); }
  };

int main() {
    Beverage* beverage = new DarkRoast( );
    beverage=new Mocha( （5） ) ;
    beverage=new Whip（ （6） ) ;
cout<<beverage->getDescription ( )<<"￥"<<beverage->cost( ) endl;
    return 0;
 }
```

编译运行上述程序，其输出结果为：

DarkRoast， Mocha， Whip ￥38

● 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】

某咖啡店当卖咖啡时，可以根据顾客的要求在其中加入各种配料，咖啡店会根据所加入的配料来计算费用。咖啡店所供应的咖啡及配料的种类和价格如下表所示。

咖啡	价格/杯	配料	价格/份
蒸馏咖啡（Espresso）	25	摩卡（Mocha）	10
深度烘焙咖啡（DarkRoast）	20	奶泡（Whip）	8

现采用装饰器（Decorator）模式来实现计算费用的功能，得到如图 6-1 所示的类图

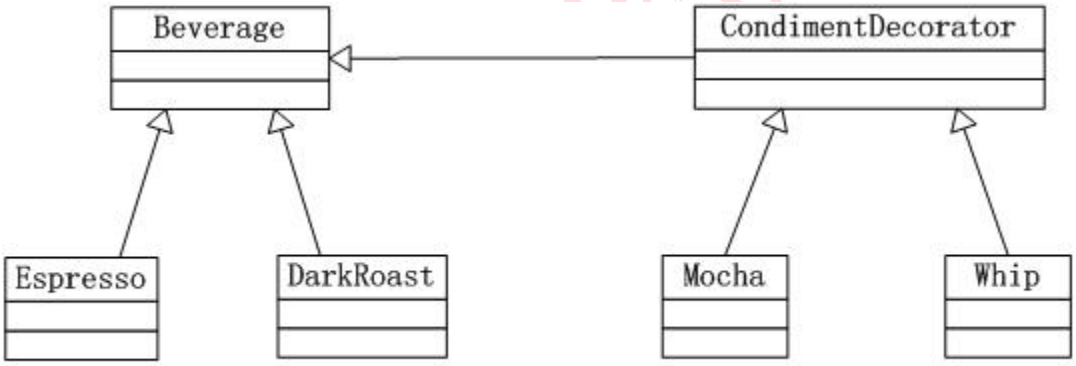

图 6-1 类图

【Java 代码】

```
      import jav
（11）A．util.*;
      （1） class Beverage {  //饮料
        String description = "Unknown Beverage";
        public  （2） (){return description;}
        public  （3） ;
      }
      abstract class CondimentDecorator extends Beverage { //配料
        （4） ;
      }

      class Espresso extends Beverage {  //蒸馏咖啡
        private final int ESPRESSO_PRICE = 25;
        public Espresso() {  description="Espresso"; }
```

```
    public int cost() {   return ESPRESSO_PRICE;   }
}

class DarkRoast extends Beverage {  //深度烘焙咖啡
    private finalint DARKROAST_PRICE = 20;
    public DarkRoast0 { description = "DarkRoast";   }
    public int cost(){ rcturn DARKROAST PRICE;   }
}
class Mocha extends CondimentDecorator {  //摩卡
    private final int MOCHA_PRICE = 10;
    public Mocha（Beverage beverage） {
    this.beverage = beverage;
}

    public String getDescription() {
    return beverage.getDescription0 + "， Mocha";
}
    public int cost() {
    return MOCHA_PRICE + beverage.cost();
    }
 }

class Whip extends CondimentDecorator {   //奶泡
    private finalint WHIP_PRICE = 8;
    public Whip（Beverage beverage） { this.beverage = beverage; }
    public String getDescription() {
    return beverage.getDescription()+"， Whip";
}
    public int cost() { return WHIP_PRICE + beverage.cost(); }
}
public class Coffee {
    public static void main(String args[]) {
    Beverage beverage = new DarkRoast();
    beverage=new Mocha(  (5)  );
beverage=new Whip (  (6)  ) ;
    System.out.println(beverage.getDescription0 +"￥" +beverage.cost());
}
}
```

编译运行上述程序，其输出结果为：

DarkRoast， Mocha， Whip ￥38