2011 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】

某医院欲开发病人监控系统。该系统通过各种设备监控病人的生命体征，并在生命体征异常时向医生和护理人员报警。该系统的主要功能如下：

（1）本地监控：定期获取病人的生命体征，如体温、血压、心率等数据。

（2）格式化生命体征：对病人的各项重要生命体征数据进行格式化，然后存入日志文件并检查生命体征。

（3）检查生命体征：将格式化后的生命体征与生命体征范围文件中预设的正常范围进行比较。如果超出了预设范围，系统就发送一条警告信息给医生和护理人员。

（4）维护生命体征范围：医生在必要时（如，新的研究结果出现时）添加或更新生命体征值的正常范围。

（5）提取报告：在医生或护理人员请求病人生命体征报告时，从日志文件中获取病人生命体征生成体征报告，并返回给请求者。

（6）生成病历：根据日志文件中的生命体征，医生对病人的病情进行描述，形成病历存入病历文件。

（7）查询病历：根据医生的病历查询请求，查询病历文件，给医生返回病历报告。

（8）生成治疗意见：根据日志文件中的生命体征和病历，医生给出治疗意见，如处方等，并存入治疗意见文件。

（9）查询治疗意见：医生和护理人员查询治疗意见，据此对病人进行治疗。

现采用结构化方法对病人监控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。

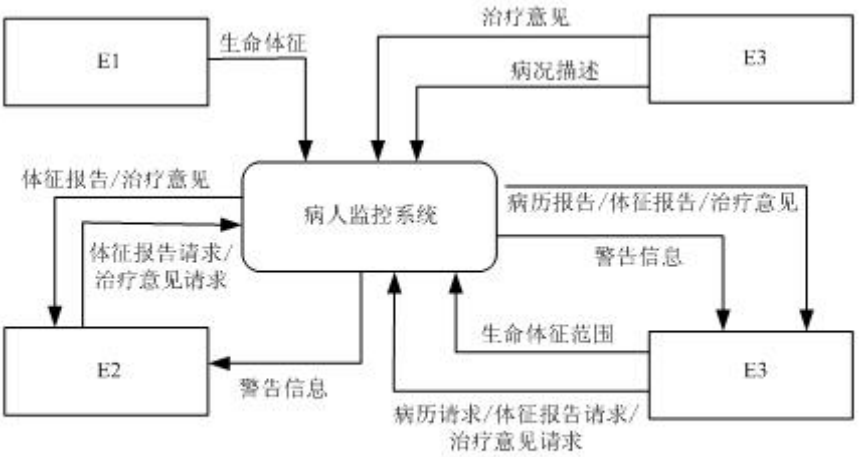


图 1-1 顶层数据流图

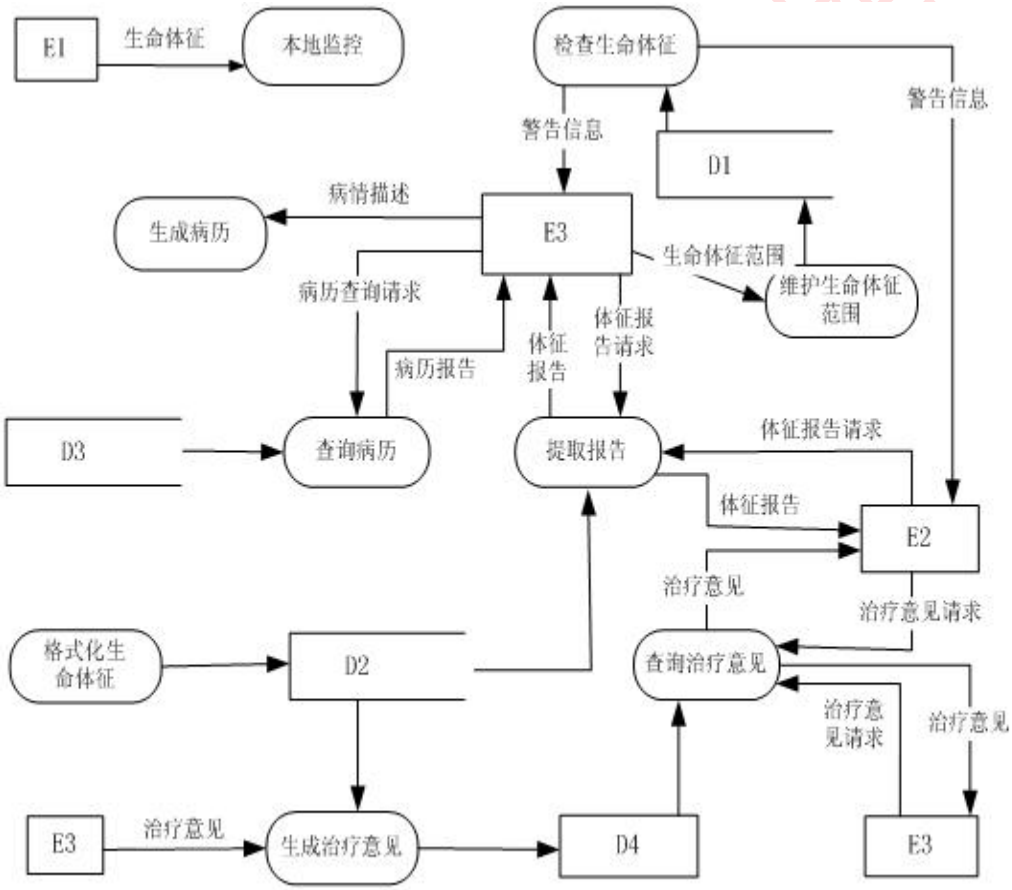


图 1-2 0 层数据流图

【问题 1】（3 分）

使用说明中的词语，给出图 1-1 中的实体 E1～E3 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】(6 分)

图 1-2 中缺失了 4 条数据流，使用说明、图 1-1 和图 1-2 中的术语，给出数据流的名称及其起点和终点。

【问题 4】（2 分）

说明实体 El 和 E3 之间可否有数据流，并解释其原因。

● 阅读下列说明，回答问题 l 至问题 3，将解答填入答题纸的对应栏内。

【说明】

某服装销售公司拟开发一套服装采购管理系统，以便对服装采购和库存进行管理。

【需求分析】

（1）采购系统需要维护服装信息及服装在仓库中的存放情况。服装信息主要包括：服装编码、服装描述、服装类型、销售价格、尺码和面料，其中，服装类型为销售分类，服装按销售分类编码。仓库信息包括：仓库编码、仓库位置、仓库容量和库管员。系统记录库管员的库管员编码、姓名和级别。一个库管员可以管理多个仓库，每个仓库有一名库管员。一个仓库中可以存放多类服装，一类服装可能存放在多个仓库中。

（2）当库管员发现有一类或者多类服装缺货时，需要生成采购订单。一个采购订单可以包含多类服装。每类服装可由多个不同的供应商供应，但具有相同的服装编码。采购订单主要记录订单编码、订货日期和应到货日期，并详细记录所采购的每类服装的数量、采购价格和对应的多个供应商。

（3）系统需记录每类服装的各个供应商信息和供应情况。供应商信息包括：供应商编码、供应商名称、地址、企业法人和联系电话。供应情况记录供应商所供应服装的服装类型和服装质量等级。一个供应商可以供应多类服装，一类服装可由多个供应商供应。库管员根据入库时的服装质量情况，设定或修改每个供应商所供应的每类服装的服装质量等级，作为后续采购服装时，选择供应商的参考标准。

【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。

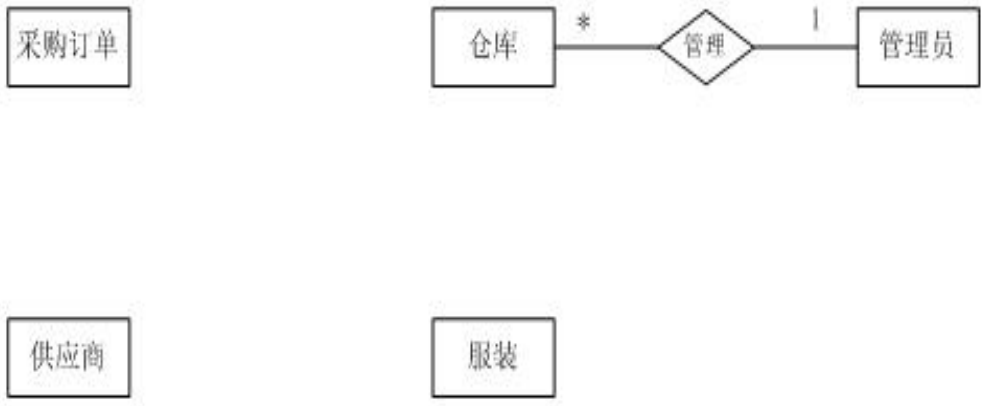


图 2-1 实体联系图

【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）：

库管员（库管员编码，姓名，级别）

仓库信息((1) ，仓库位置，仓库容量）

服装（服装编码，服装描述，服装类型，尺码，面料，销售价格）

供应商（供应商编码，供应商名称，地址，联系电话，企业法人）

供应情况((2) ，服装质量等级）
采购订单((3))
采购订单明细(____(4))

【问题 1】（6 分）
根据需求分析的描述，补充图 2.1 中的联系和联系的类型。
【问题 2】（6 分）
根据补充完整的图 2-1，将逻辑结构设计阶段生成的关系模式中的空(1)～(4)补充完整，并给出其主键（用下划线指出）。
【问题 3】（3 分）
如果库管员定期需要轮流对所有仓库中的服装质量进行抽查，对每个仓库中的每一类被抽查服装需要记录一条抽查结果，并且需要记录抽查的时间和负责抽查的库管员。请根据该要求，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。

● 阅读下列说明和图，回答问题 l 至问题 3，将解答填入答题纸的对应栏内。
【说明】
一个简单的图形编辑器提供给用户的基本操作包括：创建图形、创建元素、选择元素以及删除图形。图形编辑器的组成及其基本功能描述如下：
（1）图形由文本元素和图元元素构成，图元元素包括线条、矩形和椭圆。
（2）显示在工作空间中，一次只能显示一张图形(即当前图形，current)。
（3）提供了两种操作图形的工具：选择工具和创建工具。对图形进行操作时，一次只能使用一种工具(即当前活动工具，active)
① 创建工具用于创建文本元素和图元元素。
② 于显示在工作空间中的图形，使用选择工具能够选定其中所包含的元素，可以选择一个元素，也可以同时选择多个元素。被选择的元素称为当前选中元素(selected)。
③ 种元素都具有对应的控制点。拖拽选定元素的控制点，可以移动元素或者调整元素的大小。
现采用面向对象方法开发该图形编辑器，使用 UML 进行建模。构建出的用例图和类图分别如图 3-1 和 3-2 所示。

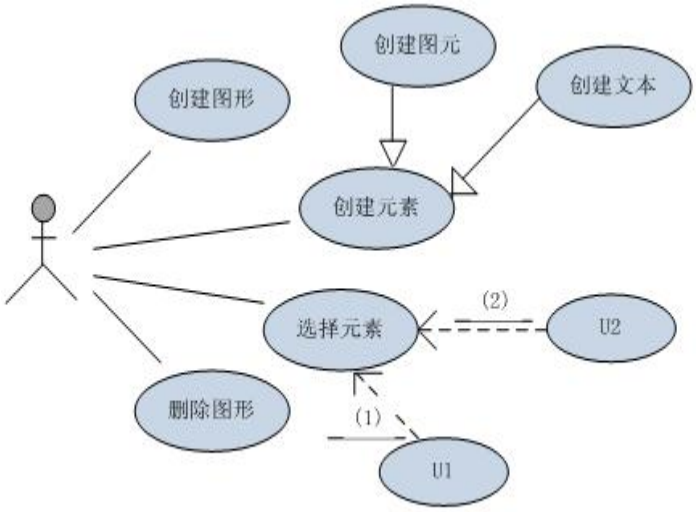


图 3-1

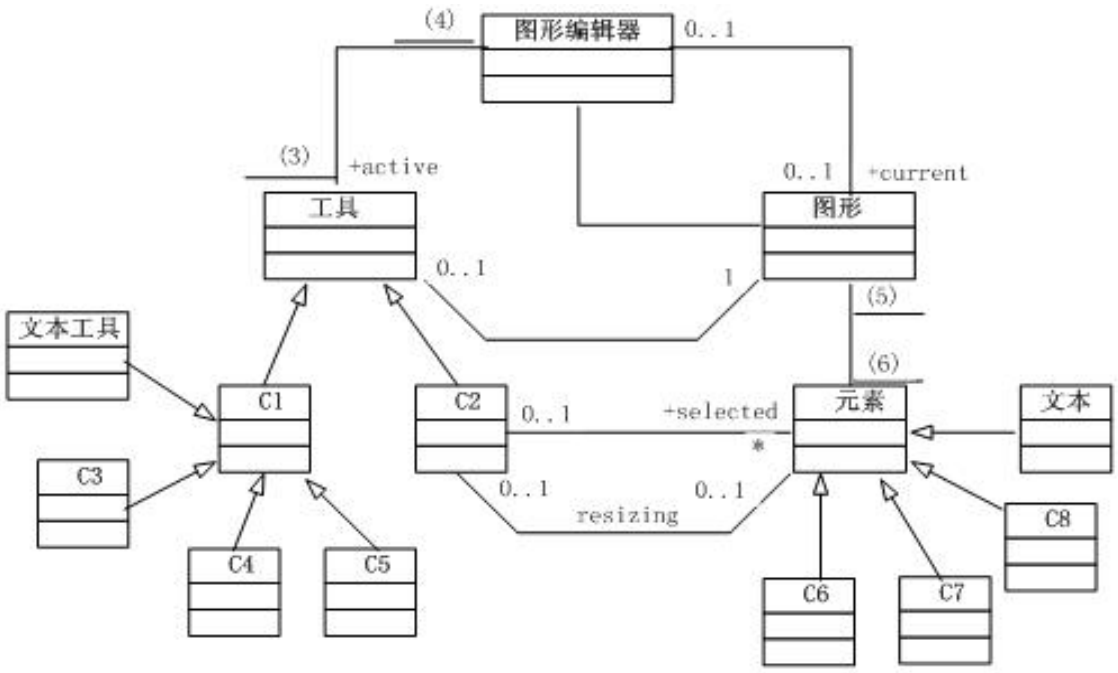


图 3-2

【问题 1】(4 分)

根据说明中的描述，给出图 3-1 中 U1 和 U2 所对应的用例，以及(1)和(2)处所对应的关系。

【问题 2】（8 分）

根据说明中的描述，给出图 3.2 中缺少的 C1～C8 所对应的类名以及(3)～(6)处所对应的多重度。

【问题 3】（3 分）

图 3-2 中的类图设计采用了桥接(Bridge)设计模式，请说明该模式的内涵。

● 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】

某应用中需要对 100000 个整数元素进行排序，每个元素的取值在 0～5 之间。排序算法的基本思想是：对每一个元素 x，确定小于等于 x 的元素个数(记为 m)，将 x 放在输出元素序列的第 m 个位置。对于元素值重复的情况，依次放入第 m-l、m-2、…个位置。例如，如果元素值小于等于 4 的元素个数有 10 个，其中元素值等于 4 的元素个数有 3 个，则 4 应该在输出元素序列的第 10 个位置、第 9 个位置和第 8 个位置上。算法具体的步骤为：

步骤 1：统计每个元素值的个数。

步骤 2：统计小于等于每个元素值的个数。

步骤 3：将输入元素序列中的每个元素放入有序的输出元素序列。

【C 代码】

下面是该排序算法的 C 语言实现。

(1)常量和变量说明

R: 常量，定义元素取值范围中的取值个数，如上述应用中 R 值应取 6

i：循环变量

n：待排序元素个数

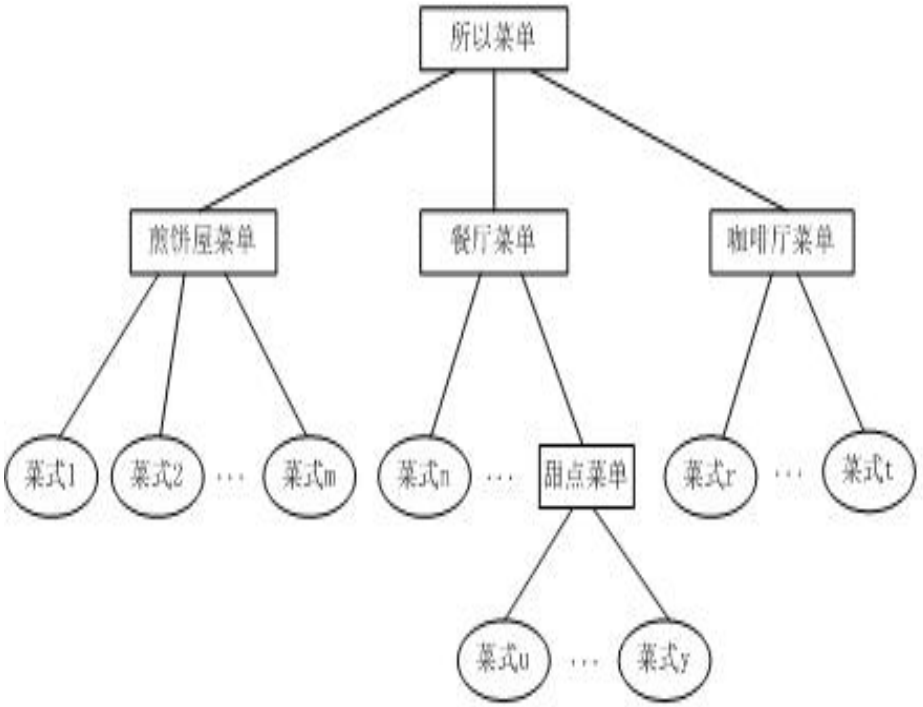
a：输入数组，长度为 n

b：输出数组，长度为 n
c：辅助数组，长度为 R，其中每个元素表示小于等于下标所对应的元素值的个数。
(2)函数 sort

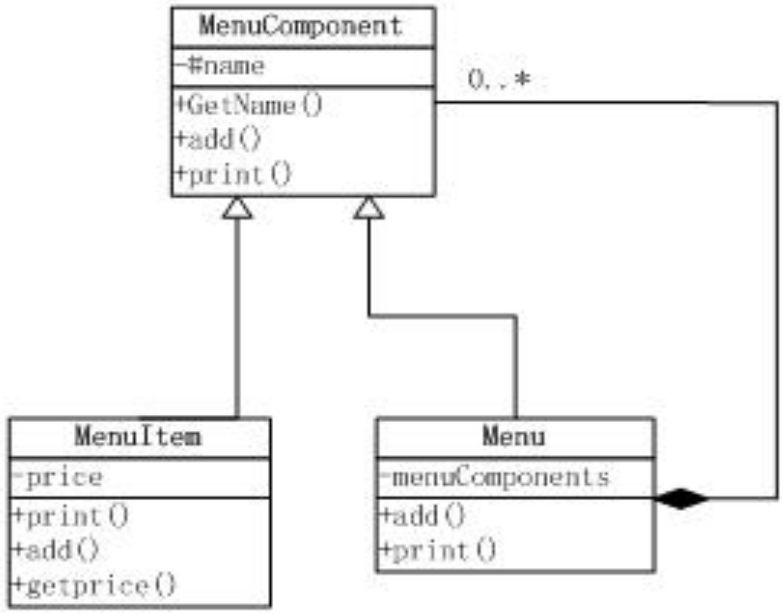
```
1   void sort(int n，int a[]，int b[]){
2     int c[R]，i；
3  for (i=0；i<   (1) ：i++){
4    c[i]=0；
5     }
6     for(i=0；i<n；i++）{
7   c[a[i]] =  (2) ；
8     }
9  for(i=1；i<R；i++）{
10   c[i]=  （3）
11    }
12 for(i=0；i<n；i++）{
13   b[c[a[i]]-1]=  (4)  ；
14   c[a[i]]=c[a[i]]-1；
15    }
16  }
```

【问题 1】（8 分）
根据说明和 C 代码，填充 C 代码中的空缺(1)～(4)。
【问题 2】（4 分）
根据 C 代码，函数的时间复杂度和空间复杂度分别为 (5) 和 (6)（用 O 符号表示）。
【问题 3】（3 分）
根据以上 C 代码，分析该排序算法是否稳定。若稳定，请简要说明（不超过 100 字）；若不稳定，请修改其中代码使其稳定（给出要修改的行号和修改后的代码）。

● 阅读下列说明和 C++代码，将应填入空(n)处的字句写在答题纸的对应栏内。
【说明】
某饭店在不同的时段提供多种不同的餐饮，其菜单的结构图如下图所示。



现在采用组合(Composition)模式来构造该饭店的菜单，使得饭店可以方便地在其中增加新的餐饮形式，得到如下图所示的类图。其中 MenuComponent 为抽象类，定义了添加(add)新菜单和打印饭店所有菜单信息(print)的方法接口。类 Menu 表示饭店提供的每种餐饮形式的菜单，如煎饼屋菜单、咖啡屋菜单等。每种菜单中都可以添加子菜单，例如图中的甜点菜单。类 MenuItem 表示菜单中的菜式。



（）

【C++代码】

```
#include<iostream>
#include<list>
#include <string>
```

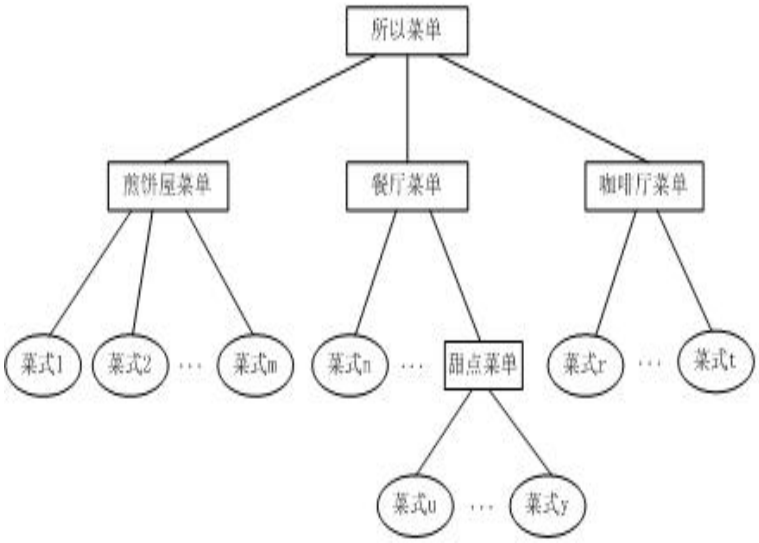
```
using namespace std;
class MenuComponent{
protected:  string name;
public:
  MenuComponent(string name){  this->name= name;)
  string getName(){ return name;)
    (1)  ;  //添加新菜单
  virtual void print()=0;  //打印菜单信息

};
class Menultem: public MenuComponent{
private:double price;
public:
    Menultem(string name,  double price):MenuComponent(name){ this->price= price;
    double getPrice(){ return price;)
    void add(MenuComponent* menuComponent){ retum;  }//添加新菜单
    void print(){ cout<<"  "  <<getName0<<",  "<<getPrice0<<endl;)
};
class Menu:public MenuComponent{
private:    list<  (2)  >  menuComponents;
public:
  Menu(string name):  MenuComponent(name){}
  void add(MenuComponent* menuComponent) //添加新菜单
  {  (3)  ;      }
  void print(){
  cout<<"\n"<<getNameO<<"\n----------"<<endl;
  std::list<MenuComponent *>::iterator iter,
  for(iter= menuComponents.begin0; iter！=menuComponents.end0; iter++)
    (4) ->print();
    }
};
void main0{
  MenuComponent* allMenus= new Menu("ALL MENUS");
  MenuComponent* dinerMenu= new Menu("DINER MENU");
  ……//创建更多的 Menu 对象，此处代码省略
  allMenus->add(dinerMenu);  //将 dinerMenu 添加到餐厅菜单中
  ……//为餐厅增加更多的菜单，此处代码省略
 (5) ->printO;  //打印饭店所有菜单的信息
```

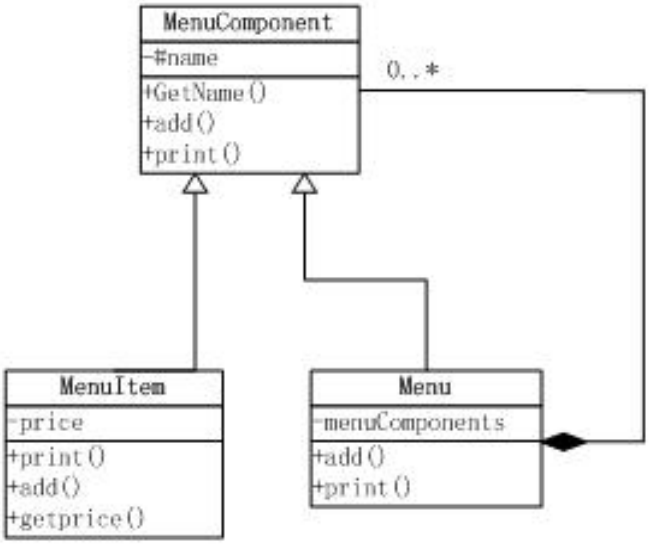
● 阅读下列说明和 Java 代码，将应填入(n)处的字句写在答题纸的对应栏内。

【说明】

某饭店在不同的时段提供多种不同的餐饮，其菜单的结构图如下图所示。



现在采用组合(Composition)模式来构造该饭店的菜单，使得饭店可以方便地在其中增加新的餐饮形式，得到如下图所示的类图。其中 MenuComponent 为抽象类，定义了添加(add)新菜单和打印饭店所有菜单信息(print)的方法接口。类 Menu 表示饭店提供的每种餐饮形式的菜单，如煎饼屋菜单、咖啡屋菜单等。每种菜单中都可以添加子菜单，例如图中的甜点菜单。类 Menultem 表示菜单中的菜式。



【Java 代码】

import jav

（6）A．util.*;

(1) MenuComponent{

protected String name;

(2) ;//添加新菜单

public abstract void print(); //打印菜单信息

public String getName(){ return name;}

}

class Menultem extends MenuComponent{

private double price;

```
    public MenuItem(String name, double price){
     this.name= name;  this.price= price;
    }
    public double getPrice(){return price;)
    public void add(MenuComponent menuComponent){ return;）//添加新菜单
    public void print(){
      System.out.print(" "+ getName());
      System.out.println(","+ getPrice());
    }
}
class Menu extends MenuComponent{
    private List<MenuComponent> menuComponents= new ArrayList<MenuComponent>();
    public Menu(String name){ this.name= name;)
    public void add(MenuComponent menuComponent){//添加新菜单
  menuComponents. (3);
}
    public void print(){
    System.out.print("\n"+ getName());
    System.out.println(","+"----------");
    Iterator iterator = menuComponents.iterator();
    while(iterator.hasNext()){
   MenuComponent menuComponent= (MenuComponent)iterator.next();
        (4) ;
     }
    }
}
class MenuTestDrive{
  public static void main(String args[]){
     MenuComponent   aIIMenus= new Menu("ALL MENUS");
     MenuComponent   dinerMenu = new Menu("DINER MENU”);
     ……//创建更多的 Menu 对象，此处代码省略
    allMenus.add(dinerMenu);   //将 dinerMenu 添加到餐厅菜单中
    ……//为餐厅增加更多的菜单，此处代码省略
      (5);   //打印饭店所有菜单的信息
    }
 }
```