2013 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 3，将解答填入答题纸的对应栏内。

【说明】

某慈善机构欲开发一个募捐系统，已跟踪记录为事业或项目向目标群体进行募捐而组织的集体性活动。该系统的主要功能如下所述。

（1）管理志愿者。根据募捐任务给志愿者发送加入邀请、邀请跟进、工作任务；管理志愿者提供的邀请响应、志愿者信息、工作时长、工作结果等。

（2）确定募捐需求和收集所募捐赠（资金及物品）。根据需求提出募捐任务、将活动请求和捐赠请求，获取所募集的资金和物品。

（3）组织募捐活动。根据活动请求，确定活动时间范围。根据活动时间，搜索场馆，即：向场馆发送场馆可用性请求，获得场馆可用性。然后根据活动时间和地点推广募捐活动，根据相应的活动信息举办活动，从募捐机构获取资金并向其发放赠品。获取和处理捐赠，根据捐赠请求，提供所募集的捐赠；处理与捐赠人之间的交互，即：

录入捐赠人西悉尼，处理后存入捐赠人信息表；从捐赠人信息表中查询捐赠人信息，向捐赠人发送捐赠请求，并将已联系的捐赠人存入已联系的捐赠人表。根据捐赠请求进行募集，募得捐赠后，将捐赠记录存入捐赠表；对捐赠记录进行记录后，存入已处理捐赠表，向捐赠人发送致谢函，根据已联系的捐赠人和捐赠记录进行跟踪，并将捐赠跟进情况发送给捐赠人。

先采用结构化方法对募捐系统进行分析和设计，获得如图 1-1、图 1-2 和图 1-3 所示分层数据流图。

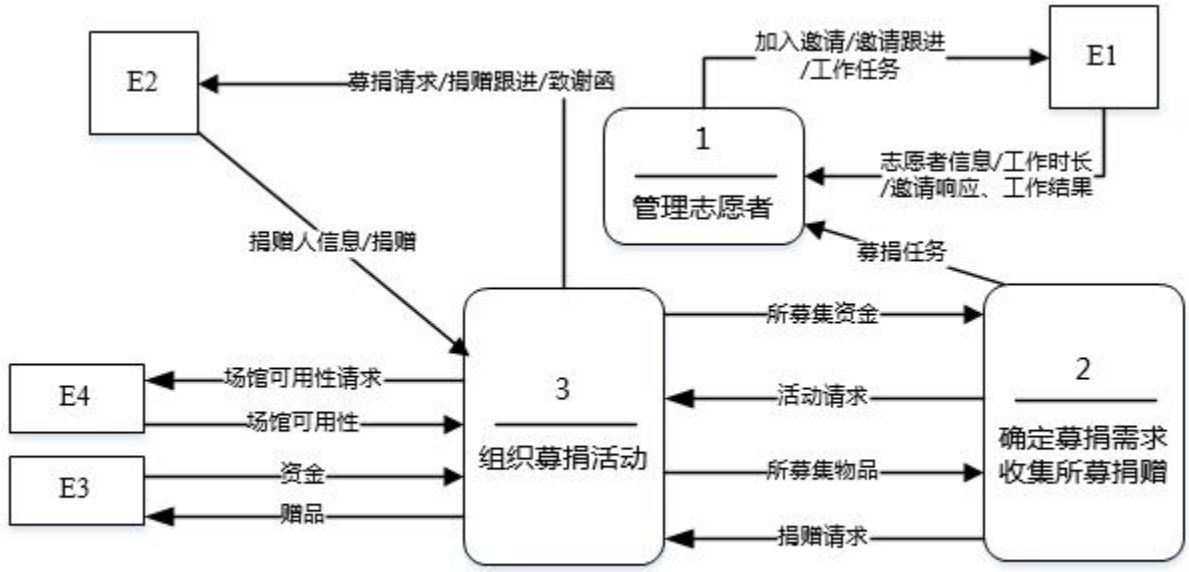

图1-1 0层数据流图

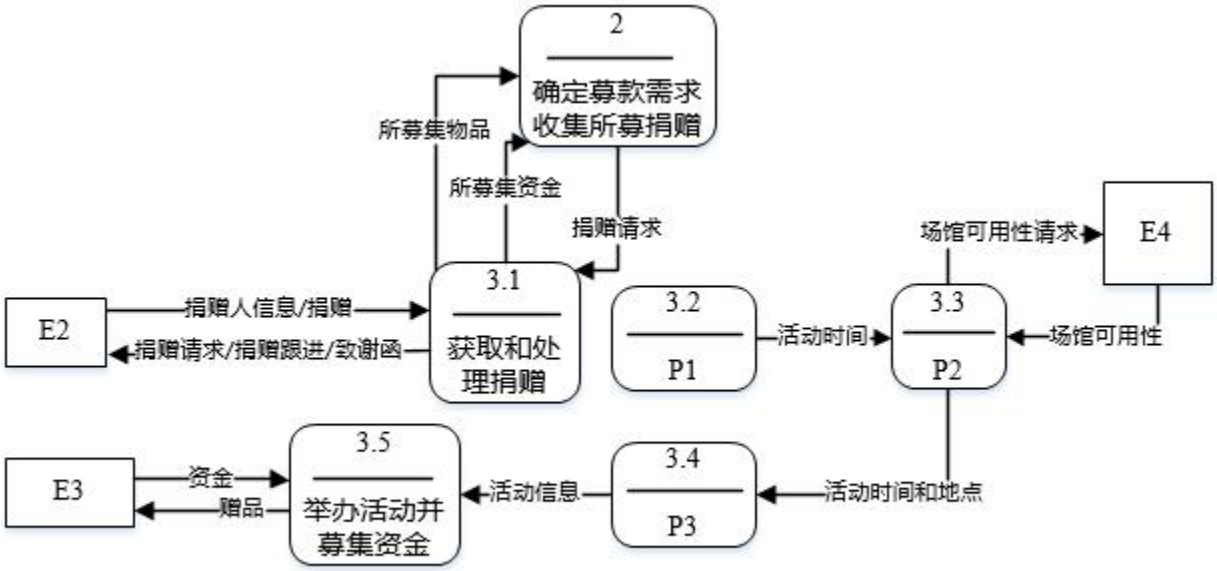

图1-2 1层数据流图

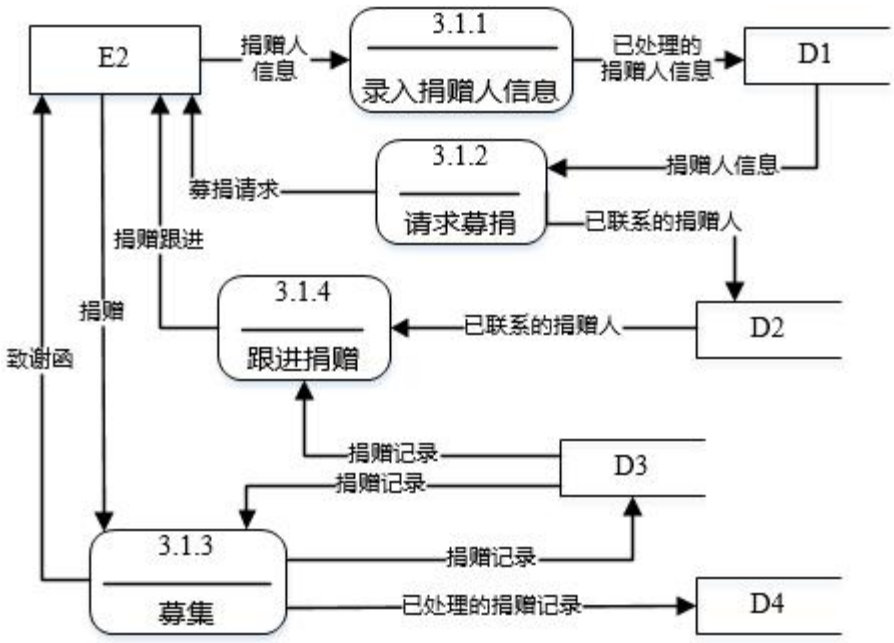

图1-3 2层数据流图

【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】（7 分）

在建模 DFD 时，需要对有些复杂加工（处理）进行进一步精化，图 1-2 为图 1-1 中处理 3 的进一步细化的 1 层数据流图，图 1-3 为图 1-2 中 3.1 进一步细化的 2 层数据流图。补全 1-2 中加工 P1、P2 和 P3 的名称和图 1-2 与图 1-3 中缺少的数据流。

【问题 3】（4 分）

使用说明中的词语，给出图 1-3 中的数据存储 D1~D4 的名称。

•

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】

某电视台拟开发一套信息管理系统，以方便对全台的员工、栏目、广告和演播室等进行管理。

【需求分析】

（1）系统需要维护全台员工的详细信息、栏目信息、广告信息和演播厅信息等。员工的信息主要包括：工号、姓名、性别、出生日期、电话、住址等。栏目信息主要包括：栏目名称、播出时间、时长等。广告信息主要包括：广告编号、价格等。演播厅信息包括：房间号、房间面积等。

（2）电视台分局调度单来协调各档栏目、演播厅和场务。一销售档栏目只会占用一个演播厅，但会使用多么场务来进行演出协调。演播厅和场务可以被多个栏目循环使用。

（3）电视台根据栏目来插播广告。每档栏目可以插播多条广告，每条广告也可以在多的栏目插播。

（4）一档栏目可以有多个主持人，但一名主持人只能支持一档节目。

（5）一名编辑人员可以编辑多条广告，一条广告只能由一名编辑人员编辑。

【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。

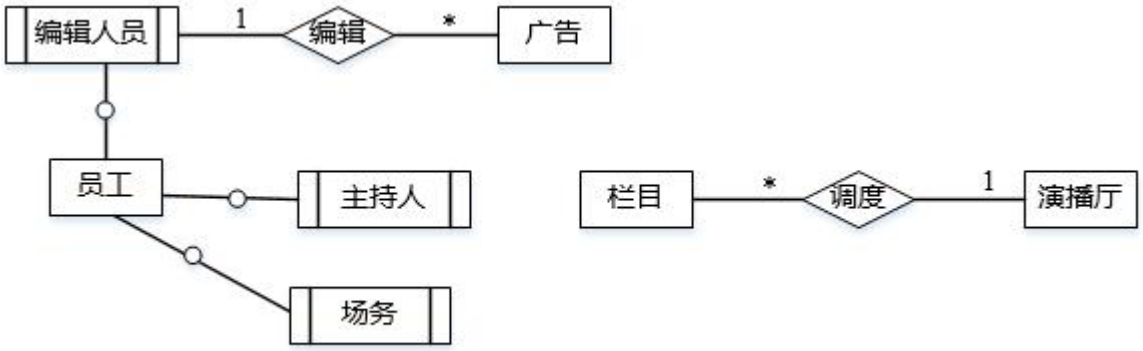

图2-1 实体联系图

【逻辑结构设计】

根据概念模式设计阶段完成的实体联系图，得出如下关系模型（不完整）：

演播厅（房间号，房间面积）

栏目（栏目名称，播出时间，时长）

广告（广告编号，销售价格，__（1）__）

员工（工号，姓名，性别，出生日期，电话，住址）

主持人（主持人工号，__（2）__）

演播单（__（3）__，播出时间）

调度单（__（4）__）

【问题 1】（7 分）

补充图 2-1 中的联系和联系类型。

【问题 2】（5 分）

根据图 2-1，将逻辑结构设计阶段生产的关系模型的空(1)~(4)补充完整，并用下划线指出（1）~（4）所在关系模型的主键。

【问题 3】（3 分）

现需要记录广告商信息，增加广告商实体。一个广告商可以提供多条广告，一条广告只由一个广告商提供。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 1 至问题 3，将解答填入答题纸的对应栏内。

【说明】

某城市拟开发一个基于 Web 城市黄页，公开发布该城市重要的组织或机构（以下统称为客户）的基本信息，方便城市生活。该系统的主要功能描述如下：

（1）搜索信息：任何使用 Internert 的网络用户都可以搜索发布在城市黄页中的信息，例如客户的名称、地址、联系电话等。

（2）认证：客户若想在城市黄页上发布信息，需通过系统的认证。认证成功后，该客户成为系统授权用户。

（3）更新信息：授权用户登录系统后，可以更改自己在城市黄页中的相关信息，例如变更联系电话等。

（4）删除客户：对于拒绝继续在城市黄页上发布信息的客户，有系统管理员删除该客户的相关信息。

系统采用面向对象方法进行开发，在开发过程中认定出如表 3-1 所示的类。系统的用例图和类图分别如图 3-1 和图 3-2 所示。

表 3-1 类列表

类名	说明
InternetClient	网络用户
CustomerList	客户集，维护城市黄页上的所有客户信息
Customer	客户信息，记录单个客户的信息
RegisteredClient	授权用户
Administrator	系统管理员

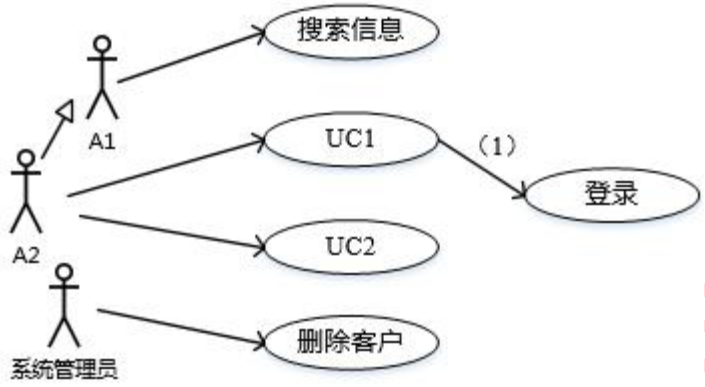

图3-1 系统用例图

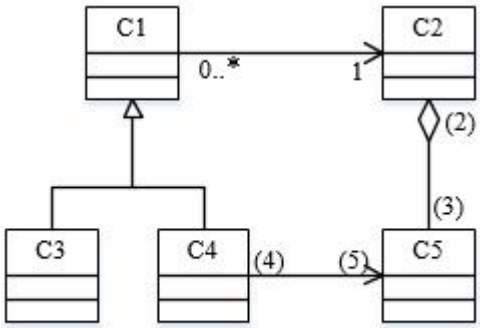

图3-2 系统类图

【问题 1】（5 分）

根据说明中的描述，给出图 3-1 中 A1 和 A2 处所对应的参与者，UC1 和 UC2 所对应的用例以及（1）处的关系。

【问题 2】（7 分）

根据说明中的描述，给出图 3-2 中 C1~C5 所对应的类名（表 3-1 中给出的类名）和（2）~（5）处所对应的多重度。

【问题 3】（3 分）

认定类是面向对象分析中非常关键的一个步骤。一般首先从问题域中得到候选类集合，在根据相应的原则从该集合中删除不作为类的，剩余的就是从问题域中认定出来的类。简要说明选择候选类的原则，以及对候选类集合进行删除的原则。

● 设有 m 台完全相同的机器运行 n 个独立的任务，运行任务 i 所需的时间为 ti，要求确定一个调度方案，使得完成所有任务所需要的时间最短。

假设任务已经按照其运行时间从大到小排序，算法基于最长运行时间作业优先的策略，按顺序先把每个任务分配到一台机器上，然后将剩余的任务一次放入最先空闲的机器。

【C 代码】
下面是算法的 C 语言实现。
1.常量和变量说明
m：机器数
n：任务数
t[]：输入数组，长度为 n，下标从 0 开始，其中每个元素表示任务的运行时间，下标从 0 开始。
s[][]：二位数组，长度为 m*n，下标从 0 开始，其中元素 s[i][j]表示机器 i 运行的任务 j 的编号。
d[]：数组，长度为 m 其中元素 d[i]表示机器 i 的运行时间，下标从 0 开始。
count[]：数组，长度为 m，下标从 0 开始，其中元素 count[i]表示机器 i 运行的任务数。
i：循环变量。
j：循环变量。
k：临时变量。
max：完成所有任务的时间。
min：临时变量。 2.函数 schedule

```
void  schedule__(8)__{
int i,j,k,max=0;
 for( i=0;i<m;i++){
     d[i]=0;
     for(j=0;j<n;j++){
       s[i][j]=0;
     }
 }
 for(i=0;i<m;i++){  //分配前 m 个任务
   s[i][0]=i;
    (1)    ;
    count[i]=1;
 }
 for(  (2)  ;i<n;i++){          //分配后 n~m 个任务
   int min = d[0];
   k=0;
   for(j=1;j<m;j++){          //确定空闲时间
     if(min>d[j]){
       min = d[j];
        k=j;                  //机器 k 空闲
     }
   }
 (3)   ;
 count[k] = count[k]+1;
 d[k] = d[k]+t[i];
}
 for(i =0;i<m;i++){            //确定完成所有任务所需要的时间
   if(  (4)  ){
     max=d[i];
   }
 }
}
```

【问题 1】（8 分）
根据说明和 C 代码，填充 C 代码中的空（1）~（4）。

【问题 2】（2 分）

根据说明和 C 代码，该问题采用了 （5） 算法设计策略，时间复杂度 （6） （用 O 符号表示）

【问题 3】（5 分）

考虑实例 m=3（编号 0~2），n=7（编号 0~6），各任务的运行时间为{16，14，6，5，4，3，2}。则在机器 0、1 和 2 上运行的任务分别为（7）、（8）和（9）（给出任务编号）。从任务开始运行到完成所需的时间为（10）。

● 现要求实现一个能够自动生成求职简历的程序，简历的基本内容包括求职者的姓名、性别、年龄及工作经历。希望每份简历中的工作经历有所不同，并尽量减少程序中的重复代码。

现采用原型模式（Prototype）来实现上述要求，得到如图 5-1 所示的类图。

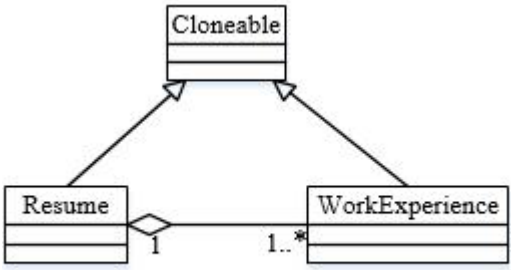

图5-1 类图

【C++代码】

```
#include<string>
using namespace std;
class  Cloneable{
public:
     (1)  ;
};
class  workExperience:public  Cloneable{   //工作经历
private:
     string  workData;
     string  company;
public:
    Cloneable * clone__(11)__{
     (2)  ;
    Obj->workDate= this->workDate;
    Obj->company = this->company;
    return Obj;
}
 //其余代码省略
}; class  Resume:public  Cloneable{      //简历
private:
       string name;
       string sex;
       string age;
       WorkExperience * work;
       Resume(WorkExperience * work){
       this->work =    (3)  ;
       }
public:
       Resume(string name){/*实现省略*/}
```

```
    void SetPersonInfo(string sex,string age){/*实现省略*/}
    void SetWorkExperience(string workDate,string company){/*实现省略*/}
    Cloneable * Clone__(12)__{
     (4)  ;
    Obj->name = this->name;
   Obj->sex = this->sex;
   Obj->age = this->age;
   return Obj;
}
}; int  main__(13)__{
    Resume * a  = new Resume(“张三”);
    a->SetPersonInfo(“男”,“29”)；
    a-> SetWorkExperience(“1998-2000”,“XXX 公司”);
    Resume * b =   (5)  ;
    b-> SetWorkExperience(“2001-2006”,“YYY 公司”);
    return 0;
}
```

● 现要求实现一个能够自动生成求职简历的程序，简历的基本内容包括求职者的姓名、性别、年龄及工作经历。希望每份简历中的工作经历有所不同，并尽量减少程序中的重复代码。
现采用原型模式（Prototype）来实现上述要求，得到如图 6-1 所示的类图。

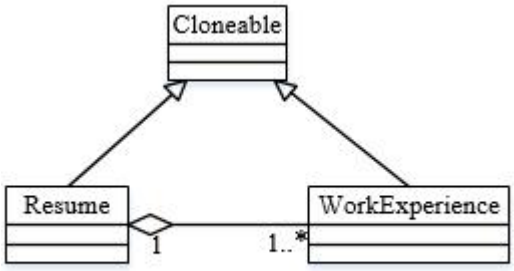

图6-1 类图

【Java 代码】

```
public class workExperience  (1)  Cloneable{  //工作经历
private String workDate;
private String company;
public Object clone__(16)__{
   (2) ;
   Obj.workDate= this.workDate;
   Obj.company = this.company;
    return Obj;
}
 //其余代码省略
} public class Resume  (3)  Cloneable{    //简历
private String name;
private String sex;
private String age;
private WorkExperience work; public Resume(string name){
    this.name = name;
    work = new WorkExperience__(17)__;
}
private Resume(WorkExperience work){
   this.work =  (4)  ;
```

```
}
public  void  SetPersonInfo(string sex,string age){/*实现省略*/}
public  void  SetWorkExperience(string workDate,string company){/*实现省略*/}
public  Object  clone__(18)__{
    Resume Obj =    (5)   ;
    return Obj;
}
}

Class WorkResume{
public  static  void  main__(19)__{
     Resume   a  = new Resume(“张三”);
     a.SetPersonInfo(“男”,“29”);
     a.SetWorkExperience(“1998-2000”,“XXX 公司”);
     Resume  b =    (6)   ;
     b. SetWorkExperience(“2001-2006”,“YYY 公司”);
}
}
```