全国计算机技术与软件专业技术资格（水平）考试

2008 年下半年　软件设计师　下午试卷（B）

（考试时间　14:00～16:30　共 150 分钟）

请按下述要求正确填写答题纸

1. 在答题纸的指定位置填写你所在的省、自治区、直辖市、计划单列市的名称。
2. 在答题纸的指定位置填写准考证号、出生年月日和姓名。
3. 答题纸上除填写上述内容外只能写解答。
4. 本试卷共 7 道题，试题一至试题四是必答题，试题五至试题七选答 1 道。每题 15 分，满分 75 分。
5. 解答时字迹务必清楚，字迹不清时，将不评分。
6. 仿照下面例题，将解答写在答题纸的对应栏内。

例题

2008 年下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。

因为正确的解答是“12 月 21 日”，故在答题纸的对应栏内写上“12”和“21”（参看下表）。

例题	解答栏
（1）	12
（2）	21

试题一（共 15 分）

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】

某营销企业拟开发一个销售管理系统，其主要功能描述如下：

1．接受客户订单，检查库存货物是否满足订单要求。如果满足，进行供货处理：即修改库存记录文件，给库房开具备货单并且保留客户订单至订单记录文件；否则进行缺货处理：将缺货订单录入缺货记录文件。

2．根据缺货记录文件进行缺货统计，将缺货通知单发给采购部门。

3．根据采购部门提供的进货通知单进行进货处理：即修改库存记录文件，并从缺货记录文件中取出缺货订单进行供货处理。

4．根据保留的客户订单进行销售统计，打印统计报表给经理。

现采用结构化方法对销售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。

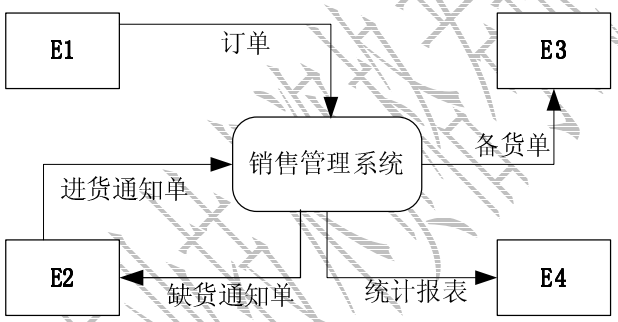

图 1-1 顶层数据流图

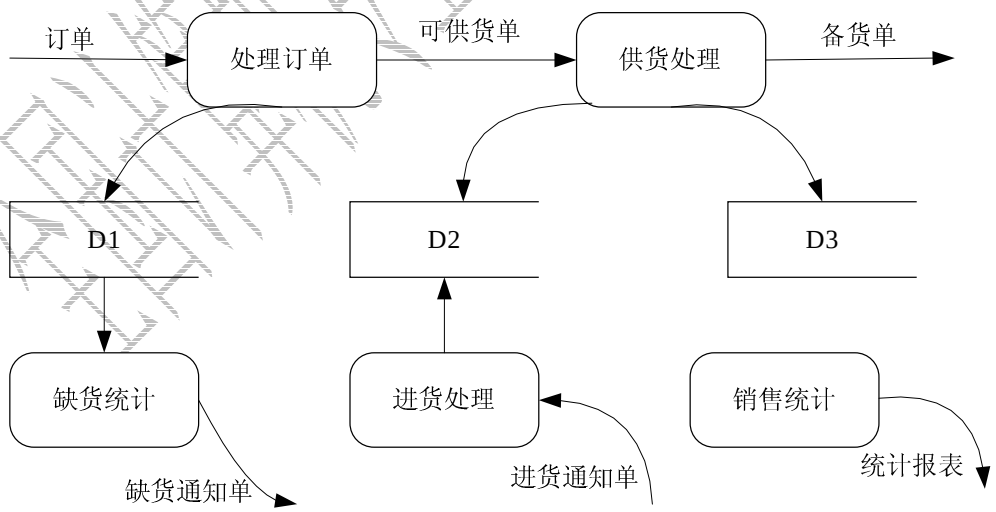

图 1-2　0 层数据流图

【问题 1】（4 分）

使用说明中的词语，给出图 1-1 的外部实体 E1~E4 的名称。

【问题 2】（3 分）

使用说明中的词语，给出图 1-2 的数据存储 D1~D3 的名称。

【问题 3】（8 分）

数据流图 1-2 缺少了四条数据流，根据说明及数据流图 1-1 提供的信息，分别指出这四条数据流的起点和终点。

起　点	终　点

试题二（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】

某宾馆拟开发一个宾馆客房预订子系统，主要是针对客房的预订和入住等情况进行管理。

【需求分析结果】

1．员工信息主要包括：员工号、姓名、出生年月、性别、部门、岗位、住址、联系电话和密码等信息。岗位有管理和服务两种。岗位为“管理”的员工可以更改（添加、删除和修改）员工表中的本部门员工的岗位和密码，要求将每一次更改前的信息保留；岗位为“服务”的员工只能修改员工表中本人的密码，且负责多个客房的清理等工作。

2．部门信息主要包括：部门号、部门名称、部门负责人、电话等信息；一个员工只能属于一个部门，一个部门只有一位负责人。

3．客房信息包括：客房号、类型、价格、状态等信息。其中类型是指单人间、三人间、普通标准间、豪华标准间等；状态是指空闲、入住和维修。

4．客户信息包括：身份证号、姓名、性别、单位和联系电话。

5．客房预定情况包括：客房号、预定日期、预定入住日期、预定入住天数、身份证号等信息。一条预定信息必须且仅对应一位客户，但一位客户可以有多条预定信息。

【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：

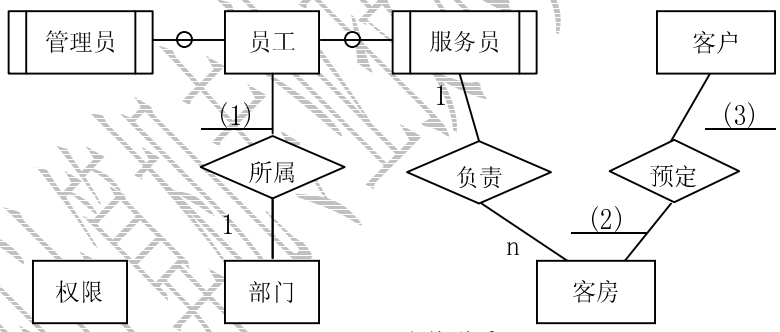

图 2-1 实体联系图

【逻辑结构设计】

逻辑结构设计阶段设计的部分关系模式（不完整）如下：

员工（ (4) ，姓名，出生年月，性别，岗位，住址，联系电话，密码）

权限（岗位，操作权限）

部门（部门号，部门名称，部门负责人，电话）

客房（ (5) ，类型，价格，状态，入住日期，入住时间，员工号）

客户（ (6) ，姓名，性别，单位，联系电话）

更改权限（员工号， (7) ，密码，更改日期，更改时间，管理员号）

预定情况（ (8) ，预定日期，预定入住日期，预定入住天数）

【问题 1】（3 分）

根据问题描述，填写图 2-1 中(1)～(3)处联系的类型。联系类型分为一对一、一对多和多对多三种，分别使用 1 : 1，1 : n 或 1 : *，m : n 或 * : *表示。

【问题 2】（2 分）

补充图 2-1 中的联系并指明其联系类型。

【问题 3】（7 分）

根据需求分析结果和图 2-1，将逻辑结构设计阶段生成的关系模式中的空(4)～(8)补充完整。（注：一个空可能需要填多个属性）

【问题 4】（3 分）

若去掉权限表，并将权限表中的操作权限属性放在员工表中（仍保持管理和服务岗位的操作权限规定），则与原有设计相比有什么优缺点（请从数据库设计的角度进行说明）？

试题三（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】

在线会议审稿系统（ORS：**O**nline **R**eviewing **S**ystem）主要处理会议前期的投稿和审稿事务，其功能描述如下：

1. 用户在初始使用系统时，必须在系统中注册（register）成为作者或审稿人。

2. 作者登录（login）后提交稿件和浏览稿件审阅结果。提交稿件必须在规定提交时间范围内，其过程为先输入标题和摘要、选择稿件所属主题类型、选择稿件所在位置（存储位置）。上述几步若未完成，则重复；若完成，则上传稿件至数据库中，系统发送通知。

3. 审稿人登录后可设置兴趣领域、审阅稿件给出意见以及罗列录用和（或）拒绝的稿件。

4. 会议委员会主席是一个特殊审稿人，可以浏览提交的稿件、给审稿人分配稿件、罗列录用和（或）拒绝的稿件以及关闭审稿过程。其中关闭审稿过程须包括罗列录用和（或）拒绝的稿件。

系统采用面向对象方法开发，使用 UML 进行建模。在建模用例图时，常用的方式是先识别参与者，然后确定参与者如何使用系统来确定用例，每个用例可以构造一个活动图。参与者名称、用例和活动名称分别参见表 3-1、表 3-2 和表 3-3。系统的部分用例图和提交稿件的活动图分别如图 3-1 和图 3-2 所示。

表 3-1 参与者列表

名称	说明	名称	说明
User	用户	Author	作者
Reviewer	审稿人	PCChair	委员会主席

表 3-2 用例名称列表

名称	说明	名称	说明
login	登录系统	register	注册
submit paper	提交稿件	browse review results	浏览稿件审阅结果
close reviewing process	关闭审稿过程	assign paper to reviewer	分配稿件给审稿人
set preferences	设定兴趣领域	enter review	审阅稿件给出意见
list accepted/rejected papers	罗列录用或/和拒绝的稿件	browse submitted papers	浏览提交的稿件

表 3-3 活动名称列表

名称	说明	名称	说明
select paper location	选择稿件位置	upload paper	上传稿件
select subject group	选择主题类型	send notification	发送通知
enter title and abstract	输入标题和摘要		

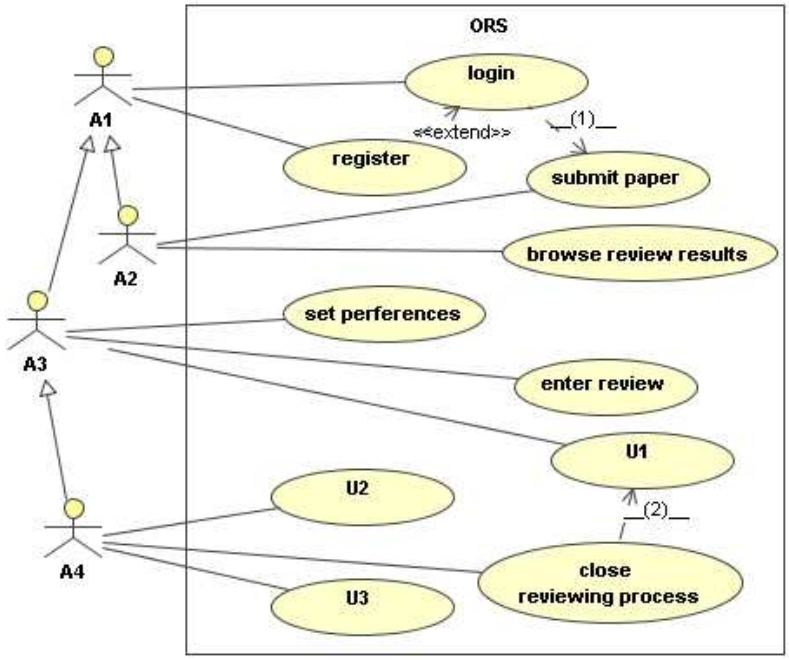

图 3-1 ORS 用例图

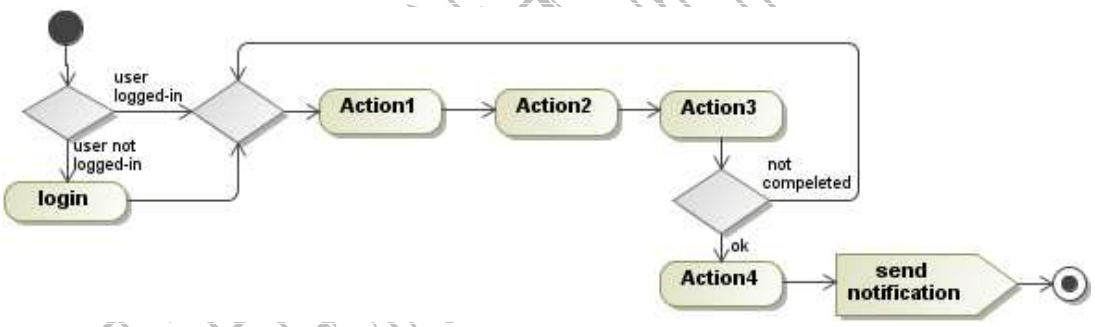

图 3-2 提交稿件过程的活动图

【问题 1】（4 分）

根据**【说明】**中的描述，使用表 3-1 中的英文名称，给出图 3-1 中 A1～A4 所对应的参与者。

【问题 2】（3 分）

根据**【说明】**中的描述，使用表 3-2 中的英文名称，给出图 3-1 中 U1～U3 所对应的用例。

【问题 3】（4 分）

根据**【说明】**中的描述，给出图 3-1 中（1）和（2）所对应的关系。

【问题 4】（4 分）

根据**【说明】**中的描述，使用表 3-2 和表 3-3 中的英文名称，给出图 3-2 中 Action1～Action4 对应的活动。

试题四（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】

某餐厅供应各种标准的营养套餐。假设菜单上共有 n 项食物 m_1，m_2，…，m_n，每项食物 m_i 的营养价值为 v_i，价格为 p_i，其中 i=1,2,…,n，套餐中每项食物至多出现一次。客人常需要一个算法来求解总价格不超过 M 的营养价值最大的套餐。

【问题 1】（9 分）

下面是用动态规划策略求解该问题的伪代码，请填充其中的空缺(1)、(2)和(3)处。

伪代码中的主要变量说明如下：

n： 总的食物项数；

v： 营养价值数组，下标从 1 到 n，对应第 1 到第 n 项食物的营养价值；

p： 价格数组，下标从 1 到 n，对应第 1 到第 n 项食物的价格；

M：总价格标准，即套餐的价格不超过 M；

x： 解向量（数组），下标从 1 到 n，其元素值为 0 或 1，其中元素值为 0 表示对应的食物不出现在套餐中，元素值为 1 表示对应的食物出现在套餐中；

nv：n+1 行 M+1 列的二维数组，其中行和列的下标均从 0 开始，nv[i][j]表示由前 i 项食物组合且价格不超过 j 的套餐的最大营养价值。问题最终要求的套餐的最大营养价值为 nv[n][M]。

伪代码如下：

```
MaxNutrientValue(n, v, p, M, x)
1   for i = 0 to n
2       nv[i][0] = 0
3   for j = 1 to M
4       nv[0][j] = 0
5   for i = 1 to n
6       for j = 1 to M
7           if j < p[i]   //若食物 mi 不能加入到套餐中
8               nv[i][j] = nv[i - 1][j]
9           else if ____（1）____
10              nv[i][j] =   nv[i - 1][j]
11          else
12              nv[i][j] =   nv[i - 1][j – p[i]] + v[i]
13  j = M
14  for i = n downto 1
15      if ____（2）____
16          x[i] = 0
17      else
18          x[i] = 1
19          ____（3）____
20  return x and nv[n][M]
```

【问题 2】（4 分）

现有 5 项食物，每项食物的营养价值和价格如表 4-1 所示。

表 4-1 食物营养价值及价格表

编　　码	营养价值	价　　格
m_1	200	50
m_2	180	30
m_3	225	45
m_4	200	25
m_5	50	5

若要求总价格不超过 100 的营养价值最大的套餐，则套餐应包含的食物有__（4）__（用食物项的编码表示），对应的最大营养价值为__（5）__。

【问题 3】（2 分）

【问题 1】中伪代码的时间复杂度为__（6）__（用 O 符号表示）。

从下列的 3 道试题（试题五至试题七）中任选 1 道解答。
如果解答的试题数超过 1 道，则题号小的 1 道解答有效。

试题五（共 15 分）

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

【说明】

已知集合 A 和 B 的元素分别用不含头结点的单链表存储，函数 Difference()用于求解集合 A 与 B 的差集，并将结果保存在集合 A 的单链表中。例如，若集合 A={5，10，20，15，25，30}，集合 B={5，15，35，25}，如图 5-1(a)所示，运算完成后的结果如图 5-1(b)所示。

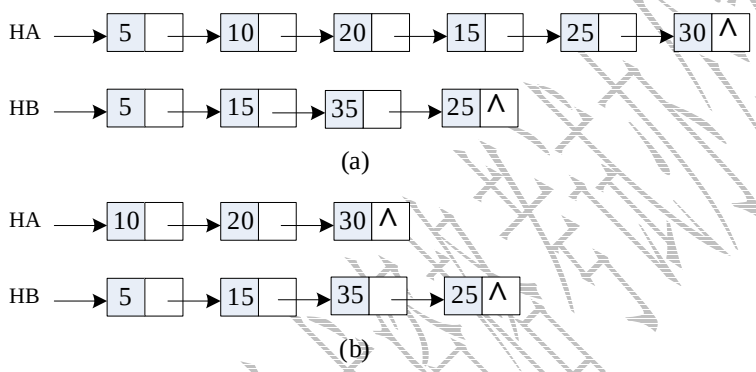

图 5-1 集合 A、B 运算前后示意图

链表结点的结构类型定义如下：

```
typedef struct Node{
    ElemType elem;
    struct Node *next;
}NodeType;
```

【C 函数】

```
void Difference(NodeType **LA, NodeType *LB)
{
    NodeType *pa, *pb, *pre, *q;

    pre = NULL;
    __（1）__;
    while (pa) {
        pb = LB;
        while (__（2）__)
            pb = pb->next;

        if (__（3）__) {
```

```
            if (!pre)
                    *LA = ___(4)___;
            else
                    ___(5)___ = pa->next;
            q = pa;
            pa = pa->next;
            free(q);
        }
        else    {
                ___(6)___;
                pa = pa->next;
        }
    }
}
```

试题六（共 15 分）

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】

已知某类库开发商提供了一套类库，类库中定义了 Application 类和 Document 类，它们之间的关系如图 6-1 所示，其中，Application 类表示应用程序自身，而 Document 类则表示应用程序打开的文档。Application 类负责打开一个已有的以外部形式存储的文档，如一个文件，一旦从该文件中读出信息后，它就由一个 Document 对象表示。

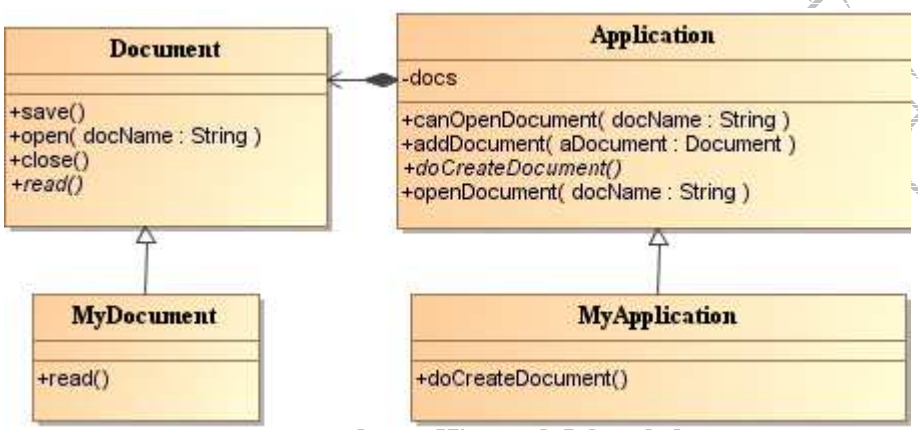

图 6-1　Application 与 Document 关系图

当开发一个具体的应用程序时，开发者需要分别创建自己的 Application 和 Document 子类，例如图 6-1 中的类 MyApplication 和类 MyDocument，并分别实现 Application 和 Document 类中的某些方法。

已知 Application 类中的 openDocument 方法采用了模板方法（Template Method）设计模式，该方法定义了打开文档的每一个主要步骤，如下所示：

1．首先检查文档是否能够被打开，若不能打开，则给出出错信息并返回；

2．创建文档对象；

3．通过文档对象打开文档；

4．通过文档对象读取文档信息；

5．将文档对象加入到 Application 的文档对象集合中。

【C++代码】

```
#include <iostream>
#include <vector>
using namespace std;

class Document{
public:
    void save(){ /*存储文档数据，此处代码省略*/ }
```

```
    void open(string docName){   /* 打开文档，此处代码省略 */ }
    void close(){   /* 关闭文档，此处代码省略*/   }
    virtual void read(string docName) = 0;
};

class Appplication{
private:
    vector <  （1）  >   docs;     /*文档对象集合*/
public:
    bool canOpenDocument(string docName){
        /*判断是否可以打开指定文档，返回真值时表示可以打开，
        返回假值表示不可打开，此处代码省略*/
    }
    void addDocument(Document * aDocument){
        /*将文档对象添加到文档对象集合中*/
        docs.push_back(   （2）   );
    }
    virtual Document * doCreateDocument() = 0; /*创建一个文档对象*/
    void openDocument(string docName){ /*打开文档*/
        if (   （3）   ){
            cout <<"文档无法打开 ！ " << endl;
            return;
        }
        ____（4）____  adoc = ____（5）____;
        ____（6）____;
        ____（7）____;
        ____（8）____;
    }
};
```

试题七（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】

已知某类库开发商提供了一套类库，类库中定义了 Application 类和 Document 类，它们之间的关系如图 7-1 所示，其中，Application 类表示应用程序自身，而 Document 类则表示应用程序打开的文档。Application 类负责打开一个已有的以外部形式存储的文档，如一个文件，一旦从该文件中读出信息后，它就由一个 Document 对象表示。

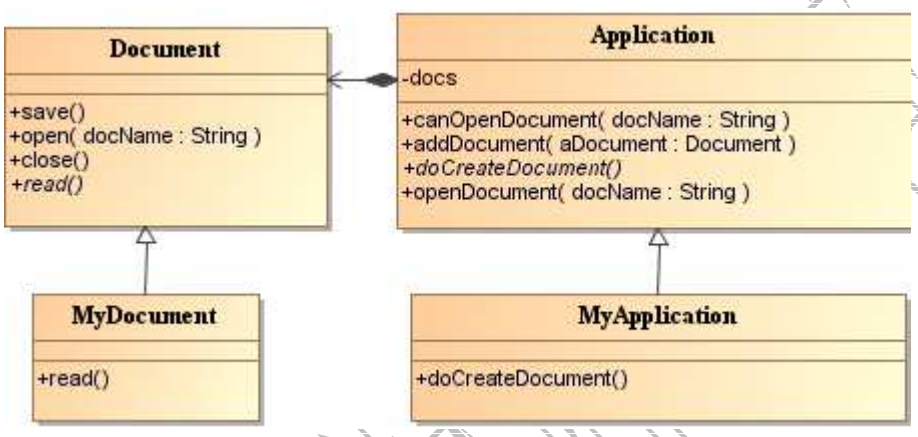

图 7-1　Application 与 Document 关系图

当开发一个具体的应用程序时，开发者需要分别创建自己的 Application 和 Document 子类，例如图 7-1 中的类 MyApplication 和类 MyDocument，并分别实现 Application 和 Document 类中的某些方法。

已知 Application 类中的 openDocument 方法采用了模板方法（Template Method）设计模式，该方法定义了打开文档的每一个主要步骤，如下所示：

1．首先检查文档是否能够被打开，若不能打开，则给出出错信息并返回；

2．创建文档对象；

3．通过文档对象打开文档；

4．通过文档对象读取文档信息；

5．将文档对象加入到 Application 的文档对象集合中。

【Java 代码】

```
abstract class Document{
      public void save(){ /*存储文档数据，此处代码省略*/     }
      public void open(String docName){    /*  打开文档，此处代码省略  */ }
      public void close(){/*  关闭文档，此处代码省略*/    }
      public abstract void read(String docName);
};
```

```
abstract class Appplication{
    private    Vector <   （1）   >   docs; /*文档对象集合*/

    public     boolean canOpenDocument(String docName){
               /*判断是否可以打开指定文档，返回真值时表示可以打开，
               返回假值表示不可打开，此处代码省略*/
    }
    public void addDocument(Document aDocument){
               /*将文档对象添加到文档对象集合中*/
               docs.add(   （2）   );
    }
    public abstract Document doCreateDocument(); /*创建一个文档对象*/
    public void openDocument(String docName){ /*打开文档*/
               if (   （3）   ){
                    System.out.println( "文档无法打开 ！ " );
                    return;
               }
               （4）  adoc =  （5）  ;
               （6）  ;
               （7）  ;
               （8）  ;
    }
};
```