

```

void quick_sort(int data[], int low, int high)
{
    int i, j, pivot;
(1)  if (low < high)
    {
        pivot=data[low];
(2)  { i=low;
        j=high;
(3)  while(i<j)
        {
            while (i<j && data[j]>=pivot)
(4)          j--;
(5)          if(i<j)
(8)          data[i++]=data[j]; //将比枢轴记录小的记录移到低端
            while (i<j && data[i]<=pivot)
(11)         i++;
(12)         if(i<j)
(13)         data[j--]=data[i]; //将比枢轴记录大的记录移到高端
(14)     }

        data[i]=pivot; //枢轴记录移到最终位置
(15) { quick_sort(data, low, i-1);
        quick_sort(data, i+1, high);
(16) }
    }
}

```

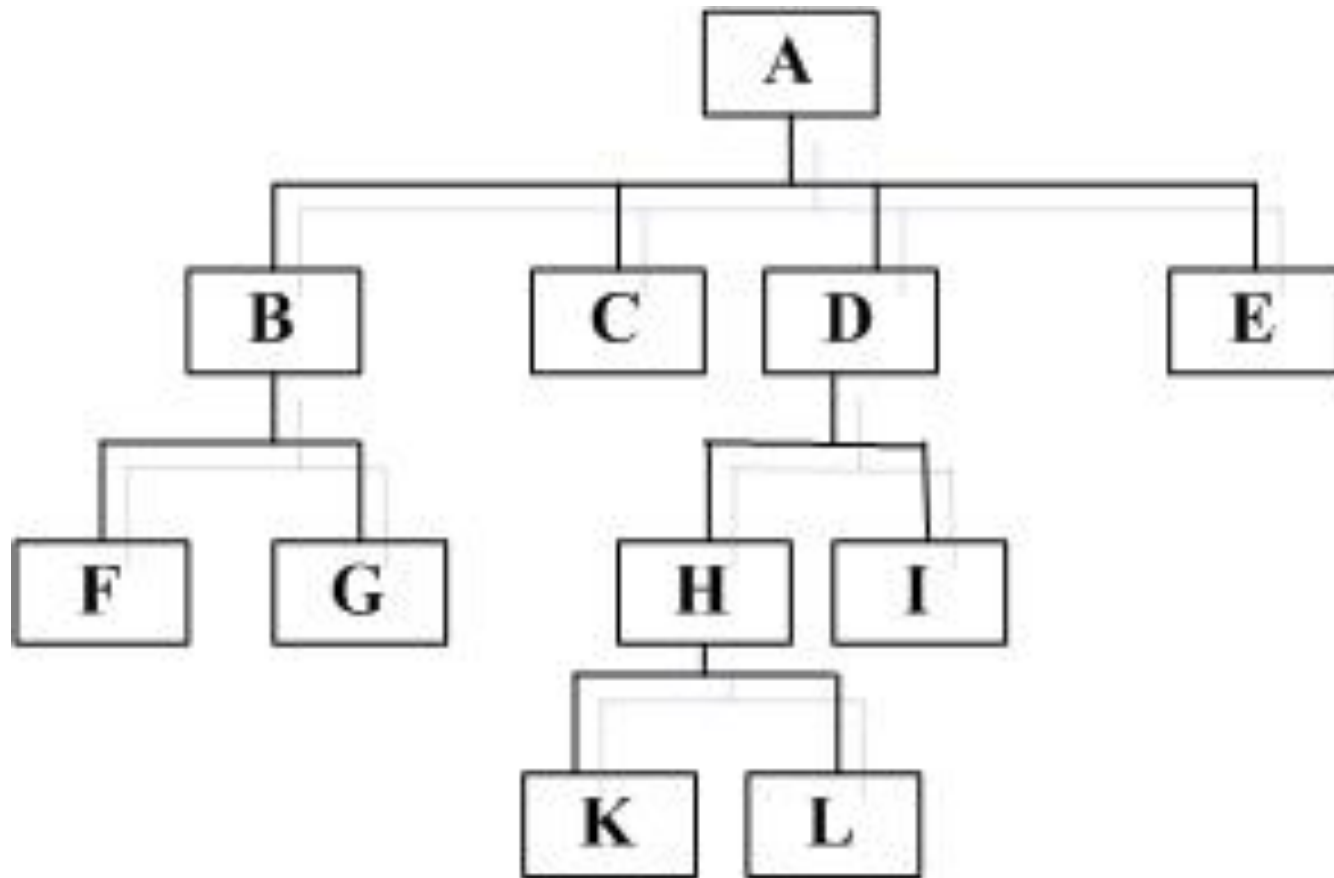
上图为一个快速排序算法的代码段，请你采用**基本路径测试**方法进行测试。

要求：

- ① 画出相应的程序控制流图。
- ② 分析该段程序的环路复杂性；
- ③ 什么是独立路径？给出该程序的一组独立路径。



- ◆ 下图是某系统的模块层次结构图，请分别给出自顶向下宽度的集成策略、自底向上集成策略、混合集成策略的步骤。

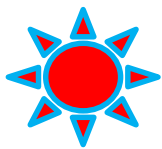


◆ BestGood公司是一家大型牛奶销售公司，有一个总部、3个仓库、5个连锁门店，每个连锁门店有2个销售经理和若干个销售员。现在BestGood公司需要开发一个信息系统，使得各个连锁门店、仓库、总部的信息能够及时上报并共享，包括：

- ◆ 总部分下达一些新的政策和规定；
- ◆ 仓库每月盘点各品种牛奶的库存并上报；
- ◆ 连锁门店要向总部填写订单，说明请求某品种牛奶和数量，总部根据各个仓库的库存量决定由哪个仓库给该门店配货；

- ◆ 总部希望对各个门店的销售情况进行统计和排名；
- ◆ 能够对操作人员的权限进行设置；

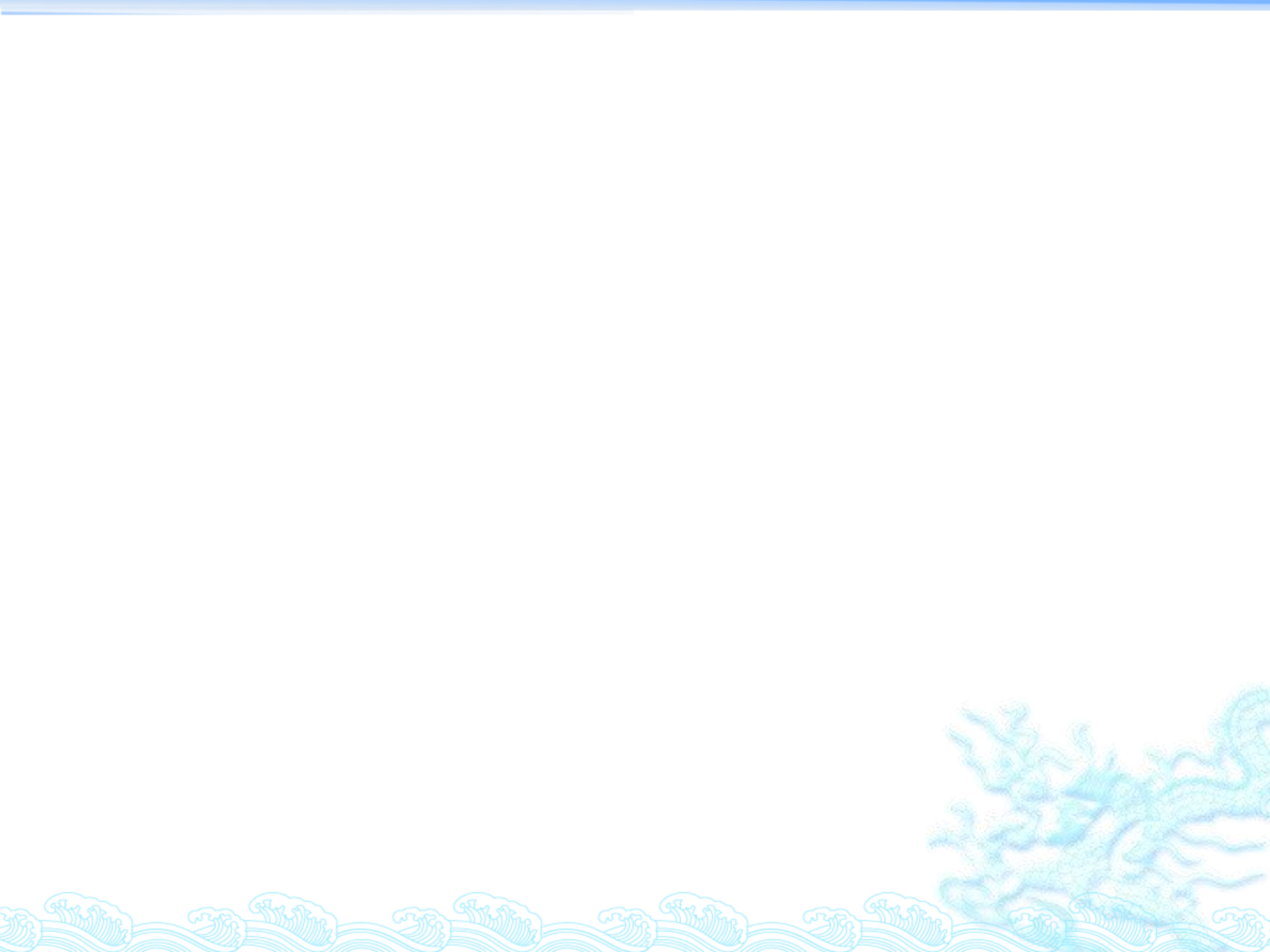
- ◆ 将来有可能开设新的门店或仓库，销售经理和销售员有可能在各个门店轮转；

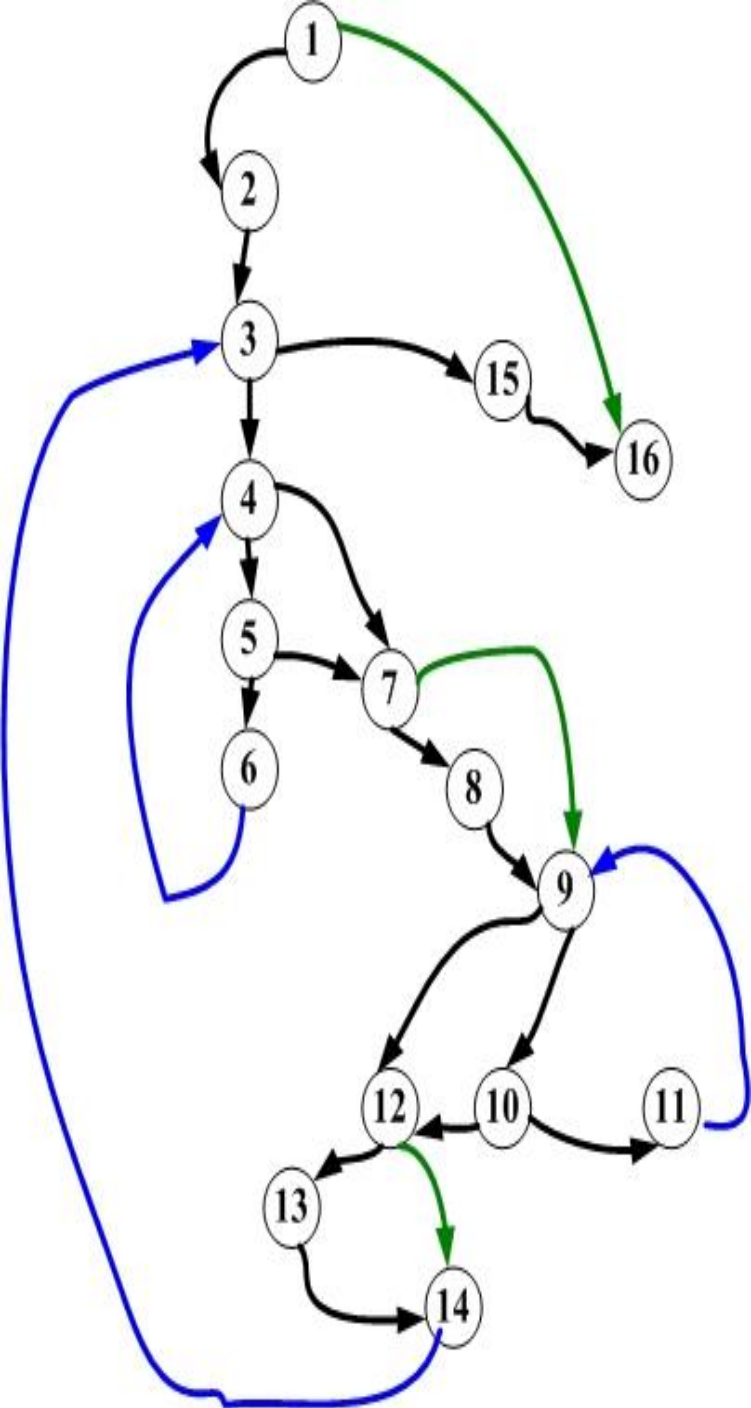


要求：

- ① 对上述的要求请给出BestGood公司信息系统的用例图。
- ② 分析并给出BestGood公司信息系统的类图，要求给出类主要的属性，并明确类之间的关系；
- ③ 给出**订单**的数据结构描述（按照数据字典的要求）。







•独立路径：至少包含一条其他路径没有的边的路径。

•Path1: 1-16

•Path2: 1-2-3-15-16

•Path3: 1-2-3-4-7-9-12-14-3-15-16

•Path4: 1-2-3-4-5-7-9-12-14-3-15-16

•Path5: 1-2-3-4-5-6-4-7-9-12-14-3-15-16

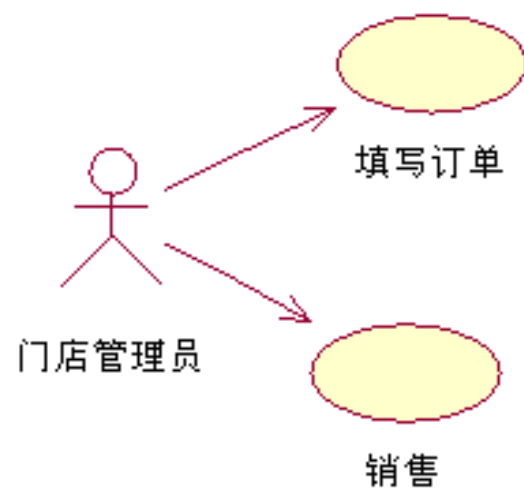
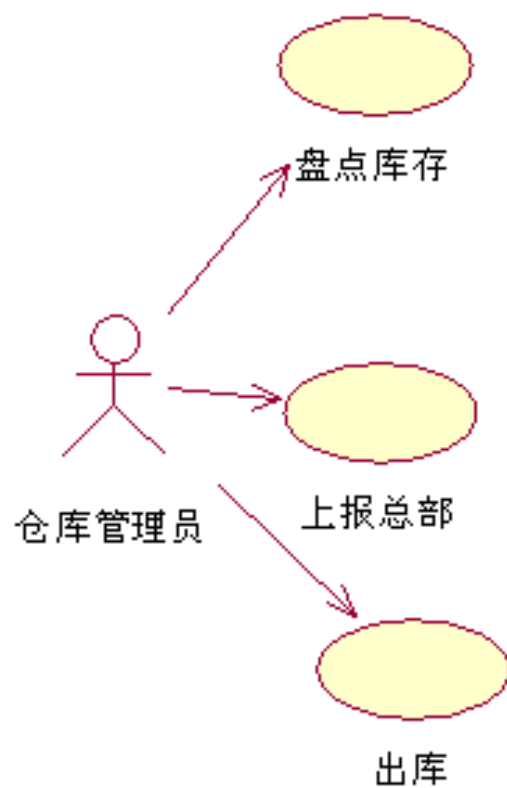
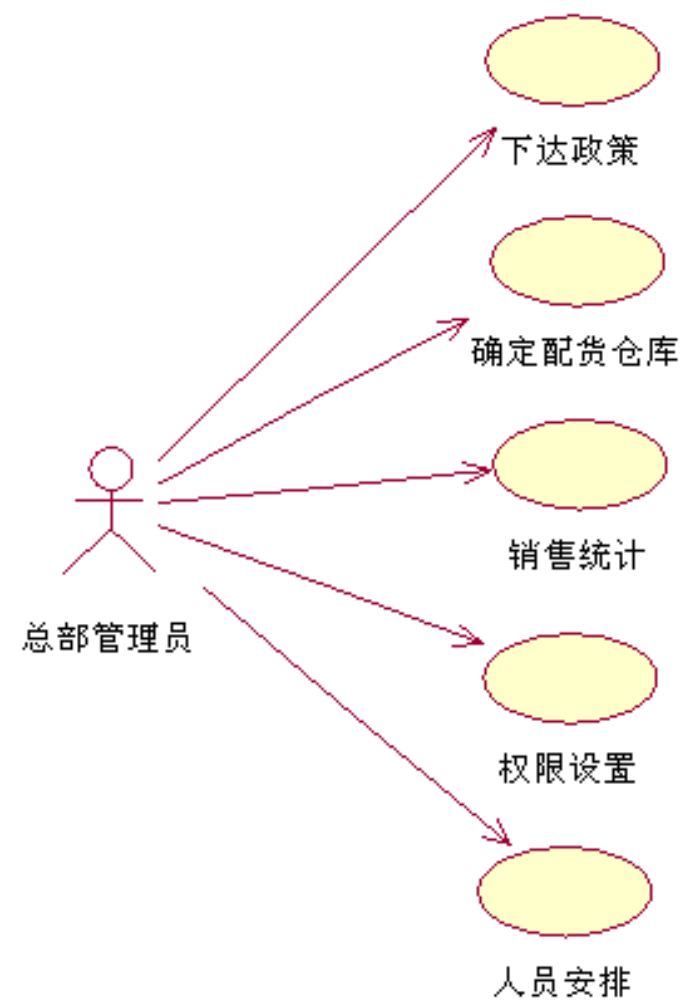
•Path6: 1-2-3-4-7-8-9-12-14-3-15-16

•Path7: 1-2-3-4-7-9-10-12-14-3-15-16

•Path8: 1-2-3-4-7-9-10-11-9-12-14-3-15-16

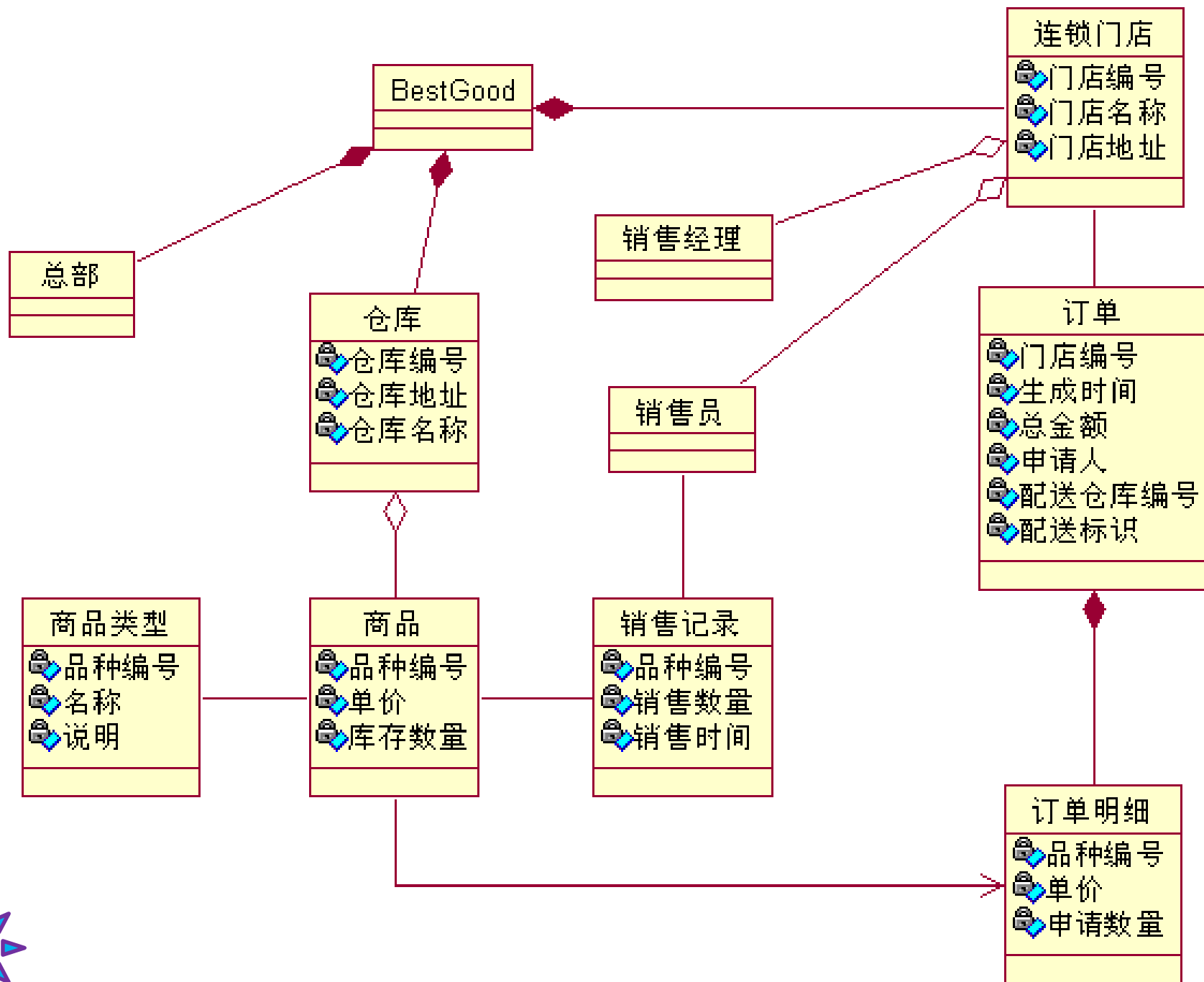
•Path9: 1-2-3-4-7-9-12-13-14-3-15-16

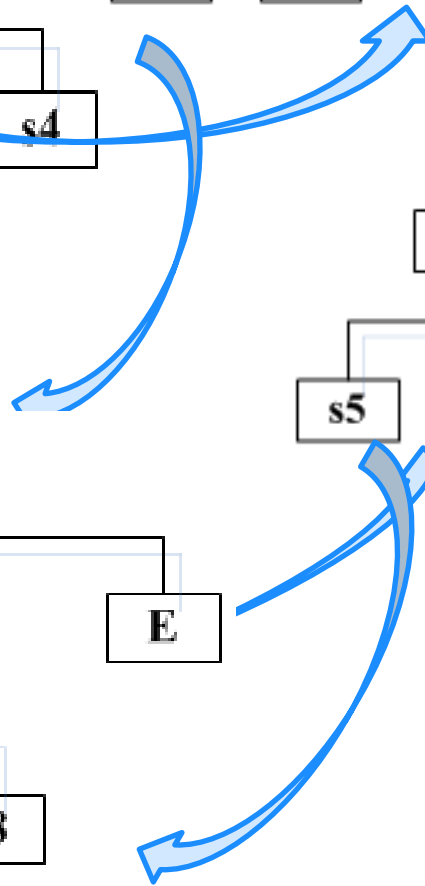
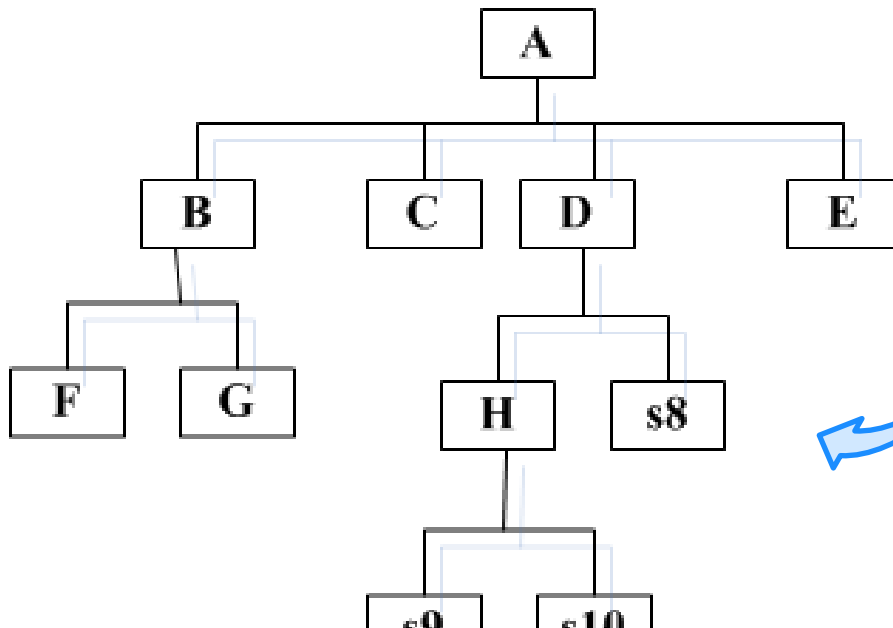
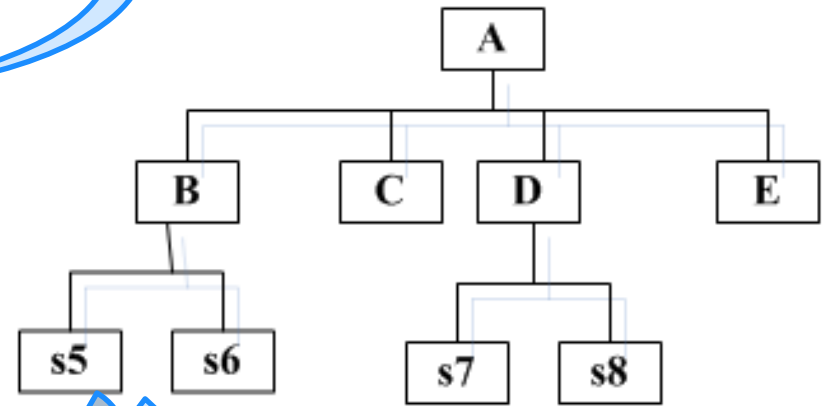
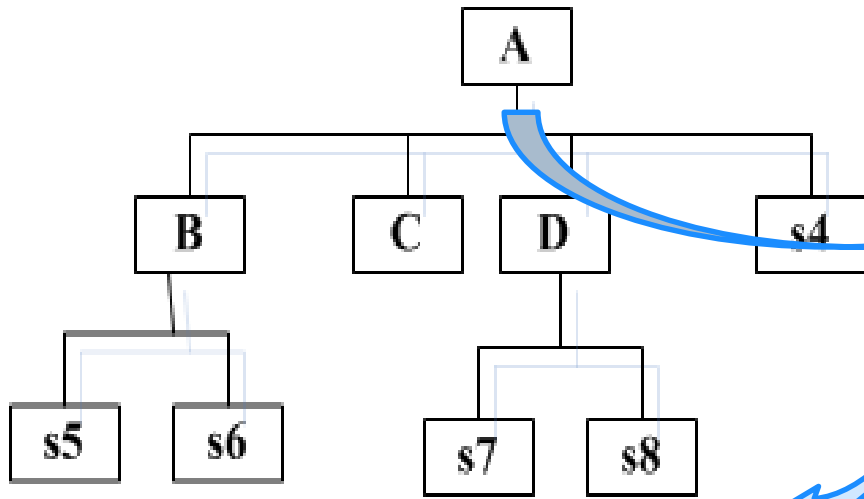
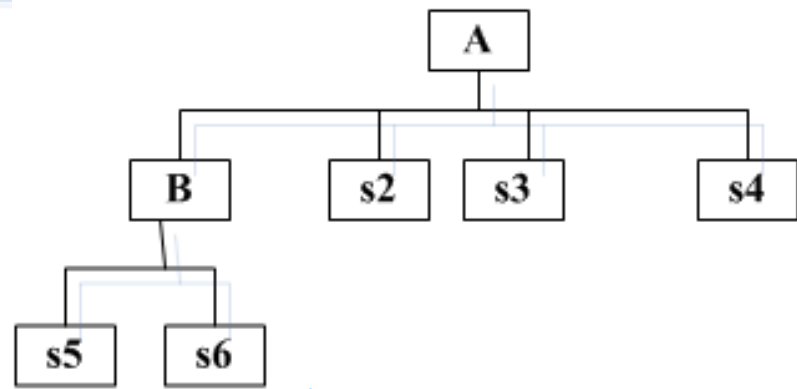
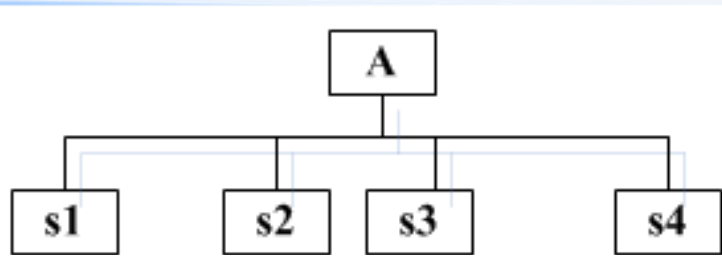


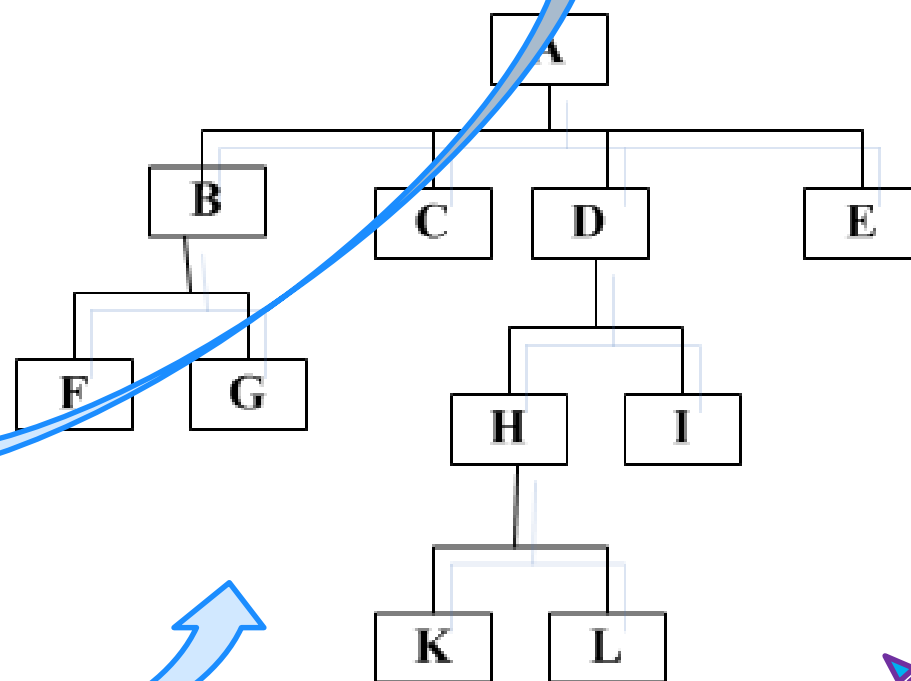
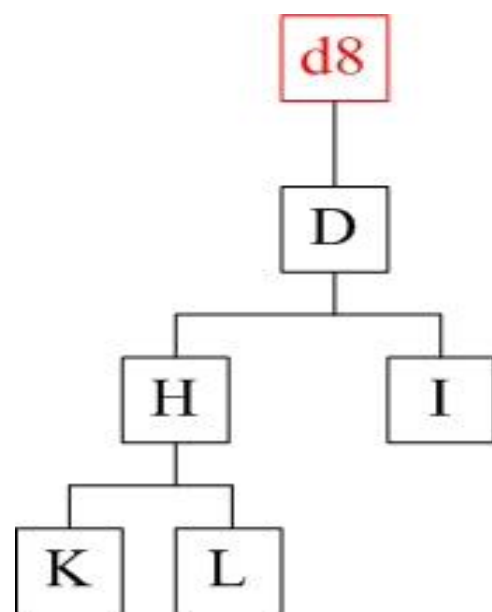
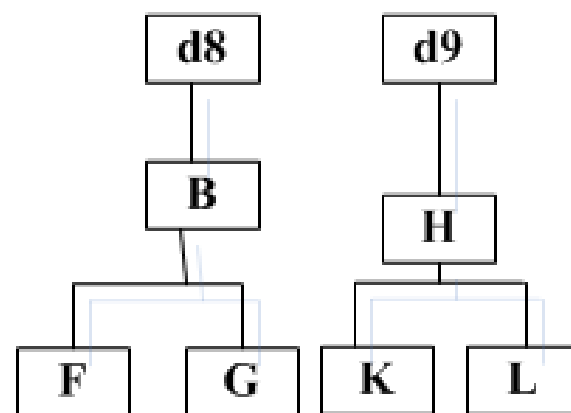
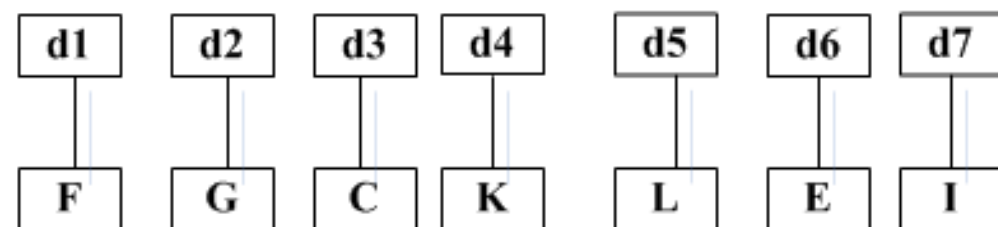


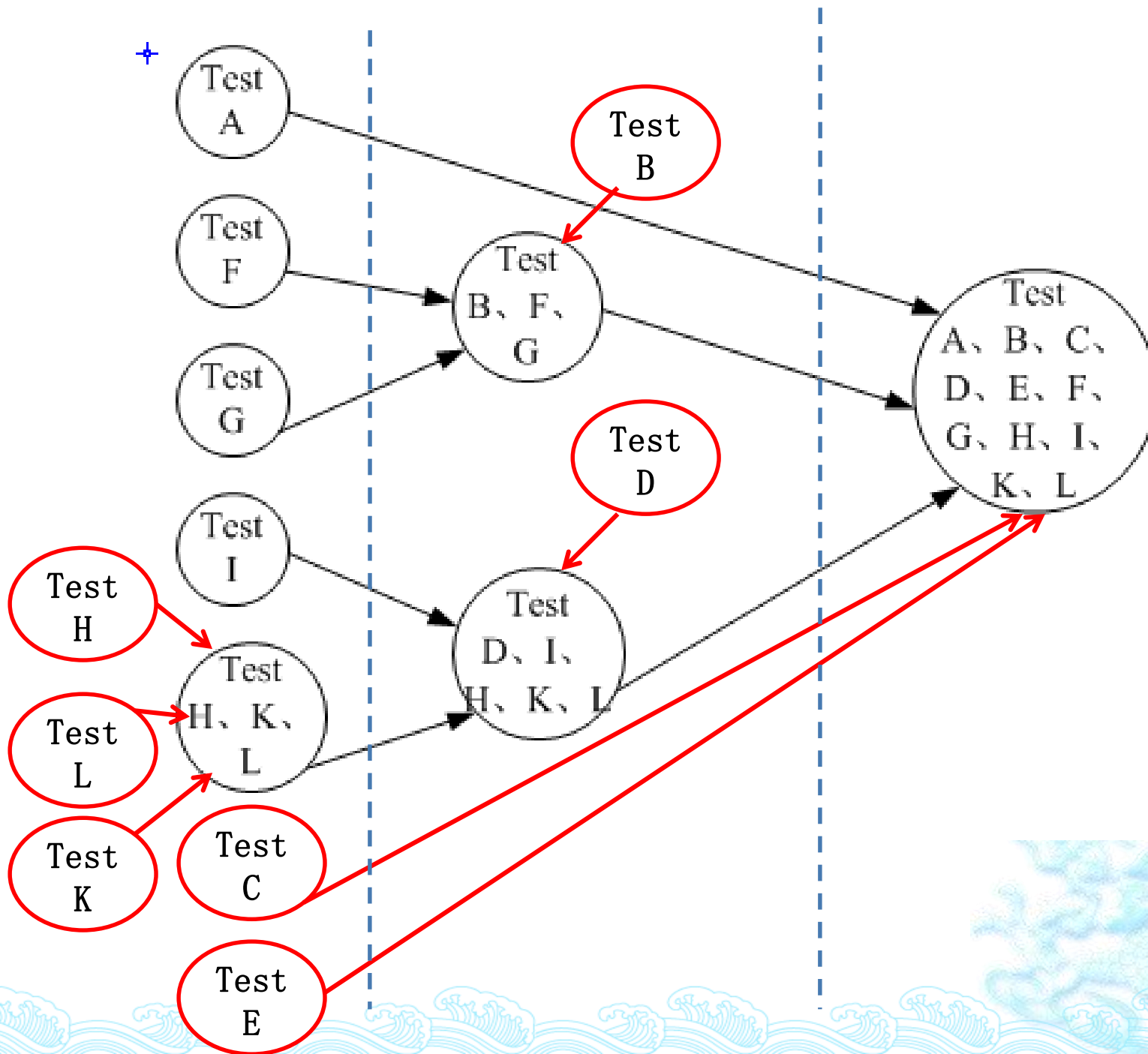
◆ BestGood公司是一家大型牛奶销售公司，有一个总部、3个仓库、5个连锁门店，每个连锁门店有2个销售经理和若干个销售员。现在BestGood公司需要开发一个信息系统，使得各个连锁门店、仓库、总部的信息能够及时上报并共享，包括：

- ◆ 总部下达一些新的政策和规定；
- ◆ 仓库每月盘点各品种牛奶的库存并上报；
- ◆ 连锁门店要向总部填写订单，说明请求某品种牛奶和数量，总部根据各个仓库的库存量决定由哪个仓库给该门店配货；
- ◆ 总部希望对各个门店的销售情况进行统计和排名；
- ◆ 能够对操作人员的权限进行设置；
- ◆ 将来有可能开设新的门店或仓库，销售经理和销售员有可能在各个门店轮转；









```
void main()
{
    int a[10];
    int i, j, t, k;
(1)   for(i=0; i<10; i++)
(2)       scanf("%d", &a[i]);
(3)   printf("\n");
(4)   for(i=0; i<9; i++)
(5)       {k=i;
(6)         for(j=i+1; j<10; j++)
(7)             if (a[j]>a[k]) k=j;
(8)
(9)         { t=a[k]; a[k]=a[i]; a[i]=t;
            }
(10)   for (i=0; i<10; i++)
(11)       printf("%d  ", a[i]);
(12)   printf("\n");
(13)}
```

```

#include<stdio.h>
#define N 10
main()
{
    int k,i;
    int table[N]={0,2,4,6,8,10,12,14,16,18};
    int mid,left=0,right=N-1;
    int find=0;
    printf("input your number.");
    scanf("%d",&k);
    while(!find&&left<right)
    {
        mid=(left+right)/2;
        if(k==table[mid])
            find=1;
        else if(k<table[mid])
            right=mid-1;
        else left=mid+1;
    }
    if(find==1)
        printf("%d in table[%d]\n",k,mid);
    else
        printf("can't find the number%d.\n",k);
}

```

Diagram annotations:

- (1) points to the initialization of `int k,i;`, `int table[N]={0,2,4,6,8,10,12,14,16,18};`, `int mid,left=0,right=N-1;`, `int find=0;`, `printf("input your number.");`, `scanf("%d",&k);`, and `while(!find&&left<right)`.
- (2) points to the opening curly brace of the `while` loop.
- (3) points to the closing curly brace of the `while` loop.
- (4) points to `mid=(left+right)/2;`
- (5) points to `if(k==table[mid])`
- (6) points to `find=1;`
- (7) points to `else if(k<table[mid])`
- (8) points to `right=mid-1;`
- (9) points to `else left=mid+1;`
- (10) points to the closing curly brace of the `while` loop.
- (11) points to `if(find==1)`
- (12) points to `printf("%d in table[%d]\n",k,mid);`
- (13) points to `else`
- (14) points to `printf("can't find the number%d.\n",k);`


```

int kmp_find(const char *text, int text_len, const char *patn, int patn_len, int *next)
{
    (1) {
        int i, j;
        assert(text != NULL && text_len > 0 && patn != NULL && patn_len > 0 &&
            next != NULL);
        for (i = 0, j = 0; i < text_len; i++)
            (2) {
                (3) while (j > 0 && text[i] != patn[j]) (12)
                    (6) j = next[j - 1];
                    (7) if (text[i] == patn[j]) (5)
                        (8) j++;
                    (9) if (j == patn_len)
                        (10) return i + 1 - patn_len;
                (11) }
            (13) return -1;
        (14) }
}

```

```

/*解码 */
void decode(linktree tree,char code[])
{
    int i=0,j=0;
    char *char0_1;
    linktree ptr=tree;
    char0_1=(char *)malloc(10*sizeof(char));/*此数组用于统计输入的0、1序列*/
    printf("霍夫曼编码-----相应字符\n\n");
    for(j=0,ptr=tree;code[i]!='\0'&&ptr->lchild;j=0,ptr=tree)
    {
        for(j=0;code[i]!='\0'&&ptr->lchild&&ptr->rchild;j++,i++)
        {
            if(code[i]=='0')
            {
                ptr=ptr->lchild;
                char0_1[j]='0';
            }
            if(code[i]=='1')
            {
                ptr=ptr->rchild;
                char0_1[j]='1';
            }
            if(!ptr->lchild&&!ptr->rchild)
            {
                char0_1[j]='\0';
                for(j=0;char0_1[j]!='\0';j++)
                {
                    printf("%c",char0_1[j]);
                    printf("\t\t%c\n",ptr->ch);
                }
                if(code[i]=='\0';ptr->lchild&&ptr->rchild)
                {
                    char0_1[j]='\0';
                    printf("没有与最后的几个0、1序列: %s相匹配的字符!\n",char0_1);
                }
            }
        }
    }
    free(char0_1);
}

```

Diagram illustrating the execution flow of the `decode` function with numbered annotations (1-27) and arrows pointing to specific code lines:

- ①: Initial setup and memory allocation.
- ②: Entry into the main loop.
- ③: Loop condition `code[i]!='\0' && ptr->lchild`.
- ④: Inner loop condition `ptr->rchild`.
- ⑤: Incrementing `i` and `j`.
- ⑥: `if(code[i]=='0')` condition.
- ⑦: `ptr=ptr->lchild;` assignment.
- ⑧: `char0_1[j]='0';` assignment.
- ⑨: `if(code[i]=='1')` condition.
- ⑩: `ptr=ptr->rchild;` assignment.
- ⑪: `char0_1[j]='1';` assignment.
- ⑫: `if(!ptr->lchild && !ptr->rchild)` condition.
- ⑬: `char0_1[j]='\0';` assignment.
- ⑭: `for(j=0; char0_1[j]!='\0'; j++)` loop.
- ⑮: `printf("%c", char0_1[j]);` statement.
- ⑯: `printf("\t\t%c\n", ptr->ch);` statement.
- ⑰: `if(code[i]=='\0'; ptr->lchild && ptr->rchild)` condition.
- ⑱: `char0_1[j]='\0';` assignment.
- ⑲: `printf("没有与最后的几个0、1序列: %s相匹配的字符!\n", char0_1);` statement.
- ⑳: `free(char0_1);` statement.
- ㉑: End of function.

某公司为本科以上学历的人重新分配工作，
原则如下：

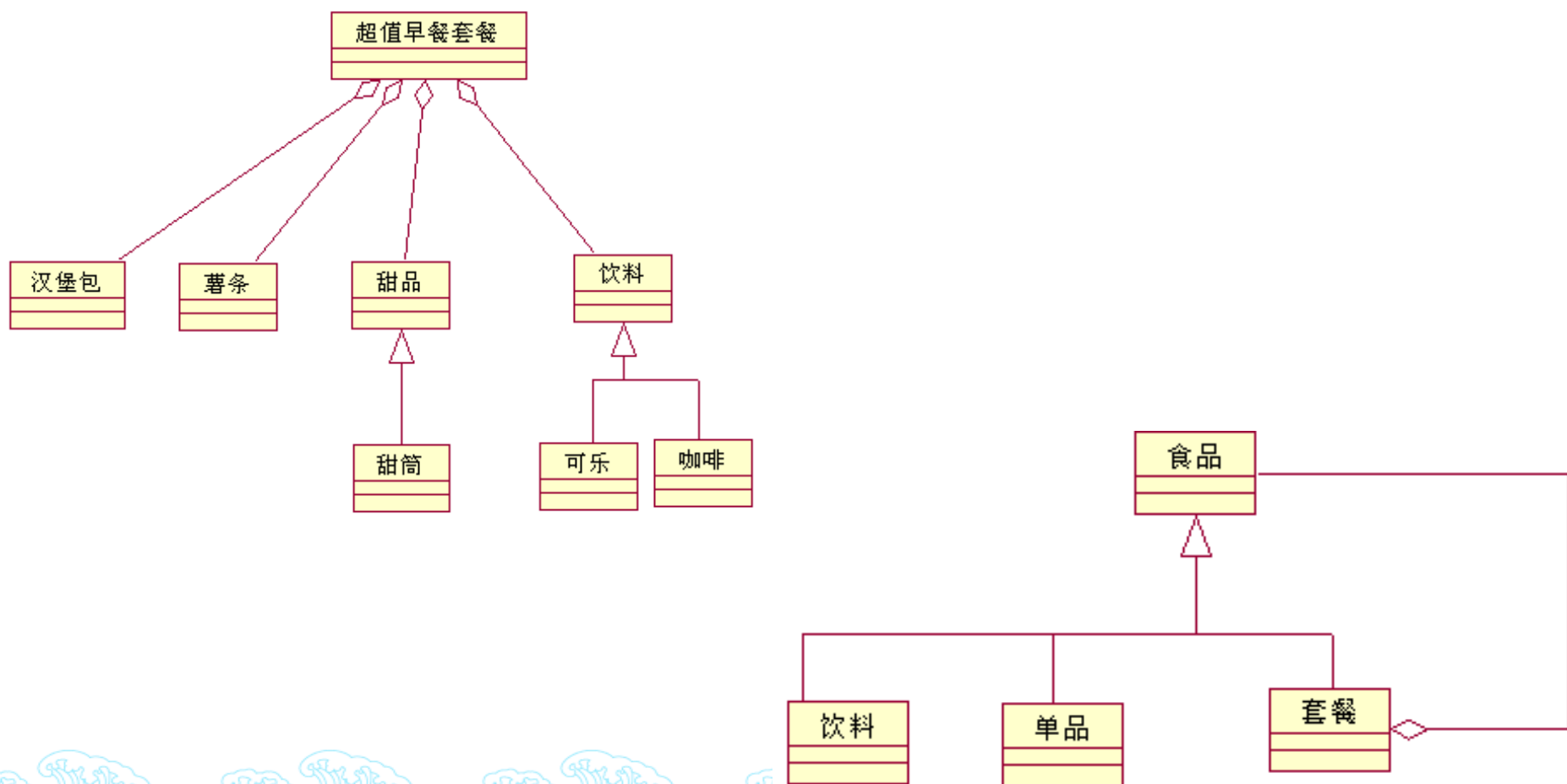
- ① 如果年龄不满23岁，学历本科，男性要求报考研究生，女性则担任行政工作。
- ② 如果年龄满23岁不满50岁，学历本科，不分男女，任中层领导职务；学历是硕士的不分男女，任课题组组长。
- ③ 如果年龄满50岁，学历本科，男性任科研工作，女性则担任资料员；学历硕士的不分男女，任课题组组长。

根据要求，给出判定表。



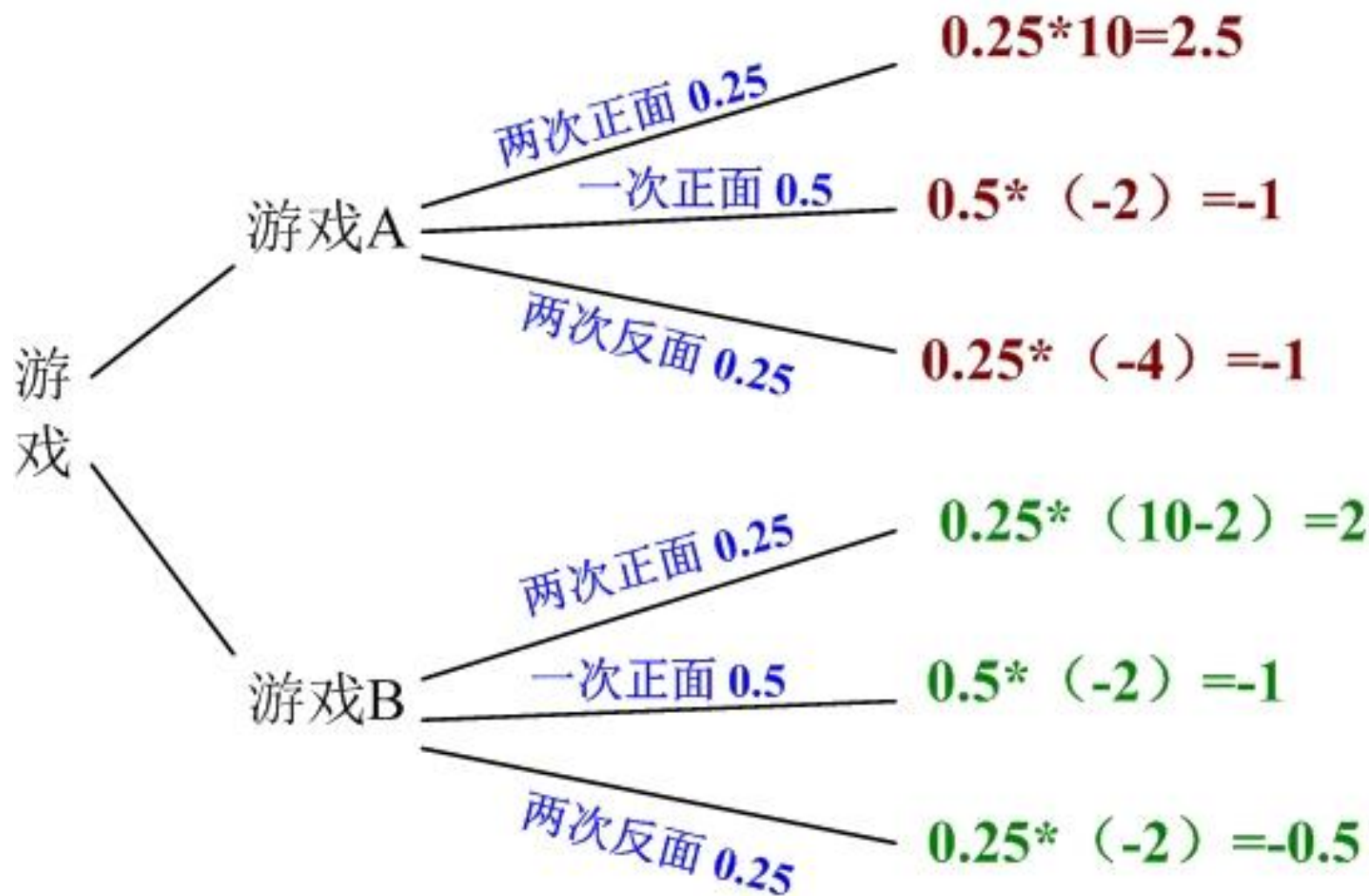
	1	2	3	4	5	6
性别为男	---	---	T	F	T	F
文化硕士	T	F	F	F	F	F
年龄<23	F	F	T	T	F	F
23≤年龄 ≤50	--	T	F	F	F	F
年龄>50	--	F	F	F	T	T
担任组长	√					
担任领导		√				
做科研					√	
做行政				√		
做资料员						√
考研			√			

用UML的类图表示：麦当劳的超值早餐套餐、薯条、咖啡、汉堡包、甜品、饮料、可乐、甜筒之间的关系。



- ◆ 一个朋友和你玩两个打赌游戏。
- ◆ 游戏A： 掷一个硬币两次，如果都是正面，他给你10元。出现一次反面，你给他2元。
- ◆ 游戏B： 还是掷一个硬币两次，但你玩一次游戏付两元，如果两次都是正面，他付你10元。
- ◆ 你的选择...





用例：ATM 机取款

主要参与者：储户


目标：储户从 ATM 机取到现金

前提条件：ATM 机正常工作，储户拥有正常的储蓄卡

触发器：储户决定从 ATM 机中取款

后置条件：相应现金从 ATM 机中吐出，储户储蓄卡中余额被相应扣除

基本事件流：

- 1.储户：将储蓄卡插入 ATM 机
 - 2.ATM：验证储蓄卡的状态。
 - 3.储户：输入密码
 - 4.储户：选择“取款”功能
 - 5.储户：输入金额
 - 6.ATM：将储蓄卡余额扣除相应金额，并吐出相应金额的现金
 - 7.储户：选择“结束”并选择“打印回单”
 - 8.ATM：打印回单并退出储蓄卡
- 

异常：

1.ATM 验证储蓄卡状态失败：换卡。

2.储户输入密码出错：如果输入错误密码的次数<3 次，则继续回到输入密码状态；否则 ATM 将储蓄卡吞掉。

3.储户输入的取款金额>储蓄卡余额，或储户输入的取款金额>2000：

重新输入取款金额。

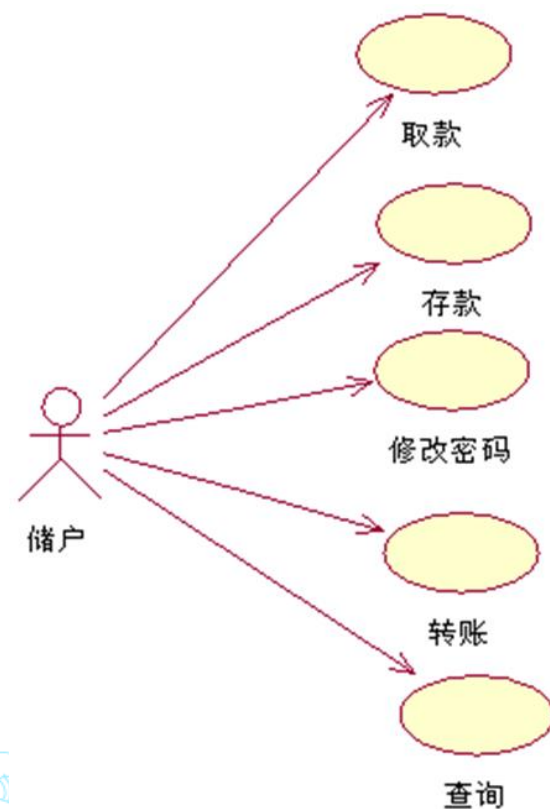
4. 储户输入的取款金额>ATM 机内现金余额：重新输入取款金额。

优先级：必须的。

何时可用：首次增量。

使用频率：每天多次。

使用方式：通过 ATM 机。



用例：ATM 机取款

主要参与者：储户

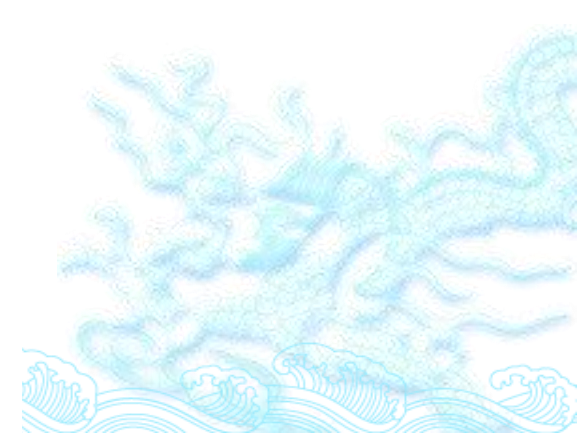
目标：储户从 ATM 机取到现金

前提条件：ATM 机正常工作，储户拥有正常的储蓄卡并且储户已登录系统进入功能选择主菜单

触发器：储户决定从 ATM 机中取款

后置条件：相应现金从 ATM 机中吐出，储户储蓄卡中余额被相应扣除

基本事件流：

- 1.储户：选择“取款”功能
 - 2.储户：输入金额
 - 3.ATM：将储蓄卡余额扣除相应金额，并吐出相应金额的现金
 - 4.储户：选择“结束”并选择“打印回单”
 - 5.ATM：打印回单并回到功能选择主菜单
- 

异常:

1. 储户输入的取款金额 > 储蓄卡余额, 或储户输入的取款金额 > 2000:

重新输入取款金额。

2. 储户输入的取款金额 > ATM 机内现金余额: 重新输入取款金额。

优先级: 必须的。

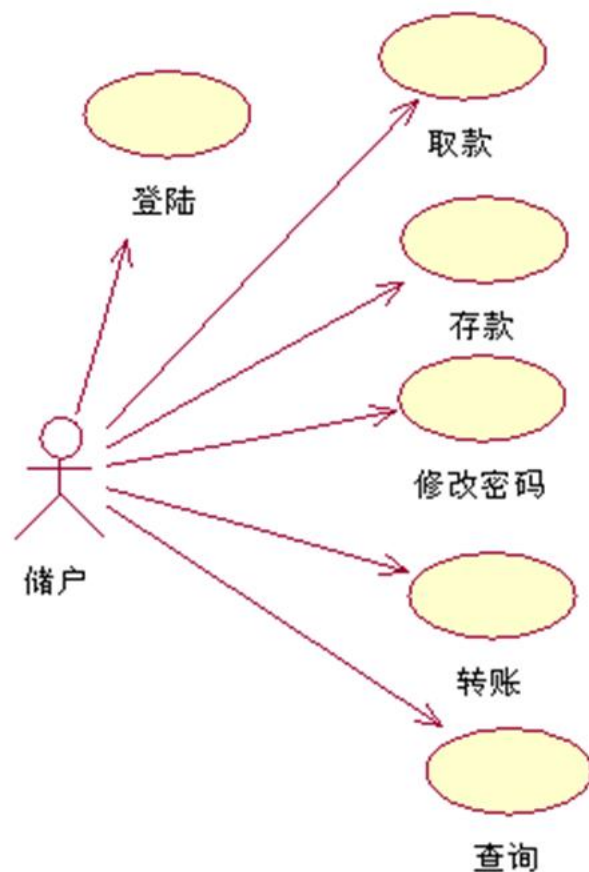
何时可用: 首次增量。

使用频率: 每天多次。

使用方式: 通过 ATM 机。

次要参与者: ATM 机

未解决的问题:



用例：图书借阅

主要参与者：读者，图书管理员

目标：管理员对读者要借阅的图书进行借阅或预定。

前提条件：系统已经输入密码并能识别读者借书权限。

触发器：读者决定借书，即进入借阅系统。

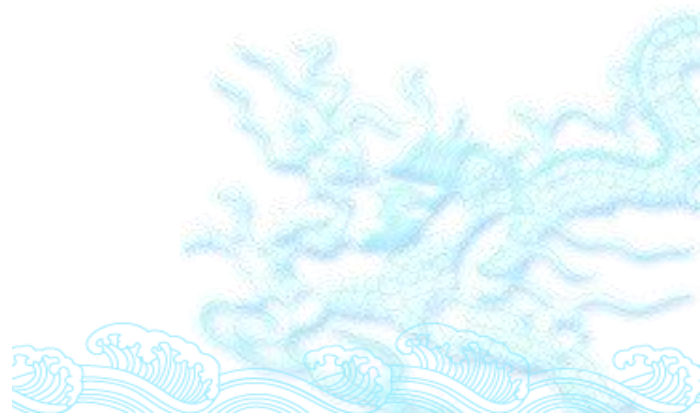
后置条件：读者结束借书

基本事件流：

- 1.读者：进入借书系统。
- 2.读者：输入用户名和密码。
- 3.读者：检索目标图书。
- 4.读者：选择借阅目标图书。
- 5.管理员：进入借书系统。
- 6.管理员：验证读者的身份和权限。
- 7.管理员：将图书的状态设为“借出”，并填写“借阅记录”，增加该读者的“已借阅图书”数量，将图书递交读者。

扩展点：该图书已被借出，读者可以选择“预定图书”，阅读用例：

“预定图书”



异常:

- 1.借书系统连接失败: 再次点击“借阅”进入借书系统。
- 2.目标图书阅读权限高于读者权限: 系统提示“由于权限不够, 无法借阅”。
- 3.用户名和密码不正确: 读者重新输入正确的用户名和密码。
- 4.读者有逾期未还的记录, 不允许借书。
- 5.读者所借图书的数量已经超过允许的数量, 不允许借书。

6.验证读者身份失败: 返回重新操作。

7.设置图书状态出错: 返回重新操作。

优先级: 必须的。

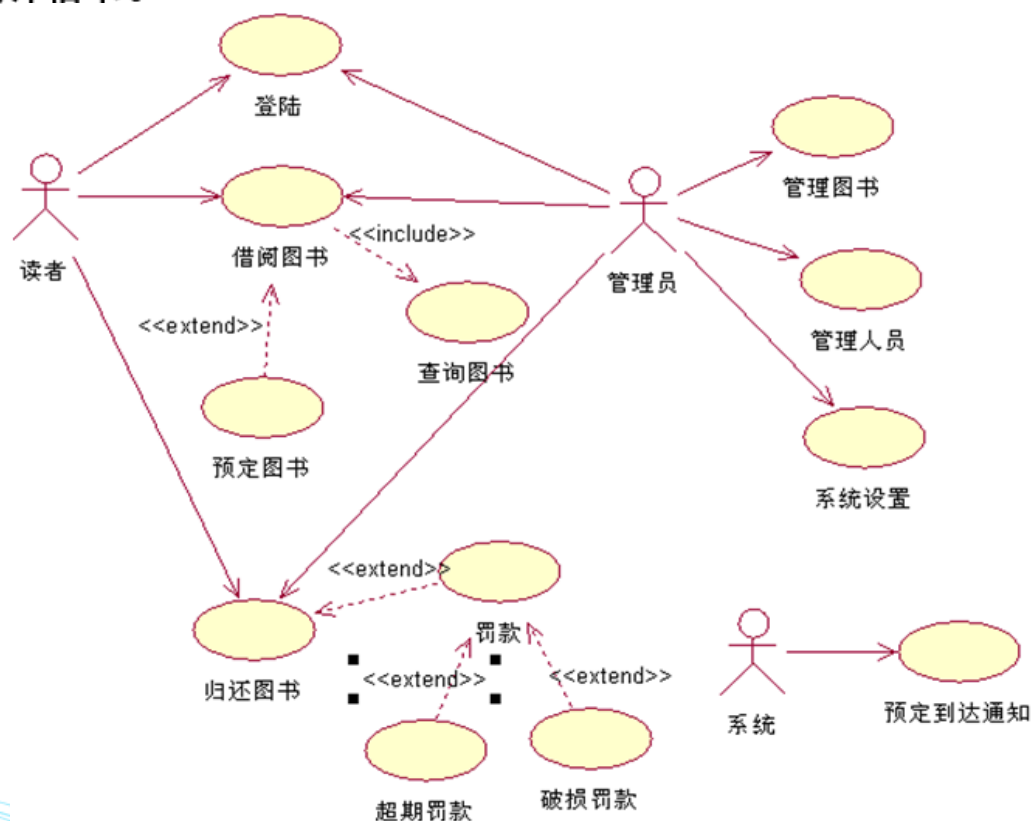
何时可用: 首次增量。

使用频率: 每天多次。

使用方式: 通过图书馆系统客户端。

次要参与者:

次要参与者使用方式:



用例：图书借阅

主要参与者：图书管理员

目标：管理员对读者要借阅的图书进行借阅登记。

前提条件：读者已经选中图书或已经通过系统预定的图书。

触发器：读者决定借书或系统通知已办理预定图书的读者前来借书。

后置条件：图书被借出，状态被置为“借出”

基本事件流：

- 1.管理员：进入借书系统。
- 2.管理员：验证读者的身份和权限。
- 3.管理员：将图书的状态设为“借出”，并填写“借阅记录”，增加该读者的“已借阅图书”数量，将图书递交读者。
- 4.管理员：结束借书。

扩展点：该图书已被借出，并且读者希望预定，读者可以选择“预定图书”，阅读用例：“预定图书”

异常:

- 1.借书系统连接失败: 再次点击“借阅”进入借书系统。
- 2.目标图书阅读权限高于读者权限: 系统提示“由于权限不够, 无法借阅”。
- 3.用户名和密码不正确: 读者重新输入正确的用户名和密码。
- 4.读者有逾期未还的记录, 不允许借书。
- 5.读者所借图书的数量已经超过允许的数量, 不允许借书。

6.验证读者身份失败: 返回重新操作。

7.设置图书状态出错: 返回重新操作。

优先级: 必须的。

何时可用: 首次增量。

使用频率: 每天多次。

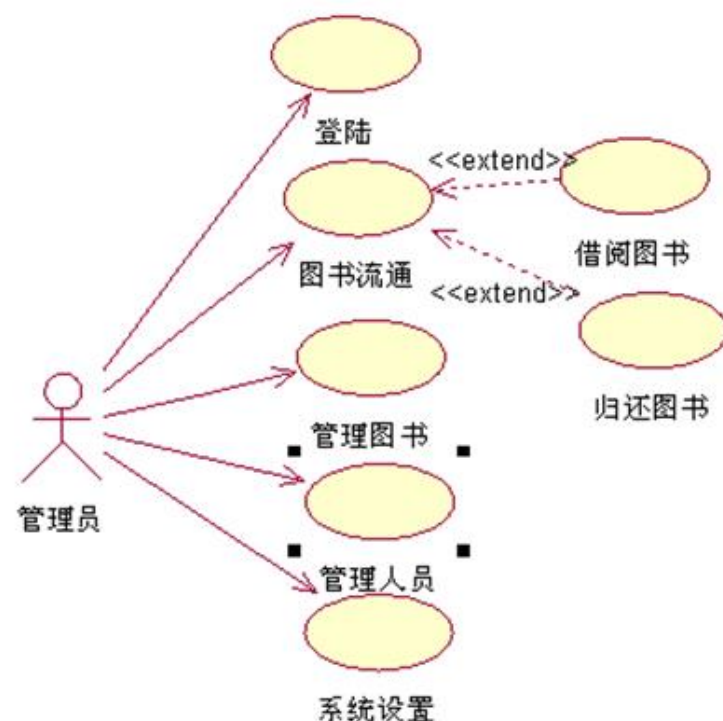
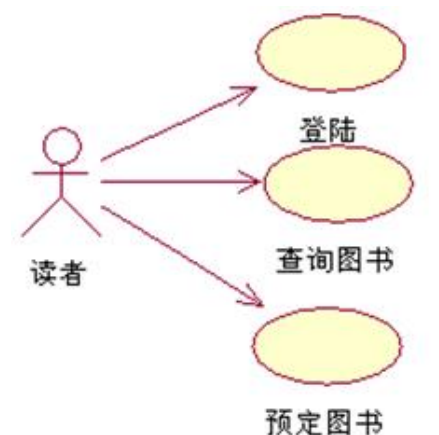
使用方式: 通过图书馆系统客户端。

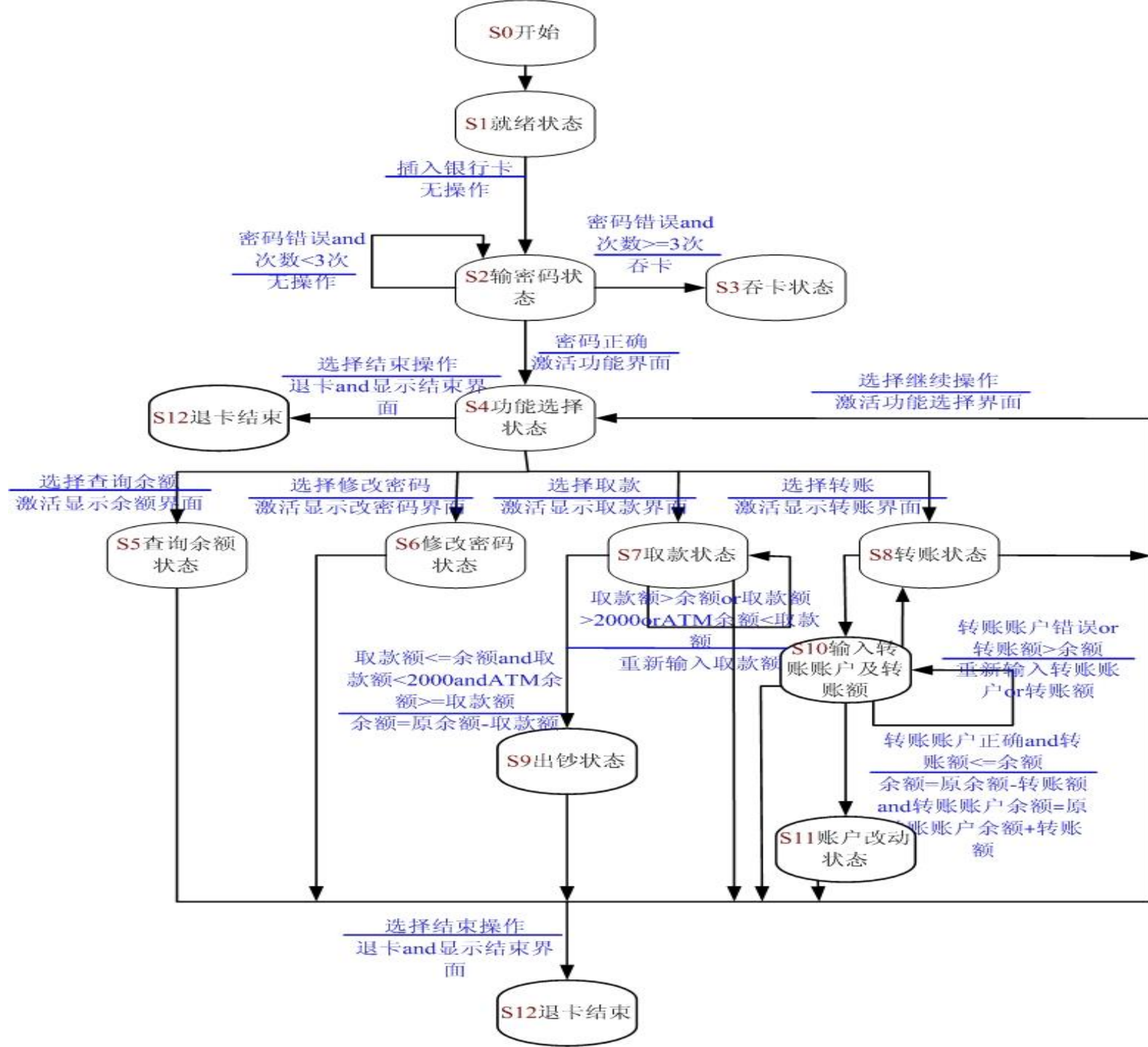
次要参与者: 读者, 图书馆系统客户端。

次要参与者使用方式:

读者: 通过语言交流。

图书馆系统客户端: 电脑。






原状态	新状态	迁移条件	迁移时的动作
S0	S1	无	无
S1	S2	插入银行卡有效	无
S2	S2	密码错误and次数<3次	无
S2	S3	密码错误and次数>=3次	吞卡
S2	S4	密码正确	激活功能选择界面
S4	S5	选择查询余额	激活显示余额界面
S4	S6	选择修改密码	激活改密码界面
S4	S7	选择取款	激活显示取款界面
S4	S8	选择转账	激活显示转账界面
S7	S7	取款额>余额or取款额>2000orATM余额<取款额	重新输入取款额
S7	S9	取款额<=余额and取款额<2000andATM余额>=取款额	余额=原余额-取款额
S8	S10	无	激活输入转账账户和转账额界面
S10	S10	输入转账账户错误or转账额>余额	重新输入转账账户和转账额
S10	S11	输入转账账户正确and账额<=余额	余额=原余额-转账额 转账账户余额=原转账账户余额+转账额
S5、S6、S7、S8、S9、S11	S4	选择继续操作	激活功能选择界面
S5、S6、S7、S8、S9、S11	S12	选择结束操作	退卡and激活结束界面

3、某个学校临近暑假就会收到许多入学申请，该校对入学申请的结论主要有：同意和拒绝。如果该生的成绩为优秀并且导师推荐级别为“良”以上，社会活动积极，则同意该生入学；如果该生的成绩为优秀，导师推荐级别为“良”以上，社会活动不积极，则同意该生入学；如果该生的成绩为优秀，导师推荐级别为“良”以下，社会活动不积极，则拒绝该生入学；如果该生的成绩为优秀，导师推荐级别为“良”以下，社会活动积极，则同意该生入学；如果该生的成绩不优秀，导师推荐级别为“良”以上，社会活动积极，则同意该生入学；如果该生的成绩不优秀，导师推荐级别为“良”以上，社会活动不积极，则拒绝该生入学；如果该生的成绩不优秀，导师推荐级别为“良”以下，社会活动不积极，则拒绝该生入学；如果该生的成绩不优秀，导师推荐级别为“良”以下，社会活动积极，则拒绝该生入学；

	Rule1	Rule2	Rule3	Rule4	Rule5	Rule6	Rule7	Rule8
成绩优秀	T	T	T	F	F	F	F	T
导师推荐 为“良”	T	T	F	T	F	F	T	F
社会活动 积极	T	F	F	F	T	F	F	T
同意入学	√	√						√
拒绝入学			√		√	√	√	

订单详情

收货人: 
收货地址: 福建省 福州 鼓楼区 2605

fancyfix创意坊专卖店



商品总价
运费(快递)
店铺优惠
订单总价

实付款

投诉商家

积分 返积分5点

更多 开票申

订单=订单编号+顾客编号+订单生成时间+订单状态+送货地址+付款方式+联

系方式+1{订单项目}100

订单编号= “000000000” .. “999999999”

顾客编号= “000000000” .. “999999999”

订单生成时间=年+月+日+小时+分钟+秒

年= “1900” .. “9999”

月= “1” .. “12”

日= “1” .. “31”

小时= “00” .. “23”

分钟= “00” .. “59”

秒= “00” .. “59”

订单状态= “1” .. “6” 其中, “1” 表示新建订单; “2” 表示订单已付款, 尚未发货; “3” 表示订单已付款, 已发货; “4” 表示订单未付款, 已发货; “5” 表示完成; “6” 表示该订单有商品退货。

送货地址=1{字母}400

付款方式=“1”..“3” 其中，“1”表示网上银行；“2”表示邮局汇款；“3”表示货到付款。

联系方式=[固定电话|手机]

固定电话=1{数字}4+8{数字}8

手机=11{数字}11

订单项目=商品编号+商品名称+商品单价+订购数量

商品编号=1{字母}2+编号

编号=“00000”..“99999”

商品名称=1{字母}40

商品单价=0.01..99999.99

订购数量=1..9999

