

---

華中科技大學

# 課程設計報告

題目： 樓盤管理系统

課程名稱： C 语言課程設計

專業班級： ACM1501 班

學 號： U201514721

姓 名： 吳肇敏

指導教師： 盧 萍

報告日期： 2016 年 9 月 30 日

計算機科學與技術學院

# 课程设计任务书

## 题目(一) 楼盘查询系统

### (1) 主要内容

建立楼盘楼盘信息系统，提供创建、编辑和综合查询等基本业务管理和服  
务。

### (2) 任务要求

收集与阅读相关文献资料，确定系统目标与范围，分析系统需求，确定系  
统功能；设计系统方案，完成系统实现；提交《课程设计报告》。

### (3) 参考文献

[1]曹计昌,卢萍,李开. C 语言程序设计,北京: 科学出版社,2013  
[2]李开,卢萍,曹计昌. C 语言实验与课程设计,北京: 科学出版社,2011  
[3]张引. C 程序设计基础课程设计, 杭州: 浙江大学出版社, 2007  
[4]黄明, 梁旭, 万洪莉. C 语言课程设计, 北京: 电子工业出版社, 2006

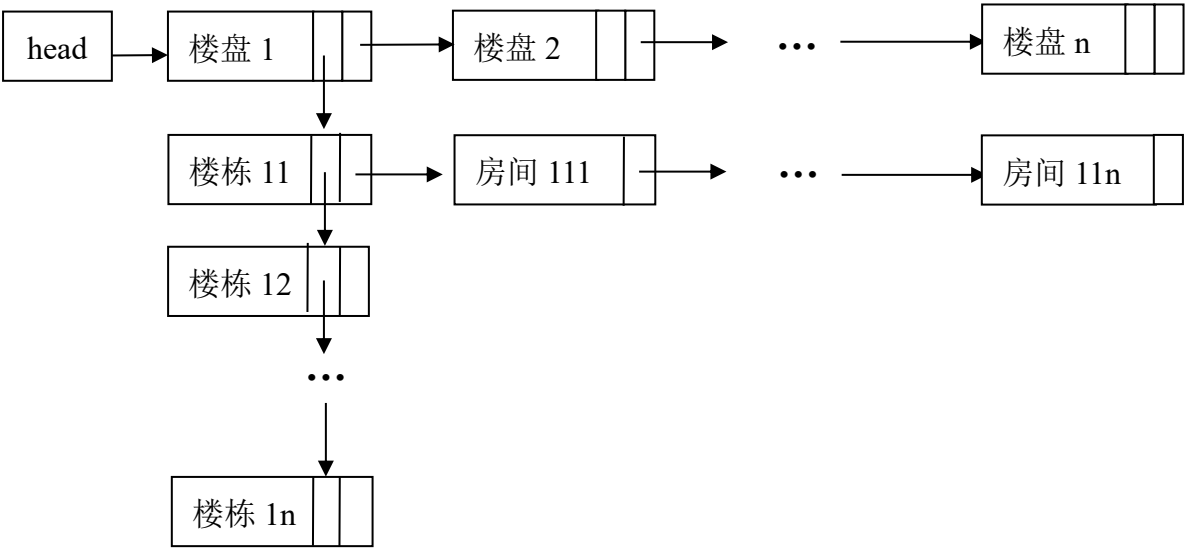


图 1 招生查询系统的链表结构示意图

---

## 目录

<b>1 绪论</b>	<b>1</b>
1.1 课题背景	1
1.2 课题研究的目的与意义	1
<b>2 系统整体设计</b>	<b>2</b>
2.1 文件模块	2
2.2 显示模块	5
2.3 编辑模块	7
2.4 查询模块	11
2.5 统计模块	14
2.6 帮助模块	14
<b>3 数据结构设计</b>	<b>16</b>
3.1 整体数据结构	16
3.2 辅助链表	18
3.3 其他结构体	19
<b>4 系统模块设计与实现</b>	<b>21</b>
4.1 文件模块	21
4.2 显示模块	24
4.3 编辑模块	26
4.4 查询模块	31
5.5 统计模块	34
5.6 帮助模块	34
<b>5 系统测试</b>	<b>35</b>
5.1 文件模块	35
5.2 显示模块	37
5.3 编辑模块	38
5.4 查询模块	43
5.5 统计模块	45
5.6 帮助模块	45
<b>6 系统界面设计</b>	<b>47</b>
<b>7 总结与体会</b>	<b>48</b>
<b>附录</b>	<b>49</b>
1、 源文件 (RoomSearch.cpp)	49
2、 头文件 (managerSys.h)	156

---

3、 头文件 (resource.h) .....	169
4、 资源文件 (楼盘管理系统.rc) .....	171
参考文献 .....	189

---

# 1 绪论

## 1.1 课题背景

房地产业是中国最近关注度非常高的一门产业。房地产商需要去管理整个公司的楼盘信息，而用户需要查询对应的楼盘信息，因此制作一个系统来满足双方的需求就显得尤为重要。一个方便的系统能够减少客户四处咨询的时间，同时也方便了公司对房地产的管理。因此，我们需要一个楼盘的管理系统来满足用户的需求。

## 1.2 课题研究的目的是与意义

(1) 给出了管理和查询的必要功能，可以实现基本的插入、修改、删除功能。方便客户与房地产商的使用。

(2) 实现了图形化界面的处理，模仿大部分 Windows 应用程序的设计思路，使设计更标准化，尽量使用户的体验达到更好。

## 2 系统整体设计

本实验设计的系统为楼盘信息管理系统，功能为管理不同地区的不同楼盘、楼栋、房间的信息。主体功能模块分为五个部分，如图 1.1 所示。接下来将依次介绍五个主要功能模块。

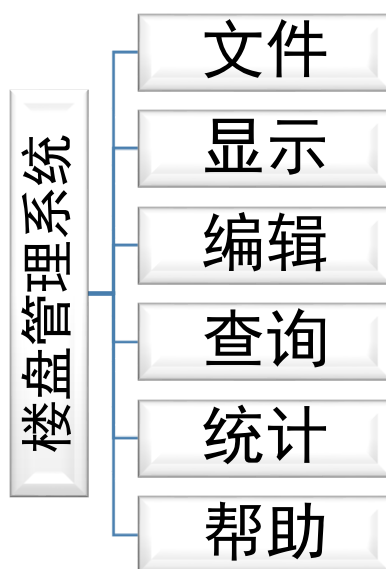


图 2.1 楼盘管理系统主体功能模块

### 2.1 文件模块

文件模块主要分为四个部分：数据文件导入、数据保存、退出系统、放弃本次修改，如图 1.2 所示。具体功能介绍如下：

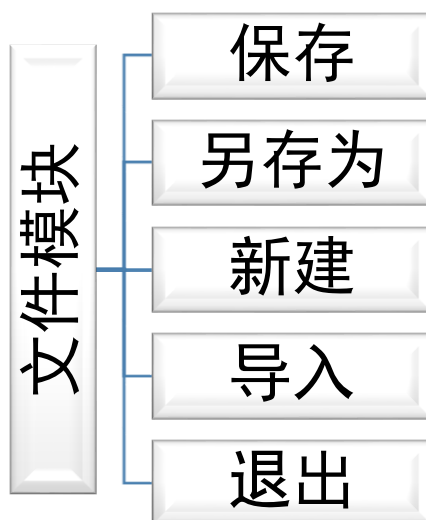


图 2.2 文件系统功能

### 2.1.1 保存

通过该功能将已经通过系统做的修改、增添和删除保存至磁盘文件。具体实现方法如下：

当用户单击菜单中的“保存”时，若磁盘若无对应文件则激活“另存为”，若磁盘中有对应文件，则直接将修改写入至磁盘对应文件。

### 2.1.2 另存为

若为“保存”激活，则保存文件至磁盘某一位置；若为直接点击，则为该文件在磁盘上创建一个副本。具体实现方法如下：

1) 当单击菜单中的“另存为”时（或对未保存至磁盘的文件单击“保存”），将弹出一个 Windows 中常用的保存窗口，如图 2.3 所示；

2) 用户可以通过这个窗口中选择将要保存的位置，以及将要保存的文件形式（目前系统仅支持\*.led 格式，该格式为自定义的引导文件格式）；

3) 用户若单击取消，则恢复原状。若用户单击确定，则文件保存至磁盘相应目录下。用户可到磁盘中查询到四个文件：一个引导文件 Leading Files(\*.led)，三个数据文件(\*.bin)，如图 2.4 所示。其文件名为系统自动确定，不允许用户更改。

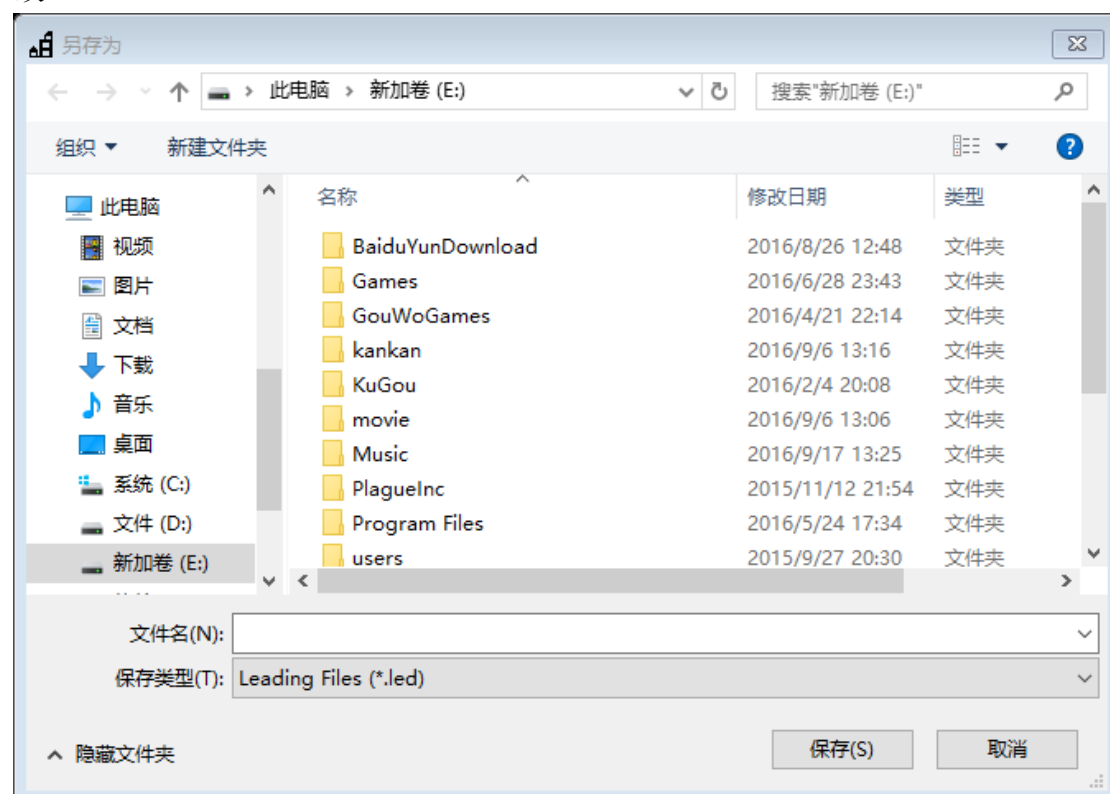


图 2.3 单击“另存为”跳出的保存窗口

### 2.1.3 新建

新建一个空的文件，供用户进行操作（添加、修改、删除等）。实际操作中

可跳过这一步，直接在空的系统中操作后保存。




 武汉市楼盘信息 - 房间.bin	2016/9/21 20:45	BIN 文件	230 KB
 武汉市楼盘信息 - 楼栋.bin	2016/9/21 20:45	BIN 文件	20 KB
 武汉市楼盘信息 - 楼盘.bin	2016/9/21 20:45	BIN 文件	7 KB
 武汉市楼盘信息.led	2016/9/21 20:45	LED 文件	1 KB

图 2.4 保存后的文件在磁盘中的显示

### 2.1.4 导入

选择磁盘中的某一个文件（要求为 Leading Files (\*.led) 文件）导入该管理系统，并将文件中的所有数据读入链表，同时在系统的窗口中显示。具体实现方法如下：

1) 点击导入之后，弹出一个类似于“保存”的窗口。可通过选择文件格式来筛选文件（默认格式为\*.led），单击文件表示选择。

2) 若点击确定，则将该文件下的数据文件导入链表，并在系统中显示第一个楼盘的信息，如图 2.5 所示。若点击取消，则恢复原状。



图 2.5 导入后系统显示的信息

### 2.1.5 退出

退出实现功能与点击右上角的退出按钮相同。首先确认文件是否保存和做过修改。若已修改但未保存，将会弹出窗口提示是否要保存，如图 2.6 所示，否则直接退出系统。



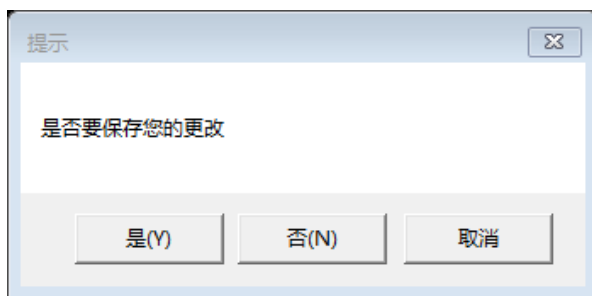


图 2.6 提示保存对话框

## 2.2 显示模块

显示模块不存在于菜单中。为了方便用户浏览信息，实现编辑、统计、添加的同步显示，做到“所见即所得”，本系统设计了显示模块。显示模块允许用户在除菜单外的客户区进行单击、双击操作，同时为其它许多功能的操作提供了便利，带来了良好的用户体验。下面具体描述显示的具体过程。

1) 当进入系统导入文件之后，显示的是第一级，楼盘与该楼盘下所属的楼栋信息，如图 2.7 所示；



图 2.7 显示第一级楼盘与楼栋

2) 单击每一个楼盘，在右上方蓝色信息区会显示该楼盘的具体信息，同时在右下方的列表中显示该楼盘下的所有楼盘。同时信息区右下角的两个按钮“编辑”与“删除”提供的快速删除、编辑当前楼盘的有效路径；

3) 在 2) 的基础上单击右下角列表中每一个楼栋，在右上方蓝色信息区会显示对应的楼栋信息。同时信息区右下角的两个按钮“编辑”与“删除”提供的快

速删除、编辑当前楼栋的有效路径；

4) 在 2) 的基础上双击右下列表中的楼栋，将可以进入该楼栋，即在右上角显示该楼栋的信息，在右下角显示该楼栋下的所有房间及基本信息。同时信息区右下角的两个按钮“编辑”与“删除”提供的快速删除、编辑当前楼栋的有效路径，如图 2.8 所示；

5) 在 4) 的基础上单击右下角列表中的房间，将在右上角蓝色信息区中显示该房间的所有信息，如图 2.9 所示。同时信息区右下角的两个按钮“编辑”与“删除”提供的快速删除、编辑当前房间的有效路径；

6) 在 4) 的基础上，单击右下方列表中的第一行楼栋基本信息，将显示该楼栋所有信息，并提供“编辑”与“删除”楼栋功能。双击右下方列表中的第一行楼栋基本信息，将返回上一级，即右下列表中显示该楼盘下所有楼栋信息，右上信息区中显示该楼栋的所有信息。同时信息区右下角的两个按钮“编辑”与“删除”提供的快速删除、编辑当前房间的有效路径；

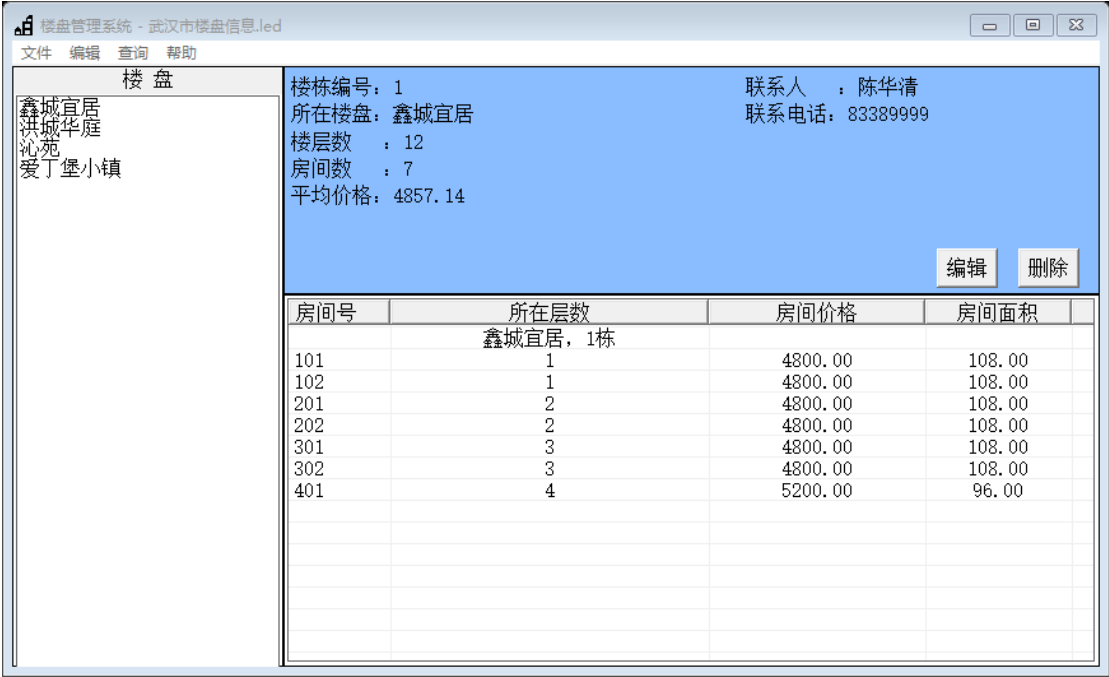


图 2.8 双击楼栋显示第二级信息

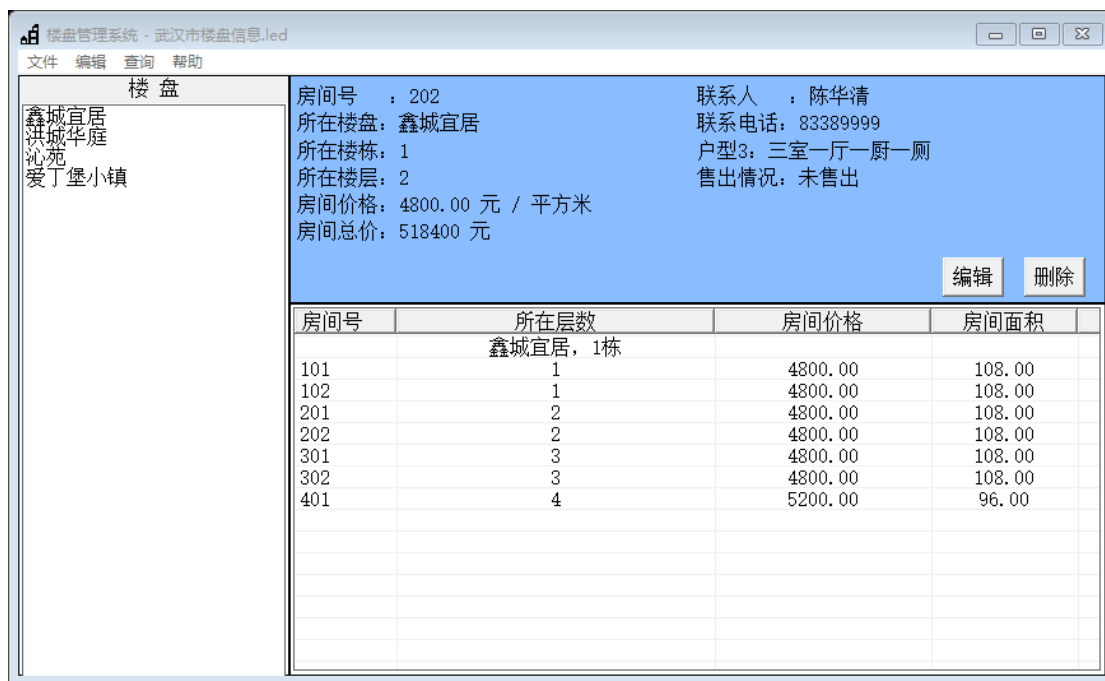


图 2.9 单击房间显示房间信息

## 2.3 编辑模块

编辑模块主要功能为对导入的数据进行编辑。包括更改、增添、删除三大部分，如图 2.10 所示。每部分的详细内容如下：

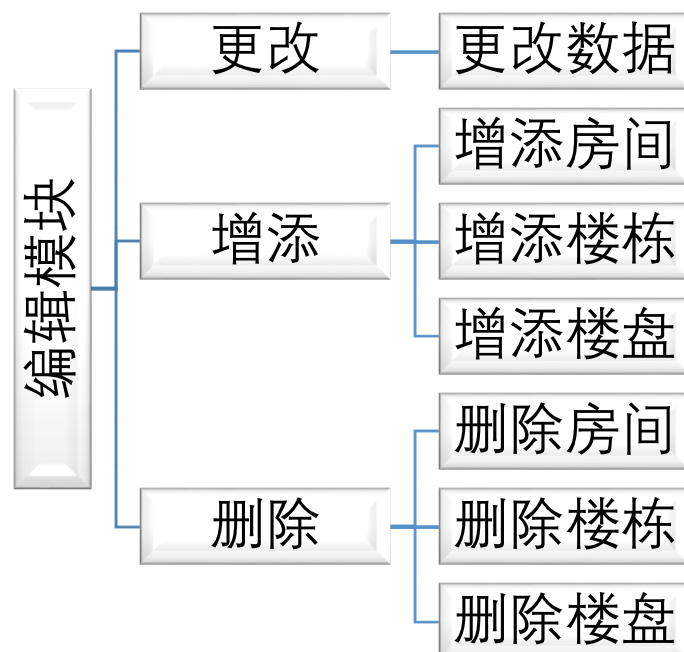


图 2.10 编辑模块功能

### 2.3.1 更改功能

更改功能又称编辑，可以通过两种方式触发：单击菜单中的“更改”项或在

浏览房间/楼栋/楼盘信息之时点击当时所在的房间/楼栋/楼盘信息右下角的编辑按钮。具体实现功能如下：

- 1) 通过以上两种方法触发弹出更改功能对话框，如图 2.11 所示；
- 2) 对话框会默认将目前所在房间/楼栋/楼盘的信息输入对应的编辑框，允许用户在编辑框上更改相关的数据。（有一些编辑框如电话号码不允许输入数字以外的符号）
- 3) 若用户点击确定，系统将检查所输入的楼栋号（或房间号、楼盘名）是否与已有数据重复，若重复，则弹出警告，如图 2.12 所示，要求用户更改所输入的数据。若成功编辑房间，则同时更新该楼盘、楼栋的平均价格，若成功编辑楼盘，则同时更新当前楼盘下的所有楼栋、房间的“所在楼盘”数据。若用户点击取消，则不做更改。
- 4) 更改完成后，重绘整个客户区，将数据更新。用户可以通过系统显示判断更改是否成功。

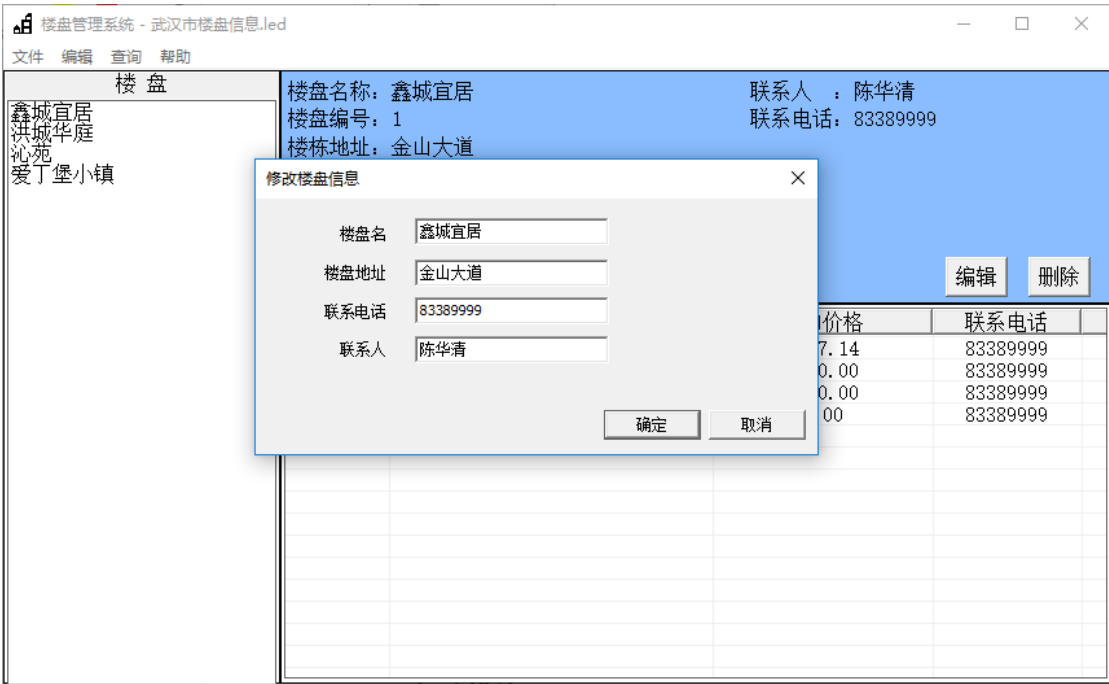
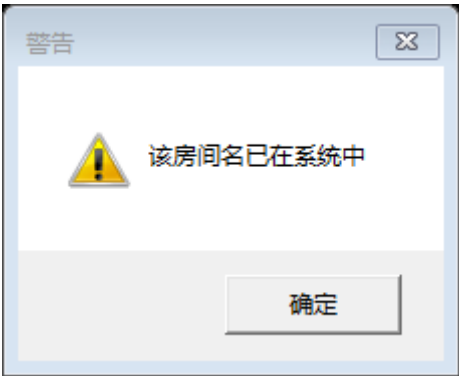


图 2.11 激活的更改对话框



2.12 弹出的修改无效警告

### 2.3.2 添加功能

添加功能有分为三大部分：添加楼盘，添加楼栋和添加房间。

#### a. 添加房间

添加房间允许在系统任何的显示状态下进行。当单击菜单中“添加 – 添加房间”时，会弹出对话框。若显示当前信息区显示为楼盘，则会在对话框中自动输入默认楼盘的相关信息；若当前信息区显示为楼栋，则会在对话框中自动输入默认楼栋的相关信息，如图 2.13 所示。

在添加房间的对话框中使用了 Edit, Static, Combo Box (下拉选择框), Check Box 等许多控件。其中下拉选择框中储存了所有的房间类型信息共 34 条，方便用户选择。

在输入完毕后，用户若点击确定，则系统会检测是否有重复的房间号，若有则报错，要求用户重新输入信息，则添加成功，用户可在列表框中查看到新添加的房间，或点击查看具体信息；若点击取消，则放弃添加，系统恢复原状。

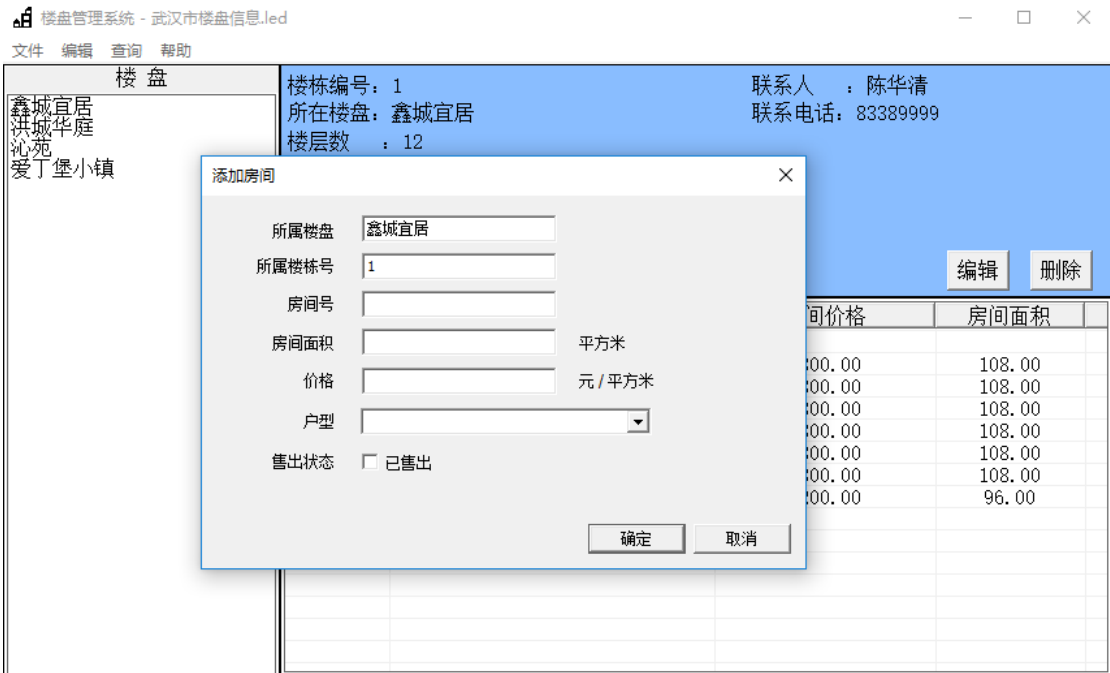


图 2.13 添加房间默认情况

#### b. 添加楼栋

添加楼栋允许在系统任何的显示状态下进行。当单击菜单中“添加 – 添加楼栋”时，会弹出对话框，且会在对话框中自动输入当前信息区显示楼盘的相关信息，如图 2.14 所示。

在输入完毕后，用户若点击确定，则系统会检测是否有重复的房间号，若有则报错，要求用户重新输入信息，若无，则添加成功，用户可在列表框中查看到新添加的楼栋，或点击查看具体信息；若点击取消，则放弃添加，系统恢复原状。

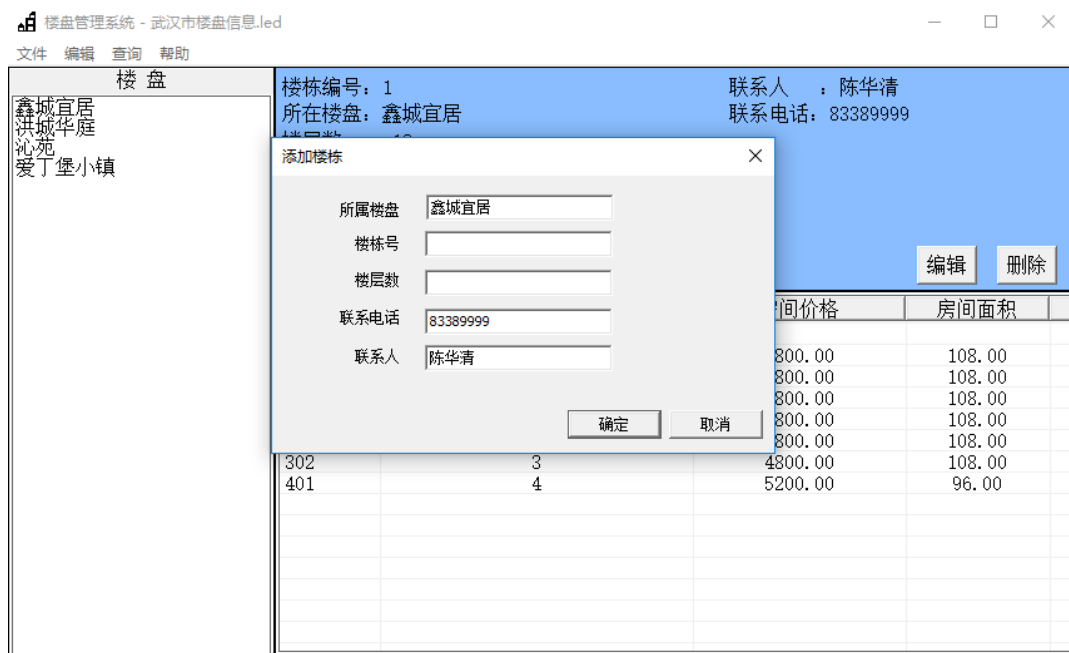


图 2.14 添加楼栋默认情况

### c. 添加楼盘

添加楼盘允许在系统任何的显示状态下进行。当单击菜单中“添加 – 添加楼栋”时，会弹出对话框，用户输入需要添加的楼盘即可。

在输入完毕后，用户若点击确定，则系统会检测是否有重复的楼盘名，若有则报错，要求用户重新输入信息，若无，则添加成功，系统自动按照顺序添加楼盘号，用户可在左边列表中查看到新添加的楼盘名称，或点击查看具体信息；若点击取消，则放弃添加，系统恢复原状。

## 2.3.3 删除功能

删除功能提供两种操作方式：准确查找删除与快速删除。

### a) 准确查找删除

该种删除方式在菜单栏中的“删除”中，分为删除楼盘、楼栋和房间三种。

点击“删除楼盘”时，会弹出对话框，如图 2.15。允许选择性地输入楼盘名称或楼盘编号，选择其中一项是，另一项的输入框会变灰色，不允许输入。若点击确定，则直接删除该楼盘；若点击取消，则放弃删除。

点击“删除楼栋”时，要求选择性地输入所在楼盘名称或楼盘编号，以及准确的楼栋编号。若点击确定，则直接删除该楼栋；若点击取消，则放弃删除。

点击“删除房间”时，要求选择性地输入所在楼盘名称或楼盘编号、所在楼栋的准确的楼栋编号，以及准确的房间号。若点击确定，则直接删除该房间；若点击取消，则放弃删除。

### b) 快速删除

考虑到用户很有可能忘记或者记不清准确的编号信息，系统提供快速删除的

功能。即用户在浏览楼盘、楼栋或房间的具体信息可以直接对相关结构进行删除。

具体操作为单击信息栏的删除按钮，系统会自动判断需要删除的是楼栋、楼盘或房间，然后弹出提示框确认是否删除（防止误点），如图 2.16 所示，若用户点击确认，则删除该结构，若用户点击取消，则放弃删除。

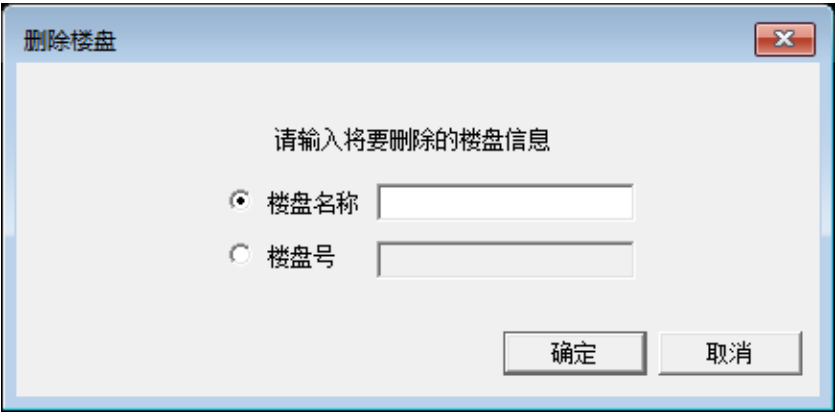


图 2.15 删除楼盘的窗口

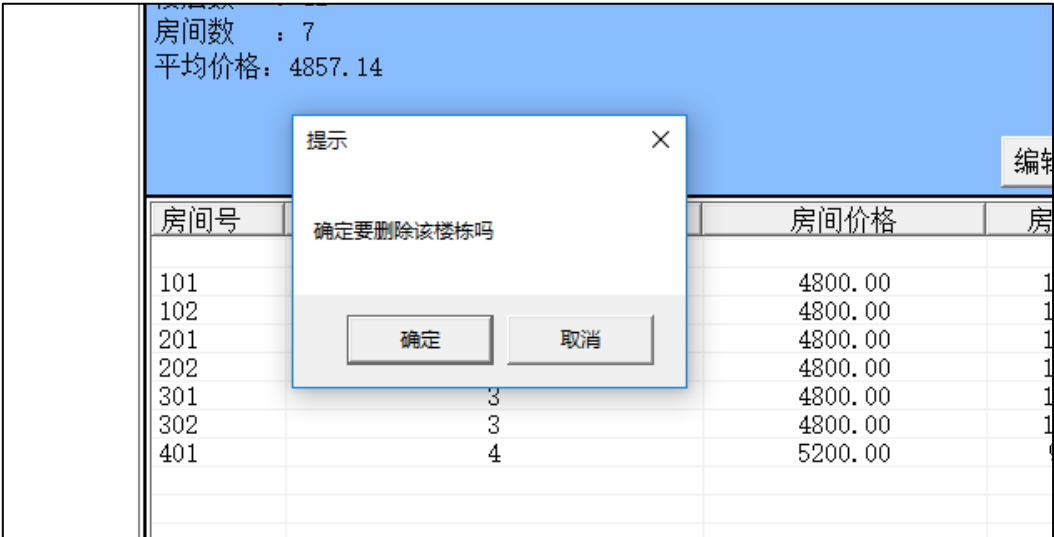


图 2.16 快速删除楼栋提示框

## 2.4 查询模块

查询模块主要功能体现在菜单中单击“查询”弹出的对话框中，如图 2.17 所示。查询功能主要分为查询楼盘，查询楼栋，查询房间三个部分，通过对话框中的选择按钮选择将要查询的部分，后面对应的信息栏将会按照选择相应的变灰，即不允许输入。

搜索

搜索楼盘

楼盘名称

楼盘编号

联系人

联系电话

搜索楼栋

楼栋编号

联系电话

联系人

搜索房间

房间编号

房间面积

所在层数

总价上限

(万元)

房间价格范围

--

元 / 平方米

户型

☐ 已售出

确定

取消

图 2.17 查询搜索对话框

选择好将要输入的信息后，用户可以在编辑框、选择框中输入自己所想要搜索的信息，但不要求准确。具体提供的搜索功能见图 2.17。

系统将会录入用户所输入的搜索信息，并根据用户的选择，将用户所输入的信息与系统中的每一个结构进行比对。系统内部会将每一条信息的重要性分配权值，并计算每一个结构与所输入每一条信息的相似度，然后所有信息相似度的加权平均值作为这一个结构与输入信息的相似度。系统会汇总所有的相似度不为 0（即至少有一条信息与所输入信息相符）的结构，并按照权值排序作为搜索结果。最终，系统将会将搜索结果按照相似度从高到低在左边列表中输出。若在搜索中输入房间号为 1，楼栋号为 1，点击确定，则搜索结果如图 2.18 所示。同时，会在列表的最后加上一条“退出搜索”的按钮。



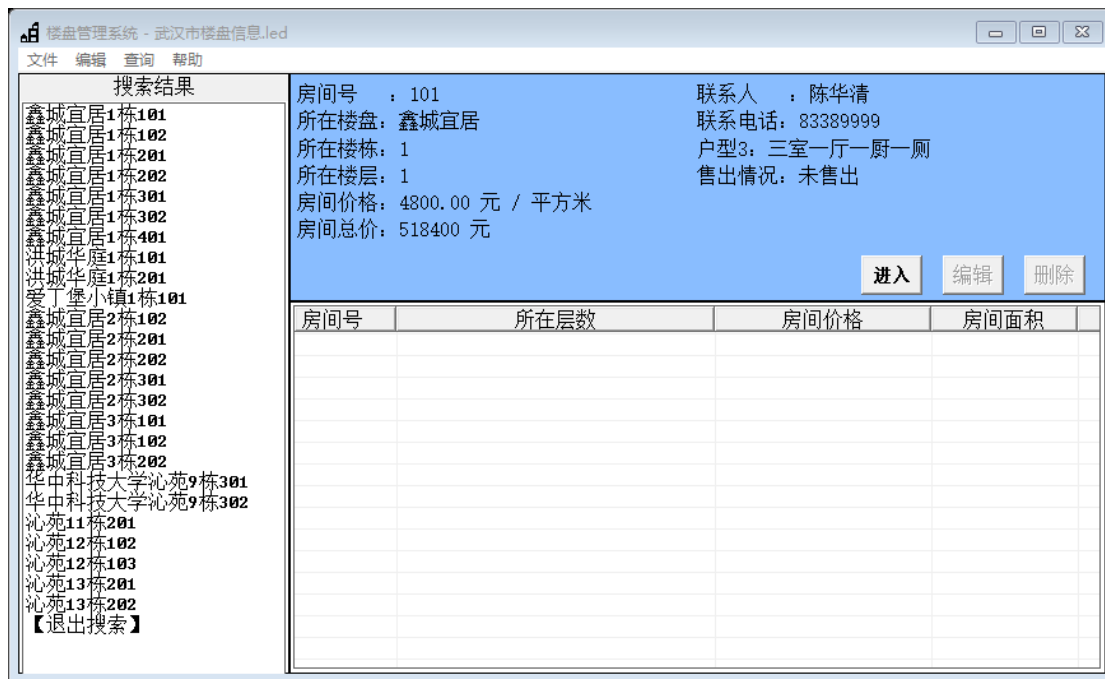


图 2.18 搜索结果示意图

显示搜索结果后，用户为了确认是否为自己想要搜索的项目，可以单击左边搜索栏中的每一个搜索结果，其全部信息将会在右边的蓝色信息区中显示。进入搜索状态后，右上信息区会出现一个“进入”按钮，当用户浏览信息后确认是想要的搜索结果时，可以点击“进入”按钮，直接进入当前搜索结果。以房间为例，点击进入后，将会在左边列表中显示楼盘信息，右上信息栏中显示该房间信息，右下列表中显示所在楼盘的所有房间，若点击沁苑 11 栋 201，则显示如图 2.19 所示。若用户未找到对应的搜索结果，则直接点击“【退出搜索】”，则恢复搜索前的状态不做改变。

按照输入的搜索结果，不存在房间号为 1 的房间，因此未找到，但是存在楼栋号为 1 的房间，因此楼栋为 1 的房间在最前面显示。同时，由于未勾选“已售出”选项，搜索所有未售出的楼盘，系统会将一些未售出但是楼栋号不是 1 的房间显示在偏后的位置，供用户查看。



图 2.19 搜索结果进入后的显示

## 2.5 统计模块

由于可视化 Windows 程序的特殊性,本程序未在菜单中加入专门的统计栏,而是将统计融入在程序的显示中。包括统计平均价格,统计楼栋所有房间数,楼盘所有房间数、楼栋数,统计已售出、未售出的房间,目前所有的楼盘数,统计平均面积等。

每次进行数据添加、更改、删除时,系统会自动将所有的相关数据全部更改,并在系统的界面上实时显示。由于楼栋编号为顺序,通过最后的楼盘编号即可了解所有盘数。且通过搜索功能可以达到更加实用和强大的统计功能,例如统计所有未售出的房间,统计所有价格在某一个价格段的房间等。

## 2.6 帮助模块

由于本系统未设置专门的参考文档,因此只在关于软件中提供版本信息、作者信息等信息。单击菜单中帮助的“关于软件”,即可得到相关信息,如图 2.20 所示。



图 2.20 关于软件

以上即为“楼盘管理系统”的主体功能模块，该功能模块较大的依赖于可视化界面，同时也充分发挥了可视化操作的优势，相较控制台带给用户更好的体验。同时，搜索模块可以通过调整权值分配的计算方法，使之更加的合理，得到的结果更符合用户的需要，此模块还有着很大的发展空间。

## 3 数据结构设计

### 3.1 整体数据结构

本系统的设计中，内存总体数据结构为十字交叉三级链表，如图 3.1 所示。

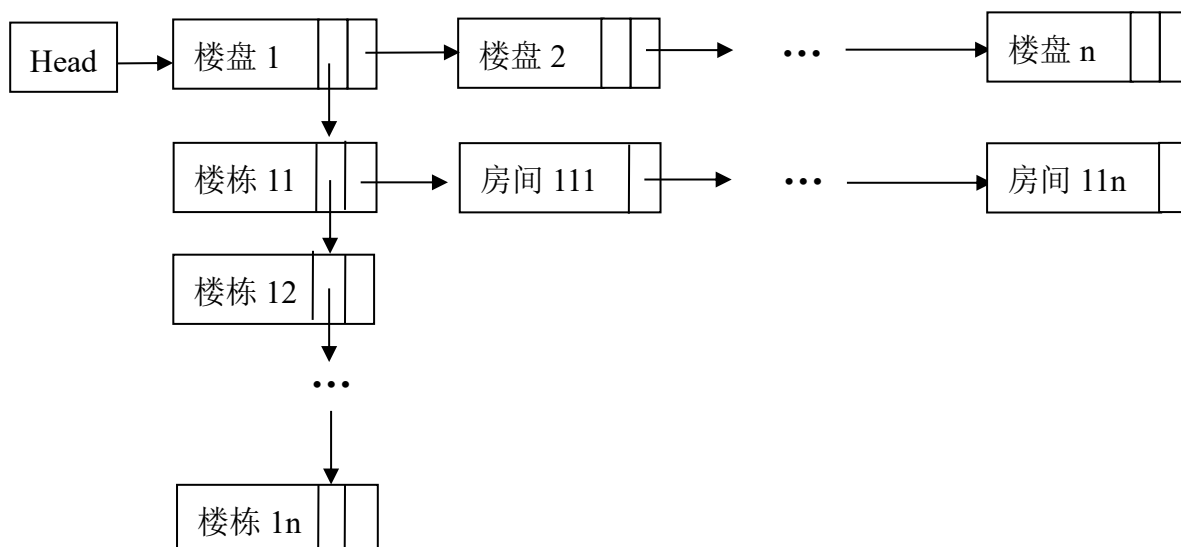


表 3.1 系统总体数据结构

每一个楼盘、楼栋、房间都作为一个结构体，这些结构体在内存中形成一个三级链表。每个成员成员储存有关的信息，每个结构体的具体设计以及各成员的意义如下：

#### 3.1.1 楼盘结构

```
////////////////////////////////////
//楼盘结构体
//功能：储存楼盘相关信息，并可用于构造链表
typedef struct _Community {
    int communityNum;           //楼盘号
    TCHAR name[100];           //楼盘名
    TCHAR address[100];        //楼盘地址
    TCHAR phone[30];           //联系电话
    int numberOfBuildings;     //所有楼栋数
    int numberOfRooms;         //所有房间数
    float avgPrice;            //平均价格
    Person host;               //联系人结构*
    struct _Community *nextCommunity = NULL; //指向下一个楼盘
    struct _Building *buildings = NULL;      //指向该楼盘第一个楼栋
}Community;
```

楼盘结构的每一个成员的具体含义如上述。其中，联系人结构将在后面指出，用于储存联系人的姓名、电话等信息。每个楼盘的唯一标识为楼盘名（name），

---

楼盘名不允许重复。

### 3.1.2 楼栋结构

```
////////////////////////////////////
//楼栋结构体
//功能：储存楼栋相关信息，并可用于构造链表
typedef struct _Building {
    int buildingNum;           //楼栋号
    int communityNum;          //所在楼盘号
    int numberOfRooms;         //所有房间数
    int numberOfFloors;        //楼层数
    float avgPrice;            //平均价格
    TCHAR inCom[100];          //所在楼盘名称
    Person host;               //联系人结构
    struct _Building *nextBuilding; //指向下一个楼栋
    struct _Room *rooms;       //指向该楼栋第一个房间
}Building;
```

楼栋结构的每一个成员的具体含义如上述。其中，联系人结构将在后面指出，用于储存联系人的姓名、电话等信息。一般而言，楼栋的联系人与楼盘的联系人相同，但是系统也提供单独修改楼栋联系人信息的方法。

### 3.1.3 房间结构

```
////////////////////////////////////
//房间结构体
//功能：储存房间相关信息，并可用于构造链表
typedef struct _Room {
    int roomNum;               //房间号
    int buildingNum;           //所在楼栋号
    int communityNum;          //所在楼盘号
    int floor;                 //所在楼层
    float roomSize;            //房间面积（平方米）
    float roomPrice;           //房间价格（人民币元/平方米）
    long allPrice;             //房间总价（人民币元）
    TCHAR roomType[100];       //户型：室，厅，厕，厨数量
    TCHAR inCom[100];          //所在楼盘名
    BOOL roomState;            //售出状态：0代表未售出，1代表已售出
    TCHAR note[1000];          //备注
    Person host;               //联系人结构
    struct _Room *nextRoom;    //指向链表中下一个房间
}Room;
```

房间结构的每一个成员的具体含义如上述。其中，联系人结构将在后面指出，用于储存联系人的姓名、电话等信息。一般而言，楼栋的联系人与楼盘、楼栋的联系人相同，但是系统也提供单独修改房间联系人信息的方法。

需要说明的是，户型的储存方式为 34 个字符串代表 34 种户型，每种户型与每个字符串严格一一对应。这些户型被定义在静态区，户型的信息如下：

```
static TCHAR *szComboBoxData[] = {
    TEXT("户型1：一室一厅一厨一厕"), TEXT("户型18：五室一厅一厨两厕"),
    TEXT("户型2：两室一厅一厨一厕"), TEXT("户型19：两室两厅两厨一厕"),
    TEXT("户型3：三室一厅一厨一厕"), TEXT("户型20：三室两厅两厨一厕"),
    TEXT("户型4：四室一厅一厨一厕"), TEXT("户型21：四室两厅两厨一厕"),
    TEXT("户型5：五室一厅一厨一厕"), TEXT("户型22：五室两厅两厨一厕"),
    TEXT("户型6：一室两厅一厨一厕"), TEXT("户型23：两室两厅一厨两厕"),
    TEXT("户型7：两室两厅一厨一厕"), TEXT("户型24：三室两厅一厨两厕"),
    TEXT("户型8：三室两厅一厨一厕"), TEXT("户型25：四室两厅一厨两厕"),
    TEXT("户型9：四室两厅一厨一厕"), TEXT("户型26：五室两厅一厨两厕"),
    TEXT("户型10：五室两厅一厨一厕"), TEXT("户型27：两室一厅两厨两厕"),
    TEXT("户型11：两室一厅两厨一厕"), TEXT("户型28：三室一厅两厨两厕"),
    TEXT("户型12：三室一厅两厨一厕"), TEXT("户型29：四室一厅两厨两厕"),
    TEXT("户型13：四室一厅两厨一厕"), TEXT("户型30：五室一厅两厨两厕"),
    TEXT("户型14：五室一厅两厨一厕"), TEXT("户型31：两室两厅两厨两厕"),
    TEXT("户型15：两室一厅一厨两厕"), TEXT("户型32：三室两厅两厨两厕"),
    TEXT("户型16：三室一厅一厨两厕"), TEXT("户型33：四室两厅两厨两厕"),
    TEXT("户型17：四室一厅一厨两厕"), TEXT("户型34：五室两厅两厨两厕"),
};
```

### 3.1.4 联系人结构

```
////////////////////////////////////////
//联系人结构体
//功能：用于存储个人信息
typedef struct _Person {
    TCHAR name[100];        //姓名
    TCHAR phone[50];        //电话
    TCHAR idNumber[50];     //证件ID号
}Person;
```

联系人结构储存一个人的相关信息，具体成员的含义见注释。

## 3.2 辅助链表

在搜索功能中，为了搜索的便捷。每找到一个有相关性的结构体，就将其接入一个新的链表，称为**搜索链表**。该链表为单向先进先出的链表，搜索链表有三类，分为楼盘搜索、楼栋搜索、房间搜索，对应三种不同的链表。将相关数据接入链表中后，将链表排序后输出，即可得到搜索结果。

---

## 3.3 其他结构体

### 3.3.1 引导文件结构体

创建引导文件的目的是方便用户导入数据，同时储存相关的信，以供读入内存时使用。因此在引导文件中存储的是一个结构体，储存了其他数据文件的基本信息。引导文件结构如下：

```
////////////////////////////////////  
//*.led文件中的结构体  
//储存com, bui, room三个文件里面所包含的数据数  
typedef struct _LenData {  
    int iCom;  
    int iBui;  
    int iRoom;  
}LenData;
```

该文件中存储的为每一种结构体的数量。便于在读取文件重新构建链表的时候顺利的构建与原来相同的链表，结构见表 3.2。

### 3.3.2 搜索结构体

在搜索设计中，系统将读取用户输入的所有数据并将其与每一个结构进行比对。因此，为了方便比对，特意将所有的输入数据读入同一个结构体，该结构具体内容如下：

```
////////////////////////////////////  
//SearchData搜索中使用的结构体，容纳搜索的所有数据  
typedef struct _SearchData {  
  
    int mask;                //确定搜索的对象  
                             //值为SD_COM, SD_BUI, SD_ROOM中的一个  
  
    //Community  
    TCHAR comName[100];  
    int comNum;  
    TCHAR comPerson[100];  
    TCHAR comPhone[100];  
  
    //Building  
    int buiNum;  
    TCHAR buiPhone[100];  
    TCHAR buiPerson[100];  
  
    //Room  
    int roomNum;  
    int roomFloor;
```

---

```
float roomSize;  
long allPrice;  
float roomLoPrice;  
float roomHiPrice;  
BOOL roomSold;  
TCHAR roomType[100];  
}SearchData;
```

该结构体分为三个部分，楼盘、楼栋和房间，未输入的信息即为 0 或\0。第一个成员 mask 储存搜索的目标，其三个值在头文件“managerSys.h”头文件中定义。使用时，将每一个成员与对应结构比对即可。

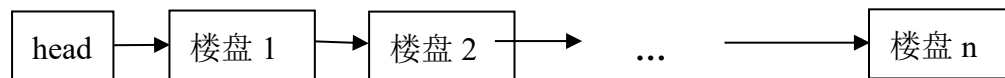


表 3.2 搜索链表结构



---

## 4 系统模块设计与实现

### 4.1 文件模块

#### 4.1.1 文件格式

##### 1、引导文件

引导文件只包含一个引导结构体，包含了所需要的相关信息。

##### 2、楼盘数据文件

楼盘数据文件以楼盘结构为单位储存了所有楼盘的信息。该结构为许多楼盘无间隔相邻顺序储存，每一个单位大小为 `sizeof(Community)`。

##### 3、楼栋数据文件

每一个楼盘中，属于该楼盘的楼栋无间隔相邻顺序储存。不同楼盘的楼栋按照楼盘存储顺序储存，并且每两个楼盘的楼栋之间相隔一个分隔楼栋，该分隔楼栋的楼栋号为-1，其与数据均为空。

##### 4、房间数据文件

每一个楼栋中，属于该楼栋的房间无间隔相邻顺序储存。不同楼栋的房间按照楼盘存储顺序储存，并且每两个楼栋的房间之间相隔一个分隔房间，该分隔房间的房间号为-1，其与数据均为空。

#### 4.1.2 文件读取

文件读取使用的函数为 `ReadFile` 函数<sup>[1]</sup>，这些函数比 `fread` 更加功能全面。通过每次在文件中读取一个结构的数据到缓冲区，确定数据全部读取然后将缓冲区写入链表，之后调用 `SetFilePointer` 函数<sup>[2]</sup> 移动文件指针到下一个结构，在调用下一个结构写入链表。若遇到文件中的分隔结构，则跳过该结构，将接下来读取的数据写入下一个楼盘（楼栋）。

整个读取由一个函数来执行，该函数原型如下：

```
////////////////////////////////////  
//FileReadWnd  
//功能：读取文件  
//参数：窗口句柄hwnd，路径名  
//返回：若成功则返回TRUE，失败则返回FALSE  
BOOL FileReadWnd(HWND hwnd, PTSTR pstrLeadFileName);
```

该函数的主要结构为：

- 1) 获取文件名；
- 2) 编辑数据文件名
- 3) 调用 `CreateFile` 函数<sup>[3]</sup> 打开引导文件和数据文件，获取文件句柄；
- 4) 检查读取是否完整，并拷贝引导文件中的数据；

- 
- 5) 将三个文件中的数据写入链表;
  - 6) 检查数据是否全部读取;
  - 7) 设置主窗口标题和回收指针、句柄;
- 函数参数以及返回值见注释, 函数实现代码见附录。

#### 4.1.3 文件写入

文件写入的函数为 WriteFile 函数<sup>[4]</sup>, 通过按照 4.1.1 所述的文件结构将链表中的数据写入链表, 同时在适当的地方加入分隔结构, 便于读取。每次写入文件后, 调用 SetFilePointer 函数将文件指针后移一个结构体, 写入新的数据(注意不可简单的将指针移到文件尾)。

整个写入过程由一个函数来执行, 该函数原型如下:

```
////////////////////////////////////  
//FileWriteWnd  
//功能: 写入文件  
//参数: 窗口句柄hwnd, 引导文件路径  
//返回: 若成功则返回TRUE, 失败则返回FALSE  
BOOL FileWriteWnd(HWND hwnd, PTSTR pstrLeadFileName);
```

该函数的主要结构为:

- 1) 获取文件名;
  - 2) 编辑数据文件名
  - 3) 调用 CreateFile 函数<sup>[3]</sup> 打开引导文件和数据文件, 获取文件句柄;
  - 4) 编辑好文件中的分隔结构, 并将其与链表中的数据一同按照 4.1.1 中的文件结构写入相应文件中;
  - 5) 检查数据是否完全读取并制作引导结构, 将其写入引导文件;
  - 6) 设置主窗口标题和回收指针、句柄;
- 函数参数以及返回值见注释, 函数实现代码见附录。

#### 4.1.4 其它文件处理函数<sup>[5]</sup>

其它的文件处理函数用于初始化以及完成一些附属功能, 例如 FileInitWnd 函数、FileImportDlg 函数、FileSaveDlg 函数等, 它们的函数原型如下:

```
////////////////////////////////////  
//FileInitWnd  
//功能: 初始化弹出文件窗口的数据  
//参数: 父窗口句柄hwnd  
//返回: 无返回值  
void FileInitWnd(HWND hwnd);
```

---

```

////////////////////////////////////
//FileImportDlg
//功能：创建导入文件对话框
//参数：父窗口句柄hwnd，文件名，路径名
//返回：若成功则返回TRUE，失败则返回FALSE
BOOL FileImportDlg(HWND hwnd, PTSTR pstrFileName, PTSTR pstrTitleName);

////////////////////////////////////
//FileSaveDlg
//功能：创建保存文件对话框
//参数：窗口句柄hwnd，文件名，路径名
//返回：若成功则返回TRUE，失败则返回FALSE
BOOL FileSaveDlg(HWND hwnd, PTSTR pstrFileName, PTSTR pstrTitleName);

```

这些函数的参数以及返回值见注释，函数实现代码见附录。

#### 4.1.5 主窗口过程中有关文件的消息响应

主窗口过程中有关文件的消息主要有 WM\_COMMAND 消息中的消息码为 IDM\_FILE\_SAVE ， IDM\_FILE\_SAVE\_AS ， IDM\_FILE\_CREATE ， IDM\_FILE\_IMPORT, IDM\_FILE\_EXIT 五种。不同的消息码对应菜单中的不同项目的 ID。在 WndProc 主窗口过程中，对各个消息码有着分别的处理：

##### 1) 保存 (IDM\_FILE\_SAVE)

调用 FileWriteWnd 函数将现有链表写入文件，若写入失败，则弹出警告。之后修改保存情况。具体操作见 2.1.1 文件模块功能。

##### 2) 另存为 (IDM\_FILE\_SAVE\_AS)

调用 FileWriteWnd 函数，弹出对话框，显示地址。将现有链表写入到一个备份的文件中，若写入失败，则弹出警告。之后修改保存情况具体操作见 2.1.2 文件模块功能。

##### 3) 新建 (IDM\_FILE\_CREATE)

更改主窗口显示的文件名。具体操作见 2.1.3 文件模块功能。

##### 4) 导入 (IDM\_FILE\_IMPORT)

若未保存，提示用户是否需要保存。调用 FileImportDlg 和 FileReadWnd 函数，弹出导入对话框，选择导入的文件（具体操作见 2.1.4 文件模块功能）。之后将所有数据导入链表，并调用 InvalidateRect 函数<sup>[6]</sup>无效化整个窗口，重绘相关信息，将信息区显示为第一个楼盘的数据。

##### 5) 退出 (IDM\_FILE\_EXIT)

向主窗口发送 WM\_CLOSE 消息。若未保存，则提示用户保存。若已经保存，向主窗口发送 WM\_DESTROY 消息请求关闭。<sup>[7]</sup>

---

## 4.2 显示模块

在头文件“managerSys”显示过程中定义四种显示状态：

```
#define PA_FIRSTEPCOM    500    //第一级，显示楼盘信息
#define PA_SECSTEPBUI    501    //第二级，显示楼栋信息
#define PA_FIRSTEPBUI    502    //第一级，显示楼栋信息
#define PA_SECSTEPROOM   503    //第二级，显示房间信息
```

### 4.2.1 显示第一级

显示第一级为在左边 ListBox<sup>[11]</sup>中显示楼盘，在右上信息区中显示楼盘或楼栋的具体信息，在右下角 ListView<sup>[12]</sup>中显示相应楼盘下的所有楼栋基本信息。

此时根据全局变量 iPaintState 的取值，判断显示的级别。若为显示楼盘信息，则在 WM\_PAINT 消息处理中调用函数 Draw1Step 与 DrawComInf；若显示为楼栋信息，则在 WM\_PAINT 消息处理中调用函数 Draw1Step 与 DrawBuiInf。

单击左边的 ListBox 中的项目，则通过 LB\_SELCHANGE 中获取选择项的文本（即为楼盘名），将楼盘名作为信息在整个链表内检索，获取该楼盘的具体信息以及其下的所有楼栋信息。使用 TextOut 函数在右上信息区中输出相关信息，通过向右下角发送 LVS\_SETITEMTEXT（或调用函数 ListView\_SetItemText）将楼栋信息发送到 ListView 中，从而显示楼盘信息（即为 Draw1Step 的工作方式）。

单击右边的楼栋项目，将会将状态设置为 PA\_FIRSTEPBUI，然后无效化窗口并发送 WM\_PAINT 消息。此时调用函数 Draw1Step 与 DrawBuiInf，此时显示对应楼栋信息。。

双击右边的楼栋项目，将会将状态设置为 PA\_SECSTEPBUI，然后无效化窗口并发送 WM\_PAINT 消息。此时调用函数 Draw2Step 与 DrawBuiInf，在右边显示楼栋信息以及在列表中列出该楼栋下所有楼盘。

### 4.2.2 显示第二级

显示第一级为在左边 ListBox<sup>[11]</sup>中显示楼盘，在右上信息区中显示楼栋或房间的具体信息，在右下角 ListView<sup>[12]</sup>中显示相应楼盘下的所有楼栋基本信息。

此时根据全局变量 iPaintState 的取值，判断显示的级别。若为显示楼盘信息，则在 WM\_PAINT 消息处理中调用函数 Draw1Step 与 DrawComInf；若显示为楼栋信息，则在 WM\_PAINT 消息处理中调用函数 Draw1Step 与 DrawBuiInf。

单击左边的 ListBox 中的项目，则通过 LB\_SELCHANGE 中获取选择项的文本（即为楼盘名），将楼盘名作为信息在整个链表内检索，获取该楼盘的具体信息以及其下的所有楼栋信息。使用 TextOut 函数在右上信息区中输出相关信息，通过向右下角发送 LVS\_SETITEMTEXT（或调用函数 ListView\_SetItemText）将楼栋信息发送到 ListView 中，从而显示楼盘信息（即为 Draw1Step 的工作方式）。

---

单击右边的房间或楼栋项目，将会将状态设置为 PA\_SECSTEPBUI 或 PA\_SECSTEPROOM，然后无效化窗口并发送 WM\_PAINT 消息。此时调用函数 Draw2Step 与 DrawBuiInf 或 DrawRoomInf，此时显示对应楼栋信息或对应房间信息。

双击右边的列表中第一行楼栋项目，通过读取 WM\_NOTIFY 中的消息获取选中项目，同时将会将状态设置为 PA\_SECSTEPBUI，然后无效化窗口并发送 WM\_PAINT 消息。此时调用函数 Draw1Step 与 DrawBuiInf，在右边显示楼栋信息以及在列表中列出所在楼盘下的所有楼栋。即实现返回上一级的目的。

在这两种显示中使用的函数声明如下：

```
////////////////////////////////////
//Draw1Step
//功能：WM_PAINT消息中绘制第一级(包含Com与Bui)，不显示数据
//参数：设备环境hdc，显示数据的com指针
//返回：无返回
void Draw1Step(HDC hdc, Community *pChosenCom);

////////////////////////////////////
//Draw2Step
//功能：WM_PAINT消息中绘制第二级，不显示数据
//参数：设备环境hdc, 指向所选楼栋的指针
//返回：无返回
void Draw2Step(HDC hdc, Building *pBui);

////////////////////////////////////
//DrawComInf
//功能：绘制楼盘信息窗口
//参数：编辑框句柄hwnd, 设备环境hdc, Community结构
//返回：成功返回TRUE，失败返回FALSE
BOOL DrawComInf(HWND hwnd, HDC hdc, Community com);

////////////////////////////////////
//DrawBuiInf
//功能：绘制楼栋信息窗口
//参数：编辑框句柄hwnd, 设备环境hdc, Building结构
//返回：成功返回TRUE，失败返回FALSE
BOOL DrawBuiInf(HWND hwnd, HDC hdc, Building bui);

////////////////////////////////////
//DrawRoomInf
//功能：绘制楼栋信息窗口
//参数：编辑框句柄hwnd, 设备环境hdc, Room结构
```

---

```
//返回：成功返回TRUE，失败返回FALSE  
BOOL DrawRoomInf(HWND hwnd, HDC hdc, Room room);
```

```
////////////////////////////////////  
//RenewData  
//功能：将数据区恢复为背景色  
//参数：设备环境hdc，Building结构  
//返回：无返回  
void RenewData(HDC hdc, RECT rectData);
```

函数具体功能见注释，函数实现代码见附录。

### 4.2.3 显示搜索结果

显示搜索结果时，不同于以上的两种显示方式。当搜索点击确定时，整个系统的一个全局变量 `iSearch` 将会更改为搜索过程，对该变量取值范围在“`managerSys.h`”中做出了相关规定：

```
#define SEARCHCOM 701  
#define NOSEARCH 702  
#define SEARCHBUI 703  
#define SEARCHROOM 704
```

同时，定义处于搜索状态如下：

```
#define SEARCHING (iSearch==SEARCHCOM||iSearch==SEARCHBUI||iSearch==SEARCHROOM)
```

当处于搜索状态时，`WM_PAINT` 消息中在左边 `ListBox` 显示为辅助链表(3.2) `pComSearch`（或 `pBuiSearch`、`pRoomSearch`）中的数据，通过不停发送 `LB_ADDSTRING` 消息，添加项目，并在最后添加一条【退出搜索】。将左边上放的标题通过 `SetWindowText` 设置为“搜索结果”。同时响应其中的单击消息，只调用 `Draw*Inf` 函数显示对应信息。

显示搜索结果未应用特殊的函数，基本通过调用之前的函数即可实现功能。

## 4.3 编辑模块

### 4.3.1 更改数据

更改数据分为两种情况：点击菜单中的更改、点击信息栏的“编辑”。

在点击菜单中的“更改”时，将会直接发送一条 `BM_CLICK` 消息给编辑按钮，相当于制造了一次点击。

在点击了信息栏中的“编辑”按钮时，保存状态转为 `False`。首先判断当前信息区的状态，即被楼盘、楼栋或是房间占用。根据情况将会弹出相应的对话框，同时，读取信息区的楼栋号（或楼盘名、房间号），遍历链表检索该楼栋（或是楼盘、房间），之后将搜索到楼盘的相关信息通过 `SetWindowText` 函数<sup>[8]</sup>设置为

对应信息编辑框的默认值。在用户对信息做出更改后，将检验是否与已在楼盘、楼栋、房间冲突，若不冲突，则将修改的信息录入链表，同时调用 UpdateData 函数，将链表的所有数据更新，并将其下的所有结构的 inCom 或 buildingNum 成员对应修改。无效化窗口，信息区规定显示编辑的信息更新版。

UpdateData 函数为自定义函数，其声明如下：

```
////////////////////////////////////  
//UpdateData  
//功能：更新整个链表的数据，包括房间数，均价等  
//参数：将要更新的链表的头指针的地址  
//返回：若完成所有更新则返回TRUE，否则返回FALSE  
BOOL UpdateData(Community **ppHead);
```

### 4.3.2 添加数据

添加数据分为三种情况，添加楼盘、楼栋和房间。

以添加房间为例。点击添加房间时，WndProc 收到 WM\_COMMAND 消息 (其 wParam 为 IDM\_EDIT\_ADD\_ROOM)，于是会调用 DialogBox 函数<sup>[10]</sup>，创建一个对话框。在对话框窗口过程 AddRoomDlgProc 来获取此时所在的楼栋和楼盘信息，将默认的信息通过 SetWindowText 函数发送到对话框中。当用户添加完所有数据之后，通过向编辑框发送 EM\_GETLINE 消息，对 Combo Box 进行 GetWindowText 函数获取文本，以及选择框发送 BM\_GETCHECK 消息获取选择状态来获取对话框中的所有信息。利用\_ttoi, \_ttof 等函数<sup>[9]</sup>将所有的信息转化为对应的数据类型，然后储存在一个新建的房间结构体中。判断这个房间的房间号是否与同一楼栋中的房间号冲突，若不冲突，将这个结构体接入链表，否则弹出警告报错。

接入链表后，无效化整个客户区，并调用 UpdateData 函数后重绘整个客户区，信息区的为房间的信息。楼盘、楼栋的添加过程与此基本相似，具体流程如图 4.1 所示。

参与添加数据的函数共有三个窗口过程，同时有三个辅助函数用于将数据写入链表。具体的函数声明如下：

```
////////////////////////////////////  
//“添加新楼盘” (AddCommunity) 对话框窗口过程  
BOOL CALLBACK AddComDlgProc(HWND hDlg, UINT message,  
    WPARAM wParam, LPARAM lParam);  
  
////////////////////////////////////  
//“添加新楼栋” (AddBuilding) 对话框窗口过程  
BOOL CALLBACK AddBuiDlgProc(HWND hDlg, UINT message,  
    WPARAM wParam, LPARAM lParam);
```



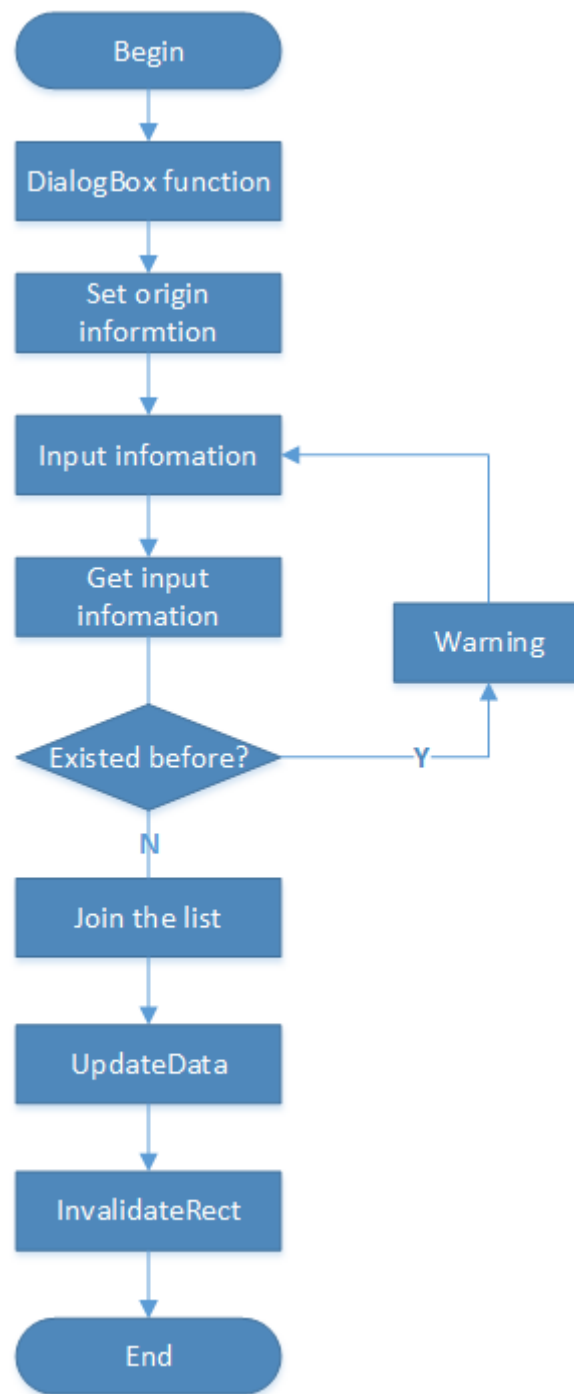


图 4.1 编辑模块的基本流程

```

////////////////////////////////////
// “添加新房间”（AddRoom）对话框窗口过程
BOOL CALLBACK AddRoomDlgProc(HWND hDlg, UINT message,
    WPARAM wParam, LPARAM lParam);

////////////////////////////////////
//AddComToList

```



---

```

//功能：向链表中添加一个community节点
//参数：将要添加的Community结构体（添加对象链表的头指针.全局变量提供）
//返回：返回更新后链表头指针
BOOL AddComToList(Community newCom, Community **head);

////////////////////////////////////
//AddBuiToList
//功能：向链表中添加一个building节点
//参数：将要添加的Building结构体，添加对象链表的头指针，所属楼盘号
//返回：成功则返回TRUE，失败返回FALSE
BOOL AddBuiToList(Building newBui, Community **head, int comNum);

////////////////////////////////////
//AddRoomToList
//功能：向链表中添加一个Room节点
//参数：将要添加的Room结构体，添加对象链表的头指针，所属楼盘号，所属楼栋号
//返回：成功则返回TRUE，失败返回FALSE
BOOL AddRoomToList(Room newRoom, Community **head,
    int comNum, int buiNum);

```

函数的参数及返回值参见注释，函数的具体实现代码见附录。

#### 4.3.1 删除数据

删除数据分为三种情况：删除楼盘、楼栋和房间。同时系统提供两种删除方式：准确删除与快速删除。下面以添加房间为例。

**准确删除。**点击添加房间时，WndProc 收到 WM\_COMMAND 消息(其 wParam 为 IDM\_EDIT\_DEL\_ROOM)，于是会调用 DialogBox 函数<sup>[10]</sup>，创建一个对话框，要求用户输入对应的准确信息。其中楼盘允许在楼盘号与楼盘名中间进行选择。即在对话框窗口过程 DelRoomDlgProc 中处理复选框消息时，若用户选择第一个选项，则通过调用 EnableWindow 函数<sup>[10]</sup>将第一个窗口有效，第二个窗口无效。当用户添加完所有数据之后，通过判断复选框的选项和向编辑框发送 EM\_GETLINE 消息来获取所有的数据。利用 \_ttoi, \_ttof 等函数<sup>[9]</sup>将所有的信息转化为对应的数据类型，然后遍历链表寻找知否有符合要求的房间。若找到，则直接从链表中删除该房间，若未找到，则弹出 MessageBox 提示该房间不存在。

删除链表后，无效化整个客户区，并调用 UpdateData 函数后重绘整个客户区，信息区的为删除的房间所在的楼栋信息。楼盘、楼栋的添加过程与此基本相似，具体流程如图 4.2 所示。

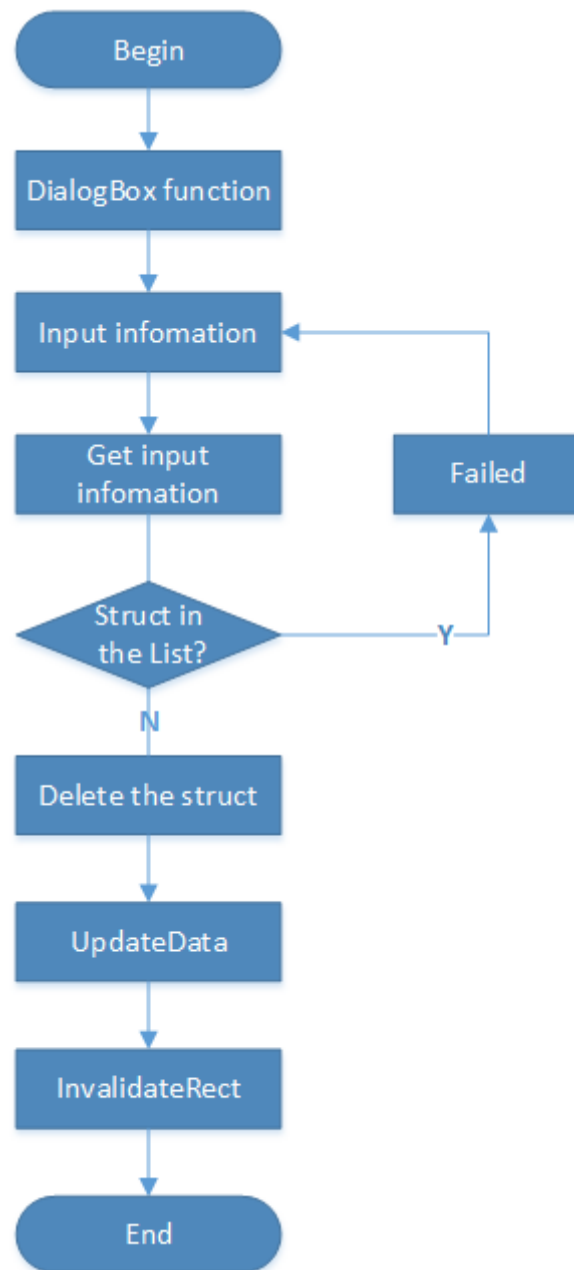


图 4.2 删除模块的基本流程

**快速删除。**快速删除主要体现在在浏览信息时，可以点击信息区右下角的“删除”按钮。此时读取目前所在的房间（或楼栋、楼盘）信息，之后遍历链表搜索该房间。搜索到该房间之后，弹出 **MessageBox** 询问用户是否确定删除该房间。若用户点击是，则调用删除函数将该房间从链表中删除。无效化整个窗口并发送 **WM\_PAINT** 消息重绘窗口。若用户点击否，则不做删除，重绘窗口。

参与删除数据的函数共有三个窗口过程，同时有三个辅助函数用于将数据从链表中删除。具体的函数声明如下：

```

////////////////////////////////////
// “删除楼盘” (DeleteCommunity) 对话框窗口过程
BOOL CALLBACK DelComDlgProc (HWND hDlg, UINT message,

```

---

```

        WPARAM wParam, LPARAM lParam);

////////////////////////////////////
// “删除楼盘” (DeleteBui) 对话框窗口过程
BOOL CALLBACK DelBuiDlgProc(HWND hDlg, UINT message,
        WPARAM wParam, LPARAM lParam);

////////////////////////////////////
// “删除楼盘” (DeleteRoom) 对话框窗口过程
BOOL CALLBACK DelRoomDlgProc(HWND hDlg, UINT message,
        WPARAM wParam, LPARAM lParam);

////////////////////////////////////
//DelComInList
//功能：删除链表中的一个Community节点
//参数：将要删除的Community编号，添加对象链表的头指针的地址
//返回：返回更新后链表头指针
BOOL DelComInList(int comNum, Community **head);

////////////////////////////////////
//DelBuiInList
//功能：删除链表中的一个Building节点
//参数：将要删除的Building编号，添加对象链表的头指针的地址
//返回：返回更新后链表头指针
BOOL DelBuiInList(int comNum, int buiNum, Community **head);

////////////////////////////////////
//DelRoomInList
//功能：删除链表中的一个Room节点
//参数：将要删除的Room编号，添加对象链表的头指针的地址
//返回：返回更新后链表头指针
BOOL DelRoomInList(int comNum, int buiNum, int roomNum, Community **head);

```

函数的参数及返回值参见注释，函数的具体实现代码见附录。

## 4.4 查询模块

查询模块为弹出对话框，之后读取用户输入的信息，通过相关搜索函数将每一项的相似度与对应结构进行比较。（详见 2.2 显示模块）具体的实现过程如图 4.3 所示。

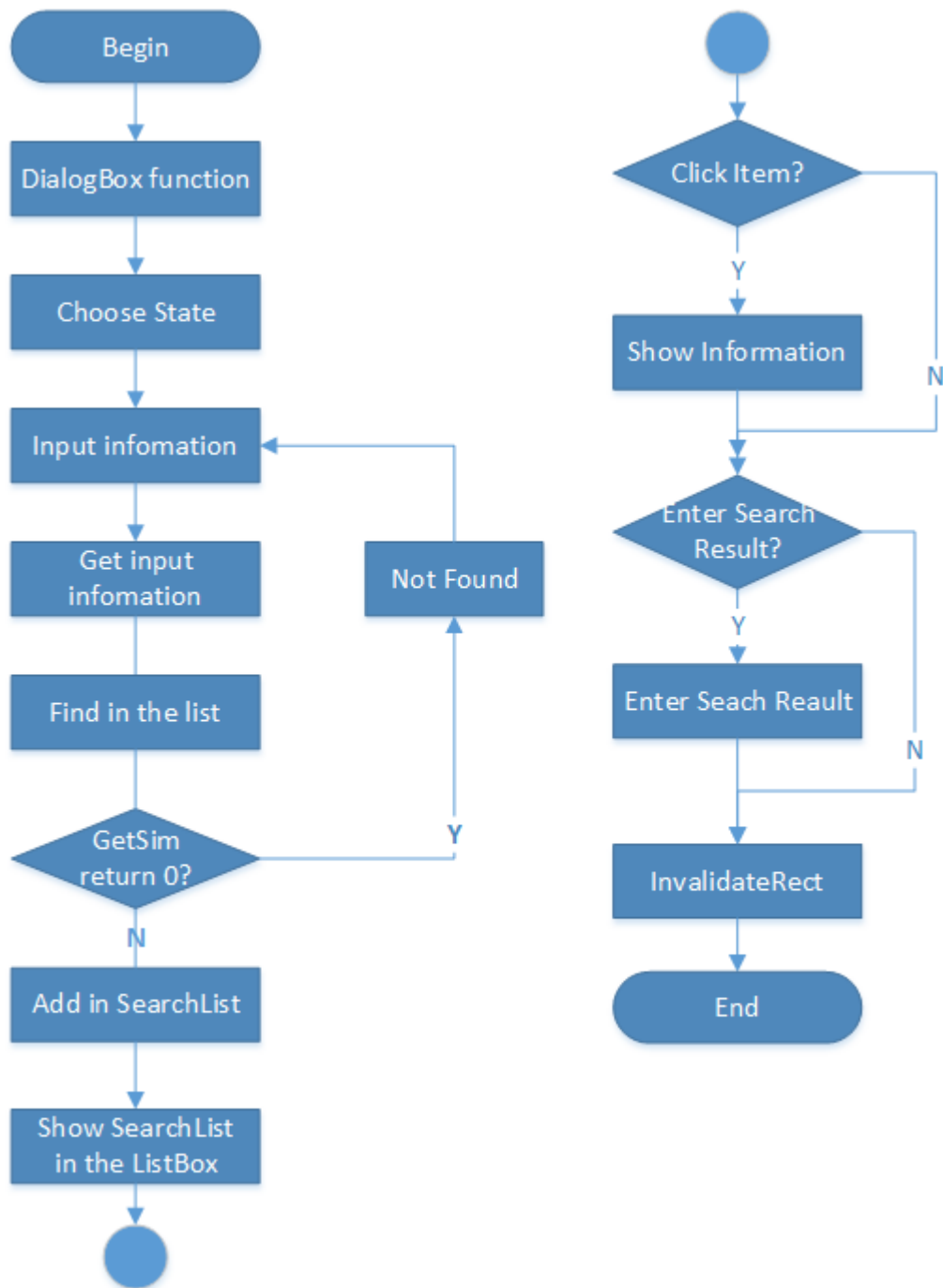


图 4.3 搜索过程主要流程

搜索结构所显示的排序方式以及原理在 2.2 显示模块中都有详细的介绍，搜索过程中显示的函数原型如下：

```

////////////////////////////////////
//计算出每一个楼栋的相似度
//参数：标准SearchData结构，对比的楼栋，该楼栋所在楼盘
//返回：这个输入信息与该楼栋的相似度

```

```

int GetBuiSim(SearchData data, Community com, Building bui);

```

---

```

////////////////////////////////////
//计算出每一个房间的相似度
//参数：标准SearchData结构，对比的房间，该房间所在楼栋、楼盘
//返回：这个输入信息与该房间的相似度
int GetRoomSim(SearchData data, Community com, Building bui, Room room);

////////////////////////////////////
//“搜索功能”（Search）对话框窗口过程
BOOL CALLBACK SearchDlgProc(HWND hDlg, UINT message,
    WPARAM wParam, LPARAM lParam);

////////////////////////////////////
//对Com链表按照权重进行排序
//参数：标准比较输入数据，排序的链表头指针
//返回：返回为排完序的链表的头指针
Community *RankCom(SearchData sData, Community *pComSearch);

////////////////////////////////////
//对Bui链表按照权重进行排序
//参数：标准比较输入数据，排序的链表头指针
//返回：返回为排完序的链表的头指针
Building *RankBui(SearchData sData, Building *pBuiSearch);

////////////////////////////////////
//AddBuiToSearch
//功能：创建一个只由bui组成的search
//参数：将要添加的Building结构体，添加对象链表的头指针
//返回：成功则返回TRUE，失败返回FALSE
BOOL AddBuiToSearch(Building newCom, Building **head);

////////////////////////////////////
//对Room链表按照权重进行排序
//参数：标准比较输入数据，排序的链表头指针
//返回：返回为排完序的链表的头指针
Room *RankRoom(SearchData sData, Room *pBuiSearch);

////////////////////////////////////
//计算出每一个楼盘的相似度
//参数：标准SearchData结构，对比的楼盘
//返回：这个输入信息与该楼盘的相似度

int GetComSim(SearchData data, Community com);

```

---

函数的参数及返回值参见注释，函数的具体实现代码见附录。

## 5.5 统计模块

统计模块的主要体现在数据的更新与实现上，由于统计模块并无专门的菜单项，因此统计功能融入其它的模块中。

统计中最重要的函数为更新数据，功能为遍历链表，将楼栋数、房间数、平均价格计算并写入对应的结构体。函数原型如下：

```
////////////////////////////////////  
//UpdateData  
//功能：更新整个链表的数据，包括房间数，均价等  
//参数：将要更新的链表的头指针的地址  
//返回：若完成所有更新则返回TRUE，否则返回FALSE
```

```
BOOL UpdateData(Community **ppHead);
```

函数的参数及返回值参见注释，函数的具体实现代码见附录。

## 5.6 帮助模块

由于本软件界面较为简单，操作简洁，无需准备无帮助文档，帮助功能仅提供关于软件功能。即弹出对话框显示软件的基本信息，该对话框在资源中定义一些 Static 控件即可。

## 5 系统测试

### 5.1 文件模块

#### 2.1.1 保存

通过该功能将已经通过系统做的修改、增添和删除保存至磁盘文件。具体实现方法如下：

当用户单击菜单中的“保存”时，若磁盘若无对应文件则激活“另存为”，若磁盘中有对应文件，则直接将修改写入至磁盘对应文件。

#### 2.1.2 另存为

若为“保存”激活，则保存文件至磁盘某一位置；若为直接点击，则为该文件在磁盘上创建一个副本。具体实现方法如下：

1) 当单击菜单中的“另存为”时（或对未保存至磁盘的文件单击“保存”），将弹出一个 Windows 中常用的保存窗口，如图 5.1 所示；

2) 用户可以通过这个窗口中选择将要保存的位置，以及将要保存的文件形式（目前系统仅支持\*.led 格式，该格式为自定义的引导文件格式）；

3) 用户若单击取消，则恢复原状。若用户单击确定，则文件保存至磁盘相应目录下。用户可到磁盘中查询到四个文件：一个引导文件 Leading Files(\*.led)，三个数据文件(\*.bin)，如图 5.2 所示。其文件名为系统自动确定，不允许用户更改。

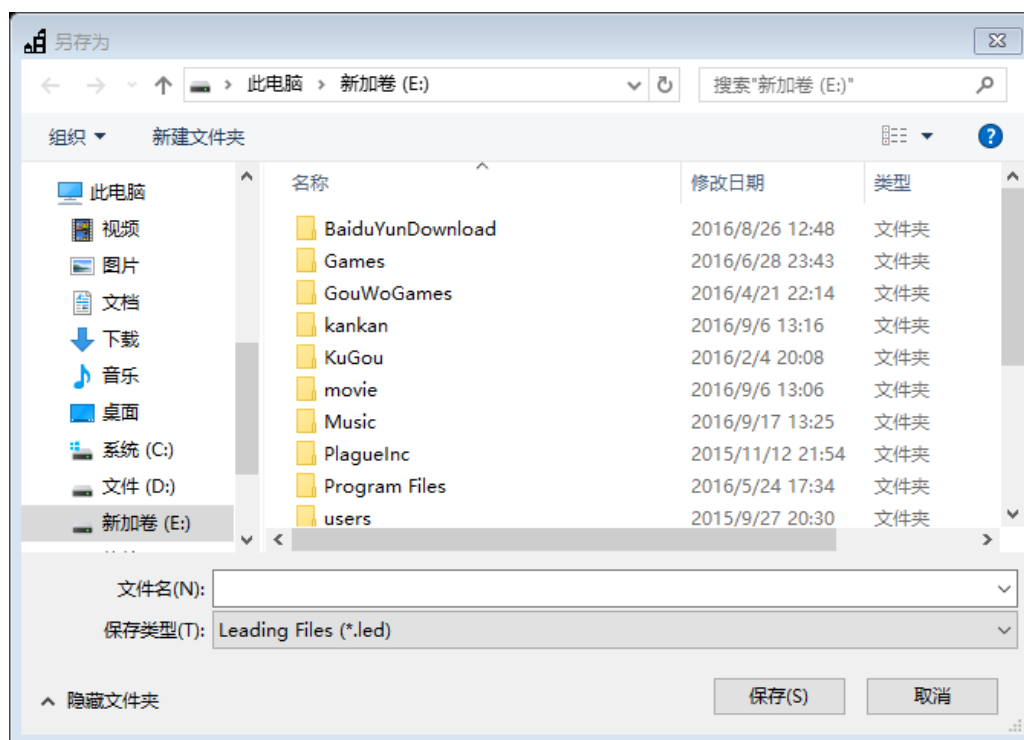


图 5.1 单击“另存为”跳出的保存窗口

### 2.1.3 新建

新建一个空的文件，供用户进行操作（添加、修改、删除等）。实际操作中可跳过这一步，直接在空的系统中操作后保存。




 武汉市楼盘信息 - 房间.bin	2016/9/21 20:45	BIN 文件	230 KB
 武汉市楼盘信息 - 楼栋.bin	2016/9/21 20:45	BIN 文件	20 KB
 武汉市楼盘信息 - 楼盘.bin	2016/9/21 20:45	BIN 文件	7 KB
 武汉市楼盘信息.led	2016/9/21 20:45	LED 文件	1 KB

图 5.2 保存后的文件在磁盘中的显示

### 2.1.4 导入

选择磁盘中的某一个文件（要求为 Leading Files (\*.led) 文件）导入该管理系统，并将文件中的所有数据读入链表，同时在系统的窗口中显示。具体实现方法如下：

- 1) 点击导入之后，弹出一个类似于“保存”的窗口。可通过选择文件格式来筛选文件（默认格式为\*.led），单击文件表示选择。
- 2) 若点击确定，则将该文件下的数据文件导入链表，并在系统中显示第一个楼盘的信息，如图 5.3 所示。若点击取消，则恢复原状。



图 5.3 导入后系统显示的信息

### 2.1.5 退出

退出实现功能与点击右上角的退出按钮相同。首先确认文件是否保存和做过修改。若已修改但未保存，将会弹出窗口提示是否要保存，如图 5.4 所示，否则直接退出系统。



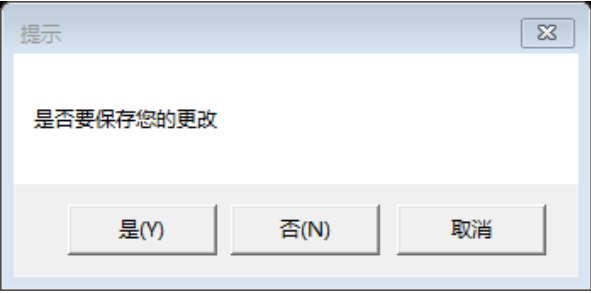


图 5.4 提示保存对话框

## 5.2 显示模块

显示模块不存在于菜单中。为了方便用户浏览信息，实现编辑、统计、添加的同步显示，做到“所见即所得”，本系统设计了显示模块。显示模块允许用户在除菜单外的客户区进行单击、双击操作，同时为其它许多功能的操作提供了便利，带来了良好的用户体验。下面具体描述显示的具体过程。

1) 当进入系统导入文件之后，显示的是第一级，楼盘与该楼盘下所属的楼栋信息，如图 5.5 所示；

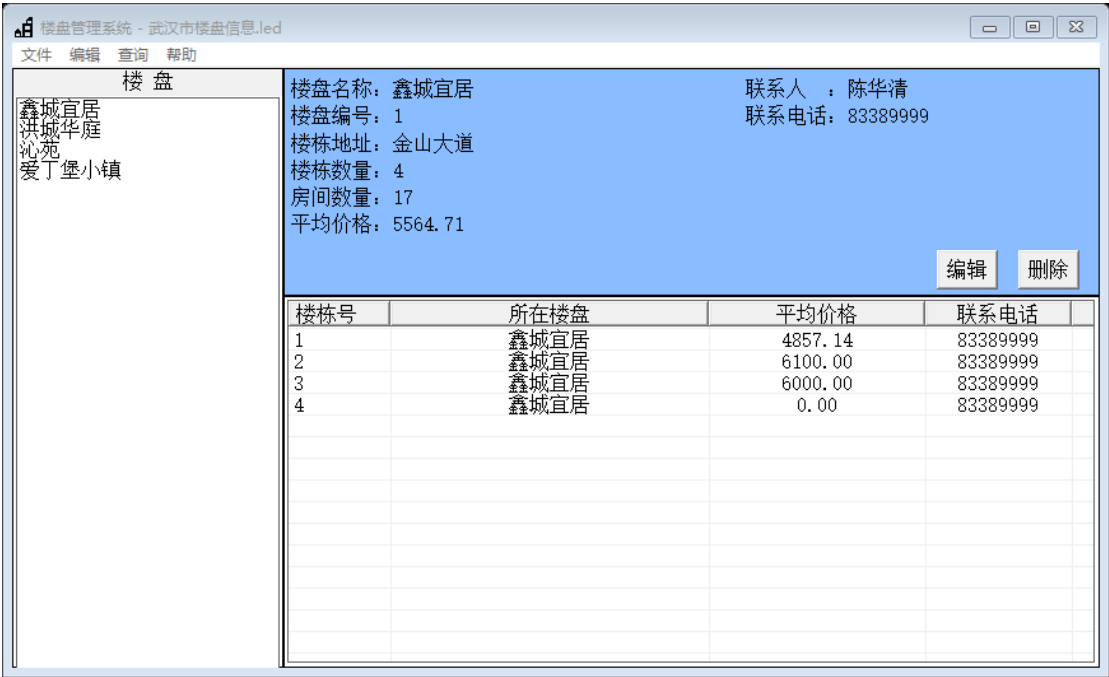


图 5.5 显示第一级楼盘与楼栋

2) 单击每一个楼盘，在右上方蓝色信息区会显示该楼盘的具体信息，同时在右下方的列表中显示该楼盘下的所有楼盘。同时信息区右下角的两个按钮“编辑”与“删除”提供的快速删除、编辑当前楼盘的有效路径；

3) 在 2) 的基础上单击右下角列表中每一个楼栋，在右上方蓝色信息区会显示对应的楼栋信息。同时信息区右下角的两个按钮“编辑”与“删除”提供的快

速删除、编辑当前楼栋的有效路径；

4) 在 2) 的基础上双击右下列表中的楼栋，将可以进入该楼栋，即在右上角显示该楼栋的信息，在右下角显示该楼栋下的所有房间及基本信息。同时信息区右下角的两个按钮“编辑”与“删除”提供的快速删除、编辑当前楼栋的有效路径，如图 5.6 所示；

5) 在 4) 的基础上单击右下角列表中的房间，将在右上角蓝色信息区中显示该房间的所有信息，如图 5.7 所示。同时信息区右下角的两个按钮“编辑”与“删除”提供的快速删除、编辑当前房间的有效路径；

6) 在 4) 的基础上，单击右下方列表中的第一行楼栋基本信息，将显示该楼栋所有信息，并提供“编辑”与“删除”楼栋功能。双击右下方列表中的第一行楼栋基本信息，将返回上一级，即右下列表中显示该楼盘下所有楼栋信息，右上信息区中显示该楼栋的所有信息。同时信息区右下角的两个按钮“编辑”与“删除”提供的快速删除、编辑当前房间的有效路径；

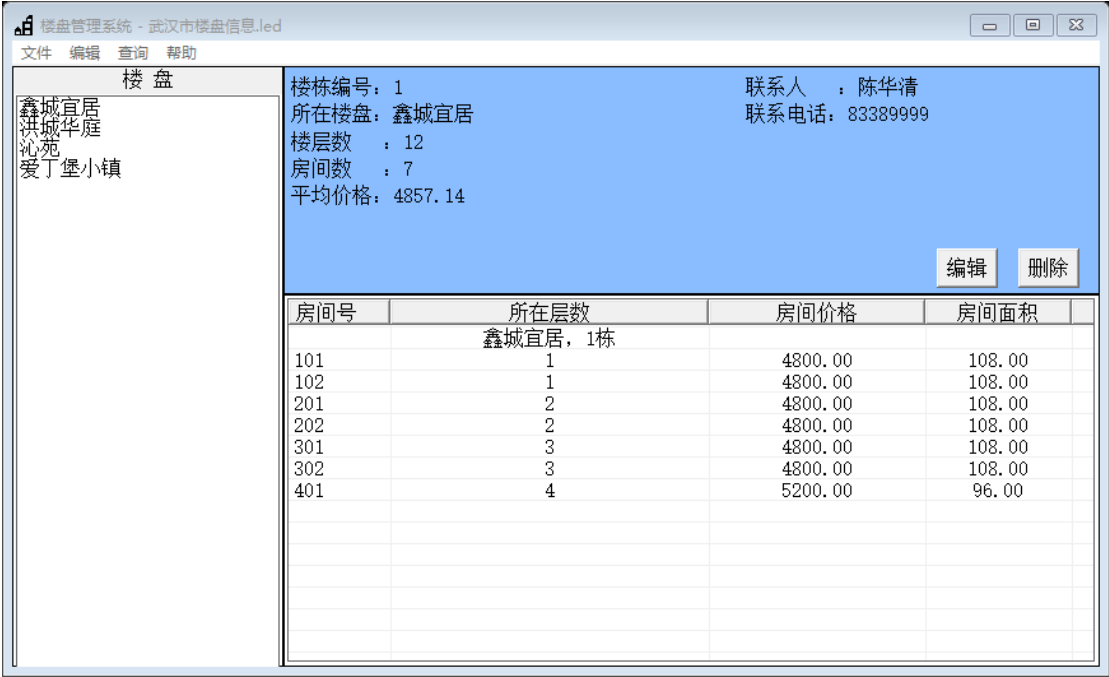


图 5.6 双击楼栋显示第二级信息

## 5.3 编辑模块

### 5.3.1 更改功能

更改功能又称编辑，可以通过两种方式触发：单击菜单中的“更改”项或在浏览房间/楼栋/楼盘信息之时点击当时所在的房间/楼栋/楼盘信息右下角的编辑按钮。具体实现功能如下：

1) 通过以上两种方法触发弹出更改功能对话框，如图 5.8 所示；

2) 对话框会默认将目前所在房间/楼栋/楼盘的信息输入对应的编辑框, 允许用户在编辑框上更改相关的数据。(有一些编辑框如电话号码不允许输入数字以外的符号)

3) 若用户点击确定, 系统将检查所输入的楼栋号(或房间号、楼盘名)是否与已有数据重复, 若重复, 则弹出警告, 如图 5.9 所示, 要求用户更改所输入的数据。若成功编辑房间, 则同时更新该楼盘、楼栋的平均价格, 若成功编辑楼盘, 则同时更新当前楼盘下的所有楼栋、房间的“所在楼盘”数据。若用户点击取消, 则不做更改。

4) 更改完成后, 重绘整个客户区, 将数据更新。用户可以通过系统显示判断更改是否成功。

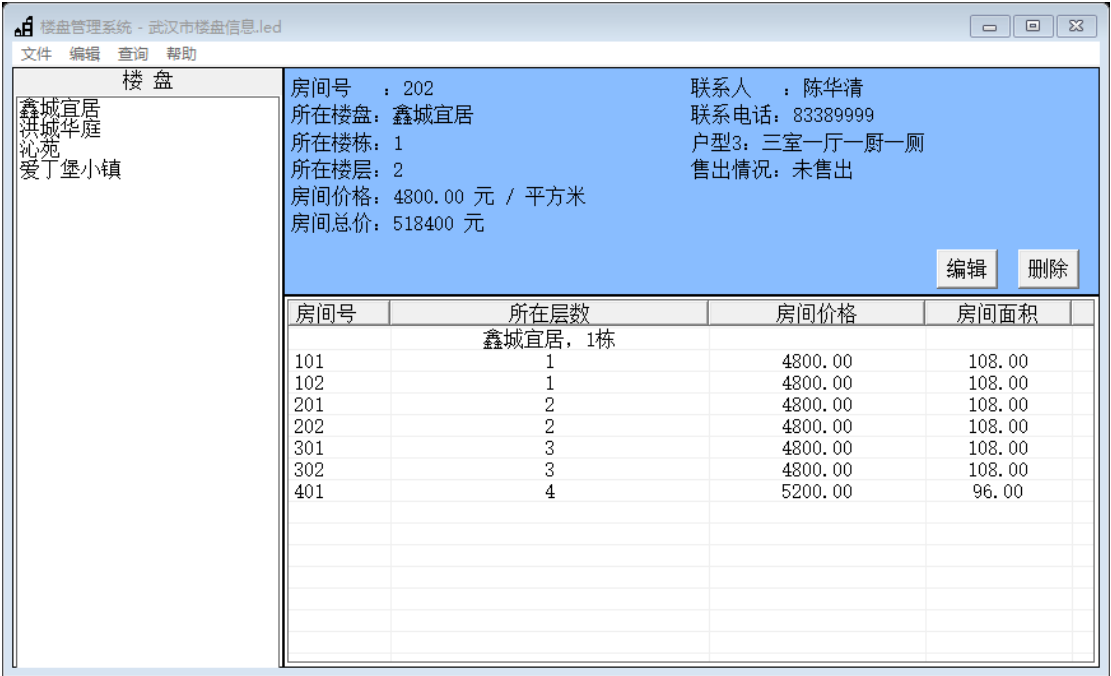


图 5.7 单击房间显示房间信息

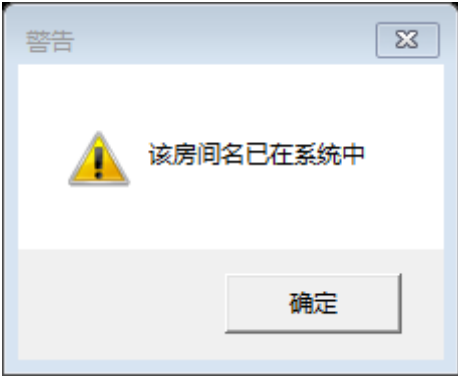


图 5.9 弹出的修改无效警告

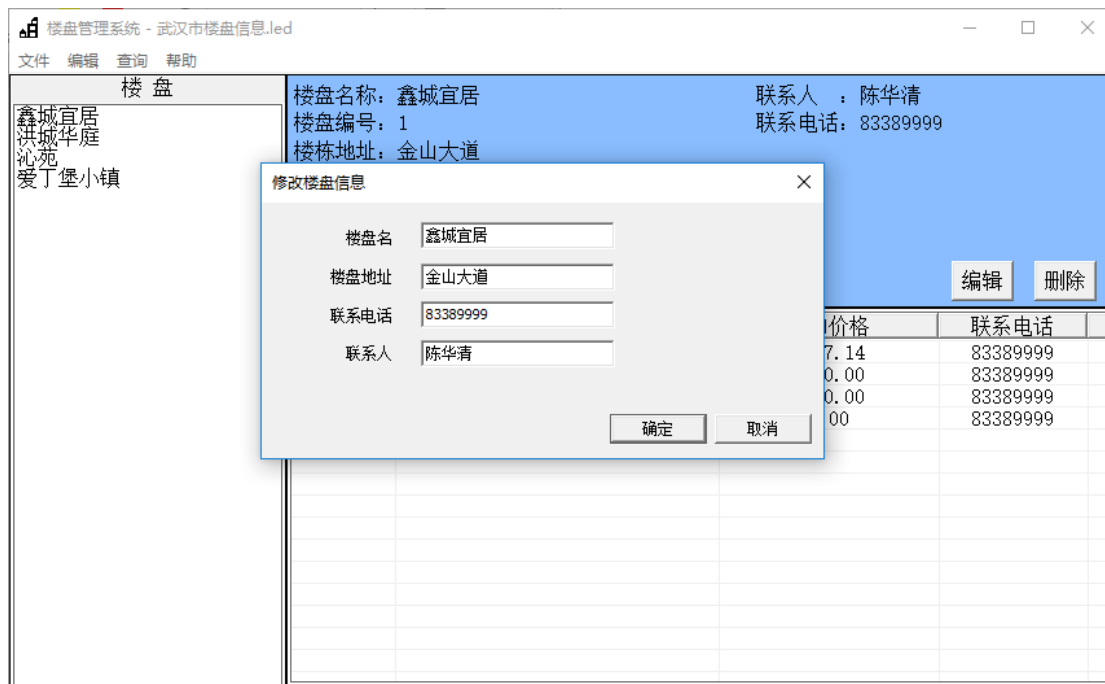


图 5.8 激活的更改对话框

### 2.3.2 添加功能

添加功能有分为三大部分：添加楼盘，添加楼栋和添加房间。

#### a. 添加房间

添加房间允许在系统任何的显示状态下进行。当单击菜单中“添加 – 添加房间”时，会弹出对话框。若显示当前信息区显示为楼盘，则会在对话框中自动输入默认楼盘的相关信息；若当前信息区显示为楼栋，则会在对话框中自动输入默认楼栋的相关信息，如图 5.10 所示。

在添加房间的对话框中使用了 Edit, Static, Combo Box (下拉选择框), Check Box 等许多控件。其中下拉选择框中储存了所有的房间类型信息共 34 条，方便用户选择。

在输入完毕后，用户若点击确定，则系统会检测是否有重复的房间号，若有则报错，要求用户重新输入信息，则添加成功，用户可在列表框中查看到新添加的房间，或点击查看具体信息；若点击取消，则放弃添加，系统恢复原状。

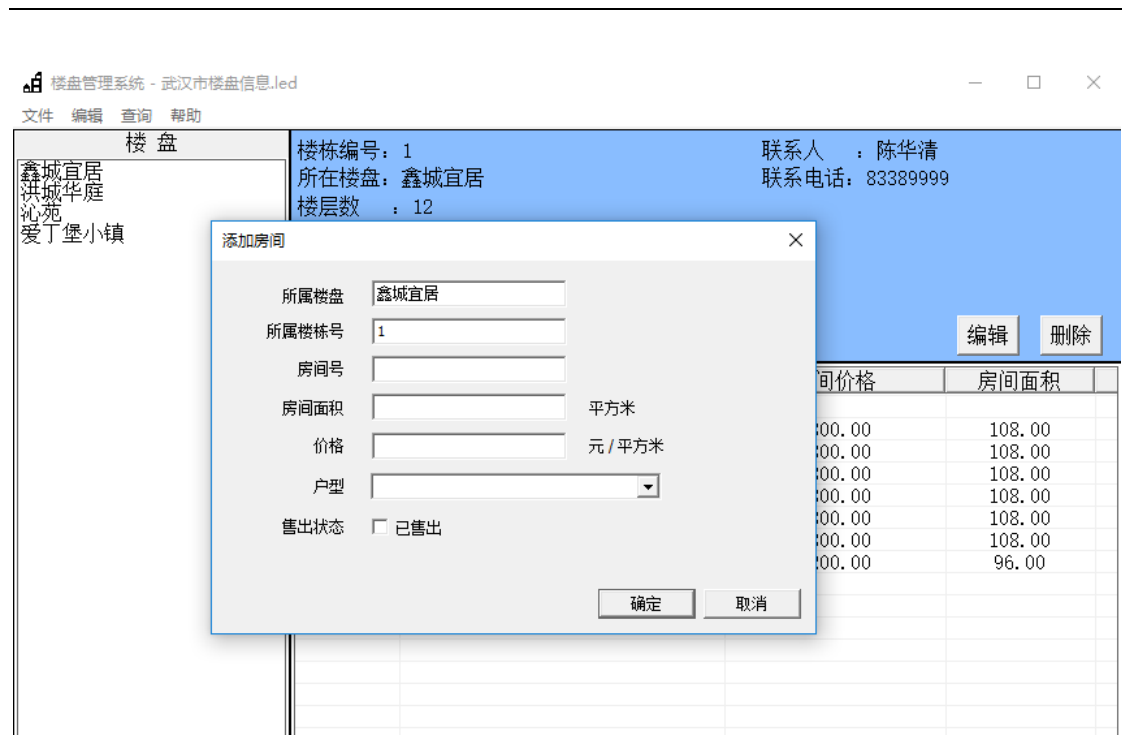


图 5.10 添加房间默认情况

#### b. 添加楼栋

添加楼栋允许在系统任何的显示状态下进行。当单击菜单中“添加 – 添加楼栋”时，会弹出对话框，且会在对话框中自动输入当前信息区显示楼盘的相关信息，如图 5.11 所示。

在输入完毕后，用户若点击确定，则系统会检测是否有重复的房间号，若有则报错，要求用户重新输入信息，若无，则添加成功，用户可在列表框中查看到新添加的楼栋，或点击查看具体信息；若点击取消，则放弃添加，系统恢复原状。

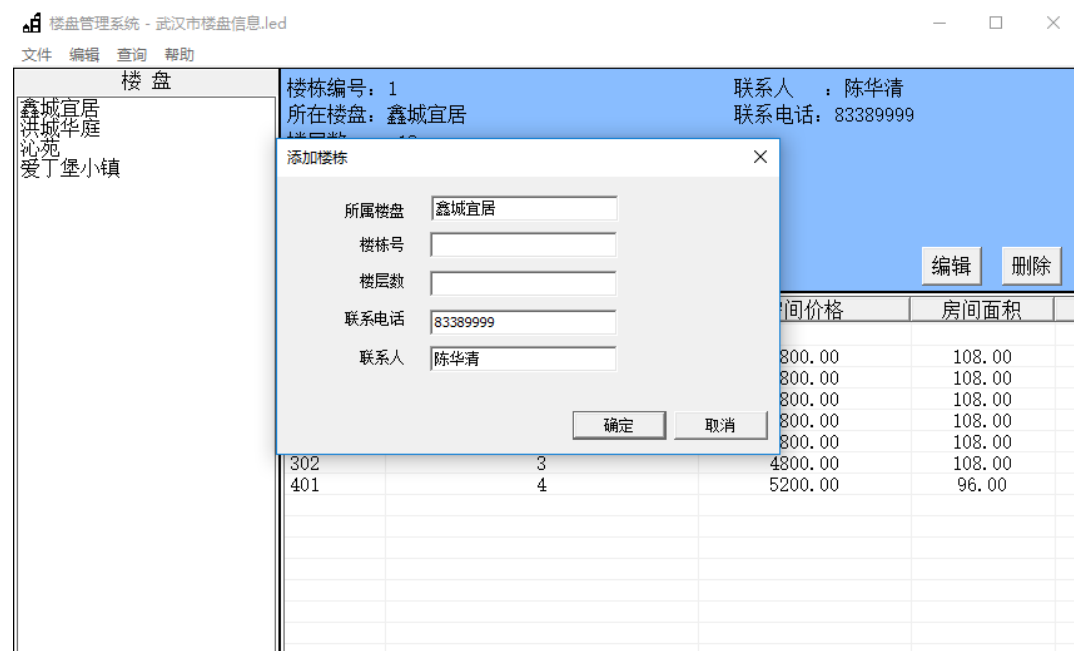


图 5.11 添加楼栋默认情况

### c. 添加楼盘

添加楼盘允许在系统任何的显示状态下进行。当单击菜单中“添加 – 添加楼栋”时，会弹出对话框，用户输入需要添加的楼盘即可。

在输入完毕后，用户若点击确定，则系统会检测是否有重复的楼盘名，若有则报错，要求用户重新输入信息，若无，则添加成功，系统自动按照顺序添加楼盘号，用户可在左边列表中查看到新添加的楼盘名称，或点击查看具体信息；若点击取消，则放弃添加，系统恢复原状。

### 2.3.3 删除功能

删除功能提供两种操作方式：准确查找删除与快速删除。

#### a) 准确查找删除

该种删除方式在菜单栏中的“删除”中，分为删除楼盘、楼栋和房间三种。

点击“删除楼盘”时，会弹出对话框，如图 5.12。允许选择性地输入楼盘名称或楼盘编号，选择其中一项是，另一项的输入框会变灰色，不允许输入。若点击确定，则直接删除该楼盘；若点击取消，则放弃删除。

点击“删除楼栋”时，要求选择性地输入所在楼盘名称或楼盘编号，以及准确的楼栋编号。若点击确定，则直接删除该楼栋；若点击取消，则放弃删除。

点击“删除房间”时，要求选择性地输入所在楼盘名称或楼盘编号、所在楼栋的准确的楼栋编号，以及准确的房间号。若点击确定，则直接删除该房间；若点击取消，则放弃删除。

#### b) 快速删除

考虑到用户很有可能忘记或者记不清准确的编号信息，系统提供快速删除的功能。即用户在浏览楼盘、楼栋或房间的具体信息可以直接对相关结构进行删除。

具体操作为单击信息栏的删除按钮，系统会自动判断需要删除的是楼栋、楼盘或房间，然后弹出提示框确认是否删除（防止误点），如图 5.13 所示，若用户点击确认，则删除该结构，若用户点击取消，则放弃删除。

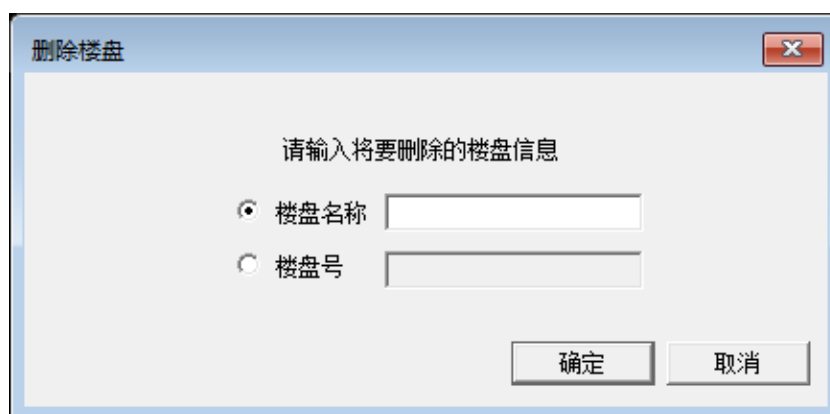


图 5.12 删除楼盘的窗口

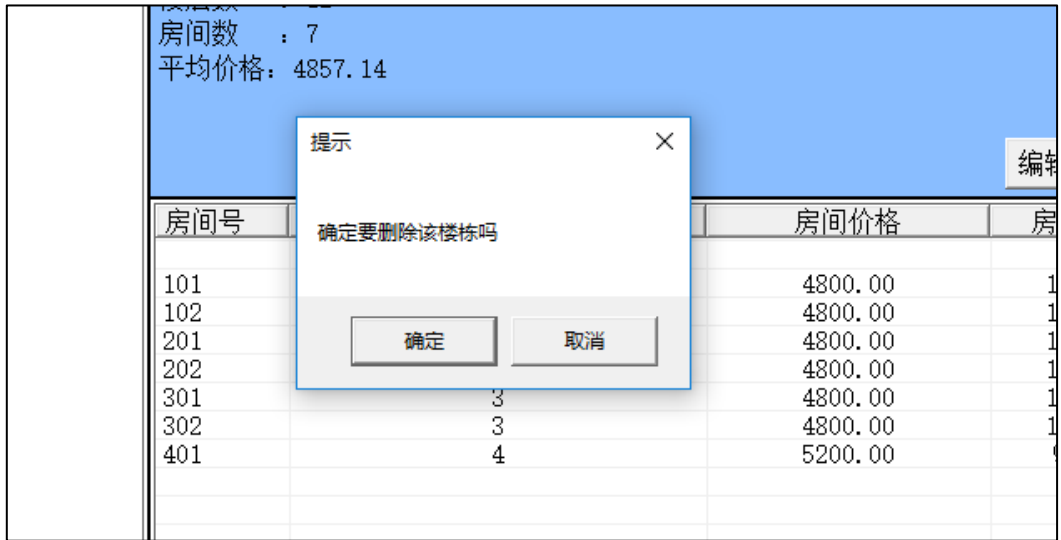


图 5.13 快速删除楼栋提示框

## 5.4 查询模块

查询模块主要功能体现在菜单中单击“查询”弹出的对话框中，如图 5.14 所示。查询功能主要分为查询楼盘，查询楼栋，查询房间三个部分，通过对话框中的选择按钮选择将要查询的部分，后面对应的信息栏将会按照选择相应的变灰，即不允许输入。

**搜索**

**楼盘信息**

☐ 搜索楼盘

楼盘名称  楼盘编号

联系人  联系电话

**楼栋信息**

☒ 搜索楼栋

楼栋编号  联系电话

联系人

**房间信息**

☐ 搜索房间

房间编号  房间面积

所在层数  总价上限  (万元)

房间价格范围  --  元 / 平方米

户型  ☐ 已售出

**确定** **取消**

图 5.14 查询搜索对话框

选择好将要输入的信息后，用户可以在编辑框、选择框中输入自己所想要搜索的信息，但不要求准确。具体提供的搜索功能见图 5.15。

系统将会录入用户所输入的搜索信息，并根据用户的选择，将用户所输入的信息与系统中的每一个结构进行比对。系统内部会将每一条信息的重要性分配权值，并计算每一个结构与所输入每一条信息的相似度，然后所有信息相似度的加权平均值作为这一个结构与输入信息的相似度。系统会汇总所有的相似度不为 0（即至少有一条信息与所输入信息相符）的结构，并按照权值排序作为搜索结果。最终，系统将会将搜索结果按照相似度从高到低在左边列表中输出。若在搜索中输入房间号为 1，楼栋号为 1，点击确定，则搜索结果如图 5.16 所示。同时，会在列表的最后加上一条“退出搜索”的按钮。

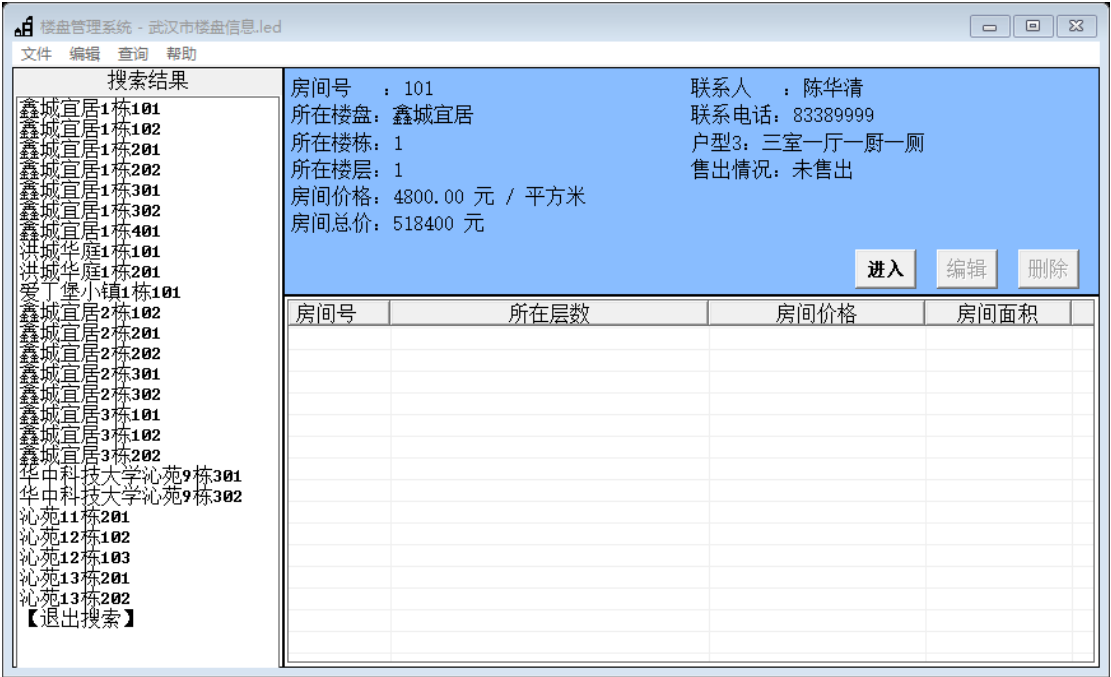


图 5.15 搜索结果示意图

显示搜索结果后，用户为了确认是否为自己想要搜索的项目，可以单击左边搜索栏中的每一个搜索结果，其全部信息将会在右边的蓝色信息区中显示。进入搜索状态后，右上信息区会出现一个“进入”按钮，当用户浏览信息后确认是想要的搜索结果时，可以点击“进入”按钮，直接进入当前搜索结果。以房间为例，点击进入后，将会在左边列表中显示楼盘信息，右上信息栏中显示该房间信息，右下列表中显示所在楼盘的所有房间，若点击沁苑 11 栋 201，则显示如图 2.19 所示。若用户未找到对应的搜索结果，则直接点击“【退出搜索】”，则恢复搜索前的状态不做改变。

按照输入的搜索结果，不存在房间号为 1 的房间，因此未找到，但是存在楼栋号为 1 的房间，因此楼栋为 1 的房间在最前面显示。同时，由于未勾选“已售出”选项，搜索所有未售出的楼盘，系统会将一些未售出但是楼栋号不是 1 的房间显示在偏后的位置，供用户查看。





图 5.16 搜索结果进入后的显示

## 5.5 统计模块

由于可视化 Windows 程序的特殊性，本程序未在菜单中加入专门的统计栏，而是将统计融入在程序的显示中。包括统计平均价格，统计楼栋所有房间数，楼盘所有房间数、楼栋数，统计已售出、未售出的房间，目前所有的楼盘数，统计平均面积等。

每次进行数据添加、更改、删除时，系统会自动将所有的相关数据全部更改，并在系统的界面上实时显示。由于楼栋编号为顺序，通过最后的楼盘编号即可了解所有盘数。且通过搜索功能可以达到更加实用和强大的统计功能，例如统计所有未售出的房间，统计所有价格在某一个价格段的房间等。

## 5.6 帮助模块

由于本系统未设置专门的参考文档，因此只在关于软件中提供版本信息、作者信息等信息。单击菜单中帮助的“关于软件”，即可得到相关信息，如图 5.17 所示。



图 5.17 关于软件

## 6 系统界面设计

本程序采用 Windows API 进行界面设计，通过调用 API 函数进行简单的窗口程序界面设计。借鉴了 Windows 资源管理器和 KeePass 的设计思路，设计了基本界面。通过单击、双击切换层级，如图 6.1 所示。



图 6.1 主界面

---

## 7 总结与体会

在本次的实验中，我的最大的体会会有以下几点：

1、查阅资料很重要。由于我是用的是 Windows API，需要在 MSDN 上查询许多的资料。同时，我在 Google，StackOverFlow 上也寻求了许多的帮助。在这次编程中，我体会到了查阅资料是编程和提高编程所必须的能力。

2、调试程序不可少。为了避免最后大量的 bug 所带来的调试困扰，我采用了完成一个功能，调试一个功能，完善一项功能的方法。发现一个问题，解决一个问题，步步为营，最终完善整个系统。

3、用户体验不能忘。我的编程过程非常重视用户体验，对于任何对用户体验有影响的地方我都会想尽办法改进。只有把系统做到自己愿意使用，方便使用，其他人才可能用起来舒服、顺手。

4、先完成主要功能在完善。主要功能是一个系统的声明，不能只注重与华而不实的功能而忽视了主要的系统功能。只有当完成了主要功能后再向其中加入较多的新功能，例如登录界面、文件头加密等，才能使系统更加的实用。

5、参考前人的经验。在设计 Windows 程序中，有很多现成的优秀的 Windows 软件可以提供参考。例如在我的设计中，参考了 Windows 文件资源管理器和 KeePass 的界面设计经验。许多优秀的软件可以提供更多合理的设计思路。

---

## 附录

### 1、 源文件（RoomSearch.cpp）

```
#include "managerSys.h"
#include "resource.h"

#pragma comment(lib, "comctl32.lib")

static OPENFILENAME ofn;
static Community *pHead = NULL;
static Community *pComSearch = NULL;
static Building *pBuiSearch = NULL;
static Room *pRoomSearch = NULL;
static int companyNum = 1;
static int iPaintState;
static BOOL bRePaint;
static TCHAR comNow[100]; //目前所在com
static int buiNow; //目前所在bui
static int roomNow; //目前所在Room
static int iSearch = NOSEARCH; //是否正在搜索
RECT rectData;
Community pcBuffer[10];
Building pbBuffer[10];
Room prBuffer[10];
HWND hwnd, hwndMainList, hwndMainTitle, hwndSubTitle;
HWND hwndButtonDel, hwndButtonEdit, hwndButtonEnter;
HWND hwndSubList;
HFONT hFontSon, hFontBlack;
enum fState {SUCCEED, FAIL};
enum fState fAddBuiState; //是否成功添加楼栋
enum fState fAddRoomState; //是否成功添加房间
int iAutoComNum = 1, iAutoBuiNum = 101;

static TCHAR *szComboBoxData[] = {
    TEXT("户型1: 一室一厅一厨一厕"), TEXT("户型18: 五室一厅一厨两厕"),
    TEXT("户型2: 两室一厅一厨一厕"), TEXT("户型19: 两室两厅两厨一厕"),
    TEXT("户型3: 三室一厅一厨一厕"), TEXT("户型20: 三室两厅两厨一厕"),
    TEXT("户型4: 四室一厅一厨一厕"), TEXT("户型21: 四室两厅两厨一厕"),
    TEXT("户型5: 五室一厅一厨一厕"), TEXT("户型22: 五室两厅两厨一厕"),
    TEXT("户型6: 一室两厅一厨一厕"), TEXT("户型23: 两室两厅一厨两厕"),
    TEXT("户型7: 两室两厅一厨一厕"), TEXT("户型24: 三室两厅一厨两厕"),
    TEXT("户型8: 三室两厅一厨一厕"), TEXT("户型25: 四室两厅一厨两厕"),
    TEXT("户型9: 四室两厅一厨一厕"), TEXT("户型26: 五室两厅一厨两厕"),
```

---

```

TEXT("户型10: 五室两厅一厨一厕"),TEXT("户型27: 两室一厅两厨两厕"),
TEXT("户型11: 两室一厅两厨一厕"),TEXT("户型28: 三室一厅两厨两厕"),
TEXT("户型12: 三室一厅两厨一厕"),TEXT("户型29: 四室一厅两厨两厕"),
TEXT("户型13: 四室一厅两厨一厕"),TEXT("户型30: 五室一厅两厨两厕"),
TEXT("户型14: 五室一厅两厨一厕"),TEXT("户型31: 两室两厅两厨两厕"),
TEXT("户型15: 两室一厅一厨两厕"),TEXT("户型32: 三室两厅两厨两厕"),
TEXT("户型16: 三室一厅一厨两厕"),TEXT("户型33: 四室两厅两厨两厕"),
TEXT("户型17: 四室一厅一厨两厕"),TEXT("户型34: 五室两厅两厨两厕"),
};

int WINAPI WinMain(HINSTANCE hInstance, HINSTANCE hPrevInstance,
    PSTR szCmdLine, int iCmdShow)
{
    static TCHAR szWinName[] = TEXT("RoomSearch");
    MSG        msg;
    WNDCLASS    wndclass;

    wndclass.style = CS_HREDRAW | CS_VREDRAW;
    wndclass.lpfnWndProc = WndProc;
    wndclass.cbClsExtra = sizeof(unsigned int);
    wndclass.cbWndExtra = sizeof(unsigned int);
    wndclass.hInstance = hInstance;
    wndclass.hIcon = LoadIcon(hInstance, MAKEINTRESOURCE(IDI_MainIcon));
    wndclass.hCursor = LoadCursor(NULL, IDC_ARROW);
    wndclass.hbrBackground = (HBRUSH)GetStockObject(WHITE_BRUSH);
    wndclass.lpszMenuName = (LPCWSTR)IDM_RoomSearch;
    wndclass.lpszClassName = szWinName;

    if (!RegisterClass(&wndclass))
    {
        MessageBox(NULL, TEXT("本程序仅支持 Windows NT 以上系统!"),
            szWinName, MB_ICONERROR);
        return 0;
    }

    hwnd = CreateWindow(szWinName, TEXT("楼盘管理系统"),
        WS_OVERLAPPEDWINDOW,
        263, 127,
        872, 532,
        NULL, NULL, hInstance, NULL);

    HDC hdc = GetDC(hwnd);

```

---

```

LOGFONT LogFont;
memset(&LogFont, 0, sizeof(LOGFONT));
lstrcpy(LogFont.lfFaceName, TEXT("宋体"));
LogFont.lfWeight = 400;
LogFont.lfHeight = GetDeviceCaps(hdc, LOGPIXELSY) * 12 / 72; // 字体大小
LogFont.lfWidth = 0;
LogFont.lfCharSet = GB2312_CHARSET;
LogFont.lfOutPrecision = 3;
LogFont.lfClipPrecision = 2;
LogFont.lfOrientation = 45;
LogFont.lfQuality = 1;
LogFont.lfPitchAndFamily = 2;

LOGFONT font;
ZeroMemory(&font, sizeof(LOGFONT));
font.lfHeight = 18;
font.lfQuality = PROOF_QUALITY;
wsprintf(font.lfFaceName, TEXT("微软雅黑"));
hFontBlack = CreateFontIndirect(&font);

ReleaseDC(hwnd, hdc);

// 创建字体“宋体”
hFontSon = CreateFontIndirect(&LogFont);

ShowWindow(hwnd, iCmdShow);
UpdateWindow(hwnd);

while (GetMessage(&msg, NULL, 0, 0))
{
    TranslateMessage(&msg);
    DispatchMessage(&msg);
}
return msg.wParam;
}

////////////////////////////////////
//主窗口窗口过程

LRESULT CALLBACK WndProc(HWND hwnd,
    UINT message, WPARAM wParam, LPARAM lParam)
{
    HDC hdc;
    PAINTSTRUCT ps;

```

---

```

TEXTMETRIC tm;

static HWND hEdit;
static TCHAR szFileName[MAX_PATH], szFileTitle[MAX_PATH];
static RECT rectClass, rectSub;
static HINSTANCE hInstance;
static Community *pComPaint;
static Building *pBuiPaint;
static Room *pRoomPaint;
static int iSaveState = FALSE;
static int cxClient, cyClient, cxChar, cyChar;
static TCHAR sInf[50];
int i, iComNum, iBuiNum, iRoomNum, iAns;
static int iIndex;
Community *pTailCom;
Building *pTailBui, *pTailBuiSearch;
Room *pTailRoom, *pTailRoomSearch;
TCHAR szBuffer[400];
TCHAR szFormatBui[] = TEXT("%-20s%-5s%-10s%-15s");

switch (message)
{
case WM_CREATE:
    hInstance = ((LPCREATESTRUCT)lParam)->hInstance;
    InitCommonControls(); //初始化listview环境

    hdc = GetDC(hwnd);

    GetTextMetrics(hdc, &tm);
    cxChar = tm.tmAveCharWidth;
    cyChar = tm.tmHeight + tm.tmExternalLeading;

    ReleaseDC(hwnd, hdc);

    hwndSubList = CreateWindowEx(NULL, TEXT("SysListView32"), NULL,
        WS_VISIBLE | WS_CHILD | WS_BORDER | LVS_REPORT,
        0, 0, 0, 0, hwnd, (HMENU)ID_SUBLIST,
        (HINSTANCE)GetWindowLong(hwnd, GWL_HINSTANCE), NULL);
    ListView_SetExtendedListViewStyle(hwndSubList,
        LVS_EX_FULLROWSELECT | LVS_EX_GRIDLINES);
    SendMessage(hwndSubList, WM_SETFONT,
        (WPARAM)hFontBlack, TRUE);

    hwndMainList = CreateWindow(TEXT("listbox"), NULL,

```



---

```

        WS_VISIBLE | WS_CHILD | LBS_NOTIFY | WS_VSCROLL | WS_BORDER,
        0, 0, 0, 0, hwnd, (HMENU)ID_MAINLIST,
        (HINSTANCE)GetWindowLong(hwnd, GWL_HINSTANCE), NULL);
SendMessage(hwndMainList, WM_SETFONT,
        (LPARAM)GetStockObject(OEM_FIXED_FONT), TRUE);

hwndMainTitle = CreateWindow(TEXT("static"), TEXT("楼 盘"),
        WS_VISIBLE | WS_CHILD | SS_CENTER,
        0, 0, 0, 0, hwnd, (HMENU)ID_MAINLISTTITLE,
        (HINSTANCE)GetWindowLong(hwnd, GWL_HINSTANCE), NULL);
SendMessage(hwndMainTitle, WM_SETFONT,
        (LPARAM)GetStockObject(OEM_FIXED_FONT), 0);

hwndButtonDel = CreateWindow(TEXT("button"), TEXT("删除"),
        WS_VISIBLE | WS_CHILD | BS_PUSHBUTTON,
        0, 0, 0, 0, hwnd, (HMENU)ID_BUTTONDEL,
        hInstance, NULL);
SendMessage(hwndButtonDel, WM_SETFONT,
        (LPARAM)GetStockObject(OEM_FIXED_FONT), 0);

hwndButtonEdit = CreateWindow(TEXT("button"), TEXT("编辑"),
        WS_VISIBLE | WS_CHILD | BS_PUSHBUTTON,
        0, 0, 0, 0, hwnd, (HMENU)ID_BUTTONEDIT,
        hInstance, NULL);
SendMessage(hwndButtonEdit, WM_SETFONT,
        (LPARAM)GetStockObject(OEM_FIXED_FONT), 0);

hwndButtonEnter = CreateWindow(TEXT("button"), TEXT("进入"),
        WS_VISIBLE | WS_CHILD | BS_PUSHBUTTON,
        0, 0, 0, 0, hwnd, (HMENU)ID_BUTTONENTER,
        hInstance, NULL);
SendMessage(hwndButtonEdit, WM_SETFONT,
        (LPARAM)GetStockObject(OEM_FIXED_FONT), 0);

FileInitWnd(hwnd);
iSaveState = TRUE;

return 0;

case WM_SIZE:

    cxClient = LOWORD(lParam);
    cyClient = HIWORD(lParam);

```

---

```

rectData.left = XLINEPOS;
rectData.right = cxClient;
rectData.top = 0;
rectData.bottom = YLINEPOS;

MoveWindow(hwndSubList, XLINEPOS + 3, YLINEPOS + 3,
           cxClient - XLINEPOS - 6, cyClient - YLINEPOS - 6, TRUE);

MoveWindow(hwndMainList, 3, cyChar * 5 / 4 + 3,
           XLINEPOS - 6, cyClient - 6, TRUE);

MoveWindow(hwndMainTitle, 2, 2,
           XLINEPOS - 3, cyChar * 5 / 4, TRUE);

return 0;

case WM_COMMAND:

    switch (LOWORD(wParam))
    {
    case ID_BUTTONDEL:

        pTailCom = pHead;

        //循环查找楼盘名
        while (pTailCom != NULL)
        {
            if (!lstrcmp(pTailCom->name, comNow))
            {
                iComNum = pTailCom->communityNum;
                break;
            }
            pTailCom = pTailCom->nextCommunity;
        }

        switch (iPaintState)
        {
        case PA_FIRSTSTEP:
            iAns = MessageBox(hwnd, TEXT("确定要删除该楼盘吗"),
                             TEXT("提示"), MB_OKCANCEL);
            if (iAns == IDOK)
            {
                iSaveState = FALSE;
                DelComInList(iComNum, &pHead);
            }
        }
    }
}

```

---

```

        if (pHead)
        {
            lstrcpy(comNow, pHead->name);
            UpdateData(&pHead);

            //存储即将绘制的Com地址
            pComPaint = (Community*)malloc(sizeof(Community));
            *pComPaint = *pHead;
        }
        else pComPaint = NULL;

        //确定将要绘制的级别并无效窗口
        iPaintState = PA_FIRSTEPCOM;
        InvalidateRect(hwnd, NULL, TRUE);
    }
    else return 0;
    break;

case PA_SECSTEPBUI:
case PA_FIRSTSTEPBUI:

    iAns = MessageBox(hwnd, TEXT("确定要删除该楼栋吗"),
        TEXT("提示"), MB_OKCANCEL);
    if (iAns == IDOK)
    {
        iSaveState = FALSE;
        DelBuiInList(iComNum, buiNow, &pHead);

        for (pTailCom = pHead; pTailCom; pTailCom =
pTailCom->nextCommunity)
        {
            if (lstrcmp(comNow, pTailCom->name) == 0)
            {
                UpdateData(&pHead);

                //存储即将绘制的Com地址
                pComPaint = (Community*)malloc(sizeof(Community));
                *pComPaint = *pTailCom;
                if (pTailCom->buildings)
                    buiNow = pTailCom->buildings->buildingNum;
                else buiNow = 0;
                break;
            }
        }
    }

```

---

```

        //确定将要绘制的级别并无效窗口
        iPaintState = PA_FIRSTEPCOM;
        InvalidateRect(hwnd, NULL, TRUE);
    }
    else return 0;
    break;

case PA_SECSTEPROOM:
    iAns = MessageBox(hwnd, TEXT("确定要删除该房间吗"),
        TEXT("提示"), MB_OKCANCEL);
    if (iAns == IDOK)
    {
        iSaveState = FALSE;
        DelRoomInList(iComNum, buiNow, roomNow, &pHead);

        for (pTailCom = pHead; pTailCom; pTailCom =
pTailCom->nextCommunity)
            if (lstrcmp(comNow, pTailCom->name) == 0)
            {
                UpdateData(&pHead);

                //存储即将绘制的Com地址
                pComPaint = (Community*)malloc(sizeof(Community));
                *pComPaint = *pTailCom;
                for (pTailBui = pTailCom->buildings; pTailBui != NULL;
                    pTailBui = pTailBui->nextBuilding)
                    if (buiNow == pTailBui->buildingNum)
                    {
                        //存储将要绘制的Bui地址
                        pBuiPaint =
(Building*)malloc(sizeof(Building));
                        *pBuiPaint = *pTailBui;
                        break;
                    }
                break;
            }

        //确定将要绘制的级别并无效窗口
        iPaintState = PA_SECSTEPBUI;
        InvalidateRect(hwnd, NULL, TRUE);
    }
    else return 0;
    break;

```

---

```

        default:
            break;
    }
    break;

case ID_BUTTONEDIT:

    iSaveState = FALSE;

    switch (iPaintState)
    {
    case PA_FIRSTEPCOM:

        DialogBox(hInstance, MAKEINTRESOURCE(IDD_COMEDIT),
            hwnd, EditComDlgProc);

        if (pHead)
        {
            lstrcpy(comNow, pHead->name);
            UpdateData(&pHead);

            //存储即将绘制的Com地址
            pComPaint = (Community*)malloc(sizeof(Community));
            *pComPaint = *pHead;
        }
        else pComPaint = NULL;

        //确定将要绘制的级别并无效窗口
        iPaintState = PA_FIRSTEPCOM;
        InvalidateRect(hwnd, NULL, TRUE);

        break;

    case PA_SECSSTEPBUI:
    case PA_FIRSTSTEPBUI:

        DialogBox(hInstance, MAKEINTRESOURCE(IDD_BUIEDIT),
            hwnd, EditBuiDlgProc);

        for (pTailCom = pHead; pTailCom; pTailCom = pTailCom->nextCommunity)
            if (lstrcmp(comNow, pTailCom->name) == 0)
            {
                UpdateData(&pHead);
            }
    }
}

```

---

```

        //存储即将绘制的Com地址
        pComPaint = (Community*)malloc(sizeof(Community));
        *pComPaint = *pTailCom;
        for (pTailBui = pTailCom->buildings; pTailBui != NULL;
             pTailBui = pTailBui->nextBuilding)
            if (buiNow == pTailBui->buildingNum)
            {
                //存储将要绘制的Bui地址
                pBuiPaint = (Building*)malloc(sizeof(Building));
                *pBuiPaint = *pTailBui;
                break;
            }
        break;
    }

    //确定将要绘制的级别并无效窗口
    iPaintState = PA_FIRSTSTEPBUI;
    InvalidateRect(hwnd, NULL, TRUE);
    break;

case PA_SECSTEPROOM:

    DialogBox(hInstance, MAKEINTRESOURCE(IDD_ROOMEDIT),
              hwnd, EditRoomDlgProc);

    for (pTailCom = pHead; pTailCom; pTailCom = pTailCom->nextCommunity)
        if (strcmp(comNow, pTailCom->name) == 0)
        {
            UpdateData(&pHead);

            //存储即将绘制的Com地址
            pComPaint = (Community*)malloc(sizeof(Community));
            *pComPaint = *pTailCom;

            for (pTailBui = pTailCom->buildings; pTailBui != NULL;
                 pTailBui = pTailBui->nextBuilding)
                if (buiNow == pTailBui->buildingNum)
                {
                    //存储将要绘制的Bui地址
                    pBuiPaint = (Building*)malloc(sizeof(Building));
                    *pBuiPaint = *pTailBui;

                    for (pTailRoom = pTailBui->rooms; pTailRoom != NULL;
                         pTailRoom = pTailRoom->nextRoom)

```

---

```

        if (roomNow == pTailRoom->roomNum)
        {
            //存储将要绘制的Bui地址
            pRoomPaint = (Room*)malloc(sizeof(Room));
            *pRoomPaint = *pTailRoom;
        }
        break;
    }
    break;
}

//确定将要绘制的级别并无效窗口
iPaintState = PA_SECSTEPROOM;
InvalidateRect(hwnd, NULL, TRUE);

break;

default:
    break;
}

return 0;

case ID_BUTTONENTER:

    switch (iSearch)
    {
    case SEARCHCOM:

        //获取选中项文本
        SendMessage(hwndMainList, LB_GETTEXT,
            iIndex, (LPARAM)szBuffer);

        for (pTailCom = pHead; pTailCom; pTailCom = pTailCom->nextCommunity)
            if (lstrcmp(szBuffer, pTailCom->name) == 0)
            {
                //存储即将绘制的Com地址
                pComPaint = (Community*)malloc(sizeof(Community));
                *pComPaint = *pTailCom;
                lstrcpy(comNow, szBuffer);
                break;
            }

        EnableWindow(hwndButtonEdit, TRUE);

```

---

```

        EnableWindow(hwndButtonDel, TRUE);

        pComSearch = NULL;
        iPaintState = PA_FIRSTSTEPCOM;
        InvalidateRect(hwnd, NULL, TRUE);
        MoveWindow(hwndButtonEnter, 0, 0, 0, 0, TRUE);

        break;

    case SEARCHBUI:

        //获取选中项代表楼栋
        for (pTailBuiSearch = pBuiSearch, i = 0; pTailBuiSearch != NULL;
            pTailBuiSearch = pTailBuiSearch->nextBuilding, i++)
            if (i == iIndex)
                break;

        for (pTailCom = pHead; pTailCom; pTailCom = pTailCom->nextCommunity)
            if (strcmp(pTailBuiSearch->inCom, pTailCom->name) == 0)
                for (pTailBui = pTailCom->buildings; pTailBui;
                    pTailBui = pTailBui->nextBuilding)
                    if (pTailBuiSearch->buildingNum ==
pTailBui->buildingNum)
                        {
                            pBuiPaint = (Building*)malloc(sizeof(Building));
                            *pBuiPaint = *pTailBui;
                            buiNow = pTailBui->buildingNum;
                            break;
                        }

        EnableWindow(hwndButtonEdit, TRUE);
        EnableWindow(hwndButtonDel, TRUE);

        pBuiSearch = NULL;
        iPaintState = PA_SECSTEPBUI;
        InvalidateRect(hwnd, NULL, TRUE);
        MoveWindow(hwndButtonEnter, 0, 0, 0, 0, TRUE);

        break;

    case SEARCHROOM:

        //获取选中项代表房间
        for (pTailRoomSearch = pRoomSearch, i = 0; pTailRoomSearch != NULL;

```



---

```

        pTailRoomSearch = pTailRoomSearch->nextRoom, i++)
        if (i == iIndex)
            break;

        for (pTailCom = pHead; pTailCom; pTailCom = pTailCom->nextCommunity)
            if (lstrcmp(pTailRoomSearch->inCom, pTailCom->name) == 0)
                for (pTailBui = pTailCom->buildings; pTailBui;
                    pTailBui = pTailBui->nextBuilding)
                    if (pTailRoomSearch->buildingNum ==
pTailBui->buildingNum)

                        for(pTailRoom = pTailBui->rooms;pTailRoom;
                            pTailRoom = pTailRoom->nextRoom)
                                if (pTailRoom->roomNum ==
pTailRoomSearch->roomNum)

                                    {
                                        pRoomPaint = (Room*)malloc(sizeof(Room));
                                        *pRoomPaint = *pTailRoom;
                                        roomNow = pTailRoom->roomNum;
                                        break;
                                    }

                                EnableWindow(hwndButtonEdit, TRUE);
                                EnableWindow(hwndButtonDel, TRUE);

                                pRoomSearch = NULL;
                                iPaintState = PA_SECSTEPROOM;
                                InvalidateRect(hwnd, NULL, TRUE);
                                MoveWindow(hwndButtonEnter, 0, 0, 0, 0, TRUE);
                                break;

                        default:
                            break;
                    }

                iSearch = NOSEARCH;
                break;

                //处理 IDM_FILE 文件系列功能

                case IDM_FILE_SAVE:                //文件-保存数据*

                    if (szFileName[0])
                    {
                        if (FileWriteWnd(hwnd, szFileName))

```

---

```

        {
            iSaveState = TRUE;
            return 1;
        }
    else
    {
        wsprintf(szBuffer, TEXT("无法打开文件 %s"), szFileName);
        MessageBox(hwnd, szBuffer, TEXT("提示"), MB_OK);
        return 0;
    }
}

//如果szFileName不存在，则跳转到“另存为”

case IDM_FILE_SAVE_AS:        //文件-数据另存为*

    if (FileSaveDlg(hwnd, szFileName, szFileTitle))
    {
        SetTitle(hwnd, szFileTitle);

        if (FileWriteWnd(hwnd, szFileName))
        {
            iSaveState = TRUE;
            return 1;
        }
    else
    {
        wsprintf(szBuffer, TEXT("无法打开文件 %s"), szFileName);
        MessageBox(hwnd, szBuffer, TEXT("提示"), MB_OK);
        return 0;
    }
}

return 0;

case IDM_FILE_CREATE:        //文件-新建*

    if (!AskConfirm(hwnd, iSaveState))
        return 0;

    szFileName[0] = '\0';
    szFileTitle[0] = '\0';

    SetTitle(hwnd, szFileTitle);
    iSaveState = TRUE;
    return 0;

```

---

```

case IDM_FILE_IMPORT:           //文件-导入数据*

    //询问用户是否需要保存
    if (!AskConfirm(hwnd, iSaveState))
        return 0;

    if (FileImportDlg(hwnd, szFileName, szFileTitle))
        if (!FileReadWnd(hEdit, szFileName))
        {
            wsprintf(szBuffer, TEXT("无法打开文件 %s"),
                szFileTitle[0] ? szFileTitle : TEXT("无标题"));
            MessageBox(hwnd, szBuffer, TEXT("楼盘管理系统"),
                MB_OK | MB_ICONEXCLAMATION);

            szFileName[0] = '\0';
            szFileTitle[0] = '\0';
        }
    else
    {
        //初始化选项
        lstrcpy(comNow, pHead->name);
        if (pHead->buildings != NULL)
            buiNow = pHead->buildings->buildingNum;

        //确认将要绘制的成员
        if (pHead)
        {
            pComPaint = (Community*)malloc(sizeof(Community));
            *pComPaint = *pHead;
            if (pHead->buildings)
            {
                pBuiPaint = (Building*)malloc(sizeof(Building));
                *pBuiPaint = *(pHead->buildings);
            }
        }
        else return 0;

        //开始绘制
        iPaintState = PA_FIRSTEPCOM;
        InvalidateRect(hwnd, NULL, TRUE);
    }

```

---

```

        SetTitle(hwnd, szFileTitle);
        iSaveState = TRUE;
        return 0;

case IDM_FILE_EXIT:                //文件-退出程序*

        SendMessage(hwnd, WM_CLOSE, 0, 0);
        break;

        //处理 IDM_EDIT 编辑部分功能

case IDM_EDIT_CHANGE_DATA:        //编辑-更改-更改数据

        SendMessage(hwndButtonEdit, BM_CLICK, 0, 0);
        break;

case IDM_EDIT_ADD_ROOM:            //编辑-添加-添加新房间*

        DialogBox(hInstance, MAKEINTRESOURCE(IDD_ADD_ROOM),
            hwnd, AddRoomDlgProc);
        if (bRePaint)
        {
            //将这个房间结构放进链表^_^
            for (pTailCom = pHead; pTailCom;
                pTailCom = pTailCom->nextCommunity)
            {
                if (lstrcmp(prBuffer[0].inCom, pTailCom->name) == 0)
                {
                    prBuffer[0].communityNum = pTailCom->communityNum;
                    for (pTailBui = pTailCom->buildings; pTailBui;
                        pTailBui = pTailBui->nextBuilding)
                        if (prBuffer[0].buildingNum == pTailBui->buildingNum)
                        {
                            if (CheckSameRoom(prBuffer[0].inCom,
                                prBuffer[0].buildingNum, prBuffer[0].roomNum,
                                &pHead))
                                AddRoomToList(prBuffer[0], &pHead,
                                    prBuffer[0].communityNum,
                                    prBuffer[0].buildingNum);
                            else
                            {
                                MessageBox(hwnd, TEXT("该房间名已在系统中"),
                                    TEXT("警告"), MB_OK | MB_ICONWARNING);
                                break;

```

---

```

    }
    UpdateData(&pHead);

    //存储即将绘制的Bui地址
    pBuiPaint = (Building*)malloc(sizeof(Building));
    *pBuiPaint = *pTailBui;
    break;
}
}

//确定将要绘制的级别并无效窗口
iPaintState = PA_SECSTEPBUI;
InvalidateRect(hwnd, NULL, TRUE);

iSaveState = FALSE;
}
break;

case IDM_EDIT_ADD_BUID:           //编辑-添加-添加新楼栋*

    DialogBox(hInstance, MAKEINTRESOURCE(IDD_ADD_BUI),
        hwnd, AddBuiDlgProc);

    if (bRePaint)
    {
        //将这个楼栋结构放进链表^^
        for (pTailCom = pHead; pTailCom; pTailCom = pTailCom->nextCommunity)
            if (lstrcmp(pbBuffer[0].inCom, pTailCom->name) == 0)
            {
                if (CheckSameBui(pbBuffer[0].inCom, pbBuffer[0].buildingNum,
&pHead))
                    AddBuiToList(pbBuffer[0], &pHead,
pTailCom->communityNum);
            }
            else
            {
                MessageBox(hwnd, TEXT("该楼栋名已在系统中"),
                    TEXT("警告"), MB_OK | MB_ICONWARNING);
                break;
            }
        UpdateData(&pHead);

        //存储即将绘制的Com地址
        pComPaint = (Community*)malloc(sizeof(Community));

```

---

```

        *pComPaint = *pTailCom;
        buiNow = pbBuffer[0].buildingNum;
        break;
    }

    //确定将要绘制的级别并无效窗口
    iPaintState = PA_FIRSTEPCOM;
    InvalidateRect(hwnd, NULL, TRUE);

    iSaveState = FALSE;
}
break;

case IDM_EDIT_ADD_COM:    //编辑-添加-添加新楼盘*

    DialogBox(hInstance, MAKEINTRESOURCE(IDD_ADD_COM),
        hwnd, AddComDlgProc);
    if (bRePaint)
    {
        if (pcBuffer[0].name)
        {
            if (CheckSameCom(pcBuffer[0].name, &pHead))
                AddComToList(pcBuffer[0], &pHead);
            else
            {
                MessageBox(hwnd, TEXT("该楼盘名已在系统中"),
                    TEXT("警告"), MB_OK | MB_ICONWARNING);
                break;
            }
            UpdateData(&pHead);
            for (pTailCom = pHead; pTailCom; pTailCom =
pTailCom->nextCommunity)
                if (lstrcmp(pcBuffer[0].name, pTailCom->name) == 0)
                {
                    //存储即将绘制的Com地址
                    pComPaint = (Community*)malloc(sizeof(Community));
                    *pComPaint = *pTailCom;
                    lstrcpy(comNow, pcBuffer[0].name);
                    break;
                }
            }
        }

        //确定将要绘制的级别并无效窗口
        iPaintState = PA_FIRSTEPCOM;

```

---

```

        InvalidateRect(hwnd, NULL, TRUE);

        iSaveState = FALSE;
    }

    break;

case IDM_EDIT_DEL_ROOM:                //编辑-删除-删除房间

    DialogBox(hInstance, MAKEINTRESOURCE(IDD_DEL_ROOM),
        hwnd, DelRoomDlgProc);
    if (bRePaint)
    {
        for (pTailCom = pHead; pTailCom; pTailCom = pTailCom->nextCommunity)
            if (lstrcmp(comNow, pTailCom->name) == 0)
            {
                UpdateData(&pHead);

                //存储即将绘制的Com地址
                pComPaint = (Community*)malloc(sizeof(Community));
                *pComPaint = *pTailCom;
                for (pTailBui = pTailCom->buildings; pTailBui != NULL;
                    pTailBui = pTailBui->nextBuilding)

                    if (buiNow == pTailBui->buildingNum)
                    {
                        //存储将要绘制的Bui地址
                        pBuiPaint = (Building*)malloc(sizeof(Building));
                        *pBuiPaint = *pTailBui;
                    }
                break;
            }

        //确定将要绘制的级别并无效窗口
        iPaintState = PA_SECSTEPBUI;
        InvalidateRect(hwnd, NULL, TRUE);

        iSaveState = FALSE;
    }
    break;

case IDM_EDIT_DEL_BUID:                //编辑-删除-删除楼栋

    DialogBox(hInstance, MAKEINTRESOURCE(IDD_DEL_BUI),

```

---

```

        hwnd, DelBuiDlgProc);
if (bRePaint)
{
    for (pTailCom = pHead; pTailCom; pTailCom = pTailCom->nextCommunity)
        if (lstrcmp(comNow, pTailCom->name) == 0)
        {
            UpdateData(&pHead);

            //存储即将绘制的Com地址
            pComPaint = (Community*)malloc(sizeof(Community));
            *pComPaint = *pTailCom;
            if(pTailCom->buildings)
                buiNow = pTailCom->buildings->buildingNum;
            else buiNow = 0;
            break;
        }

    //确定将要绘制的级别并无效窗口
    iPaintState = PA_FIRSTEPCOM;
    InvalidateRect(hwnd, NULL, TRUE);

    iSaveState = FALSE;
}
break;

case IDM_EDIT_DEL_COM:        //编辑-删除-删除楼盘

    DialogBox(hInstance, MAKEINTRESOURCE(IDD_DEL_COM),
        hwnd, DelComDlgProc);
if (bRePaint)
{
    if (pHead)
    {
        lstrcpy(comNow, pHead->name);
        UpdateData(&pHead);

        //存储即将绘制的Com地址
        pComPaint = (Community*)malloc(sizeof(Community));
        *pComPaint = *pHead;
    }
    else pComPaint = NULL;

    //确定将要绘制的级别并无效窗口
    iPaintState = PA_FIRSTEPCOM;

```



---

```

        InvalidateRect(hwnd, NULL, TRUE);

        iSaveState = FALSE;
    }

    break;

    //处理 IDM_SEARCH 查询部分功能

case IDM_SEARCH:                //查询-信息检索

    DialogBox(hInstance, MAKEINTRESOURCE(IDD_SEARCH),
        hwnd, SearchDlgProc);
    switch (iSearch)
    {
    case SEARCHCOM:
        pComPaint = (Community*)malloc(sizeof(Community));
        if(pComPaint)
            *pComPaint = *pComSearch;
        break;

    case SEARCHBUI:
        pBuiPaint = (Building*)malloc(sizeof(Building));
        if(pBuiSearch)
            *pBuiPaint = *pBuiSearch;
        break;

    case SEARCHROOM:
        pRoomPaint = (Room*)malloc(sizeof(Room));
        if (pRoomSearch)
            *pRoomPaint = *pRoomSearch;
        break;

    default:
        break;
    }

    InvalidateRect(hwnd, NULL, TRUE);
    break;

    //处理 IDM_HELP 帮助部分功能

case IDM_HELP_ABOUT:            //关于软件

```

---

```

        DialogBox(hInstance, MAKEINTRESOURCE(IDD_ABOUTBOX),
            hwnd, AboutDlgProc);
        break;

case ID_MAINLIST:

    if (HIWORD(wParam) == LBN_SELCHANGE)
    {
        iIndex = SendMessage(hwndMainList, LB_GETCURSEL, 0, 0);

        //获取选中项文本
        SendMessage(hwndMainList, LB_GETTEXT, iIndex, (LPARAM)szBuffer);

        if (SEARCHING && !strcmp(szBuffer, TEXT("【退出搜索】")))
        {
            iSearch = NOSEARCH;
            iPaintState = PA_FIRSTEPCOM;

            EnableWindow(hwndButtonEdit, TRUE);
            EnableWindow(hwndButtonDel, TRUE);

            pComSearch = NULL;
            pBuiSearch = NULL;
            pRoomSearch = NULL;
            MoveWindow(hwndButtonEnter, 0, 0, 0, 0, TRUE);
            InvalidateRect(hwnd, NULL, TRUE);
            break;
        }

        if (iSearch == SEARCHBUI)
        {
            //获取选中项代表楼栋
            for (pTailBui = pBuiSearch, i = 0; pTailBui != NULL;
                pTailBui = pTailBui->nextBuilding, i++)
                if (i == iIndex)
                    break;

            pBuiPaint = (Building*)malloc(sizeof(Building));
            *pBuiPaint = *pTailBui;

            //遍历链表寻找所在楼盘
            for (pTailCom = pHead; pTailCom;
                pTailCom = pTailCom->nextCommunity)
                if (strcmp(pTailCom->name, pBuiPaint->inCom) == 0)

```

---

```

        {
            //存储即将绘制的Com地址
            pComPaint = (Community*)malloc(sizeof(Community));
            *pComPaint = *pTailCom;
        }

        InvalidateRect(hwnd, NULL, TRUE);
        break;
    }
    else if (iSearch == SEARCHROOM)
    {
        //获取选中项代表楼栋
        for (pTailRoom = pRoomSearch, i = 0; pTailRoom != NULL;
            pTailRoom = pTailRoom->nextRoom, i++)
            if (i == iIndex)
                break;

        pRoomPaint = (Room*)malloc(sizeof(Room));
        *pRoomPaint = *pTailRoom;

        //遍历链表寻找所在楼盘、楼栋
        for (pTailCom = pHead; pTailCom;
            pTailCom = pTailCom->nextCommunity)
            if (strcmp(pTailCom->name, pRoomPaint->inCom) == 0)
            {
                //存储即将绘制的Com地址
                pComPaint = (Community*)malloc(sizeof(Community));
                *pComPaint = *pTailCom;

                for (pTailBui = pTailCom->buildings; pTailBui;
                    pTailBui = pTailBui->nextBuilding)
                    if (pTailBui->buildingNum ==
pRoomPaint->buildingNum)
                        {
                            pBuiPaint =
(Building*)malloc(sizeof(Building));
                            *pBuiPaint = *pTailBui;
                        }
            }

        InvalidateRect(hwnd, NULL, TRUE);
        break;
    }

```

---

```

        //储存所在楼盘名
        lstrcpy(comNow, szBuffer);

        //遍历链表寻找该楼盘
        for (pTailCom = pHead; pTailCom;
            pTailCom = pTailCom->nextCommunity)
            if (lstrcmp(pTailCom->name, szBuffer) == 0)
            {
                //存储即将绘制的Com地址
                pComPaint = (Community*)malloc(sizeof(Community));
                *pComPaint = *pTailCom;
            }

        //确定将要绘制的级别并无效窗口
        iPaintState = PA_FIRSTEPCOM;
        InvalidateRect(hwnd, NULL, TRUE);
    }
    break;
}

break;

case WM_NOTIFY:

    switch (wParam)
    {
        case ID_SUBLIST:

            //【双击】二级菜单，显示下一级菜单

            if (((NMHDR *)lParam)->code) == NM_DBLCLK)
            {
                //获取选中项文本
                NM_LISTVIEW *pNMListView = (NM_LISTVIEW *)lParam;
                iIndex = pNMListView->iItem;

                ListView_GetItemText(hwndSubList, iIndex, 0, szBuffer, 4096);

                //当次级列表中为楼栋时，双击进入下一级

                if (iPaintState == PA_FIRSTEBUI
                    || iPaintState == PA_FIRSTEPCOM)
                {
                    //判断双击处是否有数据

```

---

```

        if (iIndex<0 || iIndex>ListView_GetItemCount(hwndSubList))
            break;

        //获取前三位楼栋号
        szBuffer[3] = '\0\0';
        iBuiNum = _ttoi(szBuffer);

        buiNow = iBuiNum;

        //遍历链表寻找所在楼盘
        for (pTailCom = pHead; pTailCom;
            pTailCom = pTailCom->nextCommunity)
            if (lstrcmp(pTailCom->name, comNow) == 0)
            {
                //存储即将绘制的Com地址
                pComPaint = (Community*)malloc(sizeof(Community));
                *pComPaint = *pTailCom;

                //遍历链表寻找该楼栋
                for (pTailBui = pTailCom->buildings; pTailBui;
                    pTailBui = pTailBui->nextBuilding)
                    if (pTailBui->buildingNum == buiNow)
                    {
                        //存储即将绘制的Bui地址
                        pBuiPaint =
(Building*)malloc(sizeof(Building));

                        *pBuiPaint = *pTailBui;
                        buiNow = pBuiPaint->buildingNum;
                        break;
                    }
                break; //跳出循环
            }

        //绘制下一级
        iPaintState = PA_SECSTEPBUI;
        InvalidateRect(hwnd, NULL, TRUE);
    }

    //当前次级列表中为房间时，仅响应双击第一栏返回上一级的双击消息

    else if (iPaintState == PA_SECSTEPBUI
        || iPaintState == PA_SECSTEPROOM)
    {
        //判断双击处是否为第一项

```

---

```

        if (iIndex != 0)
            break;

//遍历链表寻找所在楼盘
for (pTailCom = pHead; pTailCom;
    pTailCom = pTailCom->nextCommunity)
    if (lstrcmp(pTailCom->name, comNow) == 0)
    {
        //存储即将绘制的Com地址
        pComPaint = (Community*)malloc(sizeof(Community));
        *pComPaint = *pTailCom;

        //遍历链表寻找该楼栋
        for (pTailBui = pTailCom->buildings; pTailBui;
            pTailBui = pTailBui->nextBuilding)
            if (pTailBui->buildingNum == buiNow)
            {
                //存储即将绘制的Bui地址
                pBuiPaint =
(Building*)malloc(sizeof(Building));

                *pBuiPaint = *pTailBui;
                buiNow = pBuiPaint->buildingNum;
                break;
            }
        break; //跳出循环
    }

//绘制上一级
iPaintState = PA_FIRSTEBUI;
InvalidateRect(hwnd, NULL, TRUE);
}
}

//单击二级菜单，改变显示数据

else if (((NMHDR *)lParam)->code) == NM_CLICK)
{
    //获取选中项文本
    NM_LISTVIEW *pNMListView = (NM_LISTVIEW *)lParam;
    iIndex = pNMListView->iItem;

    //判断单击处是否有数据
    if (iIndex<0 || iIndex>ListView_GetItemCount(hwndSubList))
        break;
}

```

---

```

ListView_GetItemText(hwndSubList, iIndex, 0, szBuffer, 4096);

//二级列表框为building的情况

if (iPaintState == PA_FIRSTSTEPBUI
    || iPaintState == PA_FIRSTSTEPCOM)
{
    //获取前三位楼栋号
    szBuffer[3] = '\0\0';
    iBuiNum = _ttoi(szBuffer);

    //遍历链表寻找所在楼盘
    for (pTailCom = pHead; pTailCom;
        pTailCom = pTailCom->nextCommunity)
        if (lstrcmp(pTailCom->name, comNow) == 0)
        {
            //存储即将绘制的Com地址
            pComPaint = (Community*)malloc(sizeof(Community));
            *pComPaint = *pTailCom;

            //遍历链表寻找该楼栋
            for (pTailBui = pTailCom->buildings; pTailBui;
                pTailBui = pTailBui->nextBuilding)
                if (pTailBui->buildingNum == iBuiNum)
                {
                    //存储即将绘制的Bui地址
                    pBuiPaint =
(Building*)malloc(sizeof(Building));

                    *pBuiPaint = *pTailBui;
                    buiNow = pBuiPaint->buildingNum;
                    break;
                }
            break; //跳出循环
        }

    //确定将要绘制的级别并无效窗口
    iPaintState = PA_FIRSTSTEPBUI;
    InvalidateRect(hwnd, NULL, TRUE);
}

//二级列表框为Room的情况

else if (iPaintState == PA_SECSTEPBUI

```

---

```

        || iPaintState == PA_SECSTEPROOM)
    {

        //若单击楼栋，则显示楼栋信息

        if (iIndex == 0)
        {
            iPaintState = PA_SECSTEPBUI;
            InvalidateRect(hwnd, NULL, TRUE);
            break;
        }

        //获取前三位房间号
        szBuffer[3] = '\0\0';
        iRoomNum = _ttoi(szBuffer);

        //遍历链表寻找所在楼盘
        for (pTailCom = pHead; pTailCom;
            pTailCom = pTailCom->nextCommunity)
            if (lstrcmp(pTailCom->name, comNow) == 0)
            {
                //存储即将绘制的Com地址
                pComPaint = (Community*)malloc(sizeof(Community));
                *pComPaint = *pTailCom;

                //遍历链表寻找该楼栋
                for (pTailBui = pTailCom->buildings; pTailBui;
                    pTailBui = pTailBui->nextBuilding)
                    if (pTailBui->buildingNum == buiNow)
                    {
                        //存储即将绘制的Bui地址
                        pBuiPaint =
                            (Building*)malloc(sizeof(Building));

                        *pBuiPaint = *pTailBui;

                        for (pTailRoom = pTailBui->rooms; pTailRoom;
                            pTailRoom = pTailRoom->nextRoom)
                        {
                            if (pTailRoom->roomNum == iRoomNum)
                            {
                                //存储即将绘制的Room地址
                                pRoomPaint =
                                    (Room*)malloc(sizeof(Room));

                                *pRoomPaint = *pTailRoom;

```



---

```

        }
    }

    }

    break;    //跳出循环
}

//确定将要绘制的级别并无效窗口
iPaintState = PA_SECSTEPROOM;
InvalidateRect(hwnd, NULL, TRUE);
}

}

default:
    break;
}

return 0;

case WM_PAINT:

    hdc = BeginPaint(hwnd, &ps);

    SelectObject(hdc, hFontBlack);

    //画出基本框架
    rectSub = DrawBasicBk(hwnd, hdc, cxClient, cyClient);
    UpdateData(&pHead);

    if (iSearch == NOSEARCH)
    {
        if (iPaintState == PA_FIRSTEPCOM)
        {
            Draw1Step(hdc, pComPaint);
            RenewData(hdc, rectData);
            if (pComPaint)
                DrawComInf(hwnd, hdc, *pComPaint);
        }
        else if (iPaintState == PA_SECSTEPBUI)
        {
            Draw2Step(hdc, pBuiPaint);
            RenewData(hdc, rectData);
            if (pBuiPaint)
                DrawBuiInf(hwnd, hdc, *pBuiPaint);
        }
    }
}

```

---

```

    }
    else if (iPaintState == PA_FIRSTSTEPBUI)
    {
        Draw1Step(hdc, pComPaint);
        RenewData(hdc, rectData);
        if (pBuiPaint)
            DrawBuiInf(hwnd, hdc, *pBuiPaint);
    }
    else if (iPaintState == PA_SECSTEPROOM)
    {
        Draw2Step(hdc, pBuiPaint);
        RenewData(hdc, rectData);
        if (pRoomPaint)
            DrawRoomInf(hwnd, hdc, *pRoomPaint);
    }
}

else if (SEARCHING)
{
    //重置信息
    SendMessage(hwndMainList, LB_RESETCONTENT, 0, 0);
    ListView_DeleteAllItems(hwndSubList);

    SetWindowText(hwndMainTitle, TEXT("搜索结果"));

    switch (iSearch)
    {
    case SEARCHCOM:

        //向搜索结果栏中添加信息
        for (pTailCom = pComSearch; pTailCom;
            pTailCom = pTailCom->nextCommunity)
        {
            SendMessage(hwndMainList, LB_ADDSTRING,
                0, (LPARAM)pTailCom->name);
        }
        SendMessage(hwndMainList, LB_ADDSTRING,
            0, (LPARAM)TEXT("【退出搜索】"));

        if (pComPaint)
        {
            DrawComInf(hwnd, hdc, *pComPaint);
            //iIndex = 0;
        }
        break;
    }
}

```

---

```
case SEARCHBUI:

    //向搜索结果栏中添加信息
    for (pTailBui = pBuiSearch; pTailBui;
        pTailBui = pTailBui->nextBuilding)
    {
        wsprintf(szBuffer, TEXT("%s%d栋"),
            pTailBui->inCom, pTailBui->buildingNum);
        SendMessage(hwndMainList, LB_ADDSTRING,
            0, (LPARAM)szBuffer);
    }
    SendMessage(hwndMainList, LB_ADDSTRING,
        0, (LPARAM)TEXT("【退出搜索】"));

    if (pBuiPaint)
    {
        DrawBuiInf(hwnd, hdc, *pBuiPaint);
        //iIndex = 0;
    }
    break;

case SEARCHROOM:

    //向搜索结果栏中添加信息
    for (pTailRoom = pRoomSearch; pTailRoom;
        pTailRoom = pTailRoom->nextRoom)
    {
        wsprintf(szBuffer, TEXT("%s%d栋%d"),
            pTailRoom->inCom, pTailRoom->buildingNum,
            pTailRoom->roomNum);
        SendMessage(hwndMainList, LB_ADDSTRING,
            0, (LPARAM)szBuffer);
    }
    SendMessage(hwndMainList, LB_ADDSTRING,
        0, (LPARAM)TEXT("【退出搜索】"));

    if (pRoomPaint)
    {
        DrawRoomInf(hwnd, hdc, *pRoomPaint);
        //iIndex = 0;
    }
    break;
```

---

```

        default:
            break;
    }

    EnableWindow(hwndButtonEdit, FALSE);
    EnableWindow(hwndButtonDel, FALSE);
}

EndPaint(hwnd, &ps);

return 0;

case WM_CLOSE:
    if (AskConfirm(hwnd, iSaveState))
        DestroyWindow(hwnd);
    return 0;

case WM_QUERYENDSESSION:
    if (AskConfirm(hwnd, iSaveState))
        return 1;
    else
        return 0;

case WM_DESTROY:
    PostQuitMessage(0);
    return 0;

default:
    break;
}

return DefWindowProc(hwnd, message, wParam, lParam);
}

```

```

////////////////////////////////////
// “关于‘楼盘管理系统’”（About）对话框窗口过程

```

```

BOOL CALLBACK AboutDlgProc(HWND hDlg, UINT message,
    WPARAM wParam, LPARAM lParam)
{
    switch (message)
    {
        case WM_INITDIALOG:

```

---

```

        return TRUE;

    case WM_COMMAND:
        switch (LOWORD(wParam))
        {
            case IDOK:
            case IDCANCEL:
                EndDialog(hDlg, 0);
                return TRUE;
        }

        break;
    }

    return FALSE;
}

////////////////////////////////////
//AskConfirm
//功能：当用户退出系统时，若文件未保存，则提示其是否需要保存
//      若文件已经保存，则直接退出
//参数：退出的窗口句柄，保存状态（非0为已保存，0为未保存）
//返回：若需要退出系统，则返回TRUE；若不需要，则返回FALSE

BOOL AskConfirm(HWND hwnd, int iSaveState)
{
    int iAns;
    if (!iSaveState)
    {
        iAns = MessageBox(hwnd, TEXT("是否要保存您的更改"), TEXT("提示"),
MB_YESNOCANCEL);
        switch (iAns)
        {

            case IDYES:
                SendMessage(hwnd, WM_COMMAND, (WPARAM) IDM_FILE_SAVE, 0);
                return TRUE;

            case IDNO:
                return TRUE;

            case IDCANCEL:
                return FALSE;
        }
    }
}

```

---

```

    }
}
else
    return TRUE;
}

```

```

////////////////////////////////////

```

```

//DrawBasicBk

```

```

//窗口基本布局构造函数

```

```

//功能：画出窗口的基本布局，不含任何信息

```

```

//参数：窗口句柄, 设备环境, 屏幕宽，屏幕高

```

```

//返回：数据部分的RECT矩形

```

```

RECT DrawBasicBk(HWND hwnd, HDC hdc, int cxClient, int cyClient)

```

```

{

```

```

    RECT rectClass, rectData, rectSub;

```

```

    rectClass.left = 0;

```

```

    rectClass.top = 0;

```

```

    rectClass.right = XLNEPOS;

```

```

    rectClass.bottom = cyClient;

```

```

    Rectangle(hdc, rectClass.left, rectClass.top,
        rectClass.right, rectClass.bottom);

```

```

    rectSub.left = XLNEPOS;

```

```

    rectSub.top = YLINEPOS;

```

```

    rectSub.right = cxClient;

```

```

    rectSub.bottom = cyClient;

```

```

    Rectangle(hdc, rectSub.left, rectSub.top,
        rectSub.right, rectSub.bottom);

```

```

    rectData.left = XLNEPOS;

```

```

    rectData.top = 0;

```

```

    rectData.right = cxClient;

```

```

    rectData.bottom = YLINEPOS;

```

```

    SelectObject(hdc, GetStockObject(DC_BRUSH));

```

```

    SetDCBrushColor(hdc, RGB(137, 189, 255)); //该颜色为天蓝色

```

```

    Rectangle(hdc, rectData.left, rectData.top,

```

---

```

        rectData.right, rectData.bottom);

    SelectObject(hdc, GetStockObject(WHITE_BRUSH));

    return rectData;
}

////////////////////////////////////
//create3DepthList
//功能: 创建一个三级链表
//参数: 1、第一级个数;
//      2、每个第一级所含第二级个数
//      3、每个第二级所含第三级个数
//函数返回: 返回该三级链表的头指针

Community *create3DepthList(int len1, int len2, int len3)
{
    Community *head = createComList(len1);
    Community *comTail;
    Building *buiTail;
    for (comTail = head->nextCommunity; comTail; comTail = comTail->nextCommunity)
    {
        comTail->buildings = createBuiList(len2);
        for (buiTail = comTail->buildings->nextBuilding; buiTail;
buiTail->nextBuilding)
            buiTail->rooms = createRoomList(len3);
    }
    return head;
}

////////////////////////////////////
//createComList
//功能: 创建一个room的先进先出的单向链表
//参数: 链表长度
//返回: 该链表头指针

Community *createComList(int len)
{
    Community *head, *tail;
    tail = head = (Community*)malloc(sizeof(Community));
    while (len-- > 0)
    {

```

---

```

        tail->nextCommunity = (Community*)malloc(sizeof(Community));
        tail = tail->nextCommunity;
    }
    tail->nextCommunity = NULL;
    return head;
}

```

```

////////////////////////////////////
//createBuiList
//功能：创建一个building的先进先出的单向链表
//参数：链表长度
//返回：该链表头指针

```

```

Building *createBuiList(int len)
{
    Building *head, *tail;
    tail = head = (Building*)malloc(sizeof(Building));
    while (len-- > 0)
    {
        tail->nextBuilding = (Building*)malloc(sizeof(Building));
        tail = tail->nextBuilding;
    }
    tail->nextBuilding = NULL;
    return head;
}

```

```

////////////////////////////////////
//createRoomList
//功能：创建一个room的先进先出的单向链表
//参数：链表长度
//返回：该链表头指针

```

```

Room *createRoomList(int len)
{
    Room *head, *tail;
    tail = head = (Room*)malloc(sizeof(Room));
    while (len-- > 0)
    {
        tail->nextRoom = (Room*)malloc(sizeof(Room));
        tail = tail->nextRoom;
    }
    tail->nextRoom = NULL;
}

```



---

```

        return head;
    }

////////////////////////////////////
// “添加新楼盘”（AddCommunity）对话框窗口过程

BOOL CALLBACK AddComDlgProc(HWND hDlg, UINT message,
    WPARAM wParam, LPARAM lParam)
{
    TCHAR szBuffer[100][100];
    Community *pNewCom;
    int i;
    int iLength[5];

    switch (message)
    {
    case WM_INITDIALOG:
        return TRUE;

    case WM_COMMAND:
        switch (LOWORD(wParam))
        {
        case IDOK:

            //当按下OK时，将所输入内容存进存储区

            iLength[0] = SendMessage(GetDlgItem(hDlg, IDC_ADD_COM_NAME),
                EM_GETLINE, 0, (LPARAM)szBuffer[0]);
            iLength[1] = SendMessage(GetDlgItem(hDlg, IDC_ADD_COM_ADDR),
                EM_GETLINE, 2, (LPARAM)szBuffer[1]);
            iLength[2] = SendMessage(GetDlgItem(hDlg, IDC_ADD_COM_PHONE),
                EM_GETLINE, 3, (LPARAM)szBuffer[2]);
            iLength[3] = SendMessage(GetDlgItem(hDlg, IDC_ADD_COM_PERSON),
                EM_GETLINE, 4, (LPARAM)szBuffer[3]);

            //检验每一项非空

            for (i = 0; i < 4; i++)
            {
                if (!iLength[i])
                {
                    MessageBox(hDlg, TEXT("请输入完整的信息！"), TEXT("提示"),
MB_OK);

```

---

```

        return FALSE;
    }
    else {
        *(szBuffer[i] + iLength[i]) = '\0\0';
    }
}

//分配空间，并将数据放入一个community结构体

pNewCom = (Community*)malloc(sizeof(Community));

lstrcpy(pNewCom->name, szBuffer[0]);
lstrcpy(pNewCom->address, szBuffer[1]);
lstrcpy(pNewCom->phone, szBuffer[2]);
lstrcpy(pNewCom->host.name, szBuffer[3]);

pNewCom->communityNum = iAutoComNum++;
pNewCom->avgPrice = 0.0;

//将这个楼盘存进全局缓冲区，之后加入链表
pcBuffer[0] = *pNewCom;

bRePaint = TRUE;
EndDialog(hDlg, TRUE);
return TRUE;

case IDCANCEL:
    bRePaint = FALSE;
    EndDialog(hDlg, FALSE);
    return TRUE;
}

break;
}

return FALSE;
}

```

```

////////////////////////////////////
// “添加新楼栋”（AddBuilding）对话框窗口过程

```

```

BOOL CALLBACK AddBuiDlgProc(HWND hDlg, UINT message,
    WPARAM wParam, LPARAM lParam)

```

---

```

{
    TCHAR szBuffer[100][100];
    Building *pNewBui;
    Community *ptailCom, *comTail;
    int i;
    int iLength[5];

    fAddBuiState = FAIL;

    switch (message)
    {
    case WM_INITDIALOG:

        //设置编辑框初始内容
        comTail = pHead;
        while (comTail != NULL)
        {
            if (!lstrcmp(comTail->name, comNow))
            {
                SetWindowText(GetDlgItem(hDlg, IDC_ADD_BUI_NAME), comTail->name);
                SetWindowText(GetDlgItem(hDlg, IDC_ADD_BUI_PHONE), comTail->phone);
                SetWindowText(GetDlgItem(hDlg, IDC_ADD_BUI_PERSON),
comTail->host.name);
                break;
            }
            comTail = comTail->nextCommunity;
        }

        return TRUE;

    case WM_COMMAND:
        switch (LOWORD(wParam))
        {
        case IDOK:

            //当按下OK时，将所输入内容存进存储区

            iLength[0] = SendMessage(GetDlgItem(hDlg, IDC_ADD_BUI_NAME),
                EM_GETLINE, 0, (LPARAM)szBuffer[0]);
            iLength[1] = SendMessage(GetDlgItem(hDlg, IDC_ADD_BUI_NUM),
                EM_GETLINE, 1, (LPARAM)szBuffer[1]);
            iLength[2] = SendMessage(GetDlgItem(hDlg, IDC_ADD_BUI_FLOORS),
                EM_GETLINE, 2, (LPARAM)szBuffer[2]);
            iLength[3] = SendMessage(GetDlgItem(hDlg, IDC_ADD_BUI_PHONE),

```

---

```

        EM_GETLINE, 3, (LPARAM)szBuffer[3]);
iLength[4] = SendMessage(GetDlgItem(hDlg, IDC_ADD_BUI_PERSON),
        EM_GETLINE, 4, (LPARAM)szBuffer[4]);

//检验每一项非空

for (i = 0; i < 5; i++)
{
    if (!iLength[i])
    {
        MessageBox(hDlg, TEXT("请输入完整的信息!"), TEXT("提示"),
MB_OK);

        return FALSE;
    }
    else {
        *(szBuffer[i] + iLength[i]) = '\0\0';
    }
}

//分配空间，并将数据放入一个building结构体

pNewBui = (Building*)malloc(sizeof(Building));

lstrcpy(pNewBui->inCom, szBuffer[0]);
pNewBui->buildingNum = _ttoi(szBuffer[1]);
pNewBui->numberOfFloors = _ttoi(szBuffer[2]);
lstrcpy(pNewBui->host.phone, szBuffer[3]);
lstrcpy(pNewBui->host.name, szBuffer[4]);

pNewBui->avgPrice = 0.0;
pNewBui->numberOfRooms = 0;

//确认是否存在这个地址
for (ptailCom = pHead; ptailCom; ptailCom = ptailCom->nextCommunity)
    if (lstrcmp(pNewBui->inCom, ptailCom->name) == 0)
        fAddBuiState = SUCCEED;

//检验是否有对应楼盘
if (fAddBuiState == FAIL)
{
    MessageBox(hDlg, TEXT("所输入的楼盘不存在"), TEXT("提示"), MB_OK);
    break;
}

```

---

```

        //将这个结构存入楼盘全局缓冲区
        pbBuffer[0] = *pNewBui;

        bRePaint = TRUE; //授权更新界面
        EndDialog(hDlg, TRUE);
        return TRUE;

    case IDCANCEL:
        bRePaint = FALSE; //禁止更新界面
        EndDialog(hDlg, FALSE);
        return TRUE;
    }

    break;
}

return FALSE;
}

////////////////////////////////////
// “添加新房间”（AddRoom）对话框窗口过程

BOOL CALLBACK AddRoomDlgProc(HWND hDlg, UINT message,
    WPARAM wParam, LPARAM lParam)
{
    TCHAR szBuffer[100][100];
    Room *pNewRoom;
    Community *ptailCom, *comTail;
    Building *ptailBui, *buiTail;
    int i, iSold;
    int iLength[7];

    fAddRoomState = FAIL;

    switch (message)
    {
    case WM_INITDIALOG:

        //向ComboBox框中添加数据
        for (i = 0; i < 34; i++)
            SendMessage(GetDlgItem(hDlg, IDC_ADD_ROOM_STYLE),
                CB_ADDSTRING, 0, (LPARAM)szComboBoxData[i]);
    }
}

```

---

```

//设置编辑框初始内容
comTail = pHead;
while (comTail != NULL)
{
    if (!strcmp(comTail->name, comNow))
    {
        buiTail = comTail->buildings;
        while (buiTail != NULL)
        {
            if (buiTail->buildingNum == buiNow)
            {
                SetWindowText(GetDlgItem(hDlg, IDC_ADD_ROOM_COMNAME),
buiTail->inCom);

                sprintf(szBuffer[1], TEXT("%d"), buiTail->buildingNum);
                SetWindowText(GetDlgItem(hDlg, IDC_ADD_ROOM_BUINUM),
szBuffer[1]);

                break;
            }
            buiTail = buiTail->nextBuilding;
        }
        break;
    }
    comTail = comTail->nextCommunity;
}

return TRUE;

case WM_COMMAND:
    switch (LOWORD(wParam))
    {
        case IDOK:

            //当按下OK时，将所输入内容存进存储区

            iLength[0] = SendMessage(GetDlgItem(hDlg, IDC_ADD_ROOM_COMNAME),
                EM_GETLINE, 0, (LPARAM)szBuffer[0]);
            iLength[1] = SendMessage(GetDlgItem(hDlg, IDC_ADD_ROOM_BUINUM),
                EM_GETLINE, 1, (LPARAM)szBuffer[1]);
            iLength[2] = SendMessage(GetDlgItem(hDlg, IDC_ADD_ROOM_ROOMNUM),
                EM_GETLINE, 2, (LPARAM)szBuffer[2]);
            iLength[3] = SendMessage(GetDlgItem(hDlg, IDC_ADD_ROOM_SIZE),
                EM_GETLINE, 3, (LPARAM)szBuffer[3]);
            iLength[4] = SendMessage(GetDlgItem(hDlg, IDC_ADD_ROOM_PRICE),
                EM_GETLINE, 4, (LPARAM)szBuffer[4]);
            GetWindowText(GetDlgItem(hDlg, IDC_ADD_ROOM_STYLE),

```

---

```

        szBuffer[5], lstrlen(szBuffer[5]));
iSold = SendMessage(GetDlgItem(hDlg, IDC_ADD_ROOM_SOLD),
        BM_GETCHECK, 0, 0);

//检验每一项非空

for (i = 0; i < 5; i++)
{
    if (!iLength[i])
    {
        MessageBox(hDlg, TEXT("请输入完整的信息!"), TEXT("提示"),
MB_OK);

        return FALSE;
    }
    else {
        *(szBuffer[i] + iLength[i]) = '\0\0';
    }
}

//分配空间，并将数据放入一个Room结构体

pNewRoom = (Room*)malloc(sizeof(Room));

lstrcpy(pNewRoom->inCom, szBuffer[0]);
pNewRoom->buildingNum = _ttoi(szBuffer[1]);
pNewRoom->roomNum = _ttoi(szBuffer[2]);
pNewRoom->roomSize = _ttoi(szBuffer[3]);
pNewRoom->roomPrice = _ttoi(szBuffer[4]);
lstrcpy(pNewRoom->roomType, szBuffer[5]);
pNewRoom->roomState = iSold;

//确定是否存在这个地址
for (ptailCom = pHead; ptailCom;
    ptailCom = ptailCom->nextCommunity)
{
    if (lstrcmp(pNewRoom->inCom, ptailCom->name) == 0)
    {
        for (ptailBui = ptailCom->buildings; ptailBui;
            ptailBui = ptailBui->nextBuilding)
            if (pNewRoom->buildingNum == ptailBui->buildingNum)
                fAddRoomState = SUCCEED;
    }
}

```

---

```

        //检验是否有对应楼盘和楼栋
        if (fAddRoomState == FAIL)
        {
            MessageBox(hDlg, TEXT("所输入的楼盘或楼栋不存在"), TEXT("提示"),
MB_OK);

            break;
        }

        //将这个结构存入楼盘全局缓冲区
        prBuffer[0] = *pNewRoom;

        bRePaint = TRUE; //授权更新界面
        EndDialog(hDlg, TRUE);
        return TRUE;

    case IDCANCEL:
        bRePaint = FALSE; //禁止更新界面
        EndDialog(hDlg, FALSE);
        return TRUE;
    }

    break;
}

return FALSE;
}

```

```

////////////////////////////////////
//AddComToList
//功能：向链表中添加一个community节点
//参数：将要添加的Community结构体（添加对象链表的头指针.全局变量提供）
//返回：返回更新后链表头指针

```

```

BOOL AddComToList(Community newCom, Community **head)
{
    Community *begin, *end;

    //没有楼盘的情况
    if (*head == NULL)
    {
        *head = (Community*)malloc(sizeof(Community));
        **head = newCom;
    }
}

```



---

```

        //初始化链表指针，重要！
        (**head).nextCommunity = NULL;
        (**head).buildings = NULL;
        return TRUE;
    }

    end = *head, begin = NULL;
    while (end != NULL)
    {
        begin = end;
        end = end->nextCommunity;
    }

    begin->nextCommunity =
        (Community*)malloc(sizeof(Community));
    *begin->nextCommunity = newCom;

    //初始化链表指针，重要！！
    (*begin->nextCommunity).nextCommunity = NULL;
    (*begin->nextCommunity).buildings = NULL;
    return TRUE;
}

////////////////////////////////////
//AddBuiToList
//功能：向链表中添加一个building节点
//参数：将要添加的Building结构体，添加对象链表的头指针，所属楼盘号
//返回：成功则返回TRUE，失败返回FALSE

BOOL AddBuiToList(Building newBui, Community **head, int comNum)
{
    Building *begBui, *endBui;
    Community *tailCom;

    if (*head == NULL)
        return FALSE;

    for (tailCom = *head; tailCom != NULL
        ; tailCom = tailCom->nextCommunity)
    {
        if (tailCom->communityNum == comNum)
        {

            //该楼盘下没有楼栋的情况

```

---

```

        if (tailCom->buildings == NULL)
        {
            tailCom->buildings =
                (Building*)malloc(sizeof(Building));
            *tailCom->buildings = newBui;

            //对于未初始化的结构，这一步初始化很重要！！
            tailCom->buildings->nextBuilding = NULL;
            tailCom->buildings->rooms = NULL;
            return TRUE;
        }

        endBui = tailCom->buildings;
        begBui = NULL;
        while (endBui != NULL)
        {
            begBui = endBui;
            endBui = endBui->nextBuilding;
        }

        begBui->nextBuilding =
            (Building*)malloc(sizeof(Building));
        *begBui->nextBuilding = newBui;

        //初始化链表指针
        begBui->nextBuilding->nextBuilding = NULL;
        begBui->nextBuilding->rooms = NULL;
        return TRUE;
    }
}

return FALSE;
}

////////////////////////////////////
//AddRoomToList
//功能：向链表中添加一个Room节点
//参数：将要添加的Room结构体，添加对象链表的头指针，所属楼盘号，所属楼栋号
//返回：成功则返回TRUE，失败返回FALSE

BOOL AddRoomToList(Room newRoom, Community **head,
    int comNum, int buiNum)
{

```

---

```
Room *begRoom, *endRoom;
Community *tailCom;
Building *tailBui;

if (*head == NULL)
    return FALSE;

for (tailCom = *head; tailCom != NULL;
    tailCom = tailCom->nextCommunity)
{
    if (tailCom->communityNum == comNum)
    {
        //该楼盘下没有楼栋的情况
        if (tailCom->buildings == NULL)
            return FALSE;

        for (tailBui = tailCom->buildings; tailBui != NULL;
            tailBui = tailBui->nextBuilding)
        {
            if (tailBui->buildingNum == buiNum)
            {
                //该楼栋下没有房间的情况
                if (tailBui->rooms == NULL)
                {
                    tailBui->rooms =
                        (Room*)malloc(sizeof(Room));
                    *tailBui->rooms = newRoom;

                    //对于未初始化的结构，这一步初始化很重要！！
                    tailBui->rooms->nextRoom = NULL;
                    return TRUE;
                }

                endRoom = tailBui->rooms;
                begRoom = NULL;
                while (endRoom != NULL)
                {
                    begRoom = endRoom;
                    endRoom = endRoom->nextRoom;
                }

                begRoom->nextRoom =
                    (Room*)malloc(sizeof(Room));
                *begRoom->nextRoom = newRoom;
```

---

```

        //初始化链表指针
        begRoom->nextRoom->nextRoom = NULL;
        return TRUE;
    }
}

return FALSE;
}

```

```

////////////////////////////////////

```

```

//AddBuiToSearch

```

```

//功能：创建一个只由bui组成的search

```

```

//参数：将要添加的Building结构体，添加对象链表的头指针

```

```

//返回：成功则返回TRUE，失败返回FALSE

```

```

BOOL AddBuiToSearch(Building newCom, Building **head)

```

```

{

```

```

    Building *begin, *end;

```

```

    //没有楼盘的情况

```

```

    if (*head == NULL)

```

```

    {

```

```

        *head = (Building*)malloc(sizeof(Building));

```

```

        **head = newCom;

```

```

        //初始化链表指针，重要！

```

```

        (**head).nextBuilding = NULL;

```

```

        (**head).rooms = NULL;

```

```

        return TRUE;

```

```

    }

```

```

    end = *head, begin = NULL;

```

```

    while (end != NULL)

```

```

    {

```

```

        begin = end;

```

```

        end = end->nextBuilding;

```

```

    }

```

```

    begin->nextBuilding =

```

```

        (Building*)malloc(sizeof(Building));

```

---

```

    *begin->nextBuilding = newCom;

    //初始化链表指针，重要！！
    (*begin->nextBuilding).nextBuilding = NULL;
    (*begin->nextBuilding).rooms = NULL;
    return TRUE;
}

////////////////////////////////////
//AddRoomToSearch
//功能：创建一个只由bui组成的search
//参数：将要添加的Room结构体，添加对象链表的头指针
//返回：成功则返回TRUE，失败返回FALSE

BOOL AddRoomToSearch(Room newCom, Room **head)
{
    Room *begin, *end;

    //没有楼盘的情况
    if (*head == NULL)
    {
        *head = (Room*)malloc(sizeof(Room));
        **head = newCom;

        //初始化链表指针，重要！
        (**head).nextRoom = NULL;
        return TRUE;
    }

    end = *head, begin = NULL;
    while (end != NULL)
    {
        begin = end;
        end = end->nextRoom;
    }

    begin->nextRoom =
        (Room*)malloc(sizeof(Room));
    *begin->nextRoom = newCom;

    //初始化链表指针，重要！！
    (*begin->nextRoom).nextRoom = NULL;
    return TRUE;
}

```

---

```
}
```

```
////////////////////////////////////
```

```
//FileInitWnd
```

```
//功能：初始化弹出文件窗口的数据
```

```
//参数：父窗口句柄hwnd
```

```
//返回：无返回值
```

```
void FileInitWnd(HWND hwnd)
```

```
{
```

```
    static TCHAR szFilter[] =
```

```
        TEXT("Leading Files (*.led)\0*.led\0") \
```

```
        TEXT("All Files (*.*)\0*.*\0\0");
```

```
    ofn.lStructSize = sizeof(OPENFILENAME);
```

```
    ofn.hwndOwner = hwnd;
```

```
    ofn.hInstance = NULL;
```

```
    ofn.lpstrFilter = szFilter;
```

```
    ofn.lpstrCustomFilter = NULL;
```

```
    ofn.nMaxCustFilter = 0;
```

```
    ofn.nFilterIndex = 0;
```

```
    ofn.lpstrFile = NULL; //在打开、关闭文件中设置
```

```
    ofn.nMaxFile = MAX_PATH;
```

```
    ofn.lpstrFileName = NULL; //在打开、关闭文件中设置
```

```
    ofn.nMaxFileName = MAX_PATH;
```

```
    ofn.lpstrInitialDir = NULL;
```

```
    ofn.lpstrTitle = NULL;
```

```
    ofn.Flags = 0; //在打开、关闭文件中设置
```

```
    ofn.nFileOffset = 0;
```

```
    ofn.nFileExtension = 0;
```

```
    ofn.lpstrDefExt = TEXT("bin");
```

```
    ofn.lCustData = 0L;
```

```
    ofn.lpfnHook = NULL;
```

```
    ofn.lpTemplateName = NULL;
```

```
}
```

```
////////////////////////////////////
```

```
//FileImportDlg
```

```
//功能：创建导入文件对话框
```

```
//参数：父窗口句柄hwnd，文件名，路径名
```

```
//返回：若成功则返回TRUE，失败则返回FALSE
```

---

```

BOOL FileImportDlg(HWND hwnd, PTSTR pstrFileName, PTSTR pstrFileTitle)
{
    ofn.hwndOwner = hwnd;
    ofn.lpstrFile = pstrFileName;
    ofn.lpstrFileTitle = pstrFileTitle;
    ofn.Flags = OFN_HIDEREADONLY | OFN_CREATEPROMPT;

    return GetOpenFileName(&ofn);
}

```

```

////////////////////////////////////

```

```

//FileSaveDlg

```

```

//功能：创建保存文件对话框

```

```

//参数：窗口句柄hwnd，文件名，路径名

```

```

//返回：若成功则返回TRUE，失败则返回FALSE

```

```

BOOL FileSaveDlg(HWND hwnd, PTSTR pstrFileName, PTSTR pstrFileTitle)
{
    ofn.hwndOwner = hwnd;
    ofn.lpstrFile = pstrFileName;
    ofn.lpstrFileTitle = pstrFileTitle;
    ofn.Flags = OFN_OVERWRITEPROMPT;

    return GetSaveFileName(&ofn);
}

```

```

////////////////////////////////////

```

```

//FileReadWnd

```

```

//功能：读取文件

```

```

//参数：窗口句柄hwnd，路径名

```

```

//返回：若成功则返回TRUE，失败则返回FALSE

```

```

BOOL FileReadWnd(HWND hwnd, PTSTR pstrLeadFileName)
{
    DWORD dwReadSize;
    HANDLE hFile[3], hLeadFile;
    TCHAR szPrint[50];
    int i;
    int iComNum, iBuiNum, iRoomNum;
    LenData ldData;
    Community *pcTmp;
    Building *pbTmp;
}

```

---

```
Room *prTmp;
TCHAR *szBuffers[] = {
    TEXT("%s - 楼盘.bin"),
    TEXT("%s - 楼栋.bin"),
    TEXT("%s - 房间.bin")
};
TCHAR *szPrints[3], *szCompanyName;

//获取公司名
szCompanyName = (TCHAR*)malloc(sizeof(TCHAR) * 50);
lstrcpy(szCompanyName, pstrLeadFileName);
szCompanyName[lstrlen(szCompanyName) - 4] = '\\0\\0';

//编辑数据文件名
for (i = 0; i < 3; i++)
{
    szPrints[i] = (TCHAR*)malloc(sizeof(TCHAR) * 50);
    wsprintf(szPrints[i], szBuffers[i], szCompanyName);
}

//打开引导文件
hLeadFile = CreateFile(pstrLeadFileName, GENERIC_READ, FILE_SHARE_READ,
    NULL, OPEN_EXISTING, 0, NULL);
if (INVALID_HANDLE_VALUE == hLeadFile)
    return FALSE;

//打开数据文件
for (i = 0; i < 3; i++)
    hFile[i] = CreateFile(szPrints[i], GENERIC_READ, FILE_SHARE_READ,
        NULL, OPEN_EXISTING, 0, NULL);

//读取三个文件的大小
ReadFile(hLeadFile, &ldData, sizeof(LenData), &dwReadSize, NULL);

//检查读取是否正常
if ((int)dwReadSize != sizeof(LenData))
{
    for (i = 0; i < 3; i++)
        CloseHandle(hFile[i]);
    CloseHandle(hLeadFile);
    return FALSE;
}

//复制数据
```



---

```
iComNum = ldData.iCom / sizeof(Community);
iBuiNum = ldData.iBui / sizeof(Building);
iRoomNum = ldData.iRoom / sizeof(Room);

//确定编号开始
iAutoComNum = iComNum + 1;

//分配空间
pcTmp = (Community*)malloc(sizeof(Community));
pbTmp = (Building*)malloc(sizeof(Building));
prTmp = (Room*)malloc(sizeof(Room));

pHead = NULL;
//读取三个文件并写入链表
while (iComNum > 0)
{
    iComNum--;
    ReadFile(hFile[0], pcTmp, sizeof(Community), &dwReadSize, NULL);
    SetFilePointer(hFile[0], sizeof(Community), NULL, FILE_CURRENT);

    //检验是否全部写入成功
    if ((int)dwReadSize != sizeof(Community))
    {
        CloseHandle(hFile[0]);
        return FALSE;
    }
    //加入链表
    AddComToList(*pcTmp, &pHead);

    while (iBuiNum > 0)
    {
        iBuiNum--;
        ReadFile(hFile[1], pbTmp, sizeof(Building), &dwReadSize, NULL);
        SetFilePointer(hFile[1], sizeof(Building), NULL, FILE_CURRENT);

        //检验是否全部写入成功
        if ((int)dwReadSize != sizeof(Building))
        {
            CloseHandle(hFile[1]);
            return FALSE;
        }

        //检验是否遇到楼栋分隔点
        if (pbTmp->buildingNum == -1)
```

---

```

        {
            iBuiNum++;
            break;
        }

        //将楼栋加入链表
        AddBuiToList(*pbTmp, &pHead, pcTmp->communityNum);

        while (iRoomNum > 0)
        {
            iRoomNum--;
            ReadFile(hFile[2], prTmp, sizeof(Room), &dwReadSize, NULL);
            SetFilePointer(hFile[2], sizeof(Room), NULL, FILE_CURRENT);

            //检验是否全部写入成功
            if ((int)dwReadSize != sizeof(Room))
            {
                CloseHandle(hFile[2]);
                return FALSE;
            }

            //检验是否遇到房间分隔点
            if (prTmp->roomNum == -1)
            {
                iRoomNum++;
                break;
            }

            //将房间加入链表
            AddRoomToList(*prTmp, &pHead,
                pcTmp->communityNum, pbTmp->buildingNum);
        }
    }
}

//检验数据是否全部读取
if (iRoomNum || iBuiNum || iComNum)
{
    for (i = 0; i < 3; i++)
        CloseHandle(hFile[i]);
    CloseHandle(hLeadFile);
    return FALSE;
}

```

---

```

//设置主窗口标题内容
wsprintf(szPrint, TEXT("楼盘管理系统 - %s"), szCompanyName);
SetWindowText(hwnd, (PTSTR)szPrint);

//回收句柄和指针
CloseHandle(hLeadFile);
for (i = 0; i < 3; i++)
    CloseHandle(hFile[i]);
free(pcTmp);
free(pbTmp);
free(prTmp);

return TRUE;
}

////////////////////////////////////
//FileWriteWnd
//功能：写入文件
//参数：窗口句柄hwnd，引导文件文件名
//返回：若成功则返回TRUE，失败则返回FALSE

BOOL FileWriteWnd(HWND hwnd, PTSTR pstrLeadFileName)
{
    DWORD dwWriteSize;
    HANDLE hFile[3], hLeadFile;
    int i, iComLen = 0, iBuiLen = 0, iRoomLen = 0;
    Community *pTailCom;
    Building *pTailBui, *pNullBui;
    Room *pTailRoom, *pNullRoom;
    WORD wMark = 0xFEFF;
    LenData Data;
    TCHAR *szCompanyName;
    TCHAR *szBuffers[] = {
        TEXT("%s - 楼盘.bin"),
        TEXT("%s - 楼栋.bin"),
        TEXT("%s - 房间.bin")
    };
    TCHAR *szPrints[3];

//获取公司名
szCompanyName = (TCHAR*)malloc(sizeof(TCHAR) * 50);
lstrcpy(szCompanyName, pstrLeadFileName);
szCompanyName[lstrlen(szCompanyName) - 4] = '\\0\\0';

```

---

```

//编辑数据文件名
for (i = 0; i < 3; i++)
{
    szPrints[i] = (TCHAR*)malloc(sizeof(TCHAR) * 50);
    wsprintf(szPrints[i], szBuffers[i], szCompanyName);
}

//打开文件
for (i = 0; i < 3; i++)
    hFile[i] = CreateFile(szPrints[i], GENERIC_WRITE, 0,
        NULL, CREATE_ALWAYS, 0, NULL);
hLeadFile = CreateFile(pstrLeadFileName, GENERIC_WRITE, 0,
    NULL, CREATE_ALWAYS, 0, NULL);

if (INVALID_HANDLE_VALUE == hLeadFile)
    return FALSE;

//编辑bui与room结构在文件中的分隔符

pNullBui = (Building*)malloc(sizeof(Building));
pNullRoom = (Room*)malloc(sizeof(Room));
pNullBui->buildingNum = -1;
pNullRoom->roomNum = -1;

//获取所有数据并写入三个文件中

for (pTailCom = pHead; pTailCom;
    pTailCom = pTailCom->nextCommunity)
{
    iComLen += sizeof(Community);
    WriteFile(hFile[0], pTailCom, sizeof(Community), &dwWriteSize, NULL);
    SetFilePointer(hFile[0], sizeof(Community), NULL, FILE_CURRENT);

    //检验是否全部写入成功
    if ((int)dwWriteSize != sizeof(Community))
    {
        CloseHandle(hFile[0]);
        return FALSE;
    }

    for (pTailBui = pTailCom->buildings; pTailBui;
        pTailBui = pTailBui->nextBuilding)
    {

```

---

```

    iBuiLen += sizeof(Building);
    WriteFile(hFile[1], pTailBui, sizeof(Building), &dwWriteSize, NULL);
    SetFilePointer(hFile[1], sizeof(Building), NULL, FILE_CURRENT);

    //检验是否全部写入成功
    if ((int)dwWriteSize != sizeof(Building))
    {
        CloseHandle(hFile[1]);
        return FALSE;
    }

    for (pTailRoom = pTailBui->rooms; pTailRoom;
        pTailRoom = pTailRoom->nextRoom)
    {
        iRoomLen += sizeof(Room);
        WriteFile(hFile[2], pTailRoom, sizeof(Room), &dwWriteSize, NULL);
        SetFilePointer(hFile[2], sizeof(Room), NULL, FILE_CURRENT);

        //检验是否全部写入成功
        if ((int)dwWriteSize != sizeof(Room))
        {
            CloseHandle(hFile[2]);
            return FALSE;
        }
    }

    //设置Room分隔符,并检验是否成功
    WriteFile(hFile[2], pNullRoom, sizeof(Room), &dwWriteSize, NULL);
    SetFilePointer(hFile[2], sizeof(Room), NULL, FILE_CURRENT);
    if ((int)dwWriteSize != sizeof(Room))
    {
        CloseHandle(hFile[2]);
        return FALSE;
    }
}

//设置Building分隔符,并检验是否成功
WriteFile(hFile[1], pNullBui, sizeof(Building), &dwWriteSize, NULL);
SetFilePointer(hFile[1], sizeof(Building), NULL, FILE_CURRENT);
if ((int)dwWriteSize != sizeof(Building))
{
    CloseHandle(hFile[1]);
    return FALSE;
}

```

---

```

    }

    //将三个文件的数据大小写入导入文件中
    Data.iCom = iComLen;
    Data.iBui = iBuiLen;
    Data.iRoom = iRoomLen;
    WriteFile(hLeadFile, &Data, sizeof(LenData), &dwWriteSize, NULL);
    if ((int)dwWriteSize != sizeof(LenData))
    {
        for (i = 0; i < 3; i++)
            CloseHandle(hFile[i]);
        return FALSE;
    }

    //检验文件是否为空
    if (!pcBuffer && !pbBuffer && !pcBuffer)
    {
        for (i = 0; i < 3; i++)
            CloseHandle(hFile[i]);
        return FALSE;
    }

    for (i = 0; i < 3; i++)
        CloseHandle(hFile[i]);
    CloseHandle(hLeadFile);

    return TRUE;
}

////////////////////////////////////
//SetTitle
//功能: 设置窗口内容
//参数: 编辑框句柄hwnd, 文件名
//返回: 无返回值

void SetTitle(HWND hwnd, TCHAR * szTitleName)
{
    TCHAR szCaption[64 + MAX_PATH];

    wsprintf(szCaption, TEXT("%s - %s"), TEXT("楼盘管理系统"),
        szTitleName[0] ? szTitleName : TEXT("未定义"));

    SetWindowText(hwnd, szCaption);
}

```

---

```

}

////////////////////////////////////
//DrawComInf
//功能：绘制楼盘信息窗口
//参数：编辑框句柄hwnd，设备环境hdc，Community结构
//返回：成功返回TRUE，失败返回FALSE

BOOL DrawComInf(HWND hwnd, HDC hdc, Community com)
{
    int cxChar, cyChar, cxClient, cyClient;
    TEXTMETRIC tm;
    TCHAR szBuffer[100];
    RECT rcClient;

    GetTextMetrics(hdc, &tm);
    cxChar = tm.tmAveCharWidth;
    cyChar = tm.tmHeight + tm.tmExternalLeading;
    GetClientRect(hwnd, &rcClient);
    cxClient = rcClient.right;
    cyClient = rcClient.bottom;

    SelectObject(hdc, hFontSon);
    SetBkMode(hdc, TRANSPARENT);

    TextOut(hdc, cxChar + XLINEPOS, 0.5*cyChar,
        szBuffer, wsprintf(szBuffer, TEXT("楼盘名称: %s"), com.name));

    TextOut(hdc, cxChar + XLINEPOS, 0.5*cyChar + cyChar * 1.25,
        szBuffer, wsprintf(szBuffer, TEXT("楼盘编号: %d"), com.communityNum));

    TextOut(hdc, cxChar + XLINEPOS, 0.5*cyChar + 2 * cyChar * 1.25,
        szBuffer, wsprintf(szBuffer, TEXT("楼栋地址: %s"), com.address));

    TextOut(hdc, cxChar + XLINEPOS, 0.5*cyChar + 3 * cyChar * 1.25,
        szBuffer, wsprintf(szBuffer, TEXT("楼栋数量: %d"), com.numberofBuildings));

    TextOut(hdc, cxChar + XLINEPOS, 0.5*cyChar + 4 * cyChar * 1.25,
        szBuffer, wsprintf(szBuffer, TEXT("房间数量: %d"), com.numberofRooms));

    TextOut(hdc, cxChar + XLINEPOS, 0.5*cyChar + 5 * cyChar * 1.25,
        szBuffer, swprintf(szBuffer, TEXT("平均价格: %.2f"), com.avgPrice));
}

```

---

```

    TextOut(hdc, 2.7 * XLINEPOS, 0.5*cyChar,
        szBuffer, wsprintf(szBuffer, TEXT("联系人 : %s"), com.host.name));

    TextOut(hdc, 2.7 * XLINEPOS, 0.5*cyChar + cyChar * 1.25,
        szBuffer, wsprintf(szBuffer, TEXT("联系电话: %s"), com.phone));

    DrawButton(hdc, cxClient, YLINEPOS);

    return TRUE;
}

////////////////////////////////////
//DrawBuiInf
//功能: 绘制楼栋信息窗口
//参数: 编辑框句柄hwnd, 设备环境hdc, Building结构
//返回: 成功返回TRUE, 失败返回FALSE

BOOL DrawBuiInf(HWND hwnd, HDC hdc, Building bui)
{
    int cxChar, cyChar, cxClient, cyClient;
    TEXTMETRIC tm;
    TCHAR szBuffer[100];
    RECT rcClient;

    GetTextMetrics(hdc, &tm);
    cxChar = tm.tmAveCharWidth;
    cyChar = tm.tmHeight + tm.tmExternalLeading;
    GetClientRect(hwnd, &rcClient);
    cxClient = rcClient.right;
    cyClient = rcClient.bottom;

    SelectObject(hdc, hFontSon);
    SetBkMode(hdc, TRANSPARENT);

    TextOut(hdc, cxChar + XLINEPOS, 0.5*cyChar,
        szBuffer, wsprintf(szBuffer, TEXT("楼栋编号: %d"), bui.buildingNum));

    TextOut(hdc, cxChar + XLINEPOS, 0.5*cyChar + cyChar * 1.25,
        szBuffer, wsprintf(szBuffer, TEXT("所在楼盘: %s"), bui.inCom));

    TextOut(hdc, cxChar + XLINEPOS, 0.5*cyChar + 2 * cyChar * 1.25,
        szBuffer, wsprintf(szBuffer, TEXT("楼层数 : %d"), bui.numberOfFloors));

```



---

```

    TextOut(hdc, cxChar + XLINEPOS, 0.5*cyChar + 3 * cyChar * 1.25,
        szBuffer, wsprintf(szBuffer, TEXT("房间数    : %d"), bui.numberOfRooms));

    TextOut(hdc, cxChar + XLINEPOS, 0.5*cyChar + 4 * cyChar * 1.25,
        szBuffer, swprintf(szBuffer, TEXT("平均价格: %.2f"), bui.avgPrice));

    TextOut(hdc, 2.7 * XLINEPOS, 0.5*cyChar,
        szBuffer, wsprintf(szBuffer, TEXT("联系人    : %s"), bui.host.name));

    TextOut(hdc, 2.7 * XLINEPOS, 0.5*cyChar + cyChar * 1.25,
        szBuffer, wsprintf(szBuffer, TEXT("联系电话: %s"), bui.host.phone));

    DrawButton(hdc, cxClient, YLINEPOS);

    return TRUE;
}

////////////////////////////////////
//DrawRoomInf
//功能: 绘制楼栋信息窗口
//参数: 编辑框句柄hwnd, 设备环境hdc, Room结构
//返回: 成功返回TRUE, 失败返回FALSE

BOOL DrawRoomInf(HWND hwnd, HDC hdc, Room room)
{
    int cxChar, cyChar, cxClient, cyClient;
    TEXTMETRIC tm;
    TCHAR szBuffer[100], szPrint[100];
    RECT rcClient;

    GetTextMetrics(hdc, &tm);
    cxChar = tm.tmAveCharWidth;
    cyChar = tm.tmHeight + tm.tmExternalLeading;
    GetClientRect(hwnd, &rcClient);
    cxClient = rcClient.right;
    cyClient = rcClient.bottom;

    SelectObject(hdc, hFontSon);
    SetBkMode(hdc, TRANSPARENT);

    TextOut(hdc, cxChar + XLINEPOS, 0.5*cyChar,
        szBuffer, wsprintf(szBuffer, TEXT("房间号    : %d"), room.roomNum));

```

---

```

    TextOut(hdc, cxChar + XLINEPOS, 0.5*cyChar + cyChar * 1.25,
        szBuffer, wsprintf(szBuffer, TEXT("所在楼盘: %s"), room.inCom));

    TextOut(hdc, cxChar + XLINEPOS, 0.5*cyChar + 2 * cyChar * 1.25,
        szBuffer, wsprintf(szBuffer, TEXT("所在楼栋: %d"), room.buildingNum));

    TextOut(hdc, cxChar + XLINEPOS, 0.5*cyChar + 3 * cyChar * 1.25,
        szBuffer, wsprintf(szBuffer, TEXT("所在楼层: %d"), room.floor));

    TextOut(hdc, cxChar + XLINEPOS, 0.5*cyChar + 4 * cyChar * 1.25,
        szBuffer, swprintf(szBuffer, TEXT("房间价格: %.2f 元 / 平方米"),
room.roomPrice));

    TextOut(hdc, cxChar + XLINEPOS, 0.5*cyChar + 5 * cyChar * 1.25,
        szBuffer, swprintf(szBuffer, TEXT("房间总价: %d 元"), room.allPrice));

    TextOut(hdc, 2.5 * XLINEPOS, 0.5*cyChar,
        szBuffer, wsprintf(szBuffer, TEXT("联系人   : %s"), room.host.name));

    TextOut(hdc, 2.5 * XLINEPOS, 0.5*cyChar + cyChar * 1.25,
        szBuffer, wsprintf(szBuffer, TEXT("联系电话: %s"), room.host.phone));

    TextOut(hdc, 2.5 * XLINEPOS, 0.5*cyChar + 2 * cyChar * 1.25,
        szBuffer, wsprintf(szBuffer, TEXT("%s"), room.roomType));

    if (room.roomState)
        lstrcpy(szBuffer, TEXT("已售出"));
    else lstrcpy(szBuffer, TEXT("未售出"));
    TextOut(hdc, 2.5 * XLINEPOS, 0.5*cyChar + 3 * cyChar * 1.25,
        szPrint, wsprintf(szPrint, TEXT("售出情况: %s"), szBuffer));

    DrawButton(hdc, cxClient, YLINEPOS);
    roomNow = room.roomNum;

    return TRUE;
}

```

```

////////////////////////////////////
//RenewData
//功能: 将数据区恢复为背景色
//参数: 设备环境hdc, Building结构
//返回: 无返回

```

---

```

void RenewData(HDC hdc, RECT rectData)
{
    SelectObject(hdc, GetStockObject(DC_BRUSH));
    SetDCBrushColor(hdc, RGB(137, 189, 255));    //该颜色为天蓝色

    Rectangle(hdc, rectData.left, rectData.top,
        rectData.right, rectData.bottom);

    SelectObject(hdc, GetStockObject(BLACK_BRUSH));
}

////////////////////////////////////
//UpdataData
//功能：更新整个链表的数据，包括房间数，均价等
//参数：将要更新的链表的头指针的地址
//返回：若完成所有更新则返回TRUE，否则返回FALSE

BOOL UpdateData(Community **ppHead)
{
    Community *pCom;
    Building *pBui;
    Room *pRoom;
    float avgComPrice = 0.0, avgBuiPrice = 0.0;
    int comRoomNum = 0, buiRoomNum = 0, comBuiNum = 0;
    float sumBuiPrice = 0.0, sumComPrice = 0.0;

    for (pCom = *ppHead; pCom; pCom = pCom->nextCommunity)
    {
        //初始化楼盘价格和房间数
        sumComPrice = 0.0;
        comRoomNum = 0;
        comBuiNum = 0;

        for (pBui = pCom->buildings; pBui; pBui = pBui->nextBuilding)
        {
            //初始化楼栋价格和房间数、楼栋数
            sumBuiPrice = 0.0;
            buiRoomNum = 0;

            strcpy(pBui->inCom, pCom->name);

            for (pRoom = pBui->rooms; pRoom; pRoom = pRoom->nextRoom)
            {

```

---

```

        pRoom->buildingNum = pBui->buildingNum;

        //计算房间总价
        pRoom->allPrice = pRoom->roomSize * pRoom->roomPrice;
        pRoom->floor = pRoom->roomNum / 100;    //计算楼层数
        pRoom->host = pBui->host; //房间联系人与楼栋一致

        //加入楼栋总价格和总房间数
        sumBuiPrice += pRoom->roomPrice;
        buiRoomNum++;
    }

    //将数据写入链表
    pBui->numberOfRooms = buiRoomNum;
    if (buiRoomNum)
        pBui->avgPrice = sumBuiPrice / buiRoomNum;
    else pBui->avgPrice = 0.0;

    //加入楼盘总价格和楼栋数
    sumComPrice += sumBuiPrice;
    comRoomNum += buiRoomNum;
    comBuiNum++;
}

//将数据写入链表
pCom->numberOfRooms = comRoomNum;
if (comRoomNum != 0)
    pCom->avgPrice = sumComPrice / comRoomNum;
else pCom->avgPrice = 0.0;
pCom->numberOfBuildings = comBuiNum;
}

return TRUE;
}

////////////////////////////////////
//Draw1Step
//功能: WM_PAINT消息中绘制第一级(包含Com与Bui), 不显示数据
//参数: 设备环境hdc, 显示数据的com指针(NULL表示无楼盘)
//返回: 无返回

void Draw1Step(HDC hdc, Community *pChosenCom)
{

```

---

```

HINSTANCE hInstance;
Community *pCom;
Building *pBui;
TCHAR szPrint[100];
TCHAR *pszSubTitle[] = {
    TEXT("楼栋号"),
    TEXT("所在楼盘"),
    TEXT("平均价格"),
    TEXT("联系电话")
};
int piColWid[] = { 80, 250, 170, 115 };
int iCol;
LVCOLUMN lvc;
LVITEM lvi;

hInstance = (HINSTANCE)GetWindowLong(hwnd, GWL_HINSTANCE);

lvc.mask = LVCF_SUBITEM | LVCF_TEXT | LVCF_WIDTH | LVCF_FMT;

//设置字体和背景
SendMessage(hwndSubList, WM_SETFONT, (WPARAM)hFontSon, TRUE);
SetWindowText(hwndMainTitle, TEXT("楼 盘"));

//添加listview的4列
for (iCol = 0; iCol < 4; iCol++)
{
    lvc.iSubItem = iCol;
    lvc.pszText = pszSubTitle[iCol];
    lvc.cx = piColWid[iCol];
    lvc.fmt = LVCFMT_CENTER;

    ListView_DeleteColumn(hwndSubList, iCol);
    ListView_InsertColumn(hwndSubList, iCol, &lvc);
}

//重置主菜单数据
SendMessage(hwndMainList, LB_RESETCONTENT, 0, 0);
ListView_DeleteAllItems(hwndSubList);

/*****/
/*
col1      col2      col3      col4
item0    subitem1  subitem2  subitem3
item1    subitem1  subitem2  subitem3

```

---

```

    item2    subitem1    subitem2    subitem3
    .....(item2 == subitem0)
    */

    //空表示目前无楼盘，直接退出
    if (!pChosenCom)
        return;

    //填充表格
    lvi.mask = LVIF_TEXT | LVIF_STATE;
    lvi.iSubItem = 0; //初始化行数
    lvi.pszText = LPSTR_TEXTCALLBACK;
    lvi.state = 0;
    lvi.stateMask = 0;
    lvi.iItem = 0;

    for (pCom = pHead; pCom; pCom = pCom->nextCommunity)
    {
        SendMessage(hwndMainList, LB_ADDSTRING, 0, (LPARAM)pCom->name);
        if (pCom->communityNum == pChosenCom->communityNum)
            for (pBui = pCom->buildings; pBui; pBui = pBui->nextBuilding)
            {
                //添加一行表格
                ListView_InsertItem(hwndSubList, &lvi);

                //像一行添加文本
                sprintf(szPrint, TEXT("%d"), pBui->buildingNum);
                ListView_SetItemText(hwndSubList, lvi.iItem, 0, szPrint);
                sprintf(szPrint, TEXT("%s"), pBui->inCom);
                ListView_SetItemText(hwndSubList, lvi.iItem, 1, szPrint);
                sprintf(szPrint, TEXT("%.2f"), pBui->avgPrice);
                ListView_SetItemText(hwndSubList, lvi.iItem, 2, szPrint);
                sprintf(szPrint, TEXT("%s"), pBui->host.phone);
                ListView_SetItemText(hwndSubList, lvi.iItem, 3, szPrint);

                lvi.iItem++;
            }
    }
}

////////////////////////////////////
//Draw2Step
//功能：WM_PAINT消息中绘制第二级，不显示数据
//参数：设备环境hdc, 指向所选楼栋的指针

```

---

//返回：无返回

```
void Draw2Step(HDC hdc, Building *pBui)
{
    Room *pRoom;
    Community *pCom;
    HINSTANCE hInstance;
    TCHAR szPrint[100];
    TCHAR *pszSubTitle[] = {
        TEXT("房间号"),
        TEXT("所在层数"),
        TEXT("房间价格"),
        TEXT("房间面积")
    };

    int piColWid[] = { 80, 250, 170, 115 };
    int iCol;
    LVCOLUMN lvc;
    LVITEM lvi;

    hInstance = (HINSTANCE)GetWindowLong(hwnd, GWL_HINSTANCE);

    lvc.mask = LVCF_SUBITEM | LVCF_TEXT | LVCF_WIDTH | LVCF_FMT;

    //设置字体和背景
    SendMessage(hwndSubList, WM_SETFONT, (LPARAM)hFontSon, TRUE);
    SetWindowText(hwndMainTitle, TEXT("楼 盘"));

    SendMessage(hwndMainList, LB_RESETCONTENT, 0, 0);
    for (pCom = pHead; pCom; pCom = pCom->nextCommunity)
        SendMessage(hwndMainList, LB_ADDSTRING, 0, (LPARAM)pCom->name);

    //添加listview的4列
    for (iCol = 0; iCol < 4; iCol++)
    {
        lvc.iSubItem = iCol;
        lvc.pszText = pszSubTitle[iCol];
        lvc.cx = piColWid[iCol];
        lvc.fmt = LVCFMT_CENTER;

        ListView_DeleteColumn(hwndSubList, iCol);
        ListView_InsertColumn(hwndSubList, iCol, &lvc);
    }

    //重置主菜单数据
```

---

```

    ListView_DeleteAllItems(hwndSubList);

    if (!pBui)
        return;

    //填充表格
    lvi.mask = LVIF_TEXT | LVIF_STATE;
    lvi.iSubItem = 0; //初始化行数
    lvi.pszText = LPSTR_TEXTCALLBACK;
    lvi.state = 0;
    lvi.stateMask = 0;
    lvi.iItem = 0;

    //更新子菜单

    //第一栏为该楼栋，以供查看楼栋信息和返回上一级
    wsprintf(szPrint, TEXT("%s, %d栋"), pBui->inCom, pBui->buildingNum);

    //添加一行表格
    ListView_InsertItem(hwndSubList, &lvi);
    ListView_SetItemText(hwndSubList, lvi.iItem, 1, szPrint);
    lvi.iItem++;

    //之后为房间基本信息
    for (pRoom = pBui->rooms; pRoom; pRoom = pRoom->nextRoom)
    {
        //添加一行表格
        ListView_InsertItem(hwndSubList, &lvi);

        //像一行添加文本
        wsprintf(szPrint, TEXT("%d"), pRoom->roomNum);
        ListView_SetItemText(hwndSubList, lvi.iItem, 0, szPrint);
        wsprintf(szPrint, TEXT("%d"), pRoom->floor);
        ListView_SetItemText(hwndSubList, lvi.iItem, 1, szPrint);
        swprintf(szPrint, TEXT("%.2f"), pRoom->roomPrice);
        ListView_SetItemText(hwndSubList, lvi.iItem, 2, szPrint);
        swprintf(szPrint, TEXT("%.2f"), pRoom->roomSize);
        ListView_SetItemText(hwndSubList, lvi.iItem, 3, szPrint);

        lvi.iItem++;
    }
}

```



---

```

////////////////////////////////////
//DelComInList
//功能：删除链表中的一个Community节点
//参数：将要删除的Community编号，添加对象链表的头指针的地址
//返回：返回更新后链表头指针

BOOL DelComInList(int comNum, Community **head)
{
    Community *comTail, *comLastTail;

    //没有楼盘的情况
    if (*head == NULL)
        return FALSE;

    //只有一个节点的情况
    if ((*head)->nextCommunity == NULL)
    {
        //检查第一个节点是不是所求值
        if ((*head)->communityNum == comNum)
        {
            //只有一个节点，直接指向NULL
            *head = NULL;
            return TRUE;
        }
        else return FALSE;
    }

    //检查第一个节点是不是所求值
    if ((*head)->communityNum == comNum)
    {
        //有多个节点，连到下一个
        *head = (*head)->nextCommunity;
        return TRUE;
    }

    comTail = (*head)->nextCommunity;
    comLastTail = *head;

    //循环删除该楼盘
    while (comTail != NULL)
    {
        if (comTail->communityNum == comNum)
        {
            comLastTail->nextCommunity = comTail->nextCommunity;

```

---

```

        return TRUE;
    }

    comTail = comTail->nextCommunity;
    comLastTail = comLastTail->nextCommunity;
}

return FALSE;
}

////////////////////////////////////
//DelBuiInList
//功能：删除链表中的一个Building节点
//参数：将要删除的Building编号，添加对象链表的头指针的地址
//返回：返回更新后链表头指针

BOOL DelBuiInList(int comNum, int buiNum, Community **head)
{
    Community *comTail;
    Building *buiTail, *buiLastTail;

    //没有楼盘的情况
    if (*head == NULL)
        return FALSE;

    for (comTail = *head; comTail != NULL;
        comTail = comTail->nextCommunity)
    {
        if (comNum == comTail->communityNum)
        {
            //没有楼栋的情况
            if (comTail->buildings == NULL)
                return FALSE;

            //只有一个节点的情况
            if (comTail->buildings->nextBuilding == NULL)
            {
                //检查第一个节点是不是所求值
                if (comTail->buildings->buildingNum == buiNum)
                {
                    //只有一个节点，直接指向NULL
                    comTail->buildings = NULL;
                }
            }
        }
    }
}

```

---

```

        return TRUE;
    }
    else return FALSE;
}

//检查第一个节点是不是所求值
if (comTail->buildings->buildingNum == buiNum)
{
    //指向下一个节点
    comTail->buildings = comTail->buildings->nextBuilding;
    return TRUE;
}

//初始化两个指针
buiLastTail = comTail->buildings;
buiTail = comTail->buildings->nextBuilding;

//循环删除
while (buiTail != NULL)
{
    if (buiTail->buildingNum == buiNum)
    {
        buiLastTail->nextBuilding = buiTail->nextBuilding;
        return TRUE;
    }
    buiLastTail = buiLastTail->nextBuilding;
    buiTail = buiTail->nextBuilding;
}
}

return FALSE;
}

////////////////////////////////////
//DelRoomInList
//功能：删除链表中的一个Room节点
//参数：将要删除的Room编号，添加对象链表的头指针的地址
//返回：返回更新后链表头指针

BOOL DelRoomInList(int comNum, int buiNum, int roomNum, Community **head)
{
    Community *comTail;
    Building *buiTail;
    Room *roomTail, *roomLastTail;

```

---

```
//没有楼盘的情况
if (*head == NULL)
    return FALSE;

for (comTail = *head; comTail != NULL;
    comTail = comTail->nextCommunity)
{
    if (comNum == comTail->communityNum)
    {
        //没有楼栋的情况
        if (comTail->buildings == NULL)
            return FALSE;

        for (buiTail = comTail->buildings; buiTail != NULL;
            buiTail = buiTail->nextBuilding)
        {
            if (buiTail->buildingNum == buiNum)
            {
                //没有房间的情况
                if (buiTail->rooms == NULL)
                    return FALSE;

                //只有一个节点的情况
                if (buiTail->rooms->nextRoom == NULL)
                {
                    //检查第一个节点是不是所求值
                    if (buiTail->rooms->roomNum == roomNum)
                    {
                        //只有一个节点，直接指向NULL
                        buiTail->rooms = NULL;
                        return TRUE;
                    }
                    else return FALSE;
                }

                //检查第一个节点是不是所求值
                if (buiTail->rooms->roomNum == roomNum)
                {
                    //指向下一个
                    buiTail->rooms = buiTail->rooms->nextRoom;
                    return TRUE;
                }
            }
        }
    }
}
```

---

```

        //初始化两个指针
        roomLastTail = buiTail->rooms;
        roomTail = buiTail->rooms->nextRoom;

        //循环删除
        while (roomTail != NULL)
        {
            if (roomTail->roomNum == roomNum)
            {
                roomLastTail->nextRoom = roomTail->nextRoom;
                return TRUE;
            }
            roomLastTail = roomLastTail->nextRoom;
            roomTail = roomTail->nextRoom;
        }
    }
}

return FALSE;
}

```

```

////////////////////////////////////
// “删除楼盘” (DeleteCommunity) 对话框窗口过程

BOOL CALLBACK DelComDlgProc(HWND hDlg, UINT message,
    WPARAM wParam, LPARAM lParam)
{
    TCHAR szBuffer[100][100];
    Community *comTail;
    int iComDelNum = -1;
    static int iStateEdit; //1表示输入的是名字, 2表示输入的是楼盘号
    int iLength[5];

    switch (message)
    {
        case WM_INITDIALOG:

            SendMessage(GetDlgItem(hDlg, IDC_RADIO3),
                BM_SETCHECK, BST_CHECKED, 0);
            //Disable第二个编辑框
            EnableWindow(GetDlgItem(hDlg, IDC_EDIT_DEL_COMNUM), FALSE);
            iStateEdit = 1;

```

---

```
    return TRUE;

case WM_COMMAND:
    switch (LOWORD(wParam))
    {
    case IDC_RADIO3:

        if (HIWORD(wParam) == BN_CLICKED)
        {
            //交换enable的编辑框
            EnableWindow(GetDlgItem(hDlg, IDC_EDIT_DEL_COMNAME), TRUE);
            EnableWindow(GetDlgItem(hDlg, IDC_EDIT_DEL_COMNUM), FALSE);
            SetWindowText(GetDlgItem(hDlg, IDC_EDIT_DEL_COMNUM), TEXT(""));
            iStateEdit = 1;
        }
        break;

    case IDC_RADIO4:

        if (HIWORD(wParam) == BN_CLICKED)
        {
            //交换enable的编辑框
            EnableWindow(GetDlgItem(hDlg, IDC_EDIT_DEL_COMNUM), TRUE);
            EnableWindow(GetDlgItem(hDlg, IDC_EDIT_DEL_COMNAME), FALSE);
            SetWindowText(GetDlgItem(hDlg, IDC_EDIT_DEL_COMNAME), TEXT(""));
            iStateEdit = 2;
        }
        break;

    case IDOK:

        //当按下OK时，将所输入内容存进存储区

        if (iStateEdit == 1)
            iLength[0] = SendMessage(GetDlgItem(hDlg, IDC_EDIT_DEL_COMNAME),
                                    EM_GETLINE, 0, (LPARAM)szBuffer[0]);
        else if (iStateEdit == 2)
            iLength[0] = SendMessage(GetDlgItem(hDlg, IDC_EDIT_DEL_COMNUM),
                                    EM_GETLINE, 1, (LPARAM)szBuffer[0]);

        //检验输入非空

        if (!iLength[0])
        {
```

---

```
        MessageBox(hDlg, TEXT("请输入完整的信息! "), TEXT("提示"), MB_OK);
        return FALSE;
    }
    else *(szBuffer[0] + iLength[0]) = '\0\0';

    //存储或寻找需要删除的楼栋号

    if (iStateEdit == 1)
    {
        comTail = pHead;

        //循环查找楼盘名
        while (comTail != NULL)
        {
            if (!lstrcmp(comTail->name, szBuffer[0]))
            {
                iComDelNum = comTail->communityNum;
                break;
            }
            comTail = comTail->nextCommunity;
        }
    }
    else if (iStateEdit == 2)
    {
        iComDelNum = _ttoi(szBuffer[0]);
    }

    //检查楼盘名是否正确
    if (iComDelNum == -1)
    {
        MessageBox(hDlg, TEXT("该楼盘名称不存在! "), TEXT("提示"), MB_OK);
        return FALSE;
    }

    //删除，并检验楼盘号
    if (!DelComInList(iComDelNum, &pHead))
    {
        MessageBox(hDlg, TEXT("该楼盘号不存在! "), TEXT("提示"), MB_OK);
        return FALSE;
    }

    bRepaint = TRUE;
    EndDialog(hDlg, TRUE);
    return TRUE;
```

---

```

        case IDCANCEL:
            bRePaint = FALSE;
            EndDialog(hDlg, FALSE);
            return TRUE;
        }
        break;
    }

    return FALSE;
}

////////////////////////////////////
// “删除楼盘” (DeleteBui) 对话框窗口过程

BOOL CALLBACK DelBuiDlgProc(HWND hDlg, UINT message,
    WPARAM wParam, LPARAM lParam)
{
    TCHAR szBuffer[100][100];
    Community *comTail;
    int i, iComDelNum = -1, iBuiDelNum = -1;
    static int iStateEdit; // 1表示输入的是名字, 2表示输入的是楼盘号
    int iLength[5];

    switch (message)
    {
        case WM_INITDIALOG:

            SendMessage(GetDlgItem(hDlg, IDC_RADIO3),
                BM_SETCHECK, BST_CHECKED, 0);
            //Disable第二个编辑框
            EnableWindow(GetDlgItem(hDlg, IDC_EDIT_DEL_COMNUM), FALSE);
            iStateEdit = 1;
            return TRUE;

        case WM_COMMAND:
            switch (LOWORD(wParam))
            {
                case IDC_RADIO3:

                    if (HIWORD(wParam) == BN_CLICKED)
                    {
                        //交换enable的编辑框

```



---

```

        EnableWindow(GetDlgItem(hDlg, IDC_EDIT_DEL_COMNAME), TRUE);
        EnableWindow(GetDlgItem(hDlg, IDC_EDIT_DEL_COMNUM), FALSE);
        SetWindowText(GetDlgItem(hDlg, IDC_EDIT_DEL_COMNUM), TEXT(""));
        iStateEdit = 1;
    }
    break;

case IDC_RADIO4:

    if (HIWORD(wParam) == BN_CLICKED)
    {
        //交换enable的编辑框
        EnableWindow(GetDlgItem(hDlg, IDC_EDIT_DEL_COMNUM), TRUE);
        EnableWindow(GetDlgItem(hDlg, IDC_EDIT_DEL_COMNAME), FALSE);
        SetWindowText(GetDlgItem(hDlg, IDC_EDIT_DEL_COMNAME), TEXT(""));
        iStateEdit = 2;
    }
    break;

case IDOK:

    //当按下OK时，将所输入内容存进存储区

    if (iStateEdit == 1)
        iLength[0] = SendMessage(GetDlgItem(hDlg, IDC_EDIT_DEL_COMNAME),
            EM_GETLINE, 0, (LPARAM)szBuffer[0]);
    else if (iStateEdit == 2)
        iLength[0] = SendMessage(GetDlgItem(hDlg, IDC_EDIT_DEL_COMNUM),
            EM_GETLINE, 1, (LPARAM)szBuffer[0]);

    iLength[1] = SendMessage(GetDlgItem(hDlg, IDC_EDIT_DEL_BUINUM),
        EM_GETLINE, 3, (LPARAM)szBuffer[1]);

    //检验每一项非空

    for (i = 0; i < 2; i++)
    {
        if (!iLength[i])
        {
            MessageBox(hDlg, TEXT("请输入完整的信息!"), TEXT("提示"),
MB_OK);

            return FALSE;
        }
    }
    else {

```

---

```
        *(szBuffer[i] + iLength[i]) = '\0\0';
    }
}

//存储或寻找需要删除的楼栋号

if (iStateEdit == 1)
{
    comTail = pHead;

    //循环查找楼盘名
    while (comTail != NULL)
    {
        if (!strcmp(comTail->name, szBuffer[0]))
        {
            iComDelNum = comTail->communityNum;
            break;
        }
        comTail = comTail->nextCommunity;
    }
}
else if (iStateEdit == 2)
{
    iComDelNum = _ttoi(szBuffer[0]);
}
iBuiDelNum = _ttoi(szBuffer[1]);

//检查楼盘名是否正确
if (iComDelNum == -1)
{
    MessageBox(hDlg, TEXT("该楼盘名称不存在!"), TEXT("提示"), MB_OK);
    return FALSE;
}

//删除, 并检验楼栋号
if (!DelBuiInList(iComDelNum, iBuiDelNum, &pHead))
{
    MessageBox(hDlg, TEXT("找不到该楼栋!"), TEXT("提示"), MB_OK);
    return FALSE;
}

bRepaint = TRUE;
EndDialog(hDlg, TRUE);
return TRUE;
```

---

```

        case IDCANCEL:
            bRepaint = FALSE;
            EndDialog(hDlg, FALSE);
            return TRUE;
        }
        break;
    }

    return FALSE;
}

////////////////////////////////////
// “删除楼盘” (DeleteRoom) 对话框窗口过程

BOOL CALLBACK DelRoomDlgProc(HWND hDlg, UINT message,
    WPARAM wParam, LPARAM lParam)
{
    TCHAR szBuffer[100][100];
    Community *comTail;
    int i;
    int iComDelNum = -1, iBuiDelNum = -1, iRoomDelNum = -1;
    static int iStateEdit; // 1表示输入的是名字, 2表示输入的是楼盘号
    int iLength[5];

    switch (message)
    {
        case WM_INITDIALOG:

            SendMessage(GetDlgItem(hDlg, IDC_RADIO3),
                BM_SETCHECK, BST_CHECKED, 0);
            // Disable第二个编辑框
            EnableWindow(GetDlgItem(hDlg, IDC_EDIT_DEL_COMNUM), FALSE);
            iStateEdit = 1;
            return TRUE;

        case WM_COMMAND:
            switch (LOWORD(wParam))
            {
                case IDC_RADIO3:

                    if (HIWORD(wParam) == BN_CLICKED)
                    {

```

---

```

        //交换enable的编辑框
        EnableWindow(GetDlgItem(hDlg, IDC_EDIT_DEL_COMNAME), TRUE);
        EnableWindow(GetDlgItem(hDlg, IDC_EDIT_DEL_COMNUM), FALSE);
        SetWindowText(GetDlgItem(hDlg, IDC_EDIT_DEL_COMNUM), TEXT(""));
        iStateEdit = 1;
    }
    break;

case IDC_RADIO4:

    if (HIWORD(wParam) == BN_CLICKED)
    {
        //交换enable的编辑框
        EnableWindow(GetDlgItem(hDlg, IDC_EDIT_DEL_COMNUM), TRUE);
        EnableWindow(GetDlgItem(hDlg, IDC_EDIT_DEL_COMNAME), FALSE);
        SetWindowText(GetDlgItem(hDlg, IDC_EDIT_DEL_COMNAME), TEXT(""));
        iStateEdit = 2;
    }
    break;

case IDOK:

    //当按下OK时，将所输入内容存进存储区

    if (iStateEdit == 1)
        iLength[0] = SendMessage(GetDlgItem(hDlg, IDC_EDIT_DEL_COMNAME),
            EM_GETLINE, 0, (LPARAM)szBuffer[0]);
    else if (iStateEdit == 2)
        iLength[0] = SendMessage(GetDlgItem(hDlg, IDC_EDIT_DEL_COMNUM),
            EM_GETLINE, 1, (LPARAM)szBuffer[0]);

    iLength[1] = SendMessage(GetDlgItem(hDlg, IDC_EDIT_DEL_BUINUM),
        EM_GETLINE, 2, (LPARAM)szBuffer[1]);
    iLength[2] = SendMessage(GetDlgItem(hDlg, IDC_EDIT_DEL_ROOMNUM),
        EM_GETLINE, 3, (LPARAM)szBuffer[2]);

    //检验每一项非空

    for (i = 0; i < 3; i++)
    {
        if (!iLength[i])
        {
            MessageBox(hDlg, TEXT("请输入完整的信息!"), TEXT("提示"),
                MB_OK);

```

---

```
        return FALSE;
    }
    else {
        *(szBuffer[i] + iLength[i]) = '\0\0';
    }
}

//将所在楼盘名转化为楼盘号

if (iStateEdit == 1)
{
    comTail = pHead;

    //循环查找楼盘名
    while (comTail != NULL)
    {
        if (!lstrcmp(comTail->name, szBuffer[0]))
        {
            iComDelNum = comTail->communityNum;
            break;
        }
        comTail = comTail->nextCommunity;
    }
}
else if (iStateEdit == 2)
{
    iComDelNum = _ttoi(szBuffer[0]);
}
iBuiDelNum = _ttoi(szBuffer[1]);
iRoomDelNum = _ttoi(szBuffer[2]);

//检查楼盘名是否正确
if (iComDelNum == -1)
{
    MessageBox(hDlg, TEXT("该楼盘名称不存在!"), TEXT("提示"), MB_OK);
    return FALSE;
}

//删除, 并检验房间号
if (!DelRoomInList(iComDelNum, iBuiDelNum, iRoomDelNum, &pHead))
{
    MessageBox(hDlg, TEXT("找不到该房间!"), TEXT("提示"), MB_OK);
    return FALSE;
}
```

---

```

        bRePaint = TRUE;
        EndDialog(hDlg, TRUE);
        return TRUE;

    case IDCANCEL:
        bRePaint = FALSE;
        EndDialog(hDlg, FALSE);
        return TRUE;
    }
    break;
}
return FALSE;
}

////////////////////////////////////
//绘制按钮
//参数: hdc, 信息窗口右下角的坐标

void DrawButton(HDC hdc, int x, int y)
{
    int cxChar, cyChar;
    TEXTMETRIC tm;

    GetTextMetrics(hdc, &tm);
    cxChar = tm.tmAveCharWidth;
    cyChar = tm.tmHeight + tm.tmExternalLeading;

    MoveWindow(hwndButtonDel, x - cxChar * 8, y - cyChar * 2,
        cxChar * 6, cyChar * 7 / 4, TRUE);

    MoveWindow(hwndButtonEdit, x - cxChar * 16, y - cyChar * 2,
        cxChar * 6, cyChar * 7 / 4, TRUE);

    if (SEARCHING)
        MoveWindow(hwndButtonEnter, x - cxChar * 24, y - cyChar * 2,
            cxChar * 6, cyChar * 7 / 4, TRUE);

    ShowWindow(hwndButtonEnter, SW_SHOW);
}

////////////////////////////////////
// “编辑楼盘信息” (EditComDlgProc) 对话框窗口过程

```

---

```

BOOL CALLBACK EditComDlgProc(HWND hDlg, UINT message,
    WPARAM wParam, LPARAM lParam)
{
    TCHAR szBuffer[100][100];
    Community *comTail;
    int i;
    int iLength[5];

    switch (message)
    {
    case WM_INITDIALOG:

        //设置编辑框初始内容
        comTail = pHead;
        while (comTail != NULL)
        {
            if (!lstrcmp(comTail->name, comNow))
            {
                SetWindowText(GetDlgItem(hDlg, IDC_ADD_COM_NAME), comTail->name);
                SetWindowText(GetDlgItem(hDlg, IDC_ADD_COM_ADDR), comTail->address);
                SetWindowText(GetDlgItem(hDlg, IDC_ADD_COM_PHONE), comTail->phone);
                SetWindowText(GetDlgItem(hDlg, IDC_ADD_COM_PERSON),
comTail->host.name);
                break;
            }
            comTail = comTail->nextCommunity;
        }
        return TRUE;

    case WM_COMMAND:

        switch (LOWORD(wParam))
        {
        case IDOK:

            //当按下OK时，将所输入内容存进存储区

            iLength[0] = SendMessage(GetDlgItem(hDlg, IDC_ADD_COM_NAME),
                EM_GETLINE, 0, (LPARAM)szBuffer[0]);
            iLength[1] = SendMessage(GetDlgItem(hDlg, IDC_ADD_COM_ADDR),
                EM_GETLINE, 1, (LPARAM)szBuffer[1]);
            iLength[2] = SendMessage(GetDlgItem(hDlg, IDC_ADD_COM_PHONE),
                EM_GETLINE, 2, (LPARAM)szBuffer[2]);

```

---

```

iLength[3] = SendMessage(GetDlgItem(hDlg, IDC_ADD_COM_PERSON),
    EM_GETLINE, 3, (LPARAM)szBuffer[3]);

//检验每一项非空

for (i = 0; i < 4; i++)
{
    if (!iLength[i])
    {
        MessageBox(hDlg, TEXT("请输入完整的信息!"), TEXT("提示"),
MB_OK);

        return FALSE;
    }
    else {
        *(szBuffer[i] + iLength[i]) = '\0\0';
    }
}

//修改链表内数据

comTail = pHead;
while (comTail != NULL)
{
    if (!lstrcmp(comTail->name, comNow))
    {
        if (CheckSameCom(szBuffer[0], &pHead) ||
            (lstrcmp(comTail->name, szBuffer[0]) == 0))
        {
            lstrcpy(comTail->name, szBuffer[0]);
            lstrcpy(comTail->address, szBuffer[1]);
            lstrcpy(comTail->phone, szBuffer[2]);
            lstrcpy(comTail->host.name, szBuffer[3]);

            lstrcpy(comNow, comTail->name);
        }
        else
        {
            MessageBox(hwnd, TEXT("该楼盘号已在系统中"),
                TEXT("警告"), MB_OK | MB_ICONWARNING);
            return FALSE;
        }
        break;
    }
    comTail = comTail->nextCommunity;
}

```



---

```

    }

    bRePaint = TRUE;
    EndDialog(hDlg, TRUE);
    return TRUE;

case IDCANCEL:
    bRePaint = FALSE;
    EndDialog(hDlg, FALSE);
    return TRUE;
}

break;
}

return FALSE;
}

////////////////////////////////////
// “编辑楼栋信息” (EditBuiDlgProc) 对话框窗口过程

BOOL CALLBACK EditBuiDlgProc(HWND hDlg, UINT message,
    WPARAM wParam, LPARAM lParam)
{
    TCHAR szBuffer[100][100];
    Community *comTail;
    Building *buiTail;
    int i;
    int iLength[5];

    switch (message)
    {
    case WM_INITDIALOG:

        //设置编辑框初始内容
        comTail = pHead;
        while (comTail != NULL)
        {
            if (!lstrcmp(comTail->name, comNow))
            {
                buiTail = comTail->buildings;
                while (buiTail != NULL)
                {

```

---

```

        if (buiTail->buildingNum == buiNow)
        {
            wsprintf(szBuffer[0], TEXT("%d"), buiTail->buildingNum);
            SetWindowText(GetDlgItem(hDlg, IDC_ADD_BUI_NUM),
szBuffer[0]);

            wsprintf(szBuffer[1], TEXT("%d"), buiTail->numberOfFloors);
            SetWindowText(GetDlgItem(hDlg, IDC_ADD_BUI_FLOORS),
szBuffer[1]);

            SetWindowText(GetDlgItem(hDlg, IDC_ADD_BUI_PHONE),
buiTail->host.phone);

            SetWindowText(GetDlgItem(hDlg, IDC_ADD_BUI_PERSON),
buiTail->host.name);

            break;
        }
        buiTail = buiTail->nextBuilding;
    }
    break;
}
comTail = comTail->nextCommunity;
}
return TRUE;

case WM_COMMAND:

    switch (LOWORD(wParam))
    {
        case IDOK:

            //当按下OK时，将所输入内容存进存储区

            iLength[0] = SendMessage(GetDlgItem(hDlg, IDC_ADD_BUI_NUM),
                EM_GETLINE, 0, (LPARAM)szBuffer[0]);
            iLength[1] = SendMessage(GetDlgItem(hDlg, IDC_ADD_BUI_FLOORS),
                EM_GETLINE, 1, (LPARAM)szBuffer[1]);
            iLength[2] = SendMessage(GetDlgItem(hDlg, IDC_ADD_BUI_PHONE),
                EM_GETLINE, 2, (LPARAM)szBuffer[2]);
            iLength[3] = SendMessage(GetDlgItem(hDlg, IDC_ADD_BUI_PERSON),
                EM_GETLINE, 3, (LPARAM)szBuffer[3]);

            //检验每一项非空

            for (i = 0; i < 4; i++)
            {
                if (!iLength[i])

```

---

```

        {
            MessageBox(hDlg, TEXT("请输入完整的信息!"), TEXT("提示"),
MB_OK);

            return FALSE;
        }
        else {
            *(szBuffer[i] + iLength[i]) = '\0\0';
        }
    }

//修改链表内数据

comTail = pHead;
while (comTail != NULL)
{

    if (!lstrcmp(comTail->name, comNow))
    {
        buiTail = comTail->buildings;
        while (buiTail != NULL)
        {
            if (buiTail->buildingNum == buiNow)
            {
                if (CheckSameBui (comTail->name,
                    _ttoi (szBuffer[0]), &pHead) ||
                    (buiTail->buildingNum == _ttoi (szBuffer[0])))
                {
                    buiTail->buildingNum = _ttoi (szBuffer[0]);
                    buiTail->numberOfFloors = _ttoi (szBuffer[1]);
                    lstrcpy (buiTail->host.phone, szBuffer[2]);
                    lstrcpy (buiTail->host.name, szBuffer[3]);

                    buiNow = buiTail->buildingNum;
                }
                else
                {
                    MessageBox (hwnd, TEXT ("该楼栋号已在系统中"),
                        TEXT ("警告"), MB_OK | MB_ICONWARNING);
                    return FALSE;
                }
                break;
            }
            buiTail = buiTail->nextBuilding;
        }
    }
}

```

---

```

        break;
    }
    comTail = comTail->nextCommunity;
}

bRePaint = TRUE;
EndDialog(hDlg, TRUE);
return TRUE;

case IDCANCEL:
    bRePaint = FALSE;
    EndDialog(hDlg, FALSE);
    return TRUE;
}

break;
}

return FALSE;
}

////////////////////////////////////
// “编辑房间信息” (EditRoomDlgProc) 对话框窗口过程

BOOL CALLBACK EditRoomDlgProc(HWND hDlg, UINT message,
    WPARAM wParam, LPARAM lParam)
{
    TCHAR szBuffer[100][100];
    Community *comTail;
    Building *buiTail;
    Room *roomTail;
    int i, iSold;
    int iLength[5];

    switch (message)
    {
    case WM_INITDIALOG:

        //向ComboBox框中添加数据
        for (i = 0; i < 34; i++)
            SendMessage(GetDlgItem(hDlg, IDC_ADD_ROOM_STYLE),
                CB_ADDSTRING, 0, (LPARAM)szComboBoxData[i]);
    }
}

```

---

```

//设置编辑框初始内容
comTail = pHead;
while (comTail != NULL)
{
    if (!strcmp(comTail->name, comNow))
    {
        buiTail = comTail->buildings;
        while (buiTail != NULL)
        {
            if (buiTail->buildingNum == buiNow)
            {
                roomTail = buiTail->rooms;
                while (roomTail != NULL)
                {
                    if (roomTail->roomNum == roomNow)
                    {
                        wsprintf(szBuffer[2], TEXT("%d"),
roomTail->roomNum);
                        SetWindowText(GetDlgItem(hDlg,
IDC_ADD_ROOM_ROOMNUM), szBuffer[2]);
                        swprintf(szBuffer[3], TEXT("%.2f"),
roomTail->roomSize);
                        SetWindowText(GetDlgItem(hDlg, IDC_ADD_ROOM_SIZE),
szBuffer[3]);
                        swprintf(szBuffer[4], TEXT("%.2f"),
roomTail->roomPrice);
                        SetWindowText(GetDlgItem(hDlg, IDC_ADD_ROOM_PRICE),
szBuffer[4]);

                        //设置默认户型
                        for (i = 0; i < 34; i++)
                            if (!strcmp(szComboBoxData[i],
roomTail->roomType))

                                break;
                        SendMessage(GetDlgItem(hDlg, IDC_ADD_ROOM_STYLE),
CB_SETCURSEL, i, 0);

                        //设置默认售出状态
                        if (roomTail->roomState)
                            SendMessage(GetDlgItem(hDlg,
IDC_ADD_ROOM_SOLD),

                                BM_SETCHECK, (LPARAM)BST_CHECKED, 0);
                        break;
                    }
                }
            }
        }
    }
}

```

---

```

        roomTail = roomTail->nextRoom;
    }
    break;
}
buiTail = buiTail->nextBuilding;
}
break;
}
comTail = comTail->nextCommunity;
}
return TRUE;

case WM_COMMAND:

    switch (LOWORD(wParam))
    {
    case IDOK:

        //当按下OK时，将所输入内容存进存储区

        iLength[0] = SendMessage(GetDlgItem(hDlg, IDC_ADD_ROOM_ROOMNUM),
            EM_GETLINE, 0, (LPARAM)szBuffer[0]);
        iLength[1] = SendMessage(GetDlgItem(hDlg, IDC_ADD_ROOM_SIZE),
            EM_GETLINE, 1, (LPARAM)szBuffer[1]);
        iLength[2] = SendMessage(GetDlgItem(hDlg, IDC_ADD_ROOM_PRICE),
            EM_GETLINE, 2, (LPARAM)szBuffer[2]);
        GetWindowText(GetDlgItem(hDlg, IDC_ADD_ROOM_STYLE),
            szBuffer[3], lstrlen(szBuffer[3]));
        iSold = SendMessage(GetDlgItem(hDlg, IDC_ADD_ROOM_SOLD),
            BM_GETCHECK, 0, 0);

        //检验每一项非空

        for (i = 0; i < 3; i++)
        {
            if (!iLength[i])
            {
                MessageBox(hDlg, TEXT("请输入完整的信息!"), TEXT("提示"),
MB_OK);

                return FALSE;
            }
            else {
                *(szBuffer[i] + iLength[i]) = '\0\0';
            }
        }
    }
}

```

---

```

    }

    //修改链表内数据

    comTail = pHead;
    while (comTail != NULL)
    {

        if (!strcmp(comTail->name, comNow))
        {
            buiTail = comTail->buildings;
            while (buiTail != NULL)
            {
                if (buiTail->buildingNum == buiNow)
                {
                    roomTail = buiTail->rooms;
                    while (roomTail != NULL)
                    {
                        if (roomTail->roomNum == roomNow)
                        {
                            if (CheckSameRoom(comTail->name,
                                buiTail->buildingNum, _ttoi(szBuffer[0]),

                                (roomTail->roomNum == _ttoi(szBuffer[0])))
                            {
                                roomTail->roomNum = _ttoi(szBuffer[0]);
                                roomTail->roomSize = _ttoi(szBuffer[1]);
                                roomTail->roomPrice = _ttoi(szBuffer[2]);
                                strcpy(roomTail->roomType, szBuffer[3]);
                                roomTail->roomState = iSold;

                                roomNow = roomTail->roomNum;
                            }
                            else
                            {
                                MessageBox(hwnd, TEXT("该房间名已在系统中

                                TEXT("警告"), MB_OK | MB_ICONWARNING);
                                return FALSE;
                            }
                        }
                        break;
                    }
                    roomTail = roomTail->nextRoom;
                }
            }
        }
    }
}

```

---

```

        break;
    }
    buiTail = buiTail->nextBuilding;
}
break;
}
comTail = comTail->nextCommunity;
}

bRePaint = TRUE;
EndDialog(hDlg, TRUE);
return TRUE;

case IDCANCEL:
    bRePaint = FALSE;
    EndDialog(hDlg, FALSE);
    return TRUE;
}
break;
}
return FALSE;
}

```

```

////////////////////////////////////

```

```

//计算出每一个楼盘的相似度
//参数：标准SearchData结构，对比的楼盘
//返回：这个输入信息与该楼盘的相似度

```

```

int GetComSim(SearchData data, Community com)
{
    int weigh = 0;

    if (data.mask != SD_COM)
        return -1;

    if (!strcmp(data.comName, com.name))
        weigh += 3;
    if (data.comNum == com.communityNum)
        weigh += 2;
    if (!strcmp(data.comPerson, com.host.name))
        weigh += 1;
    if (!strcmp(data.comPhone, com.phone))
        weigh += 1;
}

```



---

```

        return weigh;
    }

////////////////////////////////////
//计算出每一个楼栋的相似度
//参数: 标准SearchData结构, 对比的楼栋, 该楼栋所在楼盘
//返回: 这个输入信息与该楼栋的相似度

int GetBuiSim(SearchData data, Community com, Building bui)
{
    int weigh = 0;

    if (data.mask != SD_BUI)
        return -1;

    if (!strcmp(data.comName, com.name))
        weigh += 5;
    if (data.comNum == com.communityNum)
        weigh += 2;
    if (!strcmp(data.comPerson, com.host.name))
        weigh += 1;
    if (!strcmp(data.comPhone, com.phone))
        weigh += 1;

    if (data.buiNum == bui.buildingNum)
        weigh += 5;
    if (!strcmp(data.buiPerson, bui.host.name))
        weigh += 1;
    if (!strcmp(data.buiPhone, bui.host.phone))
        weigh += 1;

    return weigh;
}

////////////////////////////////////
//计算出每一个房间的相似度
//参数: 标准SearchData结构, 对比的房间, 该房间所在楼栋、楼盘
//返回: 这个输入信息与该房间的相似度

int GetRoomSim(SearchData data, Community com, Building bui, Room room)
{

```

---

```

    int weigh = 0;

    if (data.mask != SD_ROOM)
        return -1;

    if (!strcmp(data.comName, com.name))
        weigh += 5;
    if (data.comNum == com.communityNum)
        weigh += 3;
    if (!strcmp(data.comPerson, com.host.name))
        weigh += 1;
    if (!strcmp(data.comPhone, com.phone))
        weigh += 1;

    if (data.buiNum == bui.buildingNum)
        weigh += 5;
    if (!strcmp(data.buiPerson, bui.host.name))
        weigh += 1;
    if (!strcmp(data.buiPhone, bui.host.phone))
        weigh += 1;

    if (data.roomNum == room.roomNum)
        weigh += 5;
    if (!strcmp(data.roomType, room.roomType))
        weigh += 1;
    if (data.roomFloor == room.floor)
        weigh += 2;
    if (data.roomSize <= room.roomSize + 10 &&
        data.roomSize >= room.roomSize - 10)
        weigh += 2;
    if (room.roomPrice >= data.roomLoPrice &&
        room.roomPrice <= data.roomHiPrice)
        weigh += 2;
    if (room.allPrice <= data.allPrice)
        weigh += 3;
    if (room.roomState == data.roomSold)
        weigh += 2;

    return weigh;
}

```

```

////////////////////////////////////
// “搜索功能”（Search）对话框窗口过程

```

---

```

BOOL CALLBACK SearchDlgProc(HWND hDlg, UINT message,
    WPARAM wParam, LPARAM lParam)
{
    TCHAR szBuffer[100][100];
    Community *comTail;
    Building *buiTail;
    Room *roomTail;
    SearchData sData;
    BOOL success = FALSE;
    int i, iSold;
    static int iSearchPos = 0; //1表示com, 2表示bui, 3表示room, 0表示未选择
    int iLength[20];

    switch (message)
    {
    case WM_INITDIALOG:

        //向ComboBox框中添加数据
        for (i = 0; i < 34; i++)
            SendMessage(GetDlgItem(hDlg, IDC_S_ROOMTYPE),
                CB_ADDSTRING, 0, (LPARAM)szComboBoxData[i]);

        //设置默认选项楼盘
        SendMessage(GetDlgItem(hDlg, IDC_SCOM),
            BM_SETCHECK, BST_CHECKED, 0);
        //Disable楼栋和房间
        EnableWindow(GetDlgItem(hDlg, IDC_S_BUINUM), FALSE);
        EnableWindow(GetDlgItem(hDlg, IDC_S_BUIPERSON), FALSE);
        EnableWindow(GetDlgItem(hDlg, IDC_S_BUIPHONE), FALSE);

        EnableWindow(GetDlgItem(hDlg, IDC_S_ROOMNUM), FALSE);
        EnableWindow(GetDlgItem(hDlg, IDC_S_ROOMSIZE), FALSE);
        EnableWindow(GetDlgItem(hDlg, IDC_S_ROOMFLOOR), FALSE);
        EnableWindow(GetDlgItem(hDlg, IDC_S_ALLPRICE), FALSE);
        EnableWindow(GetDlgItem(hDlg, IDC_S_LOPRICE), FALSE);
        EnableWindow(GetDlgItem(hDlg, IDC_S_HIPRICE), FALSE);
        EnableWindow(GetDlgItem(hDlg, IDC_S_ROOMTYPE), FALSE);

        iSearchPos = 1;

        return TRUE;

    case WM_COMMAND:

```

---

```

switch (LOWORD(wParam))
{
case IDC_SCOM:

    if (HIWORD(wParam) == BN_CLICKED)
    {
        //Disable楼栋和房间
        EnableWindow(GetDlgItem(hDlg, IDC_S_BUINUM), FALSE);
        EnableWindow(GetDlgItem(hDlg, IDC_S_BUIPERSON), FALSE);
        EnableWindow(GetDlgItem(hDlg, IDC_S_BUIPHONE), FALSE);

        EnableWindow(GetDlgItem(hDlg, IDC_S_ROOMNUM), FALSE);
        EnableWindow(GetDlgItem(hDlg, IDC_S_ROOMSIZE), FALSE);
        EnableWindow(GetDlgItem(hDlg, IDC_S_ROOMFLOOR), FALSE);
        EnableWindow(GetDlgItem(hDlg, IDC_S_ALLPRICE), FALSE);
        EnableWindow(GetDlgItem(hDlg, IDC_S_LOPRICE), FALSE);
        EnableWindow(GetDlgItem(hDlg, IDC_S_HIPRICE), FALSE);
        EnableWindow(GetDlgItem(hDlg, IDC_S_ROOMTYPE), FALSE);

        iSearchPos = 1;
    }
    break;

case IDC_SBUI:

    if (HIWORD(wParam) == BN_CLICKED)
    {
        //Disable房间

        EnableWindow(GetDlgItem(hDlg, IDC_S_ROOMNUM), FALSE);
        EnableWindow(GetDlgItem(hDlg, IDC_S_ROOMSIZE), FALSE);
        EnableWindow(GetDlgItem(hDlg, IDC_S_ROOMFLOOR), FALSE);
        EnableWindow(GetDlgItem(hDlg, IDC_S_ALLPRICE), FALSE);
        EnableWindow(GetDlgItem(hDlg, IDC_S_LOPRICE), FALSE);
        EnableWindow(GetDlgItem(hDlg, IDC_S_HIPRICE), FALSE);
        EnableWindow(GetDlgItem(hDlg, IDC_S_ROOMTYPE), FALSE);

        //Enable楼栋

        EnableWindow(GetDlgItem(hDlg, IDC_S_BUINUM), TRUE);
        EnableWindow(GetDlgItem(hDlg, IDC_S_BUIPERSON), TRUE);
        EnableWindow(GetDlgItem(hDlg, IDC_S_BUIPHONE), TRUE);

```

---

```

        iSearchPos = 2;
    }
    break;

case IDC_SROOM:

    if (HIWORD(wParam) == BN_CLICKED)
    {
        //Enable房间和楼栋

        EnableWindow(GetDlgItem(hDlg, IDC_S_ROOMNUM), TRUE);
        EnableWindow(GetDlgItem(hDlg, IDC_S_ROOMSIZE), TRUE);
        EnableWindow(GetDlgItem(hDlg, IDC_S_ROOMFLOOR), TRUE);
        EnableWindow(GetDlgItem(hDlg, IDC_S_ALLPRICE), TRUE);
        EnableWindow(GetDlgItem(hDlg, IDC_S_LOPRICE), TRUE);
        EnableWindow(GetDlgItem(hDlg, IDC_S_HIPRICE), TRUE);
        EnableWindow(GetDlgItem(hDlg, IDC_S_ROOMTYPE), TRUE);

        EnableWindow(GetDlgItem(hDlg, IDC_S_BUINUM), TRUE);
        EnableWindow(GetDlgItem(hDlg, IDC_S_BUIPERSON), TRUE);
        EnableWindow(GetDlgItem(hDlg, IDC_S_BUIPHONE), TRUE);

        iSearchPos = 3;
    }
    break;

case IDOK:

    switch (iSearchPos)
    {
    case 0:        //若搜索对象为空（基本不可能出现）
        MessageBox(hDlg, TEXT("请选择将要搜索的对象"), TEXT("提示"), MB_OK);
        break;

    case 1:        //若搜索对象为楼盘

        sData.mask = SD_COM;

        iLength[0] = SendMessage(GetDlgItem(hDlg, IDC_S_COMNAME),
            EM_GETLINE, 0, (LPARAM)szBuffer[0]);
        iLength[1] = SendMessage(GetDlgItem(hDlg, IDC_S_COMNUM),
            EM_GETLINE, 0, (LPARAM)szBuffer[1]);
        iLength[2] = SendMessage(GetDlgItem(hDlg, IDC_S_COMPERSON),
            EM_GETLINE, 0, (LPARAM)szBuffer[2]);
    }
}

```

---

```

iLength[3] = SendMessage(GetDlgItem(hDlg, IDC_S_COMPHONE),
    EM_GETLINE, 0, (LPARAM)szBuffer[3]);

//将每一个字符串添加结尾
for (i = 0; i < 4; i++)
    *(szBuffer[i] + iLength[i]) = '\0\0';

//录入搜索框信息
lstrcpy(sData.comName, szBuffer[0]);
sData.comNum = _ttoi(szBuffer[1]);
lstrcpy(sData.comPerson, szBuffer[2]);
lstrcpy(sData.comPhone, szBuffer[3]);

//构建相似的搜索结果组成链表
pComSearch = NULL;
success = FALSE;
for(comTail=pHead;comTail!=NULL;
    comTail=comTail->nextCommunity)
    if (GetComSim(sData, *comTail))
    {
        AddComToList(*comTail, &pComSearch);
        success = TRUE;
    }

if (success == FALSE)
{
    MessageBox(hwnd, TEXT("未找到搜索结果"), TEXT("提示"), MB_OK);
    return FALSE;
}

//将搜索结果的链表进行排序
pComSearch = RankCom(sData, pComSearch);

iSearch = SEARCHCOM;
break;

case 2:    //若搜索对象为楼栋

sData.mask = SD_BUI;

iLength[0] = SendMessage(GetDlgItem(hDlg, IDC_S_COMNAME),
    EM_GETLINE, 0, (LPARAM)szBuffer[0]);
iLength[1] = SendMessage(GetDlgItem(hDlg, IDC_S_COMNUM),
    EM_GETLINE, 0, (LPARAM)szBuffer[1]);

```

---

```

iLength[2] = SendMessage(GetDlgItem(hDlg, IDC_S_COMPERSON),
    EM_GETLINE, 0, (LPARAM)szBuffer[2]);
iLength[3] = SendMessage(GetDlgItem(hDlg, IDC_S_COMPHONE),
    EM_GETLINE, 0, (LPARAM)szBuffer[3]);

iLength[4] = SendMessage(GetDlgItem(hDlg, IDC_S_BUINUM),
    EM_GETLINE, 0, (LPARAM)szBuffer[4]);
iLength[5] = SendMessage(GetDlgItem(hDlg, IDC_S_BUIPERSON),
    EM_GETLINE, 0, (LPARAM)szBuffer[5]);
iLength[6] = SendMessage(GetDlgItem(hDlg, IDC_S_BUIPHONE),
    EM_GETLINE, 0, (LPARAM)szBuffer[6]);

//将每一个字符串添加结尾
for (i = 0; i < 7; i++)
    *(szBuffer[i] + iLength[i]) = '\0\0';

//录入搜索框信息
lstrcpy(sData.comName, szBuffer[0]);
sData.comNum = _ttoi(szBuffer[1]);
lstrcpy(sData.comPerson, szBuffer[2]);
lstrcpy(sData.comPhone, szBuffer[3]);
sData.buiNum = _ttoi(szBuffer[4]);
lstrcpy(sData.buiPerson, szBuffer[5]);
lstrcpy(sData.buiPhone, szBuffer[6]);

//构建相似的搜索结果组成链表
success = FALSE;
pBuiSearch = NULL;
for (comTail = pHead; comTail != NULL;
    comTail = comTail->nextCommunity)
    for (buiTail = comTail->buildings; buiTail != NULL;
        buiTail = buiTail->nextBuilding)
        if (GetBuiSim(sData, *comTail, *buiTail))
        {
            AddBuiToSearch(*buiTail, &pBuiSearch);
            success = TRUE;
        }

if (success == FALSE)
{
    MessageBox(hwnd, TEXT("未找到搜索结果"), TEXT("提示"), MB_OK);
    return FALSE;
}

```

---

```

//将搜索结果的链表进行排序
pBuiSearch = RankBui(sData, pBuiSearch);

iSearch = SEARCHBUI;
break;

case 3:      //若搜索对象为房间

sData.mask = SD_ROOM;

iLength[0] = SendMessage(GetDlgItem(hDlg, IDC_S_COMNAME),
    EM_GETLINE, 0, (LPARAM)szBuffer[0]);
iLength[1] = SendMessage(GetDlgItem(hDlg, IDC_S_COMNUM),
    EM_GETLINE, 0, (LPARAM)szBuffer[1]);
iLength[2] = SendMessage(GetDlgItem(hDlg, IDC_S_COMPERSON),
    EM_GETLINE, 0, (LPARAM)szBuffer[2]);
iLength[3] = SendMessage(GetDlgItem(hDlg, IDC_S_COMPHONE),
    EM_GETLINE, 0, (LPARAM)szBuffer[3]);

iLength[4] = SendMessage(GetDlgItem(hDlg, IDC_S_BUINUM),
    EM_GETLINE, 0, (LPARAM)szBuffer[4]);
iLength[5] = SendMessage(GetDlgItem(hDlg, IDC_S_BUIPERSON),
    EM_GETLINE, 0, (LPARAM)szBuffer[5]);
iLength[6] = SendMessage(GetDlgItem(hDlg, IDC_S_BUIPHONE),
    EM_GETLINE, 0, (LPARAM)szBuffer[6]);

iLength[7] = SendMessage(GetDlgItem(hDlg, IDC_S_ROOMNUM),
    EM_GETLINE, 0, (LPARAM)szBuffer[7]);
iLength[8] = SendMessage(GetDlgItem(hDlg, IDC_S_ROOMSIZE),
    EM_GETLINE, 0, (LPARAM)szBuffer[8]);
iLength[9] = SendMessage(GetDlgItem(hDlg, IDC_S_ROOMFLOOR),
    EM_GETLINE, 0, (LPARAM)szBuffer[9]);
iLength[10] = SendMessage(GetDlgItem(hDlg, IDC_S_ALLPRICE),
    EM_GETLINE, 0, (LPARAM)szBuffer[10]);
iLength[11] = SendMessage(GetDlgItem(hDlg, IDC_S_LOPRICE),
    EM_GETLINE, 0, (LPARAM)szBuffer[11]);
iLength[12] = SendMessage(GetDlgItem(hDlg, IDC_S_HIPRICE),
    EM_GETLINE, 0, (LPARAM)szBuffer[12]);
GetWindowText(GetDlgItem(hDlg, IDC_S_ROOMTYPE),
    szBuffer[13], lstrlen(szBuffer[13]));
iSold = SendMessage(GetDlgItem(hDlg, IDC_S_SOLD), BM_GETCHECK, 0, 0);

//将每一个字符串添加结尾
for (i = 0; i < 13; i++)

```



---

```

        *(szBuffer[i] + iLength[i]) = '\0\0';

//录入搜索框信息
lstrcpy(sData.comName, szBuffer[0]);
sData.comNum = _ttoi(szBuffer[1]);
lstrcpy(sData.comPerson, szBuffer[2]);
lstrcpy(sData.comPhone, szBuffer[3]);
sData.buiNum = _ttoi(szBuffer[4]);
lstrcpy(sData.buiPerson, szBuffer[5]);
lstrcpy(sData.buiPhone, szBuffer[6]);

sData.roomNum = _ttoi(szBuffer[7]);
sData.roomSize = _ttoi(szBuffer[8]);
sData.roomFloor = _ttoi(szBuffer[9]);
sData.allPrice = 10000.0 * _ttoi(szBuffer[10]);
sData.roomLoPrice = _ttoi(szBuffer[11]);
sData.roomHiPrice = _ttoi(szBuffer[12]);
lstrcpy(sData.roomType, szBuffer[13]);
sData.roomSold = iSold;

//构建相似的搜索结果组成链表
success = FALSE;
pRoomSearch = NULL;
for (comTail = pHead; comTail != NULL;
    comTail = comTail->nextCommunity)
    for (buiTail = comTail->buildings; buiTail != NULL;
        buiTail = buiTail->nextBuilding)
        for (roomTail = buiTail->rooms; roomTail != NULL;
            roomTail = roomTail->nextRoom)
            if (GetRoomSim(sData, *comTail, *buiTail, *roomTail))
            {
                AddRoomToSearch(*roomTail, &pRoomSearch);
                success = TRUE;
            }

if (success == FALSE)
{
    MessageBox(hwnd, TEXT("未找到搜索结果"), TEXT("提示"), MB_OK);
    return FALSE;
}

//将搜索结果的链表进行排序
pRoomSearch = RankRoom(sData, pRoomSearch);

```

---

```

        iSearch = SEARCHROOM;

        break;

    default:
        break;
    }

    EndDialog(hDlg, TRUE);
    return TRUE;

case IDCANCEL:
    EndDialog(hDlg, FALSE);
    return TRUE;

default:
    break;
}
}
return FALSE;
}

////////////////////////////////////
//对Com链表按照权重进行排序
//参数：标准比较输入数据，排序的链表头指针
//返回：返回为排完序的链表的头指针

Community *RankCom(SearchData sData, Community *pComSearch)
{
    Community *comTail, *p1, *p2, *p3;
    Community *comEnd, *tmp;
    Community *aHead;

    if (pComSearch == NULL)
        return NULL;

    if (pComSearch->nextCommunity == NULL)
        return pComSearch;

    aHead = (Community*)malloc(sizeof(Community));
    aHead->nextCommunity = pComSearch;

    comEnd = NULL, tmp = NULL;

```

---

```

while (aHead != comEnd)
{
    tmp = comTail = aHead;
    while (comTail->nextCommunity->nextCommunity != comEnd)
    {
        if (GetComSim(sData, *(comTail->nextCommunity))
            < GetComSim(sData, *(comTail->nextCommunity->nextCommunity)))
        {
            //交换 p->next 和 p->next->next 的位置
            p1 = comTail->nextCommunity;
            p2 = p1->nextCommunity;
            p3 = p2->nextCommunity;
            comTail->nextCommunity = p2;
            p2->nextCommunity = p1;
            p1->nextCommunity = p3;
            tmp = comTail->nextCommunity->nextCommunity;
        }
        comTail = comTail->nextCommunity;
    }
    comEnd = tmp;
}

return aHead->nextCommunity;
}

////////////////////////////////////
//对Bui链表按照权重进行排序
//参数：标准比较输入数据，排序的链表头指针
//返回：返回为排完序的链表的头指针

Building *RankBui(SearchData sData, Building *pBuiSearch)
{
    Building *buiTail, *p1, *p2, *p3;
    Building *buiEnd, *tmp;
    Building *aHead;
    Community com, comNext, *pTailCom;

    if (pBuiSearch == NULL)
        return NULL;

    if (pBuiSearch->nextBuilding == NULL)
        return pBuiSearch;

```

---

```

aHead = (Building*)malloc(sizeof(Building));
aHead->nextBuilding = pBuiSearch;

buiEnd = NULL, tmp = NULL;
while (aHead != buiEnd)
{
    tmp = buiTail = aHead;
    while (buiTail->nextBuilding->nextBuilding != buiEnd)
    {
        //找到两个所在的楼盘
        for (pTailCom = pHead; pTailCom != NULL;
            pTailCom = pTailCom->nextCommunity)
        {
            if (strcmp(buiTail->nextBuilding->inCom,
                pTailCom->name) == 0)
                com = *pTailCom;
            if (strcmp(buiTail->nextBuilding->nextBuilding->inCom,
                pTailCom->name) == 0)
                comNext = *pTailCom;
        }

        if (GetBuiSim(sData, com, *(buiTail->nextBuilding))
            < GetBuiSim(sData, comNext,
                *(buiTail->nextBuilding->nextBuilding)))
        {
            //交换 p->next 和 p->next->next 的位置
            p1 = buiTail->nextBuilding;
            p2 = p1->nextBuilding;
            p3 = p2->nextBuilding;
            buiTail->nextBuilding = p2;
            p2->nextBuilding = p1;
            p1->nextBuilding = p3;
            tmp = buiTail->nextBuilding->nextBuilding;
        }
        buiTail = buiTail->nextBuilding;
    }
    buiEnd = tmp;
}

return aHead->nextBuilding;
}

```

```

////////////////////////////////////

```

---

```

//对Room链表按照权重进行排序
//参数：标准比较输入数据，排序的链表头指针
//返回：返回为排完序的链表的头指针

Room *RankRoom(SearchData sData, Room *pRoomSearch)
{
    Room *roomTail, *p1, *p2, *p3;
    Room *roomEnd, *tmp;
    Room *aHead;
    Building bui, buiNext, *pTailBui;
    Community com, comNext, *pTailCom;

    if (pRoomSearch == NULL)
        return NULL;

    if (pRoomSearch->nextRoom == NULL)
        return pRoomSearch;

    aHead = (Room*)malloc(sizeof(Room));
    aHead->nextRoom = pRoomSearch;

    roomEnd = NULL, tmp = NULL;
    while (aHead != roomEnd)
    {
        tmp = roomTail = aHead;
        while (roomTail->nextRoom->nextRoom != roomEnd)
        {
            //找到两个所在的楼盘
            for (pTailCom = pHead; pTailCom != NULL;
                pTailCom = pTailCom->nextCommunity)
            {
                if (strcmp(roomTail->nextRoom->inCom,
                    pTailCom->name) == 0)
                    com = *pTailCom;
                if (strcmp(roomTail->nextRoom->nextRoom->inCom,
                    pTailCom->name) == 0)
                    comNext = *pTailCom;
                for (pTailBui = pTailCom->buildings; pTailBui != NULL;
                    pTailBui = pTailBui->nextBuilding)
                {
                    if (roomTail->nextRoom->buildingNum
                        == pTailBui->buildingNum)
                        bui = *pTailBui;
                    if (roomTail->nextRoom->nextRoom->buildingNum

```

---

```

        == pTailBui->buildingNum)
        buiNext = *pTailBui;
    }
}

if (GetRoomSim(sData, com, bui, *(roomTail->nextRoom))
    < GetRoomSim(sData, comNext, buiNext,
        *(roomTail->nextRoom->nextRoom)))
{
    //交换 p->next 和 p->next->next 的位置
    p1 = roomTail->nextRoom;
    p2 = p1->nextRoom;
    p3 = p2->nextRoom;
    roomTail->nextRoom = p2;
    p2->nextRoom = p1;
    p1->nextRoom = p3;
    tmp = roomTail->nextRoom->nextRoom;
}
roomTail = roomTail->nextRoom;
}
roomEnd = tmp;
}

return aHead->nextRoom;
}

////////////////////////////////////////
//检查是否有重复的房间
//参数: 所在楼盘名, 所在楼栋号, 房间号, 排序的链表头指针
//返回: 如没有返回TRUE, 有返回FALSE

BOOL CheckSameRoom(TCHAR comName[], int buiNum, int roomNum, Community **head)
{
    Community *comTail;
    Building *buiTail;
    Room *roomTail;

    for (comTail = *head; comTail; comTail = comTail->nextCommunity)
        if (!lstrcmp(comTail->name, comName))
            for (buiTail = comTail->buildings; buiTail; buiTail =
buiTail->nextBuilding)
                if (buiNum == buiTail->buildingNum)
                    for (roomTail = buiTail->rooms; roomTail; roomTail =

```

---

```

roomTail->nextRoom)

        if (roomNum == roomTail->roomNum)
            return FALSE;

    return TRUE;
}

////////////////////////////////////
//检查是否有重复的楼栋
//参数：所在楼盘名，楼栋号，排序的链表头指针
//返回：如没有返回TRUE，有返回FALSE

BOOL CheckSameBui (TCHAR comName[], int buiNum, Community **head)
{
    Community *comTail;
    Building *buiTail;

    for (comTail = *head; comTail; comTail = comTail->nextCommunity)
        if (!lstrcmp(comTail->name, comName))
            for (buiTail = comTail->buildings; buiTail; buiTail =
buiTail->nextBuilding)
                if (buiNum == buiTail->buildingNum)
                    return FALSE;

    return TRUE;
}

////////////////////////////////////
//检查是否有重复的楼栋
//参数：楼盘号，排序的链表头指针
//返回：如没有返回TRUE，有返回FALSE

BOOL CheckSameCom (TCHAR comName[], Community **head)
{
    Community *comTail;

    for (comTail = *head; comTail; comTail = comTail->nextCommunity)
        if (!lstrcmp(comTail->name, comName))
            return FALSE;

    return TRUE;
}

```

---

## 2、头文件（managerSys.h）

```
#include <Windows.h>
#include <commdlg.h>
#include <tchar.h>
#include <stdlib.h>
#include <CommCtrl.h>

#define ID_MAINLIST 10
#define ID_MAINLISTTITLE 11
#define ID_SUBLIST 12
#define ID_SUBLISTTITLE 13
#define ID_FILEEDIT 14
#define TP_BUI 30000
#define TP_COM 30001
#define PA_FIRSTEPCOM 500
#define PA_SECSTEPBUI 501
#define PA_FIRSTEPBUI 502
#define PA_SECSTEPROOM 503
#define ID_BUTTONDEL 15
#define ID_BUTTONEDIT 16
#define SD_COM 600
#define SD_BUI 601
#define SD_ROOM 602
#define ID_BUTTONENTER 18

#define SEARCHCOM 701
#define NOSEARCH 702
#define SEARCHBUI 703
#define SEARCHROOM 704

#define SEARCHING (iSearch==SEARCHCOM || iSearch==SEARCHBUI || iSearch==SEARCHROOM)

#define XLINEPOS (1.0 / 4 * cxClient) //主界面竖直分界线
#define YLINEPOS ((1 - 0.618) * cyClient) //主界面水平分界线

#define Swap(x, y) (x)=(y)-(x); (x)=(x)+(y); (y)=(x)-(y); //交换x、y的值

////////////////////////////////////
//主窗口窗口过程

LRESULT CALLBACK WndProc(HWND, UINT, WPARAM, LPARAM);
```



---

```
////////////////////////////////////
//AskConfirm
//功能：当用户退出系统时，若文件未保存，则提示其是否需要保存
//      若文件已经保存，则直接退出
//参数：退出的窗口句柄，保存状态（非0为已保存，0为未保存）
//返回：若需要退出系统，则返回TRUE；若不需要，则返回FALSE
```

```
BOOL AskConfirm(HWND hwnd, int iSaveState);
```

```
////////////////////////////////////
//DrawBasicBk
//窗口基本布局构造函数
//功能：画出窗口的基本布局，不含任何信息
//参数：窗口句柄, 设备环境, 屏幕宽，屏幕高
//返回：数据窗口的RECT矩形
```

```
RECT DrawBasicBk(HWND hwnd, HDC hdc, int cxClient, int cyClient);
```

```
////////////////////////////////////
//“关于‘楼盘管理系统’”（About）对话框窗口过程
```

```
BOOL CALLBACK AboutDlgProc(HWND hDlg, UINT message,
    WPARAM wParam, LPARAM lParam);
```

```
////////////////////////////////////
//“添加新楼盘”（AddCommunity）对话框窗口过程
```

```
BOOL CALLBACK AddComDlgProc(HWND hDlg, UINT message,
    WPARAM wParam, LPARAM lParam);
```

```
////////////////////////////////////
//“添加新楼栋”（AddBuilding）对话框窗口过程
```

```
BOOL CALLBACK AddBuiDlgProc(HWND hDlg, UINT message,
    WPARAM wParam, LPARAM lParam);
```

```
////////////////////////////////////
//“添加新房间”（AddRoom）对话框窗口过程
```

---

```

BOOL CALLBACK AddRoomDlgProc(HWND hDlg, UINT message,
    WPARAM wParam, LPARAM lParam);

////////////////////////////////////
//联系人结构体
//功能：用于存储个人信息
typedef struct _Person {
    TCHAR name[100];        //姓名
    TCHAR phone[50];        //电话
    TCHAR idNumber[50];     //证件ID号
}Person;

////////////////////////////////////
//房间结构体
//功能：储存房间相关信息，并可用于构造链表
typedef struct _Room {
    int roomNum;            //房间号
    int buildingNum;        //所在楼栋号
    int communityNum;       //所在楼盘号
    int floor;              //所在楼层
    float roomSize;         //房间面积（平方米）
    float roomPrice;        //房间价格（人民币元/平方米）
    long allPrice;          //房间总价（人民币元）
    TCHAR roomType[100];    //户型：室，厅，厕，厨数量
    TCHAR inCom[100];       //所在楼盘名
    BOOL roomState;         //售出状态：0代表未售出，1代表已售出
    TCHAR note[1000];       //备注
    Person host;            //联系人结构
    struct _Room *nextRoom; //指向链表中下一个房间
}Room;

////////////////////////////////////
//楼栋结构体
//功能：储存楼栋相关信息，并可用于构造链表
typedef struct _Building {
    int buildingNum;        //楼栋号
    int communityNum;       //所在楼盘号
    int numberOfRooms;      //所有房间数
    int numberOfFloors;     //楼层数
    float avgPrice;         //平均价格
    TCHAR inCom[100];       //所在楼盘名称

```

---

```

    Person host;                //联系人结构
    struct _Building *nextBuilding;    //指向下一个楼栋
    struct _Room *rooms;            //指向该楼栋第一个房间
}Building;

////////////////////////////////////
//楼盘结构体
//功能：储存楼盘相关信息，并可用于构造链表
typedef struct _Community {
    int communityNum;            //楼盘号
    TCHAR name[100];            //楼盘名
    TCHAR address[100];        //楼盘地址
    TCHAR phone[30];            //联系电话
    int numberOfBuildings;      //所有楼栋数
    int numberOfRooms;          //所有房间数
    float avgPrice;             //平均价格
    Person host;                //联系人结构
    struct _Community *nextCommunity = NULL; //指向下一个楼盘
    struct _Building *buildings = NULL;      //指向该楼盘第一个楼栋
}Community;

////////////////////////////////////
//*.led文件中的结构体
//储存com, bui, room三个文件里面所包含的数据数
typedef struct _LenData {
    int iCom;
    int iBui;
    int iRoom;
}LenData;

////////////////////////////////////
//搜索中使用的结构体，容纳搜索的所有数据
typedef struct _SearchData {

    int mask;                    //确定搜索的对象
                                //值为SD_COM, SD_BUI, SD_ROOM中的一个

    //Community
    TCHAR comName[100];
    int comNum;
    TCHAR comPerson[100];
    TCHAR comPhone[100];

```

---

```

//Building
int buiNum;
TCHAR buiPhone[100];
TCHAR buiPerson[100];

//Room
int roomNum;
int roomFloor;
float roomSize;
long allPrice;
float roomLoPrice;
float roomHiPrice;
BOOL roomSold;
TCHAR roomType[100];
}SearchData;

////////////////////////////////////
//createComList
//功能：创建一个community的先进先出的单向链表
//参数：链表长度
//返回：该链表头指针

Community *createComList(int len);

////////////////////////////////////
//createBuiList
//功能：创建一个building的先进先出的单向链表
//参数：链表长度
//返回：该链表头指针

Building *createBuiList(int len);

////////////////////////////////////
//createRoomList
//功能：创建一个room的先进先出的单向链表
//参数：链表长度
//返回：该链表头指针

Room *createRoomList(int len);

////////////////////////////////////

```

---

```

//create3DepthList
//功能：创建一个三级链表
//参数：1、第一级个数；
//      2、每个第一级所含第二级个数
//      3、每个第二级所含第三级个数
//返回：返回该三级链表的头指针

Community *create3DepthList(int len1, int len2, int len3);

////////////////////////////////////
//AddComToList
//功能：向链表中添加一个community节点
//参数：将要添加的Community结构体（添加对象链表的头指针.全局变量提供）
//返回：返回更新后链表头指针

BOOL AddComToList(Community newCom, Community **head);

////////////////////////////////////
//AddBuiToList
//功能：向链表中添加一个building节点
//参数：将要添加的Building结构体，添加对象链表的头指针，所属楼盘号
//返回：成功则返回TRUE，失败返回FALSE

BOOL AddBuiToList(Building newBui, Community **head, int comNum);

////////////////////////////////////
//AddRoomToList
//功能：向链表中添加一个Room节点
//参数：将要添加的Room结构体，添加对象链表的头指针，所属楼盘号，所属楼栋号
//返回：成功则返回TRUE，失败返回FALSE

BOOL AddRoomToList(Room newRoom, Community **head,
    int comNum, int buiNum);

////////////////////////////////////
//FileInitWnd
//功能：初始化弹出文件窗口的数据
//参数：父窗口句柄hwnd
//返回：无返回值

```

---

```

void FileInitWnd(HWND hwnd);

////////////////////////////////////
//FileImportDlg
//功能: 创建导入文件对话框
//参数: 父窗口句柄hwnd, 文件名, 路径名
//返回: 若成功则返回TRUE, 失败则返回FALSE

BOOL FileImportDlg(HWND hwnd, PTSTR pstrFileName, PTSTR pstrTitleName);

////////////////////////////////////
//FileSaveDlg
//功能: 创建保存文件对话框
//参数: 窗口句柄hwnd, 文件名, 路径名
//返回: 若成功则返回TRUE, 失败则返回FALSE

BOOL FileSaveDlg(HWND hwnd, PTSTR pstrFileName, PTSTR pstrTitleName);

////////////////////////////////////
//FileReadWnd
//功能: 创建保存文件对话框
//参数: 窗口句柄hwnd, 文件名, 路径名
//返回: 若成功则返回TRUE, 失败则返回FALSE

BOOL FileReadWnd(HWND hEdit, PTSTR pstrFileName);

////////////////////////////////////
//FileWriteWnd
//功能: 写入文件
//参数: 编辑框句柄hwnd, 文件名
//返回: 若成功则返回TRUE, 失败则返回FALSE

BOOL FileWriteWnd(HWND hEdit, PTSTR pstrFileName);

////////////////////////////////////
//SetTitle
//功能: 设置窗口内容
//参数: 编辑框句柄hwnd, 文件名
//返回: 无返回值

```

---

```
void SetTitle(HWND hwnd, TCHAR * szTitleName);
```

```
////////////////////////////////////
```

```
//DrawComInf
```

```
//功能：绘制楼盘信息窗口
```

```
//参数：编辑框句柄hwnd，设备环境hdc，Community结构
```

```
//返回：成功返回TRUE，失败返回FALSE
```

```
BOOL DrawComInf(HWND hwnd, HDC hdc, Community com);
```

```
////////////////////////////////////
```

```
//DrawBuiInf
```

```
//功能：绘制楼栋信息窗口
```

```
//参数：编辑框句柄hwnd，设备环境hdc，Building结构
```

```
//返回：成功返回TRUE，失败返回FALSE
```

```
BOOL DrawBuiInf(HWND hwnd, HDC hdc, Building bui);
```

```
////////////////////////////////////
```

```
//DrawRoomInf
```

```
//功能：绘制楼栋信息窗口
```

```
//参数：编辑框句柄hwnd，设备环境hdc，Room结构
```

```
//返回：成功返回TRUE，失败返回FALSE
```

```
BOOL DrawRoomInf(HWND hwnd, HDC hdc, Room room);
```

```
////////////////////////////////////
```

```
//RenewData
```

```
//功能：将数据区恢复为背景色
```

```
//参数：设备环境hdc，Building结构
```

```
//返回：无返回
```

```
void RenewData(HDC hdc, RECT rectData);
```

```
////////////////////////////////////
```

```
//Draw1Step
```

```
//功能：WM_PAINT消息中绘制第一级(包含Com与Bui)，不显示数据
```

```
//参数：设备环境hdc，显示数据的com指针
```

---

//返回：无返回

void Draw1Step(HDC hdc, Community \*pChosenCom);

////////////////////////////////////

//Draw2Step

//功能：WM\_PAINT消息中绘制第二级，不显示数据

//参数：设备环境hdc,指向所选楼栋的指针

//返回：无返回

void Draw2Step(HDC hdc, Building \*pBui);

////////////////////////////////////

//UpdataData

//功能：更新整个链表的数据，包括房间数，均价等

//参数：将要更新的链表的头指针的地址

//返回：若完成所有更新则返回TRUE，否则返回FALSE

BOOL UpdateData(Community \*\*ppHead);

////////////////////////////////////

//DelComInList

//功能：删除链表中的一个Community节点

//参数：将要删除的Community编号，添加对象链表的头指针的地址

//返回：返回更新后链表头指针

BOOL DelComInList(int comNum, Community \*\*head);

////////////////////////////////////

//DelBuiInList

//功能：删除链表中的一个Building节点

//参数：将要删除的Building编号，添加对象链表的头指针的地址

//返回：返回更新后链表头指针

BOOL DelBuiInList(int comNum, int buiNum, Community \*\*head);

////////////////////////////////////

//DelRoomInList



---

```

//功能：删除链表中的一个Room节点
//参数：将要删除的Room编号，添加对象链表的头指针的地址
//返回：返回更新后链表头指针

BOOL DelRoomInList(int comNum, int buiNum, int roomNum, Community **head);

////////////////////////////////////
// “删除楼盘” (DeleteCommunity) 对话框窗口过程

BOOL CALLBACK DelComDlgProc(HWND hDlg, UINT message,
    WPARAM wParam, LPARAM lParam);

////////////////////////////////////
// “删除楼盘” (DeleteBui) 对话框窗口过程

BOOL CALLBACK DelBuiDlgProc(HWND hDlg, UINT message,
    WPARAM wParam, LPARAM lParam);

////////////////////////////////////
// “删除楼盘” (DeleteRoom) 对话框窗口过程

BOOL CALLBACK DelRoomDlgProc(HWND hDlg, UINT message,
    WPARAM wParam, LPARAM lParam);

////////////////////////////////////
//绘制按钮
//参数：hdc，信息窗口右下角的坐标

void DrawButton(HDC hdc, int x, int y);

////////////////////////////////////
// “编辑楼盘信息” (EditComDlgProc) 对话框窗口过程

BOOL CALLBACK EditComDlgProc(HWND hDlg, UINT message,
    WPARAM wParam, LPARAM lParam);

////////////////////////////////////
// “编辑楼栋信息” (EditBuiDlgProc) 对话框窗口过程

```

---

```

BOOL CALLBACK EditBuiDlgProc(HWND hDlg, UINT message,
    WPARAM wParam, LPARAM lParam);

////////////////////////////////////
// “编辑房间信息”（EditRoomDlgProc）对话框窗口过程

BOOL CALLBACK EditRoomDlgProc(HWND hDlg, UINT message,
    WPARAM wParam, LPARAM lParam);

////////////////////////////////////
//计算出每一个楼盘的相似度
//参数：标准SearchData结构，对比的楼盘
//返回：这个输入信息与该楼盘的相似度

int GetComSim(SearchData data, Community com);

////////////////////////////////////
//计算出每一个楼栋的相似度
//参数：标准SearchData结构，对比的楼栋，该楼栋所在楼盘
//返回：这个输入信息与该楼栋的相似度

int GetBuiSim(SearchData data, Community com, Building bui);

////////////////////////////////////
//计算出每一个房间的相似度
//参数：标准SearchData结构，对比的房间，该房间所在楼栋、楼盘
//返回：这个输入信息与该房间的相似度

int GetRoomSim(SearchData data, Community com, Building bui, Room room);

////////////////////////////////////
// “搜索功能”（Search）对话框窗口过程

BOOL CALLBACK SearchDlgProc(HWND hDlg, UINT message,
    WPARAM wParam, LPARAM lParam);

////////////////////////////////////
//对Com链表按照权重进行排序

```

---

```
//参数：标准比较输入数据，排序的链表头指针
//返回：返回为排完序的链表的头指针
```

```
Community *RankCom(SearchData sData, Community *pComSearch);
```

```
////////////////////////////////////
//对Bui链表按照权重进行排序
//参数：标准比较输入数据，排序的链表头指针
//返回：返回为排完序的链表的头指针
```

```
Building *RankBui(SearchData sData, Building *pBuiSearch);
```

```
////////////////////////////////////
//AddBuiToSearch
//功能：创建一个只由bui组成的search
//参数：将要添加的Building结构体，添加对象链表的头指针
//返回：成功则返回TRUE，失败返回FALSE
```

```
BOOL AddBuiToSearch(Building newCom, Building **head);
```

```
////////////////////////////////////
//对Room链表按照权重进行排序
//参数：标准比较输入数据，排序的链表头指针
//返回：返回为排完序的链表的头指针
```

```
Room *RankRoom(SearchData sData, Room *pBuiSearch);
```

```
////////////////////////////////////
//检查是否有重复的楼栋
//参数：楼盘号，排序的链表头指针
//返回：如没有返回TRUE，有返回FALSE
```

```
BOOL CheckSameCom(TCHAR comName[], Community **head);
```

```
////////////////////////////////////
//检查是否有重复的楼栋
//参数：所在楼盘名，楼栋号，排序的链表头指针
//返回：如没有返回TRUE，有返回FALSE
```

```
BOOL CheckSameBui(TCHAR comName[], int buiNum, Community **head);
```

---

```
////////////////////////////////////
```

```
//检查是否有重复的房间
```

```
//参数：所在楼盘名，所在楼栋号，房间号，排序的链表头指针
```

```
//返回：如没有返回TRUE，有返回FALSE
```

```
BOOL CheckSameRoom(TCHAR comName[], int buiNum, int roomNum, Community **head);
```

---

### 3、头文件（resource.h）

```
//{{NO_DEPENDENCIES}}
// Microsoft Visual C++ 生成的包含文件。
// 供 楼盘管理系统.rc 使用
//

#define IDI_MainIcon 5
#define IDD_DEL_COM 9
#define IDM_RoomSearch 101
#define IDD_ABOUTBOX 103
#define IDD_ADD_COM 109
#define IDD_ADD_BUI 112
#define IDD_ADD_ROOM 116
#define IDD_DEL_ROOM 121
#define IDD_DIALOG1 123
#define IDD_DEL_BUI 124
#define IDD_COMEDIT 125
#define IDD_BUIEDIT 127
#define IDD_ROOMEDIT 129
#define IDD_SEARCH 132
#define IDC_ADD_COM_NAME 1002
#define IDC_ADD_COM_NAME3 1003
#define IDC_ADD_COM_NUM 1003
#define IDC_ADD_COM_NAME2 1004
#define IDC_ADD_COM_ADDR1 1004
#define IDC_ADD_COM_ADDR 1004
#define IDC_ADD_COM_NAME4 1005
#define IDC_ADD_COM_PHONE 1005
#define IDC_ADD_COM_NAME5 1006
#define IDC_ADD_COM_PERSON 1006
#define IDC_ADD_COM_NAME6 1007
#define IDC_ADD_BUI_PERSON 1007
#define IDC_ADD_BUI_NAME 1008
#define IDC_ADD_BUI_NUM 1009
#define IDC_ADD_BUI_ROOMS 1010
#define IDC_ADD_BUI_FLOORS 1011
#define IDC_ADD_BUI_PHONE 1012
#define IDC_ADD_ROOM_COMNAME 1013
#define IDC_ADD_ROOM_BUINUM 1014
#define IDC_ADD_ROOM_ROOMNUM 1015
#define IDC_ADD_ROOM_SIZE 1016
#define IDC_ADD_ROOM_PRICE 1017
#define IDC_ADD_ROOM_STYLE 1027
#define IDC_ADD_ROOM_SOLD 1041
```

---

#define IDC_EDIT_DEL_COMNAME	1043
#define IDC_EDIT_DEL_COMNUM	1044
#define IDC_EDIT_DEL_COMNAME2	1045
#define IDC_S_COMPERSON	1045
#define IDC_EDIT_DEL_ROOMNUM	1046
#define IDC_S_BUINUM	1046
#define IDC_EDIT_DEL_COMNAME3	1047
#define IDC_S_COMPHONE	1047
#define IDC_S_BUIPERSON	1048
#define IDC_S_BUIPHONE	1049
#define IDC_S_ROOMNUM	1050
#define IDC_S_ROOMSIZE	1051
#define IDC_RADIO3	1052
#define IDC_S_ROOMFLOOR	1052
#define IDC_RADIO4	1053
#define IDC_S_LOPRICE	1053
#define IDC_S_HIPRICE	1054
#define IDC_S_ALLPRICE	1055
#define IDC_EDIT_DEL_BUINUM	1060
#define IDC_SCOM	1064
#define IDC_SBUI	1065
#define IDC_ROOMTYPE	1066
#define IDC_S_COMNAME	1067
#define IDC_S_COMNUM	1068
#define IDC_SROOM	1069
#define IDC_S_ROOMTYPE	1070
#define IDC_CHECK1	1070
#define IDC_S_SOLD	1071
#define IDM_FILE	40021
#define IDM_FILE_SAVE	40025
#define IDM_FILE_SAVE_AS	40026
#define IDM_FILE_EXIT	40028
#define IDM_EDIT	40029
#define IDM_EDIT_CHANGE	40030
#define IDM_EDIT_DELETE	40031
#define IDM_EDIT_CHANGE_DATA	40032
#define IDM_EDIT_CHANGE_POS	40033
#define IDM_SEARCH	40034
#define IDM_SEARCH_ACUT	40035
#define IDM_SEARCH_INF	40036
#define IDM_SEARCH_COND	40037
#define IDM_HELP	40038
#define IDM_HELP_USE	40039
#define IDM_HELP_ABOUT	40040

---

```

#define IDM_FILE_IMPORT 40043
#define IDM_FILE_CANCEL 40044
#define IDM_FILE_CREATE 40044
#define IDM_EDIT_ADD_ROOM 40045
#define IDM_FILE_NEW 40045
#define IDM_EDIT_ADD_BUID 40046
#define IDM_EDIT_ADD_COM 40047
#define IDM_EDIT_DEL_ROOM 40048
#define IDM_EDIT_DEL_BUID 40049
#define IDM_EDIT_DEL_COM 40050

```

```

// Next default values for new objects
//
#ifdef APSTUDIO_INVOKED
#ifndef APSTUDIO_READONLY_SYMBOLS
#define _APS_NEXT_RESOURCE_VALUE 134
#define _APS_NEXT_COMMAND_VALUE 40047
#define _APS_NEXT_CONTROL_VALUE 1072
#define _APS_NEXT_SYMED_VALUE 101
#endif
#endif

```

## 4、资源文件（楼盘管理系统.rc）

```

// Microsoft Visual C++ generated resource script.
//
#include "resource.h"

#define APSTUDIO_READONLY_SYMBOLS
////////////////////////////////////
//
// Generated from the TEXTINCLUDE 2 resource.
//
#include "winres.h"

////////////////////////////////////
#undef APSTUDIO_READONLY_SYMBOLS

////////////////////////////////////
// 中文(简体, 中国) resources

#if !defined(AFX_RESOURCE_DLL) || defined(AFX_TARG_CHS)
LANGUAGE LANG_CHINESE, SUBLANG_CHINESE_SIMPLIFIED

```

---

```

#ifdef APSTUDIO_INVOKED
////////////////////////////////////
//
// TEXTINCLUDE
//

1 TEXTINCLUDE
BEGIN
    "resource.h\0"
END

2 TEXTINCLUDE
BEGIN
    "#include ""winres.h""\r\n"
    "\0"
END

3 TEXTINCLUDE
BEGIN
    "\r\n"
    "\0"
END

#endif    // APSTUDIO_INVOKED

////////////////////////////////////
//
// Menu
//

IDM_RoomSearch MENU
BEGIN
    POPUP "文件"
    BEGIN
        MENUITEM "保存",           IDM_FILE_SAVE
        MENUITEM "另存为",         IDM_FILE_SAVE_AS
        MENUITEM "新建",           IDM_FILE_CREATE
        MENUITEM "导入",           IDM_FILE_IMPORT
        MENUITEM "退出",           IDM_FILE_EXIT
    END
    POPUP "编辑"
    BEGIN
        MENUITEM "更改",           IDM_EDIT_CHANGE_DATA

```



---

```

    POPUP "添加"
    BEGIN
        MENUITEM "添加新房间",          IDM_EDIT_ADD_ROOM
        MENUITEM "添加新楼栋",          IDM_EDIT_ADD_BUID
        MENUITEM "添加新楼盘",          IDM_EDIT_ADD_COM
    END
    POPUP "删除"
    BEGIN
        MENUITEM "删除整个房间",        IDM_EDIT_DEL_ROOM
        MENUITEM "删除整个楼栋",        IDM_EDIT_DEL_BUID
        MENUITEM "删除整个楼盘",        IDM_EDIT_DEL_COM
    END
END
MENUITEM "查询",                      IDM_SEARCH
POPUP "帮助"
BEGIN
    MENUITEM "关于软件",                IDM_HELP_ABOUT
END
END

////////////////////////////////////
//
// Dialog
//

IDD_ABOUTBOX DIALOGEX 32, 32, 311, 177
STYLE DS_SETFONT | DS_MODALFRAME | DS_FIXEDSYS | DS_CENTER | WS_POPUP | WS_CAPTION |
WS_SYSMENU
CAPTION "关于“楼盘管理系统”"
FONT 8, "MS Shell Dlg", 400, 0, 0x1
BEGIN
    DEFPUSHBUTTON "确定", IDOK, 129, 156, 50, 14
    CTEXT "楼盘管理系统 1.0", IDC_STATIC, 123, 18, 62, 10
    LTEXT "By  计算机学院201509班    吴肇敏\n\n版本号
1.0", IDC_STATIC, 95, 54, 118, 47
    CONTROL "", IDC_STATIC, "Static", SS_BLACKFRAME, 42, 35, 222, 1
END

IDD_ADD_COM DIALOGEX 0, 0, 251, 121
STYLE DS_SETFONT | DS_MODALFRAME | DS_FIXEDSYS | DS_CENTER | WS_POPUP | WS_CAPTION |
WS_SYSMENU
CAPTION "新建楼盘"
FONT 8, "MS Shell Dlg", 400, 0, 0x1

```

---

BEGIN

DEFPUSHBUTTON    "确定", IDOK, 143, 100, 50, 14  
PUSHBUTTON       "取消", IDCANCEL, 195, 100, 50, 14  
EDITTEXT          IDC\_ADD\_COM\_NAME, 82, 9, 100, 13, ES\_AUTOHSCROLL  
RTEXT             "楼盘名", IDC\_STATIC, 7, 12, 60, 8  
EDITTEXT          IDC\_ADD\_COM\_ADDR, 82, 28, 100, 13, ES\_AUTOHSCROLL  
EDITTEXT          IDC\_ADD\_COM\_PHONE, 82, 46, 100, 13, ES\_AUTOHSCROLL  
EDITTEXT          IDC\_ADD\_COM\_PERSON, 82, 64, 100, 13, ES\_AUTOHSCROLL  
RTEXT             "联系电话", IDC\_STATIC, 7, 48, 60, 8  
RTEXT             "楼盘地址", IDC\_STATIC, 7, 30, 60, 11  
RTEXT             "联系人", IDC\_STATIC, 7, 66, 60, 10

END

IDD\_ADD\_BUI DIALOGEX 0, 0, 267, 136

STYLE DS\_SETFONT | DS\_MODALFRAME | DS\_FIXEDSYS | DS\_CENTER | WS\_POPUP | WS\_CAPTION |  
WS\_SYSMENU

CAPTION "添加楼栋"

FONT 8, "MS Shell Dlg", 400, 0, 0x1

BEGIN

DEFPUSHBUTTON    "确定", IDOK, 157, 115, 50, 14  
PUSHBUTTON       "取消", IDCANCEL, 211, 115, 50, 14  
EDITTEXT          IDC\_ADD\_BUI\_NAME, 82, 9, 100, 13, ES\_AUTOHSCROLL  
RTEXT             "所属楼盘", IDC\_STATIC, 8, 12, 60, 8  
EDITTEXT          IDC\_ADD\_BUI\_NUM, 81, 27, 100, 13, ES\_AUTOHSCROLL | ES\_NUMBER  
EDITTEXT          IDC\_ADD\_BUI\_FLOORS, 81, 46, 100, 13, ES\_AUTOHSCROLL | ES\_NUMBER  
RTEXT             "楼层数", IDC\_STATIC, 8, 47, 60, 8  
RTEXT             "联系电话", IDC\_STATIC, 8, 65, 60, 10  
EDITTEXT          IDC\_ADD\_BUI\_PHONE, 81, 65, 100, 13, ES\_AUTOHSCROLL  
RTEXT             "联系人", IDC\_STATIC, 8, 84, 60, 10  
RTEXT             "楼栋号", IDC\_STATIC, 8, 29, 60, 11  
EDITTEXT          IDC\_ADD\_BUI\_PERSON, 81, 83, 100, 13, ES\_AUTOHSCROLL | ES\_NUMBER

END

IDD\_ADD\_ROOM DIALOGEX 0, 0, 309, 176

STYLE DS\_SETFONT | DS\_MODALFRAME | DS\_FIXEDSYS | DS\_CENTER | WS\_POPUP | WS\_CAPTION |  
WS\_SYSMENU

CAPTION "添加房间"

FONT 8, "MS Shell Dlg", 400, 0, 0x1

BEGIN

DEFPUSHBUTTON    "确定", IDOK, 198, 155, 50, 14  
PUSHBUTTON       "取消", IDCANCEL, 252, 155, 50, 14  
EDITTEXT          IDC\_ADD\_ROOM\_COMNAME, 82, 9, 100, 13, ES\_AUTOHSCROLL  
RTEXT             "所属楼栋号", IDC\_STATIC, 8, 29, 60, 8  
EDITTEXT          IDC\_ADD\_ROOM\_BUINUM, 82, 27, 100, 13, ES\_AUTOHSCROLL | ES\_NUMBER

---

```

EDITTEXT      IDC_ADD_ROOM_ROOMNUM, 82, 45, 100, 13, ES_AUTOHSCROLL | ES_NUMBER
EDITTEXT      IDC_ADD_ROOM_SIZE, 82, 63, 100, 13, ES_AUTOHSCROLL | ES_NUMBER
EDITTEXT      IDC_ADD_ROOM_PRICE, 82, 81, 100, 13, ES_AUTOHSCROLL | ES_NUMBER
RTEXT         "房间面积", IDC_STATIC, 8, 65, 60, 8
RTEXT         "房间号", IDC_STATIC, 8, 47, 60, 11
RTEXT         "价格", IDC_STATIC, 8, 83, 60, 10
RTEXT         "售出状态", IDC_STATIC, 8, 121, 60, 10
LTEXT         "元 / 平方米", IDC_STATIC, 193, 83, 60, 10
LTEXT         "平方米", IDC_STATIC, 193, 65, 60, 10
RTEXT         "所属楼盘", IDC_STATIC, 8, 12, 60, 8
COMBOBOX      IDC_ADD_ROOM_STYLE, 81, 100, 149, 93, CBS_DROPDOWNLIST |
CBS_NOINTEGRALHEIGHT | WS_VSCROLL | WS_TABSTOP
RTEXT         "户型", IDC_STATIC, 8, 102, 60, 10
CONTROL       "已售出", IDC_ADD_ROOM_SOLD, "Button", BS_AUTOCHECKBOX |
WS_TABSTOP, 82, 121, 40, 10
END

```

```

IDD_DEL_COM DIALOGEX 0, 0, 279, 107
STYLE DS_SETFONT | DS_MODALFRAME | DS_FIXEDSYS | DS_CENTER | WS_POPUP | WS_VISIBLE |
WS_CAPTION | WS_SYSMENU
EXSTYLE WS_EX_NOACTIVATE
CAPTION "删除楼盘"
FONT 8, "MS Shell Dlg", 400, 0, 0x1
BEGIN
    DEFPUSHBUTTON "确定", IDOK, 169, 86, 50, 14
    PUSHBUTTON "取消", IDCANCEL, 223, 86, 50, 14
    EDITTEXT IDC_EDIT_DEL_COMNAME, 125, 39, 90, 12, ES_AUTOHSCROLL
    EDITTEXT IDC_EDIT_DEL_COMNUM, 125, 57, 90, 12, ES_AUTOHSCROLL | ES_NUMBER
    CONTROL " 楼盘名称", IDC_RADIO3, "Button", BS_AUTORADIOBUTTON, 73, 40, 50, 10
    CONTROL " 楼盘号", IDC_RADIO4, "Button", BS_AUTORADIOBUTTON, 73, 57, 42, 10
    LTEXT "请输入将要删除的楼盘信息", IDC_STATIC, 90, 20, 97, 8
END

```

```

IDD_DEL_ROOM DIALOGEX 0, 0, 305, 189
STYLE DS_SETFONT | DS_MODALFRAME | DS_FIXEDSYS | DS_CENTER | WS_POPUP | WS_CAPTION |
WS_SYSMENU
CAPTION "s"
FONT 8, "MS Shell Dlg", 400, 0, 0x1
BEGIN
    DEFPUSHBUTTON "确定", IDOK, 198, 168, 50, 14
    PUSHBUTTON "取消", IDCANCEL, 248, 168, 50, 14
    EDITTEXT IDC_EDIT_DEL_COMNAME, 125, 27, 90, 12, ES_AUTOHSCROLL
    EDITTEXT IDC_EDIT_DEL_COMNUM, 125, 45, 90, 12, ES_AUTOHSCROLL | ES_NUMBER
    CONTROL " 楼盘名称", IDC_RADIO3, "Button", BS_AUTORADIOBUTTON, 73, 28, 50, 10

```

---

```

CONTROL      " 楼盘号", IDC_RADIO4, "Button", BS_AUTORADIOBUTTON, 73, 45, 42, 10
CONTROL      "", IDC_STATIC, "Static", SS_BLACKFRAME | WS_BORDER, 43, 75, 212, 1
EDITTEXT     IDC_EDIT_DEL_BUINUM, 125, 87, 90, 12, ES_AUTOHSCROLL | ES_NUMBER
LTEXT        " 请输入将要删除的房间所属楼栋号", IDC_STATIC, 87, 71, 123, 8
CONTROL      "", IDC_STATIC, "Static", SS_BLACKFRAME | WS_BORDER, 43, 14, 212, 1
LTEXT        " 请输入将要删除的房间所属楼盘信息", IDC_STATIC, 81, 11, 131, 8
CONTROL      "", IDC_STATIC, "Static", SS_BLACKFRAME | WS_BORDER, 44, 116, 212, 1
EDITTEXT     IDC_EDIT_DEL_ROOMNUM, 126, 128, 90, 12, ES_AUTOHSCROLL | ES_NUMBER
LTEXT        " 请输入将要删除的房间号", IDC_STATIC, 102, 112, 91, 8
LTEXT        "楼栋号", IDC_STATIC, 86, 89, 25, 8
LTEXT        "房间号", IDC_STATIC, 86, 130, 25, 8

```

```
END
```

```
IDD_DEL_BUI DIALOGEX 0, 0, 301, 165
```

```
STYLE DS_SETFONT | DS_MODALFRAME | DS_FIXEDSYS | DS_CENTER | WS_POPUP | WS_CAPTION |
WS_SYSMENU
```

```
CAPTION "删除楼栋"
```

```
FONT 8, "MS Shell Dlg", 400, 0, 0x1
```

```
BEGIN
```

```

DEFPUSHBUTTON "确定", IDOK, 191, 144, 50, 14
PUSHBUTTON    "取消", IDCANCEL, 245, 144, 50, 14
EDITTEXT      IDC_EDIT_DEL_COMNAME, 123, 33, 90, 12, ES_AUTOHSCROLL
EDITTEXT      IDC_EDIT_DEL_COMNUM, 123, 51, 90, 12, ES_AUTOHSCROLL | ES_NUMBER
CONTROL       " 楼盘名称", IDC_RADIO3, "Button", BS_AUTORADIOBUTTON, 71, 34, 50, 10
CONTROL       " 楼盘号", IDC_RADIO4, "Button", BS_AUTORADIOBUTTON, 71, 51, 42, 10
CONTROL       "", IDC_STATIC, "Static", SS_BLACKFRAME | WS_BORDER, 41, 83, 212, 1
EDITTEXT      IDC_EDIT_DEL_BUINUM, 123, 98, 90, 12, ES_AUTOHSCROLL | ES_NUMBER
LTEXT         " 请输入将要删除的楼栋号", IDC_STATIC, 100, 80, 91, 8
CONTROL       "", IDC_STATIC, "Static", SS_BLACKFRAME | WS_BORDER, 41, 20, 212, 1
LTEXT         " 请输入将要删除的楼栋所属楼盘信息", IDC_STATIC, 79, 17, 131, 8
LTEXT         "楼栋号", IDC_STATIC, 84, 100, 25, 8

```

```
END
```

```
IDD_COMEDIT DIALOGEX 0, 0, 289, 121
```

```
STYLE DS_SETFONT | DS_MODALFRAME | DS_FIXEDSYS | DS_CENTER | WS_POPUP | WS_CAPTION |
WS_SYSMENU
```

```
CAPTION "修改楼盘信息"
```

```
FONT 8, "MS Shell Dlg", 400, 0, 0x1
```

```
BEGIN
```

```

DEFPUSHBUTTON "确定", IDOK, 179, 100, 50, 14
PUSHBUTTON    "取消", IDCANCEL, 233, 100, 50, 14
EDITTEXT      IDC_ADD_COM_NAME, 82, 9, 100, 13, ES_AUTOHSCROLL
RTEXT         "楼盘名", IDC_STATIC, 7, 12, 60, 8
EDITTEXT      IDC_ADD_COM_ADDR, 82, 29, 100, 13, ES_AUTOHSCROLL

```

---

```

EDITTEXT      IDC_ADD_COM_PHONE, 82, 47, 100, 13, ES_AUTOHSCROLL | ES_NUMBER
EDITTEXT      IDC_ADD_COM_PERSON, 82, 65, 100, 13, ES_AUTOHSCROLL
RTEXT         "联系电话", IDC_STATIC, 7, 49, 60, 8
RTEXT         "楼盘地址", IDC_STATIC, 7, 31, 60, 11
RTEXT         "联系人", IDC_STATIC, 7, 67, 60, 10
END

IDD_BUIEDIT DIALOGEX 0, 0, 273, 126
STYLE DS_SETFONT | DS_MODALFRAME | DS_FIXEDSYS | DS_CENTER | WS_POPUP | WS_CAPTION |
WS_SYSMENU
CAPTION "编辑楼栋信息"
FONT 8, "MS Shell Dlg", 400, 0, 0x1
BEGIN
    DEFPUSHBUTTON "确定", IDOK, 162, 106, 50, 14
    PUSHBUTTON    "取消", IDCANCEL, 216, 106, 50, 14
    EDITTEXT      IDC_ADD_BUI_NUM, 82, 9, 100, 13, ES_AUTOHSCROLL
    RTEXT         "楼栋号", IDC_STATIC, 8, 12, 60, 8
    EDITTEXT      IDC_ADD_BUI_FLOORS, 82, 29, 100, 13, ES_AUTOHSCROLL | ES_NUMBER
    EDITTEXT      IDC_ADD_BUI_PHONE, 82, 48, 100, 13, ES_AUTOHSCROLL | ES_NUMBER
    RTEXT         "楼层数", IDC_STATIC, 8, 31, 60, 8
    RTEXT         "联系电话", IDC_STATIC, 8, 50, 60, 10
    EDITTEXT      IDC_ADD_BUI_PERSON, 81, 68, 100, 13, ES_AUTOHSCROLL
    RTEXT         "联系人", IDC_STATIC, 8, 70, 60, 10
END

IDD_ROOMEDIT DIALOGEX 0, 0, 285, 138
STYLE DS_SETFONT | DS_MODALFRAME | DS_FIXEDSYS | DS_CENTER | WS_POPUP | WS_CAPTION |
WS_SYSMENU
CAPTION "编辑房间信息"
FONT 8, "MS Shell Dlg", 400, 0, 0x1
BEGIN
    DEFPUSHBUTTON "确定", IDOK, 176, 117, 50, 14
    PUSHBUTTON    "取消", IDCANCEL, 228, 117, 50, 14
    EDITTEXT      IDC_ADD_ROOM_ROOMNUM, 82, 10, 100, 13, ES_AUTOHSCROLL | ES_NUMBER
    EDITTEXT      IDC_ADD_ROOM_SIZE, 82, 28, 100, 13, ES_AUTOHSCROLL | ES_NUMBER
    EDITTEXT      IDC_ADD_ROOM_PRICE, 82, 46, 100, 13, ES_AUTOHSCROLL | ES_NUMBER
    RTEXT         "房间面积", IDC_STATIC, 8, 30, 60, 8
    RTEXT         "房间号", IDC_STATIC, 8, 12, 60, 11
    RTEXT         "价格", IDC_STATIC, 8, 48, 60, 10
    RTEXT         "售出状态", IDC_STATIC, 8, 86, 60, 10
    COMBOBOX      IDC_ADD_ROOM_STYLE, 81, 65, 149, 93, CBS_DROPDOWNLIST |
CBS_NOINTEGRALHEIGHT | WS_VSCROLL | WS_TABSTOP
    RTEXT         "户型", IDC_STATIC, 8, 67, 60, 10
    CONTROL       "已售出", IDC_ADD_ROOM_SOLD, "Button", BS_AUTOCHECKBOX |

```

---

WS\_TABSTOP, 82, 86, 40, 10

END

IDD\_SEARCH\_DIALOGEX 0, 0, 415, 238

STYLE DS\_SETFONT | DS\_MODALFRAME | DS\_FIXEDSYS | DS\_CENTER | WS\_POPUP | WS\_CAPTION |  
WS\_SYSMENU

CAPTION “搜索”

FONT 8, “MS Shell Dlg”, 400, 0, 0x1

BEGIN

DEFPUSHBUTTON “确定”, IDOK, 303, 218, 50, 14

PUSHBUTTON “取消”, IDCANCEL, 358, 218, 50, 14

CONTROL “”, IDC\_STATIC, “Static”, SS\_BLACKFRAME | WS\_BORDER, 43, 67, 343, 1

LTEXT “楼栋信息”, IDC\_STATIC, 202, 63, 35, 8

CONTROL “”, IDC\_STATIC, “Static”, SS\_BLACKFRAME | WS\_BORDER, 43, 14, 341, 1

LTEXT “楼盘信息”, IDC\_STATIC, 201, 10, 35, 8

CONTROL “”, IDC\_STATIC, “Static”, SS\_BLACKFRAME | WS\_BORDER, 44, 119, 344, 1

LTEXT “房间信息”, IDC\_STATIC, 202, 115, 35, 8

CONTROL “搜索楼盘”, IDC\_SCOM, “Button”, BS\_AUTORADIOBUTTON, 11, 36, 48, 10

CONTROL “搜索楼栋”, IDC\_SBUI, “Button”, BS\_AUTORADIOBUTTON, 11, 89, 48, 10

CONTROL “搜索房间”, IDC\_SROOM, “Button”, BS\_AUTORADIOBUTTON, 11, 149, 48, 10

EDITTEXT IDC\_S\_COMNAME, 125, 27, 90, 12, ES\_AUTOHSCROLL

EDITTEXT IDC\_S\_COMNUM, 279, 27, 90, 12, ES\_AUTOHSCROLL | ES\_NUMBER

LTEXT “楼盘名称”, IDC\_STATIC, 84, 29, 33, 8

LTEXT “楼盘编号”, IDC\_STATIC, 237, 28, 33, 8

EDITTEXT IDC\_S\_COMPERSON, 125, 44, 90, 12, ES\_AUTOHSCROLL

LTEXT “联系人”, IDC\_STATIC, 91, 45, 25, 8

EDITTEXT IDC\_S\_BUINUM, 125, 79, 90, 12, ES\_AUTOHSCROLL | ES\_NUMBER

LTEXT “楼栋编号”, IDC\_STATIC, 83, 81, 33, 8

EDITTEXT IDC\_S\_COMPHONE, 279, 44, 90, 12, ES\_AUTOHSCROLL | ES\_NUMBER

LTEXT “联系电话”, IDC\_STATIC, 237, 45, 33, 8

EDITTEXT IDC\_S\_BUIPERSON, 126, 96, 90, 12, ES\_AUTOHSCROLL

LTEXT “联系人”, IDC\_STATIC, 92, 97, 25, 8

EDITTEXT IDC\_S\_BUIPHONE, 280, 79, 90, 12, ES\_AUTOHSCROLL | ES\_NUMBER

LTEXT “联系电话”, IDC\_STATIC, 238, 80, 33, 8

EDITTEXT IDC\_S\_ROOMNUM, 127, 129, 90, 12, ES\_AUTOHSCROLL | ES\_NUMBER

LTEXT “房间编号”, IDC\_STATIC, 85, 131, 33, 8

EDITTEXT IDC\_S\_ROOMSIZE, 281, 129, 90, 12, ES\_AUTOHSCROLL | ES\_NUMBER

LTEXT “房间面积”, IDC\_STATIC, 239, 131, 33, 8

EDITTEXT IDC\_S\_LOPRICE, 144, 162, 54, 12, ES\_AUTOHSCROLL | ES\_NUMBER

LTEXT “元 / 平方米”, IDC\_STATIC, 275, 164, 40, 8

LTEXT “—”, IDC\_STATIC, 203, 164, 8, 8

EDITTEXT IDC\_S\_HIPRICE, 214, 162, 54, 12, ES\_AUTOHSCROLL | ES\_NUMBER

LTEXT “房间价格范围”, IDC\_STATIC, 85, 164, 49, 8

LTEXT “户型”, IDC\_STATIC, 85, 181, 17, 8

---

```

        COMBOBOX        IDC_S_ROOMTYPE, 127, 180, 140, 96, CBS_DROPDOWNLIST | WS_VSCROLL |
WS_TABSTOP
        EDITTEXT        IDC_S_ROOMFLOOR, 127, 145, 90, 12, ES_AUTOHSCROLL | ES_NUMBER
        LTEXT           "所在层数", IDC_STATIC, 85, 147, 33, 8
        EDITTEXT        IDC_S_ALLPRICE, 281, 145, 57, 12, ES_AUTOHSCROLL | ES_NUMBER
        LTEXT           "总价上限", IDC_STATIC, 239, 147, 33, 8
        LTEXT           "（万元）", IDC_STATIC, 342, 147, 33, 8
        CONTROL         "已售出", IDC_S_SOLD, "Button", BS_AUTOCHECKBOX |
WS_TABSTOP, 281, 181, 40, 10
END

```

```

////////////////////////////////////
//
// DESIGNINFO
//

```

```

#ifdef APSTUDIO_INVOKED
GUIDELINES DESIGNINFO
BEGIN
    IDD_ABOUTBOX, DIALOG
    BEGIN
        LEFTMARGIN, 7
        RIGHTMARGIN, 304
        VERTGUIDE, 154
        TOPMARGIN, 7
        BOTTOMMARGIN, 170
    END

    IDD_ADD_COM, DIALOG
    BEGIN
        LEFTMARGIN, 7
        RIGHTMARGIN, 244
        VERTGUIDE, 79
        VERTGUIDE, 132
        VERTGUIDE, 223
        TOPMARGIN, 7
        BOTTOMMARGIN, 114
        HORZGUIDE, 25
        HORZGUIDE, 42
        HORZGUIDE, 60
        HORZGUIDE, 77
        HORZGUIDE, 97
        HORZGUIDE, 116
    END

```

---

END

IDD\_ADD\_BUI, DIALOG  
BEGIN  
    LEFTMARGIN, 7  
    RIGHTMARGIN, 260  
    TOPMARGIN, 7  
    BOTTOMMARGIN, 129  
END

IDD\_ADD\_ROOM, DIALOG  
BEGIN  
    LEFTMARGIN, 7  
    RIGHTMARGIN, 302  
    TOPMARGIN, 7  
    BOTTOMMARGIN, 169  
END

IDD\_DEL\_COM, DIALOG  
BEGIN  
    LEFTMARGIN, 7  
    RIGHTMARGIN, 272  
    TOPMARGIN, 7  
    BOTTOMMARGIN, 100  
    HORZGUIDE, 40  
END

IDD\_DEL\_ROOM, DIALOG  
BEGIN  
    LEFTMARGIN, 7  
    RIGHTMARGIN, 298  
    VERTGUIDE, 85  
    TOPMARGIN, 7  
    BOTTOMMARGIN, 182  
END

IDD\_DEL\_BUI, DIALOG  
BEGIN  
    LEFTMARGIN, 7  
    RIGHTMARGIN, 294  
    TOPMARGIN, 7  
    BOTTOMMARGIN, 158  
END



---

```
IDD_COMEDIT, DIALOG
BEGIN
    LEFTMARGIN, 7
    RIGHTMARGIN, 282
    TOPMARGIN, 7
    BOTTOMMARGIN, 114
END

IDD_BUIEDIT, DIALOG
BEGIN
    LEFTMARGIN, 7
    RIGHTMARGIN, 266
    TOPMARGIN, 7
    BOTTOMMARGIN, 119
END

IDD_ROOMEDIT, DIALOG
BEGIN
    LEFTMARGIN, 7
    RIGHTMARGIN, 278
    TOPMARGIN, 7
    BOTTOMMARGIN, 131
END

IDD_SEARCH, DIALOG
BEGIN
    LEFTMARGIN, 7
    RIGHTMARGIN, 408
    VERTGUIDE, 11
    VERTGUIDE, 202
    TOPMARGIN, 7
    BOTTOMMARGIN, 231
END

#endif    // APSTUDIO_INVOKED
```

```
////////////////////////////////////
//
// AFX_DIALOG_LAYOUT
//
```

```
IDD_ABOUTBOX AFX_DIALOG_LAYOUT
BEGIN
```

---

```
    0
END
```

```
IDD_ADD_COM AFX_DIALOG_LAYOUT
BEGIN
    0
END
```

```
IDD_ADD_BUI AFX_DIALOG_LAYOUT
BEGIN
    0
END
```

```
IDD_ADD_ROOM AFX_DIALOG_LAYOUT
BEGIN
    0
END
```

```
IDD_DEL_COM AFX_DIALOG_LAYOUT
BEGIN
    0
END
```

```
IDD_DEL_ROOM AFX_DIALOG_LAYOUT
BEGIN
    0
END
```

```
IDD_DEL_BUI AFX_DIALOG_LAYOUT
BEGIN
    0
END
```

```
IDD_COMEDIT AFX_DIALOG_LAYOUT
BEGIN
    0
END
```

```
IDD_BUIEDIT AFX_DIALOG_LAYOUT
BEGIN
    0
END
```

```
IDD_ROOMEDIT AFX_DIALOG_LAYOUT
```

---

```
BEGIN
```

```
    0
```

```
END
```

```
IDD_SEARCH AFX_DIALOG_LAYOUT
```

```
BEGIN
```

```
    0
```

```
END
```

```
////////////////////////////////////
```

```
//
```

```
// Icon
```

```
//
```

```
// Icon with lowest ID value placed first to ensure application icon
```

```
// remains consistent on all systems.
```

```
IDI_MainIcon          ICON          "icon1.ico"
```

```
////////////////////////////////////
```

```
//
```

```
// Dialog Info
```

```
//
```

```
IDD_ADD_ROOM DLGINIT
```

```
BEGIN
```

```
    IDC_ADD_ROOM_STYLE, 0x403, 24, 0
```

```
0xa7bb, 0xcdd0, 0xa331, 0xd2ba, 0xcabb, 0xd2d2, 0xccbb, 0xd2fc, 0xb3bb,  
0xd2f8, 0xb2bb, 0x00de,
```

```
    IDC_ADD_ROOM_STYLE, 0x403, 24, 0
```

```
0xa7bb, 0xcdd0, 0xa332, 0xc1ba, 0xcabd, 0xd2d2, 0xccbb, 0xd2fc, 0xb3bb,  
0xd2f8, 0xb2bb, 0x00de,
```

```
    IDC_ADD_ROOM_STYLE, 0x403, 24, 0
```

```
0xa7bb, 0xcdd0, 0xa333, 0xc8ba, 0xcafd, 0xd2d2, 0xccbb, 0xd2fc, 0xb3bb,  
0xd2f8, 0xb2bb, 0x00de,
```

```
    IDC_ADD_ROOM_STYLE, 0x403, 24, 0
```

```
0xa7bb, 0xcdd0, 0xa334, 0xcbba, 0xcac4, 0xd2d2, 0xccbb, 0xd2fc, 0xb3bb,  
0xd2f8, 0xb2bb, 0x00de,
```

```
    IDC_ADD_ROOM_STYLE, 0x403, 24, 0
```

```
0xa7bb, 0xcdd0, 0xa335, 0xceba, 0xcae5, 0xd2d2, 0xccbb, 0xd2fc, 0xb3bb,  
0xd2f8, 0xb2bb, 0x00de,
```

```
    IDC_ADD_ROOM_STYLE, 0x403, 24, 0
```

```
0xa7bb, 0xcdd0, 0xa336, 0xd2ba, 0xcabb, 0xc1d2, 0xccbd, 0xd2fc, 0xb3bb,
```

---

```

0xd2f8, 0xb2bb, 0x00de,
    IDC_ADD_ROOM_STYLE, 0x403, 24, 0
0xa7bb, 0xcdd0, 0xa337, 0xc1ba, 0xcabd, 0xc1d2, 0xccbd, 0xd2fc, 0xb3bb,
0xd2f8, 0xb2bb, 0x00de,
    IDC_ADD_ROOM_STYLE, 0x403, 24, 0
0xa7bb, 0xcdd0, 0xa338, 0xc8ba, 0xcafd, 0xc1d2, 0xccbd, 0xd2fc, 0xb3bb,
0xd2f8, 0xb2bb, 0x00de,
    IDC_ADD_ROOM_STYLE, 0x403, 24, 0
0xa7bb, 0xcdd0, 0xa339, 0xcbba, 0xcac4, 0xc1d2, 0xccbd, 0xd2fc, 0xb3bb,
0xd2f8, 0xb2bb, 0x00de,
    IDC_ADD_ROOM_STYLE, 0x403, 25, 0
0xa7bb, 0xcdd0, 0x3031, 0xbaa3, 0xe5ce, 0xd2ca, 0xbdc1, 0xfccc, 0xbbd2,
0xf8b3, 0xbbd2, 0xdeb2, "\000"
    IDC_ADD_ROOM_STYLE, 0x403, 25, 0
0xa7bb, 0xcdd0, 0x3131, 0xbaa3, 0xbdc1, 0xd2ca, 0xbbd2, 0xfccc, 0xbdc1,
0xf8b3, 0xbbd2, 0xdeb2, "\000"
    IDC_ADD_ROOM_STYLE, 0x403, 25, 0
0xa7bb, 0xcdd0, 0x3231, 0xbaa3, 0xfdc8, 0xd2ca, 0xbbd2, 0xfccc, 0xbdc1,
0xf8b3, 0xbbd2, 0xdeb2, "\000"
    IDC_ADD_ROOM_STYLE, 0x403, 25, 0
0xa7bb, 0xcdd0, 0x3331, 0xbaa3, 0xc4cb, 0xd2ca, 0xbbd2, 0xfccc, 0xbdc1,
0xf8b3, 0xbbd2, 0xdeb2, "\000"
    IDC_ADD_ROOM_STYLE, 0x403, 25, 0
0xa7bb, 0xcdd0, 0x3431, 0xbaa3, 0xe5ce, 0xd2ca, 0xbbd2, 0xfccc, 0xbdc1,
0xf8b3, 0xbbd2, 0xdeb2, "\000"
    IDC_ADD_ROOM_STYLE, 0x403, 25, 0
0xa7bb, 0xcdd0, 0x3531, 0xbaa3, 0xbdc1, 0xd2ca, 0xbbd2, 0xfccc, 0xbbd2,
0xf8b3, 0xbdc1, 0xdeb2, "\000"
    IDC_ADD_ROOM_STYLE, 0x403, 25, 0
0xa7bb, 0xcdd0, 0x3631, 0xbaa3, 0xfdc8, 0xd2ca, 0xbbd2, 0xfccc, 0xbbd2,
0xf8b3, 0xbdc1, 0xdeb2, "\000"
    IDC_ADD_ROOM_STYLE, 0x403, 25, 0
0xa7bb, 0xcdd0, 0x3731, 0xbaa3, 0xc4cb, 0xd2ca, 0xbbd2, 0xfccc, 0xbbd2,
0xf8b3, 0xbdc1, 0xdeb2, "\000"
    IDC_ADD_ROOM_STYLE, 0x403, 25, 0
0xa7bb, 0xcdd0, 0x3831, 0xbaa3, 0xe5ce, 0xd2ca, 0xbbd2, 0xfccc, 0xbbd2,
0xf8b3, 0xbdc1, 0xdeb2, "\000"
    IDC_ADD_ROOM_STYLE, 0x403, 25, 0
0xa7bb, 0xcdd0, 0x3931, 0xbaa3, 0xbdc1, 0xd2ca, 0xbdc1, 0xfccc, 0xbdc1,
0xf8b3, 0xbbd2, 0xdeb2, "\000"
    IDC_ADD_ROOM_STYLE, 0x403, 25, 0
0xa7bb, 0xcdd0, 0x3032, 0xbaa3, 0xfdc8, 0xd2ca, 0xbdc1, 0xfccc, 0xbdc1,
0xf8b3, 0xbbd2, 0xdeb2, "\000"
    IDC_ADD_ROOM_STYLE, 0x403, 25, 0

```

---

```

0xa7bb, 0xcdd0, 0x3132, 0xbaa3, 0xc4cb, 0xd2ca, 0xbdc1, 0xfccc, 0xbdc1,
0xf8b3, 0xbbd2, 0xdeb2, "\000"
    IDC_ADD_ROOM_STYLE, 0x403, 25, 0
0xa7bb, 0xcdd0, 0x3232, 0xbaa3, 0xe5ce, 0xd2ca, 0xbdc1, 0xfccc, 0xbdc1,
0xf8b3, 0xbbd2, 0xdeb2, "\000"
    IDC_ADD_ROOM_STYLE, 0x403, 25, 0
0xa7bb, 0xcdd0, 0x3332, 0xbaa3, 0xbdc1, 0xd2ca, 0xbdc1, 0xfccc, 0xbbd2,
0xf8b3, 0xbdc1, 0xdeb2, "\000"
    IDC_ADD_ROOM_STYLE, 0x403, 25, 0
0xa7bb, 0xcdd0, 0x3432, 0xbaa3, 0xfdc8, 0xd2ca, 0xbdc1, 0xfccc, 0xbbd2,
0xf8b3, 0xbdc1, 0xdeb2, "\000"
    IDC_ADD_ROOM_STYLE, 0x403, 25, 0
0xa7bb, 0xcdd0, 0x3532, 0xbaa3, 0xc4cb, 0xd2ca, 0xbdc1, 0xfccc, 0xbbd2,
0xf8b3, 0xbdc1, 0xdeb2, "\000"
    IDC_ADD_ROOM_STYLE, 0x403, 25, 0
0xa7bb, 0xcdd0, 0x3632, 0xbaa3, 0xe5ce, 0xd2ca, 0xbdc1, 0xfccc, 0xbbd2,
0xf8b3, 0xbdc1, 0xdeb2, "\000"
    IDC_ADD_ROOM_STYLE, 0x403, 25, 0
0xa7bb, 0xcdd0, 0x3732, 0xbaa3, 0xbdc1, 0xd2ca, 0xbbd2, 0xfccc, 0xbdc1,
0xf8b3, 0xbdc1, 0xdeb2, "\000"
    IDC_ADD_ROOM_STYLE, 0x403, 25, 0
0xa7bb, 0xcdd0, 0x3832, 0xbaa3, 0xfdc8, 0xd2ca, 0xbbd2, 0xfccc, 0xbdc1,
0xf8b3, 0xbdc1, 0xdeb2, "\000"
    IDC_ADD_ROOM_STYLE, 0x403, 25, 0
0xa7bb, 0xcdd0, 0x3932, 0xbaa3, 0xc4cb, 0xd2ca, 0xbbd2, 0xfccc, 0xbdc1,
0xf8b3, 0xbdc1, 0xdeb2, "\000"
    IDC_ADD_ROOM_STYLE, 0x403, 25, 0
0xa7bb, 0xcdd0, 0x3033, 0xbaa3, 0xe5ce, 0xd2ca, 0xbbd2, 0xfccc, 0xbdc1,
0xf8b3, 0xbdc1, 0xdeb2, "\000"
    IDC_ADD_ROOM_STYLE, 0x403, 25, 0
0xa7bb, 0xcdd0, 0x3133, 0xbaa3, 0xbdc1, 0xd2ca, 0xbdc1, 0xfccc, 0xbdc1,
0xf8b3, 0xbdc1, 0xdeb2, "\000"
    IDC_ADD_ROOM_STYLE, 0x403, 25, 0
0xa7bb, 0xcdd0, 0x3233, 0xbaa3, 0xfdc8, 0xd2ca, 0xbdc1, 0xfccc, 0xbdc1,
0xf8b3, 0xbdc1, 0xdeb2, "\000"
    IDC_ADD_ROOM_STYLE, 0x403, 25, 0
0xa7bb, 0xcdd0, 0x3333, 0xbaa3, 0xc4cb, 0xd2ca, 0xbdc1, 0xfccc, 0xbdc1,
0xf8b3, 0xbdc1, 0xdeb2, "\000"
    IDC_ADD_ROOM_STYLE, 0x403, 25, 0
0xa7bb, 0xcdd0, 0x3433, 0xbaa3, 0xe5ce, 0xd2ca, 0xbdc1, 0xfccc, 0xbdc1,
0xf8b3, 0xbdc1, 0xdeb2, "\000"
0
END

```

---

IDD\_ROOMEDIT DLGINIT

BEGIN

    IDC\_ADD\_ROOM\_STYLE, 0x403, 24, 0  
0xa7bb, 0xcdd0, 0xa331, 0xd2ba, 0xcabb, 0xd2d2, 0xccbb, 0xd2fc, 0xb3bb,  
0xd2f8, 0xb2bb, 0x00de,

    IDC\_ADD\_ROOM\_STYLE, 0x403, 24, 0  
0xa7bb, 0xcdd0, 0xa332, 0xc1ba, 0xcabd, 0xd2d2, 0xccbb, 0xd2fc, 0xb3bb,  
0xd2f8, 0xb2bb, 0x00de,

    IDC\_ADD\_ROOM\_STYLE, 0x403, 24, 0  
0xa7bb, 0xcdd0, 0xa333, 0xc8ba, 0xcafd, 0xd2d2, 0xccbb, 0xd2fc, 0xb3bb,  
0xd2f8, 0xb2bb, 0x00de,

    IDC\_ADD\_ROOM\_STYLE, 0x403, 24, 0  
0xa7bb, 0xcdd0, 0xa334, 0xcbba, 0xcac4, 0xd2d2, 0xccbb, 0xd2fc, 0xb3bb,  
0xd2f8, 0xb2bb, 0x00de,

    IDC\_ADD\_ROOM\_STYLE, 0x403, 24, 0  
0xa7bb, 0xcdd0, 0xa335, 0xceba, 0xcae5, 0xd2d2, 0xccbb, 0xd2fc, 0xb3bb,  
0xd2f8, 0xb2bb, 0x00de,

    IDC\_ADD\_ROOM\_STYLE, 0x403, 24, 0  
0xa7bb, 0xcdd0, 0xa336, 0xd2ba, 0xcabb, 0xc1d2, 0xccbd, 0xd2fc, 0xb3bb,  
0xd2f8, 0xb2bb, 0x00de,

    IDC\_ADD\_ROOM\_STYLE, 0x403, 24, 0  
0xa7bb, 0xcdd0, 0xa337, 0xc1ba, 0xcabd, 0xc1d2, 0xccbd, 0xd2fc, 0xb3bb,  
0xd2f8, 0xb2bb, 0x00de,

    IDC\_ADD\_ROOM\_STYLE, 0x403, 24, 0  
0xa7bb, 0xcdd0, 0xa338, 0xc8ba, 0xcafd, 0xc1d2, 0xccbd, 0xd2fc, 0xb3bb,  
0xd2f8, 0xb2bb, 0x00de,

    IDC\_ADD\_ROOM\_STYLE, 0x403, 24, 0  
0xa7bb, 0xcdd0, 0xa339, 0xcbba, 0xcac4, 0xc1d2, 0xccbd, 0xd2fc, 0xb3bb,  
0xd2f8, 0xb2bb, 0x00de,

    IDC\_ADD\_ROOM\_STYLE, 0x403, 25, 0  
0xa7bb, 0xcdd0, 0x3031, 0xbaa3, 0xe5ce, 0xd2ca, 0xbdc1, 0xfccc, 0xbbd2,  
0xf8b3, 0xbbd2, 0xdeb2, "\000"

    IDC\_ADD\_ROOM\_STYLE, 0x403, 25, 0  
0xa7bb, 0xcdd0, 0x3131, 0xbaa3, 0xbdc1, 0xd2ca, 0xbbd2, 0xfccc, 0xbdc1,  
0xf8b3, 0xbbd2, 0xdeb2, "\000"

    IDC\_ADD\_ROOM\_STYLE, 0x403, 25, 0  
0xa7bb, 0xcdd0, 0x3231, 0xbaa3, 0xfdc8, 0xd2ca, 0xbbd2, 0xfccc, 0xbdc1,  
0xf8b3, 0xbbd2, 0xdeb2, "\000"

    IDC\_ADD\_ROOM\_STYLE, 0x403, 25, 0  
0xa7bb, 0xcdd0, 0x3331, 0xbaa3, 0xc4cb, 0xd2ca, 0xbbd2, 0xfccc, 0xbdc1,  
0xf8b3, 0xbbd2, 0xdeb2, "\000"

    IDC\_ADD\_ROOM\_STYLE, 0x403, 25, 0  
0xa7bb, 0xcdd0, 0x3431, 0xbaa3, 0xe5ce, 0xd2ca, 0xbbd2, 0xfccc, 0xbdc1,  
0xf8b3, 0xbbd2, 0xdeb2, "\000"

---

```

IDC_ADD_ROOM_STYLE, 0x403, 25, 0
0xa7bb, 0xcdd0, 0x3531, 0xbaa3, 0xbdc1, 0xd2ca, 0xbbd2, 0xfccc, 0xbbd2,
0xf8b3, 0xbdc1, 0xdeb2, "\000"

IDC_ADD_ROOM_STYLE, 0x403, 25, 0
0xa7bb, 0xcdd0, 0x3631, 0xbaa3, 0xfdc8, 0xd2ca, 0xbbd2, 0xfccc, 0xbbd2,
0xf8b3, 0xbdc1, 0xdeb2, "\000"

IDC_ADD_ROOM_STYLE, 0x403, 25, 0
0xa7bb, 0xcdd0, 0x3731, 0xbaa3, 0xc4cb, 0xd2ca, 0xbbd2, 0xfccc, 0xbbd2,
0xf8b3, 0xbdc1, 0xdeb2, "\000"

IDC_ADD_ROOM_STYLE, 0x403, 25, 0
0xa7bb, 0xcdd0, 0x3831, 0xbaa3, 0xe5ce, 0xd2ca, 0xbbd2, 0xfccc, 0xbbd2,
0xf8b3, 0xbdc1, 0xdeb2, "\000"

IDC_ADD_ROOM_STYLE, 0x403, 25, 0
0xa7bb, 0xcdd0, 0x3931, 0xbaa3, 0xbdc1, 0xd2ca, 0xbdc1, 0xfccc, 0xbdc1,
0xf8b3, 0xbbd2, 0xdeb2, "\000"

IDC_ADD_ROOM_STYLE, 0x403, 25, 0
0xa7bb, 0xcdd0, 0x3032, 0xbaa3, 0xfdc8, 0xd2ca, 0xbdc1, 0xfccc, 0xbdc1,
0xf8b3, 0xbbd2, 0xdeb2, "\000"

IDC_ADD_ROOM_STYLE, 0x403, 25, 0
0xa7bb, 0xcdd0, 0x3132, 0xbaa3, 0xc4cb, 0xd2ca, 0xbdc1, 0xfccc, 0xbdc1,
0xf8b3, 0xbbd2, 0xdeb2, "\000"

IDC_ADD_ROOM_STYLE, 0x403, 25, 0
0xa7bb, 0xcdd0, 0x3232, 0xbaa3, 0xe5ce, 0xd2ca, 0xbdc1, 0xfccc, 0xbdc1,
0xf8b3, 0xbbd2, 0xdeb2, "\000"

IDC_ADD_ROOM_STYLE, 0x403, 25, 0
0xa7bb, 0xcdd0, 0x3332, 0xbaa3, 0xbdc1, 0xd2ca, 0xbdc1, 0xfccc, 0xbbd2,
0xf8b3, 0xbdc1, 0xdeb2, "\000"

IDC_ADD_ROOM_STYLE, 0x403, 25, 0
0xa7bb, 0xcdd0, 0x3432, 0xbaa3, 0xfdc8, 0xd2ca, 0xbdc1, 0xfccc, 0xbbd2,
0xf8b3, 0xbdc1, 0xdeb2, "\000"

IDC_ADD_ROOM_STYLE, 0x403, 25, 0
0xa7bb, 0xcdd0, 0x3532, 0xbaa3, 0xc4cb, 0xd2ca, 0xbdc1, 0xfccc, 0xbbd2,
0xf8b3, 0xbdc1, 0xdeb2, "\000"

IDC_ADD_ROOM_STYLE, 0x403, 25, 0
0xa7bb, 0xcdd0, 0x3632, 0xbaa3, 0xe5ce, 0xd2ca, 0xbdc1, 0xfccc, 0xbbd2,
0xf8b3, 0xbdc1, 0xdeb2, "\000"

IDC_ADD_ROOM_STYLE, 0x403, 25, 0
0xa7bb, 0xcdd0, 0x3732, 0xbaa3, 0xbdc1, 0xd2ca, 0xbbd2, 0xfccc, 0xbdc1,
0xf8b3, 0xbdc1, 0xdeb2, "\000"

IDC_ADD_ROOM_STYLE, 0x403, 25, 0
0xa7bb, 0xcdd0, 0x3832, 0xbaa3, 0xfdc8, 0xd2ca, 0xbbd2, 0xfccc, 0xbdc1,
0xf8b3, 0xbdc1, 0xdeb2, "\000"

IDC_ADD_ROOM_STYLE, 0x403, 25, 0
0xa7bb, 0xcdd0, 0x3932, 0xbaa3, 0xc4cb, 0xd2ca, 0xbbd2, 0xfccc, 0xbdc1,

```

---

```

0xf8b3, 0xbdc1, 0xdeb2, "\000"
    IDC_ADD_ROOM_STYLE, 0x403, 25, 0
0xa7bb, 0xcdd0, 0x3033, 0xbaa3, 0xe5ce, 0xd2ca, 0xbbd2, 0xfccc, 0xbdc1,
0xf8b3, 0xbdc1, 0xdeb2, "\000"
    IDC_ADD_ROOM_STYLE, 0x403, 25, 0
0xa7bb, 0xcdd0, 0x3133, 0xbaa3, 0xbdc1, 0xd2ca, 0xbdc1, 0xfccc, 0xbdc1,
0xf8b3, 0xbdc1, 0xdeb2, "\000"
    IDC_ADD_ROOM_STYLE, 0x403, 25, 0
0xa7bb, 0xcdd0, 0x3233, 0xbaa3, 0xfdc8, 0xd2ca, 0xbdc1, 0xfccc, 0xbdc1,
0xf8b3, 0xbdc1, 0xdeb2, "\000"
    IDC_ADD_ROOM_STYLE, 0x403, 25, 0
0xa7bb, 0xcdd0, 0x3333, 0xbaa3, 0xc4cb, 0xd2ca, 0xbdc1, 0xfccc, 0xbdc1,
0xf8b3, 0xbdc1, 0xdeb2, "\000"
    IDC_ADD_ROOM_STYLE, 0x403, 25, 0
0xa7bb, 0xcdd0, 0x3433, 0xbaa3, 0xe5ce, 0xd2ca, 0xbdc1, 0xfccc, 0xbdc1,
0xf8b3, 0xbdc1, 0xdeb2, "\000"
    0
END

#endif    // 中文(简体, 中国) resources
//////////

#ifndef APSTUDIO_INVOKED
//////////
//
// Generated from the TEXTINCLUDE 3 resource.
//
//////////
#endif    // not APSTUDIO_INVOKED

```



---

## 参考文献

- [1] MSDN, Windows API 官方文档: ReadFile Function, URL:  
[https://msdn.microsoft.com/en-us/library/windows/desktop/aa365467\(v=vs.85\).aspx](https://msdn.microsoft.com/en-us/library/windows/desktop/aa365467(v=vs.85).aspx)
- [2] MSDN, Windows API 官方文档: SetFilePointer Function, URL:  
[https://msdn.microsoft.com/en-us/library/windows/desktop/aa365541\(v=vs.85\).aspx](https://msdn.microsoft.com/en-us/library/windows/desktop/aa365541(v=vs.85).aspx)
- [3] MSDN, Windows API 官方文档: CreateFile Function, URL:  
[https://msdn.microsoft.com/en-us/library/windows/desktop/aa363858\(v=vs.85\).aspx](https://msdn.microsoft.com/en-us/library/windows/desktop/aa363858(v=vs.85).aspx)
- [4] MSDN, Windows API 官方文档: WriteFile Function, URL:  
[https://msdn.microsoft.com/en-us/library/windows/desktop/aa365747\(v=vs.85\).aspx](https://msdn.microsoft.com/en-us/library/windows/desktop/aa365747(v=vs.85).aspx)
- [5] Charles Petzold, 《Windows 程序设计》, 方敏、张胜、梁路平等译, 北京: 清华大学出版社, 2015 年 9 月第 5 版: 426 页- 440 页
- [6] Charles Petzold, 《Windows 程序设计》, 方敏、张胜、梁路平等译, 北京: 清华大学出版社, 2015 年 9 月第 5 版: 64 页- 65 页
- [7] Charles Petzold, 《Windows 程序设计》, 方敏、张胜、梁路平等译, 北京: 清华大学出版社, 2015 年 9 月第 5 版: 57 页- 58 页
- [8] MSDN, Windows API 官方文档: SetWindowText Function, URL:  
[https://msdn.microsoft.com/en-us/library/windows/desktop/ms633546\(v=vs.85\).aspx](https://msdn.microsoft.com/en-us/library/windows/desktop/ms633546(v=vs.85).aspx)
- [9] MSDN, Developer Network 文档: atoi, \_atoi\_l, \_wtoi, \_wtoi\_l, URL:  
<https://msdn.microsoft.com/en-us/library/yd5xkb5c.aspx>
- [10] MSDN, Windows API 官方文档: EnableWindow Function, URL:  
[https://msdn.microsoft.com/en-us/library/windows/desktop/ms646291\(v=vs.85\).aspx](https://msdn.microsoft.com/en-us/library/windows/desktop/ms646291(v=vs.85).aspx)
- [11] MSDN, Windows API 官方文档: ListBox Control, URL:  
[https://msdn.microsoft.com/en-us/library/windows/desktop/aa369761\(v=vs.85\).aspx](https://msdn.microsoft.com/en-us/library/windows/desktop/aa369761(v=vs.85).aspx)
- [12] MSDN, Windows API 官方文档: ListView Control, URL:  
[https://msdn.microsoft.com/en-us/library/windows/desktop/aa369763\(v=vs.85\).aspx](https://msdn.microsoft.com/en-us/library/windows/desktop/aa369763(v=vs.85).aspx)