通讯录管理系统

# 1.问题描述

**通讯录**

设计一个通讯录管理系统，每条记录包括：联系人姓名、电话、住址。

系统功能要求：

（1）显示通讯录所有内容。

（2）查询通讯录。按联系人姓名，将查询结果整条记录输出到屏幕。

（3）修改通讯录。通过查询功能，找到需要修改的记录，并修改内容。

（4）添加联系人。新增联系人信息。

（5）删除联系人。通过查询功能，找到要删除的联系人信息。

（6）清空联系人。清空联系人文本文件内容。

（7）退出通讯录管理系统。

# 2.需求分析

通讯录管理系统是为了方便用户管理个人联系人信息而设计的应用程序。通过该系统，用户可以添加、删除、显示、查找、修改以及清空联系人信息。

# 3.概要设计

**3.1 数据结构设计**

结构体加指针

//联系人

struct PeopleTel {

char name[20];

char Phone[12];

char Address[20];

};

//链表

struct list {

struct PeopleTel data;

struct list\* next;

};

**3.2 系统功能设计**

系统采用链表数据结构，每个节点表示一个联系人，节点之间通过指针进行连接。包含头节点，头节点不存储实际联系人信息。

**3.3创新点：用文件保存**

# 4.模块分析

* **链表模块：**
  + **createHeadList()**: 创建头节点。
  + **createNode(PeopleTel data)**: 创建新节点。
  + **insertList(list\*& Head, PeopleTel data)**: 在链表中插入新节点。
* **文件操作模块：**
  + **readFile(list\* headNode, const char\* fileURL)**: 从文件中读取联系人信息到链表。
  + **saveFile(list\* headNode, const char\* fileURL)**: 将链表中的联系人信息保存到文件。
* **联系人管理模块：**
  + **deleteContact(list\*& Head, const char\* nameToDelete)**: 删除指定联系人。
  + **searchContact(list\* Head, const char\* nameToSearch)**: 查找指定联系人。
  + **updateContact(list\* Head, const char\* nameToUpdate, const PeopleTel& newData)**: 更新指定联系人信息。
  + **clearContacts(list\*& Head)**: 清空所有联系人信息。
* **用户交互模块：**
  + **menu()**: 显示用户菜单。
  + **keyDown(list\* &Headlist)**: 处理用户输入，调用相应模块实现功能。

# 5．详细设计

## 1.链表模块

**1.1createHeadList()**

**该函数返回一个新的头节点，初始时头节点的指针为空。**

**list\* createHeadList()**

**{**

**list\* HeadList = NULL;**

**HeadList = new list;**

**HeadList->next = NULL;**

**return HeadList;**

**}**

**1.2createNode(PeopleTel data)**

**该函数创建一个新的节点，将给定的联系人信息存储在节点中，并返回节点指针。**

**list\* createNode(PeopleTel data)**

**{**

**list\* newNode = new list;**

**newNode->data = data;**

**newNode->next = NULL;**

**return newNode;**

**}**

**1.3 insertList(list\*& Head, PeopleTel data)**

**该函数在链表中插入一个新的节点，将给定的联系人信息存储在节点中。**

**void insertList(list\*& Head, PeopleTel data)**

**{**

**list\* newList = createNode(data);**

**newList->next = Head->next;**

**Head->next = newList;**

**}**

## 2. 文件操作模块

**2.1**

**readFile(list\* headNode, const char\* fileURL)**

该函数从文件中读取联系人信息到链表。

void readFile(list\* headNode, const char\* fileURL)

{

FILE\* fp = fopen(fileURL, "r");

if (fp == NULL)

{

fp = fopen(fileURL, "w");

fclose(fp);

return;

}

PeopleTel tmp;

while (fscanf(fp, "%s\t%s\t%s\n", tmp.name, tmp.Phone, tmp.Address) != EOF)

{

insertList(headNode, tmp);

}

fclose(fp);

}

**2.2**

**saveFile(list\* headNode, const char\* fileURL)**

该函数将链表中的联系人信息保存到文件。

void saveFile(list\* headNode, const char\* fileURL)

{

FILE\* fp = fopen(fileURL, "w");

list\* Move = headNode->next;

while (Move)

{

fprintf(fp, "%s\t%s\t%s\n",

Move->data.name,

Move->data.Phone,

Move->data.Address);

Move = Move->next;

}

fclose(fp);

}

## 3. 联系人管理模块

**3.1**

**deleteContact(list\*& Head, const char\* nameToDelete)**

该函数删除指定姓名的联系人信息。

**void deleteContact(list\*& Head, const char\* nameToDelete)**

**{**

**list\* current = Head->next;**

**list\* prev = Head;**

**while (current != NULL)**

**{**

**if (strcmp(current->data.name, nameToDelete) == 0)**

**{**

**prev->next = current->next;**

**delete current;**

**cout << "联系人已删除" << endl;**

**return;**

**}**

**prev = current;**

**current = current->next;**

**}**

**cout << "未找到该联系人" << endl;**

**}**

**3.2**

**searchContact(list\* Head, const char\* nameToSearch)**

该函数查找指定姓名的联系人信息并显示。

**void searchContact(list\* Head, const char\* nameToSearch)**

**{**

**list\* current = Head->next;**

**while (current != NULL)**

**{**

**if (strcmp(current->data.name, nameToSearch) == 0)**

**{**

**cout << "联系人信息：" << endl;**

**cout << current->data.name << "\t" << current->data.Phone << "\t" << current->data.Address << endl;**

**return;**

**}**

**current = current->next;**

**}**

**cout << "未找到该联系人" << endl;**

**}**

**3.3**

**updateContact(list\* Head, const char\* nameToUpdate, const PeopleTel& newData)**

该函数更新指定联系人的信息。

**void updateContact(list\* Head, const char\* nameToUpdate, const PeopleTel& newData)**

**{**

**list\* current = Head->next;**

**while (current != NULL)**

**{**

**if (strcmp(current->data.name, nameToUpdate) == 0)**

**{**

**current->data = newData;**

**cout << "联系人信息已更新" << endl;**

**return;**

**}**

**current = current->next;**

**}**

**cout << "未找到该联系人" << endl;**

**}**

**3.4**

**clearContacts(list\*& Head)**

该函数清空所有联系人信息。

void clearContacts(list\*& Head)

{

list\* current = Head->next;

list\* nextNode;

while (current != NULL)

{

nextNode = current->next;

delete current;

current = nextNode;

}

Head->next = NULL;

cout << "所有联系人已清空" << endl;

}

## 4.用户交互模块

**4.1**

**menu()**

该函数显示用户菜单。

void menu()

{

cout << "\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*" << endl;

cout << "\*\*\*\*\* 1.添加联系人 \*\*\*\*\*" << endl;

cout << "\*\*\*\*\* 2.删除联系人 \*\*\*\*\*" << endl;

cout << "\*\*\*\*\* 3.显示联系人 \*\*\*\*\*" << endl;

cout << "\*\*\*\*\* 4.查找联系人 \*\*\*\*\*" << endl;

cout << "\*\*\*\*\* 5.修改联系人 \*\*\*\*\*" << endl;

cout << "\*\*\*\*\* 6.清空联系人 \*\*\*\*\*" << endl;

cout << "\*\*\*\*\* 0.退出系统 \*\*\*\*\*" << endl;

cout << "\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*" << endl;

}

**4.2**

**keyDown(list\* &Headlist)**

该函数处理用户输入，调用相应的模块实现功能

void keyDown(list\* &Headlist)

{

PeopleTel Peo;

int userKey = 0;

cout << "请输入你的选择(0-6)：";

cin >> userKey;

switch (userKey)

{

case 0:

cout << "退出成功" << endl;

exit(0);

break;

case 1:

cout << "请输入信息：姓名 电话 联系地址" << endl;

cin >> Peo.name >> Peo.Phone >> Peo.Address;

insertList(Headlist, Peo);

saveFile(Headlist, "通讯录文件.txt");

cout << "添加成功" << endl;

break;

case 2:

cout << "请输入要删除的联系人姓名：";

cin >> Peo.name;

deleteContact(Headlist, Peo.name);

saveFile(Headlist, "通讯录文件.txt");

break;

case 3:

insearch(Headlist);

break;

case 4:

cout << "请输入要查找的联系人姓名：";

cin >> Peo.name;

searchContact(Headlist, Peo.name);

break;

case 5:

cout << "请输入要修改的联系人姓名：";

cin >> Peo.name;

cout << "请输入新的联系人信息：姓名 电话 联系地址" << endl;

cin >> Peo.name >> Peo.Phone >> Peo.Address;

updateContact(Headlist, Peo.name, Peo);

saveFile(Headlist, "通讯录文件.txt");

break;

case 6:

clearContacts(Headlist);

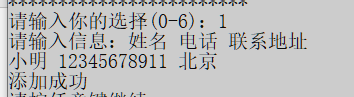
saveFile(Headlist, "通讯录文件.txt");

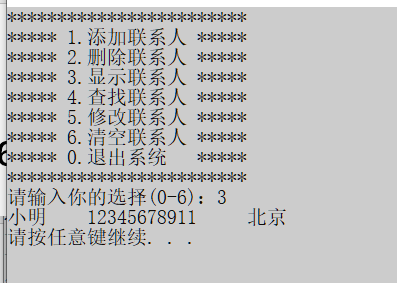
break;

}

}

# 6.测试分析





文本, 信件

描述已自动生成

文本

描述已自动生成

图形用户界面, 应用程序

描述已自动生成

文本, 信件

描述已自动生成

文本

中度可信度描述已自动生成

图形用户界面, 文本, 应用程序, 电子邮件

描述已自动生成

# 7.用户手册

进入项目文件下双击这个文件直接运行在dev中

图形用户界面, 文本

中度可信度描述已自动生成

就可以直接使用

源代码在main.cpp中

