

第26讲：顺序表的应用

目录

1. 基于动态顺序表实现通讯录项目
2. 顺序表经典算法
3. 顺序表的问题及思考

正文开始

顺序表的应用

1. 基于动态顺序表实现通讯录

C语言基础要求：结构体、动态内存管理、顺序表、文件操作

1、功能要求

- 1) 至少能够存储100个人的通讯信息
- 2) 能够保存用户信息：名字、性别、年龄、电话、地址等
- 3) 增加联系人信息
- 4) 删除指定联系人
- 5) 查找制定联系人
- 6) 修改指定联系人
- 7) 显示联系人信息

2、代码实现

【思考1】用静态顺序表和动态顺序表分别如何实现

【思考2】如何保证程序结束后，历史通讯录信息不会丢失

```
1 //SeqList.h
2
3 #pragma once
4 #define _CRT_SECURE_NO_WARNINGS
5 #include<stdio.h>
6 #include<assert.h>
```

```
7 #include<stdlib.h>
8 #include"contact.h"
9
10 //数据类型为PersonInfo
11 typedef struct PersonInfo SQDataType;
12 //typedef int SQDataType;
13
14 //动态顺序表
15 typedef struct SeqList {
16     SQDataType* a;
17     int size;//保存有效数据个数
18     int capacity;//空间的大小
19 }SLT;
20
21 //初始化与销毁
22 void SeqListInit(SLT* psl);
23 void SeqListDesTroy(SLT* psl);
24 void SeqListPrint(SLT sl);
25 void CheckCapacity(SLT* psl);
26
27 // 头部插入删除 / 尾部插入删除
28 void SeqListPushBack(SLT* psl, SQDataType x);
29 void SeqListPushFront(SLT* psl, SQDataType x);
30 void SeqListPopBack(SLT* psl);
31 void SeqListPopFront(SLT* psl);
32
33 //查找
34 int SeqListFind(SLT* psl, SQDataType x);
35 // 在指定位置之前插入/删除
36 //void SeqListInsert(SLT* psl, int pos, SQDataType x);
37 void SeqListInsert(SLT* psl, size_t pos, SQDataType x);
38 void SeqListErase(SLT* psl, size_t pos);
39
40 size_t SeqListSize(SLT* psl);
41 //修改指定位置的值
42 void SeqListAt(SLT* psl, size_t pos, SQDataType x);
```

```
1 //contact.h
2 #pragma once
3 #define NAME_MAX 100
4 #define SEX_MAX 4
5 #define TEL_MAX 11
6 #define ADDR_MAX 100
7
8 //前置声明
```

```

9  typedef struct SeqList Contact;
10
11 //用户数据
12 typedef struct PersonInfo
13 {
14     char name[NAME_MAX];
15     char sex[SEX_MAX];
16     int age;
17     char tel[TEL_MAX];
18     char addr[ADDR_MAX];
19 }PeoInfo;
20
21 //初始化通讯录
22 void InitContact(Contact* con);
23 //添加通讯录数据
24 void AddContact(Contact* con);
25 //删除通讯录数据
26 void DelContact(Contact* con);
27 //展示通讯录数据
28 void ShowContact(Contact* con);
29 //查找通讯录数据
30 void FindContact(Contact* con);
31 //修改通讯录数据
32 void ModifyContact(Contact* con);
33 //销毁通讯录数据
34 void DestroyContact(Contact* con);

```

```

1 //contact.c
2 #define _CRT_SECURE_NO_WARNINGS
3 #include "contact.h"
4 #include "SeqList.h"
5
6 void LoadContact(Contact* con) {
7     FILE* pf = fopen("contact.txt", "rb");
8     if (pf == NULL) {
9         perror("fopen error!\n");
10        return;
11    }
12
13    //循环读取文件数据
14    PeoInfo info;
15    while (fread(&info, sizeof(PeoInfo), 1, pf))
16    {
17        SeqListPushBack(con, info);
18    }

```

```

19     printf("历史数据导入通讯录成功! \n");
20 }
21
22 void InitContact(contact* con) {
23     SeqListInit(con);
24     LoadContact(con);
25 }
26
27 void AddContact(contact* con) {
28     PeoInfo info;
29     printf("请输入姓名: \n");
30     scanf("%s", &info.name);
31     printf("请输入性别: \n");
32     scanf("%s", &info.sex);
33     printf("请输入年龄: \n");
34     scanf("%d", &info.age);
35     printf("请输入联系电话: \n");
36     scanf("%s", &info.tel);
37     printf("请输入地址: \n");
38     scanf("%s", &info.addr);
39
40     SeqListPushBack(con, info);
41     printf("插入成功! \n");
42
43 }
44
45 int FindByName(contact* con, char name[]) {
46     for (int i = 0; i < con->size; i++)
47     {
48         if (0 == strcmp(con->a[i].name, name)) {
49             return i;
50         }
51     }
52     return -1;
53 }
54
55 void DelContact(contact* con){
56     char name[NAME_MAX];
57     printf("请输入要删除的用户姓名: \n");
58     scanf("%s", name);
59
60     int pos = FindByName(con, name);
61     if (pos < 0) {
62         printf("要删除的用户不存在, 删除失败! \n");
63         return;
64     }
65

```

```
66     SeqListErase(con, pos);
67     printf("删除成功! \n");
68 }
69
70 void ShowContact(contact* con){
71     printf("%-10s %-4s %-4s %15s %-20s\n", "姓名", "性别", "年龄", "联系电话",
72     for (int i = 0; i < con->size; i++)
73     {
74         printf("%-10s %-4s %-4d %15s %-20s\n",
75             con->a[i].name,
76             con->a[i].sex,
77             con->a[i].age,
78             con->a[i].tel,
79             con->a[i].addr);
80     }
81 }
82
83 void FindContact(contact* con){
84     char name[NAME_MAX];
85     printf("请输入要查找的用户姓名: \n");
86     scanf("%s", name);
87
88     int pos = FindByName(con, name);
89     if (pos < 0) {
90         printf("要查找的用户不存在, 查找失败! \n");
91         return;
92     }
93
94     printf("查找成功! \n");
95     printf("%-10s %-4s %-4d %15s %-20s\n",
96         con->a[pos].name,
97         con->a[pos].sex,
98         con->a[pos].age,
99         con->a[pos].tel,
100         con->a[pos].addr);
101 }
102
103 void ModifyContact(contact* con) {
104     char name[NAME_MAX];
105     printf("请输入要修改的用户名称: \n");
106     scanf("%s", name);
107
108     int pos = FindByName(con, name);
109     if (pos < 0) {
110         printf("要查找的用户不存在, 修改失败! \n");
111         return;
112     }
```

```
113
114     PeoInfo info;
115     printf("请输入要修改的姓名: \n");
116     scanf("%s", &con->a[pos].name);
117     printf("请输入要修改的性别: \n");
118     scanf("%s", &con->a[pos].sex);
119     printf("请输入要修改的年龄: \n");
120     scanf("%d", &con->a[pos].age);
121     printf("请输入要修改的联系电话: \n");
122     scanf("%s", &con->a[pos].tel);
123     printf("请输入要修改的地址: \n");
124     scanf("%s", &con->a[pos].addr);
125
126     printf("修改成功! \n");
127 }
128
129 void SaveContact(contact* con) {
130     FILE* pf = fopen("contact.txt", "wb");
131     if (pf == NULL) {
132         perror("fopen error!\n");
133         return;
134     }
135     //将通讯录数据写入文件
136     for (int i = 0; i < con->size; i++)
137     {
138         fwrite(con->a + i, sizeof(PeoInfo), 1, pf);
139     }
140
141     printf("通讯录数据保存成功! \n");
142 }
143
144 void DestroyContact(contact* con) {
145     SaveContact(con);
146     SeqListDesTroy(con);
147 }
```

```
1 //test.c
2
3 #include"SeqList.h"
4 #include"contact.h"
5 void menu() {
6     //通讯录初始化
7     contact con;
8     InitContact(&con);
9     int op = -1;
```

```

10
11     do {
12
13         printf("*****\n");
14         printf("*****1、添加用户  2、删除用户*****\n");
15         printf("*****3、查找用户  4、修改用户*****\n");
16         printf("*****5、展示用户  0、退出    *****\n");
17         printf("*****\n");
18
19         printf("请选择您的操作：\n");
20         scanf("%d", &op);
21
22         switch (op)
23         {
24             case 1:
25                 AddContact(&con);
26                 break;
27             case 2:
28                 DelContact(&con);
29                 break;
30             case 3:
31                 FindContact(&con);
32                 break;
33             case 4:
34                 ModifyContact(&con);
35                 break;
36             case 5:
37                 ShowContact(&con);
38                 break;
39             default:
40                 printf("输入有误，请重新输入\n");
41                 break;
42         }
43     } while (op!=0);
44     //销毁通讯录
45     DestroyContact(&con);
46 }

```

课后练习：静态顺序表实现通讯录。

2. 顺序表经典算法

经典算法OJ题1： [移除元素](#)

经典算法OJ题2： [合并两个有序数组](#)

3. 顺序表的问题及思考

1. 中间/头部的插入删除，时间复杂度为 $O(N)$

2. 增容需要申请新空间，拷贝数据，释放旧空间。会有不小的消耗。

3. 增容一般是呈2倍的增长，势必会有一定的空间浪费。例如当前容量为100，满了以后增容到200，我们再继续插入了5个数据，后面没有数据插入了，那么就浪费了95个数据空间。

思考：如何解决以上问题呢？

完

比特就业课