实验一 顺序表的应用

一、实验目的

1、掌握建立顺序表的基本方法。

2、掌握顺序表的插入、删除算法的思想和实现，并能灵活运用

二、实验内容

    用顺序表实现病历信息的管理与查询功能。具体要求如下:

1. 利用教材中定义顺序表类型存储病人病历信息(病历号,姓名，症状)；要求使用头文件。

2.设计顺序表定位查找算法，写成一个函数，完成的功能为:在线性表L中查找数据元素x，如果存在则返回线性表中和x值相等的第1个数据元素的序号；如果不存在，则返回-1。

    函数定义为 int ListFind(SequenceList L,char \*x)

请在主函数中测试查找是否存在姓名为x的病人，并根据返回的序号打印出病人信息。

三、实验源代码

typedef struct

{

    ElemType list[MaxSize];

    int size;

}SequenceList;

void ListInitialize(SequenceList \*L)  /\*初始化顺序表\*/

{

    L->size = 0;    /\*定义初始数据的元素个数\*/

}

int ListLength(SequenceList L)  /\*返回顺序表L的当前数据元素个数\*/

{

    return L.size;

}

int ListInsert(SequenceList \*L, int i, ElemType x)  /\*在顺序表L的位置i(0<=i<=size)前插入数据元素值x，插入成功返回1，插入失败返回0\*/

{

    int j;

    if (L->size >= MaxSize)

    {

        printf("The sequential table is full and cannot be inserted!\n");

        return 0;

    }

    else if (i < 0 || i > L->size)

    {

        printf("Parameter i is invalid!\n");

        return 0;

    }

    else

    {

        for (j = L->size; j > i; j--)   /\*为插入做准备\*/

        {

            L->list[j] = L->list[j - 1];

        }

        L->list[i] = x; /\*插入\*/

        L->size++;  /\*元素个数加1\*/

        return 1;

    }

}

/\*删除顺序表L中位置i(0<=i<=size-1)的数据元素并存放到参数x中\*/

/\*删除成功返回1，删除失败返回0\*/

int ListDelete(SequenceList \*L, int i, ElemType \*x)

{

    int j;

    if (L->size <= 0)

    {

        printf("The sequential table is empty and there is no data to delete!\n");

        return 0;

    }

    else if (i < 0 || i > L->size - 1)

    {

        printf("Parameter i is invalid!\n");

        return 0;

    }

    else

    {

        \*x = L->list[i];    /\*保存删除的元素到参数x中\*/

        for (j = i + 1; j <= L->size - 1; j++)

        {

            L->list[j - 1] = L->list[j];    /\*依次前移\*/

        }

        L->size--;  /\*数据元素个数减1\*/

        return 1;

    }

}

/\*取顺序表L中第i个数据元素的值存于x中，成功则返回1，失败返回0\*/

int ListGet(SequenceList L, int i, ElemType \*x)

{

    if (i < 0 || i> L.size - 1)

    {

        printf("Parameter i is invalid!\n");

        return 0;

    }

    else

    {

        \*x = L.list[i];

        return 1;

    }

}

#include <stdio.h>

#include <string.h>

#define MaxSize 100

#define N 2

typedef struct

{

    char number[5];

    char name[20];

    char symptom[50];

}patient;

typedef patient ElemType;

#include "SequenceList.h"

int ListFind(SequenceList L,char \*x);

int main(void)

{

    patient s;

    SequenceList mylist;

    int i;

    ListInitialize(&mylist);    /\*顺序表初始化\*/

    for (i = 0; i < N; i++)     /\*输入数据\*/

    {

        printf("------Input %d Patient Information\n", i + 1);

        printf("Input %d Patient number:", i + 1);

        scanf("%s", s.number);

        printf("Input %d Patient name:", i + 1);

        scanf("%s", s.name);

        printf("Input %d Patient symptom:", i + 1);

        scanf("%s", s.symptom);

        ListInsert(&mylist, i, s);      /\*插入到顺序表中\*/

    }

    printf("\n\*\*\*\*\*Data in a sequential table\*\*\*\*\*\n");

    for (i = 0; i < ListLength(mylist); i++)

    {

        ListGet(mylist, i, &s);

        printf("%s %s %s\n", s.number, s.name, s.symptom);

    }

    char find\_name[20];     /\*存储要查找的患者姓民\*/

    printf("\nPlease enter the name of the person you want to find:");

    scanf("%s", find\_name);

    i = ListFind(mylist, find\_name);

    if (i != -1)

    {

        ListGet(mylist, i, &s);     /\*取出病人信息\*/

        printf("%s %s %s\n", s.number, s.name, s.symptom);

    }

    else

    {

        printf("Not Found");

    }

    return 0;

}

/\*在线性表L中查找数据元素x，如果存在则返回线性表中和x值相等的第1个数据元素的序号；如果不存在，则返回-1。\*/

int ListFind(SequenceList L, char \*x)

{

    int i;

    for (i = 0; i < L.size; i++)

    {

        if (strcmp(L.list[i].name, x) == 0)

        {

            return i;

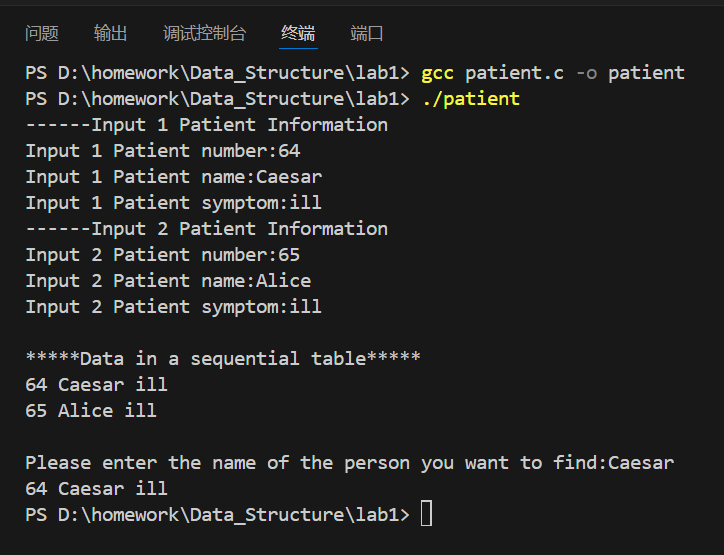
        }

    }

    return -1;

}

四、实验结果



五、实验总结

答：在本次实验中，我学会了顺序表的存储结构，学会了顺序表的基本操作实现，并进行顺序表的应用。在实验过程中我也遇到一个问题：在设计顺序表的查找函数时，运行后没有结果显示。经过检查，发现strcmp(L.list[i].name, x) 中name被我错误的写为number，导致结果没有正确显示，修改后代码得以成功运行得出结果。