1. Operation.h头文件
2. //一、实验目的
3. // 1、掌握建立顺序表的基本方法。
4. // 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
5. // 二、实验内容
6. //     用顺序表实现病历信息的管理与查询功能。具体要求如下:
7. // 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。
8. // 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。
9. //     函数定义为 int ListFind(SequenceList L,char \*x)
10. // 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。
11. #define MaxSize 100
12. typedef struct {
13. int number;
14. char name[4];
15. char sym[100];
16. }patient;
17. typedef struct{
18. patient list[MaxSize];
19. int size;
20. }SequenceList;
21. int ListFind(SequenceList L,char \*x)//顺序表定位查找算法
22. {
23. int i,j=-1;
24. for(i=0;i<L.size;i++){
25. if(x==L.list[i].name)
26. return i;
27. }
29. return j;
30. }
31. //顺序表初始化
32. void SequenceInit(SequenceList \*L){
33. L->size=1;
34. }
35. //顺序表插入
36. int Listsert(SequenceList \*L,int i,patient x)
37. {
38. int j;
39. if(L->size > MaxSize)
40. {
41. printf("满了，插不了");
42. return 0;
43. }
44. else if((i<0) ||(i>L->size))
45. {
46. printf("插入不合法\n");
47. return 0;
48. }
49. else{
50. for(j = L->size;j>i;j--)
51. {
52. L->list[j] = L->list[j-1];
53. L->list[i] = x;
54. L->size ++;
55. return 1;
56. }
57. }
58. }
59. //顺序表删除
60. // void listdelete(SequenceList \*L,int i;patient \*x)
61. // {
63. // }

2. main.c文件

#include<stdio.h>

#include<stdlib.h>

#include"operation.h"

int main(){

    int i,conti=1;

    SequenceList L;

    SequenceInit(&L);//初始化

    patient patient1;

    //输入功能

    printf("请你按照病人的名字，症状的顺序来输入，切不可输错,输入完成到下一个请按1,输入0即可退出输入程序\n");

    printf("开始吧");

while(conti){

   scanf("%s,%s",L.list[i].name,L.list[i].sym);

    Listsert(&L,i,L.list[i]);

     scanf("%d",conti);

    i++;

}

//查询功能

printf("请输入你想要查询的病人的名字");

scanf("%c",patient1.name);

if(ListFind(L,patient1.name)!=-1)

{

    printf("%s\n",ListFind(L.patient1.name));

     printf("%s\n",ListFind(L.patient1.sym));

}

else

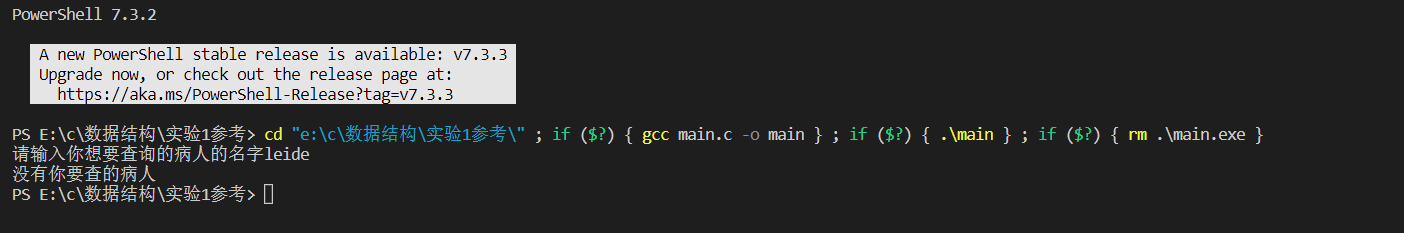
printf("没有你要查的病人");

return 0;

}

3. 运行结果

查询功能



输入功能

啥