## 实验任务描述：

给出一组数据，创建散列表（线性探测法和拉链法处理冲突）

## 二、实验环境

Visual Studio 2017

**三．实验步骤和流程描述**

哈希表的定义不赘述，关键在于怎么设计结构？在注释中说明了。

1.基本的存储节点

typedef struct

{

int key; //这个key是存储的关键字

char data;

}ElemType; //定义结点类型

2.线性探测采用的存储结构

typedef struct //线性探测法 数据结构

{

ElemType elems[max];

int len;

}HashTable; //定义散列表类型

int stored[max]; //标志数组

3.拉链法时采用的存储结构

typedef struct ElemNode { //拉链法时采用的存储结构

int key; //这个key是除余函数直接计算的值

ElemType data; //关键字值

struct ElemNode\* next; //拉链法每个位置实际是一个链表，所以应有指针

}ElemNode;

typedef struct {

ElemNode\* first;

int len;

}ElemHeader, hashtable[max];

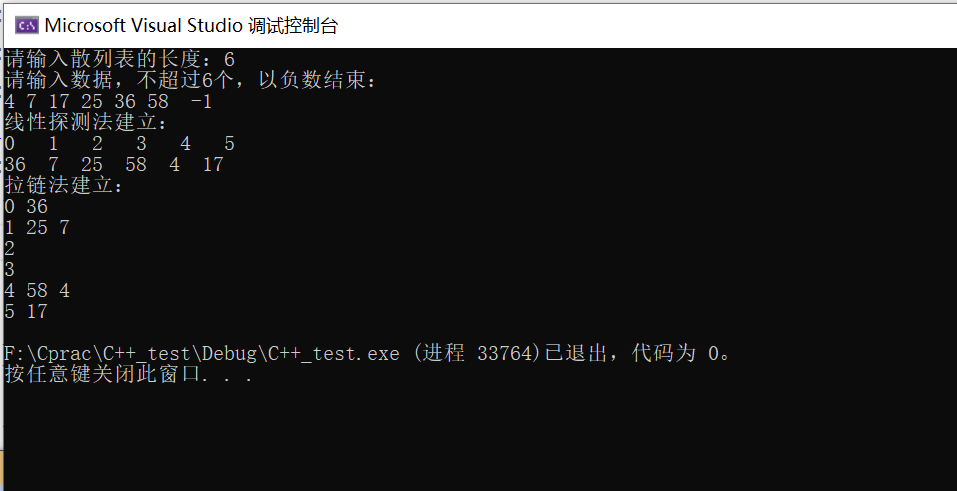
定义好这些结构后实际上逻辑的运算并不难，规规矩矩按照两种解决冲突的办法设计函数填入就好了。

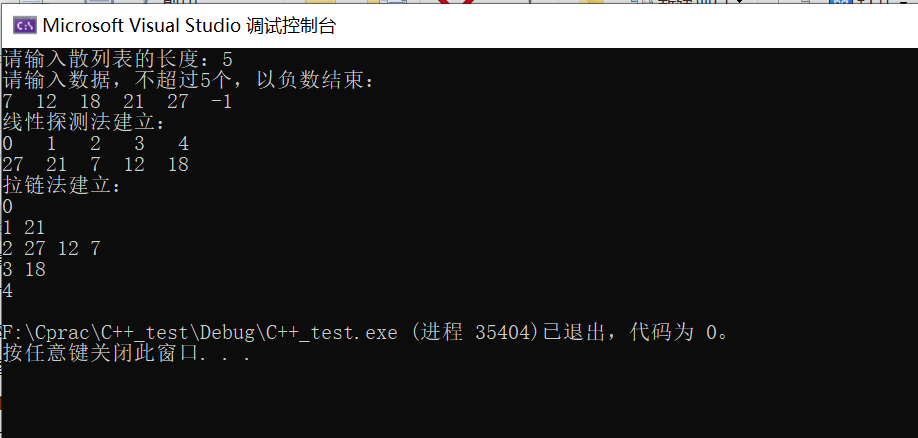
测试用例和截图：

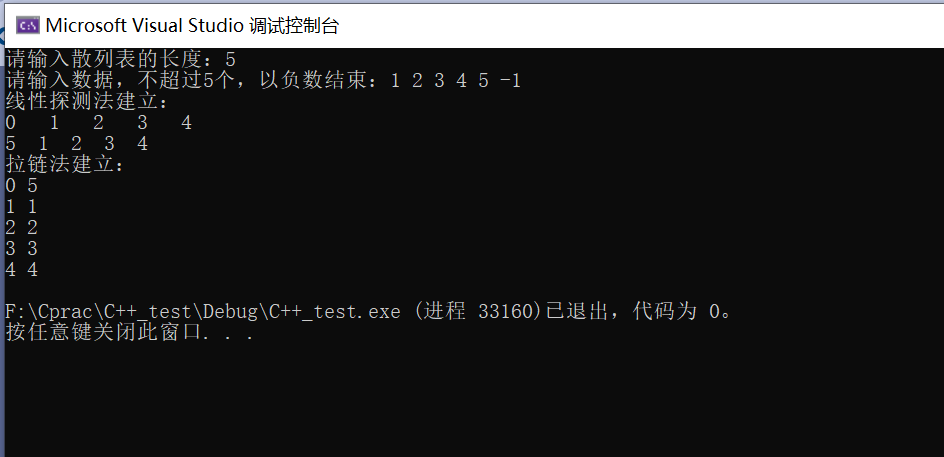
4 7 17 25 36 58

7 12 18 21 27

1 2 3 4 5







五、**问题分析及收获**

两种解决冲突的函数设计并不难，感觉清楚了结构体的定义分工好，就不是很难了。

通过这个实验更加清楚了散列表的查找方式和线性探测法及拉链法的冲突处理过程，巩固了知识的理解。