**2.7假设顺序表L中的元素按从小到大的次序排列，设计算法以删除表中重复的元素, 并 要求时间尽可能少。要求：**

**（1）对顺序表（1,1,2,2,2,3,4,5,5,5,6,6,7,7,8,8,8,9）模拟执行本算法，并统计移动元素的 次数。**

**（2）分析算法的时间性能。**

void SetSubtraction(seqList & A, seqList & B)  
{  
    int i=-1; //指示 A 中已经处理的最后元素  
    int ia=0, ib=0; //指示 A、B 中，当前待处理的元素，初始指向第一个元素  
    while( ia<A.listLen && ib<B.listLen )  
   {  
    if(A.data[ia]==B.data[ib]) //非 A-B 中元素，ia、ib 同时后移  
    {  
        ia++;  
        ib++;  
    }  
    else if(A.data[ia]>B.data[ib])   
        ib++; //此时，ia 指示元素可能在 B 中 ib 指示的元素后面，移动 ib。  
    else //此为，A.data[ia]<B.data[ib]，因为递增性，ia 指示的元素不可能在 B 中。  
    { //所以 ia 指示元素必在 A-B 中。  
      //如果(i+1)==ia，说明 ia 元素不需迁移位置，直接为 A-B 中元素  
    if(i+1!=ia) //(i+1!=ia)，需要将 ia 指示元素迁移到目标位置 i+1  
        A.data[i+1]=A.data[ia];  
        i++; //A-B 集合最后元素指示器后移  
        ia++; //A 的指示器后移  
    }  
}  
 //处理 B 已经，A 尚未结束情况，A 中剩下部分元素全部为 A-B 元素  
    while(ia<A.listLen)   
    {  
        if(i+1!=ia)  
            A.data[i+1]=A.data[ia];  
            i++; //A-B 集合最后元素指示器后移  
            ia++; //A 的指示器后移  
    }  
    A.listLen=i+1;  
}

时间性能：O(n3 )

**2.9 递增有序顺序表 A、B 分别表示一个集合，设计算法求解 A=A－B，并分析其时间 性能。**  
#include<iostream>  
#include<stdlib.h>  
#define MAXSIZE 100  
#define Error 0  
#define OK 1  
typedef int Status;  
using namespace std;  
  
//顺序表的储存结构   
typedef struct sqlist{  
int \*elem;  
int length;  
}SqList; //顺序表的结构类型为Sqlist   
  
//初始化顺序表  
Status InitList(SqList &L){   
    //构造一个空的顺序表  
//为顺序表分配一个大小为MAXSIZE的数组空间   
L.elem=new int [MAXSIZE];   
if(!L.elem)   
return Error;  
//空表长度为0   
L.length=0;  
return OK;  
}  
void CreateList(SqList &L, int n){  
int e;  
if(n>MAXSIZE)  
    cout<<"超出最大范围"<<endl;  
else{  
for(int i=0;i<n;i++){  
cout<<"请输入第"<<i+1<<"个元素数据"<<endl;  
cin>>e;  
L.elem[i]=e;  
L.length++;  
}   
    }  
}  
void ShowList(SqList &L){  
for(int i=0;i<L.length;i++){  
cout<<L.elem[i]<<"  ";  
}   
}  
void DeleteList(SqList &L){  
int i=0;  
int j=1,c=0;  
  
    while (j<=L.length) {  
     L.elem[c]=L.elem[i];  
     while(L.elem[i]==L.elem[j]){  
     j++;  
}  
             i=j;  
         j=i+1;  
         c++;      
}  
      
L.length=c;   
}  
  
int main(){  
SqList L;  
int n,m;  
cout<<"请先初始化一个顺序表并赋初值"<<endl;  
cout<<"请输入你要填写的数子个数"<<endl;  
cin>>n;//         题目中元素有18个。   
InitList( L);  
CreateList(L,n); //    逐渐输入题目要求的元素。   
cout<<"输入数据如下"<<endl;  
ShowList(L);  
DeleteList(L);//    对重复元素的删除。   
cout<<"删除相同元素后数据如下"<<endl;  
ShowList(L);  
  
  
return 0;  
}

时间性能 O(|A| + |B|)