顺序表代码题

1. 从顺序表中删除具有最小值的元素（假设唯一）并由函数返回被删元素的值。空出的位置由最后一个元素填补，若顺序表为空，则显示出错信息并退出运行。

思路

遍历比较，最后交换.

代码

```
bool Del_Min(SqList &L,ElemType &value){
        if(L.length==0)
        {
                return false;
        }
        value = L.data[0];
        int pos=0;
        for(int i=0;i<L.length;i++)
        {
                if(L.data[i]<value)
                {
                        value=L.data[i];
                        pos=i;
                }
        }
        L.data[pos]=L.data[length-1];
        L.length--;
        return true;
}
```

1. 设计一个高效算法，将顺序表L的所有元素逆置，要求算法的空间复杂度为$O(1)$

思路

折半交换

代码

```
void Reverse(Sqlist &L)
{
        ElemType temp;
```

```
    for(int i=0;i<L.length/2;i++)
    {
        temp =L.data[i];
        L.data[i]=L.data[length-i-1];
        L.data[length-i-1] = temp;
    }
}
```

1. 对长度为n的顺序表L,编写一个时间复杂度为$O(n)$、空间复杂度为$O(1)$的算法，该算法别删除顺序表中所有值为x的数据元素。

思路

直接覆盖，计数法

代码

```
void Del_x(SqList &L,ElemType x)
{
    int k,i=0;
    for(i=0;i<L.length;i++)
    {
        if(L.data[i]!=x)
        {
            L.data[k]=L.data[i];
            k++;
        }
    }
    L.length=K++;
}
```

1. 从顺序表中删除其值在给定值s和t之间（包含s和t，要求s<t)的所有元素，若s,t不合理或顺序表为空，则显示出错信息并退出运行。

思路

同上题

代码

```
bool Del_s_t(SqList &L,int s,int t)
{
```

```
        if(L.length==0||s>=t)
        return false;
        int k,i=0;
        for(i=0;i<L.length;i++)
        {
                if(L.data[i]<s||L.data[i]>t)
                {
                        L.data[k]=L.data[i];
                        k++;
                }
        }
        L.length = K++;
}
```

1. 从有序顺序表中删除所有其值重复的元素，使表中所有元素的值均不同。

思路

暴力解法

代码

```
void Del_repeat(SqList &L)
{
        int k,i=0;
        for(i=1;i<L.length;i++){
                if(L.data[k]!=L.data[i]){
                        L.data[k++]=L.data[i];
                }
        }
        L.length = k++;
}
```

1. 将两个有序顺序表合并为一个新的有序顺序表，并由函数返回结果顺序表。

思路

归并

代码

```
bool Merge(SeqList &A,SeqList &B,SeqList &C){
        if(A.length+B.length>C.Maxsize)
```

```
            return false;
        int i,j,k=0;
        while(i<A.length&&j<B.length){
            if(A.data[i]<B.data[j])
                C.data[k++]=A.data[i++];
            else
                C.data[k++]=B.data[j++];
        }
        while(i<A.length)
            C.data[k++]=A.data[i++];
        while(j<B.length)
            C.data[K++]=B.data[j++];
        C.length = K;
        return true;
}
```

1. 已知在一维数组A[m+n]中依次存放两个线性表(a1，a2,a3,…,am)和(b1,b2,b3,…,bn)。编写一个函数，将数组中两个顺序表的位置互换，即将(b1,b2,b3,…,bn)放在(a1,a2,a3,…,am)的前面。

思路

先逆置再分别逆置

代码

```
typedef int DataType;
void Reverse(DataType A[],int left,int right,int arraySize){
        if(left>right||right>=arraySize)
            return
        int mid=(left+right)/2;
        if(int i=0;i<=mid-left;i++)
        {
            int temp = A[left+i];
            A[left+i]=A[right-i];
            A[right-i]=temp;
        }
}
void Exchange(DataType A[],int m,int n,int arrraySize)
{
        Reverse(A,0,m+n-1,arraySize);
        Reverse(A,0,n-1,arraySize);
        Reverse(A,n,n+m-1,arraySize);
}
```

1. 线性表(a1,a2,,•,an)中的元素递增有序且按顺序存储于计算机内。要求设计一个算法，完成用最少时间在表中查找数值为x的元素，若找到，则将其与后继元素位置相交换，若找不到，则将其插入表中并使表中元素仍递增有序。

思路

折半查找，交换，插入

代码

```
void SearchExchangeInsert(ElemType A[],ElemType x)
{
	int low=0,height=n-1,mid;
	while(low<=height)
	{
		mid = (low+height)/2;
		if(A[mid]==x)
			break;
		if(A[mid]<x)
		{
			low=mid+1;
		}
		else
			height = mid-1;
	}
	if(A[mid]==x&&mid!=1)
	{
		t=A[mid];A[mid]=A[mid+1];A[mid+1]=t
	}
	if(low>height)
	{
		for(i=n-1;i>height;i--)
		{
			A[i+1]=A[i];
		}
		A[i+1]=x;
	}
}
```

1. 给定三个序列A、B、C,长度均为n，且均为无重复元素的递增序列，请设计一个时间上尽可能高效的算法，逐行输出同时存在于这三个序列中的所有元素。例如，数组A为{1,2,3},数组B为{2,3,4}，数组C为{-1,0,2}，则输出2。要求：1)给出算法的基本设计思想，2)根据设计思想，采用C或C+语言描述算法，关健之处给出注释。3)说明你的算法的时间复杂度和空间复杂度。

思路

三下标遍历

代码

```
void samekey(int A[],int B[],int C[],int n)
{
	int i=0,j=0,k=0;
	while(i<n&&j<n&&k<n)
	{
		if(A[i]==B[j]&&B[j]==C[k])
			{
				printf("%d\n",A[i]);
				i++;j++;k++;
			}
		else{
			int maxnum =max(A[i],max(B[j],C[k]));
			if(A[i]<maxnum) i++;
			if(B[j]<maxnum)j++;
			if(C[k]<maxnum)k++;
		}
	}
}
```

时间复杂度分析

时间复杂度$O(n)$,空间复杂度$O(1)$

1. 【2010统考真题】设将n(n>1)个整数存放到一维数组R中。设计一个在时间和空间两方面都尽可能高效的算法。将R中保存的序列循环左移p(0<p<n)个位置，即将R中的数据由(X,X,⋯,Xn-1)变换为(XmXp+1,,Xn-,X,X,⋯,Xp-)要求：1)给出算法的基本设计思想。2)根据设计思想，采用C或C+或Java语言描述算法，关键之处给出注释。3)说明你所设计算法的时间复杂度和空间复杂度。

思路

两次转置，第一次整体，第二次前n-p个后p个分别

代码

```
void Reverse(int R[],int from.int to)
{
```

```
        int i,temp;
        for(i=0;i<(from-to)/2;i++)
        {
                temp=R[from+i];
                R[from+i]=R[to-i];
                R[to-i]=temp;
        }
}
void Converse(int R[],int n,int p)
{
        Reverse(R[],0,n-1);
        Reverse(R[],0,n-p-1);
        Reverse(R[],n-p,n-1);
}
```

时间复杂度

时间复杂度$O(n)$,空间复杂度$O(1)$

1. 【2011统考真题】一个长度为L(L≥1)的升序序列S,处在第L/2个位置的数称为S的中位数。例如，若序列S,=(11,13,15,17,19),则S1的中位数是15，两个序列的中位数是含它们所有元素的升序序列的中位数。例如，若S2=(2,4,6,8,20),则S1和S2的中位数是11。现在有两个等长升序序列A和B,试设计一个在时间和空间两方面都尽可能高效的算法，找出两个序列A和B的中位数。要求：1)给出算法的基本设计思想。2)根据设计思想，采用C或C+或Jva语言描述算法，关键之处给出注释。3)说明你所设计算法的时间复杂度和空间复杂度。

思路

分别求两个升序序列A、B的中位数，设为a和b,求序列A、B的中位数过程如下：
①若a=b,则a或b即为所求中位数，算法结束。
②若a<b,则舍弃序列A中较小的一半，同时舍弃序列B中较大的一半，要求两次舍弃的长度相等。
③若a>b,则舍弃序列A中较大的一半，同时舍弃序列B中较小的一半，要求两次舍弃的长度相等。
在保留的两个升序序列中，重复过程①、②、③，直到两个序列中均只含一个元素时为止，小者即为所求的中位数。

代码

```
int M_Search(int A[],int B[],int n)
{
```

```
    int s1,d1,s2,d2,m1,m2;
    s1=0;d1=n-1;
    s2=0;d2=n-1;
    while(s1!=d1||s2!=d2)
    {
        m1=(d1+s1)/2;
        m2=(d2+s2)/2;
        if(A[m1]==B[m2])
            return A[m1];
        if(A[m1]<B[m2])
        {
            if((s1+d1)%2==0)
            {
                s1=m1;
                d2=m2;
            }
            else
            {
                s1=m+1;
                d2=m2;
            }
        }
        else
        {
            if((s1+d1)%2==0)
            {
                s2=m2;
                d1=m1;
            }
            else
            {
                s2=m2+1;
                d1=m1;
            }
        }
    }
    return A[s1]<B[s2]?A[s1]:B[s2];
}
```

时间复杂度

时间复杂度$O(log_2^n)$,空间复杂度$O(1)$

1. 【2013统考真题】已知一个整数序列A=(a,a1,⋯,am-),其中0≤a_i<n(0≤i<n)。若存在$a_{p1}=a_{p2}=\ldots=a_{pm}$=x且m>n/2(0≤$p_k$<m,1≤k≤m),则称x为A的主元素。例如A=(0,5,5,3,5,7,5,5),则5为主元素；又如A=(0,5,5,3,5,1,5,7),则A中没有主元素。

假设A中的n个元素保存在一个一维数组中，请设计一个尽可能高效的算法，找出A的主元素。若存在主元素，则输出该元素；否则输出-1。要求：1)给出算法的基本设计思想。2)根据设计思想，采用C或C++或Java语言描述算法，关键之处给出注释。3)说明你所设计算法的时间复杂度和空间复杂度。

思路

算法的基本设计思想：算法的策略是从前向后扫描数组元素，标记出一个可能成为主元的元素Num。然后重新计数，确认Num是否是主元素。算法可分为以下两步：
①选取候选的主元素。依次扫描所给数组中的每个整数，将第一个遇到的整数Nm保存到c中，记录Num的出现次数为1：若遇到的下一个整数仍等于Num,则计数加1，否则计数减1：当计数减到0时，将遇到的下一个整数保存到c中，计数重新记为1，开始新一轮计数，即从当前位置开始重复上述过程，直到扫描完全部数组元素。
②判断c中元素是否是真正的主元素。再次扫描该数组，统计c中元素出现的次数，若大于2，则为主元素：否则，序列中不存在主元素。

代码

```
int Majority(A[],int n)
{
	int i,c,count=1;
	c=A[0];
	for(i=1;i<n;i++){
		if(A[i]==c)
			count++;
		else
			if(count>0)
				count--;
			else
			{
				c=A[i];
				count=1;
			}
	}
	if(count>0)
	{
		for(i=count=0;i<n;i++)
		{
			if(A[i]==c)
				count++;
		}
	}
	if(count>n/2)
		return c;
```

```
        else
                return -1;
}
```

时间复杂度

时间复杂度$O(n)$,空间复杂度$O(1)$

1. 【2018统考真题】给定一个含n(n≥1)个整数的数组，请设计一个在时间上尽可能高效的算法，找出数组中未出现的最小正整数。例如，数组{-5,3,2,3}中未出现的最小正整数是1；数组{1,2,3}中未出现的最小正整数是4。要求：1)给出算法的基本设计思想。2)根据设计思想，采用C或C++语言描述算法，关键之处给出注释。3)说明你所设计算法的时间复杂度和空间复杂度。

思路

1)算法的基本设计思想：
要求在时间上尽可能高效，因此采用空间换时间的办法。分配一个用于标记的数组B[]，用来记录A中是否出现了1~n中的正整数，B[0]对应正整数1，B[n-1]对应正整数n,初始化B
中全部为0。由于A中含有n个整数，因此可能返回的值是1~n+1,当A中n个数恰好为1~n时返回n+1。当数组A中出现了小于或等于0或大于n的值时，会导致1~n中出现空余位置，返
回结果必然在1~n中，因此对于A中出现了小于或等于0或大于n的值，可以不采取任何操作。经过以上分析可以得出算法流程：从A[0]开始遍历A,若0<A[i]<=n,则令B[A[i]-1]=1:否则不做操作。对A遍历结束后，开始遍历数组B,若能查找到第一个满足B[1]==0的下标1，返回i+1即为结果，此时说明A中未出现的最小正整数在1和n之间。若B[i]全部不为0，返
回i+1(跳出循环时i=n,1+1等于n+1),此时说明A中未出现的最小正整数是n+1。

代码

```
int findMissMin(int A[],int n)
{
        int i *B;
        B=(int *)malloc(sizeof(int)*n);
        memset(B,0,sizeof(int)*n);
        for(i=0;i<n;i++){
                if(A[i]>0&&A[i]<=n)
                                B[A[i]-1]=1;
        }
        for(i=0;i<n;i++){
```

```
            if(B[i]==0)
                break;
        }
            return i+1;
}
```

时间复杂度

时间复杂度$O(n)$,空间复杂度$O(n)$

1. 【2020统考真题】定义三元组(a,b,c)(a,b,c均为整数)的距离D=|a-b|+|b-c|+|c-d|给定3个非空整数集合S、S和S,按升序分别存储在3个数组中。请设计一个尽可能高效的算法，计算并输出所有可能的三元组(a,b,c)(a∈S,b∈S2,c∈S3)中的最小距离。例如S1={-1,0,9},S2={-25,-10,10,11},S3={2,9,17,30,41},则最小距离为2，相应的三元组为(9,10,9)。要求：1)给出算法的基本设计思想。2)根据设计思想，采用C语言或C++语言描述算法，关键之处给出注释3)说明你所设计算法的时间复杂度和空间复杂度。

思路

1)算法的基本设计思想
①使用D记录所有已处理的三元组的最小距离，初值为一个足够大的整数。
②集合S、S2和S3分别保存在数组A、B、C中。数组的下标变量i=j=k=0,当$i<|S_1|,j<|S_2|,k<|S_3|$(S表示集合S中的元素个数)，循环执行下面的a)~c)。
a)计算(A[i],B[j],C[k])的距离D:(计算D)
b)若D<Dmin,则Dmin=D:(更新D)
c)将A、B、C中的最小值的下标+1:
(对照分析：最小值为a,最大值为c,这里c
不变而更新a,试图寻找更小的距离D)
③输出Dmin,结束。

代码

```
# define INT_MAX 0x7fffffff
int abs_(int a){
        if(a<0) return -a;
        else return a;
}
bool xls_min(int a,int b,int c){
        if(a<=b&&a<=c) return true;
        return false;
```

```
}
int findMinofTrip(int A[],int n,int B[],int m,int C[],int p){
        int i=0,j=0,k=0,D_min = INT_MAX,D;
        while(i<n&&j<m&&k<p&&D_min>0)
        {
                D=abs_(A[i]-B[j])+abs_(B[j]-C[k])+abs_(C[k]-A[i]);
                if(D<D_min) D_min =D;
                if(xls_min(A[i],B[j],C[k])) i++;
                else if(xls_min(B[j],A[i],C[k])) j++;
                else k++;

        }
        return D_min;
}
```

时间复杂度

时间复杂度$O(n)$,空间复杂度$O(1)$