作业2 线性结构 参考答案

1. 从时间、空间及操作等层面比较线性表静态存储和动态存储两种存储结构的优缺点。

（略）

（2）输入一个已经按升序排序过的数组A和一个数字*x*，在数组A中查找两个数，使得它们的和正好是输入的*x*。

思路：

* 让指针指向数组的头部和尾部，相加，如果小于x，则增大头指针，如果大于则减小尾指针；
* 退出的条件，相等或者头部=尾部。

算法：

void function(int a[],int n,int x)

{

int i=0,j=n-1;

while(i!=j)

{

if(a[i]+a[j]==x)

{

printf("%d,%d",a[i],a[j]);

break;

}

a[i]+a[j]>x？j--:i++;

}

}

（3）将一个有序数组以一个点为中心进行旋转（前后颠倒）；例如{4，5 ，1，2 ，3}是{1,2,3,4,5}以3为中心的旋转。设计算法，查找一给定的元素x。

int bsearch\_rotate(int a[], int n, int t) //2次二分查找

{ int p = split(a, n); //找到分割位置

if (p == -1)

return bsearch\_first(a, n, t); //如果原数组有序，则直接二分查找即可

else

{

int left = bsearch\_first(a, p+1, t); //查找左半部分

if (left == -1)

{ //左半部分没有找到，则查找右半部分

int right = bsearch\_first(a+p+1, n-p-1, t); //查找右半部分

if (right != -1)

return right+p+1; //返回位置，注意要加上p+1

return -1;

}

return left; //左半部分找到，则直接返回

}

}

int bsearch\_rotate(int a[], int n, int t)   //1次二分查找

{       int low = 0, high = n-1;

     while (low <= high)

 {    int mid = low + (high-low) / 2;

         if (t == a[mid])

             return mid;

         if (a[mid] >= a[low]) //数组左半有序

{      if (t >= a[low] && t < a[mid])

                 high = mid - 1;

             else

                  low = mid + 1;

         }

else //数组右半段有序

{     if (t > a[mid] && t <= a[high])

                  low = mid + 1;

             else

                 high = mid - 1;

         }

     }

     return -1;

}

（4）已知线性表中的元素以值递增有序排列，并以单链表作存储结构。试写一算法，删除表中所有大于*x*且小于*y*的元素（若表中存在这样的元素）同时释放被删除结点空间。

Void Delete\_list（Lnode \*head，ElemType x，ElemType y）//代表头

{ Lnode \*p,\*q.\*r； /假设x,y都存在

if（head!->next）return 空表；

p=head->next；

while（p&&p->data<=x）//找第一个大于x的结点p,q为p的直接前驱

{ q=p;

p=p->next; }

if（!p）return ERROR1;

while（p&&p->data<=y）//继续找第一个左后一个y的结点p

p=p->next;

if（!p）return ERROR2;

r=q->next;

q->next=p; //将从q->next开始到p之前的结点从链表中删除

while(r!=p) //释放结点

{ q=r->next;

free(q);

r=q;

}

}