**实验五 分治法**

1. **实验目的**

1．熟练掌握分治法的设计思想。

2. 掌握运用分治法解决实际问题。

1. **实验要求**
2. 加深对分治算法的理解和运用。
3. 掌握分治算法的复杂度分析方法。
4. 能得到正确的程序运行结果
5. **主要仪器及耗材**

计算机及相关软件

1. **基础实验**
2. 任选一种语言实现合并排序算法。

**C语言：**

**#include<stdio.h>**

**void Merge(int \*A, int f, int m, int e) {**

**//两个有序数组的合并为一个数组**

**int temp[e-f+1];**

**int i,first=f,last=m+1;**

**for(i=0; i<(e-first+1)&&f<=m&&last<=e; i++) {**

**if(A[f]<=A[last]) {**

**temp[i]=A[f];**

**f++;**

**} else {**

**temp[i]=A[last];**

**last++;**

**}**

**}**

**while(f>m&&last<=e) {**

**temp[i]=A[last];**

**i++;**

**last++;**

**}**

**while(f<=m&&last>e) {**

**temp[i]=A[f];**

**i++;**

**f++;**

**}**

**for(i=0; first<=e; i++,first++) {**

**A[first]=temp[i];**

**}**

**}**

**void Merge\_Sort(int \*a,int f,int e ) {**

**int mid=(e+f)/2;**

**if(f<e) {**

**Merge\_Sort(a,f,mid);**

**Merge\_Sort(a,mid+1,e);**

**Merge(a,f,mid,e);**

**}**

**}**

**int main() {**

**int a[100];**

**int n;//n为输入的数组的长度；**

**printf("输入要排序的数组长度：\n");**

**scanf("%d",&n);**

**for(int i=0; i<n; i++) {**

**scanf("%d",&a[i]);**

**}**

**Merge\_Sort(a,0,n-1);**

**printf("排序后的数组元素顺序为：\n");**

**for(int i=0; i<n; i++) {**

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

**}**

**}**

2. 设A是n个非0实数构成的数组，设计一个算法重新排列数组中的数，使得负数都排在正数的前面.要求算法使用О（n）的时间和О（1）的空间

类似于快速排序的划分过程，从后往前把每个数与0比较，找到第一个负数A[p].从前向后把每个数与0比较，找到第一个整数A[q],如果p>q,则A[p]和A[q]交换。交换后如果p-q=1，算法停止；否则继续这个过程，或者找到下一对可以交换的数，或者扫描到p<=q停止。

**#include<stdio.h>**

**void intSort(int a[],int n){**

**int i=0,j=n-1;**

**int temp;**

**while(j>i){**

**if(a[j]>0)**

**j--;**

**if(a[i]<0)**

**i++;**

**if(j>i){**

**temp=a[i];**

**a[i++]=a[j];**

**a[j--]=temp;**

**}**

**}**

**for(int i=0;i<n;i++){**

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

**}**

**}**

**int main(){**

**int n;scanf("%d",&n);**

**printf("\n");**

**int a[n];**

**for(int i=0;i<n;i++){**

**scanf("%d",&a[i]);**

**}**

**intSort(a,n);**

**}**

3.任选一种语言实现快速排序算法。用该程序处理一批输入样本，来检验该算法的理论效率的正确性。

**#include <stdio.h>**

**int a[100005];**

**void quicksort(int left,int right)**

**{**

**int i,j,t;**

**if(left>right)**

**return ;**

**int temp=a[left];//存基准数**

**i=left;**

**j=right;**

**while(i!=j)**

**{**

**//顺序很重要，先从右向左找**

**while(a[j]>=temp&&i<j)**

**j--;**

**//在从左向右找**

**while(a[i]<=temp&&i<j)**

**i++;**

**if(i<j)//未相遇**

**{**

**t=a[j];**

**a[j]=a[i];**

**a[i]=t;**

**}**

**}**

**//将基准数归位**

**a[left]=a[i];**

**a[i]=temp;**

**quicksort(left,i-1);//递归左边**

**quicksort(i+1,right);//递归右边**

**}**

**int main()**

**{**

**int n;**

**scanf("%d",&n);**

**int i;**

**for(i=1; i<=n; i++)**

**scanf("%d",&a[i]);**

**quicksort(1,n);**

**for(i=1; i<=n; i++)**

**printf("%d\n",a[i]);**

**return 0;**

**}**

4. 设计一个算法，它能根据二叉树的中序和后序遍历序列列表构造该树。列表是由n个标为0,1,2，…,n-1的结点构成，这个算法对无解的输入也应正确识别。

#define MAXN 50

typedef struct TreeNode \*BinTree;

struct TreeNode

{

char data;

BinTree Left;

BinTree Right;

};

BinTree CreateBinTree(char \*Pre,char \*In,int Len)

{/\*按先序遍历序列和中序遍历序列构建二叉树\*/

BinTree T;

int i;

if(!Len) return NULL; //递归的终止条件

/\*建立当前根结点\*/

T=(BinTree)malloc(sizeof(struct TreeNode));

T->data=Pre[0];

for(i=0;i<Len;i++) //在中序遍历序列中查找根结点

if(Pre[0]==In[i]) break; //找到就返回

/\*递归构建左右子树\*/

T->Left=CreateBinTree(Pre+1,In,i);

T->Right=CreateBinTree(Pre+i+1,In+i+1,Len-i-1);

return T;

}

int Height(BinTree T)

{/\*求树的高度\*/

int THeight,LHeight,RHeight;

if(!T)

THeight=0;

else{

LHeight=Height(T->Left);

RHeight=Height(T->Right);

THeight=(LHeight>RHeight)?LHeight:RHeight;

THeight++;//树的高度为左右子树高较大者加1

}

return THeight;

}

int main()

{

int N;

char Pre[MAXN+1],In[MAXN+1];//用来保存先序遍历序列和中序遍历序列

BinTree T=NULL;

scanf("%d",&N);

scanf("%s\n%s",Pre,In);

T=CreateBinTree(Pre,In,N);

printf("%d\n",Height(T));

return 0;

}

5. 设A是n个不等的整数数组，n=2k,设计一个分治算法找出A中的最大数max和最小数min。

算法的设计思想：如果A中只有2个数，那么比较1次就可以确定最大数与最小数。否则，将A划分成相等的两个子集A1与A2.用算法MaxMin递归地在A1和A2中找最大与最小.令a1,a2分别表示A1与A2中的最大数，b1与b2分别表示A1与A2中的最小数，那么max{a1,a2}与min{b1,b2}就是所需要的结果。算法的伪码如下：

算法 MaxMin(A,s,l)

//求数组A[s..l]的最大元素max与最小元素min

//输入：数组A和数组首元素的地址和最后一个元素的地址

//输出：数组中的最大数max和最小数min

if s=l-1

then if A[s] < A[l] then max←A[l];min←A[s];

else min←A[l];max←A[s]

else

k←(s+l)/2

MaxMin(A,s,k)

a1←max;b1←min

MaxMin(A,k+1,l)

a2←max;b2←min

if a1>a2 then max←a1

else max←a2

if b1<b2 then min←b1

else min←b2

return max,min

设比较次数为,则

用换元得到

-2

**#include<stdio.h>**

**#include<algorithm>**

**using namespace std;**

**void Fx(int a[],int low,int high,int &maxe,int &mine)**

**{**

**if(low==high)**

**{**

**maxe=a[low];**

**mine=a[low];**

**}**

**else if(high-low==1)**

**{**

**maxe=max(a[low],a[high]);**

**mine=min(a[low],a[high]);**

**}**

**else**

**{**

**int mid=(low+high)/2;**

**int lmaxe,lmine;**

**Fx(a,low,mid,lmaxe,lmine);**

**int rmaxe,rmine;**

**Fx(a,mid+1,high,rmaxe,rmine);**

**maxe=max(lmaxe,rmaxe);**

**mine=min(lmine,rmine);**

**}**

**}**

**int main(){**

**int n;**

**scanf("%d",&n);**

**int a[n];**

**for(int i=0;i<n;i++){**

**scanf("%d",&a[i]);**

**}**

**int maxe;**

**int mine;**

**Fx(a,0,n-1,maxe,mine);**

**printf("max=%d,min=%d",maxe,mine);**

**}**

6. 有n个人，其中某些人是诚实的，其他人可能会说谎。现在需要进行一项调查，该调查由一系列测试构成。每次测试如下进行：选2个人，然后提问：对方是否诚实？每个人的回答只能是“是”或者是“否”。假定在这些人中，所有诚实的人回答都是正确的，而其他人的回答则不能肯定是否正确。如果诚实的人数>n/2,试设计一个调查算法，以最小的测试次数从其中找出诚实的人。

算法设计思想：采用芯片测试算法Test.将n个人分成组，每组2个人，如果2个人的回答相同，则他们或都是诚实的，或都说了谎。如果回答不同，则至少1个人说了谎。回答相同的组中任意保留1人进入下一轮，回答不同的2人都淘汰。当有轮空的人时需要单独处理。根据芯片测试算法的分析，经过一轮测试后，人数至少减半，且进入下一轮的诚实的人数多于可能说谎的人数，该算法的时间复杂度为.

算法：Test(n)

//输入：n个人构成的数组，其中诚实的人至少比说谎的人多1

//输出：1个诚实的人

k←n

while k>3 do

将人数分成k/2组 //如有轮空的人，单独测试，根据情况丢弃或保留

for i=1 to k/2 do

if 2个回答相同，则任取1人留下

else 2人同时丢弃

k←剩下的人数

if k=3

then 任取2人 进行测试

if 2人回答不同，留没测过的人

else 任取1人留下

if k=2 or 1 then 任取1人

1. **注意事项**

1.注意分治法是一种常见的算法设计技术，最常见的分治法是二分法，当然也可以设计成三分法或者更复杂的划分方法，对输入集合的划分一般采用均衡的原则。

2.注意子问题的边界。