**算法实习报告：实验四 分治算法的应用**

题目一：

1、a.为一个分治算法编写伪代码，该算法同时求出一个n元数组的最大元素和最小元素的值。

b.请拿该算法与解同样问题的蛮力算法做一个比较。

解答：

a、

minmax\_element(A[l..r], min, max){

if(r<0) return;      // 空数组，没有最大最小元

if(r==0)

max=min=A[0]; // 只有一个元素时

if(r==1) {

=A[0]<=A[1]?A[0]:A[1]; max=A[0]<=A[1]?A[1]:A[0];

}  // 有两元素时

else {

　　 m=int((l+r)/2);   // 去中间值，把数组分成两个部分

　　 minmax\_element(l, m, fMin, fMax);   // 递归解决前一部分

　　 minmax\_element(m+1, r, sMin, sMax); // 递归解决后一部分

　　 max= max(fMax, sMax); // 从两部分的两个最大值中选择大值

　　 min= min(fMin, sMin); // 从两部分的两个最小值中选择小值

}

}

b、

该算法的时间复杂度为：  
t(n)=2\*t(n/2)+2 n>2  
t(1)=0 t(2)=1  
设n=2^k，则n/2=2^(k-1)  
t(n)=t(2^k)=2\*t[2^(k-1)]+2  
=2[2\*t(2^(k-2))+2]+2  
=2^2\*t[2^(k-2))]+2^2+2  
=2^2[2\*t[2^(k-3)]+2]+2^2+2  
=2^3\*t[2^(k-3)]+2^3+2^2+2  
=...  
=2^(k-1)\*t(2)+2^(k-1)+2^(k-2)+...+2   // t(2)=1  
=2^(k-1)+2^(k-1)+2^(k-2)+...+2 // 后面部分为等比数列求和  
=2^(k-1)+2^k-2    // 2^(k-1)=n/2, 2^k=n  
=n/2+n-2  
=3\*n/2-2

蛮力算法如：

simple\_minmax(A[0..n]){

    max=min=A[0];

    for(i=0; i<n; ++i) {

　　  if(A[i]>max)

            max=A[i];

　　  if(A[i]<min)

            min=A[i];

    }

}

时间复杂度：t(n)=2\*(n-1)。

算法 minmax\_element 的时间复杂度为3\*n/2-2，simple\_minmax 的时间复杂度为2n-2，都属于Θ(n)复杂度。

但比较可得，minmax\_element 减少了不必要的比较，速度上比 simple\_minmax 的快一些。

题目二：

2、为最近对问题的一维版本设计一个直接基于分治技术的算法,并确定它的时间复杂度。（最近点对问题定义：已知上m个点的集合，找出对接近的一对点。）

解答：

ClosestNumber(A[l..r])

    if(r=l)

        return ∞

    else{

        if(r－l=1)

            return A[r]－A[l]

        else return min{ClosestNumber(A[l… (l+r)/2 ]),ClosestNumber(A[(l+r)/2 ...r]),A[ (l+r)/2 +1]－A[ (l+r)/2 ]

    }

}

设递归的时间效率为T(n):

对n=2k, 则: T(n)=2T(n/2)+c

利用主定理求解.T(n)=Θ(n)

题目三：

1. 设计一个分治算法来计算二叉树的层数.(空树返回0,单顶点树返回1),并确定它的时间复杂度.

解答：

CONUT\_DEEP(Tree ht){

    if(ht = null)

        return 0;

    else{

        return max{CONUT\_DEEP(TL),CONUT\_DEEP(TR)}+1

    }

}