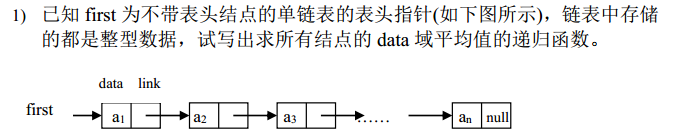
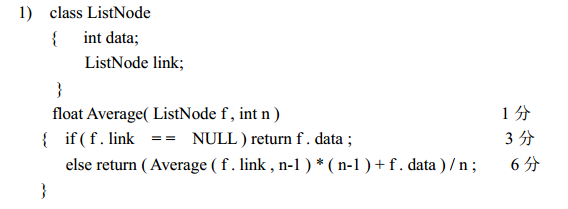
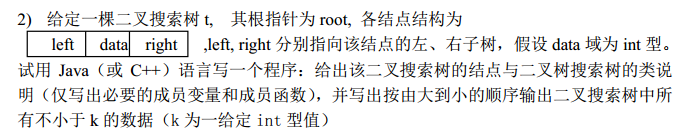
======================



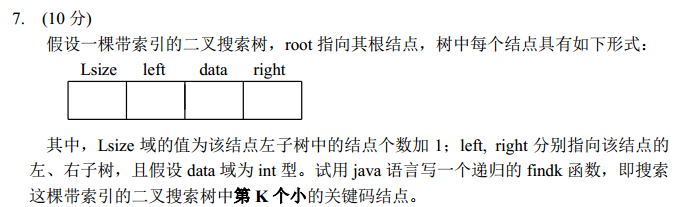


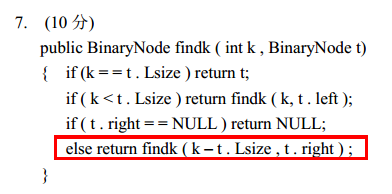
===================





=====================





{

public Comparable findK( BinaryNode root, int k)

{

if( root==null) return null;//空

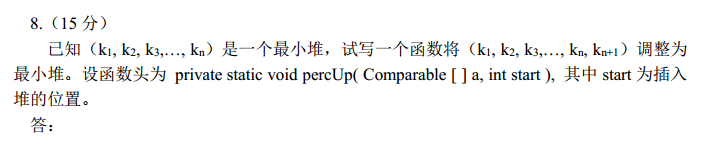
if( k< root. leftSize) //在左子树

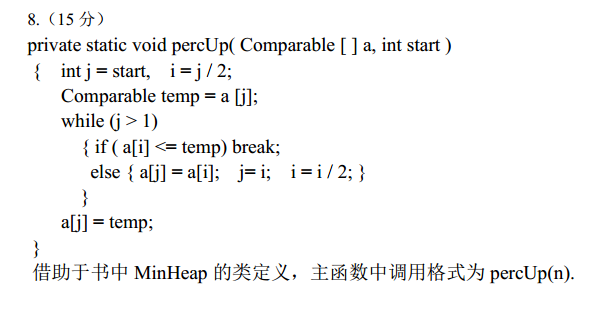
findK( root. left, k);

else if( k>root. leftSize) //在右子树

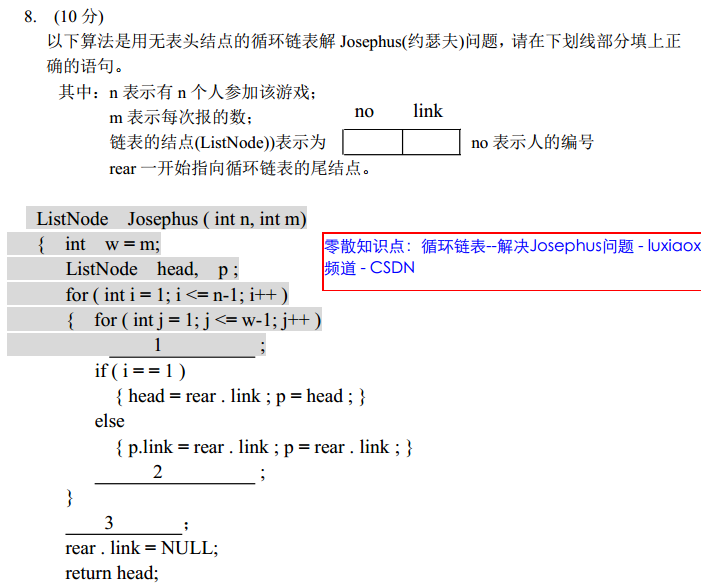
findK( root. right, k-root. leftSize);//注意减去

else return root.element;

}

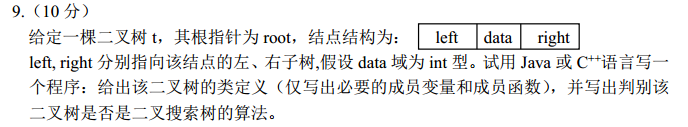


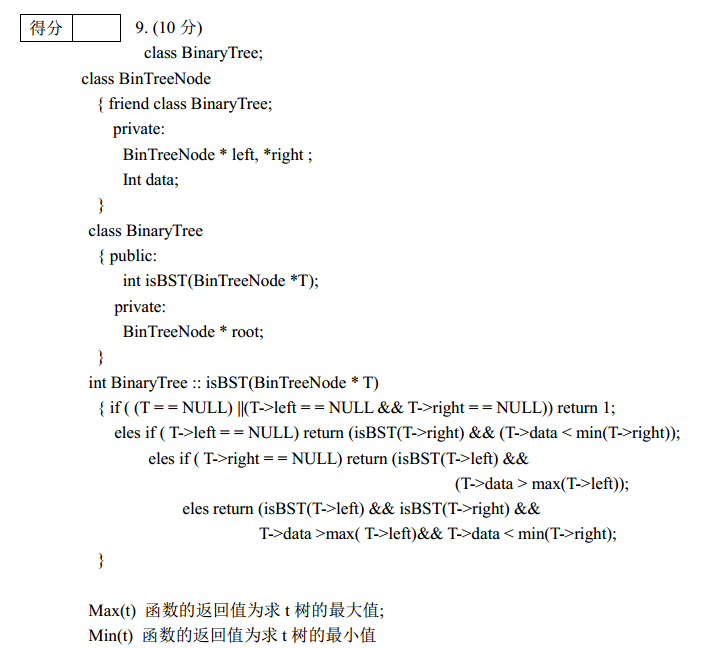
============





===============

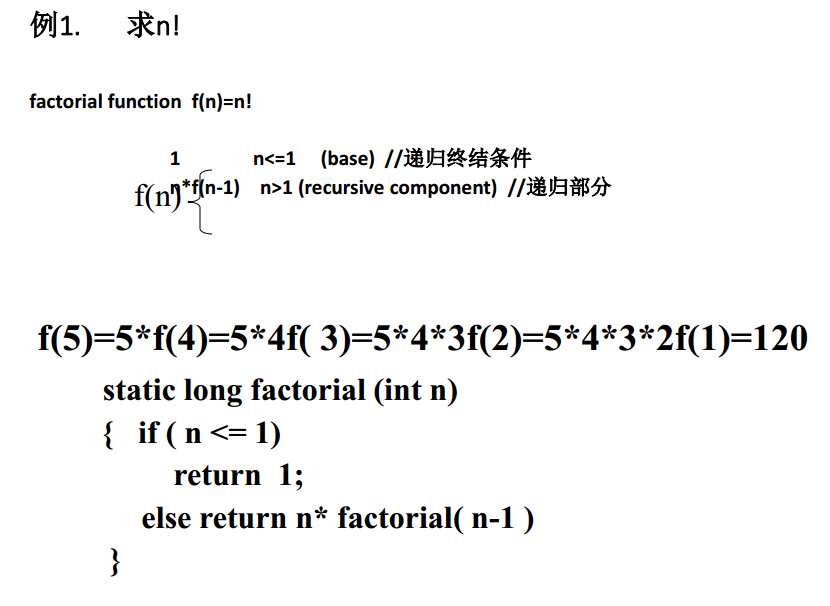




============

递归

===================



复习例题---在O(n)时间内实现将负数排在所有非负数之前。

void sort ( float [ ] a, int n )

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

while ( i != j )

{ while ( a[j] >= 0.0 && i < j ) j-- ;

while ( a[i] < 0 && i < j ) i++ ;

float temp = a[i] ; a[i] = a[j]; a[j] = temp;

j-- ; i++ ;

}

}

左孩子右子树

class TreeNode:

T data;

TreeNode \*firstchild, \*nextsibling;

class Tree:

TreeNode \* root, \*current;

给定一个由英文字母组成的字符串S (假设S用数组实现), 编制一个递归函数, 测试S是否为回文串( a palindrome ), “回文串”是指从左向右读该字符串和从右向左读该字符串完全相同，例如 “noon”, “radar”等。

int palindrome(char s[ ], int start, int end )

{

if ( start >= end ) return 1;

else if ((s[start] = = s[end]) && palindrome(s, start +1, end-

1 ))return 1;

else return 0;

}