# 第一题

#pragma once  
#include "BTree.h"  
int SumsLeafNodes(BTree<int>\* head)  
{  
 if (head == nullptr) return 0;  
 if (head->Left == nullptr && head->Right == nullptr)   
 return head->value;  
 return SumsLeafNodes(head->Left) + SumsLeafNodes(head->Right);  
}

# 第二题

#pragma once  
#include "BTree.h"  
int SumsBiggerNodes(BTree<int>\* head, int k)  
{  
 if (head == nullptr) return 0;  
 return SumsBiggerNodes(head->Left, k) +   
 SumsBiggerNodes(head->Right, k) +   
 head->value >= k ? 1 : 0;  
}

# 第三题

#pragma once  
#include "BTree.h"  
template<typename T>  
int GetDepth(BTree<T>\* head, T x)  
{  
 if (head == nullptr) return 0;  
 if (head->value == x) return 1;  
 int leftDepth = GetDepth(head->Left, x);  
 if (leftDepth > 0) return leftDepth + 1;  
 int rightDepth = GetDepth(head->Right, x);  
 if (rightDepth > 0) return rightDepth + 1;  
 return 0;  
}

# 第四题

#pragma once  
int FindSubscript(int\* array, int length)  
{  
 if (array[0] >= length || array[length - 1] < 0) return -1;  
 int mid = length / 2;  
 int bias = 0;  
 while (length != 0)  
 {  
 if (array[mid] == mid) return mid;  
 else if (array[mid] > mid)  
 {  
 mid = (mid + bias) / 2;  
 }  
 else  
 {  
 int tmpBias = mid;  
 mid = mid + std::max(1, (mid - bias) / 2);  
 bias = tmpBias;  
 }  
 length /= 2;  
 }  
 return -1;  
}

# 第五题

#pragma once  
template<typename T>  
T\* TriSearch(T\* First, T target, int length, bool (\*compare)(T a,T b))  
{  
 T\* Second = First + length / 3;  
 T\* Third = First + length \* 2 / 3;  
 T\* End = First + length;  
 if (length <= 2)  
 return \*First == target ? First : \*(End - 1) == target ? End - 1 : nullptr;  
 if (compare(target, \*First) || compare(\*(End - 1), target))  
 return nullptr;  
 if (compare(target, \*Second))  
 return TriSearch(First, target, length / 3, compare);  
 else if (compare(target,\*Third))  
 return TriSearch(Second, target, Third - Second, compare);  
 else  
 return TriSearch(Third, target, End - Third, compare);  
}

递归：T(n) = T(n/3) + O(1) 因此时间复杂度为

# 第六题

#pragma once  
int SepaExpNum(int num)  
{  
 if (num <= 2) return 1;  
 else  
 {  
 int sum = 0;  
 for (int i = 2;i <=num ;i++)  
 {  
 if (num % i == 0)  
 {  
 sum += SepaExpNum(num / i);  
 }  
 }  
 return sum;  
 }  
}