### segment-tree

#include<stdio.h>  
#define MAX\_LEN 1000  
  
void bulid\_tree(int arr[], int tree[], int node, int start, int end) {  
 if (start == end) {  
 tree[node] = arr[start];  
 }  
 else {  
 int mid = (start + end) / 2;  
 int left\_node = 2 \* node + 1;  
 int right\_node = 2 \* node + 2;  
  
 bulid\_tree(arr, tree, left\_node, start, mid);  
 bulid\_tree(arr, tree, right\_node, mid+1, end);  
 tree[node] = tree[left\_node] + tree[right\_node];  
 }  
}  
  
void update\_tree(int arr[], int tree[], int node, int start, int end, int idx, int val) {  
 if (start == end) {  
 arr[idx] = val;  
 tree[node] = val;  
 }  
 else {  
 int mid = (start + end) / 2;  
 int left\_node = 2 \* node + 1;  
 int right\_node = 2 \* node + 2;  
 if (idx >= start && idx <= mid) {  
 update\_tree(arr, tree, left\_node, start, mid, idx, val);  
 }  
 else {  
 update\_tree(arr, tree, right\_node, mid+1, end, idx, val);  
 }  
 tree[node] = tree[left\_node] + tree[right\_node];  
 }  
}  
  
int query\_tree(int arr[], int tree[], int node, int start, int end, int L, int R) {  
 printf("start = %d\n", start);  
 printf("end = %d\n", end);  
 printf("\n");  
  
 if (R < start || L > end) {  
 return 0;  
 }  
 else if (L <= start && end <= R) {  
 return tree[node];  
 }  
 else if (start == end) {  
 return tree[node];  
 }  
 else {  
 int mid = (start + end) / 2;  
 int left\_node = 2 \* node + 1;  
 int right\_node = 2 \* node + 2;  
 int sum\_left = query\_tree(arr, tree, left\_node, start, mid, L, R);  
 int sum\_right = query\_tree(arr, tree, right\_node, mid+1, end, L, R);  
 return sum\_left + sum\_right;  
 }  
}  
  
int main()  
{  
 int arr[] = {1, 3, 5, 7, 9, 11};  
 int size = 6;  
 int tree[MAX\_LEN] = {0};  
  
 bulid\_tree(arr, tree, 0, 0, size-1);  
  
 int i;  
 for(i = 0; i <= 14; i++) {  
 printf("tree[%d] = %d\n", i, tree[i]);  
 }  
  
 printf("\n");  
 update\_tree(arr, tree, 0, 0, size-1, 4, 6);  
 for(i = 0; i <= 14; i++) {  
 printf("tree[%d] = %d\n", i, tree[i]);  
 }  
  
 printf("\n");  
 int s = query\_tree(arr, tree, 0, 0, size-1, 2, 5);  
 printf("s = %d\n", s);  
  
 return 0;  
}