2-3

int get\_L\_R(int arr[], int size,int x)

{

int middle;

int left = 0, right = x;

while (left <=right ) {

middle = (left + right) / 2;

if (x == arr[middle])

return middle;

if (x >= arr[middle])

left = middle + 1;

else

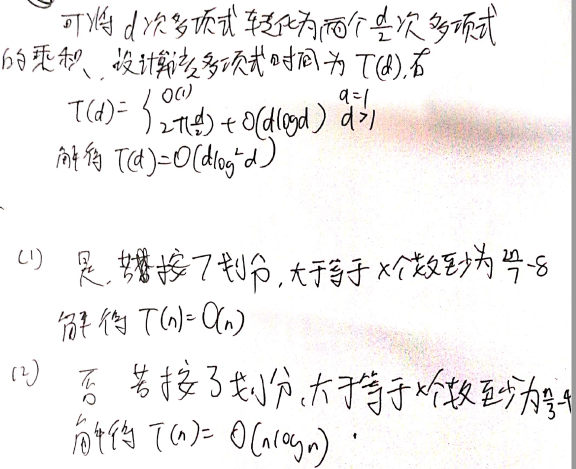
right = middle - 1;

}

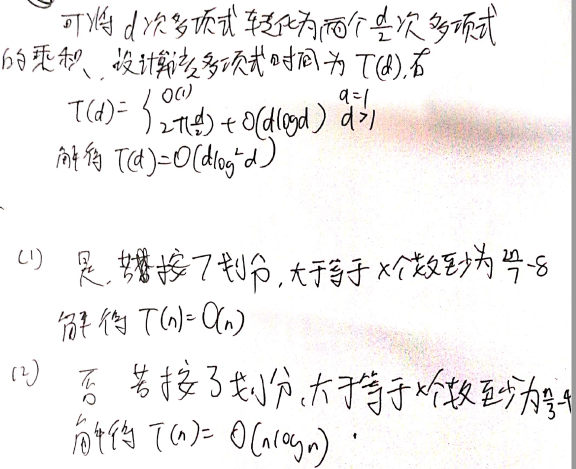
return left, right;

}

2-7



2-25



2-28

#include <iostream>

#include <vector>

using namespace std;

void guibing(vector<int>& a, vector<int>& b, vector<int>& c, int l1, int l2)

{

int i = 0, j = 0, temp = 0;

while (i < l1 && j < l2 && temp <= (l1 + l2) / 2)

{

if (a[i] < b[j])

c[temp] = a[i++];

else

c[temp] = b[j++];

temp++;

}

if (i == l1 && temp <= (l1 + l2) / 2)

{

while (j < l2)

c[temp++] = b[j++];

}

else if (j == l2 && temp <= (l1 + l2) / 2)

{

while (i < l1)

c[temp++] = a[i++];

}

return;

}

double findMedianSortedArrays(vector<int>& nums1, vector<int>& nums2) {

int l1 = nums1.size();

int l2 = nums2.size();

vector<int>temp(l1 + l2, 0);

guibing(nums1, nums2, temp, l1, l2);

if ((l1 + l2) % 2 == 0)

{

return (double(temp[(l1 + l2) / 2]) + double(temp[(l1 + l2) / 2 - 1])) / 2;

}

else

return double(temp[(l1 + l2) / 2]);

}