Q1)You are transporting some boxes through a tunnel, where each box is a parallelepiped, and is characterized by its length, width and height.

The height of the tunnel 41 feet and the width can be assumed to be infinite. A box can be carried through the tunnel only if its height is strictly less than the tunnel's height. Find the volume of each box that can be successfully transported to the other end of the tunnel. Note: Boxes cannot be rotated.

Input Format

The first line contains a single integer n, denoting the number of boxes.

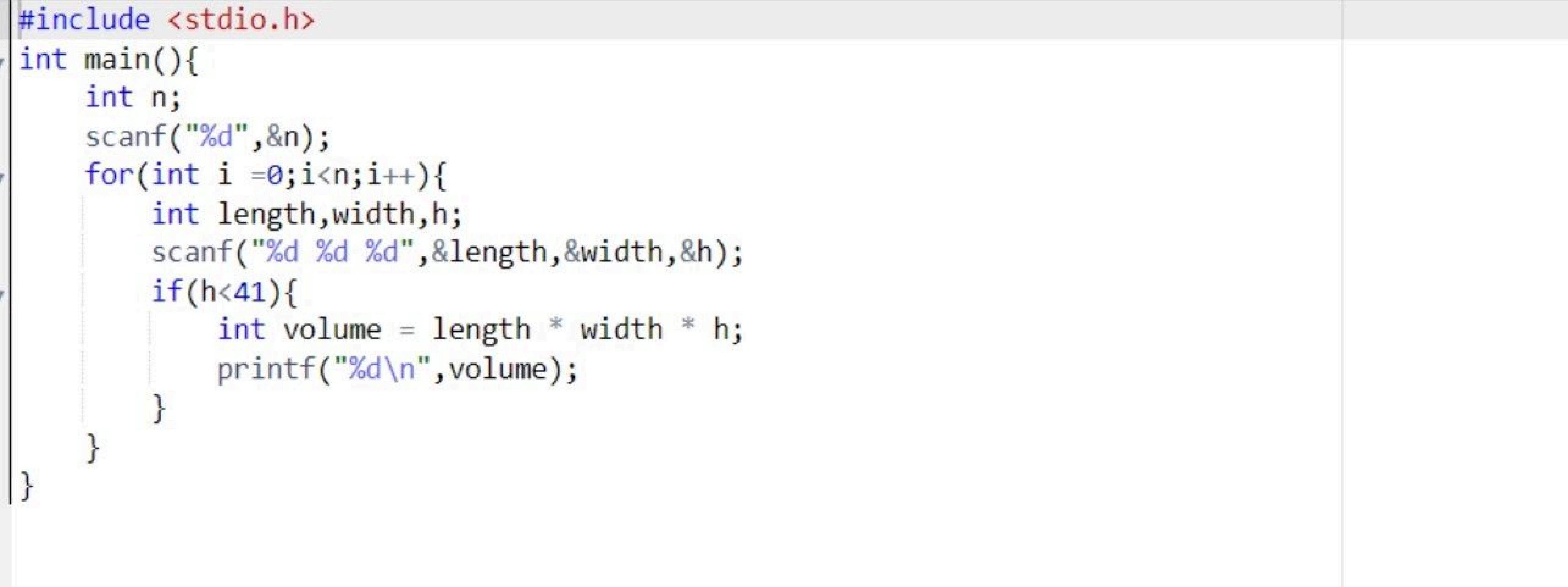
n lines follow with three integers on each separated by single spaces - lengthi, widthi and heighti which are length, width and height in feet of the i-th box.

Constraints

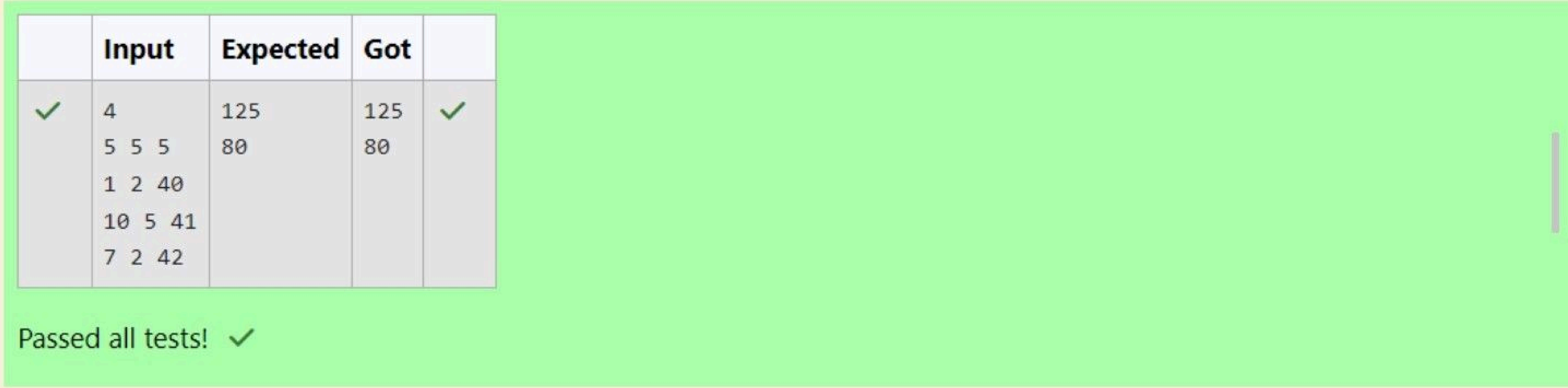
1 ≤ n ≤ 100

1 ≤ lengthi, widthi, heighti ≤ 100

Output Format



```c
#include <stdio.h>
int main(){
    int n;
    scanf("%d",&n);
    for(int i =0;i<n;i++){
        int length,width,h;
        scanf("%d %d %d",&length,&width,&h);
        if(h<41){
            int volume = length * width * h;
            printf("%d\n",volume);
        }
    }
}
```



	Input	Expected	Got	
✓	4 5 5 5 1 2 40 10 5 41 7 2 42	125 80	125 80	✓

Passed all tests! ✓

Q2)You are given n triangles, specifically, their sides ai, bi and ci. Print them in the same style but sorted by their areas from the smallest one to the largest one. It is guaranteed that all the areas are different.

The best way to calculate a volume of the triangle with sides a, b and c is Heron's formula:

S = Ö p * (p – a) * (p - b) * (p – c) where p = (a + b + c) / 2.

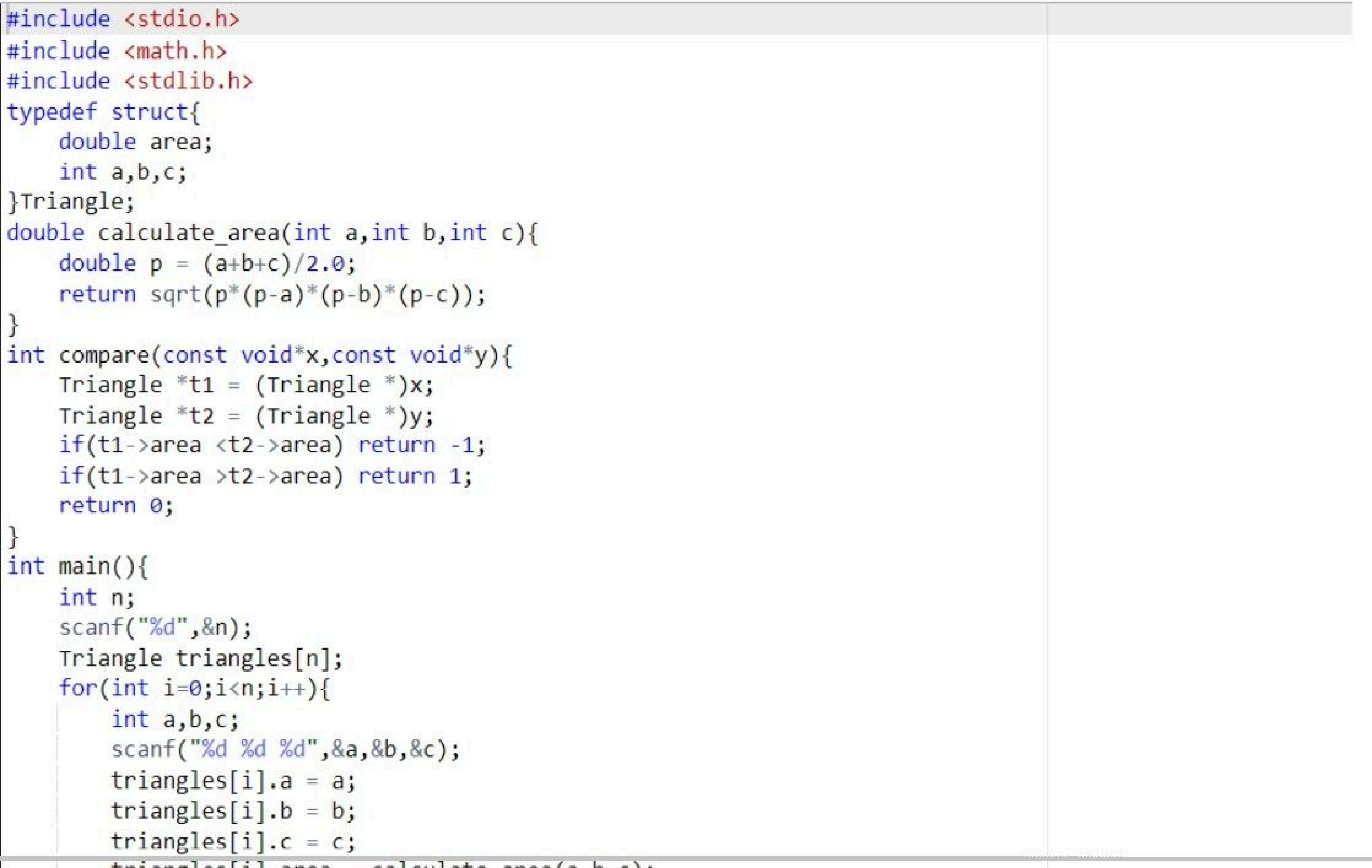
Input Format

First line of each test file contains a single integer n. n lines follow with ai, bi and ci on each separated by single spaces.

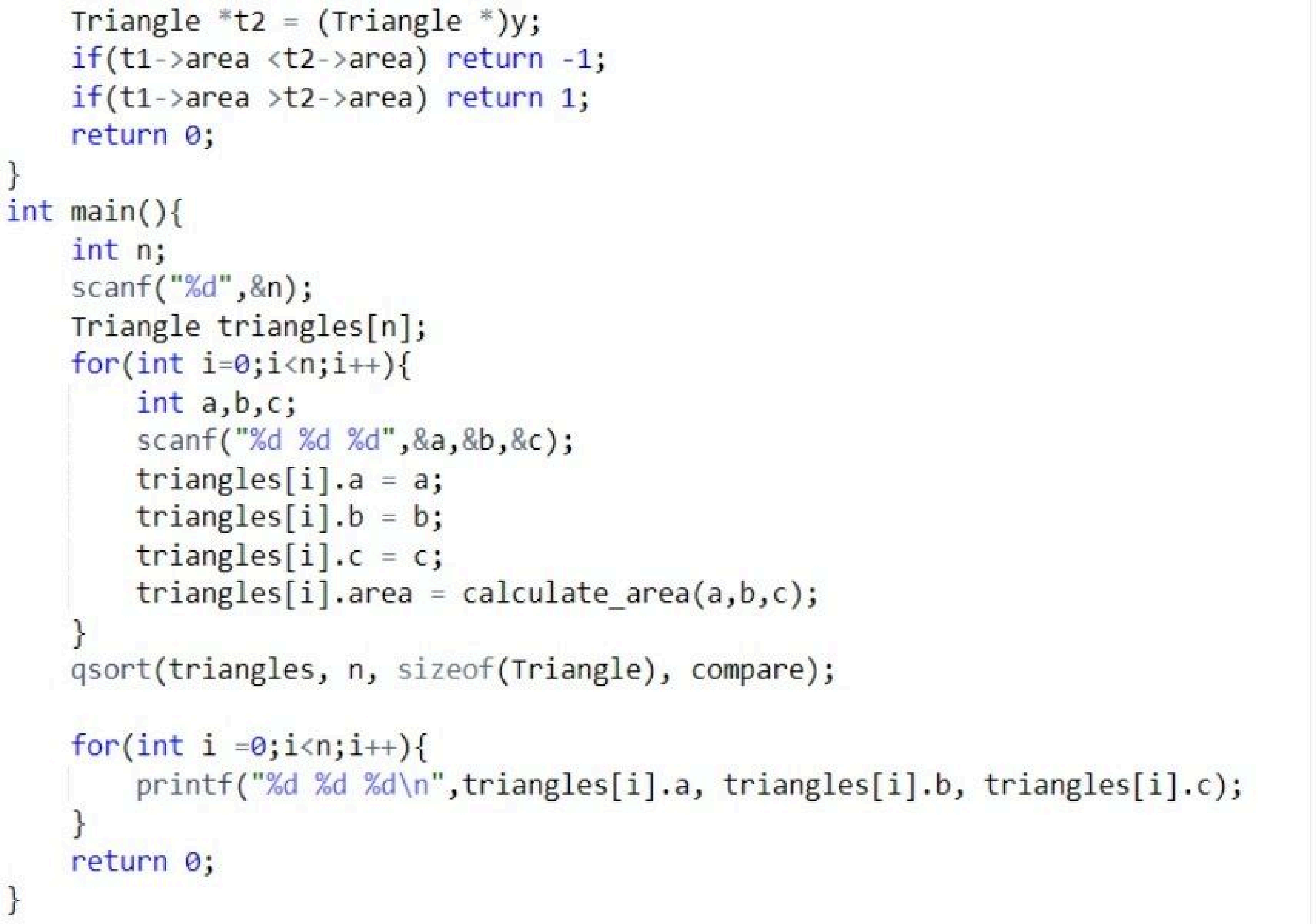
Constraints

1 ≤ n ≤ 100

1 ≤ ai, bi, ci ≤ 70

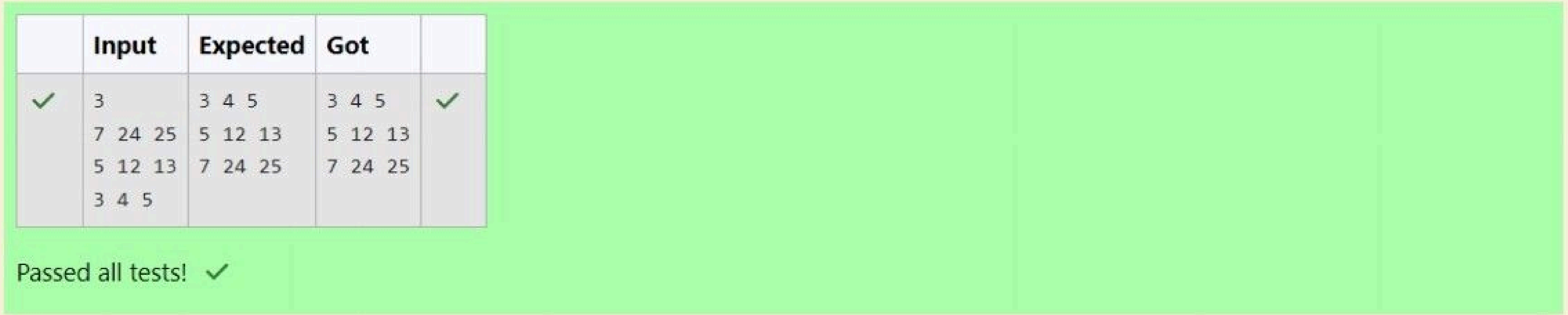


```c
#include <stdio.h>
#include <math.h>
#include <stdlib.h>
typedef struct{
    double area;
    int a,b,c;
}Triangle;
double calculate_area(int a,int b,int c){
    double p = (a+b+c)/2.0;
    return sqrt(p*(p-a)*(p-b)*(p-c));
}
int compare(const void*x,const void*y){
    Triangle *t1 = (Triangle *)x;
    Triangle *t2 = (Triangle *)y;
    if(t1->area <t2->area) return -1;
    if(t1->area >t2->area) return 1;
    return 0;
}
int main(){
    int n;
    scanf("%d",&n);
    Triangle triangles[n];
    for(int i=0;i<n;i++){
        int a,b,c;
        scanf("%d %d %d",&a,&b,&c);
        triangles[i].a = a;
        triangles[i].b = b;
        triangles[i].c = c;
        triangles[i].area = calculate_area(a,b,c);
```



```c
    Triangle *t2 = (Triangle *)y;
    if(t1->area <t2->area) return -1;
    if(t1->area >t2->area) return 1;
    return 0;
}
int main(){
    int n;
    scanf("%d",&n);
    Triangle triangles[n];
    for(int i=0;i<n;i++){
        int a,b,c;
        scanf("%d %d %d",&a,&b,&c);
        triangles[i].a = a;
        triangles[i].b = b;
        triangles[i].c = c;
        triangles[i].area = calculate_area(a,b,c);
    }
    qsort(triangles, n, sizeof(Triangle), compare);

    for(int i =0;i<n;i++){
        printf("%d %d %d\n",triangles[i].a, triangles[i].b, triangles[i].c);
    }
    return 0;
}
```



	Input	Expected	Got	
✓	3	3 4 5	3 4 5	✓
	7 24 25	5 12 13	5 12 13	
	5 12 13	7 24 25	7 24 25	
	3 4 5			

Passed all tests! ✓