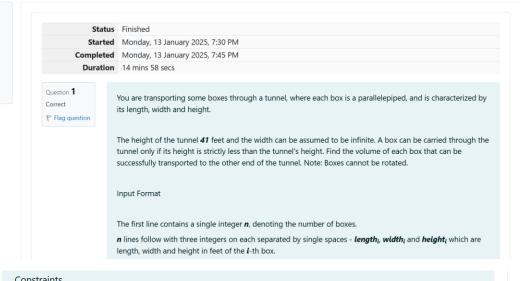
REC-CIS

GE23131-Programming Using C-2024





Constraints
1 ≤ n ≤ 100 1 ≤ length _i , width _i , height _i ≤ 100
Output Format
For every box from the input which has a height lesser than 41 feet, print its volume in a separate line.
Sample Input 0
4
5 5 5
1 2 40
10 5 41
7 2 42
Sample Output 0
125
80

Explanation 0

The first box is really low, only 5 feet tall, so it can pass through the tunnel and its volume is 5 x 5 x 5 = 125.

The second box is sufficiently low, its volume is $1 \times 2 \times 4 = 80$.

The third box is exactly 41 feet tall, so it cannot pass. The same can be said about the fourth box.

Answer: (penalty regime: 0 %)

```
1 #include<stdio.h>
    int main()
3 ₹ {
4
        int n;
        scanf("%d", &n);
for(int i = 0; i < n; i++)
 5
6
7 ▽
         int lenght,width,height;
scanf("%d %d %d",&lenght,&width,&height);
if(height < 41)</pre>
8
9
10
11 *
12
                 int volume = lenght * width * height;
                  printf("%d\n",volume);
13
14
15
         }
         return 0;
16
17 }
```

Question **2**Correct

Flag question

You are given n triangles, specifically, their sides a_i , b_i and c_i . 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 \boldsymbol{a} , \boldsymbol{b} and \boldsymbol{c} is Heron's formula:

$$S = \ddot{O} p * (p - a) * (p - b) * (p - c)$$
 where $p = (a + b + c) / 2$.

Input Format

First line of each test file contains a single integer n. n lines follow with a_i , b_i and c_i on each separated by single spaces.

Constraints

 $1 \le n \le 100$

 $1 \leq a_i, \, b_i, \, c_i \leq 70$

 $a_i + b_i > c_i$, $a_i + c_i > b_i$ and $b_i + c_i > a_i$

Output Format

Print exactly n lines. On each line print 3 integers separated by single spaces, which are a_i , b_i and c_i of the corresponding triangle.

Sample Input 0

3

7 24 25

5 12 13

3 4 5

Sample Output 0

3 4 5

5 12 13

7 24 25

Explanation 0

The square of the first triangle is **84**. The square of the second triangle is **30**. The square of the third triangle is **6**. So the sorted order is the reverse one.

Answer: (penalty regime: 0 %)

```
1 #include<stdio.h>
    #include<math.h>
3
    #include<stdlib.h>
4 v typedef struct{
5
      int a, b, c;
6
       double area;
8 }triangle;
9 double calculate_area(int a,int b,int c)
10 🔻 {
11
        double p = (a+b+c)/2.0;
12
        return sqrt(p*(p-a)*(p-b)*(p-c));
13
14 int compare(const void *t1, const void *t2)
15 ₹ {
16
        triangle *tri1 = (triangle*)t1;
       triangle *tri2 = (triangle*)t2;
17
       if(tri1 -> area < tri2 -> area)
18
19
       return -1;
20
      if(tri1 -> area > tri2 -> area)
21
       return 1;
22
        return 0;
23 }
24 int main()
```

```
25 ₹ {
26
        int n;
        scanf("%d", &n);
27
28
        triangle triangles[n];
29
        for(int i =0; i <n ;i++)</pre>
30 ₹
            int a,b,c;
31
32
           scanf("%d %d %d",&a,&b,&c);
           triangles[i].a = a;
33
34
           triangles[i].b = b;
           triangles[i].c = c;
35
            triangles[i].area = calculate_area(a,b,c);
36
37
38
        qsort(triangles,n,sizeof(triangle),compare);
        for(int i = 0; i < n; i++)</pre>
39
40 🔻
41
            printf("%d %d %d\n",triangles[i].a,triangles[i].b,triangles[i].c);
42
43
        return 0;
44 }
```



Finish review