# GE23131-Programming Using C-2024-2025 Week 14

ABENANTHAN P 240701005 CSE-'A'

# **Structures and Unions**

#### Question 1

# **Boxes Through Tunnel**

#### **Problem Statement**

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 - length, width; and height; which are length, width and height in feet of the i-th box.

#### **Constraints**

 $1 \le n \le 100$ 

 $1 \le \text{length}$ , width, height  $\le 100$ 

#### **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

```
1 2 40
```

10 5 41

7 2 42

### Sample Output 0

125

80

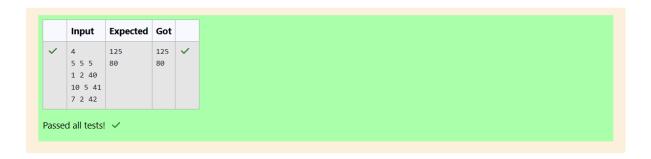
# **Explanation 0**

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

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 %)
      #include<stdio.h>
       #include<stdlib.h>
       #define MAX_HEIGHT 41
       struct box
   6 ₹ {
            int len;
   8
           int wid;
   9
           int hgt;
   10 };
   11
   12 typedef struct box box;
   13
   14 v int get_volume(box b){
   15
           return b.len * b.wid * b.hgt;
   16
   17 v int is_lower_than_max_height(box b){
   18
           return b.hgt < MAX_HEIGHT;</pre>
   19
      int main()
   20
   21 • {
           int n;
   22
           scanf("%d",&n);
box*boxes=(box *)malloc(n*sizeof(box));
   23
   24
           for(int i=0;i<n;i++){
    scanf("%d %d %d", &boxes[i].len,&boxes[i].wid,&boxes[i].hgt);</pre>
   25 1
   26
   27
                if(is_lower_than_max_height(boxes[i])){
   28
                    printf("%d\n",get_volume(boxes[i]));
   29
   30
   31
           free(boxes);
   32
   33
           return 0;
   34 }
```



# Question 2

# **Small Triangles , Large Triangles**

#### **Problem Statement**

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:

$$S= \sqrt{p + (p-a) + (p-b) + (p-c)}$$
 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

 $1 \le n \le 100$ 

 $1 \le a$ , b, c  $\le 70$ 

ai + bi > ci , ai + ci > bi and bi + ci > ai

#### **Output Format**

Print exactly n lines. On each line print 3 integers separated by single spaces, which are di, bi and ci of the corresponding triangle.

### Sample Input 0

7 24 25

5 12 13

345

### Sample Output 0

345

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>
        struct triangle
    4 + {
            int a;
int b;
            int c;
        typedef struct triangle triangle;
  11 v double area(triangle t){
12 double s=(t.a+t.b+t.c)/2.0;
  13
14
             return sqrt(s*(s-t.a)*(s-t.b)*(s-t.c));
  if (area(tr[j])>area(tr[j+1])){
    triangle temp=tr[j];
    tr[j]=tr[j+1];
    tr[j+1]=temp;
   18 •
   19
   20
   21
   22
                      }
   23
                 }
  24
25
            }
   26 v int main(){
   27
            int n;
scanf("%d",&n);
   28
            for(int i=0;i<n;i++){
    scanf("%d %d %d",&tr[i].a,&tr[i].b,&tr[i].c);
   29
   30 +
  31
32
           sort_triangles_by_area(tr,n);
for(int i=0;i<n;i++){
    printf("%d %d %d \n",tr[i].a,tr[i].b,tr[i].c);
}</pre>
   33
34 v
   36
   37
             return 0;
  38
39
```

```
Input Expected Got

3 3 4 5 3 4 5 5 12 13 5 12 13 5 12 13 5 12 13 7 24 25 7 24 25 7 24 25

Passed all tests! 

V

Passed all tests!
```