1.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 - $length_i$, $width_i$ and $height_i$ which are length, width and height in feet of the *i*-th box.

Constraints

$1 \le n \le 100$

$1 \le length_i, width_i, height_i \le 100$

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

5 5 5

1 2 40

10 5 41

7 2 42

Sample Output 0

125

80

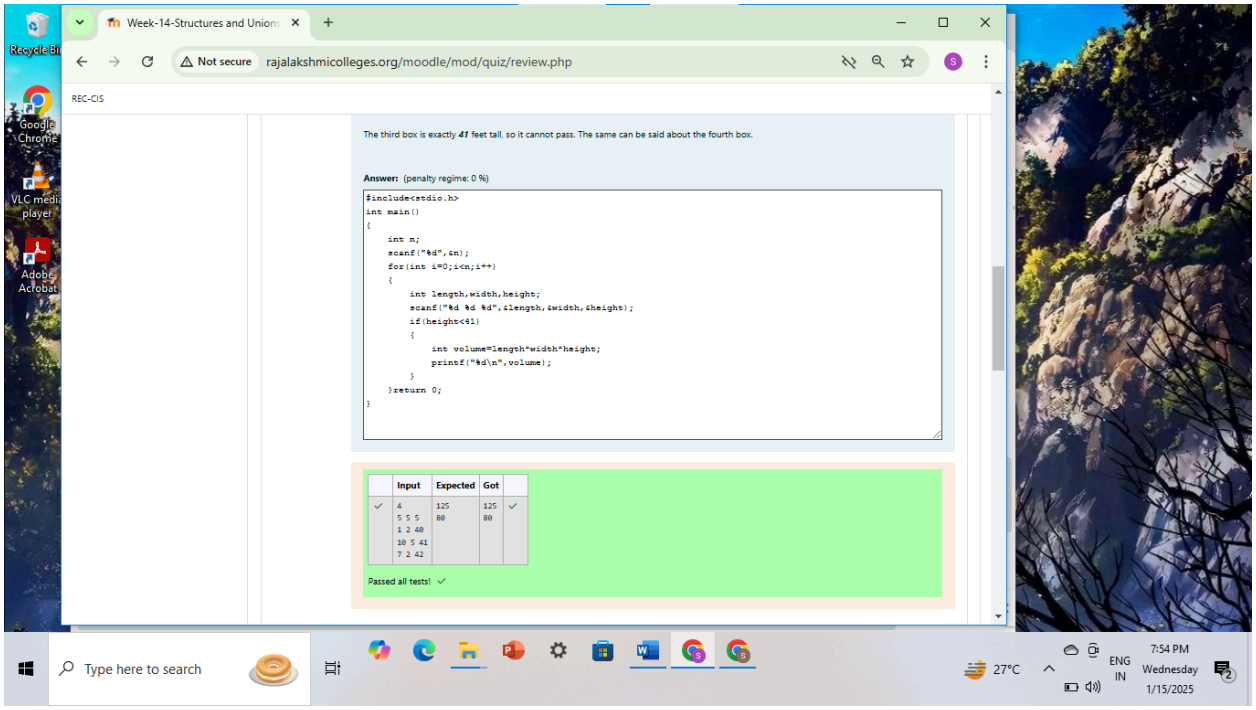
Explanation 0

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 x 2 x 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 %)



2.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 *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 a_i, b_i and c_i on each separated by single spaces.

Constraints

$1 \le n \le 100$

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

$a_i + b_i > c_i$, $a_i + c_i > b_i$ *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

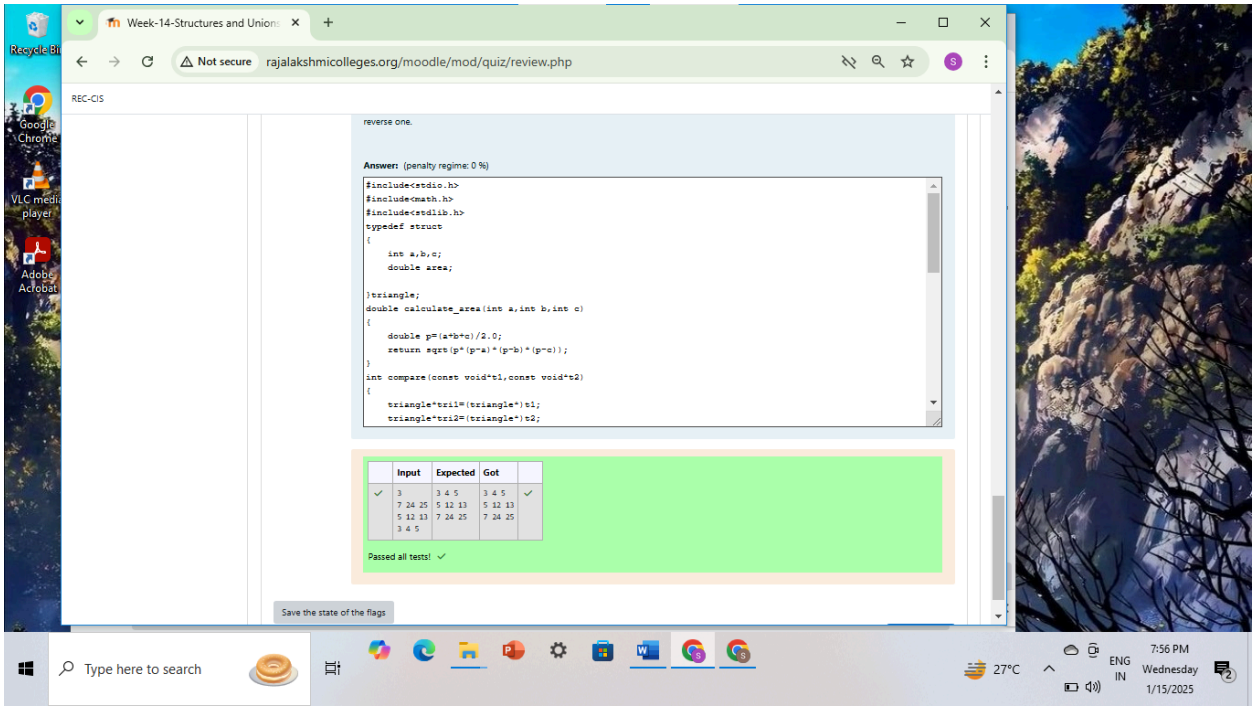
5 12 13

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 %)



2.Given an array of integers, reverse the given array in place using an index and loop rather than a built-in function.

Example

arr = [1, 3, 2, 4, 5]

Return the array *[5, 4, 2, 3, 1]* which is the reverse of the input array.

Function Description

Complete the function *reverseArray* in the editor below.

reverseArray has the following parameter(s):

int arr[n]: an array of integers

Return

int[n]: the array in reverse order

Constraints

$1 \leq n \leq 100$

$0 < arr[i] \leq 100$

Input Format For Custom Testing

The first line contains an integer, *n*, the number of elements in *arr*.

Each line *i* of the *n* subsequent lines (where $0 \leq i < n$) contains an integer, *arr[i]*.

Sample Case 0

Sample Input For Custom Testing

5

1

3

2

4

5

Sample Output

5

4

2

3

1

Explanation

The input array is [1, 3, 2, 4, 5], so the reverse of the input array is [5, 4, 2, 3, 1].

Sample Case 1

Sample Input For Custom Testing

4

17

10

21

45

Sample Output

45

21

10

17

Explanation

The input array is [17, 10, 21, 45], so the reverse of the input array is [45, 21, 10, 17].

Answer:(penalty regime: 0 %)