Week 15

**Question 1:**

**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 ≤ n ≤ 100**

**0 < arr[i] ≤ 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 ≤ i < n) contains an integer, arr[i].**

**Sample Input For Custom Testing**

**4**

**17**

**10**

**21**

**45**

**Sample Output**

**45**

**21**

**10**

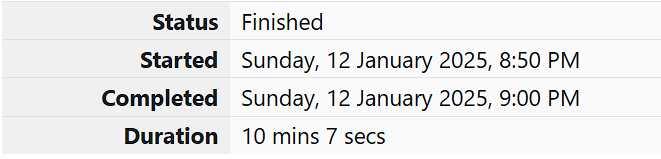
**17**

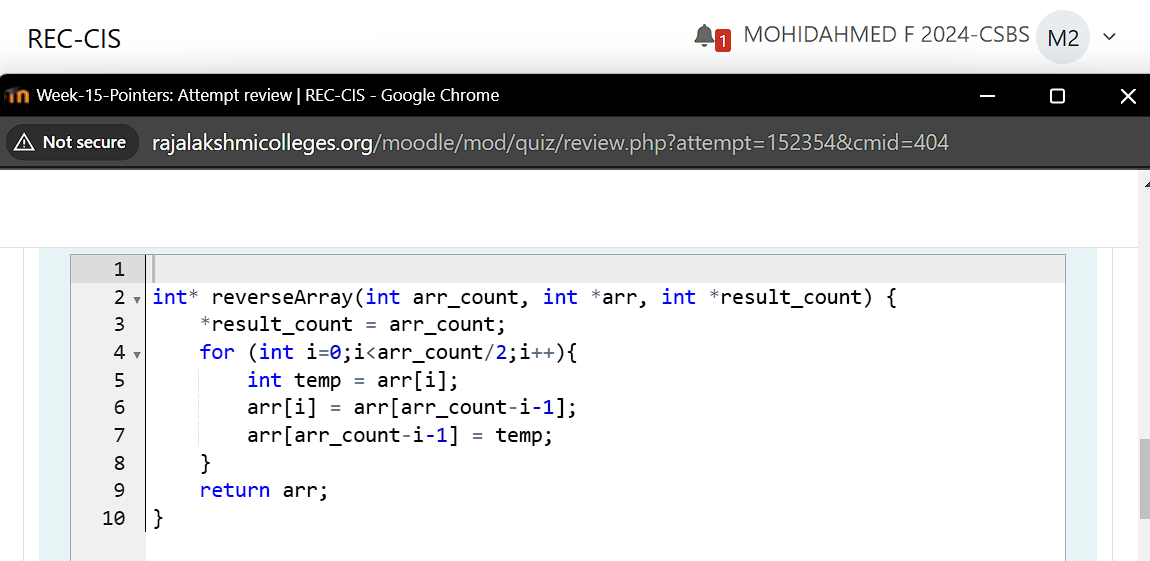
**4**

**Explanation**

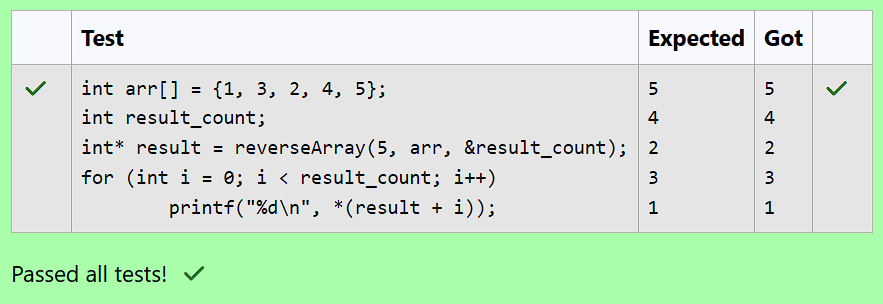
**The input array is [4, 17, 10, 21, 45], so the reverse of the input array is [45, 21, 10, 17, 4].**

**Program:**

****

****

**Output:**

****

**Question 2:**

**An automated cutting machine is used to cut rods into segments. The cutting machine can**

**only hold a rod of minLength or more, and it can only make one cut at a time. Given the**

**array lengths [] representing the desired lengths of each segment, determine if it is**

**possible to make the necessary cuts using this machine. The rod is marked into lengths**

**already, in the order given.**

**Example**

**n = 3**

**lengths = [4, 3, 2]**

**minLength = 7**

**The rod is initially sum(lengths) = 4 + 3 + 2 = 9 units long. First cut off the segment of length**

**4 + 3 = 7 leaving a rod 9 - 7 = 2. Then check that the length 7 rod can be cut into segments**

**of lengths 4 and 3. Since 7 is greater than or equal to minLength = 7, the final cut can be**

**made. Return "Possible".**

**Example**

**n = 3**

**lengths = [4, 2, 3]**

**minLength = 7**

**The rod is initially sum(lengths) = 4 + 2 + 3 = 9 units long. In this case, the initial cut can**

**be of length 4 or 4 + 2 = 6. Regardless of the length of the first cut, the remaining piece**

**will be shorter than minLength. Because n - 1 = 2 cuts cannot be made, the answer is**

**"Impossible".**

**Function Description**

**Complete the function cutThemAll in the editor below.**

**cutThemAll has the following parameter(s):**

**int lengths[n]: the lengths of the segments, in order**

**int minLength: the minimum length the machine can accept**

**Returns**

**string: "Possible" if all n-1 cuts can be made. Otherwise, return the string**

**"Impossible".**

**Constraints**

**• 2 ≤ n ≤ 105**

**• 1 ≤ t ≤ 109**

**• 1 ≤ lengths[i] ≤ 109**

**• The sum of the elements of lengths equals the uncut rod length.**

**Input Format For Custom Testing**

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

**Each line i of the n subsequent lines (where 0 ≤ i < n) contains an integer, lengths[i].**

**The next line contains an integer, minLength, the minimum length accepted by the**

**machine.**

**Sample Input For Custom Testing**

**STDIN Function**

**----- --------**

**4 → lengths[] size n = 4**

**3 → lengths[] = [3, 5, 4, 3]**

**5**

**4**

**3**

**9 → minLength= 9**

**Sample Output:**

**Possible**

**Explanation:**

**The uncut rod is 3 + 5 + 4 + 3 = 15 units long. Cut the rod into lengths of 3 + 5 + 4**

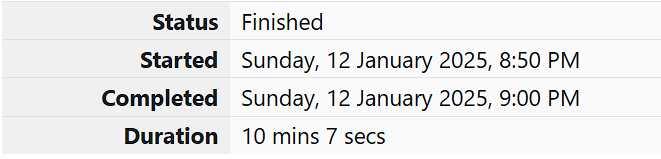
**= 12 and 3.**

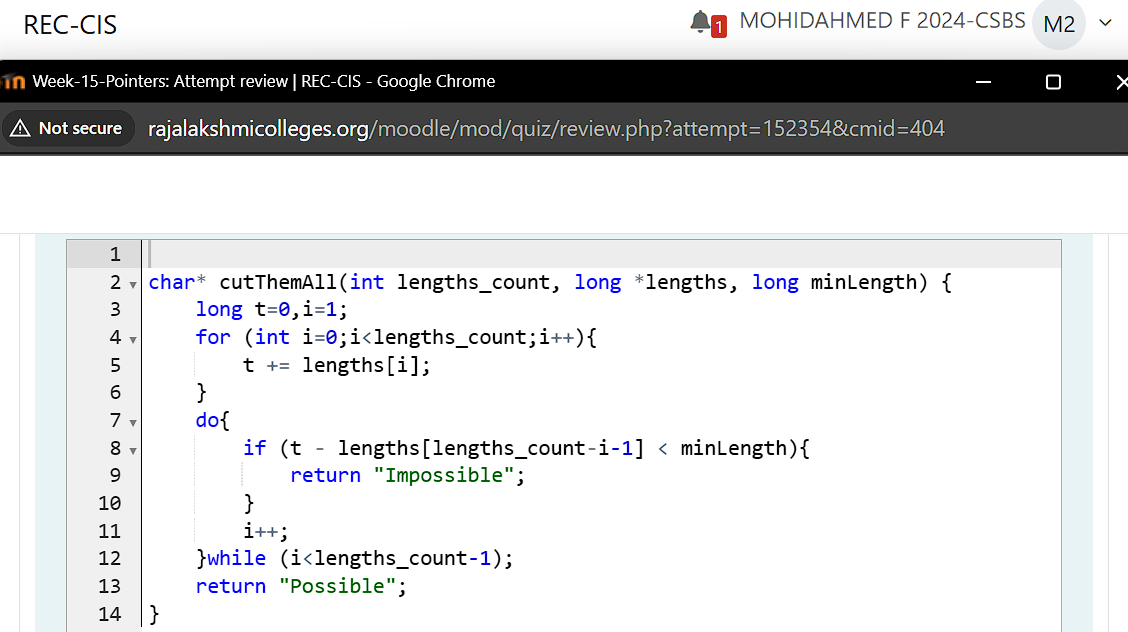
**Then cut the 12-unit piece into lengths 3 and 5 + 4 = 9.**

**The remaining segment is 5 + 4 = 9 units and that is long enough to make the final**

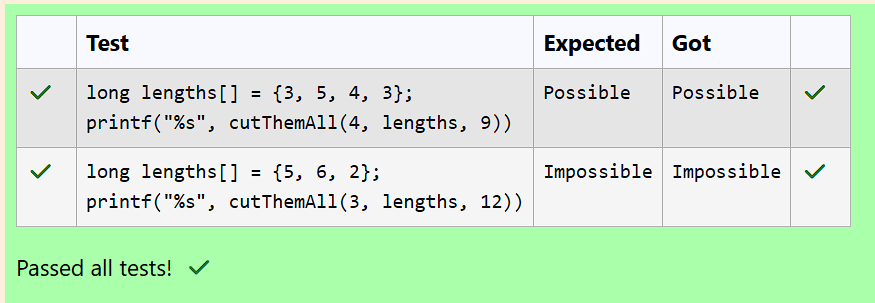
**cut.**

**Program:**

****

****

**Output:**

****