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

$1 \leq n \leq 100$

$0 < arr[i] \leq 100$

```c
#include<stdio.h>
#include<stdlib.h>
int* reverseArray(int arr_count, int *arr, int *result_count) {
    int* result=(int*)malloc(arr_count*sizeof(int));
    if(result==NULL){
        return NULL;
    }
    for(int i=0;i<arr_count;i++){
        result[i]=arr[arr_count-i-1];
    }
    *result_count=arr_count;
    return result;

}
```

	Test	Expected	Got	

✓	int arr[] = {1, 3, 2, 4, 5};
	int result_count;
	int* result = reverseArray(5, arr, &result_count);
	for (int i = 0; i < result_count; i++)
	printf("%d\n", *(result + i));
		5
		4
		2
		3
		1
			5
			4
			2
			3
			1
				✓

Passed all tests! ✓

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*".

```c
#include<stdio.h>
char* cutThemAll(int lengths_count, long *lengths, long minLength) {
    long t=0,i=1;
    for(int i=0;i<=lengths_count-1;i++){
        t+=lengths[i];
    }
    do{
        if(t-lengths[lengths_count-1]<minLength){
            return "Impossible";
        }
        i++;
    }
    while(i<lengths_count-i);
    return "Possible";

}
```

	Test	Expected	Got	
✓	long lengths[] = {3, 5, 4, 3}; printf("%s", cutThemAll(4, lengths, 9))	Possible	Possible	✓
✓	long lengths[] = {5, 6, 2}; printf("%s", cutThemAll(3, lengths, 12))	Impossible	Impossible	✓

Passed all tests! ✓