

Reset answer

```
1 1 /*
2 2 * Complete the 'reverseArray' function below.
3 3 *
4 4 * The function is expected to return an INTEGER_ARRAY.
5 5 * The function accepts INTEGER_ARRAY arr as parameter.
6 6 */
7 7
8 8 /*
9 9 * To return the integer array from the function, you should:
10 10 * - Store the size of the array to be returned in the result_count variable
11 11 * - Allocate the array statically or dynamically
12 12
13 13 * For example,
14 14 * int* return_integer_array_using_static_allocation(int* result_count) {
15 15 *     *result_count = 5;
16 16 *     static int a[5] = {1, 2, 3, 4, 5};
17 17 *     return a;
18 18 * }
19 19
20 20 * int* return_integer_array_using_dynamic_allocation(int* result_count) {
21 21 *     *result_count = 5;
22 22 *     int *a = malloc(5 * sizeof(int));
23 23 *     for (int i = 0; i < 5; i++) {
24 24 *         *(a + i) = i + 1;
25 25 *     }
26 26 *     return a;
27 27 * }
28 28
29 29 */
30 30
31 31 int* reverseArray(int n, int *a, int *rC) {
32 32     *rC=n;
33 33     int *b=(int*)malloc(sizeof(int)*n);
34 34     for(int i=0;i<n;i++)
35 35     {
36 36         b[i]=a[n-1-i];
37 37     }
38 38     return b;
39 39 }
40 40
41 41
42 42
43 43
44 44
```

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! ✓

Reset answer

```
1 1 /*
2 2 * Complete the 'cutThenAll' function below.
3 3 *
4 4 * The function is expected to return a STRING.
5 5 * The function accepts following parameters:
6 6 * 1. LONG_INTEGER_ARRAY lengths
7 7 * 2. LONG_INTEGER minLength
8 8 */
9 9
10 10 /*
11 11 * To return the string from the function, you should either do static allocation or dynamic allocation
12 12 *
13 13 * For example,
14 14 * char* return_string_using_static_allocation() {
15 15 *     static char s[] = "static allocation of string";
16 16 *     return s;
17 17 * }
18 18
19 19 * char* return_string_using_dynamic_allocation() {
20 20 *     char* s = malloc(100 * sizeof(char));
21 21 *     s = "dynamic allocation of string";
22 22 *     return s;
23 23 * }
24 24
25 25 */
26 26
27 27 #include<stdio.h>
28 28 int cmp(const void*a,const void*b){
29 29     return(*(int*)a)-(*(int*)b);
30 30 }
31 31
32 32 char* cutThenAll(int n , long *a, long m){
33 33     int s=0;
34 34     for(int i=0;i<n;i++)
35 35     {
36 36         s+=a[i];
37 37     }
38 38     long r=s;
39 39     qsort(a,n,sizeof(long),cmp);
40 40     for(int i=0;i<n;i++)
41 41     {
42 42         if(r==m){
43 43             return "Possible";
44 44         }
45 45         if(r<m){
46 46             r+=a[i];
47 47         }
48 48         else{
49 49             return "Impossible";
50 50         }
51 51     }
52 52 }
```

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

Passed all tests! ✓