

GE23131-Programming Using C-2024

Attempts allowed: 4

This quiz has been configured so that students may only attempt it using the Safe Exam Browser.

Time limit: 1 hour 30 mins

Grading method: Highest grade

Your attempts

Attempt 1	
Status	Finished
Started	Thursday, 16 January 2025, 6:04 PM
Completed	Thursday, 16 January 2025, 6:14 PM
Duration	10 mins 36 secs
<div>Review</div>	

The Safe Exam Browser keys could not be validated. Check that you're using Safe Exam Browser with the correct configuration file.

GE23131-Programming Using C-2024

Status	Finished
Started	Thursday, 16 January 2025, 6:04 PM
Completed	Thursday, 16 January 2025, 6:14 PM
Duration	10 mins 36 secs

Question 1

Correct

Marked out of 1.00

🚩 [Flag question](#)

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):

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

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

Answer: (penalty regime: 0 %)

Reset answer

```
1  /*
2  * Complete the 'reverseArray
3  *
4  * The function is expected t
5  * The function accepts INTEG
6  */
7
8  /*
9  * To return the integer arra
10 *     - Store the size of th
11 *     - Allocate the array s
12 *
13 * For example,
14 * int* return_integer_array_
15 *     *result_count = 5;
16 *
17 *     static int a[5] = {1,
18 *
19 *     return a;
20 * }
21 *
22 * int* return_integer_array_
23 *     *result_count = 5;
24 *
25 *     int *a = malloc(5 * si
26 *
27 *     for (int i = 0; i < 5;
28 *         *(a + i) = i + 1;
29 *     }
30 *
31 *     return a;
32 * }
33 *
34 */
35 #include <stdio.h>
36 #include <stdlib.h>
37 int* reverseArray(int arr, cou
```

```

30     *
31     *     return a;
32     * }
33     *
34     */
35     #include <stdio.h>
36     #include <stdlib.h>
37     int* reverseArray(int arr_count,
38                       int* result=(int*)malloc(
39                       if(result==NULL){
40                           return NULL;
41                       }
42                       for(int i=0;i<arr_count;i
43                       {
44                           result[i]=arr[arr_count-i-1];
45                       }
46                       *result_count=arr_count;
47                       return result;
48
49 }
50

```

	Test
✓	<pre> 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)); </pre>

Passed all tests! ✓

Question 2

Incorrect

Marked out of 1.00

```
30
31
32
33
34
35
36
37 ▼ *arr, int *result_count) {
38     t);
39 ▼
40
41
42
43 ▼
44
45
46
47
48
49
50
```

	Test
✓	<pre>int arr[] = {1, 3, 2, 4, 5}; int result_count; int* result = reverseArray(5, a for (int i = 0; i < result_coun printf("%d\n", *(result</pre>

Passed all tests! ✓

Question 2

Incorrect

Marked out of 1.00

Question 2

Incorrect

Marked out of 1.00

🚩 [Flag question](#)

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"*.

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 $0 \leq i < n$) contains an integer, *lengths*[i].

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

Sample Case 0

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

lengths 3 and $5 + 4 = 9$. The remaining segment is $5 + 4 = 9$ units and that is long enough to make the final cut.

Sample Case 1

Sample Input For Custom Testing

STDIN Function

3 \rightarrow lengths[] size $n = 3$

5 \rightarrow lengths[] = [5, 6, 2]

6

2

12 \rightarrow minLength = 12

Sample Output

Impossible

Explanation

The uncut rod is $5 + 6 + 2 = 13$ units long. After making either cut, the rod will be too short to make the second cut.

Answer: (penalty regime: 0 %)

Reset answer

Reset answer

```
1  /*
2   * Complete the 'cutThemAll'
3   *
4   * The function is expected t
5   * The function accepts follo
6   * 1. LONG_INTEGER_ARRAY len
7   * 2. LONG_INTEGER minLength
8   */
9
10 /*
11  * To return the string from
12  *
13  * For example,
14  * char* return_string_using_
15  *     static char s[] = "sta
16  *
17  *     return s;
18  * }
19  *
20  * char* return_string_using_
21  *     char* s = malloc(100 *
22  *
23  *     s = "dynamic allocatio
24  *
25  *     return s;
26  * }
27  *
28  */
29  #include <stdio.h>
30  char* cutThemAll(int lengths_
31      long t=0,i=1;
32      for(int i=0;i<=lengths_co
33  {
34      t+=lengths[i];
35  }
36  do{
37      if(t-lengths[lengths_
38      return "Impossible";
39  }
40  i++;
41  }while(i<lengths_count-i)
42  return "Possible";
```

```

20
27
28
29 include <stdio.h>
30 * cutThemAll(int lengths_count
31 long t=0,i=1;
32 for(int i=0;i<=lengths_count;i
33 {
34     t+=lengths[i];
35 }
36 do{
37     if(t-lengths[lengths_count
38     return "Impossible";
39     }
40 i++;
41 }while(i<lengths_count-i);
42 return "Possible";
43
44

```

	Test
✗	long lengths[] = {3, 5, 4, 3}; printf("%s", cutThemAll(4, leng
✓	long lengths[] = {5, 6, 2}; printf("%s", cutThemAll(3, leng

Your code must pass all tests to earn any marks. Try again.

Show differences

Finish review