

```

11      *      Allocate the array statically or
12      *
13      * For example,
14      * int* return_integer_array_using_static
15      *      *result_count = 5;
16      *
17      *      static int a[5] = {1, 2, 3, 4, 5};
18      *
19      *      return a;
20      * }
21      *
22      * int* return_integer_array_using_dynami
23      *      *result_count = 5;
24      *
25      *      int *a = malloc(5 * sizeof(int));
26      *
27      *      for (int i = 0; i < 5; i++) {
28      *          *(a + i) = i + 1;
29      *      }
30      *
31      *      return a;
32      * }
33      *
34      */
35      int* reverseArray(int arr_count, int *arr
36          *result_count=arr_count;
37          int *reversed=(int*)malloc(arr_count*
38          if(reversed==NULL)exit(1);
39          for(int i=0;i<arr_count;i++)
40              reversed[i]=arr[arr_count-1-i];
41
42          return reversed;
43      }

```

	Expected	Got	
4, 5};	5	5	✓
reverseArray(5, arr, &result_count);	4	4	
for(int i=0;i<result_count;i++)	2	2	
*(result + i));	3	3	
	1	1	

Passed all tests! ✓


```

10  /*
11  * To return the string from the function
12  *
13  * For example,
14  * char* return_string_using_static_alloc
15  *     static char s[] = "static allocati
16  *
17  *     return s;
18  * }
19  *
20  * char* return_string_using_dynamic_allo
21  *     char* s = malloc(100 * sizeof(char
22  *
23  *     s = "dynamic allocation of string"
24  *
25  *     return s;
26  * }
27  *
28  */
29  char* cutThemAll(int lengths_count, long
30      long totallength=0;
31      for(int i=0;i<lengths_count;i++)
32          totallength+=lengths[i];
33      long currentlength=0;
34      for(int i=0;i<lengths_count-1;i++){
35          currentlength+=lengths[i];
36          long rem_length=totallength-curre
37          if(rem_length>minLength)return "P
38
39      }return "Impossible";
40  }
41

```

	Expected	Got	
5, 4, 3}; All(4, lengths, 9))	Possible	Possible	✓
6, 2}; All(3, lengths, 12))	Impossible	Impossible	✓

Passed all tests! ✓