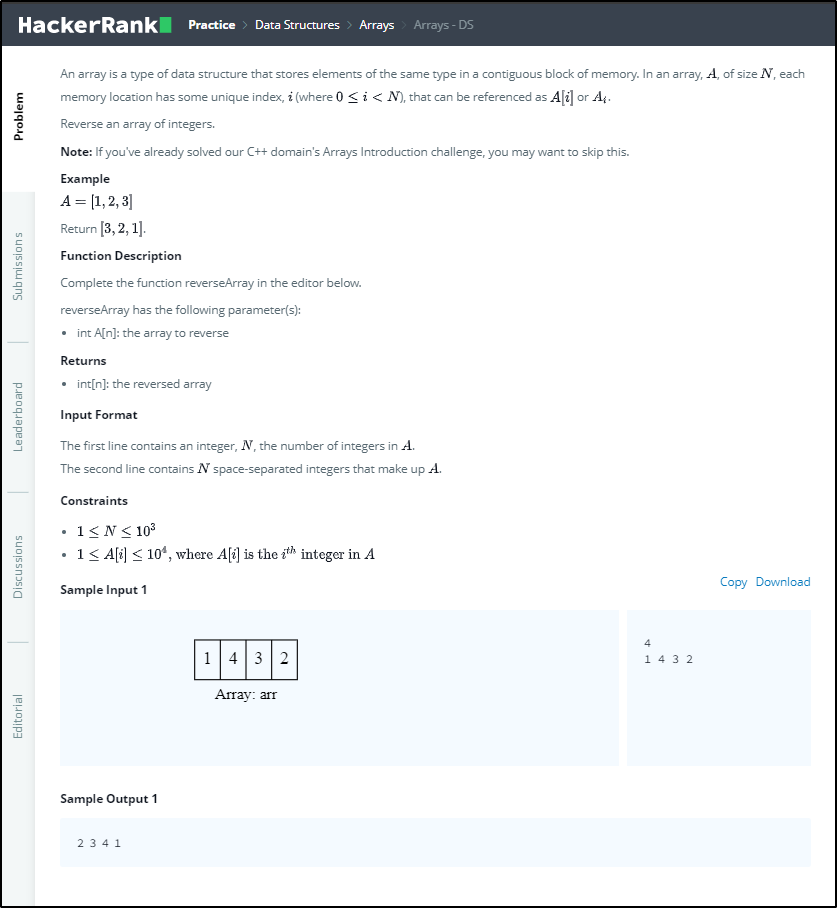
**PROBLEM STATEMENT**



**PROGRAM USED TO SOLVE THE PROBLEM STATEMENT**

#include <assert.h>

#include <limits.h>

#include <math.h>

#include <stdbool.h>

#include <stddef.h>

#include <stdint.h>

#include <stdio.h>

#include <stdlib.h>

#include <string.h>

char\* readline();

char\*\* split\_string(char\*);

// Complete the reverseArray function below.

// Please store the size of the integer array to be returned in result\_count pointer. For example,

// int a[3] = {1, 2, 3};

//

// \*result\_count = 3;

//

// return a;

//

int\* reverseArray(int a\_count, int\* a, int\* result\_count) {

    \*result\_count = a\_count;

    int temp;

    int end = a\_count-1;

    for(int i = 0; i < a\_count/2; i++){

        temp = \*(a+end);

        \*(a+end) = \*(a+i);

        \*(a+i) = temp;

        end--;

    }

    return a;

}

int main()

{

    FILE\* fptr = fopen(getenv("OUTPUT\_PATH"), "w");

    char\* arr\_count\_endptr;

    char\* arr\_count\_str = readline();

    int arr\_count = strtol(arr\_count\_str, &arr\_count\_endptr, 10);

    if (arr\_count\_endptr == arr\_count\_str || \*arr\_count\_endptr != '\0') { exit(EXIT\_FAILURE); }

    char\*\* arr\_temp = split\_string(readline());

    int\* arr = malloc(arr\_count \* sizeof(int));

    for (int i = 0; i < arr\_count; i++) {

        char\* arr\_item\_endptr;

        char\* arr\_item\_str = \*(arr\_temp + i);

        int arr\_item = strtol(arr\_item\_str, &arr\_item\_endptr, 10);

        if (arr\_item\_endptr == arr\_item\_str || \*arr\_item\_endptr != '\0') { exit(EXIT\_FAILURE); }

        \*(arr + i) = arr\_item;

    }

    int res\_count;

    int\* res = reverseArray(arr\_count, arr, &res\_count);

    for (int i = 0; i < res\_count; i++) {

        fprintf(fptr, "%d", \*(res + i));

        if (i != res\_count - 1) {

            fprintf(fptr, " ");

        }

    }

    fprintf(fptr, "\n");

    fclose(fptr);

    return 0;

}

char\* readline() {

    size\_t alloc\_length = 1024;

    size\_t data\_length = 0;

    char\* data = malloc(alloc\_length);

    while (true) {

        char\* cursor = data + data\_length;

        char\* line = fgets(cursor, alloc\_length - data\_length, stdin);

        if (!line) {

            break;

        }

        data\_length += strlen(cursor);

        if (data\_length < alloc\_length - 1 || data[data\_length - 1] == '\n') {

            break;

        }

        alloc\_length <<= 1;

        data = realloc(data, alloc\_length);

        if (!line) {

            break;

        }

    }

    if (data[data\_length - 1] == '\n') {

        data[data\_length - 1] = '\0';

        data = realloc(data, data\_length);

    } else {

        data = realloc(data, data\_length + 1);

        data[data\_length] = '\0';

    }

    return data;

}

char\*\* split\_string(char\* str) {

    char\*\* splits = NULL;

    char\* token = strtok(str, " ");

    int spaces = 0;

    while (token) {

        splits = realloc(splits, sizeof(char\*) \* ++spaces);

        if (!splits) {

            return splits;

        }

        splits[spaces - 1] = token;

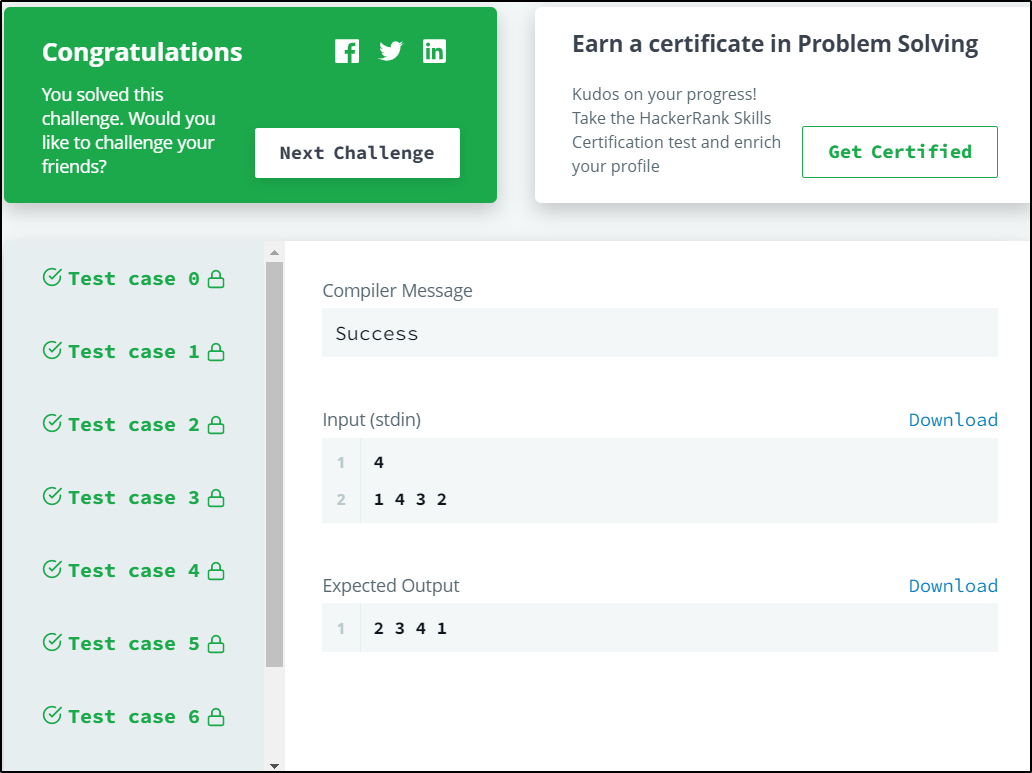
        token = strtok(NULL, " ");

    }

    return splits;

}

**TEST CASES**

****