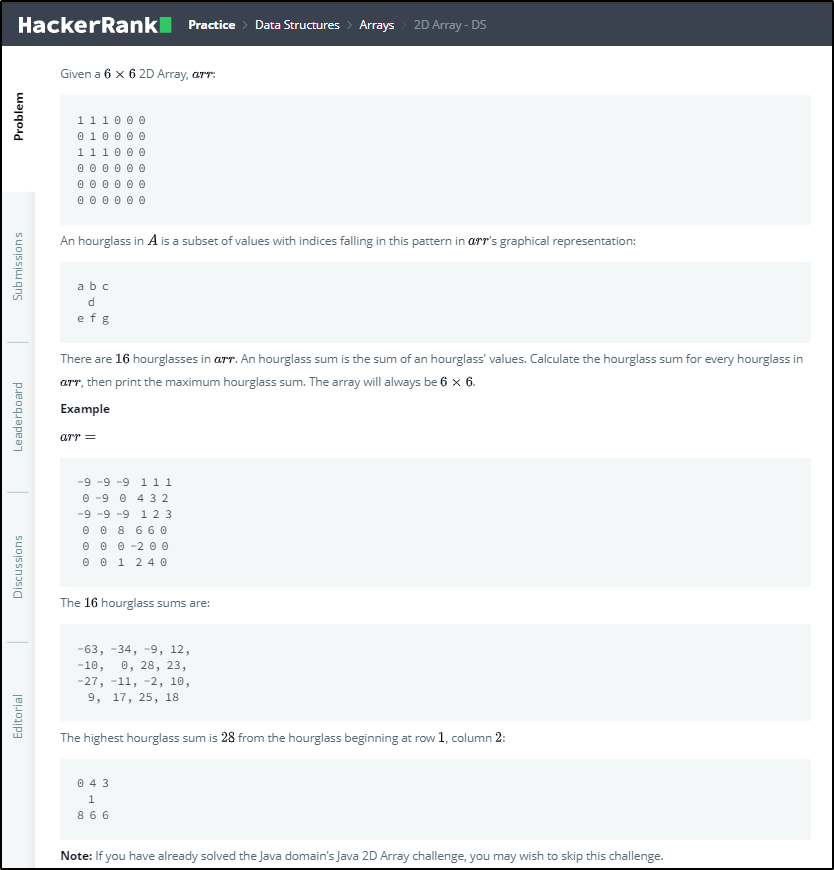
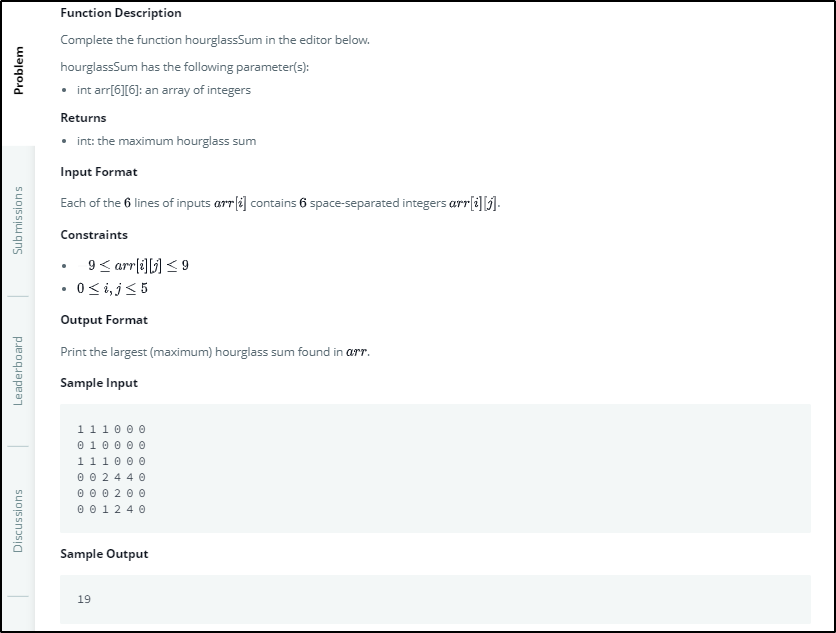
**PROBLEM STATEMENT**







**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 hourglassSum function below.

int hourglassSum(int arr\_rows, int arr\_columns, int\*\* arr) {

   int sum = 0,max = -63;

   for(int i=0; i<4; i++){

   for(int j=0; j<4; j++){

       sum = 0;

       sum += arr[i][j] + arr[i][j+1] + arr[i][j+2];      // x y z

       sum += arr[i+1][j+1];                              //   o

       sum += arr[i+2][j] + arr[i+2][j+1] + arr[i+2][j+2];// a b c

           if(sum>max)

           max = sum;

          }

      }

 return max;

}

int main()

{

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

    int\*\* arr = malloc(6 \* sizeof(int\*));

    for (int i = 0; i < 6; i++) {

        \*(arr + i) = malloc(6 \* (sizeof(int)));

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

        for (int j = 0; j < 6; j++) {

            char\* arr\_item\_endptr;

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

            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) + j) = arr\_item;

        }

    }

    int arr\_rows = 6;

    int arr\_columns = 6;

    int result = hourglassSum(arr\_rows, arr\_columns, arr);

    fprintf(fptr, "%d\n", result);

    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; }

        size\_t new\_length = alloc\_length << 1;

        data = realloc(data, new\_length);

        if (!data) { break; }

        alloc\_length = new\_length;

    }

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

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

    }

    data = realloc(data, data\_length);

    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**

