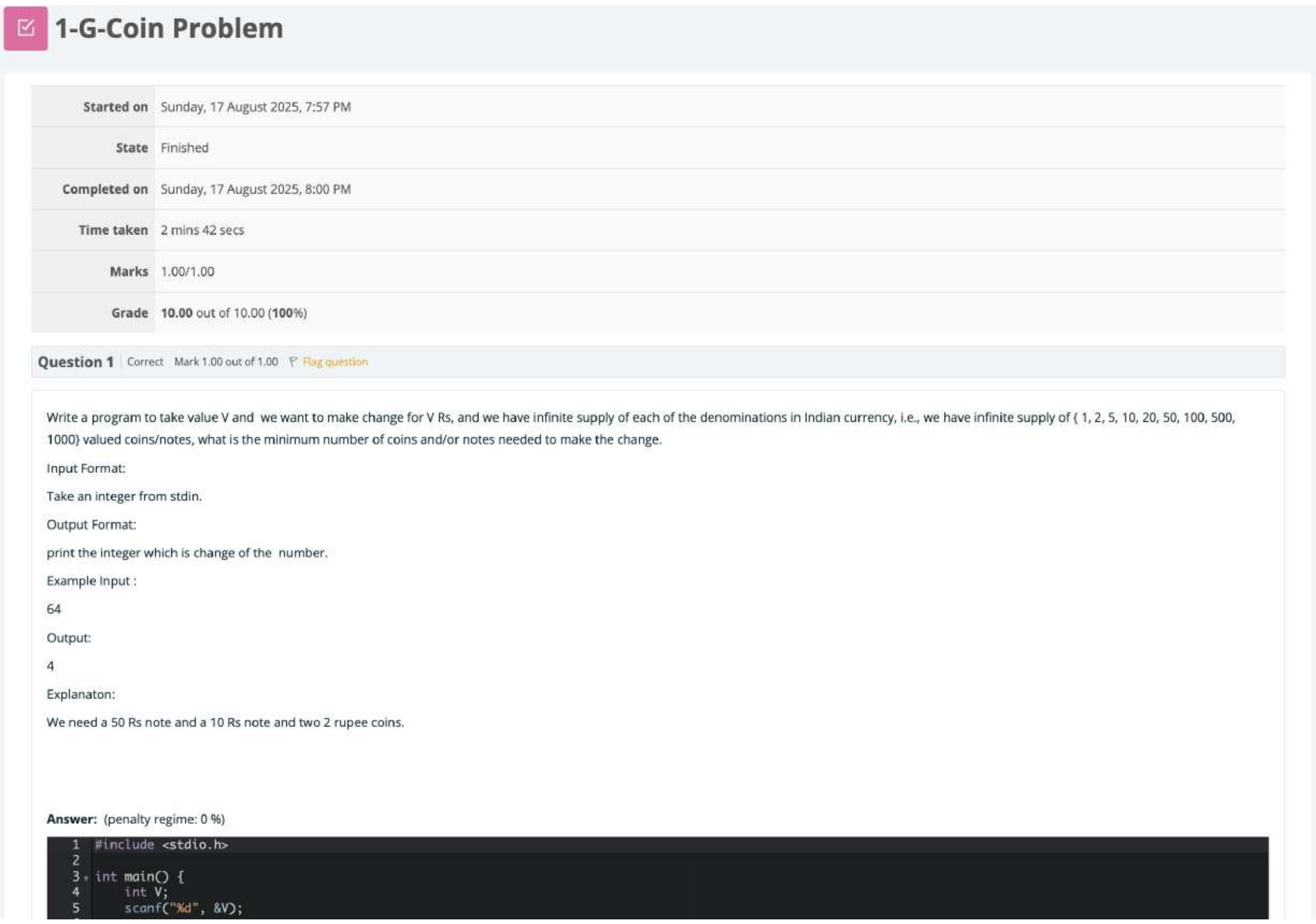


1-G-Coin Problem

Started on	Sunday, 17 August 2025, 7:57 PM
State	Finished
Completed on	Sunday, 17 August 2025, 8:00 PM
Time taken	2 mins 42 secs
Marks	1.00/1.00
Grade	**10.00** out of 10.00 (**100**%)

Question 1 Correct Mark 1.00 out of 1.00 Flag question

Write a program to take value V and we want to make change for V Rs, and we have infinite supply of each of the denominations in Indian currency, i.e., we have infinite supply of { 1, 2, 5, 10, 20, 50, 100, 500, 1000} valued coins/notes, what is the minimum number of coins and/or notes needed to make the change.

Input Format:

Take an integer from stdin.

Output Format:

print the integer which is change of the number.

Example Input :

64

Output:

4

Explanaton:

We need a 50 Rs note and a 10 Rs note and two 2 rupee coins.

Answer: (penalty regime: 0 %)

```
#include <stdio.h>

int main() {
    int V;
    scanf("%d", &V);
```

Answer: (penalty regime: 0 %)

```c
#include <stdio.h>

int main() {
    int V;
    scanf("%d", &V);

    int denominations[] = {1000, 500, 100, 50, 20, 10, 5, 2, 1};
    int num_denominations = sizeof(denominations) / sizeof(denominations[0]);
    int count = 0;

    for (int i = 0; i < num_denominations; i++) {
        while (V >= denominations[i]) {
            V -= denominations[i];
            count++;
        }
    }

    printf("%d\n", count);

    return 0;
}
```

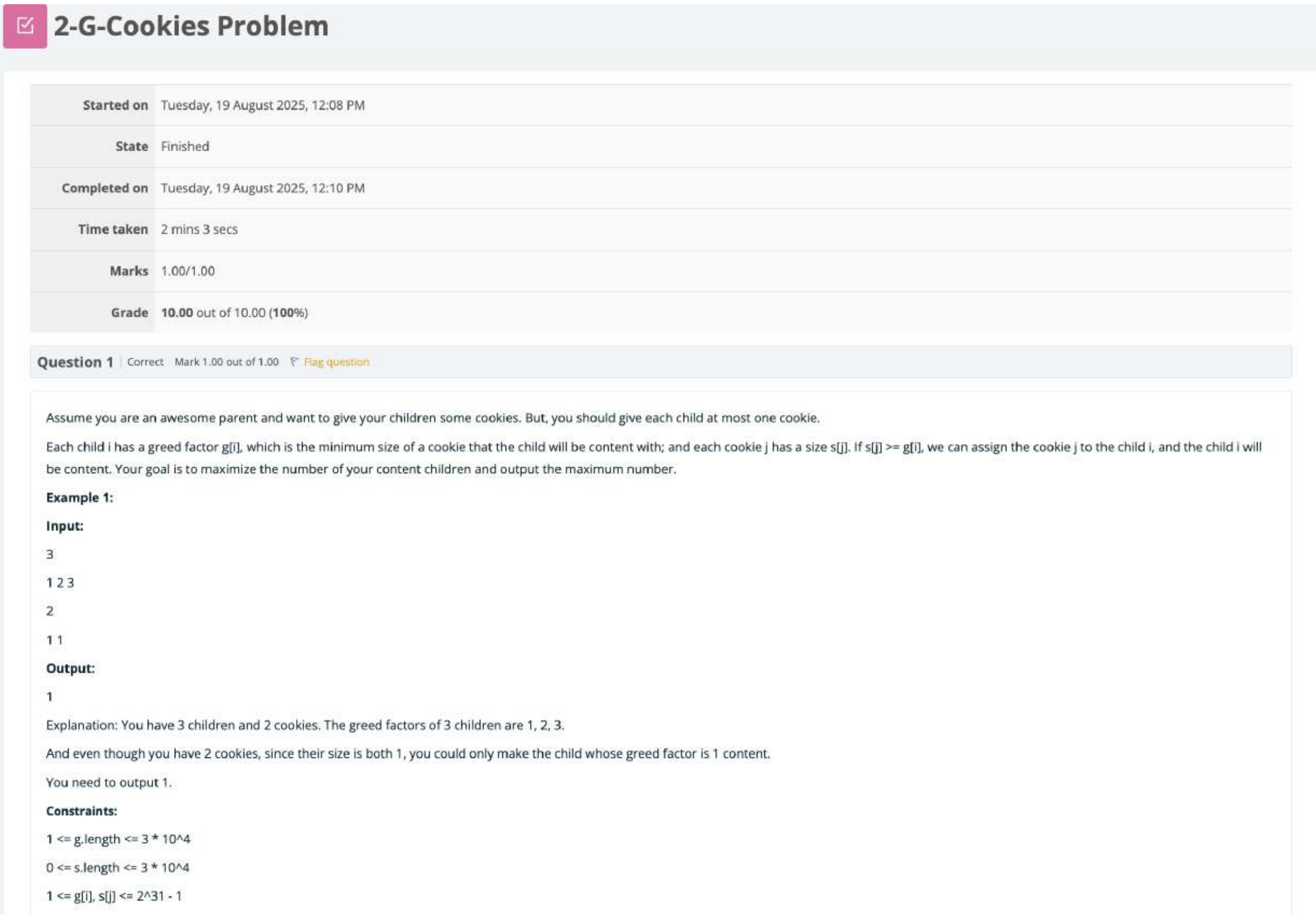
	Input	Expected	Got	
✓	49	5	5	✓

Passed all tests! ✓

Correct

Marks for this submission: 1.00/1.00.

Finish review



2-G-Cookies Problem

Started on	Tuesday, 19 August 2025, 12:08 PM
State	Finished
Completed on	Tuesday, 19 August 2025, 12:10 PM
Time taken	2 mins 3 secs
Marks	1.00/1.00
Grade	**10.00** out of 10.00 (**100%**)

Question 1 Correct Mark 1.00 out of 1.00 Flag question

Assume you are an awesome parent and want to give your children some cookies. But, you should give each child at most one cookie.

Each child i has a greed factor g[i], which is the minimum size of a cookie that the child will be content with; and each cookie j has a size s[j]. If s[j] >= g[i], we can assign the cookie j to the child i, and the child i will be content. Your goal is to maximize the number of your content children and output the maximum number.

Example 1:

Input:

3

1 2 3

2

1 1

Output:

1

Explanation: You have 3 children and 2 cookies. The greed factors of 3 children are 1, 2, 3.

And even though you have 2 cookies, since their size is both 1, you could only make the child whose greed factor is 1 content.

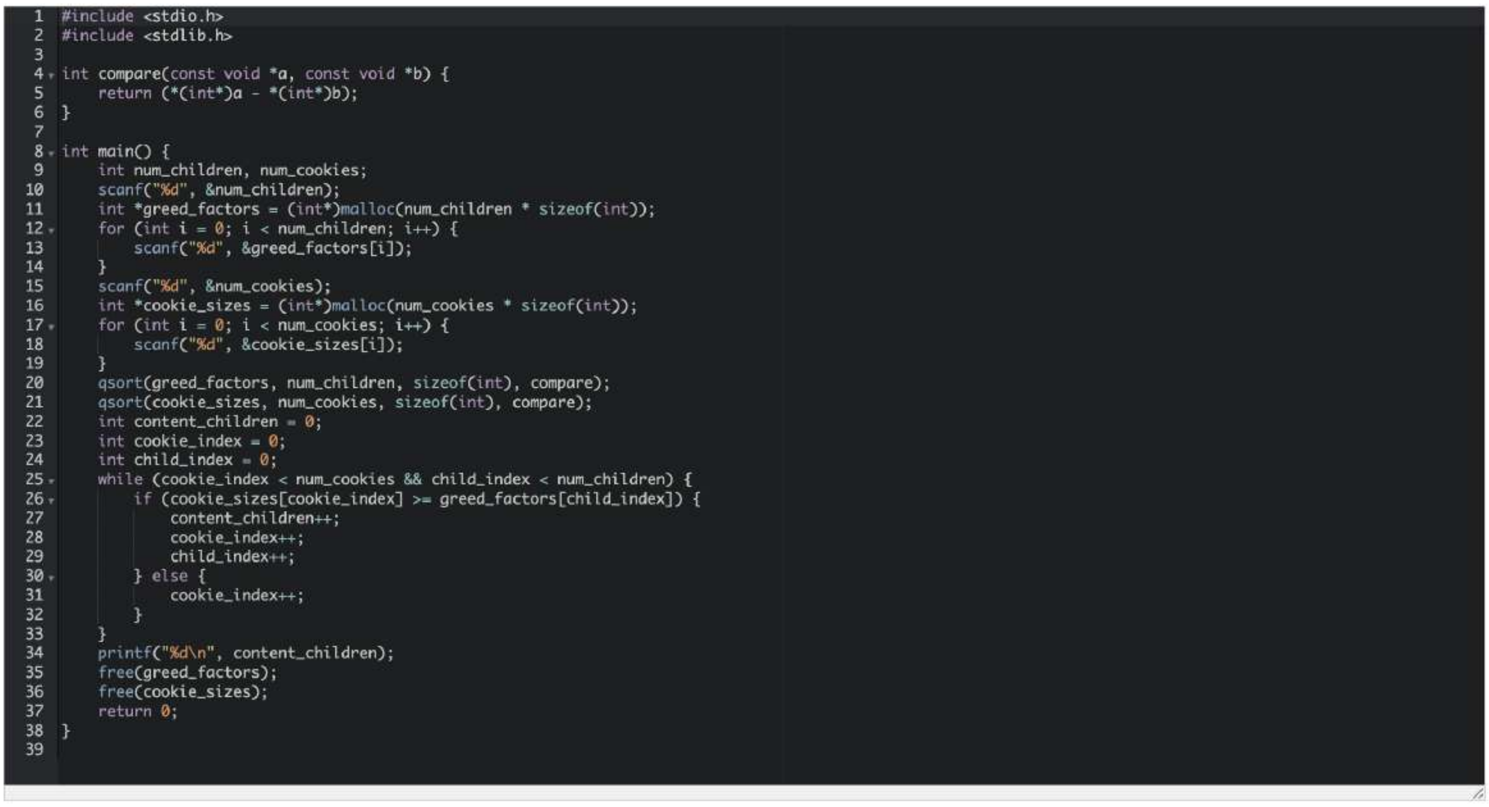
You need to output 1.

Constraints:

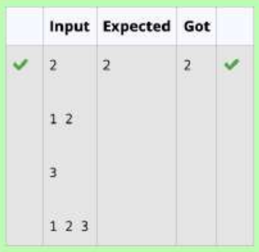
1 <= g.length <= 3 * 10^4

0 <= s.length <= 3 * 10^4

1 <= g[i], s[j] <= 2^31 - 1



```c
#include <stdio.h>
#include <stdlib.h>

int compare(const void *a, const void *b) {
    return (*(int*)a - *(int*)b);
}

int main() {
    int num_children, num_cookies;
    scanf("%d", &num_children);
    int *greed_factors = (int*)malloc(num_children * sizeof(int));
    for (int i = 0; i < num_children; i++) {
        scanf("%d", &greed_factors[i]);
    }
    scanf("%d", &num_cookies);
    int *cookie_sizes = (int*)malloc(num_cookies * sizeof(int));
    for (int i = 0; i < num_cookies; i++) {
        scanf("%d", &cookie_sizes[i]);
    }
    qsort(greed_factors, num_children, sizeof(int), compare);
    qsort(cookie_sizes, num_cookies, sizeof(int), compare);
    int content_children = 0;
    int cookie_index = 0;
    int child_index = 0;
    while (cookie_index < num_cookies && child_index < num_children) {
        if (cookie_sizes[cookie_index] >= greed_factors[child_index]) {
            content_children++;
            cookie_index++;
            child_index++;
        } else {
            cookie_index++;
        }
    }
    printf("%d\n", content_children);
    free(greed_factors);
    free(cookie_sizes);
    return 0;
}

```



	Input	Expected	Got	
✓	2 1 2 3 1 2 3	2	2	✓

3-G-Burger Problem

Started on	Sunday, 24 August 2025, 6:02 PM
State	Finished
Completed on	Tuesday, 26 August 2025, 12:58 PM
Time taken	1 day 18 hours
Marks	1.00/1.00
Grade	**10.00** out of 10.00 (**100**%)

Question 1 Correct Mark 1.00 out of 1.00 Flag question

A person needs to eat burgers. Each burger contains a count of calorie. After eating the burger, the person needs to run a distance to burn out his calories. If he has eaten i burgers with c calories each, then he has to run at least $3^i * c$ kilometers to burn out the calories. For example, if he ate 3 burgers with the count of calorie in the order: [1, 3, 2], the kilometers he needs to run are $(3^0 * 1) + (3^1 * 3) + (3^2 * 2) = 1 + 9 + 18 = 28$. But this is not the minimum, so need to try out other orders of consumption and choose the minimum value. Determine the minimum distance he needs to run. Note: He can eat burger in any order and use an efficient sorting algorithm.Apply greedy approach to solve the problem.

Input Format

First Line contains the number of burgers

Second line contains calories of each burger which is n space-separate integers

Output Format

Print: Minimum number of kilometers needed to run to burn out the calories

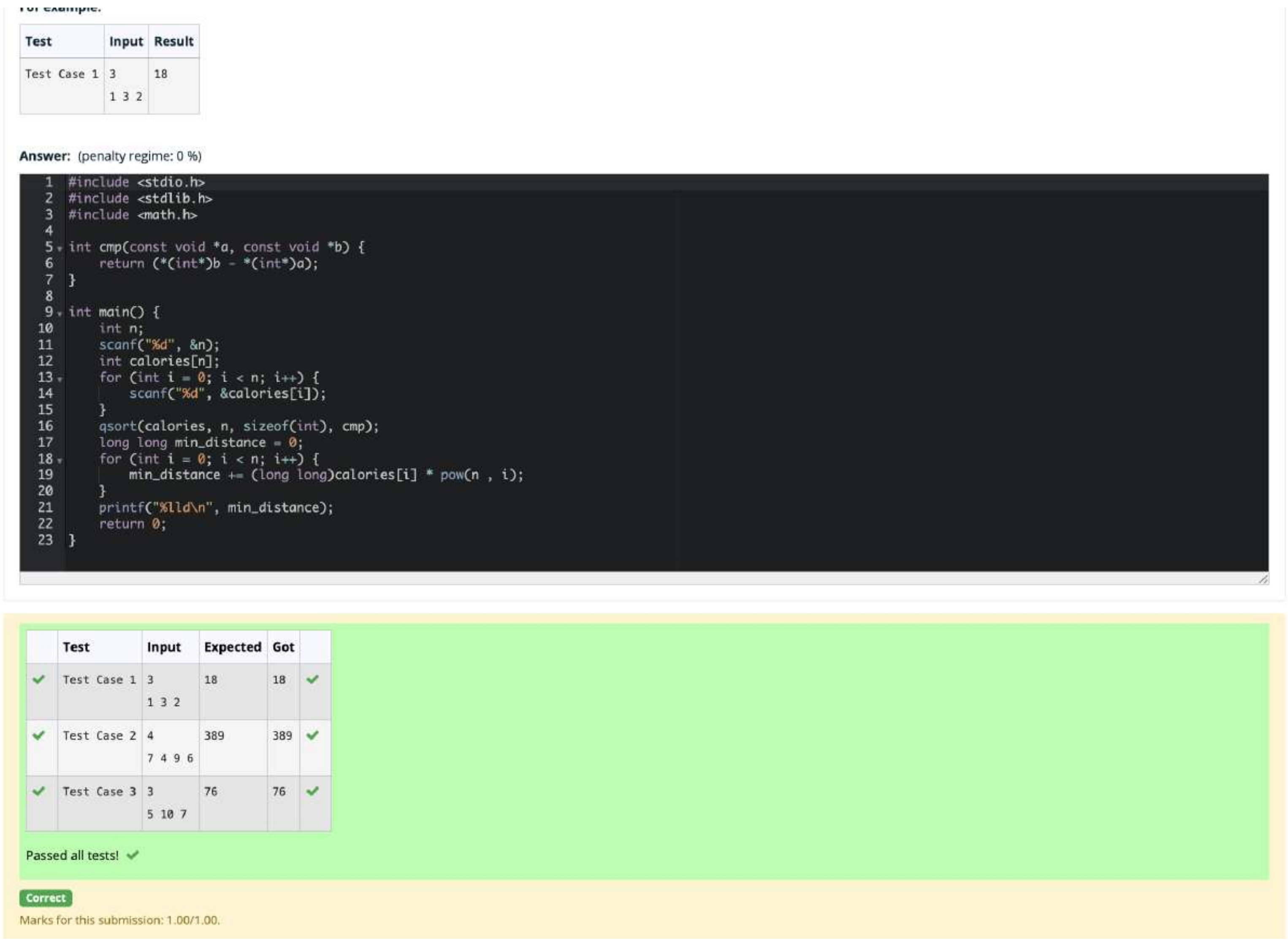
Sample Input

```
3
5 10 7
```

Sample Output

```
76
```

For example:



For example:

Test	Input	Result
Test Case 1	3 1 3 2	18

Answer: (penalty regime: 0 %)

```c
#include <stdio.h>
#include <stdlib.h>
#include <math.h>

int cmp(const void *a, const void *b) {
    return (*(int*)b - *(int*)a);
}

int main() {
    int n;
    scanf("%d", &n);
    int calories[n];
    for (int i = 0; i < n; i++) {
        scanf("%d", &calories[i]);
    }
    qsort(calories, n, sizeof(int), cmp);
    long long min_distance = 0;
    for (int i = 0; i < n; i++) {
        min_distance += (long long)calories[i] * pow(n , i);
    }
    printf("%lld\n", min_distance);
    return 0;
}
```

	Test	Input	Expected	Got	
✓	Test Case 1	3 1 3 2	18	18	✓
✓	Test Case 2	4 7 4 9 6	389	389	✓
✓	Test Case 3	3 5 10 7	76	76	✓

Passed all tests! ✓

Correct

Marks for this submission: 1.00/1.00.

4-G-Array Sum max problem

Started on	Tuesday, 19 August 2025, 12:39 PM
State	Finished
Completed on	Tuesday, 19 August 2025, 1:01 PM
Time taken	21 mins 59 secs
Marks	1.00/1.00
Grade	**10.00** out of 10.00 (**100**%)

Question 1 Correct Mark 1.00 out of 1.00 Flag question

Given an array of N integer, we have to maximize the sum of arr[i] * i, where i is the index of the element (i = 0, 1, 2, ..., N).Write an algorithm based on Greedy technique with a Complexity O(nlogn).

Input Format:

First line specifies the number of elements-n

The next n lines contain the array elements.

Output Format:

Maximum Array Sum to be printed.

Sample Input:

5

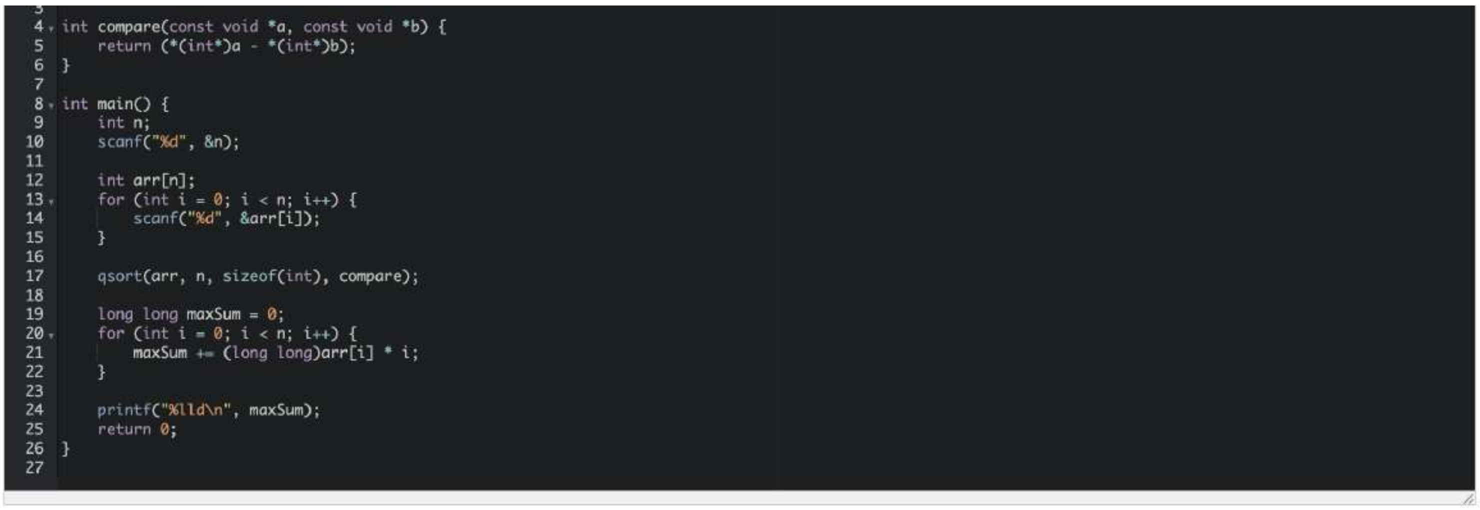
2 5 3 4 0

Sample output:

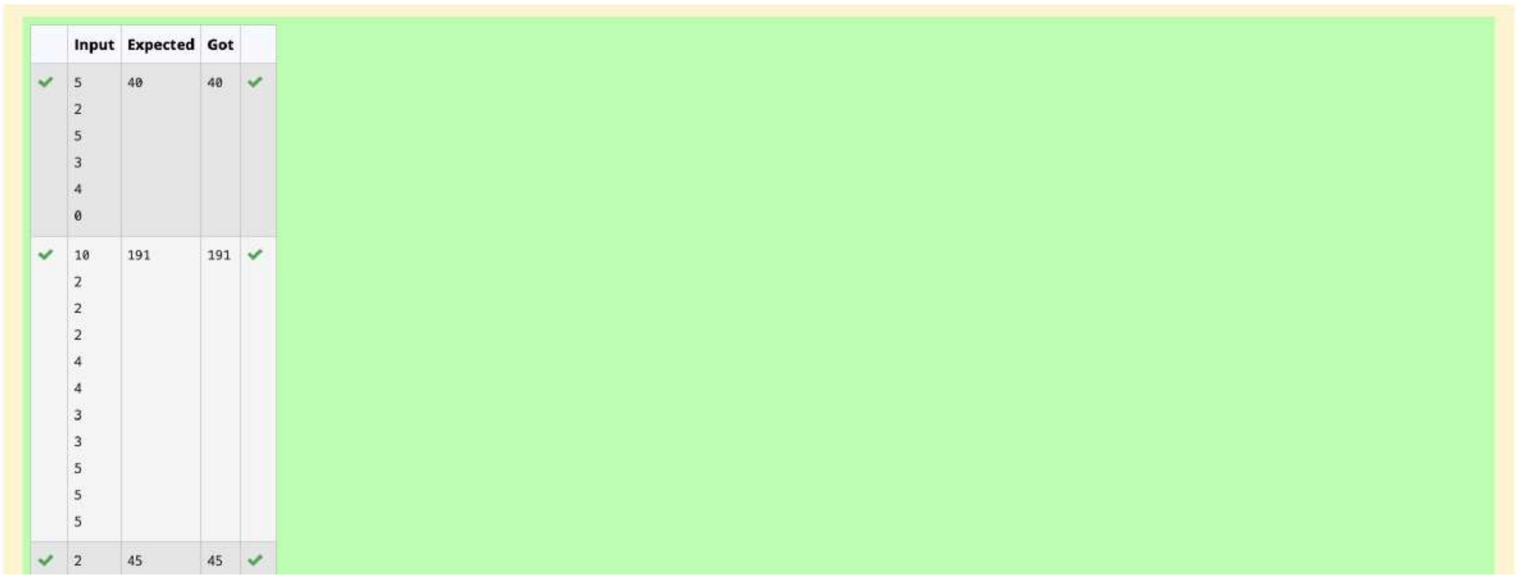
40

Answer: (penalty regime: 0 %)

```
#include <stdio.h>
#include <stdlib.h>

int compare(const void *a, const void *b) {
    return (*(int*)a - *(int*)b);
}

int main() {
    int n;
```



```c
int compare(const void *a, const void *b) {
    return (*(int*)a - *(int*)b);
}

int main() {
    int n;
    scanf("%d", &n);

    int arr[n];
    for (int i = 0; i < n; i++) {
        scanf("%d", &arr[i]);
    }

    qsort(arr, n, sizeof(int), compare);

    long long maxSum = 0;
    for (int i = 0; i < n; i++) {
        maxSum += (long long)arr[i] * i;
    }

    printf("%lld\n", maxSum);
    return 0;
}

```



	Input	Expected	Got	
✓	5 2 5 3 4 0	40	40	✓
✓	10 2 2 2 4 4 3 3 5 5 5	191	191	✓
✓	2	45	45	✓

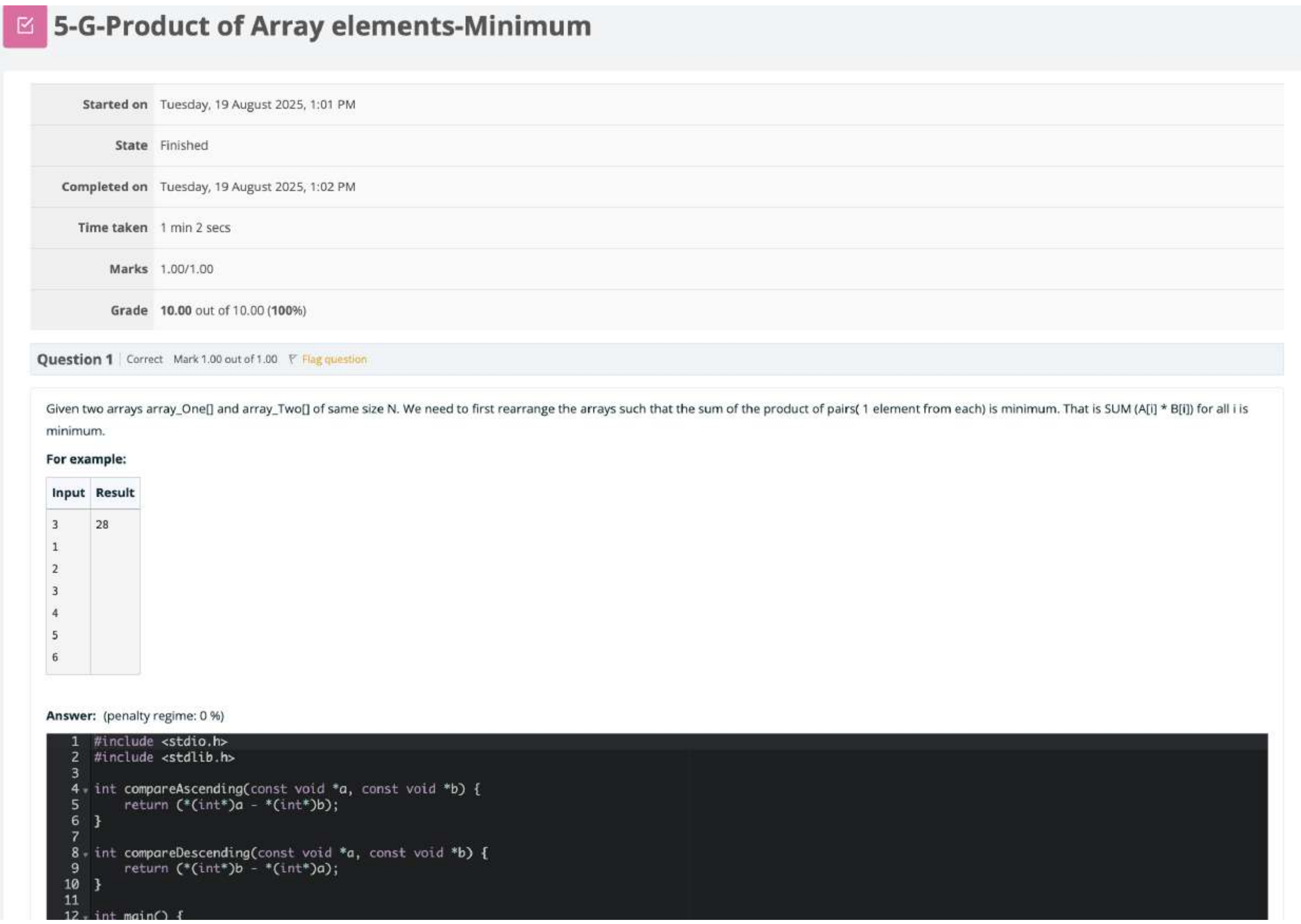
	Input	Expected	Got	
✓	5 2 5 3 4 0	40	40	✓
✓	10 2 2 2 4 4 3 3 5 5 5	191	191	✓
✓	2 45 3	45	45	✓

Passed all tests! ✓

Correct

Marks for this submission: 1.00/1.00.

Finish review



5-G-Product of Array elements-Minimum

Started on	Tuesday, 19 August 2025, 1:01 PM
State	Finished
Completed on	Tuesday, 19 August 2025, 1:02 PM
Time taken	1 min 2 secs
Marks	1.00/1.00
Grade	**10.00** out of 10.00 (**100%**)

Question 1 Correct Mark 1.00 out of 1.00 Flag question

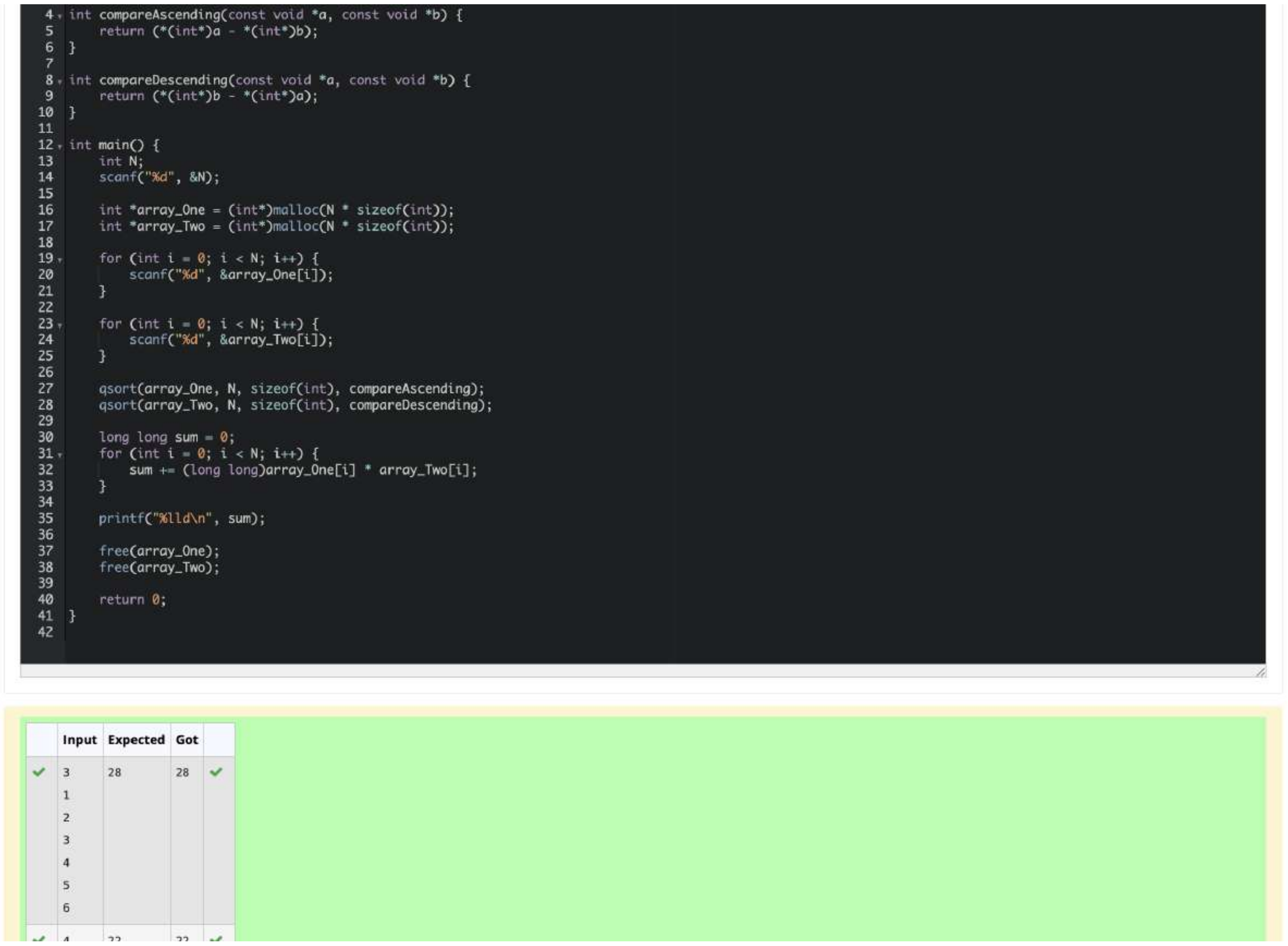
Given two arrays array_One[] and array_Two[] of same size N. We need to first rearrange the arrays such that the sum of the product of pairs(1 element from each) is minimum. That is SUM (A[i] * B[i]) for all i is minimum.

For example:

Input	Result
3	28
1	
2	
3	
4	
5	
6	

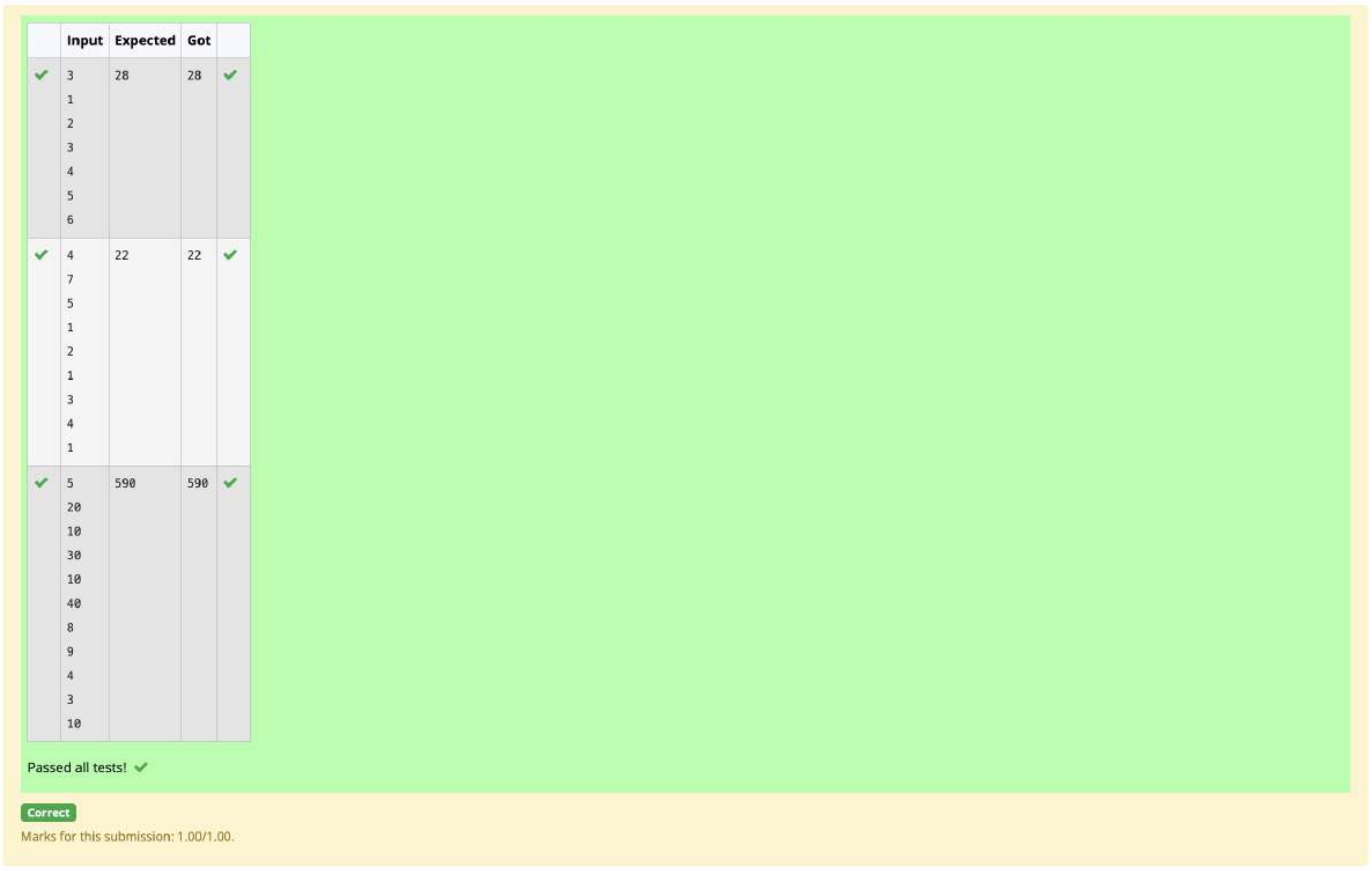
Answer: (penalty regime: 0 %)

```c
#include <stdio.h>
#include <stdlib.h>

int compareAscending(const void *a, const void *b) {
    return (*(int*)a - *(int*)b);
}

int compareDescending(const void *a, const void *b) {
    return (*(int*)b - *(int*)a);
}

int main() {
```



```c
int compareAscending(const void *a, const void *b) {
    return (*(int*)a - *(int*)b);
}

int compareDescending(const void *a, const void *b) {
    return (*(int*)b - *(int*)a);
}

int main() {
    int N;
    scanf("%d", &N);

    int *array_One = (int*)malloc(N * sizeof(int));
    int *array_Two = (int*)malloc(N * sizeof(int));

    for (int i = 0; i < N; i++) {
        scanf("%d", &array_One[i]);
    }

    for (int i = 0; i < N; i++) {
        scanf("%d", &array_Two[i]);
    }

    qsort(array_One, N, sizeof(int), compareAscending);
    qsort(array_Two, N, sizeof(int), compareDescending);

    long long sum = 0;
    for (int i = 0; i < N; i++) {
        sum += (long long)array_One[i] * array_Two[i];
    }

    printf("%lld\n", sum);

    free(array_One);
    free(array_Two);

    return 0;
}

```

	Input	Expected	Got	
✓	3 1 2 3 4 5 6	28	28	✓
✓	4	22	22	✓



	Input	Expected	Got	
✓	3	28	28	✓
	1			
	2			
	3			
	4			
	5			
	6			
✓	4	22	22	✓
	7			
	5			
	1			
	2			
	1			
	3			
	4			
	1			
✓	5	590	590	✓
	20			
	10			
	30			
	10			
	40			
	8			
	9			
	4			
	3			
	10			

Passed all tests! ✓

Correct

Marks for this submission: 1.00/1.00.

Finish review