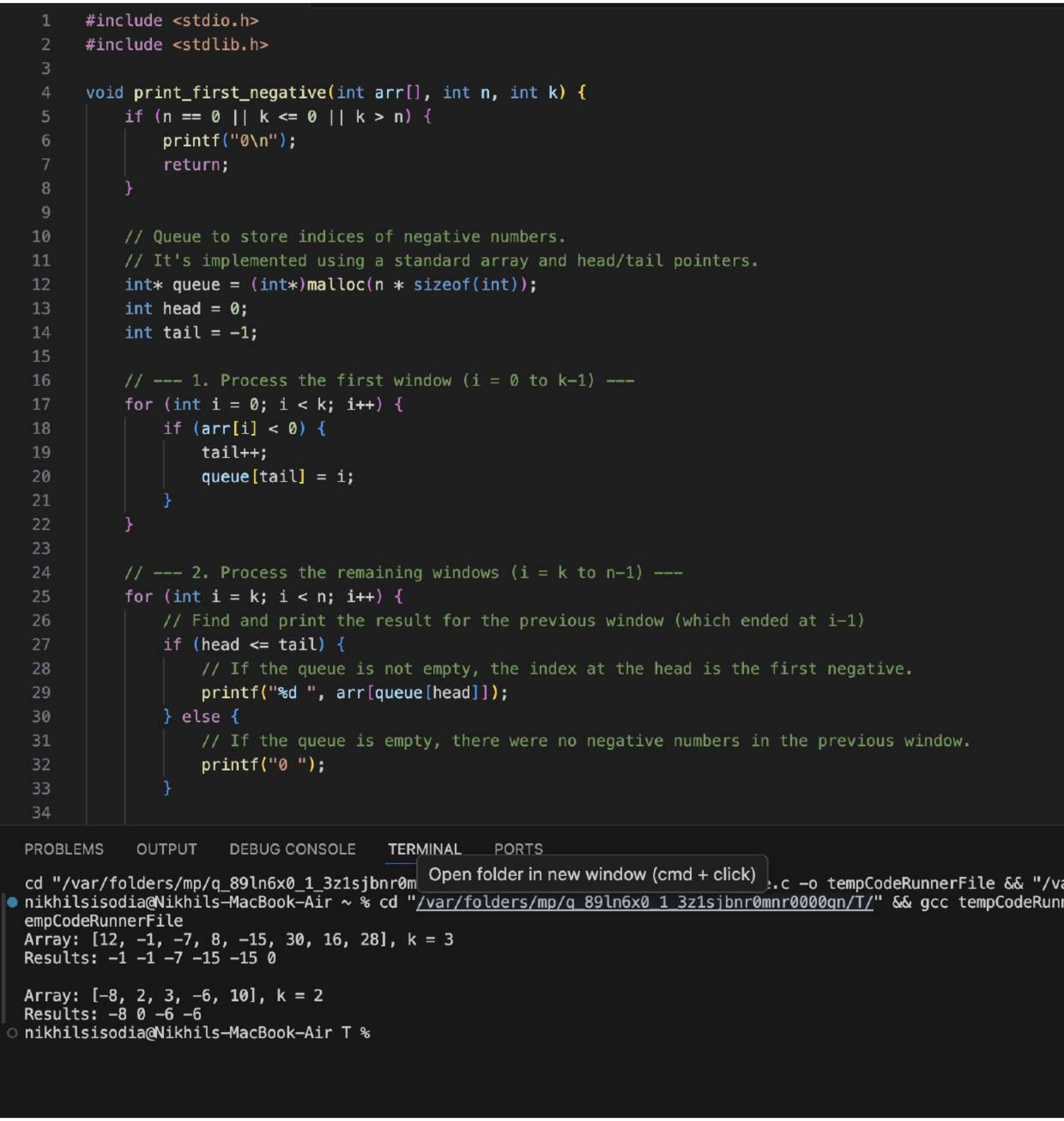
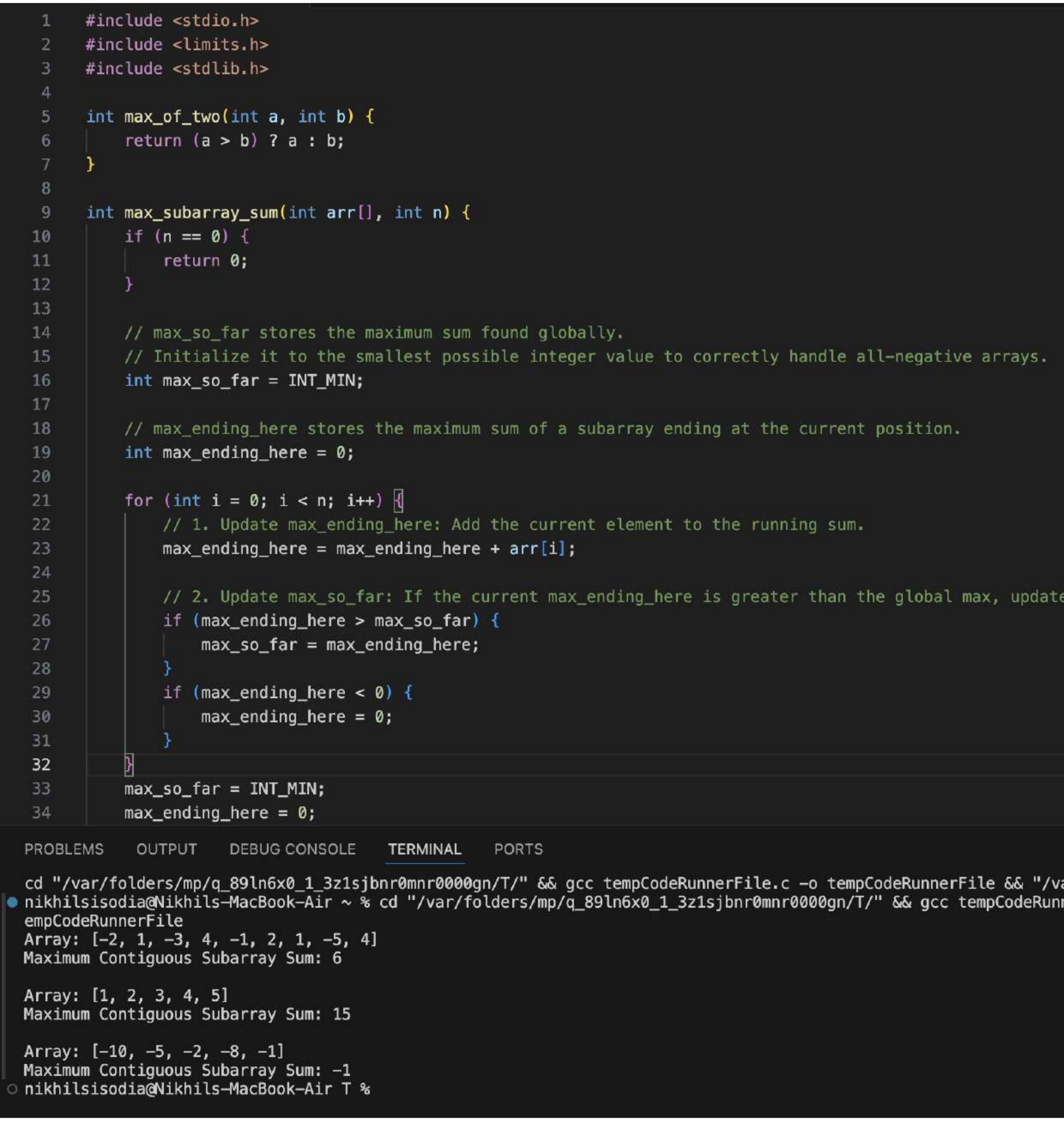


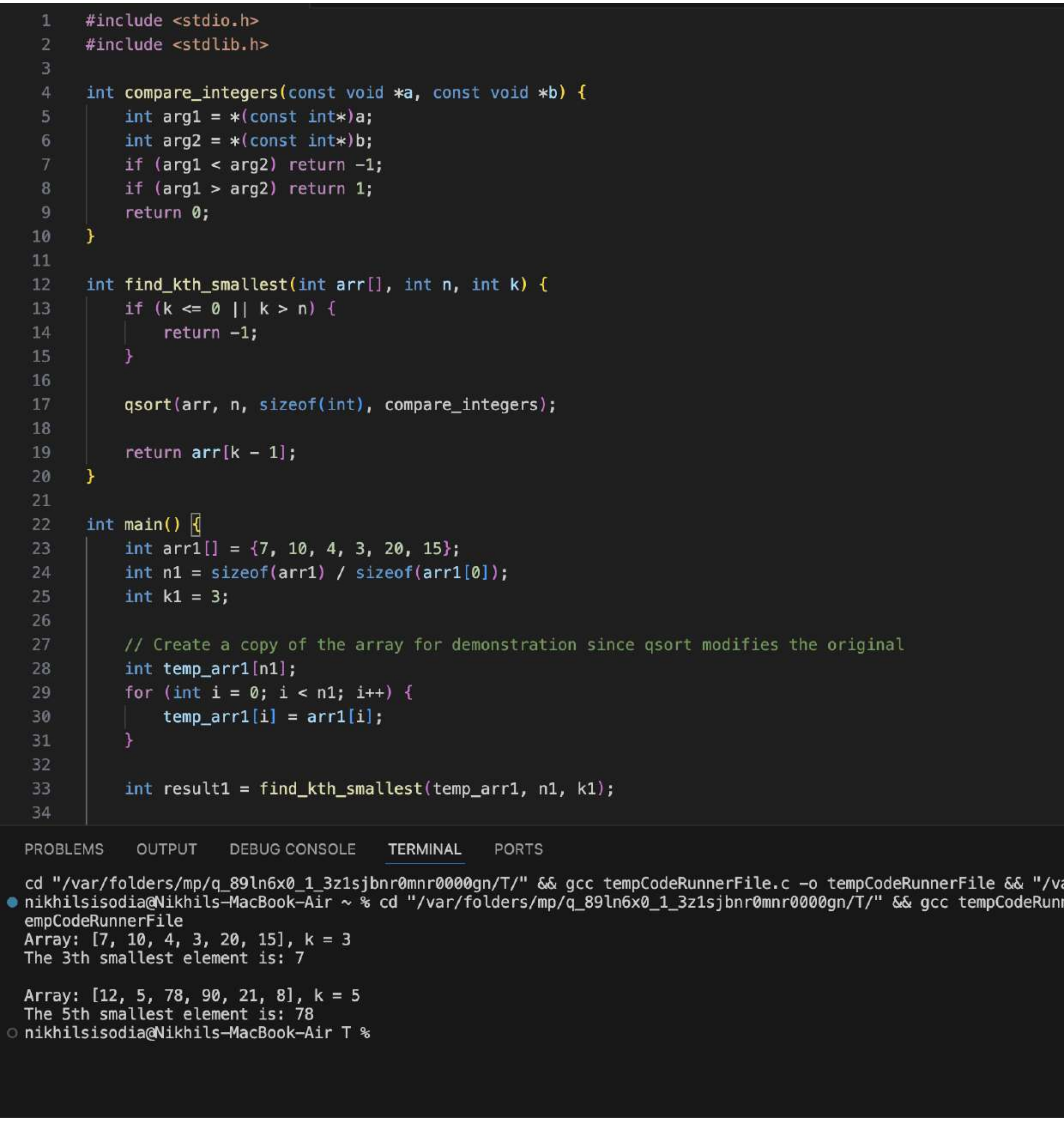
```c
#include <stdio.h>
#include <stdlib.h>

void print_first_negative(int arr[], int n, int k) {
    if (n == 0 || k <= 0 || k > n) {
        printf("0\n");
        return;
    }

    // Queue to store indices of negative numbers.
    // It's implemented using a standard array and head/tail pointers.
    int* queue = (int*)malloc(n * sizeof(int));
    int head = 0;
    int tail = -1;

    // --- 1. Process the first window (i = 0 to k-1) ---
    for (int i = 0; i < k; i++) {
        if (arr[i] < 0) {
            tail++;
            queue[tail] = i;
        }
    }

    // --- 2. Process the remaining windows (i = k to n-1) ---
    for (int i = k; i < n; i++) {
        // Find and print the result for the previous window (which ended at i-1)
        if (head <= tail) {
            // If the queue is not empty, the index at the head is the first negative.
            printf("%d ", arr[queue[head]]);
        } else {
            // If the queue is empty, there were no negative numbers in the previous window.
            printf("0 ");
        }

```

```
cd "/var/folders/mp/q_89ln6x0_1_3z1sjbnr0m                                          .c -o tempCodeRunnerFile && "/va
nikhilsisodia@Nikhils-MacBook-Air ~ % cd "/var/folders/mp/q_89ln6x0_1_3z1sjbnr0mnr0000gn/T/" && gcc tempCodeRunn
empCodeRunnerFile
Array: [12, -1, -7, 8, -15, 30, 16, 28], k = 3
Results: -1 -1 -7 -15 -15 0

Array: [-8, 2, 3, -6, 10], k = 2
Results: -8 0 -6 -6
nikhilsisodia@Nikhils-MacBook-Air T %
```



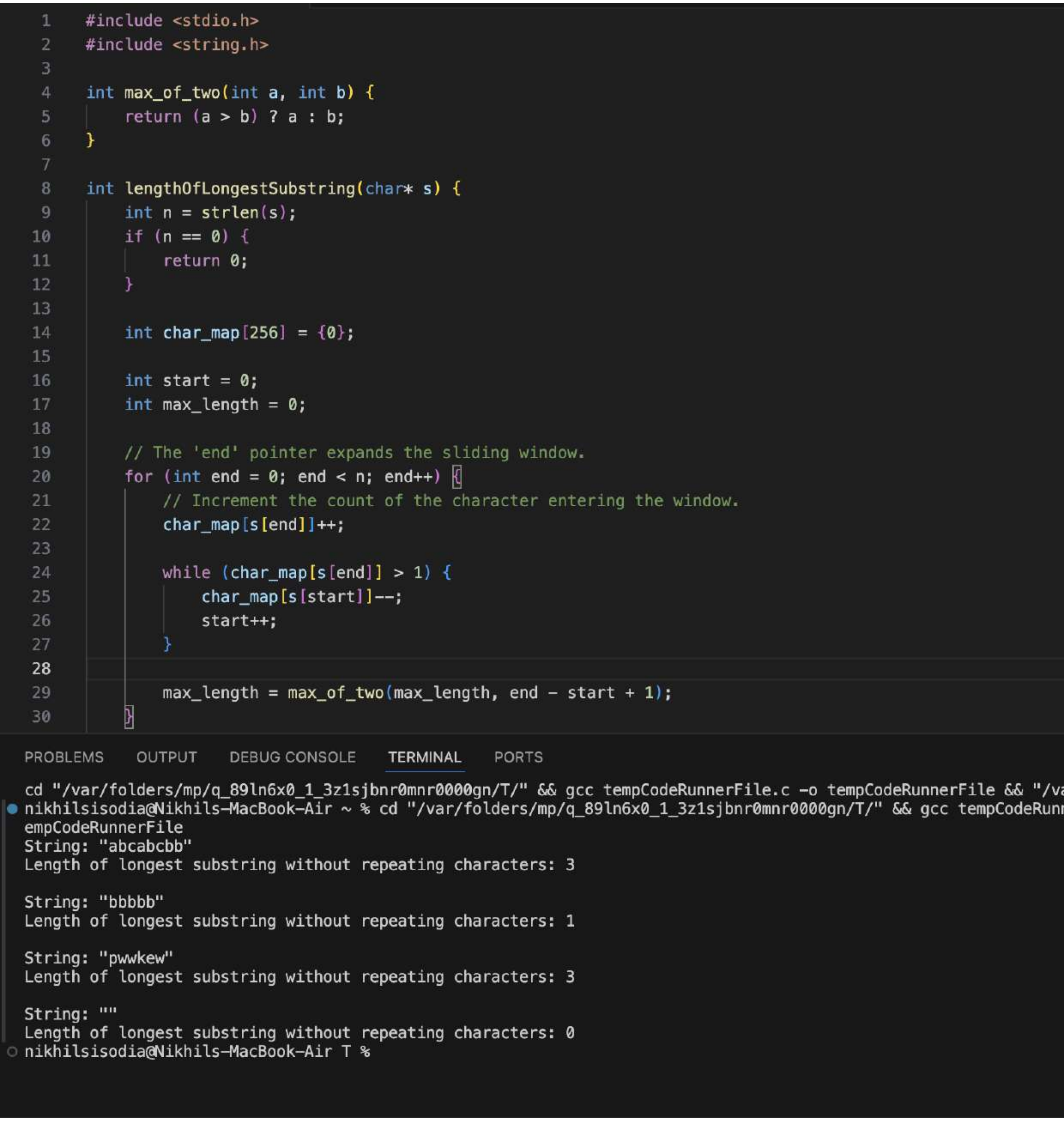
```c
#include <stdio.h>
#include <limits.h>
#include <stdlib.h>

int max_of_two(int a, int b) {
    return (a > b) ? a : b;
}

int max_subarray_sum(int arr[], int n) {
    if (n == 0) {
        return 0;
    }

    // max_so_far stores the maximum sum found globally.
    // Initialize it to the smallest possible integer value to correctly handle all-negative arrays.
    int max_so_far = INT_MIN;

    // max_ending_here stores the maximum sum of a subarray ending at the current position.
    int max_ending_here = 0;

    for (int i = 0; i < n; i++) {
        // 1. Update max_ending_here: Add the current element to the running sum.
        max_ending_here = max_ending_here + arr[i];

        // 2. Update max_so_far: If the current max_ending_here is greater than the global max, update
        if (max_ending_here > max_so_far) {
            max_so_far = max_ending_here;
        }
        if (max_ending_here < 0) {
            max_ending_here = 0;
        }
    }
    max_so_far = INT_MIN;
    max_ending_here = 0;
```

```
cd "/var/folders/mp/q_89ln6x0_1_3z1sjbnr0mnr0000gn/T/" && gcc tempCodeRunnerFile.c -o tempCodeRunnerFile && "/va
nikhilsisodia@Nikhils-MacBook-Air ~ % cd "/var/folders/mp/q_89ln6x0_1_3z1sjbnr0mnr0000gn/T/" && gcc tempCodeRunn
empCodeRunnerFile
Array: [-2, 1, -3, 4, -1, 2, 1, -5, 4]
Maximum Contiguous Subarray Sum: 6

Array: [1, 2, 3, 4, 5]
Maximum Contiguous Subarray Sum: 15

Array: [-10, -5, -2, -8, -1]
Maximum Contiguous Subarray Sum: -1
nikhilsisodia@Nikhils-MacBook-Air T %
```



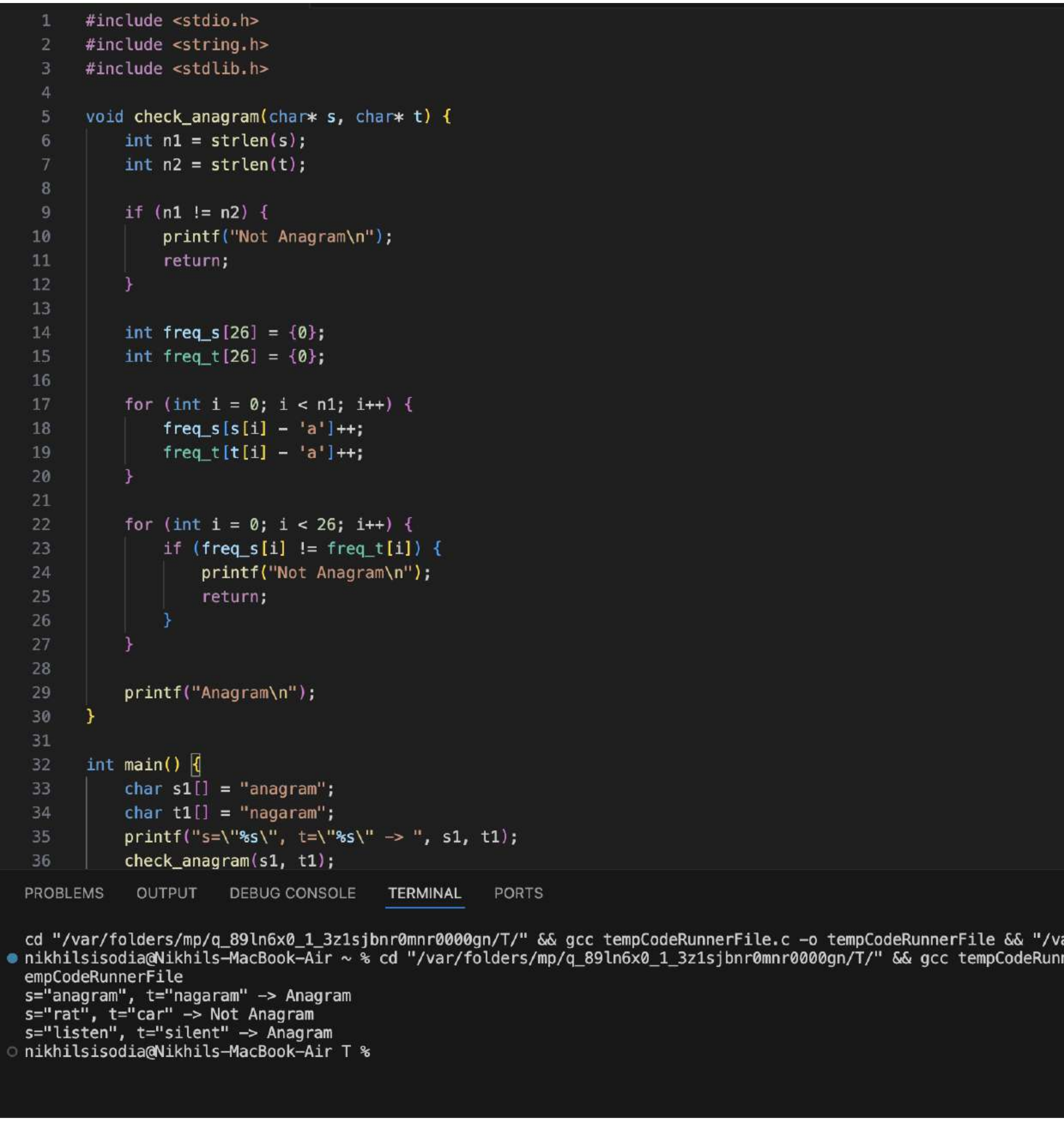
```c
#include <stdio.h>
#include <stdlib.h>

int compare_integers(const void *a, const void *b) {
    int arg1 = *(const int*)a;
    int arg2 = *(const int*)b;
    if (arg1 < arg2) return -1;
    if (arg1 > arg2) return 1;
    return 0;
}

int find_kth_smallest(int arr[], int n, int k) {
    if (k <= 0 || k > n) {
        return -1;
    }

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

    return arr[k - 1];
}

int main() {
    int arr1[] = {7, 10, 4, 3, 20, 15};
    int n1 = sizeof(arr1) / sizeof(arr1[0]);
    int k1 = 3;

    // Create a copy of the array for demonstration since qsort modifies the original
    int temp_arr1[n1];
    for (int i = 0; i < n1; i++) {
        temp_arr1[i] = arr1[i];
    }

    int result1 = find_kth_smallest(temp_arr1, n1, k1);

```

```
cd "/var/folders/mp/q_89ln6x0_1_3z1sjbnr0mnr0000gn/T/" && gcc tempCodeRunnerFile.c -o tempCodeRunnerFile && "/va
● nikhilsisodia@Nikhils-MacBook-Air ~ % cd "/var/folders/mp/q_89ln6x0_1_3z1sjbnr0mnr0000gn/T/" && gcc tempCodeRunn
empCodeRunnerFile
Array: [7, 10, 4, 3, 20, 15], k = 3
The 3th smallest element is: 7

Array: [12, 5, 78, 90, 21, 8], k = 5
The 5th smallest element is: 78
○ nikhilsisodia@Nikhils-MacBook-Air T %
```



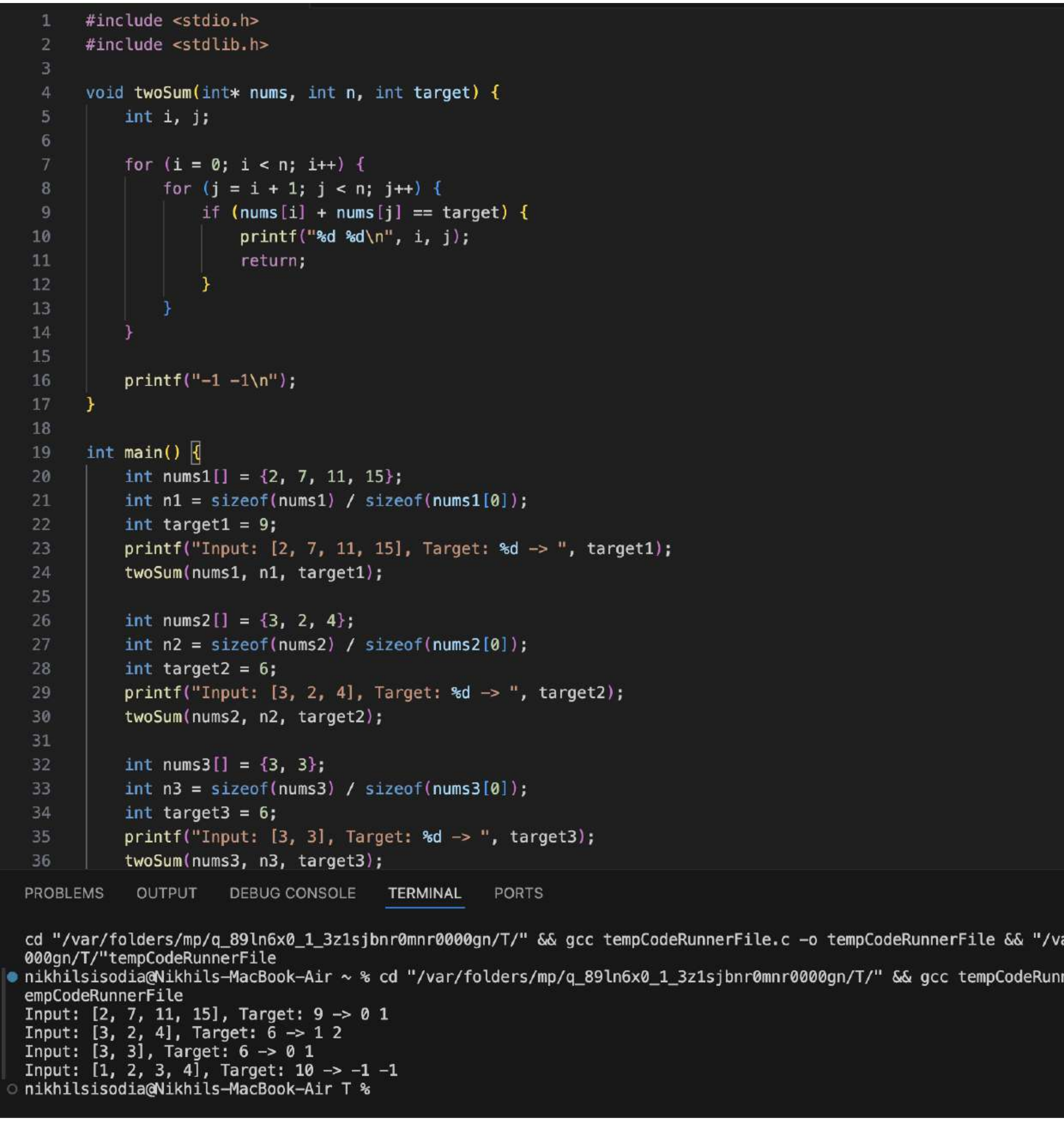
```c
#include <stdio.h>
#include <string.h>

int max_of_two(int a, int b) {
    return (a > b) ? a : b;
}

int lengthOfLongestSubstring(char* s) {
    int n = strlen(s);
    if (n == 0) {
        return 0;
    }

    int char_map[256] = {0};

    int start = 0;
    int max_length = 0;

    // The 'end' pointer expands the sliding window.
    for (int end = 0; end < n; end++) {
        // Increment the count of the character entering the window.
        char_map[s[end]]++;

        while (char_map[s[end]] > 1) {
            char_map[s[start]]--;
            start++;
        }

        max_length = max_of_two(max_length, end - start + 1);
    }
```

```
cd "/var/folders/mp/q_89ln6x0_1_3z1sjbnr0mnr0000gn/T/" && gcc tempCodeRunnerFile.c -o tempCodeRunnerFile && "/va
nikhilsisodia@Nikhils-MacBook-Air ~ % cd "/var/folders/mp/q_89ln6x0_1_3z1sjbnr0mnr0000gn/T/" && gcc tempCodeRunn
empCodeRunnerFile
String: "abcabcbb"
Length of longest substring without repeating characters: 3

String: "bbbbb"
Length of longest substring without repeating characters: 1

String: "pwwkew"
Length of longest substring without repeating characters: 3

String: ""
Length of longest substring without repeating characters: 0
nikhilsisodia@Nikhils-MacBook-Air T %
```



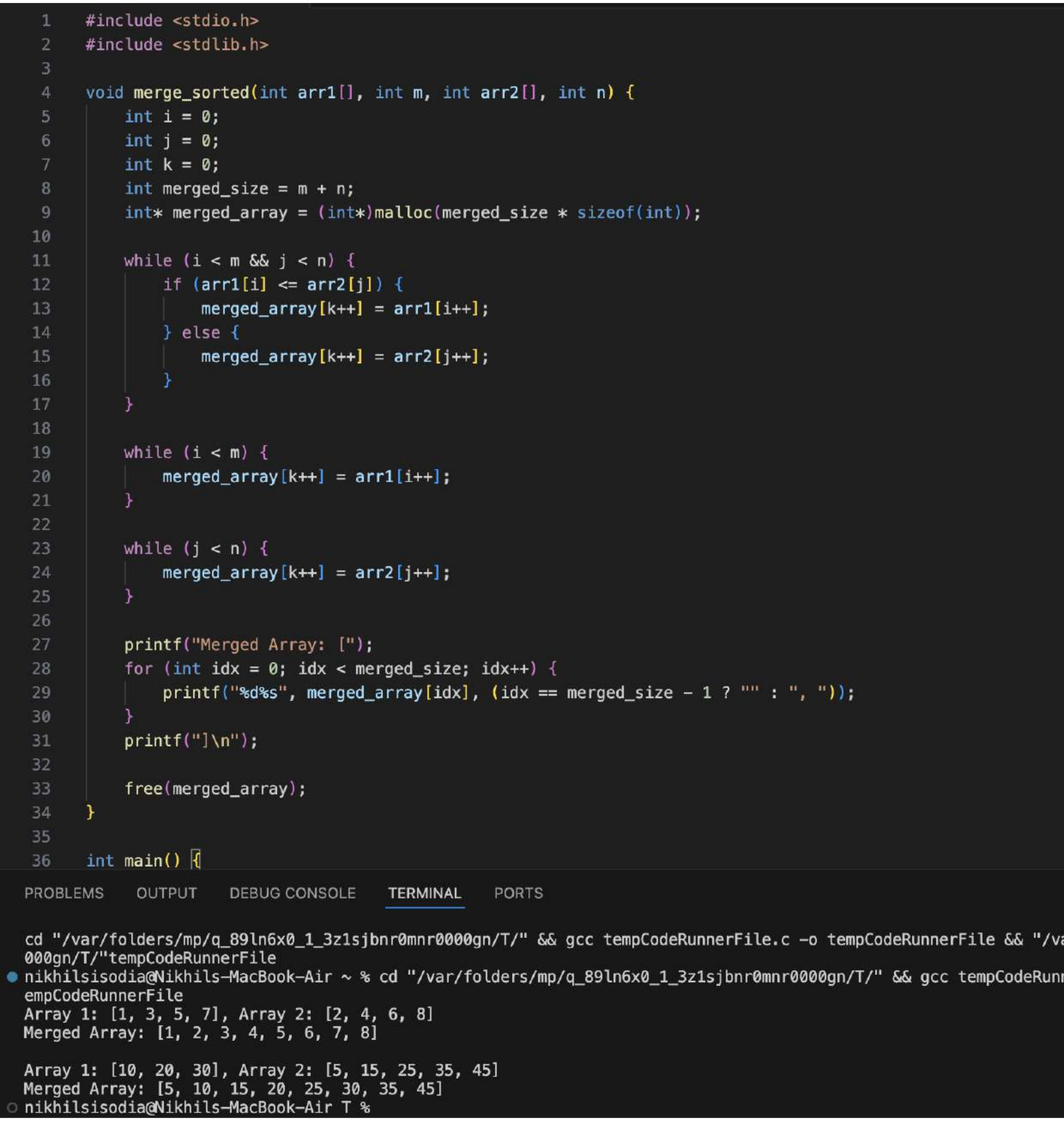
```c
#include <stdio.h>
#include <string.h>
#include <stdlib.h>

void check_anagram(char* s, char* t) {
    int n1 = strlen(s);
    int n2 = strlen(t);

    if (n1 != n2) {
        printf("Not Anagram\n");
        return;
    }

    int freq_s[26] = {0};
    int freq_t[26] = {0};

    for (int i = 0; i < n1; i++) {
        freq_s[s[i] - 'a']++;
        freq_t[t[i] - 'a']++;
    }

    for (int i = 0; i < 26; i++) {
        if (freq_s[i] != freq_t[i]) {
            printf("Not Anagram\n");
            return;
        }
    }

    printf("Anagram\n");
}

int main() {
    char s1[] = "anagram";
    char t1[] = "nagaram";
    printf("s=\"%s\", t=\"%s\" -> ", s1, t1);
    check_anagram(s1, t1);
```

```
cd "/var/folders/mp/q_89ln6x0_1_3z1sjbnr0mnr0000gn/T/" && gcc tempCodeRunnerFile.c -o tempCodeRunnerFile && "/va
● nikhilsisodia@Nikhils-MacBook-Air ~ % cd "/var/folders/mp/q_89ln6x0_1_3z1sjbnr0mnr0000gn/T/" && gcc tempCodeRunn
empCodeRunnerFile
s="anagram", t="nagaram" -> Anagram
s="rat", t="car" -> Not Anagram
s="listen", t="silent" -> Anagram
○ nikhilsisodia@Nikhils-MacBook-Air T %
```



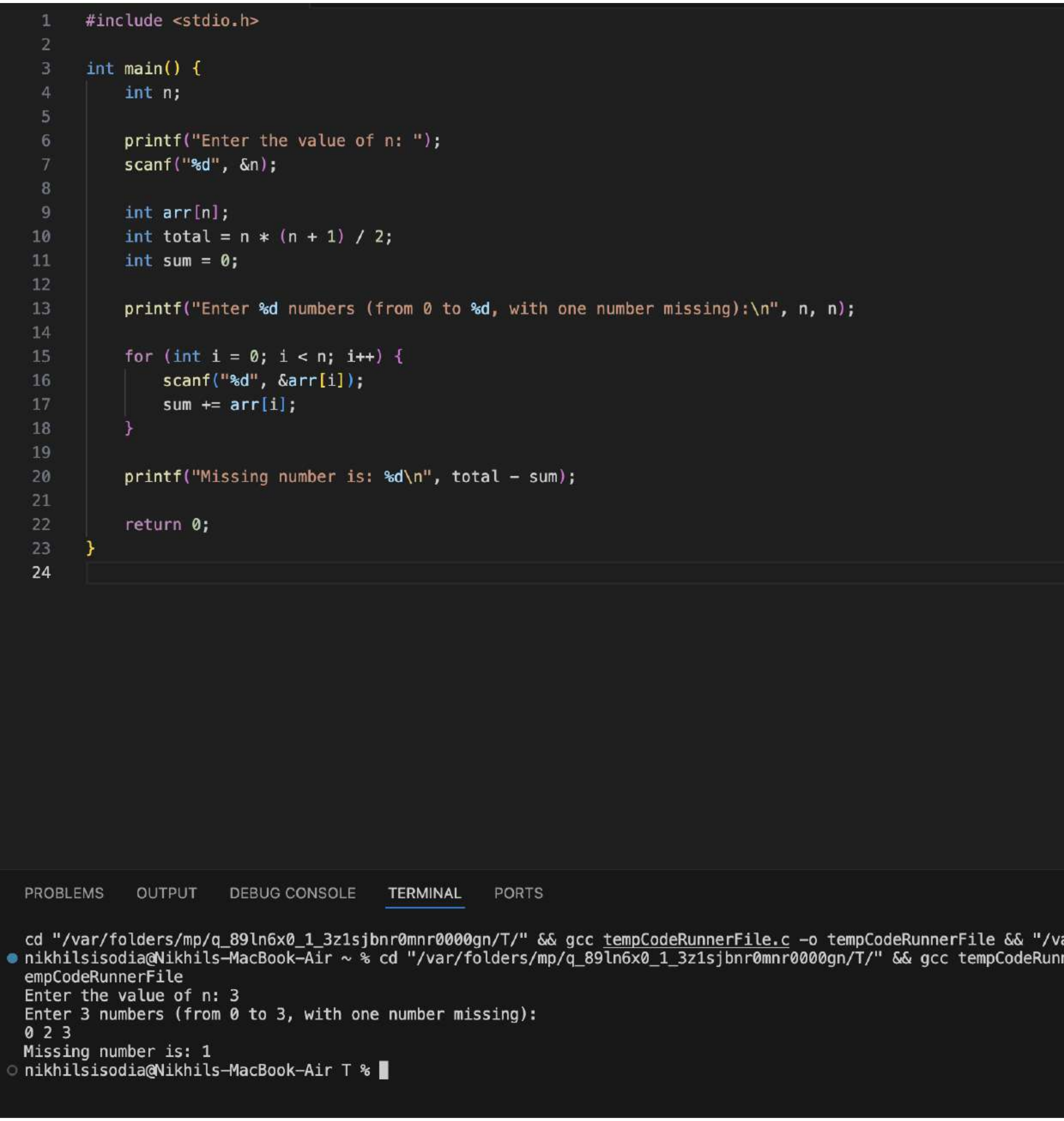
```c
#include <stdio.h>
#include <stdlib.h>

void twoSum(int* nums, int n, int target) {
    int i, j;

    for (i = 0; i < n; i++) {
        for (j = i + 1; j < n; j++) {
            if (nums[i] + nums[j] == target) {
                printf("%d %d\n", i, j);
                return;
            }
        }
    }

    printf("-1 -1\n");
}

int main() {
    int nums1[] = {2, 7, 11, 15};
    int n1 = sizeof(nums1) / sizeof(nums1[0]);
    int target1 = 9;
    printf("Input: [2, 7, 11, 15], Target: %d -> ", target1);
    twoSum(nums1, n1, target1);

    int nums2[] = {3, 2, 4};
    int n2 = sizeof(nums2) / sizeof(nums2[0]);
    int target2 = 6;
    printf("Input: [3, 2, 4], Target: %d -> ", target2);
    twoSum(nums2, n2, target2);

    int nums3[] = {3, 3};
    int n3 = sizeof(nums3) / sizeof(nums3[0]);
    int target3 = 6;
    printf("Input: [3, 3], Target: %d -> ", target3);
    twoSum(nums3, n3, target3);
```

```
cd "/var/folders/mp/q_89ln6x0_1_3z1sjbnr0mnr0000gn/T/" && gcc tempCodeRunnerFile.c -o tempCodeRunnerFile && "/va
000gn/T/"tempCodeRunnerFile
● nikhilsisodia@Nikhils-MacBook-Air ~ % cd "/var/folders/mp/q_89ln6x0_1_3z1sjbnr0mnr0000gn/T/" && gcc tempCodeRunn
empCodeRunnerFile
Input: [2, 7, 11, 15], Target: 9 -> 0 1
Input: [3, 2, 4], Target: 6 -> 1 2
Input: [3, 3], Target: 6 -> 0 1
Input: [1, 2, 3, 4], Target: 10 -> -1 -1
○ nikhilsisodia@Nikhils-MacBook-Air T %
```



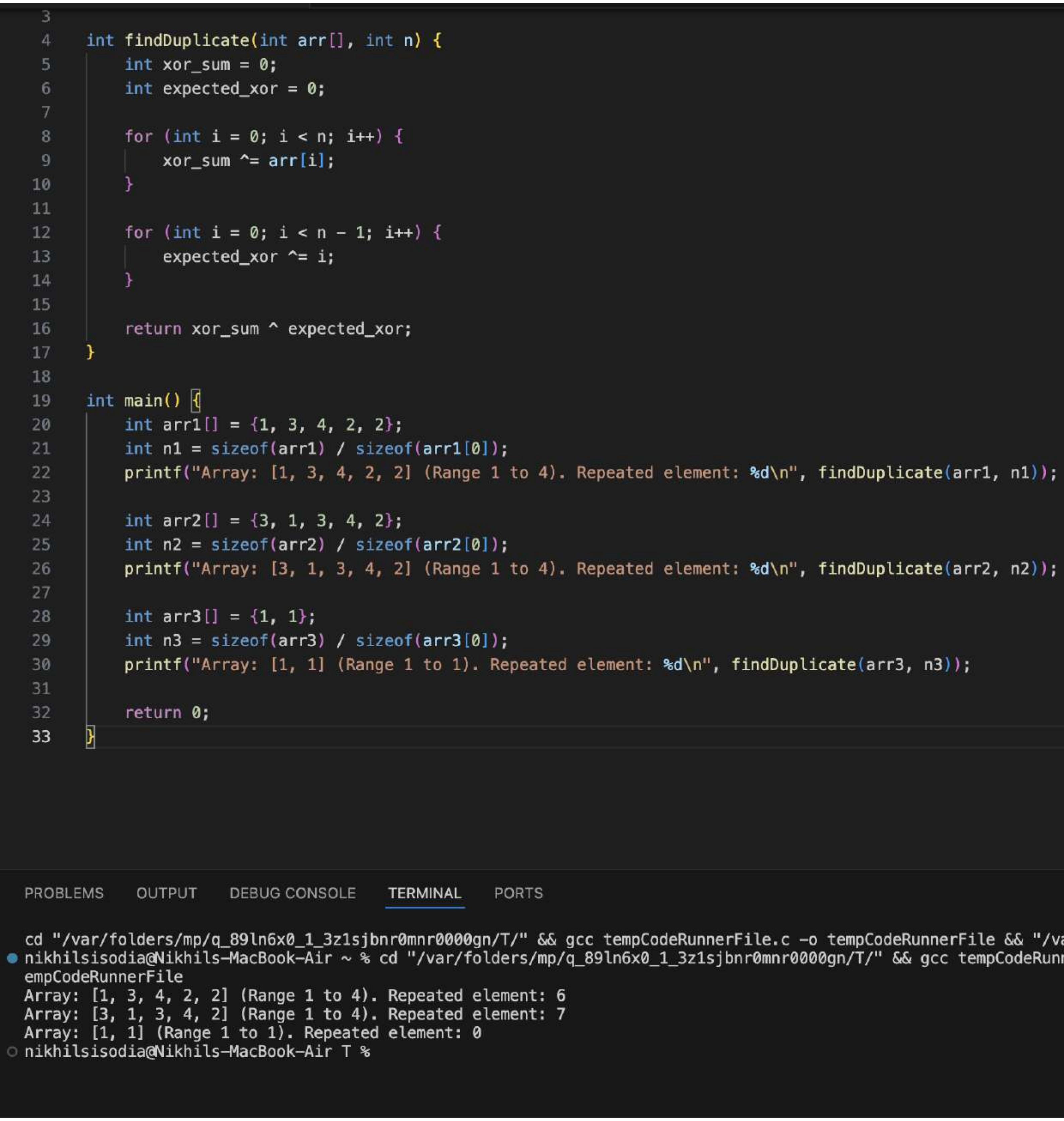
```c
#include <stdio.h>
#include <stdlib.h>

void merge_sorted(int arr1[], int m, int arr2[], int n) {
    int i = 0;
    int j = 0;
    int k = 0;
    int merged_size = m + n;
    int* merged_array = (int*)malloc(merged_size * sizeof(int));

    while (i < m && j < n) {
        if (arr1[i] <= arr2[j]) {
            merged_array[k++] = arr1[i++];
        } else {
            merged_array[k++] = arr2[j++];
        }
    }

    while (i < m) {
        merged_array[k++] = arr1[i++];
    }

    while (j < n) {
        merged_array[k++] = arr2[j++];
    }

    printf("Merged Array: [");
    for (int idx = 0; idx < merged_size; idx++) {
        printf("%d%s", merged_array[idx], (idx == merged_size - 1 ? "" : ", "));
    }
    printf("]\n");

    free(merged_array);
}

int main() {
```

```
cd "/var/folders/mp/q_89ln6x0_1_3z1sjbnr0mnr0000gn/T/" && gcc tempCodeRunnerFile.c -o tempCodeRunnerFile && "/va
000gn/T/"tempCodeRunnerFile
● nikhilsisodia@Nikhils-MacBook-Air ~ % cd "/var/folders/mp/q_89ln6x0_1_3z1sjbnr0mnr0000gn/T/" && gcc tempCodeRunn
empCodeRunnerFile
Array 1: [1, 3, 5, 7], Array 2: [2, 4, 6, 8]
Merged Array: [1, 2, 3, 4, 5, 6, 7, 8]

Array 1: [10, 20, 30], Array 2: [5, 15, 25, 35, 45]
Merged Array: [5, 10, 15, 20, 25, 30, 35, 45]
○ nikhilsisodia@Nikhils-MacBook-Air T %
```



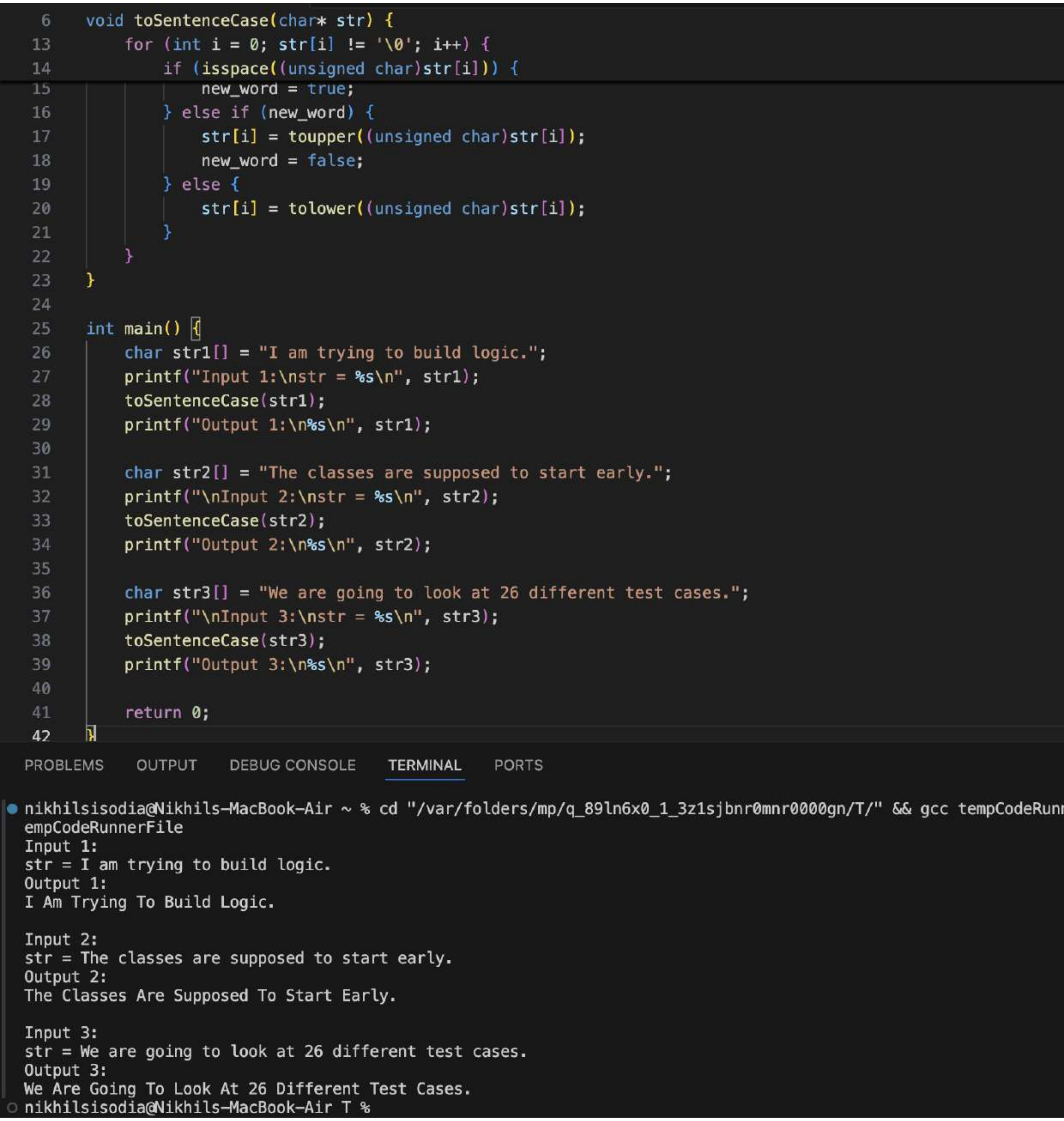
```c
#include <stdio.h>

int main() {
    int n;

    printf("Enter the value of n: ");
    scanf("%d", &n);

    int arr[n];
    int total = n * (n + 1) / 2;
    int sum = 0;

    printf("Enter %d numbers (from 0 to %d, with one number missing):\n", n, n);

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

    printf("Missing number is: %d\n", total - sum);

    return 0;
}
```

```
cd "/var/folders/mp/q_89ln6x0_1_3z1sjbnr0mnr0000gn/T/" && gcc tempCodeRunnerFile.c -o tempCodeRunnerFile && "/va
nikhilsisodia@Nikhils-MacBook-Air ~ % cd "/var/folders/mp/q_89ln6x0_1_3z1sjbnr0mnr0000gn/T/" && gcc tempCodeRunn
empCodeRunnerFile
Enter the value of n: 3
Enter 3 numbers (from 0 to 3, with one number missing):
0 2 3
Missing number is: 1
nikhilsisodia@Nikhils-MacBook-Air T % 
```



```c

int findDuplicate(int arr[], int n) {
    int xor_sum = 0;
    int expected_xor = 0;

    for (int i = 0; i < n; i++) {
        xor_sum ^= arr[i];
    }

    for (int i = 0; i < n - 1; i++) {
        expected_xor ^= i;
    }

    return xor_sum ^ expected_xor;
}

int main() {
    int arr1[] = {1, 3, 4, 2, 2};
    int n1 = sizeof(arr1) / sizeof(arr1[0]);
    printf("Array: [1, 3, 4, 2, 2] (Range 1 to 4). Repeated element: %d\n", findDuplicate(arr1, n1));

    int arr2[] = {3, 1, 3, 4, 2};
    int n2 = sizeof(arr2) / sizeof(arr2[0]);
    printf("Array: [3, 1, 3, 4, 2] (Range 1 to 4). Repeated element: %d\n", findDuplicate(arr2, n2));

    int arr3[] = {1, 1};
    int n3 = sizeof(arr3) / sizeof(arr3[0]);
    printf("Array: [1, 1] (Range 1 to 1). Repeated element: %d\n", findDuplicate(arr3, n3));

    return 0;
}
```

```
cd "/var/folders/mp/q_89ln6x0_1_3z1sjbnr0mnr0000gn/T/" && gcc tempCodeRunnerFile.c -o tempCodeRunnerFile && "/va
● nikhilsisodia@Nikhils-MacBook-Air ~ % cd "/var/folders/mp/q_89ln6x0_1_3z1sjbnr0mnr0000gn/T/" && gcc tempCodeRunn
empCodeRunnerFile
Array: [1, 3, 4, 2, 2] (Range 1 to 4). Repeated element: 6
Array: [3, 1, 3, 4, 2] (Range 1 to 4). Repeated element: 7
Array: [1, 1] (Range 1 to 1). Repeated element: 0
○ nikhilsisodia@Nikhils-MacBook-Air T %
```



```c
void toSentenceCase(char* str) {
    for (int i = 0; str[i] != '\0'; i++) {
        if (isspace((unsigned char)str[i])) {
            new_word = true;
        } else if (new_word) {
            str[i] = toupper((unsigned char)str[i]);
            new_word = false;
        } else {
            str[i] = tolower((unsigned char)str[i]);
        }
    }
}

int main() {
    char str1[] = "I am trying to build logic.";
    printf("Input 1:\nstr = %s\n", str1);
    toSentenceCase(str1);
    printf("Output 1:\n%s\n", str1);

    char str2[] = "The classes are supposed to start early.";
    printf("\nInput 2:\nstr = %s\n", str2);
    toSentenceCase(str2);
    printf("Output 2:\n%s\n", str2);

    char str3[] = "We are going to look at 26 different test cases.";
    printf("\nInput 3:\nstr = %s\n", str3);
    toSentenceCase(str3);
    printf("Output 3:\n%s\n", str3);

    return 0;
}
```

```
nikhilsisodia@Nikhils-MacBook-Air ~ % cd "/var/folders/mp/q_89ln6x0_1_3z1sjbnr0mnr0000gn/T/" && gcc tempCodeRunn
empCodeRunnerFile
Input 1:
str = I am trying to build logic.
Output 1:
I Am Trying To Build Logic.

Input 2:
str = The classes are supposed to start early.
Output 2:
The Classes Are Supposed To Start Early.

Input 3:
str = We are going to look at 26 different test cases.
Output 3:
We Are Going To Look At 26 Different Test Cases.
nikhilsisodia@Nikhils-MacBook-Air T %
```