EE 610 Fundamentals of VLSI CAD

Submitted By

Karan Markandey

210102042

Assignment-01

Design an algorithm to find the smallest element in an array such that the time complexity of the algorithm is linear, denoted as O(kn), where k is a parameter that can vary. Your goal is to minimise the value of k while ensuring that the algorithm still operates in linear time. What is the optimal value of k for which the algorithm remains linear, and how can you achieve this.

Code Link :- Github Link

Code Explanation

1. The code starts by including the <bits/stdc++.h> header, which includes most standard C++ libraries. It also defines a constant INF set to a large value (10^9 + 7) to represent infinity.
2. The main function begins by reading an integer n, which represents the size of the input array.
3. If n is less than 3, the program prints that the third smallest element doesn't exist and exits.
4. A vector a of size n is created to store the input elements.
5. The program then reads n integers into the vector a.
6. Three variables mi, mi1, and mi2 are initialised to INF. These will store the smallest, second smallest, and third smallest elements respectively.
7. The code iterates through the array once, updating these variables as follows:

- If the current element is smaller than or equal to mi, it shifts mi to mi1, mi1 to mi2, and updates mi with the current element.
- If it's larger than mi but smaller than or equal to mi1, it shifts mi1 to mi2 and updates mi1.
- If it's larger than mi1 but smaller than mi2, it updates mi2.

8. Finally, the program prints the third smallest element, which is stored in mi2.

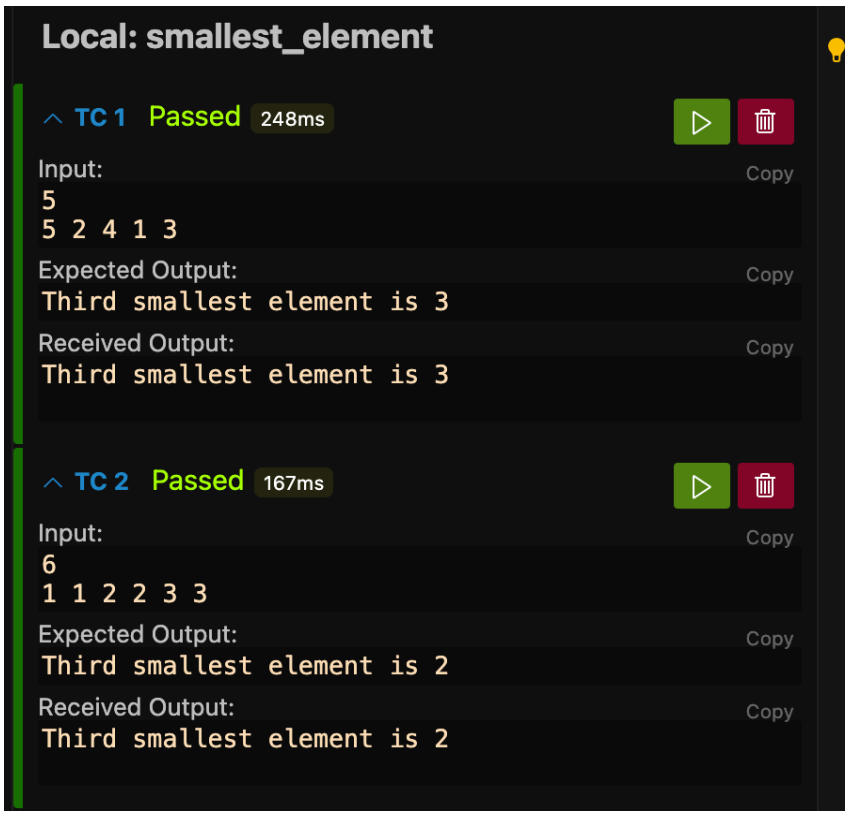
Time Complexity

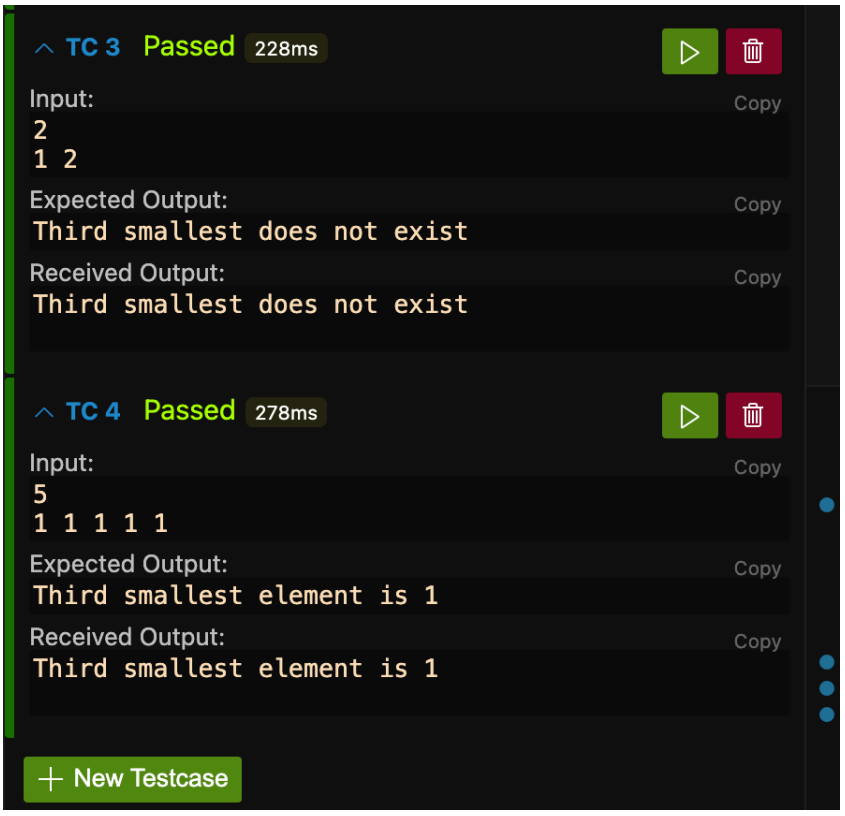
The time complexity of this algorithm is O(n), where n is the number of elements in the array. The program makes a single pass through the array, performing constant-time operations for each element.

Space Complexity

The space complexity is O(n), where n is the number of elements in the array. This is due to the vector a which stores all the input elements. The additional variables (mi, mi1, mi2) use constant space.

Results


Local: smallest_element
TC 1 Passed 248ms
Input:
5
5 2 4 1 3
Copy
Expected Output:
Third smallest element is 3
Copy
Received Output:
Third smallest element is 3
Copy
TC 2 Passed 167ms
Input:
6
1 1 2 2 3 3
Copy
Expected Output:
Third smallest element is 2
Copy
Received Output:
Third smallest element is 2
Copy


TC 3 Passed 228ms
Input:
2
1 2
Copy
Expected Output:
Third smallest does not exist
Copy
Received Output:
Third smallest does not exist
Copy
TC 4 Passed 278ms
Input:
5
1 1 1 1 1
Copy
Expected Output:
Third smallest element is 1
Copy
Received Output:
Third smallest element is 1
Copy
New Testcase