Hands On 2: Competitive Programming and Contests

1.1 Introduction

The solutions for both problems in this section have been implemented using Segment Trees with lazy propagation. These data structures are tailored for handling `i32` numbers efficiently, focusing on fast query and update operations. The segment tree is designed to optimize the process of range updates and queries by utilizing lazy propagation, which helps in reducing unnecessary computations.

1.2 Min Max Problem

In this problem, a recursive structure was used to implement the Segment Tree. A Segment Tree is a naturally recursive data structure, and this property was leveraged to design an efficient solution. The Segment Tree consists of the following components:

- **Key:** The value stored at the node.
- **Lazy:** The lazy value for the node, which starts as the key value.
- **Range:** The range of applicability for the current node.
- **Left:** The left child node (optional).
- **Right:** The right child node (optional).

The construction of the tree requires a linear time complexity of O(n), where `n` is the size of the input array. Each query, whether it's an update or a maximum query, requires O(logn) due to the segment tree properties. Therefore, the overall complexity of the problem, considering `m` queries, is: O(n+m·logn)

Where:

- n is the size of the initial array.
- m is the number of queries.

1.3 IsThere Problem

For this problem, an array-based approach was implemented using a Segment Tree. The tree is initialized with zeros, and each segment is treated as an update of 1 within its range. After all updates, each node will store the maximum number of overlapping segments for its range. The structure of each node in the segment tree includes:

- **Key:** The value stored at the node.
- **Lazy:** The lazy value, initially set to zero.

Each query, whether it's an update or an `IsThere()` query, operates with a time complexity of O(logn). Given `n` segments and `m` queries, the overall time complexity is: O((n+m)·logn)

Where:

- n is the number of segments.
- m is the number of `IsThere()` queries.