Segment Tree

1. Introduction

A Segment Tree (ST) is a binary tree that is build on top of an (usually integer) array so that we can solve the Range Min/Max/Sum Query as well as any Range Update Query of this array in O(log **N**) time instead of the naive O(**N**) time. Given an array A of **N** (usually integer) elements, we can build the corresponding RMinQ/RMaxQ/RSumQ Segment Tree in O(**N**) time.

2. Modes

To toggle between the RMinQ/RMaxQ/RSumQ Segment Tree, select the respective header.

3. Visualisation

View the visualisation of Segment Tree here!
The top side shows the Segment Tree structure where each vertex shows the Min/Max/Sum value of the corresponding range (red colored with this format [L,R]).
The bottom row shows the original array A content (yellow colored) from which the Segment Tree structure is built.
Vertices that are lazily updated will have this purple ring highlight.
Each leaf vertex in the Segment Tree corresponds to an individual index in the corresponding array A.

4. Operations

There are three basic operations that are available in Segment Tree data structure visualization (for all 3 modes: RMinQ/RMaxQ/RSumQ):
1. You can create RMinQ/RMaxQ/RSumQ Segment Tree from either a user-defined array of integers (maximum of 16 two-digits integer), or let the system provide a small random integer array or a small random but sorted integer array.
2. You can do RMinQ/RMaxQ/RSumQ by specifying a left (L) and a right (R) indices.
3. You can do Range Update by specifying a left (L) index, a right (R) index, and a new VALUE for this range [L,R]. We employ lazy update strategy for fast performance.

5. Implementation

Unfortunately, this data structure is not yet available in C++ STL, Java API, Python or OCaml Standard Library as of 2020. Therefore, we have to write our own implementation.

Please look at the following C++/Java/Python/OCaml implementations of this Segment Tree data structure in Object-Oriented Programming (OOP) fashion:
segmenttree_ds.cpp
segmenttree_ds.java
segmenttree_ds.py
segmenttree_ds.ml

Again, you are free to customize this custom library implementation to suit your needs.