
SEGMENT TREE
Basics

Let's start with a brief explanation of segment trees. They are used when we have an array, perform some changes and queries on continuous segments. In the first example we'll consider 2 operations:

1. modify one element in the array;
2. find the sum of elements on some segment. .

5.

<table>
<tr><td colspan="16">1: [0, 16)</td></tr>
<tr><td colspan="8">2: [0, 8)</td><td colspan="8">3: [8, 16)</td></tr>
<tr><td colspan="4">4: [0, 4)</td><td colspan="4">5: [4, 8)</td><td colspan="4">6: [8, 12)</td><td colspan="4">7: [12, 16)</td></tr>
<tr><td colspan="2">8:
[0, 2)</td><td colspan="2">9:
[2, 4)</td><td colspan="2">10:
[4, 6)</td><td colspan="2">11:
[6, 8)</td><td colspan="2">12:
[8, 10)</td><td colspan="2">13:
[10, 12)</td><td colspan="2">14:
[12, 14)</td><td colspan="2">15:
[14, 16)</td></tr>
<tr><td>16:
0</td><td>17:
1</td><td>18:
2</td><td>19:
3</td><td>20:
4</td><td>21:
5</td><td>22:
6</td><td>23:
7</td><td>24:
8</td><td>25:
9</td><td>26:
10</td><td>27:
11</td><td>28:
12</td><td>29:
13</td><td>30:
14</td><td>31:
15</td></tr>
</table>

Perfect Binary Tree

- I like to visualize a segment tree in the following way Notation is node_index: corresponding segment (left border included, right excluded).
- At the bottom row we have our array (0-indexed), the leaves of the tree. For now suppose it's length is a power of 2 (16 in the example), so we get perfect binary tree.

  When going up the tree we take pairs of nodes with indices $(2 * i, 2 * i + 1)$ and combine their values in their parent with index i.
- This way when we're asked to find a sum on interval $3,11$ we need to sum up only values in the nodes 19, 5, 12 and 26 (marked with bold), not all 8 values inside the interval. Let's jump directly to implementation (in C++) to see how it works: