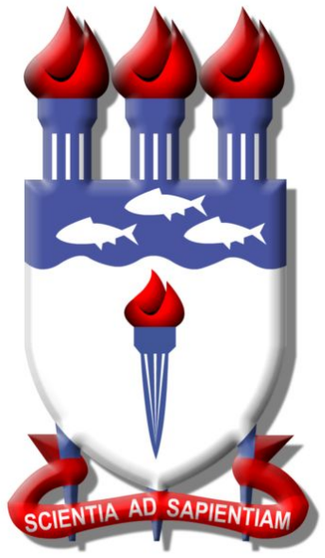


Segment Tree

Ana Ferreira e Kelly Bianca

https://github.com/AnaFerreira015/little-huff

Problem

- Imagine that you have ***N*** integer values and, from time to time, are interested in finding the sum or looking in a range of these values

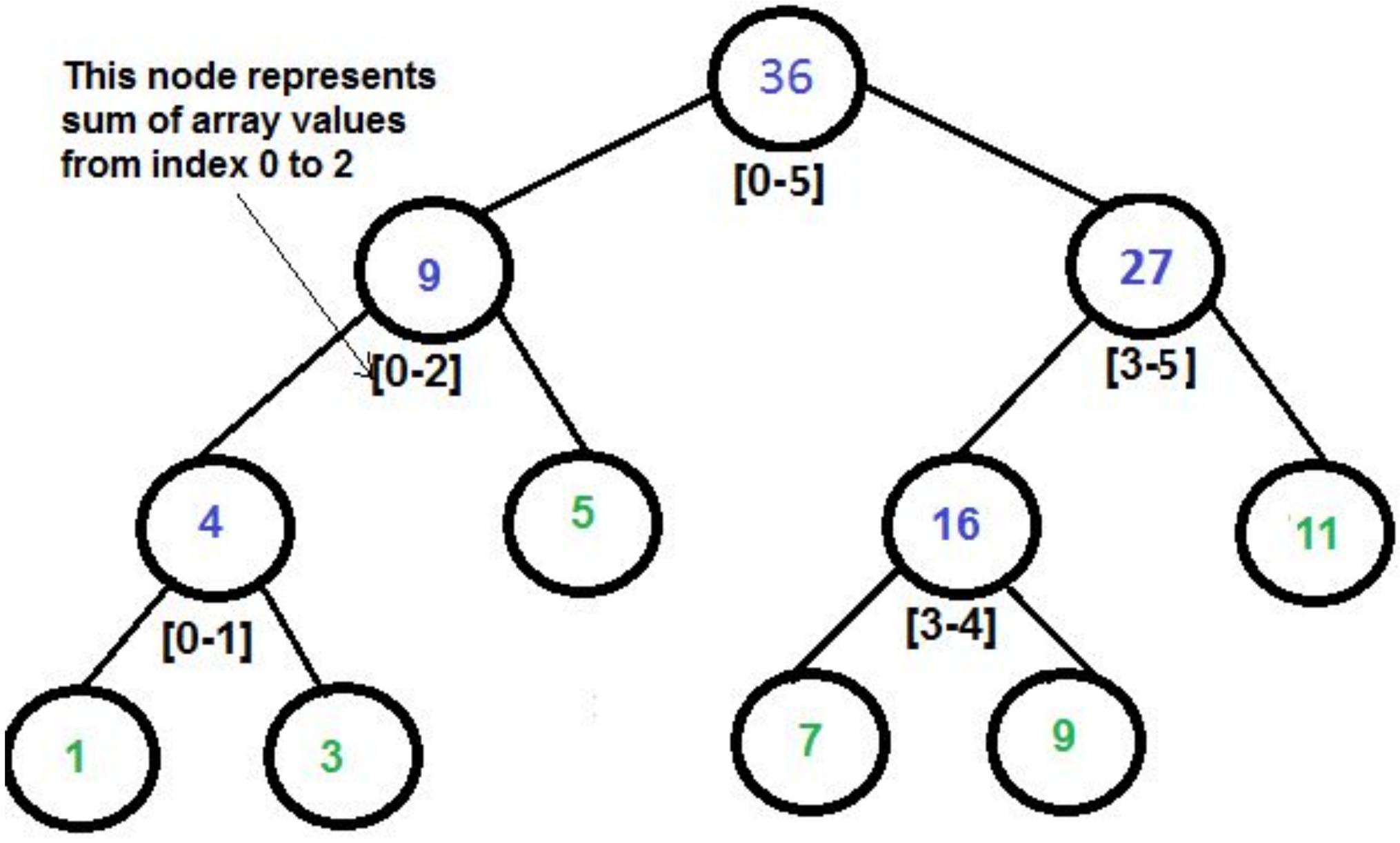
Segment Tree

- Segment tree is a basically a binary tree used for storing the intervals or segments
- Each node in the segment tree represents an interval. Considering an array ***A*** of size ***N*** and a corresponding segment tree ***T***:
  i. The root of ***T*** will represent the whole array ***A***[0 : N - 1]
  ii. Each leaf in the segment tree ***T*** will represent a single element ***A***[i]
  iii. The internal nodes in the segment tree T represents the union of elementary intervals ***A***[i : j]

Segment Tree

- Segment Tree can be used to solve range min/max and sum queries and range update queries in O(log n) time.

- is one of the most widely used data structures in competitive programming because of its efficiency and versatility.



Segment Tree for input array {1, 3, 5, 7, 9, 11}

Código

```
#include <stdio.h>
#include <stdlib.h>
typedef long long int lli;
// lli segTree[n * 4 + 2];

void build(lli segTree[], lli vec[], int n)
{
    int i;
    for (i = 0; i < n; i++)
    {
        segTree[n + i] = vec[i];
    }
    for (i = n - 1; i > 0; --i)
    {
        segTree[i] = segTree[i * 2] + segTree[i * 2 + 1];
    }
}
```

Código

```
lli query(lli segTree[], lli left, lli right, lli indexSeg, lli leftSeg, lli
rightSeg)
{
    if (leftSeg > right || rightSeg < left)
    {
         return 0;
    }
    if (leftSeg >= left && rightSeg <= right)
    {
         return segTree[indexSeg];
    }
    lli middle = (leftSeg + rightSeg) / 2;
    return query(segTree, left, right, indexSeg * 2, leftSeg, middle) +
query(segTree, left, right, index * 2 + 1, middle + 1, right);
}
```