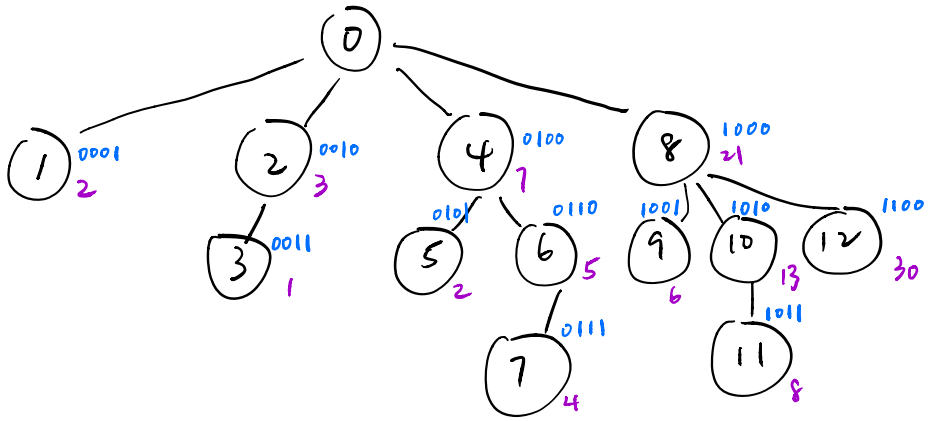


- BIT[y] is the parent of BIT[x] if y can be obtained by removing last set bit of x.

4 is the parent of 5 and 6.

- Each node BIT[x] stores the sum of elements between [y, x)
  - y: inclusive of parent
  - x: exclusive of child.

- BIT is 1-indexed array.

	0	1	2	3	4	5	6	7	8	9	10	11
arr:		2	1	1	3	2	3	4	5	6	7	8	9
BIT:	0	1	2	3	4	5	6	7	8	9	10	11	12
		2	3	1	7	2	5	4	21	6	13	8	30

```
Binary Indexed Tree {
O(logn)   getSum(i) {}
O(logn)   update(index, val) {}
}
```

To construct BIT, simply loop through arr and update for each value.
So it takes O(n log n).

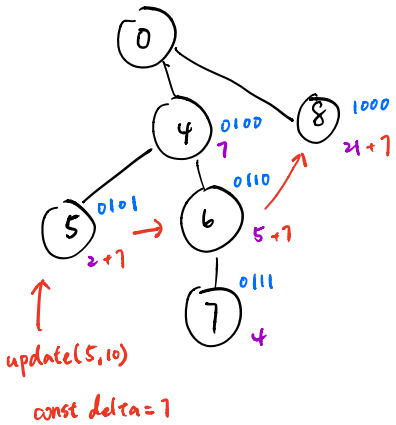
getSum(7): $7 = 2^2 + 2^1 + 2^0$

= range(0,4) + range(4,6) + range(6,7)

= BIT[4] + BIT[6] + BIT[7]

100 ← 110 ← 111

x => x - lastSetBit(x)



update(5, 10):

const delta = 10 - arr[5];

0101 BIT[5]
0110 BIT[6]
1000 BIT[8]

} => += delta.

x => x + lastSetBit(x)