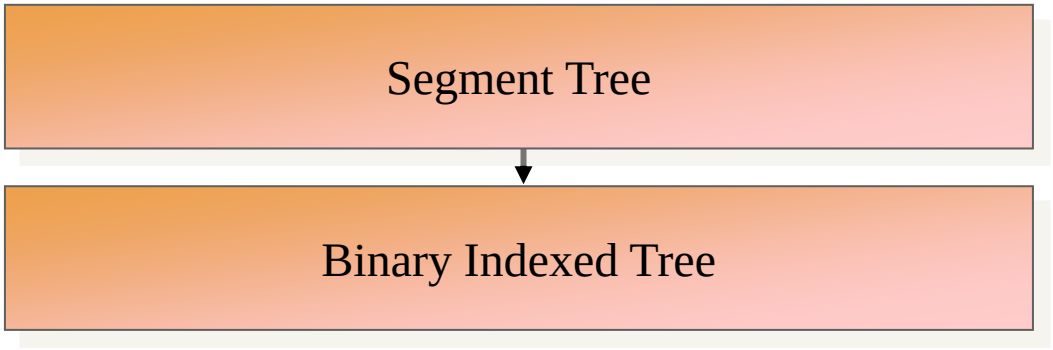
NCKU Programming Contest Training Course
2013/08/09

Pin-Chieh Huang (free999)

http://myweb.ncku.edu.tw/~p76014143/20130809.rar

Department of Computer Science and Information Engineering
National Cheng Kung University
Tainan, Taiwan

Outline



Segment Tree

Problem

求區間內數字最大

陣列長度 = N

Query 數 = Q

時間複雜度 = O (?)

3	8	9	2	5	6	7	9	1	4

Segment Tree

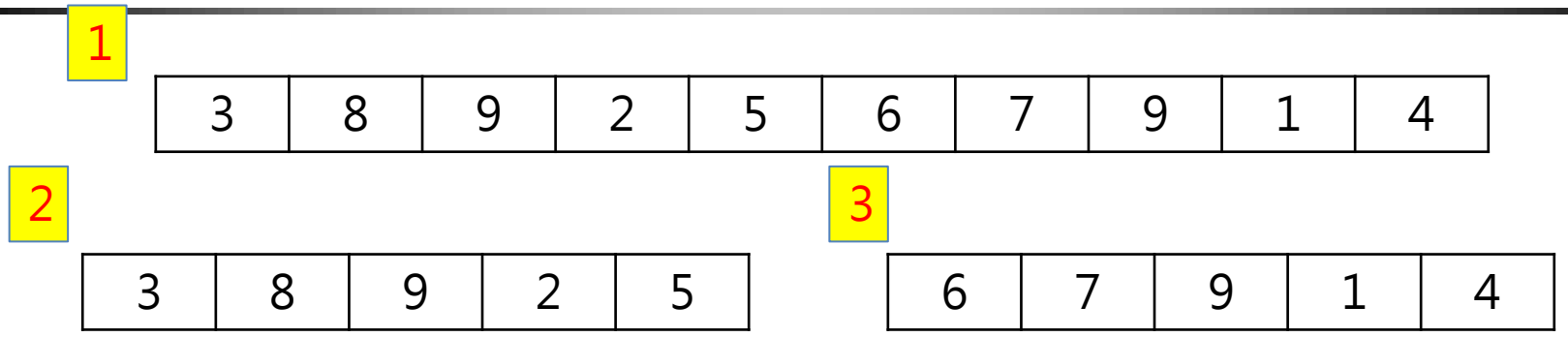
- Segment Tree
  - A tree-based data structure
  - Construct tree with O(N)
  - RMQ (range minimum/maximum query problem) in O(logN)

Segment Tree

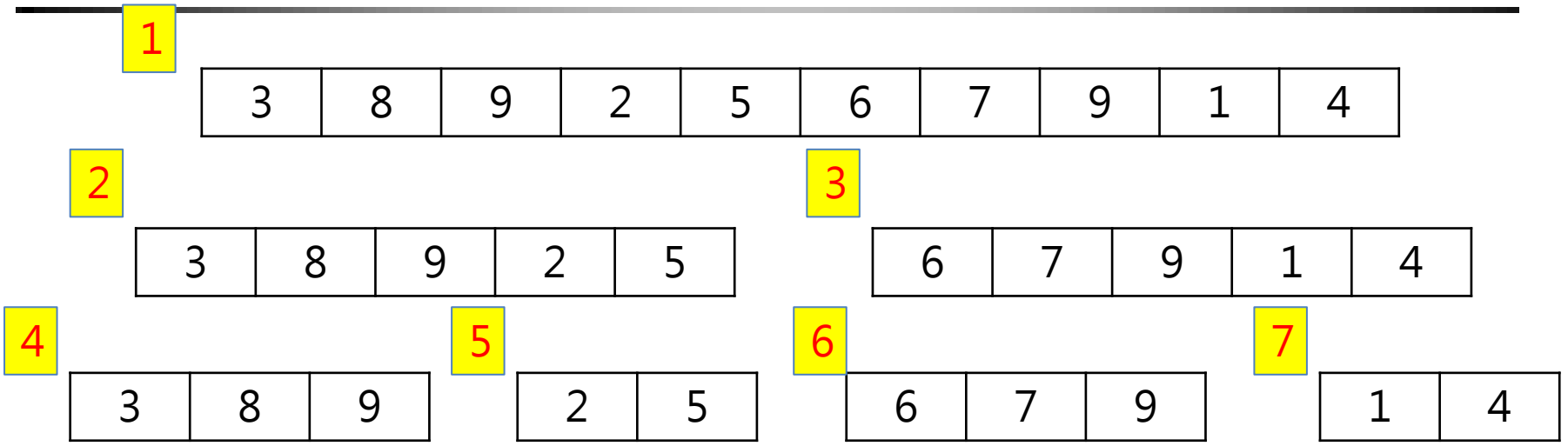
1

3	8	9	2	5	6	7	9	1	4

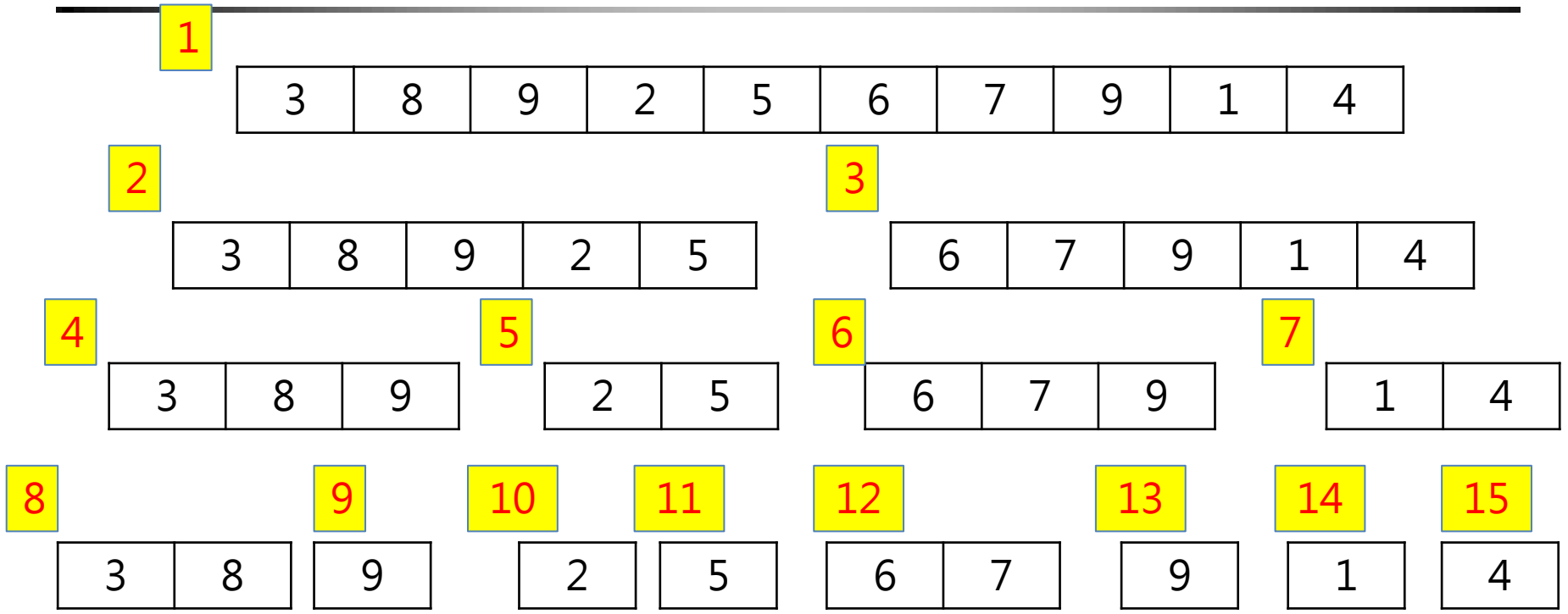
Segment Tree



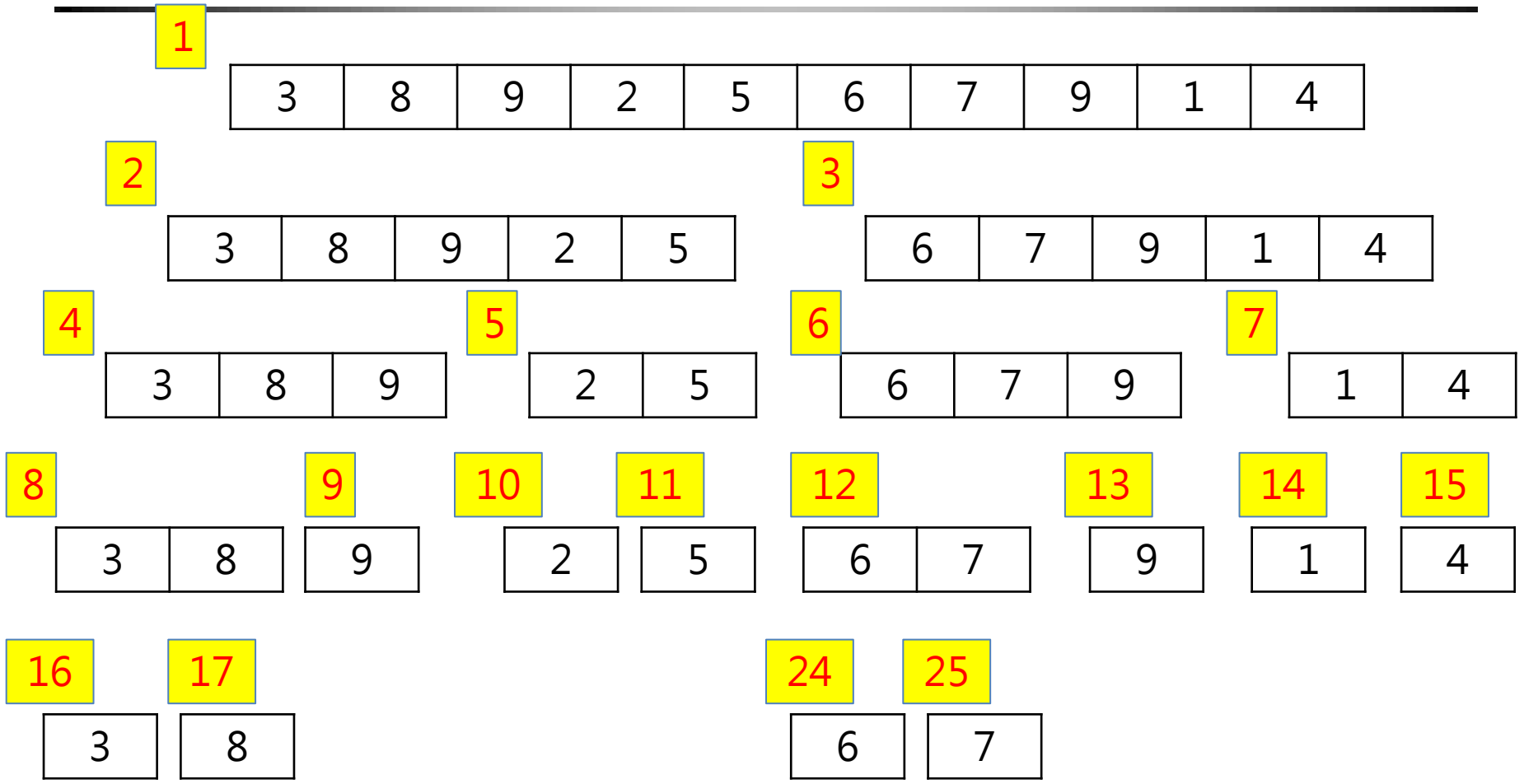
Segment Tree



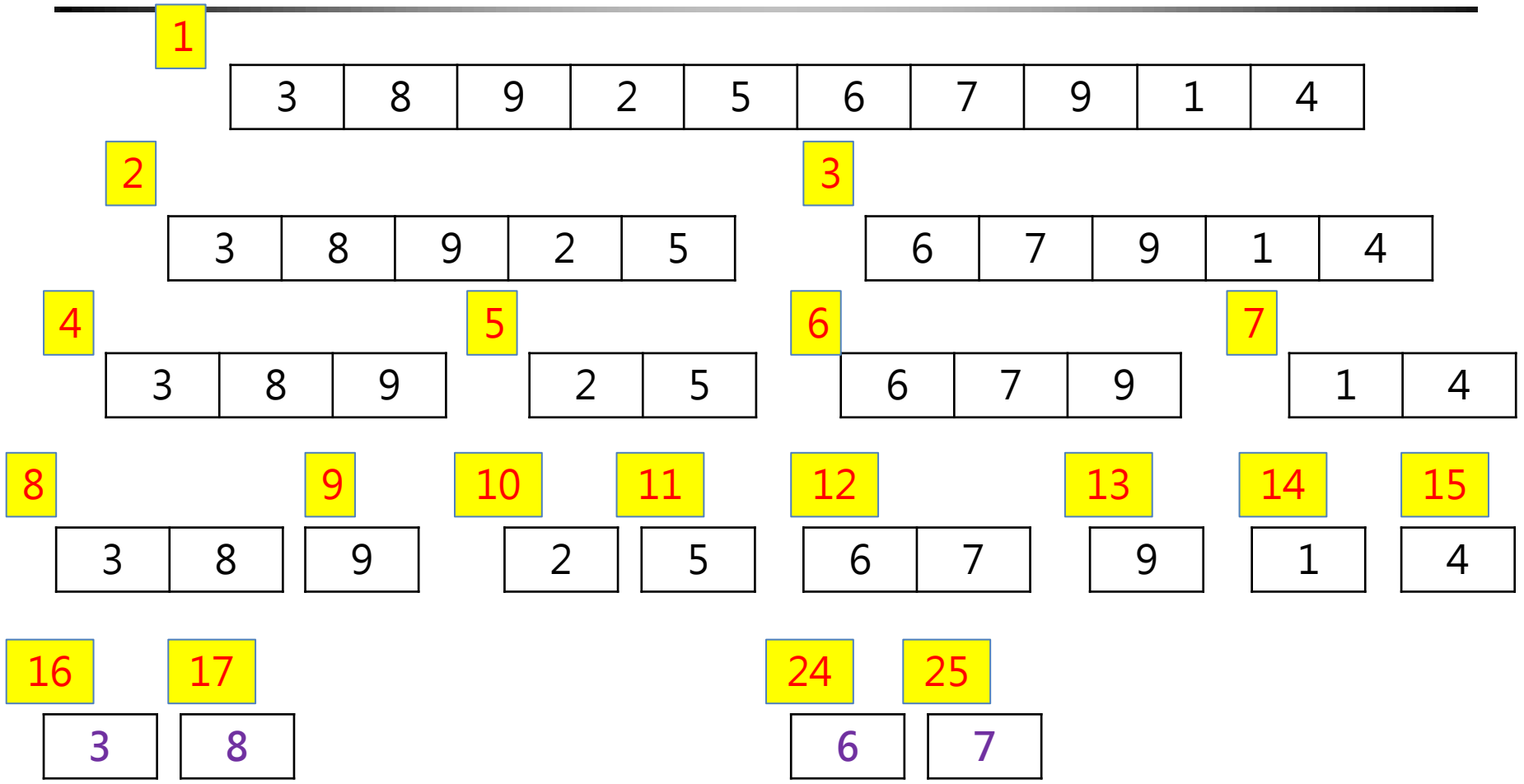
Segment Tree



Segment Tree

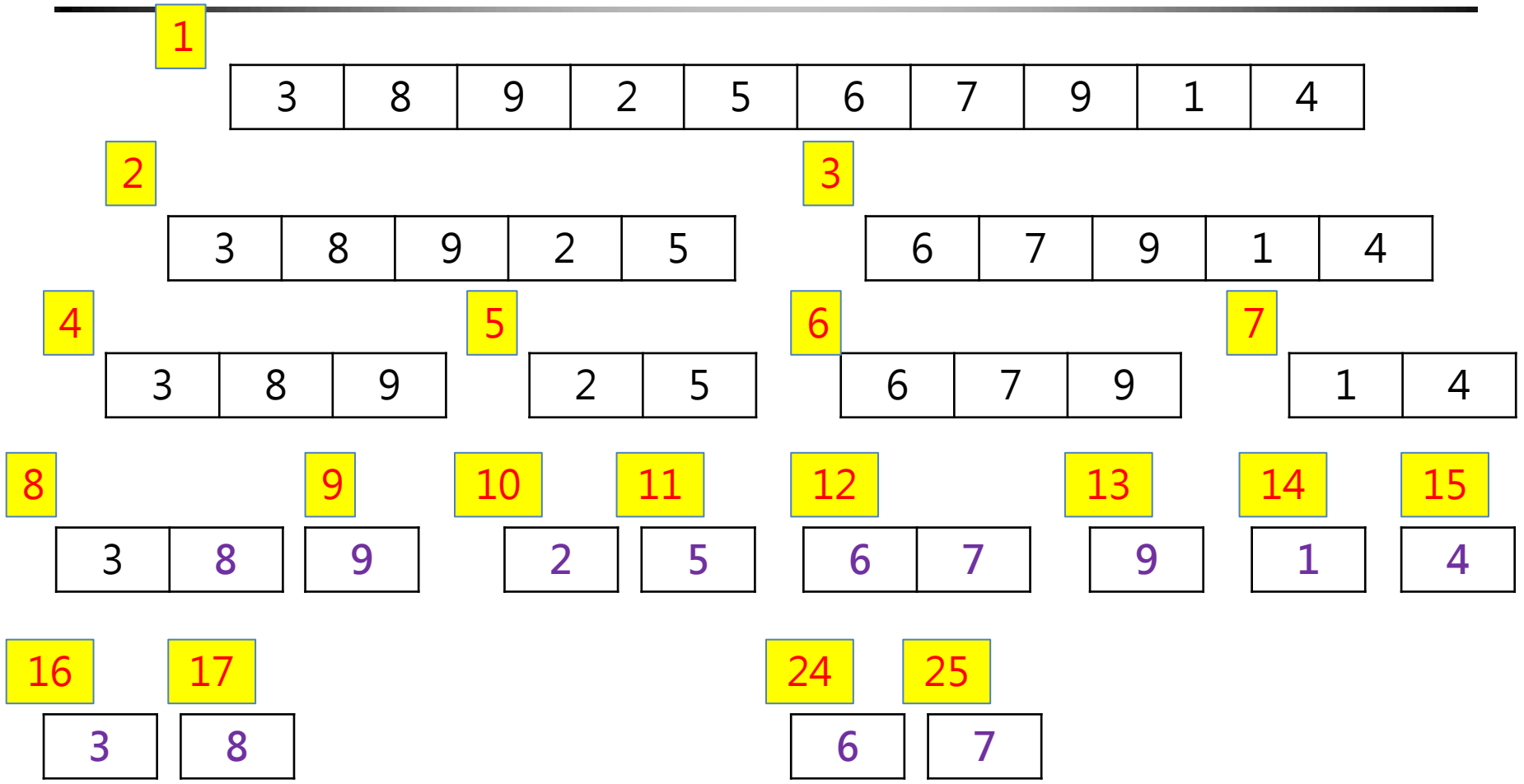


Segment Tree



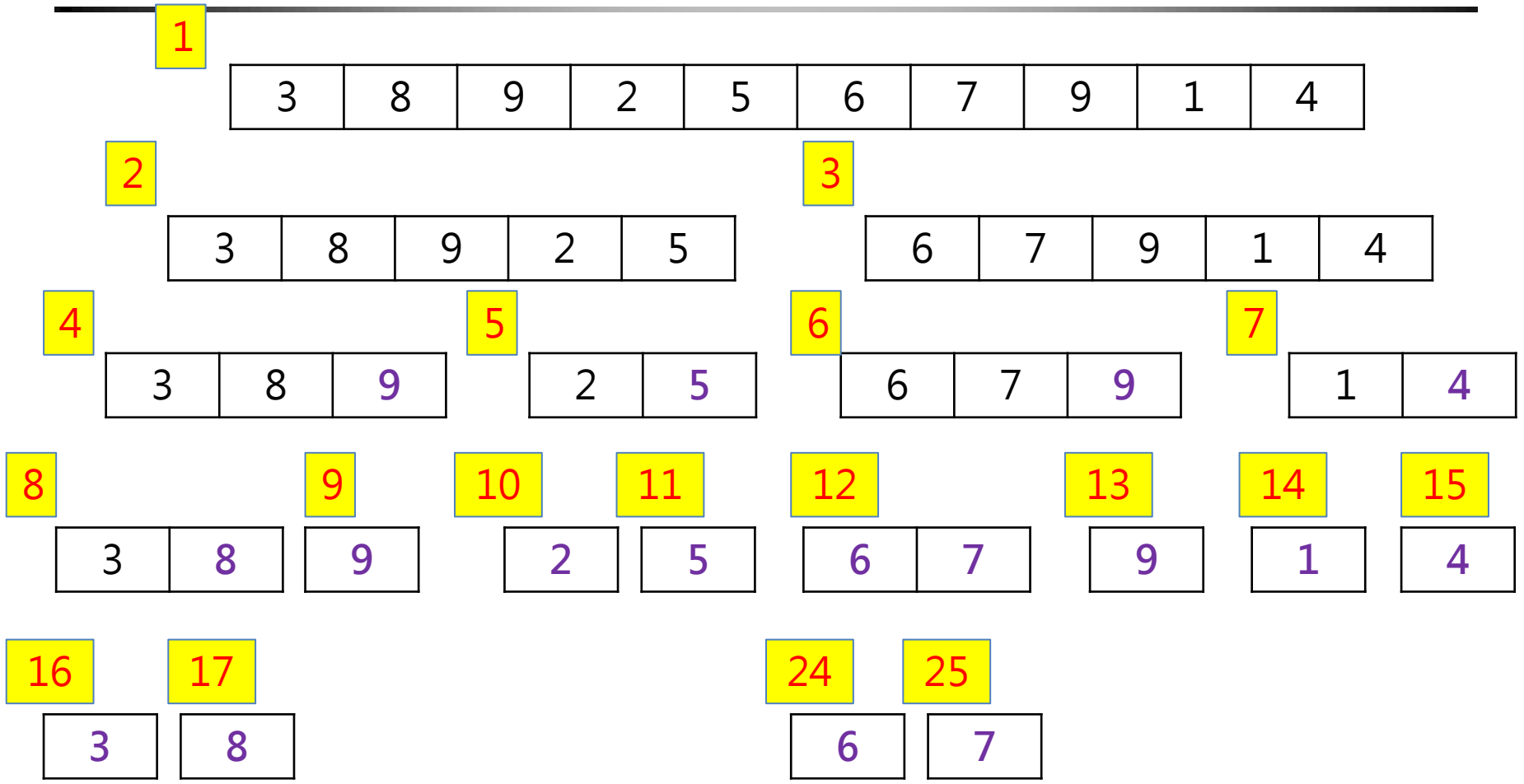
made by electron & free999

Segment Tree



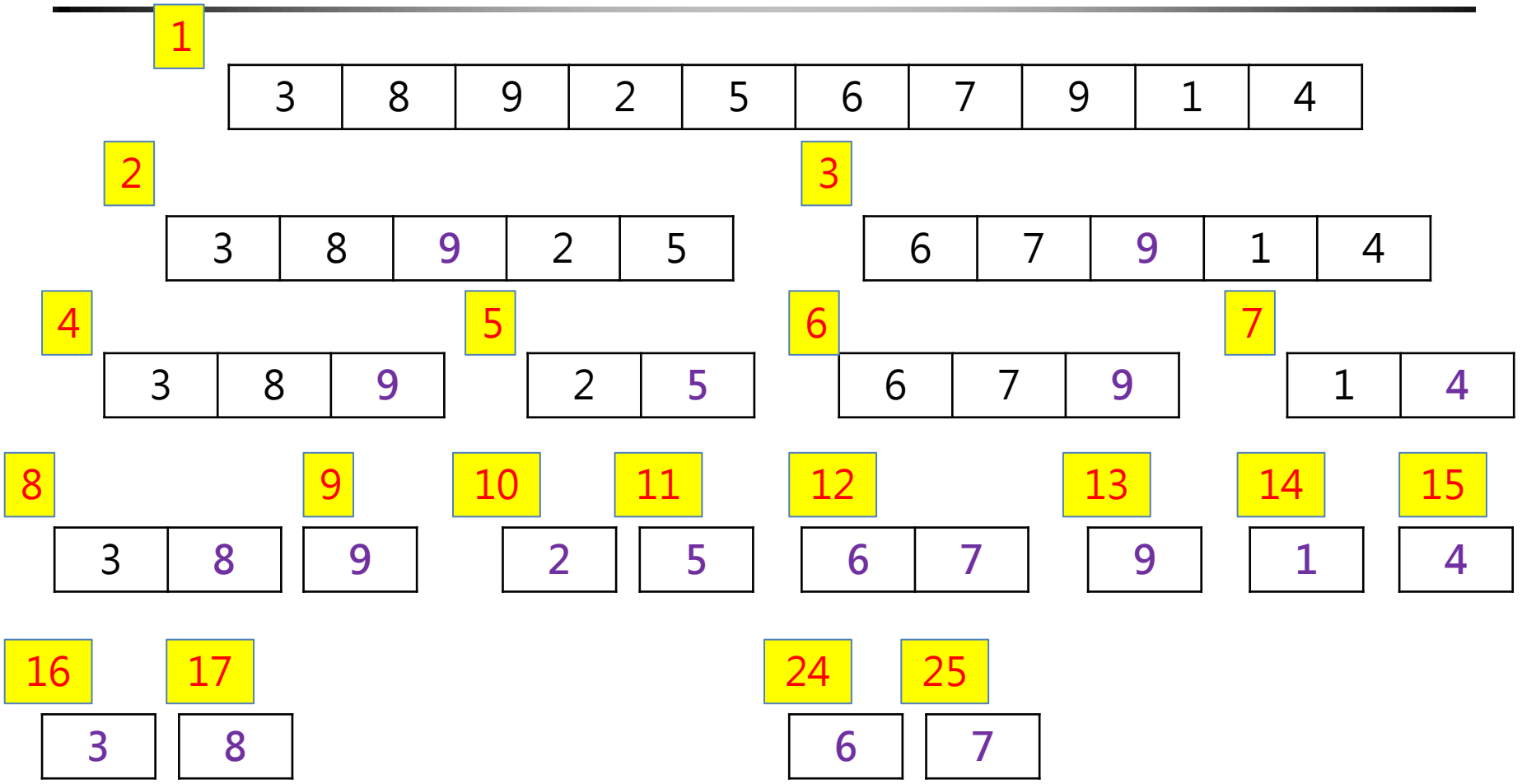
made by electron & free999

Segment Tree



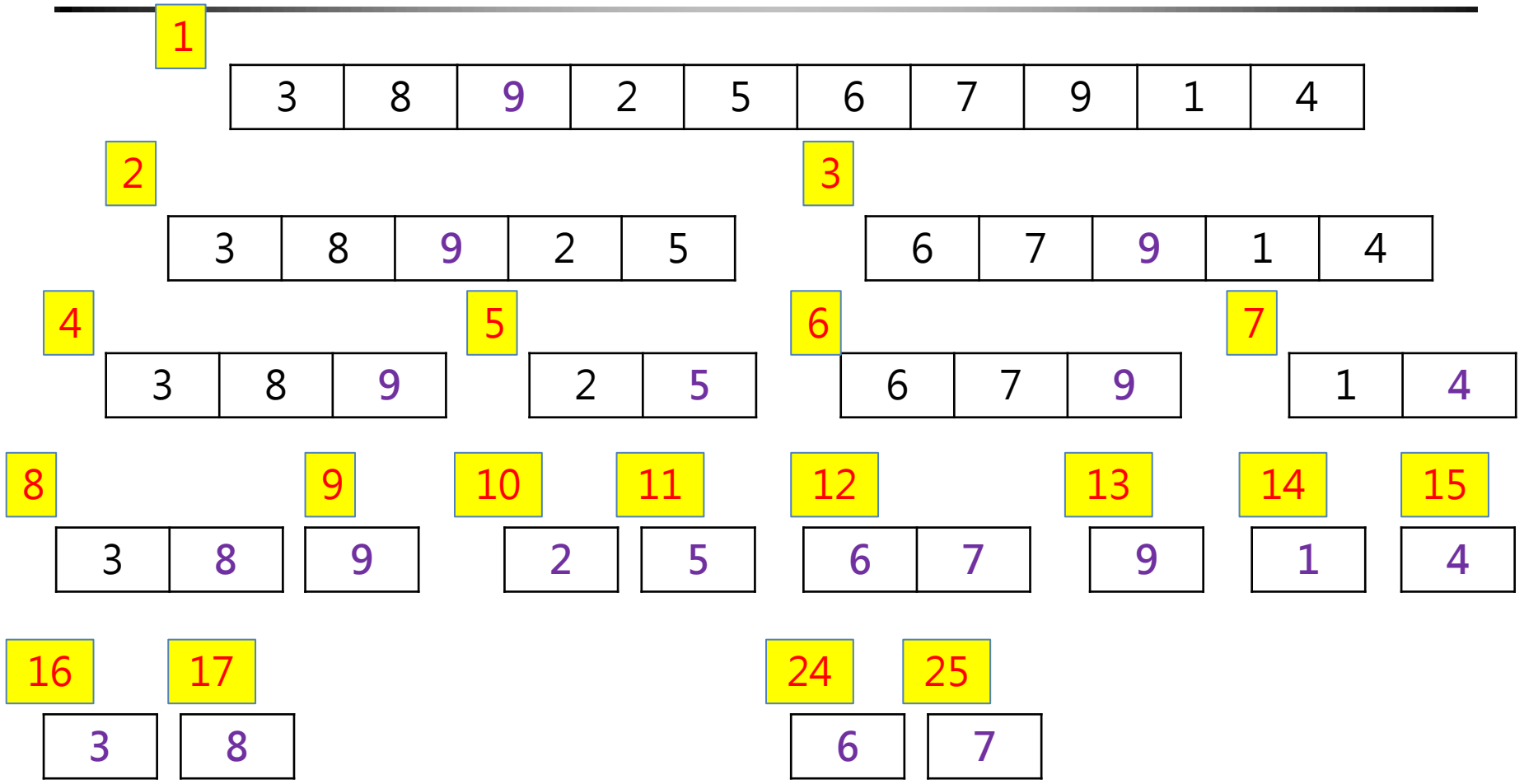
made by electron & free999

Segment Tree

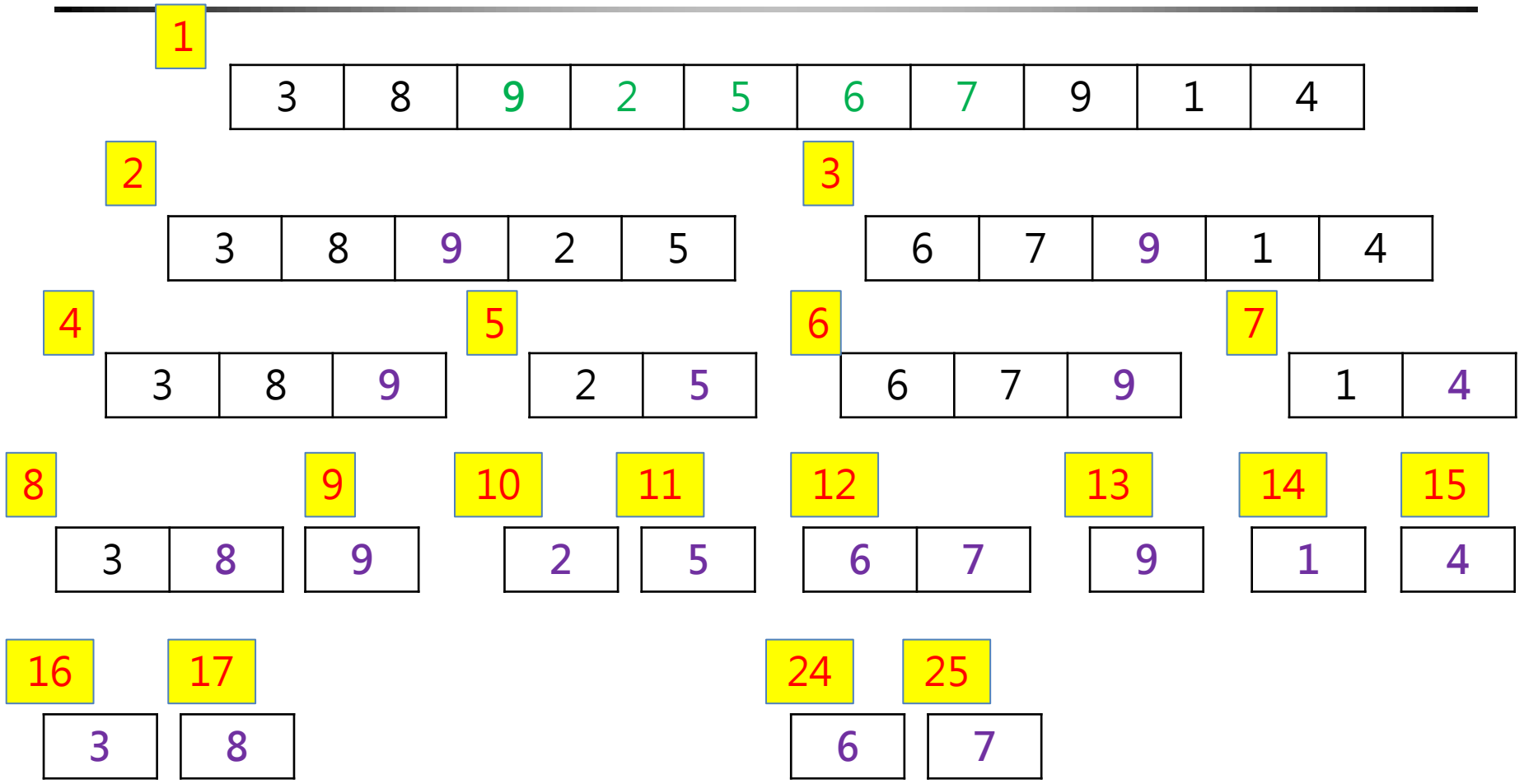


made by electron & free999

Segment Tree

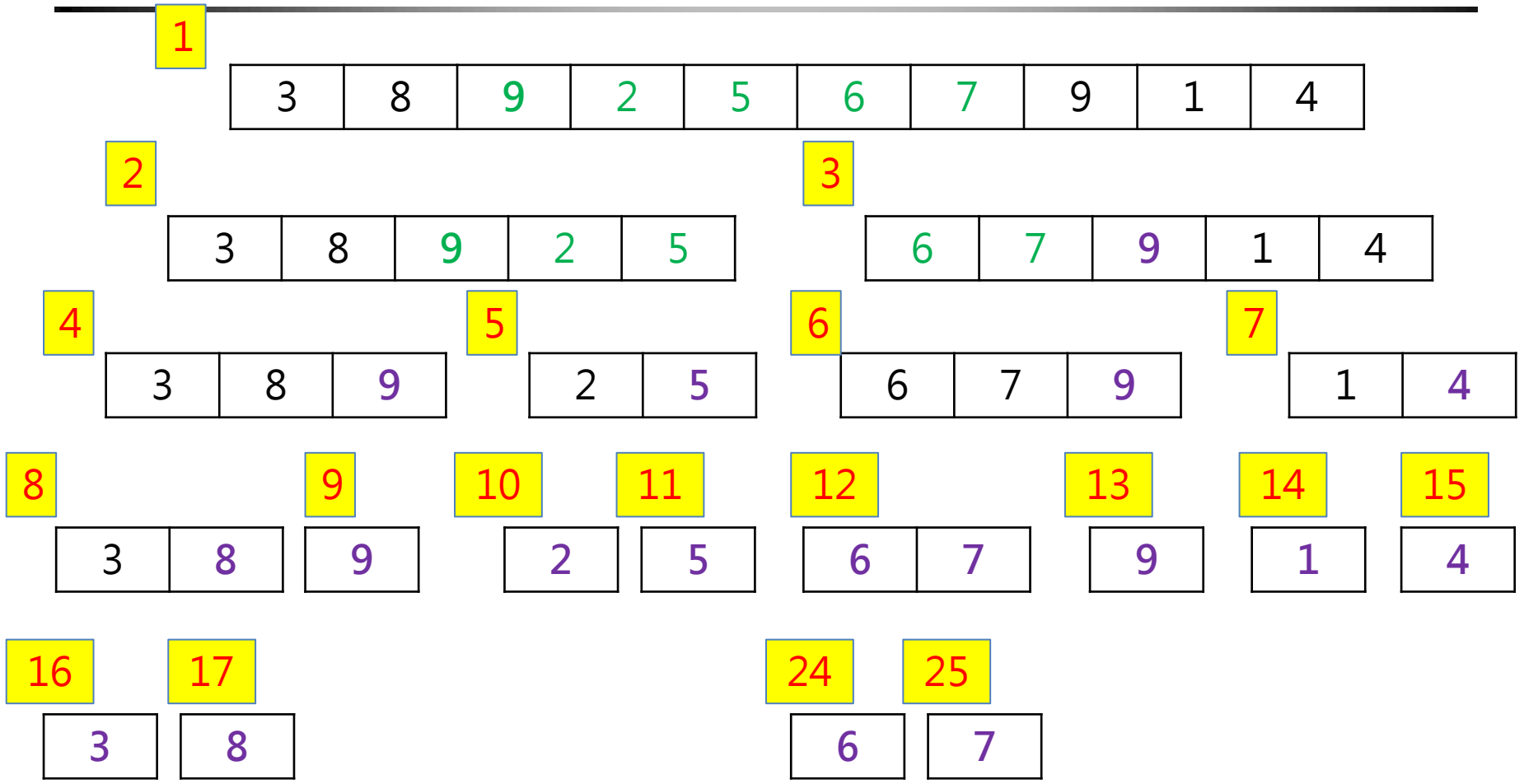


Segment Tree



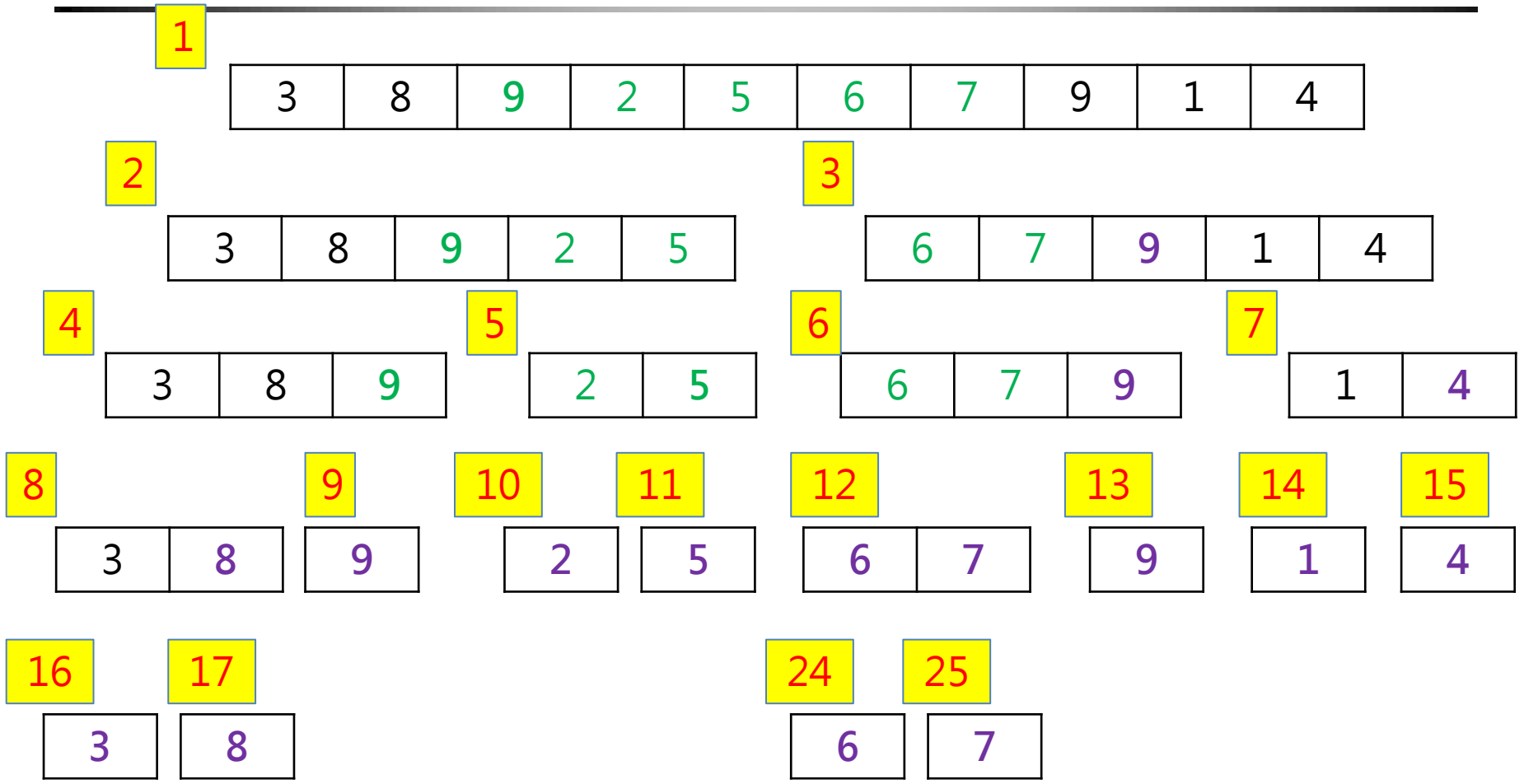
made by electron & free999

Segment Tree



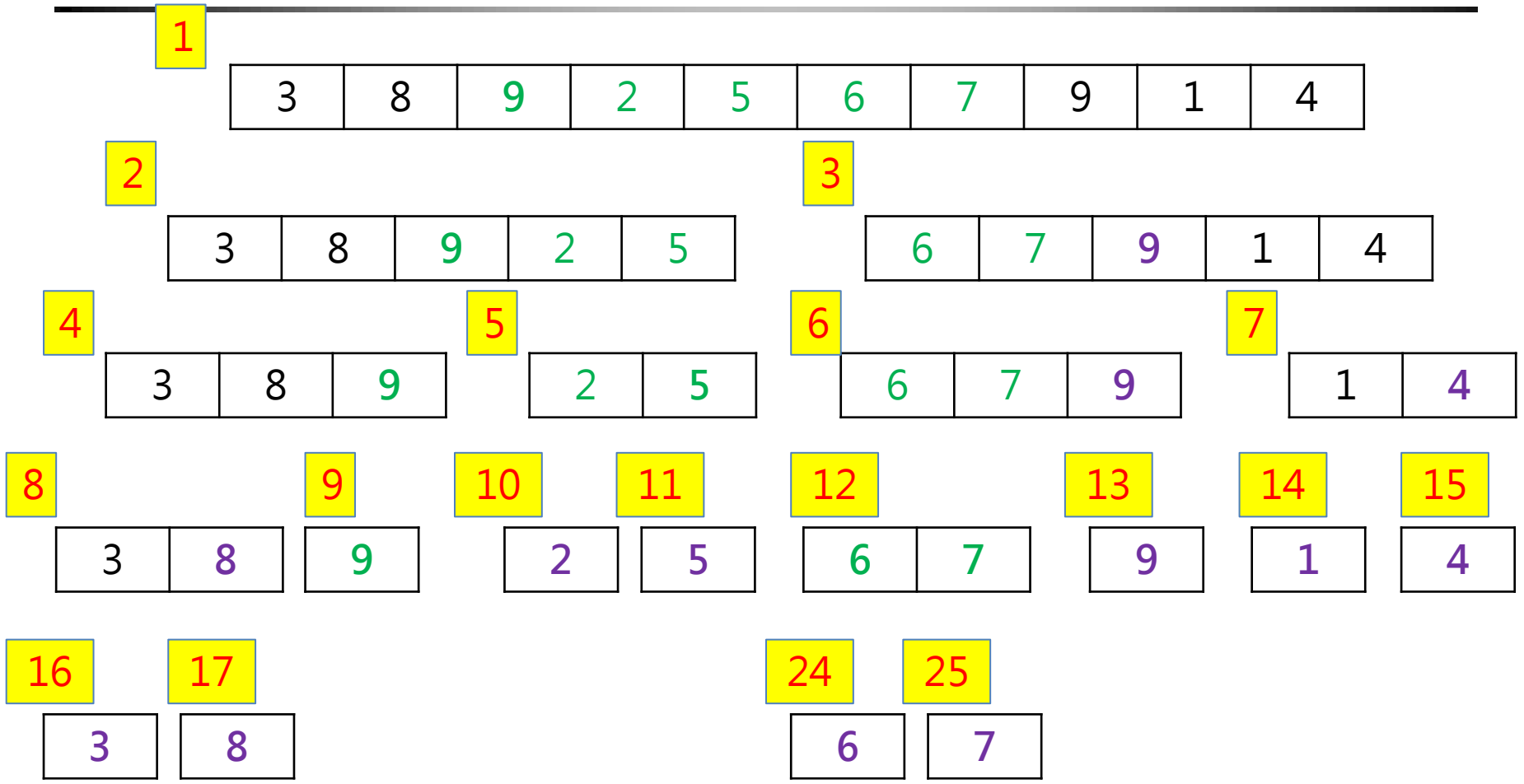
made by electron & free999

Segment Tree



made by electron & free999

Segment Tree



made by electron & free999

Example

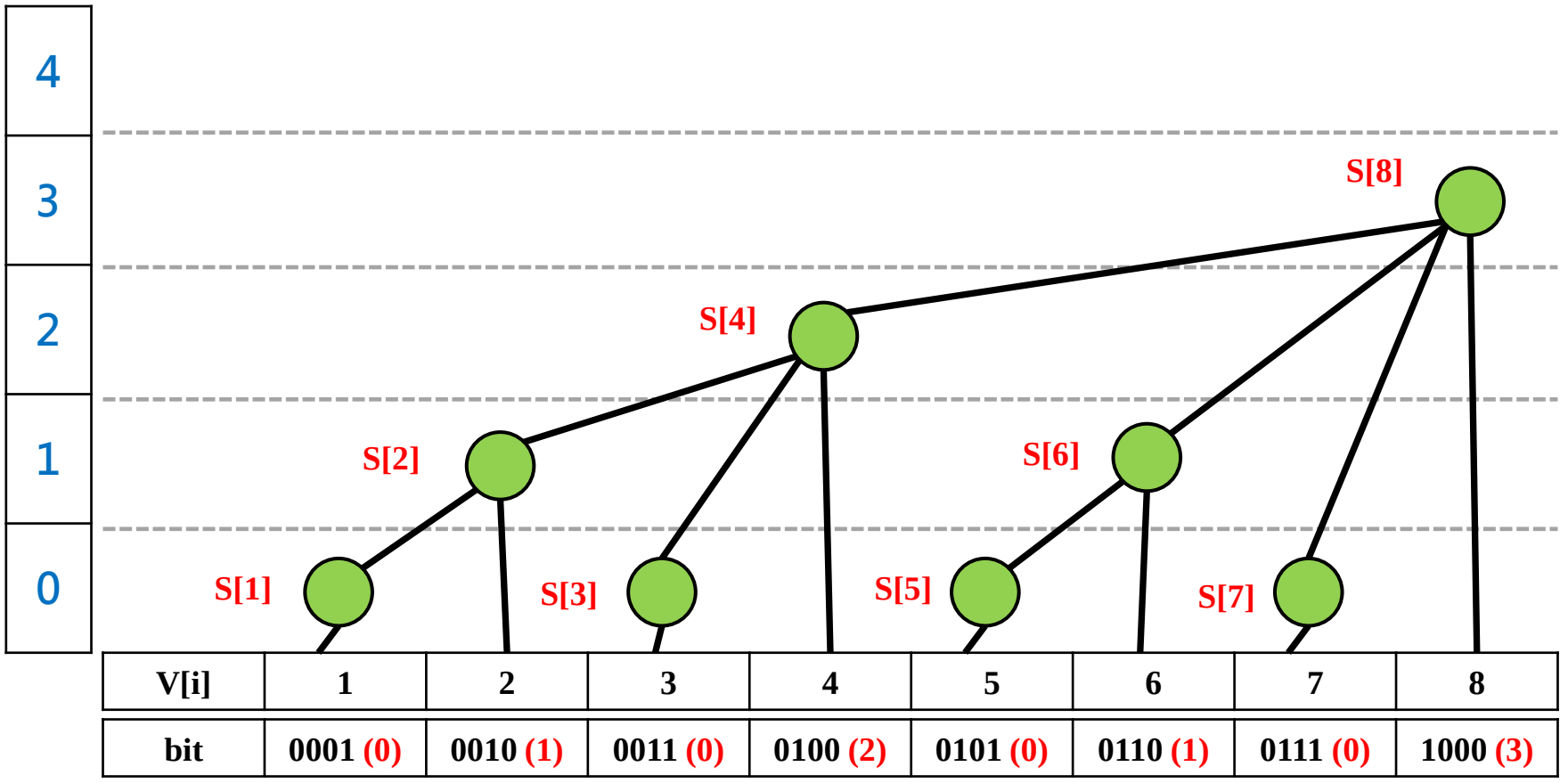
- POJ-3264

Binary Indexed Tree

- Operation
  - Increase/decrease the value of an element
  - Query the summation within the interval [i…j]
  - How to solve this problem efficiently?

- Naïve Solution
  - Each query requires O(N)
  - Total **O(QN)**…

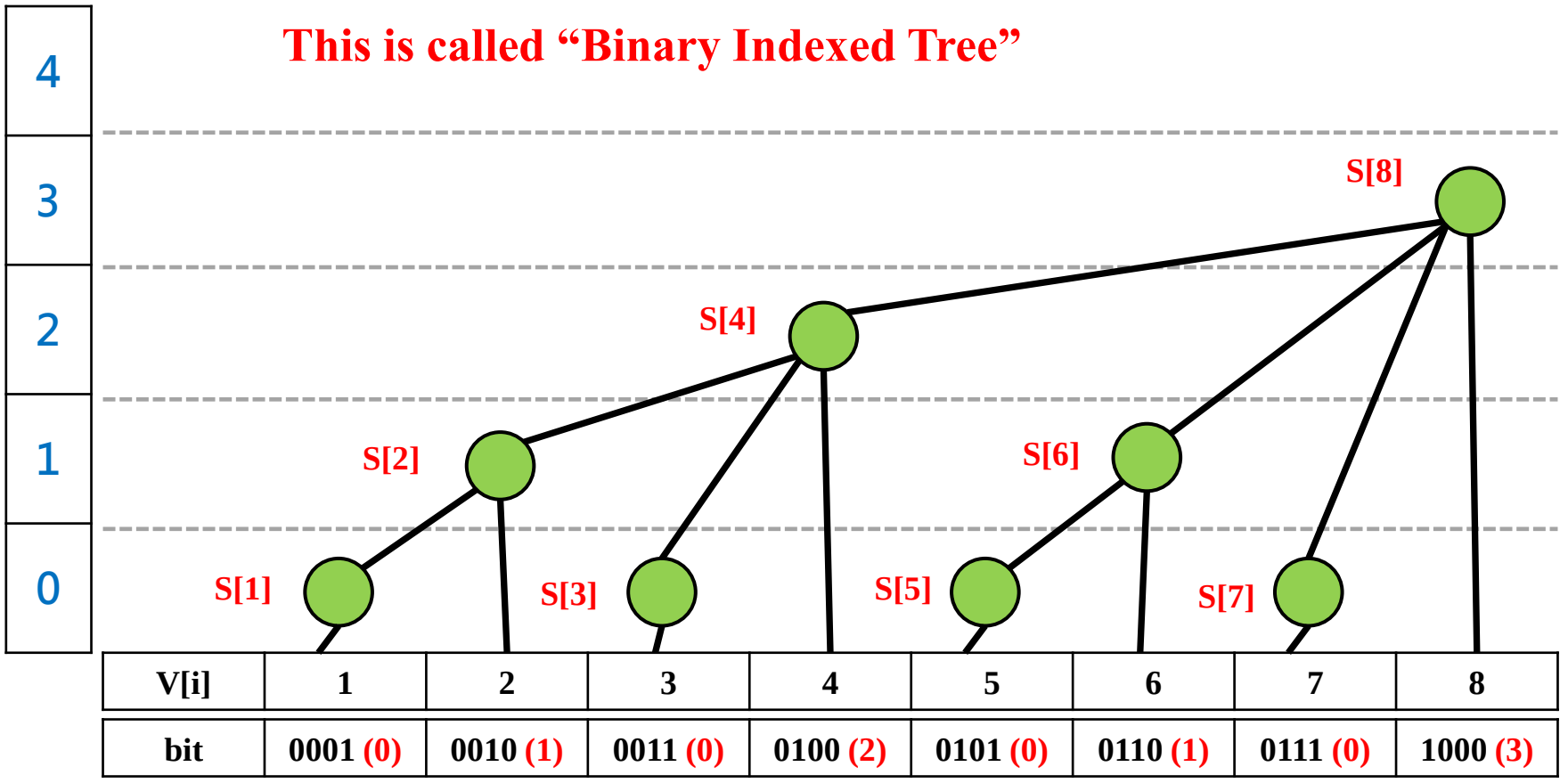
Binary Indexed Tree



V[i]	1	2	3	4	5	6	7	8
bit	0001 (0)	0010 (1)	0011 (0)	0100 (2)	0101 (0)	0110 (1)	0111 (0)	1000 (3)

made by electron & free999

Binary Indexed Tree



V[i]	1	2	3	4	5	6	7	8
bit	0001 (0)	0010 (1)	0011 (0)	0100 (2)	0101 (0)	0110 (1)	0111 (0)	1000 (3)

made by electron & free999

Binary Indexed Tree

s[1] = v[1]

s[2] = v[2] + **s[1]**

s[3] = v[3]

s[4] = v[4] + **s[3] + s[2]**

s[5] = v[5]

s[6] = v[6] + **s[5]**

s[7] = v[7]

s[8] = v[8] + **s[7] + s[6] + s[4]**

…

What's the regularity?

Binary Indexed Tree

Define:

```
int lowbit (int in)
{
        return in&(-in);
}
```

ex:

lowbit(1) = 1

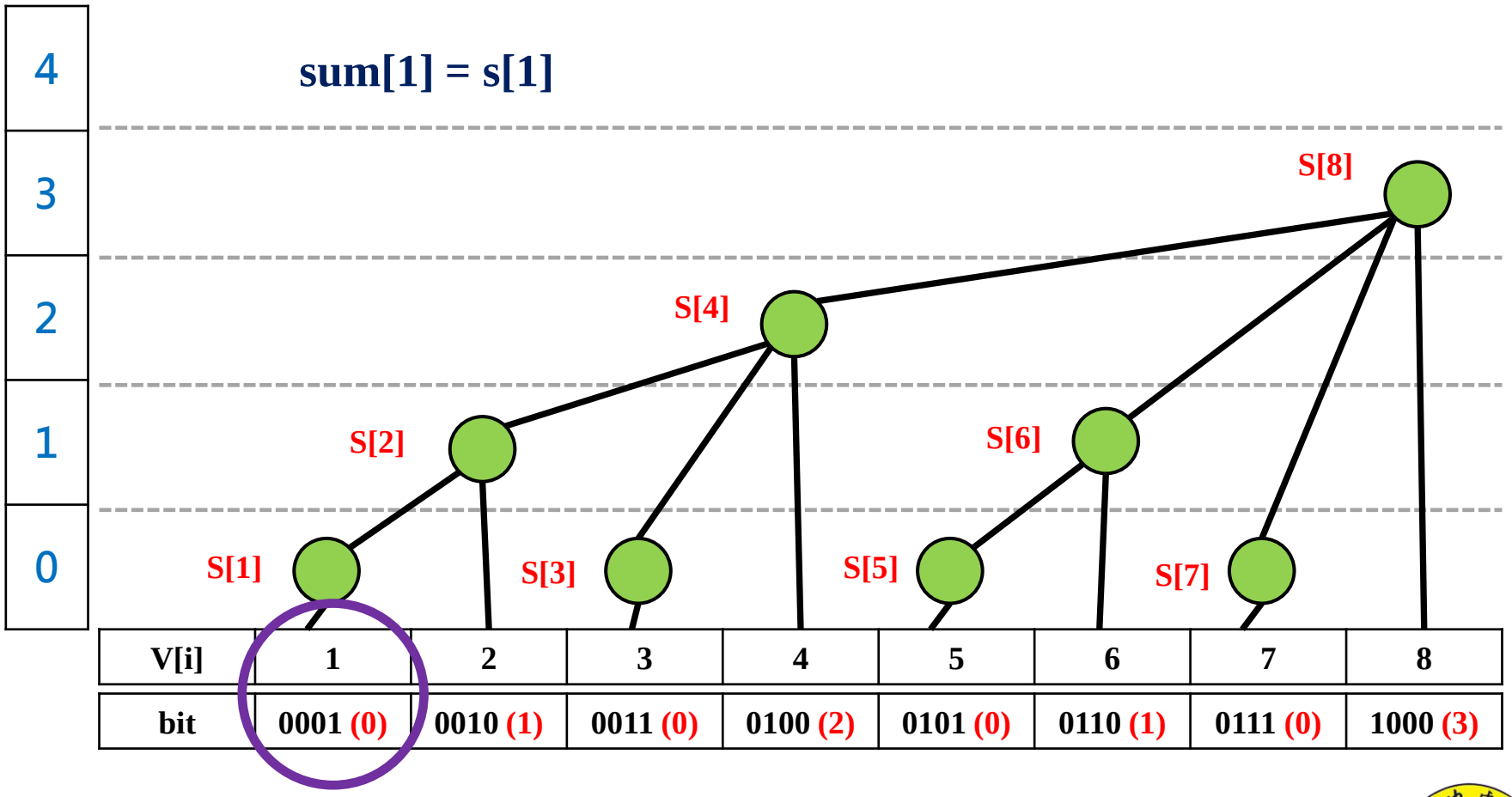
lowbit(2) = 2

lowbit(3) = 1

lowbit(4) = 4

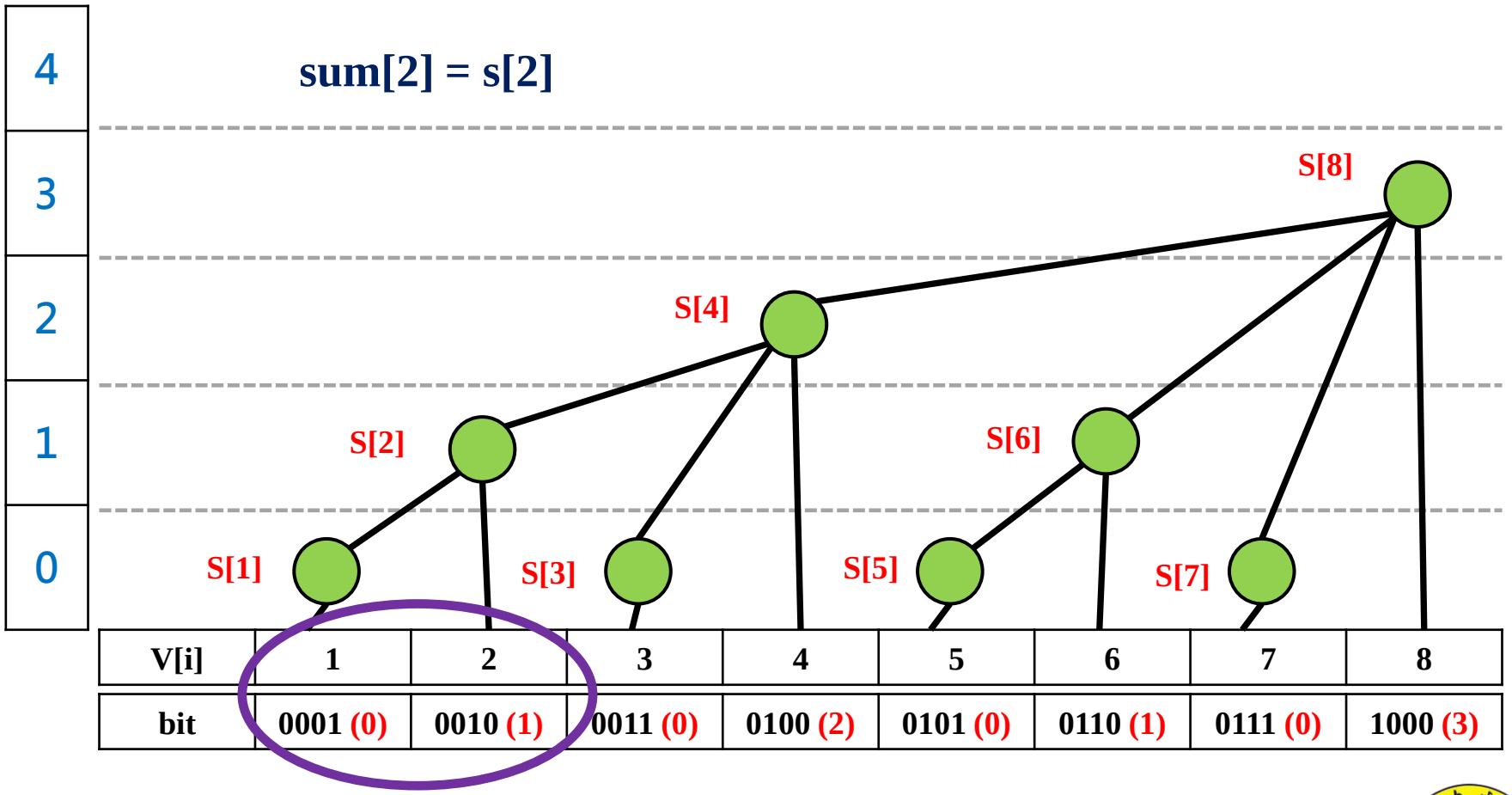
…

Binary Indexed Tree



made by electron & free999

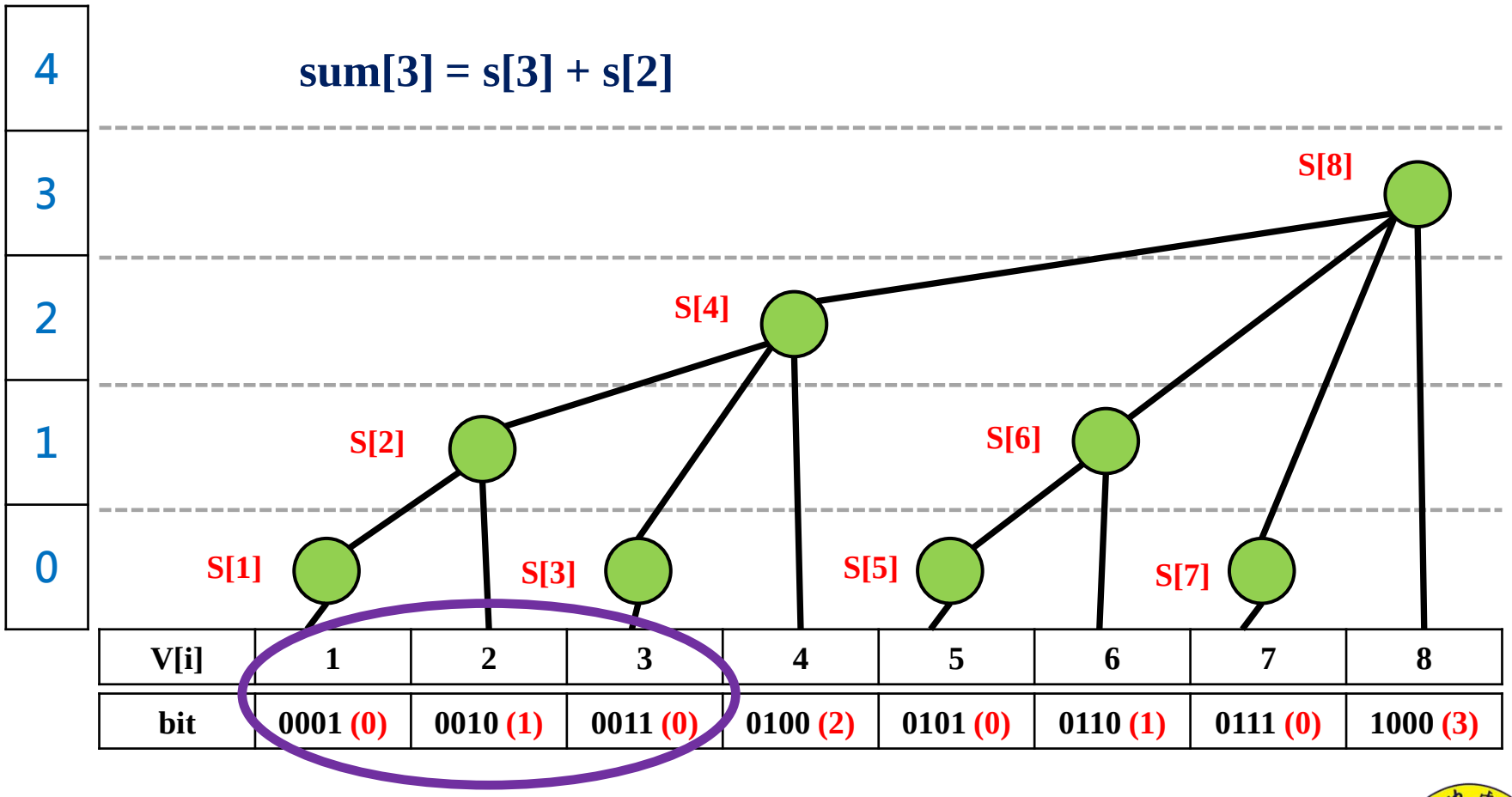
Binary Indexed Tree



V[i]	1	2	3	4	5	6	7	8
bit	0001 (0)	0010 (1)	0011 (0)	0100 (2)	0101 (0)	0110 (1)	0111 (0)	1000 (3)

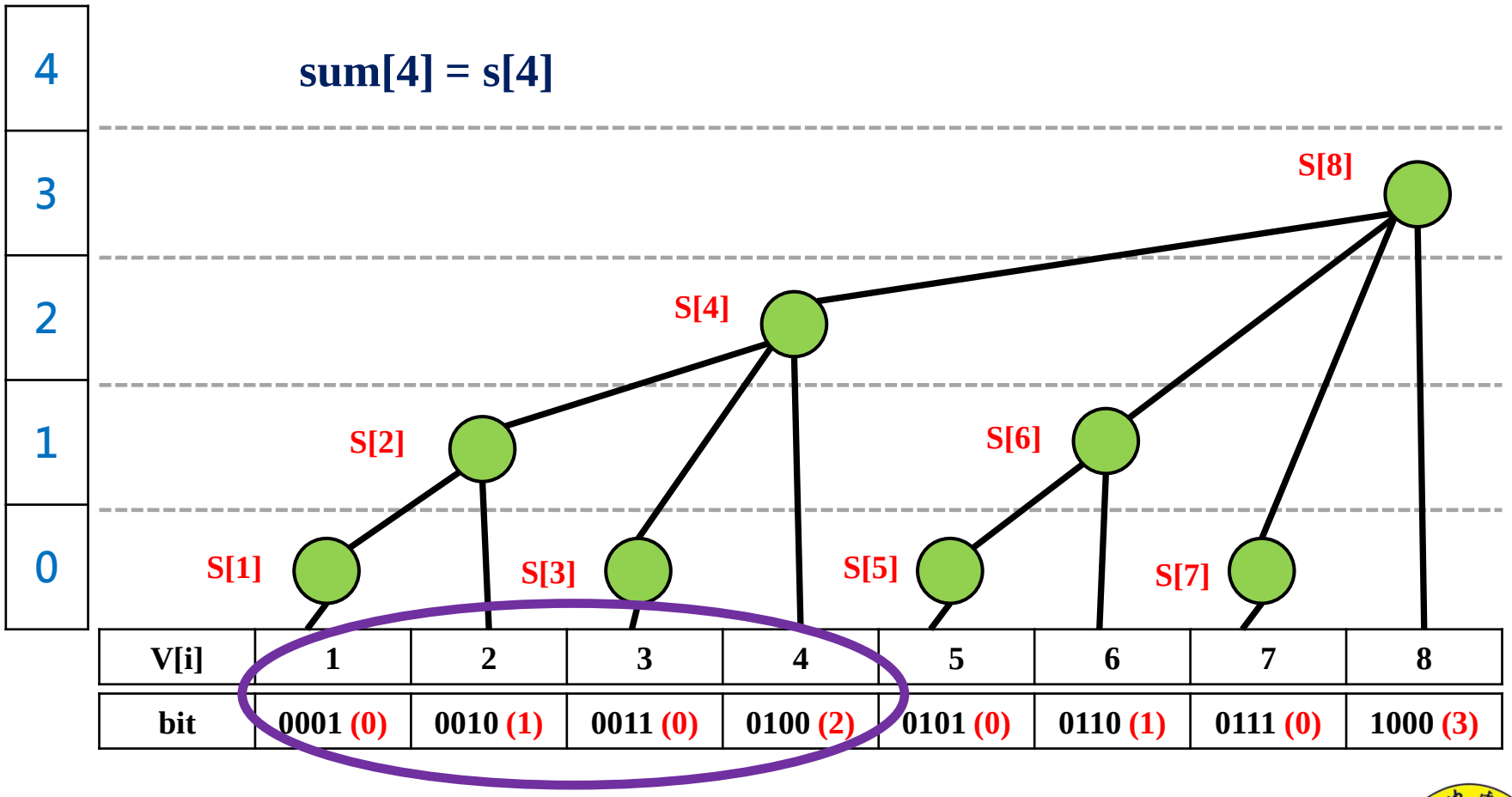
made by electron & free999

Binary Indexed Tree



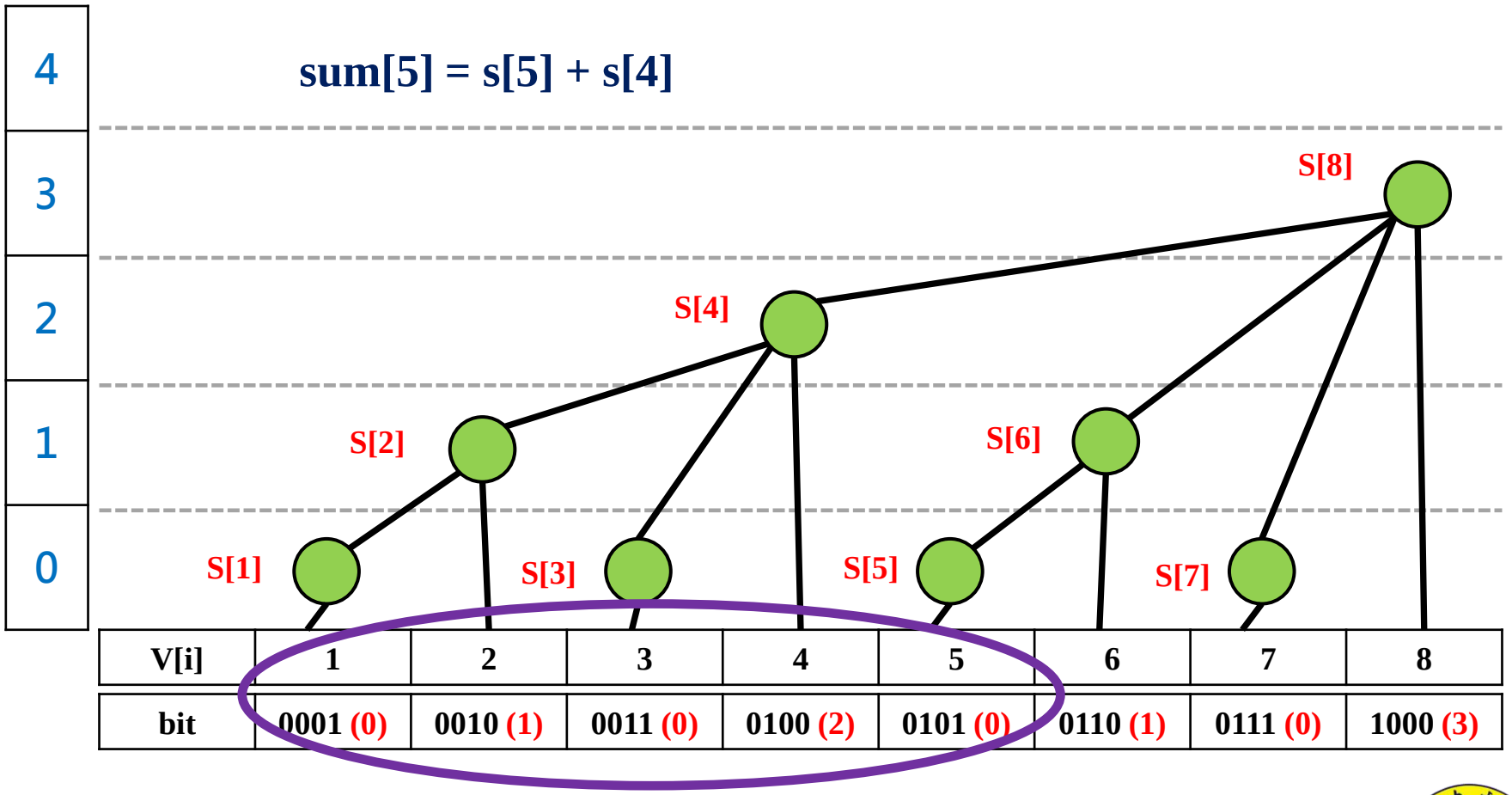
made by electron & free999

Binary Indexed Tree



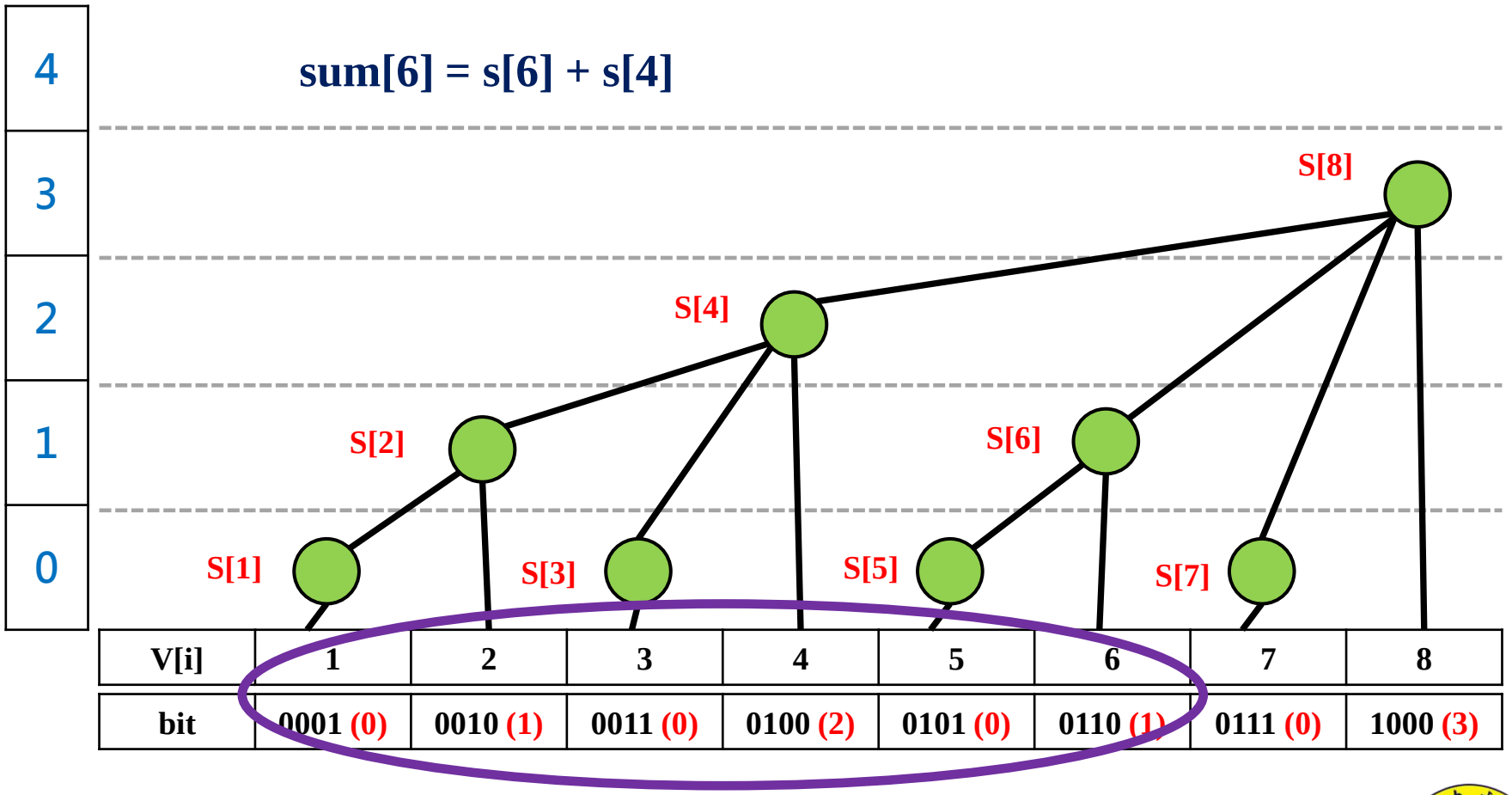
made by electron & free999

Binary Indexed Tree



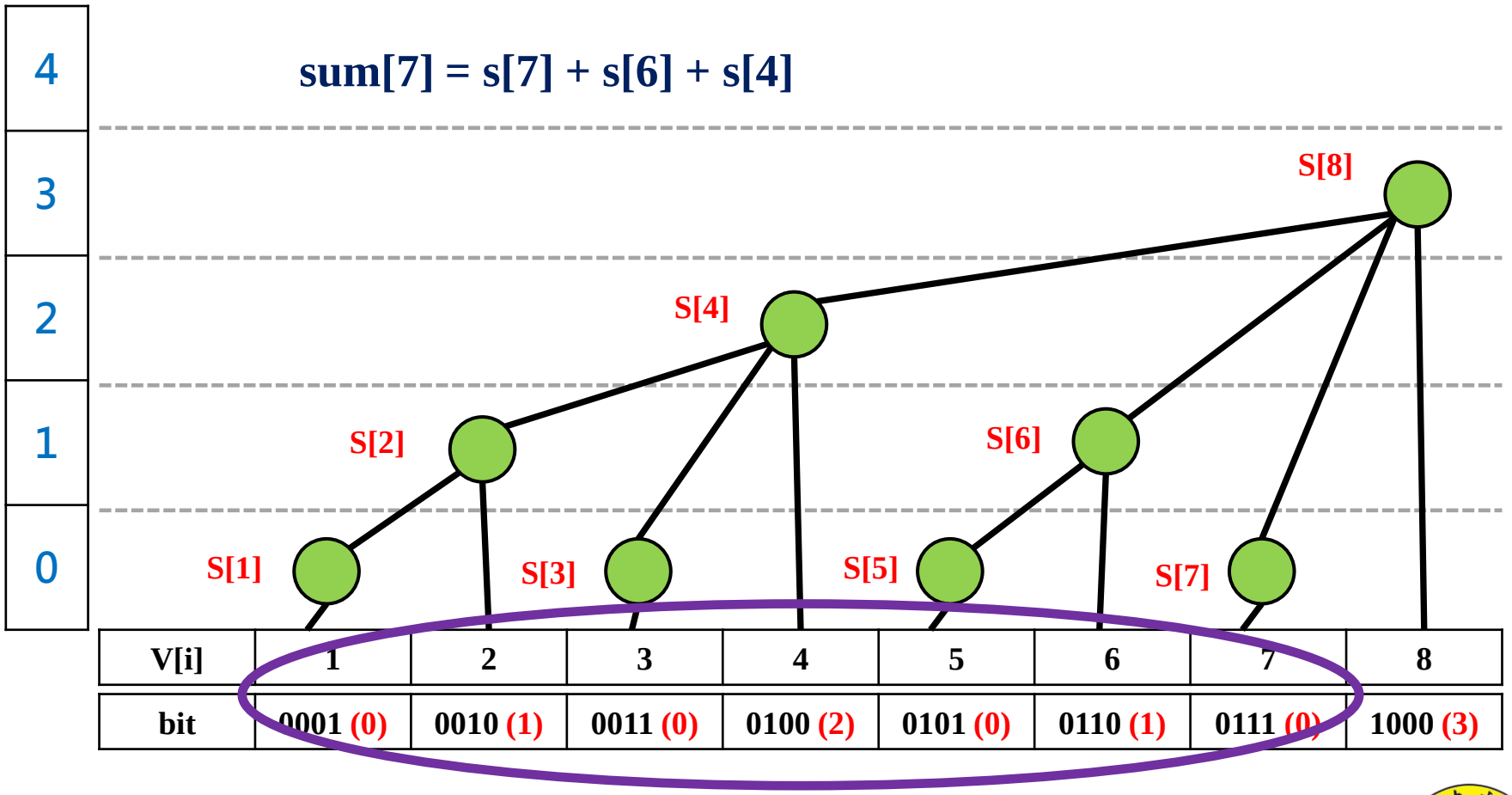
made by electron & free999

Binary Indexed Tree



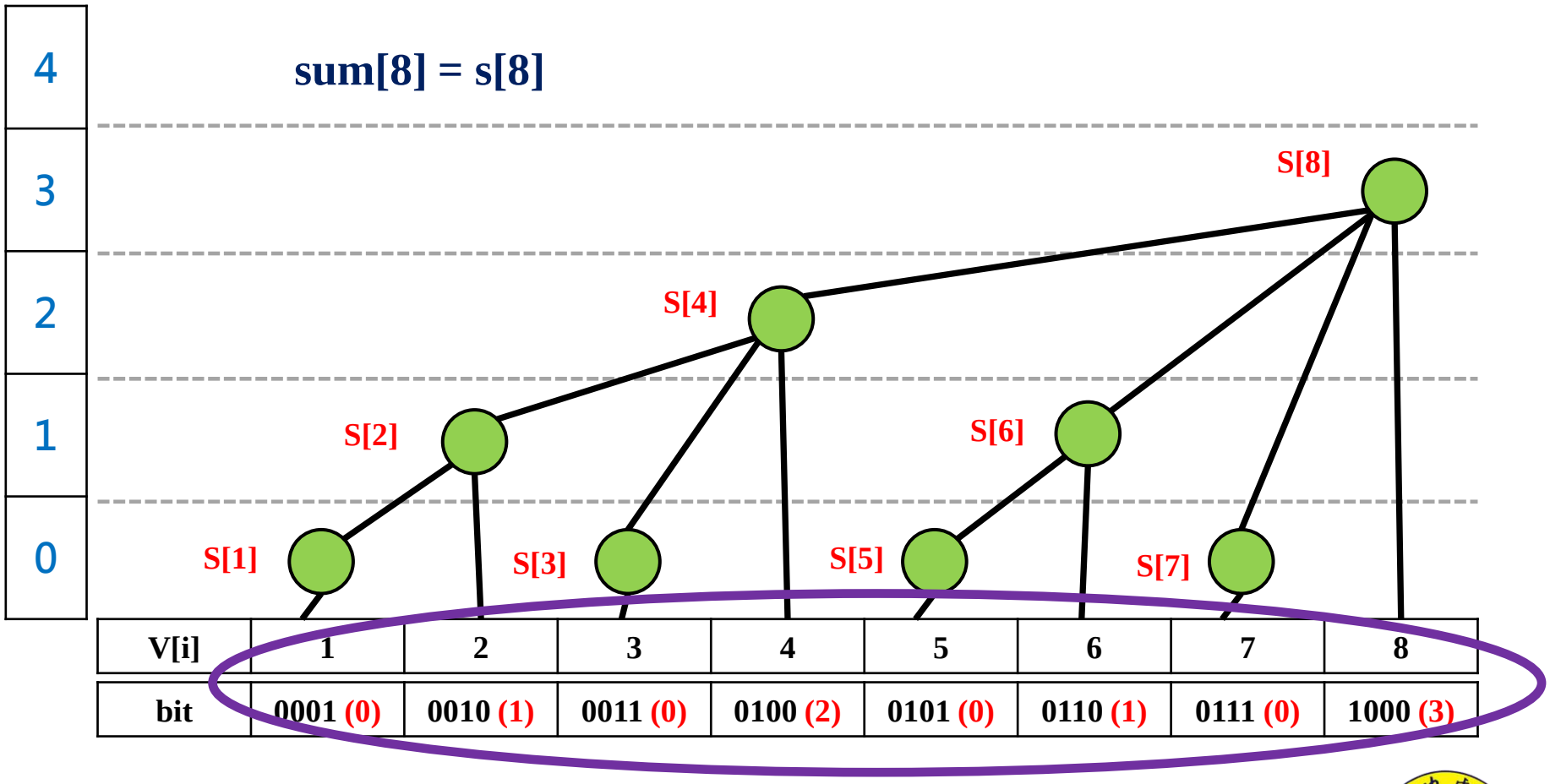
made by electron & free999

Binary Indexed Tree



made by electron & free999

Binary Indexed Tree



V[i]	1	2	3	4	5	6	7	8
bit	0001 (0)	0010 (1)	0011 (0)	0100 (2)	0101 (0)	0110 (1)	0111 (0)	1000 (3)

made by electron & free999

Binary Indexed Tree

Define:

```
int lowbit (int in)
{
        return in&(-in);
}
```

ex:

lowbit(1) = 1

lowbit(2) = 2

lowbit(3) = 1

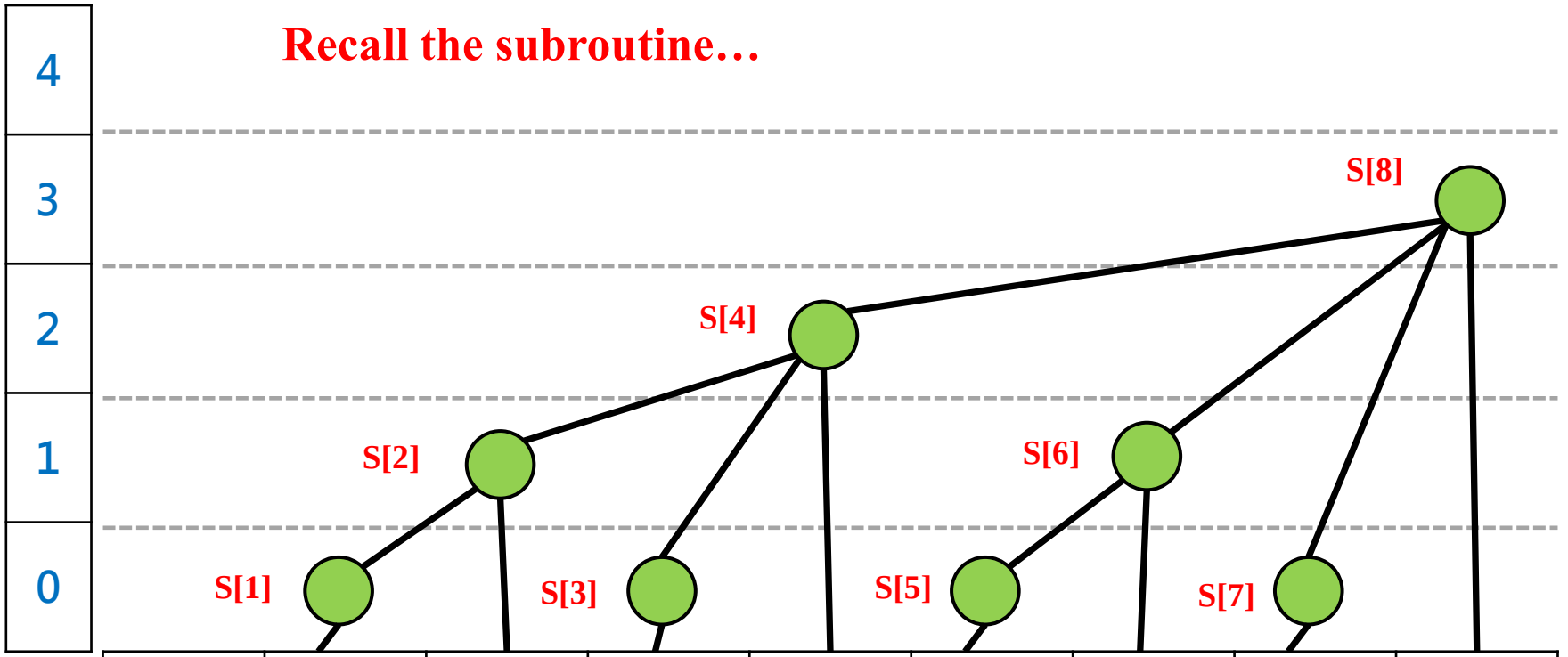
lowbit(4) = 4

…

V[i]	1	2	3	4	5	6	7	8
lowbit	1	2	1	4	1	2	1	8

Binary Indexed Tree

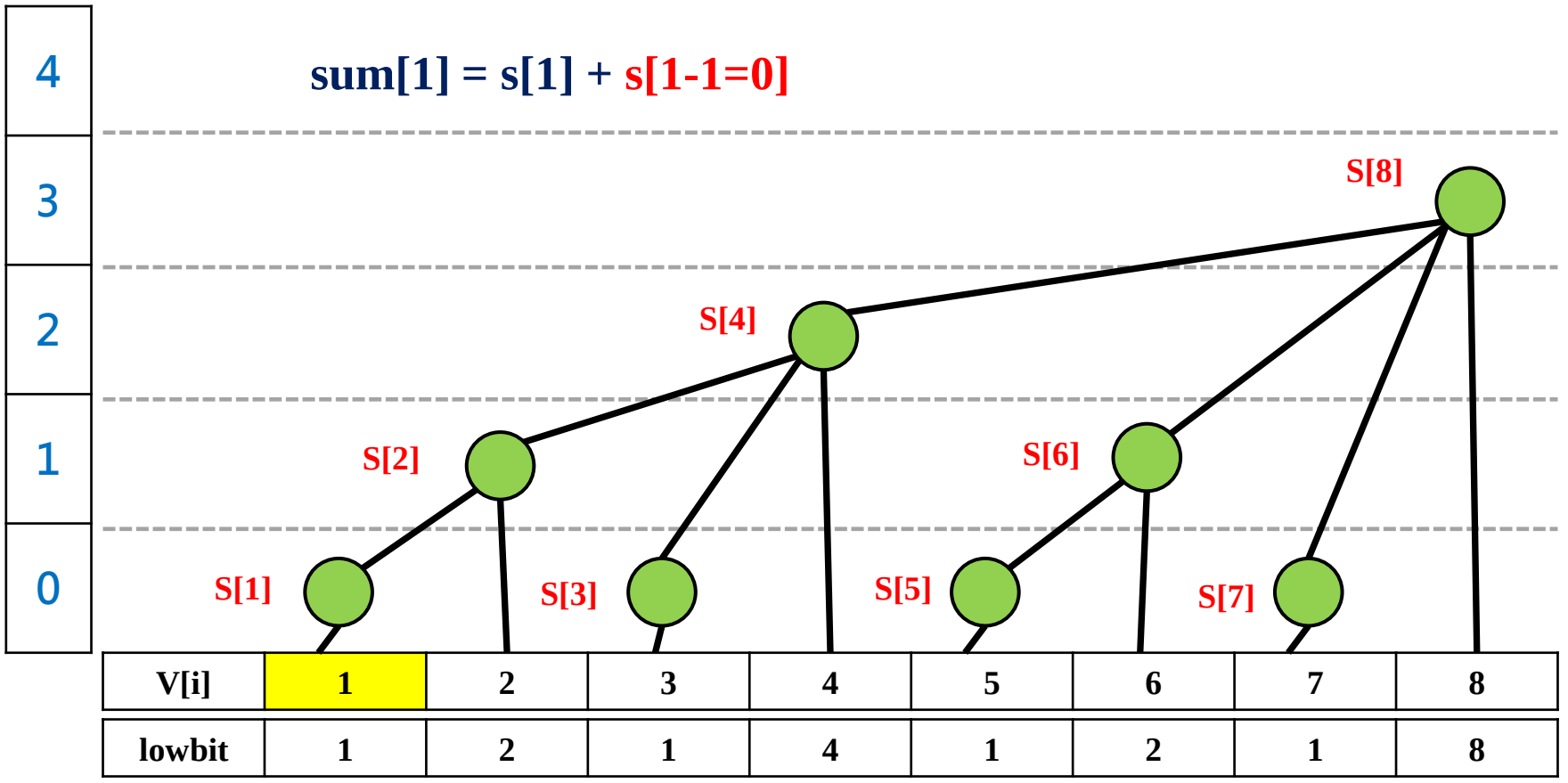
Recall the subroutine…



V[i]	1	2	3	4	5	6	7	8
lowbit	1	2	1	4	1	2	1	8

made by electron & free999

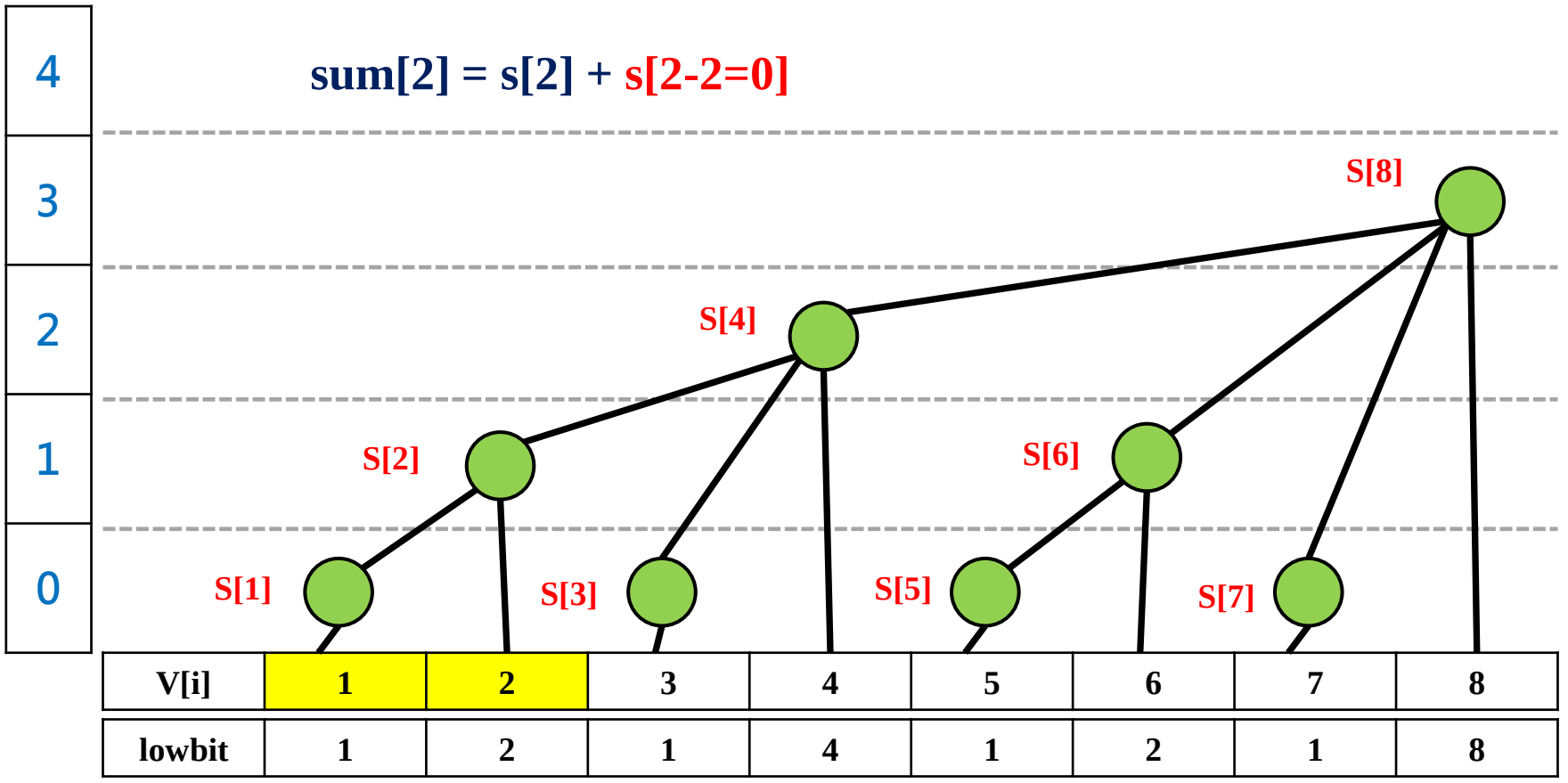
Binary Indexed Tree



V[i]	1	2	3	4	5	6	7	8
lowbit	1	2	1	4	1	2	1	8

made by electron & free999

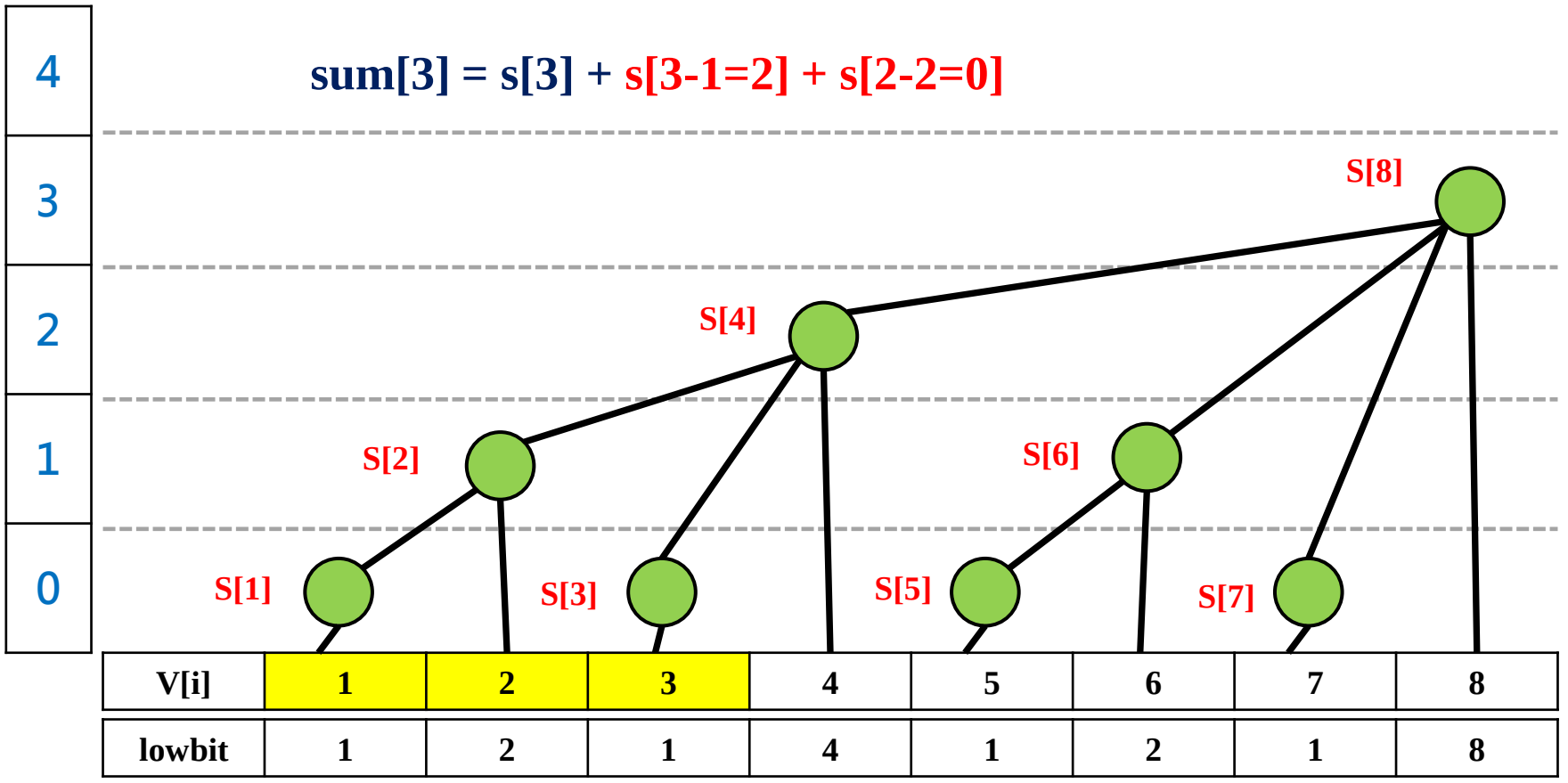
Binary Indexed Tree



V[i]	1	2	3	4	5	6	7	8
lowbit	1	2	1	4	1	2	1	8

made by electron & free999

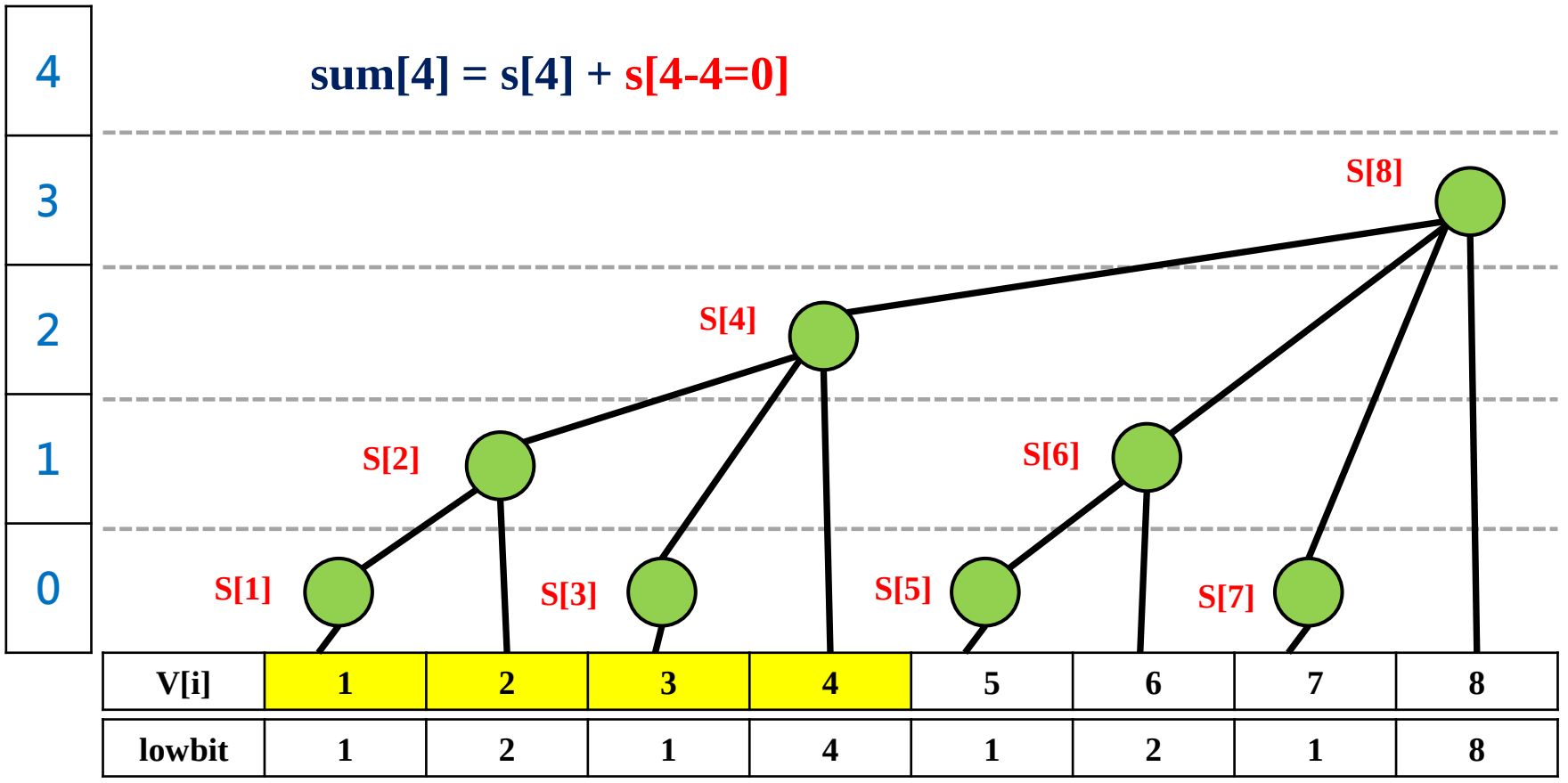
Binary Indexed Tree



V[i]	1	2	3	4	5	6	7	8
lowbit	1	2	1	4	1	2	1	8

made by electron & free999

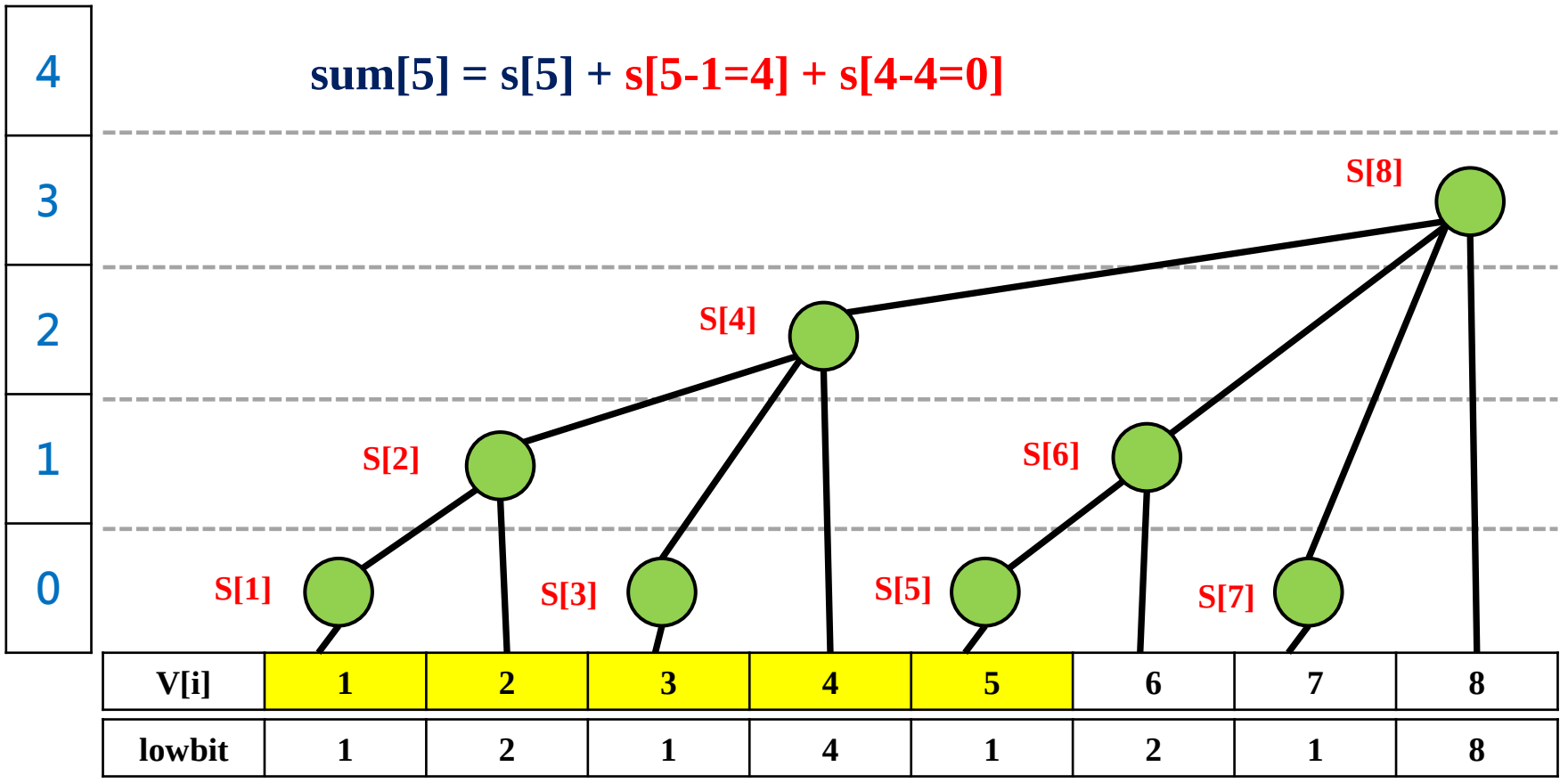
Binary Indexed Tree



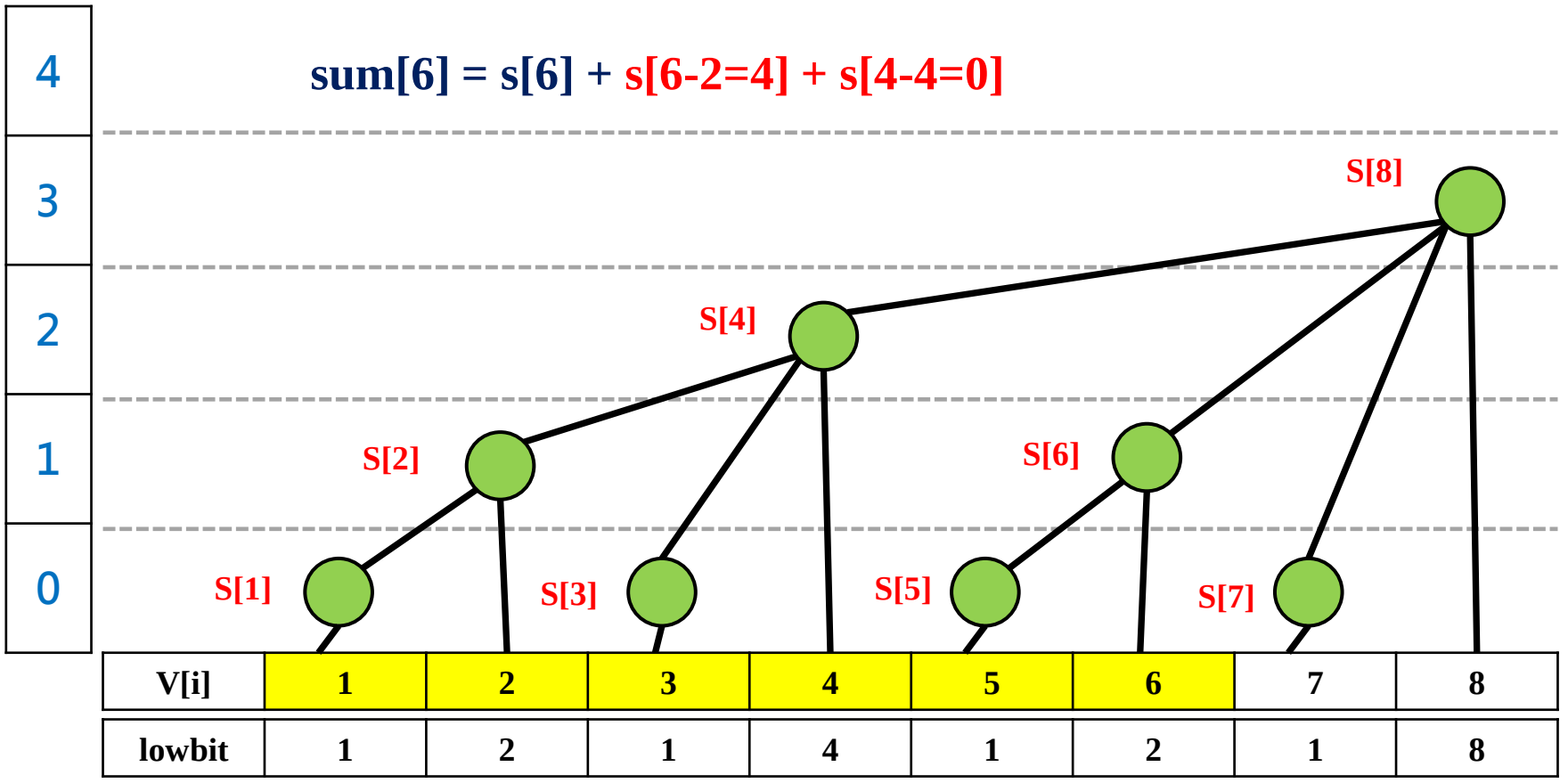
V[i]	1	2	3	4	5	6	7	8
lowbit	1	2	1	4	1	2	1	8

made by electron & free999

Binary Indexed Tree

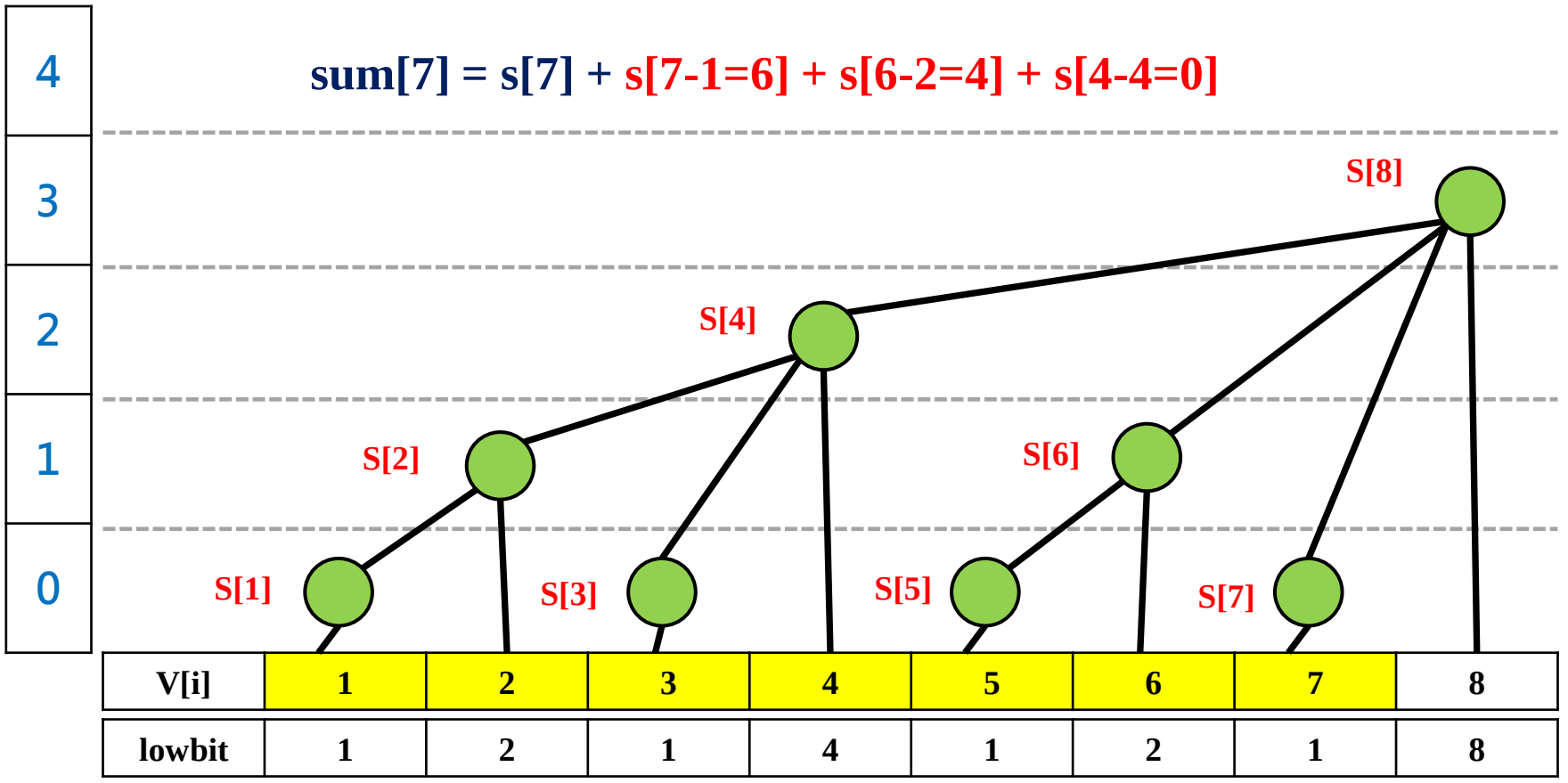


Binary Indexed Tree



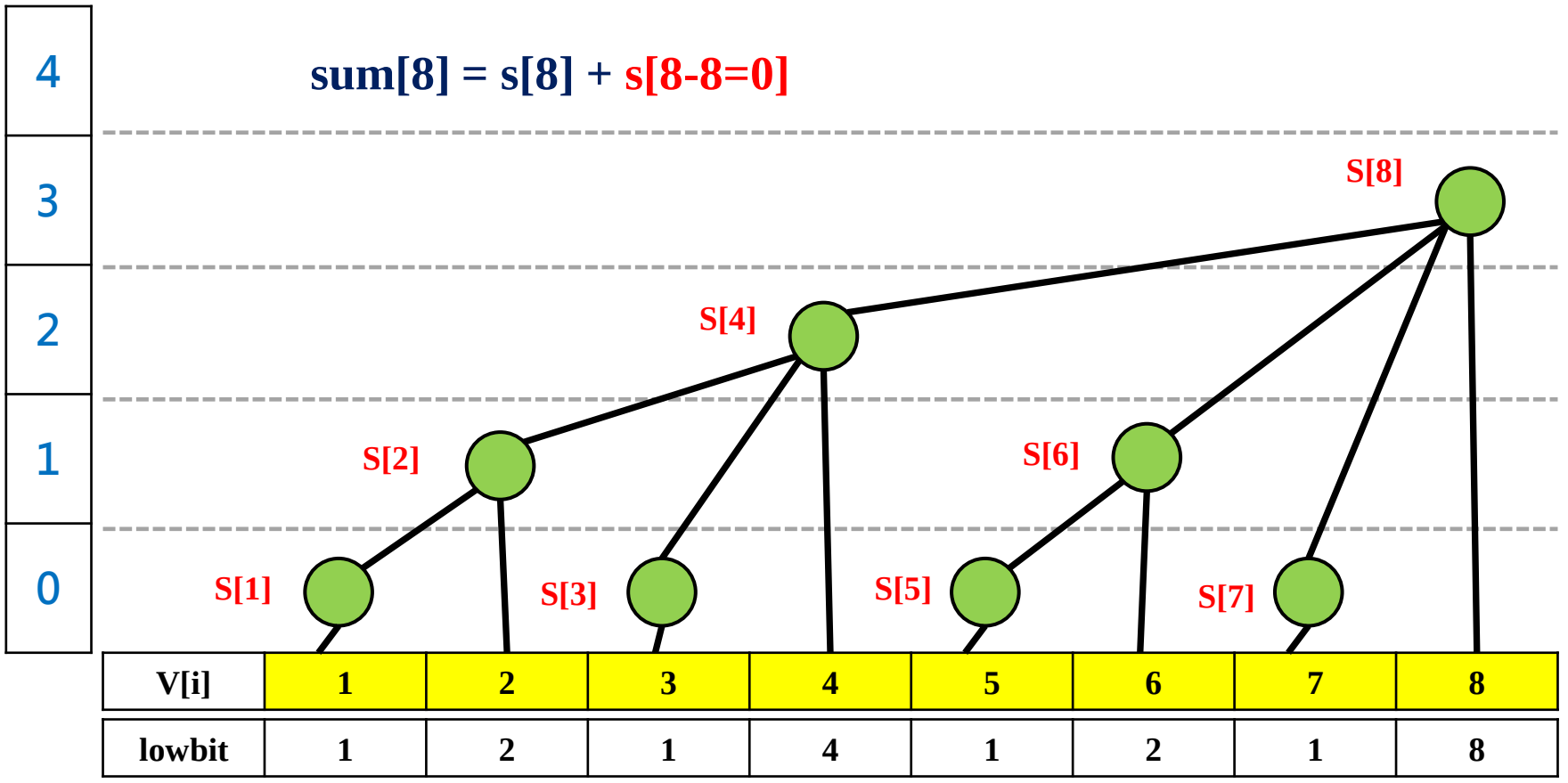
made by electron & free999

Binary Indexed Tree



made by electron & free999

Binary Indexed Tree



made by electron & free999

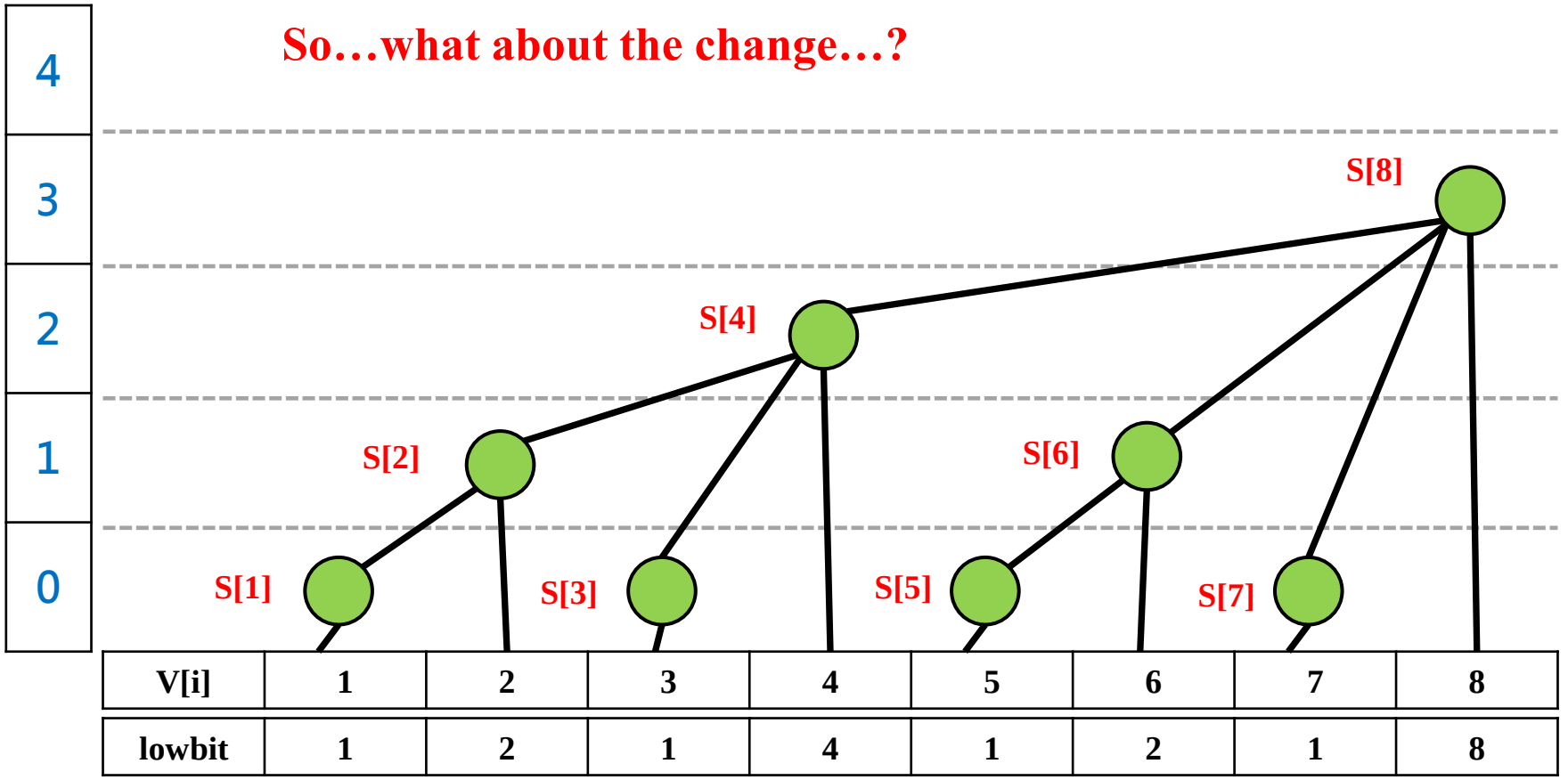
Binary Indexed Tree

Define:

```
int lowbit (int in)
{
    return in&(-in);
}

int getsum (int end)
{
    int ans = 0;
    while(end>0)
    {
        ans += s[end];
        end -= lowbit(end);
    }
}
```

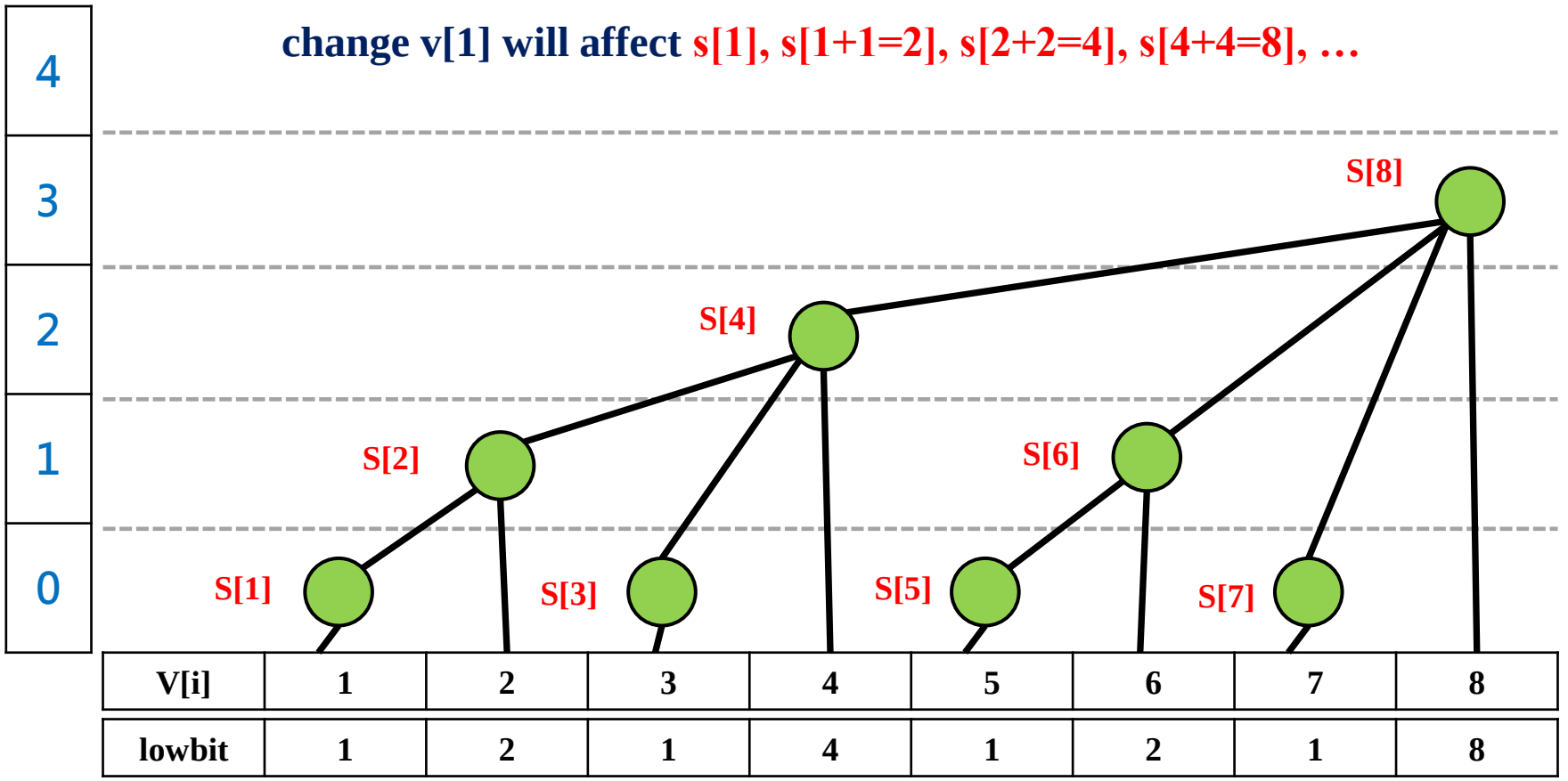
Binary Indexed Tree



V[i]	1	2	3	4	5	6	7	8
lowbit	1	2	1	4	1	2	1	8

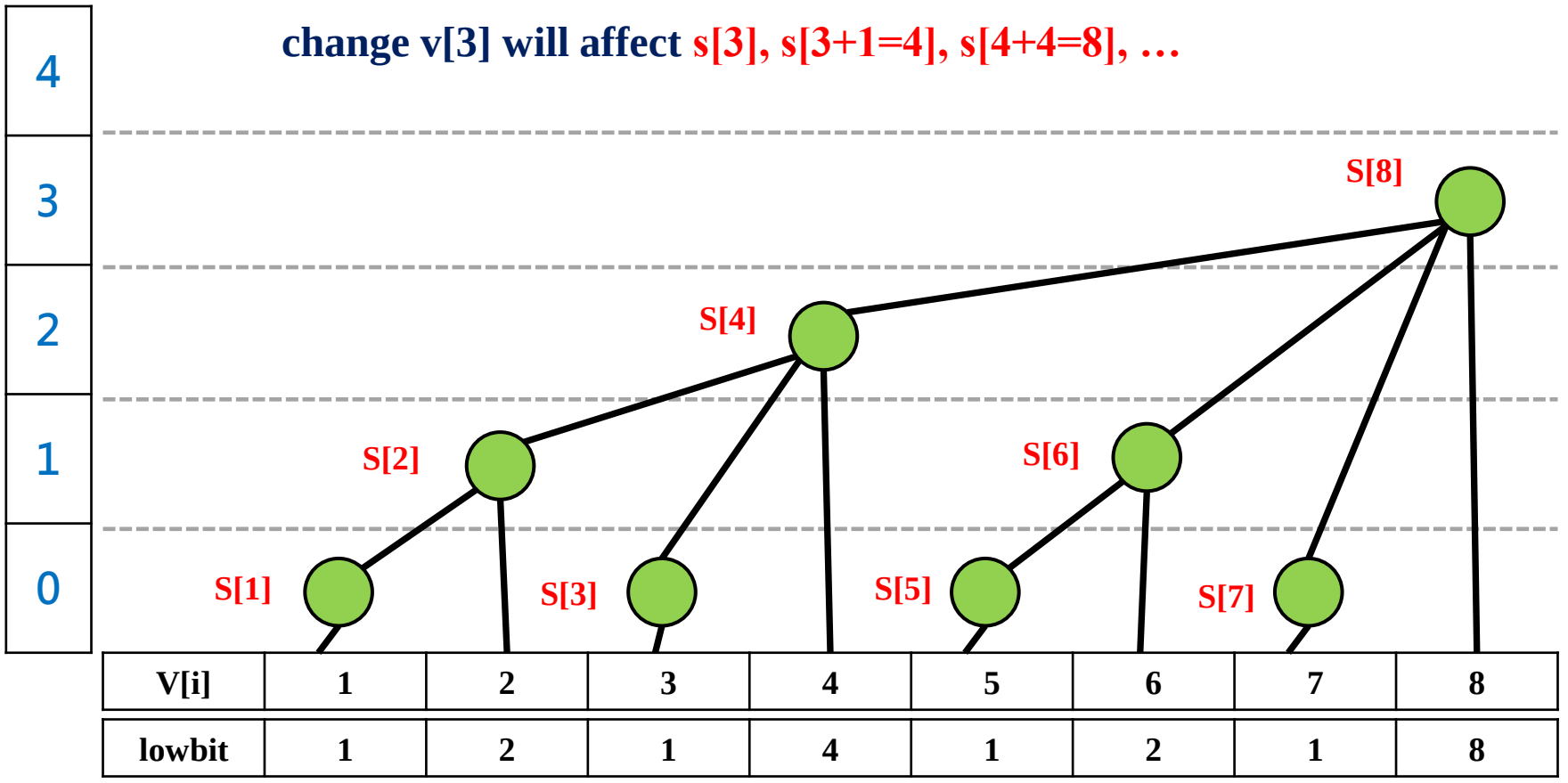
made by electron & free999

Binary Indexed Tree



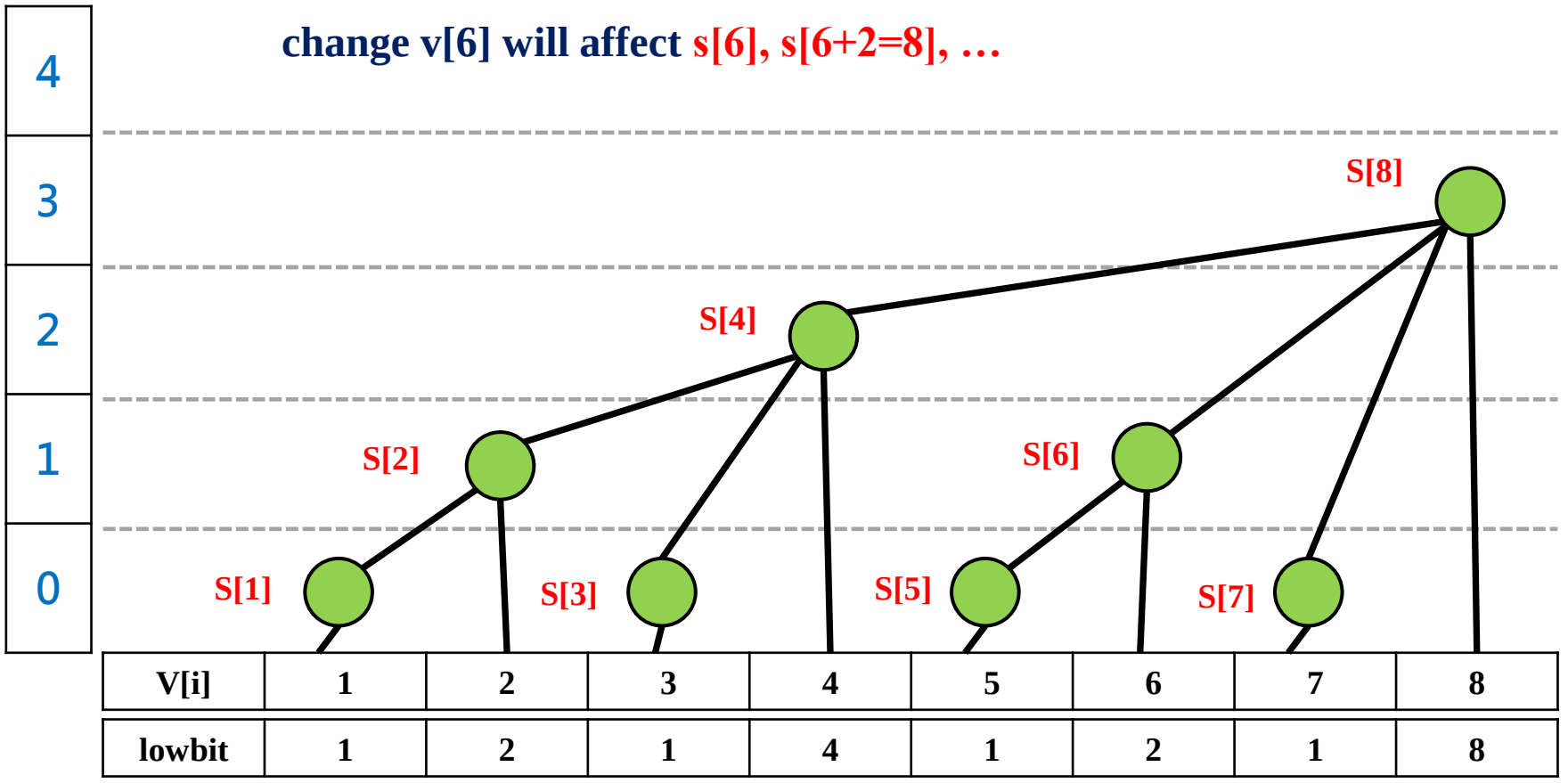
V[i]	1	2	3	4	5	6	7	8
lowbit	1	2	1	4	1	2	1	8

Binary Indexed Tree



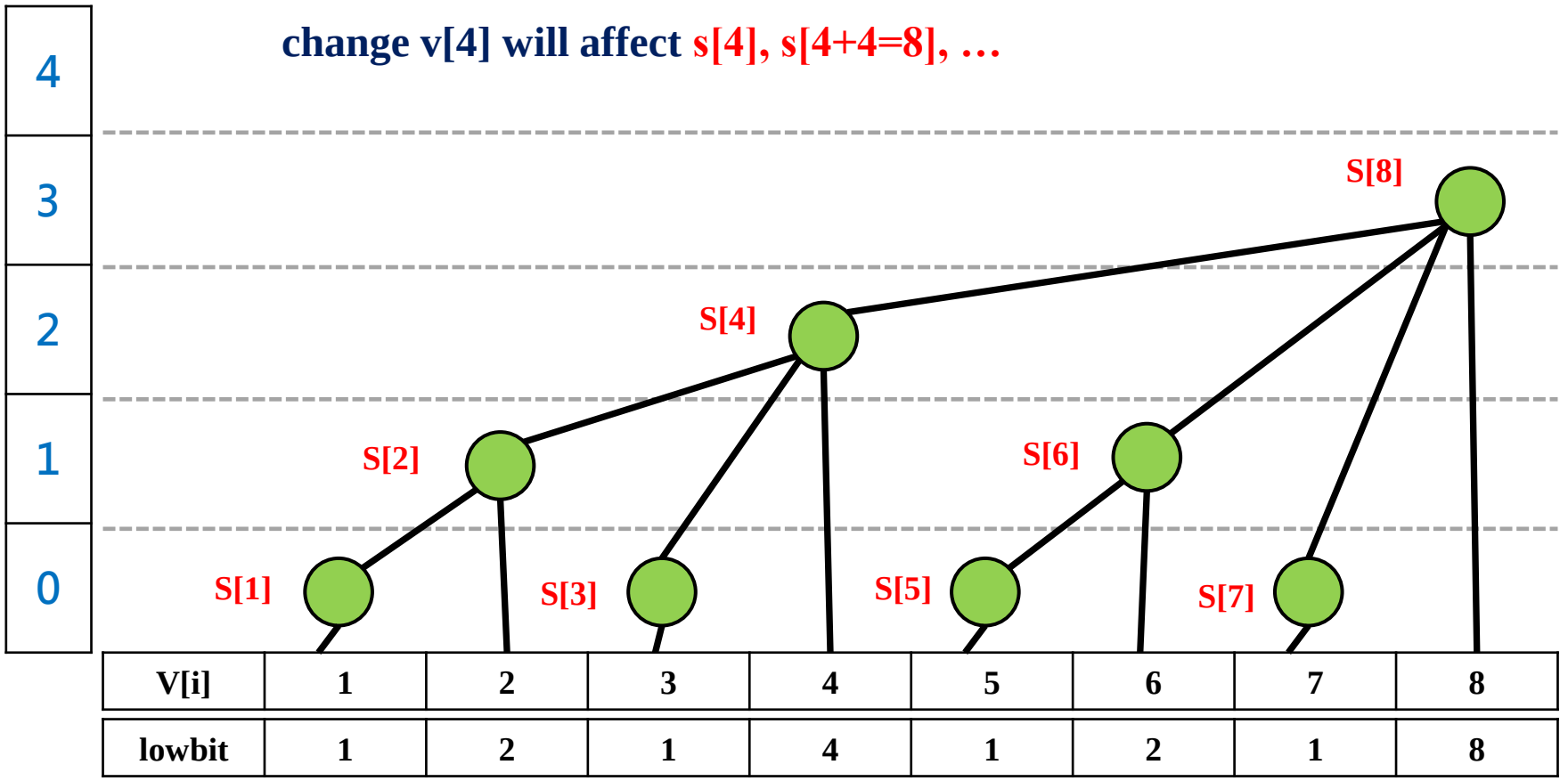
V[i]	1	2	3	4	5	6	7	8
lowbit	1	2	1	4	1	2	1	8

Binary Indexed Tree



V[i]	1	2	3	4	5	6	7	8
lowbit	1	2	1	4	1	2	1	8

Binary Indexed Tree



V[i]	1	2	3	4	5	6	7	8
lowbit	1	2	1	4	1	2	1	8

made by electron & free999

Binary Indexed Tree

Define:

```
int change (int end, int delta)
{
    for(int i=end; i<=maxsize; i+=lowbit(i))
        s[i] += delta;
}
```

Binary Indexed Tree

- How to find the summation between interval [i…j] ?
  - call the subroutine **“getsum[j] – getsum[i-1]”**
- Expand the 1 dimension into 2 dimension by yourself
- Replace such routines with a segment tree by yourself

Example

- POJ 2352

HomeWork

pku-1195

pku-3321

pku-2155

pku-2352

pku-3067

pku-2481

pku-2299

PKU-3368
PKU-2528
PKU-2828
PKU-2777
PKU-2886
PKU-2750
PKU-2482
PKU-2352