A_1

Q_1

1. True

PF: WTS: $\exists c_1, c_2 \in R^+, \exists n_0 \in N, \forall n \in N\, [n \geq n_0 \Rightarrow c_1 \cdot n^2 \leq 9n^2 + 5n - 17 \leq c_2 n^2]$

Choose $c_1 = 8, c_2 = 14, n_0 = 4$

Consider n be an arbitrary number of N

Suppose $n \geq 4$, we have,

$$\begin{aligned} 9n^2 + 5n - 17 &> 8n^2 + 5n - 17 && (9n^2 > 8n^2 \text{ if } n \geq 4) \\ &> 8n^2 && (5n - 17 \geq 0 \text{ if } n \geq 4) \\ 9n^2 + 5n - 17 &< 9n^2 + 5n && (-17 < 0) \\ &< 9n^2 + 5n^2 && (5n^2 > 5n \text{ if } n \geq 4) \\ &= 14n^2 \end{aligned}$$

$\Rightarrow 8n^2 \leq 9n^2 + 5n - 17 \leq 14n^2$

As n is an arbitrary number,

$$9n^2 + 5n - 17 \in \theta\left(n^2\right)$$

QED

2. True

PF: WTS: $\exists c_1 \in R^+, \exists n_0 \in N, \forall n \in N\, [n \geq n_0 \Rightarrow n \log(n) \leq c_1 n^2]$

Choose $c_1 = 1, n_0 = 10$

Consider n be an arbitrary number of N

Suppose $n \geq 10$

We have $n \log(n) < n \cdot n = n^2 \quad (n > \log(n) > 0 \text{ if } n \geq 10)$

$\Rightarrow n \log(n) \leq n^2$

As n is an arbitrary number of R,

$n(\log n) \in O(n^2)$

QED.

3. False

PF: WTS: $\forall c_1 \in R^+, \forall n_0 \in N, \exists n \in N\, [n \geq n_0 \wedge n^2 > c_1 n \log(n)]$

Consider c_1, n_0 be arbitrary number from $R^+$and N.

Choose n to be $\max\left(n_0, 10^{\lceil c_1 \rceil + 5}\right) \in N$

Then we have $n \geq n_0 > 0 \wedge n \geq 10^{\lceil c_1 \rceil + 5}$

Consider $h(n) = \frac{n}{\log(n)} \quad (n \geq 5)$

$$
\begin{aligned}
h(n)' &= \left(\log(n) - n \cdot \frac{1}{n \ln(10)}\right) / \log^2(n) \\
&= \left(\log(n) - \frac{1}{\ln(10)}\right) / \log^2(n) \\
&> 0 \quad \text{if } n \geq 5
\end{aligned}
$$

As $10^{\lceil c_1 \rceil + 5} > 10^5 > 5 \wedge n \geq 10^{\lceil c_1 \rceil + 5} > 5$

$$
\begin{aligned}
h(n) &= \frac{n}{\log n} \\
&\geq \frac{10^{\lceil c_1 \rceil + 5}}{\lceil c_1 \rceil + 5} \\
&> \frac{(\lceil c_1 \rceil + 5)^2}{\lceil c_1 \rceil + 5} \quad (\forall n \in R^+, 10^n > n^2) \\
&= \lceil c_1 \rceil + 5 \\
&> c_1
\end{aligned}
$$

$\Rightarrow \frac{n}{\log(n)} > c_1$

As $n > 5, \quad \log(n) > 0$

$\Rightarrow n > c_1 \log(n)$

$\Rightarrow n^2 > c_1 n \log(n)$

$\therefore n \geq n_0, n^2 > c_1 n \log(n),$ as wanted

$\Rightarrow n^2 \notin O(n \log(n))$

QED.

4.True

PF: WTS: $\exists c \in R^+, \exists n_0 \in N, \forall n \in N\,[n \geq n_0 \Rightarrow \log(n^{2025} + n + 1 \leq c\log(n)]$

Choose $c = 2026, n_0 = 2$

Consider n to be an arbitrary number $\in N$

Suppose $n \geq n_0 = 2$

Consider $g(n) = n^{2026} - n^{2025} - n - 1 \quad (n \geq 2)$

$g'(n) = 2026n^{2025} - 2025n^{2024} - 1$

$= n^{2024}(2026n - 2025) - 1$

$n^{2024}(2026n - 2005) > n^{2024} \geq 2^{2024} > 1$

$\therefore g'(n) > 0 \quad (n \geq 2)$

$\therefore g(n) > g(2) = 2^{2026} - 2^{2025} - 2 - 1 > 0 \quad (n \geq 2)$

$\therefore n^{2026} > n^{2025} + n + 1 > 0$

$\therefore \log\left(n^{2026}\right) > \log\left(n^{2026} + n + 1\right)$

$\therefore \log\left(n^{2025} + n + 1\right) < \log\left(n^{2026}\right) = 2026\log(n)$

$\therefore \log\left(n^{2025} + n + 1\right) \leq 2026\log(n),$ as wanted

As n is an arbitrary number,

$$\log\left(n^{2025} + n + 1\right) \in O(\log(n))$$

QED.

5.False

PF: WTS: $\forall c \in R^+, \forall n_0 \in N, \exists n \in N\,[n \geq n_0 \wedge (n+1)! > c \cdot n!]$

Consider $c \in R^+, n_0 \in N$ to be arbitrary numbers.

Choose $n = \max\,(n_0,\,\lceil c \rceil)$

$\Rightarrow n \geq n_0, n \geq \lceil c \rceil$

$(n+1)! = n + 1 \cdot n! \geq (\lceil \bar{c} \rceil + 1) - n! > c \cdot n!$, as wanted $\Rightarrow (n+1)! \notin O(n!)$

QED

Q_2

1. False

PF: WTS: $\exists f, g \in N \to R^+, f(n) \notin O(g(n)) \wedge g(n) \notin O(f(x))$

Choose $f(n) = \begin{cases} n & n \text{ is even} \\ 1 & n \text{ is odd} \end{cases} \qquad g(n) = \begin{cases} 1 & n \text{ is even} \\ n & n \text{ is odd} \end{cases}$

WTS: $\forall c \in R^+, \forall n_0 \in N, \exists n_1, n_2 \in N,$

$[(n_1 \geq n_0 \wedge f(n_1) > cg(n_1)) \wedge (n_2 \geq n_0 \wedge g(n_2) > cf(n_2))]$

Consider $c \in R^+, n_0 \in N$ to be an arbitrary number.

Choose $u_1 = 2\lceil c \rceil (n_0 + 1), u_2 = 1 + 2\lceil c \rceil (n_0 + 1)$

$(n_1 = 2\lceil c \rceil (n+1) \geq 2(n_0+1) > n_0, n_2 > n_1 > n_0) \Rightarrow (n_1 \geq n_0, n_2 \geq n_0)$

$f(n_1) = 2\lceil c \rceil (n_0 + 1) > \lceil c \rceil > c \cdot 1 = c \cdot g(n_1)$, as wanted

$g(n_2) = 1 + 2\lceil c \rceil (n_0 + 1) > \lceil c \rceil > c \cdot 1 = c \cdot f(n_2)$, as wanted

QED

2. True:

PF: Consider f_1, f_2, g_2, g_2 be arbitrary functions $\in N \to R^+$

Suppose $f_1 \in O(g_1) \wedge f_2 \in O(g_2)$

we have $\exists c_1 \in R^+, \exists n_0 \in N, \forall n \in N [n \geq n_0 \Rightarrow f_1(n) \leq c_1 g_1(n)]$

$\exists c_2 \in R^+, \exists n_1 \in N, \forall n \in N [n \geq n_1 \Rightarrow f_2(n) \leq c_2 g_2(n)]$

WTS: $\exists c_3 \in R^+, \exists n_2 \in N, \forall n \in N$

$[n \geq n_2 \Rightarrow f_1(n) + f_2(n) \leq c_3 (g_2(n) + g_2(n))]$

Consider n be an arbitrary number from N

$\Rightarrow \exists c_1, c_2 \in R^+, n_0, n_1 \in N,$

s.t. $(n \geq n_0 \Rightarrow f_1(n) \leq c_1 g_1(n)) \wedge (n \geq n_1 \Rightarrow f_2(n) \leq c_2 g_2(n))$

Choose $c_3 = \max(c_1, c_2) \quad n_2 = \max(n_0, n_1)$

Suppose $n \geq n_2$

$\Rightarrow n \geq n_2 \geq n_0, n \geq n_2 \geq n_1, c_3 \geq c_1, c_3 \geq c_2$

$\Rightarrow f_1(n) \leq c_1 g_1(n) \leq c_3 g_1(n) \quad (g_1(n) > 0 \wedge c_1 > 0)$

$\quad f_2(n) \leq c_2 g_2(n) \leq c_3 g_2(n) \quad (g_2(n) > 0 \wedge c_2 > 0)$

$\Rightarrow f_1(n) + f_2(n) \leq c_3 g_1(n) + c_3 g_2(n) = c_3 \left(g_1(n) + g_2(n)\right),$ as wanted
QED.

3. False

PF: WTS: $\exists f, g \in N \to R^+, f \in O(g) \wedge 2^f \notin O(g)$

Choose $f(n) = 2n, g(n) = n$

WTS: $\exists c_1 \in R^+, \exists n_0 \in N, \forall n \in N \left[n \geq n_0 \Rightarrow 2n \leq c_1 n\right]$

$\wedge \forall c_2 \in R^+, \forall n_1 \in N, \exists n' \in N \left[n' \geq n_1 \wedge 2^{2n'} > c_2 2^{n'}\right]$

Consider n be an arbitrary number from N

Choose $c_1 = 3, n_0 = 1$

Suppose $n \geq n_0 = 1, 2n < 3n = c_1 n$, as wanted

Consider $c_2 \in R^+, n_1 \in N$ be arbitrary numbers.

Choose $n' = \max\left(n_1, \lceil \log_2^{\lceil c_2 \rceil + 1} \rceil\right)$

$n' \geq n_1, n' \geq \left\lceil \log_2^{\lceil c_2 \rceil + 1} \right\rceil \geq \log_2^{\lceil c_2 \rceil + 1}$

$\Rightarrow 2^{2n'} = 2^{n'} \cdot 2^{n'} \geq 2^{n'} \cdot 2^{\log_2^{\lceil c_2 \rceil + 1}} = 2^{n'} \cdot (\lceil c_2 \rceil + 1)$

$> 2^{n'} \cdot c_2 = c_2 \cdot 2^{n'}$, as wanted.
QED.

4. True

Consider f, g be arbitrary from $N \to R^+$

Suppose $f \in \Omega(g) \wedge g \in \Omega(h)$

$\Rightarrow \exists c_0 \in R^+, \exists n_0 \in N, \forall n \in N (n \geq n_0 \Rightarrow f(n) \geq c_0 g(n))$

$\wedge \exists c_1 \in R^+, \exists n_1 \in N_1, \forall n \in N \left(n \geq n_1 \Rightarrow g(n) \geq c_1 h(n)\right)$

WTS: $\exists c_2 \in R^+, \exists n_2 \in N, \forall n \in N \left(n \geq n_2 \Rightarrow f(n) \geq c_2 h(n)\right)$

Consider n be an arbitrary number from N

Choose $c_2 = c_0c_1, n_2 = \max(n_0, n_1)$

Suppose $n \geq n_2$

$\Rightarrow n \geq n_0, n \geq n_1$

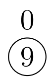
$f(n) \geq c_0g(n) > 0, g(n) \geq c_1h(n) > 0$

$f(n) \geq c_0g(n) \geq c_0c_1h(n) = c_2h(n)$, as wanted

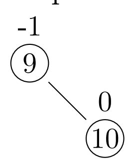
QED.

Q3.

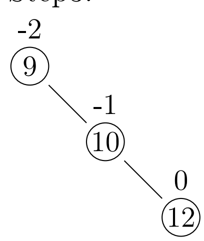
Step1:



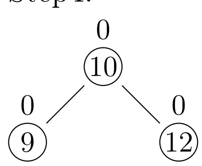
Step2:



Step3:

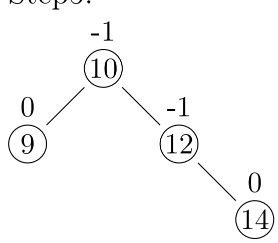


Step4:

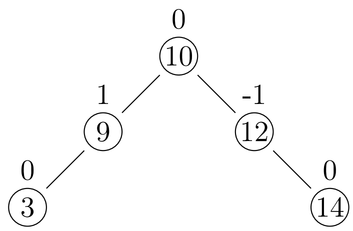


type: counter-clockwize
node: 10

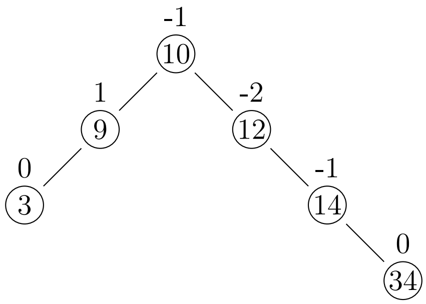
Step5:



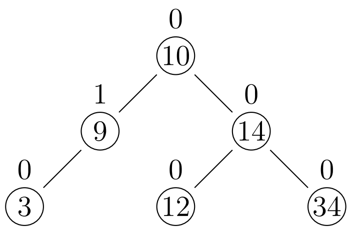
Step6:


0
10
1
9
-1
12
0
3
0
14

Step7:

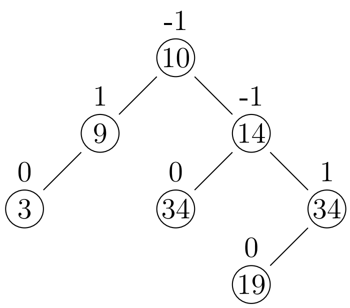


Step8:

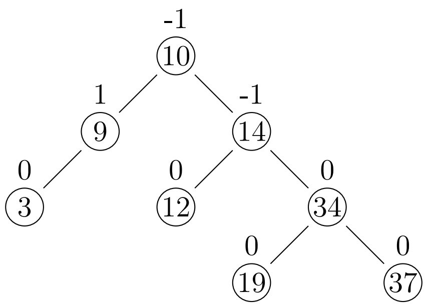


type: counter-clockwize
node: 14

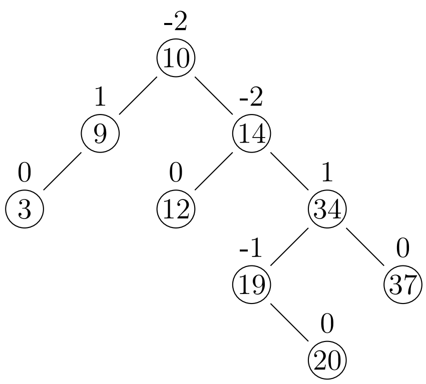
Step9:



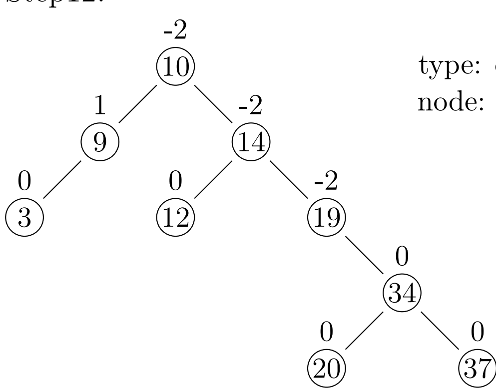
Step10:



Step11:



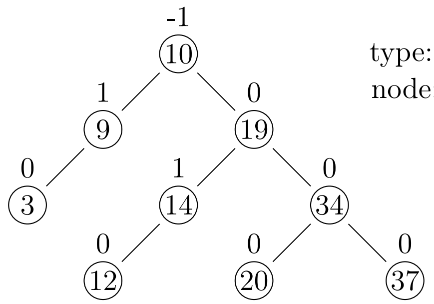
Step12:



type: clockwize

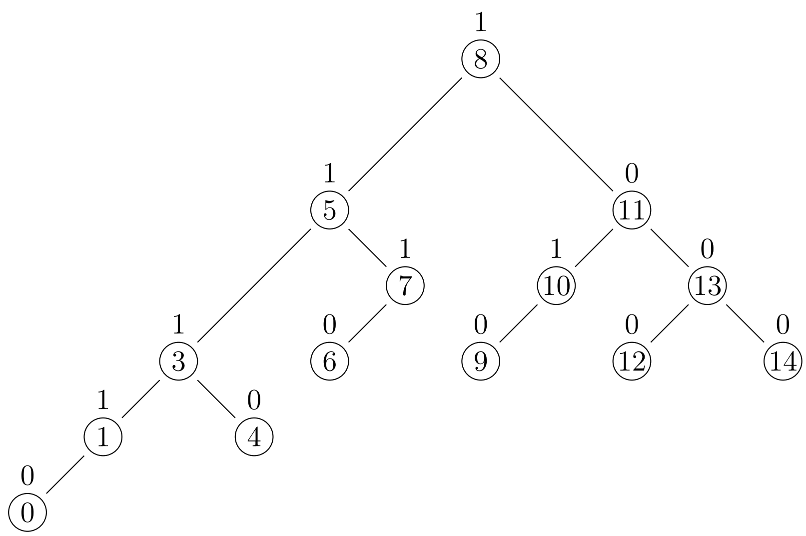
node: 19

Step13:

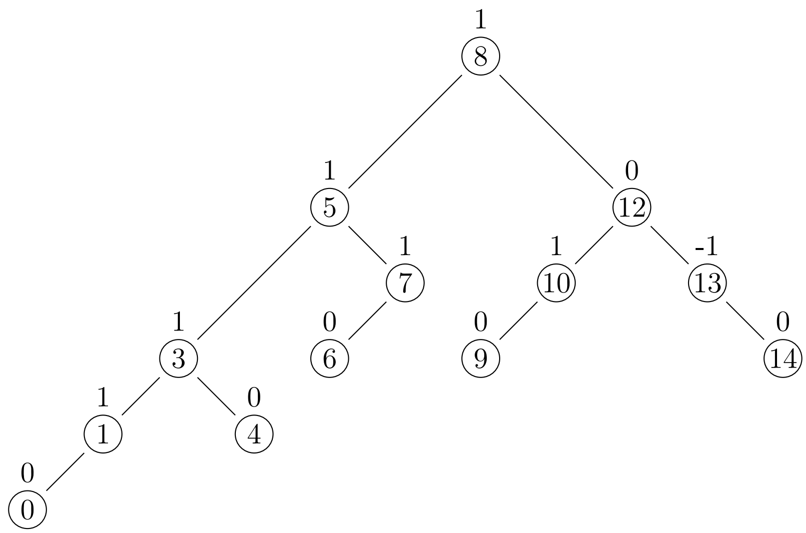


type: counter-clockwize
node: 19

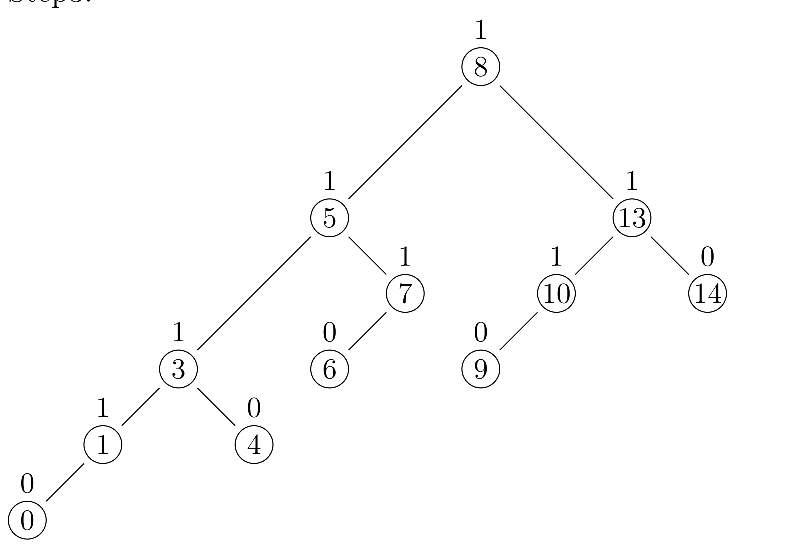
2. Step1:



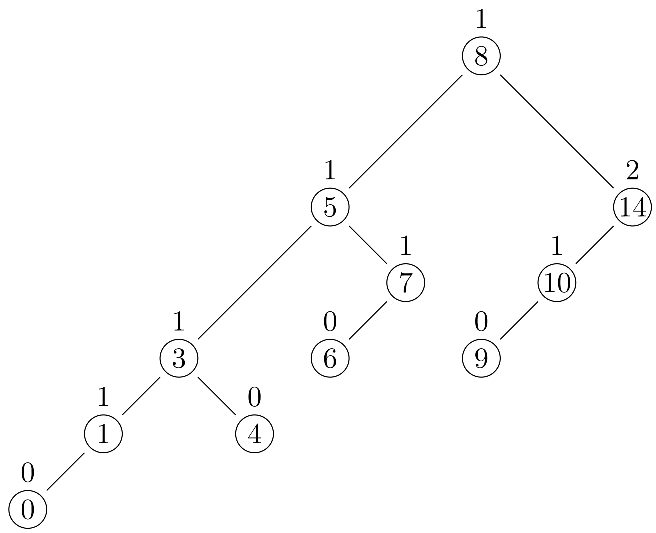
Step2:



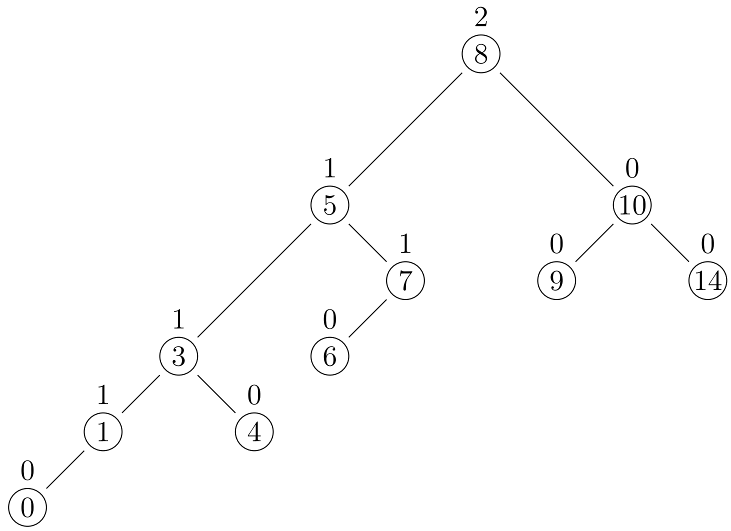
Step3:



Step4:

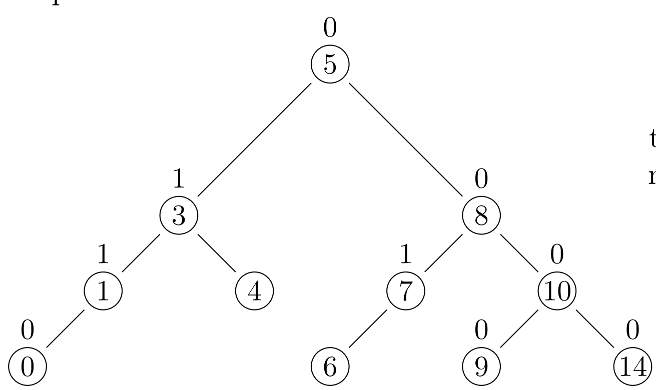

1
8
1
5
2
14
1
7
1
10
1
3
0
6
0
9
1
1
0
4
0
0

Step5:



type: clockwize
node: 10

Step6:



type: clockwize
node: 5

Q4

1.

The node S has:

#key: the left end point x of interval [x, y]

#right_end: the right end point y of interval [x, y]

#height: the height of the tree whose root is S

#max_y: the max value of y in the tree whose root is S

#left: the left node

#right: the right node

2.

For the rebalance mentioned below, we need to add update_height and update_max_y functions in the end of each single and double rotation by using the course rebalance algorithm.

```
update_height(S) -- pseudocode
if S == nil:
    return
if S.left == nil and S.right == nil:
    S.height = 1
    return
if S.left == nil and S.right != nil:
    S.height = S.right.height + 1
    return
if S.left != nil and S.right == nil:
    S.height = S.left.height + 1
    return
```

```
S.height = max(S.left.height, S.right.height) + 1
return

update_max_y(S) -- pseudocode
if S == nil:
    return
if S.left == nil and S.right == nil:
    S.max_y = S.right_end
    return
if S.left == nil and S.right != nil:
    S.max_y = max(S.right.max_y, S.right_end)
    return
if S.left != nil and S.right == nil:
    S.max_y = max(S.left.max_y, S.right_end)
    return
S.max_y = max(S.left.max_y, S.right.max_y, S.right_end)
return

insert(S, x, y) -- pseudocode
if S == nil:
    S = new node(key=x, right_end=y, height=1, max_y=y, left=nil, right=nil)
    return S
if S.key == x and S.right_end == y:
    return S
if S.key > x:
    S.left = insert(S.left, x, y)
```

```
    update_height(S)
    update_max_y(S)
    rebalance at S
    return S
else:
    S.right = insert(S.right, x, y)
    update_height(S)
    update_max_y(S)
    rebalance at S
    return S
```

```
delete(S, x, y) -- pseudocode
if S == nil:
    return nil
if S.key == x and S.right_end == y:
    if S.left == nil:
        return S.right
    if S.right == nil:
        return S.left;
    search from S.right to the left most node p of the right subtree of S
    S.key = q.key
    S.right_end = q.right_end
    S.right = delete(S.right, q.key, q.right_end)
    update_height(S)
    update_max_y(S)
    rebalance at S
```

```
    return S
if S.key > x:
    S.left = delete(S.left, x, y)
    update_height(S)
    update_max_y(S)
    rebalance at S
    return S
S.right = delete(S.right, x, y)
update_height(S)
update_max_y(S)
rebalance at S
return S
```

```
eclipse(S, l, h) -- pseudocode
if S == nil or S.max_y <= l:
    return None
if l <= S.key and S.right_end <= h:
    return [S.key, S.right_end]
if S.left != nil and S.left.max_y > l:
    interval = eclipse(S.left, l, h)
    if interval == None:
        if S.right == nil or S.right.max_y <= l:
            return None
        return eclipse(S.right, l, h)
    return interval
return eclipse(S.right, l, h)
```

3.

For insert and delete, my algorithm is based on insert and delete operations of BST(i.e. I only search in one sub-tree), whose complexity is $O(log(n))$ if the tree with root S is AVL with height $log(n)$. Also my update functions for height and max_y are with complexity $O(1)$, so the whole complexity is $O(log(n))$. Also,as I have helper function for updating height and max_y, similar insert and delete approaches to BST and the use of rebalanced, my algorithms is right. By the way, I first update height and then rebalanced the tree, which let all nodes have correct balance factors. At last, I update the max_y based on new AVL.

For eclipse, in base case I consider the easiest cases that must have and must not have, with complexity $O(1)$, then I consider cases that the sub-trees of S may have such interval. I first search on left sub-tree and return if find(similar to search in BST), the complexity is $O(log(n))$. If I can not find, then I goes to the right sub-tree with similar approach, also with $O(log(n))$ complexity. So in worst case, my complexity is $O(log(n)) + O(log(n))$ which is still $O(log(n))$. As I have consider from base cases to all sub-tree cases, my algorithm is correct.