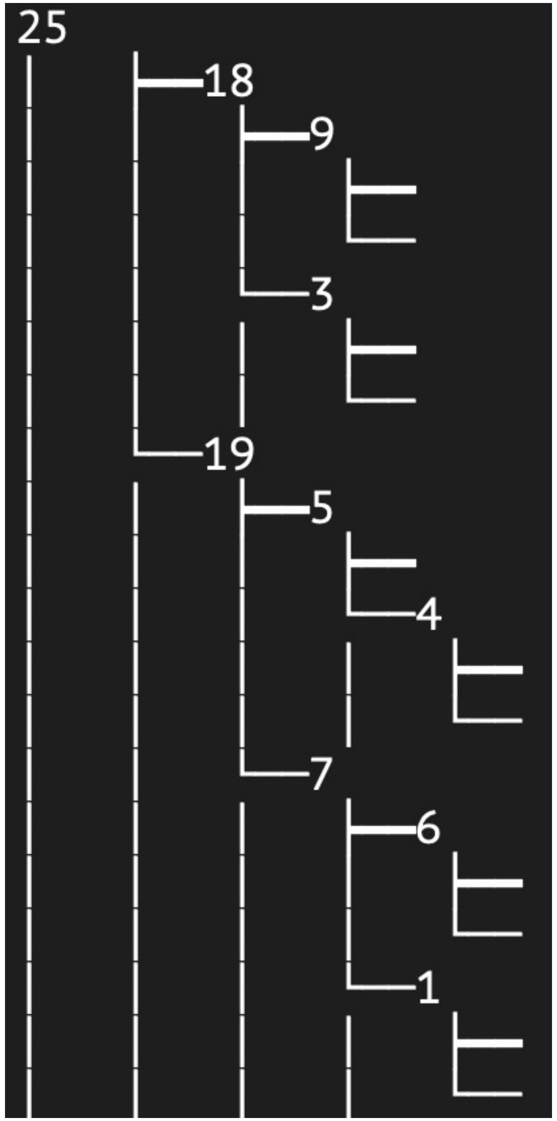
Heap

delete() Method

Prepared by Mahdi Ghamkhari

delete()

We store 25 in a temporary variable so that we can return it

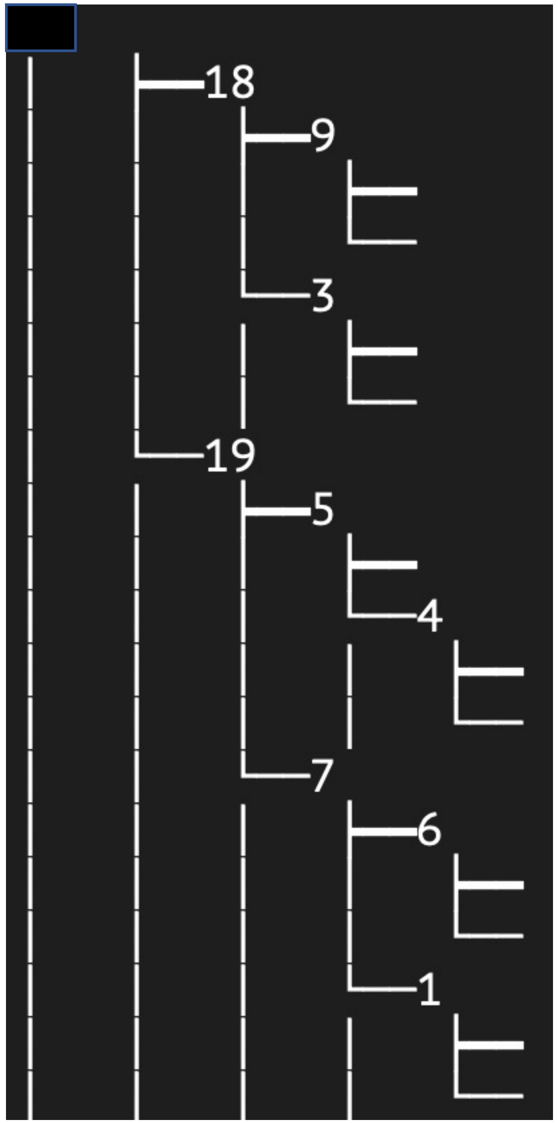


```
25
│   ├──18
│   │   ├──9
│   │   │   ├──
│   │   │   └──
│   │   └──3
│   │   │   ├──
│   │   │   └──
│   └──19
│   │   ├──5
│   │   │   ├──
│   │   │   └──4
│   │   │   │   ├──
│   │   │   │   └──
│   │   └──7
│   │   │   ├──6
│   │   │   │   ├──
│   │   │   │   └──
│   │   │   └──1
│   │   │   │   ├──
│   │   │   │   └──
```

delete()

We store 25 in a temporary variable so that we can return it

We delete 25



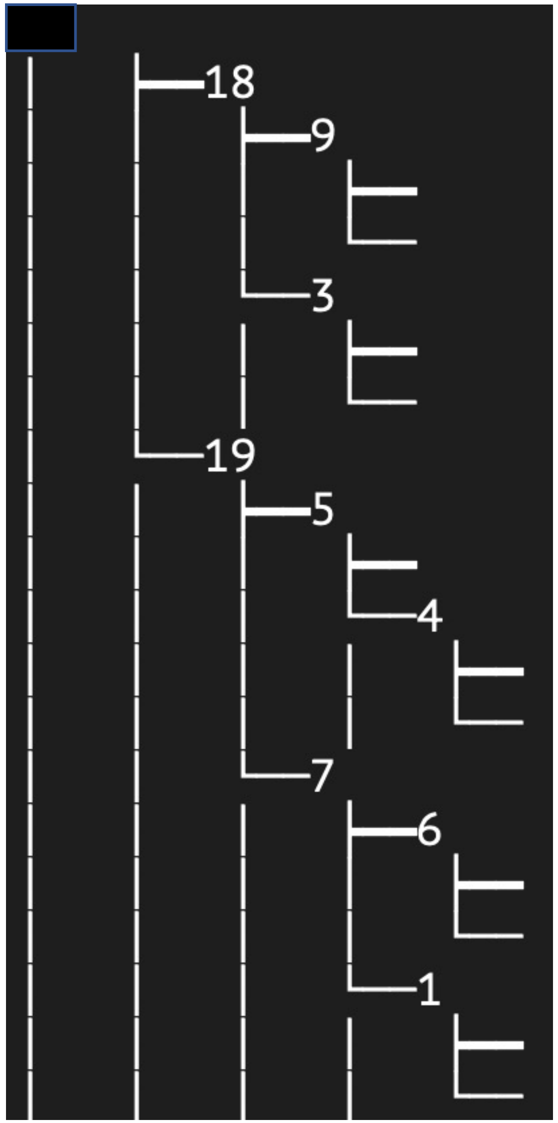
```
│    ├──18
│    │    ├──9
│    │    │    ├──
│    │    │    └──
│    │    └──3
│    │    │    ├──
│    │    │    └──
│    └──19
│    │    ├──5
│    │    │    ├──
│    │    │    └──4
│    │    │    │    ├──
│    │    │    │    └──
│    │    └──7
│    │    │    ├──6
│    │    │    │    ├──
│    │    │    │    └──
│    │    │    └──1
│    │    │    │    ├──
│    │    │    │    └──
```

delete()

We store 25 in a temporary variable so that we can return it

We delete 25

This action disturbs the order of nodes.



[redacted]

```
│    ├──18
│    │    ├──9
│    │    │    ├──
│    │    │    └──
│    │    └──3
│    │    │    ├──
│    │    │    └──
│    └──19
│    │    ├──5
│    │    │    ├──
│    │    │    └──4
│    │    │    │    ├──
│    │    │    │    └──
│    │    └──7
│    │    │    ├──6
│    │    │    │    ├──
│    │    │    │    └──
│    │    │    └──1
│    │    │    │    ├──
│    │    │    │    └──
│    │    │    │
```

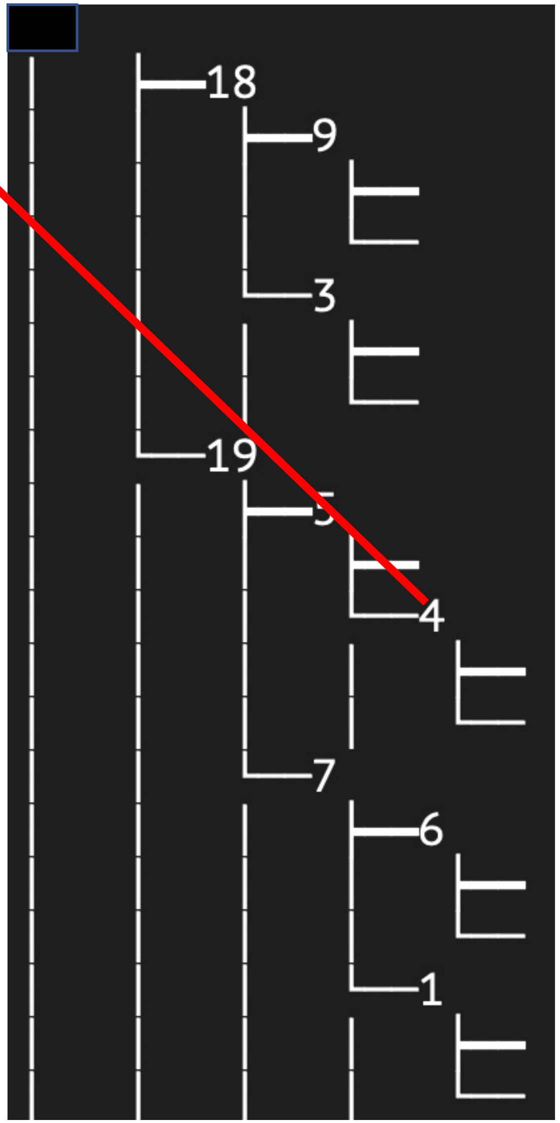
delete()

Can we place the last value in the root?

We store 25 in a temporary variable so that we can return it

We delete 25

This action disturbs the order of nodes.



```
│   ├──18
│   │   ├──9
│   │   │   ├──
│   │   │   └──
│   │   └──3
│   │   │   ├──
│   │   │   └──
│   └──19
│   │   ├──5
│   │   │   ├──
│   │   │   └──4
│   │   │   │   ├──
│   │   │   │   └──
│   │   └──7
│   │   │   ├──6
│   │   │   │   ├──
│   │   │   │   └──
│   │   │   └──1
│   │   │   │   ├──
│   │   │   │   └──
```

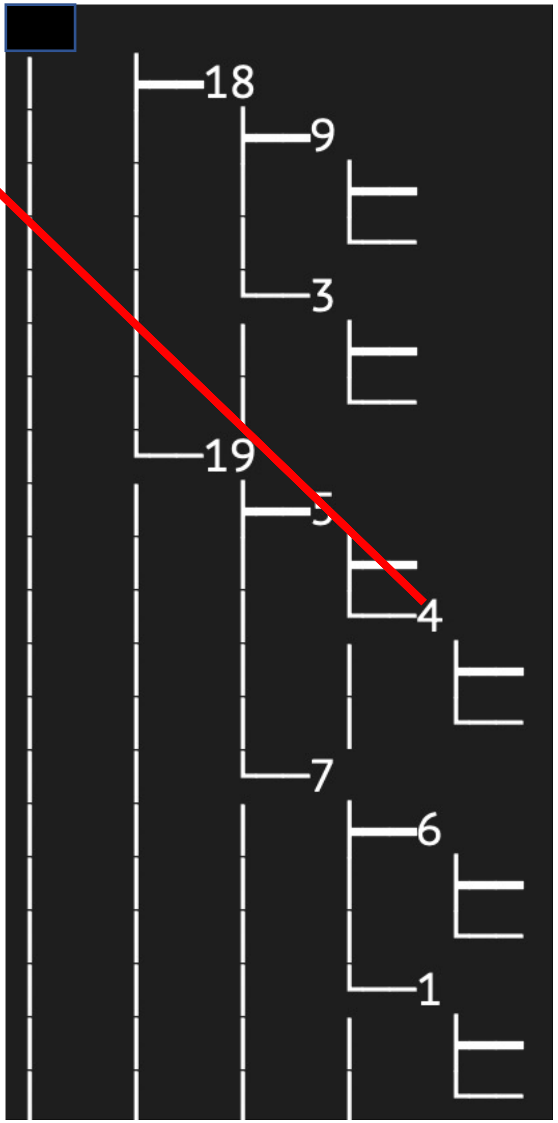
Can we place the last value in the root? No, this will disturb the tree

delete()

We store 25 in a temporary variable so that we can return it

We delete 25

This action disturbs the order of nodes.



```
[redacted]
│    ├──18
│    │    ├──9
│    │    │    ├──
│    │    │    └──
│    │    └──3
│    │    │    ├──
│    │    │    └──
│    └──19
│    │    ├──5
│    │    │    ├──
│    │    │    └──4
│    │    │    │    ├──
│    │    │    │    └──
│    │    └──7
│    │    │    ├──6
│    │    │    │    ├──
│    │    │    │    └──
│    │    │    └──1
│    │    │    │    ├──
│    │    │    │    └──
```

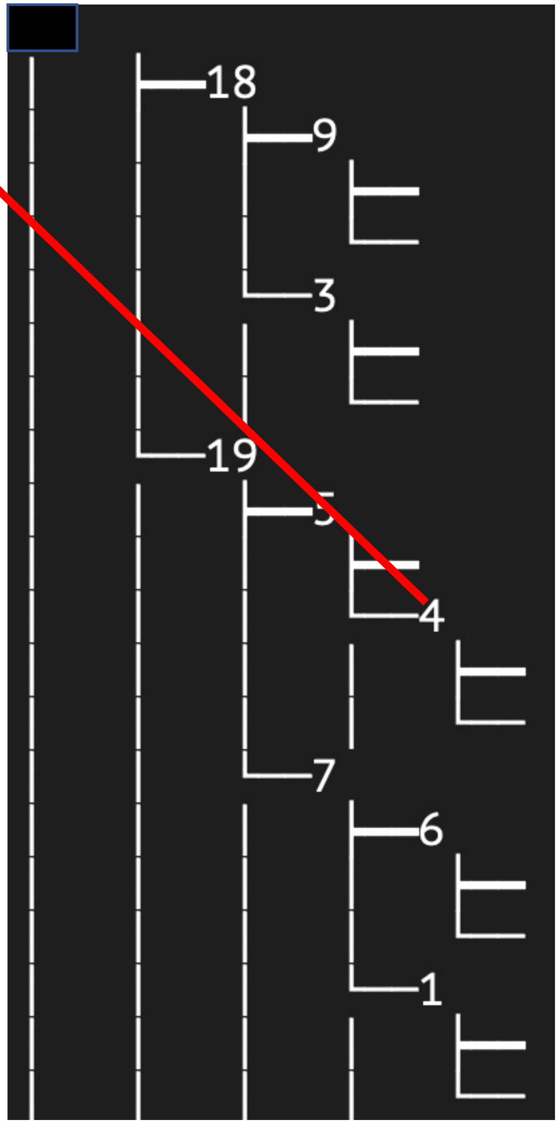
delete()

We remove the last value and place it in a temporary variable LastValue

We store 25 in a temporary variable so that we can return it

We delete 25

This action disturbs the order of nodes.



```
│   ├──18
│   │   ├──9
│   │   │   ├──
│   │   │   └──
│   │   └──3
│   │   │   ├──
│   │   │   └──
│   └──19
│   │   ├──5
│   │   │   ├──
│   │   │   └──4
│   │   │   │   ├──
│   │   │   │   └──
│   │   └──7
│   │   │   ├──6
│   │   │   │   ├──
│   │   │   │   └──
│   │   │   └──1
│   │   │   │   ├──
│   │   │   │   └──
```

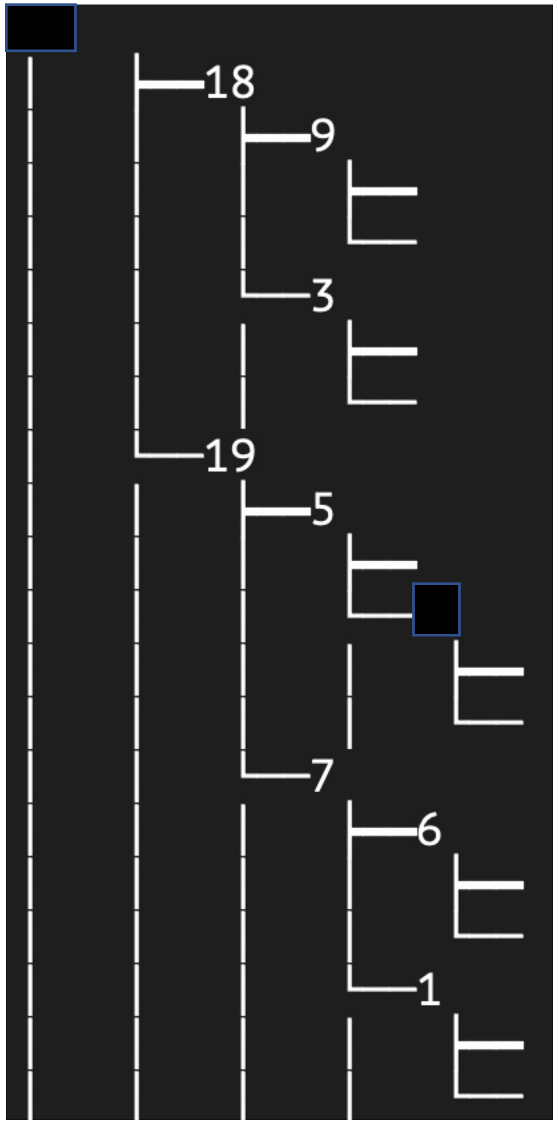
delete()

LastValue = 4

We store 25 in a temporary variable so that we can return it

We delete 25

This action disturbs the order of nodes.



```
│    ├──18
│    │    ├──9
│    │    │    ├──
│    │    │    └──
│    │    └──3
│    │    │    ├──
│    │    │    └──
│    └──19
│    │    ├──5
│    │    │    ├──
│    │    │    └──
│    │    │    │    ├──
│    │    │    │    └──
│    │    └──7
│    │    │    ├──6
│    │    │    │    ├──
│    │    │    │    └──
│    │    │    └──1
│    │    │    │    ├──
│    │    │    │    └──
```

delete()

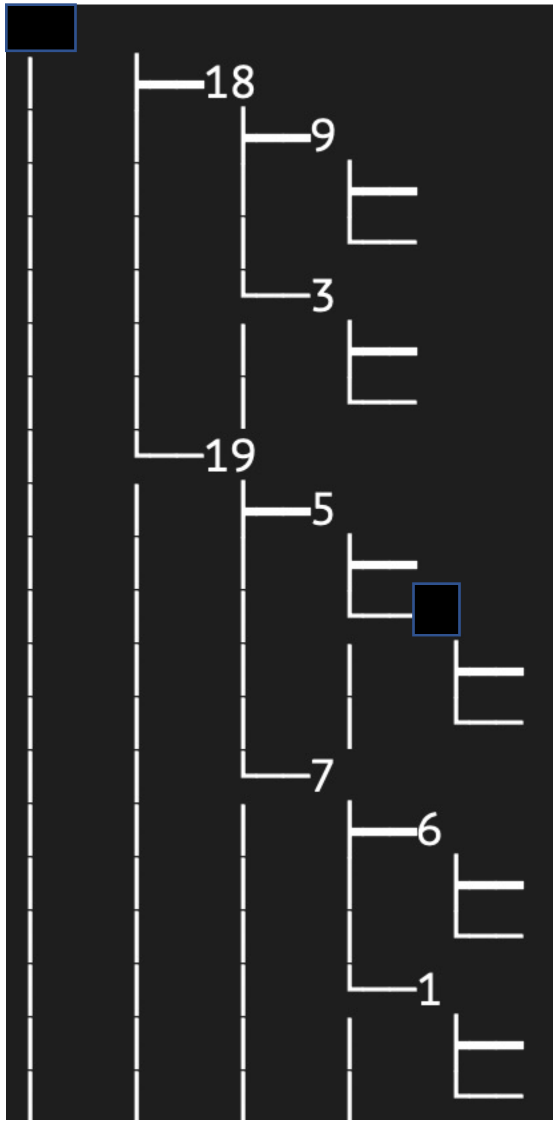
LastValue = 4

We store 25 in a temporary variable so that we can return it

We delete 25

This action disturbs the order of nodes

To reorder the nodes we must take certain actions



```
[redacted]
│   ├──18
│   │   ├──9
│   │   │   ├──
│   │   │   └──
│   │   └──3
│   │   │   ├──
│   │   │   └──
│   └──19
│   │   ├──5
│   │   │   ├──
│   │   │   └──[redacted]
│   │   │   │   ├──
│   │   │   │   └──
│   │   └──7
│   │   │   ├──6
│   │   │   │   ├──
│   │   │   │   └──
│   │   │   └──1
│   │   │   │   ├──
│   │   │   │   └──
```

delete()

LastValue = 4

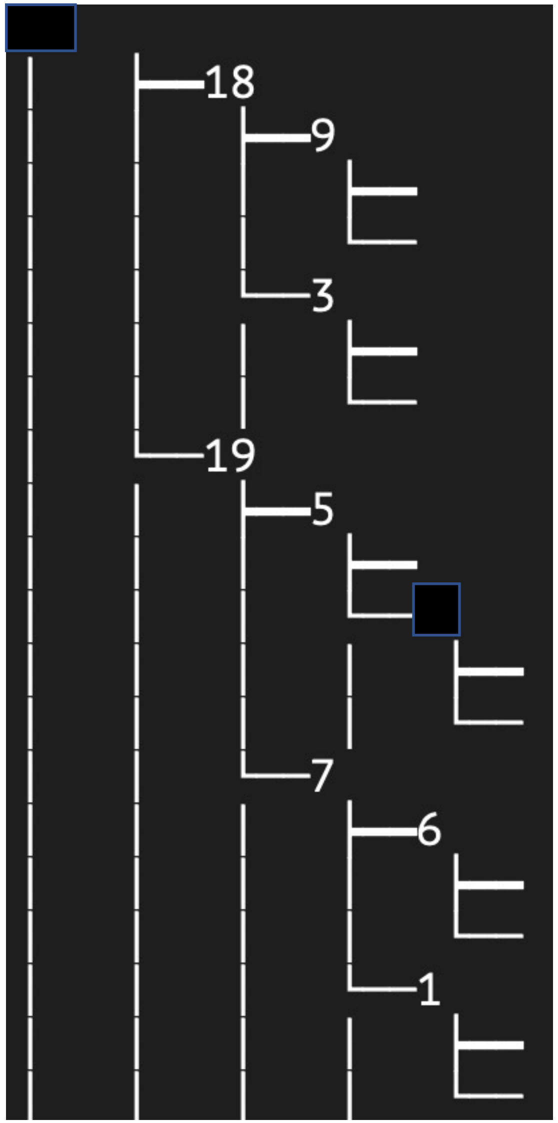
We store 25 in a temporary variable so that we can return it

We delete 25

This action disturbs the order of nodes

To reorder the nodes we must take certain actions

Root has two children: 18 and 19.



```
[redacted]
│    ├──18
│    │    ├──9
│    │    │    ├──
│    │    │    └──
│    │    └──3
│    │    │    ├──
│    │    │    └──
│    └──19
│    │    ├──5
│    │    │    ├──
│    │    │    └──[redacted]
│    │    │    │    ├──
│    │    │    │    └──
│    │    └──7
│    │    │    ├──6
│    │    │    │    ├──
│    │    │    │    └──
│    │    │    └──1
│    │    │    │    ├──
│    │    │    │    └──
```

delete()

LastValue = 4

We store 25 in a temporary variable so that we can return it

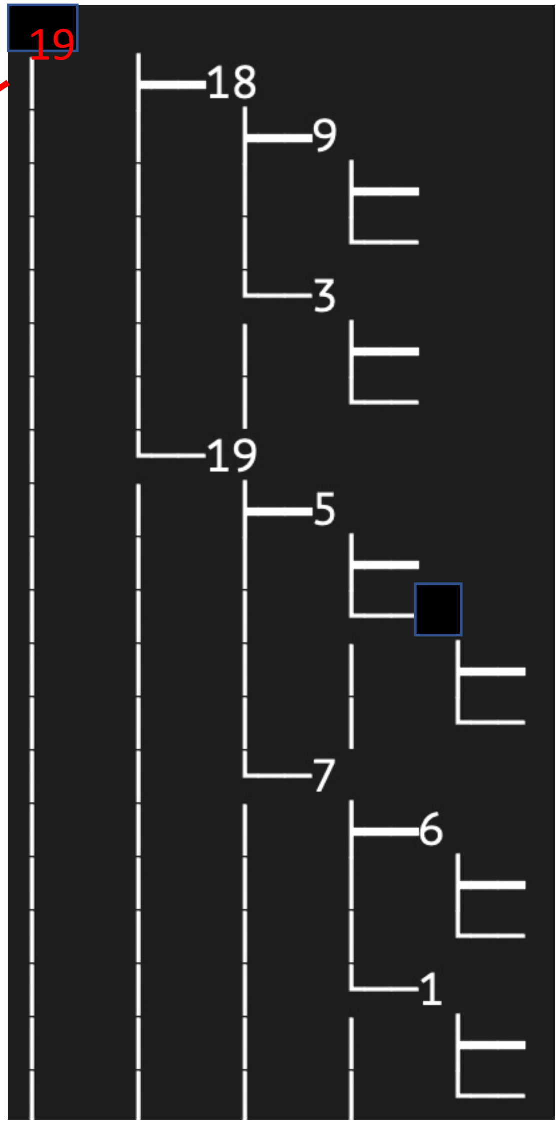
We delete 25

This action disturbs the order of nodes

To reorder the nodes we must take certain actions

Root has two children: 18 and 19.

Since 19 is larger, we copy 19 to the root



```
19
│   ├──18
│   │   ├──9
│   │   │   ├──
│   │   │   └──
│   │   └──3
│   │   │   ├──
│   │   │   └──
│   └──19
│   │   ├──5
│   │   │   ├──
│   │   │   └──
│   │   │   │   ├──
│   │   │   │   └──
│   │   └──7
│   │   │   ├──6
│   │   │   │   ├──
│   │   │   │   └──
│   │   │   └──1
│   │   │   │   ├──
│   │   │   │   └──
```

delete()

LastValue = 4

19 > 4 so no need for an action

We store 25 in a temporary variable so that we can return it

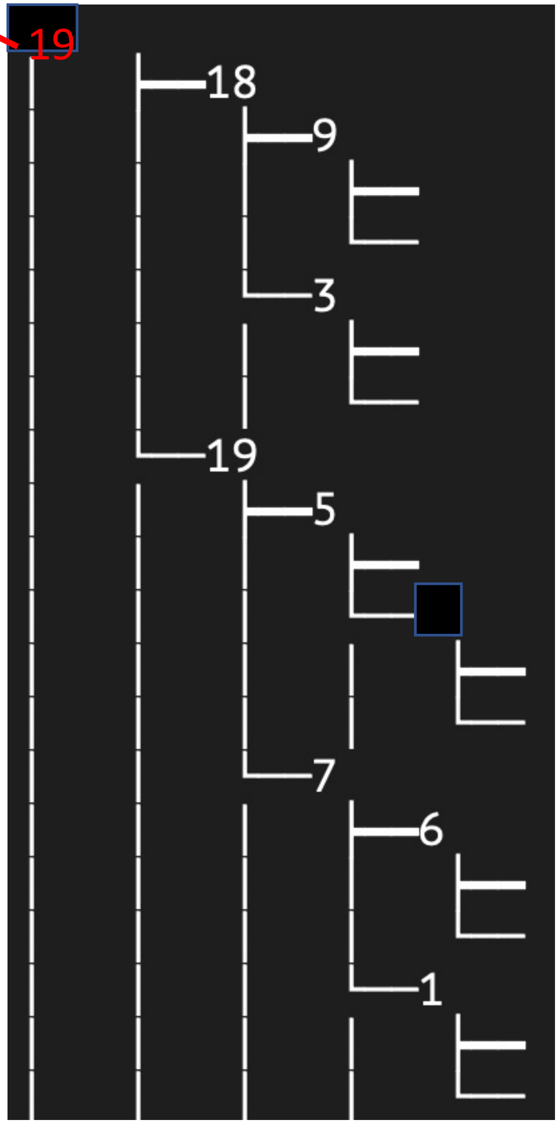
We delete 25

This action disturbs the order of nodes

To reorder the nodes we must take certain actions

Root has two children: 18 and 19.

Since 19 is larger, we copy 19 to the root



```
19
│    ├──18
│    │    ├──9
│    │    │    ├──
│    │    │    └──
│    │    └──3
│    │    │    ├──
│    │         └──
│    └──19
│    │    ├──5
│    │    │    ├──
│    │    │    └──
│    │    │    │    ├──
│    │    │         └──
│    │    └──7
│    │    │    ├──6
│    │    │    │    ├──
│    │    │    │    └──
│    │    │    └──1
│    │    │    │    ├──
│    │    │    │    └──
```

delete()

LastValue = 4

We store 25 in a temporary variable so that we can return it

We delete 25

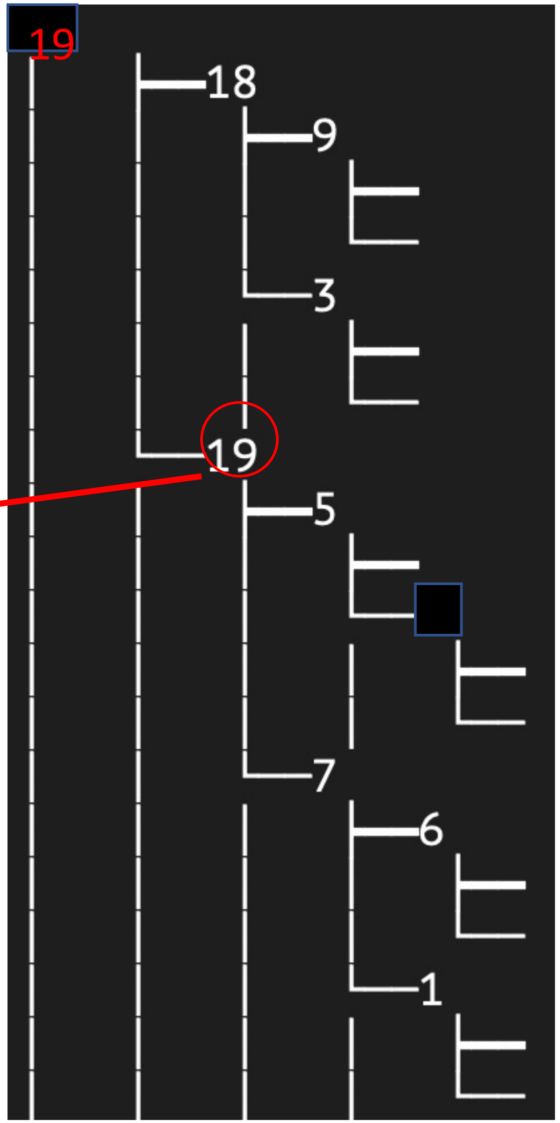
This action disturbs the order of nodes

To reorder the nodes we must take certain actions

Root has two children: 18 and 19.

Since 19 is larger, we copy 19 to the root

We move to 19



delete()

LastValue = 4

We store 25 in a temporary variable so that we can return it

We delete 25

This action disturbs the order of nodes

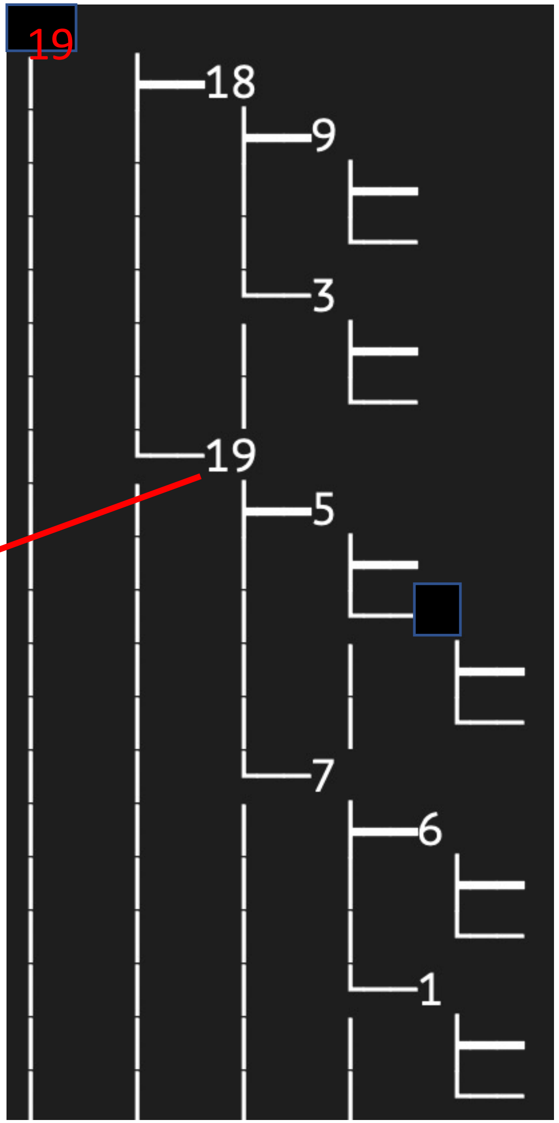
To reorder the nodes we must take certain actions

Root has two children: 18 and 19.

Since 19 is larger, we copy 19 to the root

We move to 19

This 19 is redundant and should be overwritten



```
19
│    ├──18
│    │    ├──9
│    │    │    ├──
│    │    │    └──
│    │    └──3
│    │    │    ├──
│    │    │    └──
│    └──19
│    │    ├──5
│    │    │    ├──
│    │    │    └──
│    │    │    │    ├──
│    │    │    │    └──
│    │    └──7
│    │    │    ├──6
│    │    │    │    ├──
│    │    │    │    └──
│    │    │    └──1
│    │    │    │    ├──
│    │    │    │    └──
```

delete()

LastValue = 4

We store 25 in a temporary variable so that we can return it

We delete 25

This action disturbs the order of nodes

To reorder the nodes we must take certain actions

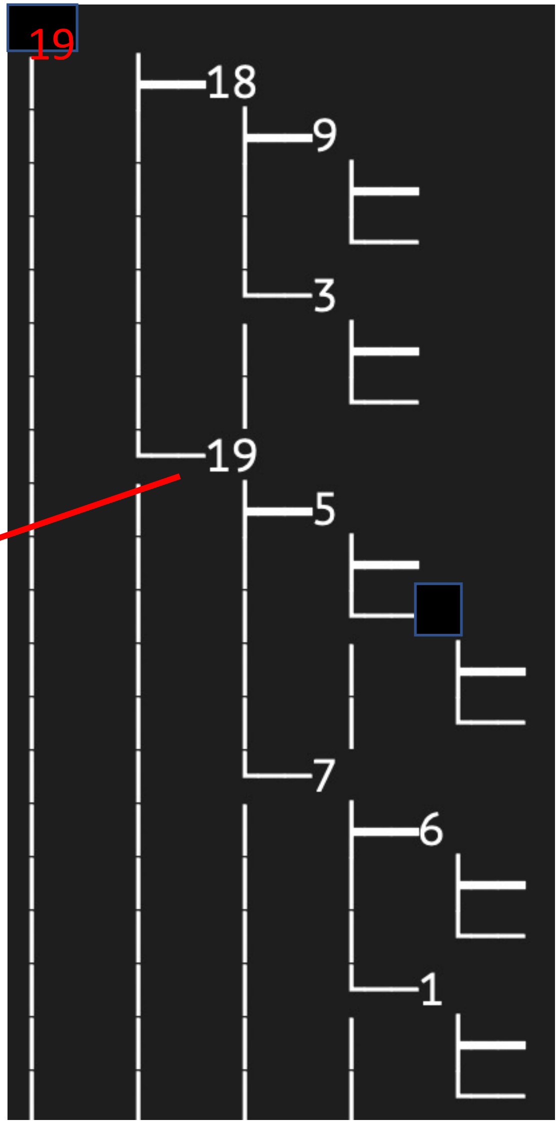
Root has two children: 18 and 19.

Since 19 is larger, we copy 19 to the root

We move to 19

This 19 is redundant and should be overwritten

19 has two children: 5 and 7



delete()

LastValue = 4

We store 25 in a temporary variable so that we can return it

We delete 25

This action disturbs the order of nodes

To reorder the nodes we must take certain actions

Root has two children: 18 and 19.

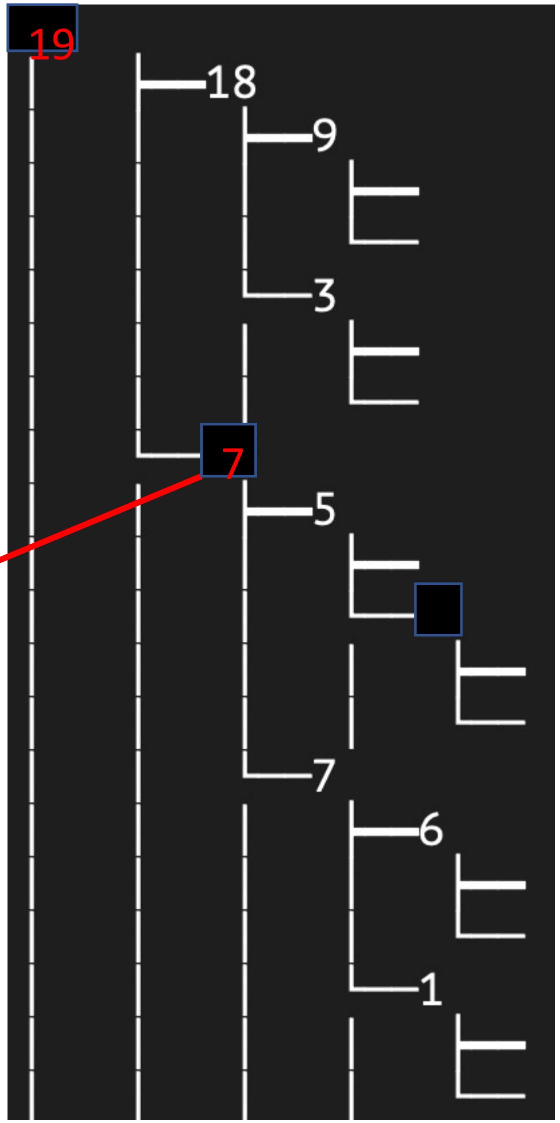
Since 19 is larger, we copy 19 to the root

We move to 19

This 19 is redundant and should be overwritten

19 has two children: 5 and 7.

Since 7 is larger, we copy 7 to 19



delete()

LastValue = 4

We store 25 in a temporary variable so that we can return it

We delete 25

This action disturbs the order of nodes

To reorder the nodes we must take certain actions

Root has two children: 18 and 19.

Since 19 is larger, we copy 19 to the root

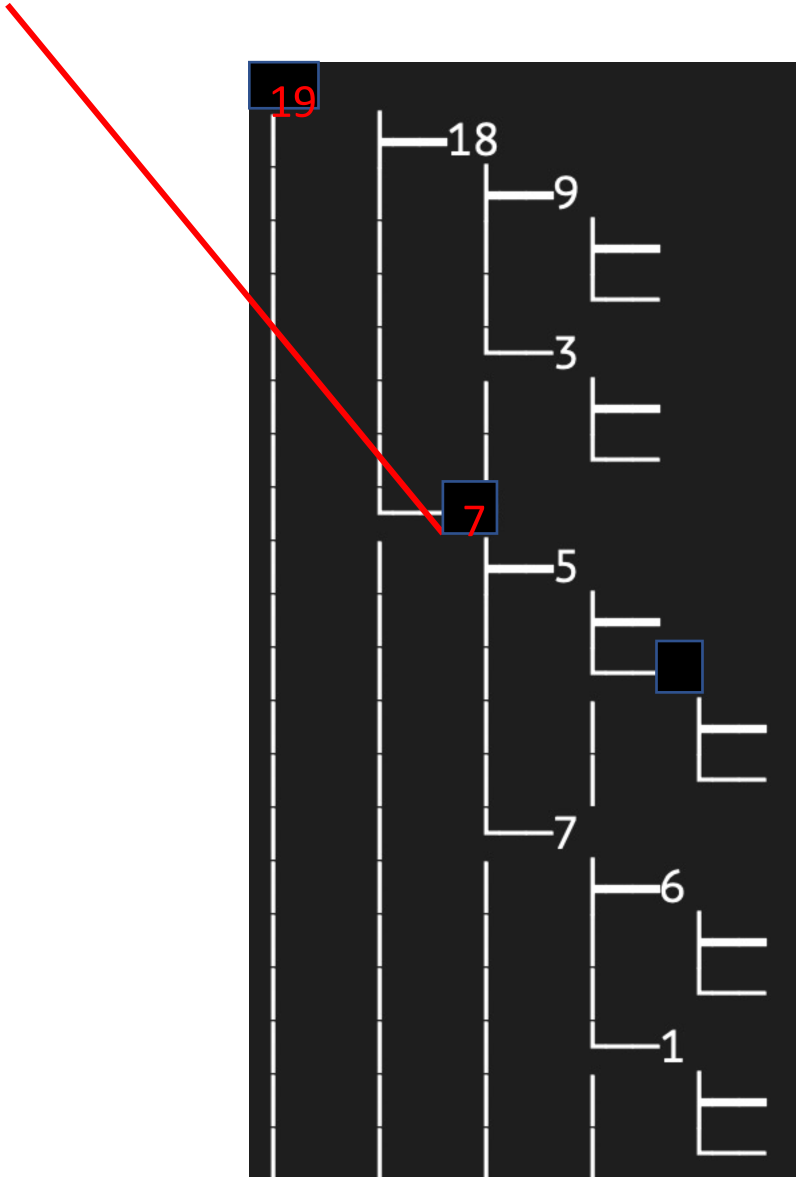
We move to 19

This 19 is redundant and should be overwritten

19 has two children: 5 and 7.

Since 7 is larger, we copy 7 to 19

7 > 4 so no need for an action



delete()

LastValue = 4

We store 25 in a temporary variable so that we can return it

We delete 25

This action disturbs the order of nodes

To reorder the nodes we must take certain actions

Root has two children: 18 and 19.

Since 19 is larger, we copy 19 to the root

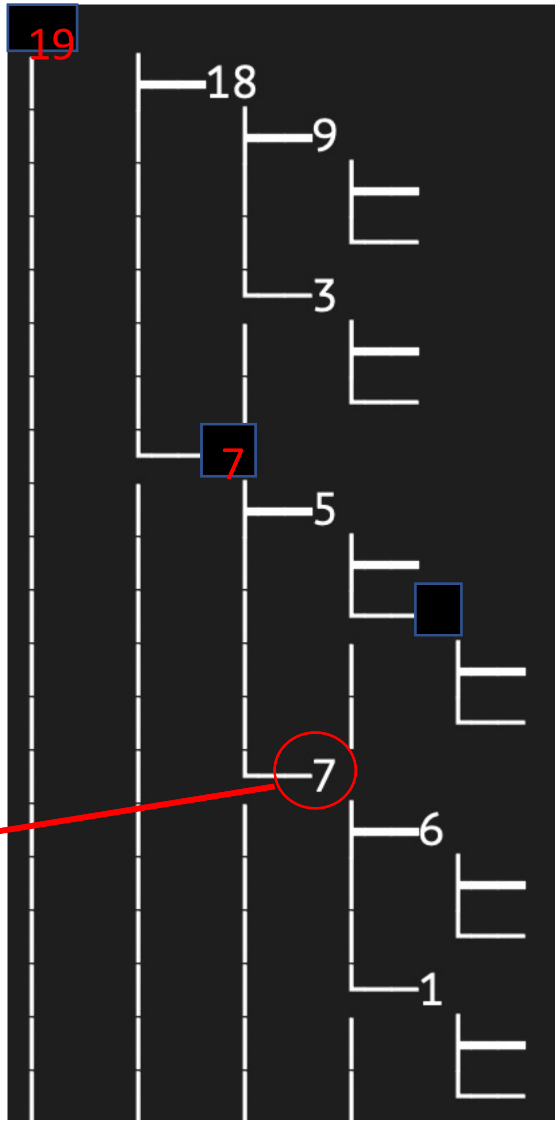
We move to 19

This 19 is redundant and should be overwritten

19 has two children: 5 and 7.

Since 7 is larger, we copy 7 to 19

We move to 7



delete()

LastValue = 4

We store 25 in a temporary variable so that we can return it

We delete 25

This action disturbs the order of nodes

To reorder the nodes we must take certain actions

Root has two children: 18 and 19.

Since 19 is larger, we copy 19 to the root

We move to 19

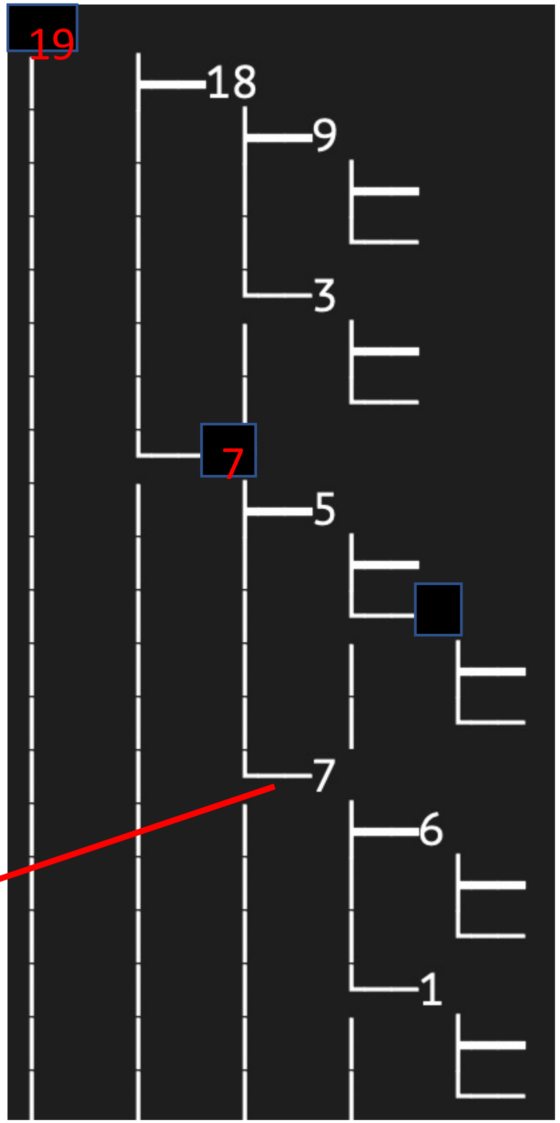
This 19 is redundant and should be overwritten

19 has two children: 5 and 7.

Since 7 is larger, we copy 7 to 19

We move to 7

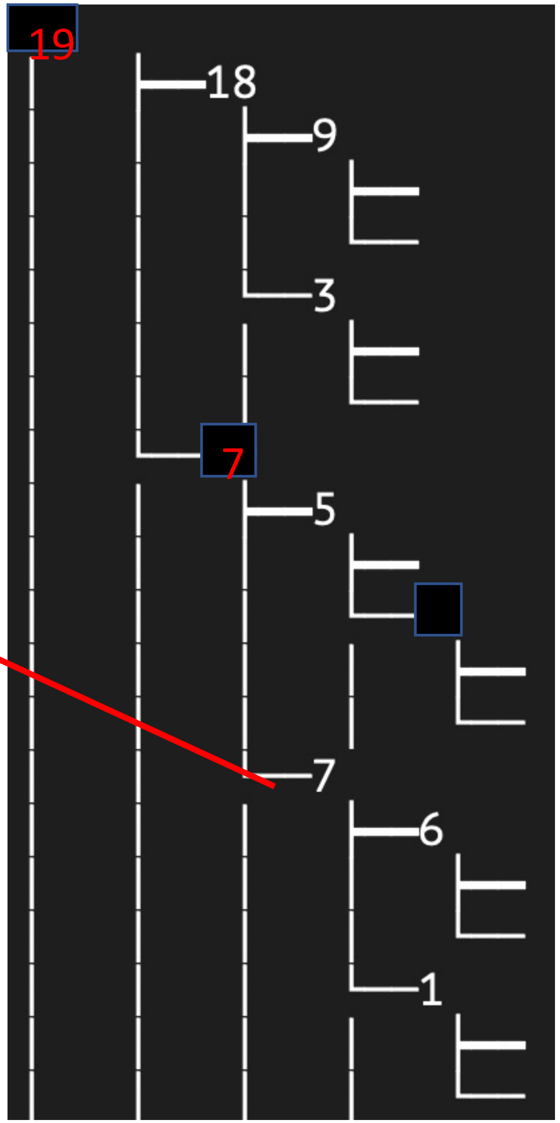
This 7 is redundant and needs to be removed



delete()

LastValue = 4

7 has two children: 6 and 1

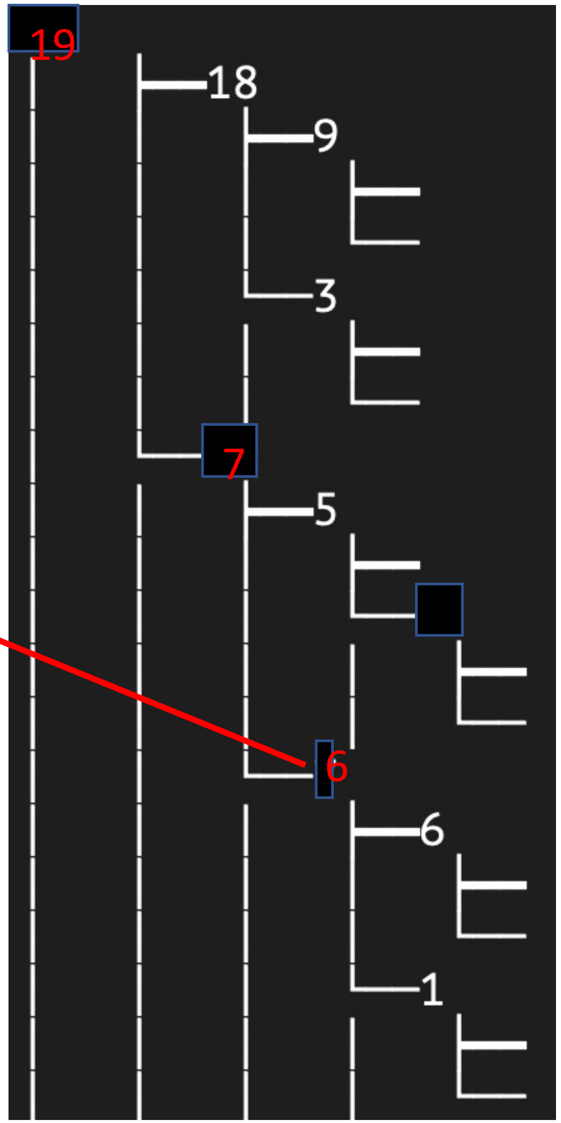


delete()

LastValue = 4

7 has two children: 6 and 1

Since 6 is larger, we copy 6 to 7



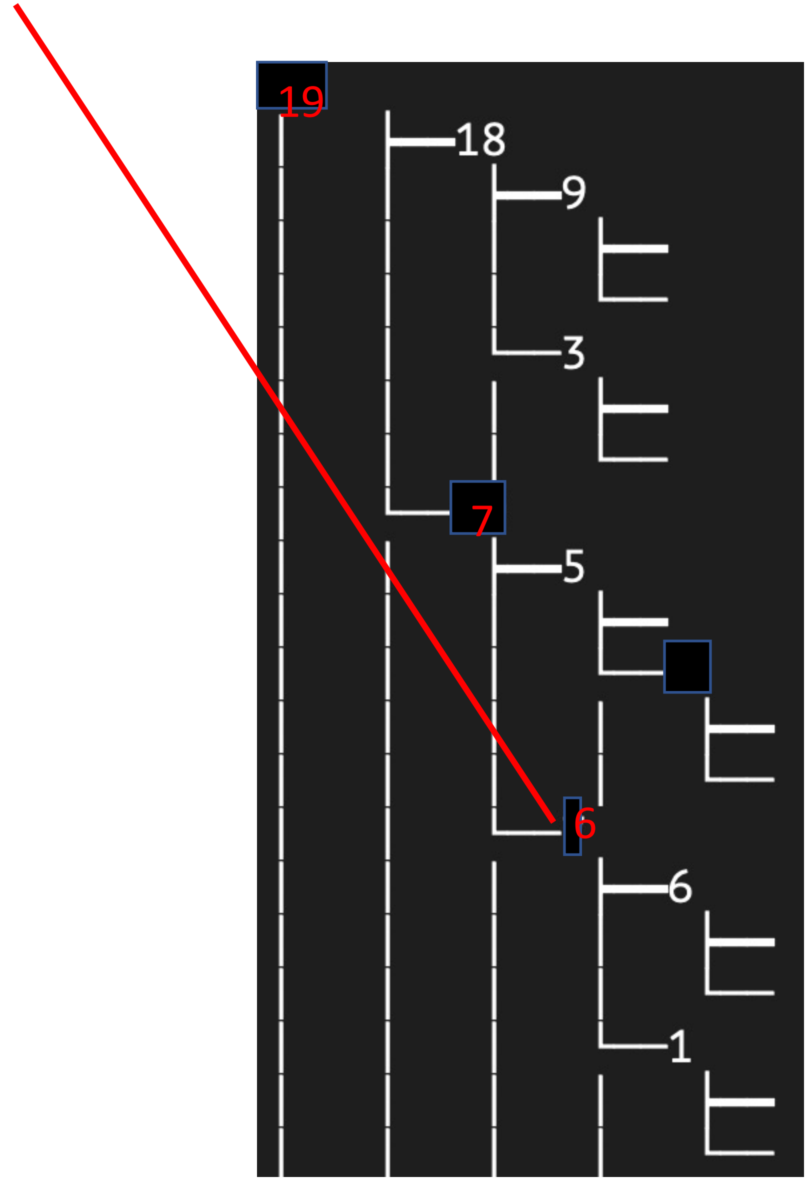
delete()

LastValue = 4

6 > 4 so no need for an action

7 has two children: 6 and 1

Since 6 is larger, we copy 6 to 7



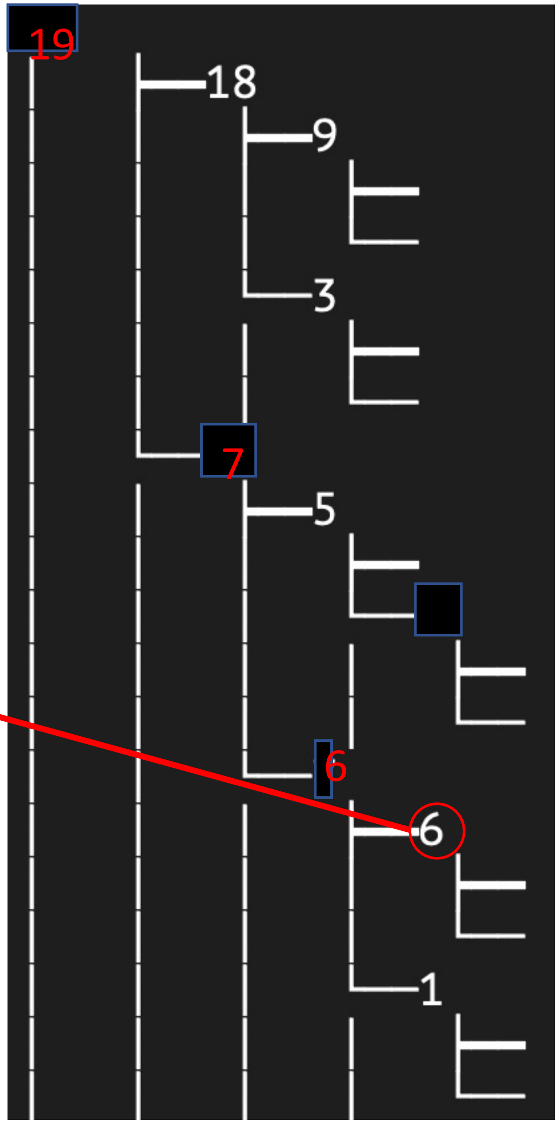
delete()

LastValue = 4

7 has two children: 6 and 1

Since 6 is larger, we copy 6 to 7

We move to 6



delete()

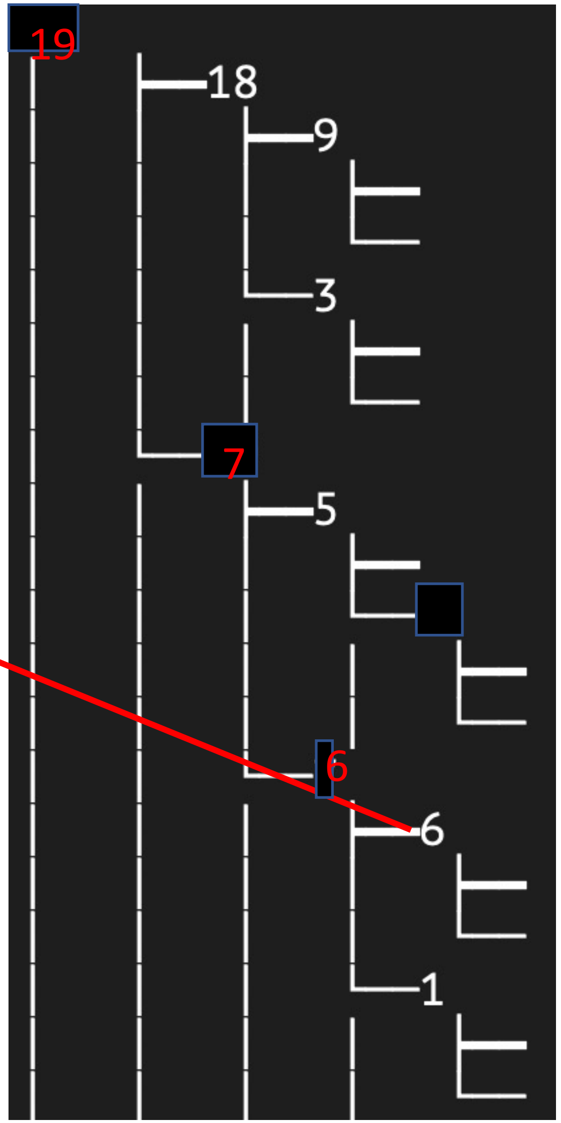
LastValue = 4

7 has two children: 6 and 1

Since 6 is larger, we copy 6 to 7

We move to 6

This 6 is redundant and needs to be removed.



delete()

LastValue = 4

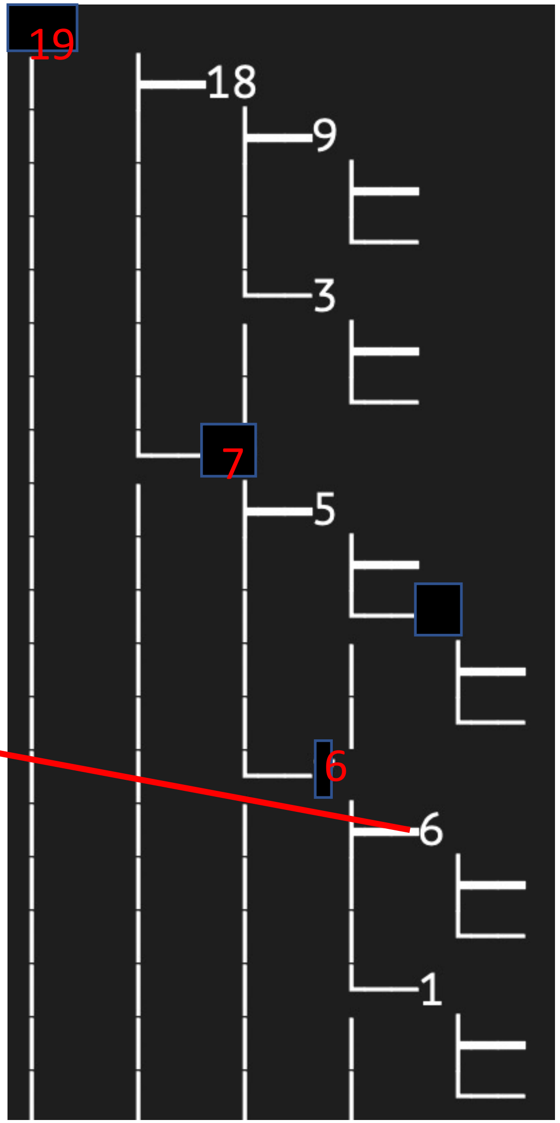
7 has two children: 6 and 1

Since 6 is larger, we copy 6 to 7

We move to 6

This 6 is redundant and needs to be removed.

6 has no children



delete()

7 has two children: 6 and 1

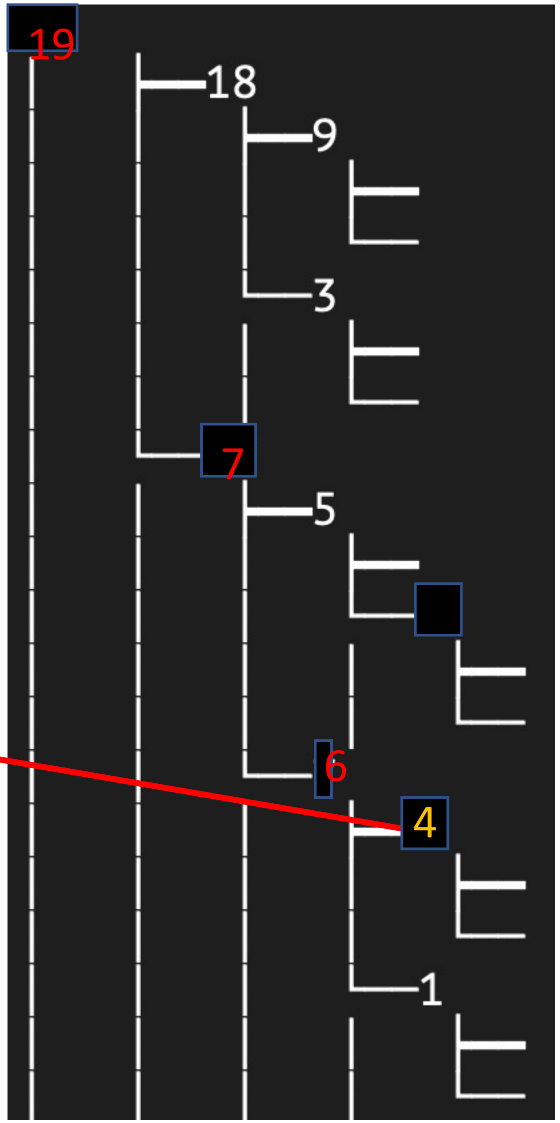
Since 6 is larger, we copy 6 to 7

We move to 6

This 6 is redundant and needs to be removed.

6 has no children

We replace 6 with LastValue



delete()

Remark:

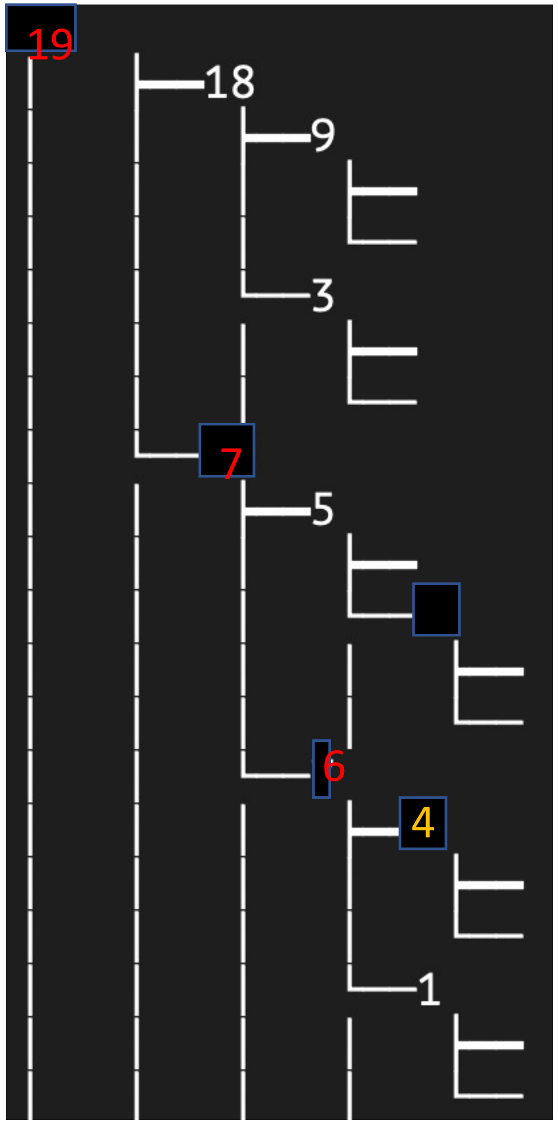
As we move forward level-to-level, we compare node’s value to LastValue

If it turns out that LastValue is larger than node’s value, we swap:

tmp = node’s value

node’s value = LastValue

LastValue =tmp



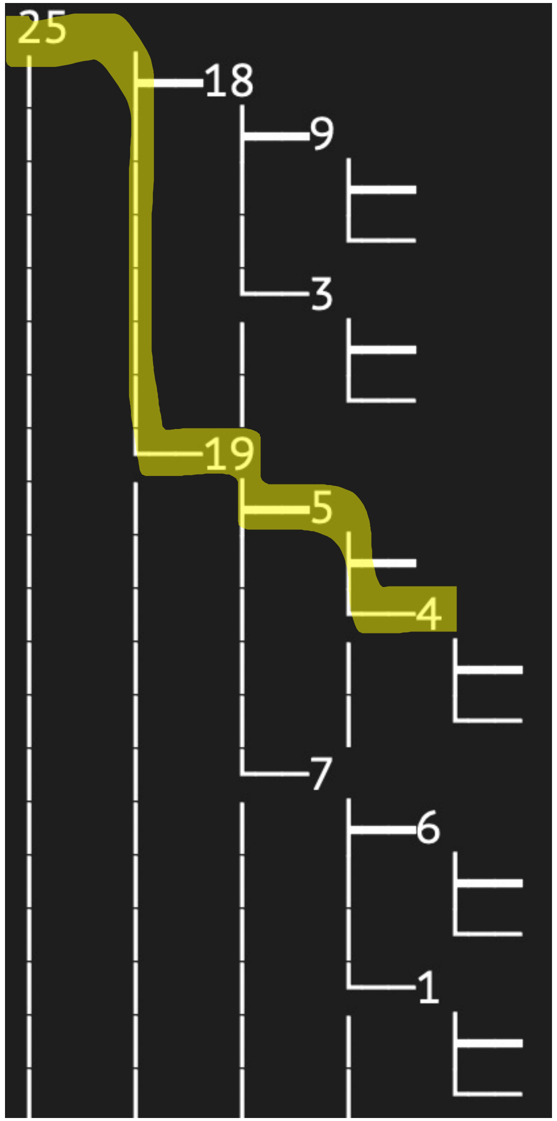
delete()

Remark:

For accessing the last value, we take this path using route(Size) method

Size is the number of nodes when delete() method was called

The time complexity of route() method is log(n)



Time Complexity of delete()

- Time complexity of delete() is O(log(n))