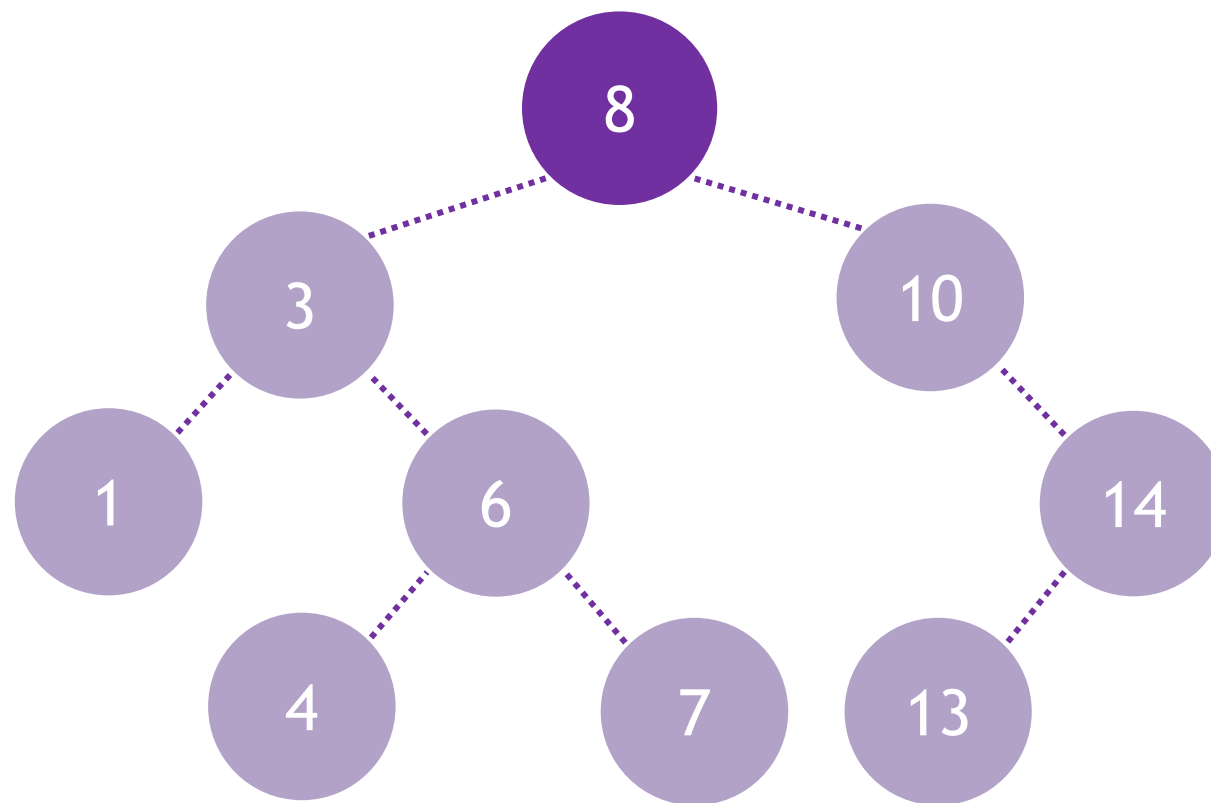


# Binárny vyhľadávací strom



# OBSAH

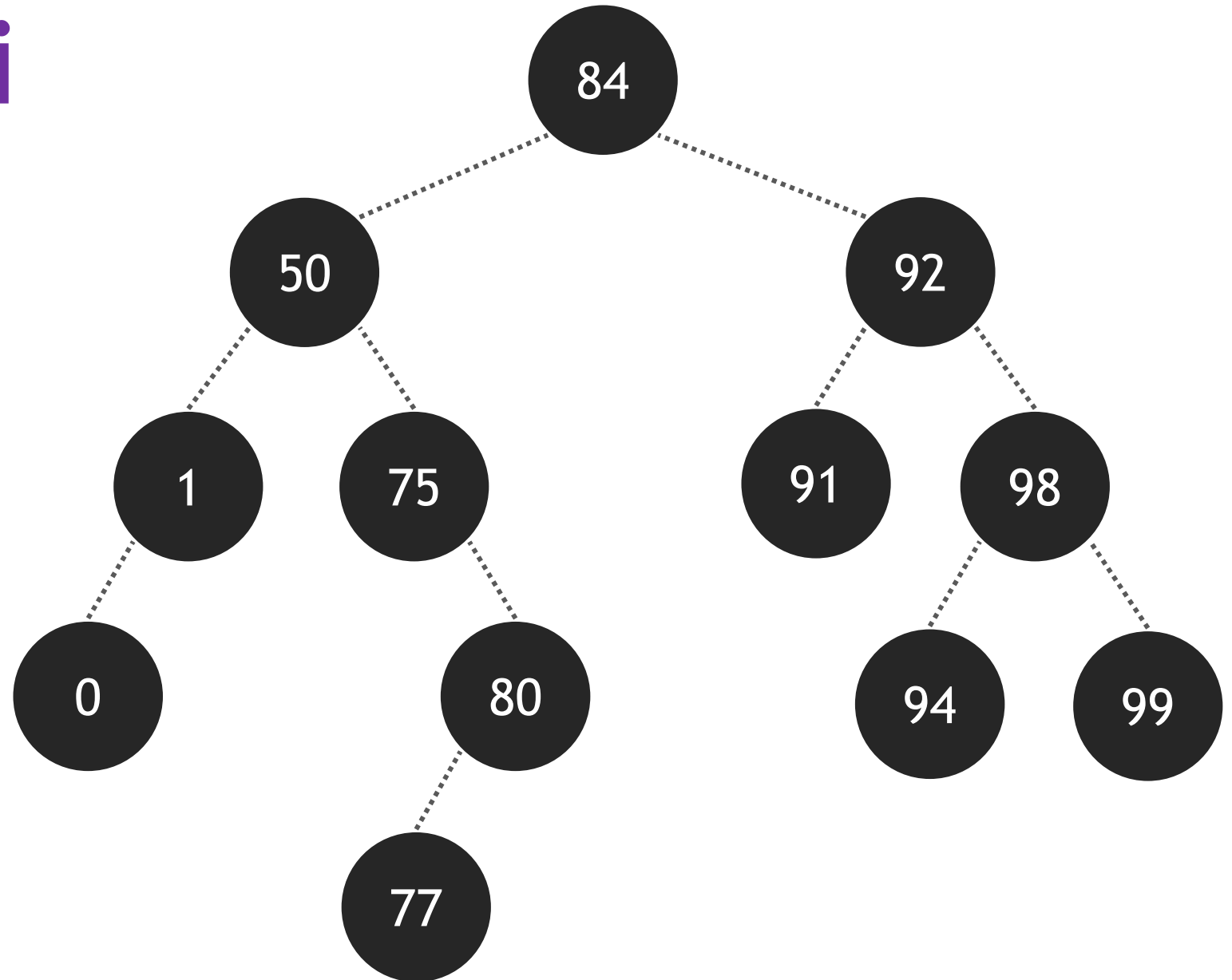
## Binárny vyhľadávací strom (BST)

- Vlastnosti a reprezentácia
- Operácie:
  - Pridávanie uzlov
  - Prechody stromom (BFS,DFS)
  - Vyhľadávanie uzlov
  - Vymazávanie uzlov

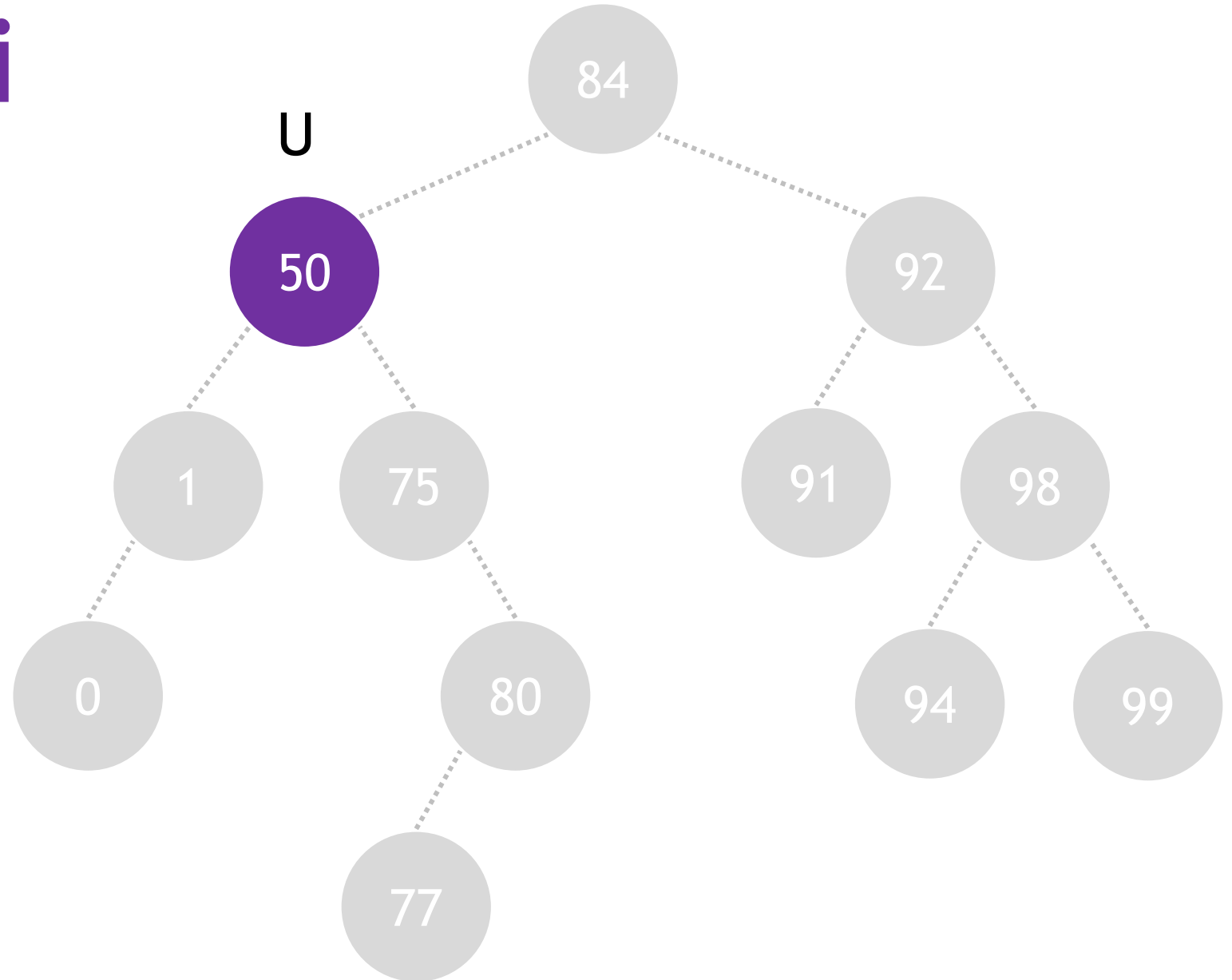
# Vlastnosti

- BST je dátová štruktúra, ktorej uzly sú uložené v takom usporiadaní, aby ich bolo možné efektívne vyhľadávať.
- Pre ľubovoľný uzol  $U$  v BST platí, že:
  - Hodnota  $U$  je väčšia ako všetky hodnoty v ľavom podstrome.
  - Hodnota  $U$  je menšia ako všetky hodnoty v pravom podstrome.
- BST obsahuje len unikátne hodnoty uzlov.

# Vlastnosti

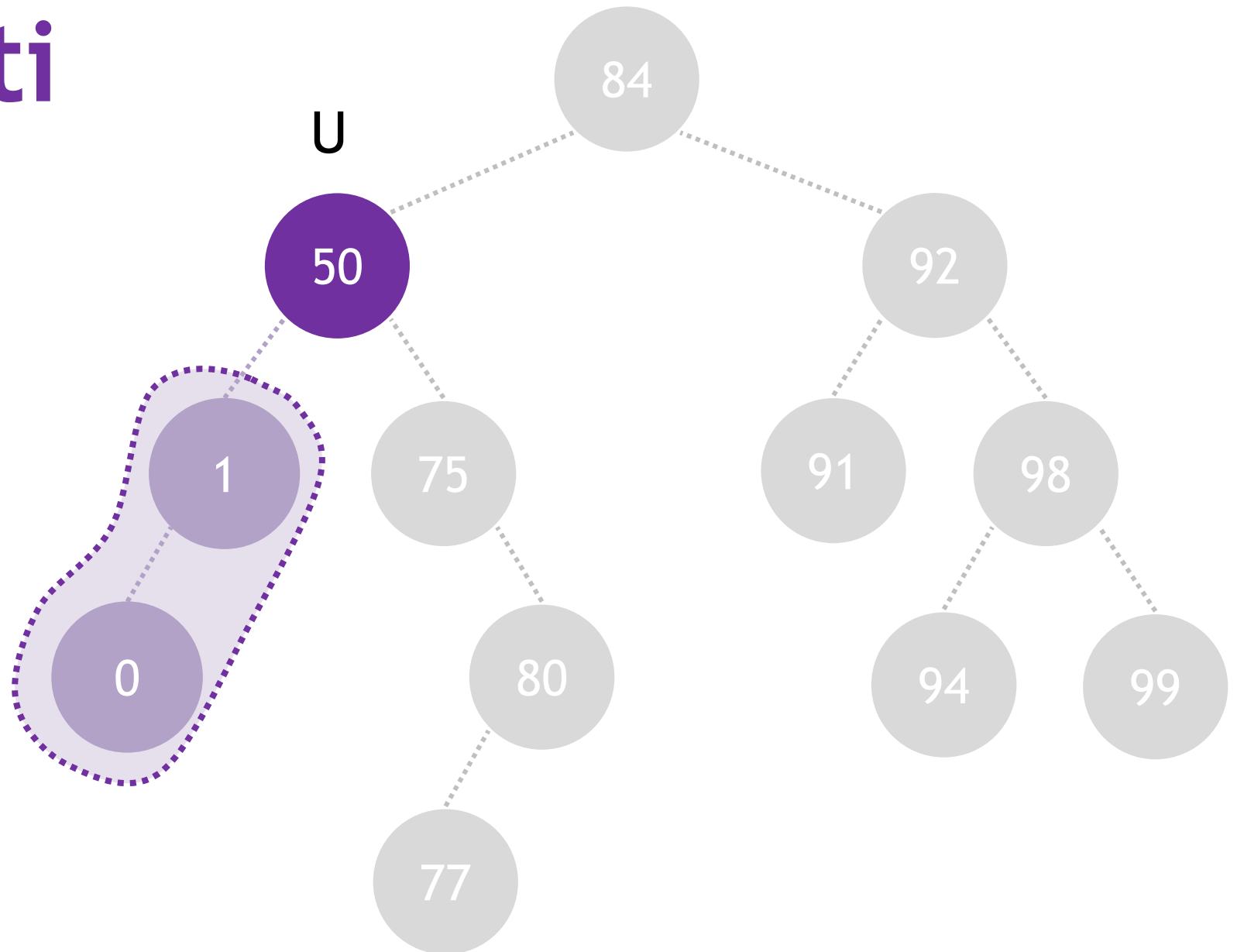


# Vlastnosti



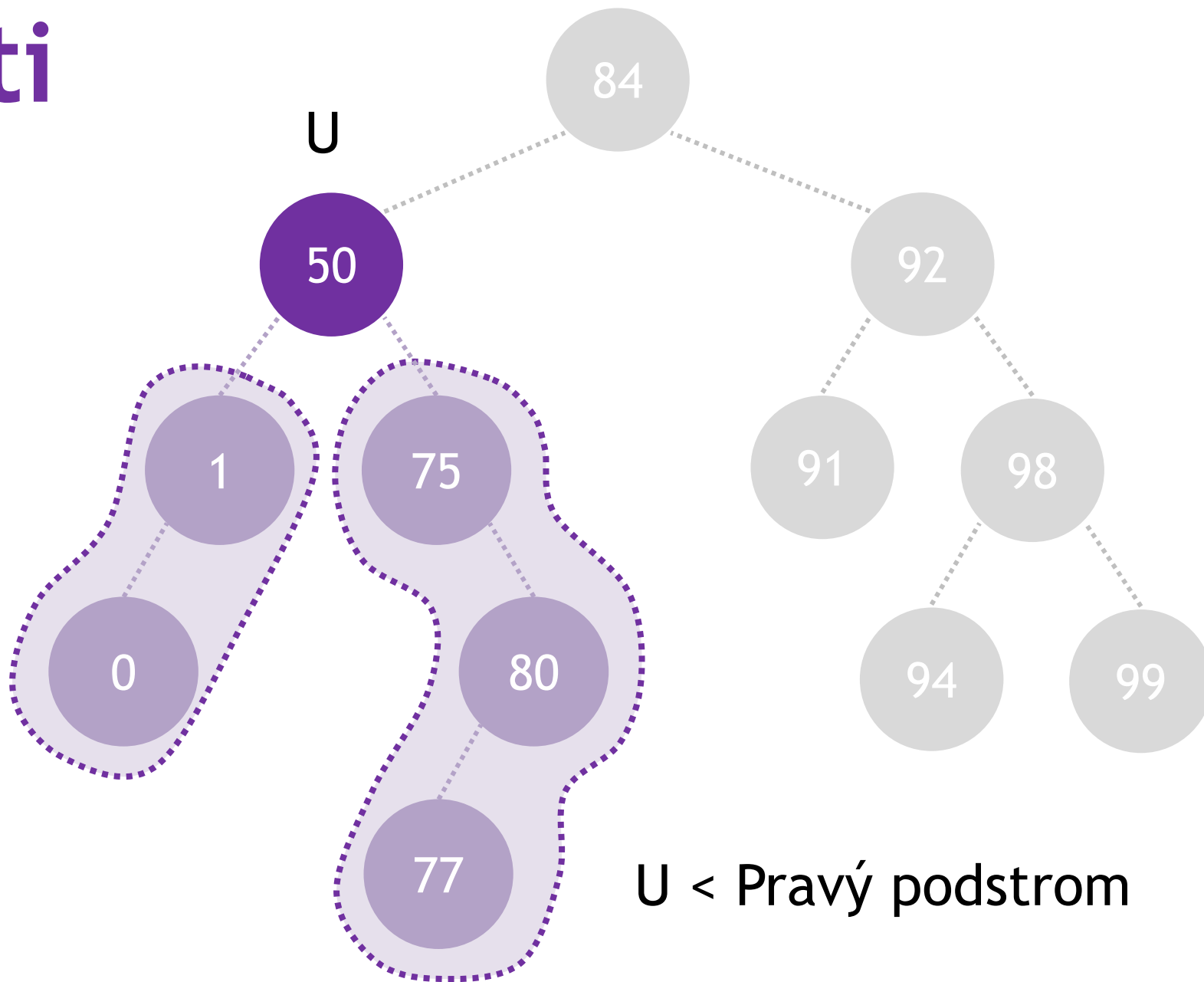
# Vlastnosti

$U > \text{Ľavý podstrom}$



# Vlastnosti

$U > \text{Ľavý podstrom}$

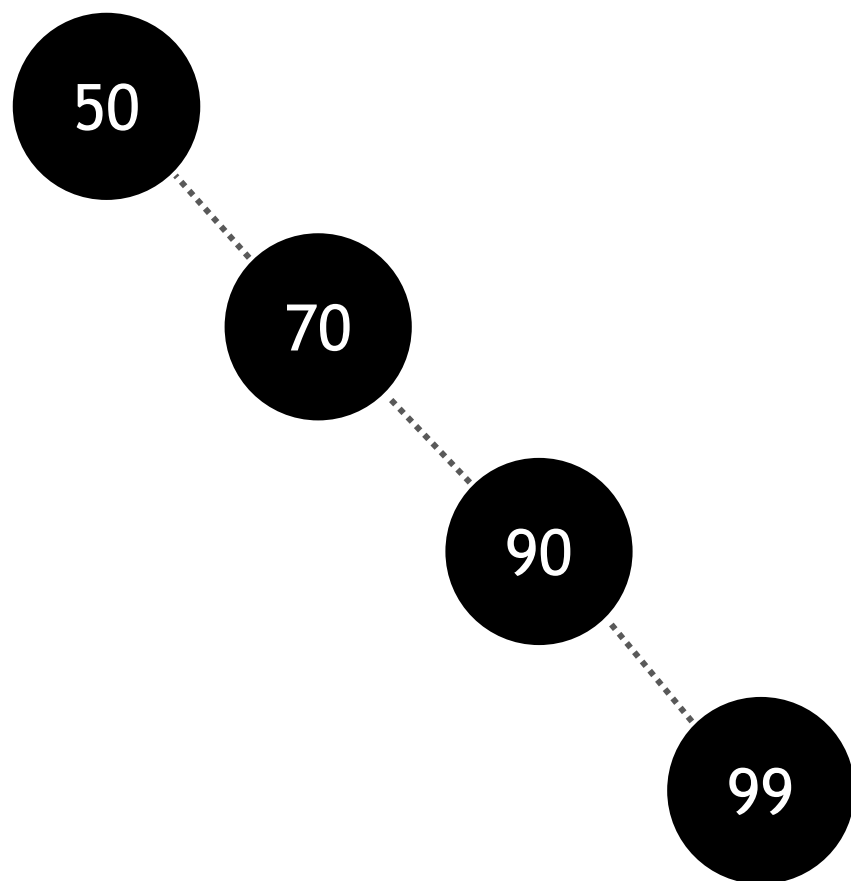


# Vlastnosti

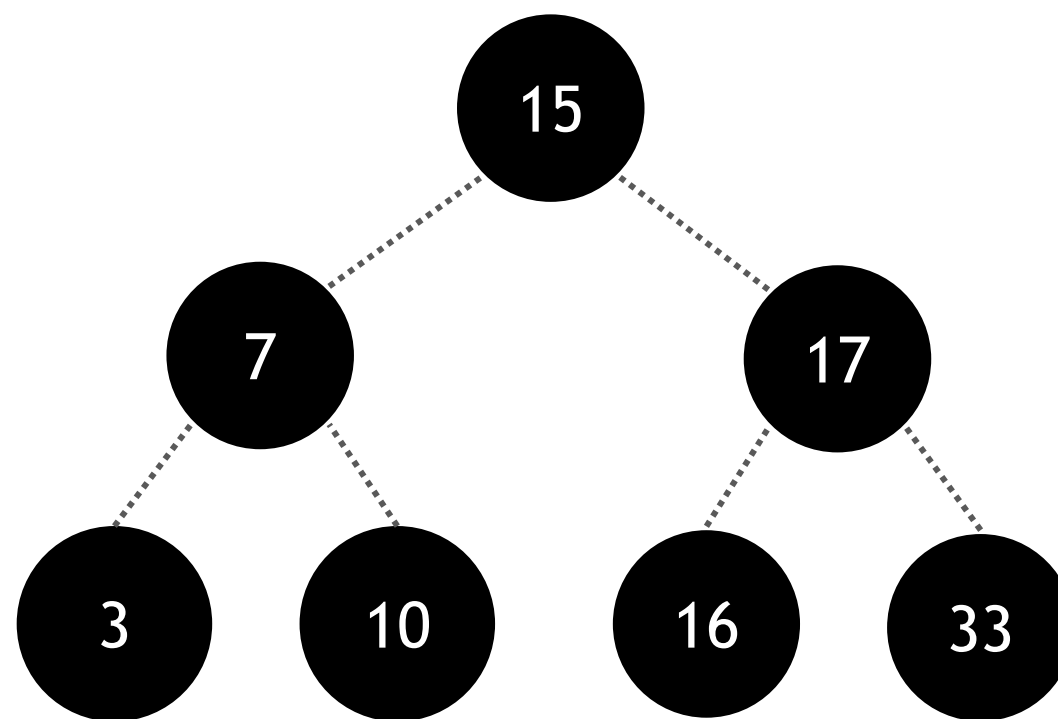
- Rýchlosť vyhľadávania uzlov závisí od toho, do akej miery je strom vyvážený.
- Vo vyváženom strome je rozdiel hĺbok ľubovoľných dvoch listov maximálne 1.
- Časová zložitosť vyhľadávania závisí od výšky stromu.
- Zložitosť vyhľadávania uzla v BST:
  - priemerná:  $O(\log n)$
  - worst case:  $O(n)$



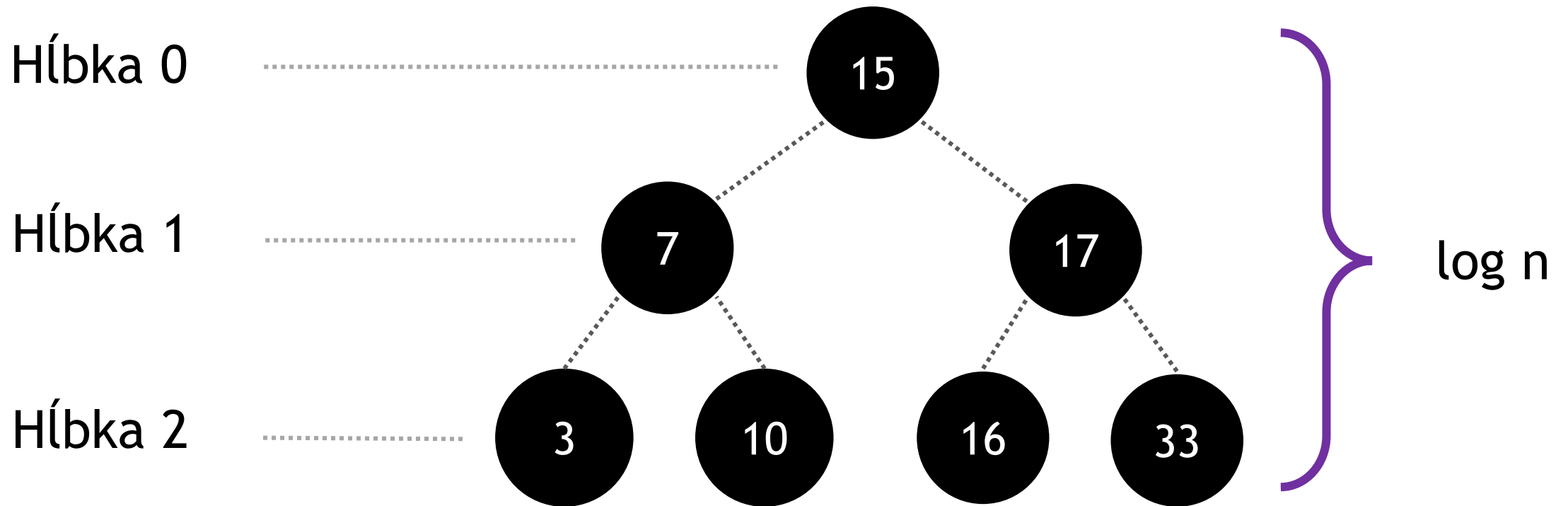
Nevyvážený strom



Vyvážený strom



# Híbká stromu/uzlov



# Reprezentácia uzla

```
struct Node {  
  
    int value; // hodnota uzla  
  
    // smernik na laveho potomka  
    // vsetky uzly v podstromi s korenom v uzle 'smaller'  
    // maju mensie hodnoty ako 'value'  
    Node* smaller;  
  
    // smernik na praveho potomka  
    // vsetky uzly v podstromi s korenom v uzle 'greater'  
    // maju vacsie hodnoty ako 'value'  
    Node* greater;  
};
```

# Reprezentácia BST

```
struct BinarySearchTree {  
    Node *root; // koren BST  
};
```

# Pridávanie uzlov

20,10,70,0,15,50,80,18,55



✓  
20, 10, 70, 0, 15, 50, 80, 18, 55

koreň

20

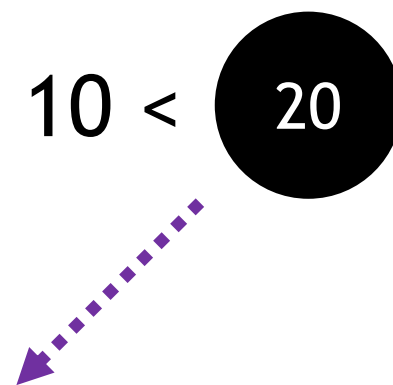
✓  
20, 10, 70, 0, 15, 50, 80, 18, 55  
↑

20

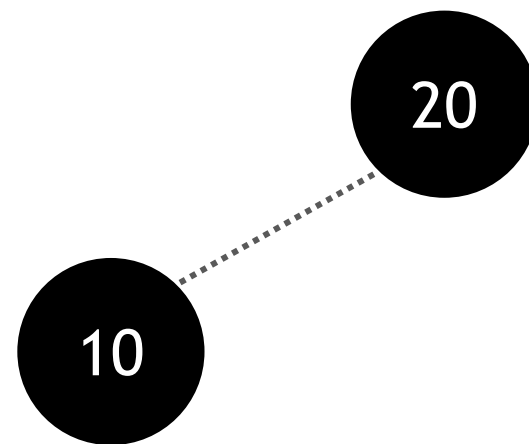


✓  
20, 10, 70, 0, 15, 50, 80, 18, 55  
↑

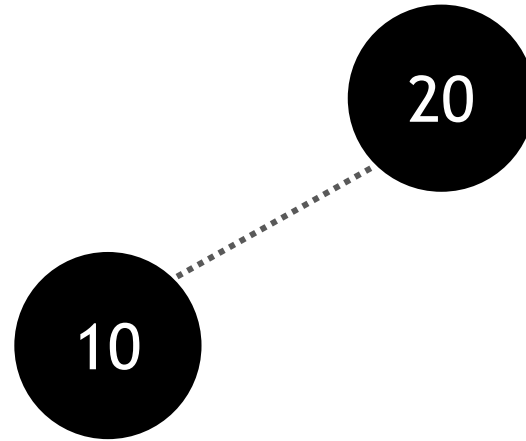
10 < 20



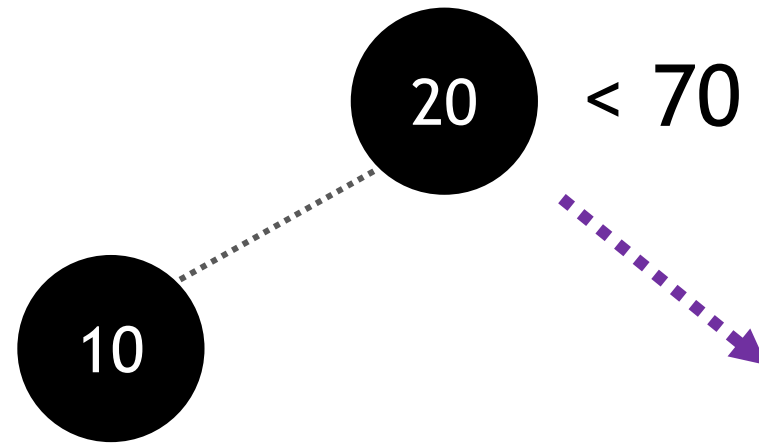
✓ ✓  
20, 10, 70, 0, 15, 50, 80, 18, 55



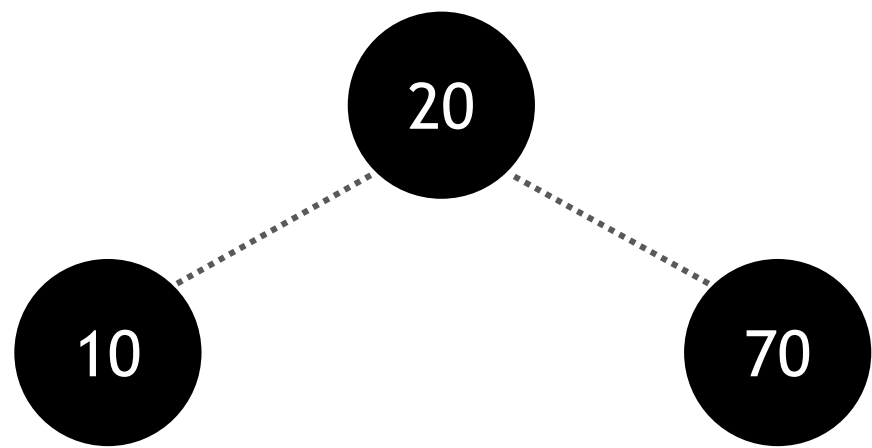
✓ ✓  
20, 10, 70, 0, 15, 50, 80, 18, 55  
↑



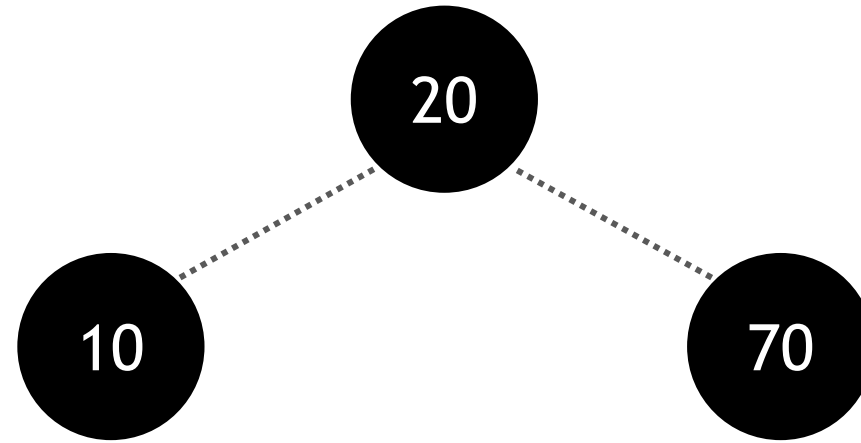
✓ ✓  
20, 10, 70, 0, 15, 50, 80, 18, 55  
↑



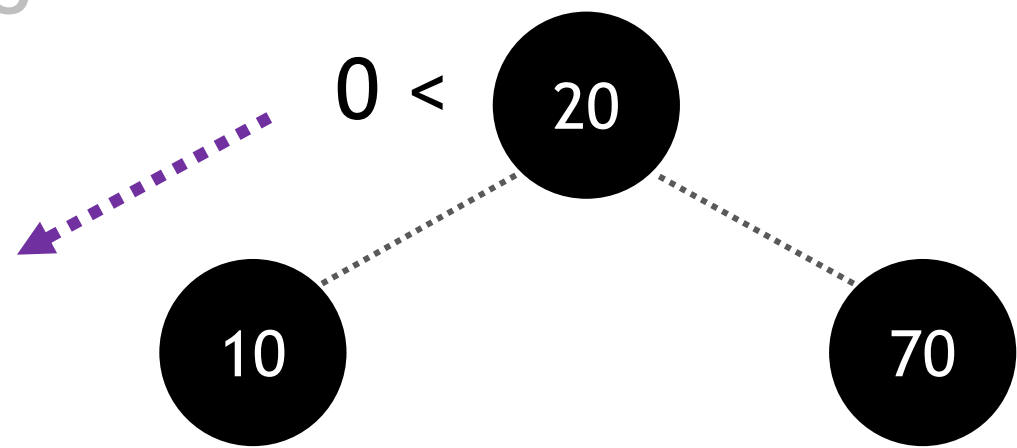
✓ ✓ ✓  
20, 10, 70, 0, 15, 50, 80, 18, 55



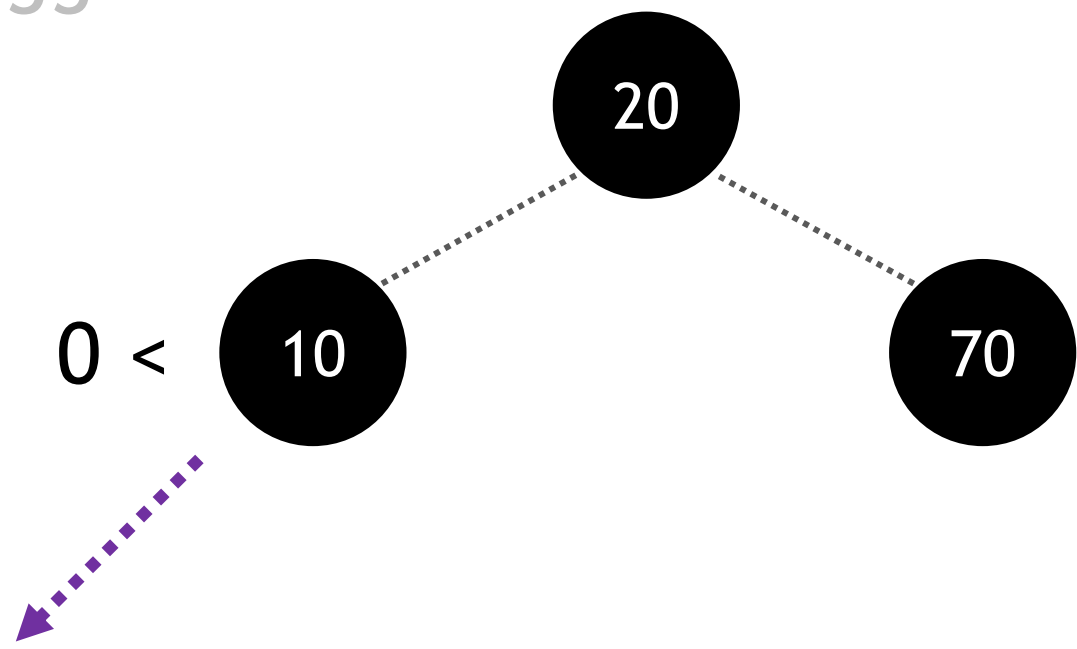
✓ ✓ ✓  
20, 10, 70, 0, 15, 50, 80, 18, 55  
↑



✓ ✓ ✓  
20, 10, 70, 0, 15, 50, 80, 18, 55  
↑

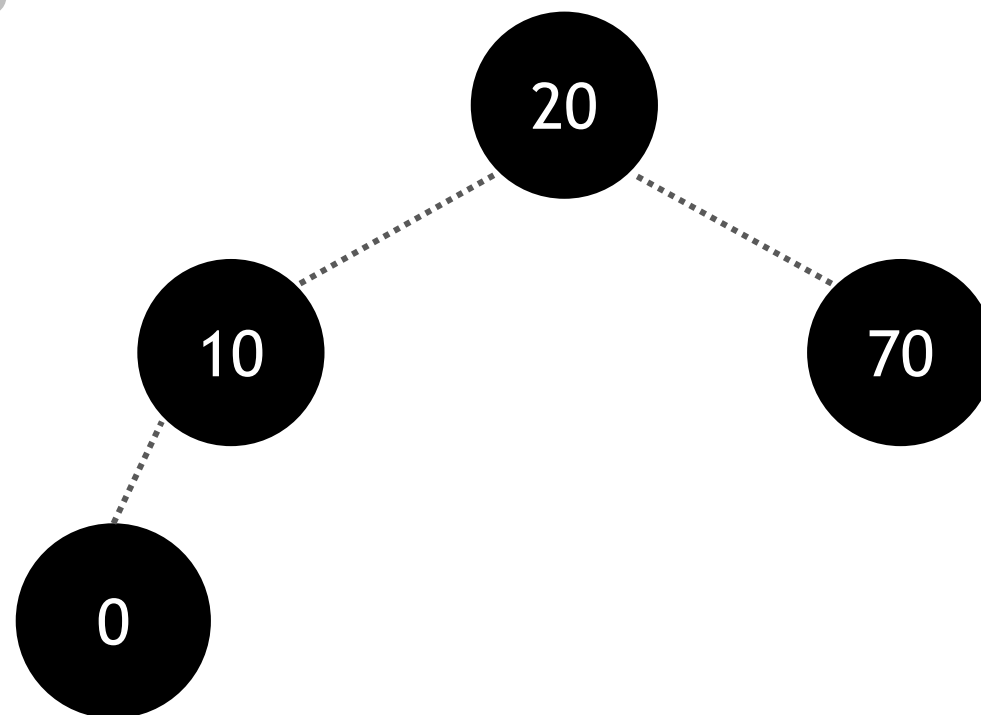


✓ ✓ ✓  
20, 10, 70, 0, 15, 50, 80, 18, 55  
↑

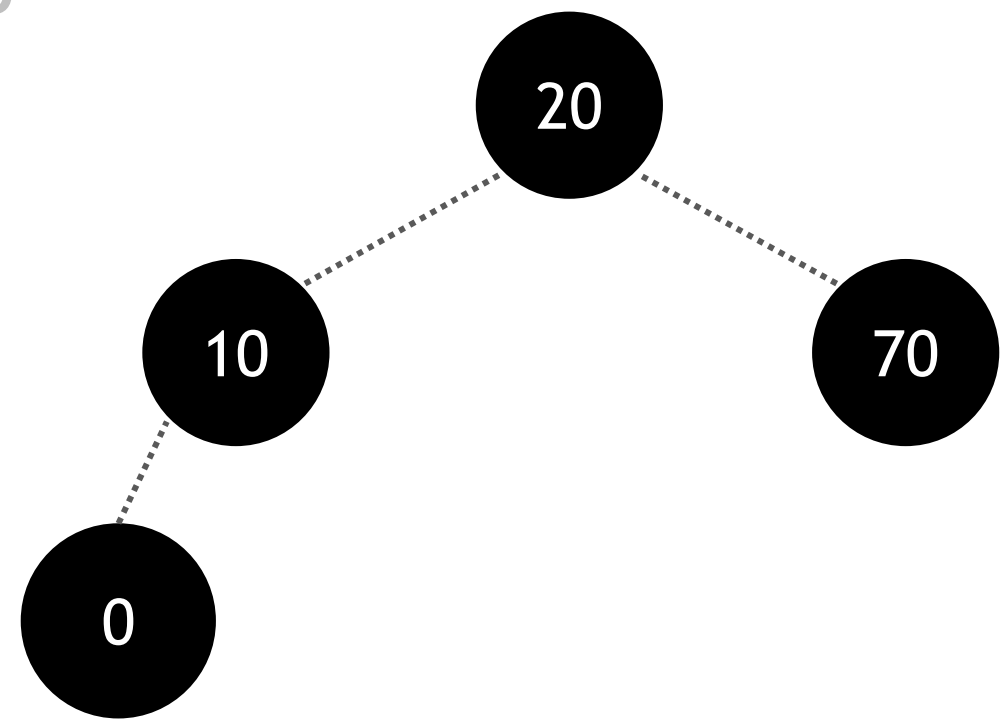




✓ ✓ ✓ ✓  
20, 10, 70, 0, 15, 50, 80, 18, 55

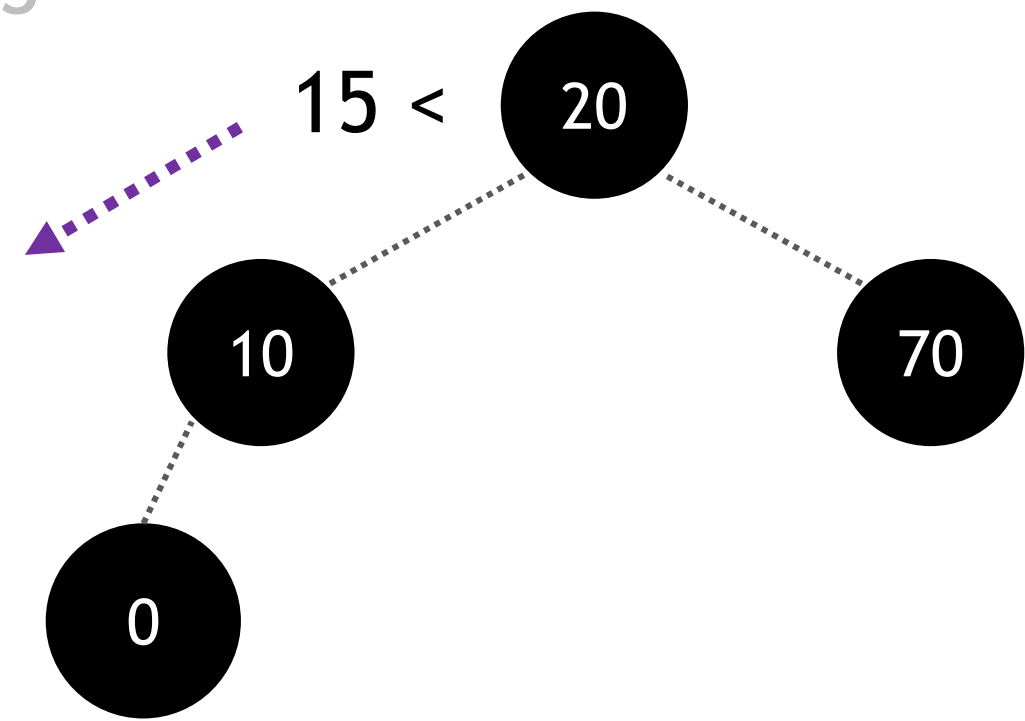


✓ ✓ ✓ ✓  
20, 10, 70, 0, 15, 50, 80, 18, 55  
↑

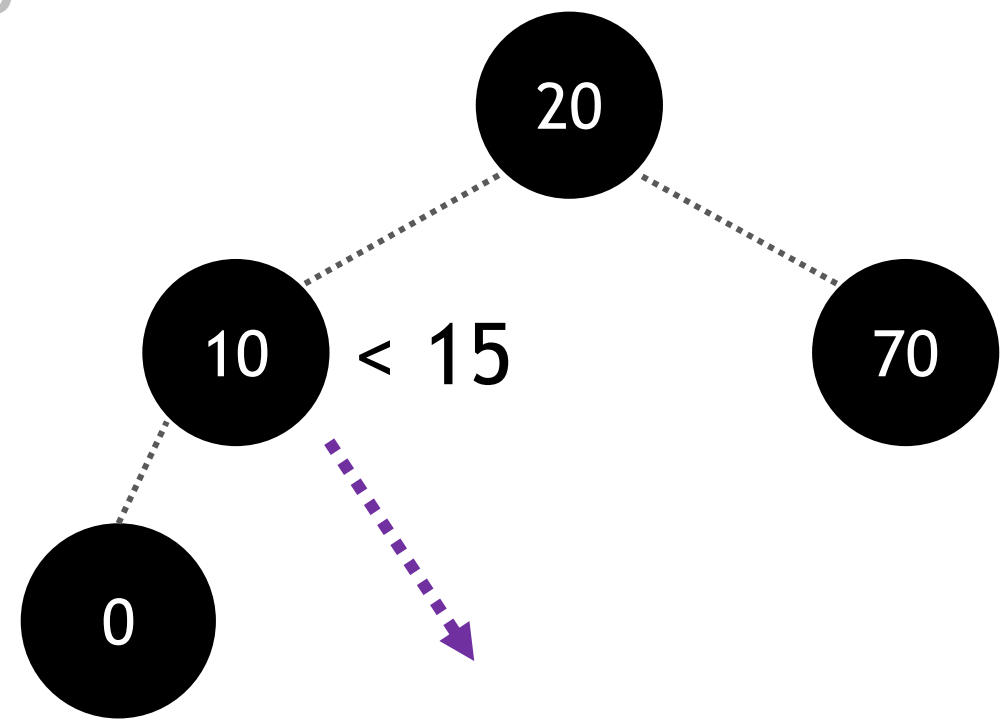


✓ ✓ ✓ ✓  
20, 10, 70, 0, 15, 50, 80, 18, 55

↑

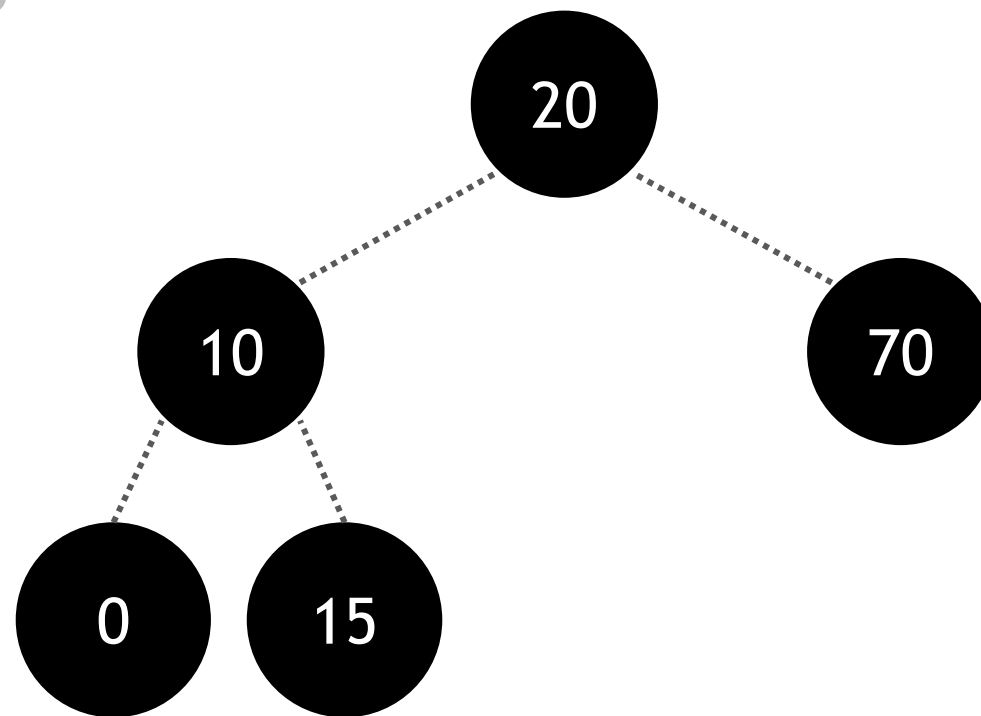


✓ ✓ ✓ ✓  
20, 10, 70, 0, 15, 50, 80, 18, 55  
↑

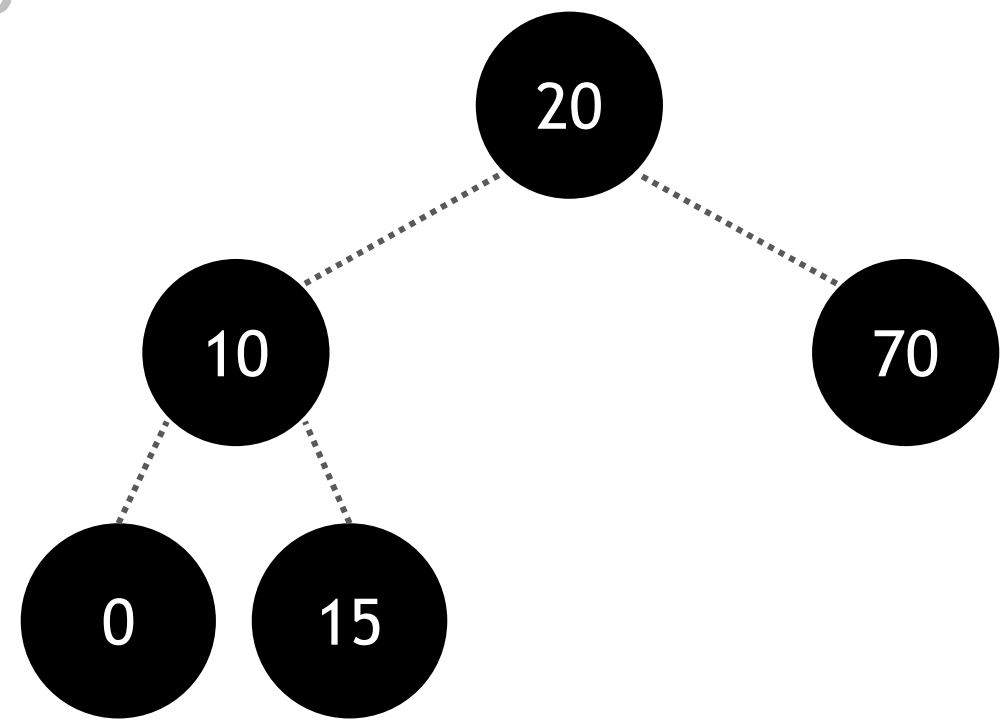


✓ ✓ ✓ ✓ ✓

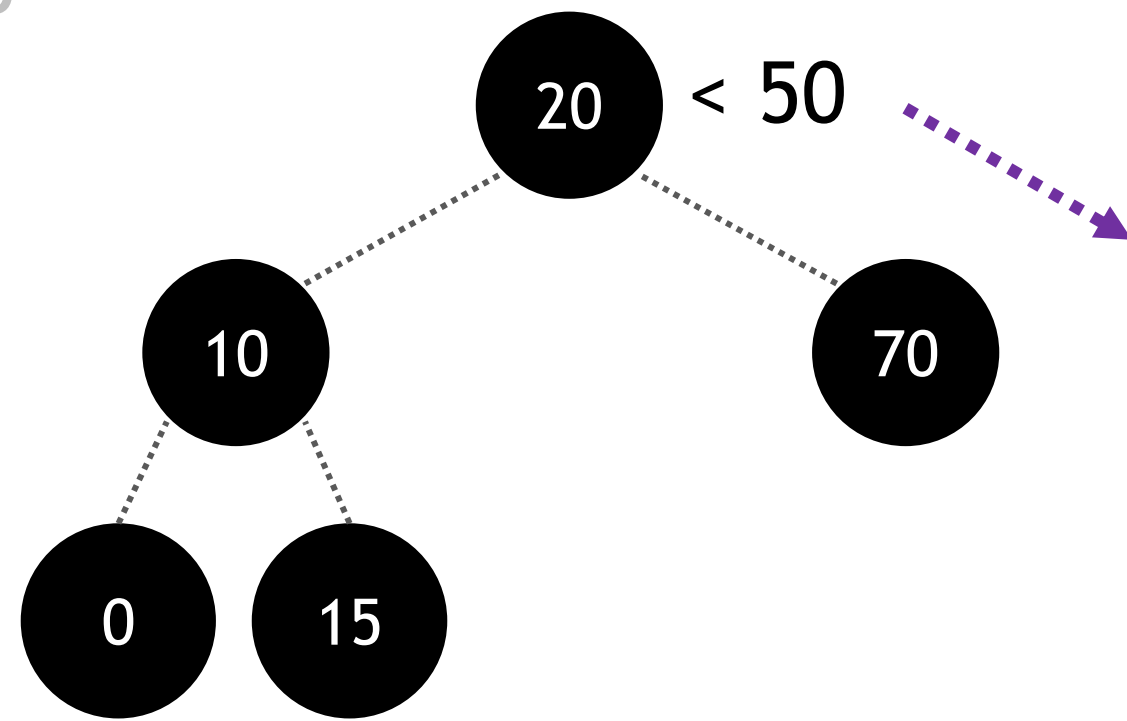
20, 10, 70, 0, 15, 50, 80, 18, 55



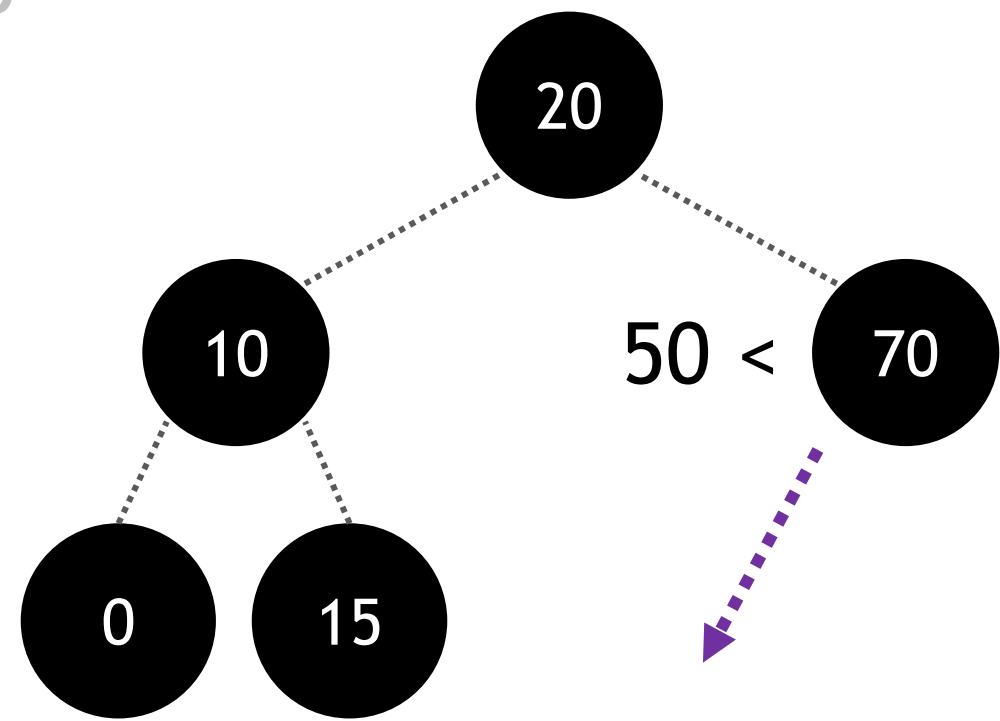
✓ ✓ ✓ ✓ ✓  
20, 10, 70, 0, 15, 50, 80, 18, 55  
↑



✓ ✓ ✓ ✓ ✓  
20, 10, 70, 0, 15, 50, 80, 18, 55  
↑

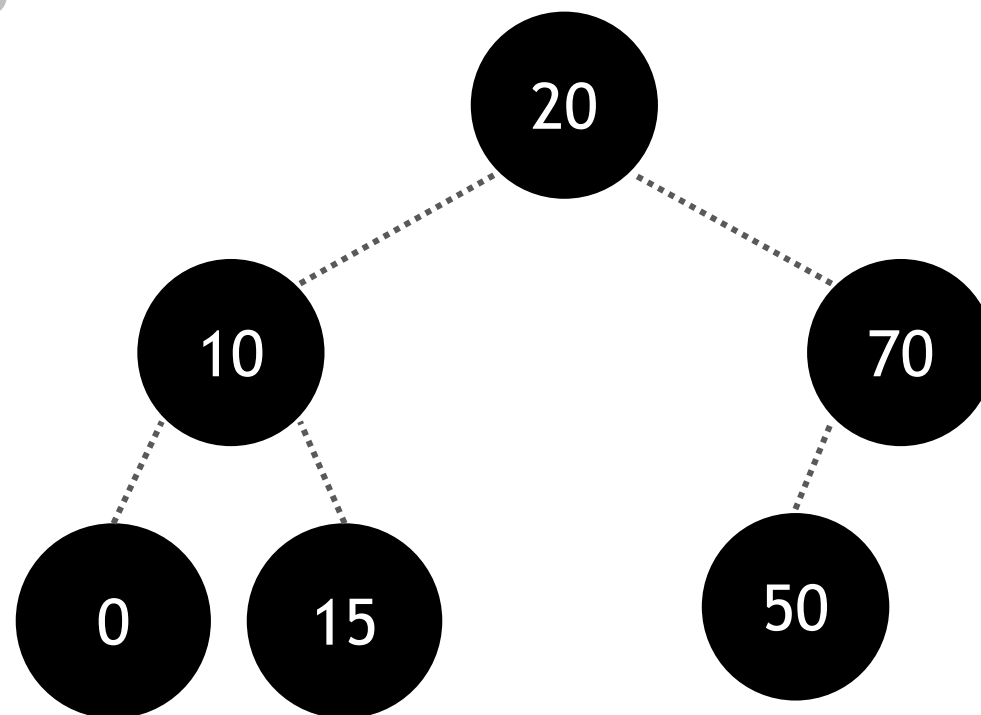


✓ ✓ ✓ ✓ ✓  
20, 10, 70, 0, 15, 50, 80, 18, 55  
↑

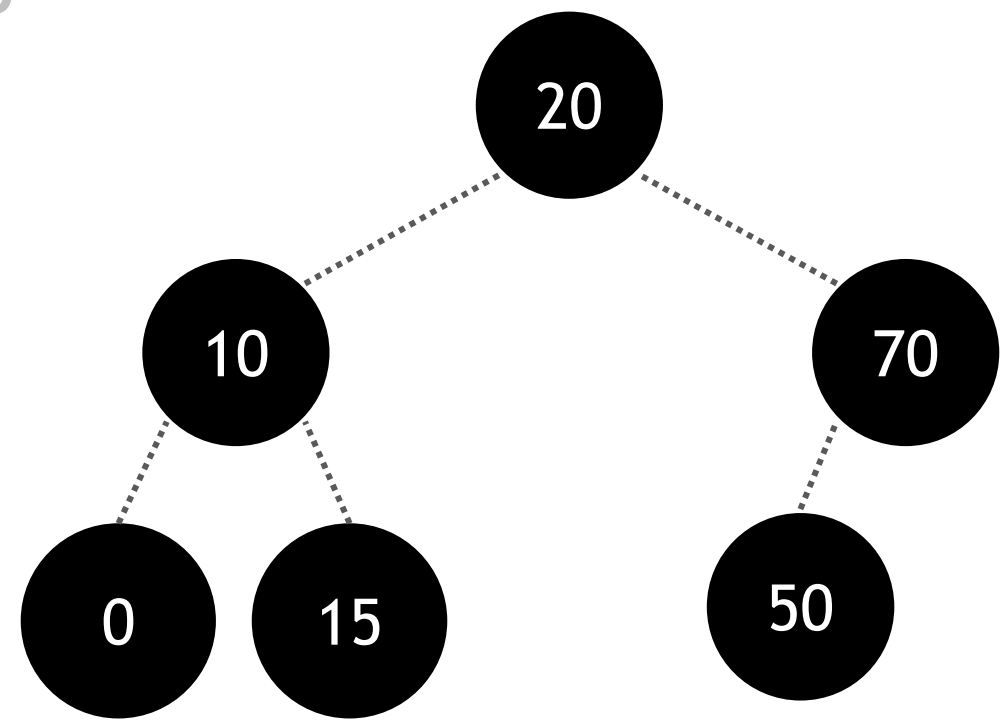




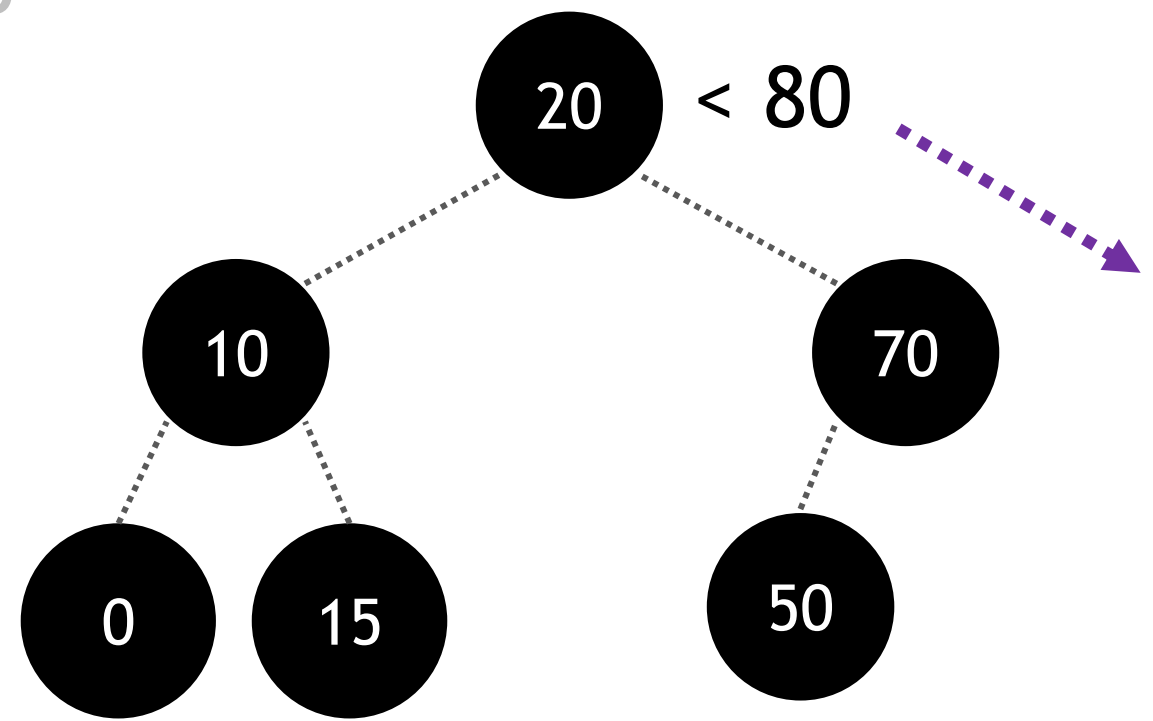
✓ ✓ ✓ ✓ ✓ ✓  
20, 10, 70, 0, 15, 50, 80, 18, 55



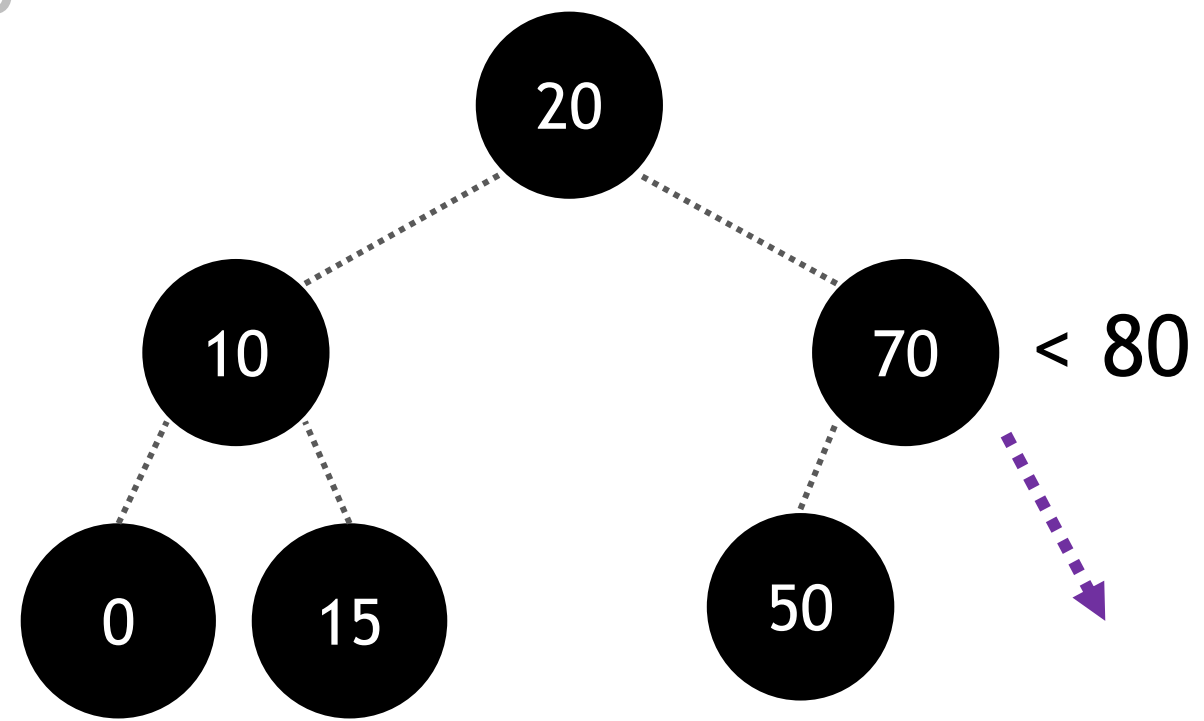
✓ ✓ ✓ ✓ ✓ ✓  
20, 10, 70, 0, 15, 50, 80, 18, 55  
↑



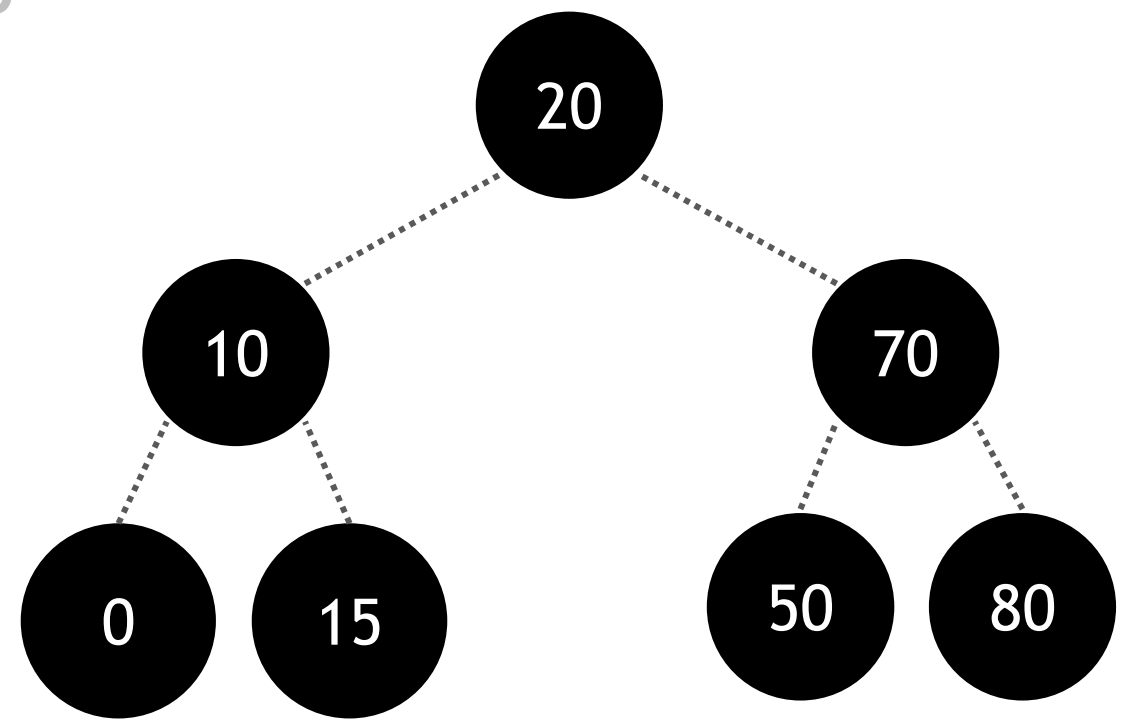
✓ ✓ ✓ ✓ ✓ ✓  
20, 10, 70, 0, 15, 50, 80, 18, 55  
↑



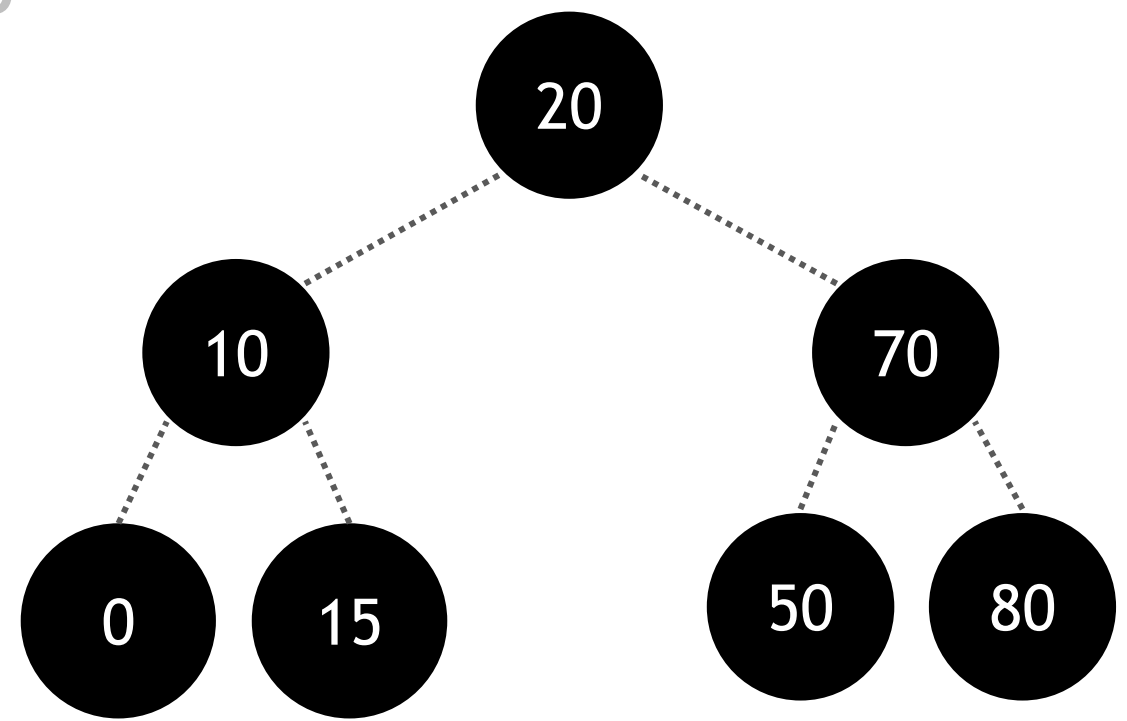
✓ ✓ ✓ ✓ ✓ ✓  
20, 10, 70, 0, 15, 50, 80, 18, 55  
↑



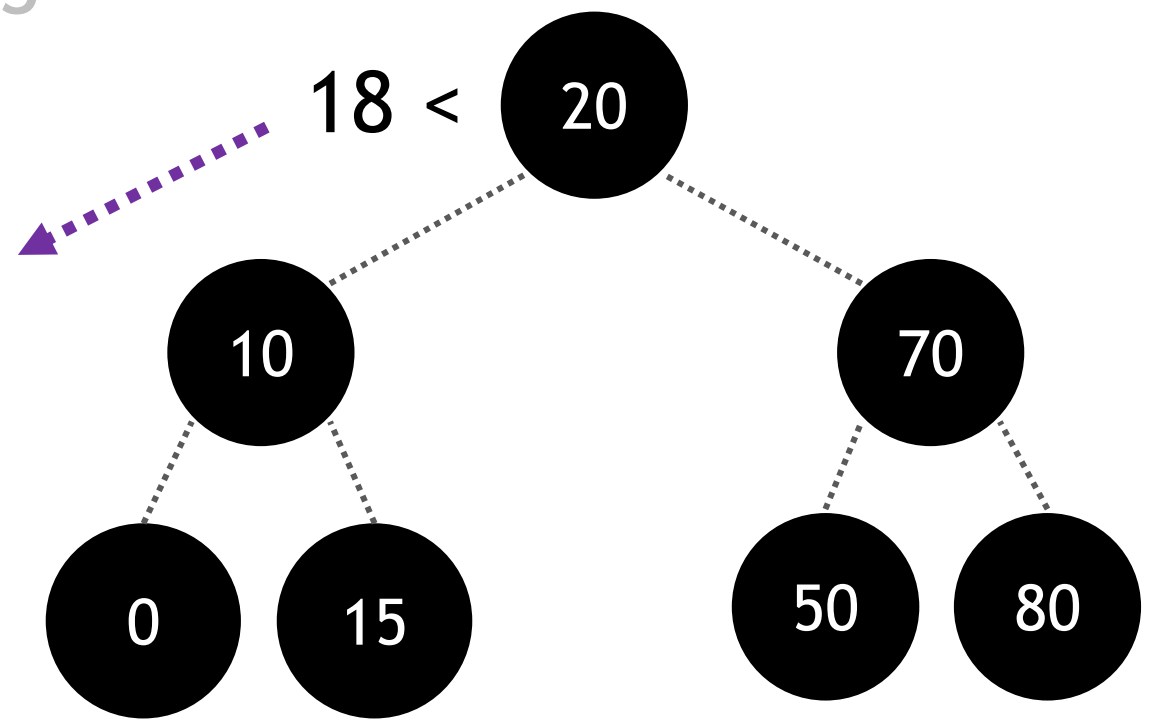
✓ ✓ ✓ ✓ ✓ ✓ ✓  
20, 10, 70, 0, 15, 50, 80, 18, 55



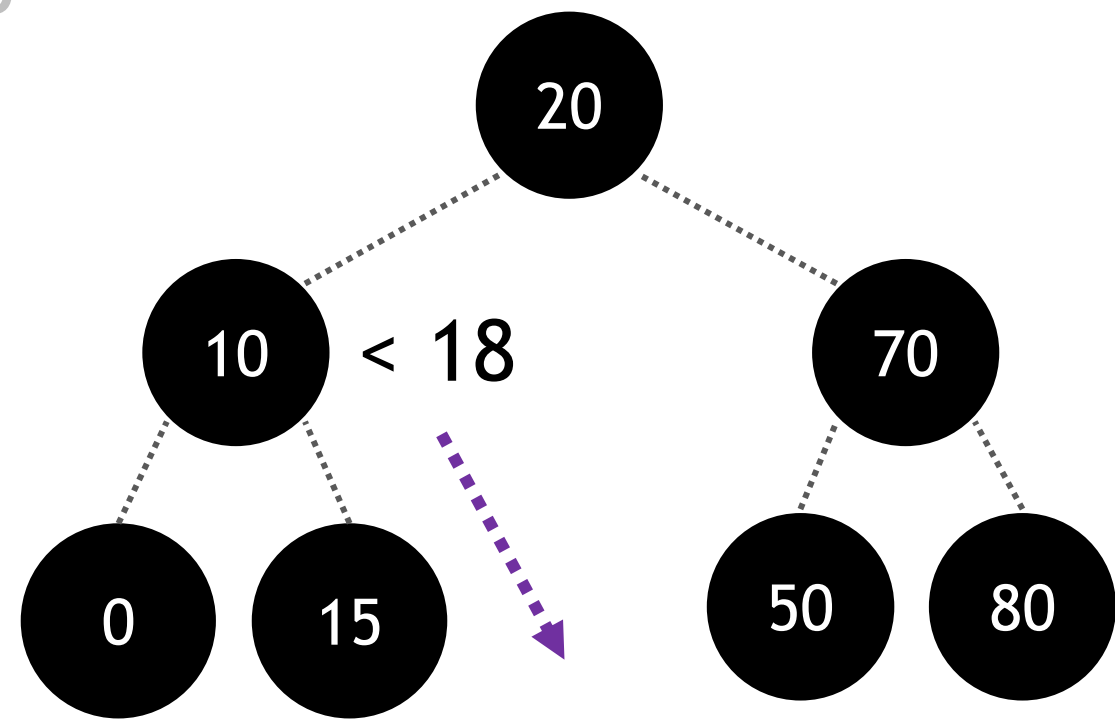
✓ ✓ ✓ ✓ ✓ ✓ ✓  
20, 10, 70, 0, 15, 50, 80, 18, 55  
↑



✓ ✓ ✓ ✓ ✓ ✓ ✓  
20, 10, 70, 0, 15, 50, 80, 18, 55



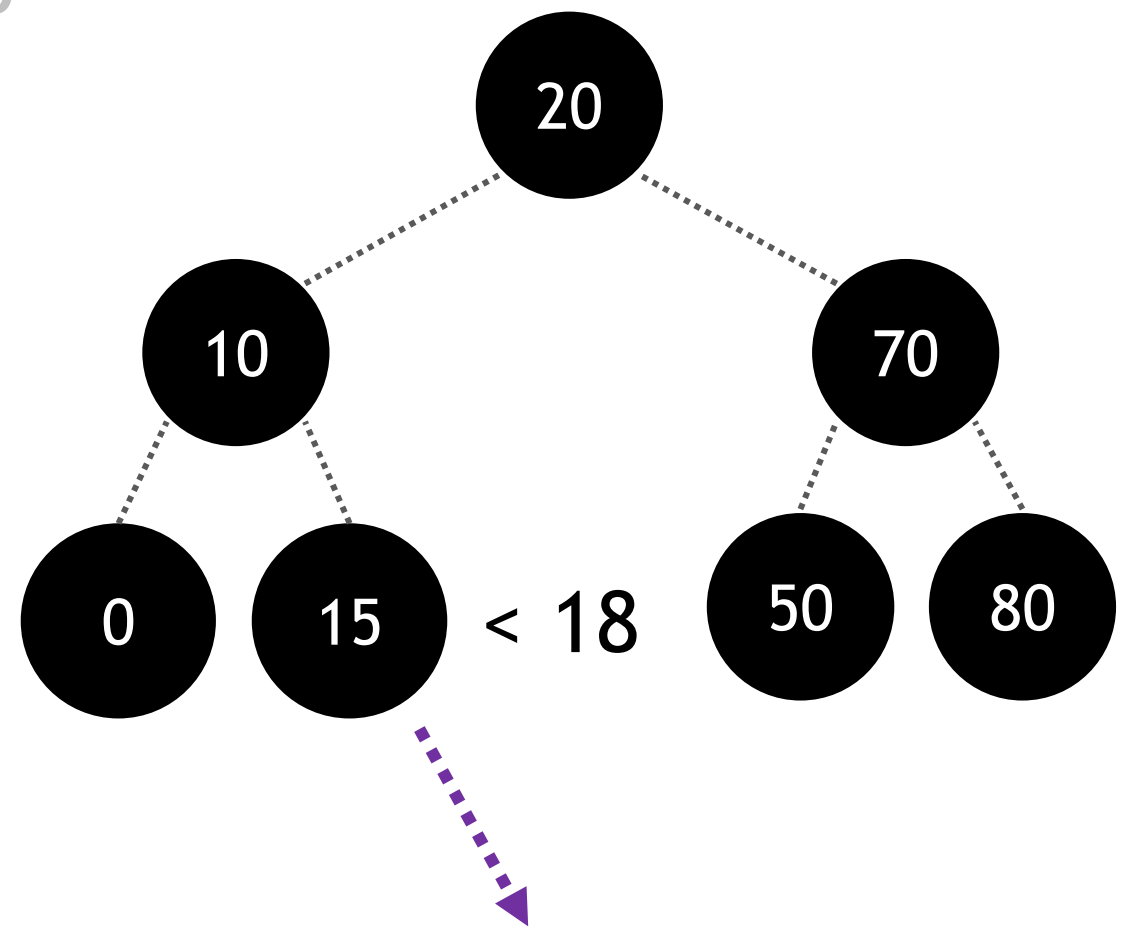
✓ ✓ ✓ ✓ ✓ ✓ ✓  
20, 10, 70, 0, 15, 50, 80, 18, 55  
↑



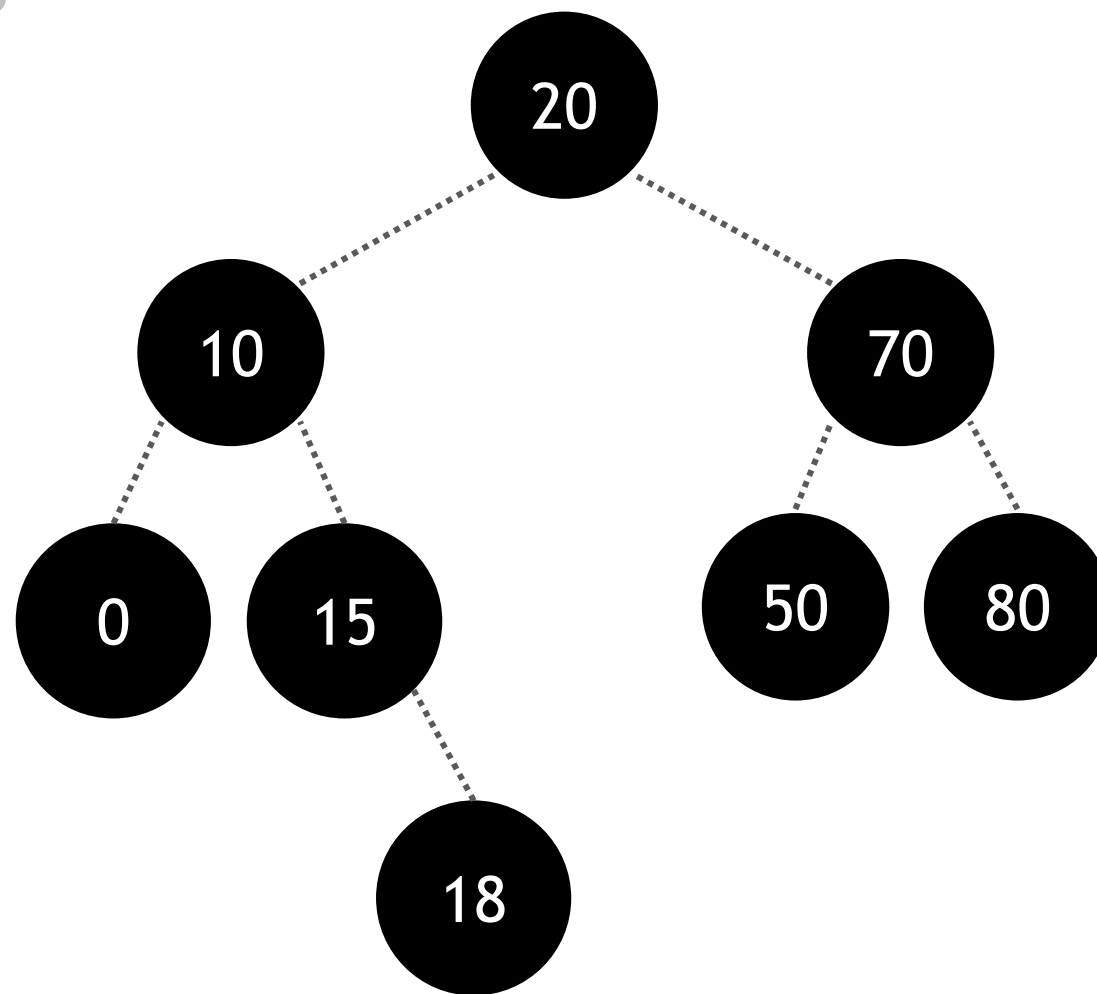


✓ ✓ ✓ ✓ ✓ ✓ ✓  
20, 10, 70, 0, 15, 50, 80, 18, 55

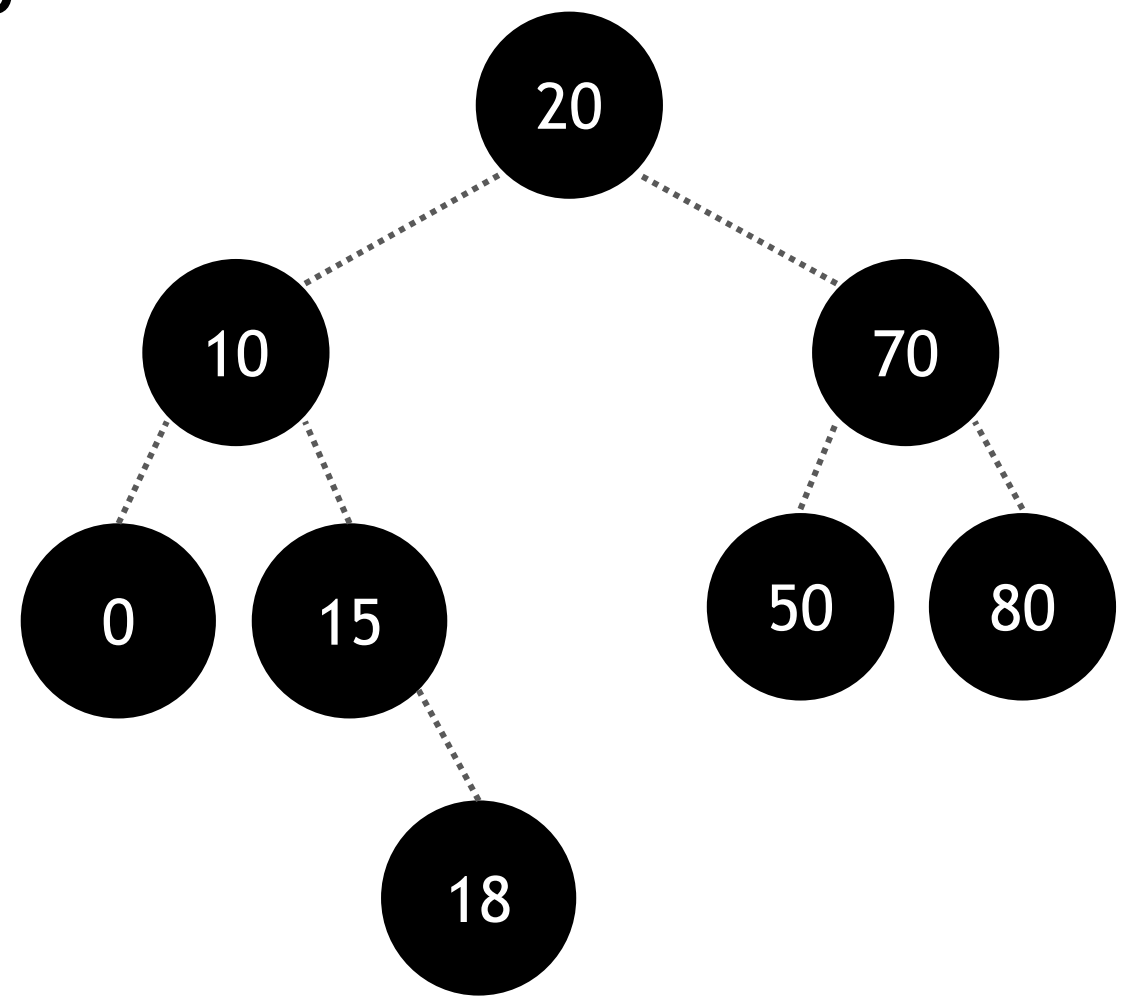
↑



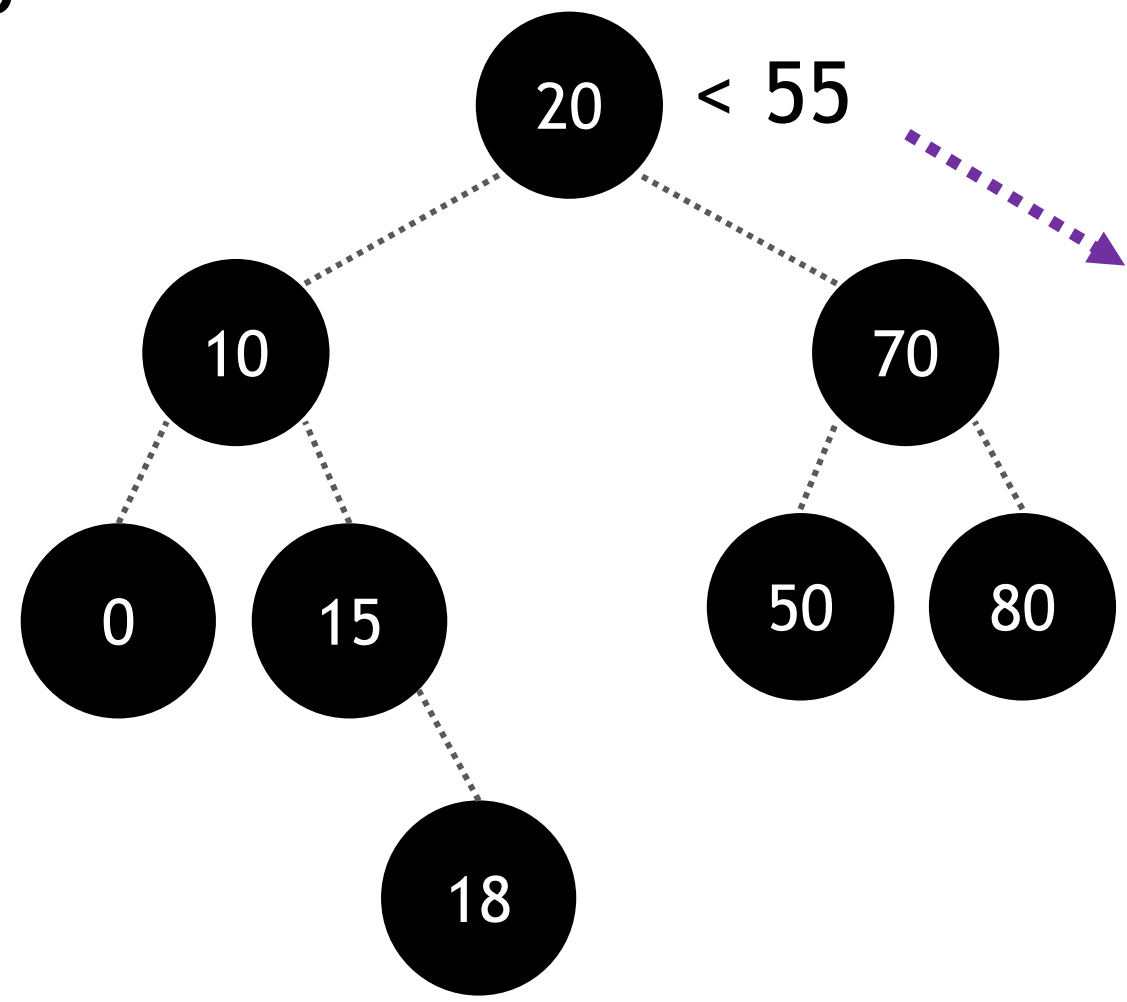
✓ ✓ ✓ ✓ ✓ ✓ ✓  
20, 10, 70, 0, 15, 50, 80, 18, 55



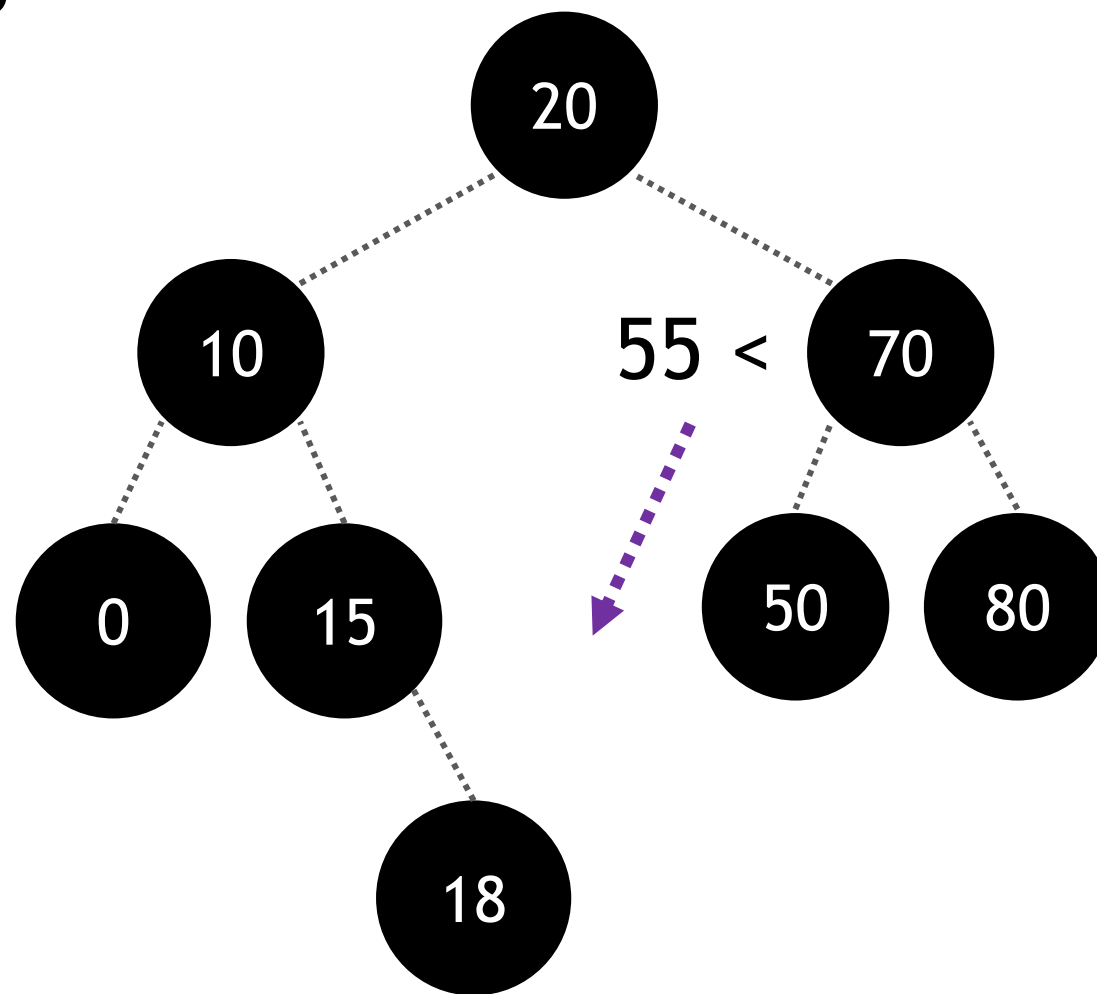
✓ ✓ ✓ ✓ ✓ ✓ ✓  
20, 10, 70, 0, 15, 50, 80, 18, 55  
↑



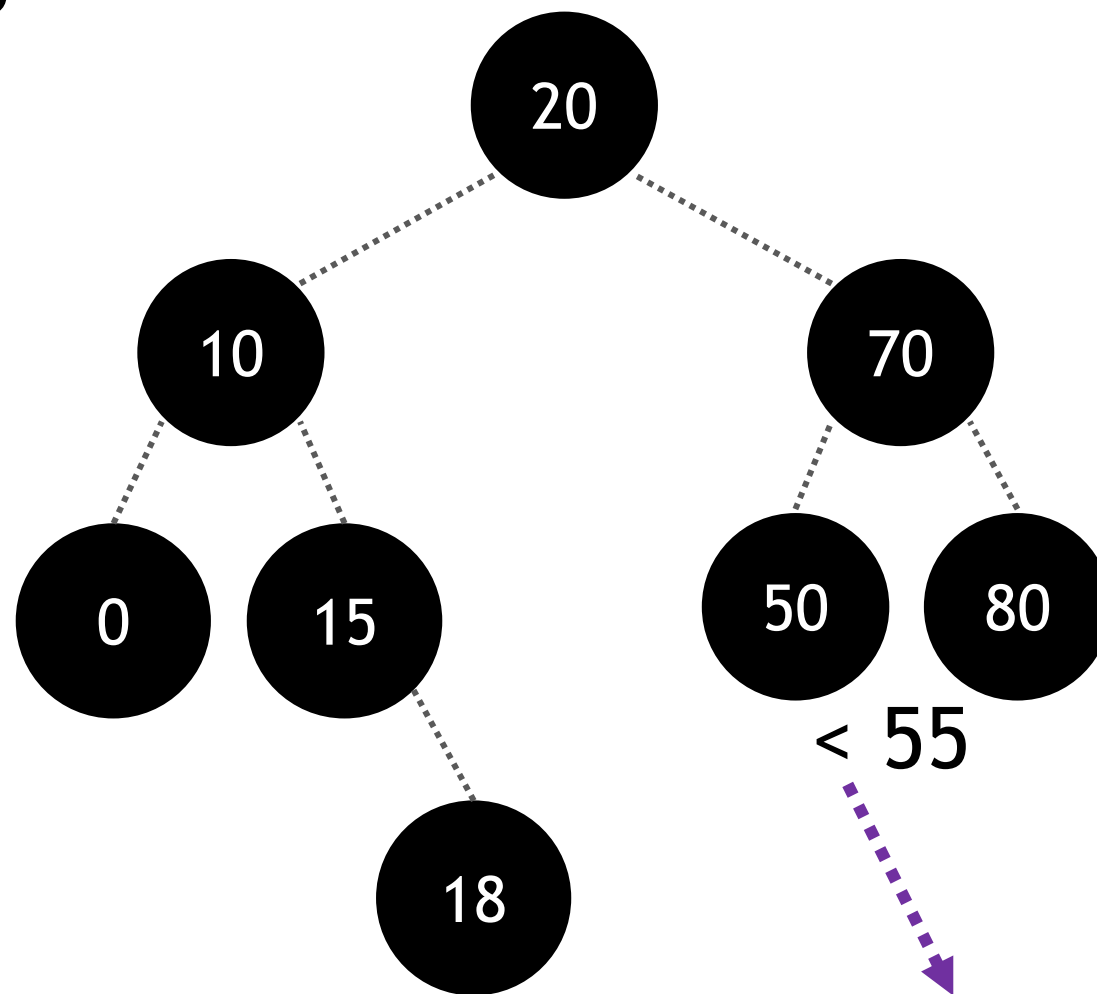
✓ ✓ ✓ ✓ ✓ ✓ ✓  
20, 10, 70, 0, 15, 50, 80, 18, 55  
↑



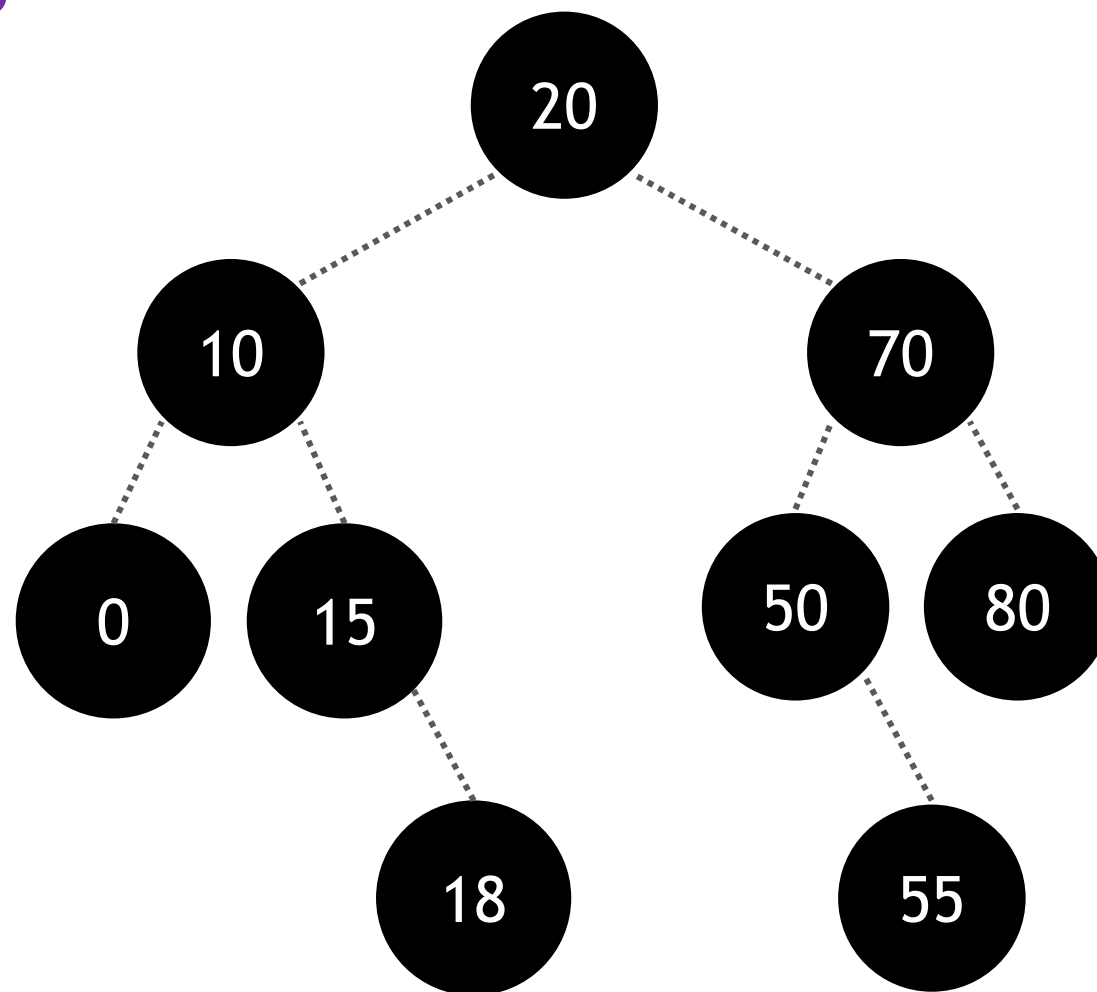
✓ ✓ ✓ ✓ ✓ ✓ ✓  
20, 10, 70, 0, 15, 50, 80, 18, 55  
↑



✓ ✓ ✓ ✓ ✓ ✓ ✓  
20, 10, 70, 0, 15, 50, 80, 18, 55  
↑



✓ ✓ ✓ ✓ ✓ ✓ ✓ ✓ ✓  
20, 10, 70, 0, 15, 50, 80, 18, 55



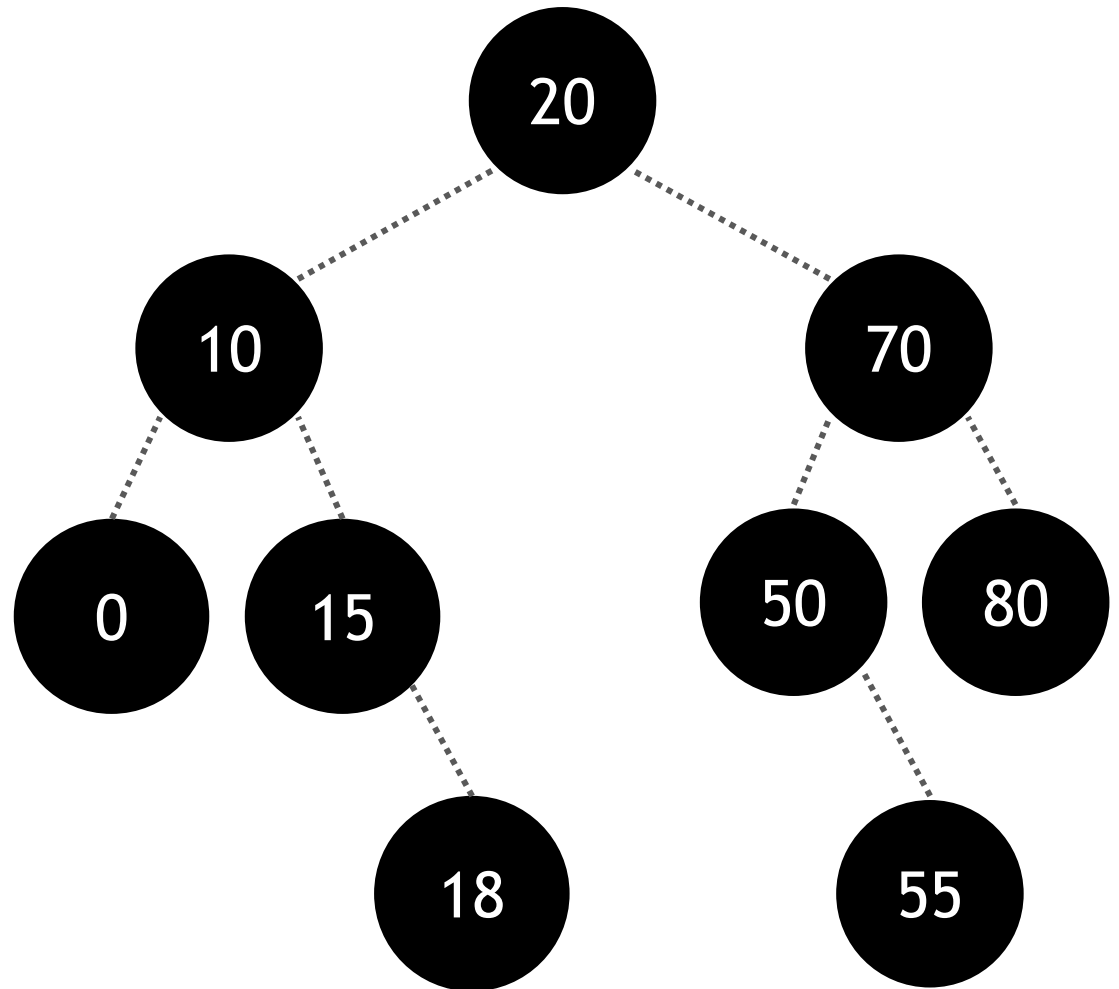
# Pre-order

Prechod BST



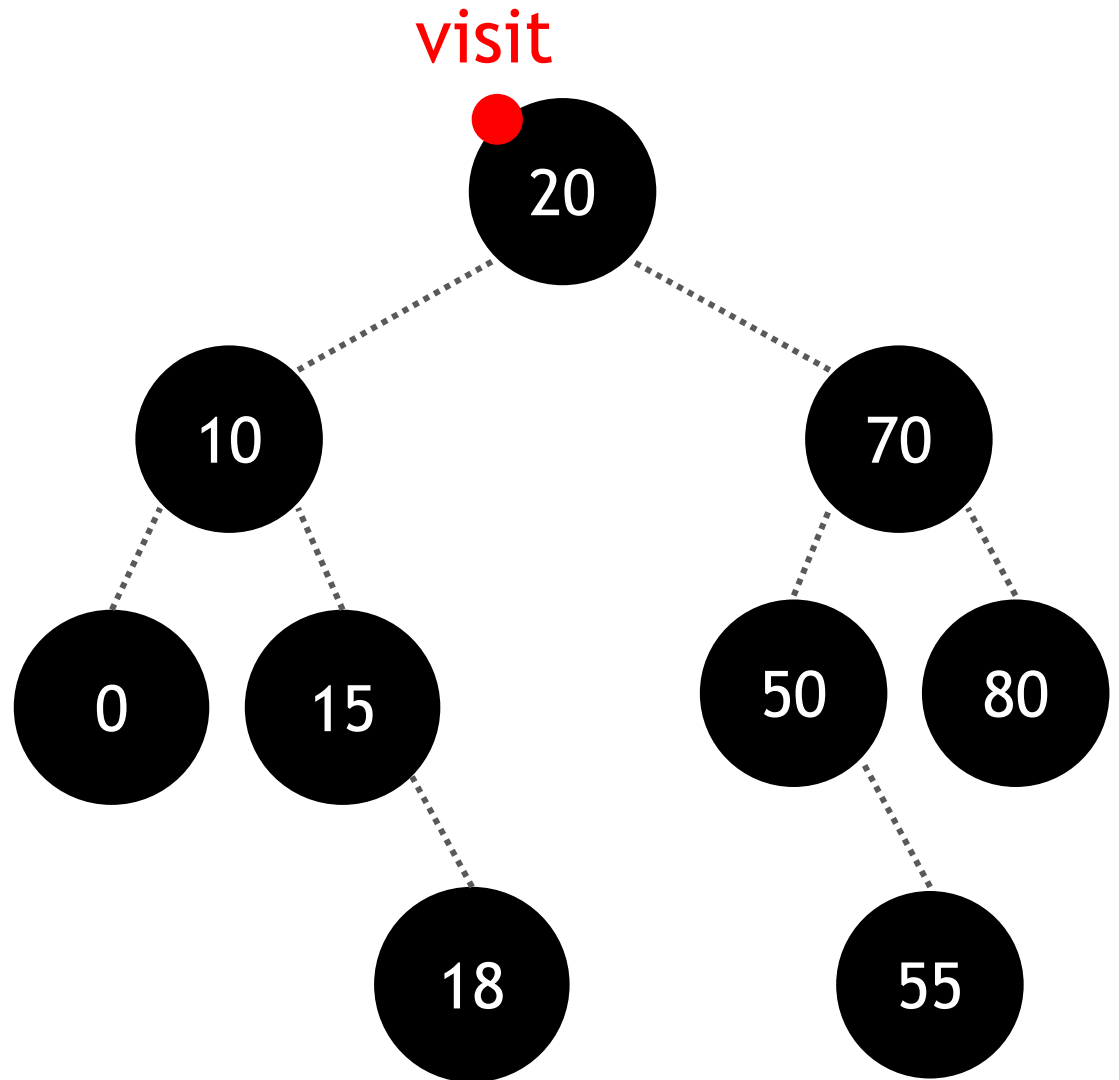
# Pre-order

1. Visit the node.
2. `preOrder(node->smaller)`
3. `preOrder(node->greater)`



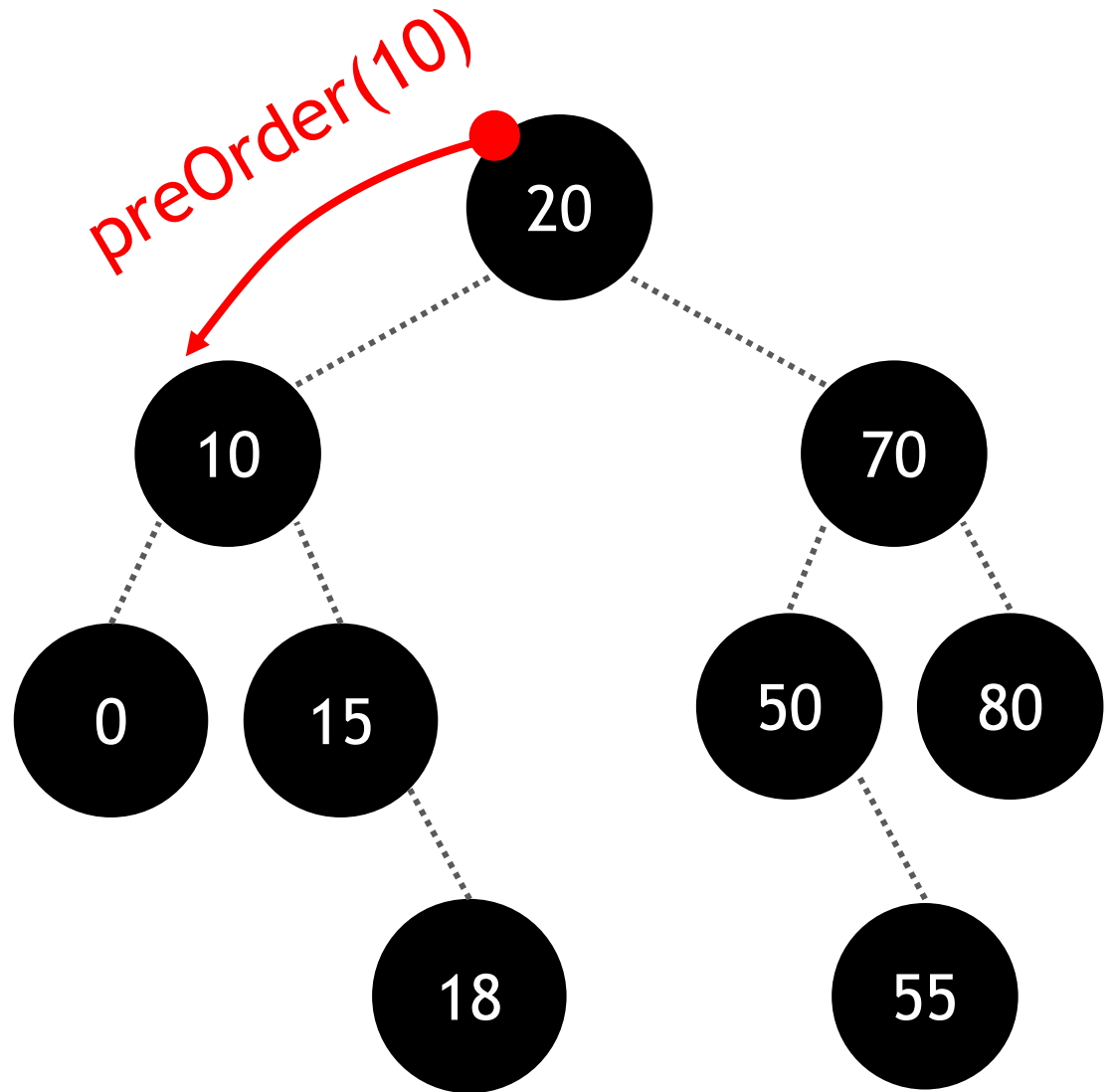
# Pre-order

1. Visit the node.
2. `preOrder(node->smaller)`
3. `preOrder(node->greater)`



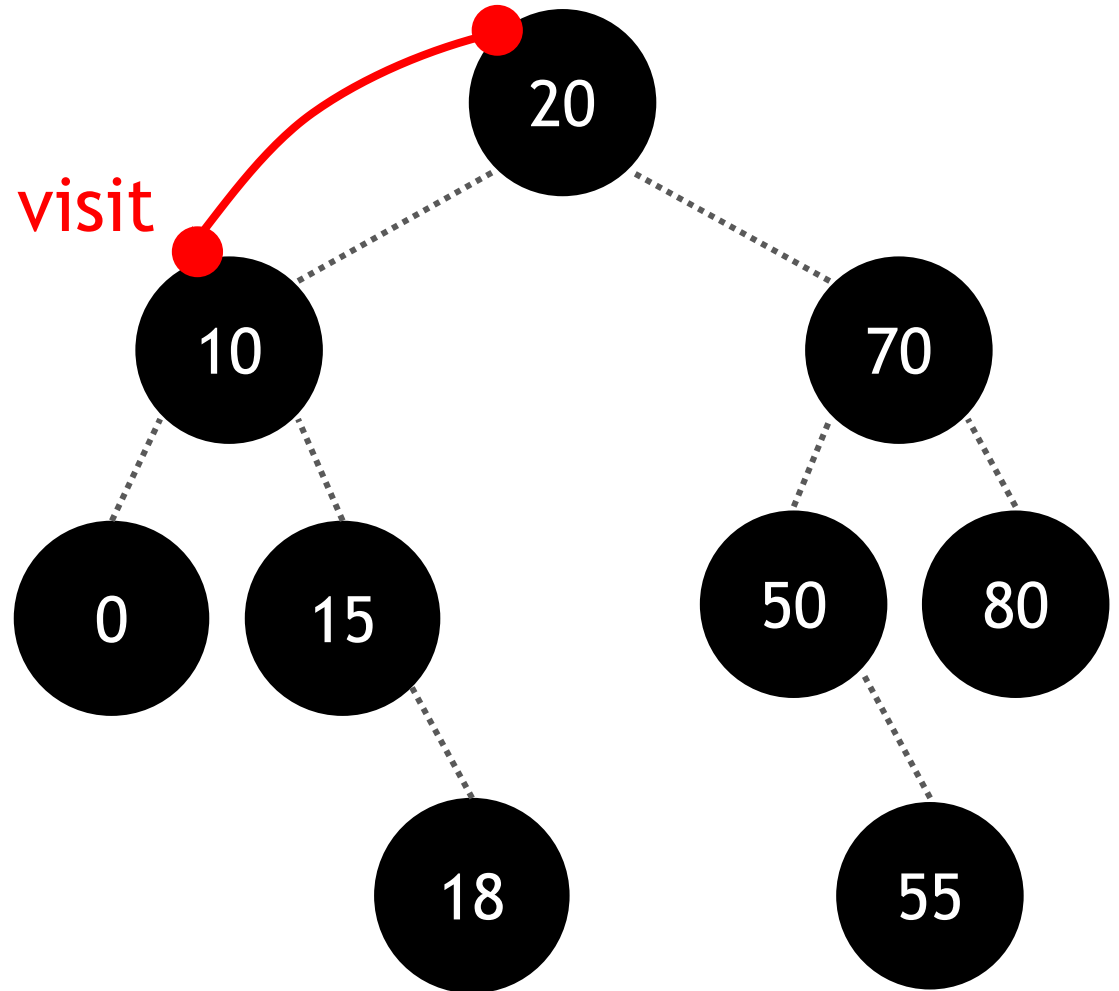
# Pre-order

1. Visit the node.
2. `preOrder(node->smaller)`
3. `preOrder(node->greater)`



# Pre-order

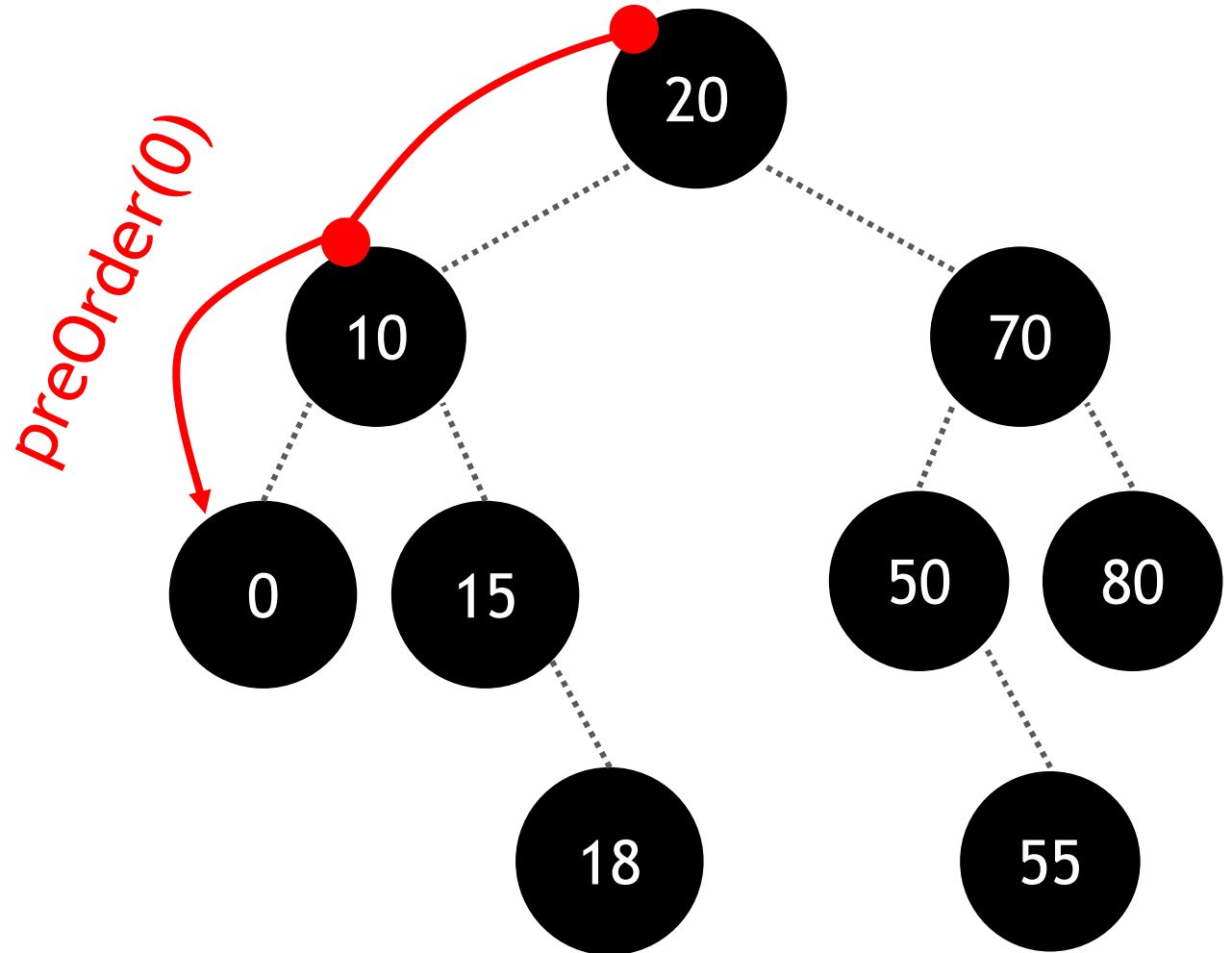
1. Visit the node.
2. `preOrder(node->smaller)`
3. `preOrder(node->greater)`



20, 10,

# Pre-order

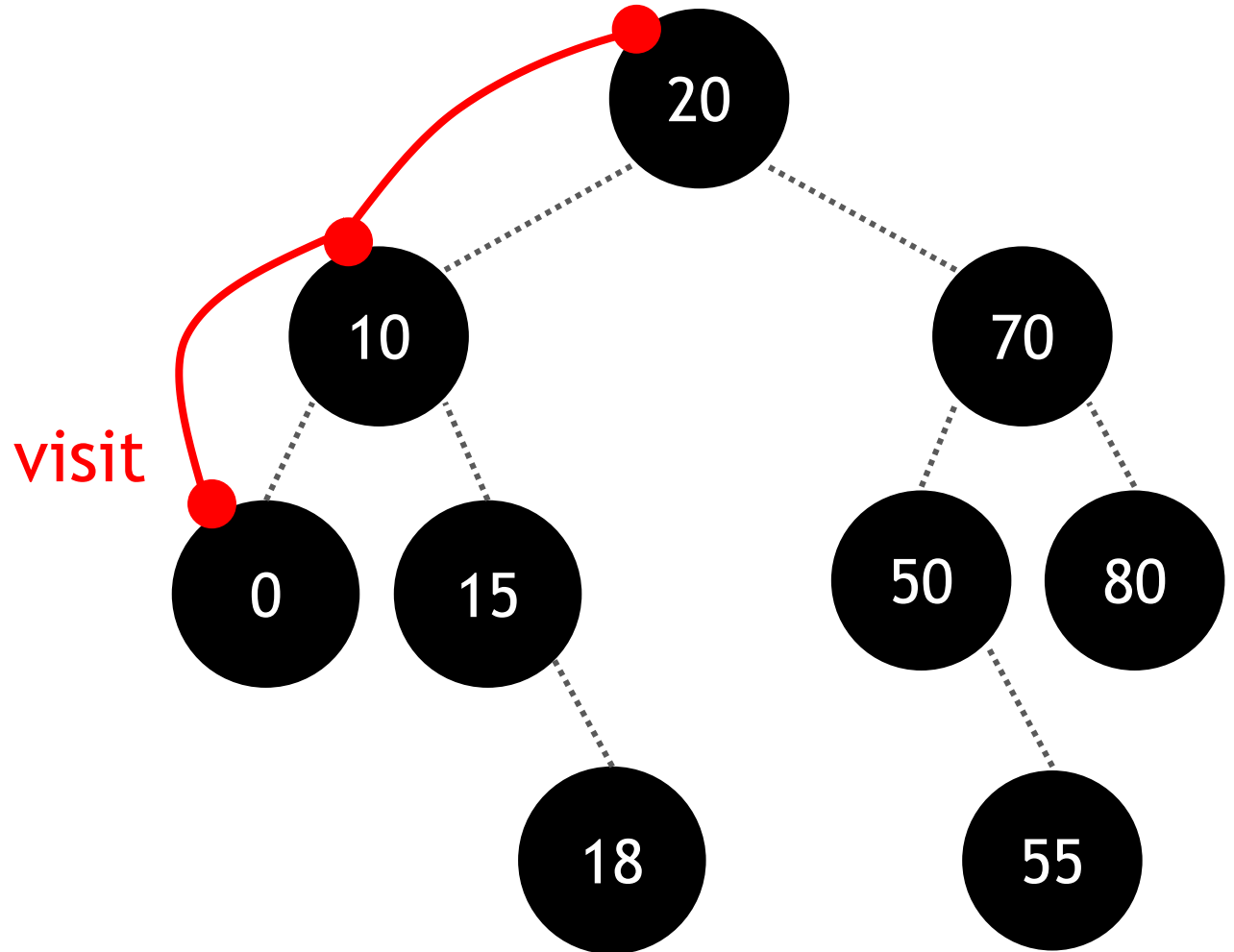
1. Visit the node.
2. `preOrder(node->smaller)`
3. `preOrder(node->greater)`



20,10,

# Pre-order

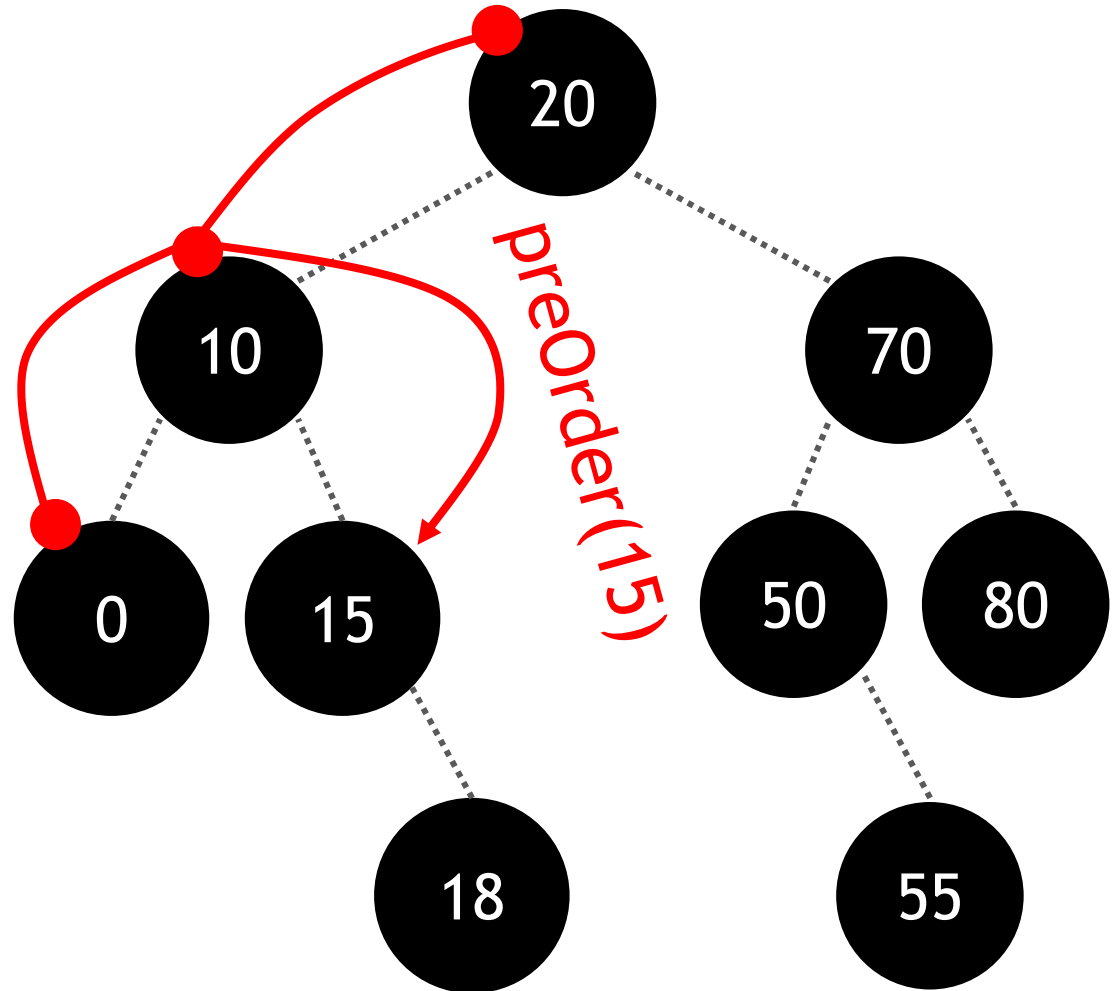
1. Visit the node.
2. `preOrder(node->smaller)`
3. `preOrder(node->greater)`



20,10,0

# Pre-order

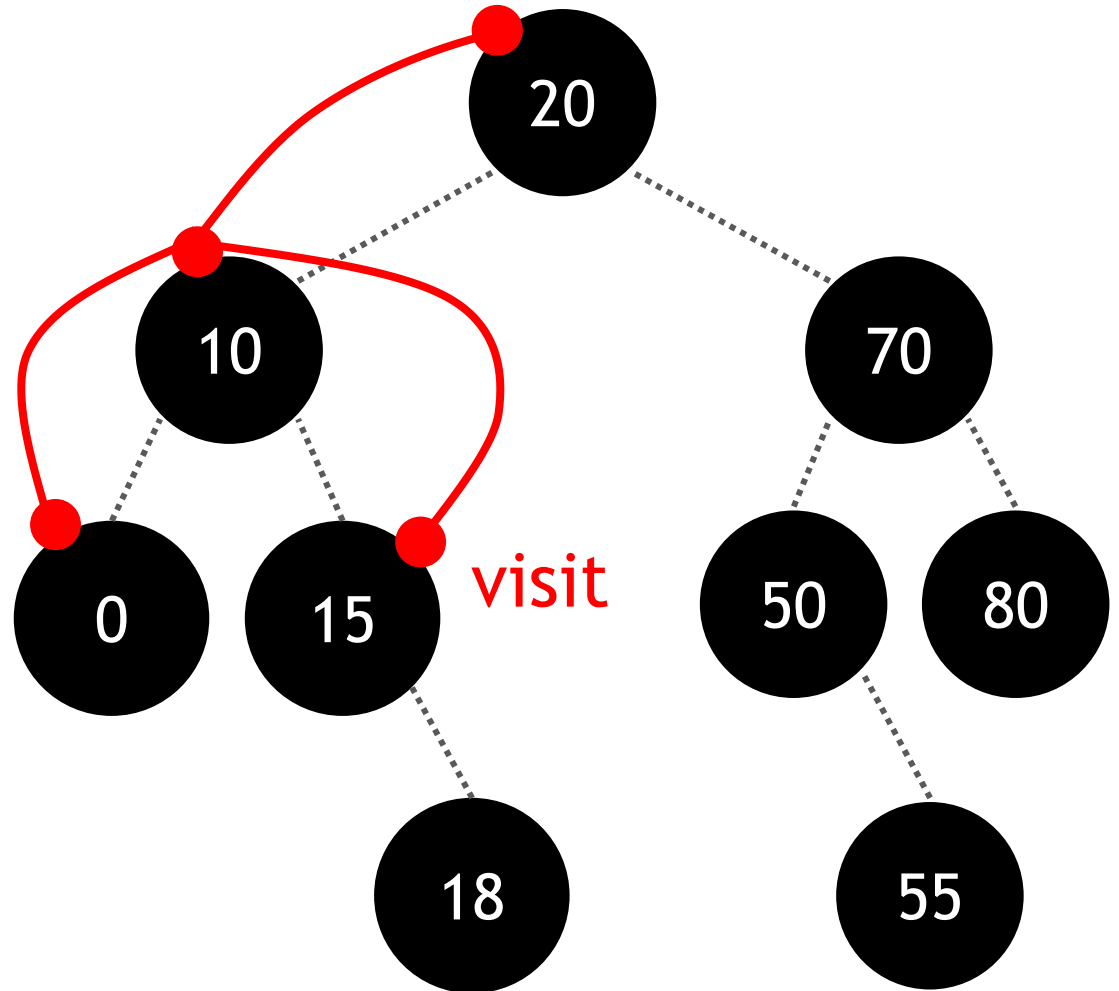
1. Visit the node.
2. `preOrder(node->smaller)`
3. `preOrder(node->greater)`



20,10,0

# Pre-order

1. Visit the node.
2. `preOrder(node->smaller)`
3. `preOrder(node->greater)`

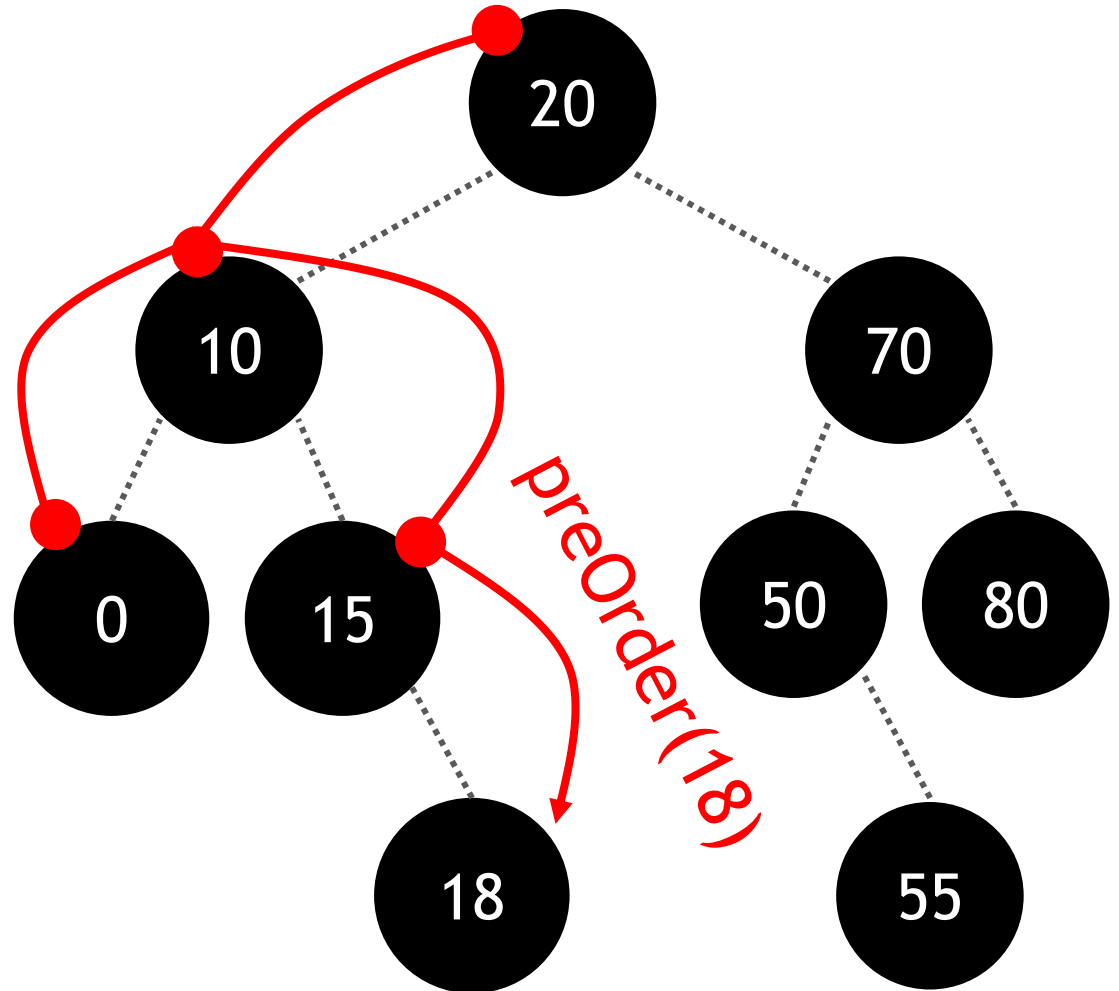


20,10,0,15



# Pre-order

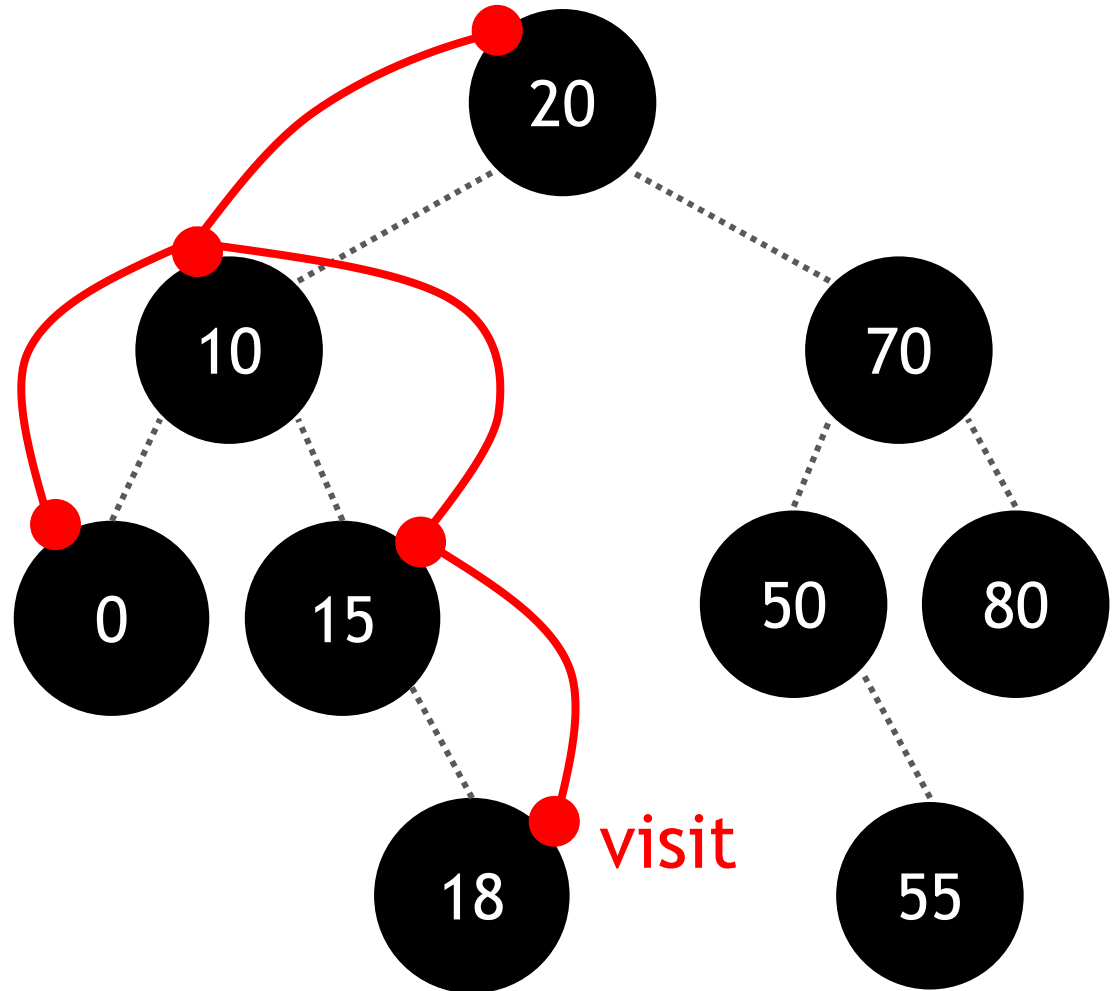
1. Visit the node.
2. `preOrder(node->smaller)`
3. `preOrder(node->greater)`



20,10,0,15

# Pre-order

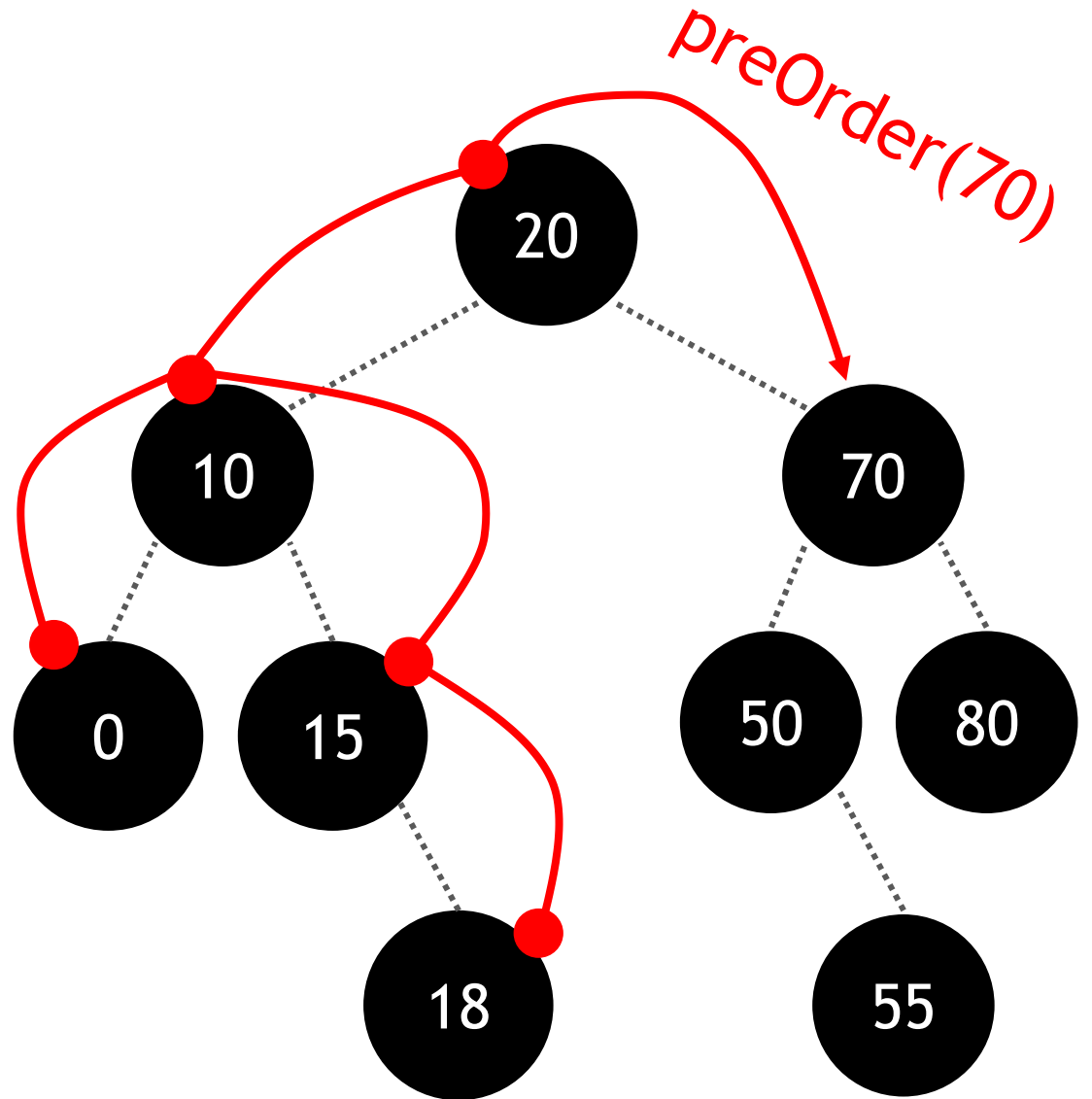
1. Visit the node.
2. `preOrder(node->smaller)`
3. `preOrder(node->greater)`



20,10,0,15,18,

# Pre-order

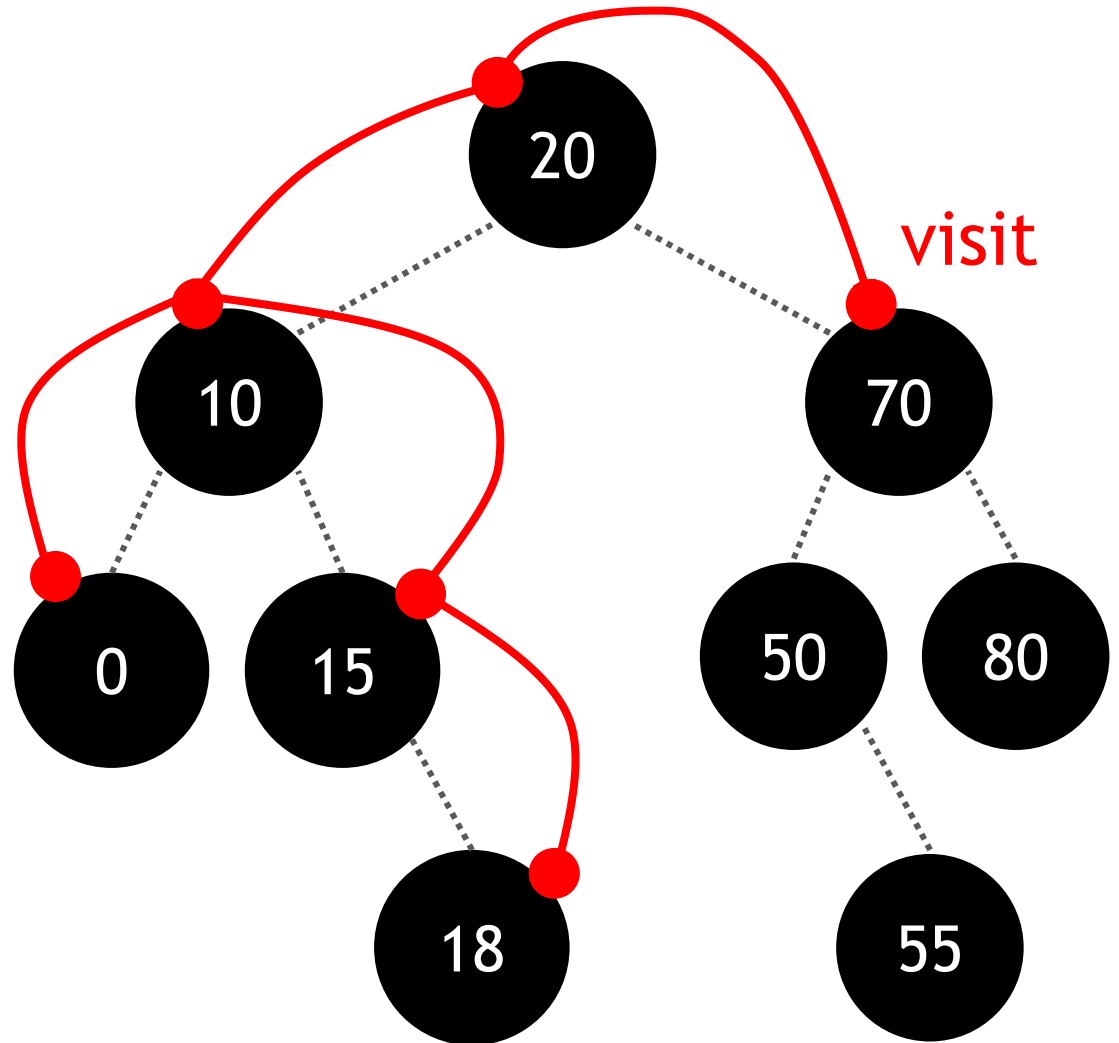
1. Visit the node.
2. `preOrder(node->smaller)`
3. `preOrder(node->greater)`



20,10,0,15,18,

# Pre-order

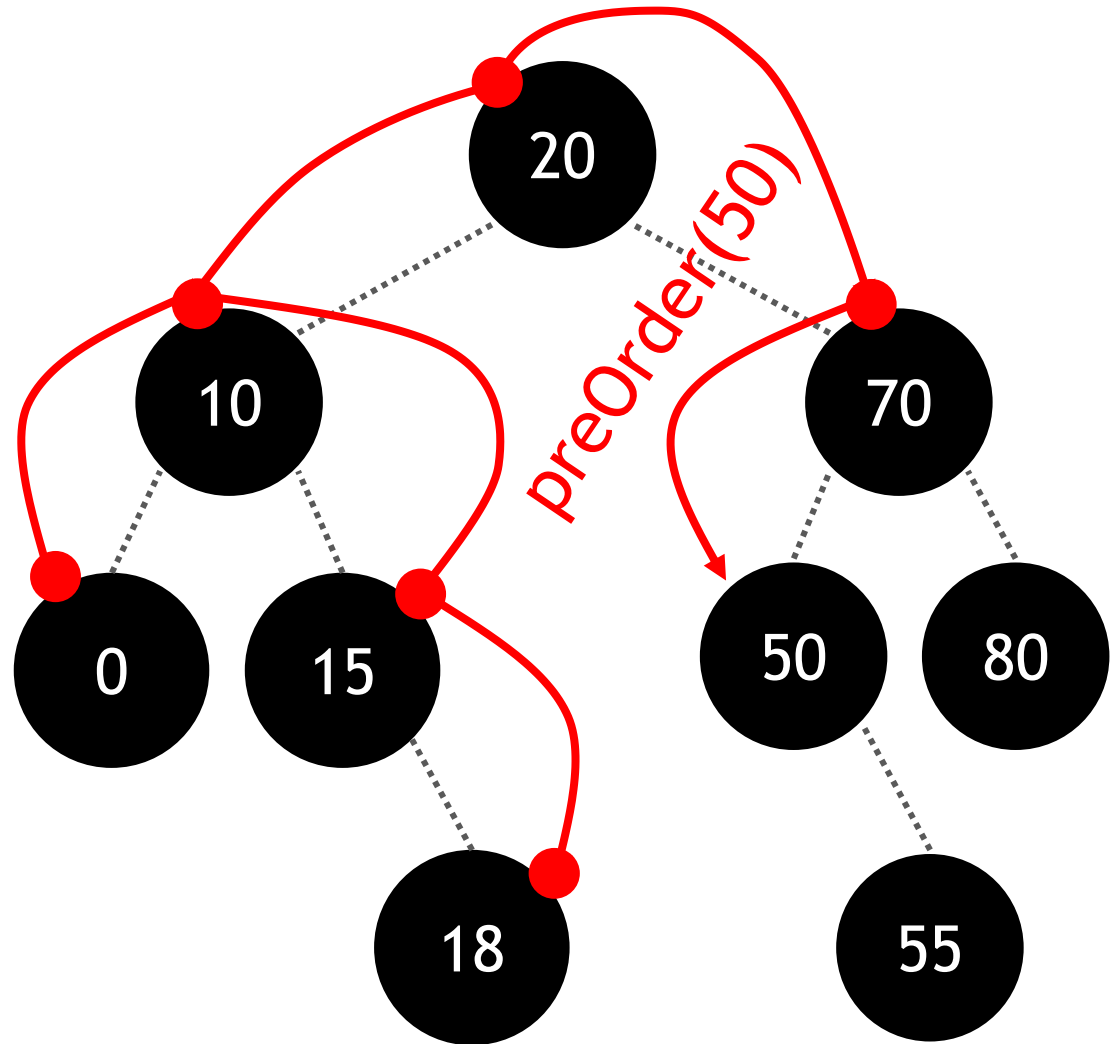
1. Visit the node.
2. `preOrder(node->smaller)`
3. `preOrder(node->greater)`



20,10,0,15,18,70

# Pre-order

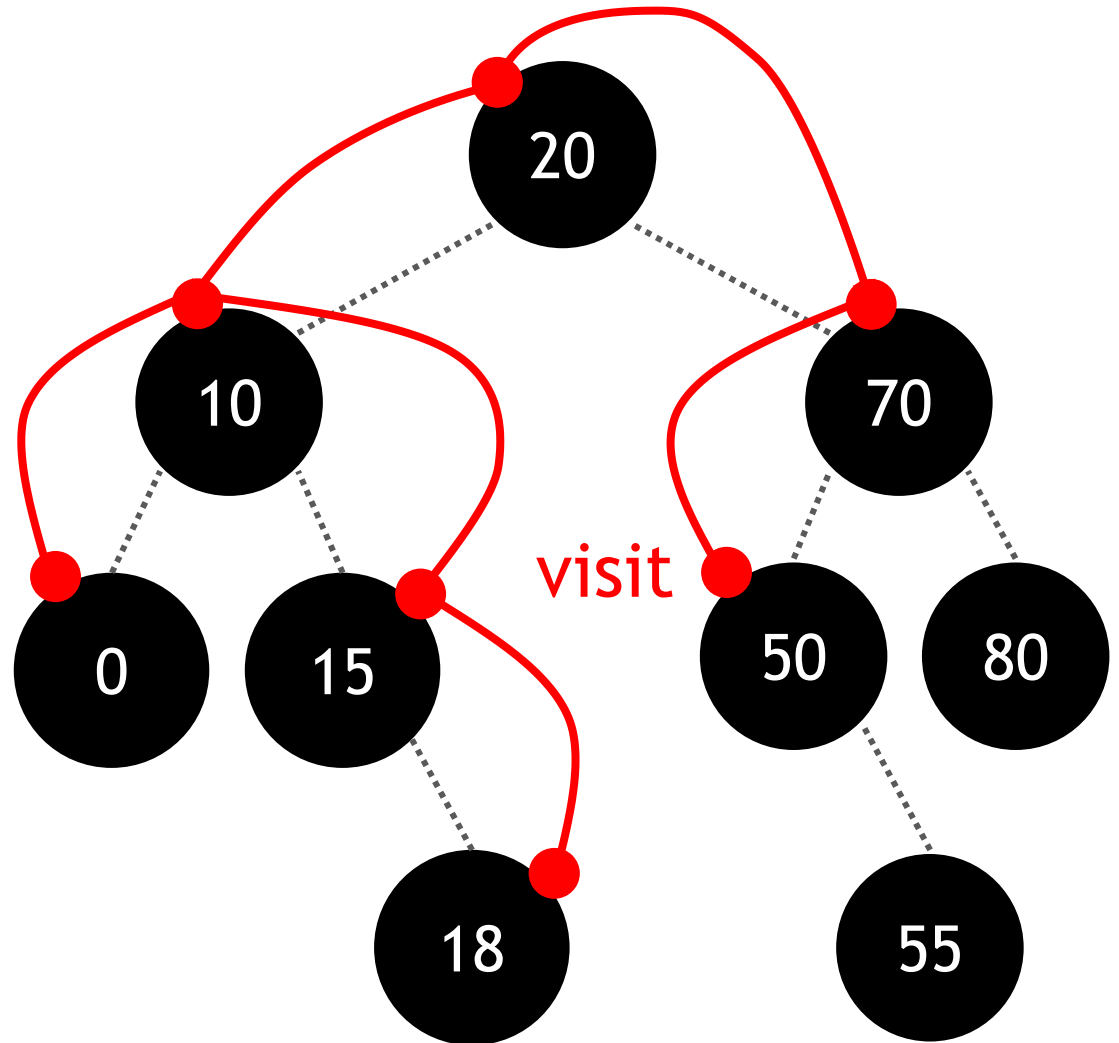
1. Visit the node.
2. `preOrder(node->smaller)`
3. `preOrder(node->greater)`



20,10,0,15,18,70

# Pre-order

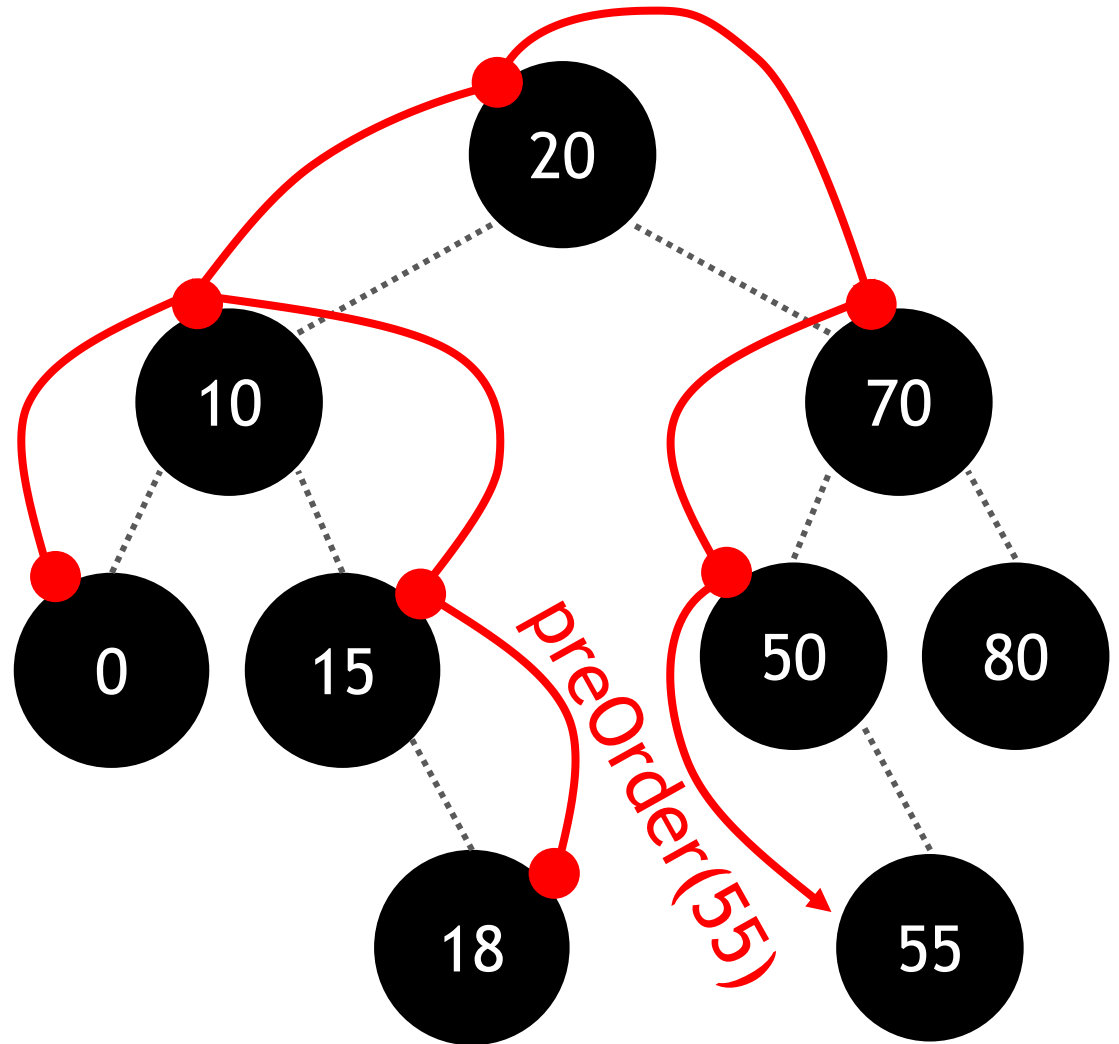
1. Visit the node.
2. `preOrder(node->smaller)`
3. `preOrder(node->greater)`



20,10,0,15,18,70,50

# Pre-order

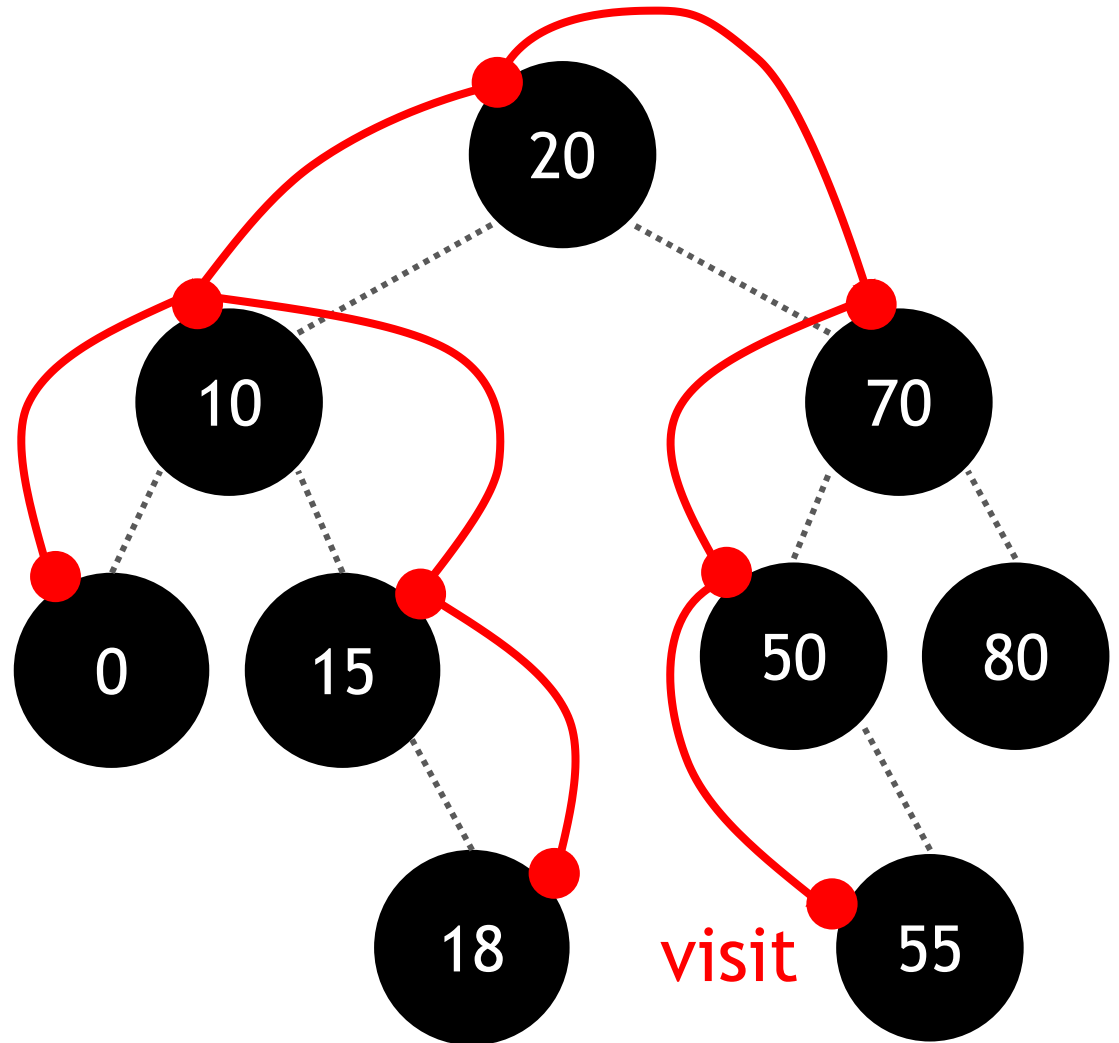
1. Visit the node.
2. `preOrder(node->smaller)`
3. `preOrder(node->greater)`



20,10,0,15,18,70,50

# Pre-order

1. Visit the node.
2. `preOrder(node->smaller)`
3. `preOrder(node->greater)`

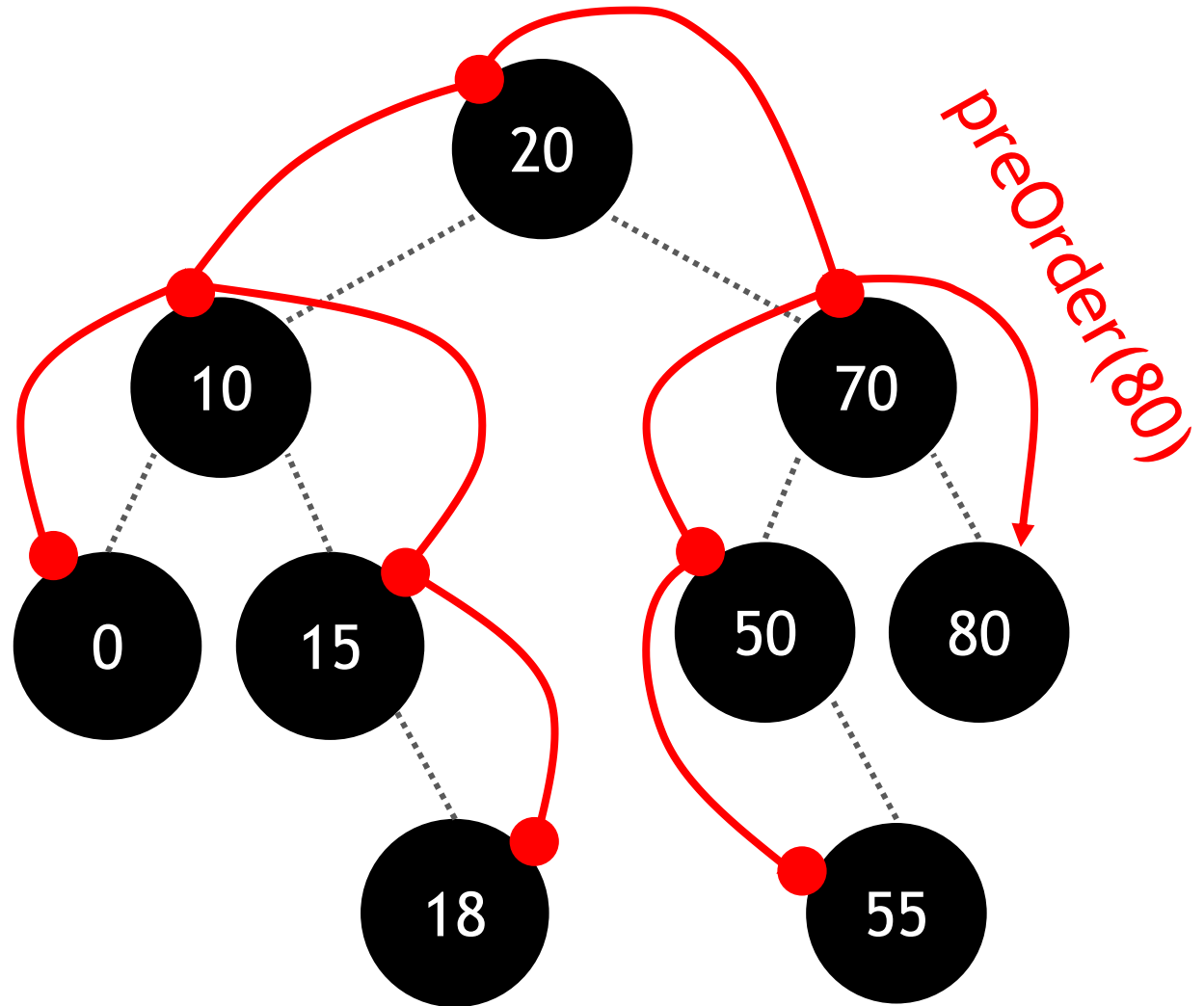


20,10,0,15,18,70,50,55



# Pre-order

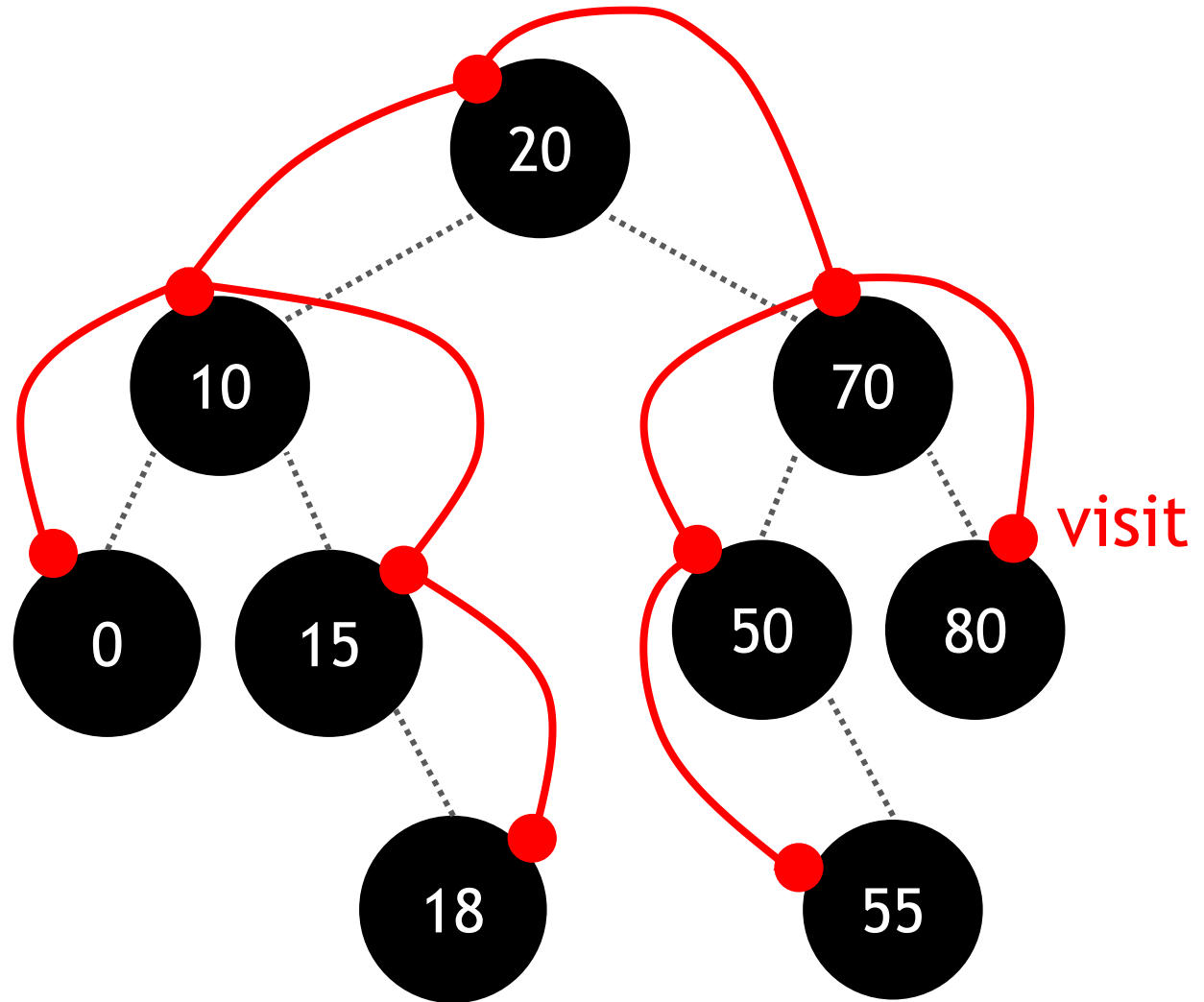
1. Visit the node.
2. `preOrder(node->smaller)`
3. `preOrder(node->greater)`



20,10,0,15,18,70,50,55

# Pre-order

1. Visit the node.
2. `preOrder(node->smaller)`
3. `preOrder(node->greater)`



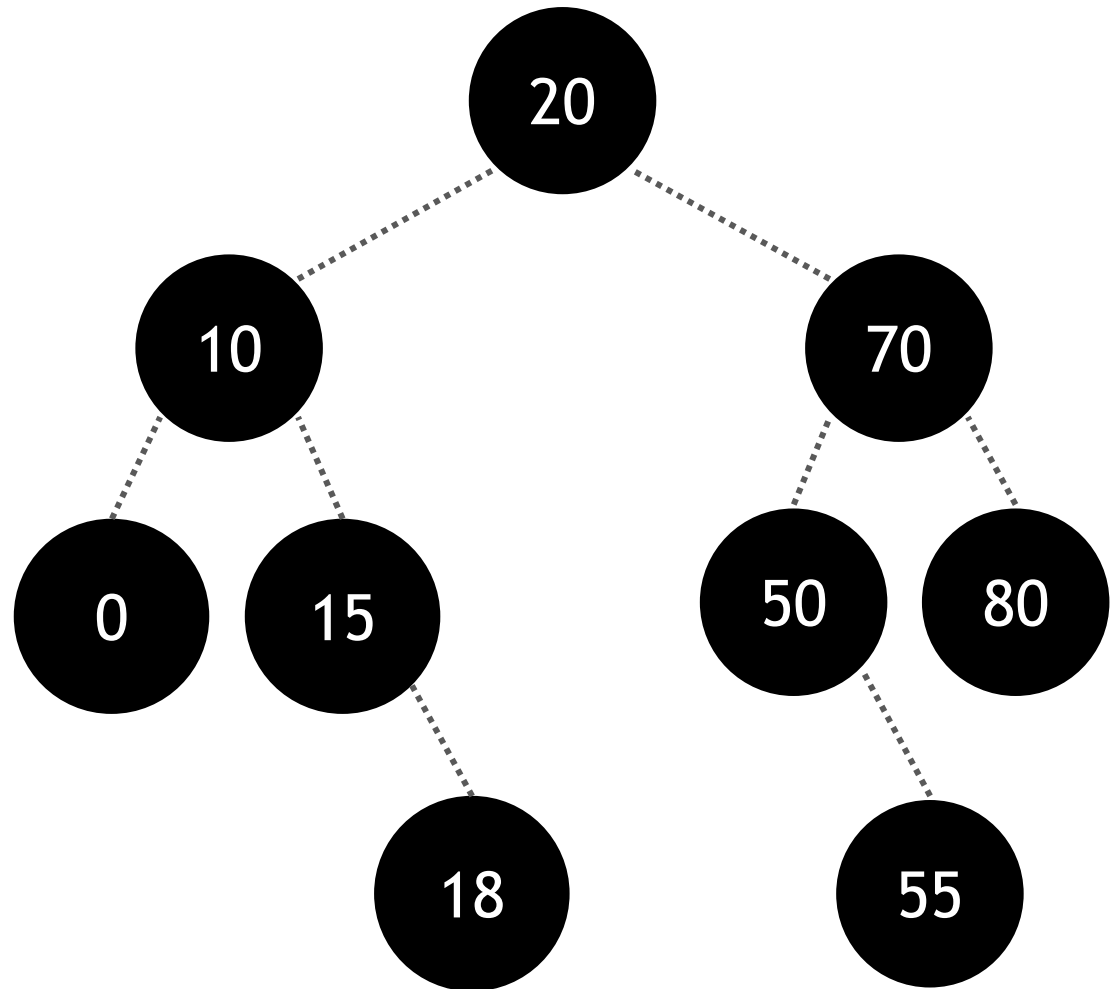
20,10,0,15,18,70,50,55,80

# In-order

## Prechod BST

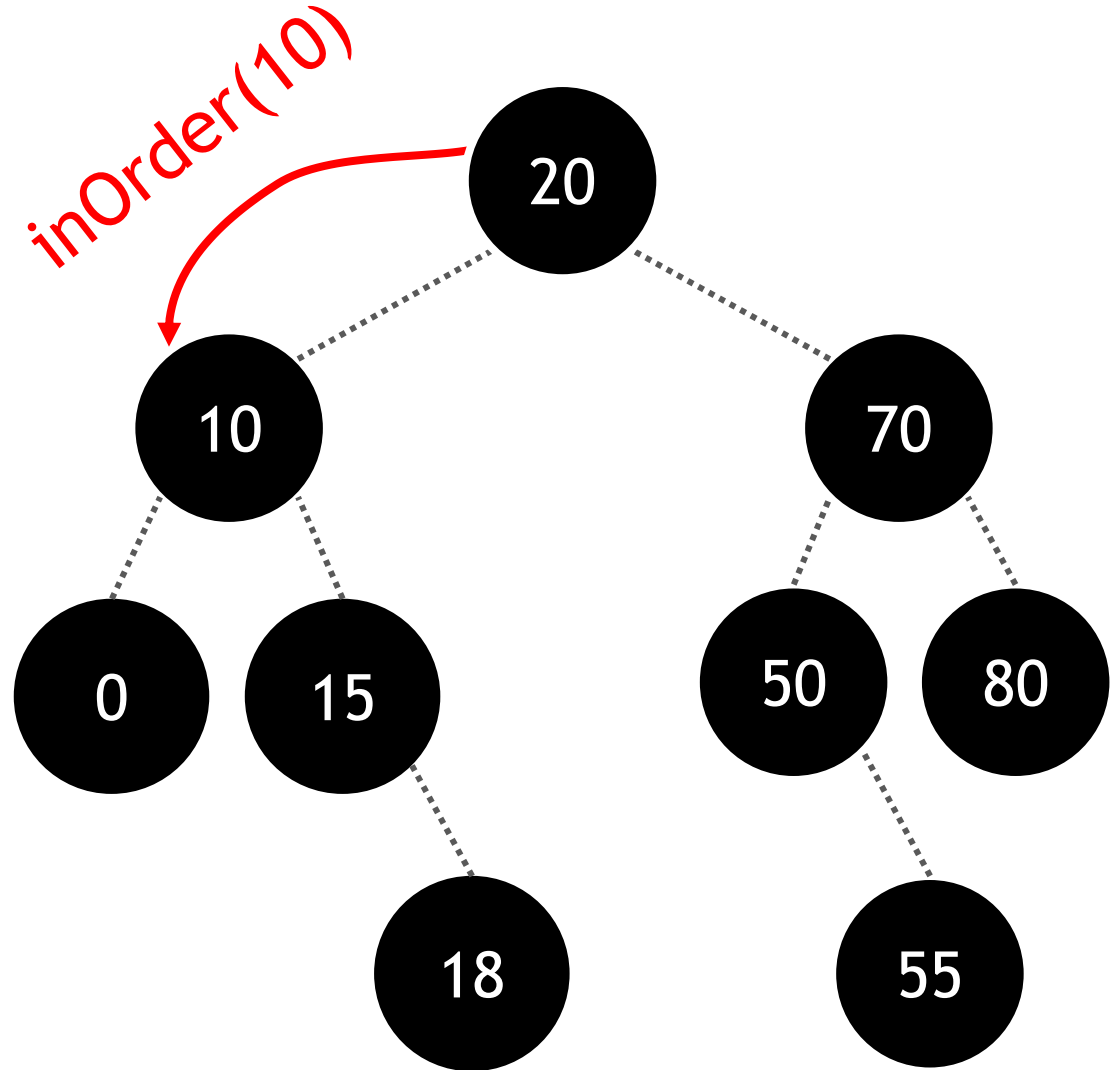
# In-order

1. inOrder(node->smaller)
2. Visit the node.
3. inOrder(node->greater)



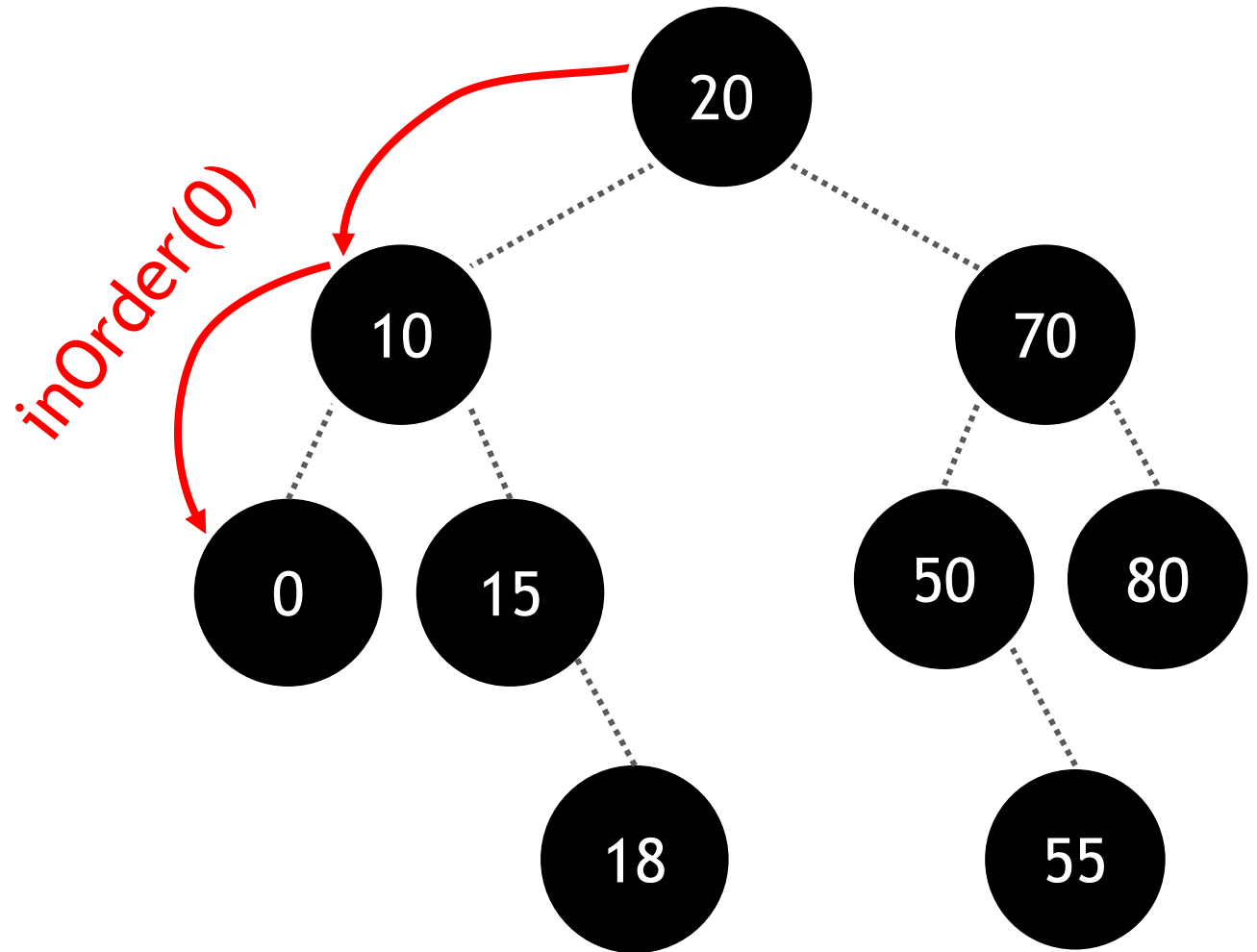
# In-order

1. inOrder(node->smaller)
2. Visit the node.
3. inOrder(node->greater)



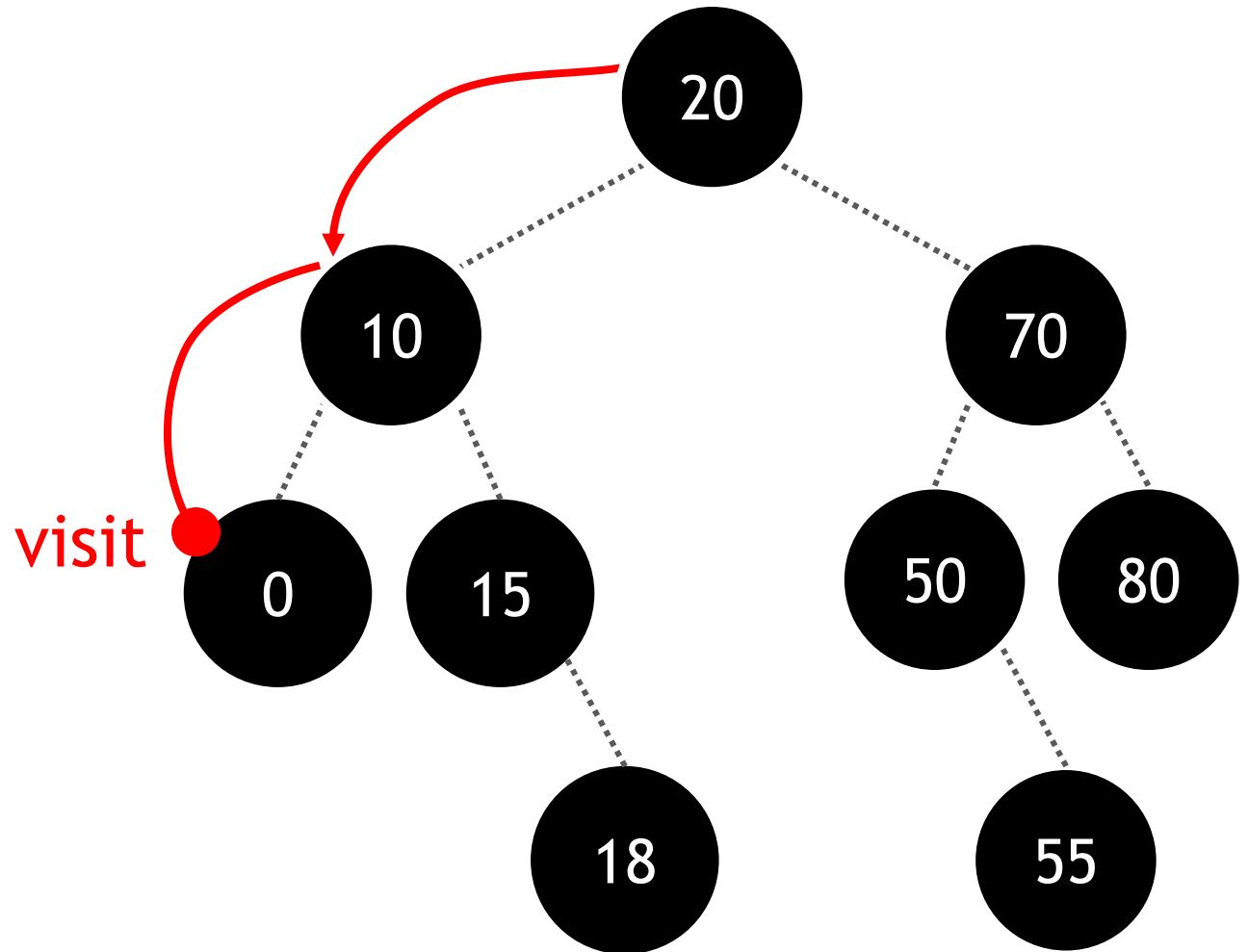
# In-order

1. inOrder(node->smaller)
2. Visit the node.
3. inOrder(node->greater)



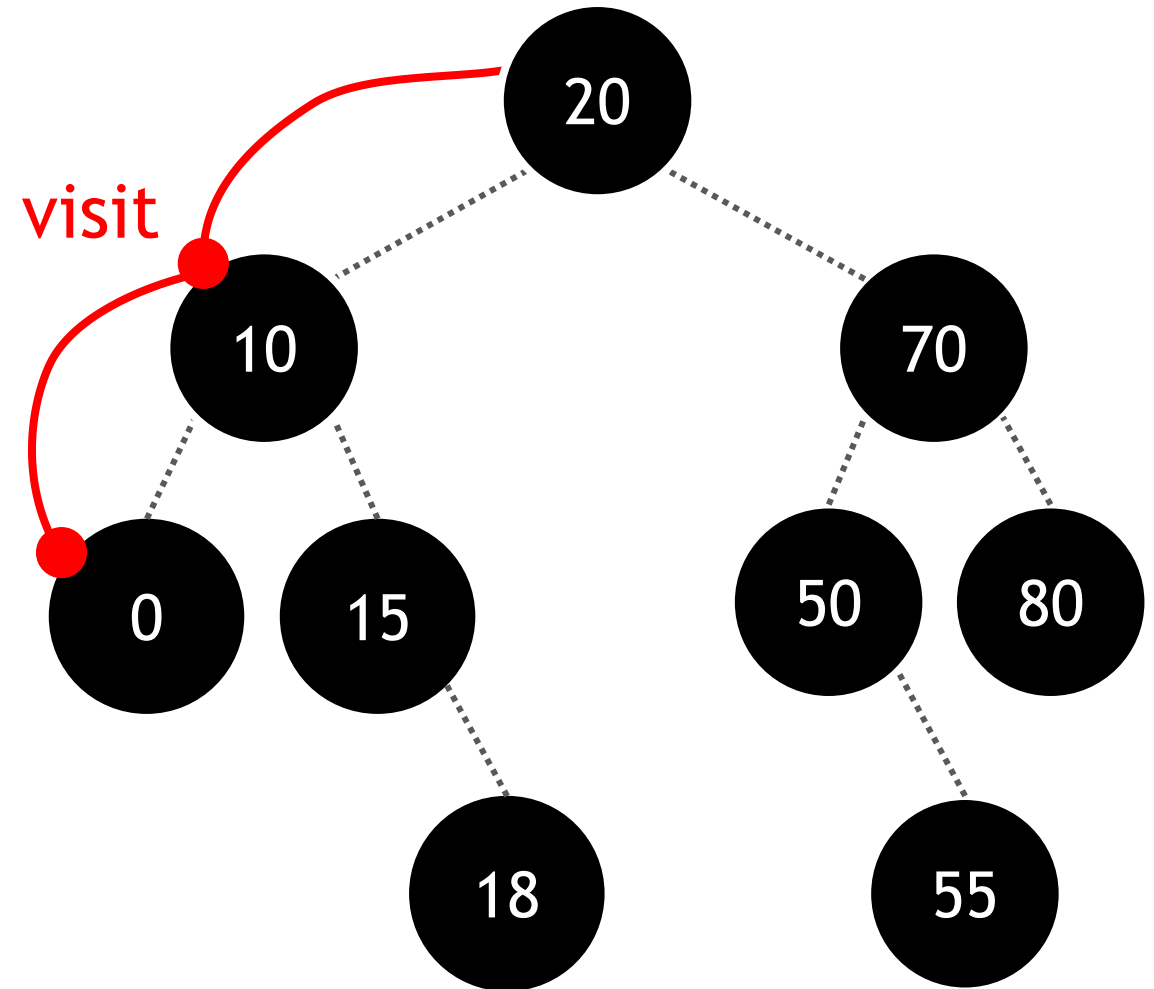
# In-order

1. inOrder(node->smaller)
2. Visit the node.
3. inOrder(node->greater)



# In-order

1. inOrder(node->smaller)
2. Visit the node.
3. inOrder(node->greater)

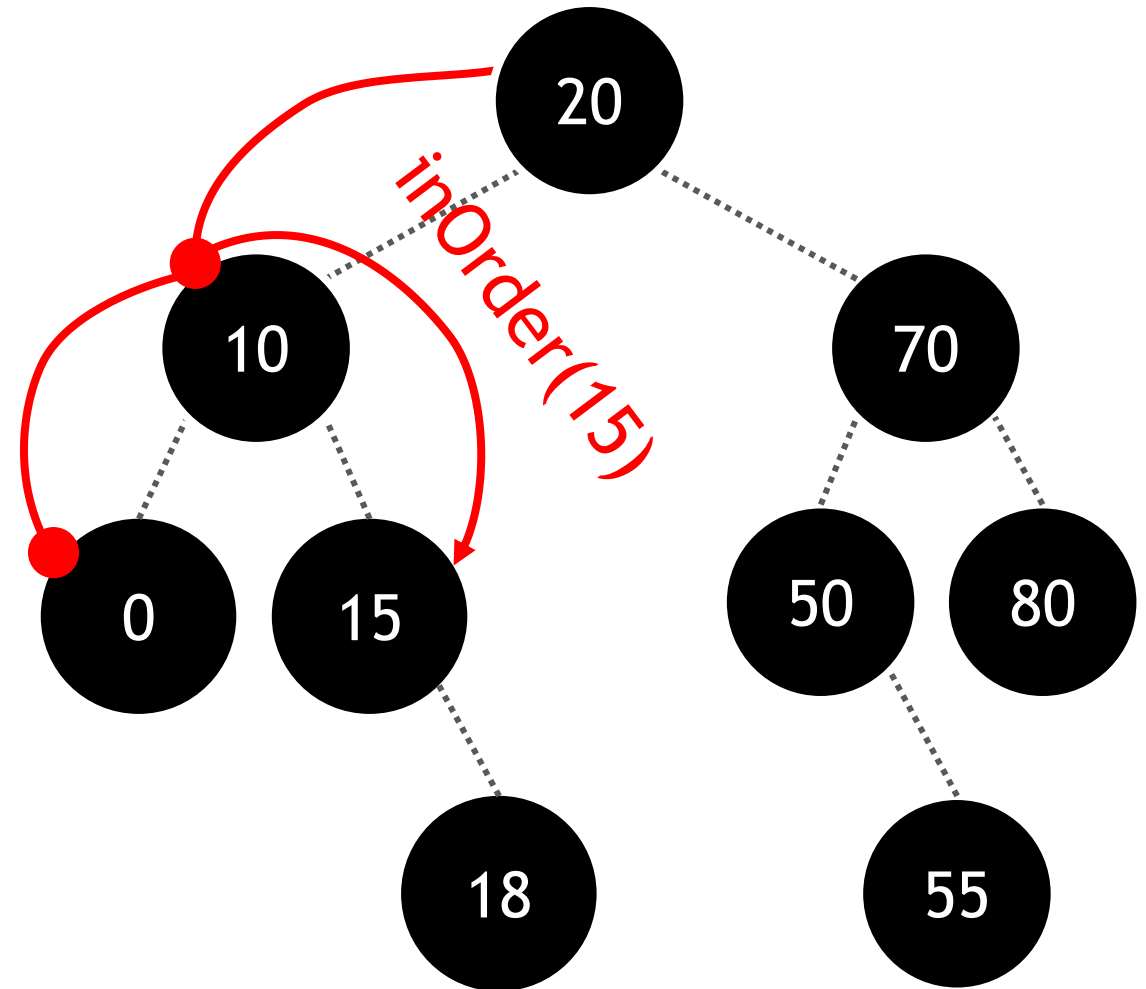


0,10,



# In-order

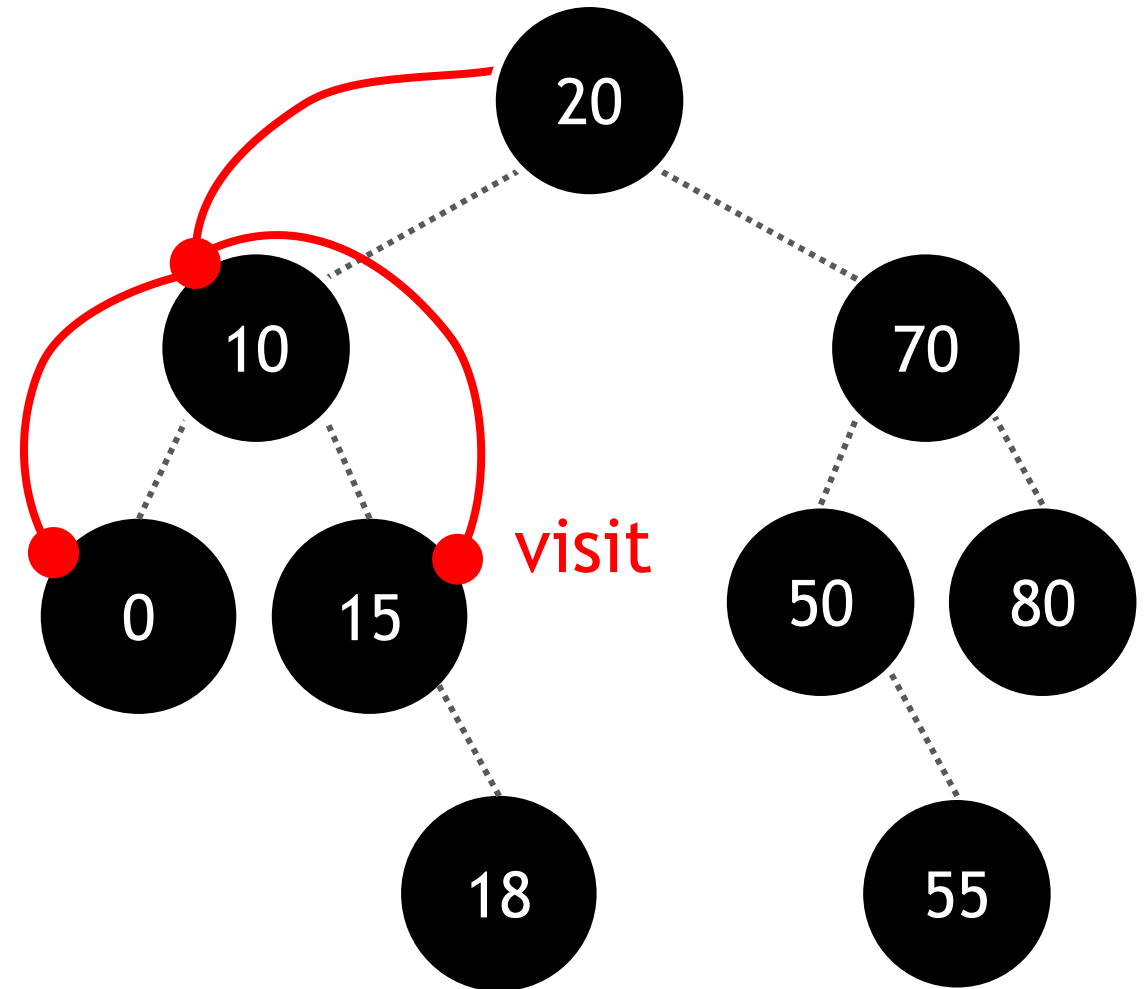
1. inOrder(node->smaller)
2. Visit the node.
3. inOrder(node->greater)



0,10,

# In-order

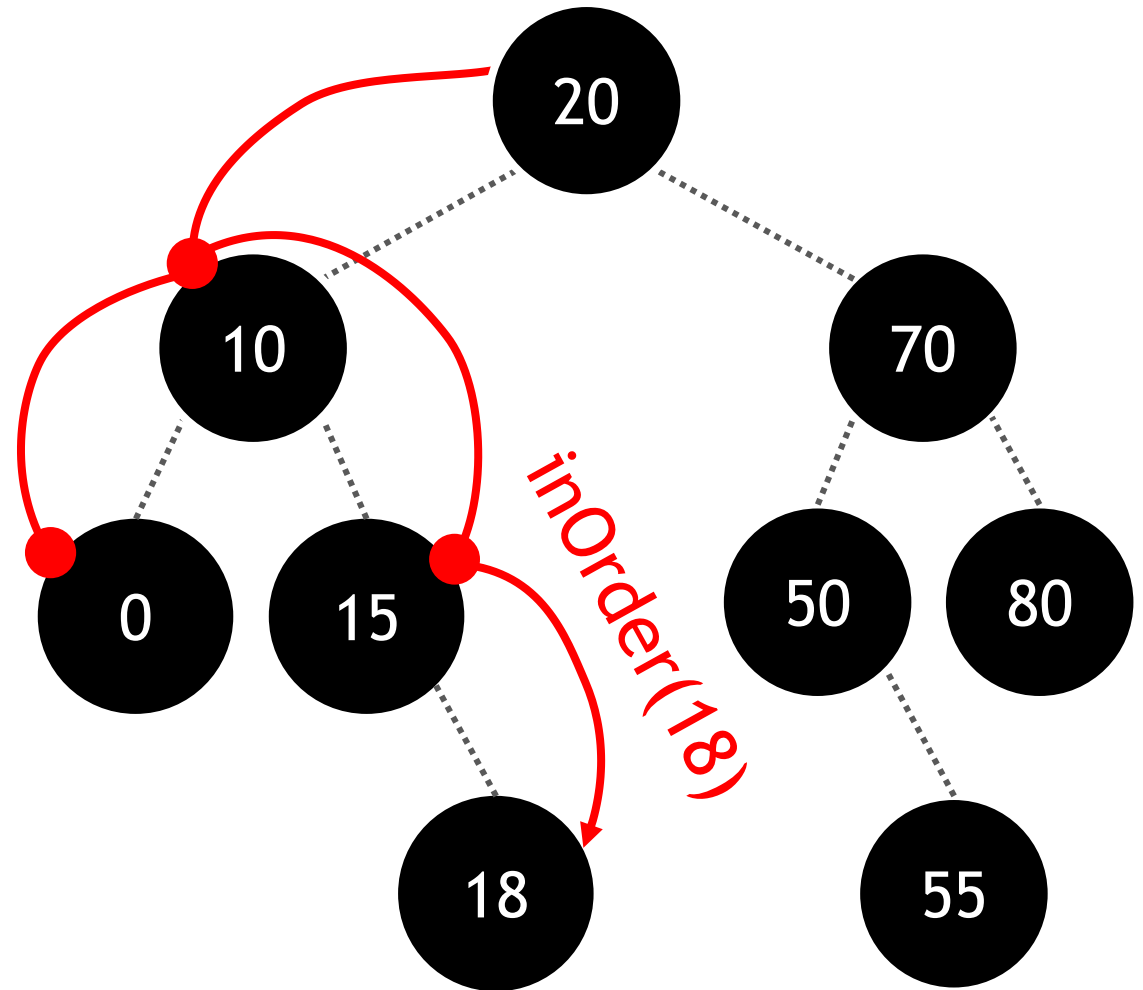
1. inOrder(node->smaller)
2. Visit the node.
3. inOrder(node->greater)



0,10,15,

# In-order

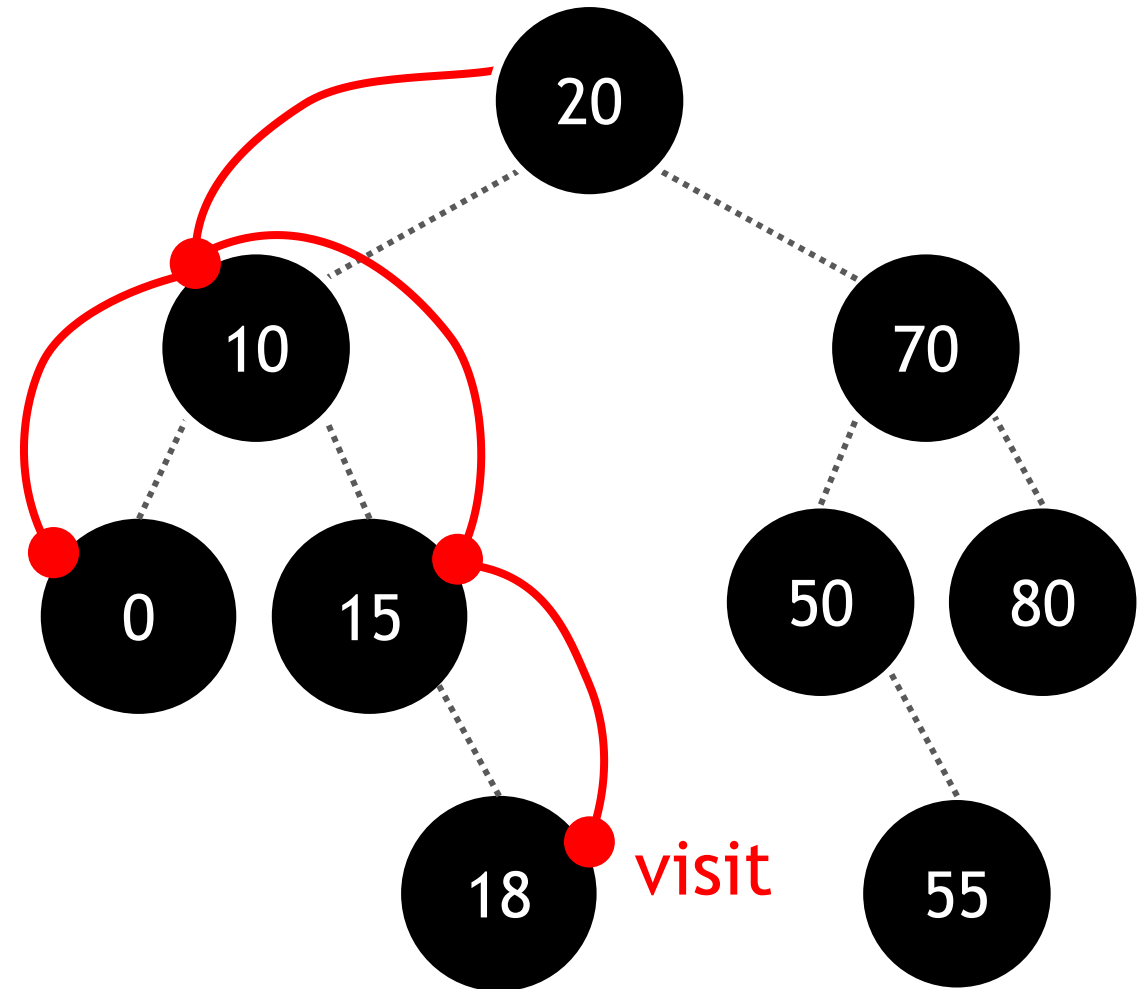
1. inOrder(node->smaller)
2. Visit the node.
3. inOrder(node->greater)



0,10,15,

# In-order

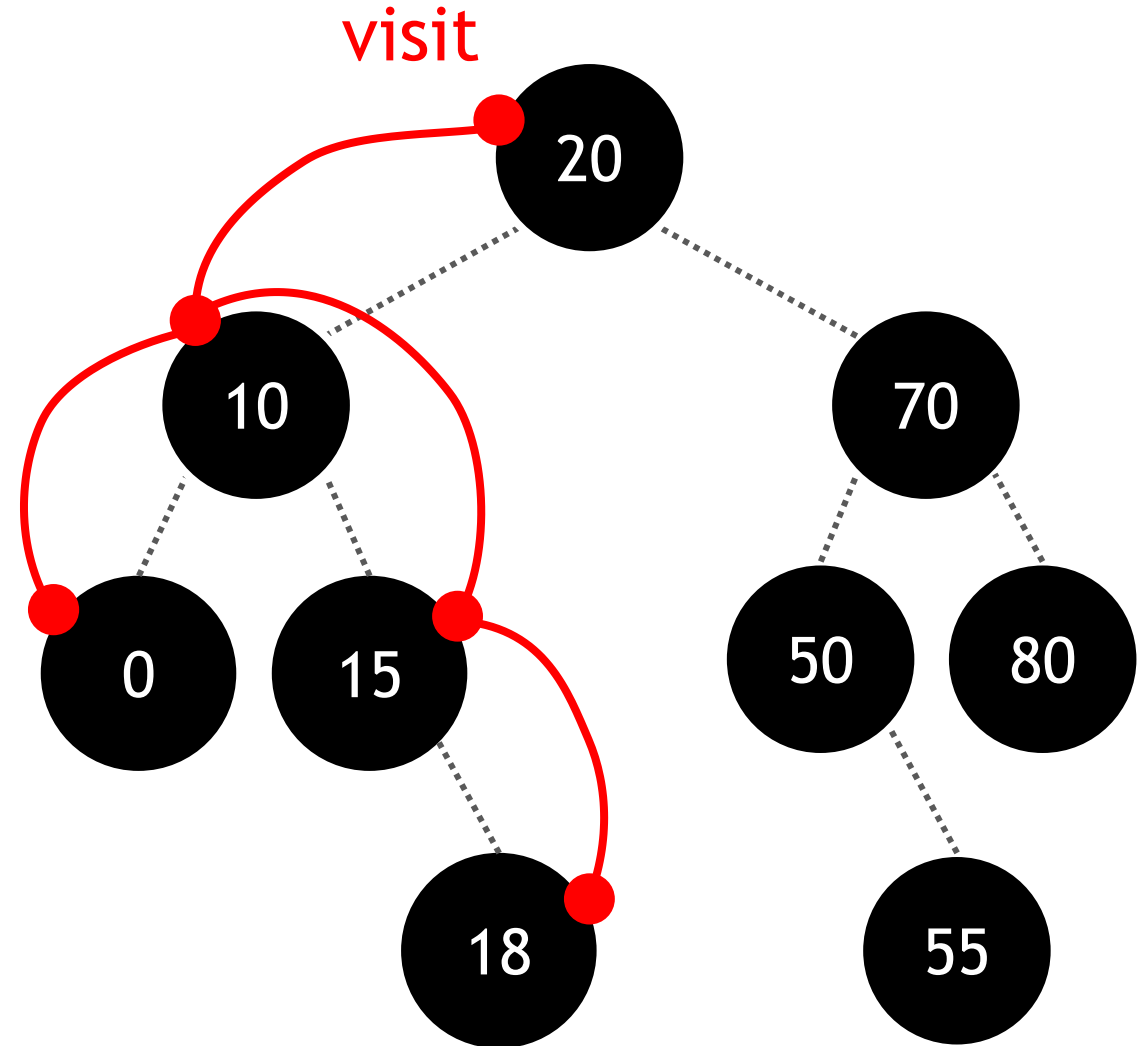
1. inOrder(node->smaller)
2. Visit the node.
3. inOrder(node->greater)



0,10,15,18

# In-order

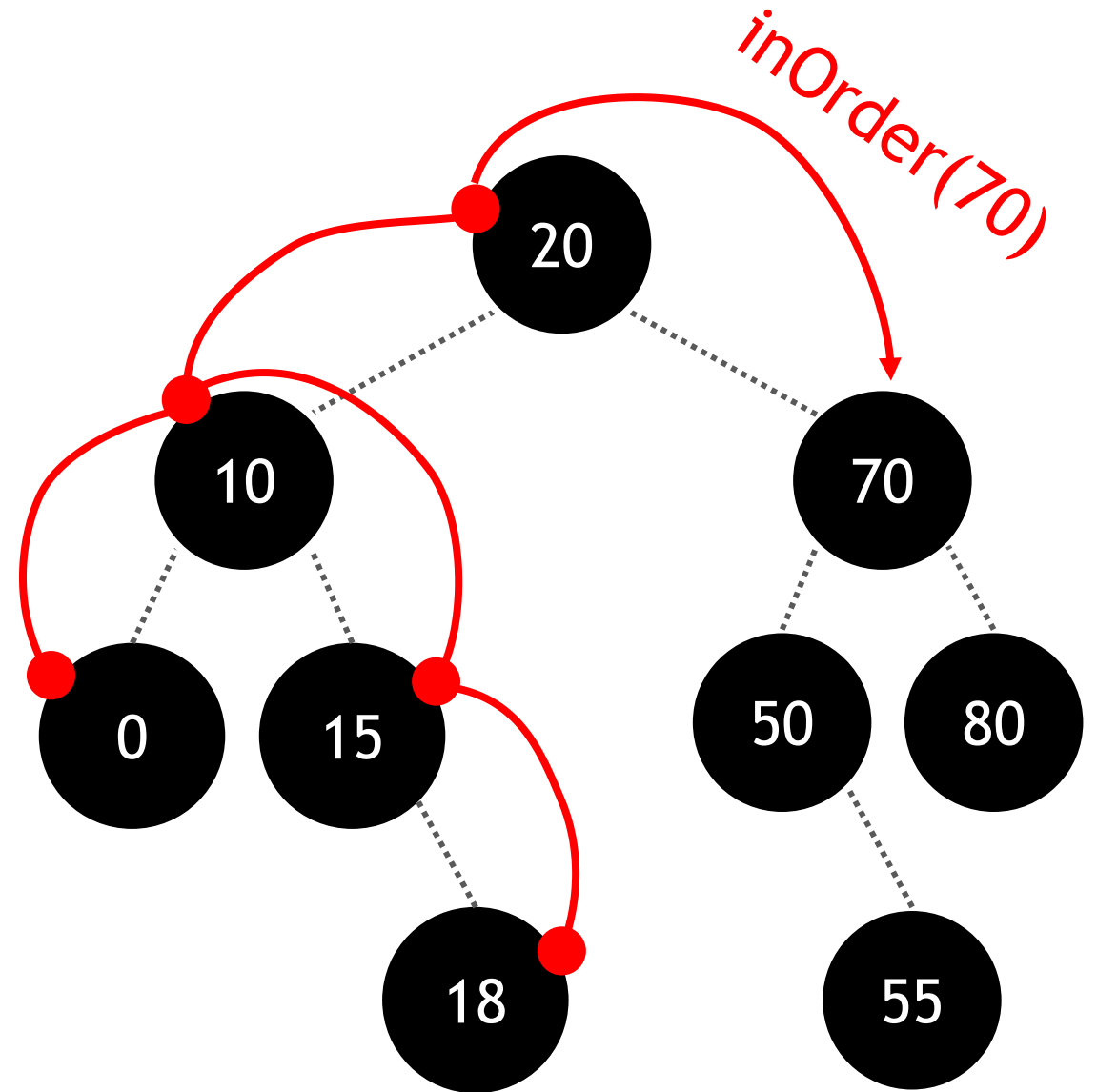
1. inOrder(node->smaller)
2. Visit the node.
3. inOrder(node->greater)



0,10,15,18,20,

# In-order

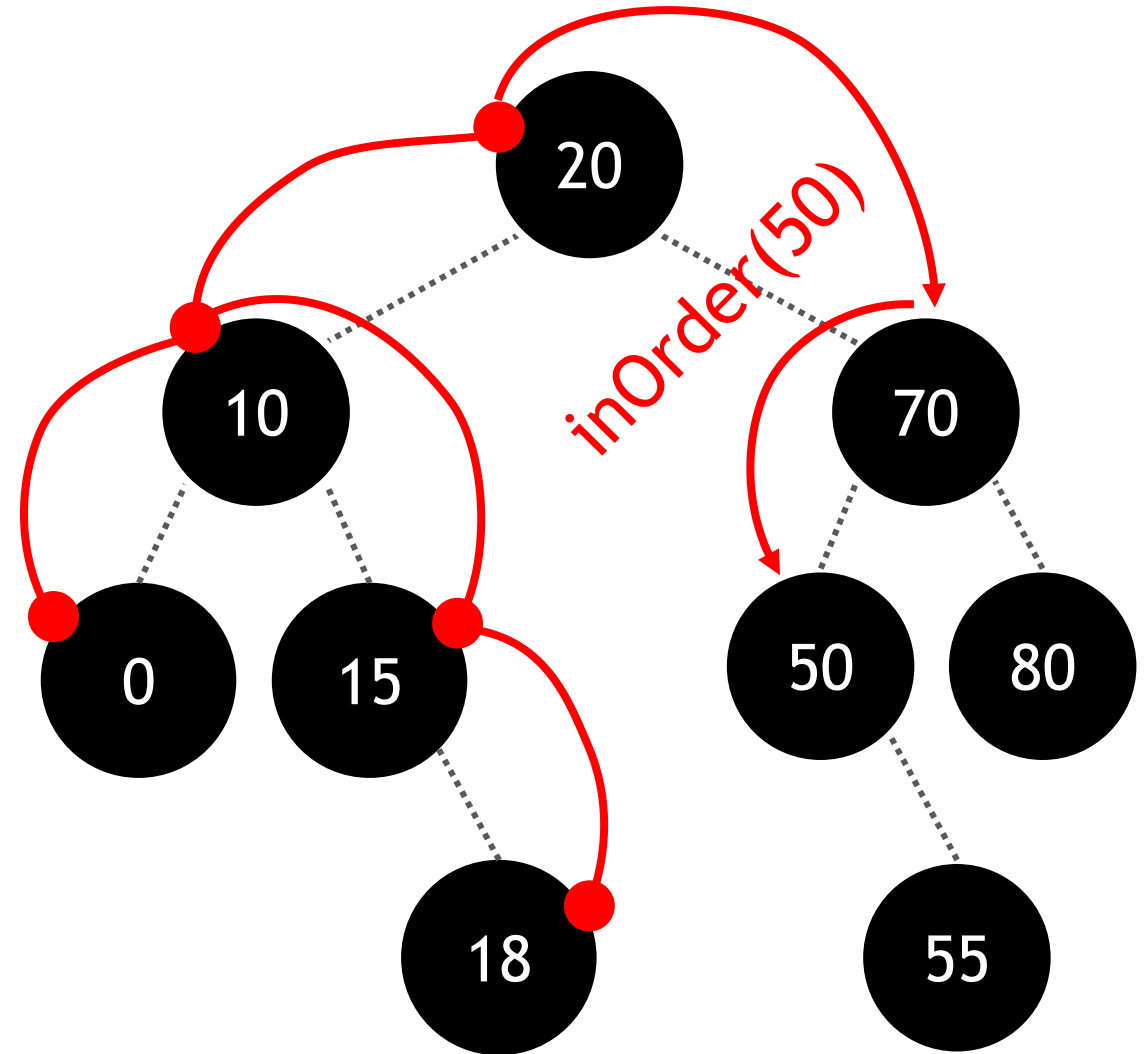
1. inOrder(node->smaller)
2. Visit the node.
3. inOrder(node->greater)



0,10,15,18,20,

# In-order

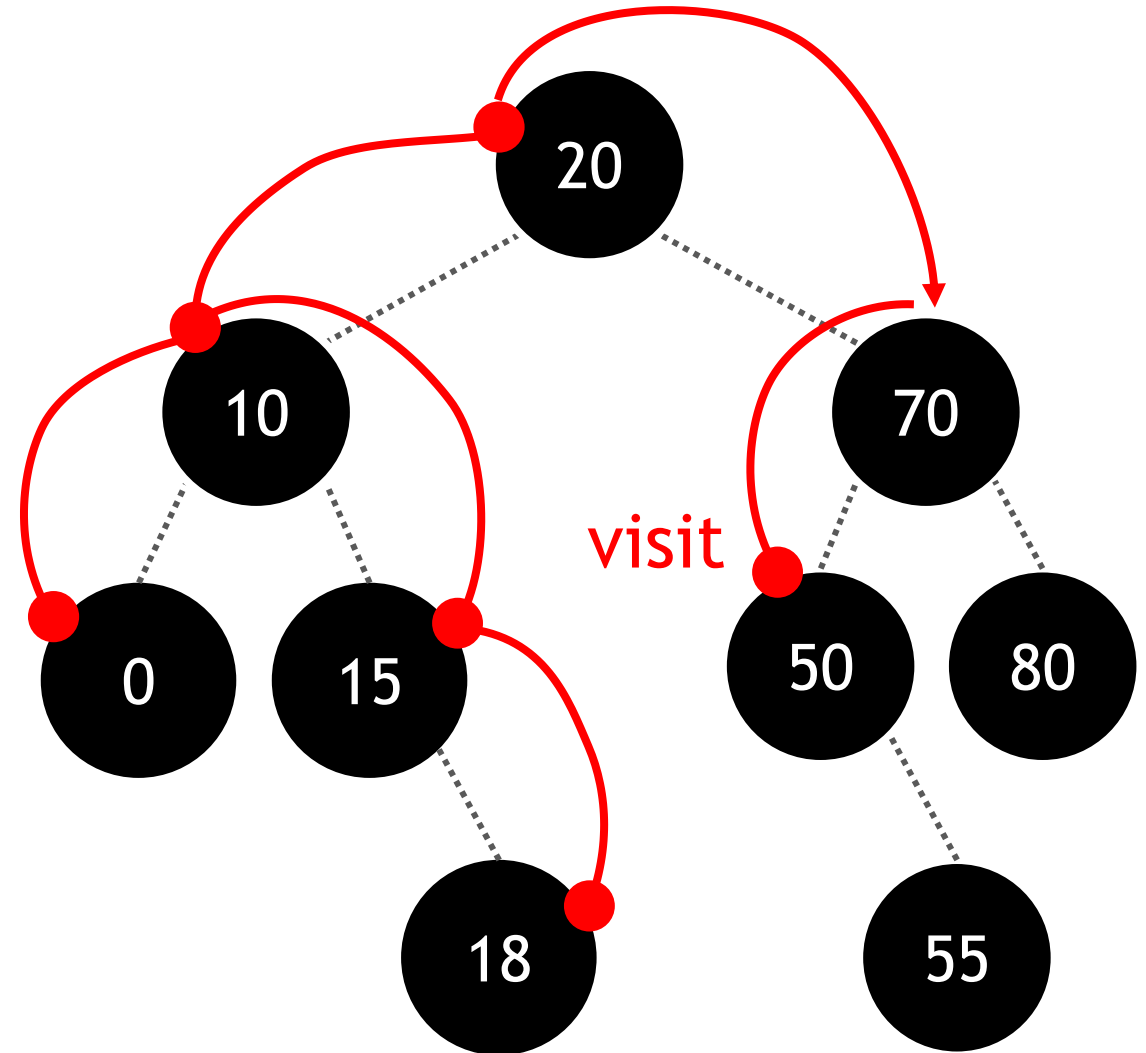
1. inOrder(node->smaller)
2. Visit the node.
3. inOrder(node->greater)



0,10,15,18,20,

# In-order

1. inOrder(node->smaller)
2. Visit the node.
3. inOrder(node->greater)

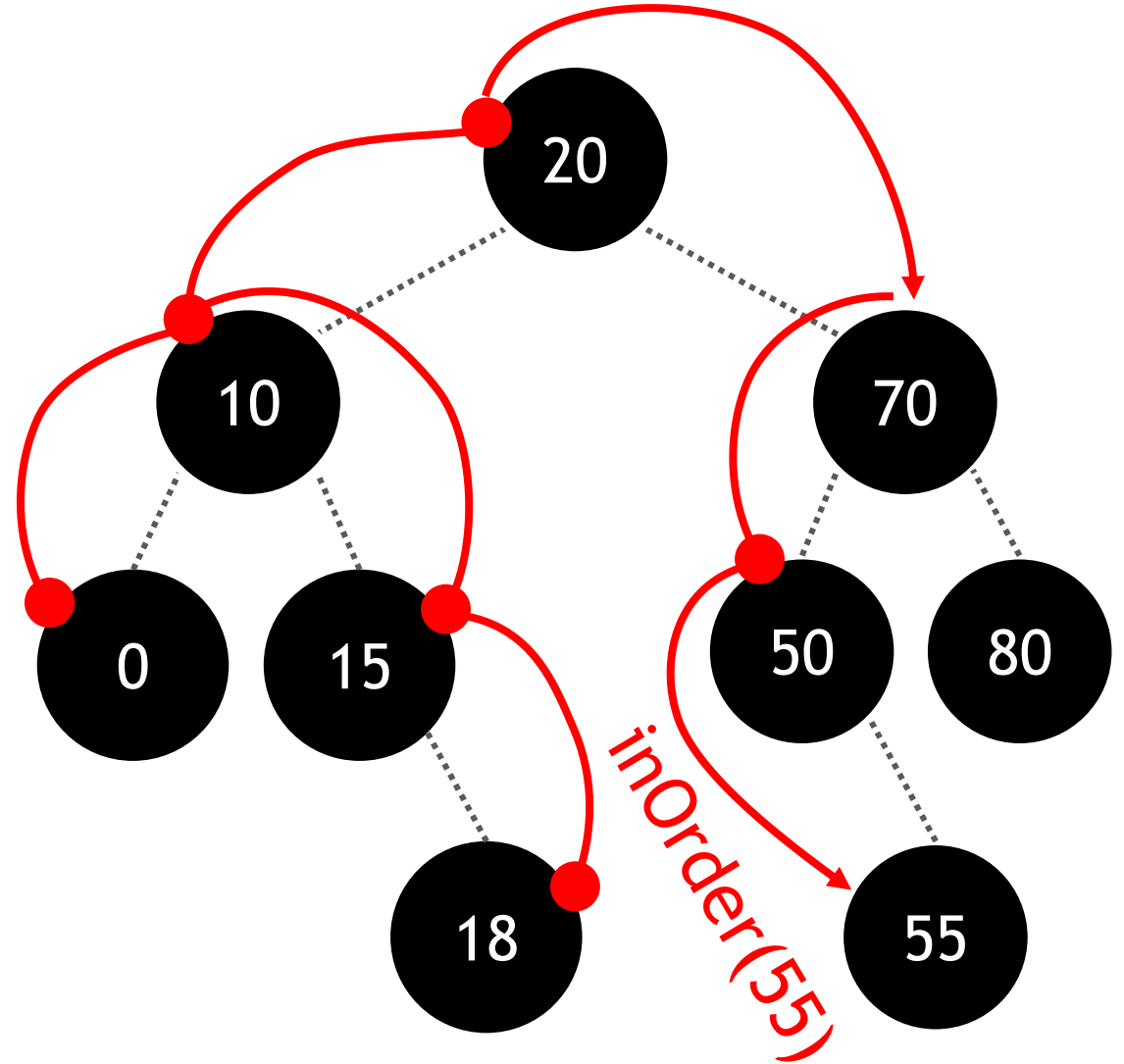


0,10,15,18,20,50,



# In-order

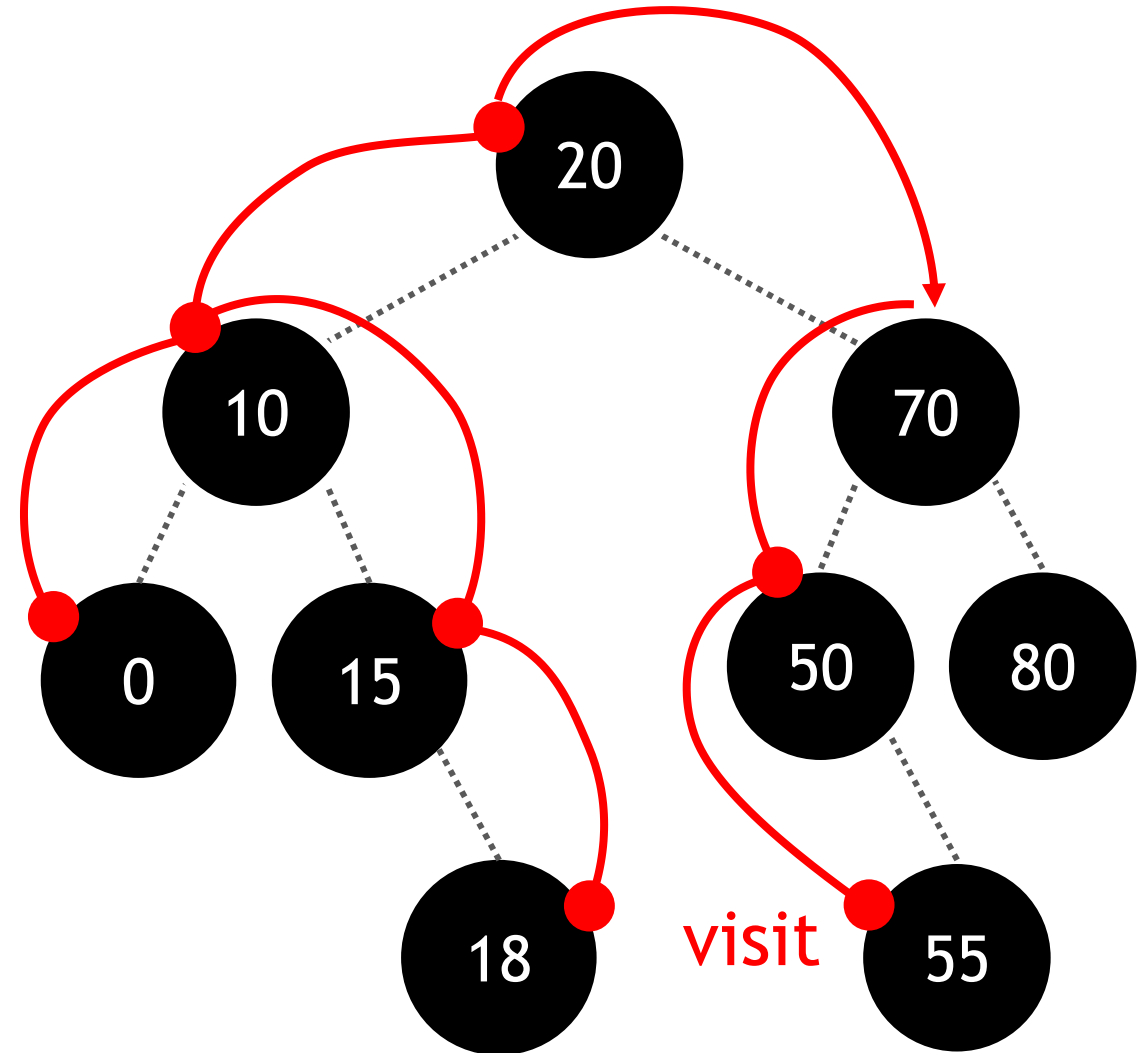
1. inOrder(node->smaller)
2. Visit the node.
3. inOrder(node->greater)



0,10,15,18,20,50,

# In-order

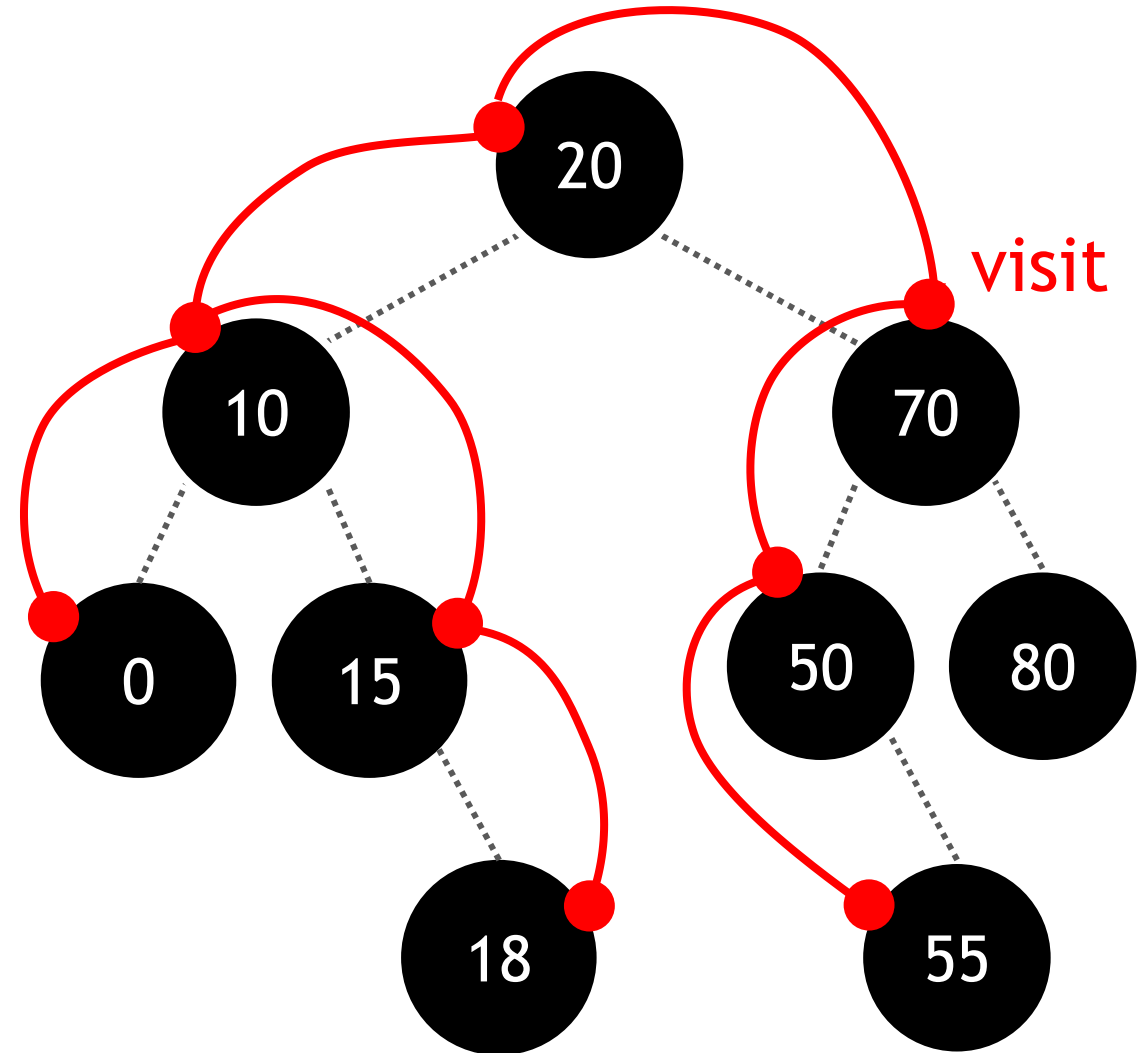
1. inOrder(node->smaller)
2. Visit the node.
3. inOrder(node->greater)



0,10,15,18,20,50,55

# In-order

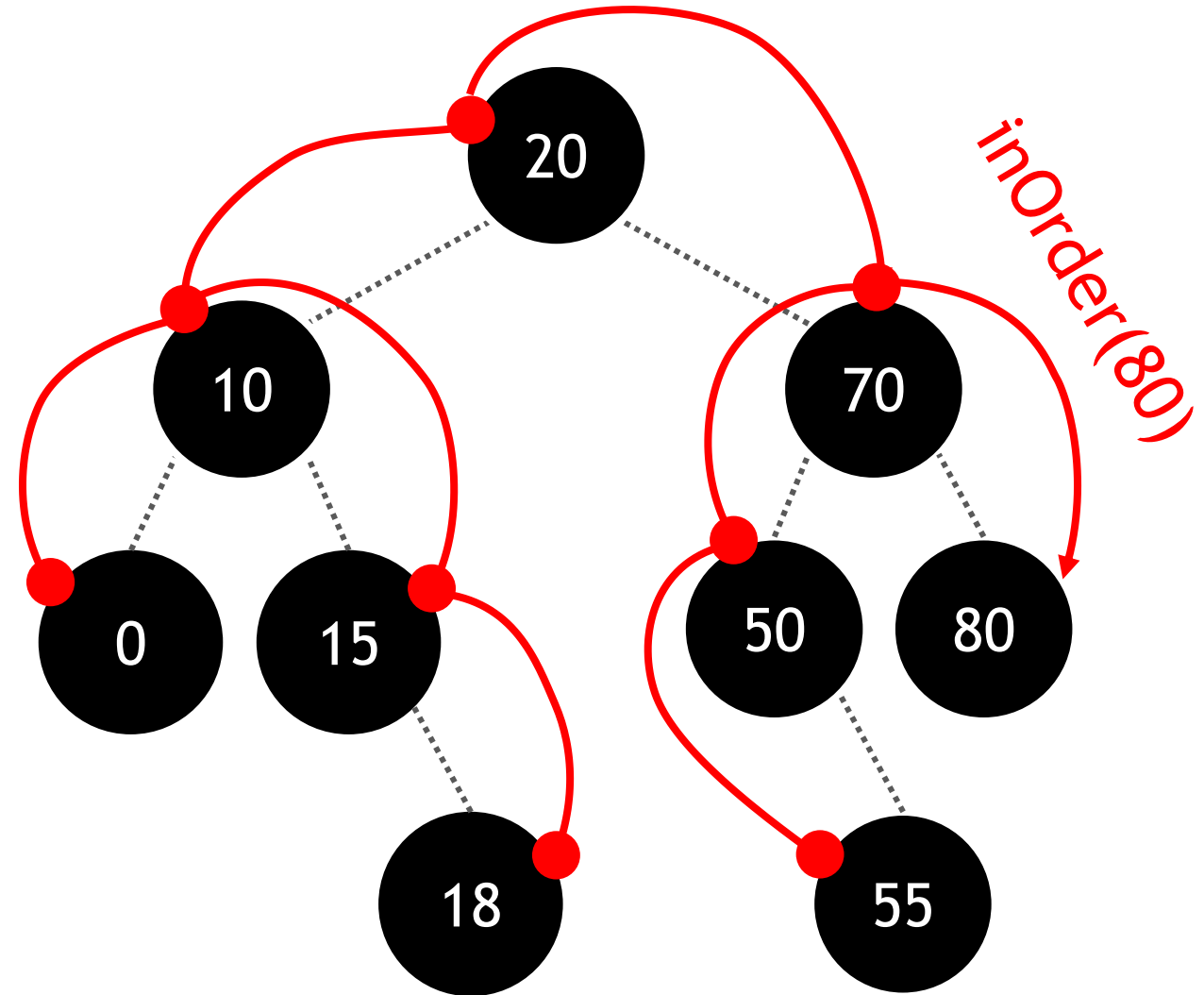
1. inOrder(node->smaller)
2. Visit the node.
3. inOrder(node->greater)



0,10,15,18,20,50,55,70,

# In-order

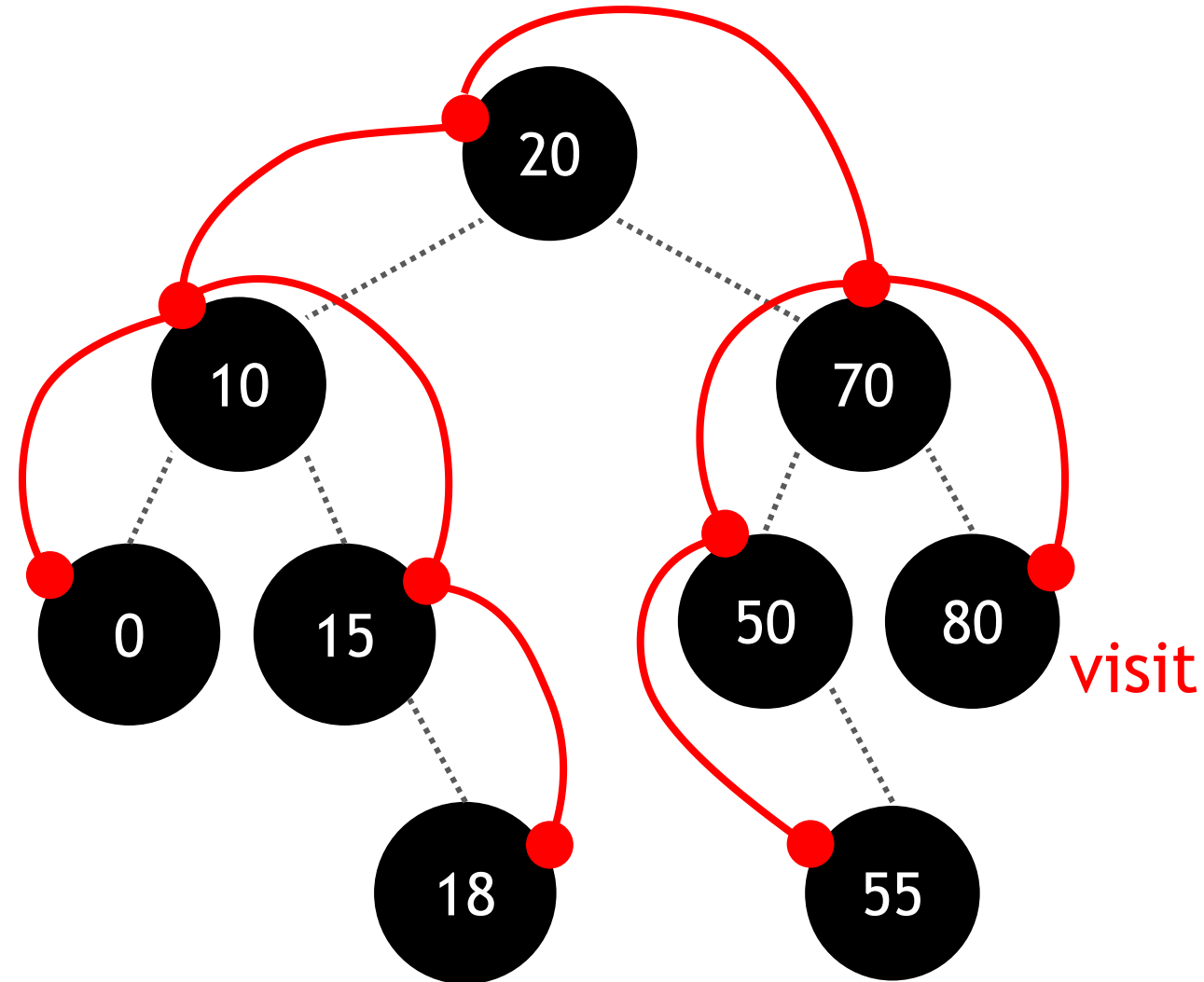
1. inOrder(node->smaller)
2. Visit the node.
3. inOrder(node->greater)



0,10,15,18,20,50,55,70,

# In-order

1. inOrder(node->smaller)
2. Visit the node.
3. inOrder(node->greater)



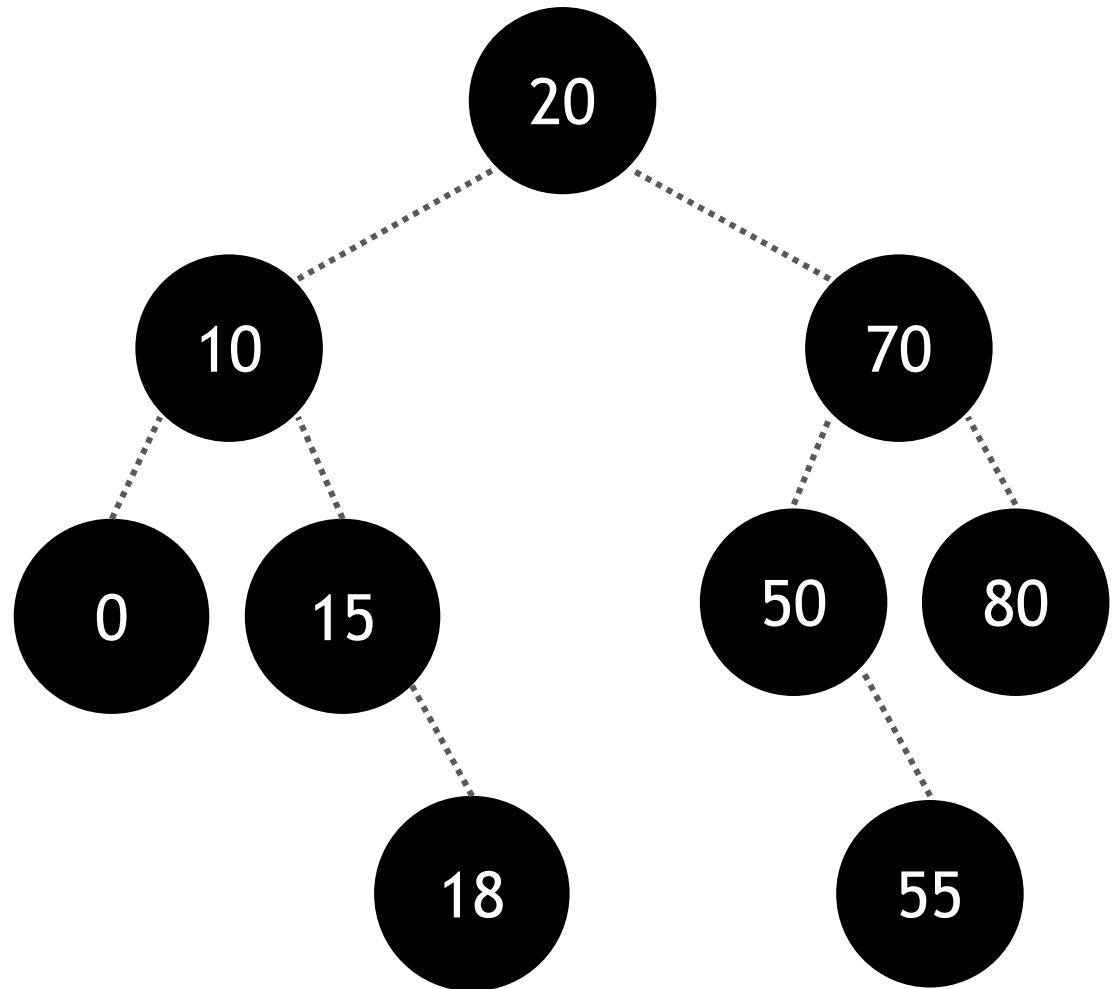
0,10,15,18,20,50,55,70,80

# Post-order

## Prechod BST

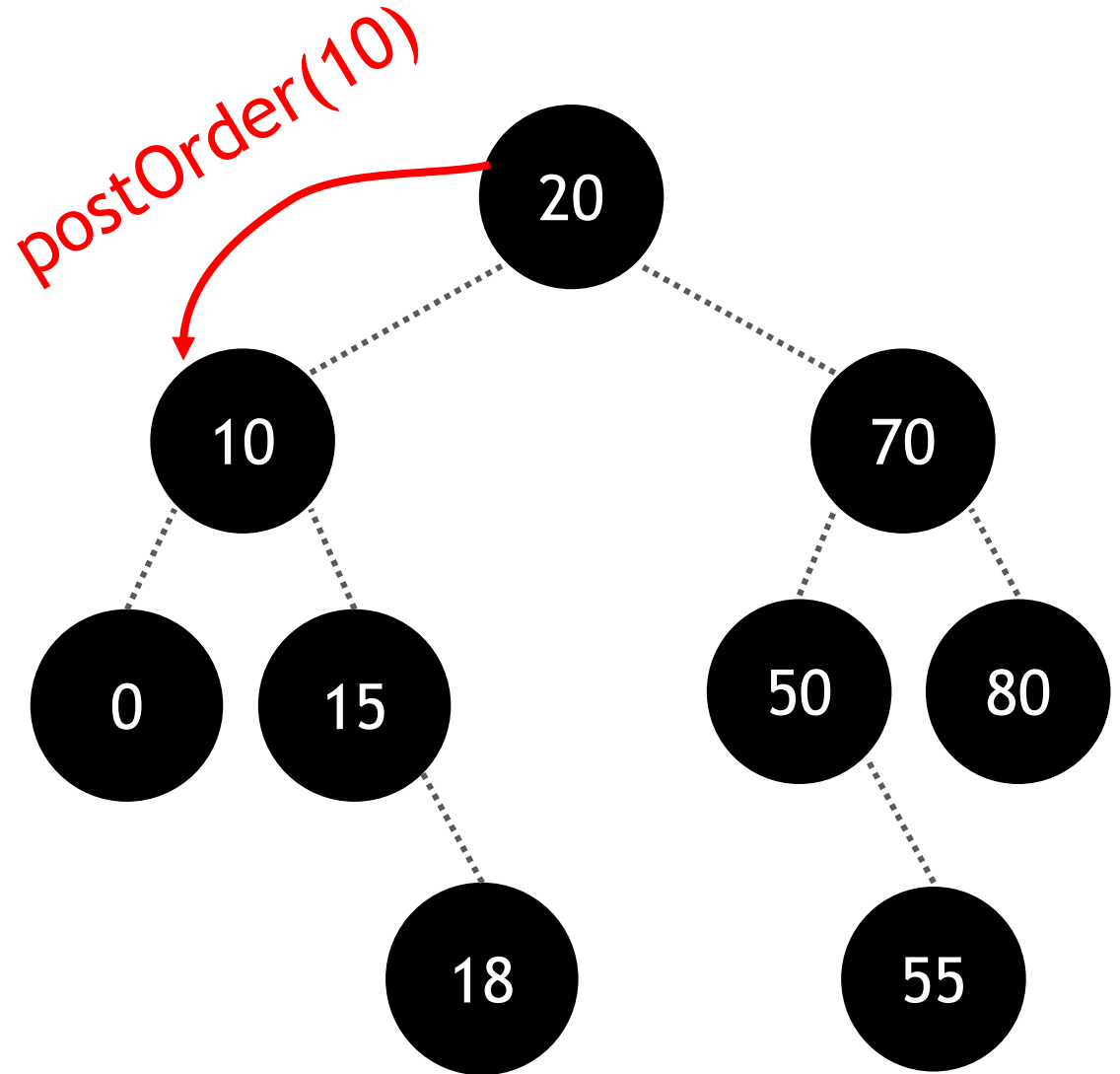
# Post-order

1. `postOrder(node->smaller)`
2. `postOrder(node->greater)`
3. Visit the node.



# Post-order

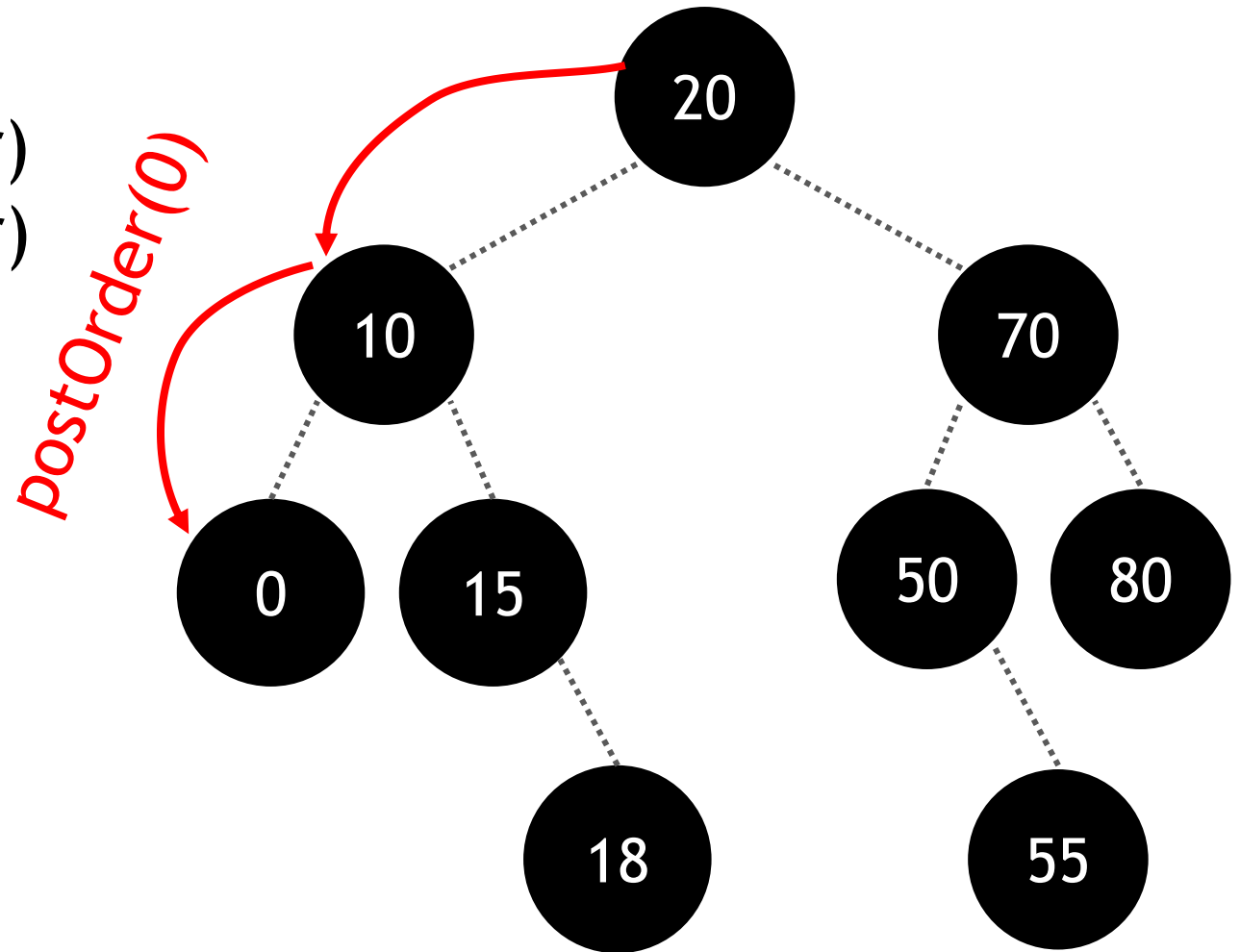
1. `postOrder(node->smaller)`
2. `postOrder(node->greater)`
3. Visit the node.





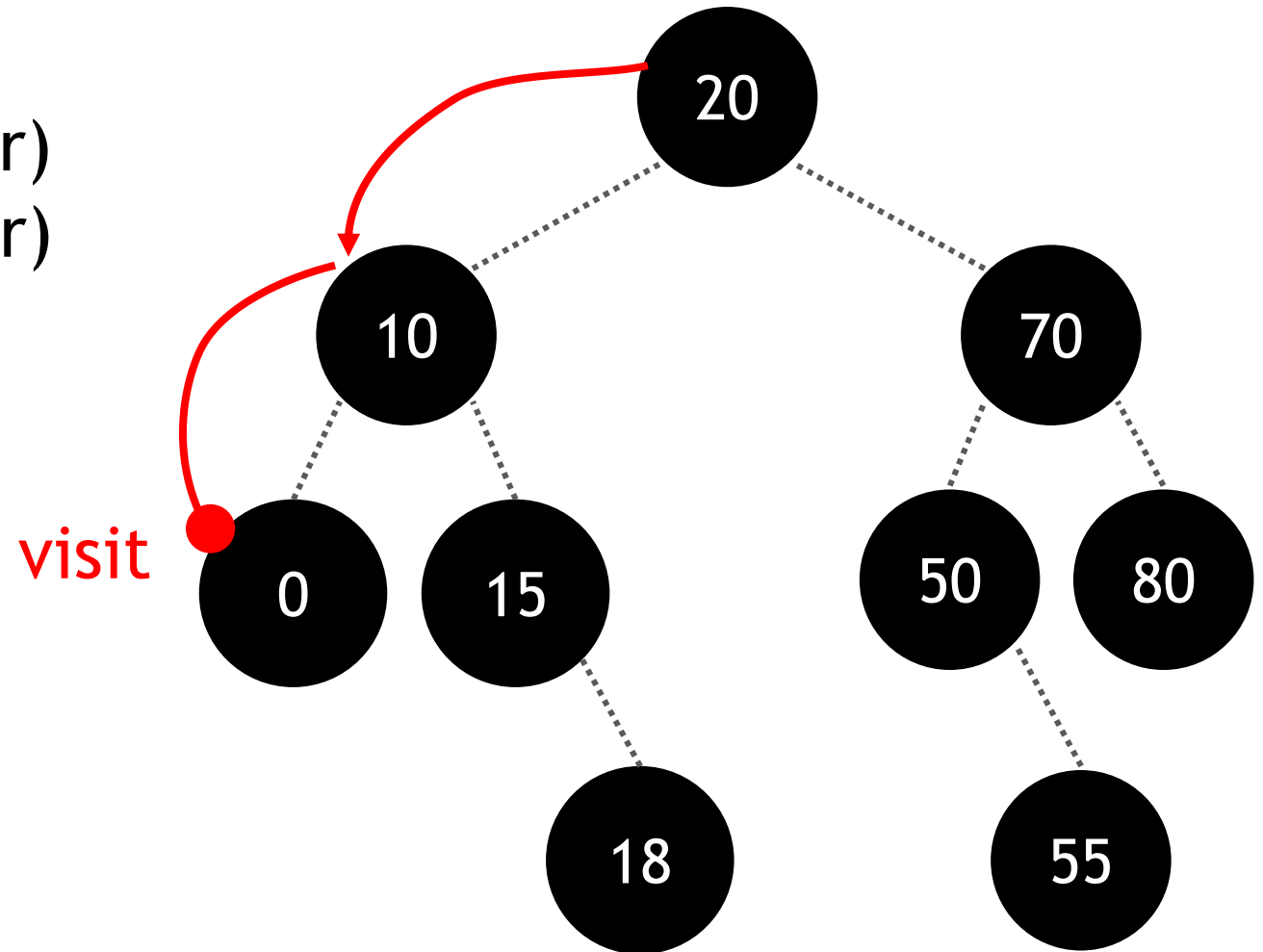
# Post-order

1. `postOrder(node->smaller)`
2. `postOrder(node->greater)`
3. Visit the node.



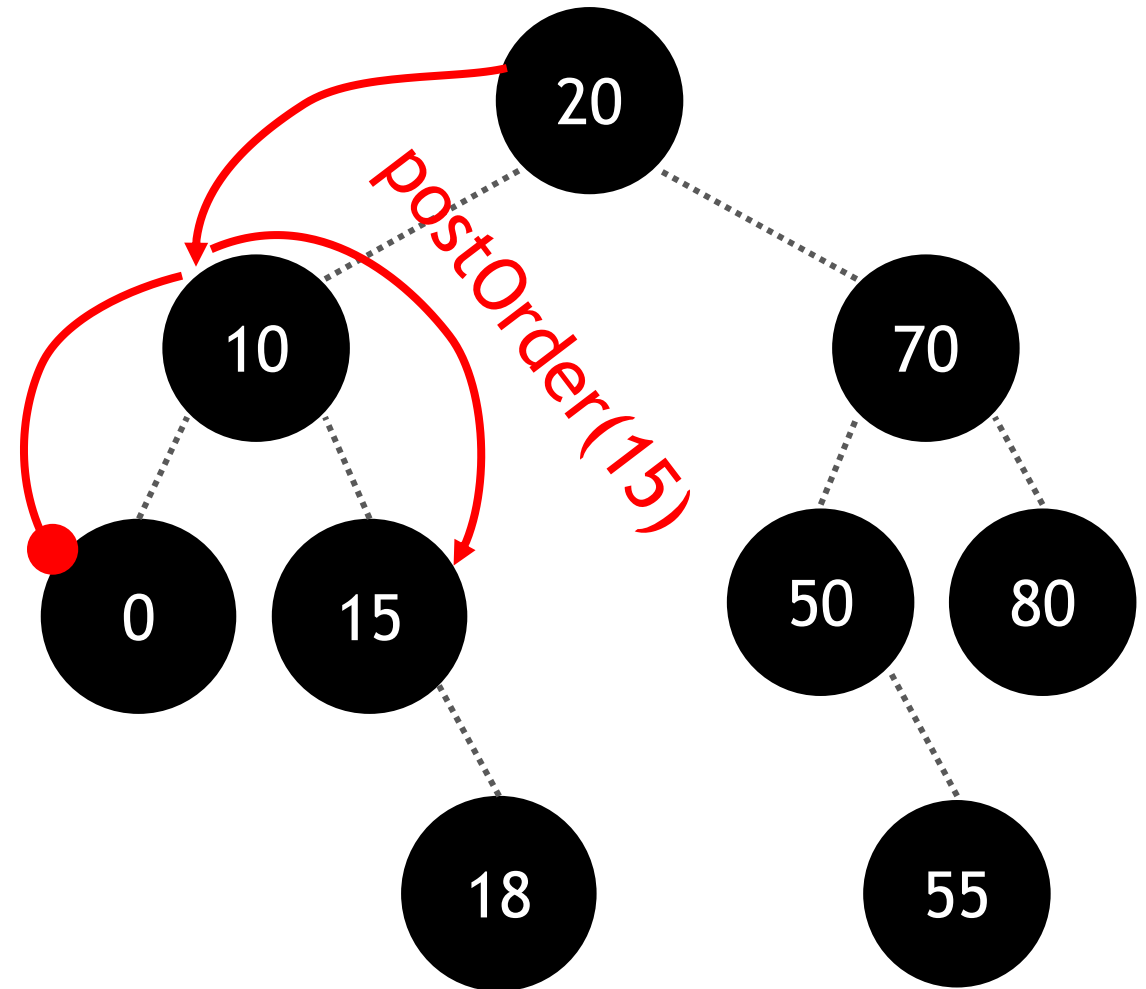
# Post-order

1. postOrder(node->smaller)
2. postOrder(node->greater)
3. Visit the node.



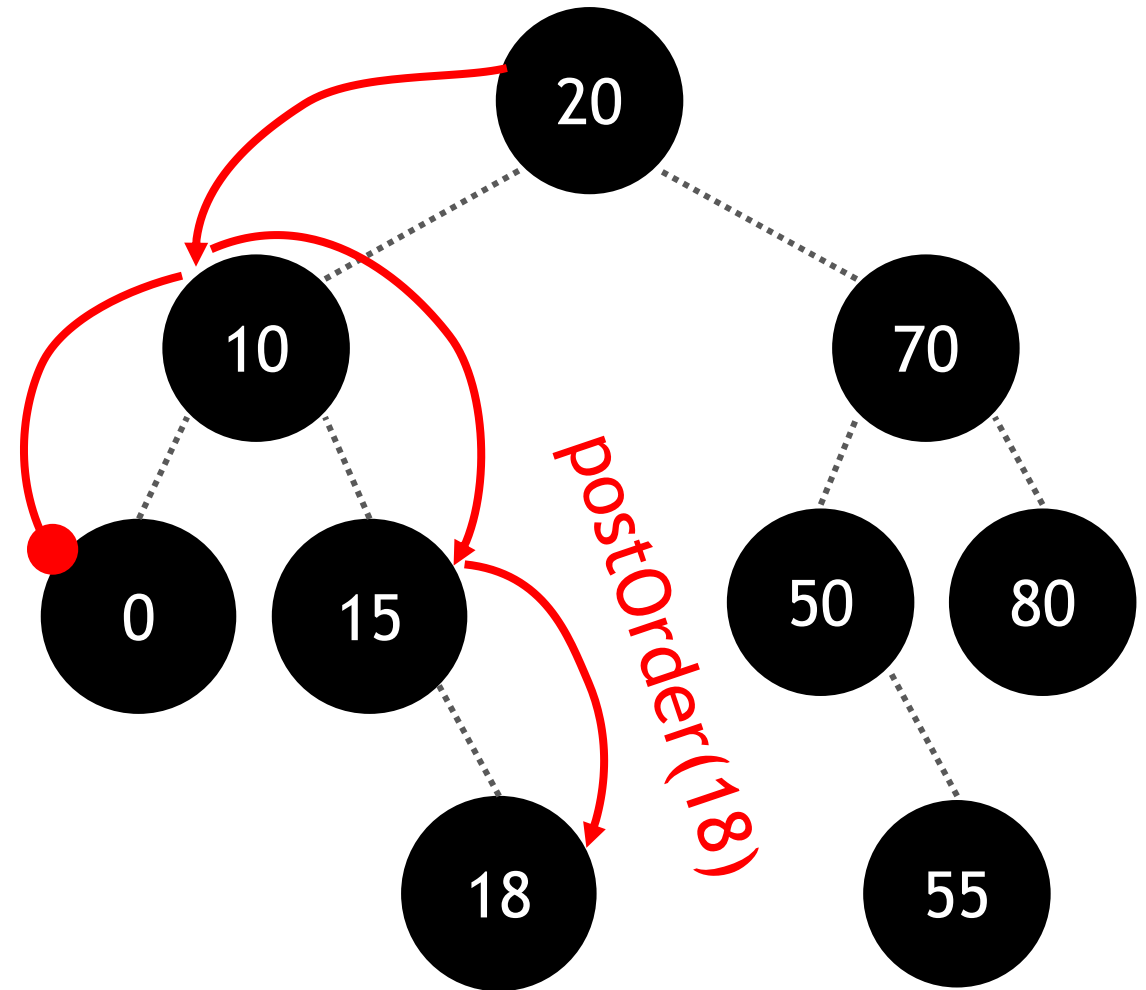
# Post-order

1. postOrder(node->smaller)
2. postOrder(node->greater)
3. Visit the node.



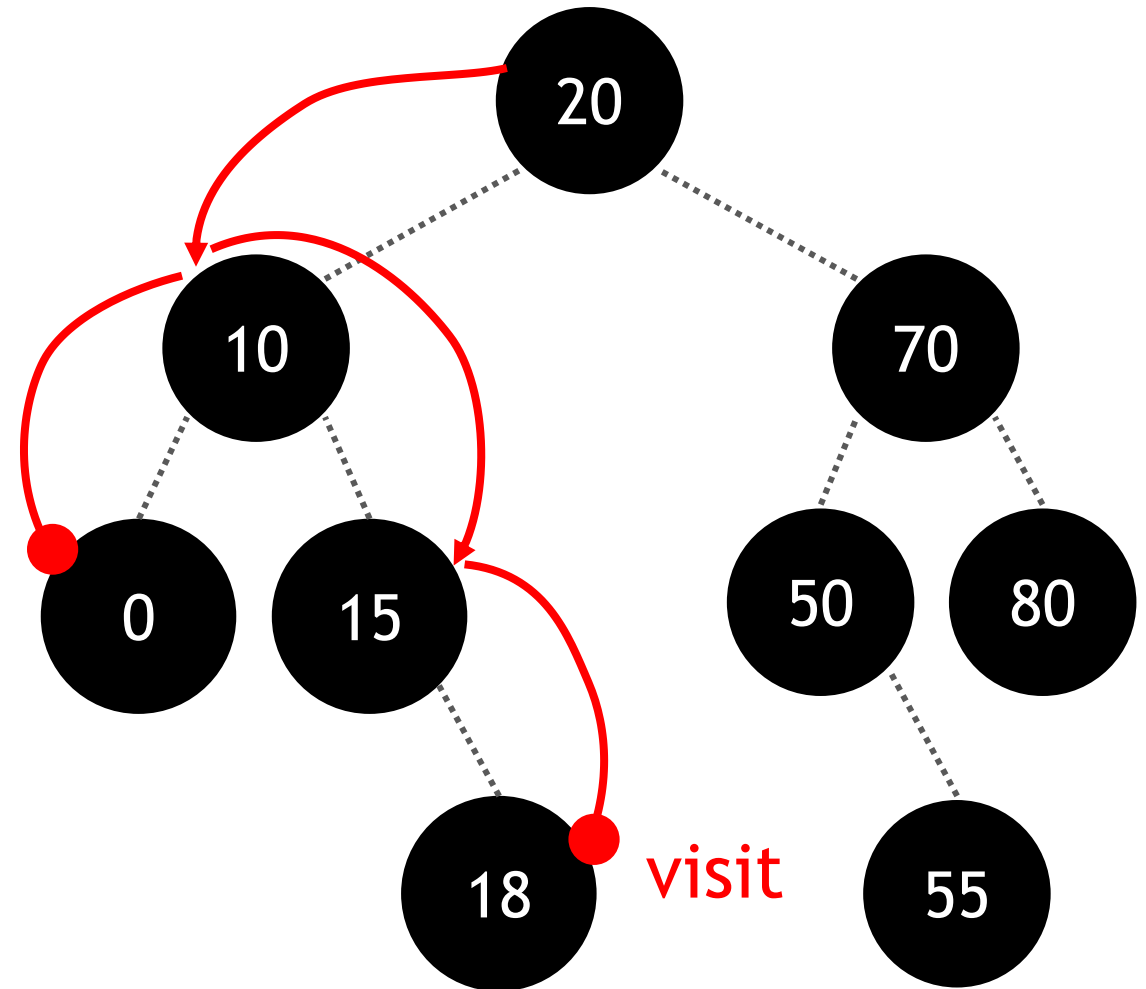
# Post-order

1. postOrder(node->smaller)
2. postOrder(node->greater)
3. Visit the node.



# Post-order

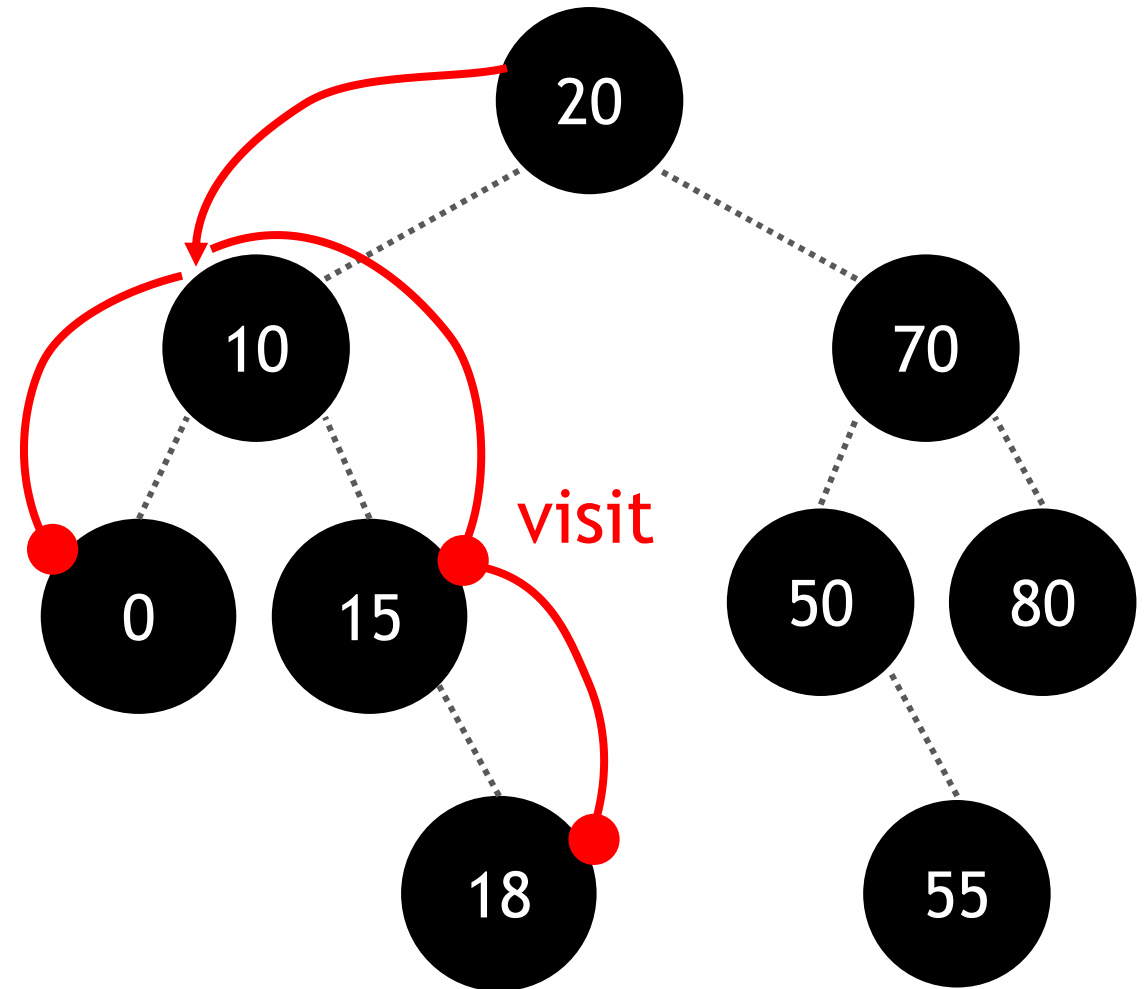
1. postOrder(node->smaller)
2. postOrder(node->greater)
3. Visit the node.



0,18,

# Post-order

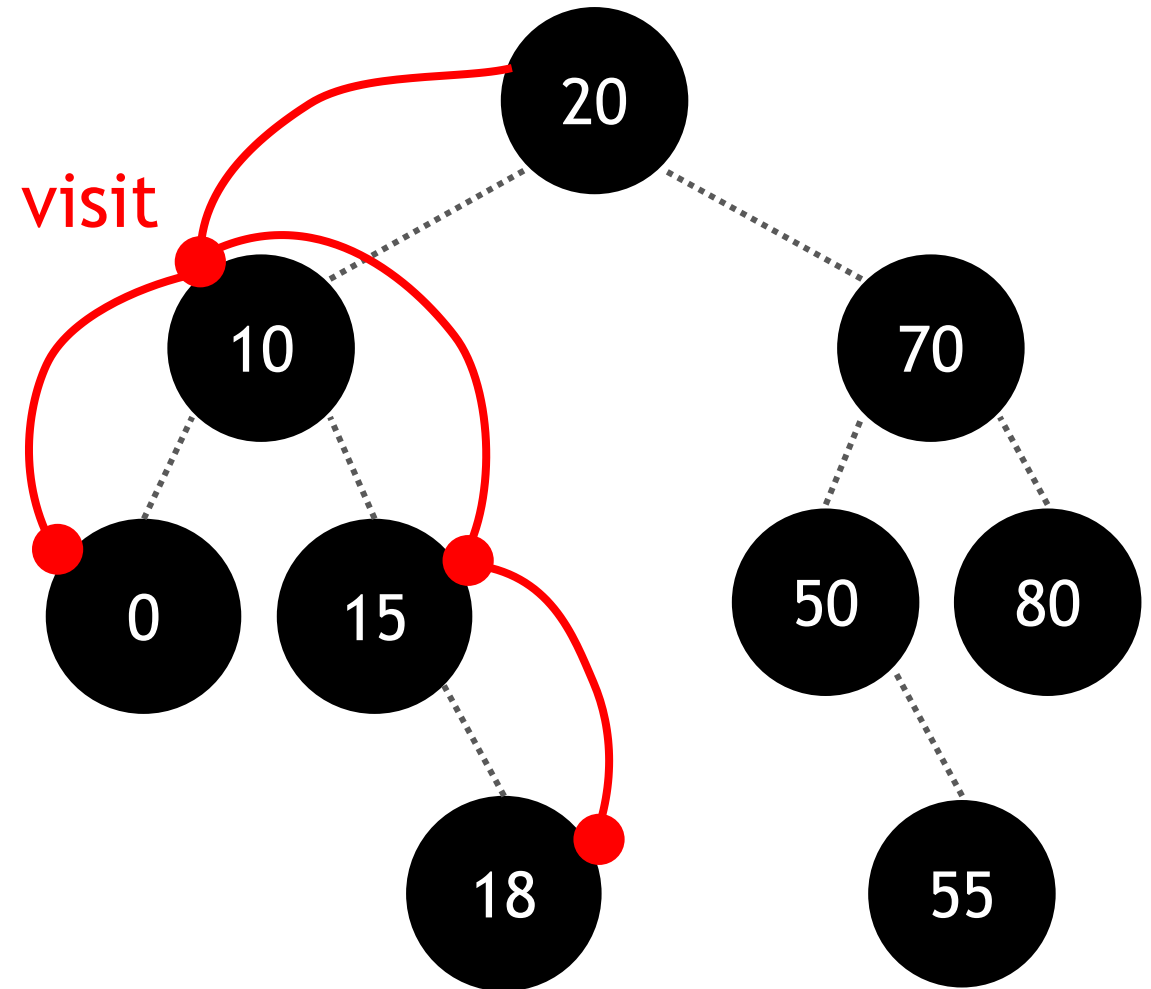
1. postOrder(node->smaller)
2. postOrder(node->greater)
3. Visit the node.



0,18,15

# Post-order

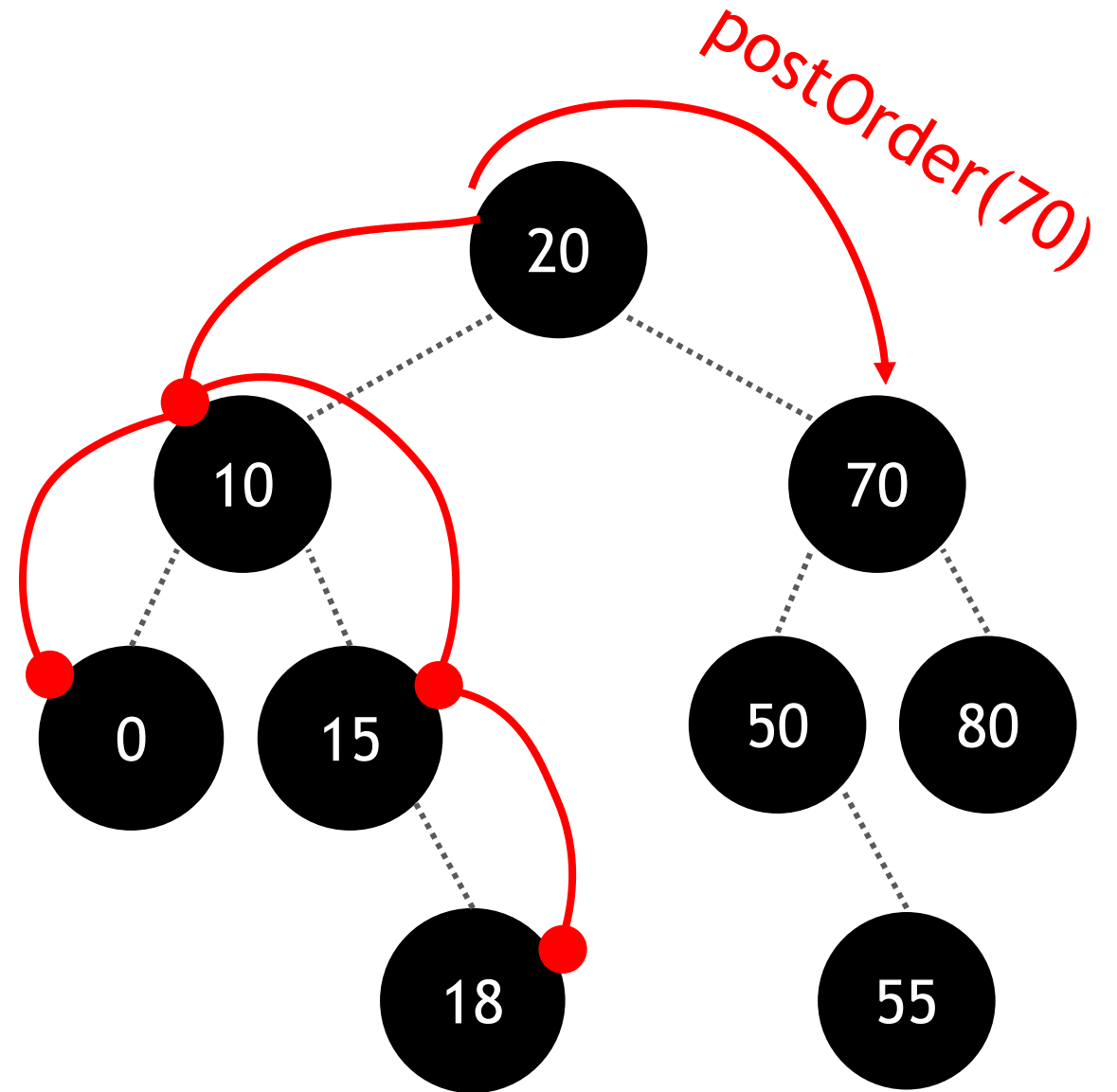
1. postOrder(node->smaller)
2. postOrder(node->greater)
3. Visit the node.



0,18,15,10

# Post-order

1. `postOrder(node->smaller)`
2. `postOrder(node->greater)`
3. Visit the node.

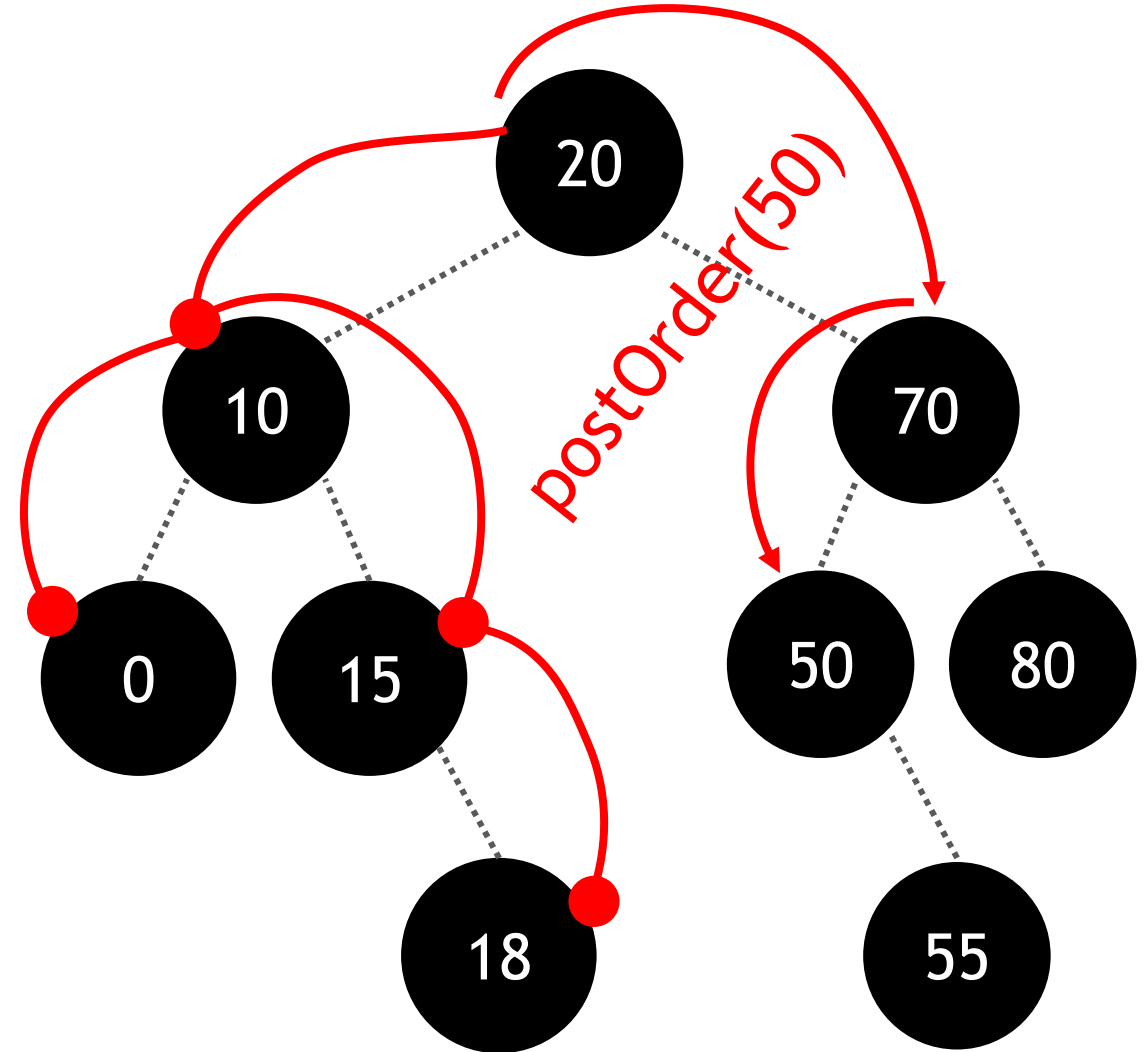


0,18,15,10



# Post-order

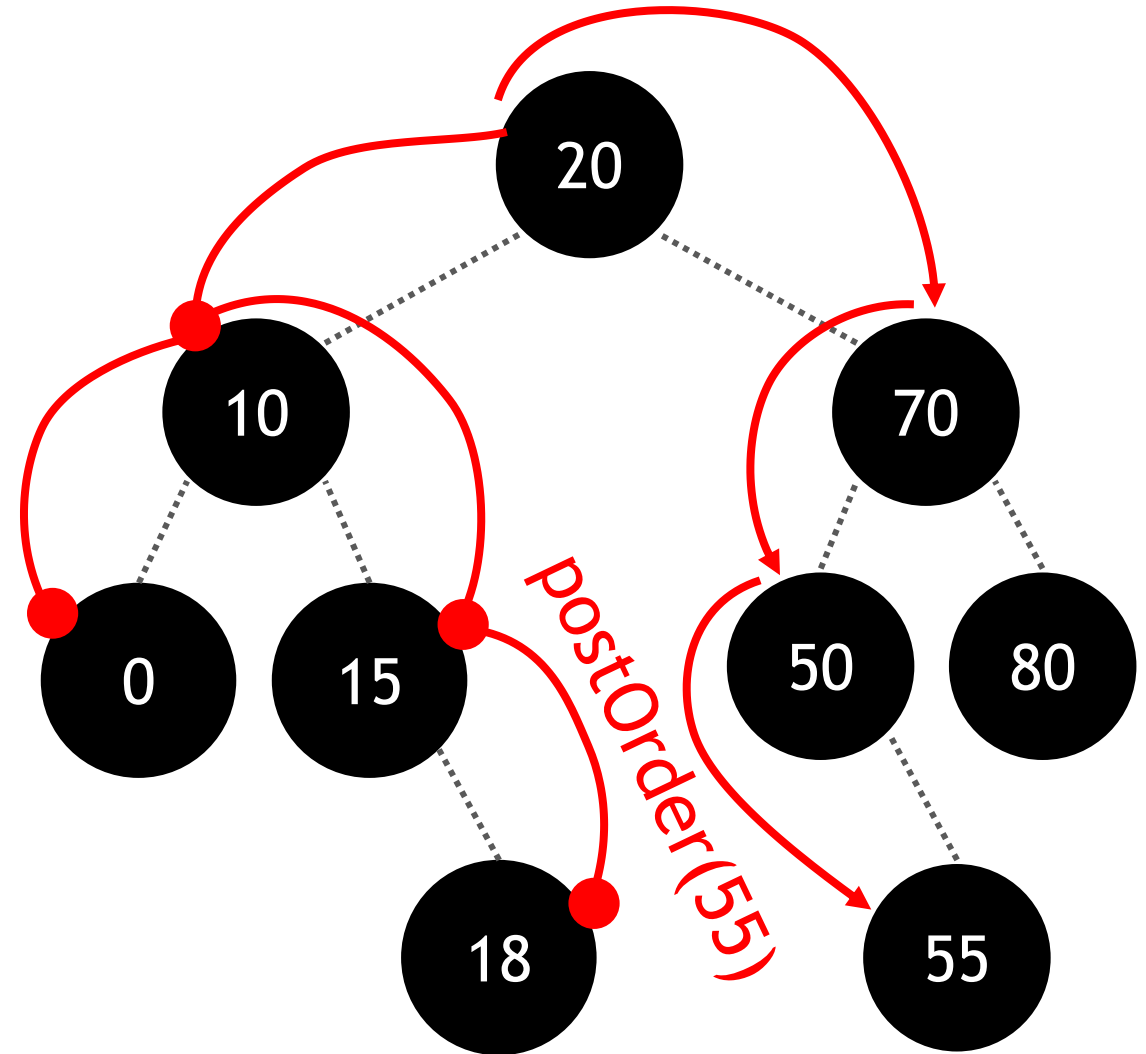
1. postOrder(node->smaller)
2. postOrder(node->greater)
3. Visit the node.



0,18,15,10

# Post-order

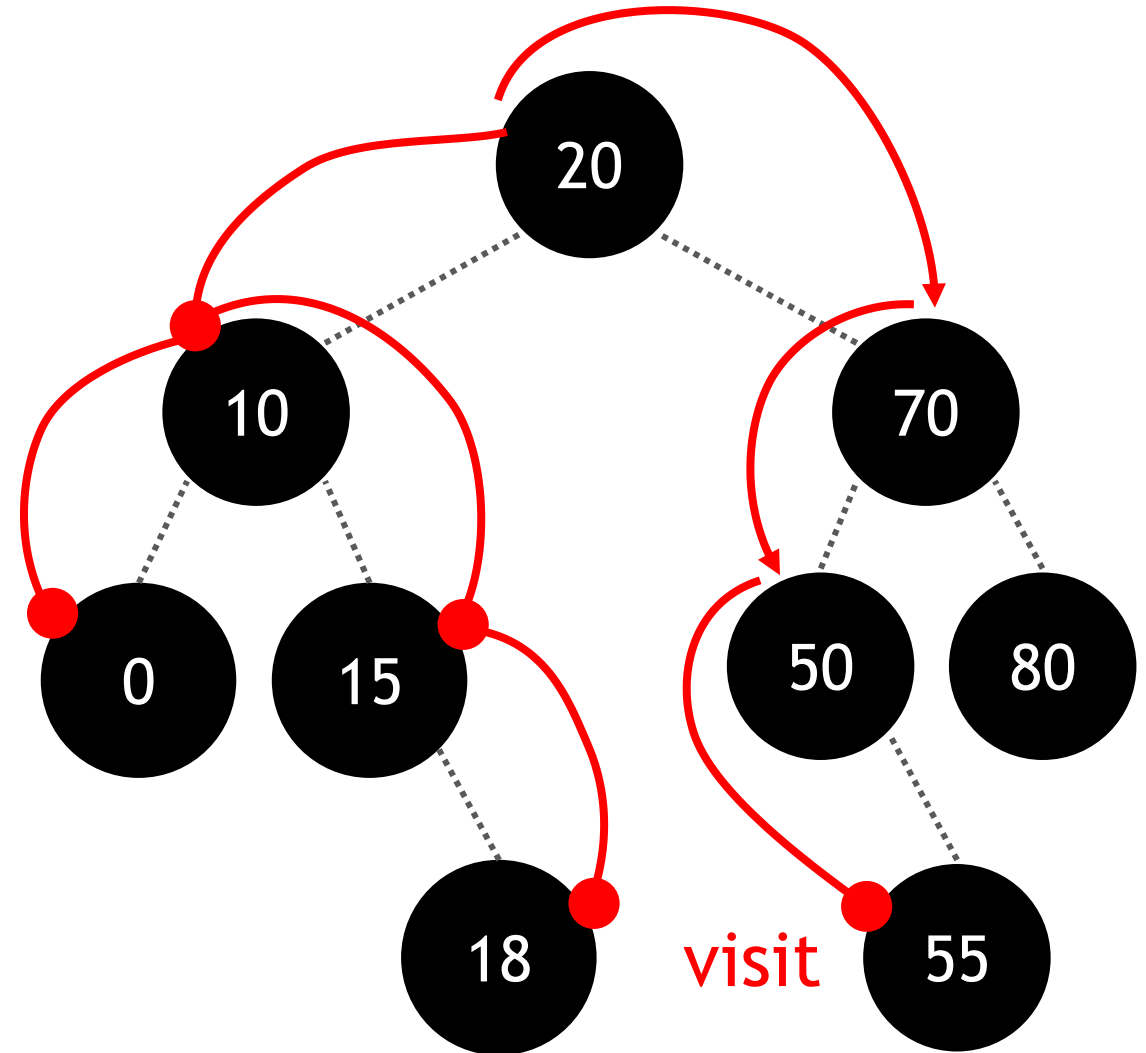
1. `postOrder(node->smaller)`
2. `postOrder(node->greater)`
3. Visit the node.



0,18,15,10

# Post-order

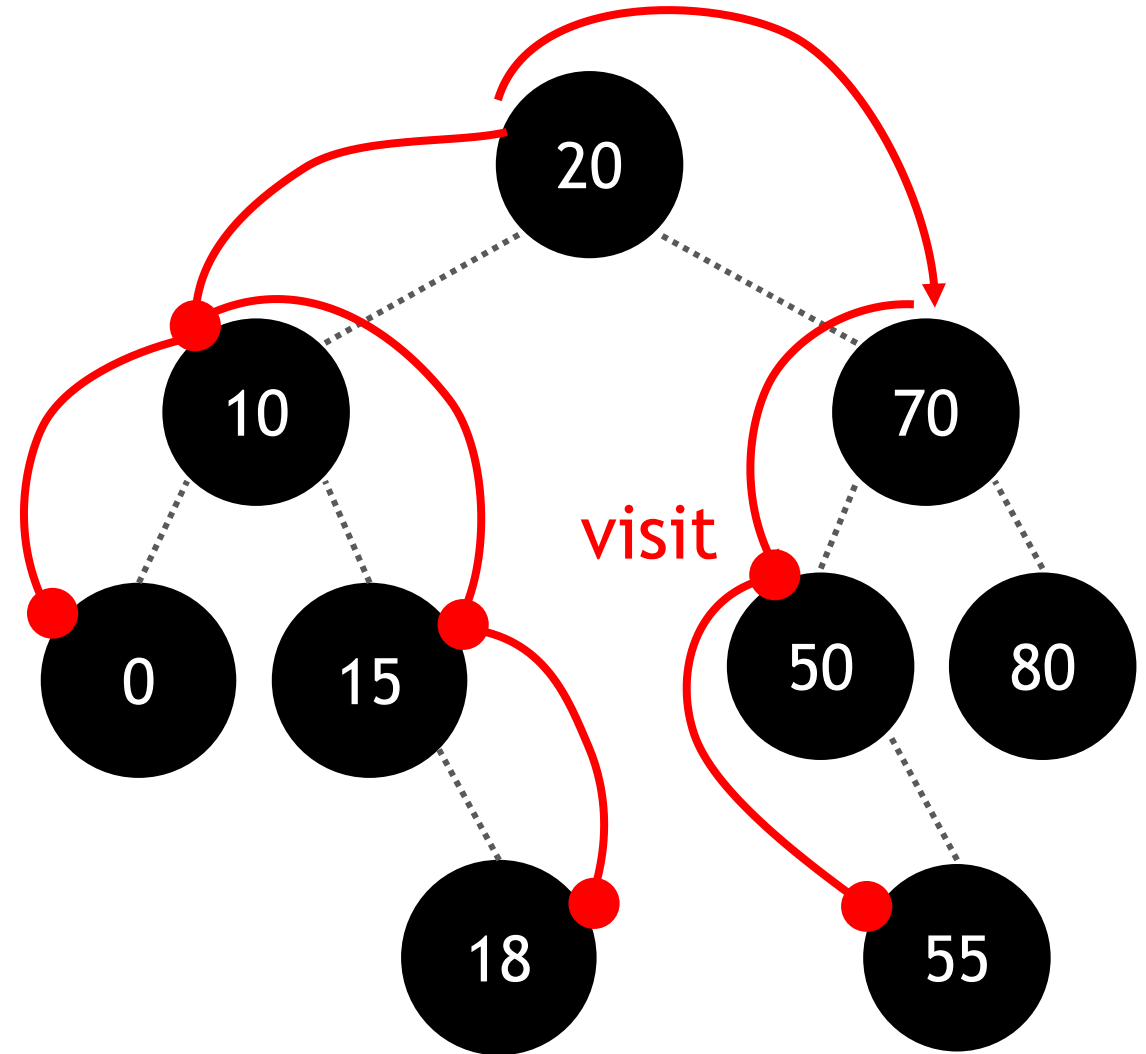
1. postOrder(node->smaller)
2. postOrder(node->greater)
3. Visit the node.



0,18,15,10,55

# Post-order

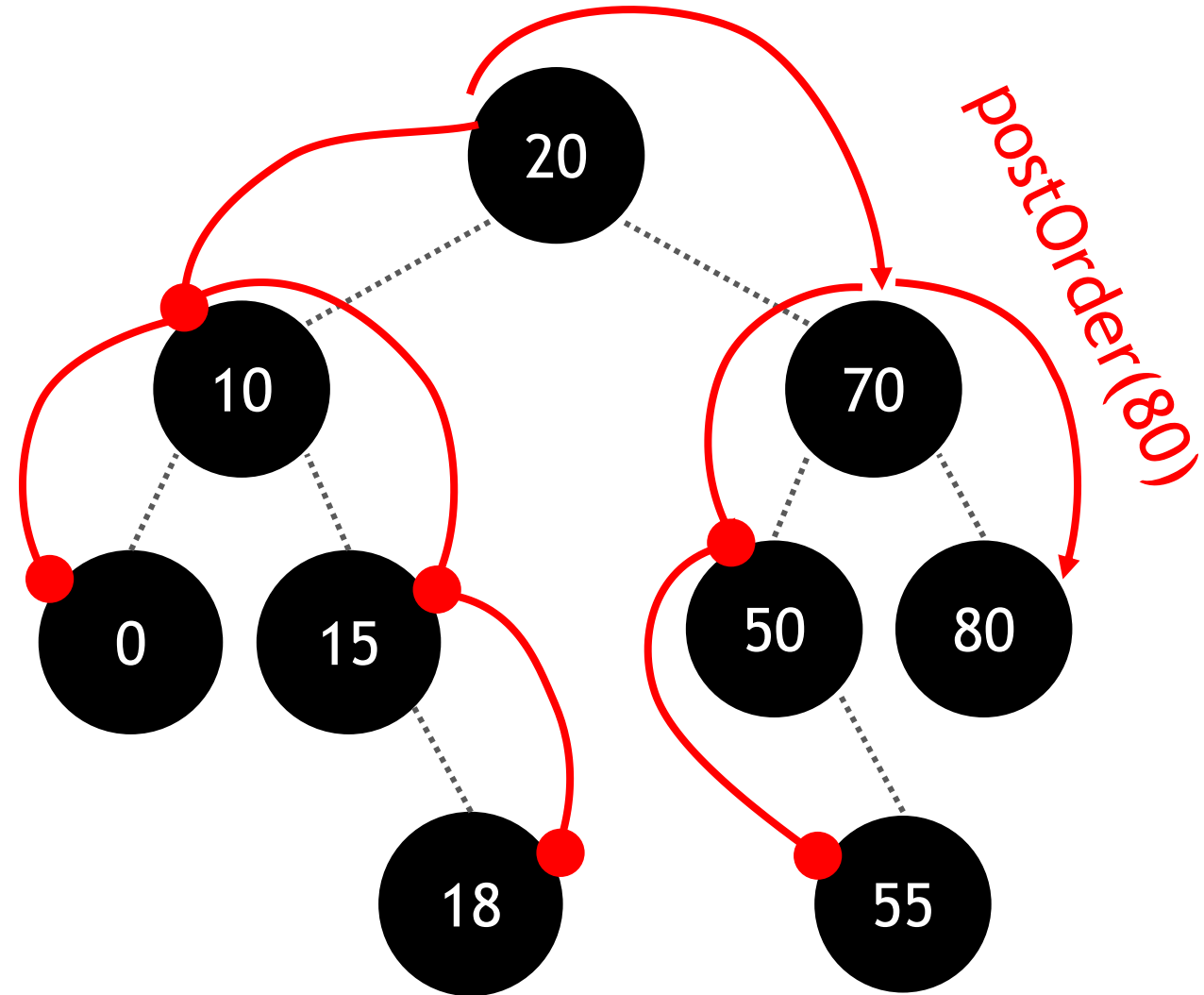
1. postOrder(node->smaller)
2. postOrder(node->greater)
3. Visit the node.



0,18,15,10,55,50

# Post-order

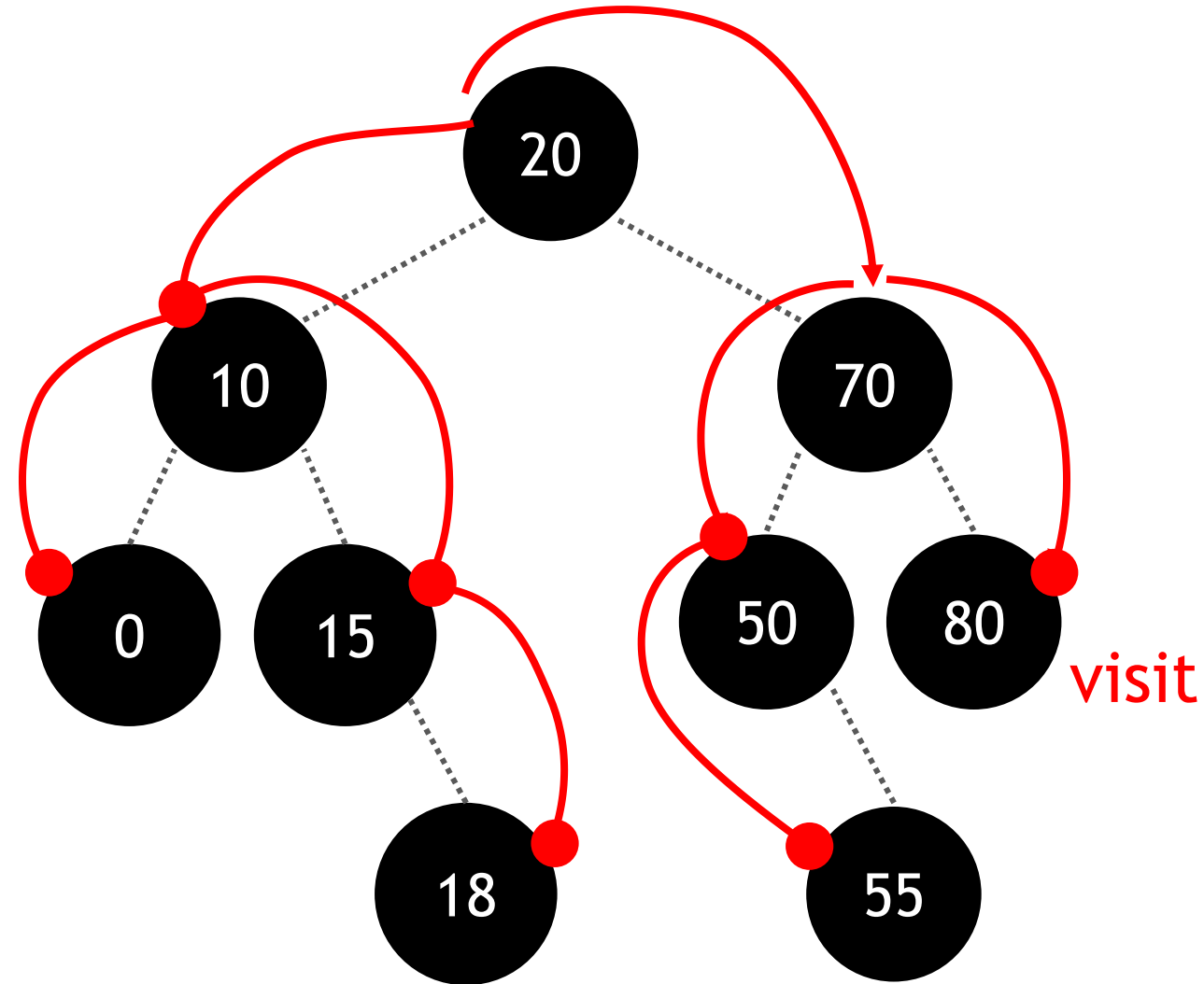
1. `postOrder(node->smaller)`
2. `postOrder(node->greater)`
3. Visit the node.



0,18,15,10,55,50

# Post-order

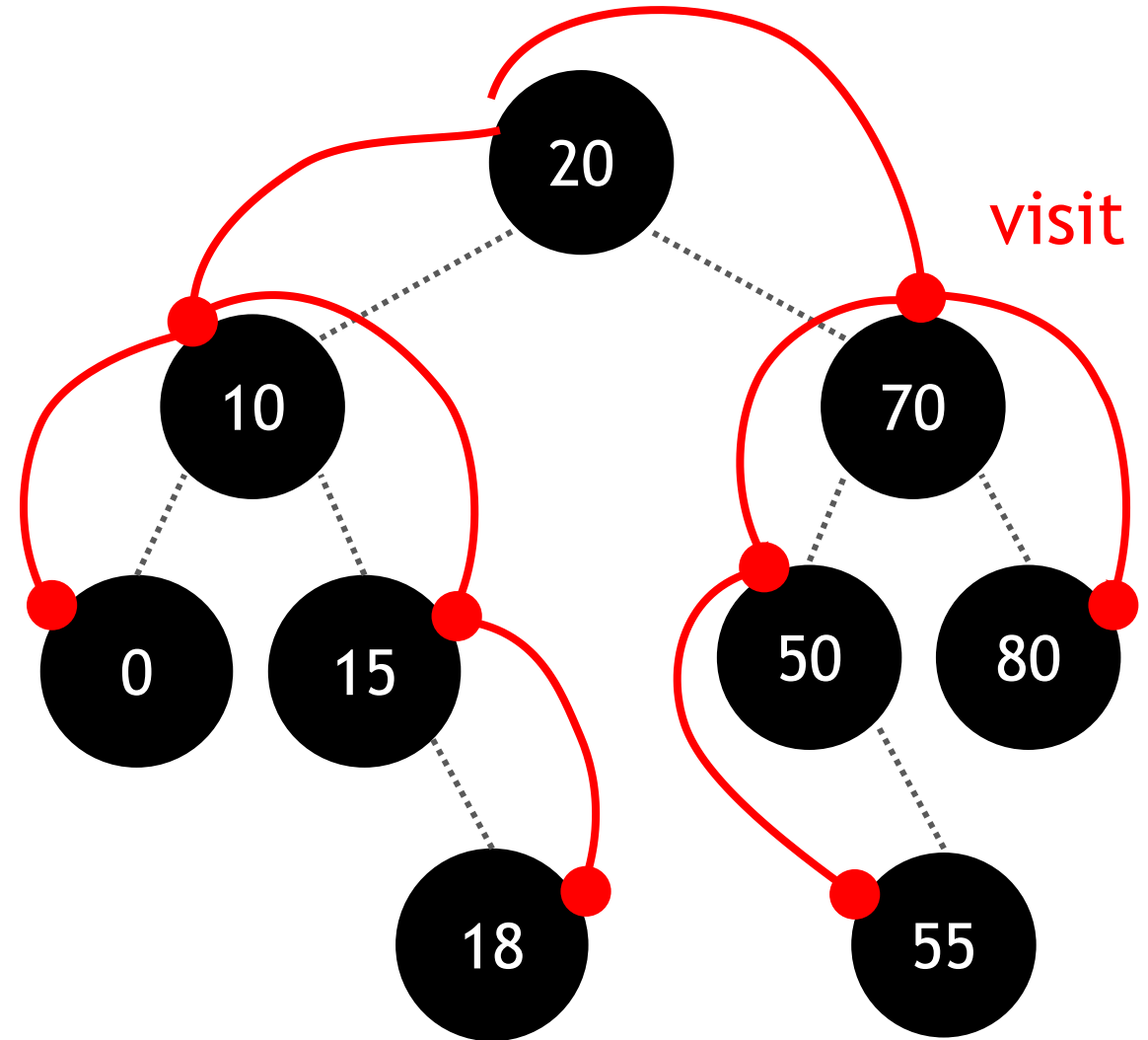
1. postOrder(node->smaller)
2. postOrder(node->greater)
3. Visit the node.



0,18,15,10,55,50,80

# Post-order

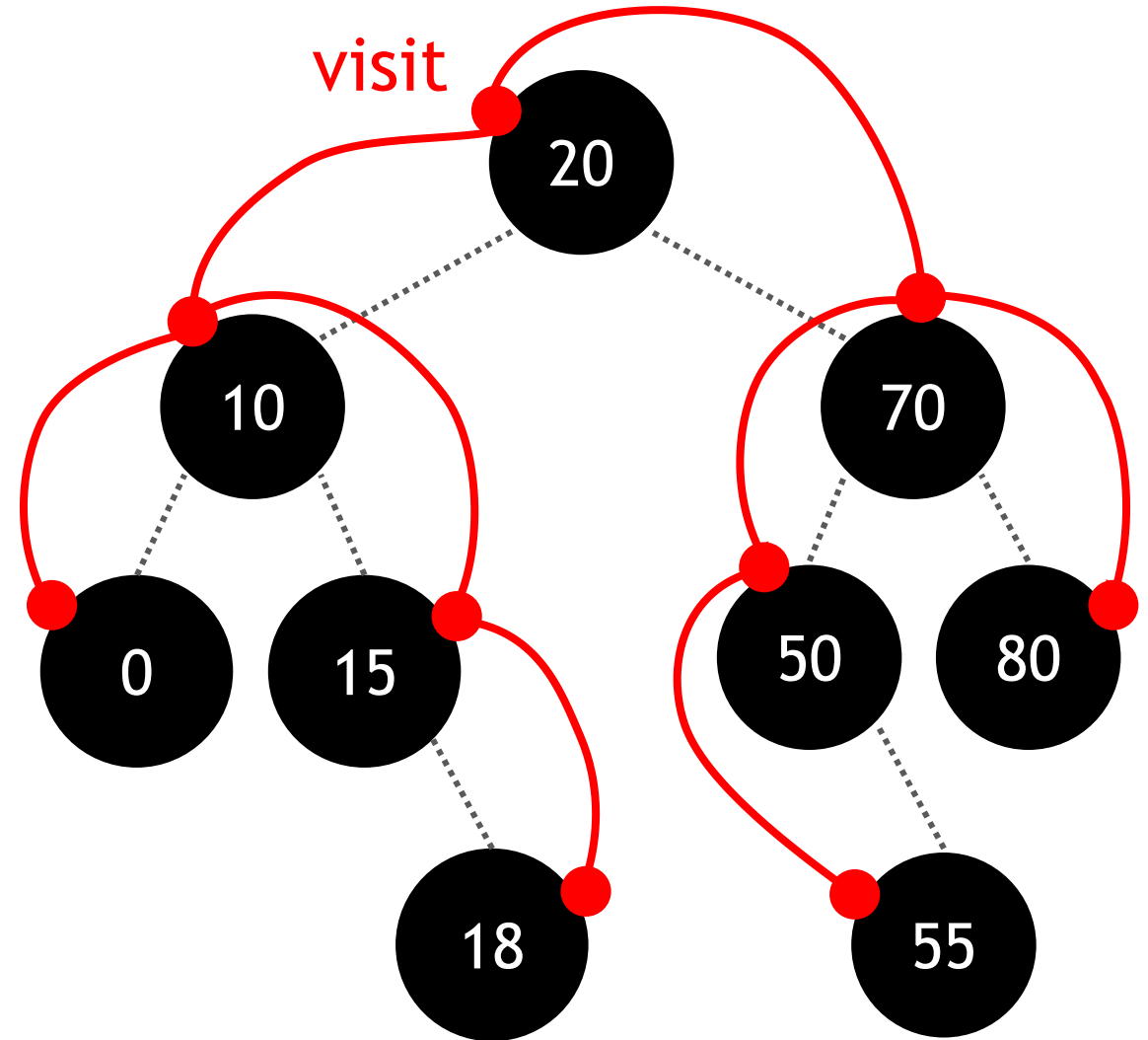
1. postOrder(node->smaller)
2. postOrder(node->greater)
3. Visit the node.



0, 18, 15, 10, 55, 50, 80, 70,

# Post-order

1. postOrder(node->smaller)
2. postOrder(node->greater)
3. Visit the node.



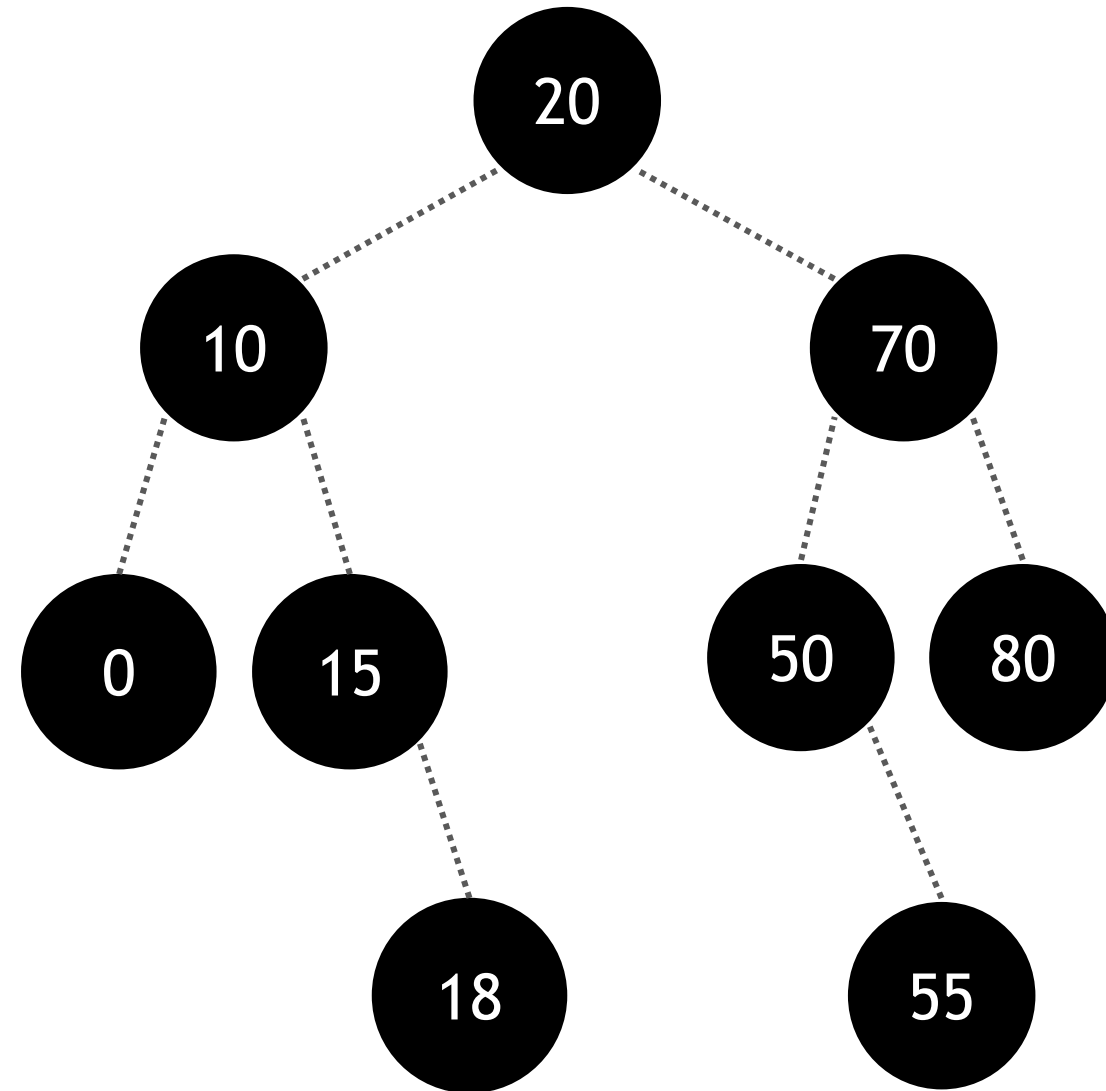
0,18,15,10,55,50,80,70,20



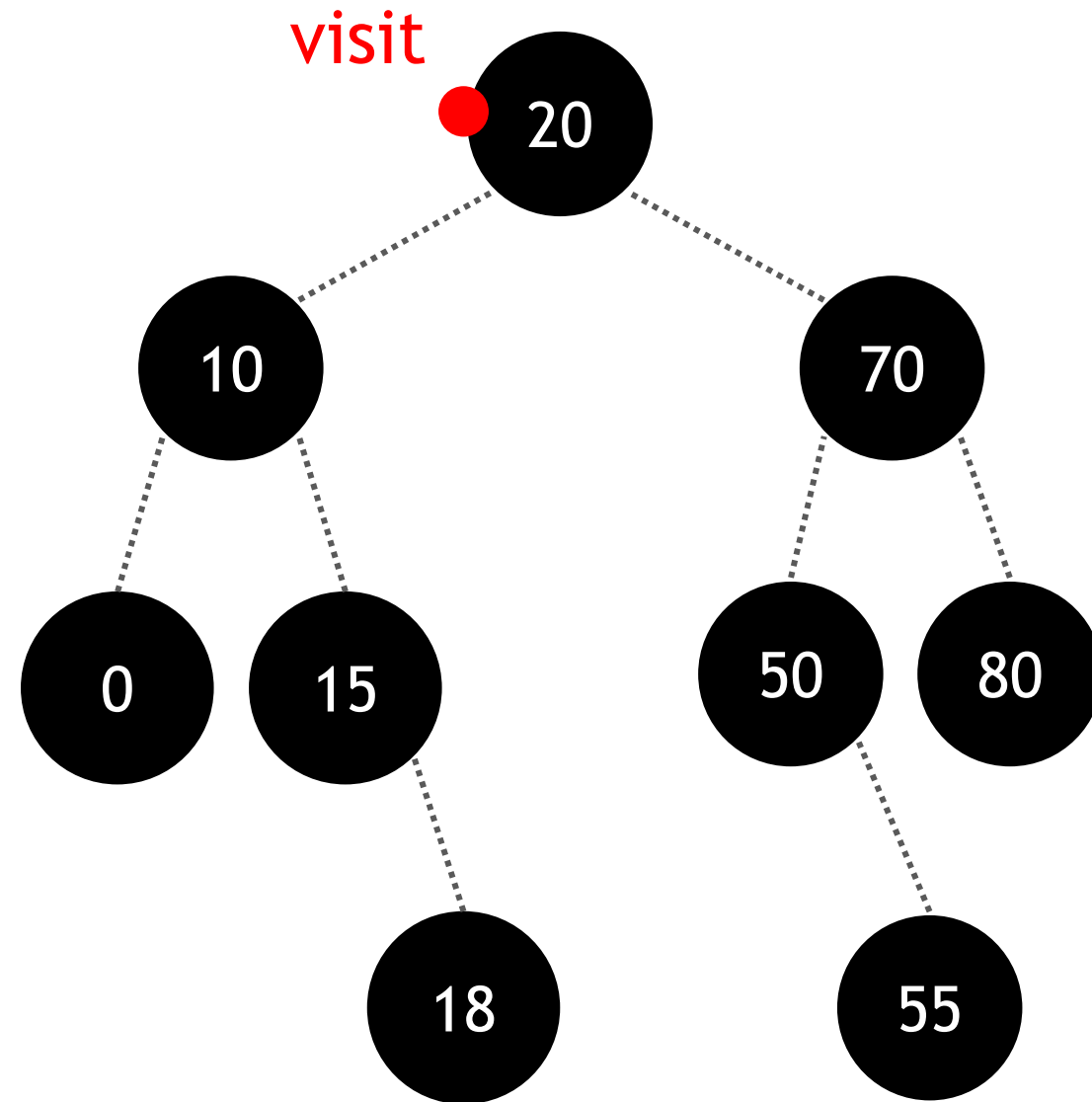
# Level-order

## Prechod BST

# Level-order

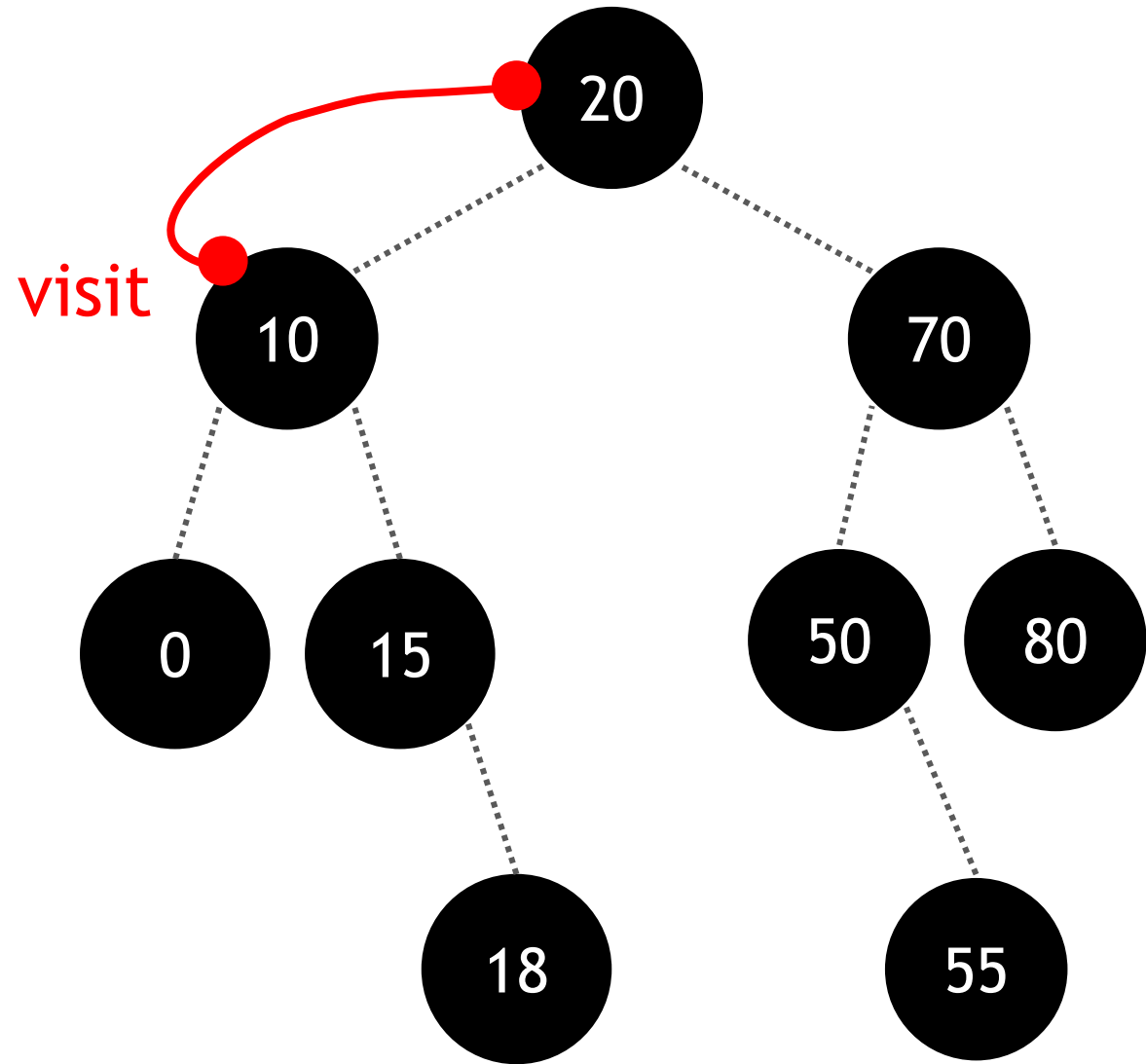


# Level-order



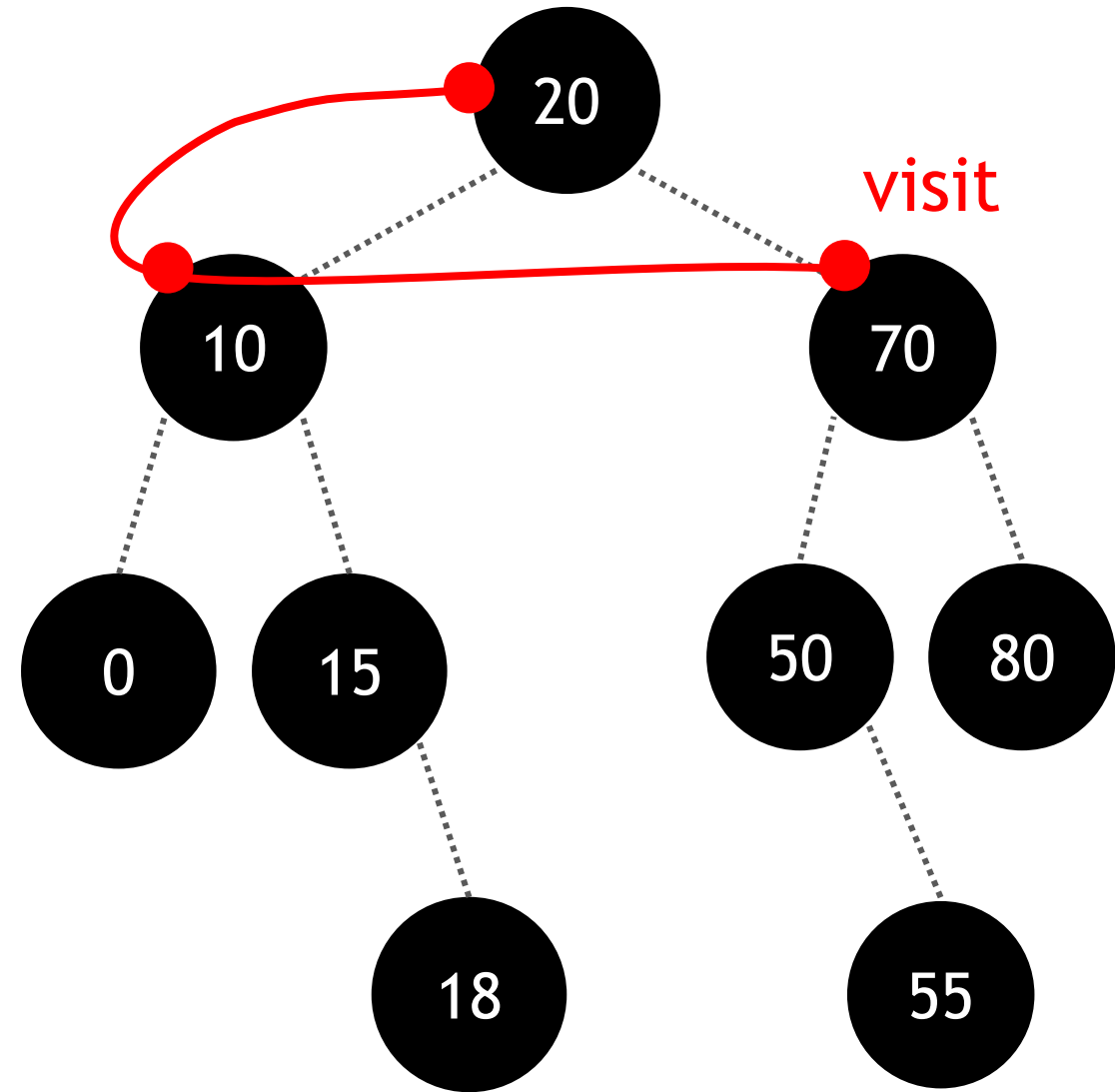
20,

# Level-order



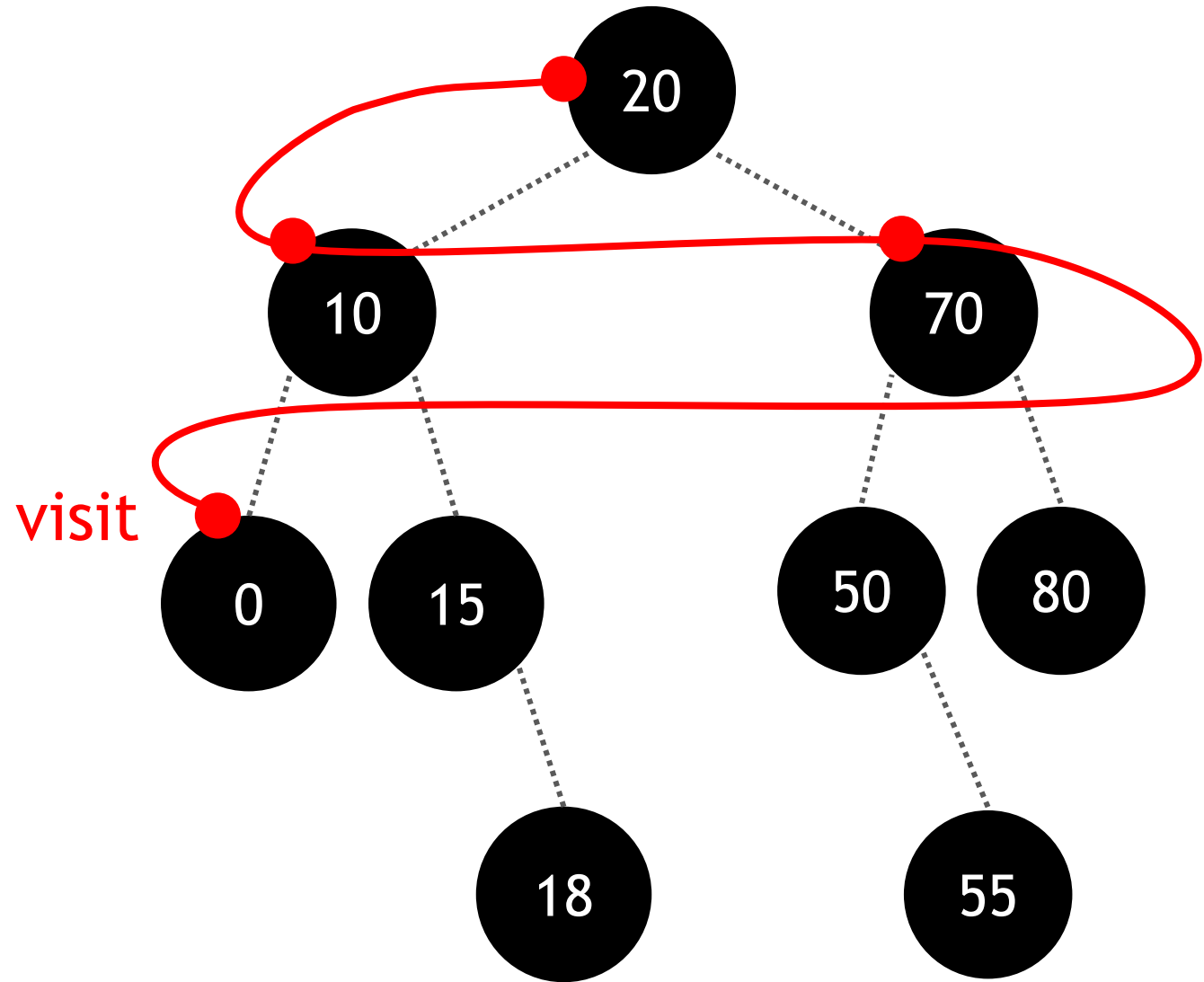
20, 10,

# Level-order



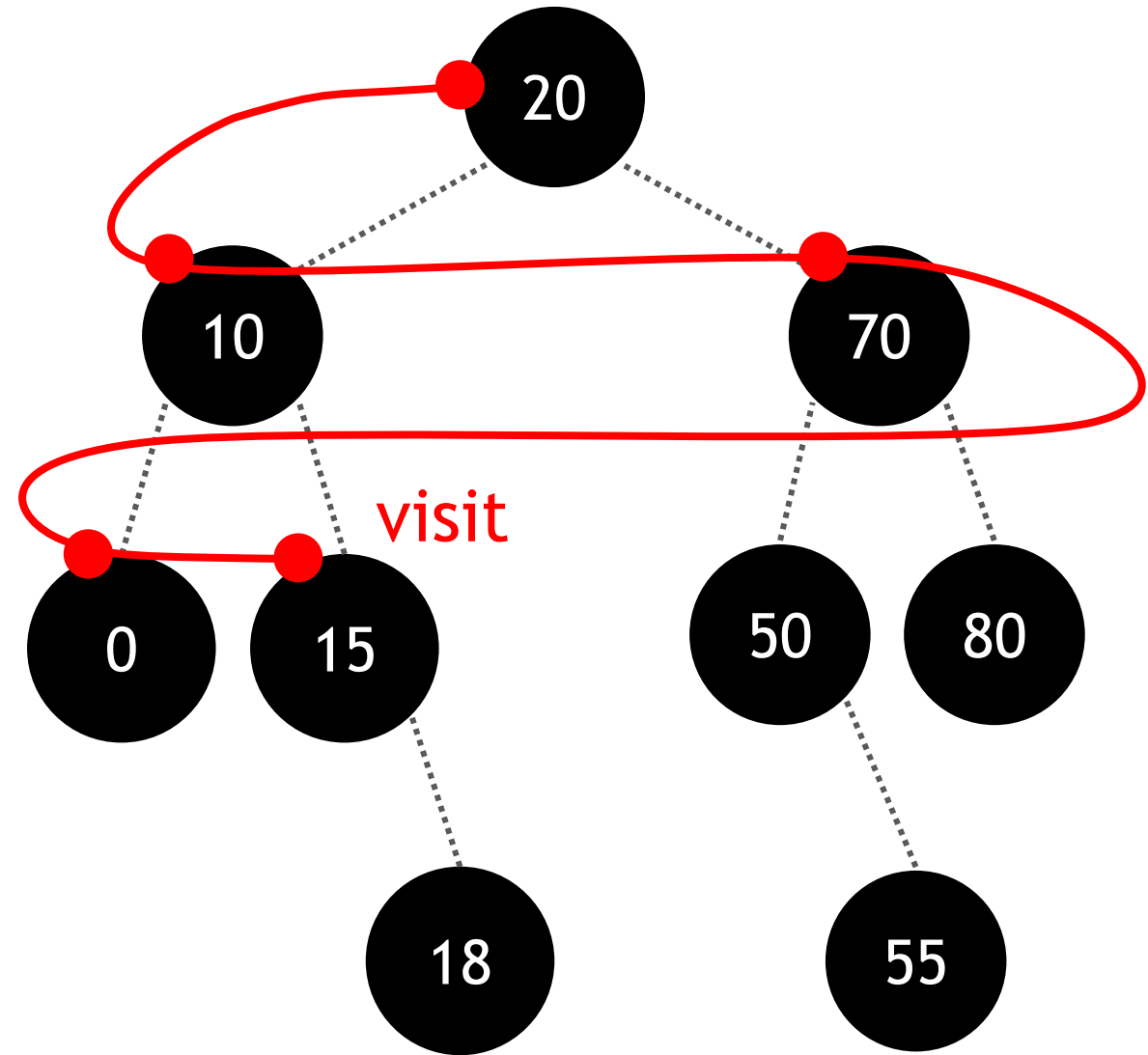
20,10,70,

# Level-order



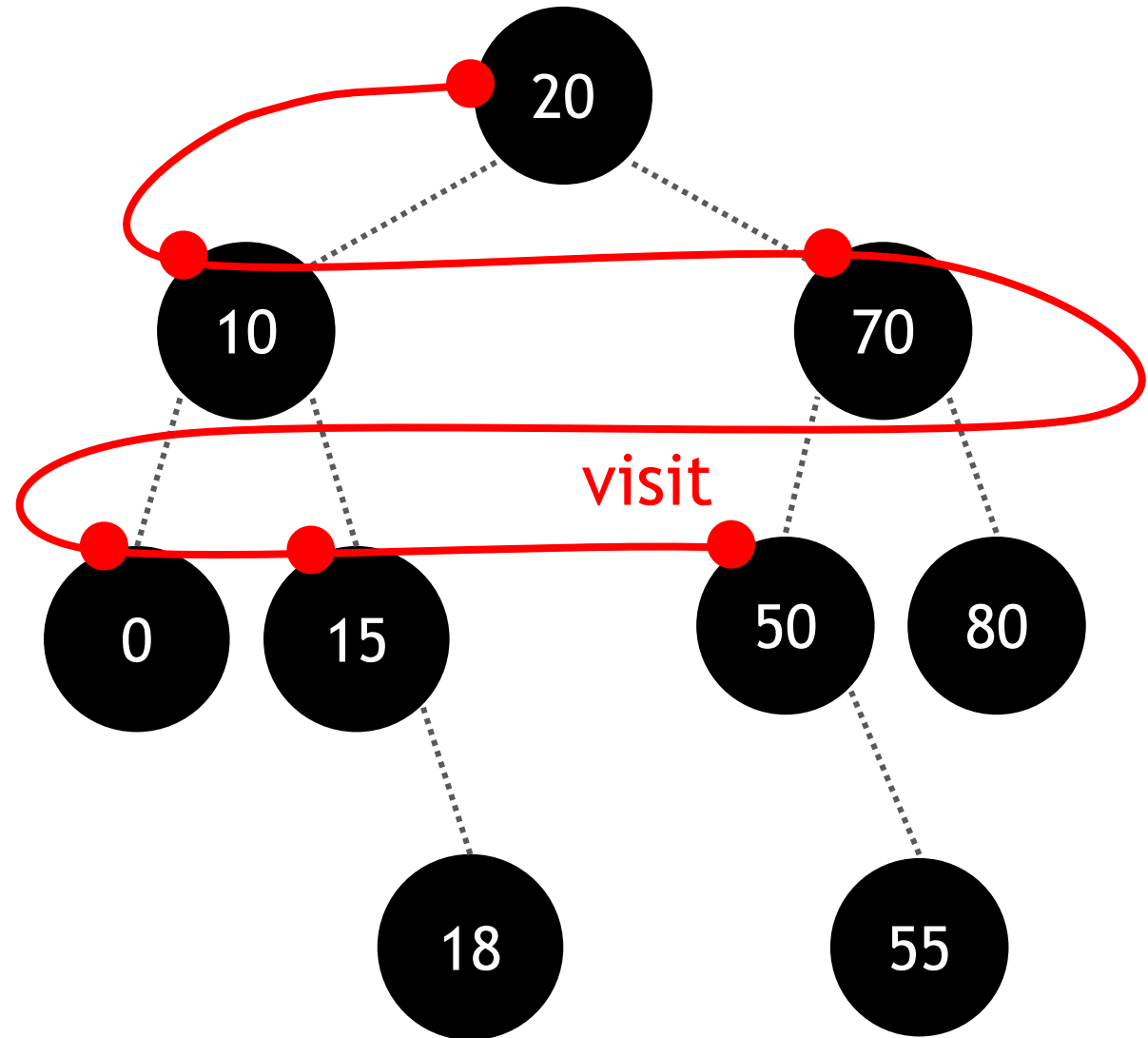
20,10,70,0,

# Level-order



20,10,70,0,15,

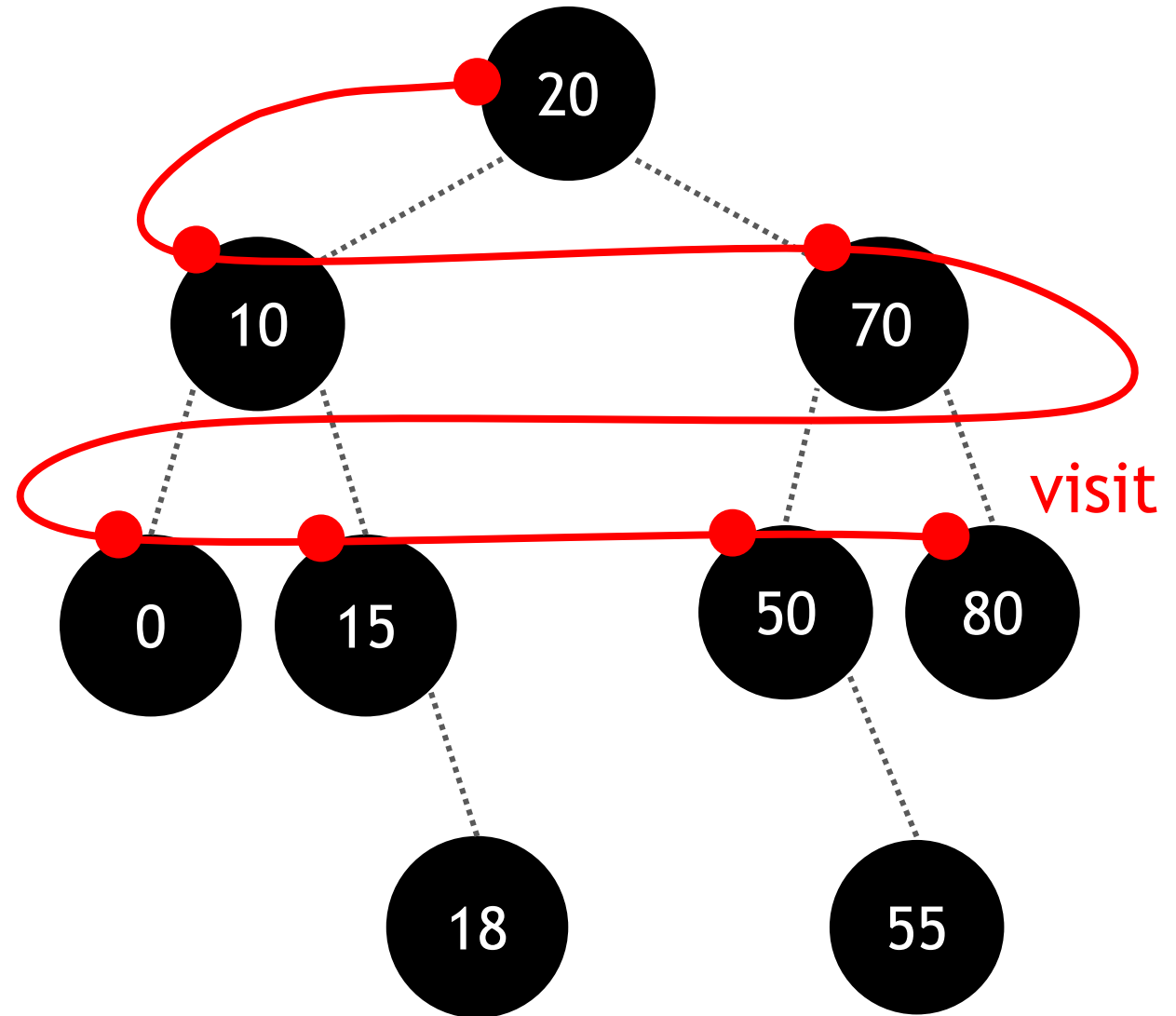
# Level-order



20,10,70,0,15,50,

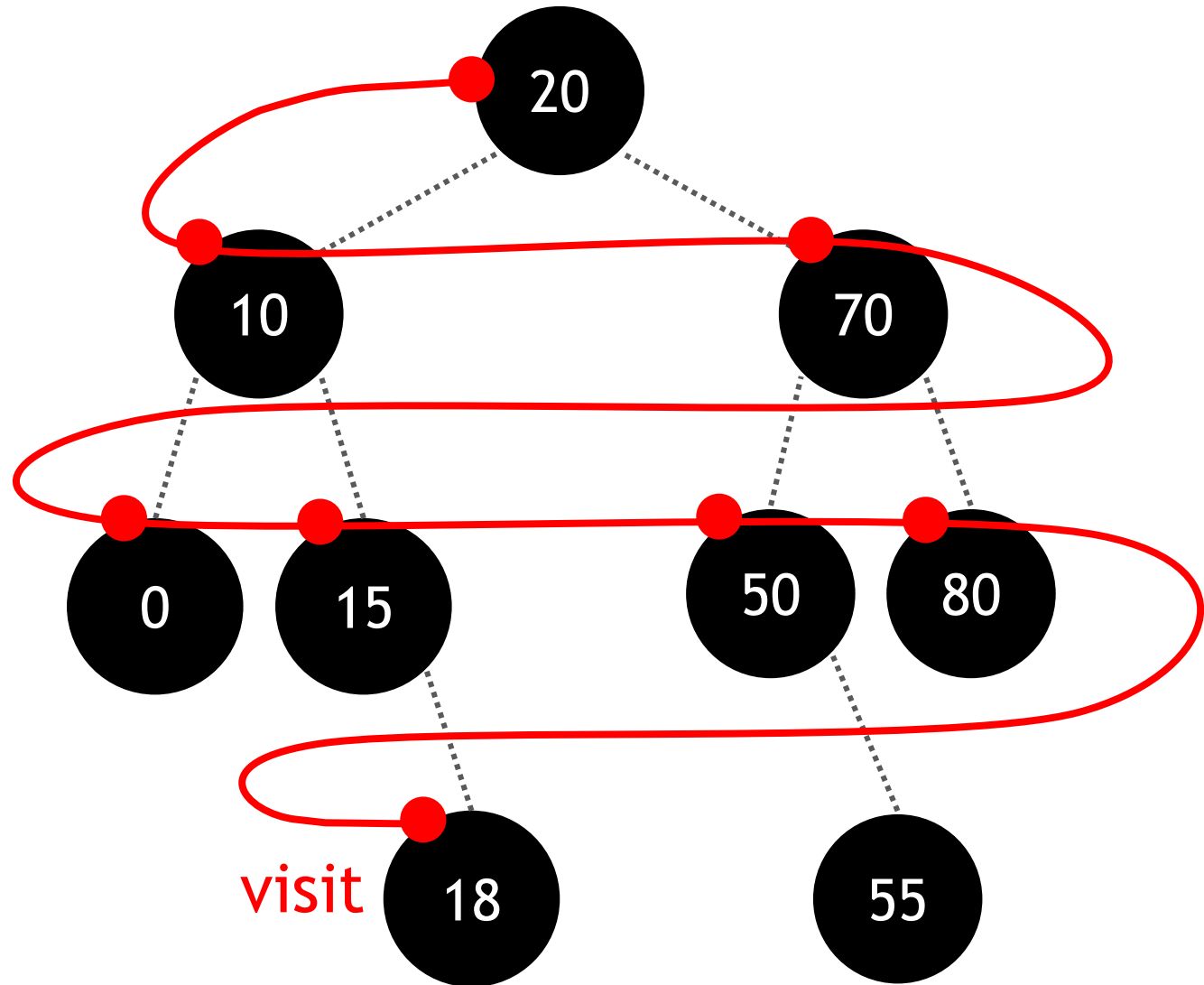


# Level-order



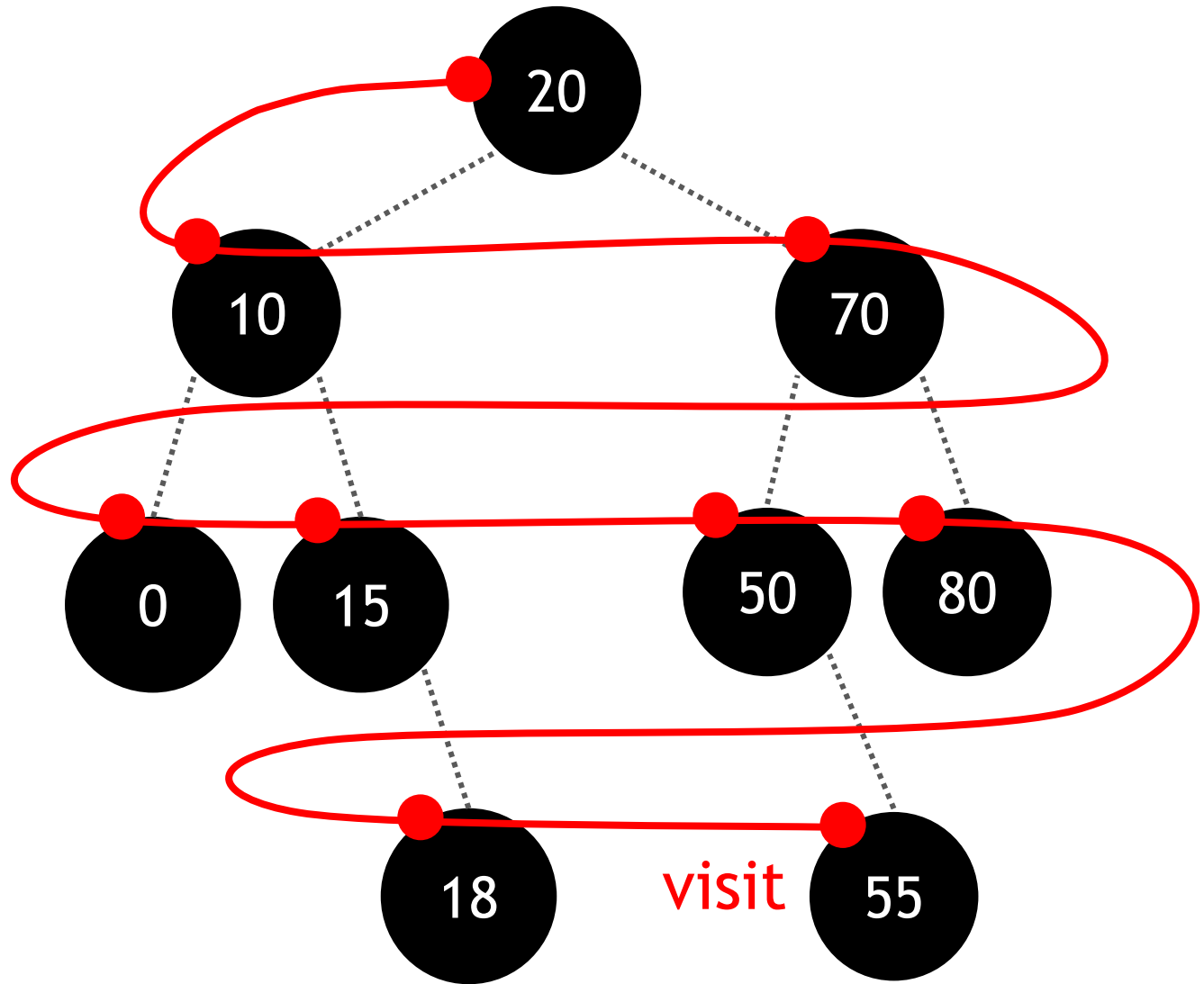
20,10,70,0,15,50,80,

# Level-order



20,10,70,0,15,50,80,18,

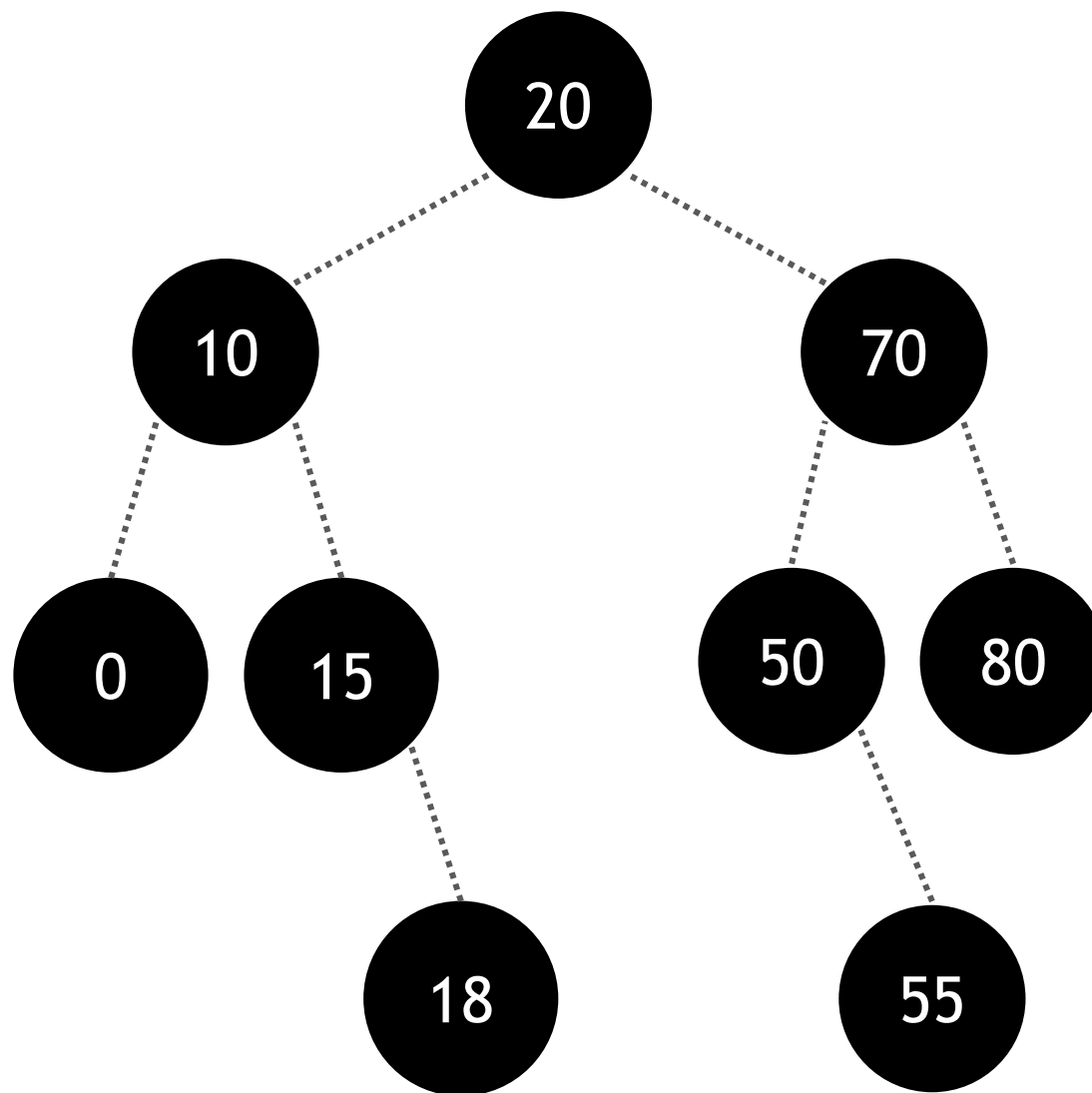
# Level-order



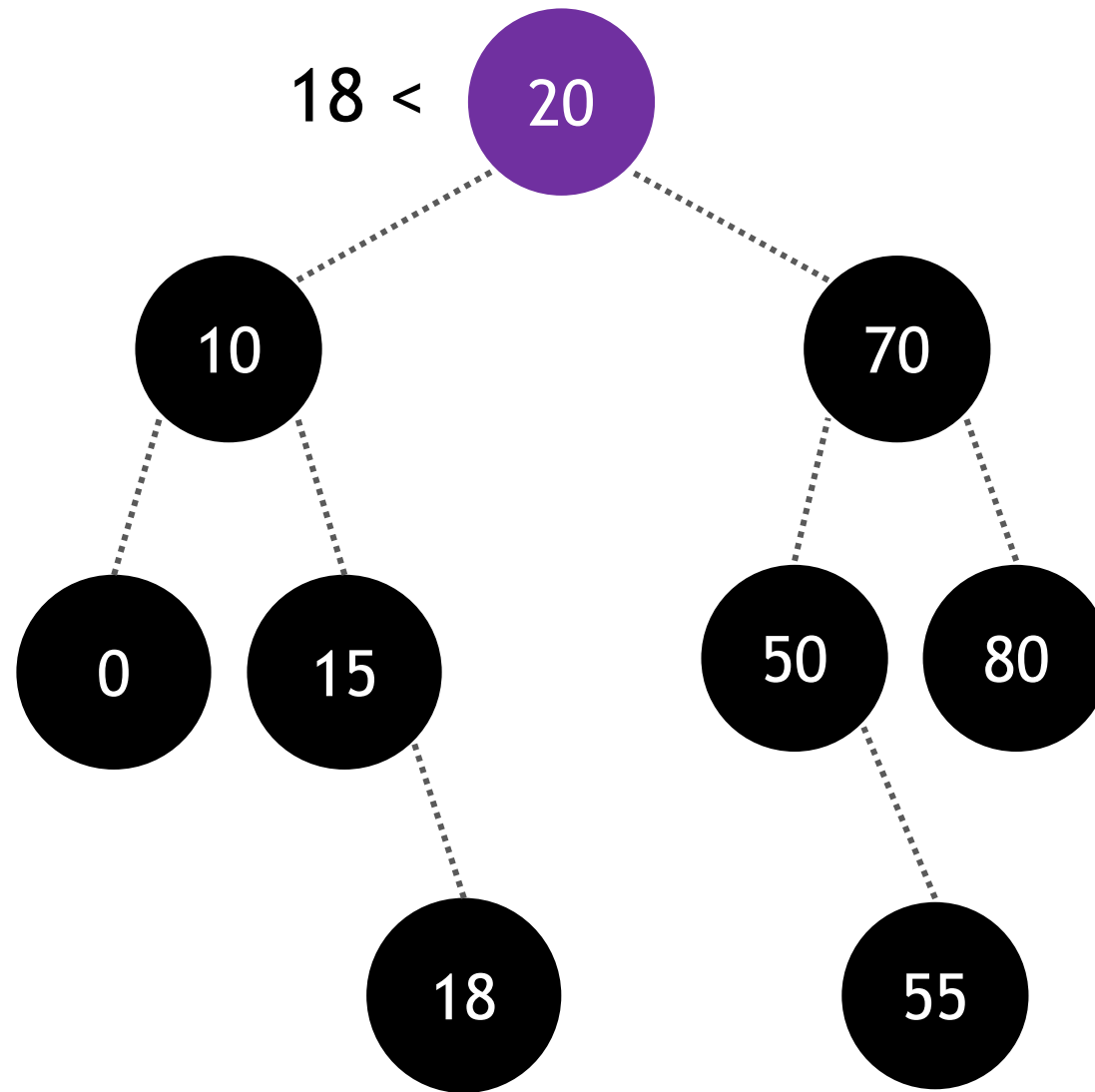
20,10,70,0,15,50,80,18,55

# Vyhľadávanie uzlov

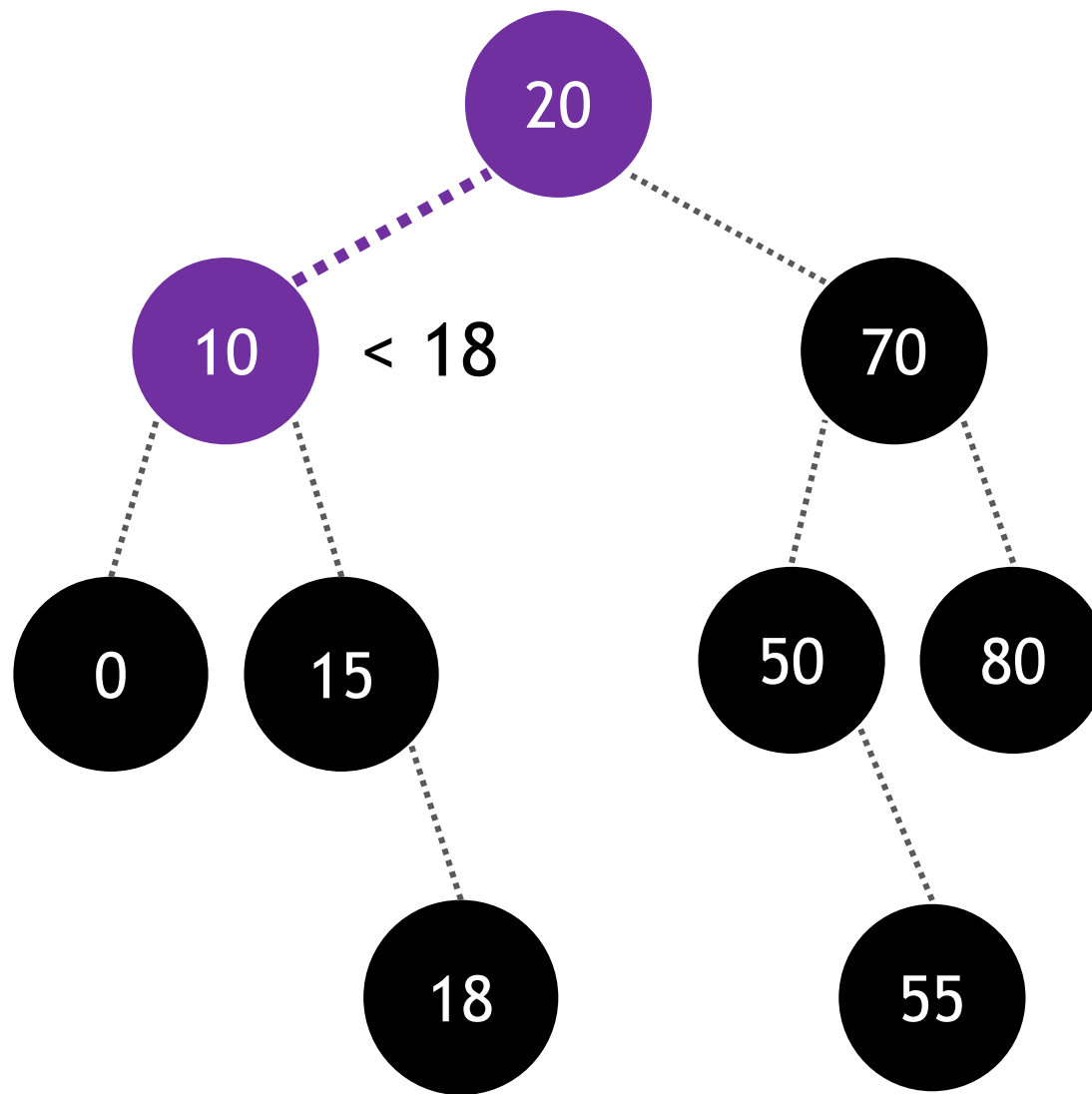
Hľadáme: 18



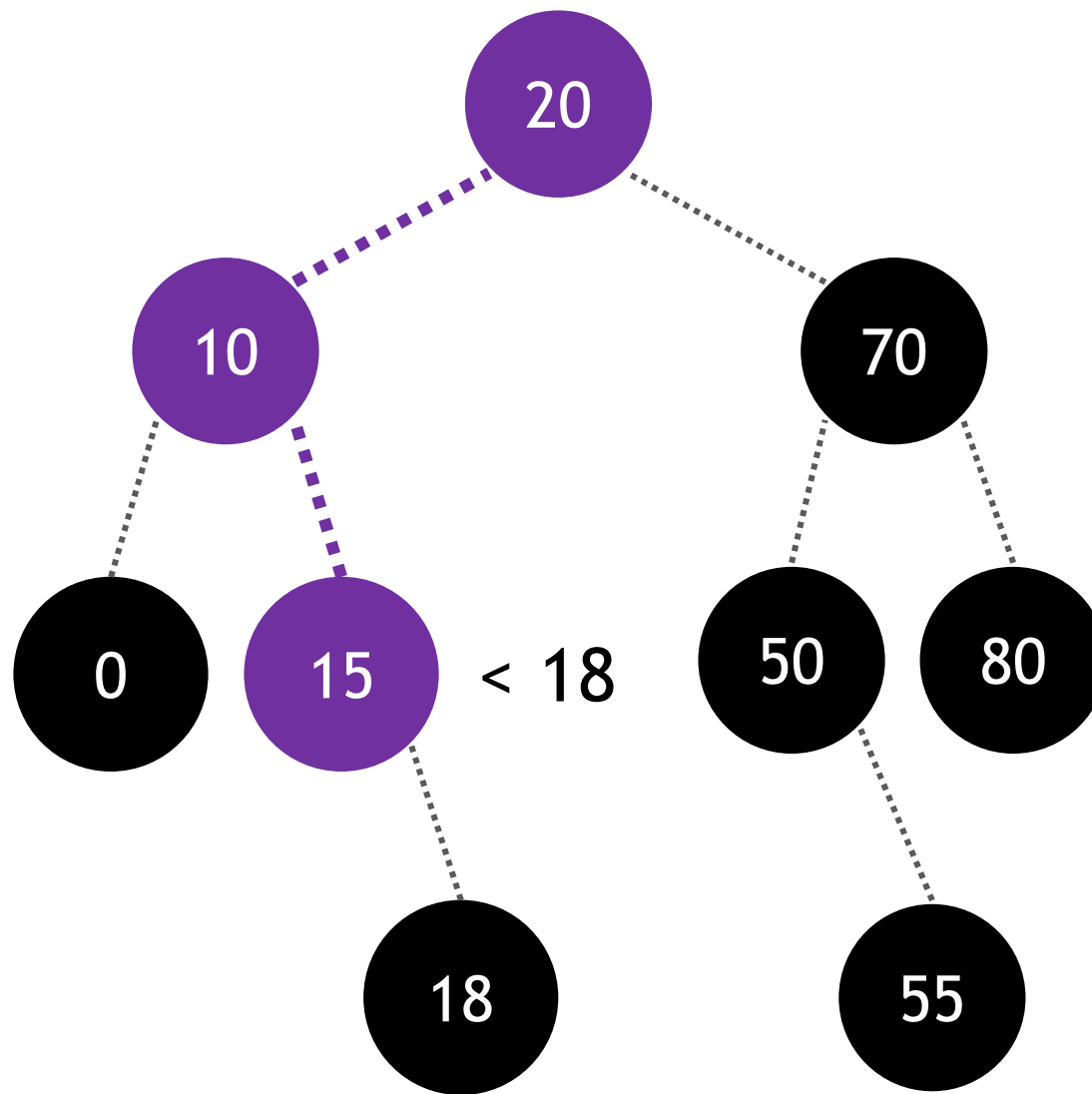
Hľadáme: 18



Hľadáme: 18

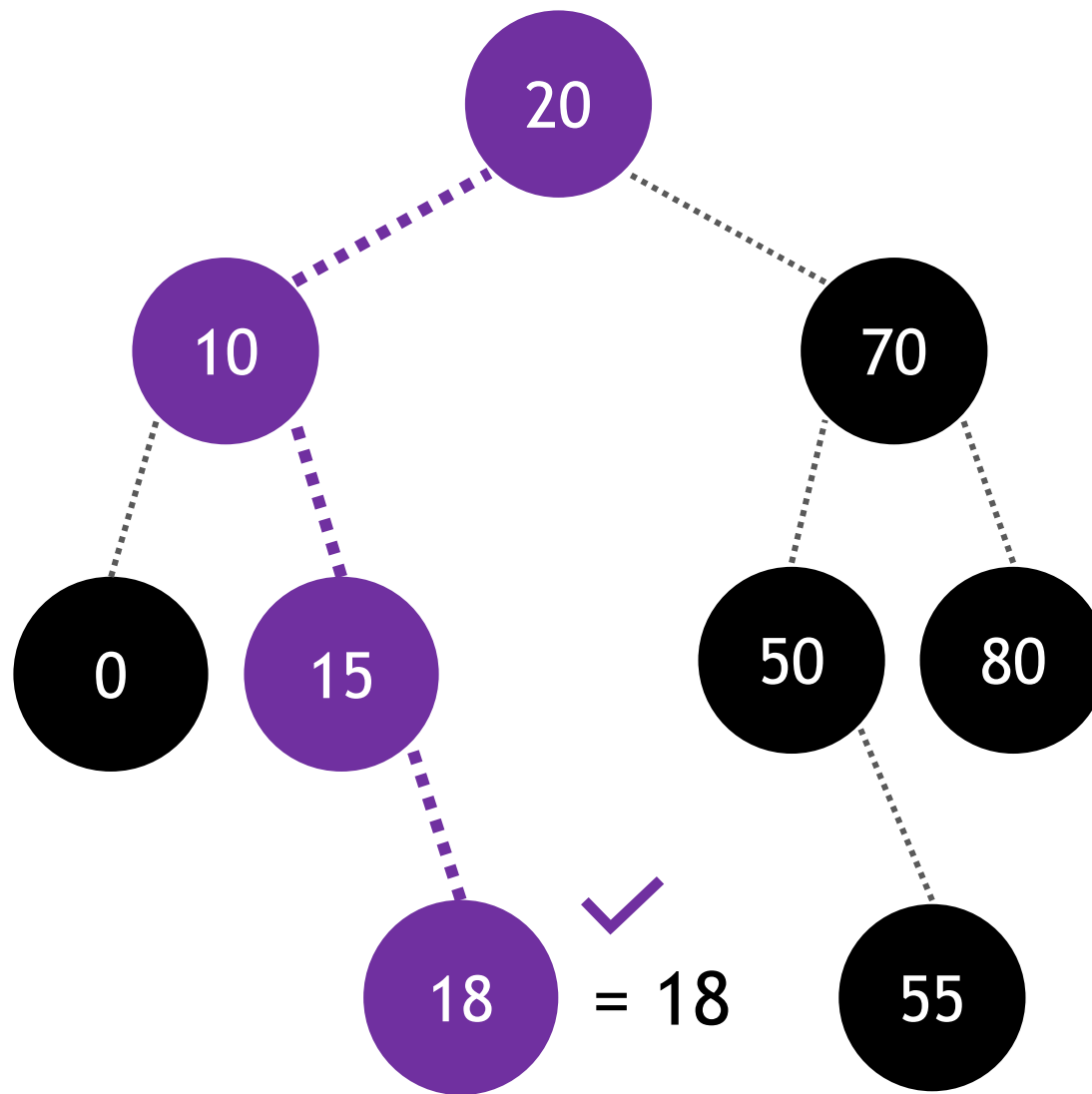


Hľadáme: 18





Hľadáme: 18



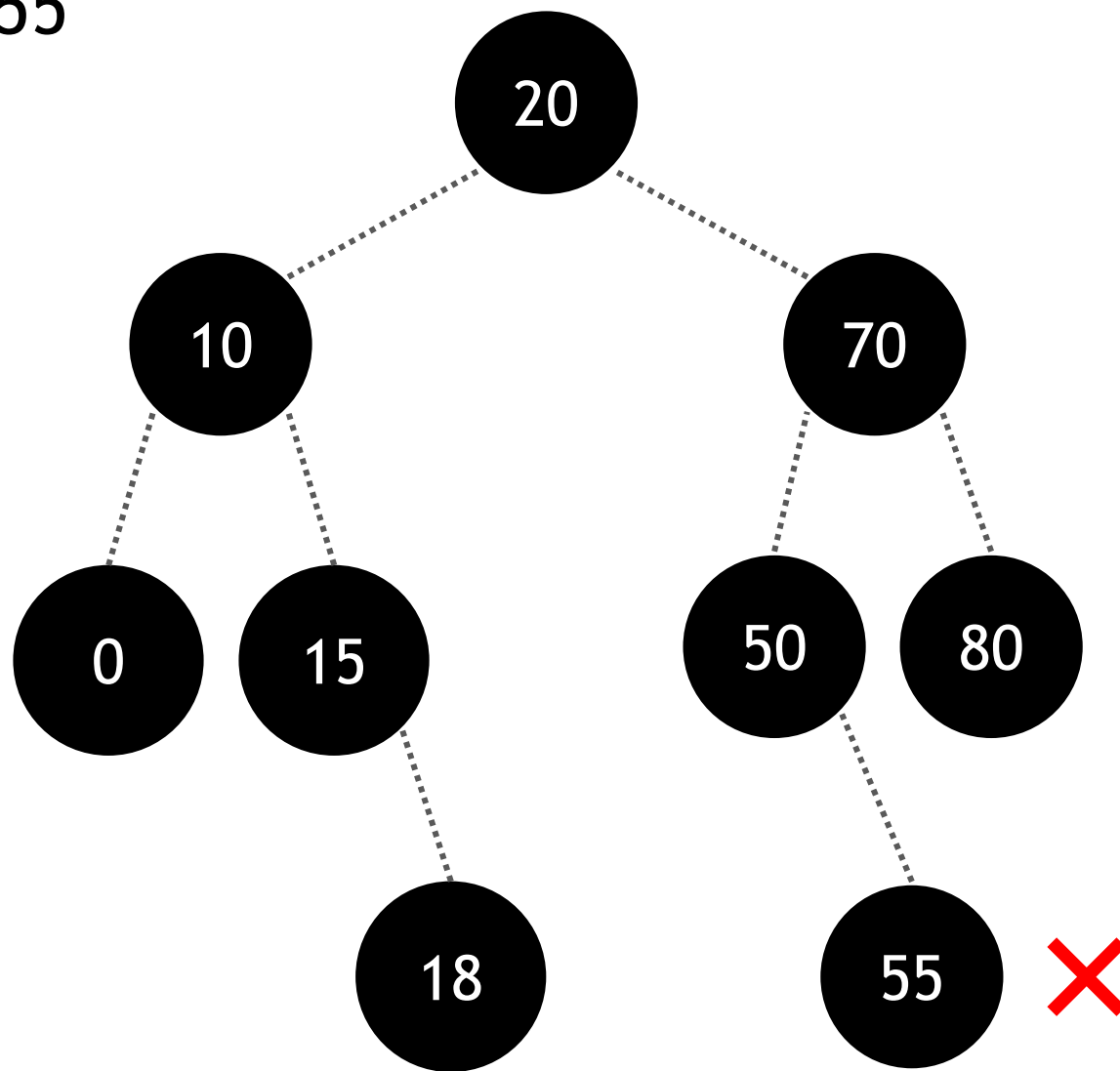
**Vymazávanie uzlov**

# Vymazávanie uzlov

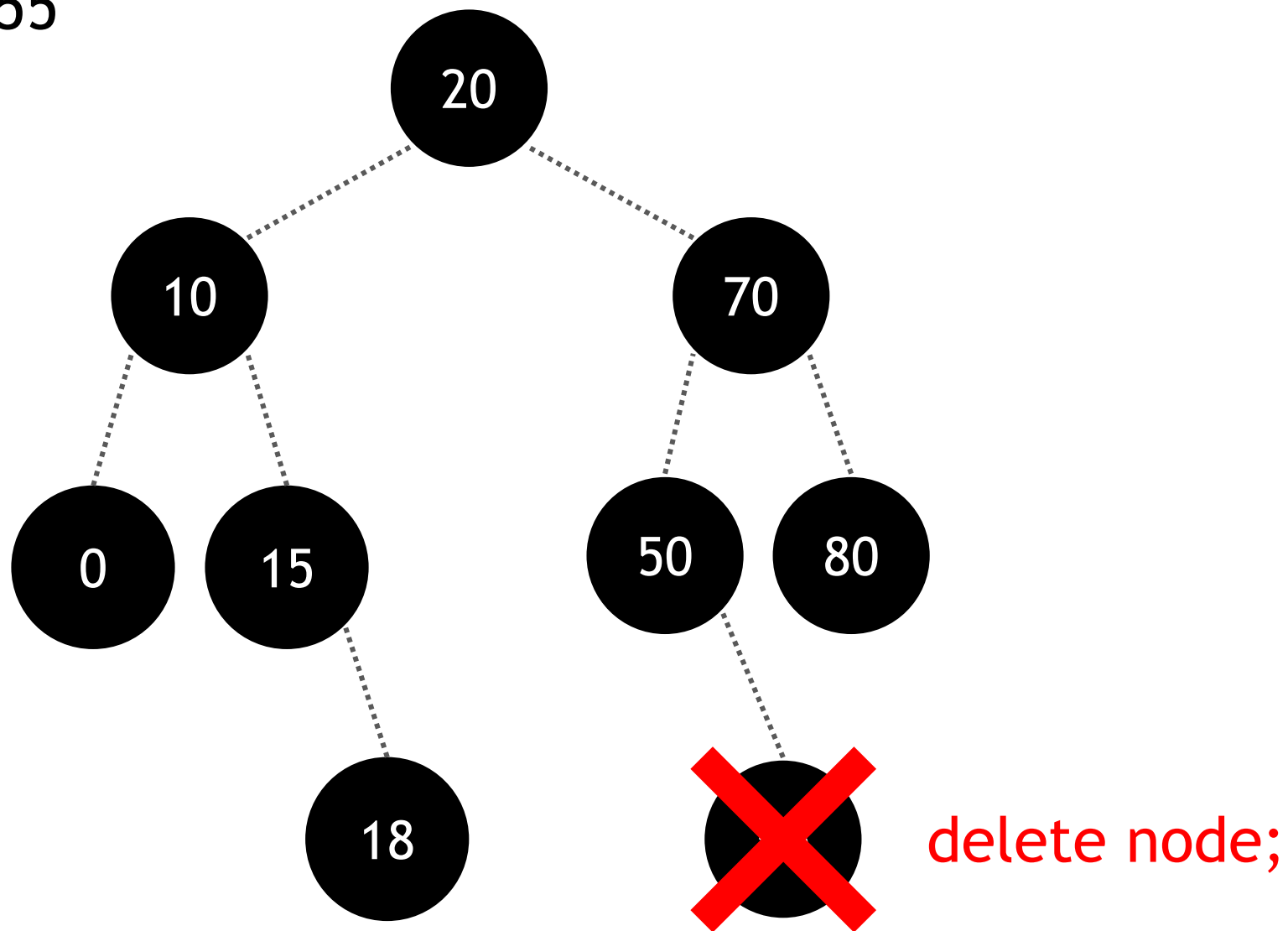
## Situácie

- Vymazávanie uzla s 0 potomkami.
- Vymazávanie uzlov s 1 potomkom.
- Vymazávanie uzlov s 2 potomkami.

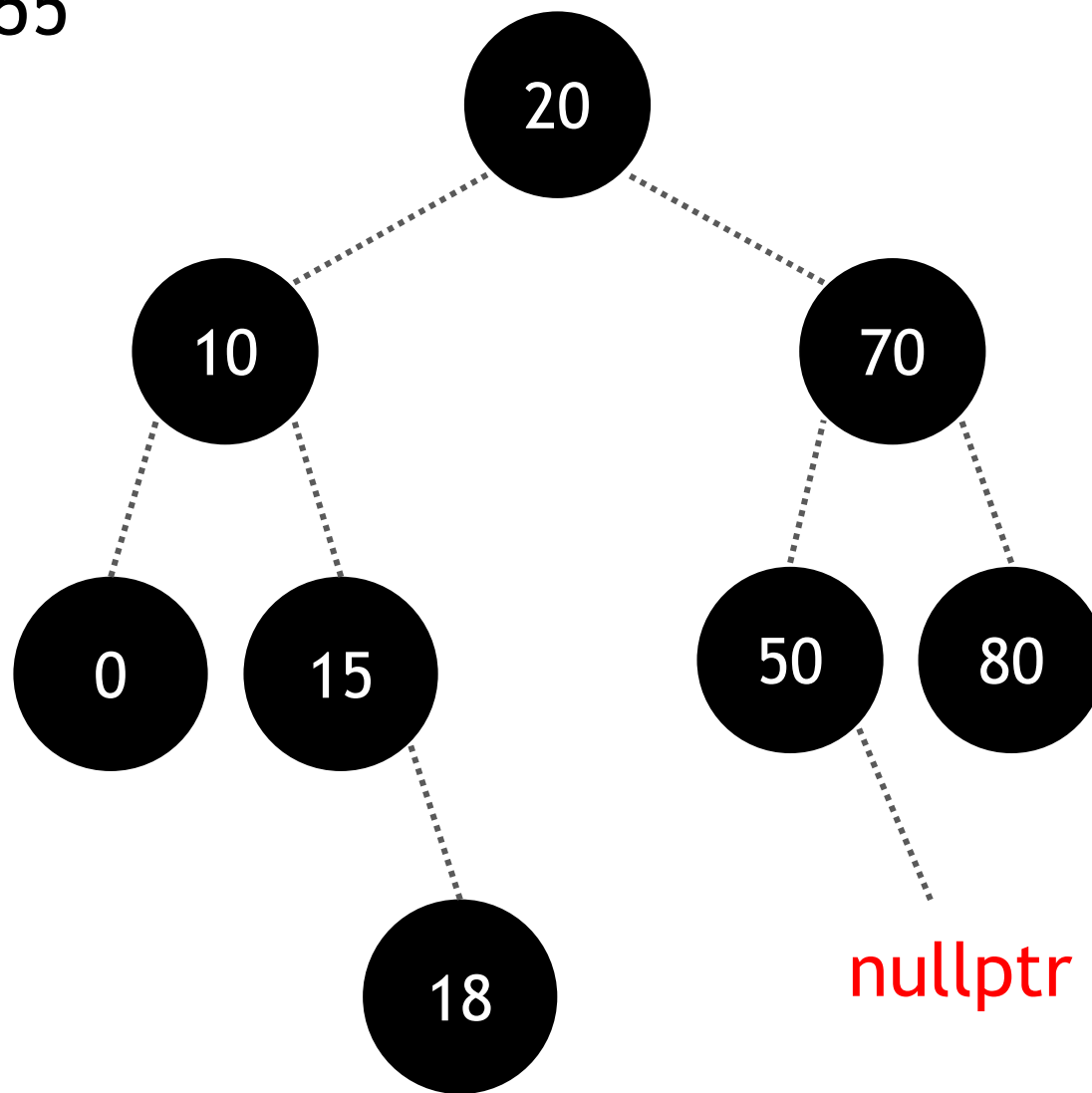
Vymazávame: 55



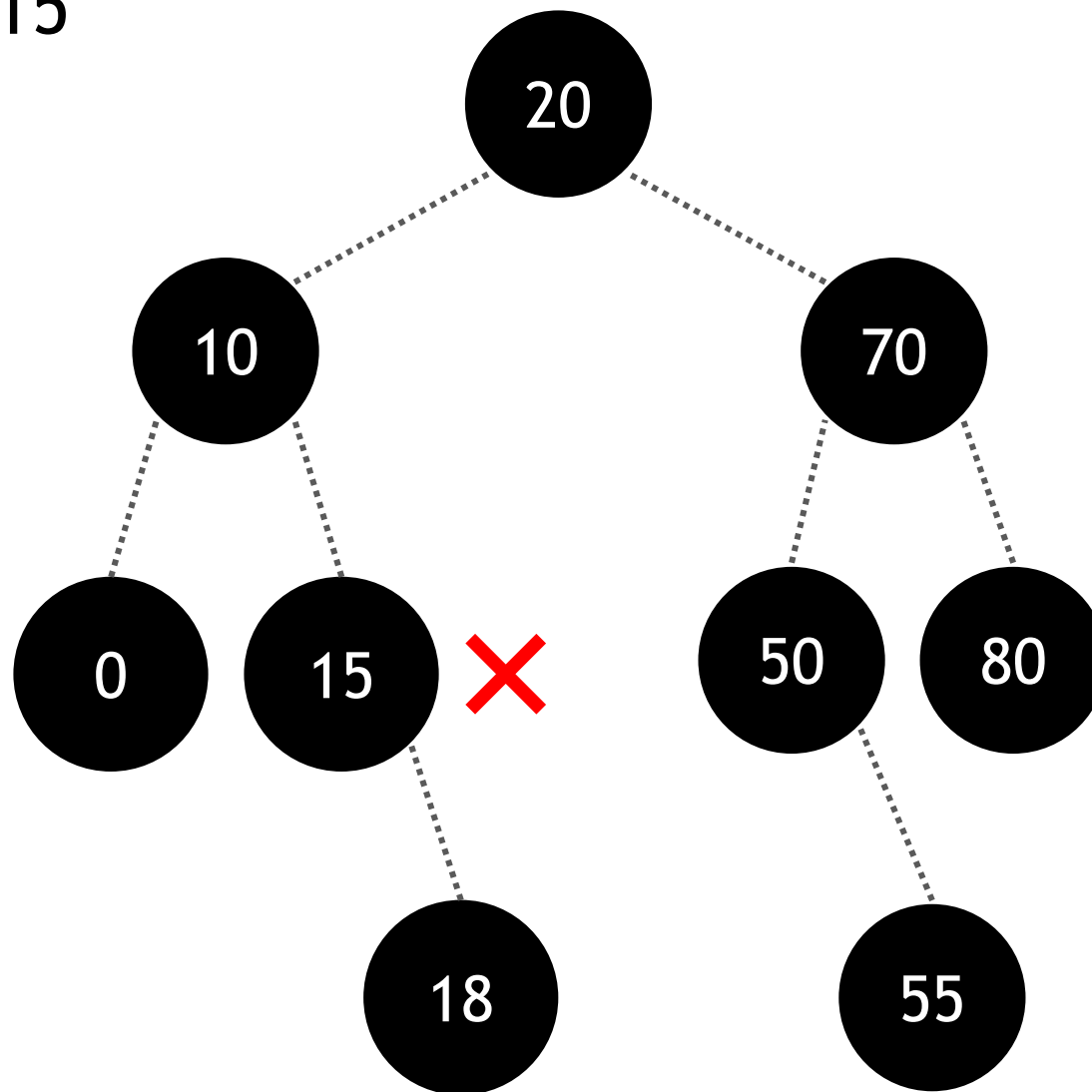
Vymazávame: 55



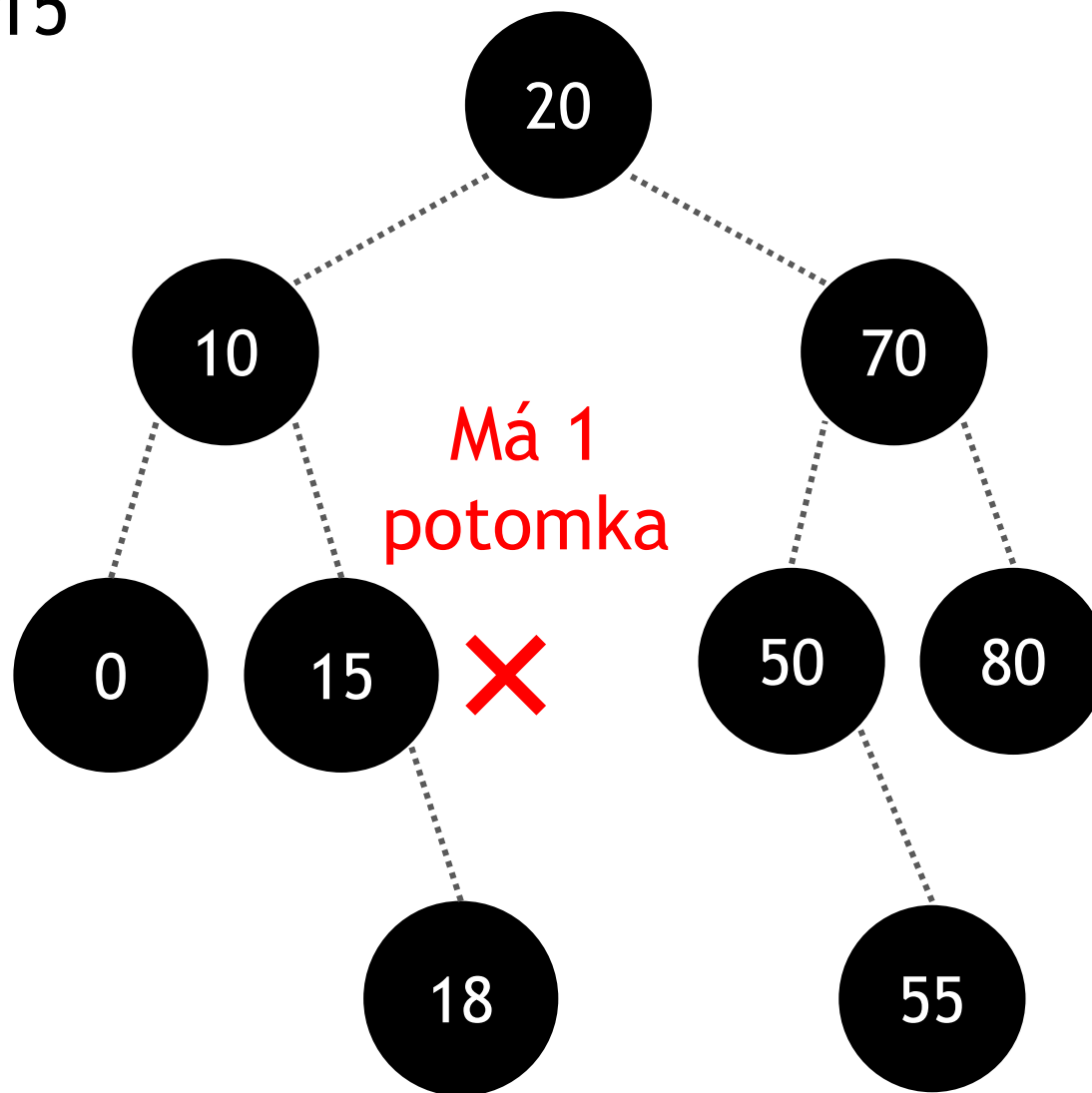
Vymazávame: 55



Vymazávame: 15

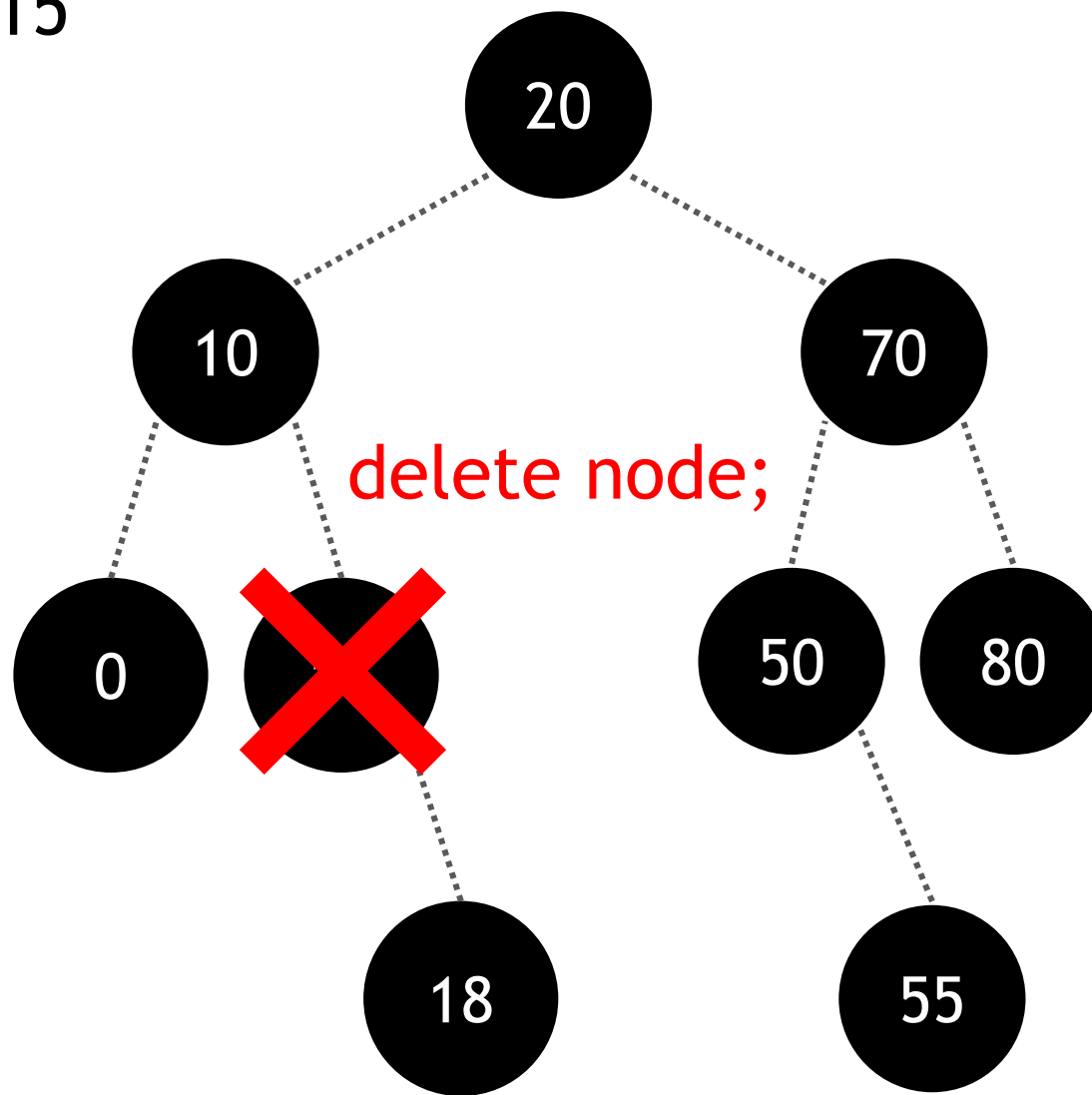


Vymazávame: 15

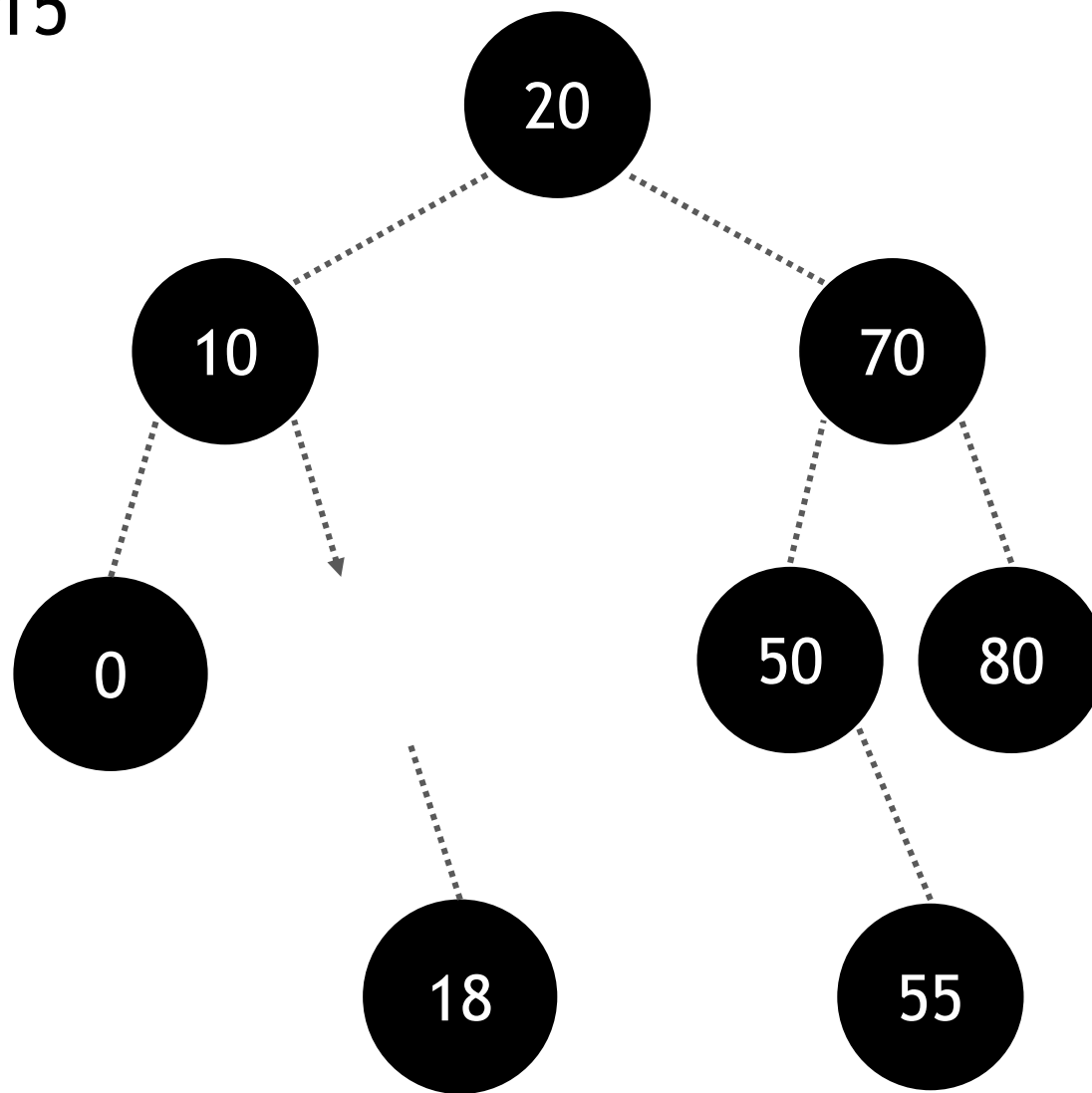




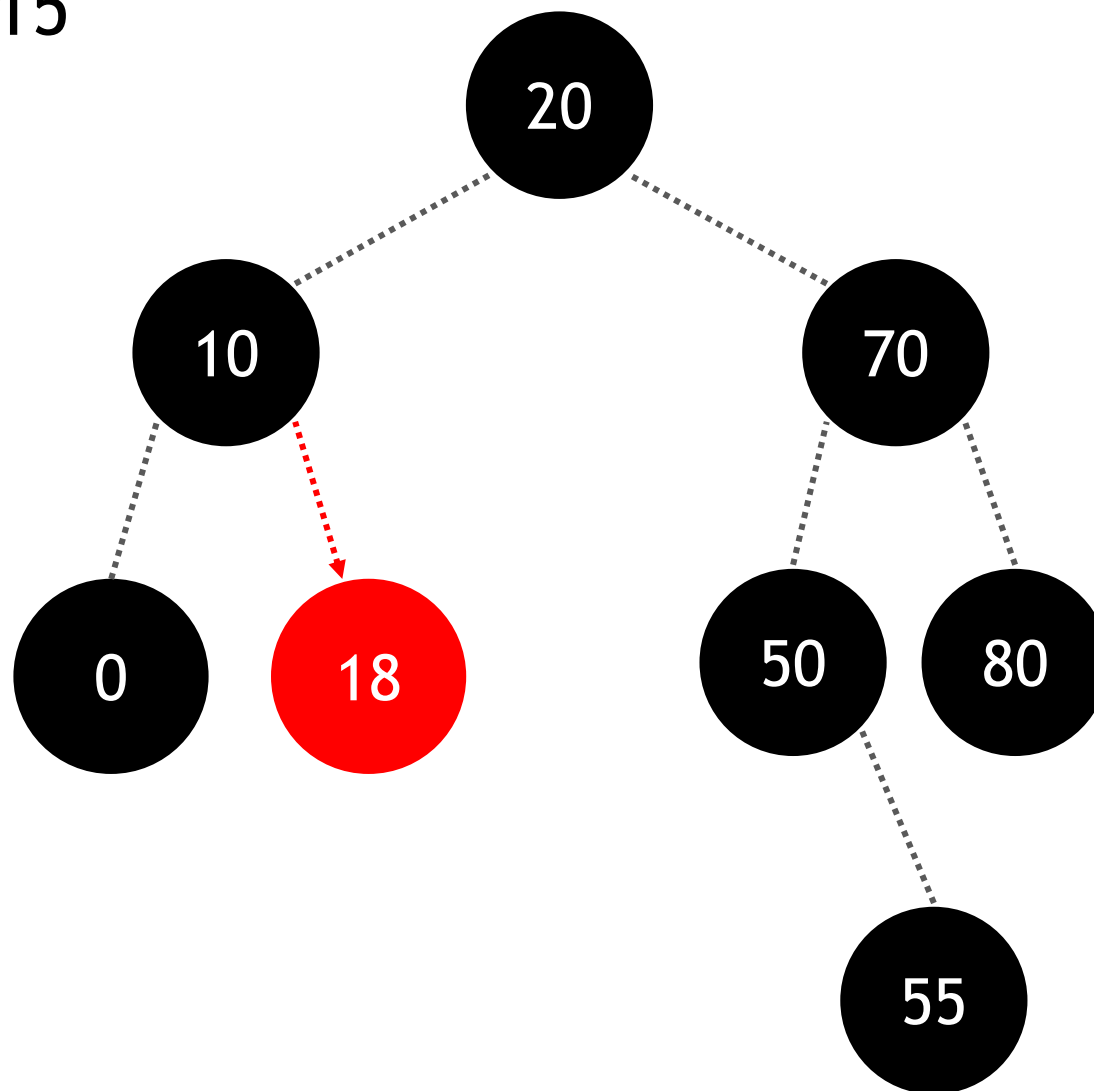
Vymazávame: 15



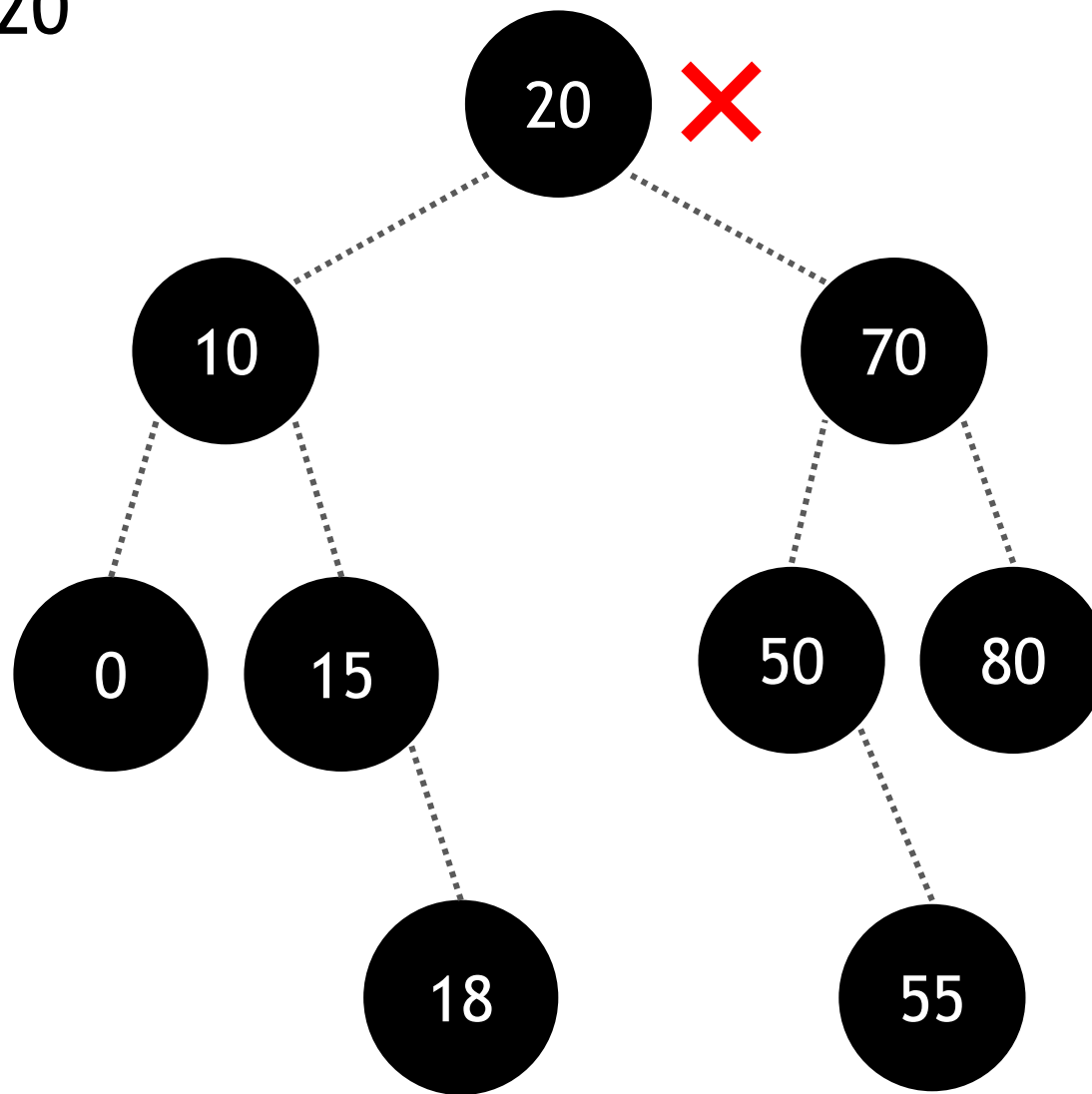
Vymazávame: 15



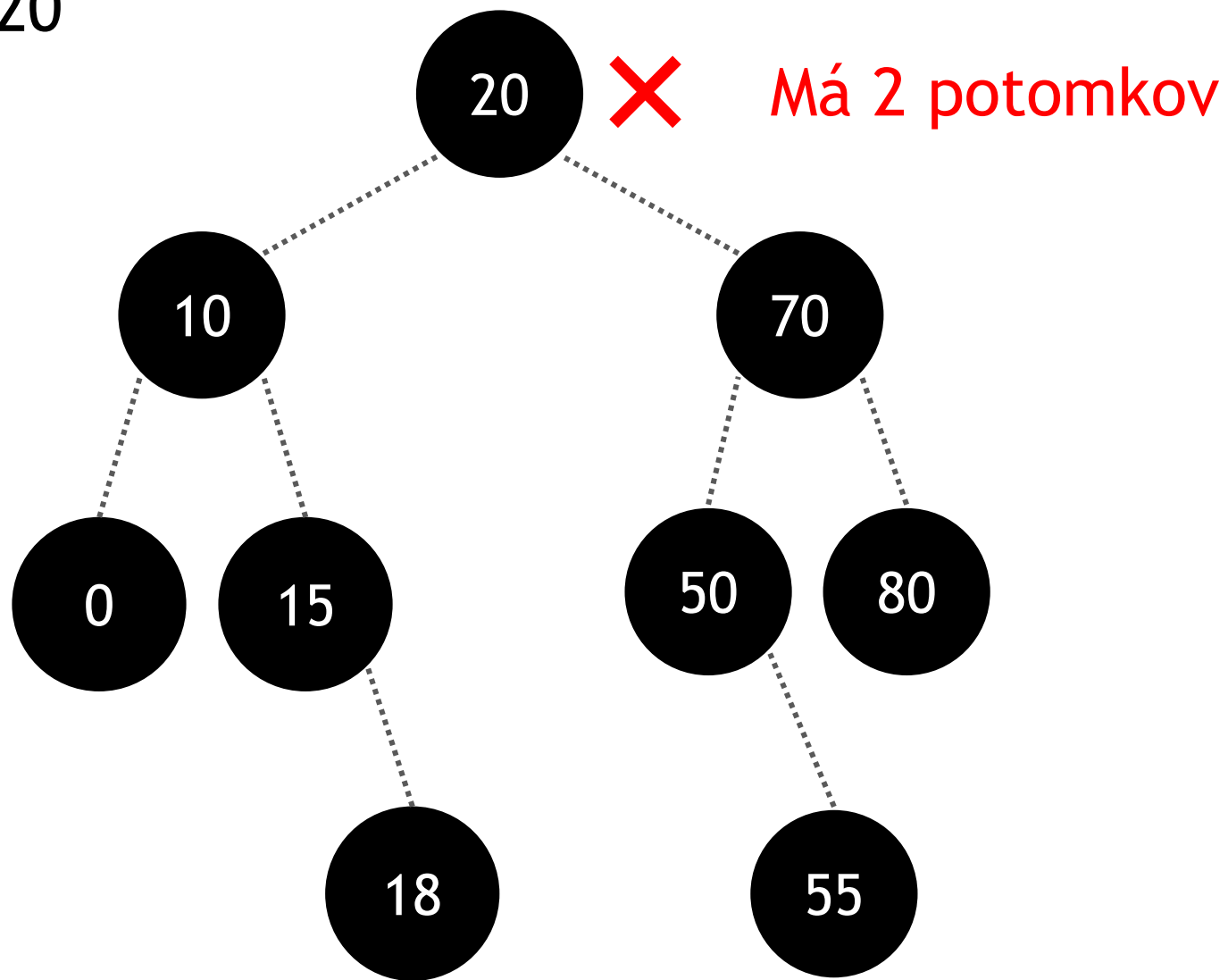
Vymazávame: 15



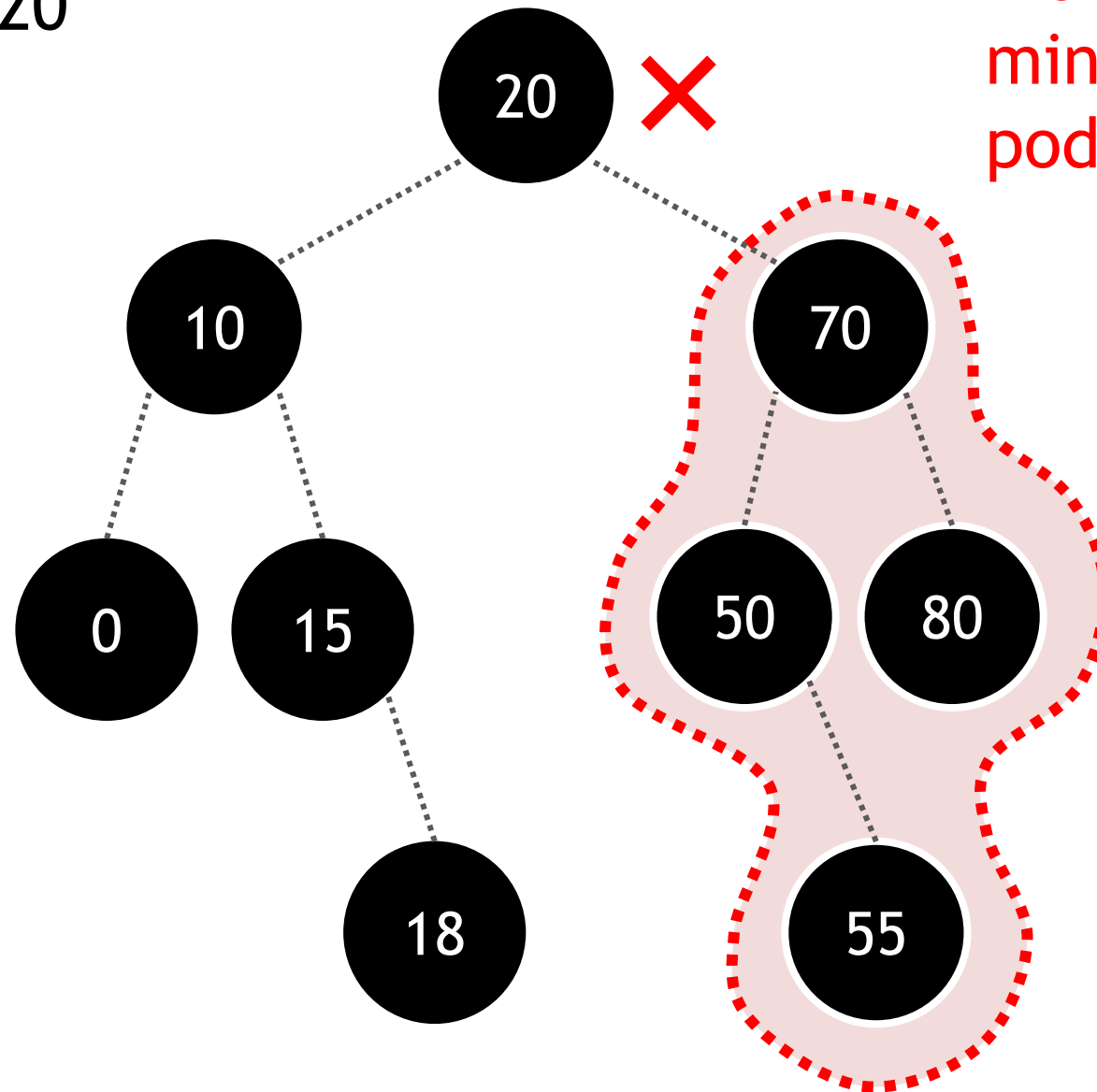
Vymazávame: 20



Vymazávame: 20

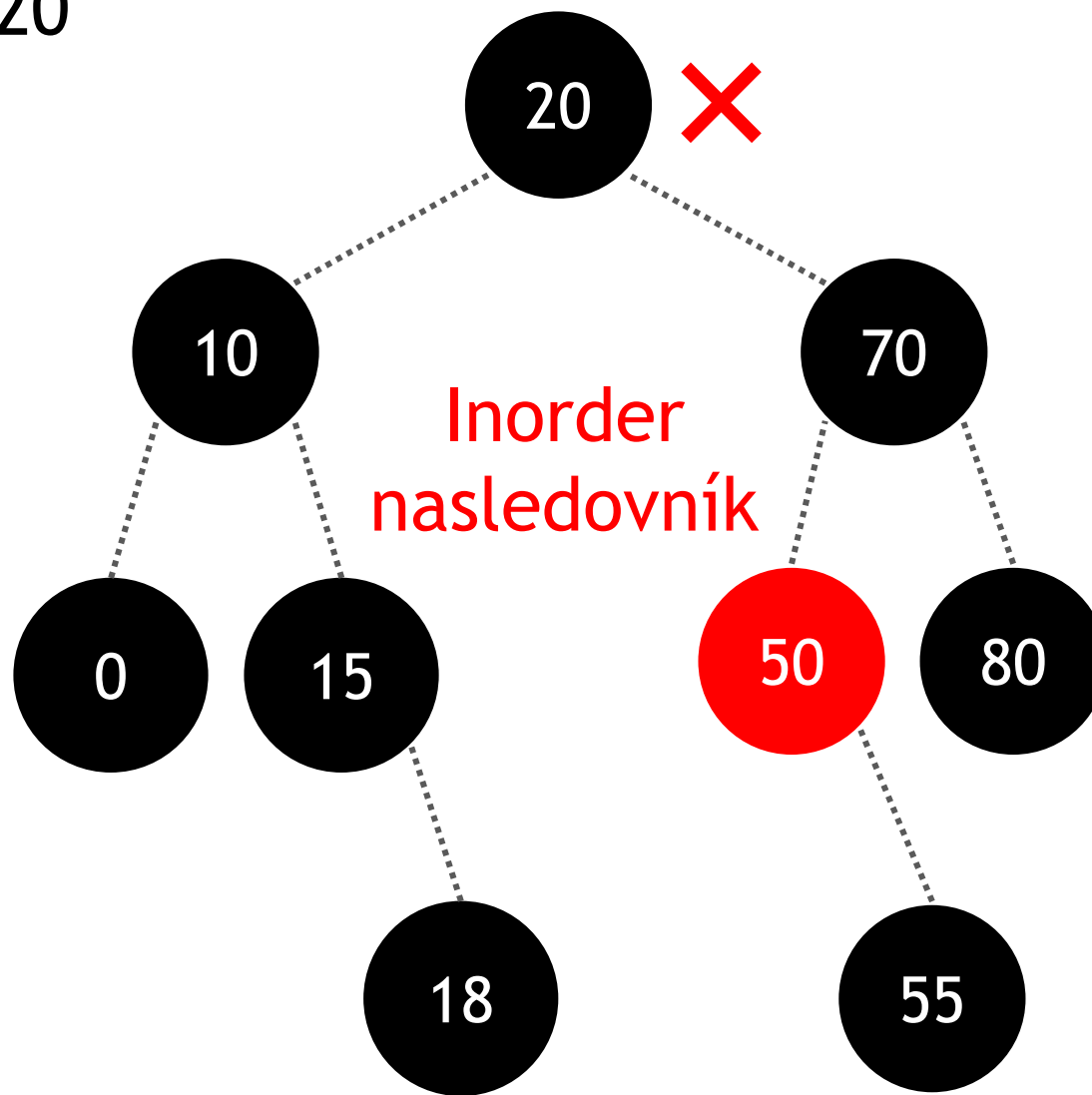


Vymazávame: 20

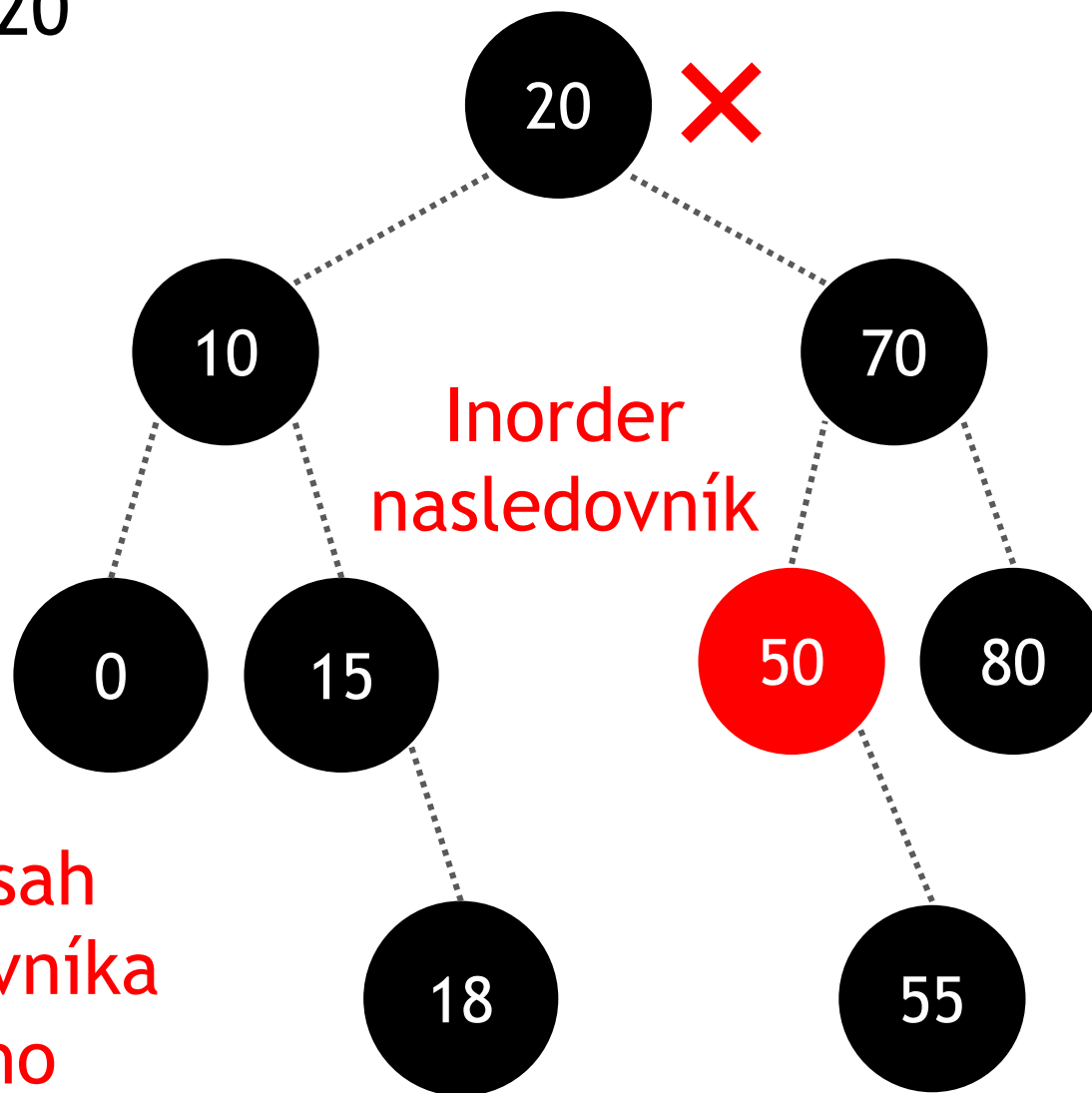


Inorder nasledovník je minimum v pravom podstrome.

Vymazávame: 20



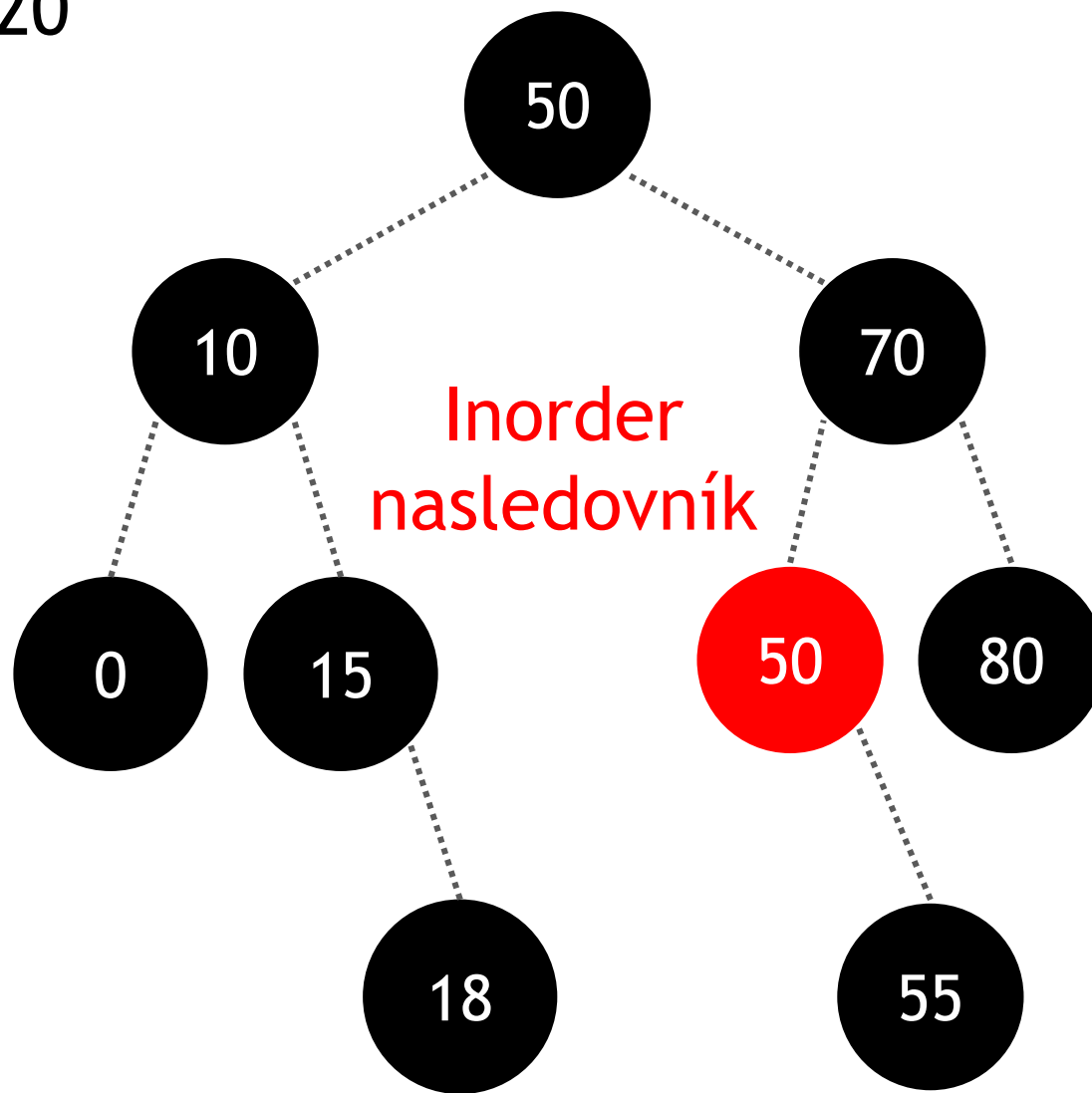
Vymazávame: 20



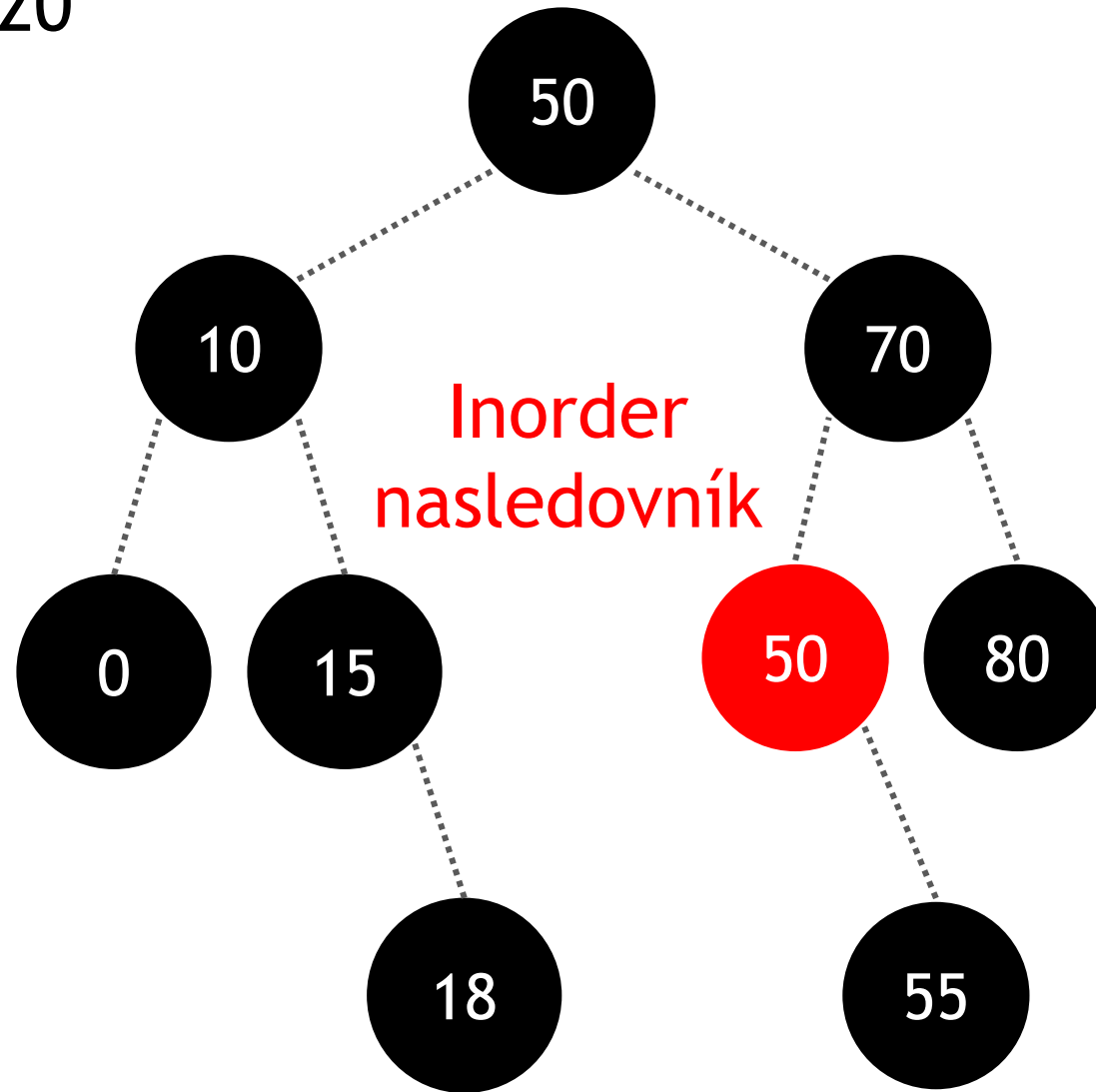
Skopírujeme obsah  
inorder nasledovníka  
do vymazávaného  
uzla.



Vymazávame: 20

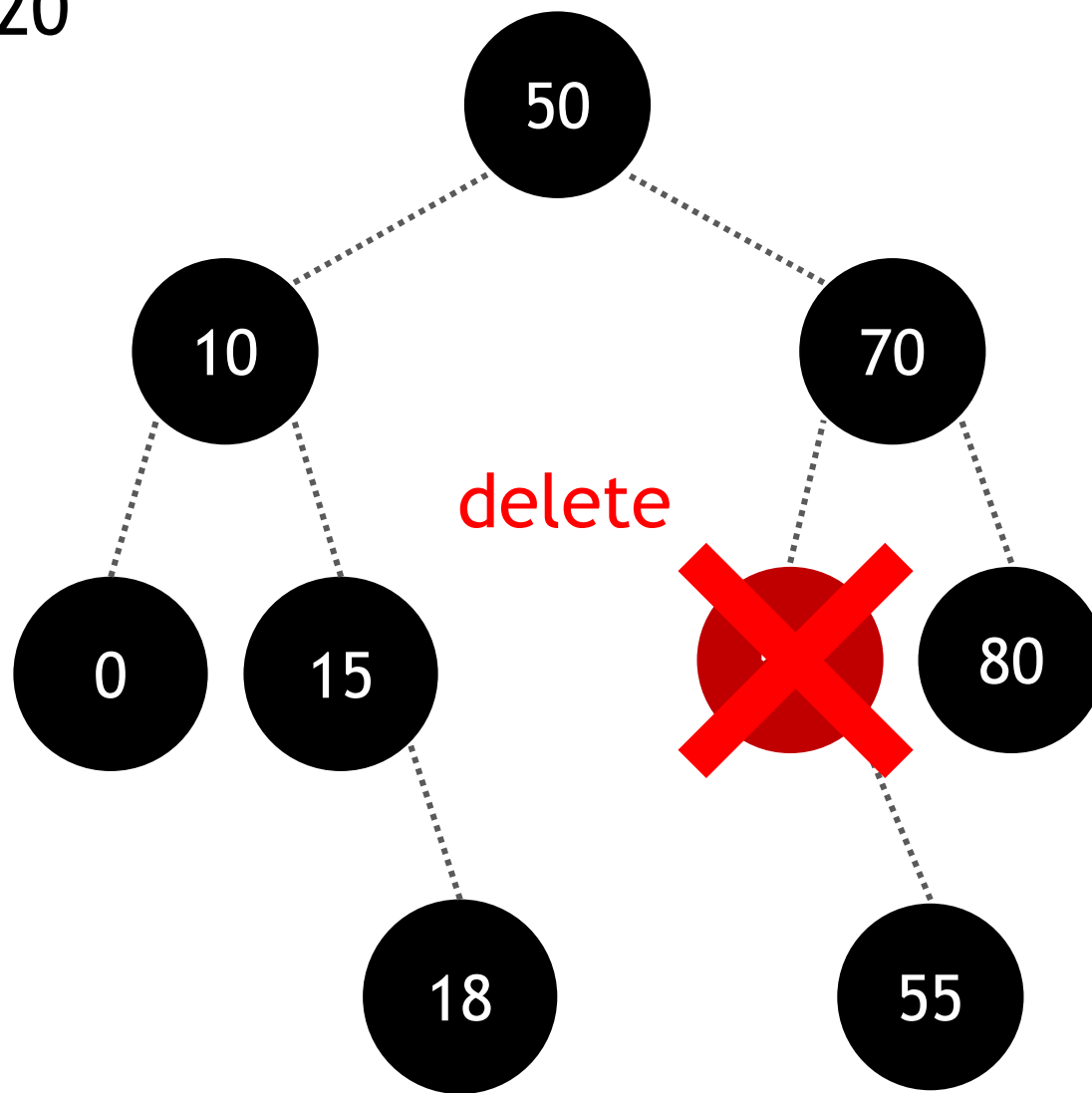


Vymazávame: 20

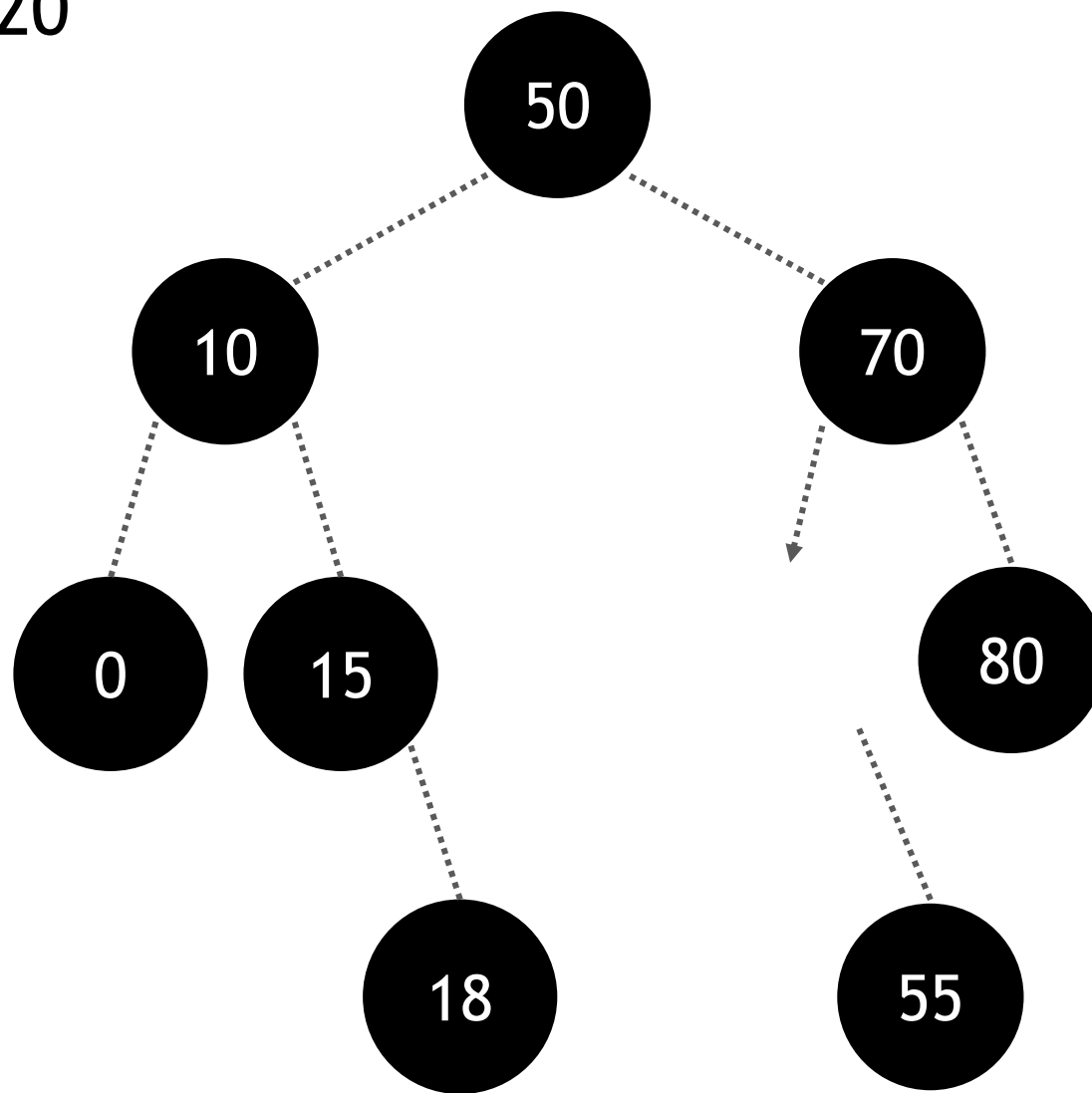


Vymažeme inorder  
nasledovníka.

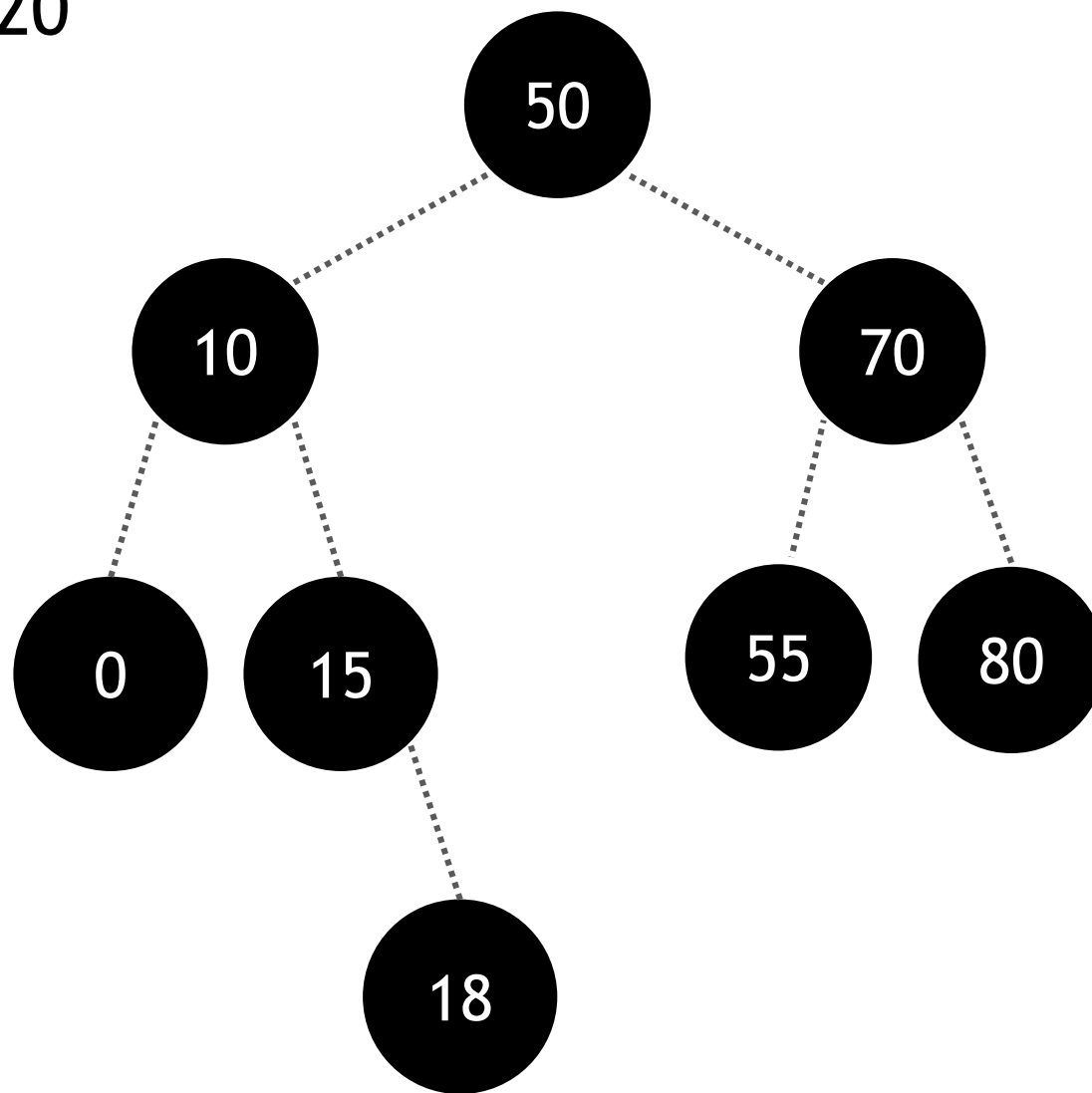
Vymazávame: 20



Vymazávame: 20



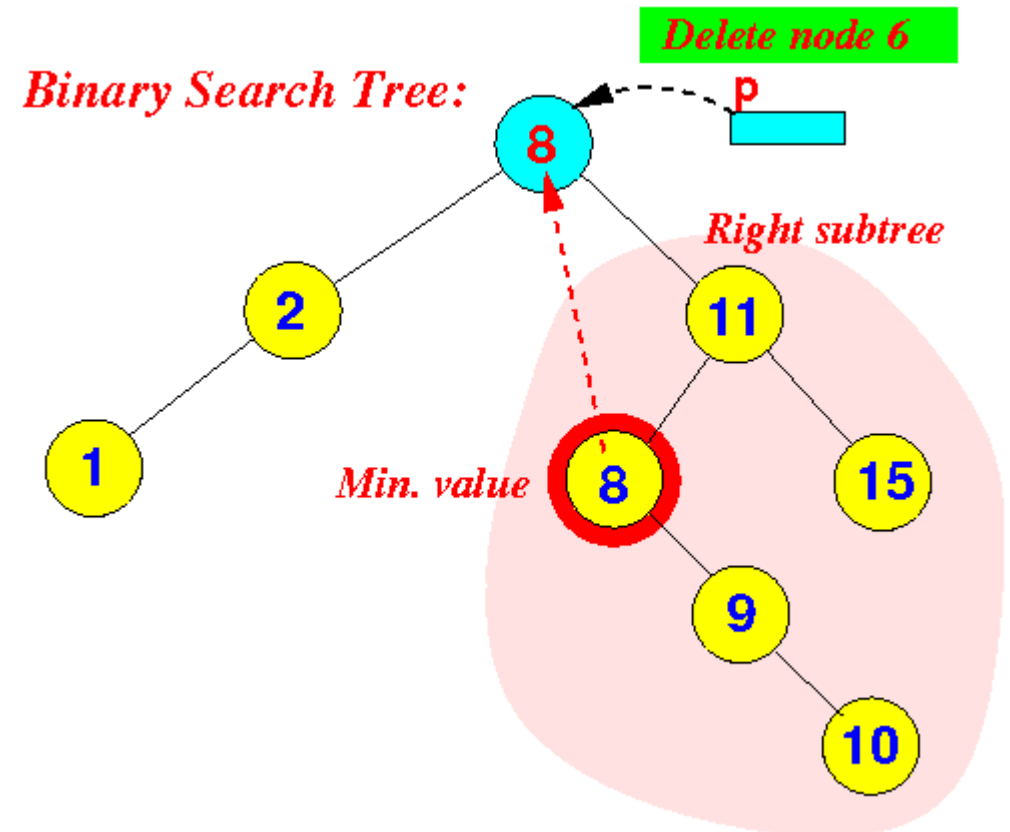
Vymazávame: 20



# Literatúra (vymazávanie uzlov)

<http://www.mathcs.emory.edu/~cheung/Courses/171/Syllabus/9-BinTree/BST-delete.html>

<http://www.mathcs.emory.edu/~cheung/Courses/171/Syllabus/9-BinTree/BST-delete2.html>



# Vzorová implementácia v C/C++