Lecture 11

Augmenting Data Structures

Source: Introduction to Algorithms, CLRS

Finding the Element with ith Rank

Recall that rank of an element is its position in the sorted order (w.r.t. keys) of the set.

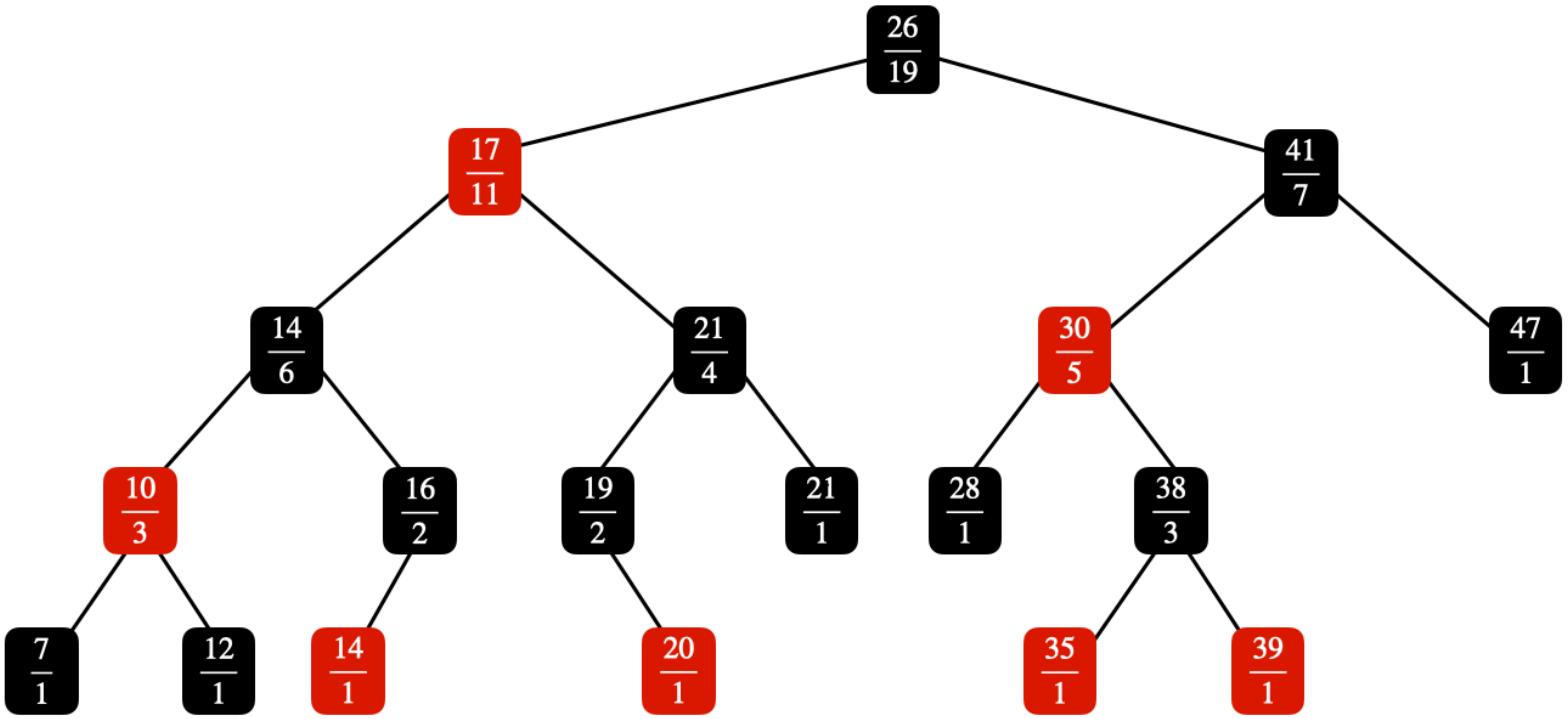
Finding the Element with ith Rank

Finding the Element with ith Rank

Example: Find an element with 15th rank in the below **set** or R**B-tree**.

Finding the Element with ith Rank

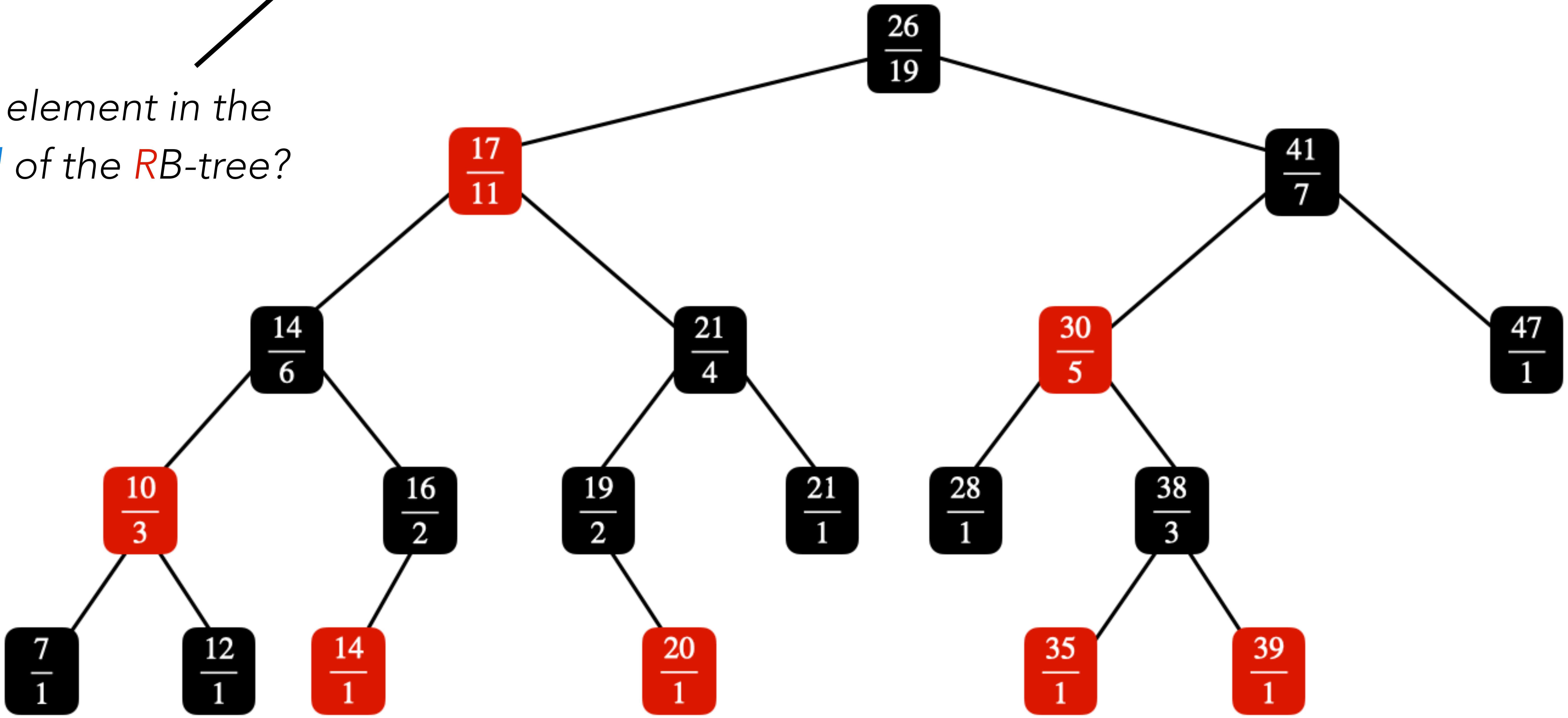
Example: Find an element with 15th rank in the below **set** or RB-tree.



Finding the Element with ith Rank

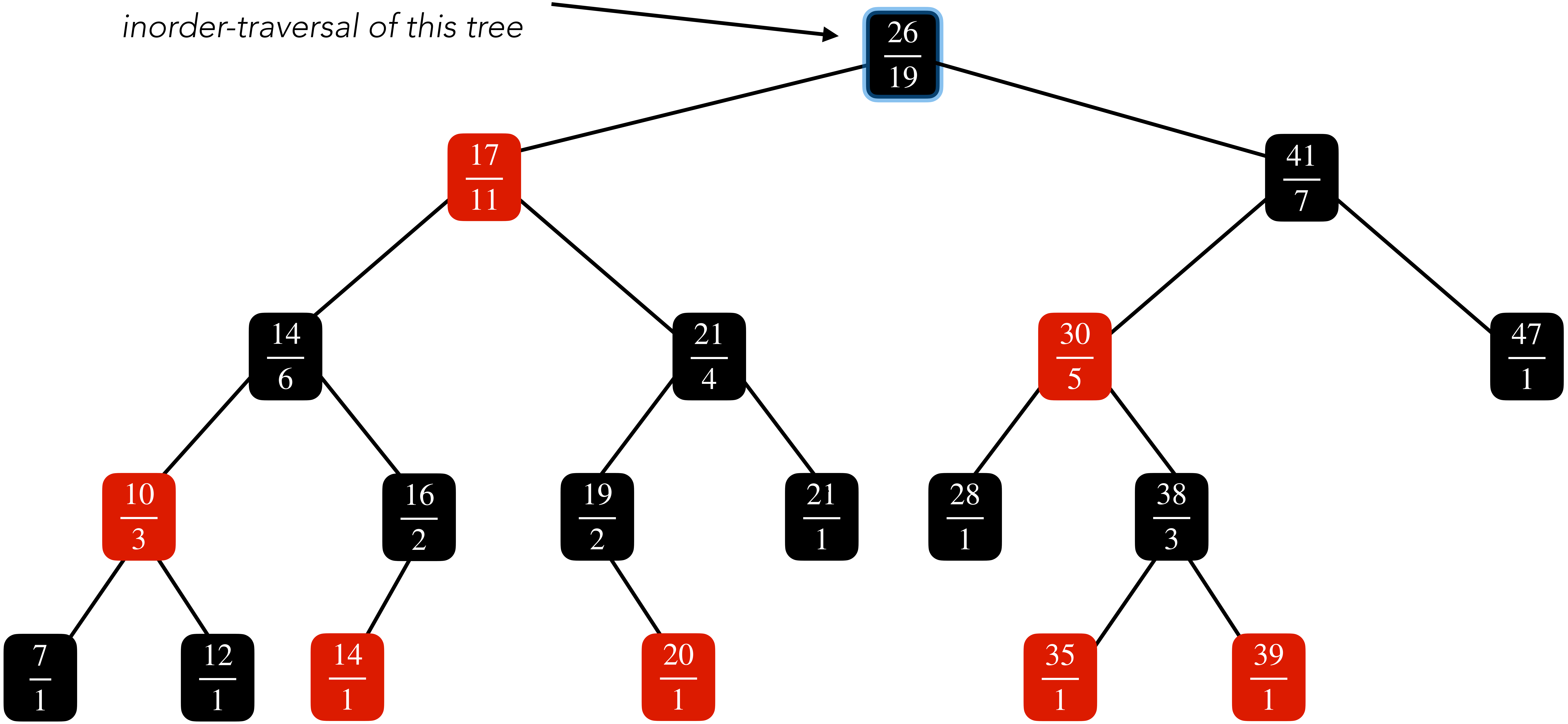
Example: Find an element with 15th rank in the below **set** or **RB-tree**.

Isn't it the 15th element in the inorder-traversal of the RB-tree?

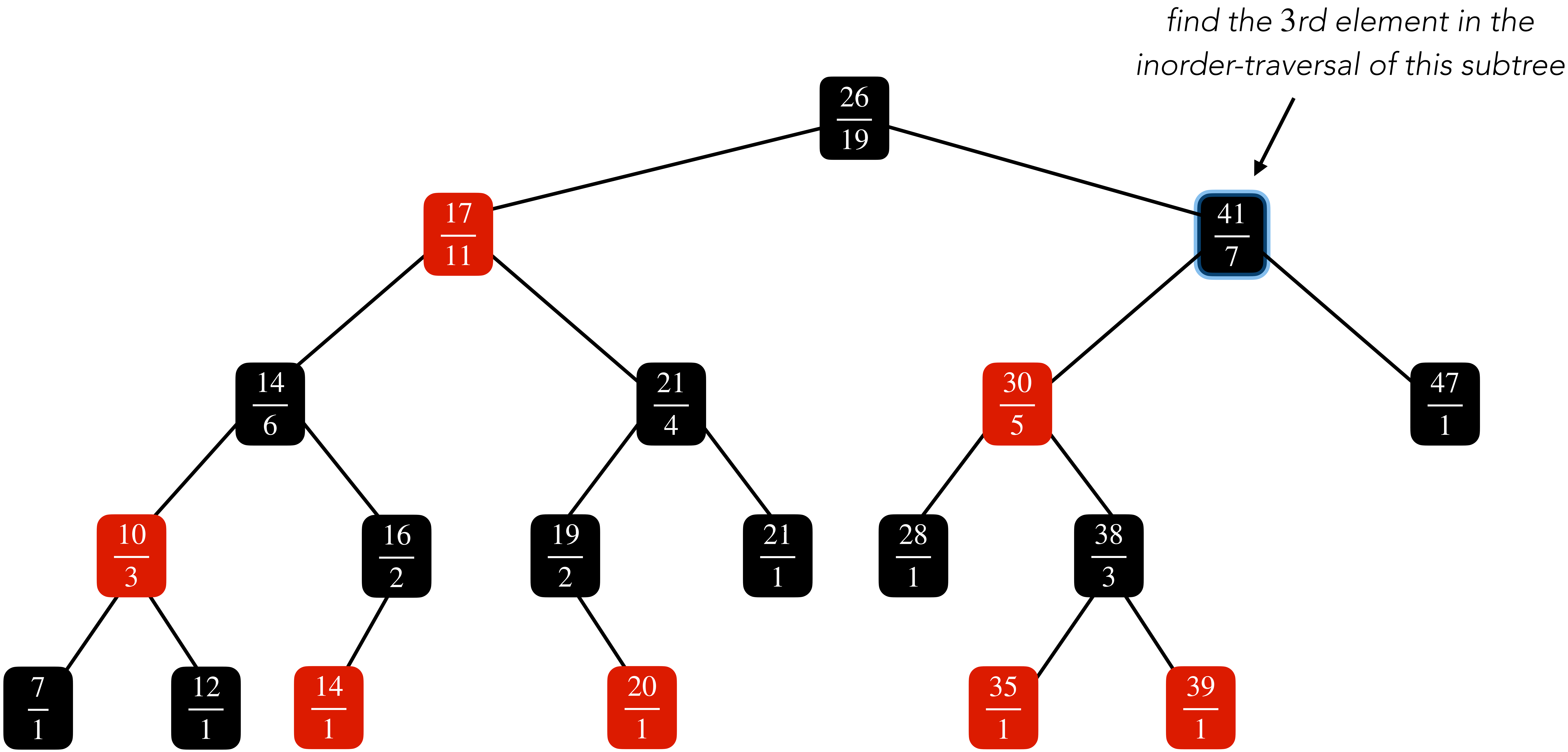


Finding the Element with ith Rank

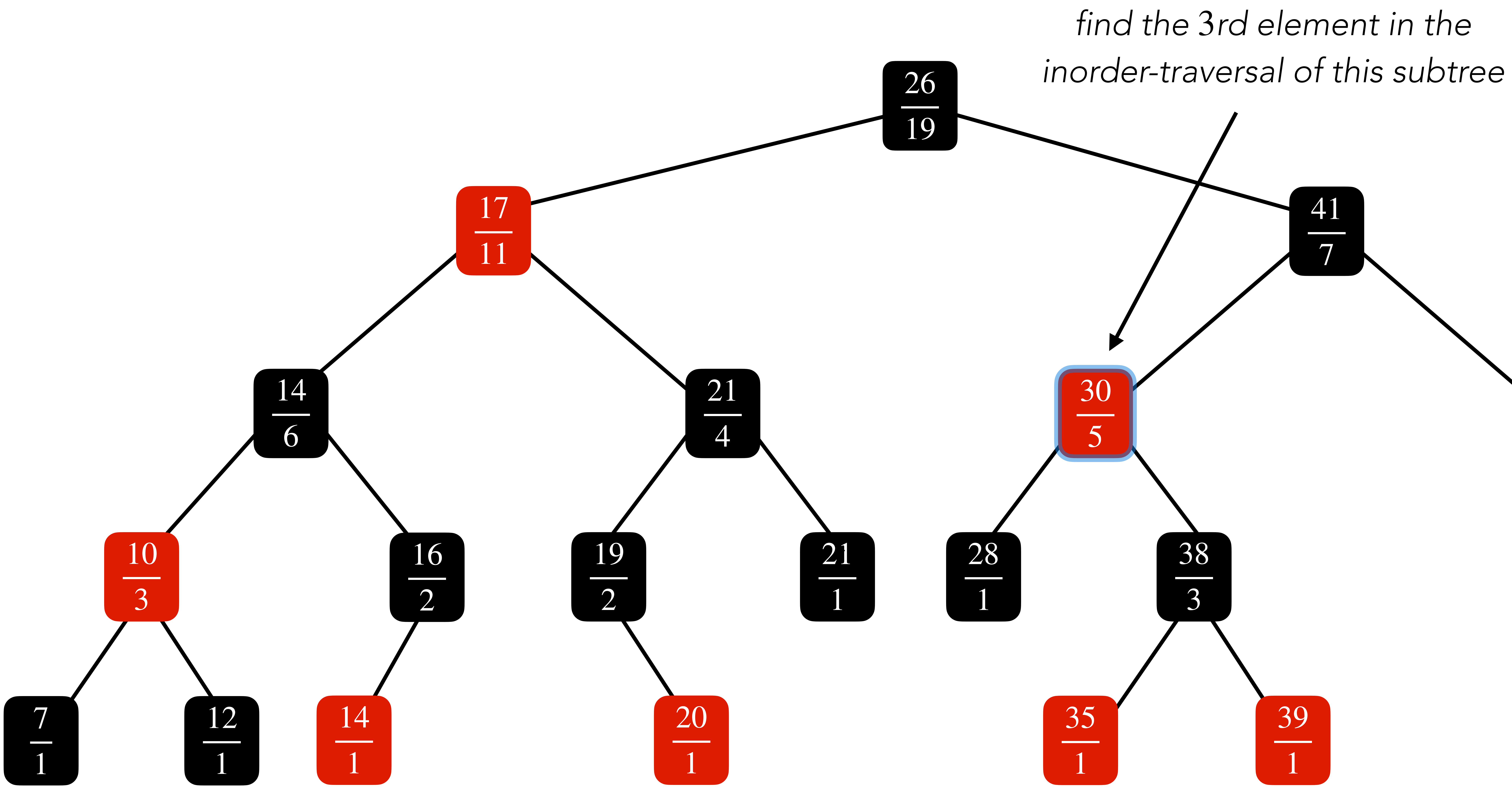
find the 15th element in the inorder-traversal of this tree



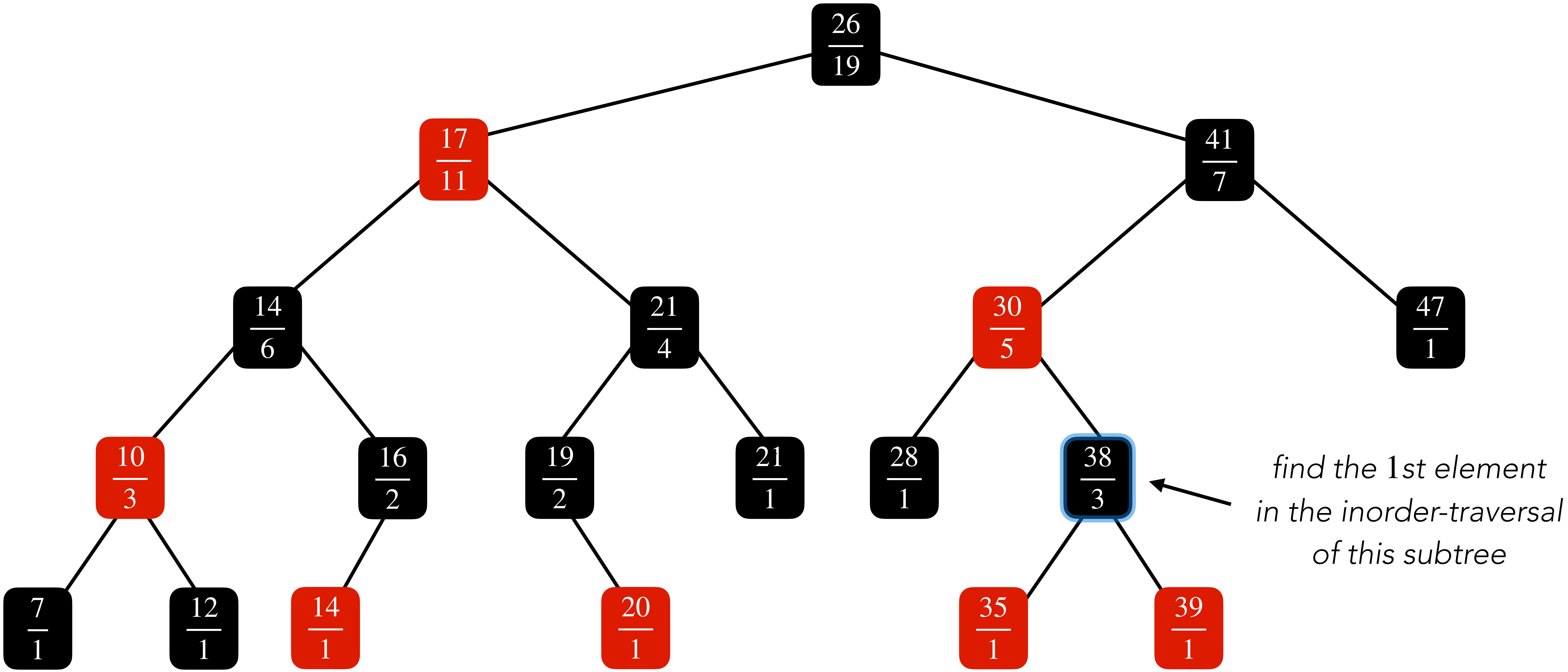
Finding the Element with ith Rank



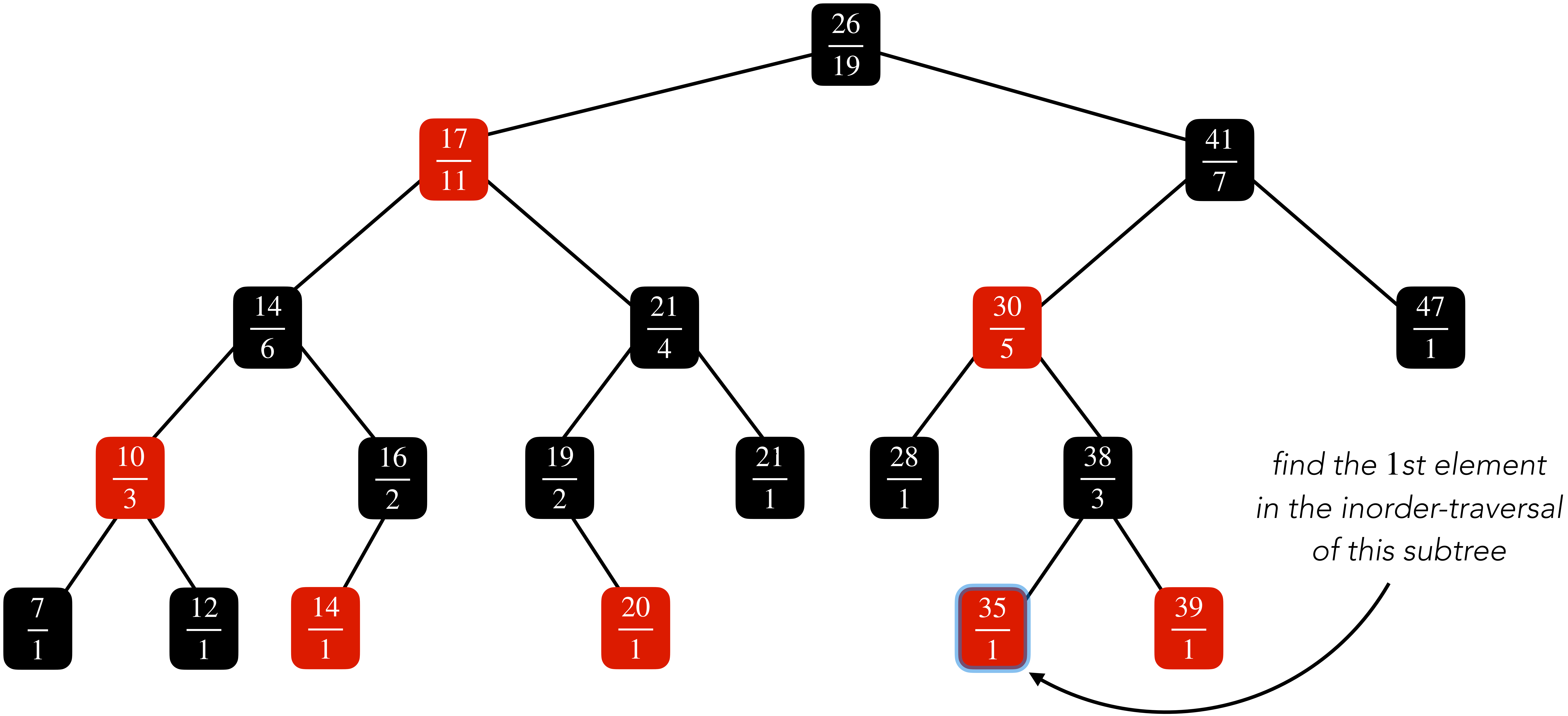
Finding the Element with ith Rank



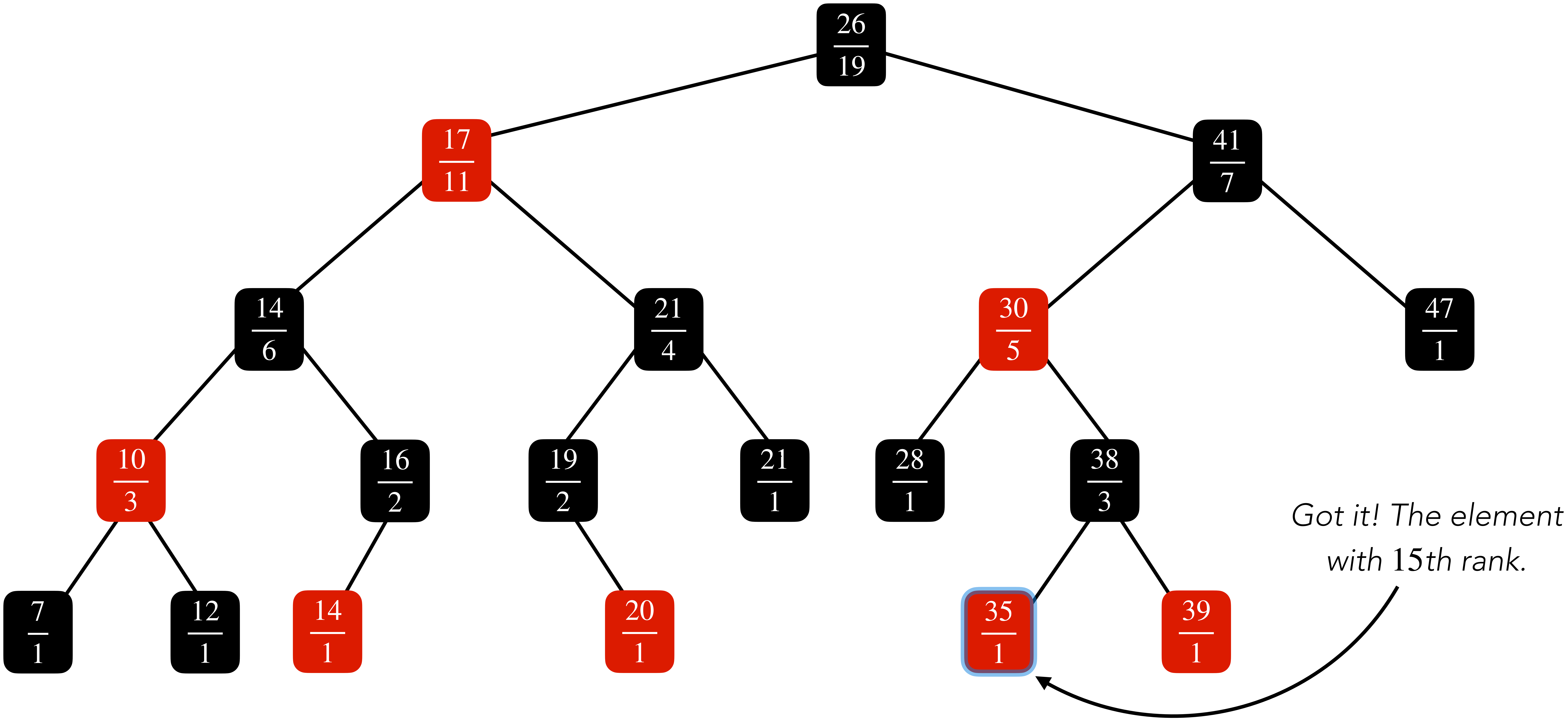
Finding the Element with ith Rank



Finding the Element with ith Rank



Finding the Element with ith Rank



Finding the Element with ith Rank

Finding the Element with ith Rank

To find the element with ith rank in T call **Select**$(T.root, i)$.

Finding the Element with ith Rank

To find the element with ith rank in T call **Select**$(T.root, i)$.

Select(x, i):

Finding the Element with ith Rank

To find the element with ith rank in T call **Select**$(T.root, i)$.

Select(x, i):

1. $r = x.left.size + 1$

Finding the Element with ith Rank

To find the element with ith rank in T call **Select**$(T.root, i)$.

Select(x, i):

1. $r = x.left.size + 1$

calculate the rank of x

Finding the Element with ith Rank

To find the element with ith rank in T call **Select**$(T.root, i)$.

Select(x, i)**:**

1. $r = x.left.size + 1$
2. **if** $i == r$

calculate the rank of x

Finding the Element with ith Rank

To find the element with ith rank in T call **Select**$(T.root, i)$.

Select(x, i):

1. $r = x.left.size + 1$
2. **if** $i == r$
3.   **return** x

calculate the rank of x

Finding the Element with ith Rank

To find the element with ith rank in T call **Select**$(T\,.\,root, i)$.

Select(x, i):

1. $r = x\,.\,left\,.\,size + 1$
2. **if** $i == r$
3. **return** x
4. **else if** $i < r$

calculate the rank of x

Finding the Element with ith Rank

To find the element with ith rank in T call **Select**$(T.root, i)$.

Select(x, i):

1. $r = x.left.size + 1$
2. **if** $i == r$
3. **return** x
4. **else if** $i < r$

calculate the rank of x

element with ith rank must fall in the left-subtree

Finding the Element with ith Rank

To find the element with ith rank in T call **Select**$(T.root, i)$.

Select(x, i):

$r = x.left.size + 1$

if $i == r$

    return x

else if $i < r$

    return ______________

calculate the rank of x

*element with **ith rank** must fall in the **left-subtree***

Finding the Element with ith Rank

To find the element with ith rank in T call **Select**$(T.root, i)$.

```
Select(x, i):
    r = x.left.size + 1
    if i == r
        return x
    else if i < r
        return
    else
```

calculate the rank of x

element with ***ith rank*** *must fall in the* ***left-subtree***

Finding the Element with ith Rank

To find the element with ith rank in T call **Select**$(T.root, i)$.

Select(x, i):

$r = x.left.size + 1$

if $i == r$

  return x

else if $i < r$

  return ______

else

calculate the rank of x

element with ith rank must fall in the left-subtree

element with ith rank must fall in the right-subtree

Finding the Element with ith Rank

To find the element with ith rank in T call **Select**$(T\,.\,root, i)$.

Select(x, i):
1. $r = x\,.\,left\,.\,size + 1$
2. **if** $i == r$
3. **return** x
4. **else if** $i < r$
5. **return** ______________
6. **else**
7. **return** ______________

calculate the rank of x

element with ith rank must fall in the left-subtree

element with ith rank must fall in the right-subtree

Finding the Element with ith Rank

To find the element with ith rank in T call **Select**$(T\,.\,root, i)$.

Select(x, i):

$r = x\,.\,left\,.\,size + 1$

if $i == r$

    return x

else if $i < r$

    return **Select**$(x\,.\,left, i)$

else

    return

calculate the rank of x

element with ith rank must fall in the left-subtree

element with ith rank must fall in the right-subtree

Finding the Element with ith Rank

To find the element with ith rank in T call **Select**$(T\,.\,root, i)$.

Select(x, i)**:**

$r = x\,.\,left\,.\,size + 1$

if $i == r$

    return x

else if $i < r$

    return Select$(x\,.\,left, i)$

else

    return Select$(x\,.\,right, i - r)$

calculate the rank of x

element with ith rank must fall in the left-subtree

element with ith rank must fall in the right-subtree

Finding the Element with ith Rank

To find the element with ith rank in T call **Select**$(T\,.\,root, i)$.

Select(x, i):

$r = x\,.\,left\,.\,size + 1$

if $i == r$

  return x

else if $i < r$

  return Select$(x\,.\,left, i)$

else

  return Select$(x\,.\,right, i - r)$

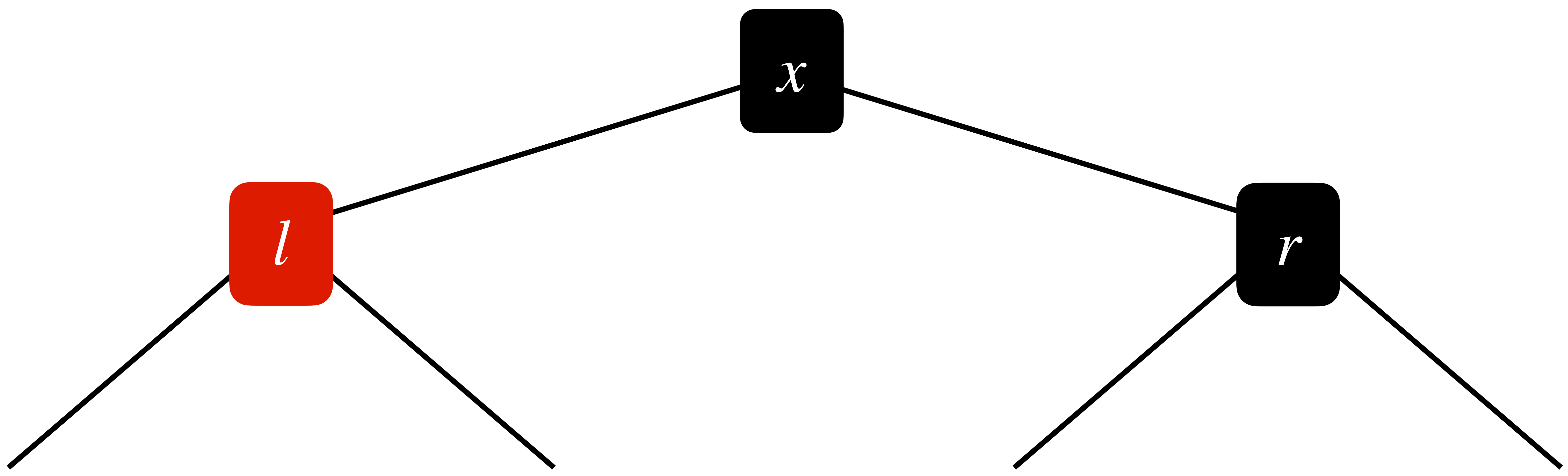
calculate the rank of x

element with ith rank must fall in the left-subtree

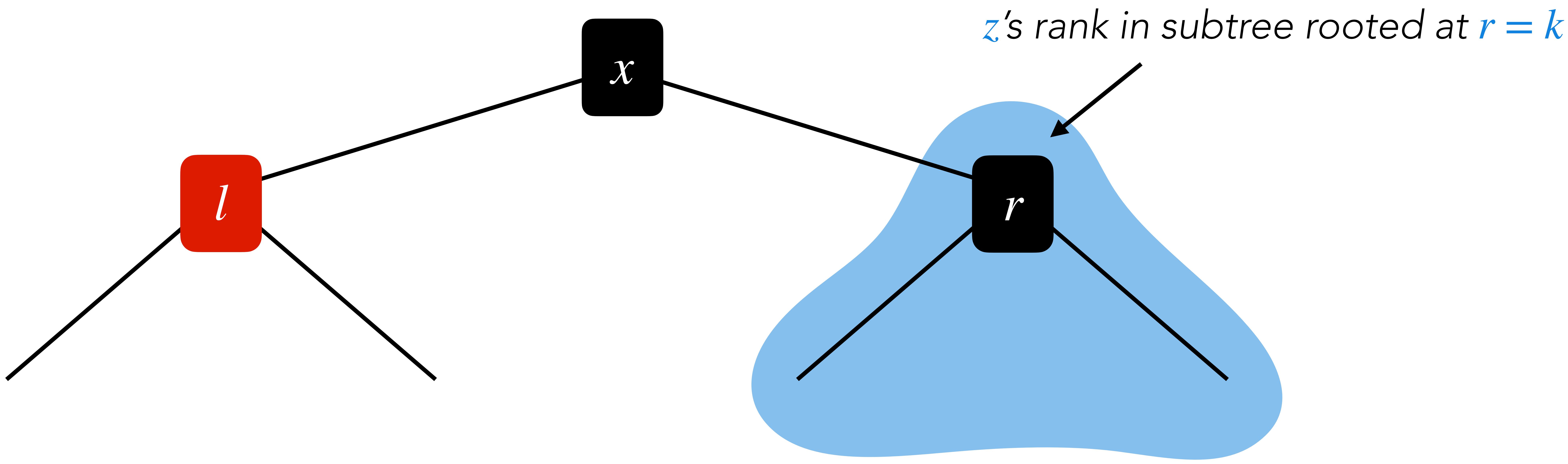
element with ith rank must fall in the right-subtree

Time Complexity: $O(h) = O(\log n)$ as with every recursive call algorithm goes one level down.

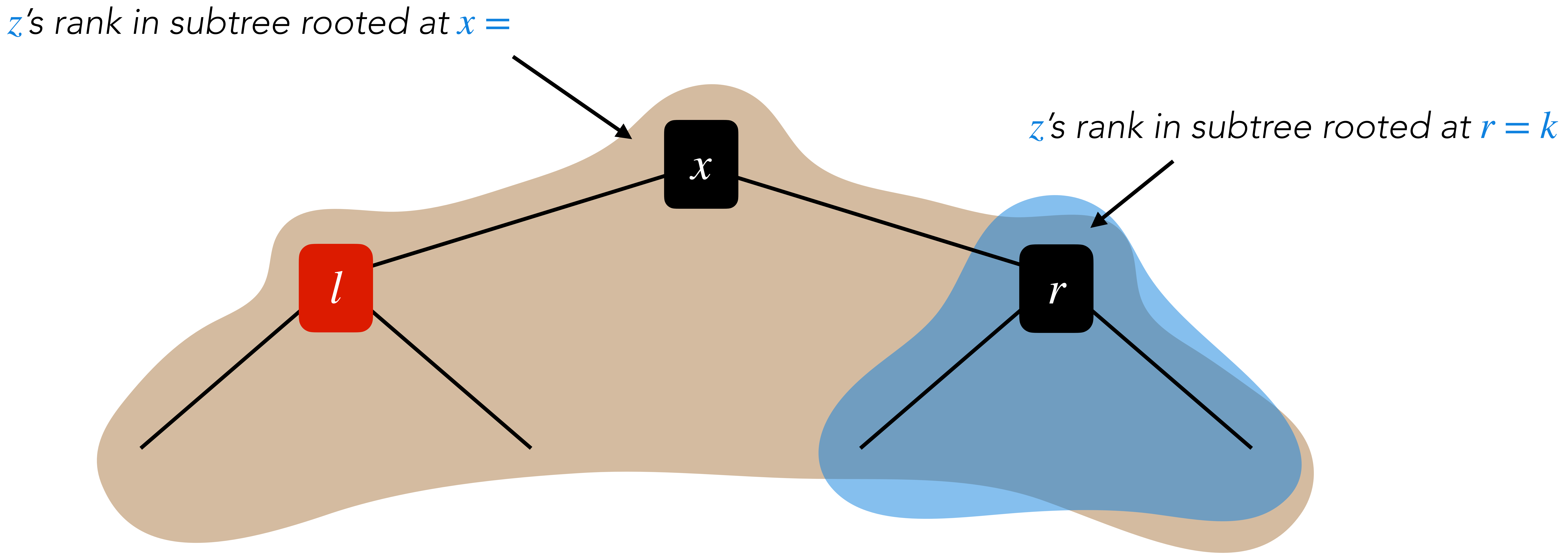
Finding The Rank of an Element



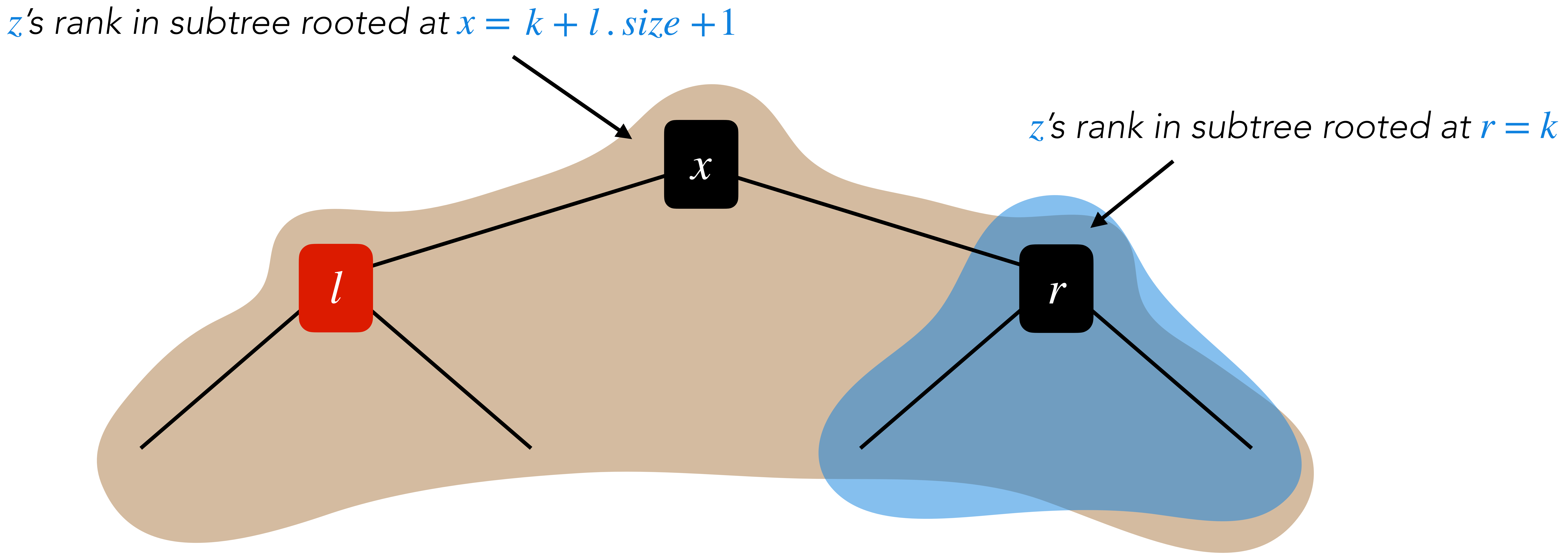
Finding The Rank of an Element



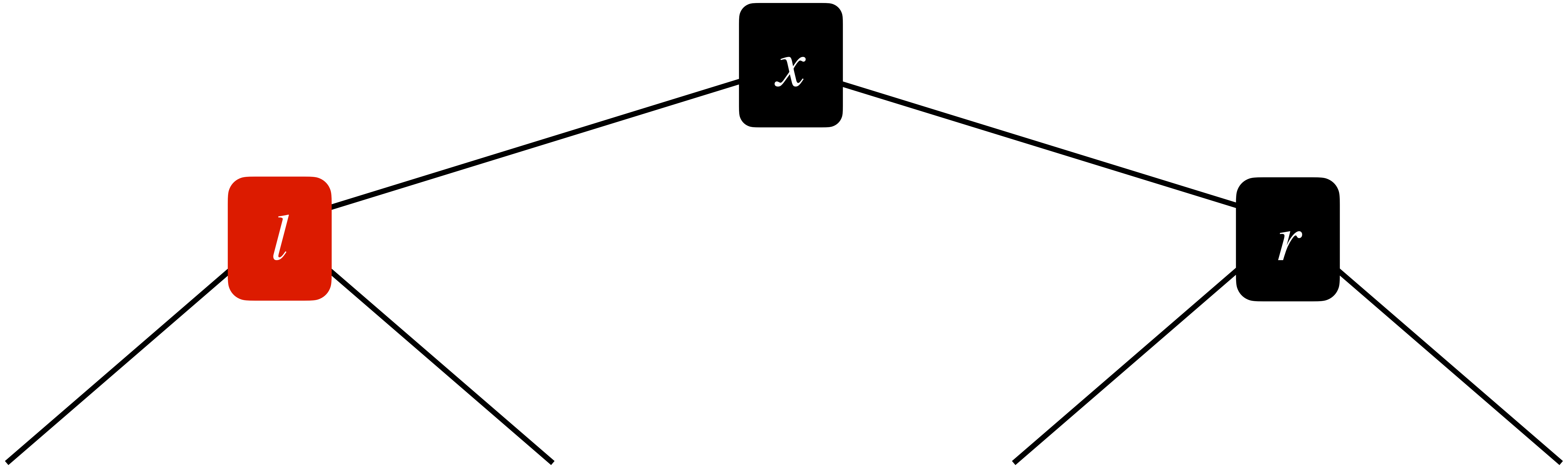
Finding The Rank of an Element



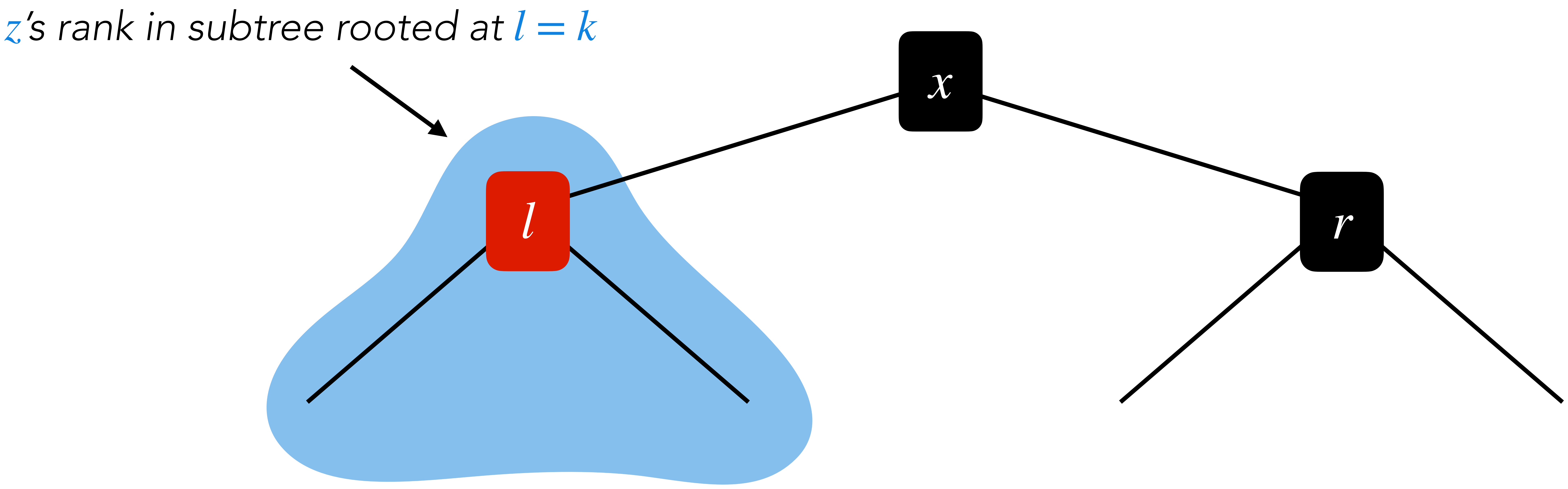
Finding The Rank of an Element



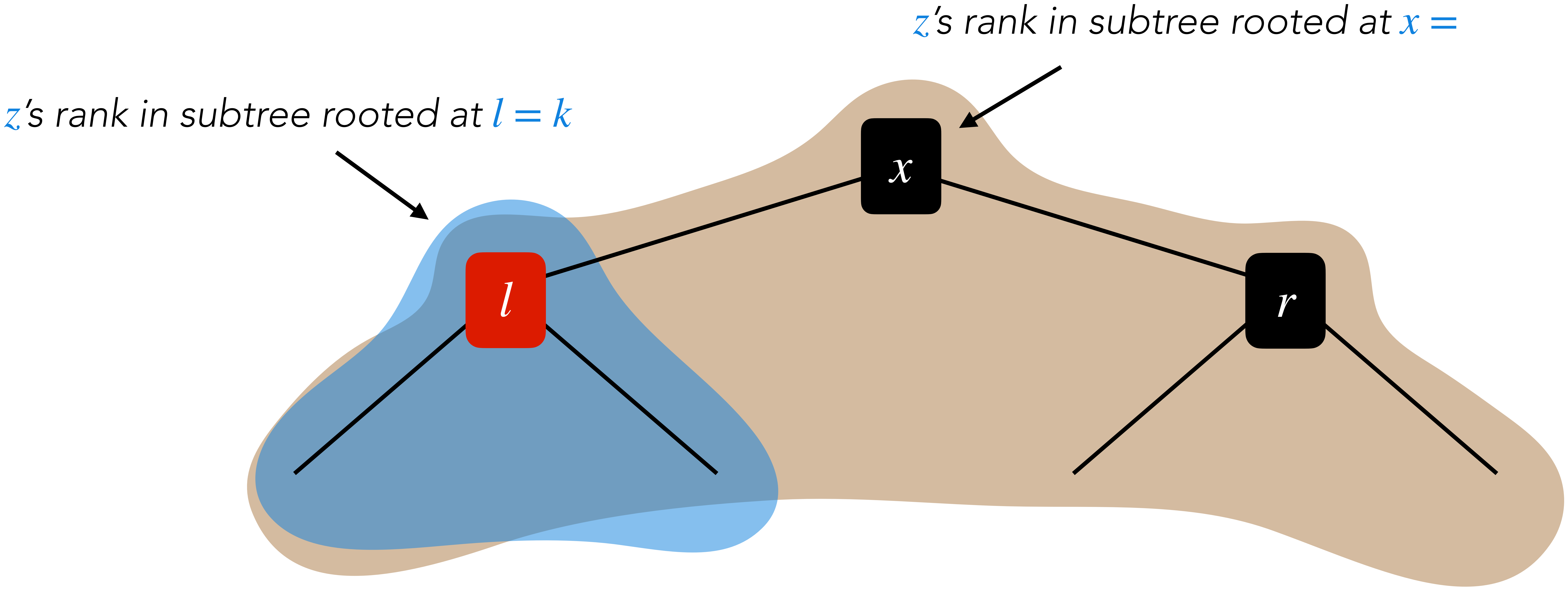
Finding The Rank of an Element



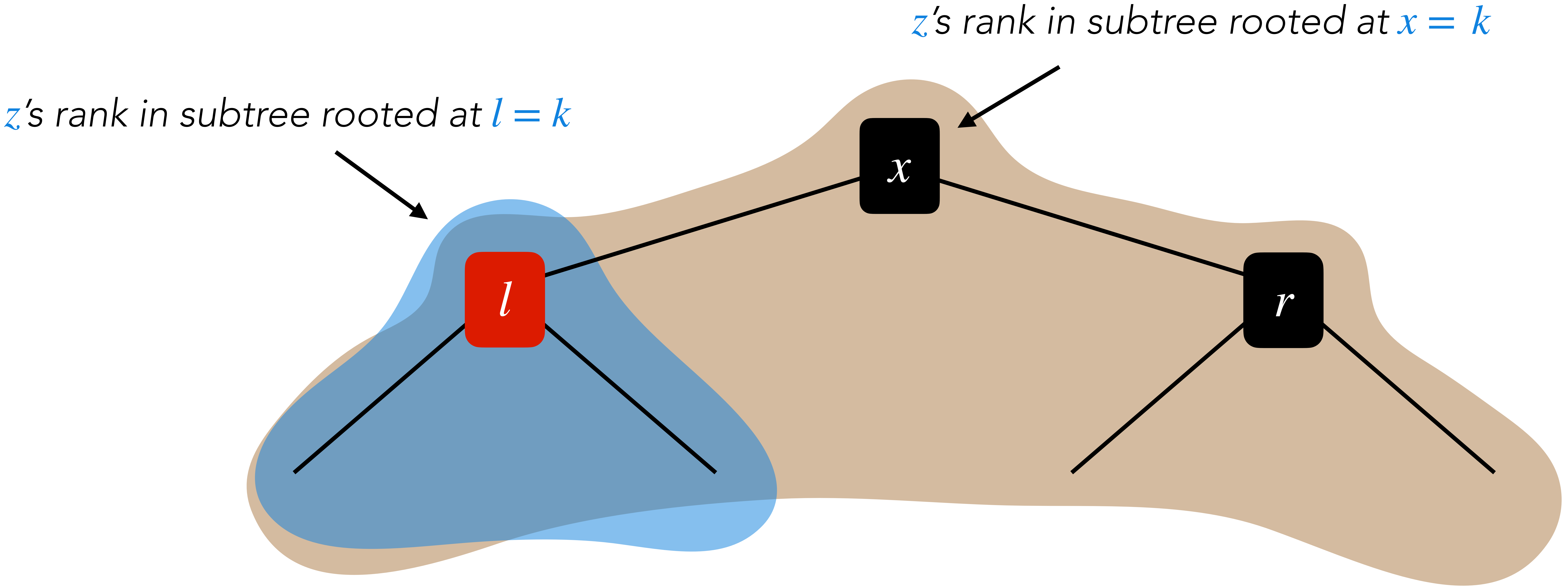
Finding The Rank of an Element



Finding The Rank of an Element



Finding The Rank of an Element



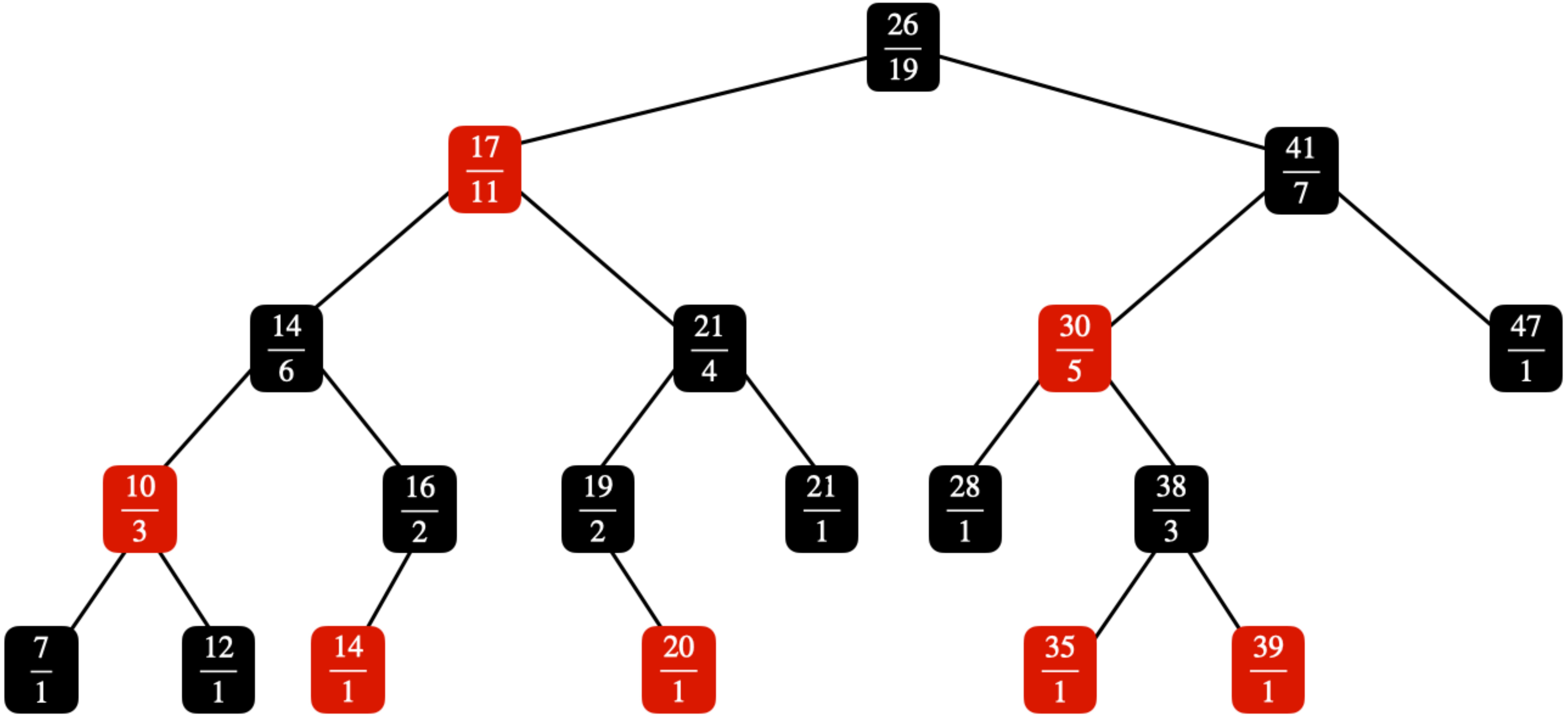
Finding The Rank of an Element

Finding The Rank of an Element

Example: Find the rank of 38 in the below **set** or R**B-tree**.

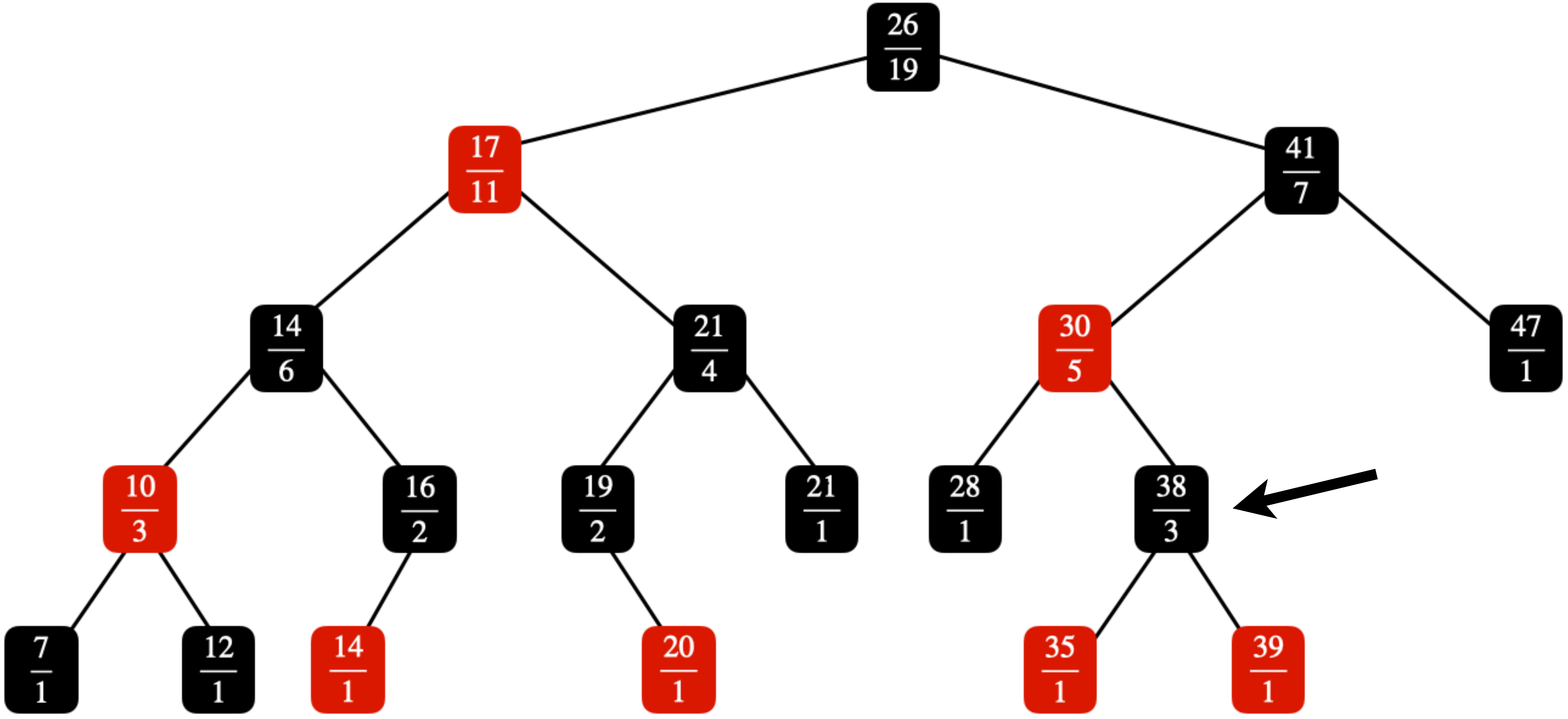
Finding The Rank of an Element

Example: Find the rank of 38 in the below **set** or RB-**tree**.

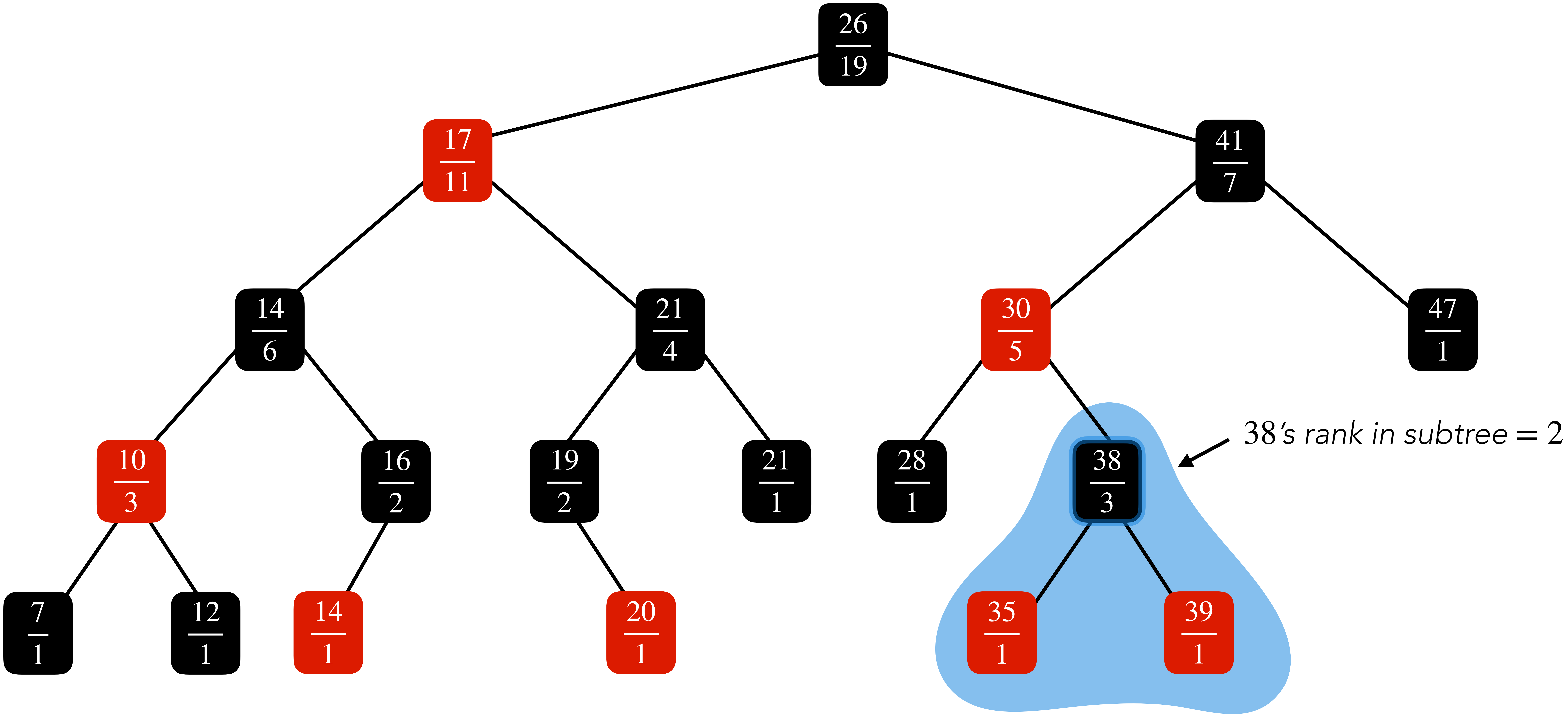


Finding The Rank of an Element

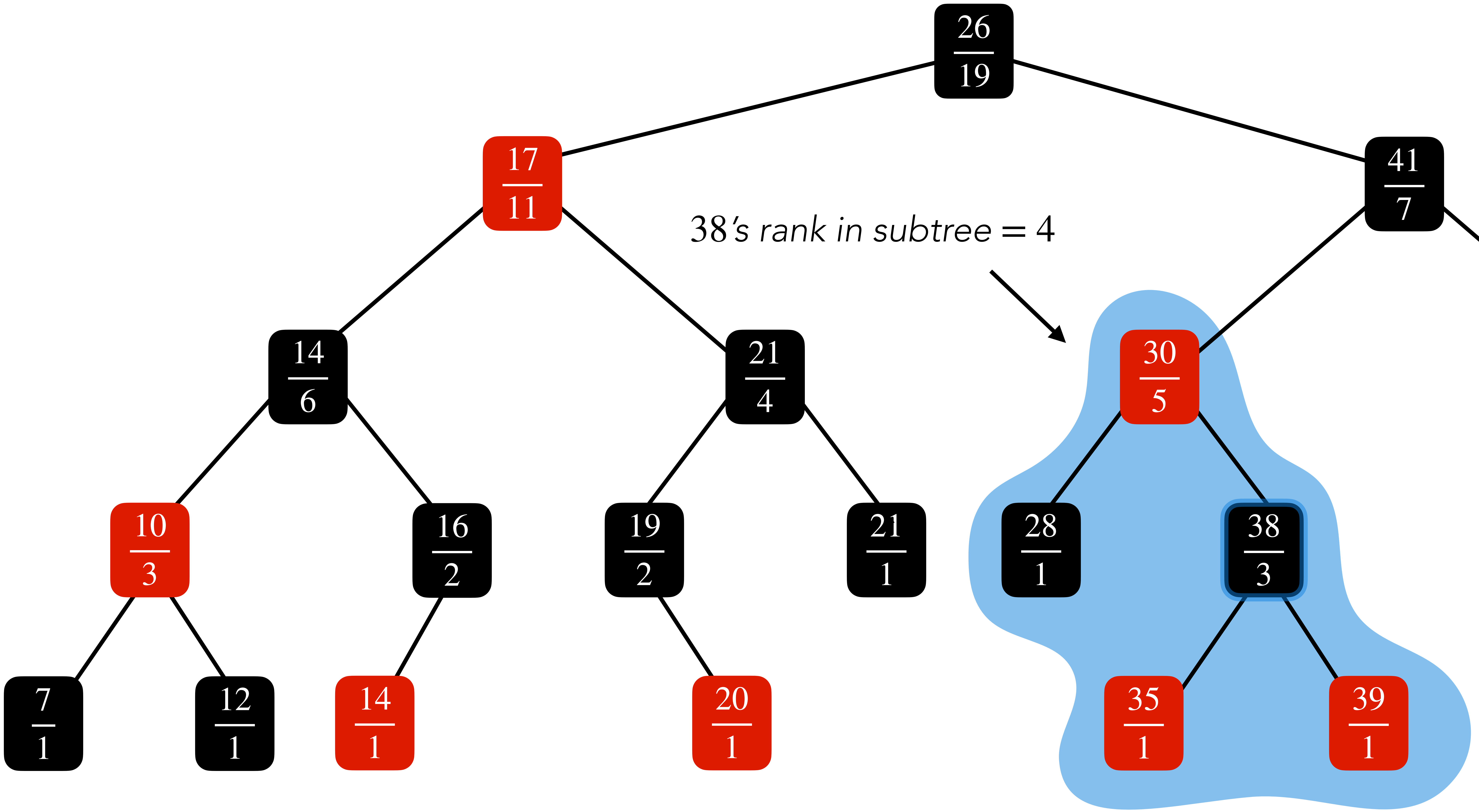
Example: Find the rank of 38 in the below **set** or RB-**tree**.



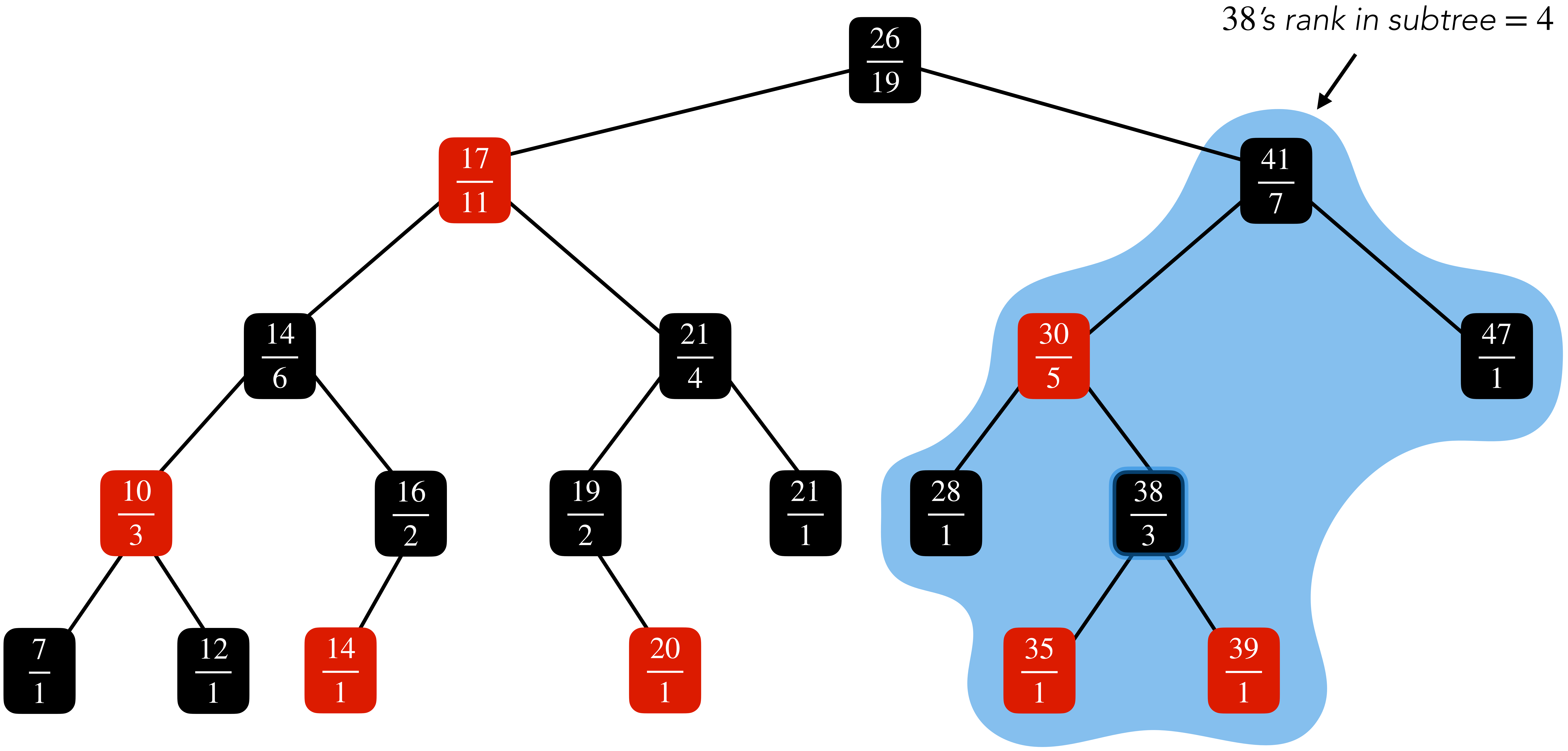
Finding The Rank of an Element



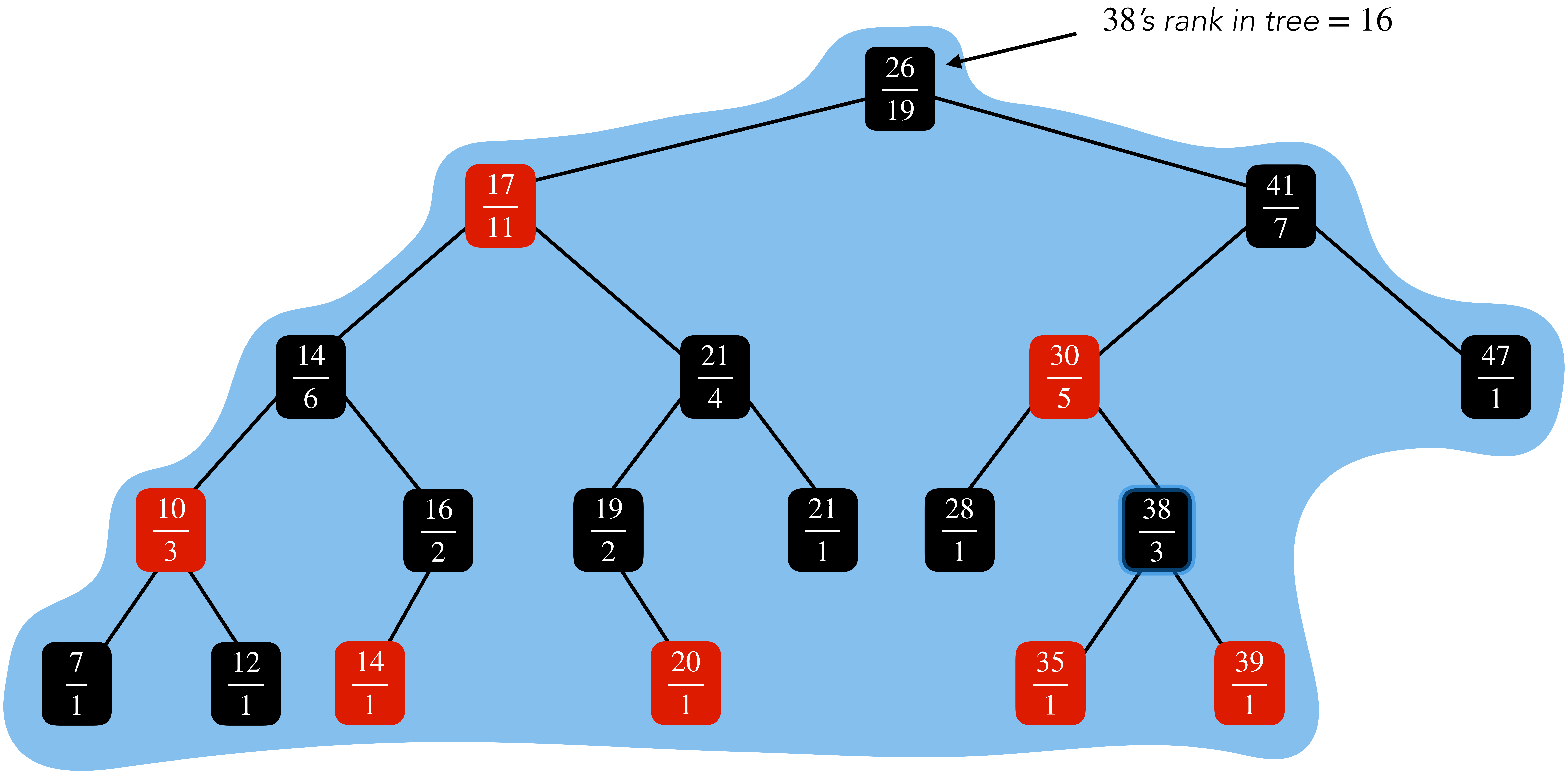
Finding The Rank of an Element



Finding The Rank of an Element



Finding The Rank of an Element



Finding The Rank of an Element

Finding The Rank of an Element

To find the rank of an element x in T call $\textbf{Rank}(T, x)$.

Finding The Rank of an Element

To find the rank of an element x in T call **Rank**(T, x).

Rank(T, x):

Finding The Rank of an Element

To find the rank of an element x in T call **Rank**(T, x).

Rank(T, x)**:**

1. $r = x\,.\,\mathit{left}\,.\,\mathit{size} + 1$

Finding The Rank of an Element

To find the rank of an element x in T call **Rank**(T, x).

Rank(T, x)**:**

1. $r = x \,.\, left \,.\, size + 1$

x's rank within the subtree rooted at x

Finding The Rank of an Element

To find the rank of an element x in T call **Rank**(T, x).

Rank(T, x):

1. $r = x\,.\,left\,.\,size + 1$
2. $y = x$

x's rank within the subtree rooted at x

Finding The Rank of an Element

To find the rank of an element x in T call **Rank**(T, x).

Rank(T, x)**:**

1. $r = x\,.\,left\,.\,size + 1$
2. $y = x$
3. **while** $y \neq T\,.\,root$

x's rank within the subtree rooted at x

Finding The Rank of an Element

To find the rank of an element x in T call **Rank**(T, x).

Rank(T, x)**:**

```
r = x.left.size + 1
y = x
while y ≠ T.root
    if y = y.p.right
```

x's rank within the subtree rooted at x

Finding The Rank of an Element

To find the rank of an element x in T call **Rank**(T, x).

Rank(T, x):

1. $r = x\,.\,left\,.\,size + 1$
2. $y = x$
3. **while** $y \neq T\,.\,root$
4. **if** $y = y\,.\,p\,.\,right$

*x's rank within the **subtree rooted at** x*

*updating x's rank when y is the **right child** of its parent.*

Finding The Rank of an Element

To find the rank of an element x in T call **Rank**(T, x).

Rank(T, x):

1. $r = x\,.\,left\,.\,size + 1$
2. $y = x$
3. **while** $y \neq T\,.\,root$
4.   **if** $y = y\,.\,p\,.\,right$
5. $r =$ ________________

*x's rank within the **subtree rooted at** x*

*updating x's rank when y is the **right child** of its parent.*

Finding The Rank of an Element

To find the rank of an element x in T call **Rank**(T, x).

Rank(T, x):

1. $r = x\,.\,left\,.\,size + 1$
2. $y = x$
3. **while** $y \neq T\,.\,root$
4. **if** $y = y\,.\,p\,.\,right$
5. $r =$ ______________________
6. $y = y\,.\,p$

*x's rank within the **subtree rooted at** x*

*updating x's rank when y is the **right child** of its parent.*

Finding The Rank of an Element

To find the rank of an element x in T call **Rank**(T, x).

Rank(T, x):

1. $r = x\,.\,left\,.\,size + 1$
2. $y = x$
3. **while** $y \neq T\,.\,root$
4.   **if** $y = y\,.\,p\,.\,right$
5. $r =$ ________________
6. $y = y\,.\,p$
7. **return** r

x's rank within the ***subtree rooted at*** *x*

updating x's rank when y is the ***right child*** *of its parent.*

Finding The Rank of an Element

To find the rank of an element x in T call **Rank**(T, x).

Rank(T, x):

```
r = x.left.size + 1
y = x
while y ≠ T.root
    if y = y.p.right
        r = r + y.p.left.size + 1
    y = y.p
return r
```

x's rank within the ***subtree rooted at*** *x*

updating x's rank when y is the ***right child*** *of its parent.*

Finding The Rank of an Element

To find the rank of an element x in T call **Rank**(T, x).

Rank(T, x):

$r = x\,.\,left\,.\,size + 1$

$y = x$

while $y \neq T\,.\,root$

  if $y = y\,.\,p\,.\,right$

    $r = r + y\,.\,p\,.\,left\,.\,size + 1$

  $y = y\,.\,p$

return r

x's rank within the subtree rooted at x

updating x's rank when y is the right child of its parent.

Time Complexity: $O(h) = O(\log n)$ as with every iteration of while loop, y moves closer to root.

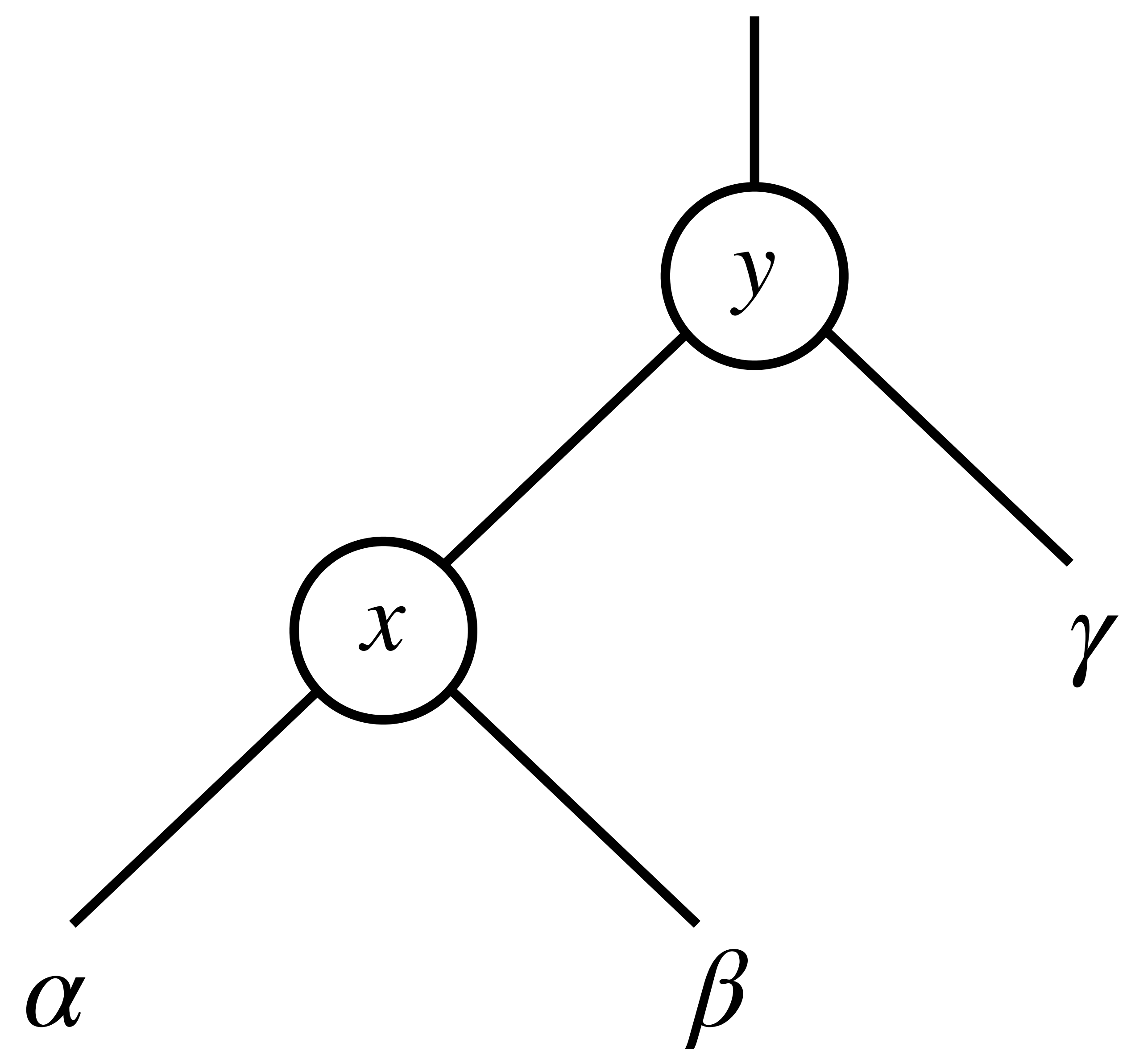
Maintaining Subtree Sizes: Rotations

Maintaining Subtree Sizes: Rotations

Maintaining subtree sizes during rotations is easy.

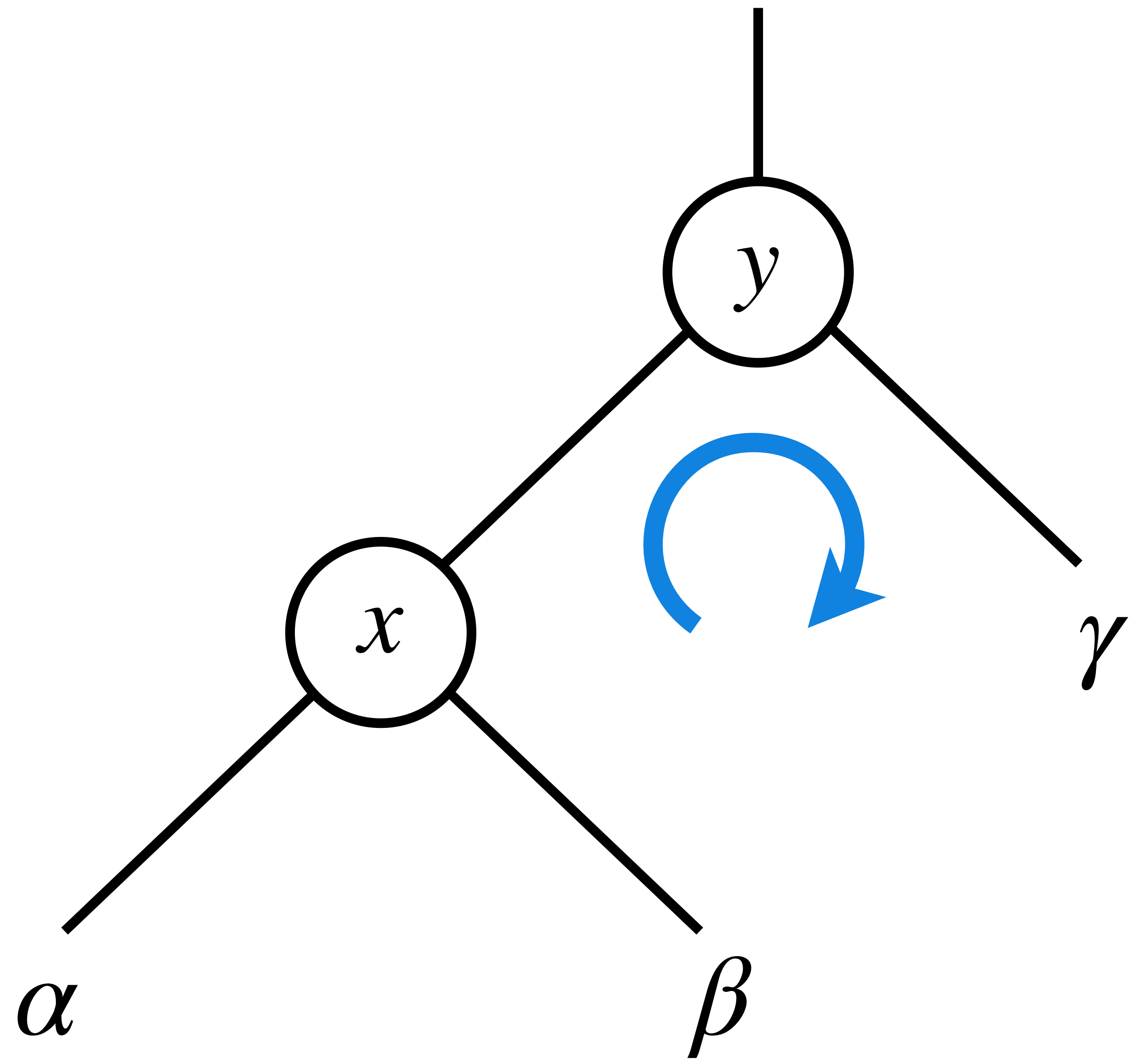
Maintaining Subtree Sizes: Rotations

Maintaining subtree sizes during rotations is easy.



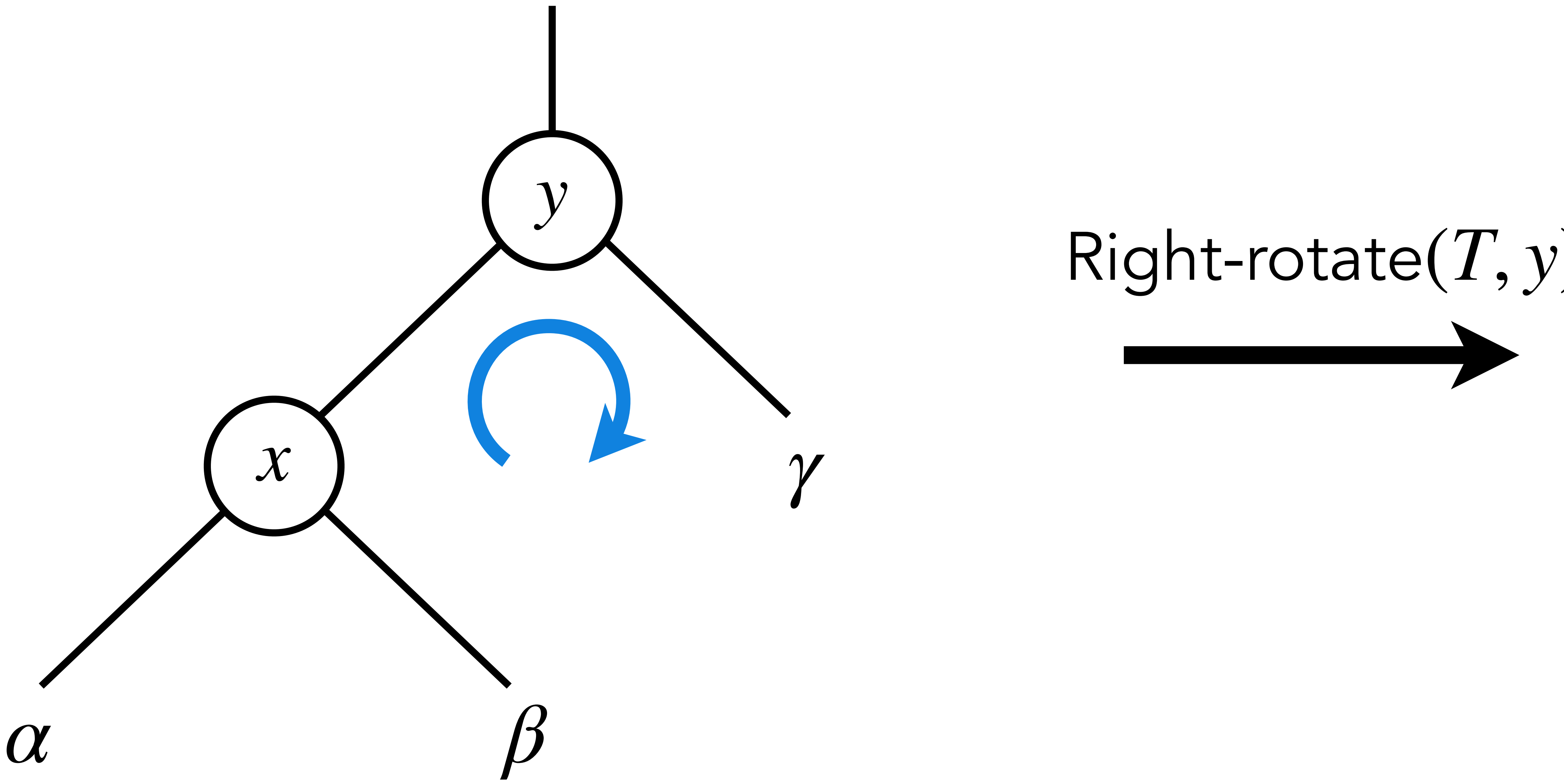
Maintaining Subtree Sizes: Rotations

Maintaining subtree sizes during rotations is easy.



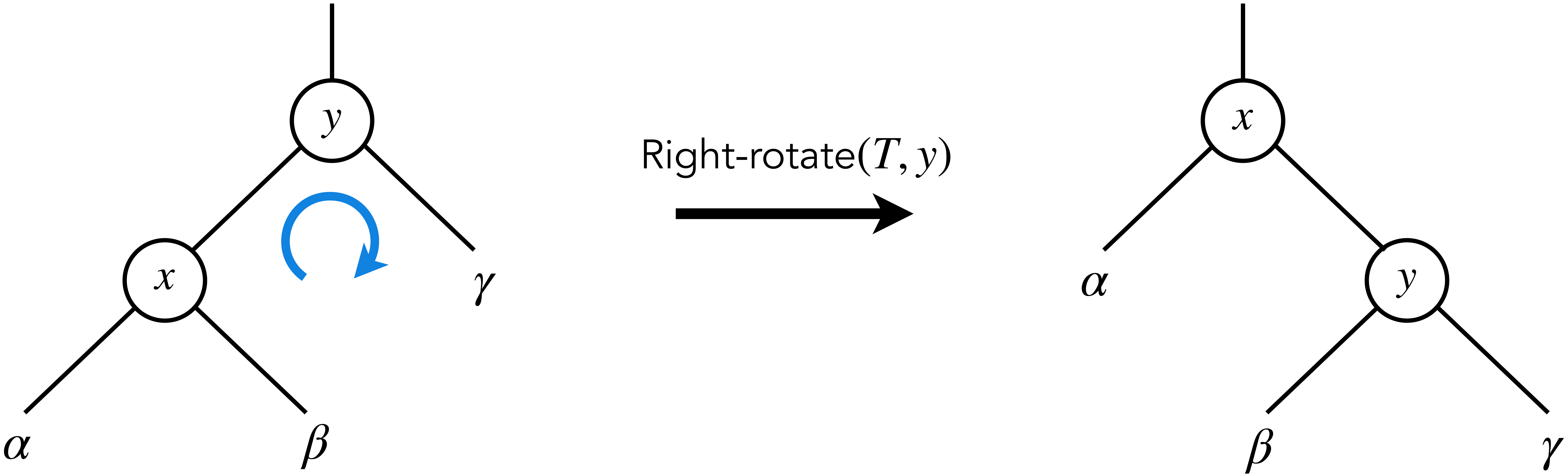
Maintaining Subtree Sizes: Rotations

Maintaining subtree sizes during rotations is easy.



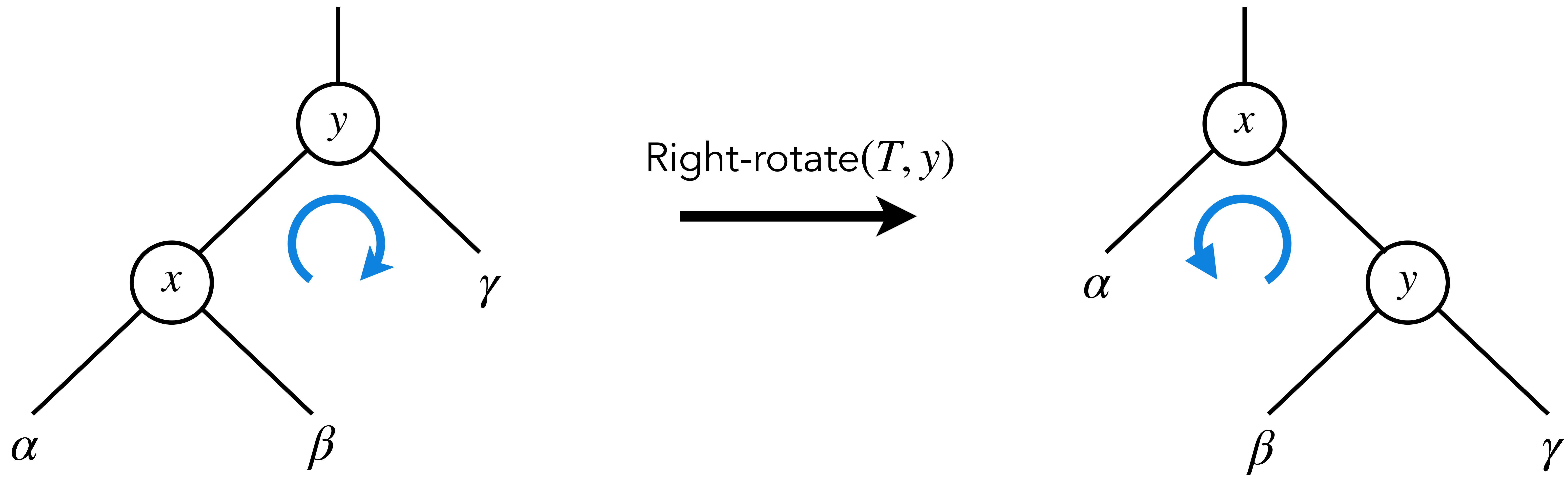
Maintaining Subtree Sizes: Rotations

Maintaining subtree sizes during rotations is easy.



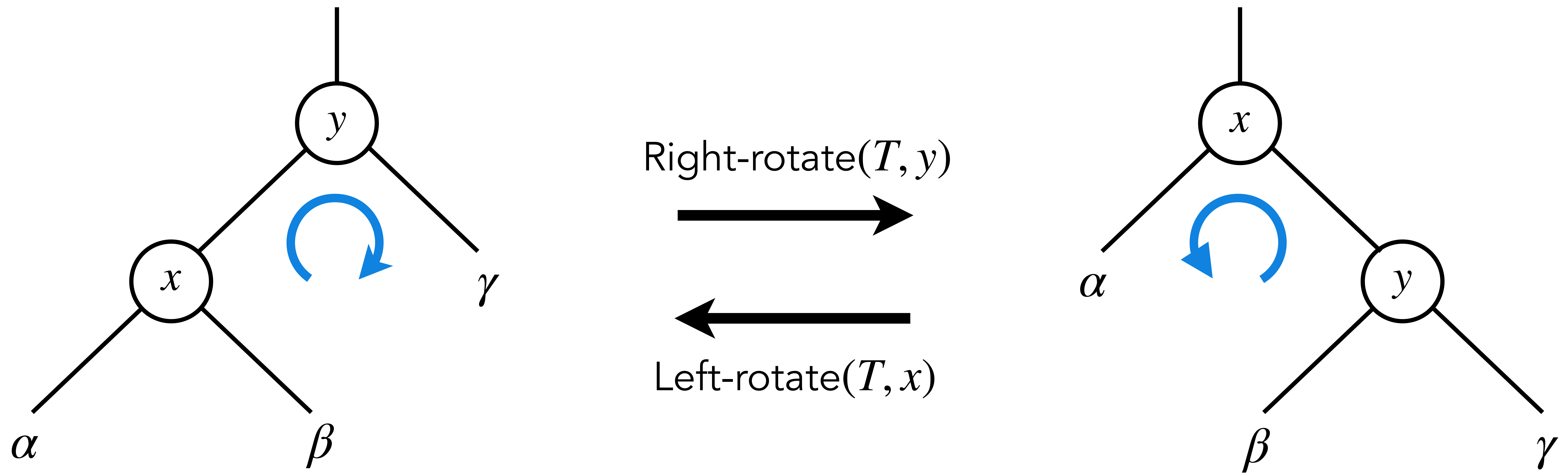
Maintaining Subtree Sizes: Rotations

Maintaining subtree sizes during rotations is easy.



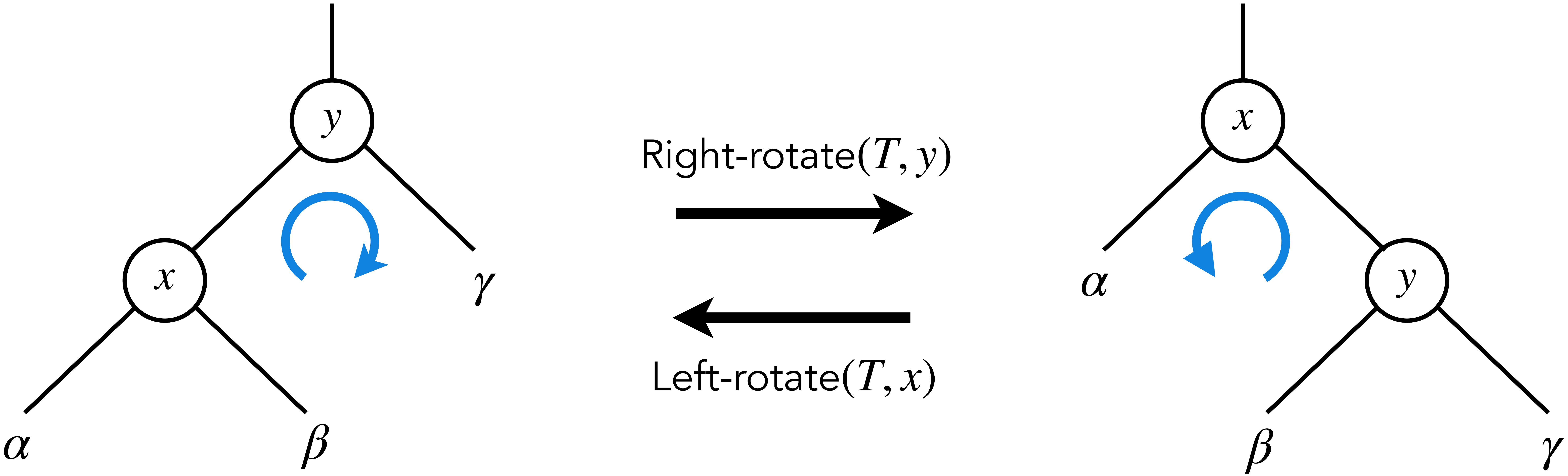
Maintaining Subtree Sizes: Rotations

Maintaining subtree sizes during rotations is easy.



Maintaining Subtree Sizes: Rotations

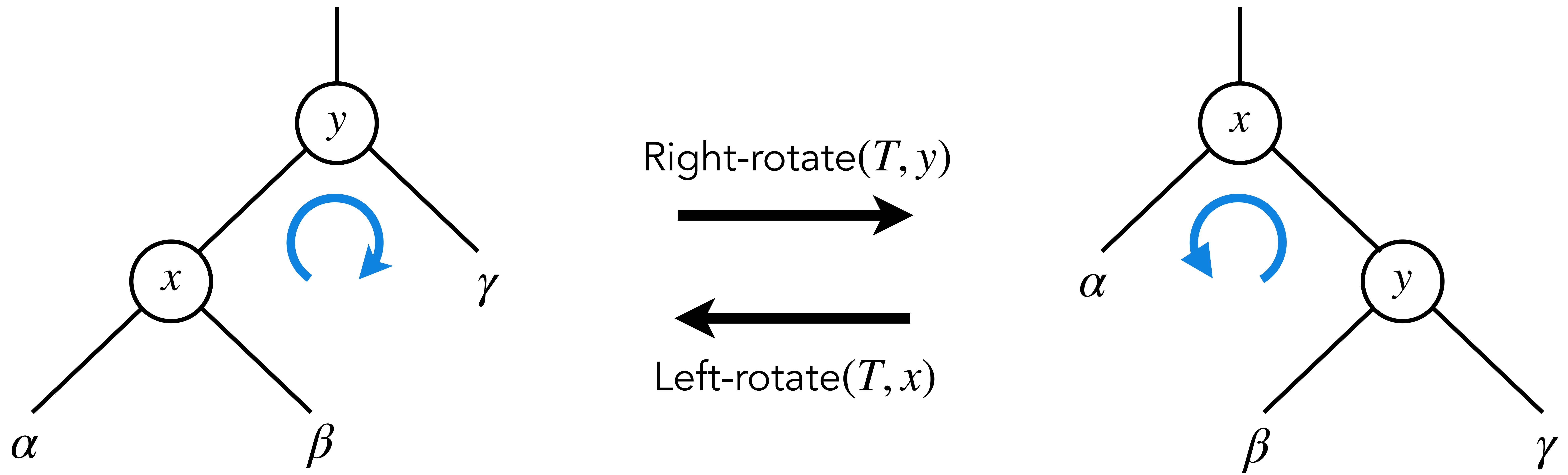
Maintaining subtree sizes during rotations is easy.



After **Left-rotate**(T, x), do the following:

Maintaining Subtree Sizes: Rotations

Maintaining subtree sizes during rotations is easy.

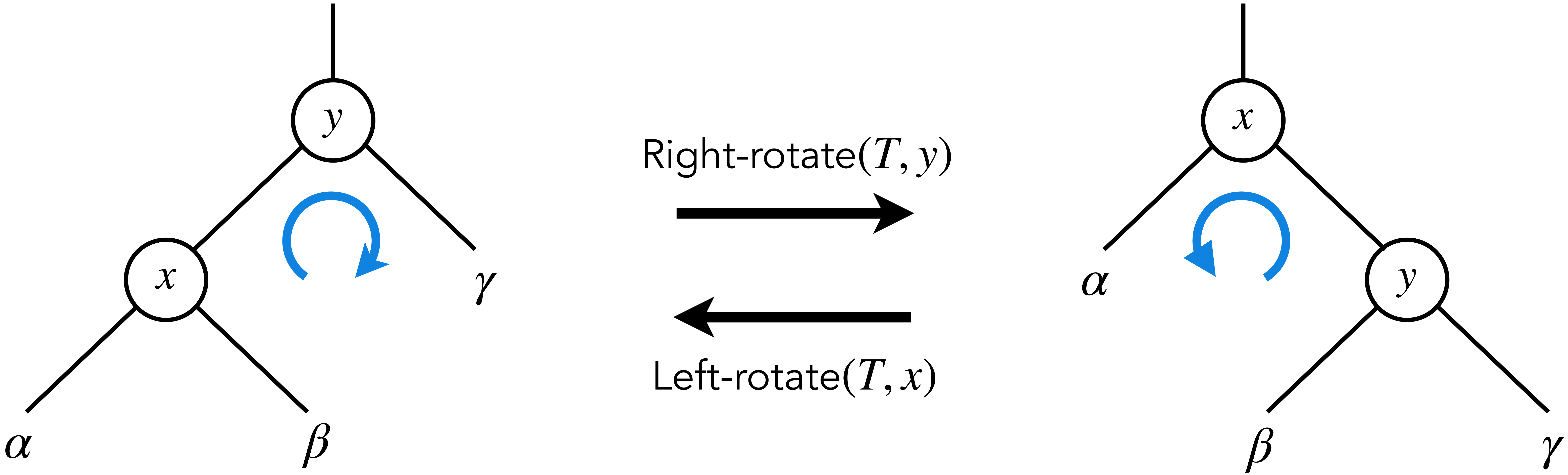


After **Left-rotate**(T, x), do the following:

1. $y\,.\,size =$

Maintaining Subtree Sizes: Rotations

Maintaining subtree sizes during rotations is easy.

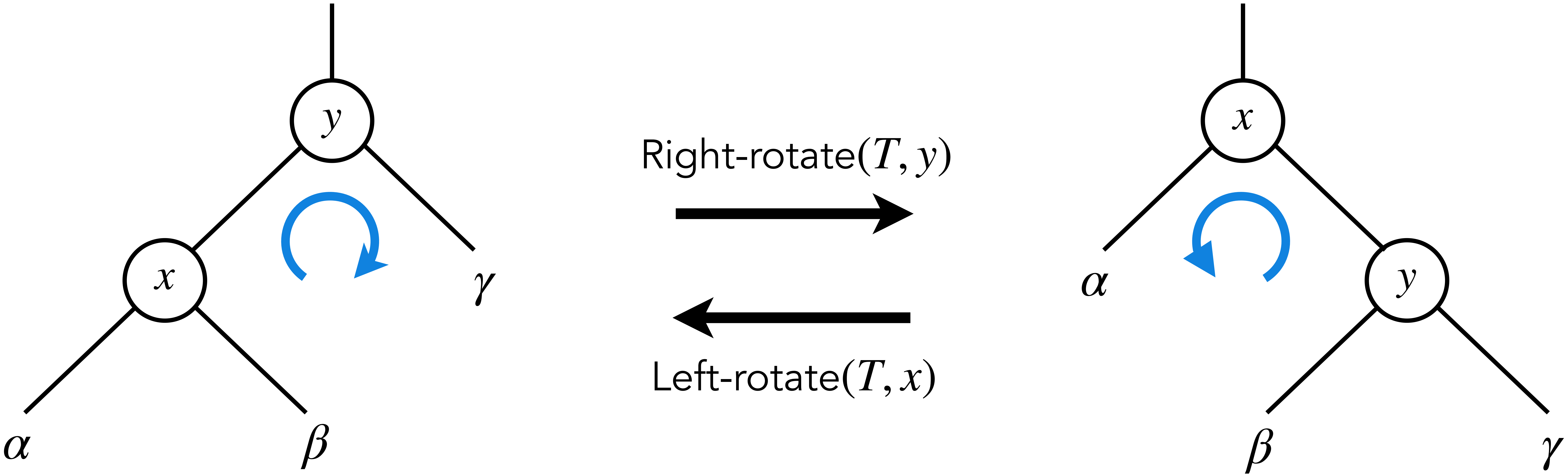


After **Left-rotate**(T, x), do the following:

1. $y\,.\,size =$
2. $x\,.\,size =$

Maintaining Subtree Sizes: Rotations

Maintaining subtree sizes during rotations is easy.

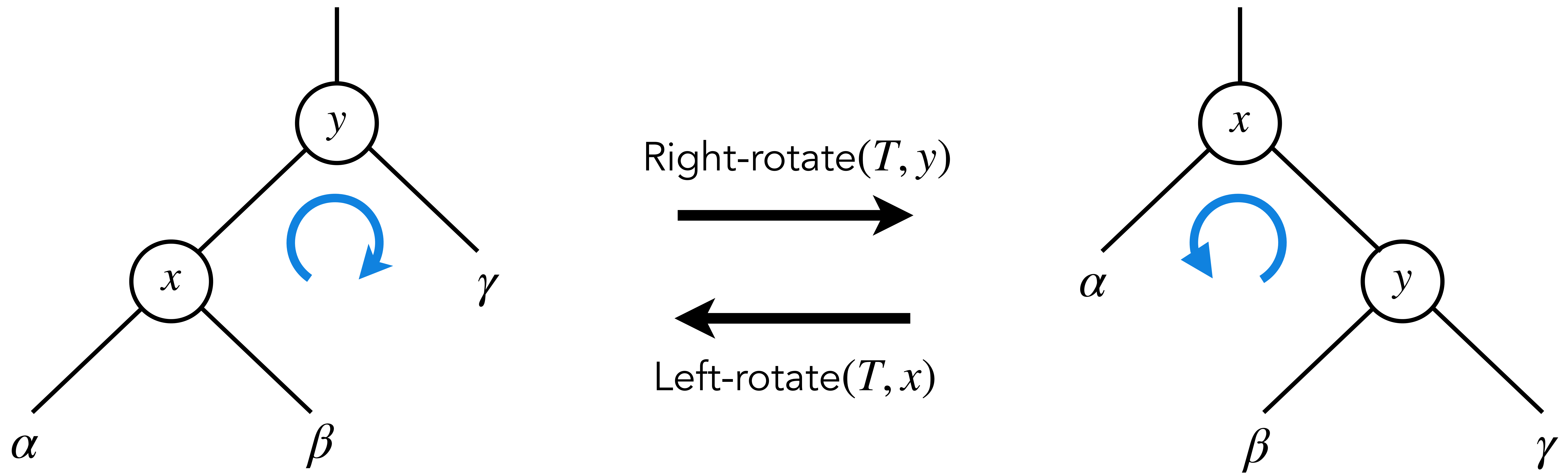


After **Left-rotate**(T, x), do the following:

1. $y\,.\,size = x\,.\,size$
2. $x\,.\,size =$

Maintaining Subtree Sizes: Rotations

Maintaining subtree sizes during rotations is easy.



After **Left-rotate**(T, x), do the following:

1. $y\,.\,size = x\,.\,size$
2. $x\,.\,size = x\,.\,left\,.\,size + x\,.\,right\,.\,size + 1$