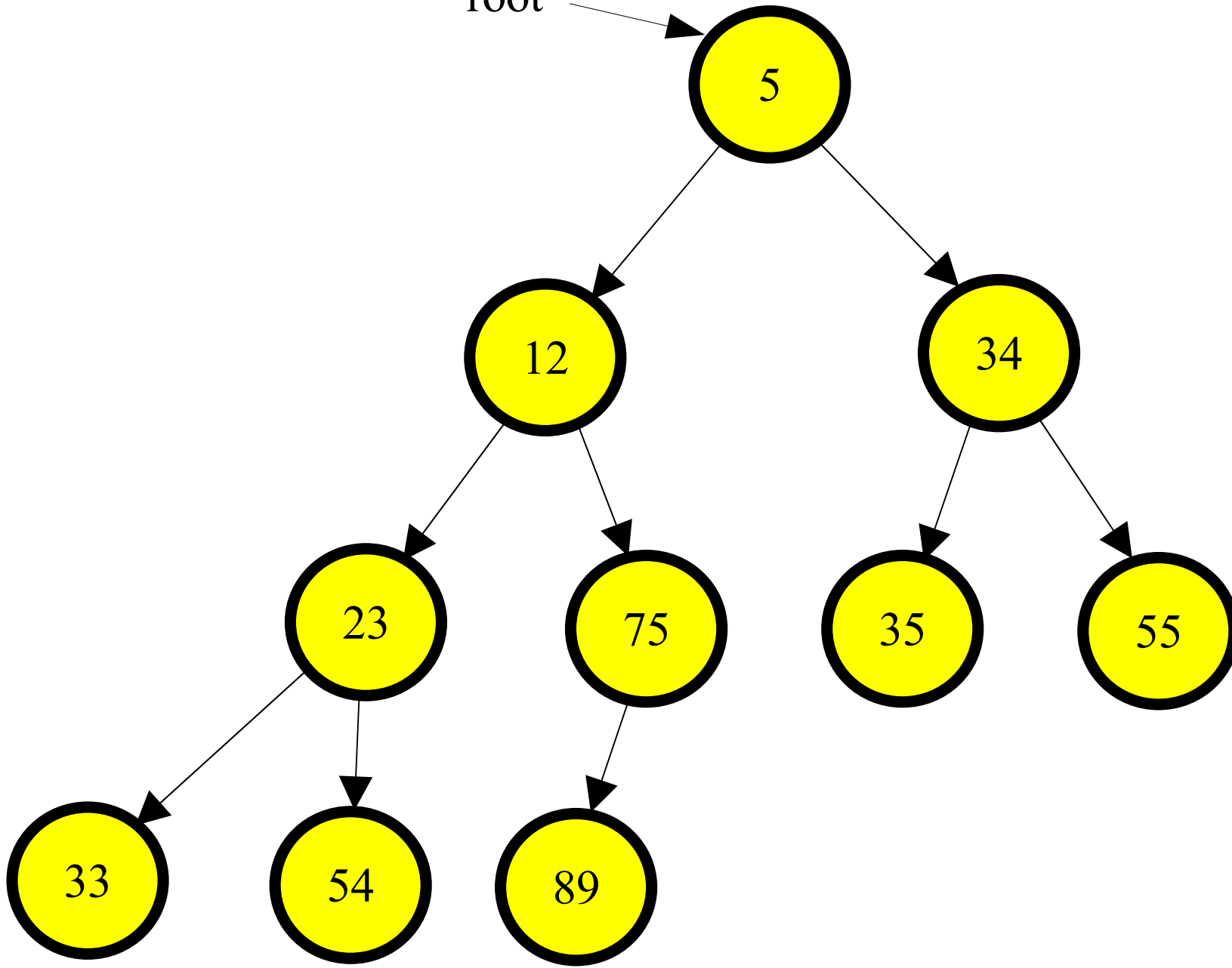
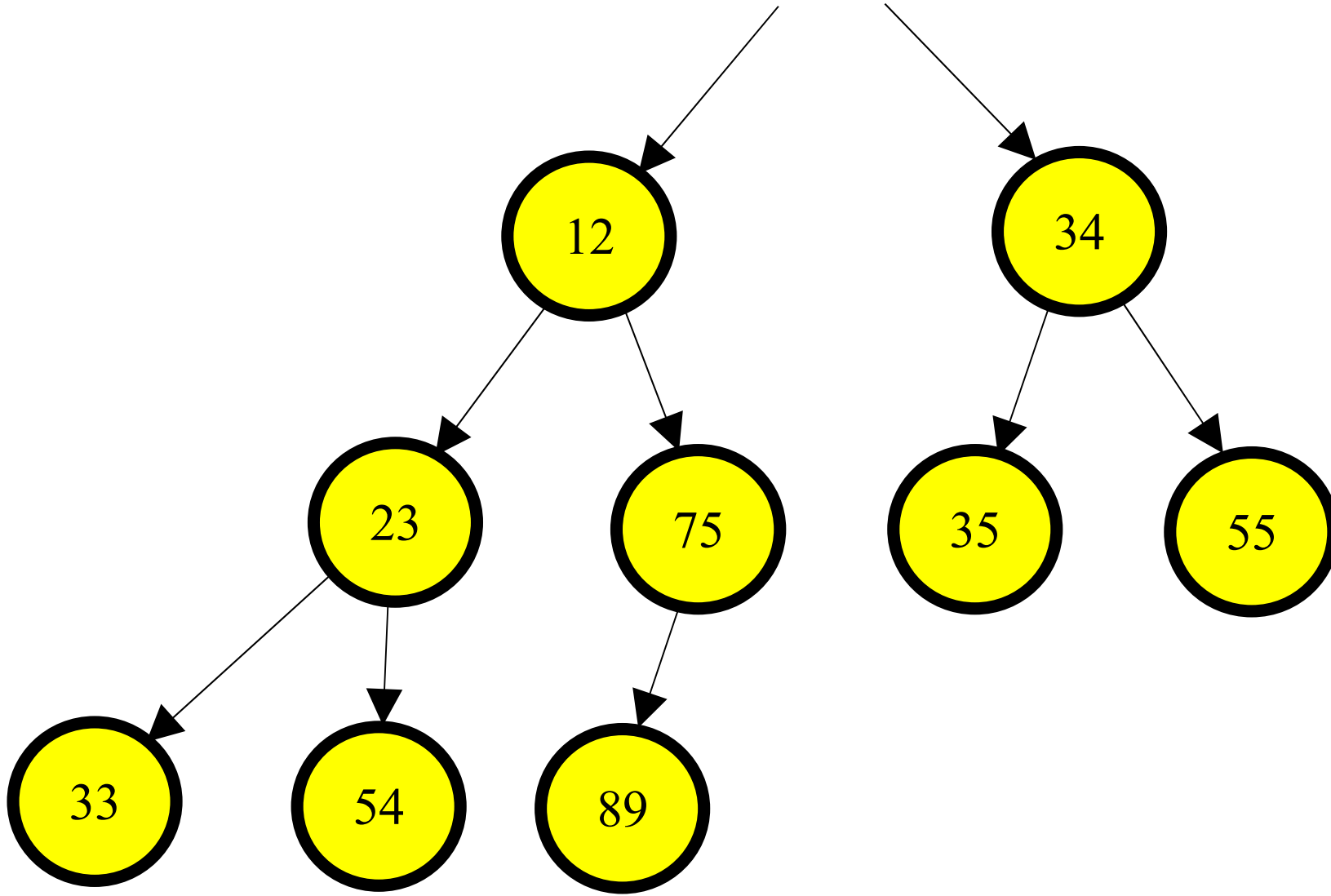
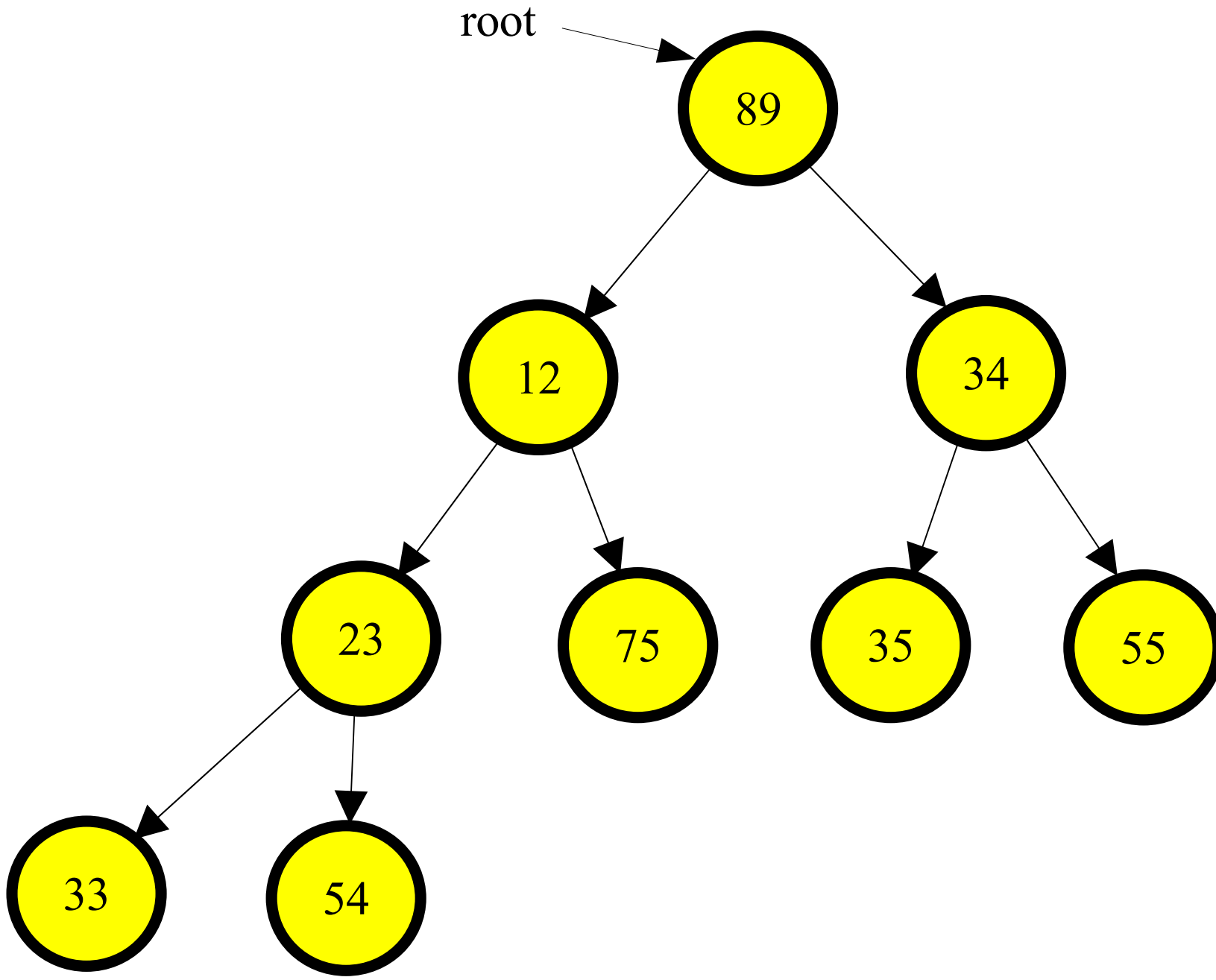


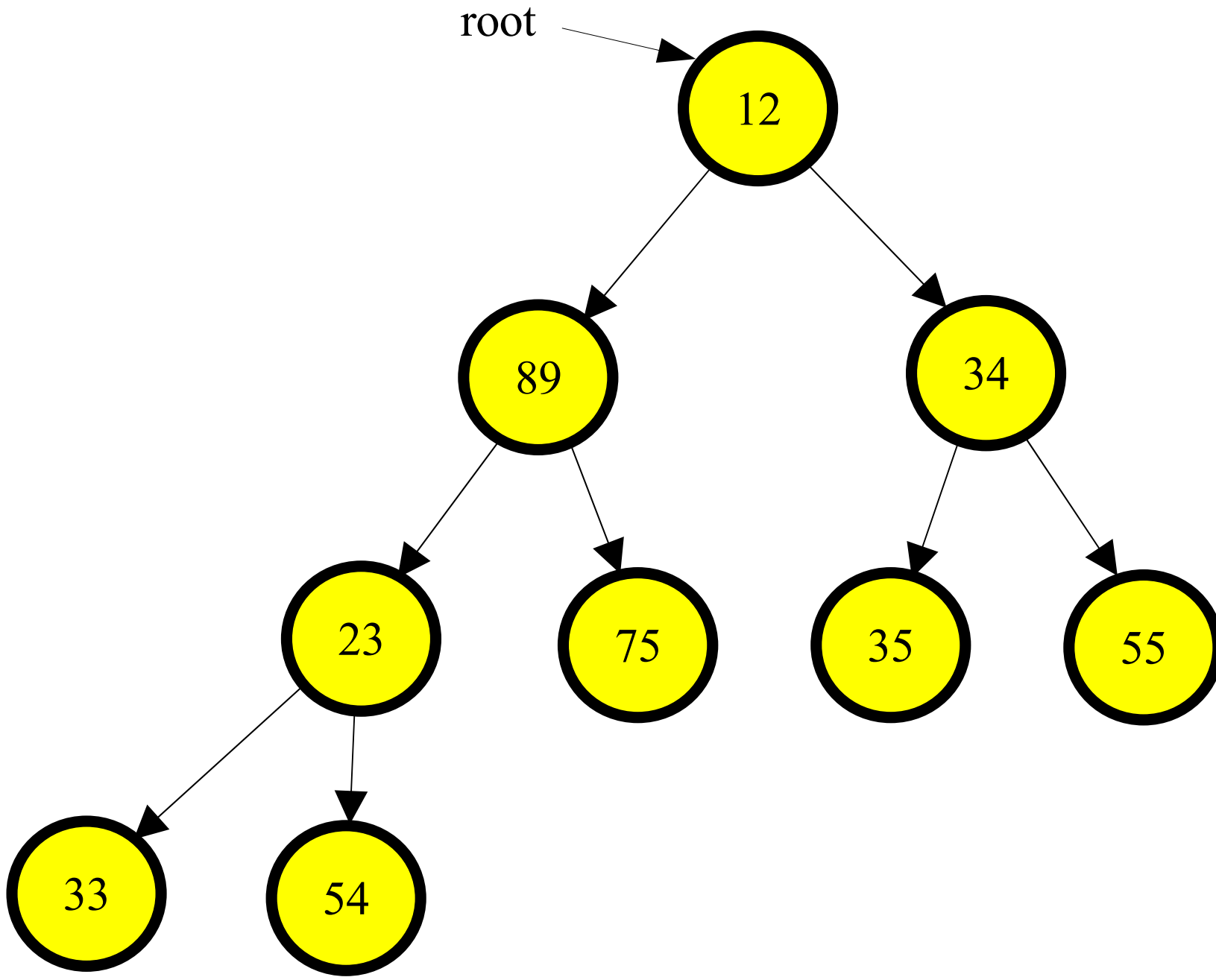
root

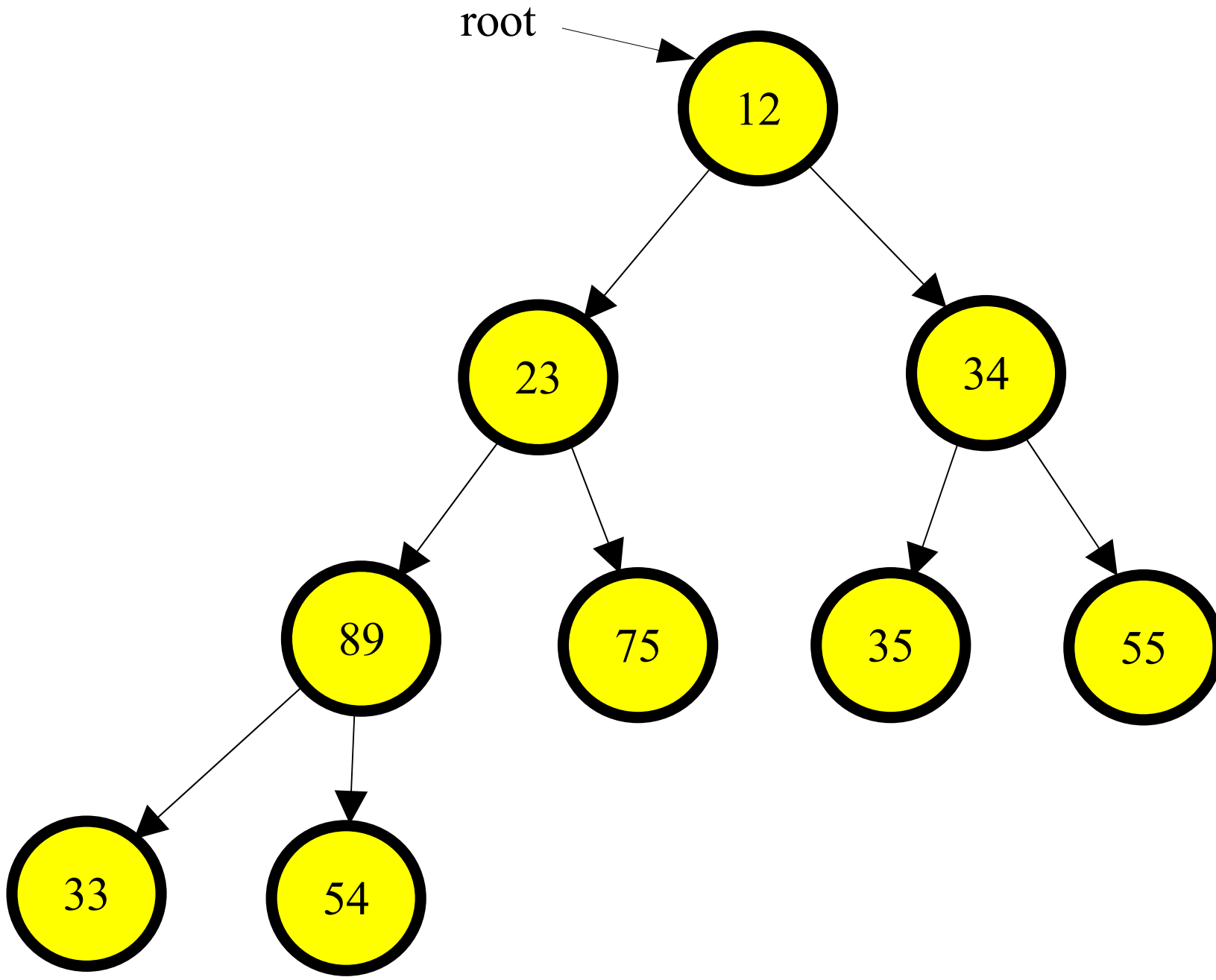


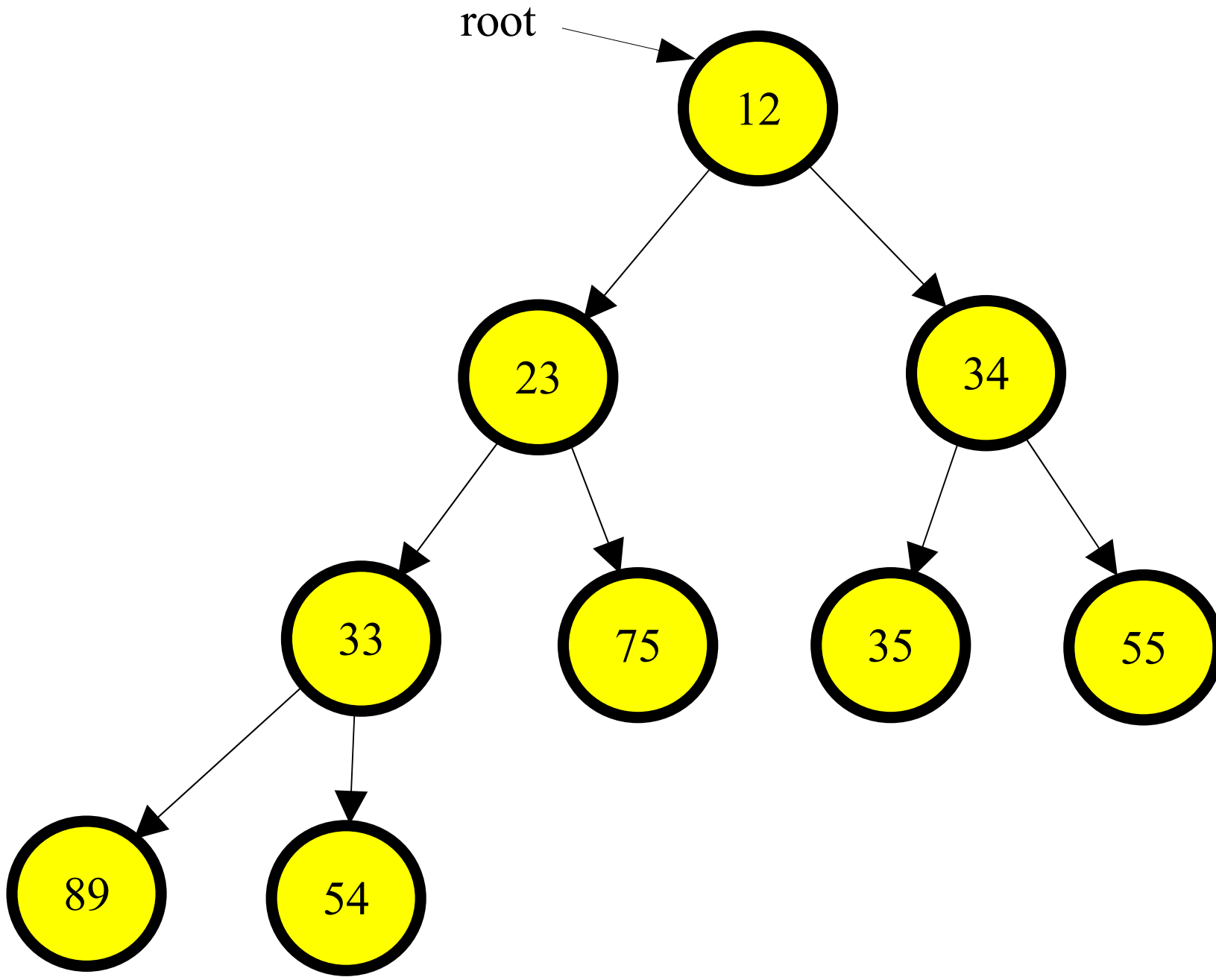
root →

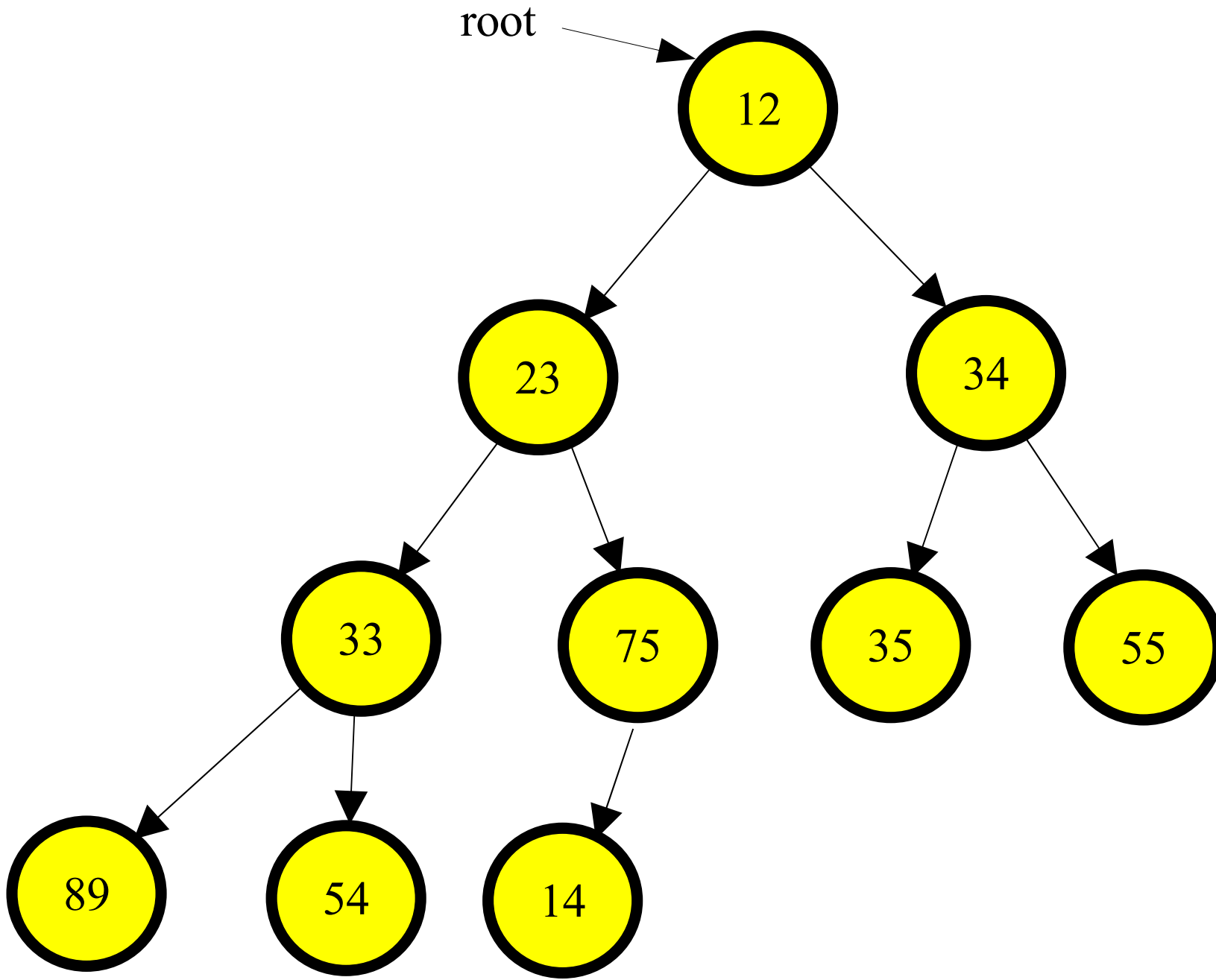


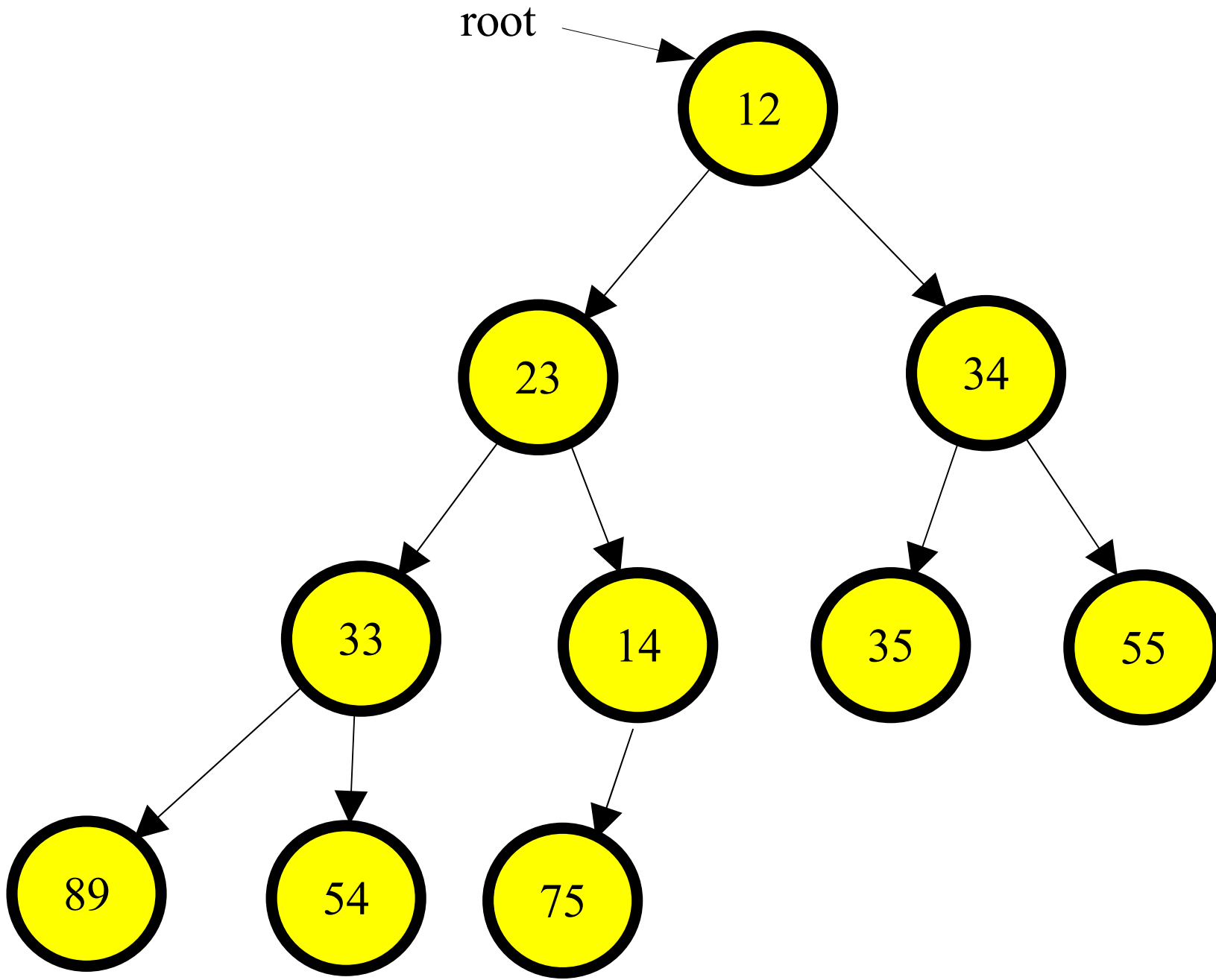




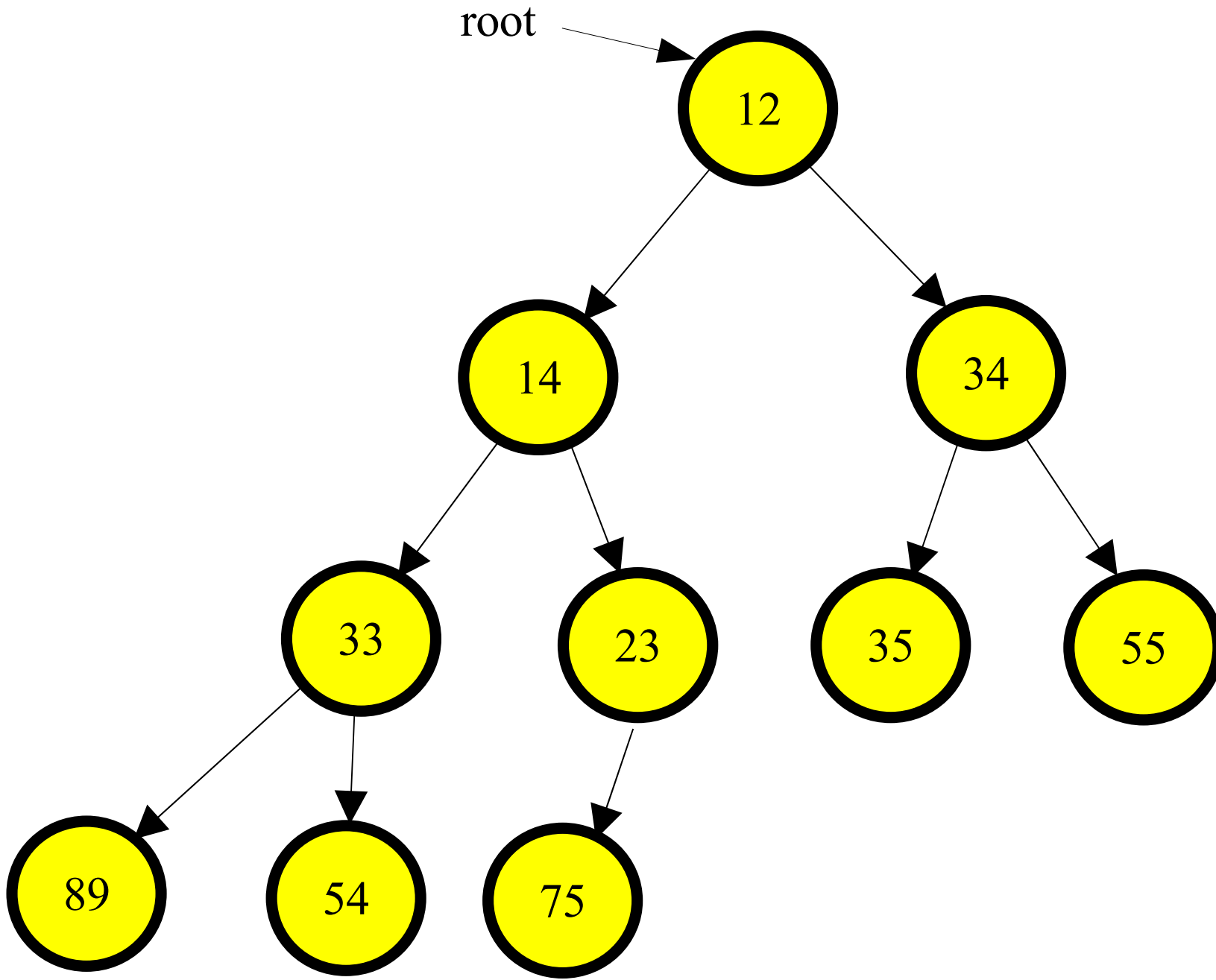






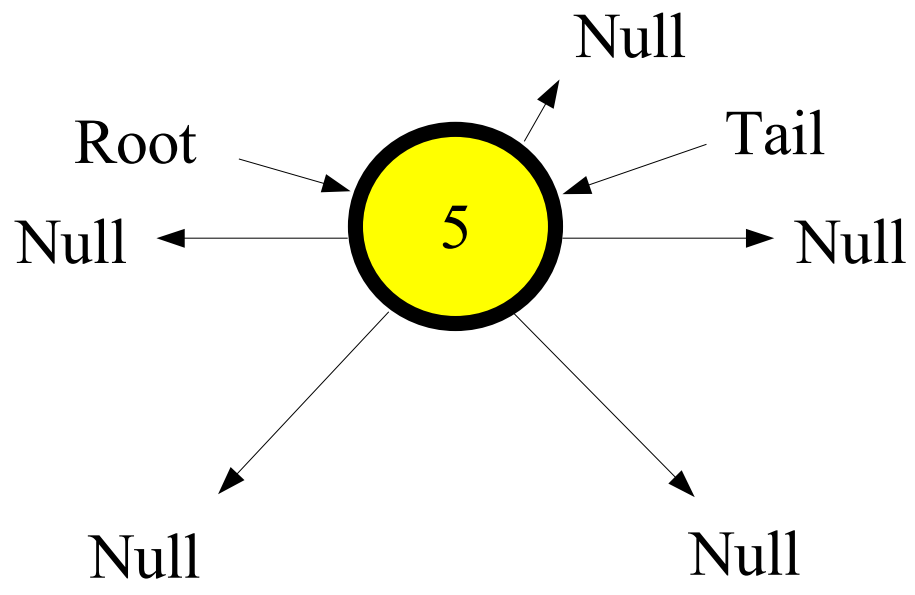


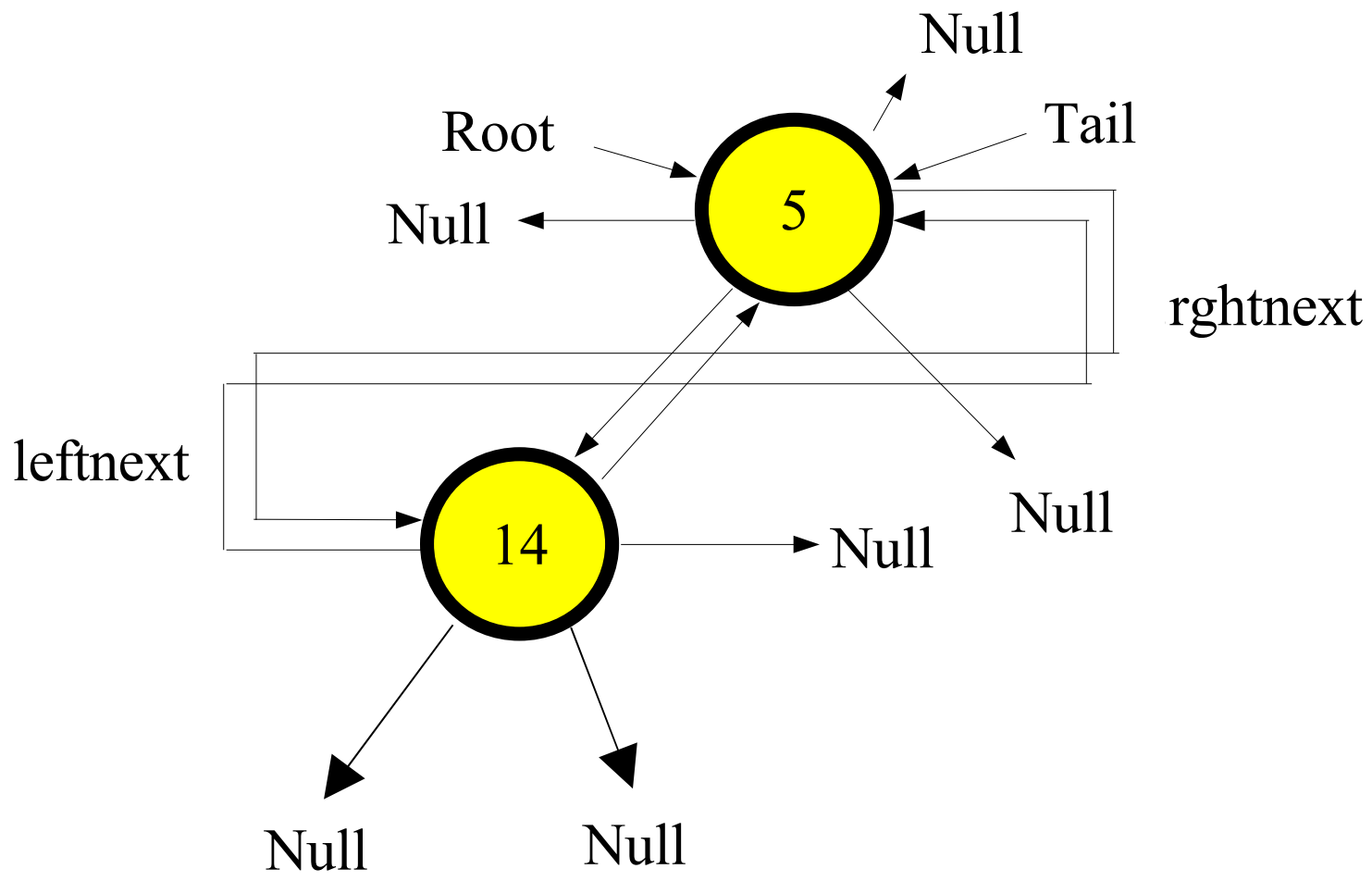


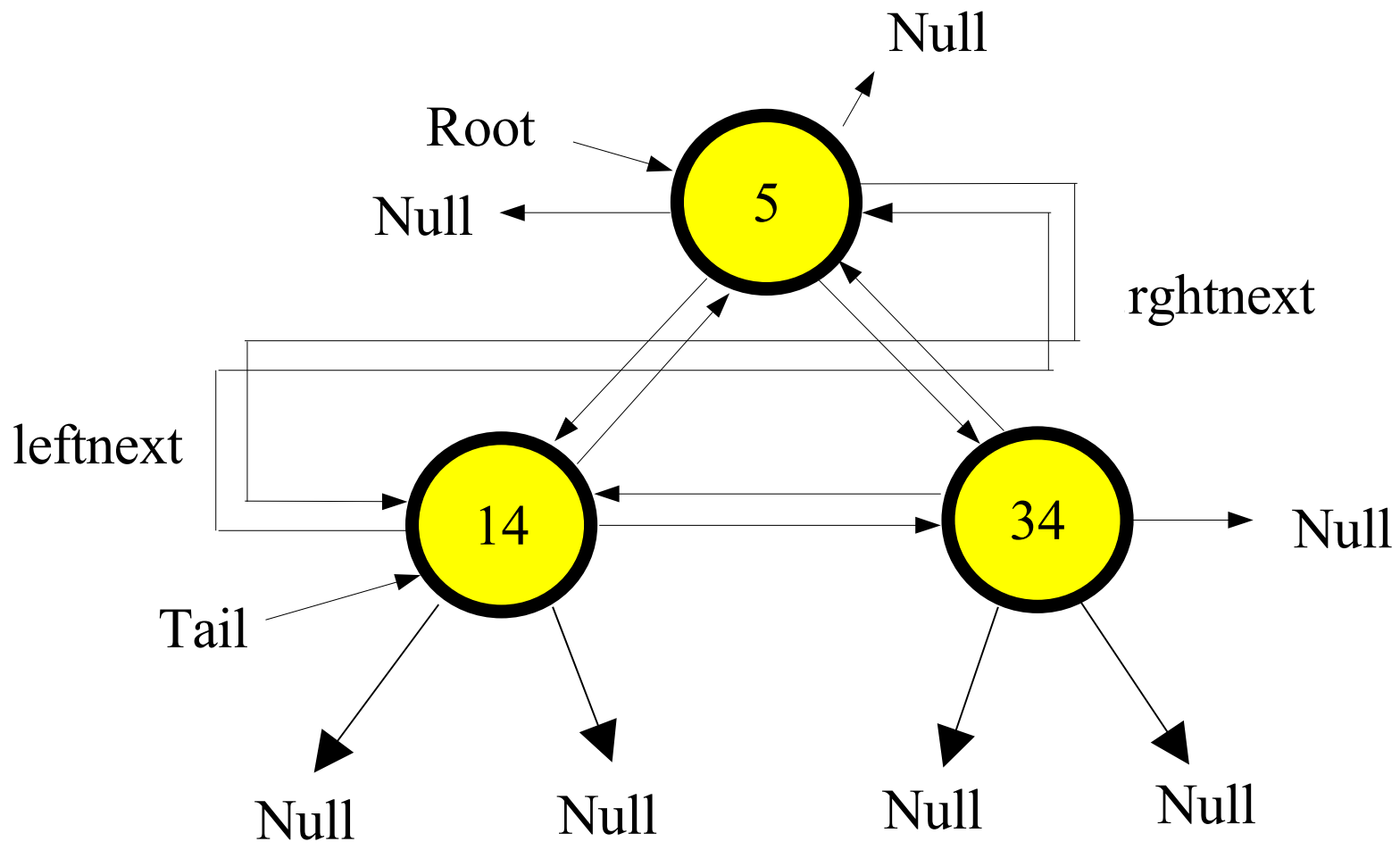


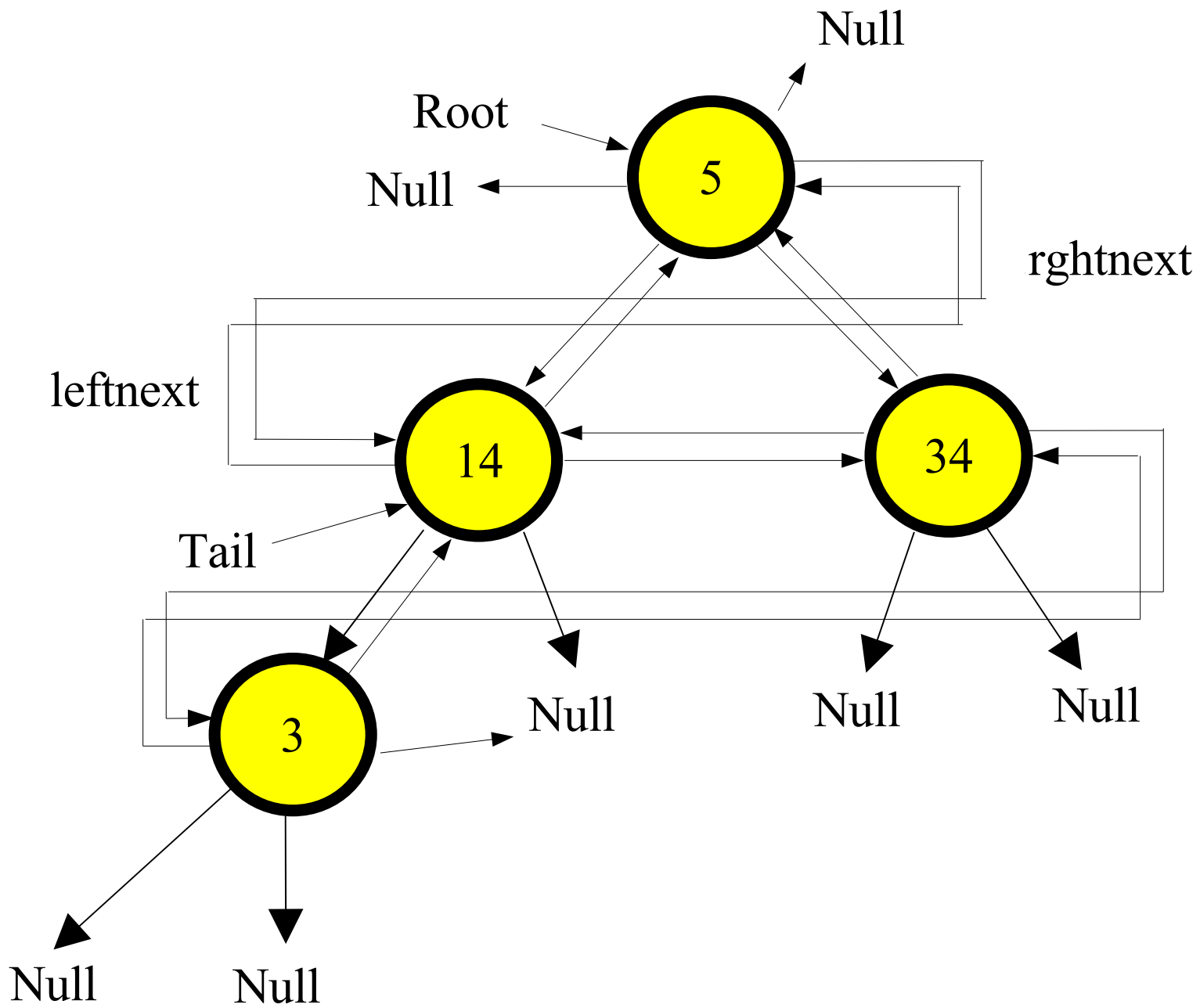
# Implementation Details

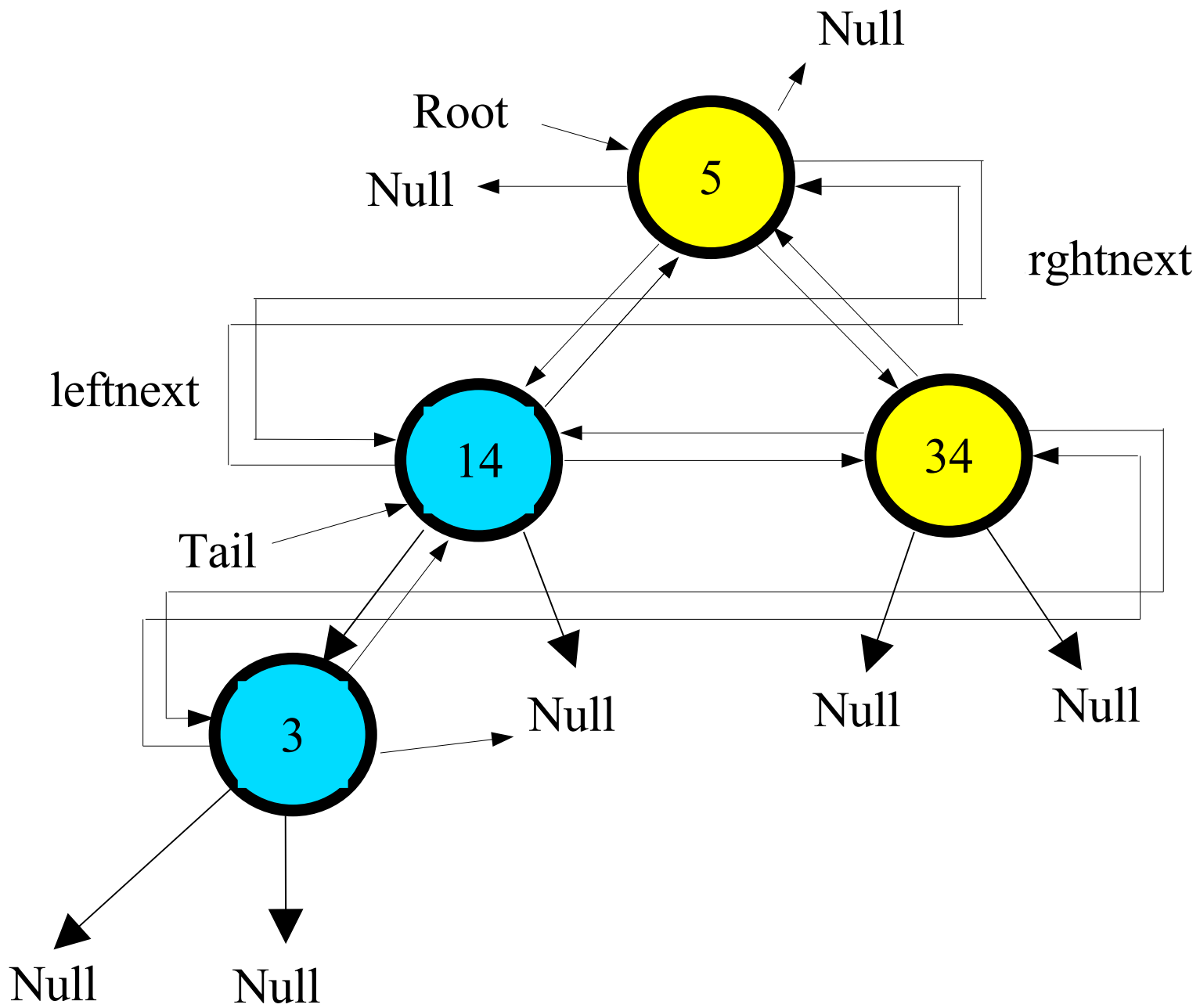




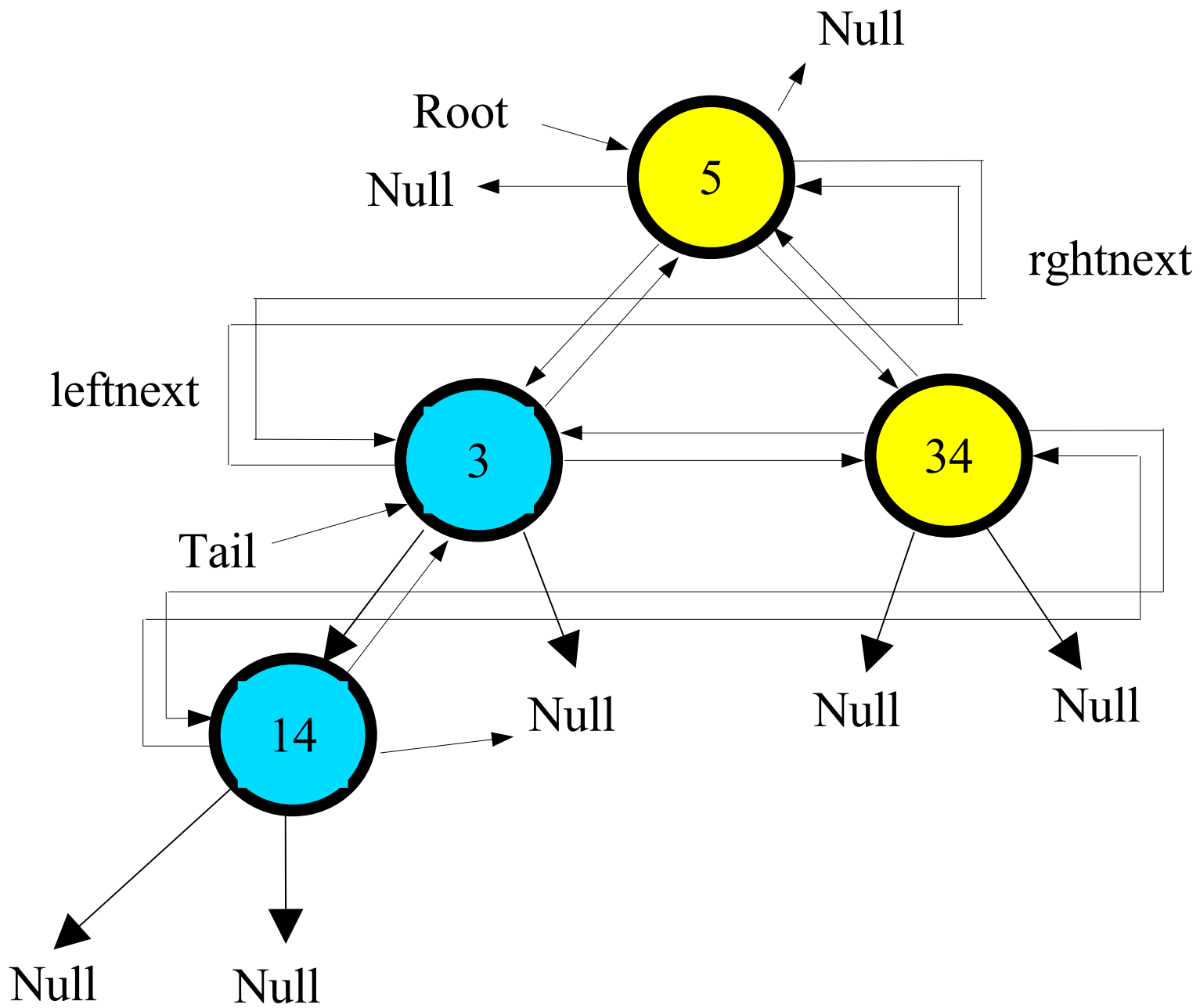


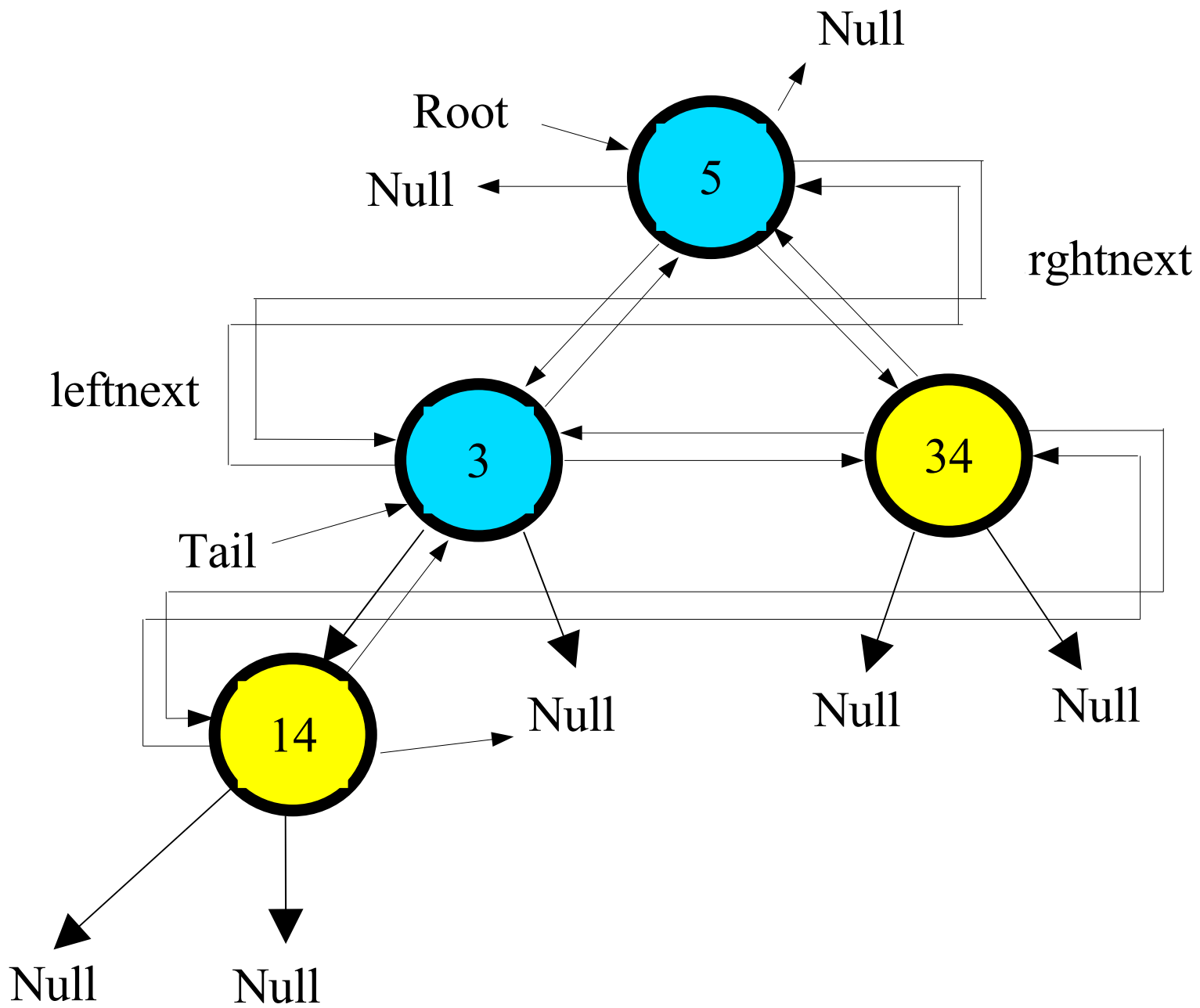


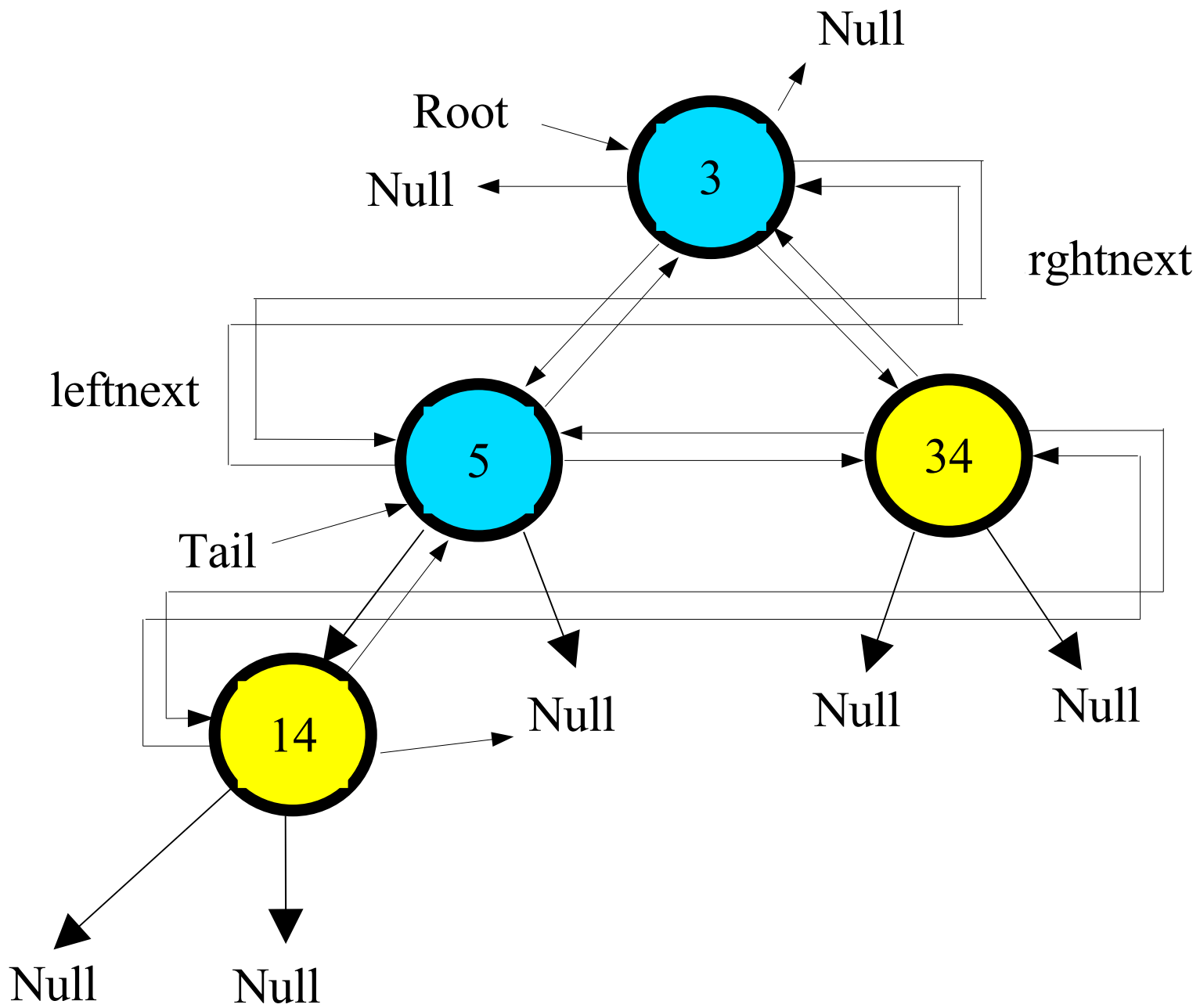






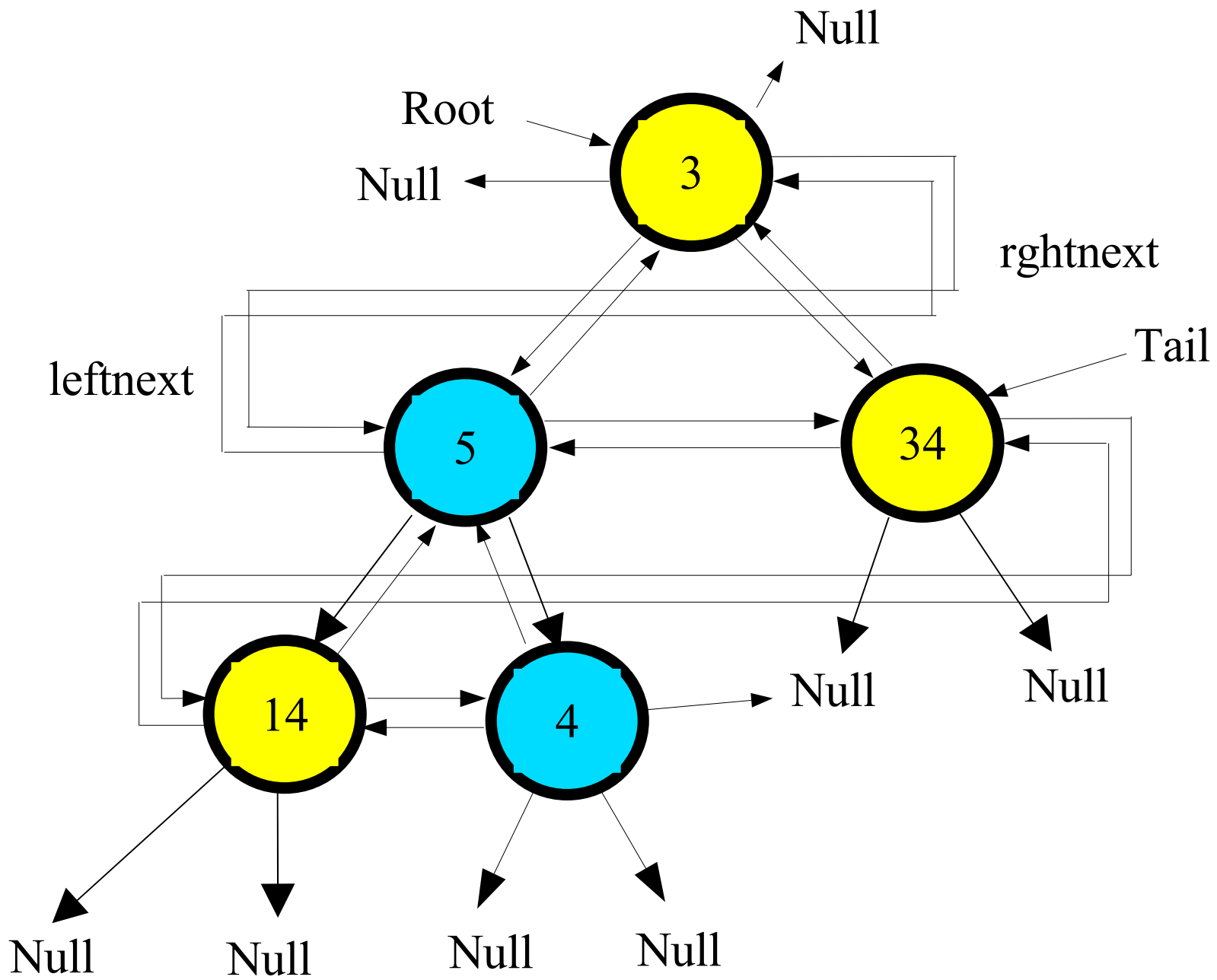


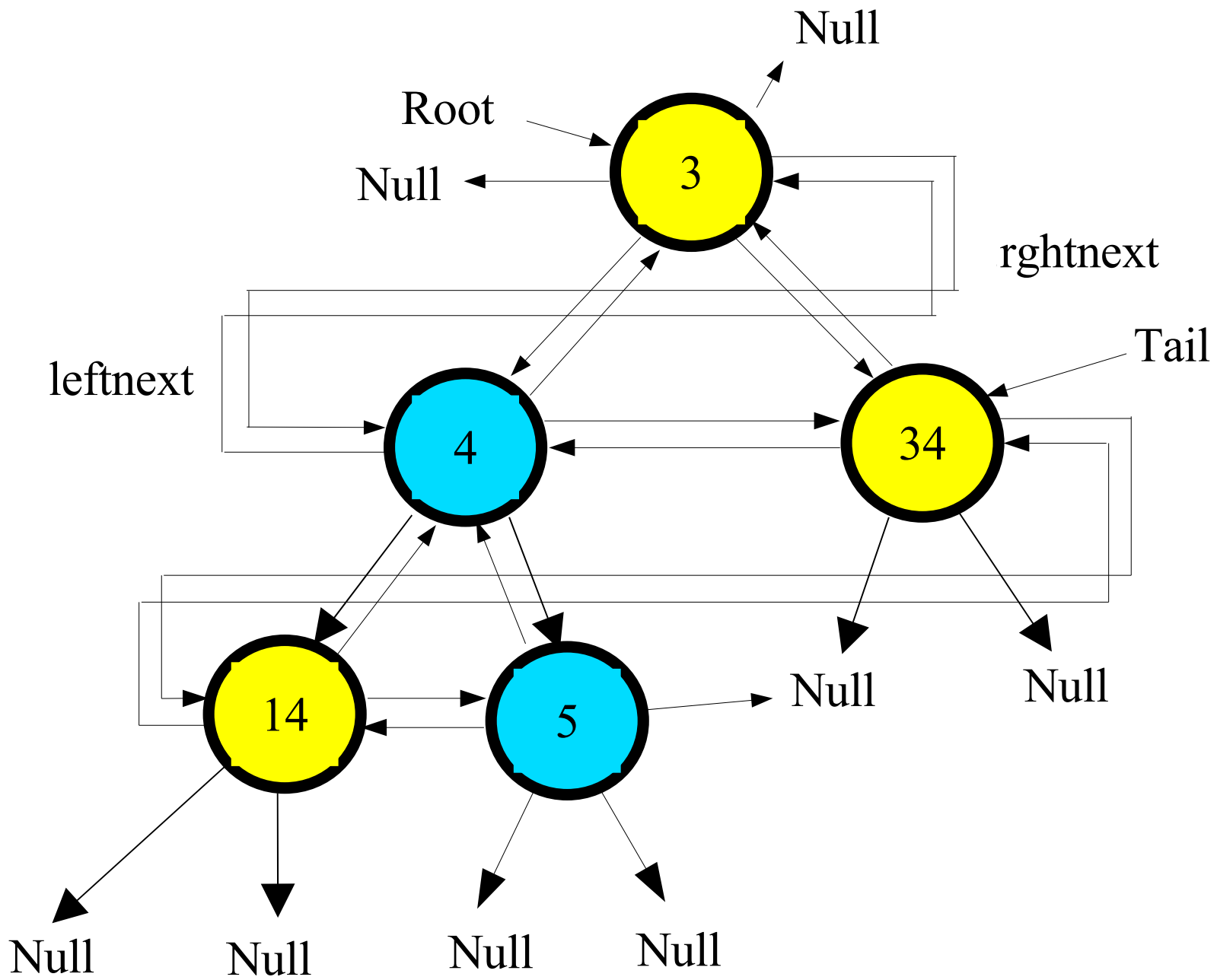


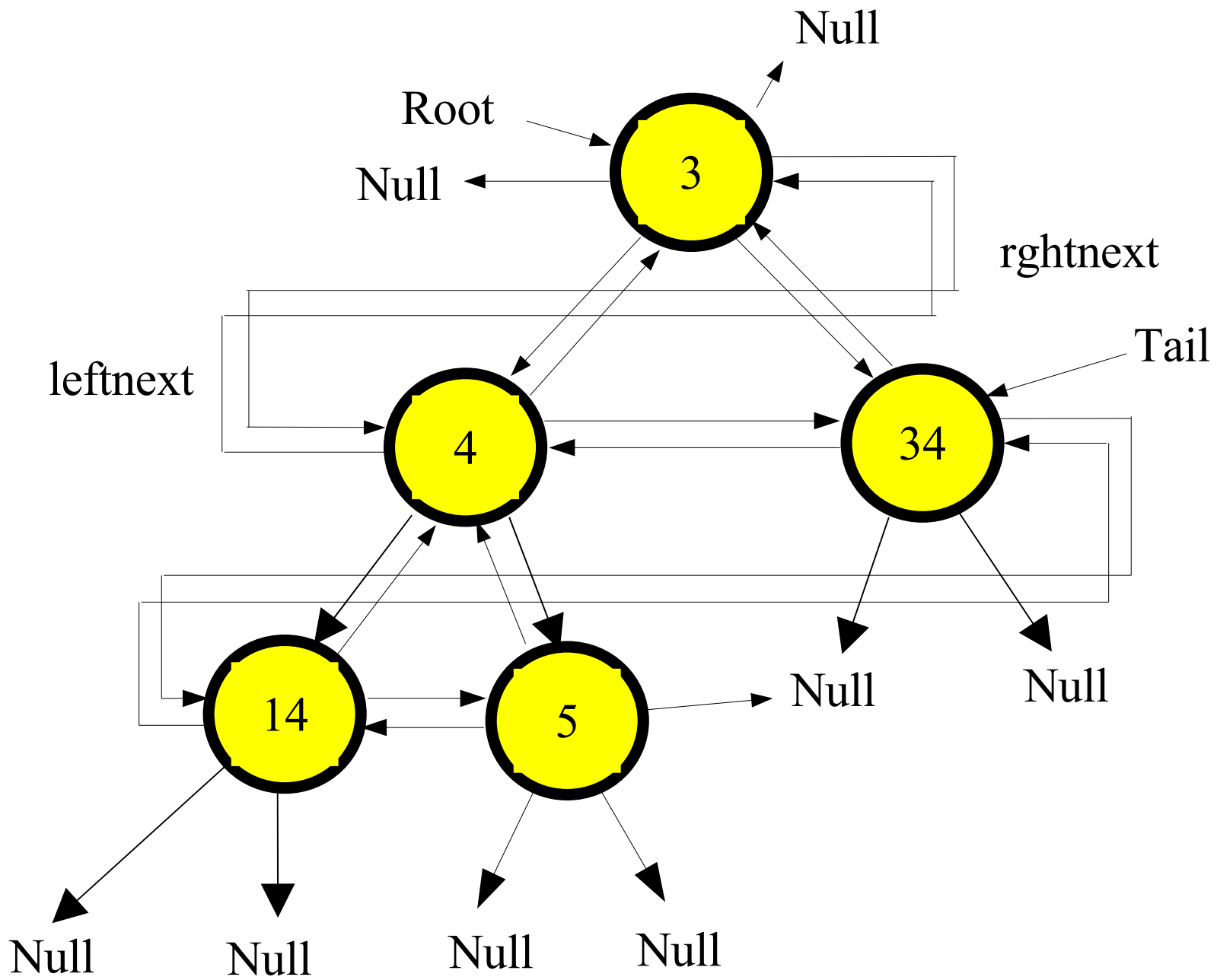






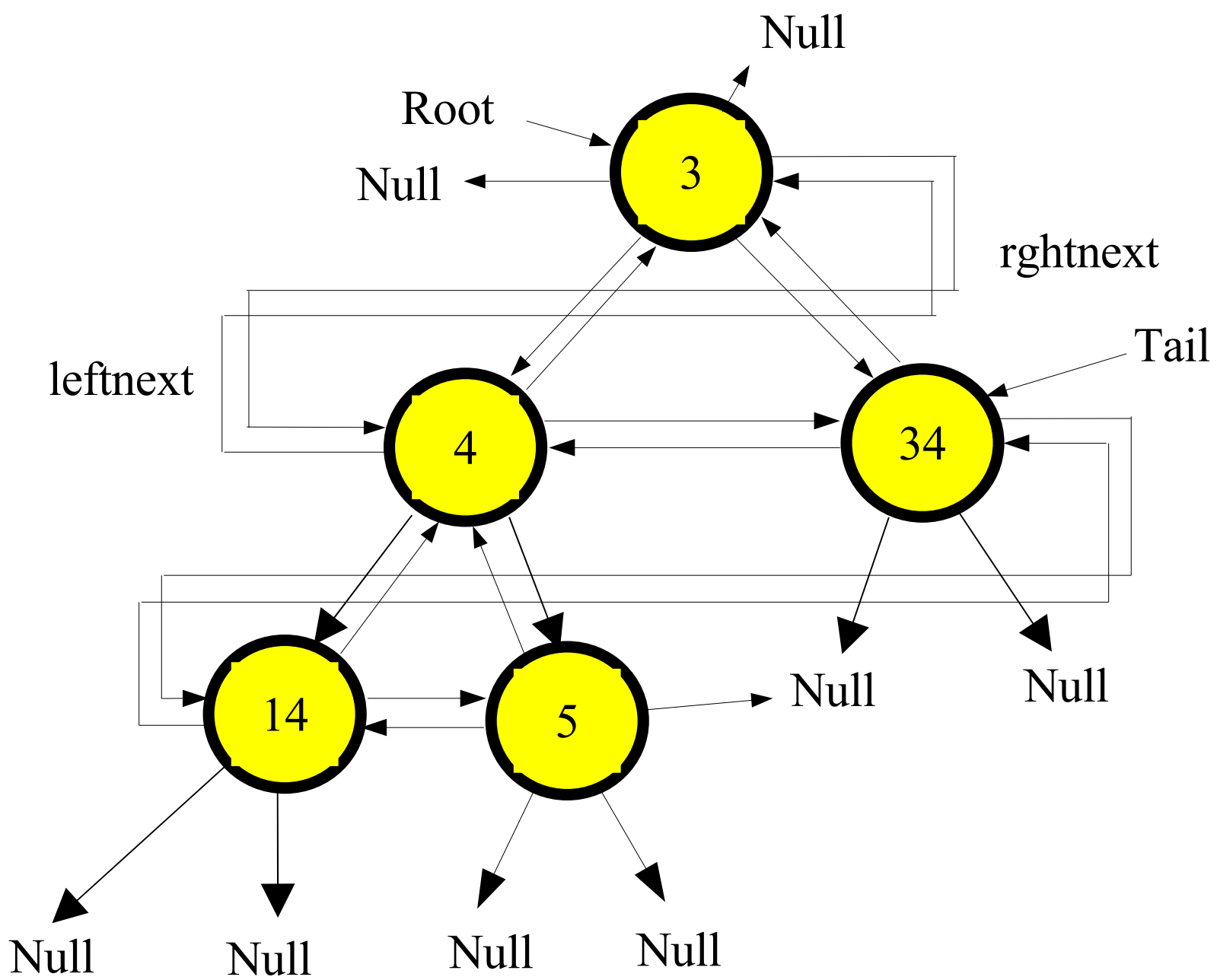


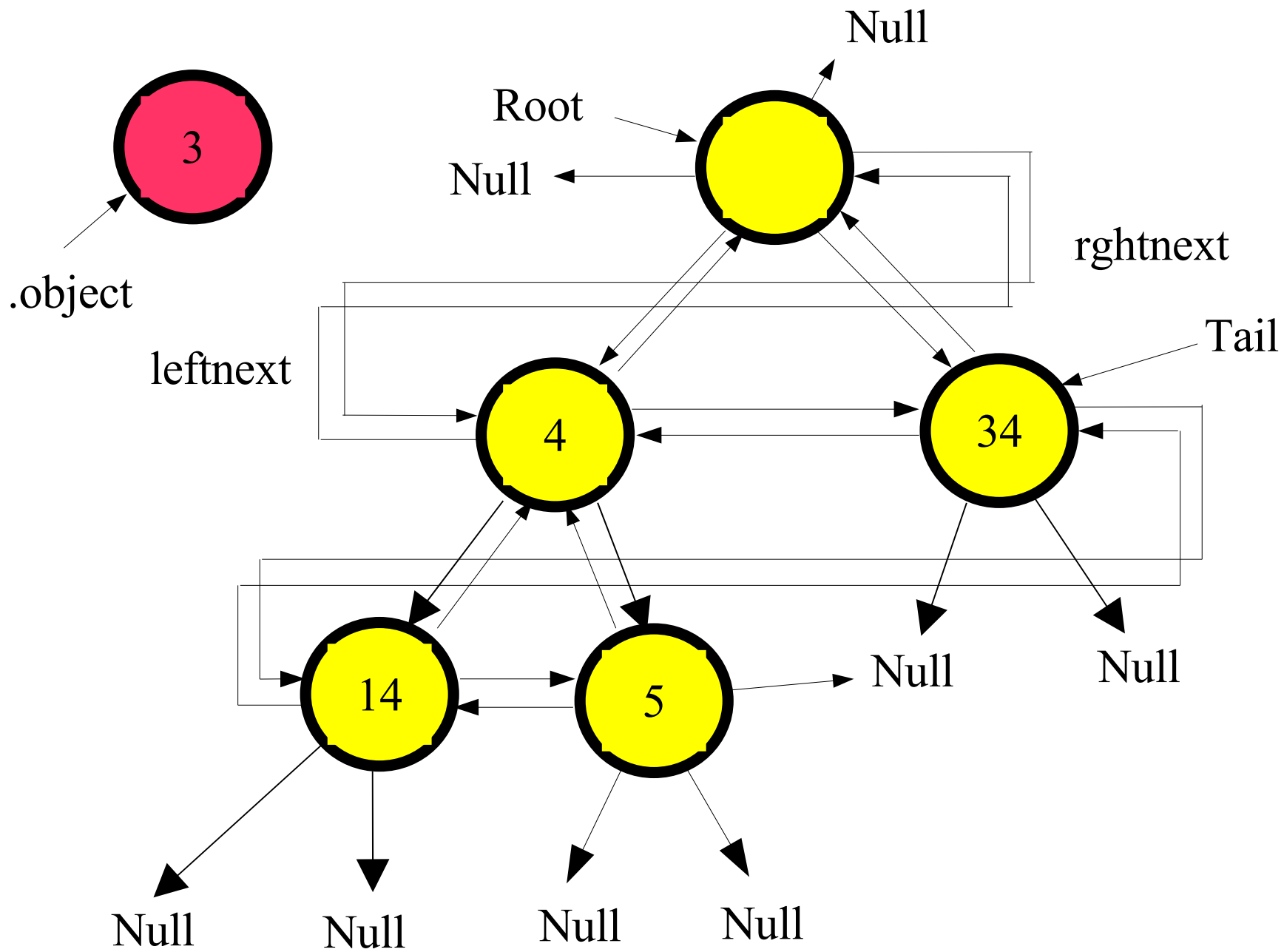


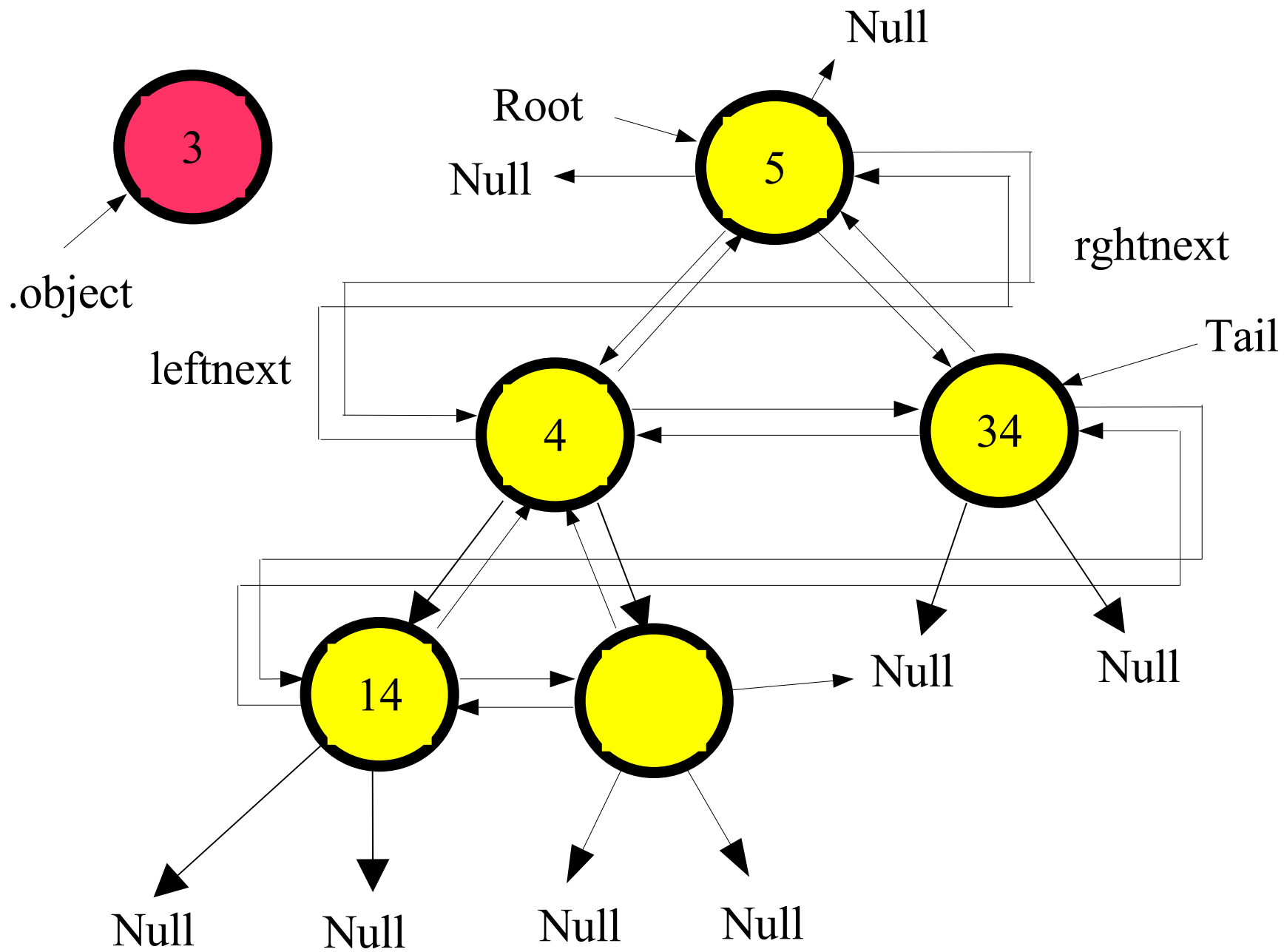


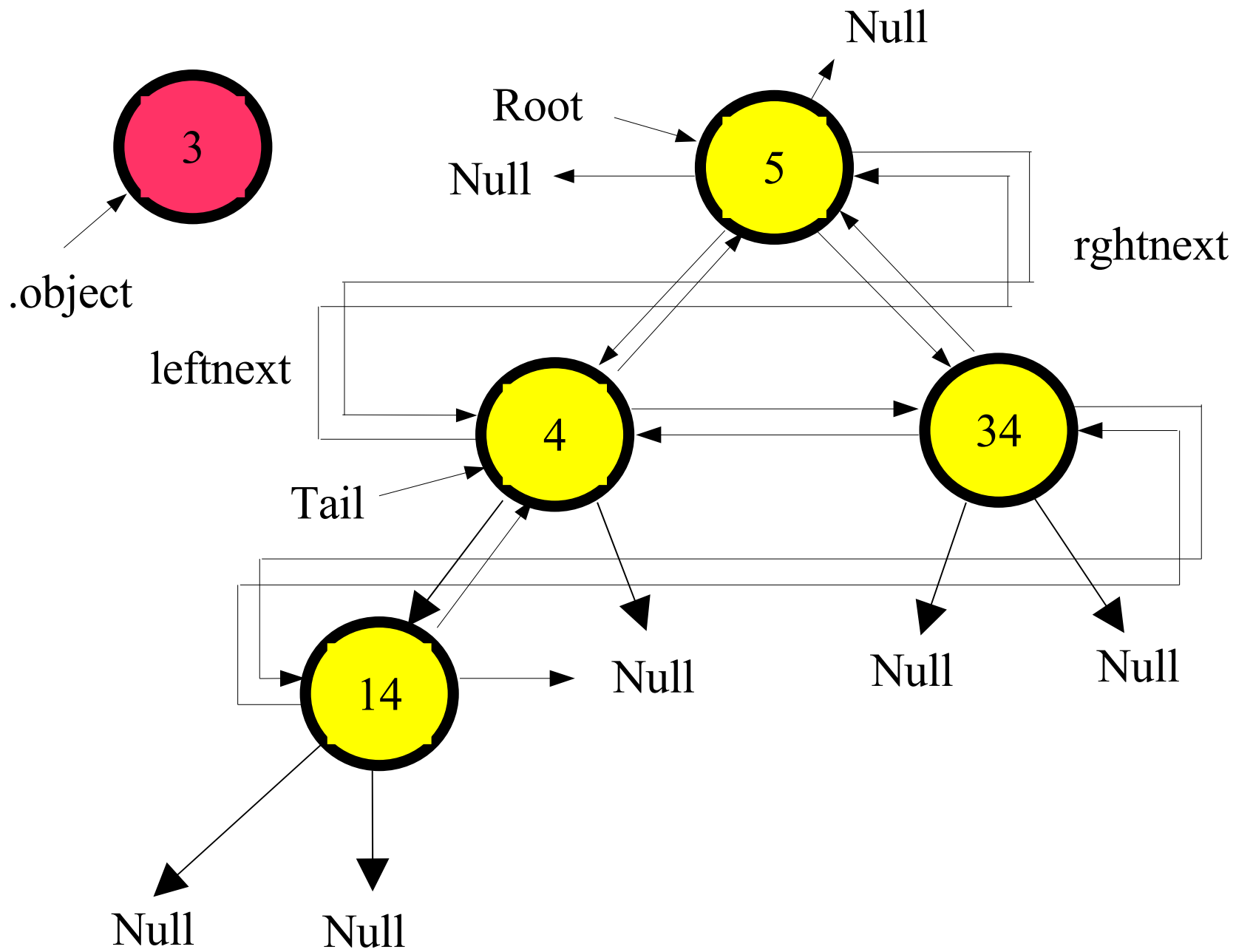


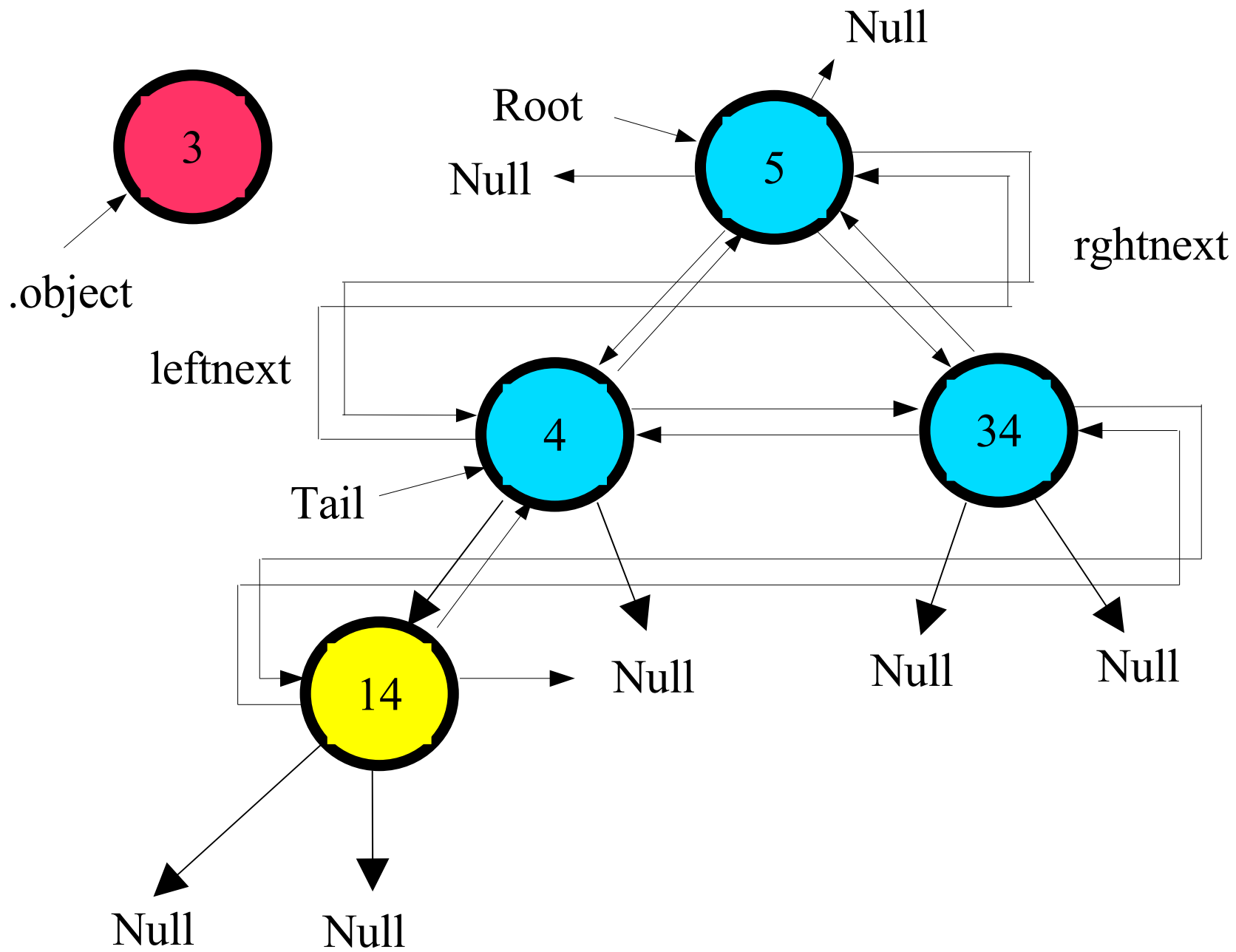
“Pop” an item off the top

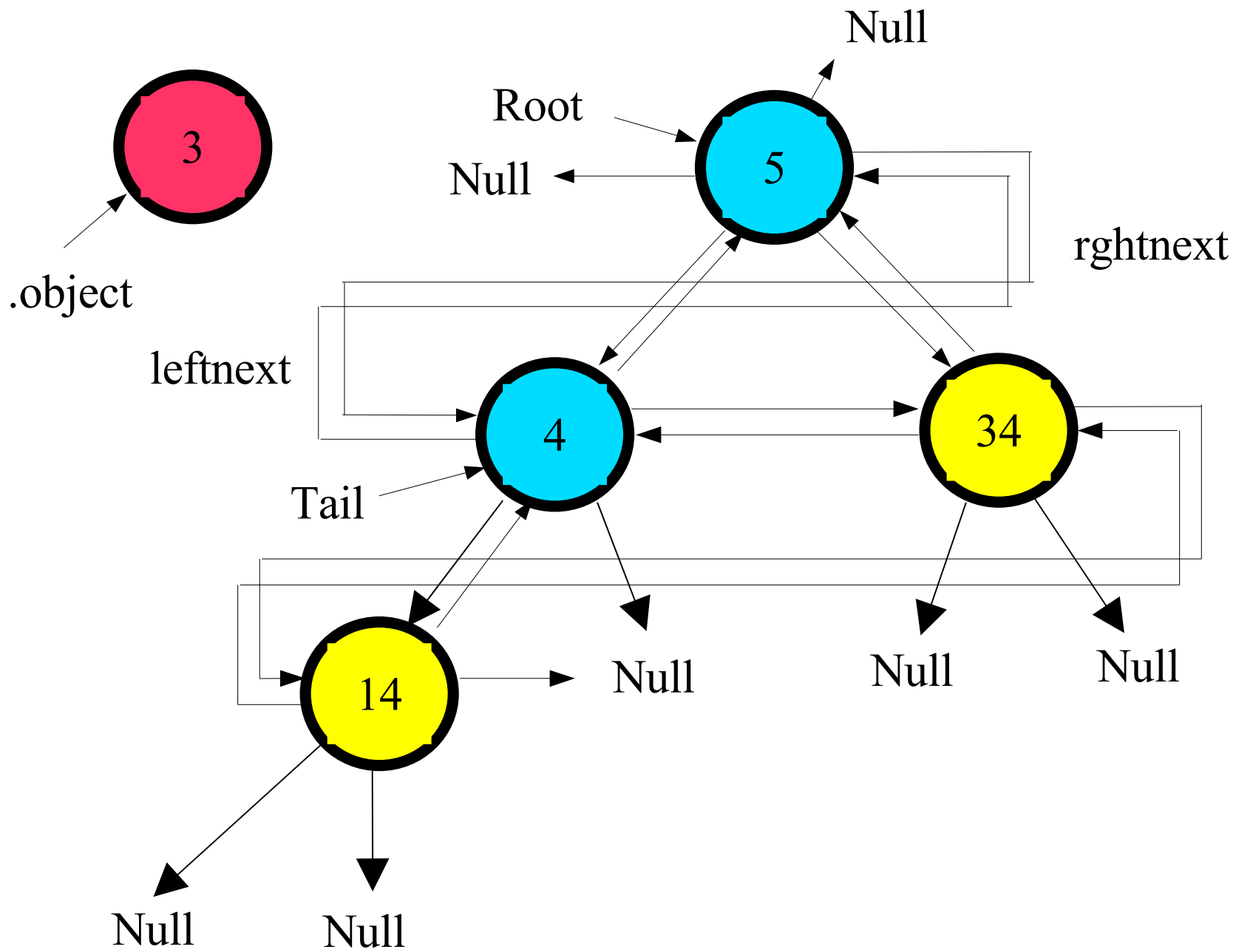


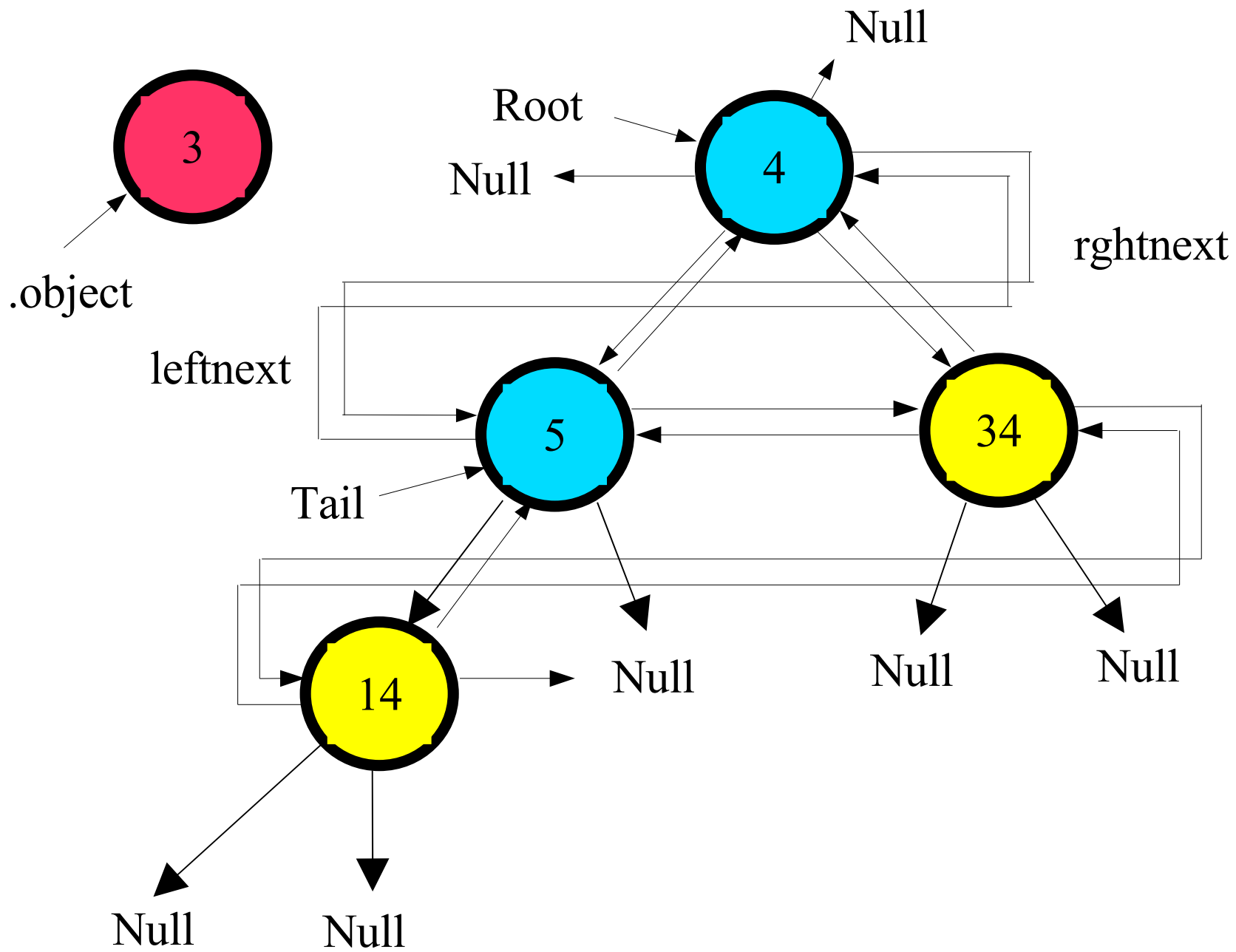




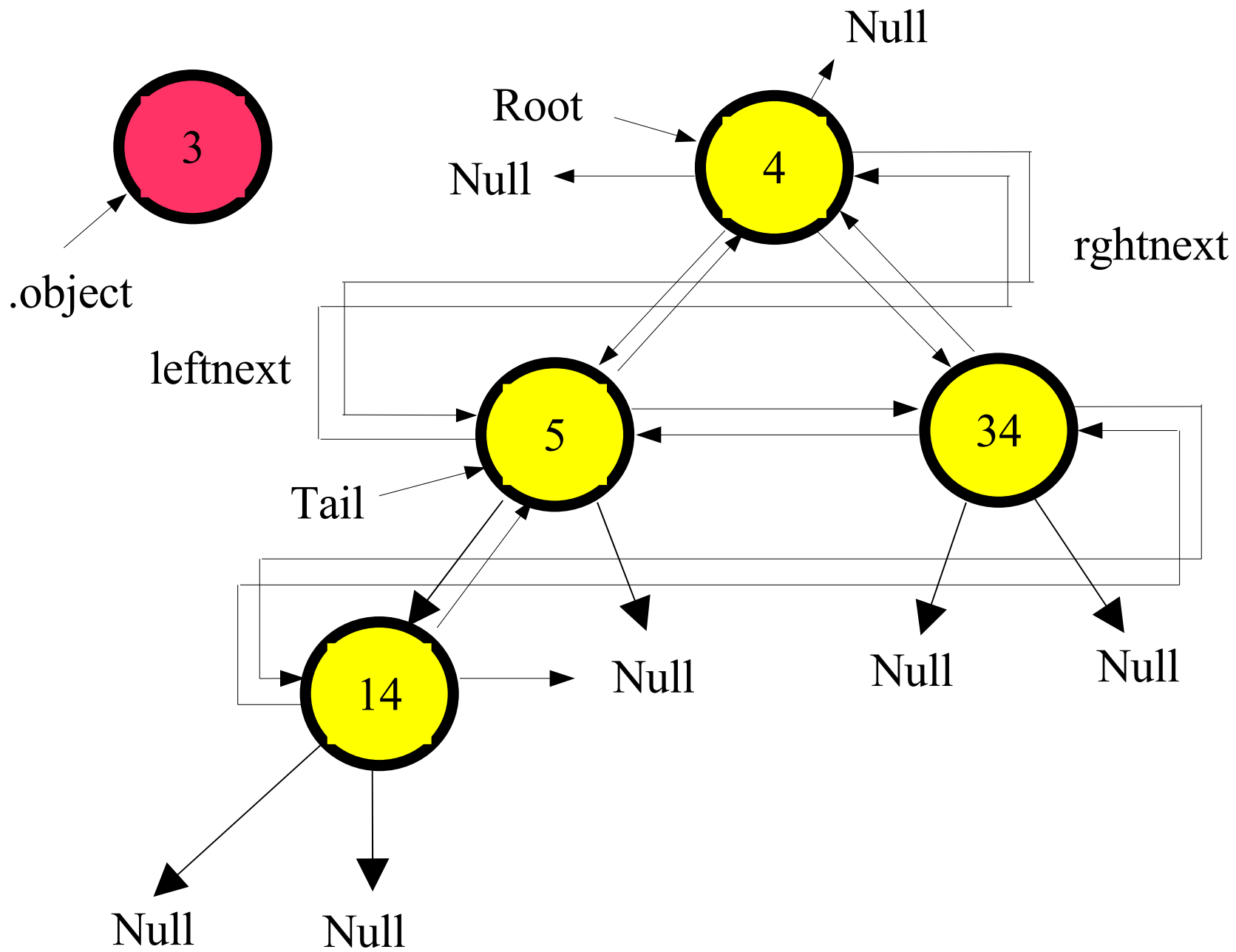








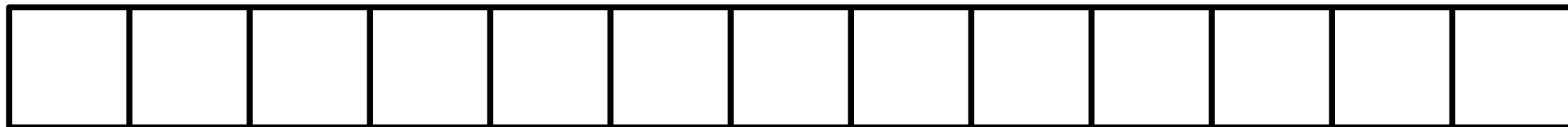






Implementation using  
an array of pointers

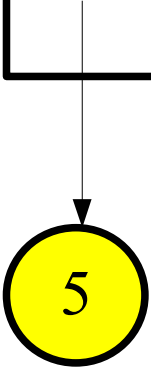
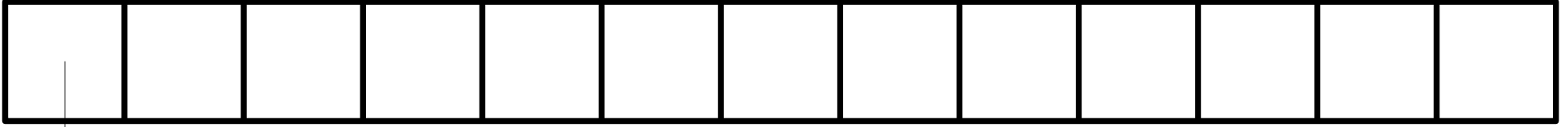
0 1 2 3 4 5 6 7 8 9 10 11 12



root = 0

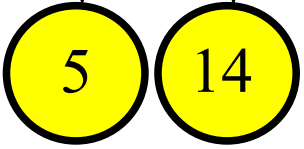
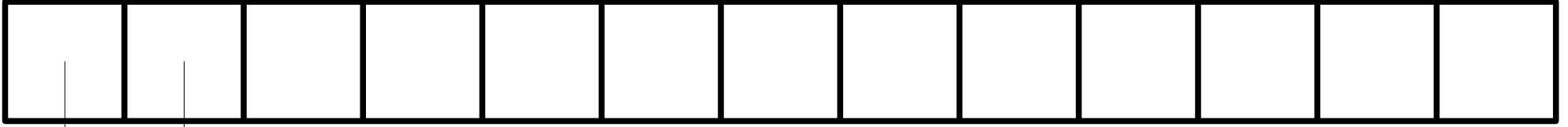
tail = 0

0 1 2 3 4 5 6 7 8 9 10 11 12



root = 0  
tail = 1

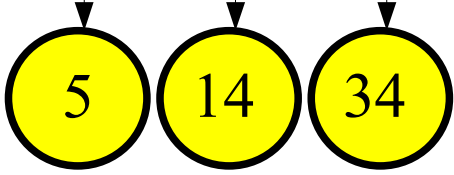
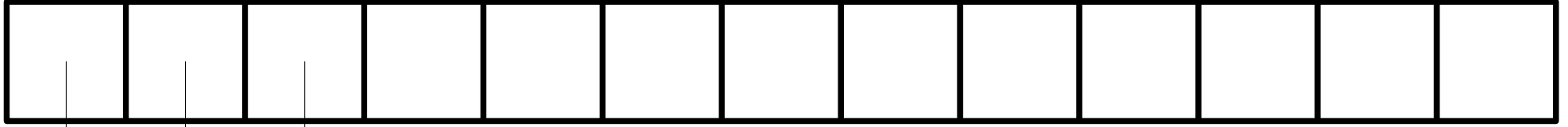
0 1 2 3 4 5 6 7 8 9 10 11 12



root = 0

tail = 2

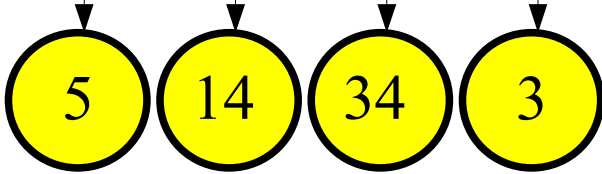
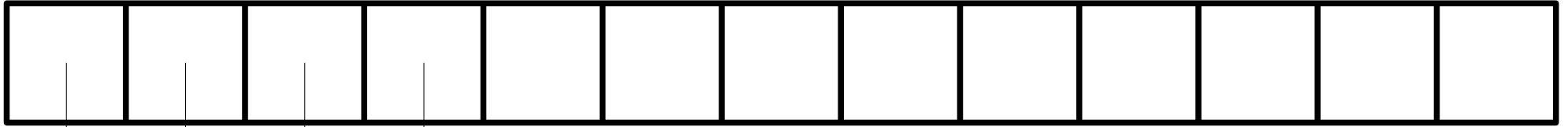
0 1 2 3 4 5 6 7 8 9 10 11 12



root = 0

tail = 3

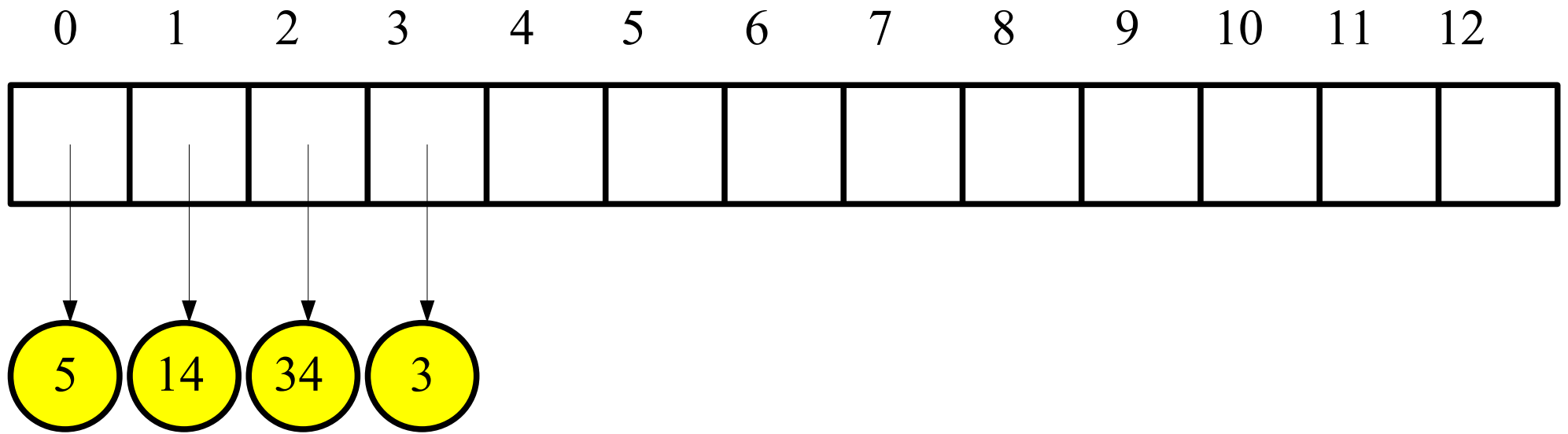
0 1 2 3 4 5 6 7 8 9 10 11 12



root = 0

tail = 4

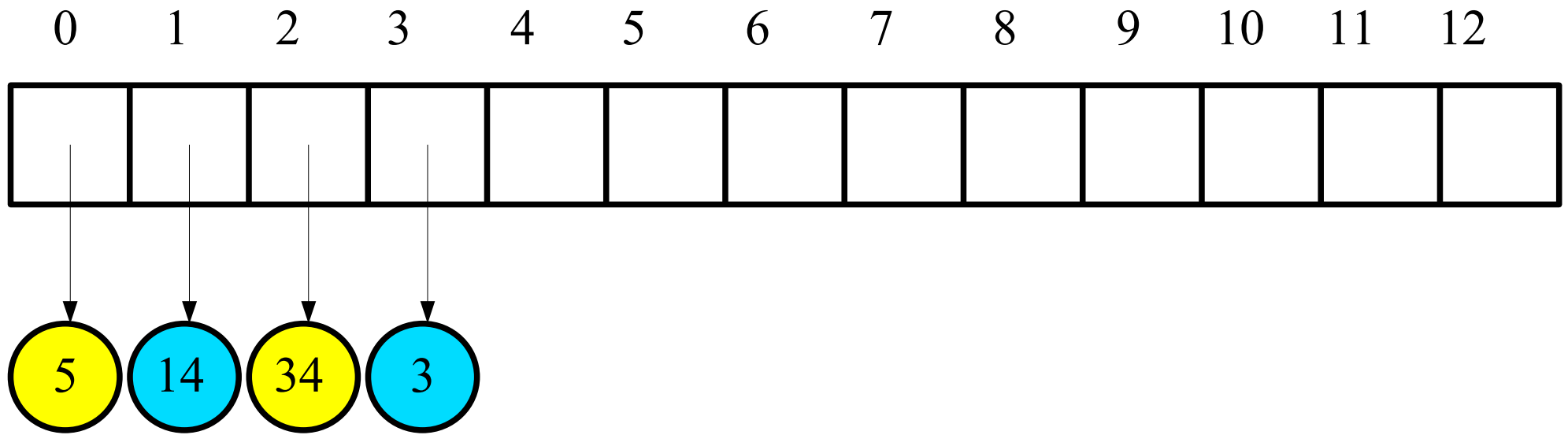




root = 0  
tail = 4

Parent of “3” node is determined by subtracting the index of that node by 1 and dividing by 2.  
In this case its

$$(3-1)/2 = 1$$

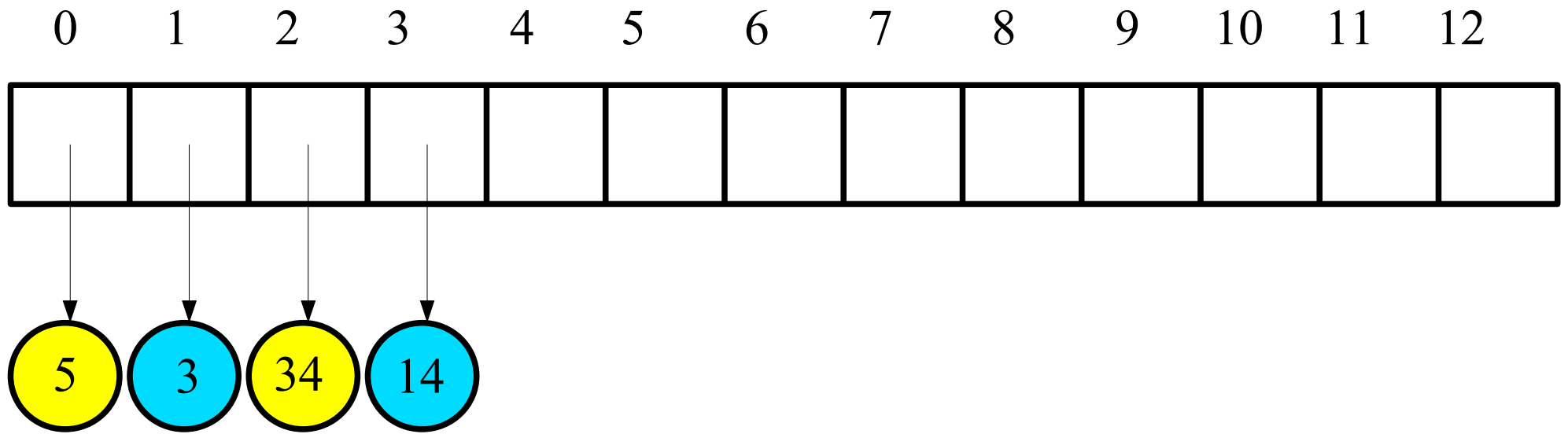


root = 0  
tail = 4

Parent of “3” node is determined by subtracting the index of that node by 1 and dividing by 2.  
In this case its

$$(3-1)/2 = 1$$

Compare parent with child.



root = 0  
tail = 4

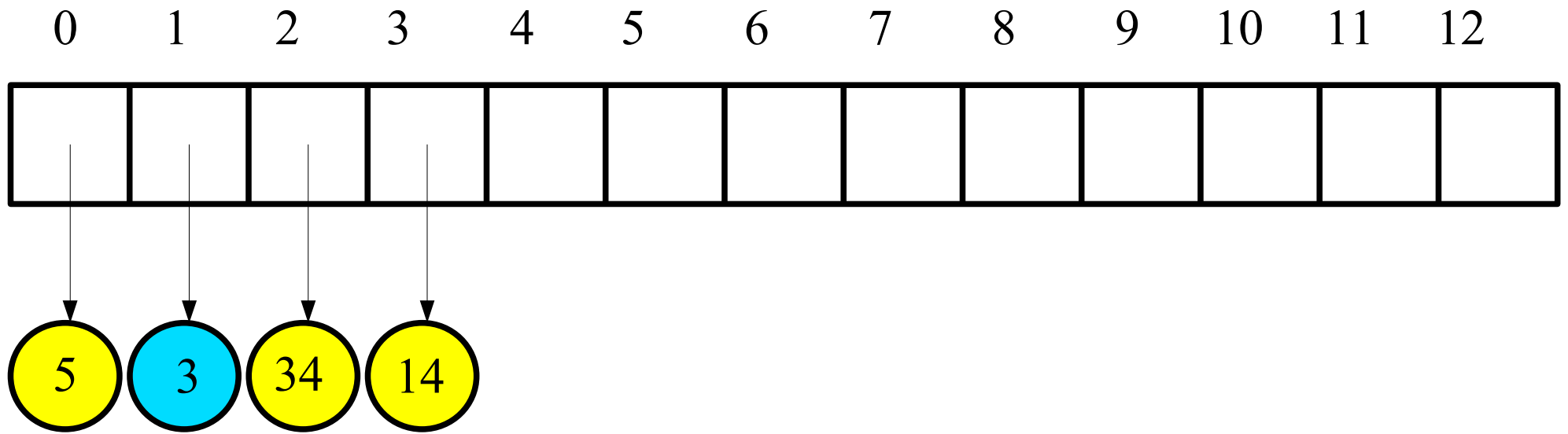
Parent of “3” node is determined by subtracting the index of that node by 1 and dividing by 2.

In this case its

$$(3-1)/2 = 1$$

Compare parent with child.

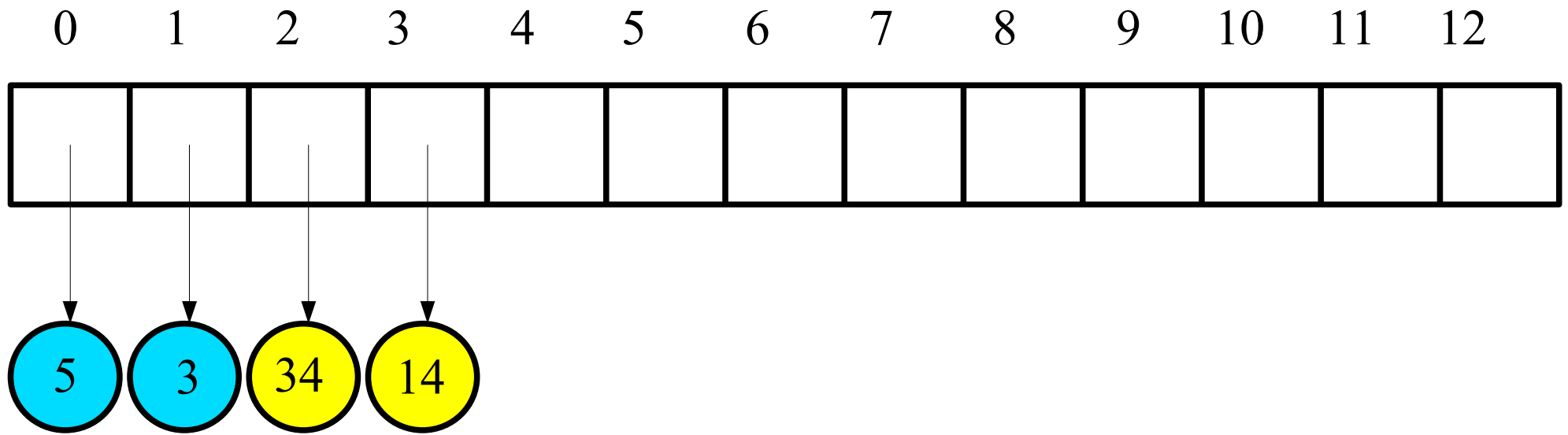
Move the child up by one level.



root = 0  
tail = 4

Parent of “3” node is determined by subtracting the index of that node by 1 and dividing by 2.  
In this case its

$$(1-1)/2 = 0$$

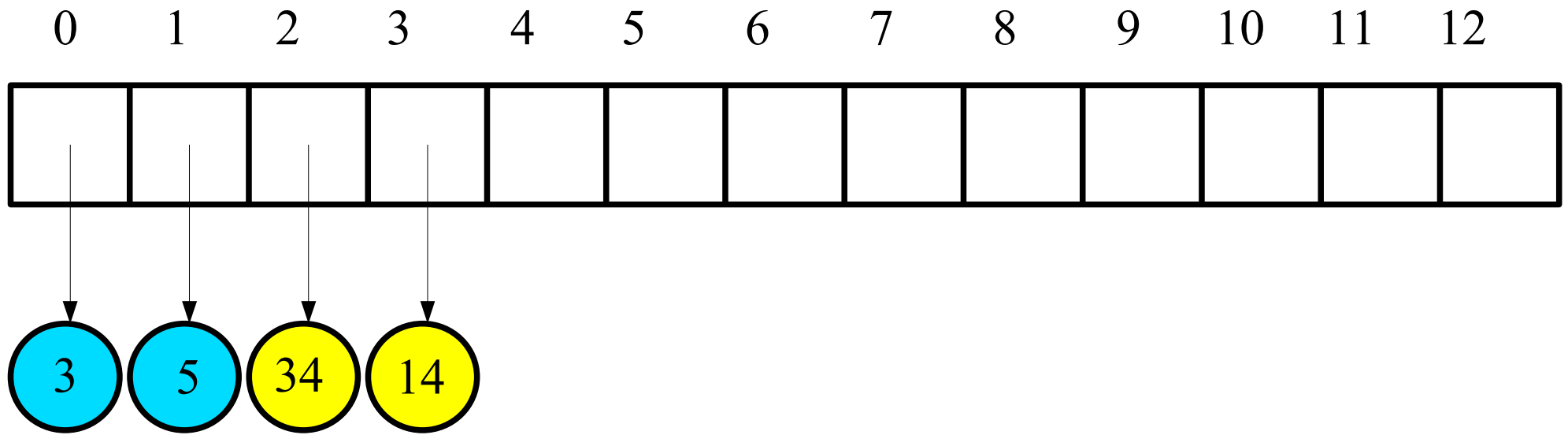


root = 0  
tail = 4

Parent of “3” node is determined by subtracting the index of that node by 1 and dividing by 2.  
In this case its

$$(1-1)/2 = 0$$

Compare parent and child.



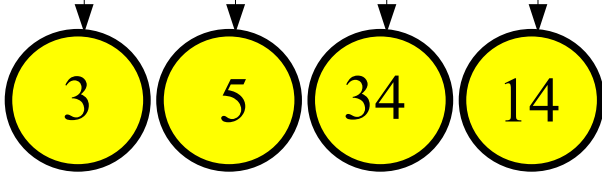
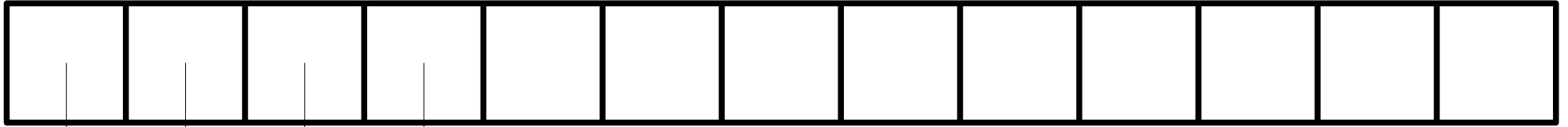
root = 0  
tail = 4

Parent of “3” node is determined by subtracting the index of that node by 1 and dividing by 2.  
In this case its

$$(1-1)/2 = 0$$

Compare parent and child.  
Move child up one level.

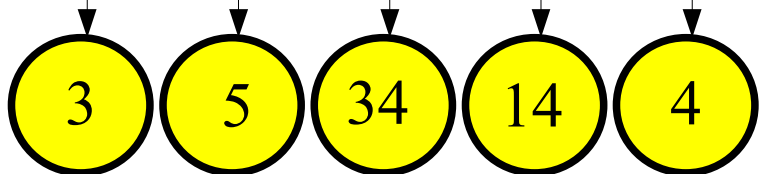
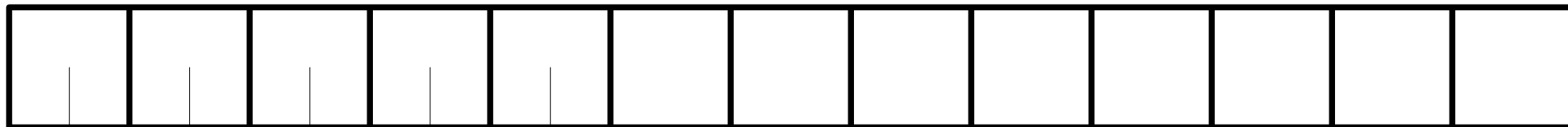
0 1 2 3 4 5 6 7 8 9 10 11 12



root = 0

tail = 4

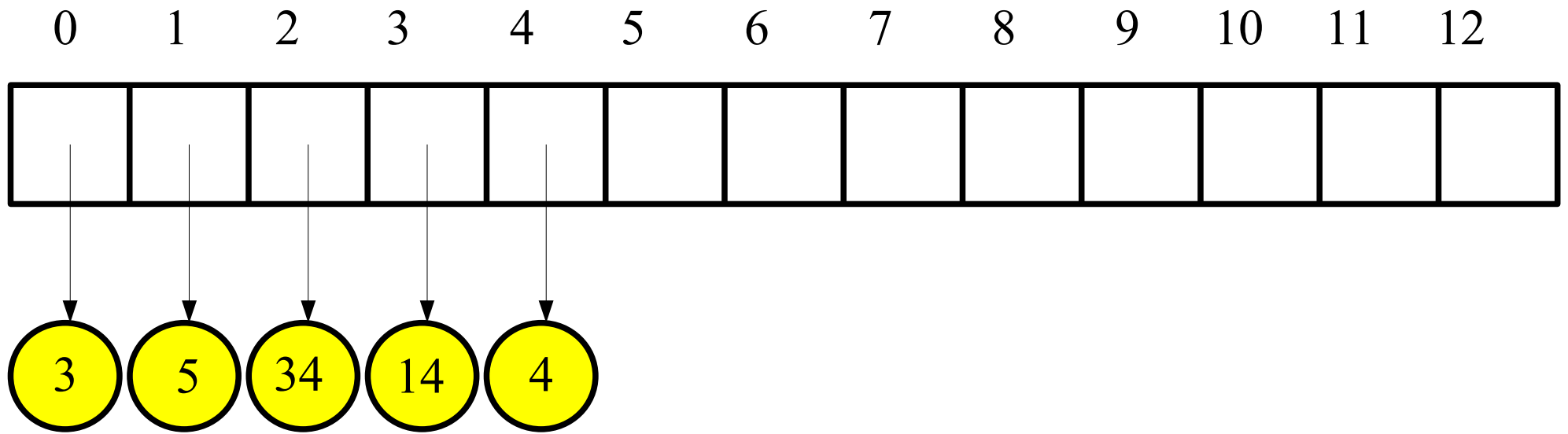
0 1 2 3 4 5 6 7 8 9 10 11 12



root = 0

tail = 5

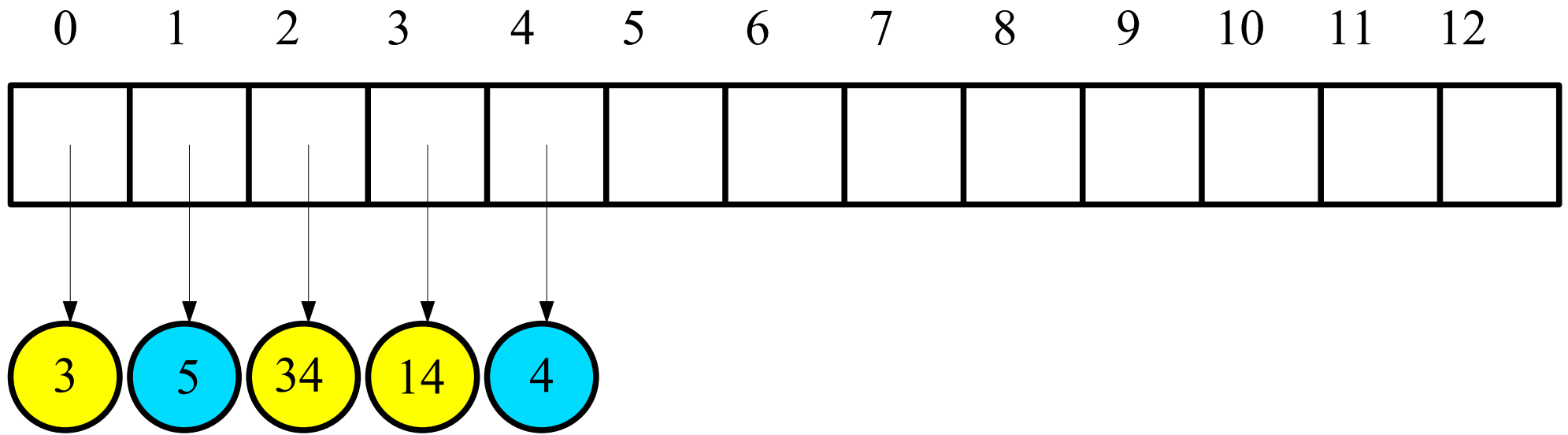




root = 0  
tail = 5

Parent of “4” node is determined by subtracting the index of that node by 1 and dividing by 2.  
In this case its

$$(4-1)/2 = 1$$

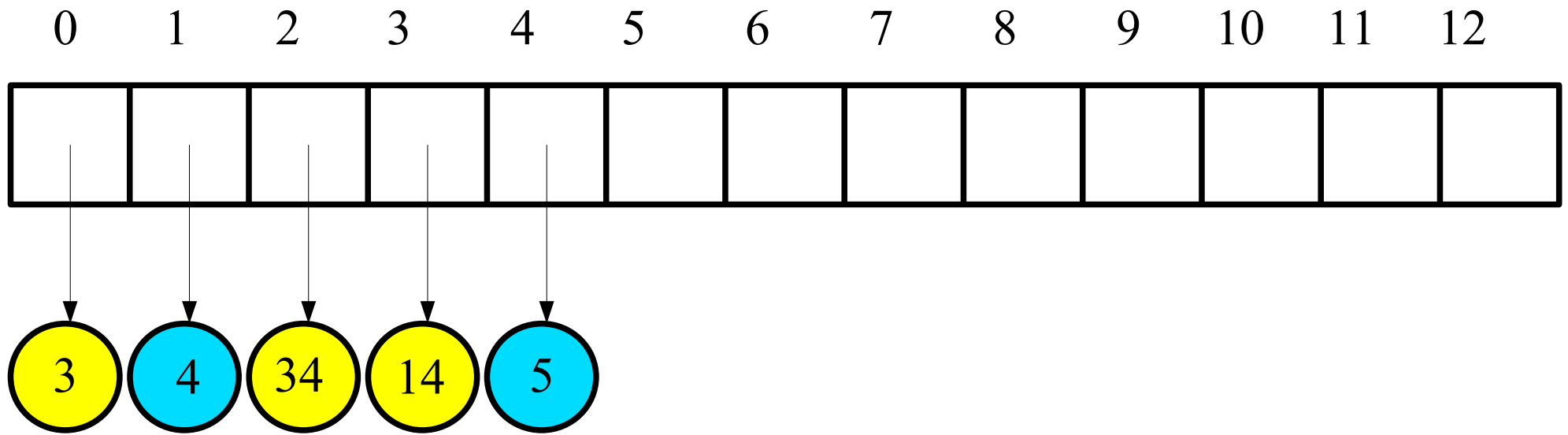


root = 0  
tail = 5

Parent of “4” node is determined by subtracting the index of that node by 1 and dividing by 2.  
In this case its

$$(4-1)/2 = 1$$

Compare parent and child.

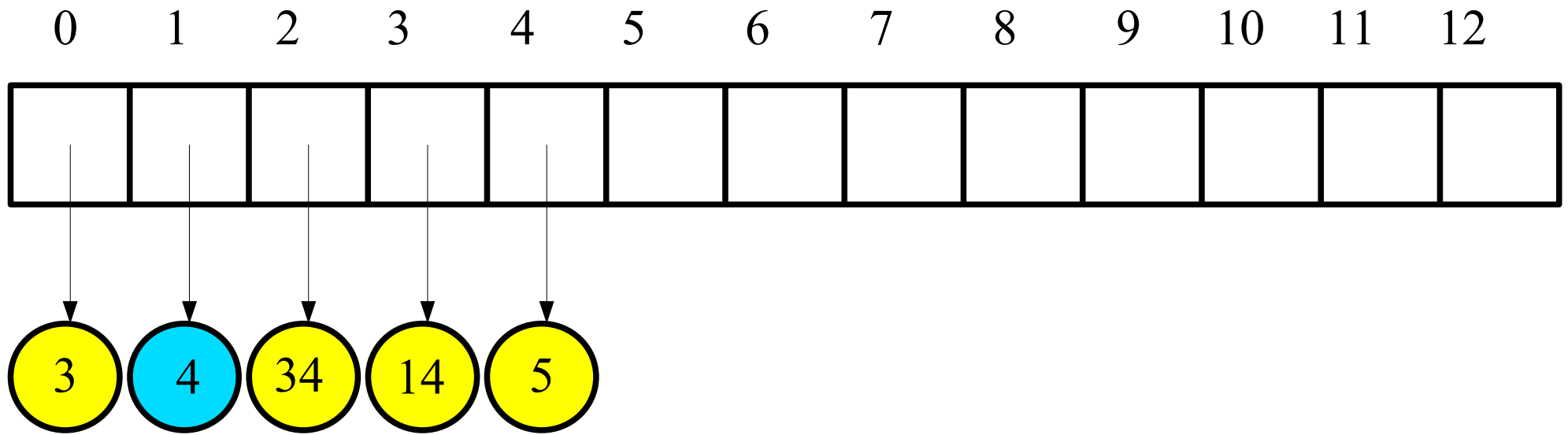


root = 0  
tail = 5

Parent of “4” node is determined by subtracting the index of that node by 1 and dividing by 2.  
In this case its

$$(4-1)/2 = 1$$

Compare parent and child.  
Move child up one level.

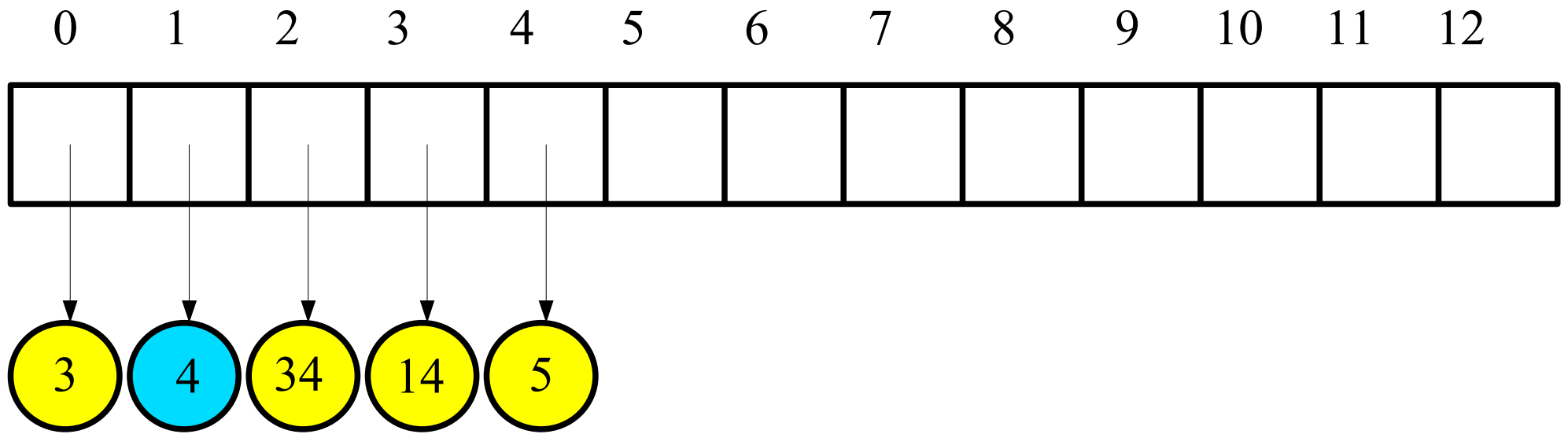


root = 0  
tail = 5

Parent of “4” node is determined by subtracting the index of that node by 1 and dividing by 2.  
In this case its

$$(4-1)/2 = 1$$

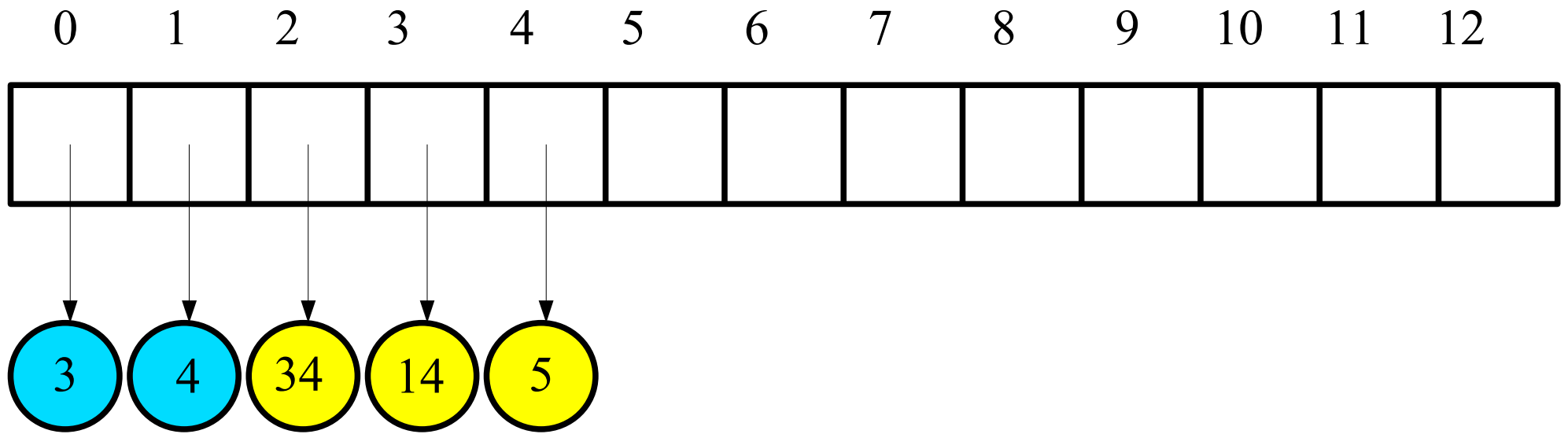
Compare parent and child.  
Move child up one level.



root = 0  
tail = 5

Parent of “4” node is determined by subtracting the index of that node by 1 and dividing by 2.  
In this case its

$$(1-1)/2 = 0$$

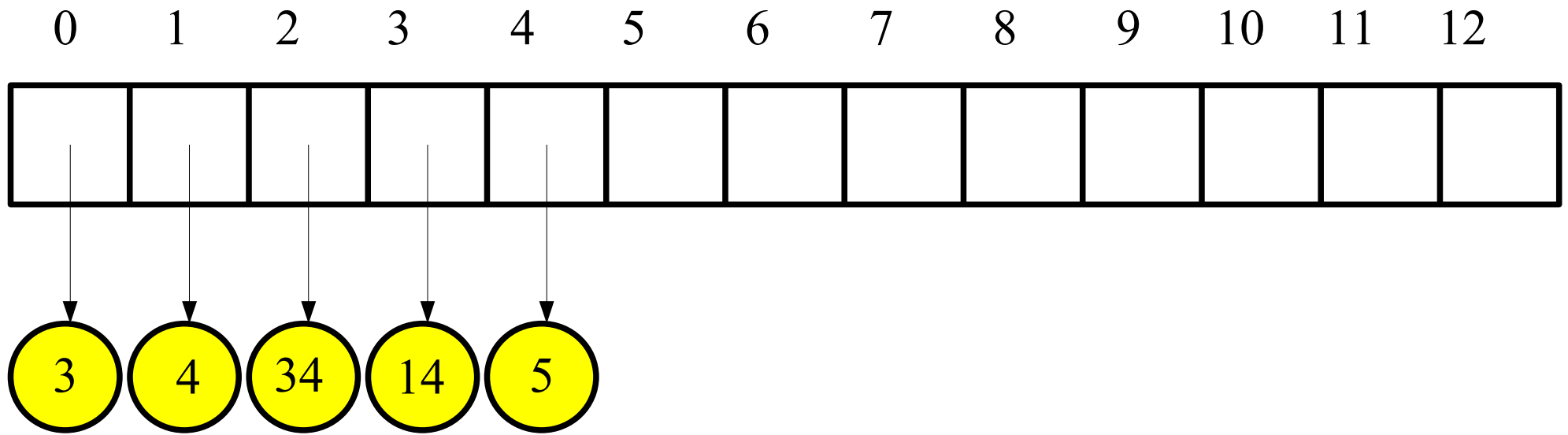


root = 0  
tail = 5

Parent of “4” node is determined by subtracting the index of that node by 1 and dividing by 2.  
In this case its

$$(1-1)/2 = 0$$

Compare parent and child.



root = 0  
tail = 5

Parent of “4” node is determined by subtracting the index of that node by 1 and dividing by 2.  
In this case its

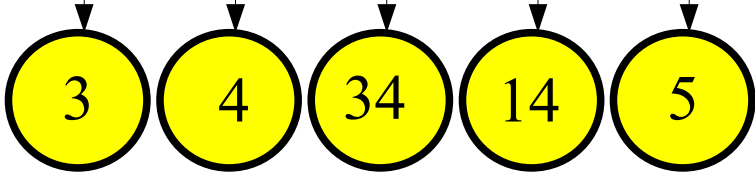
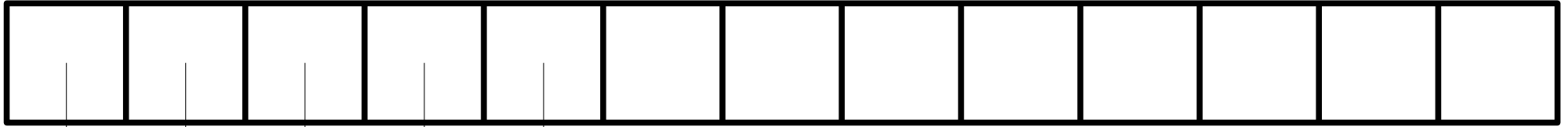
$$(1-1)/2 = 0$$

Compare parent and child.  
Leave child alone.

Now “pop” the “lowest” valued object

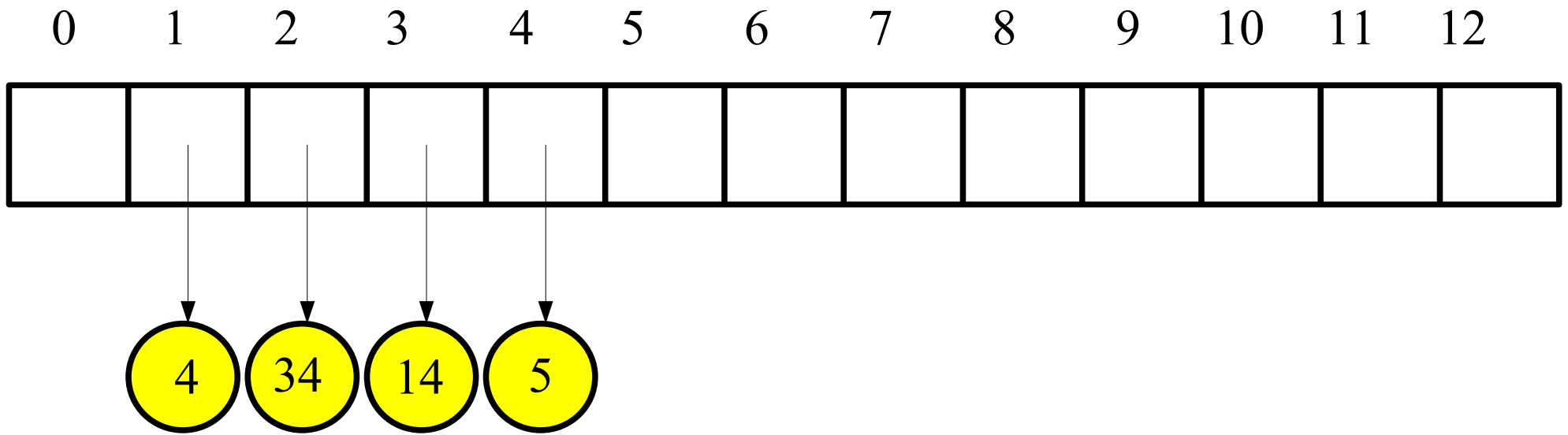


0 1 2 3 4 5 6 7 8 9 10 11 12



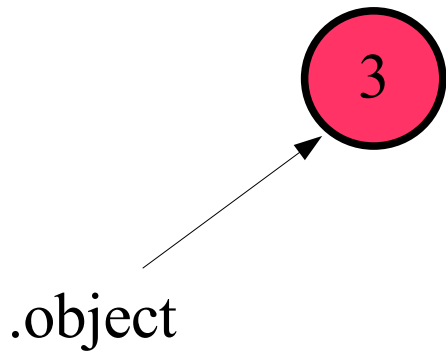
root = 0  
tail = 5

Now to “pop” the root node

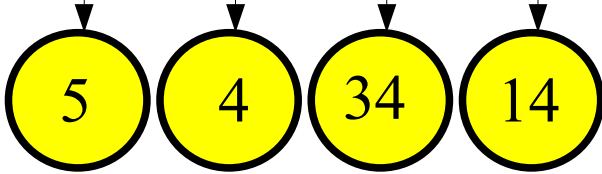
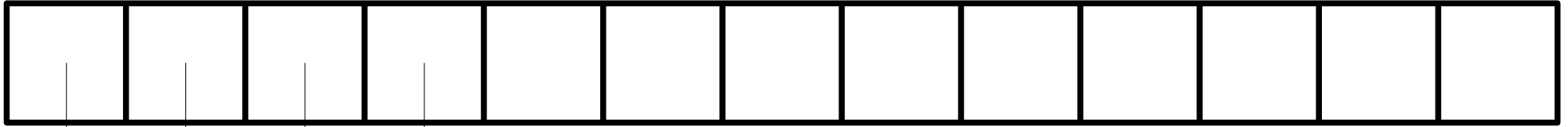


root = 0  
tail = 5

Now to “pop” the root node.

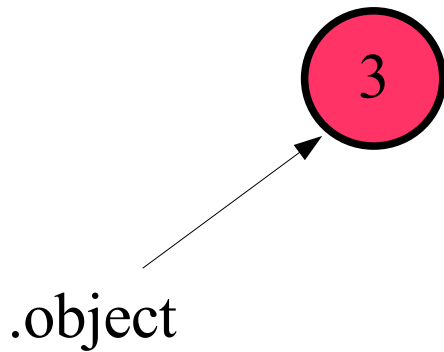


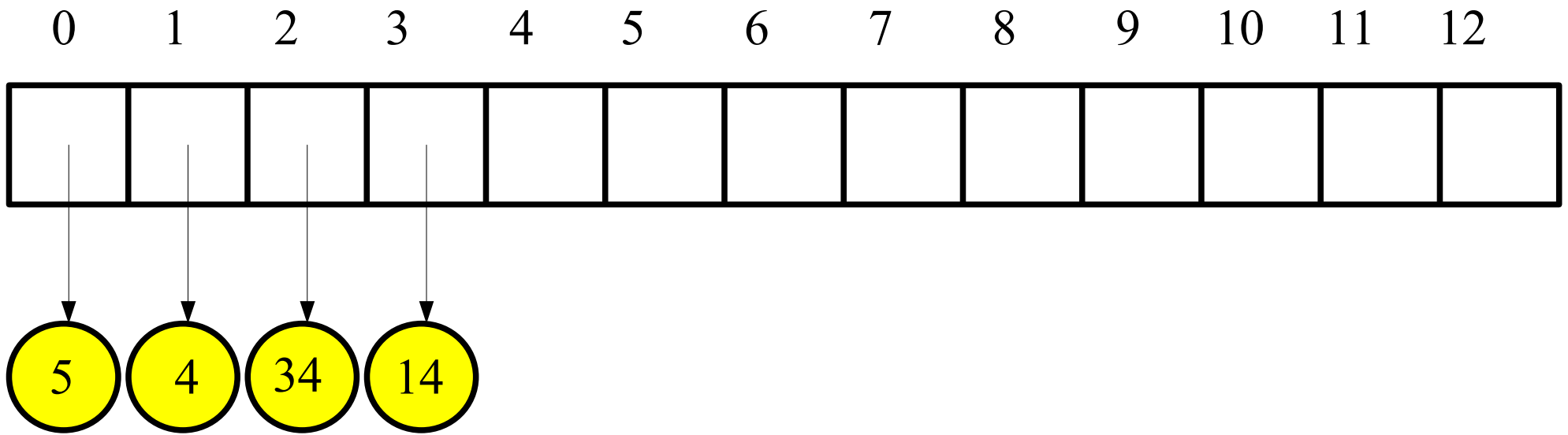
0 1 2 3 4 5 6 7 8 9 10 11 12



root = 0  
tail = 5

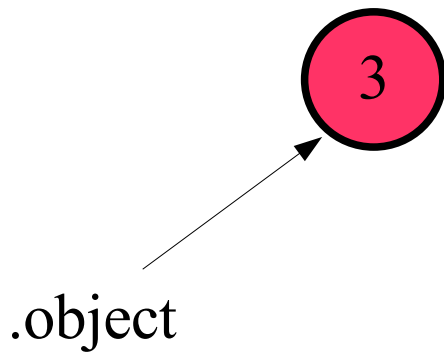
Now to “pop” the root node.  
Move “tail” object to root.

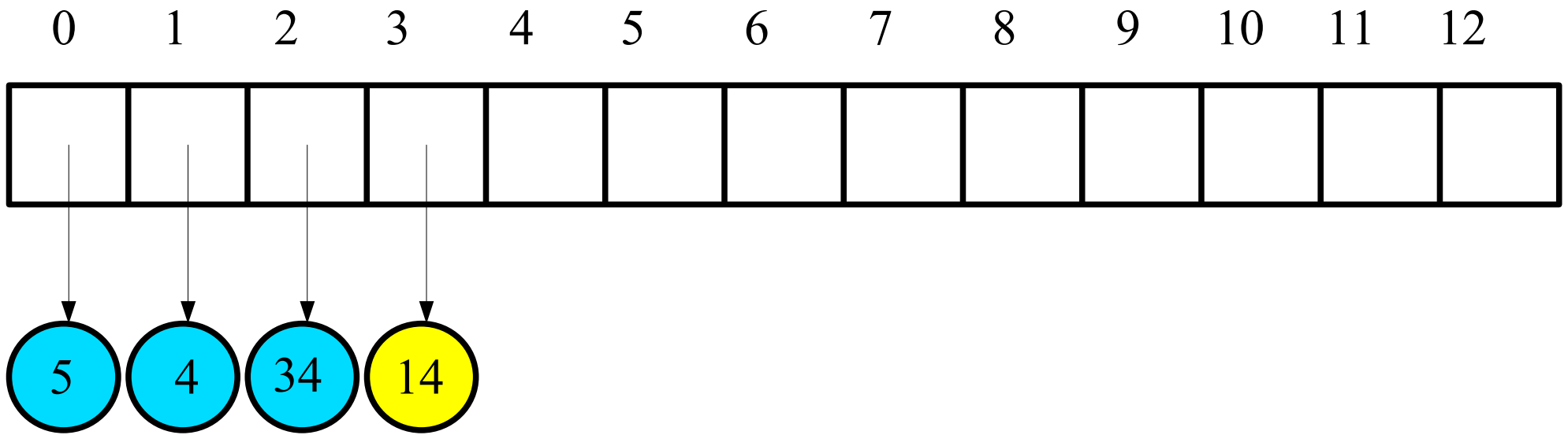




root = 0  
tail = 4

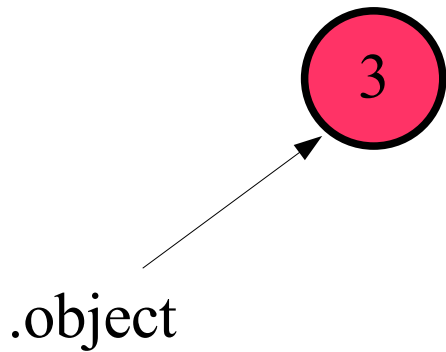
Now to “pop” the root node.  
Move “tail” object to root.  
Change the “tail” index

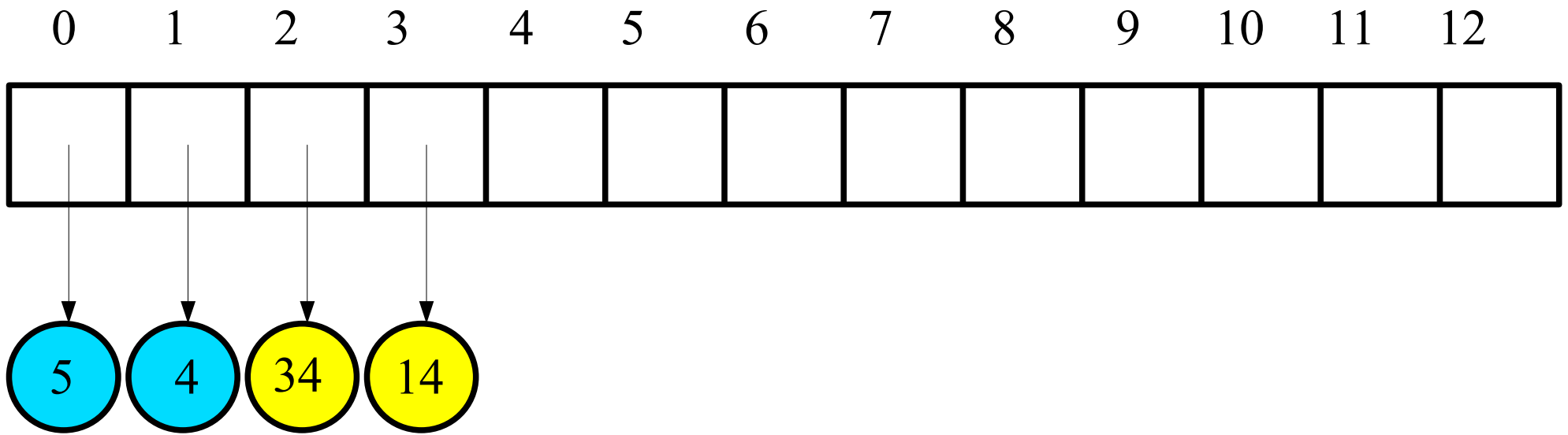




root = 0  
tail = 4

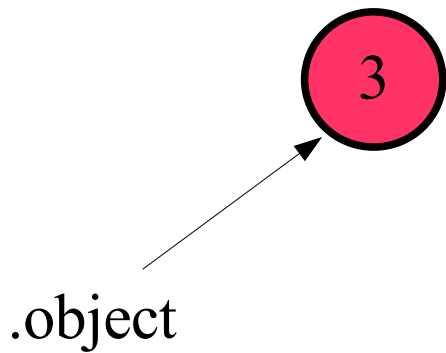
Now to “pop” the root node.  
Move “tail” object to root.  
Change the “tail” index  
Compare root with two children

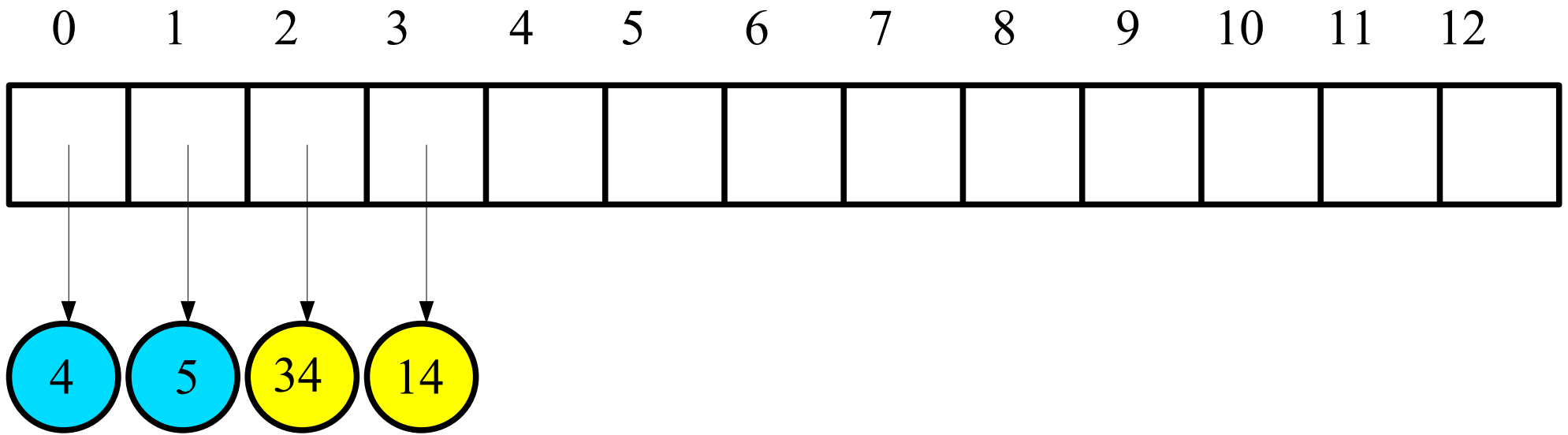




root = 0  
tail = 4

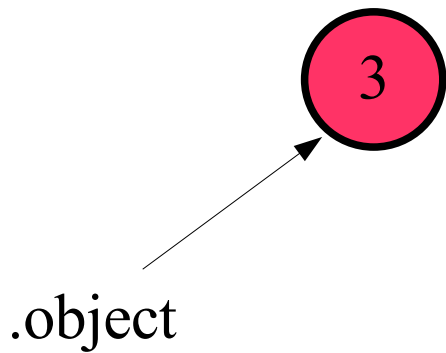
Now to “pop” the root node.  
Move “tail” object to root.  
Change the “tail” index  
Compare root with two children  
If greater, swap with smallest of the children

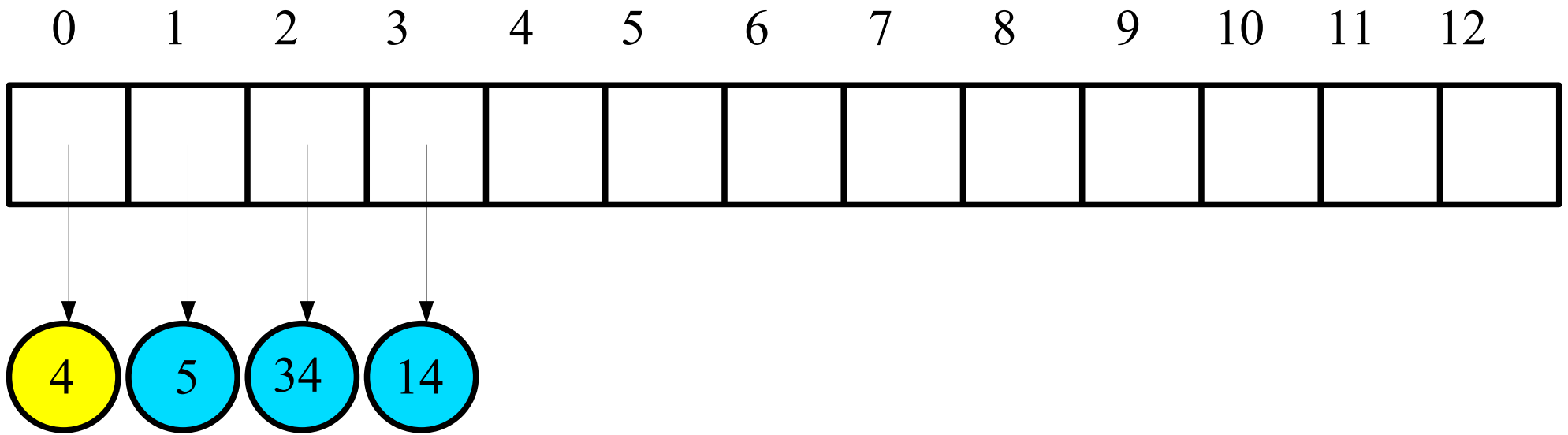




root = 0  
tail = 4

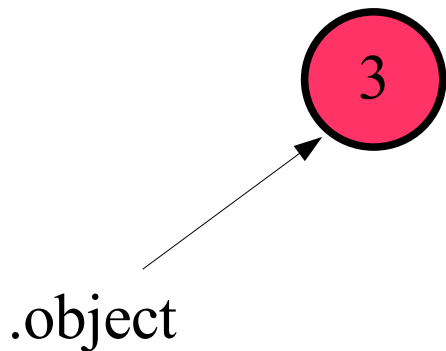
Now to “pop” the root node.  
Move “tail” object to root.  
Change the “tail” index  
Compare root with two children  
If greater, swap with smallest of the children



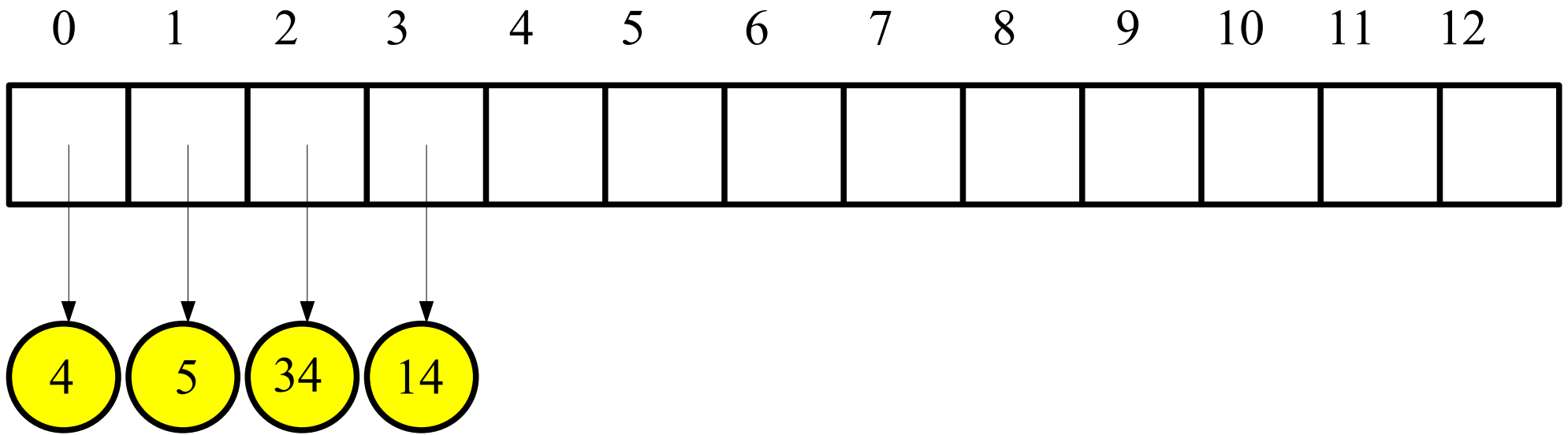


root = 0  
tail = 4

Now to “pop” the root node.  
 Move “tail” object to root.  
 Change the “tail” index  
 Compare root with two children  
 If greater, swap with smallest of the children  
 Compare swapped object with children  
 at indices  $1*2+1=3$  and  $1*2+2=4$

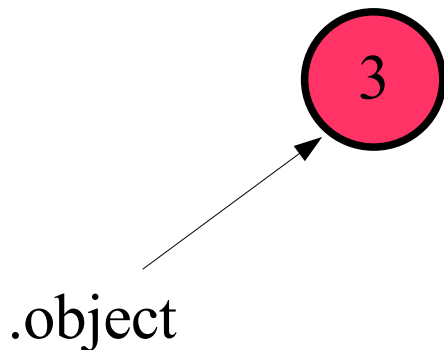


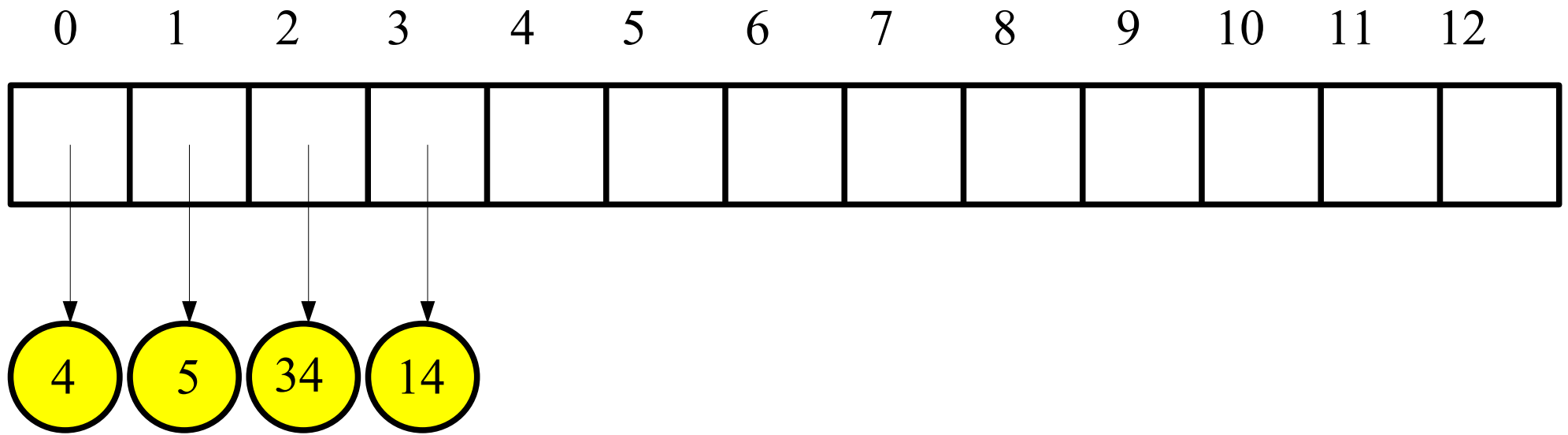




root = 0  
tail = 4

Now to “pop” the root node.  
 Move “tail” object to root.  
 Change the “tail” index  
 Compare root with two children  
 If greater, swap with smallest of the children  
 Compare swapped object with children  
     at indices  $1*2+1=3$  and  $1*2+2=4$   
 Node is smaller than both: stop





root = 0  
tail = 4

Now to “pop” the root node.  
Move “tail” object to root.  
Change the “tail” index  
Compare root with two children  
If greater, swap with smallest of the children  
Compare swapped object with children  
    at indices  $1*2+1=3$  and  $1*2+2=4$   
Node is smaller than both: stop  
Return the object