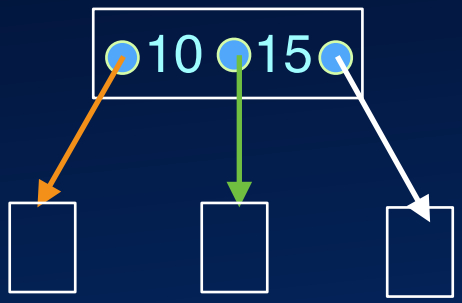
Data Structures & Algorithms

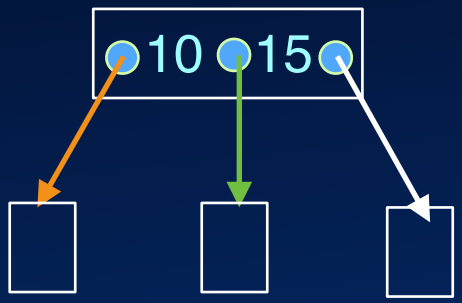
Subodh Kumar
(subodh@iitd.ac.in, Bharti 422)
Dept of Computer Sc. & Engg.

Binar-izing a 2-4 Tree

Binar-izing a 2-4 Tree

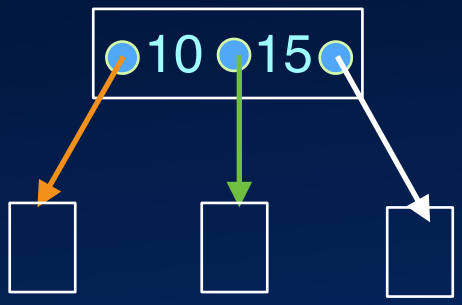


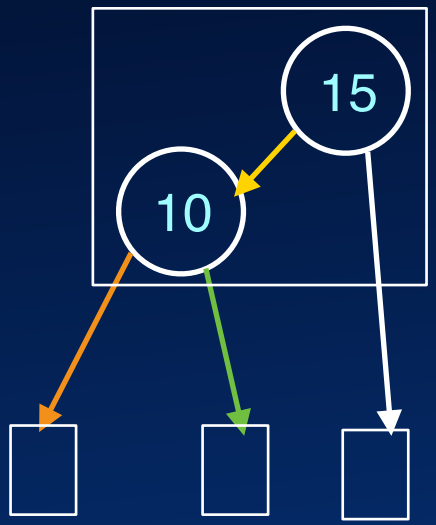
Binar-izing a 2-4 Tree



$\lg_3 n = .63 \lg_2 n$

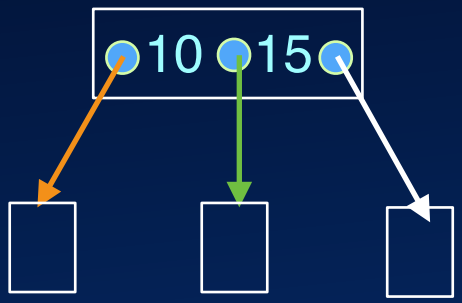
Binar-izing a 2-4 Tree

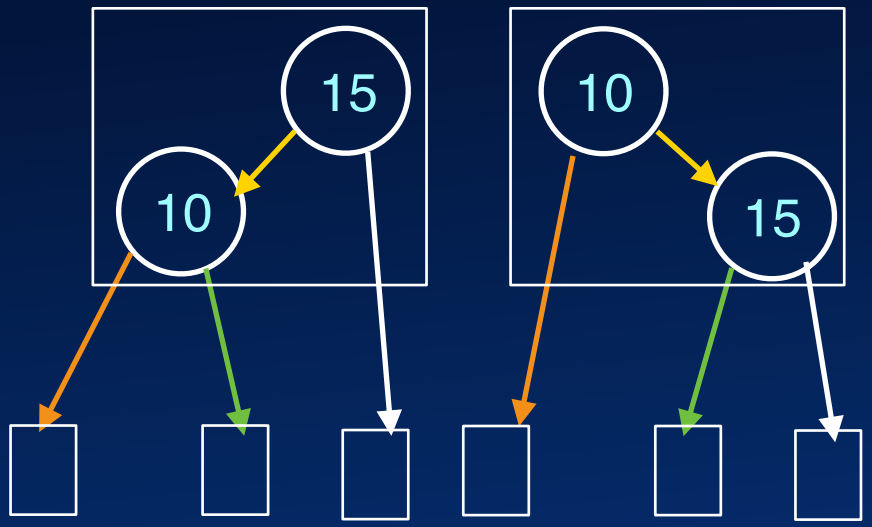




$\lg_3 n = .63 \lg_2 n$

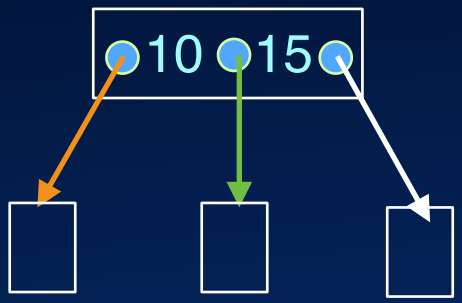
Binar-izing a 2-4 Tree

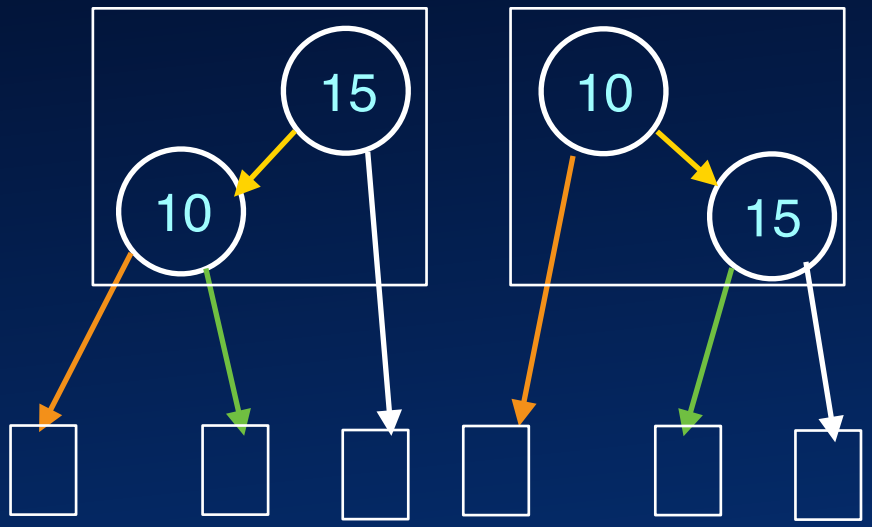




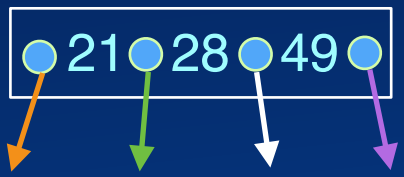
$\lg_3 n = .63 \lg_2 n$

Binar-izing a 2-4 Tree

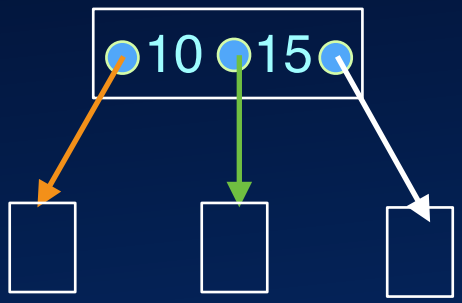


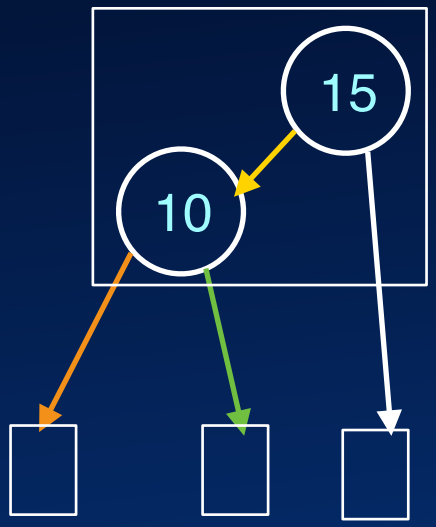


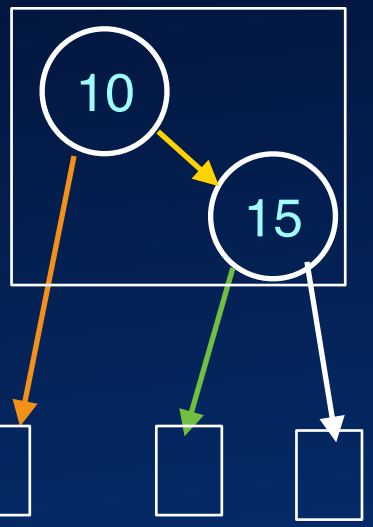
$\lg_3 n = .63 \lg_2 n$



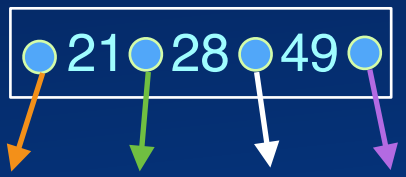
Binar-izing a 2-4 Tree

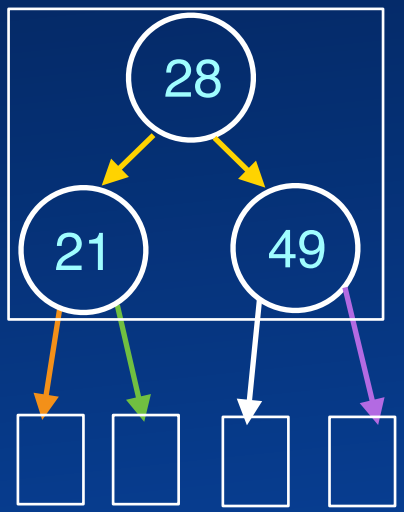




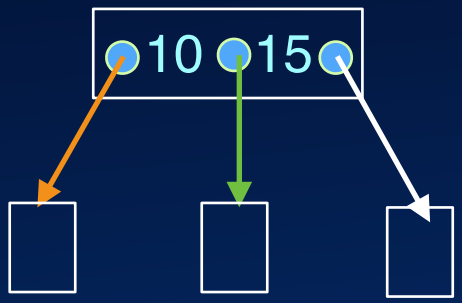


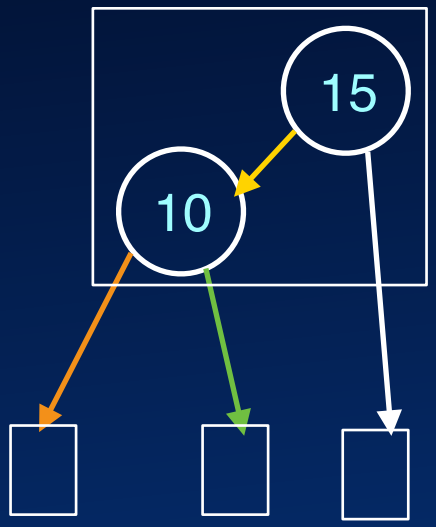
$\lg_3 n = .63 \lg_2 n$

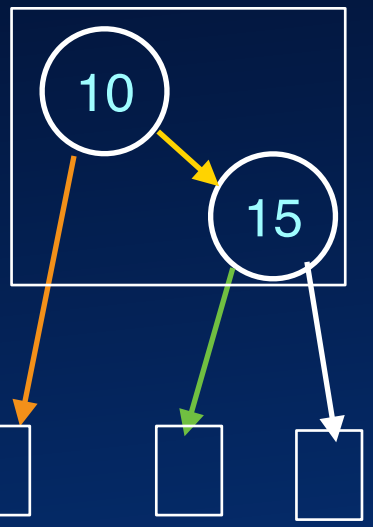




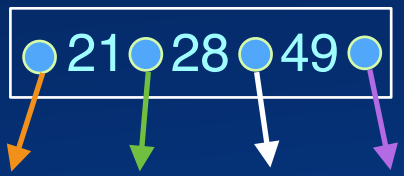
Binar-izing a 2-4 Tree

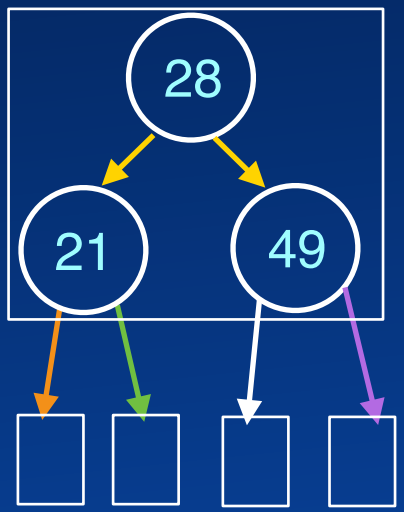


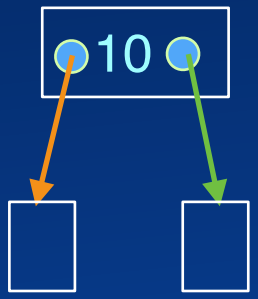




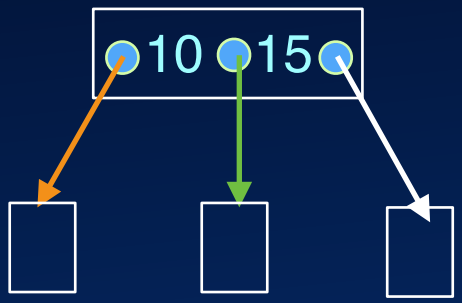
$\lg_3 n = .63 \lg_2 n$

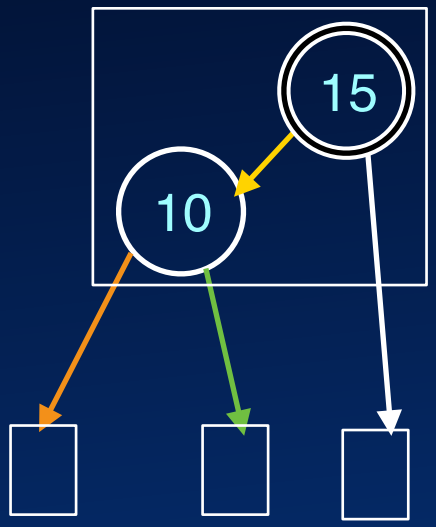


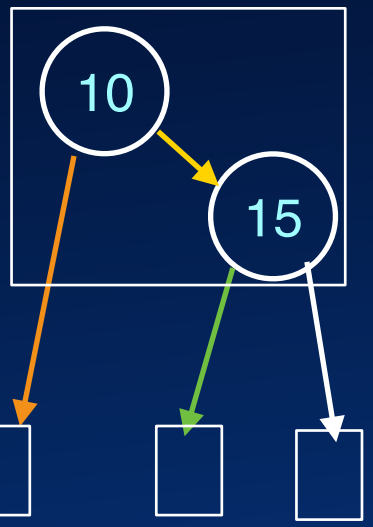




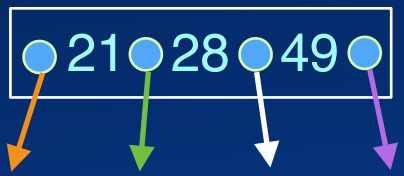
Binar-izing a 2-4 Tree

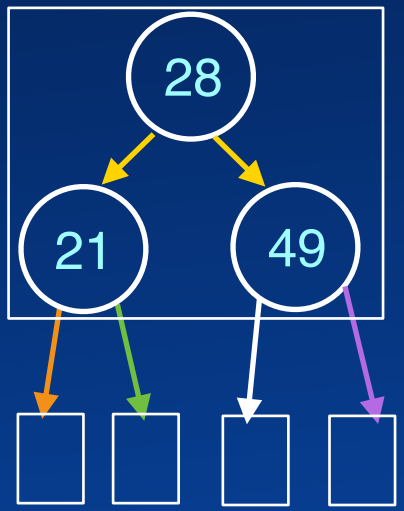


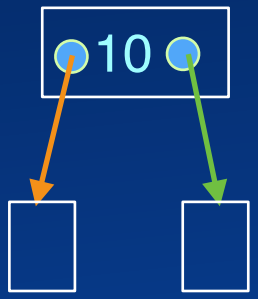




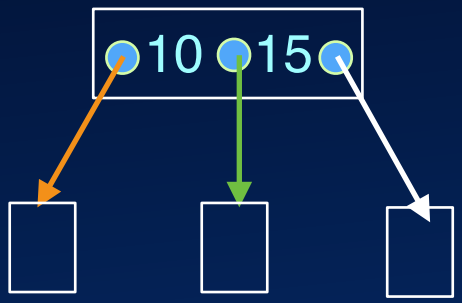
$\lg_3 n = .63 \lg_2 n$

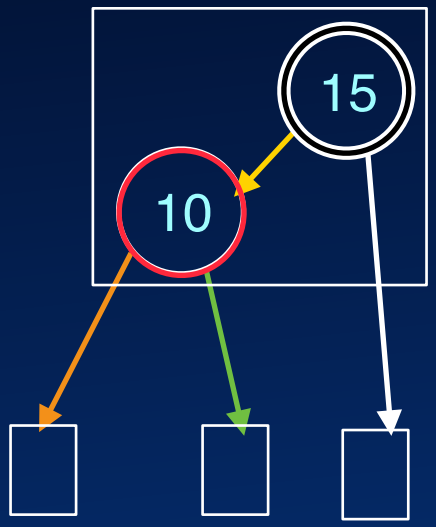


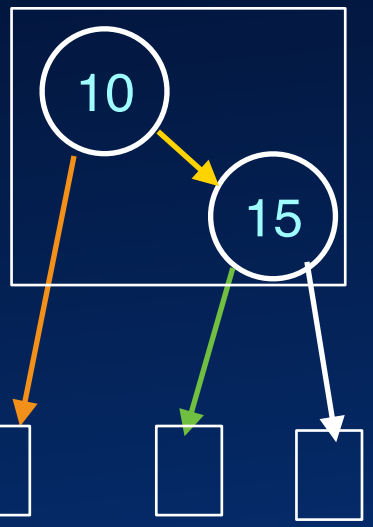




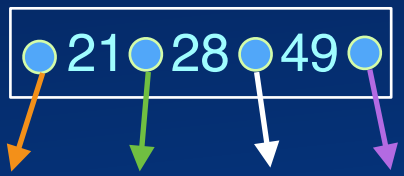
Binar-izing a 2-4 Tree

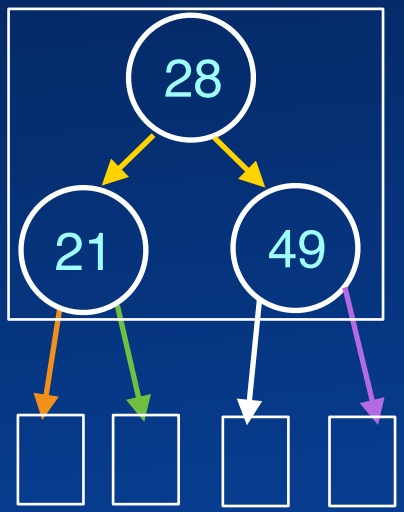


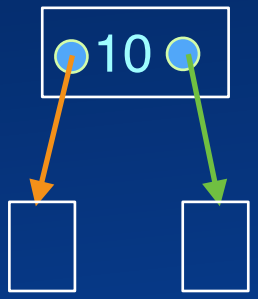




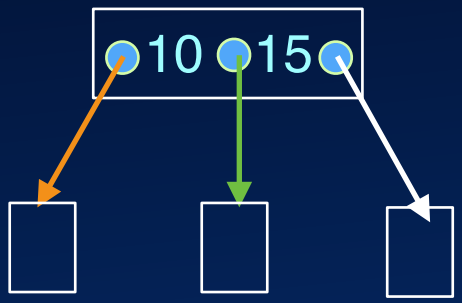
$\lg_3 n = .63 \lg_2 n$

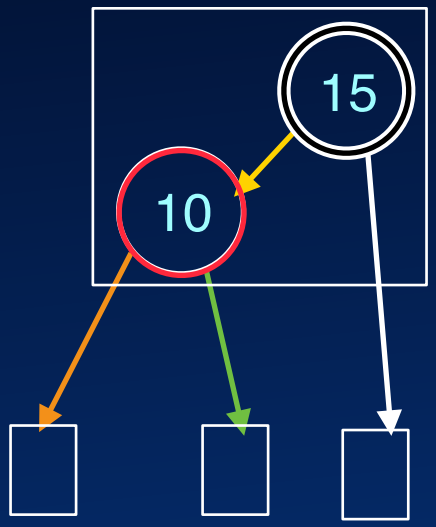


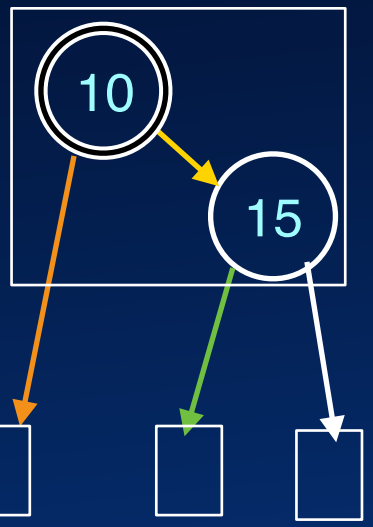




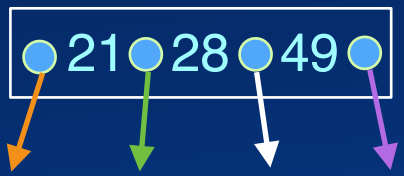
Binar-izing a 2-4 Tree

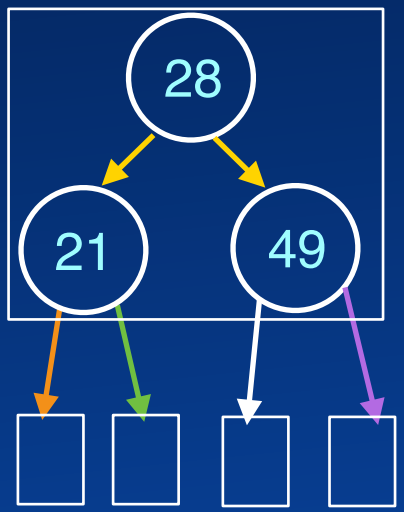


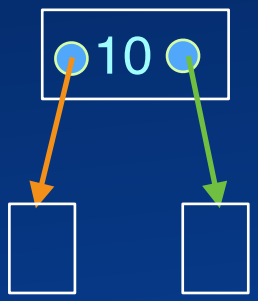




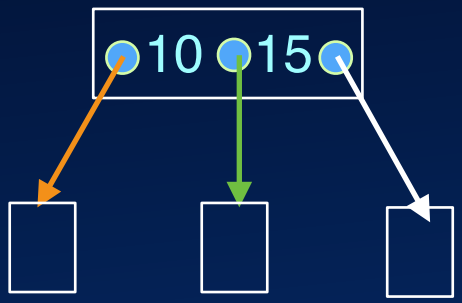
$\lg_3 n = .63 \lg_2 n$

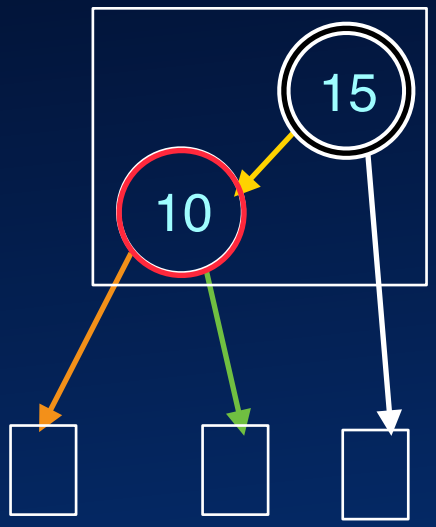


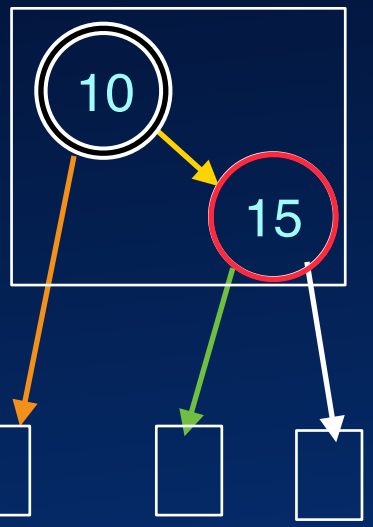




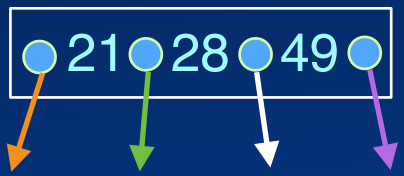
Binar-izing a 2-4 Tree

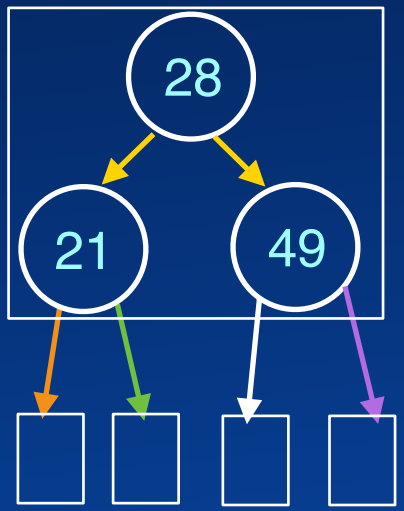


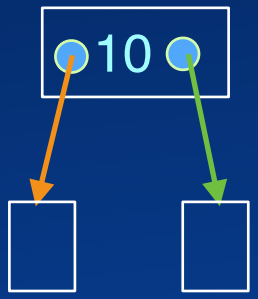




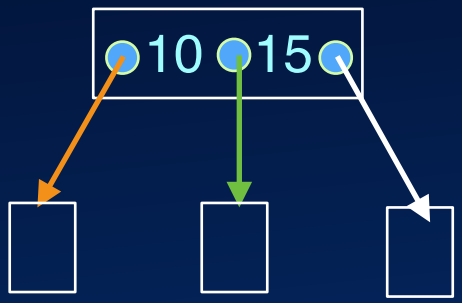
$\lg_3 n = .63 \lg_2 n$

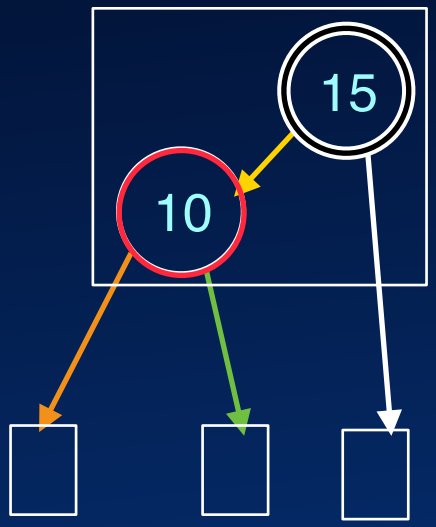


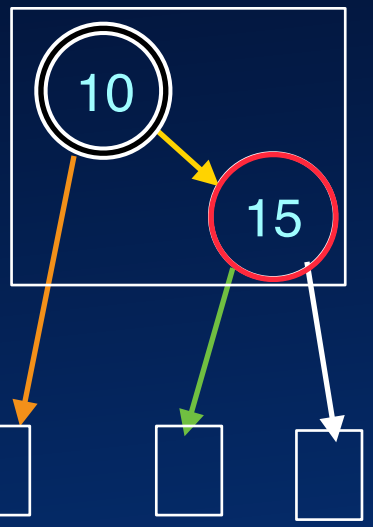




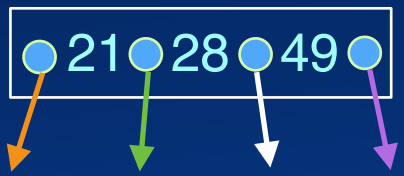
Binar-izing a 2-4 Tree

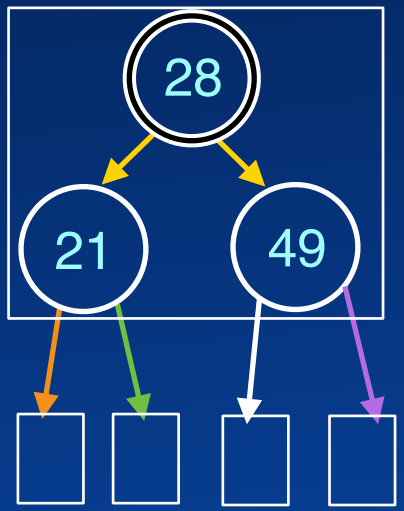


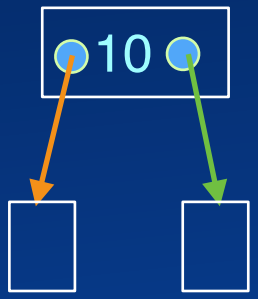




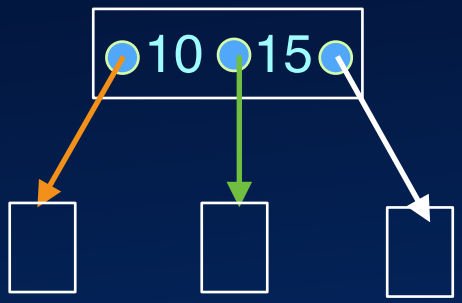
$\lg_3 n = .63 \lg_2 n$

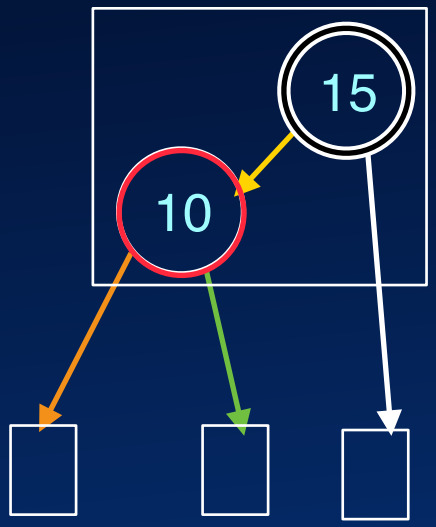


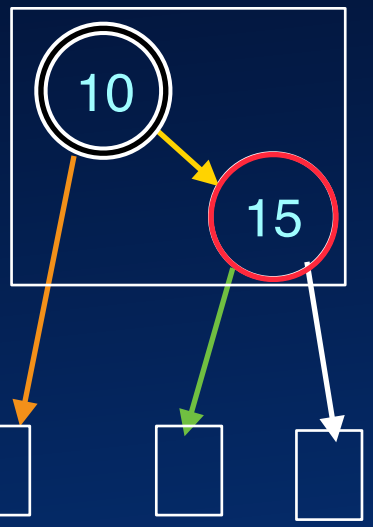




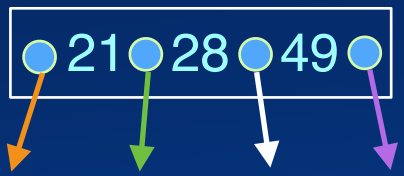
Binar-izing a 2-4 Tree

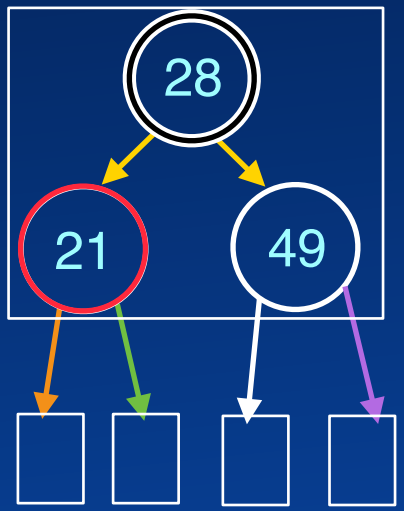


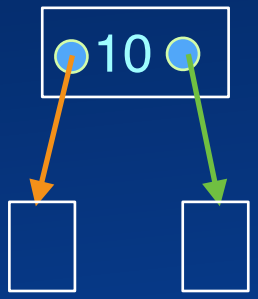




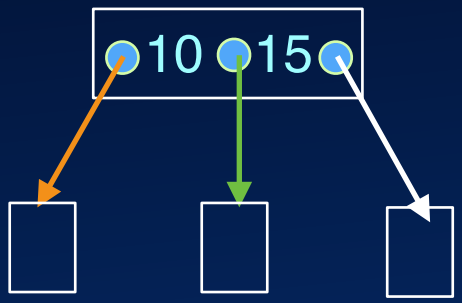
$\lg_3 n = .63 \lg_2 n$

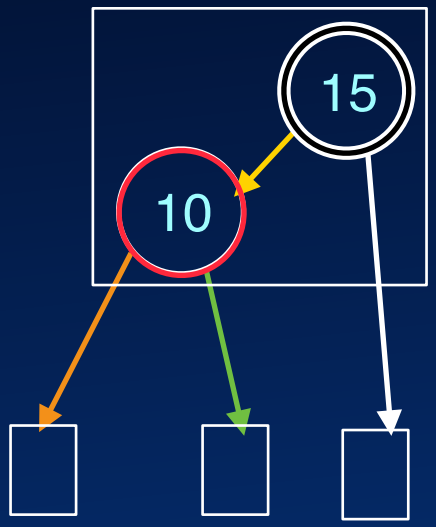


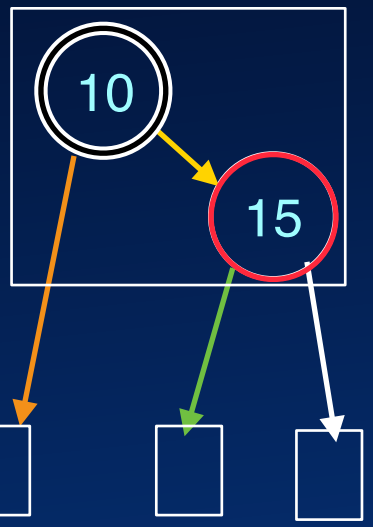




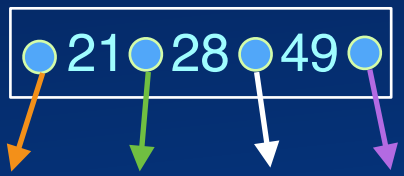
Binar-izing a 2-4 Tree

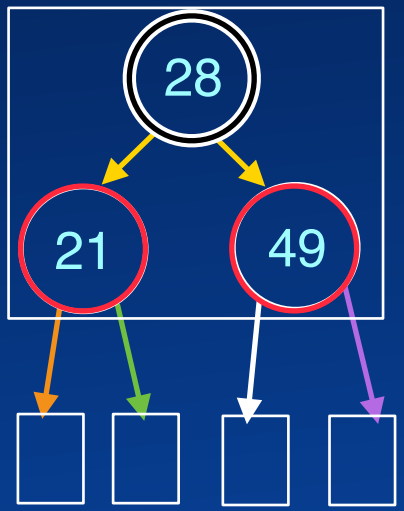


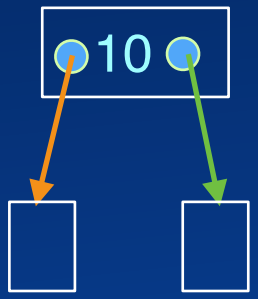




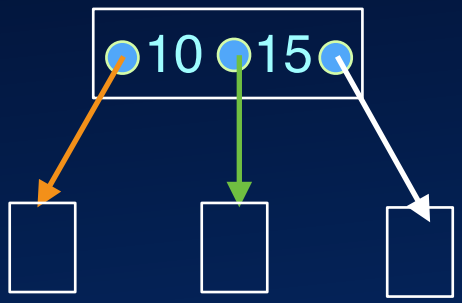
$\lg_3 n = .63 \lg_2 n$

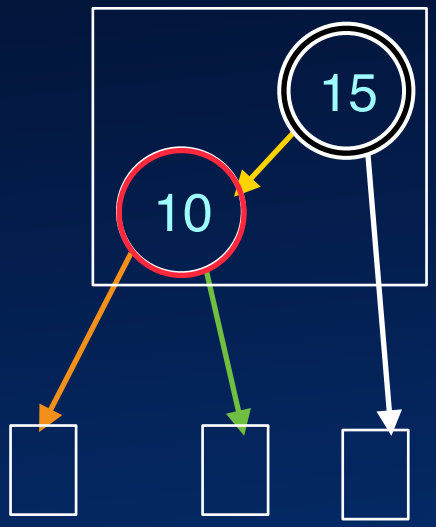


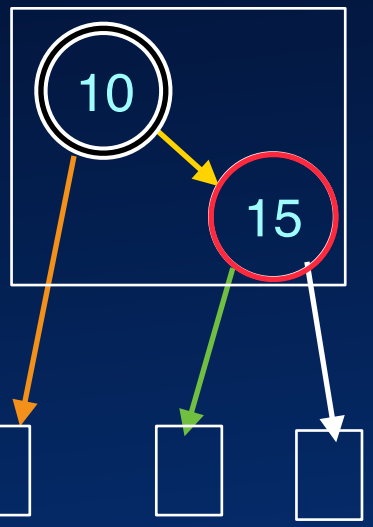




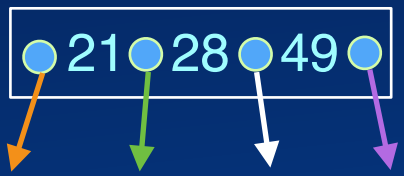
Binar-izing a 2-4 Tree

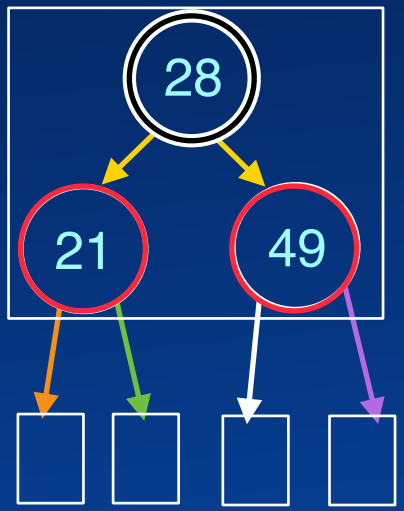


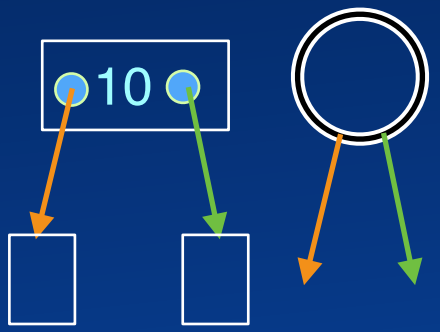




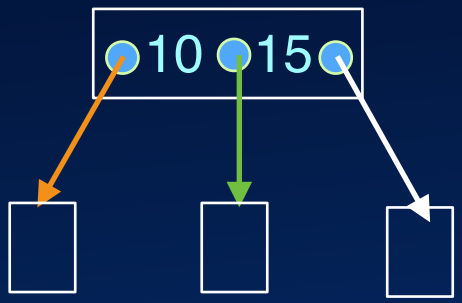
$\lg_3 n = .63 \lg_2 n$

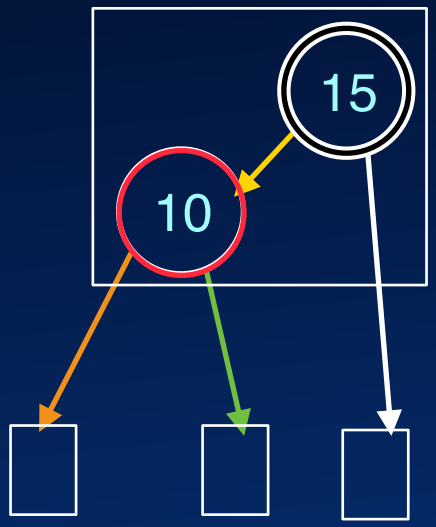


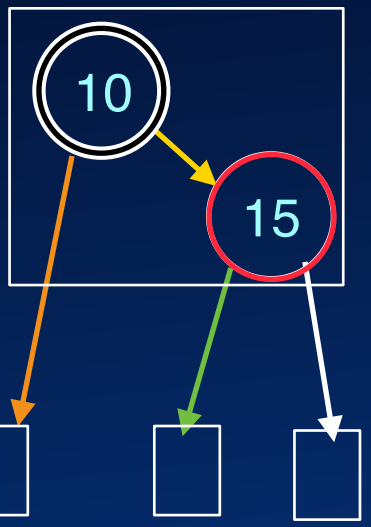




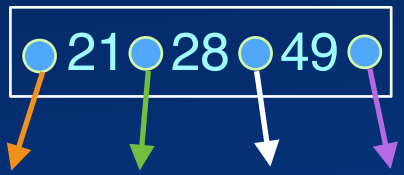
Binar-izing a 2-4 Tree



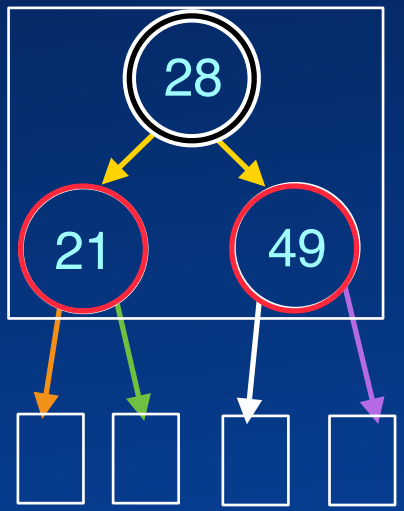


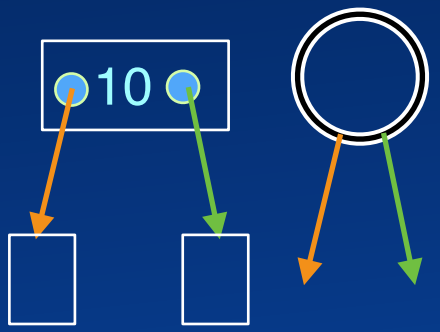


$\lg_3 n = .63 \lg_2 n$



Red Black Tree





Red Black Tree

- No red colored node has a red colored child
- The number of black colored node on the path from the root to each null reference is the same

Red Black Tree

- **No red colored node has a red colored child**
- **The number of black colored node on the path from the root to each null reference is the same**

$$\Rightarrow \text{height} \leq 2 \log (n)$$

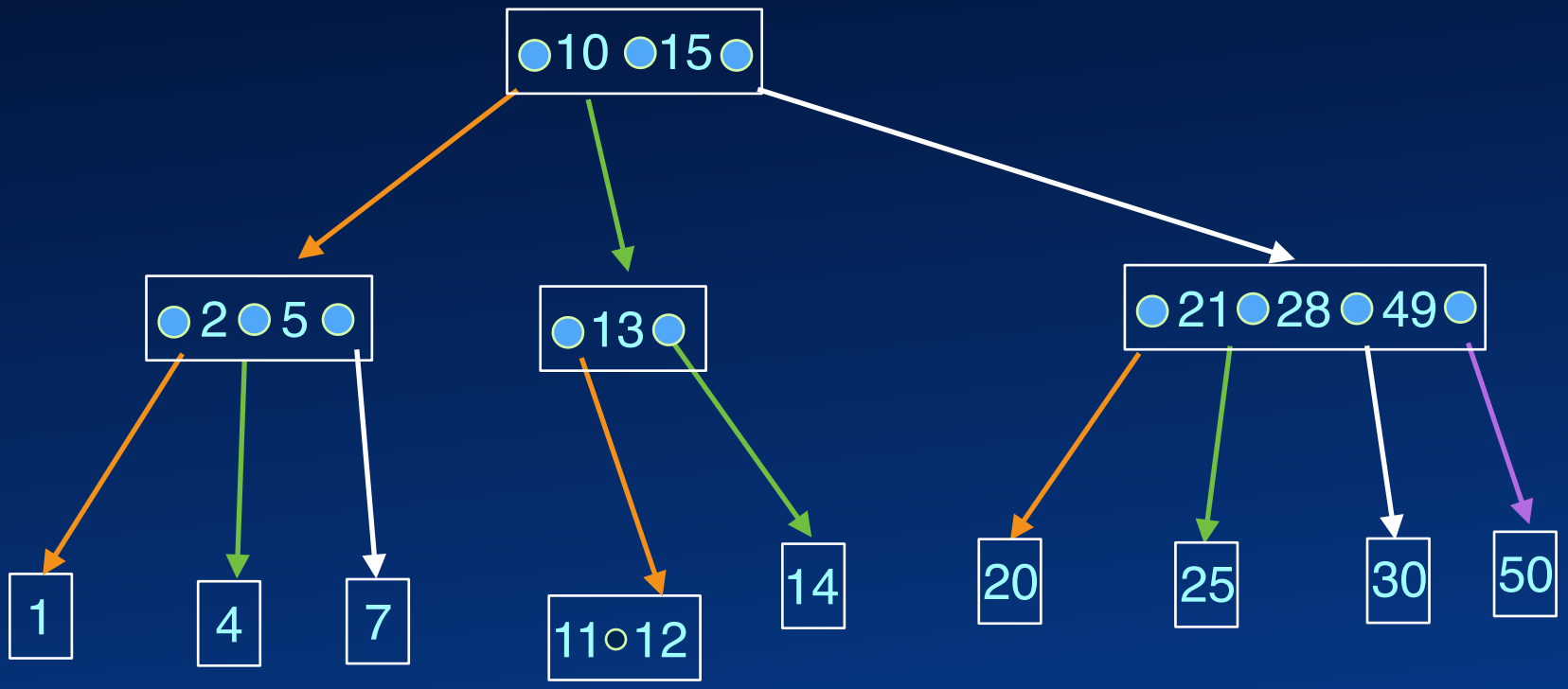
Red Black Tree

- **No red colored node has a red colored child**
- **The number of black colored node on the path from the root to each null reference is the same**

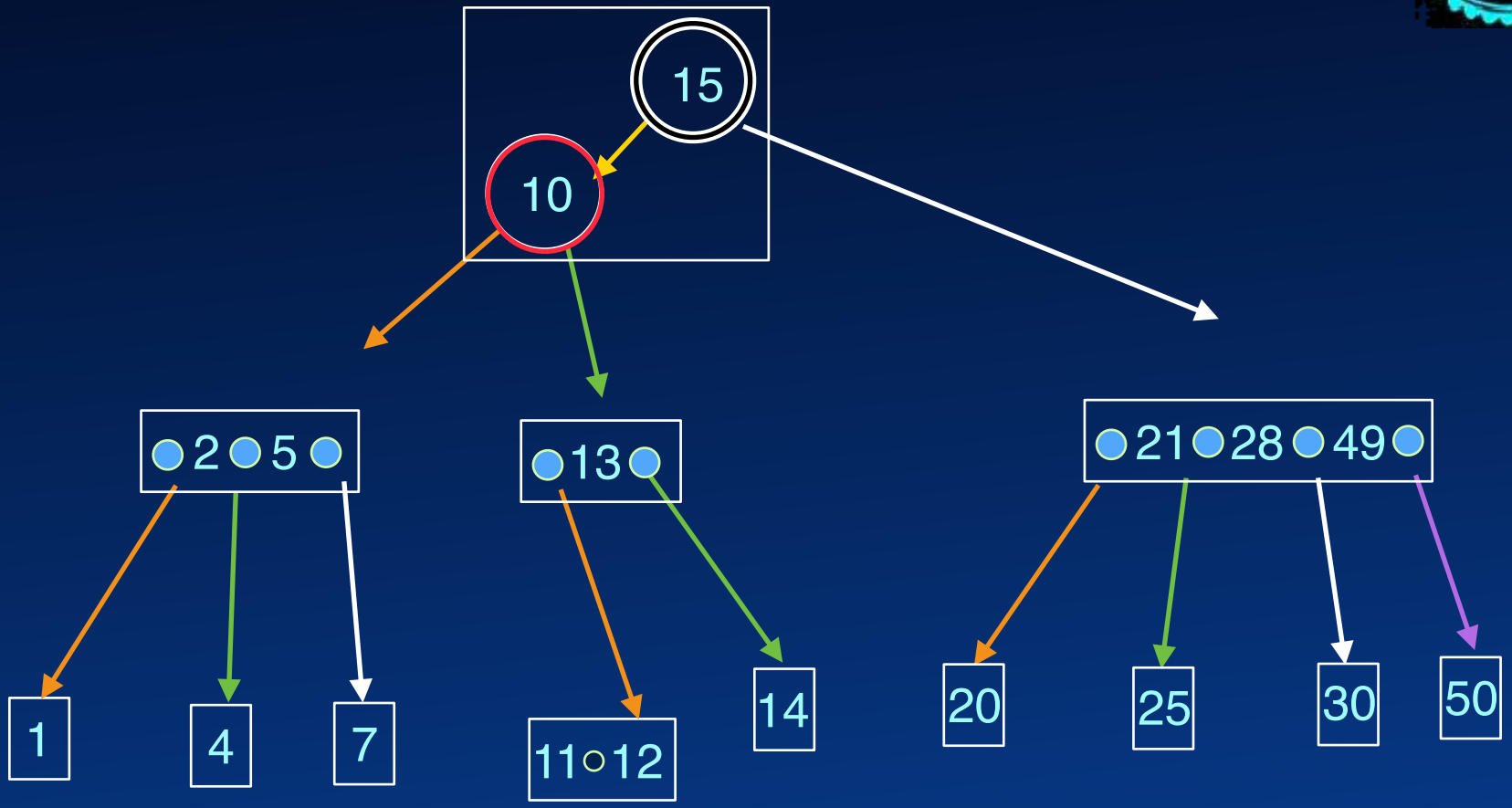
  => height ≤ 2 log (n)

  - Null references are assumed black
  - Root is always black

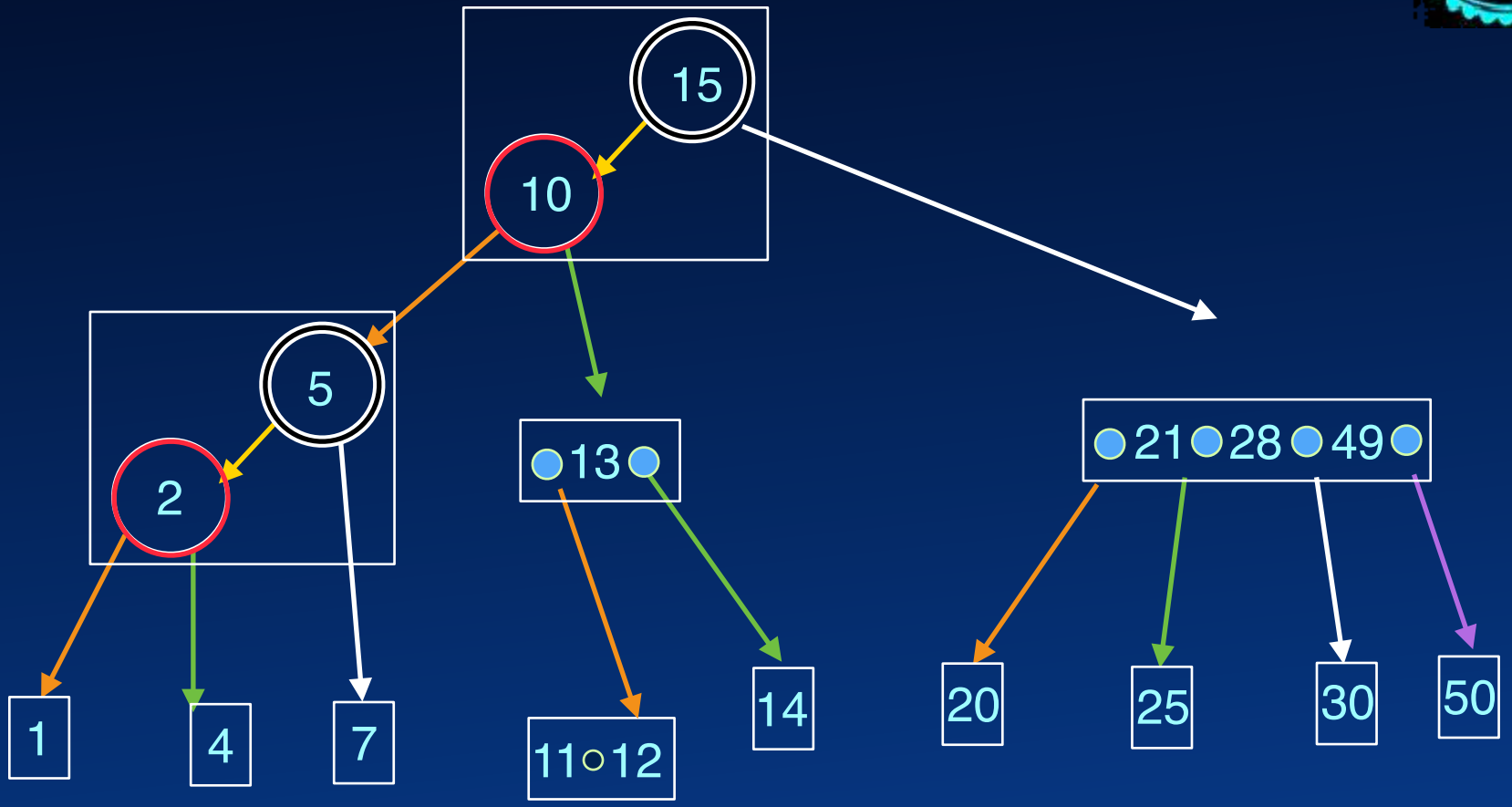
Binar-ized 2-4 Tree



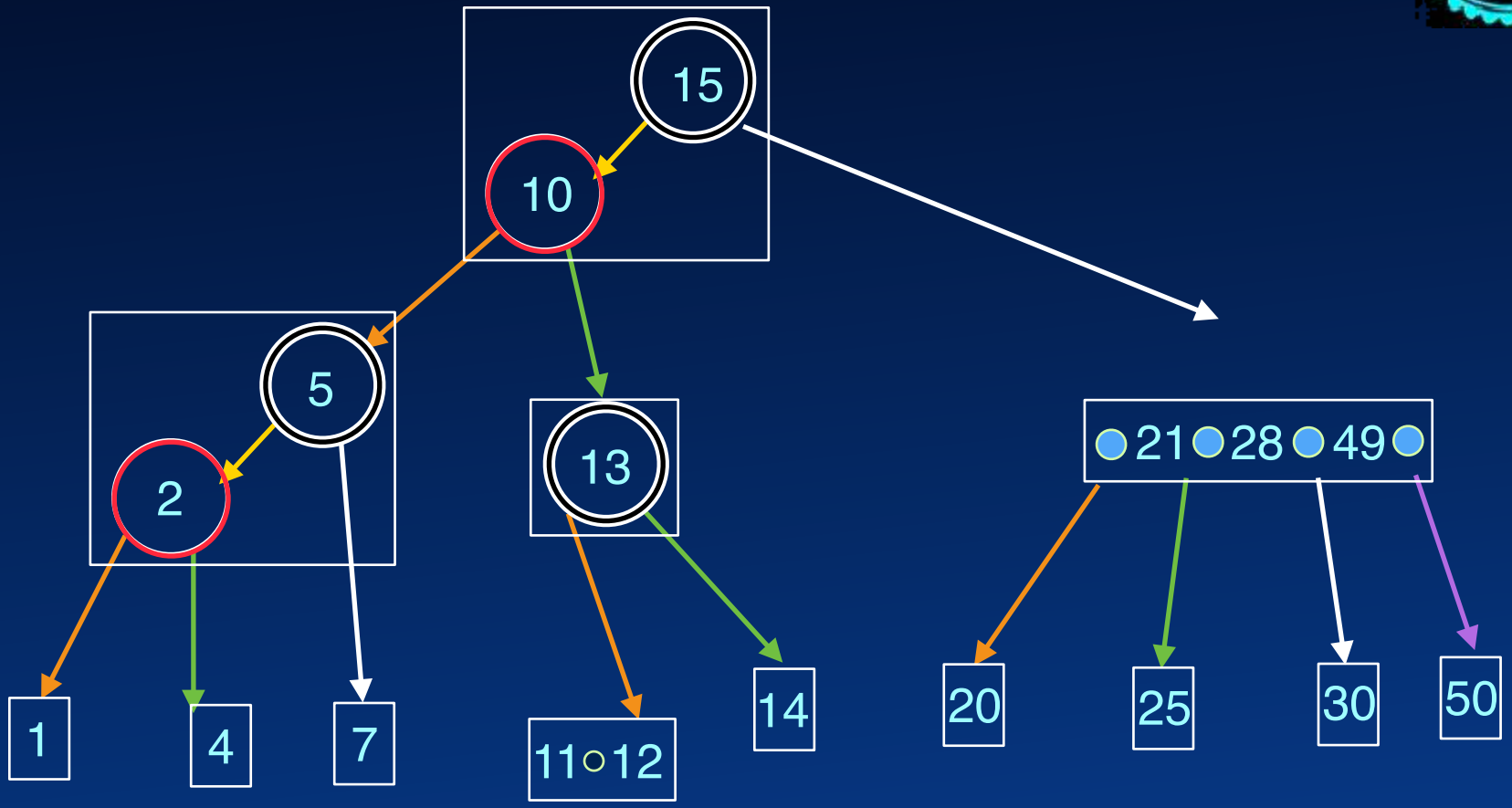
Binar-ized 2-4 Tree



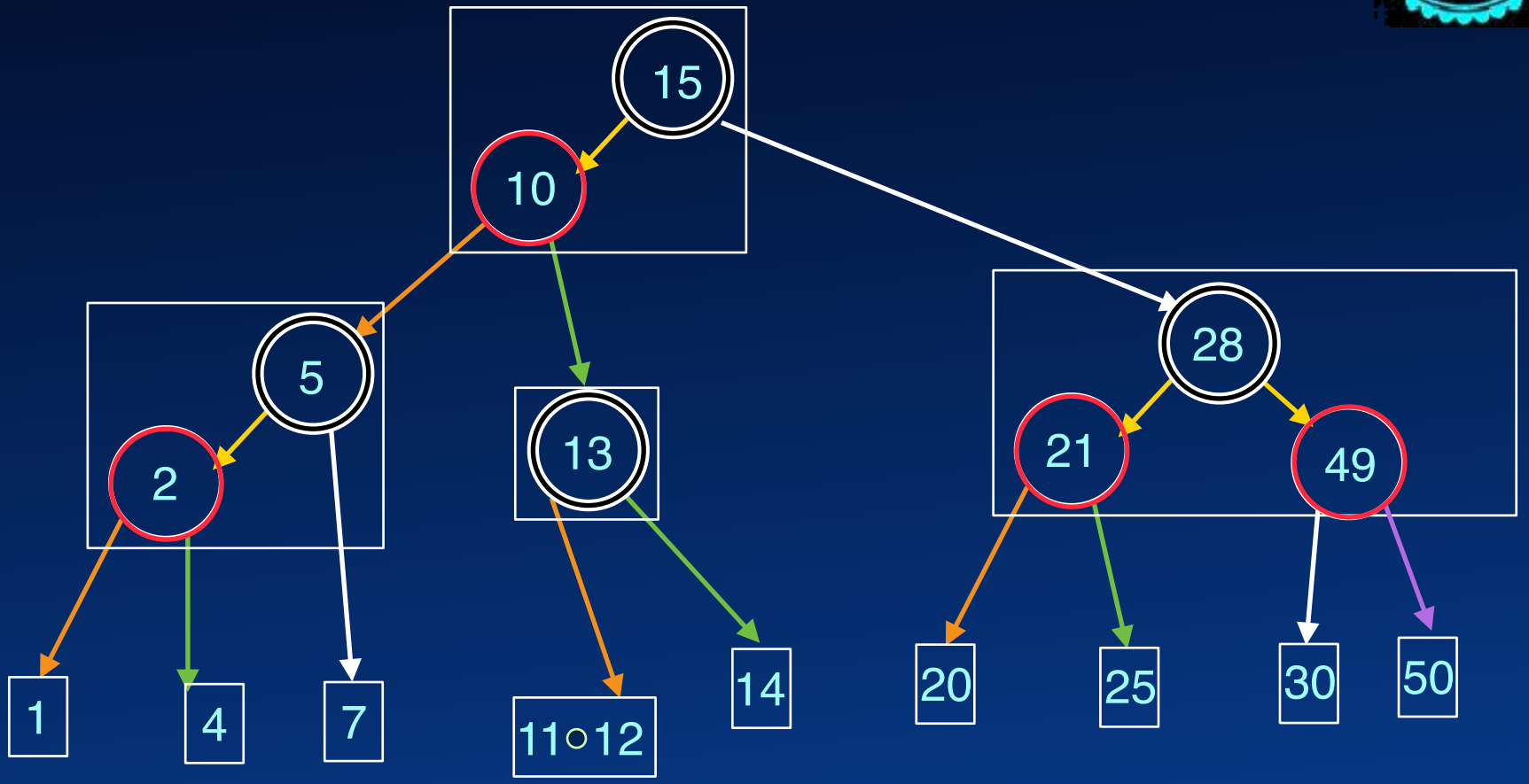
Binar-ized 2-4 Tree



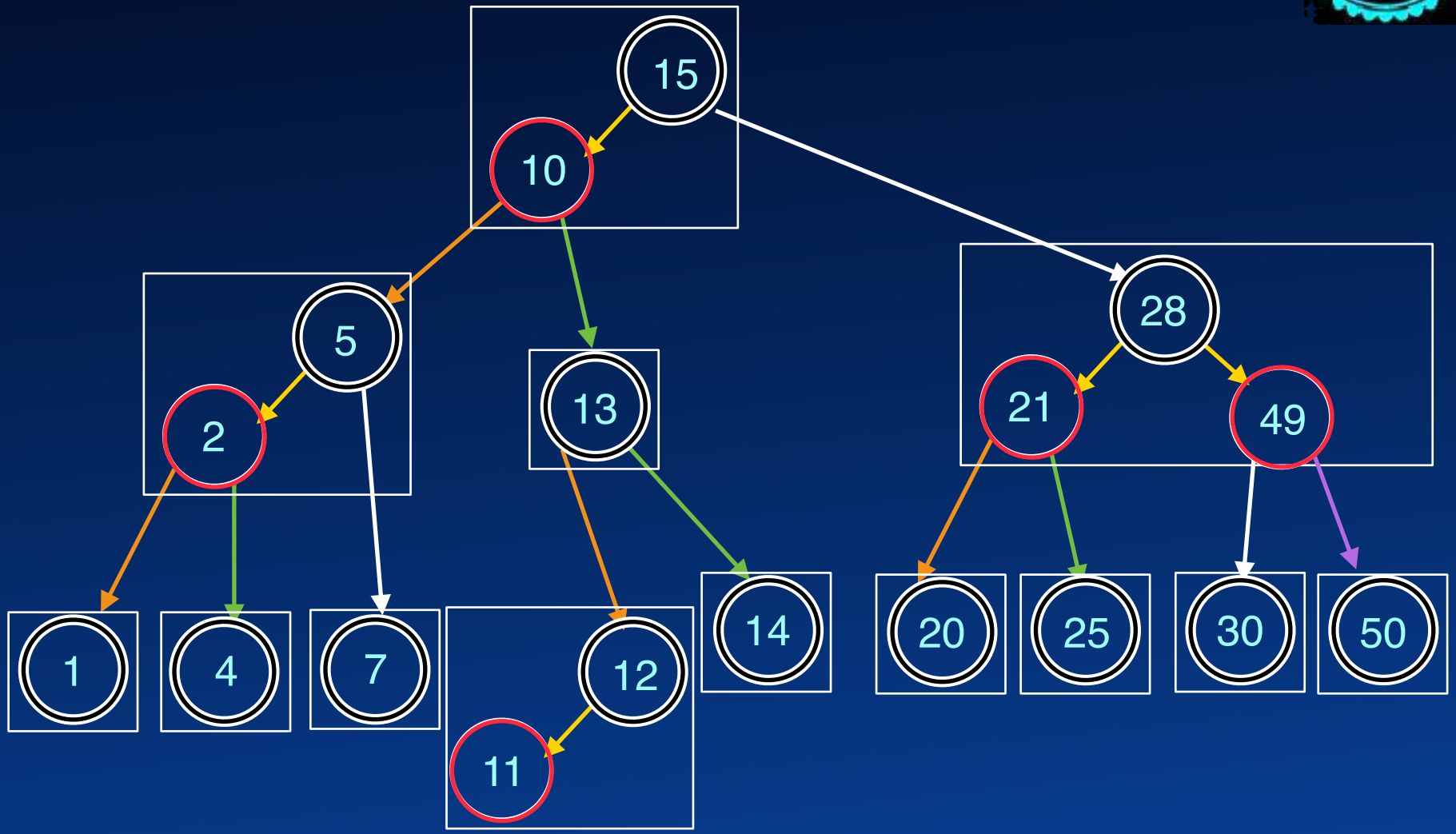
Binar-ized 2-4 Tree



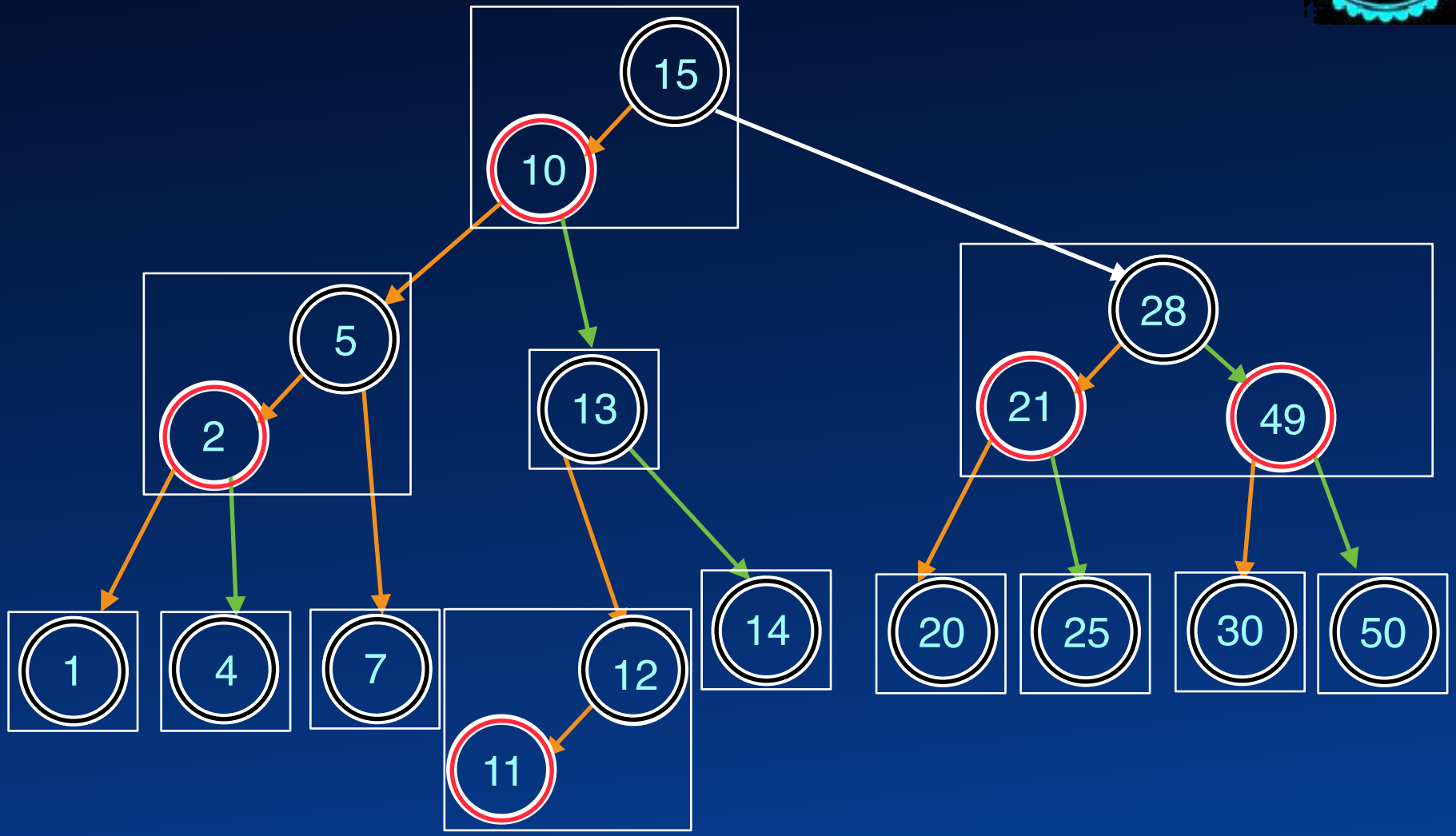
Binar-ized 2-4 Tree



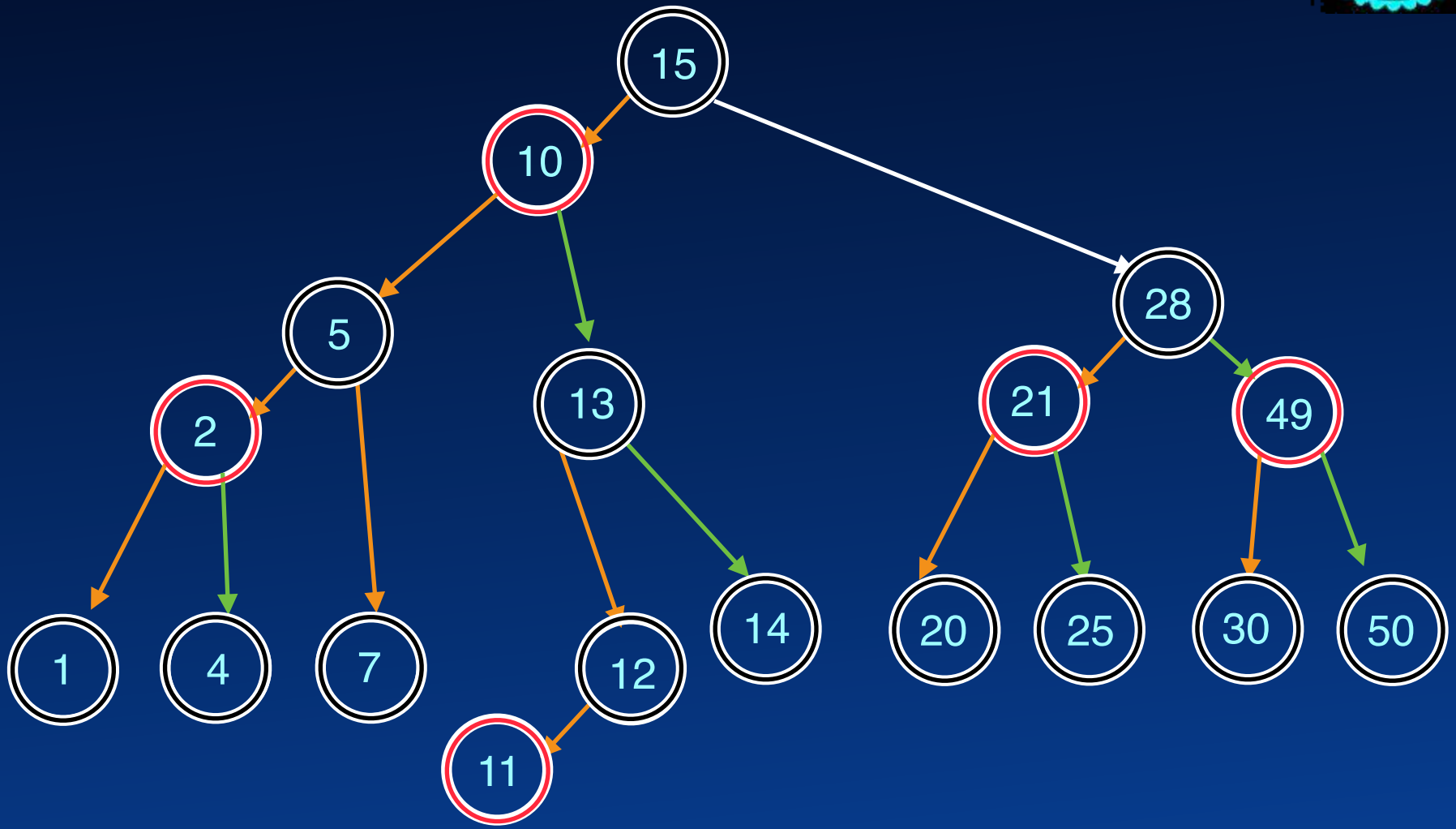
Binar-ized 2-4 Tree



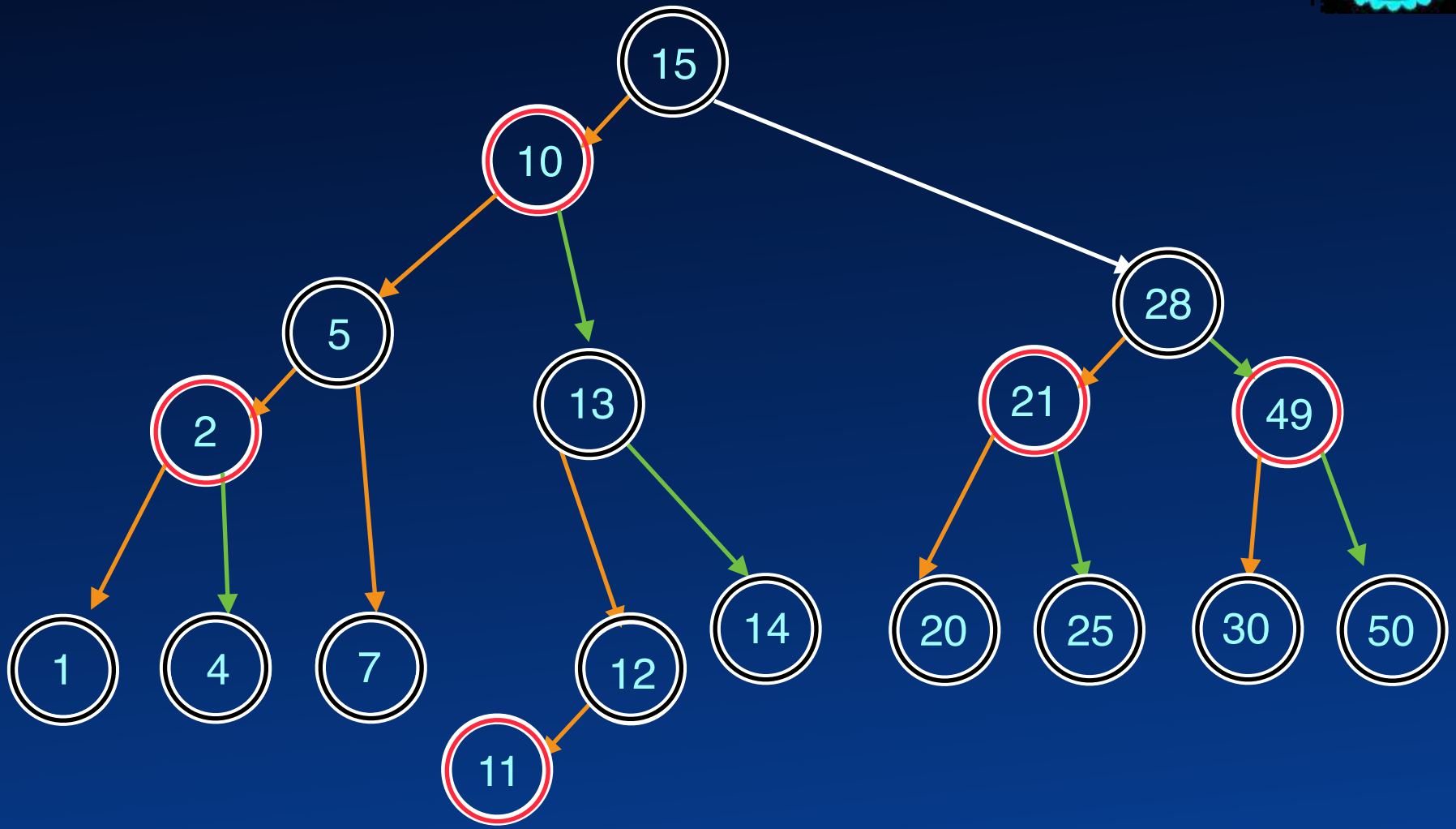
Red-Black Tree Insert



Red-Black Tree Insert

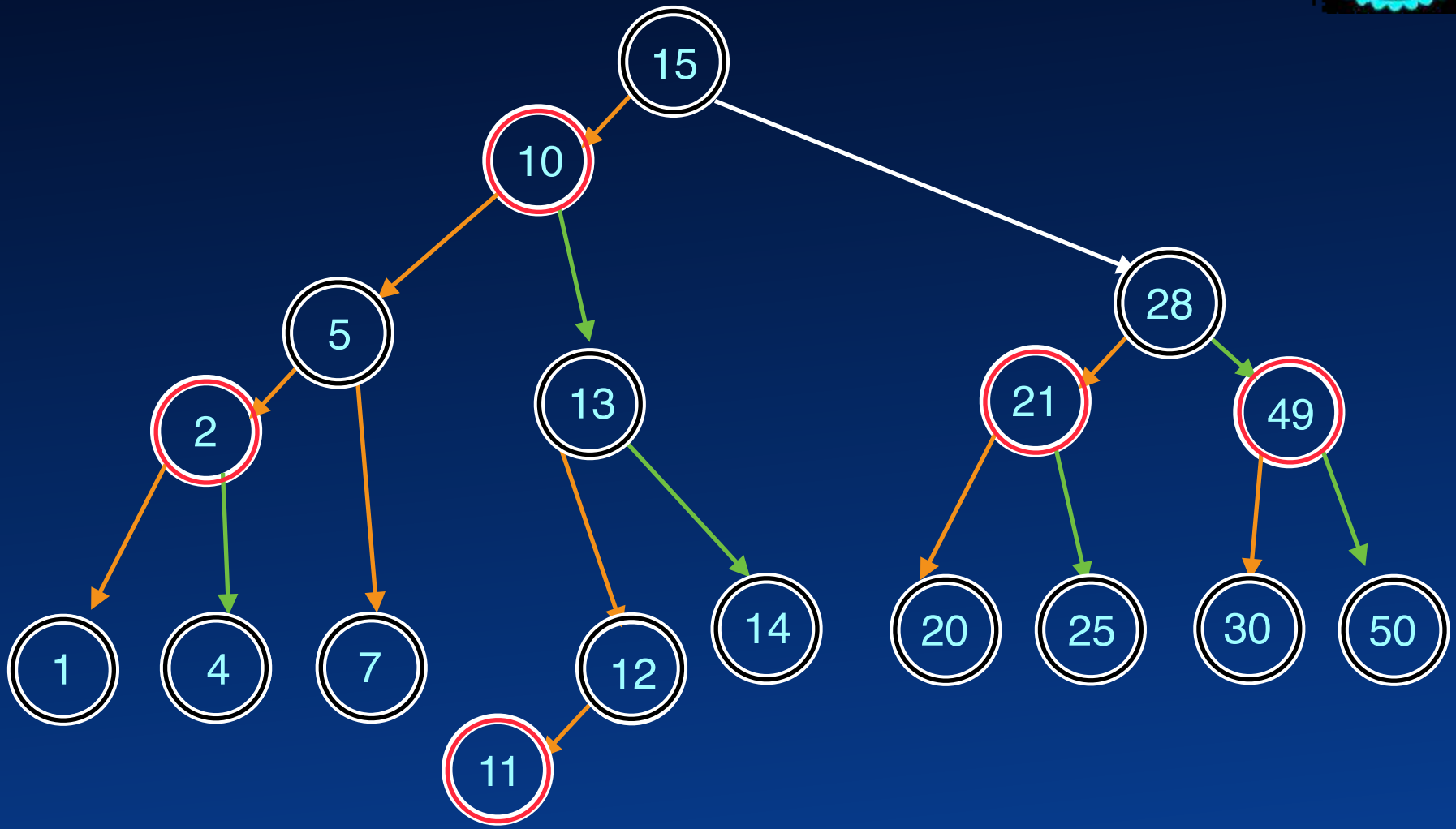


Red-Black Tree Insert



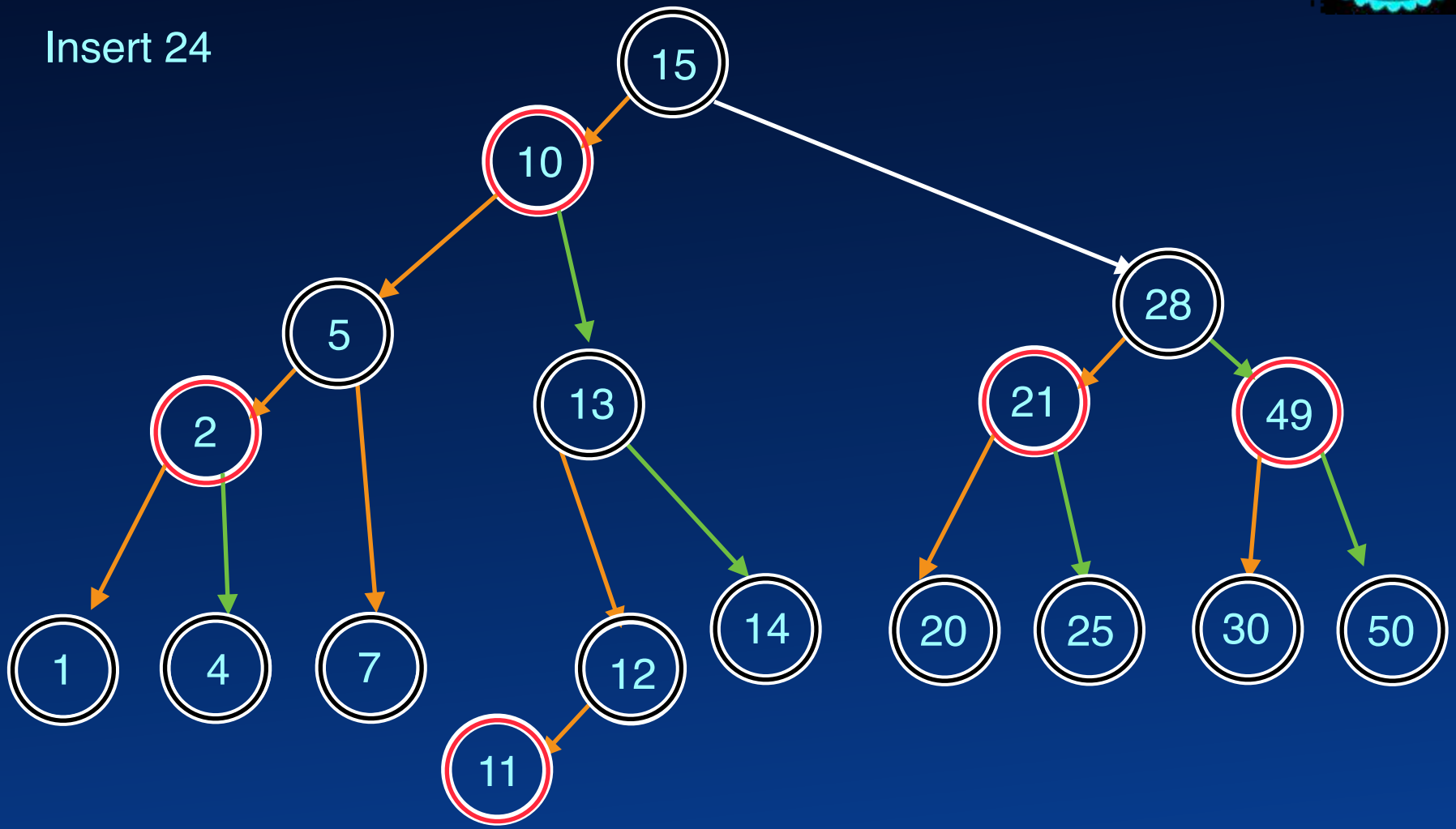
Insert

Red-Black Tree Insert



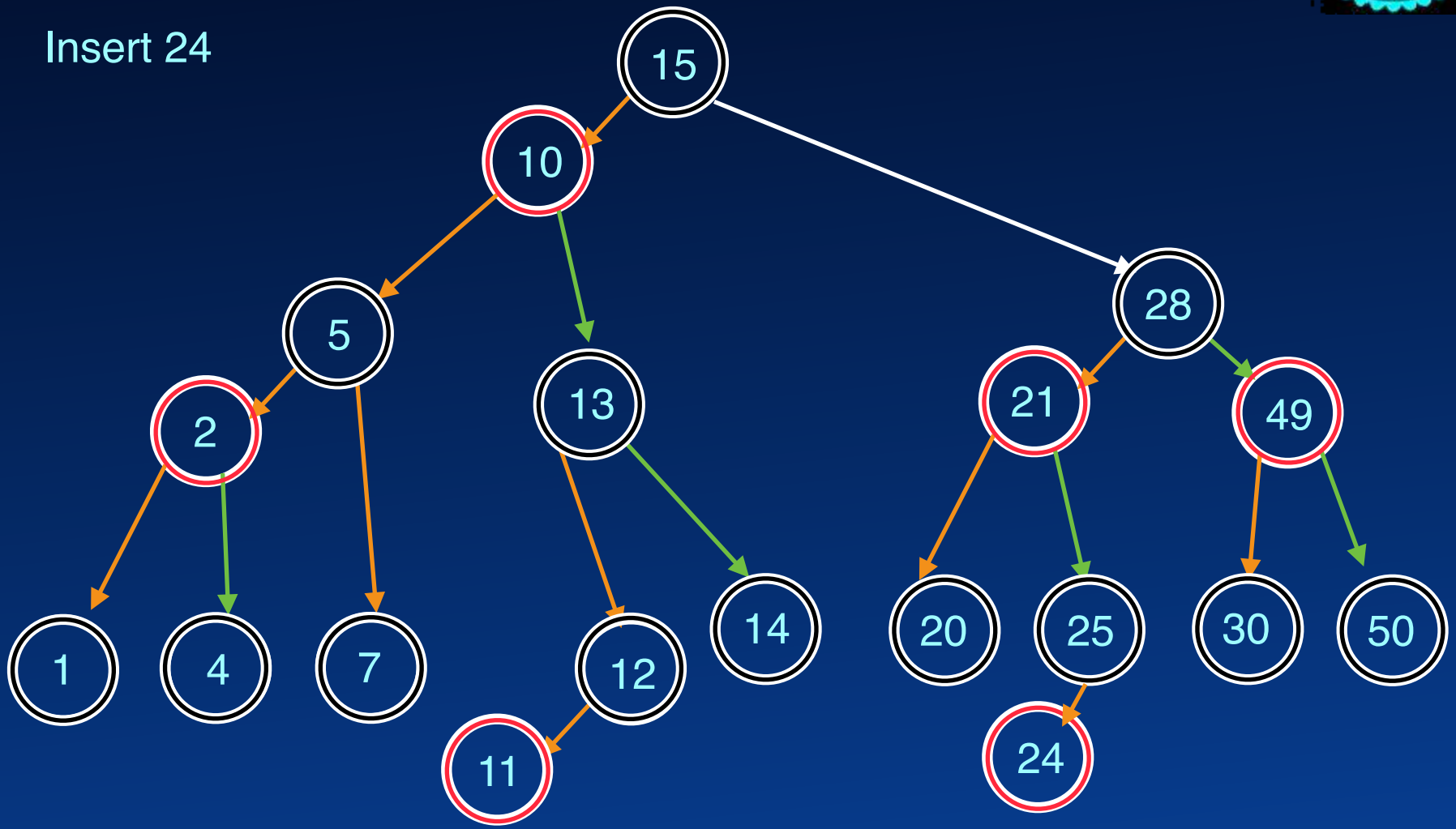
Insert @null in node ***n***

Red-Black Tree Insert



Insert @null in node ***n***

Red-Black Tree Insert

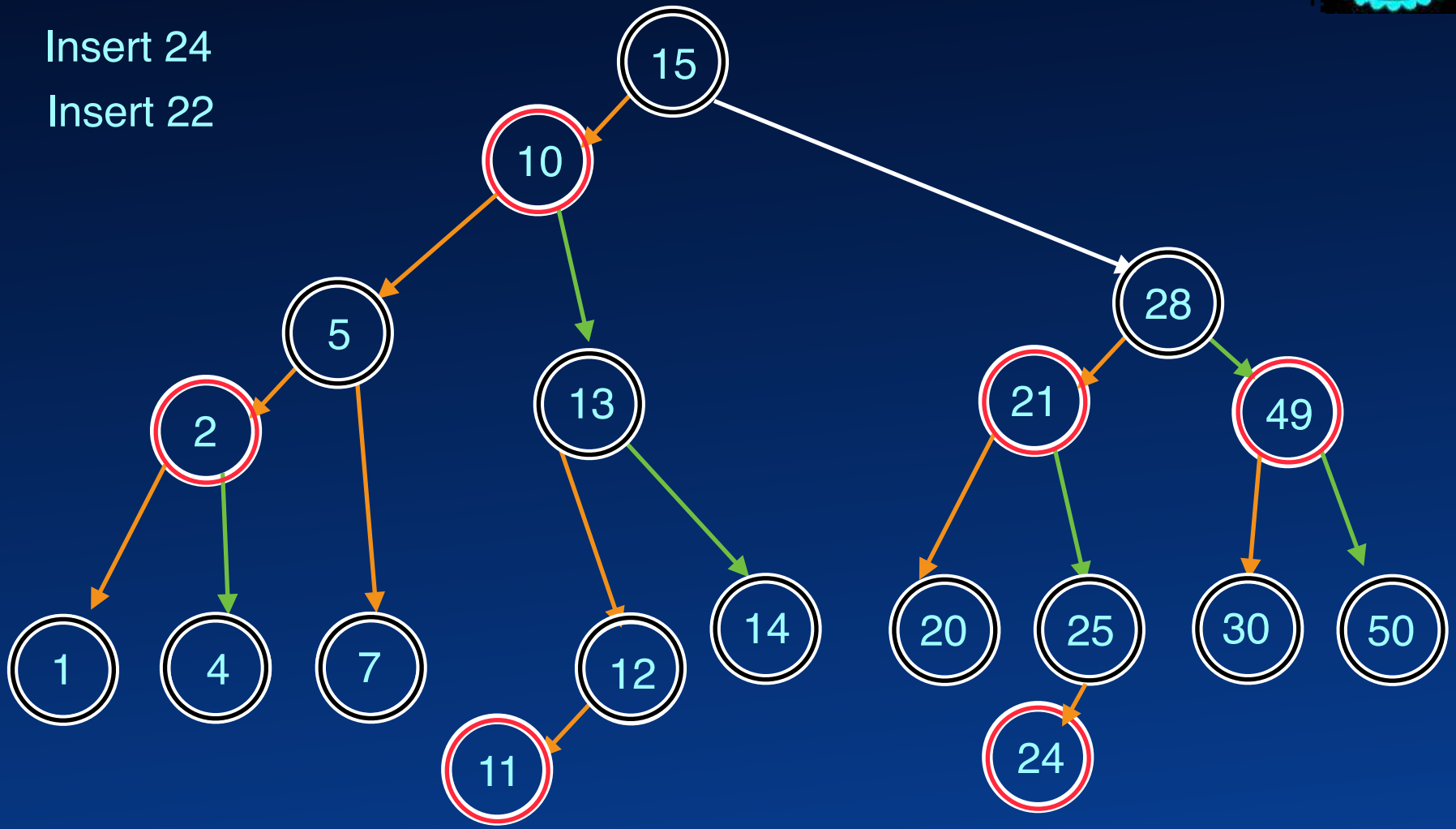


Insert @null in node ***n***

Red-Black Tree Insert

Insert 24

Insert 22

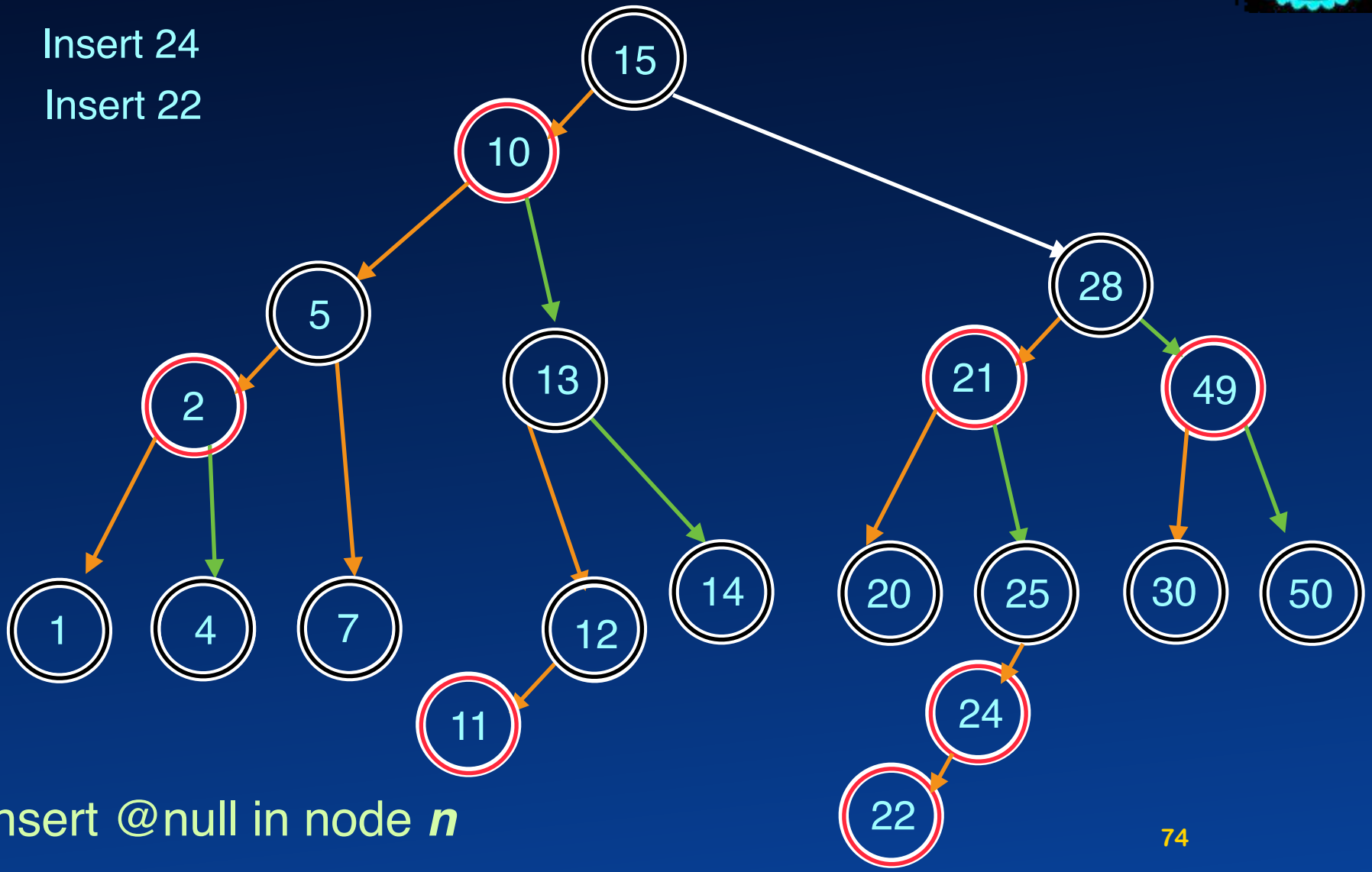


Insert @null in node ***n***

Red-Black Tree Insert

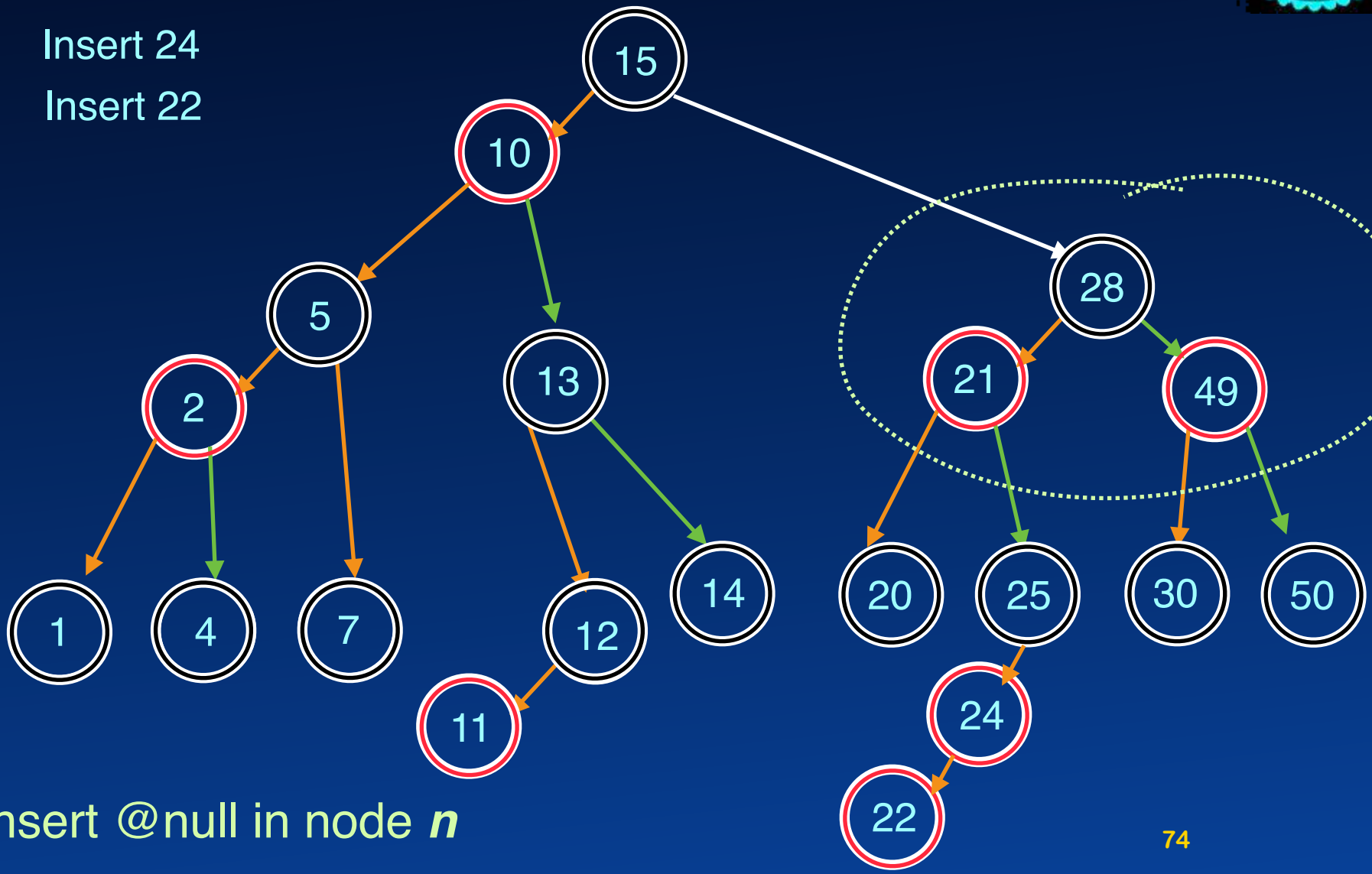
Insert 24

Insert 22



Insert @null in node ***n***

Red-Black Tree Insert

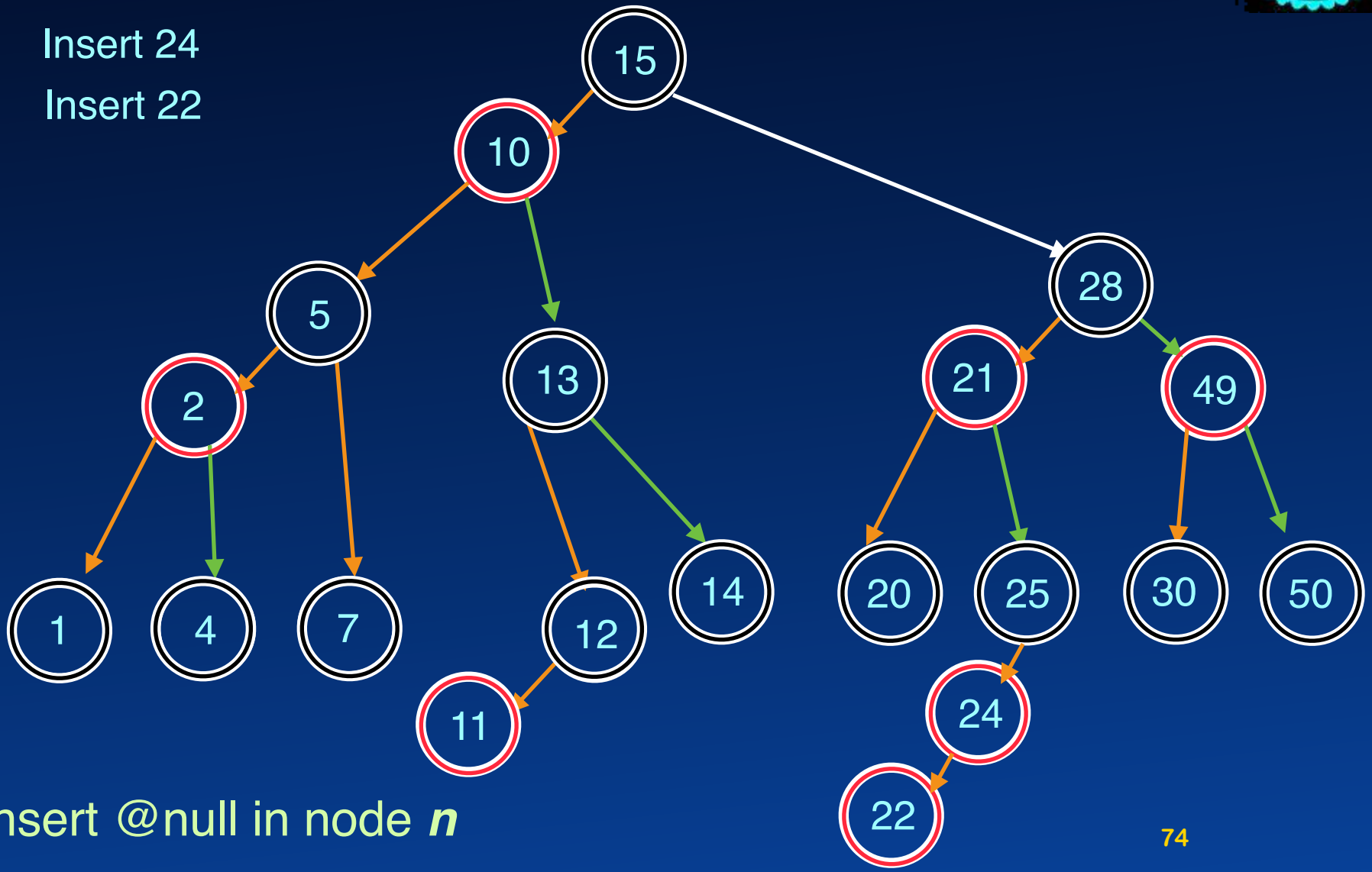


Insert @null in node ***n***

Red-Black Tree Insert

Insert 24

Insert 22

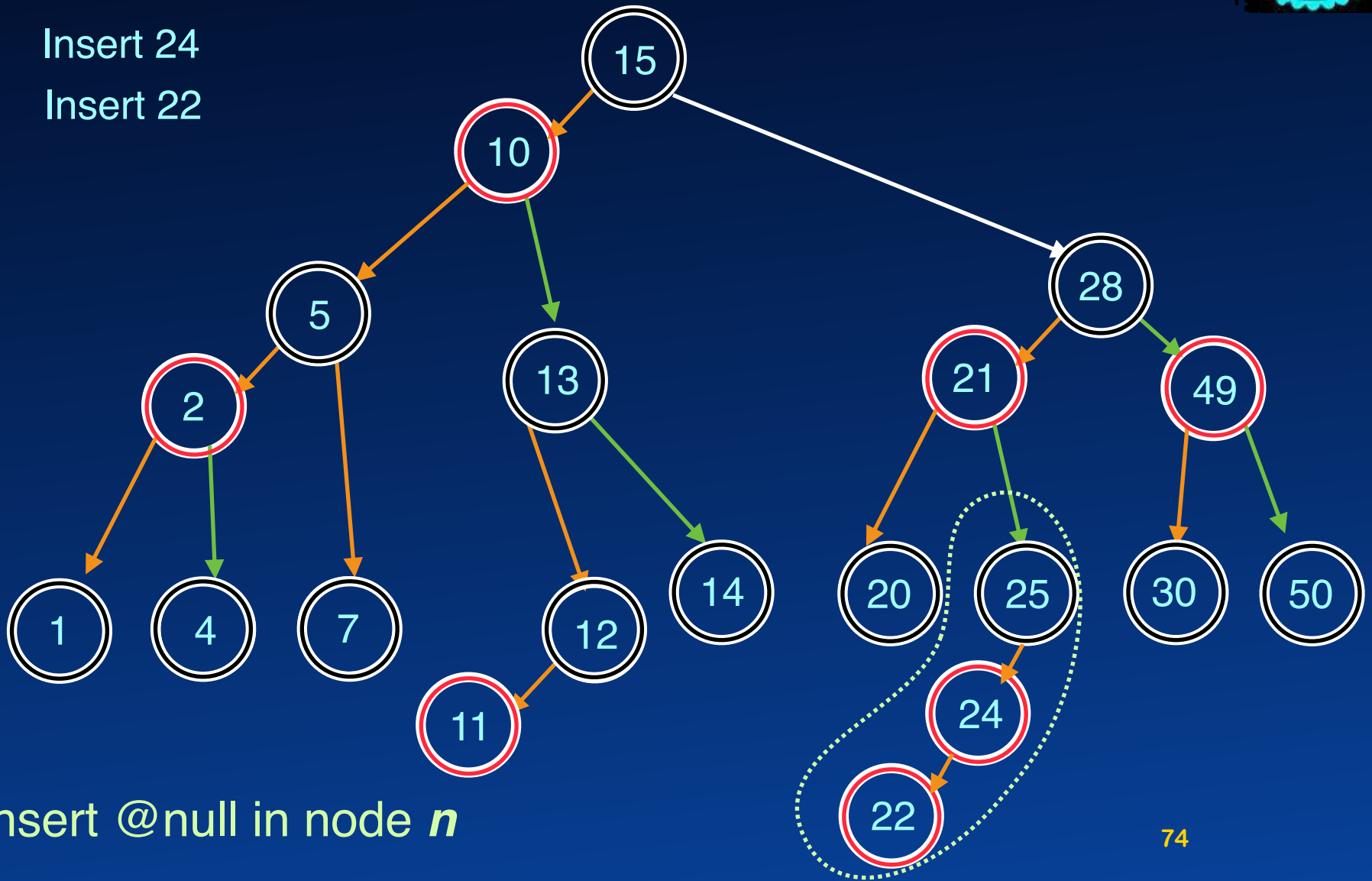


Insert @null in node ***n***

Red-Black Tree Insert

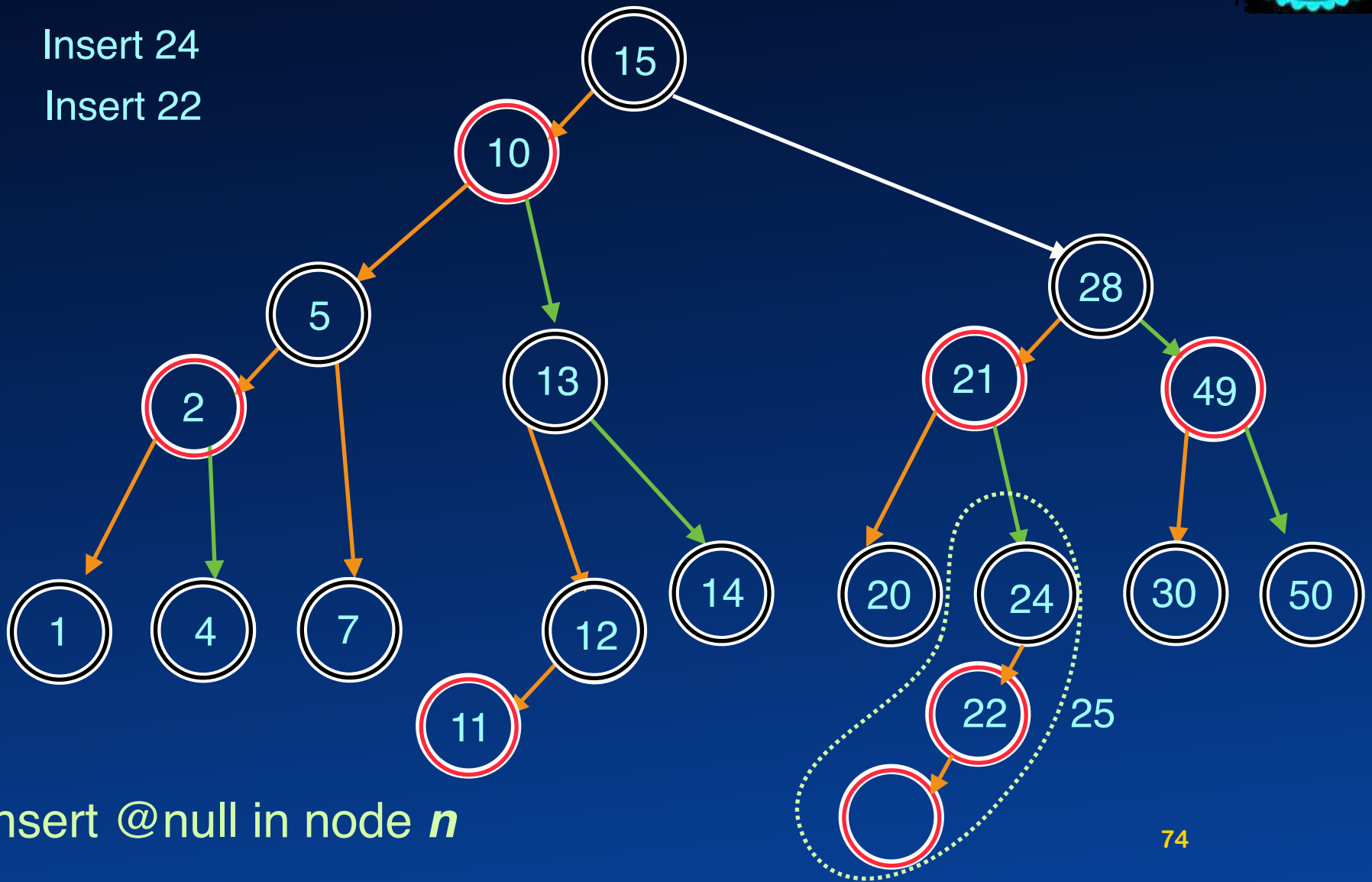
Insert 24

Insert 22

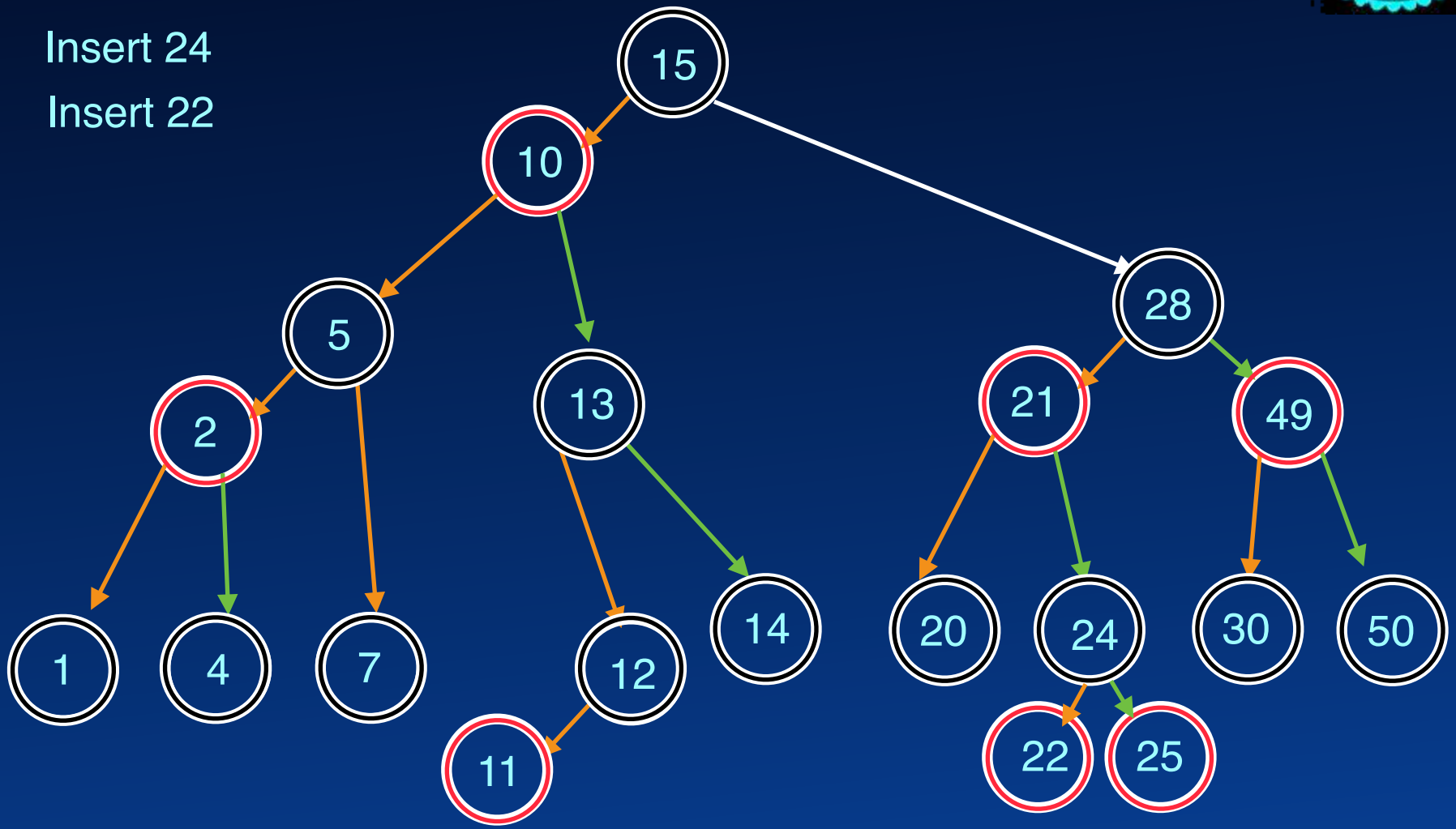


Insert @null in node ***n***

Red-Black Tree Insert

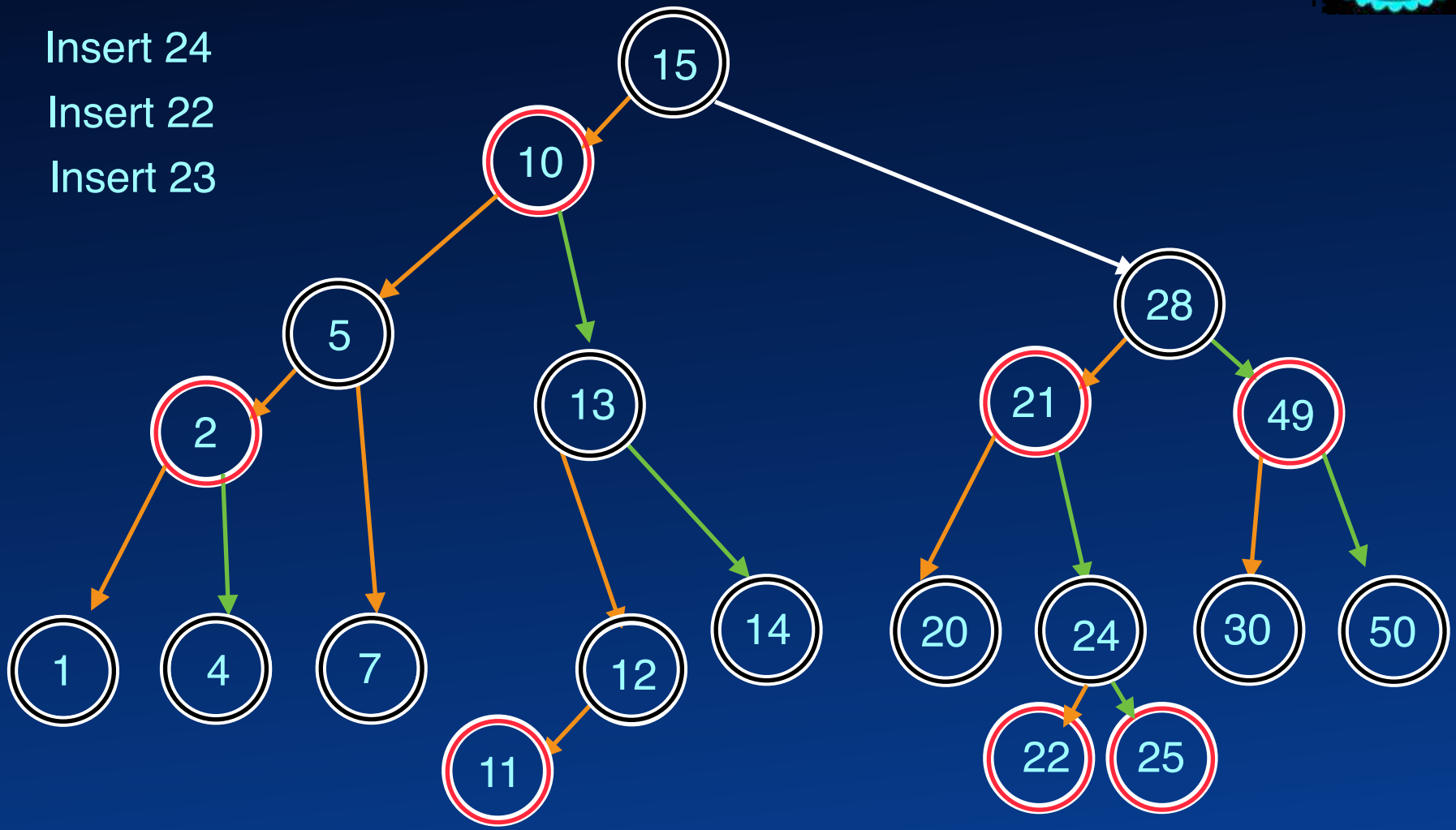


Red-Black Tree Insert



Insert @null in node ***n***

Red-Black Tree Insert



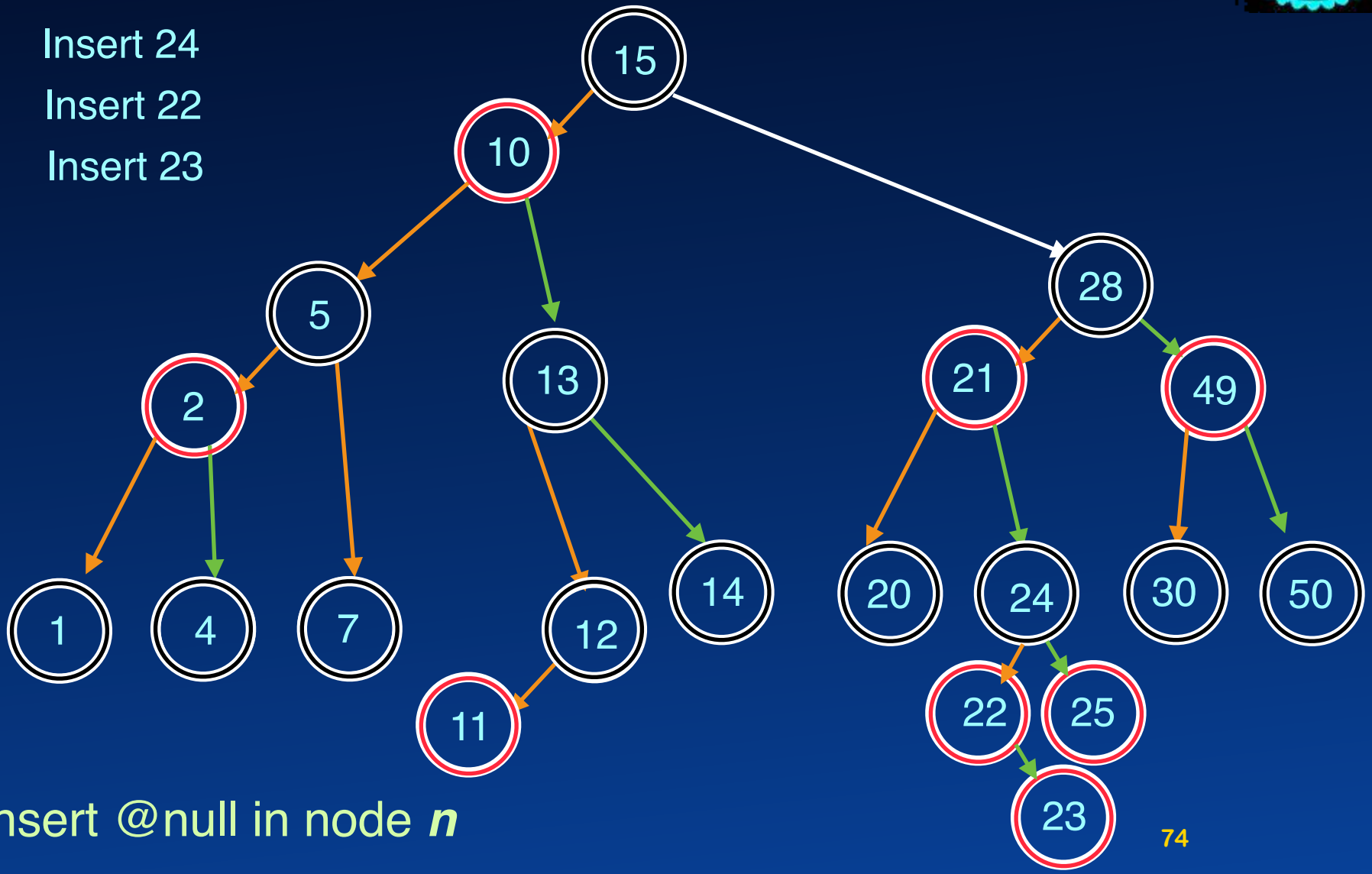
Insert @null in node ***n***

Red-Black Tree Insert

Insert 24

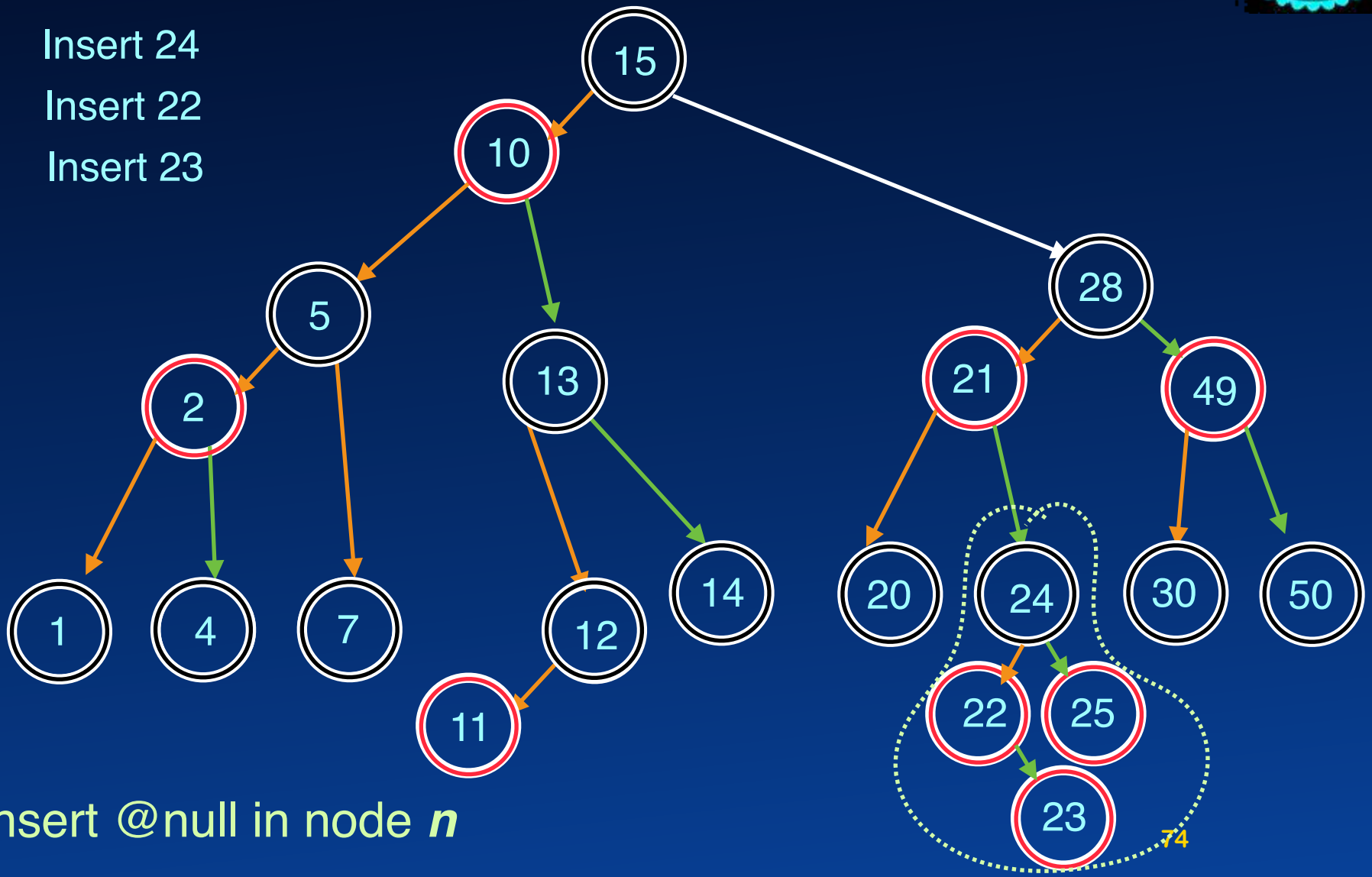
Insert 22

Insert 23

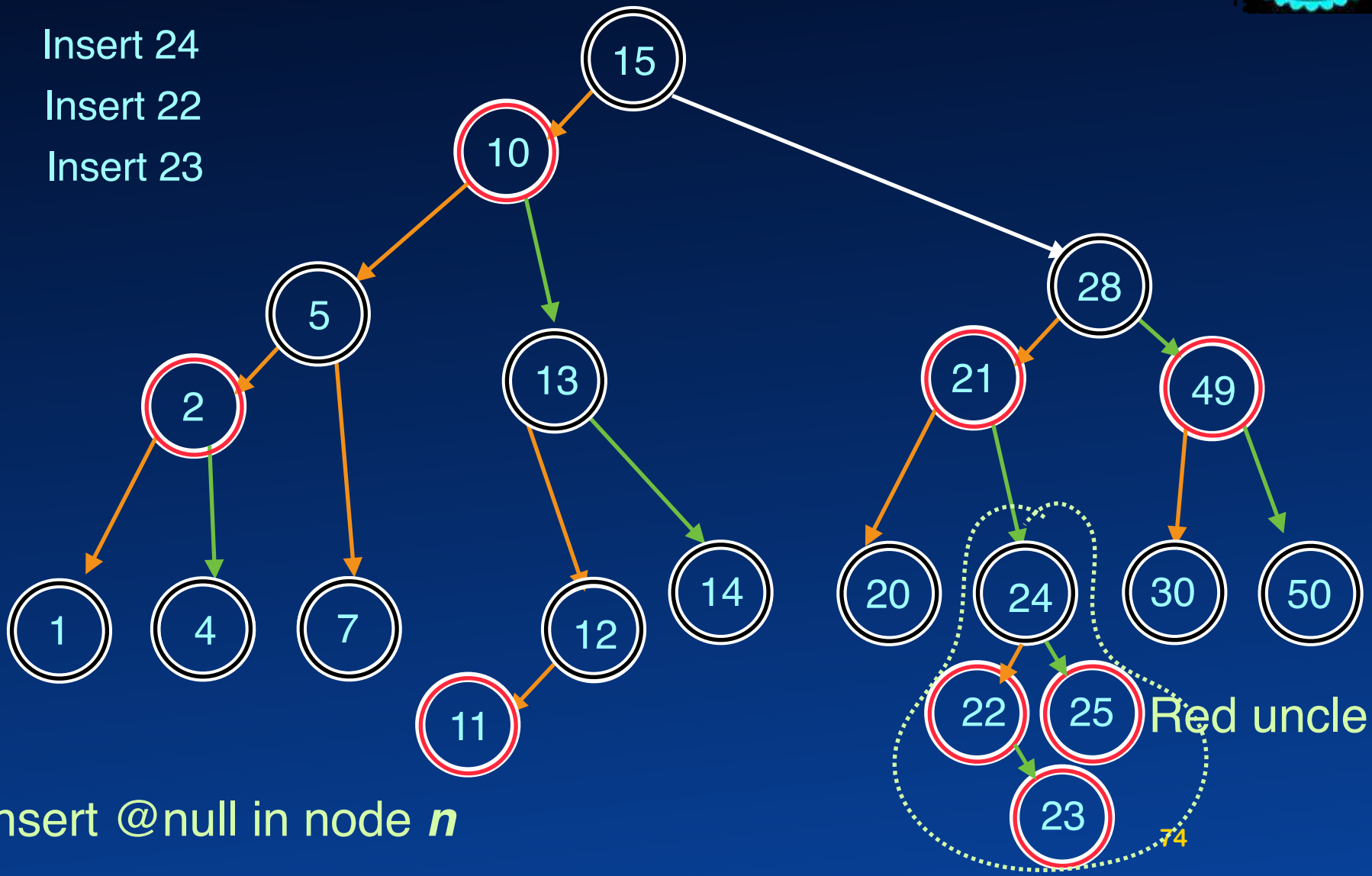


Insert @null in node ***n***

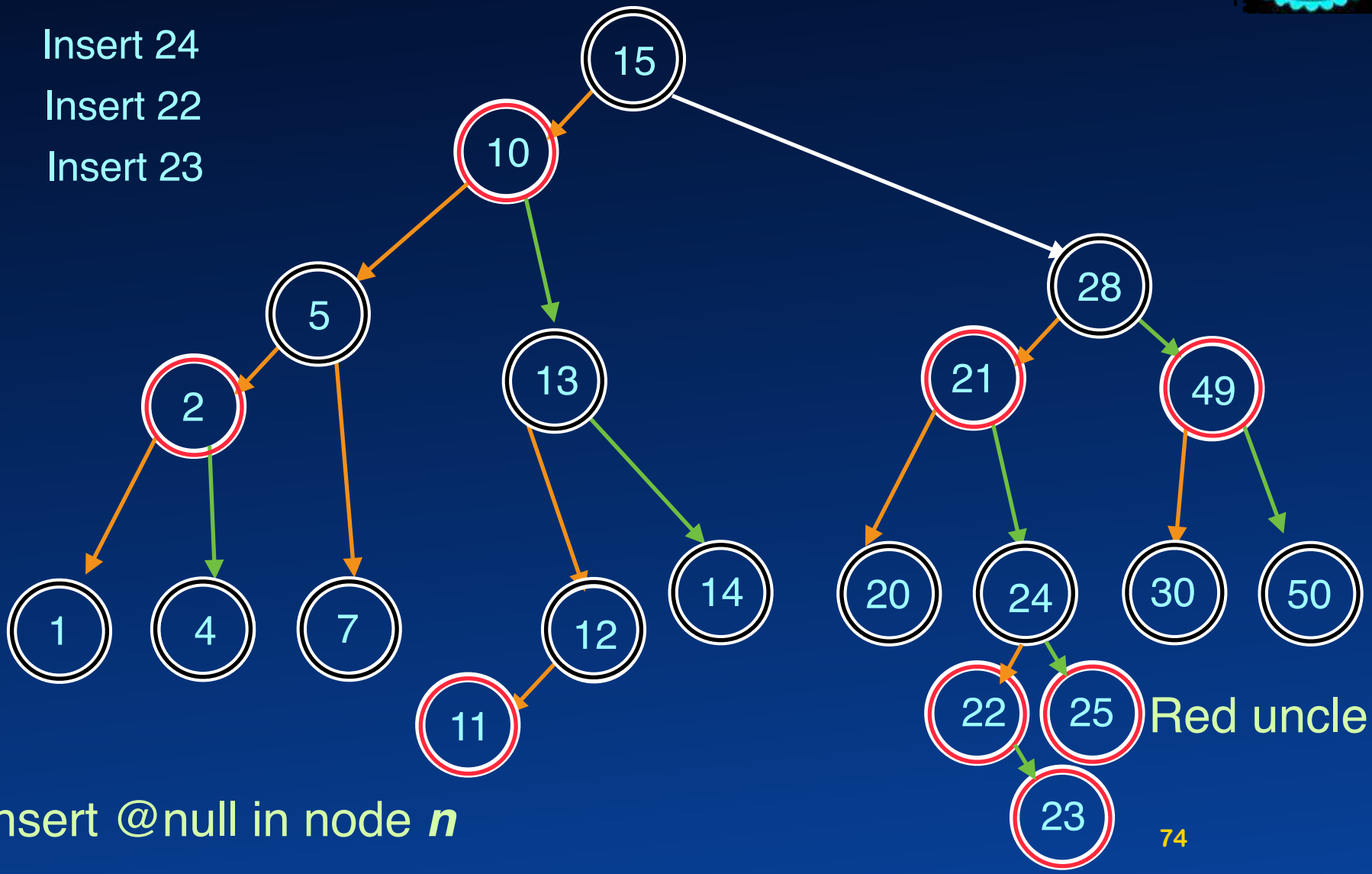
Red-Black Tree Insert



Red-Black Tree Insert

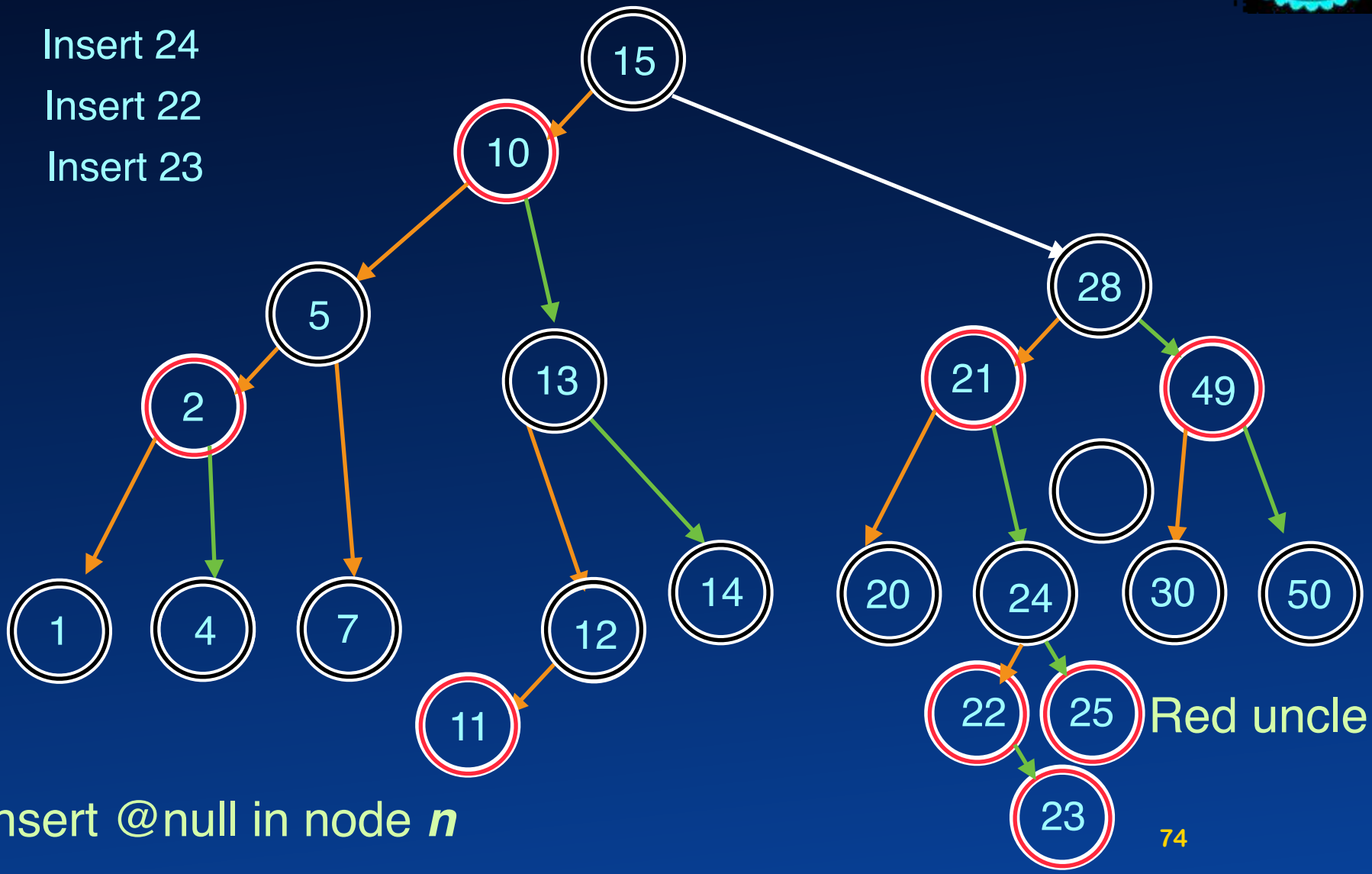


Red-Black Tree Insert



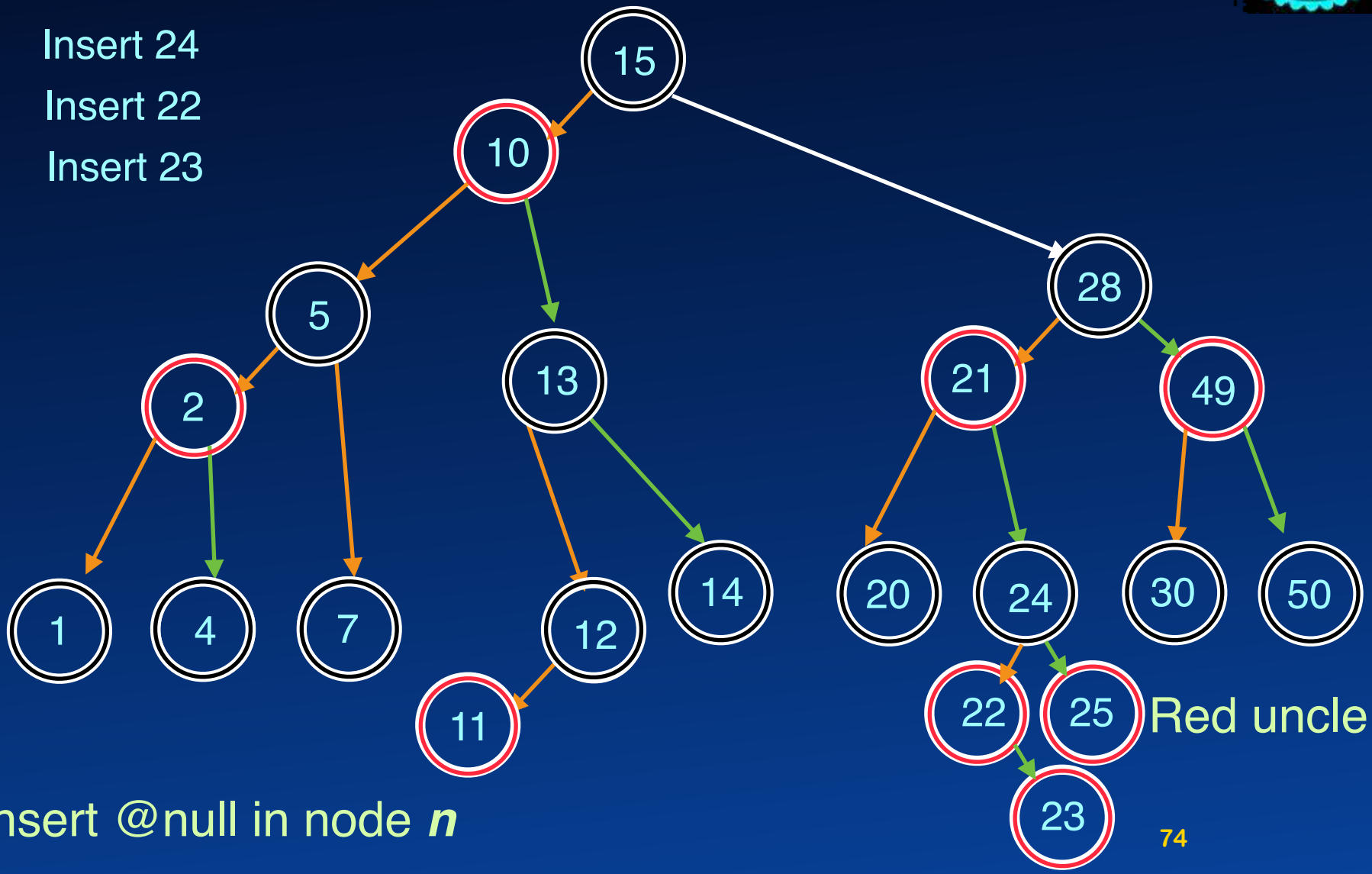
Insert @null in node ***n***

Red-Black Tree Insert



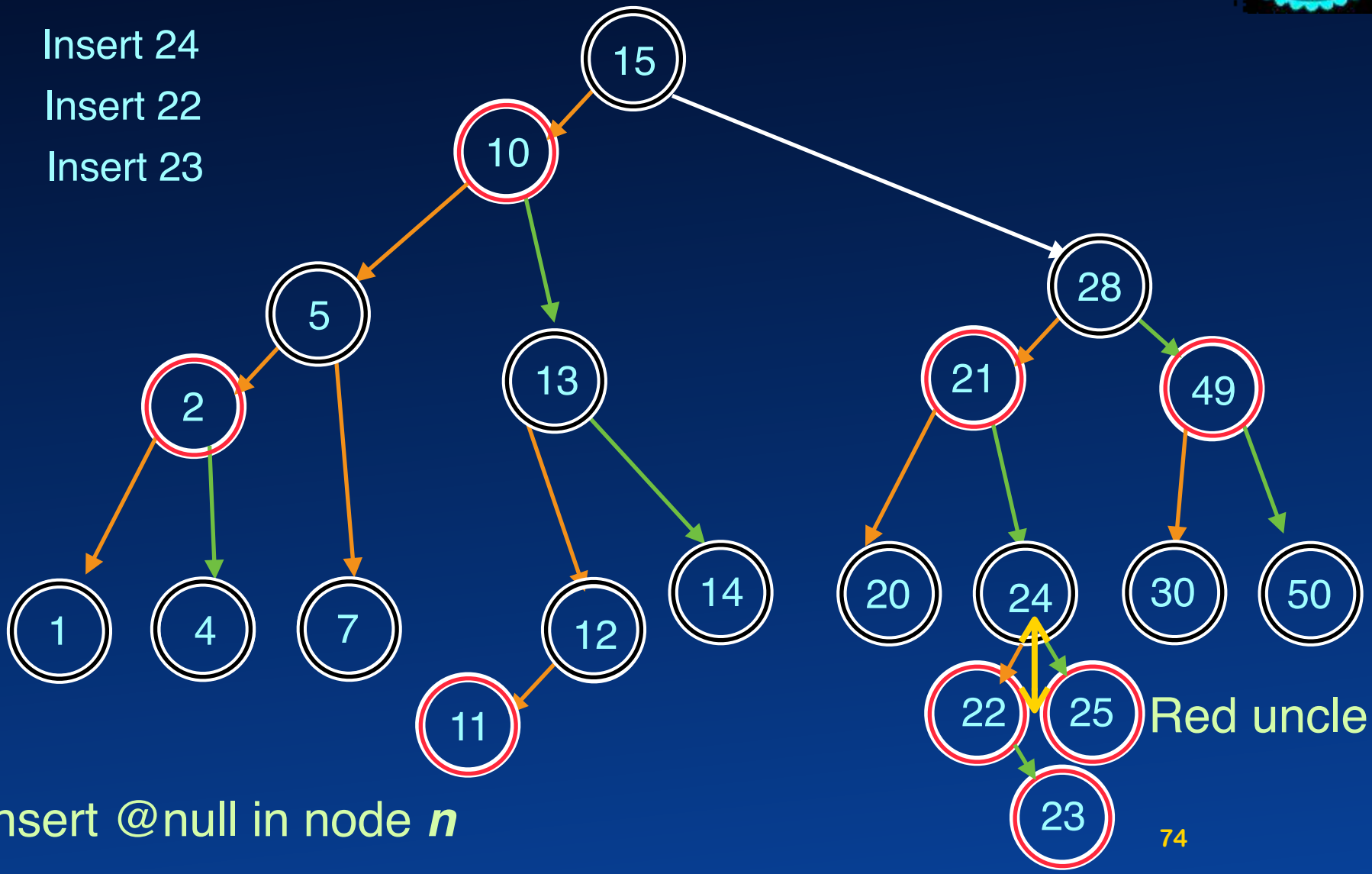
Insert @null in node ***n***

Red-Black Tree Insert



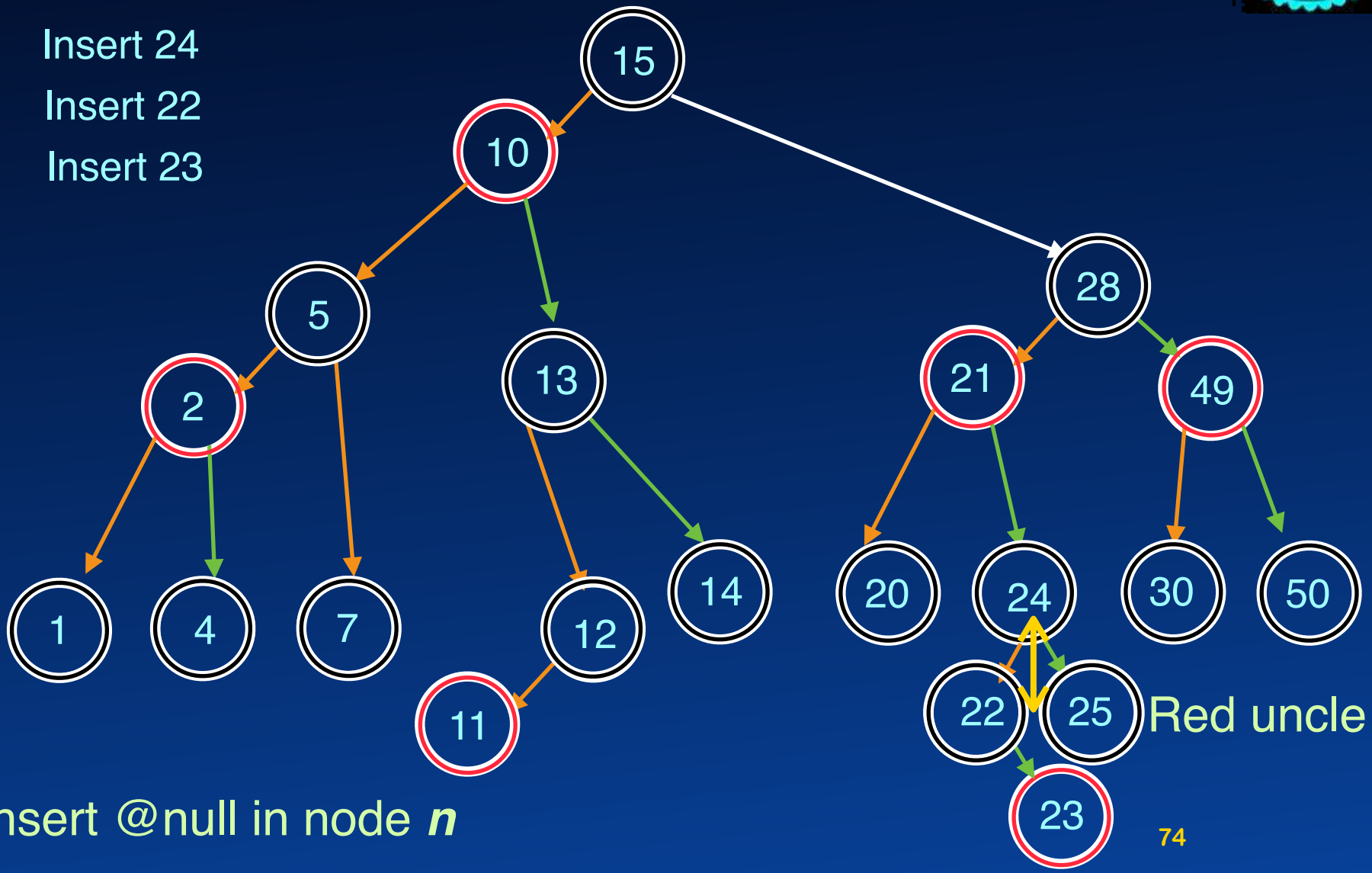
Insert @null in node ***n***

Red-Black Tree Insert



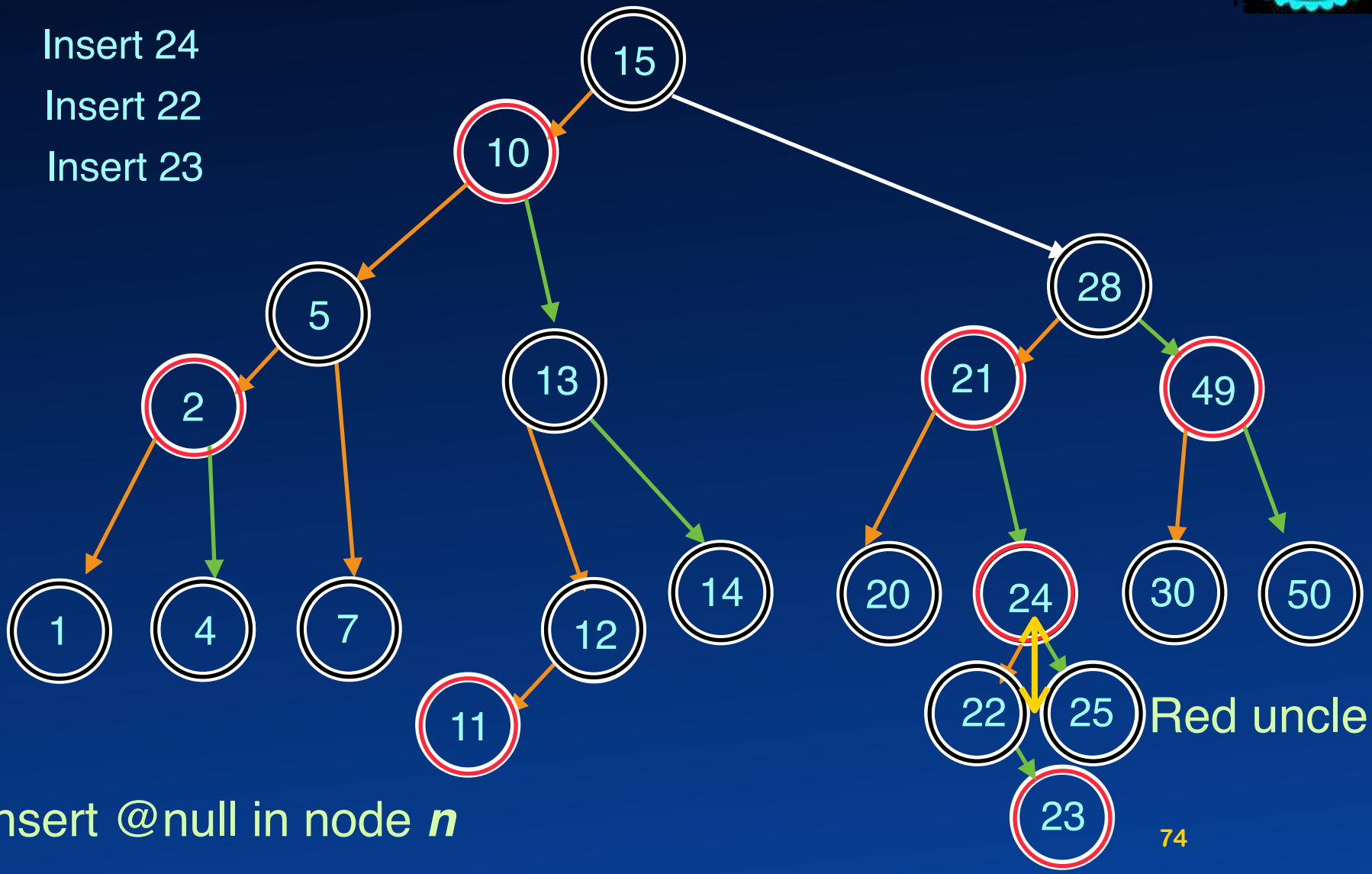
Insert @null in node ***n***

Red-Black Tree Insert



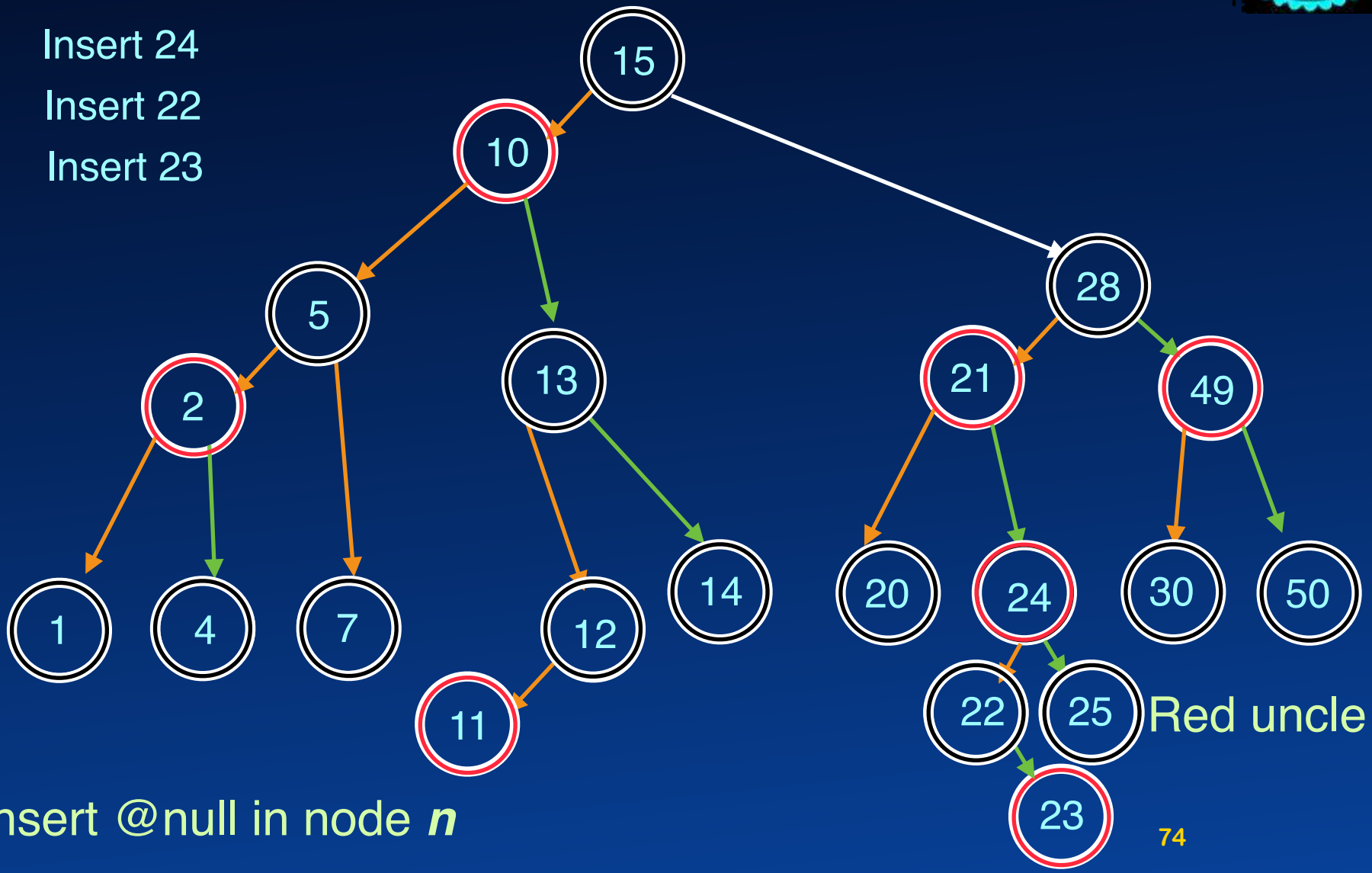
Insert @null in node ***n***

Red-Black Tree Insert



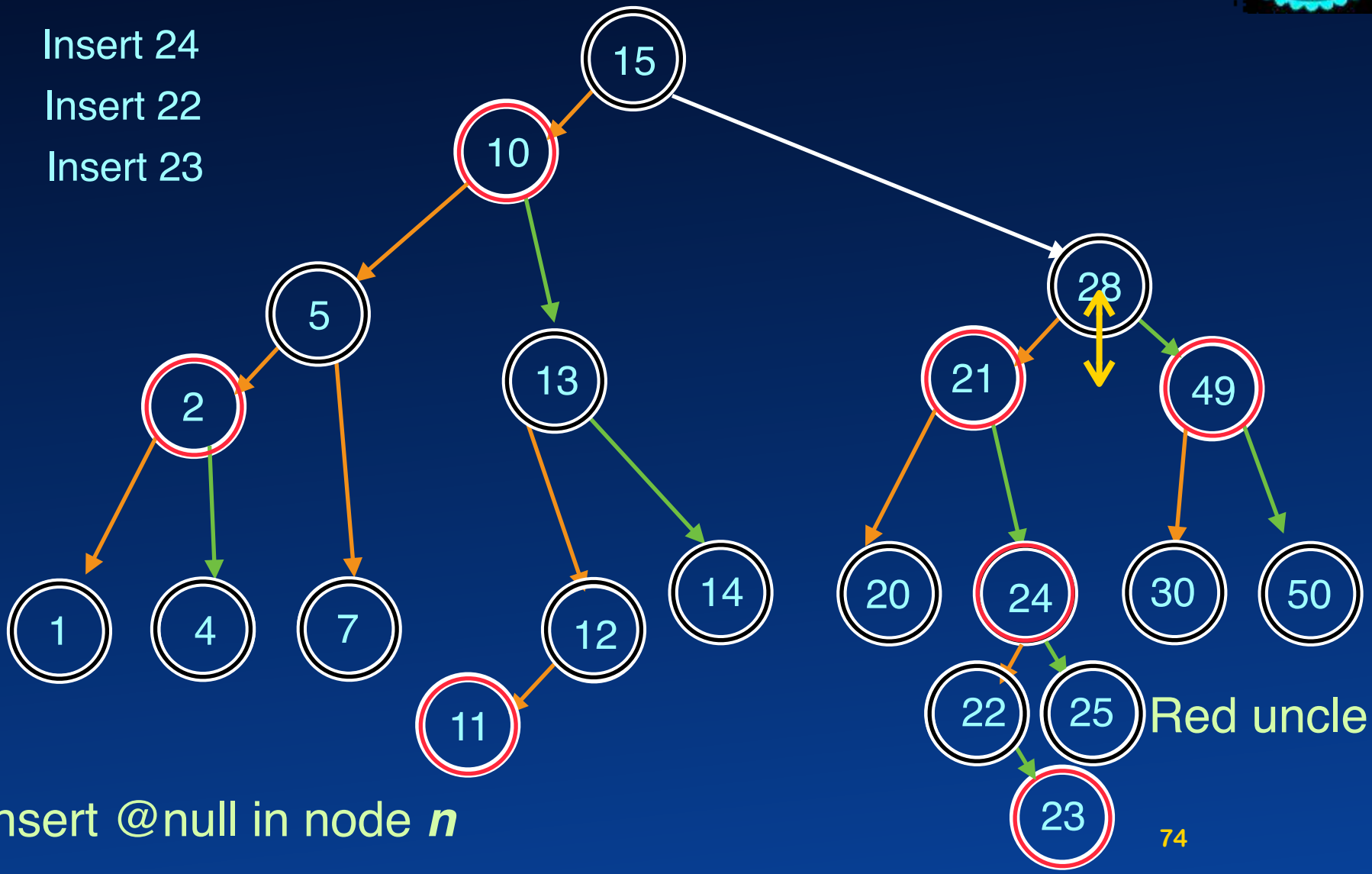
Insert @null in node ***n***

Red-Black Tree Insert



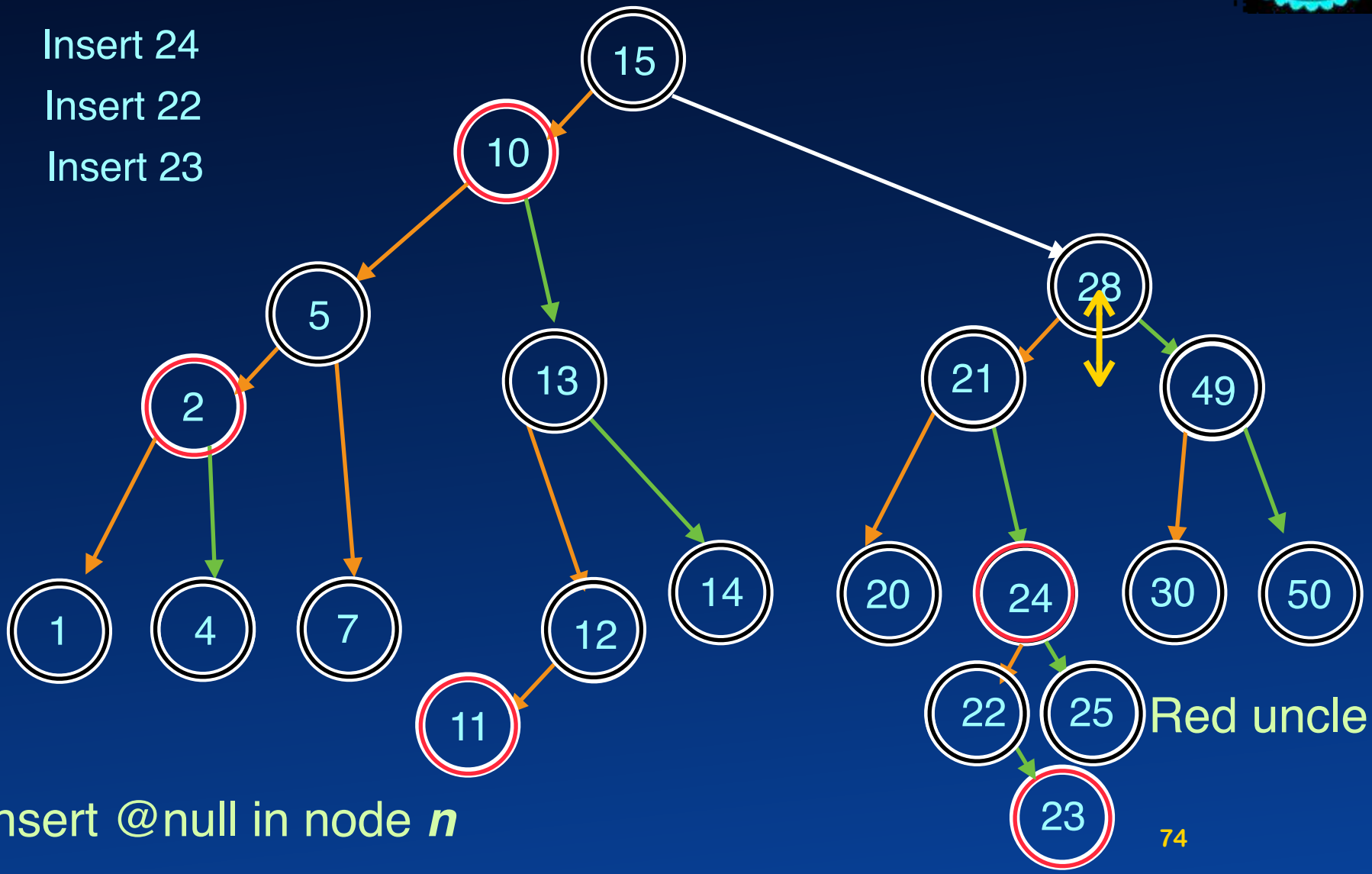
Insert @null in node ***n***

Red-Black Tree Insert



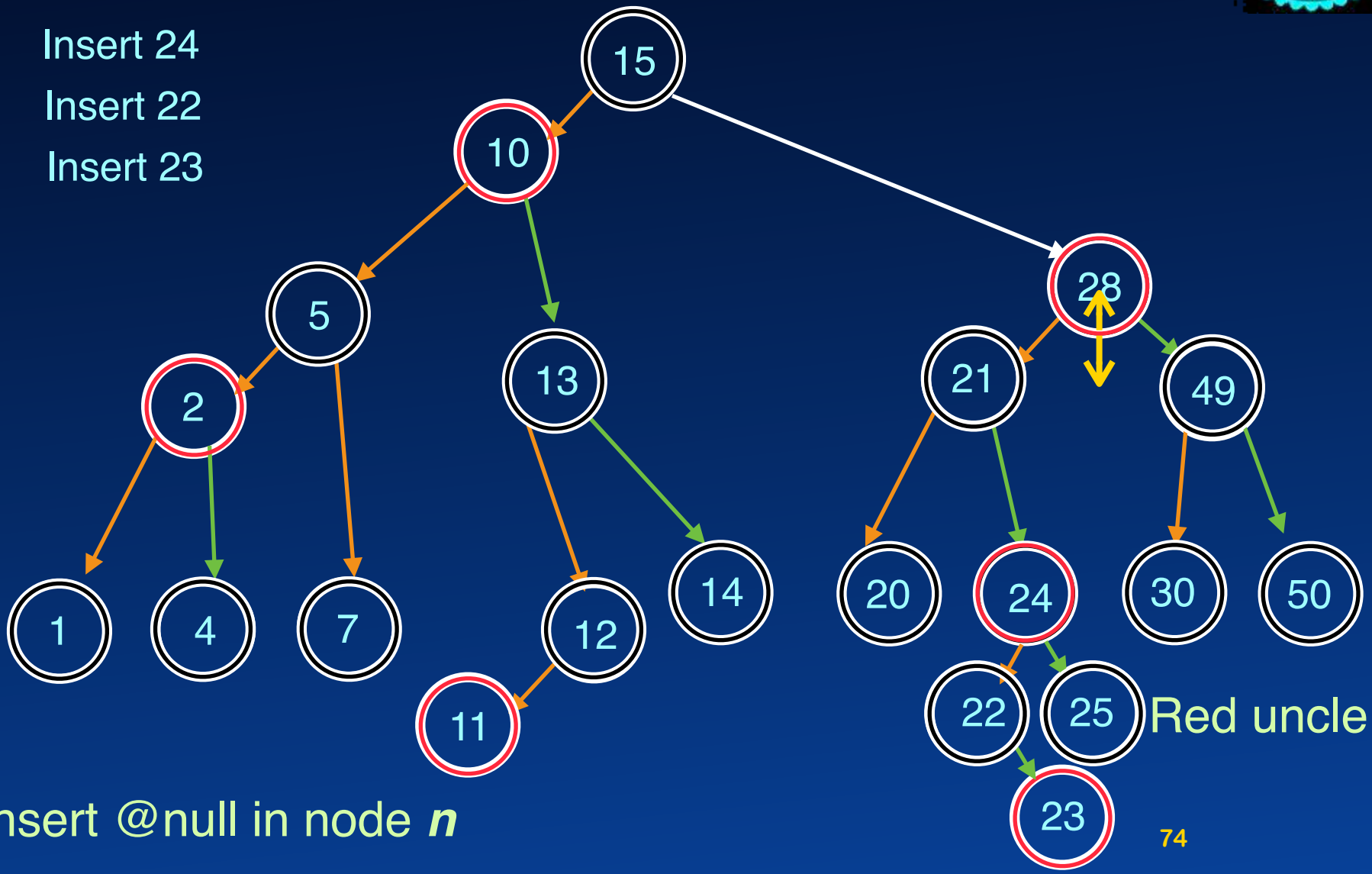
Insert @null in node ***n***

Red-Black Tree Insert



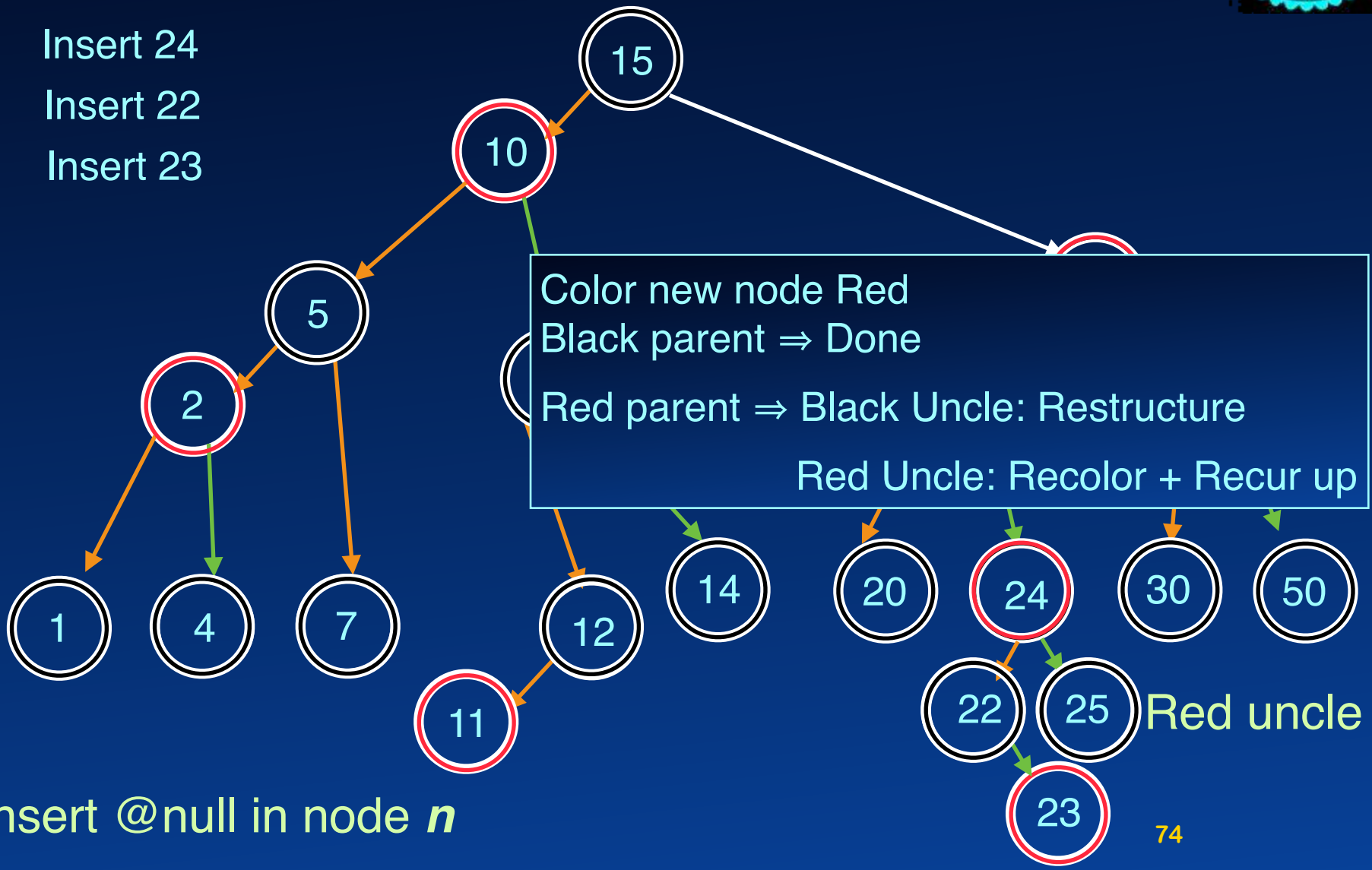
Insert @null in node ***n***

Red-Black Tree Insert

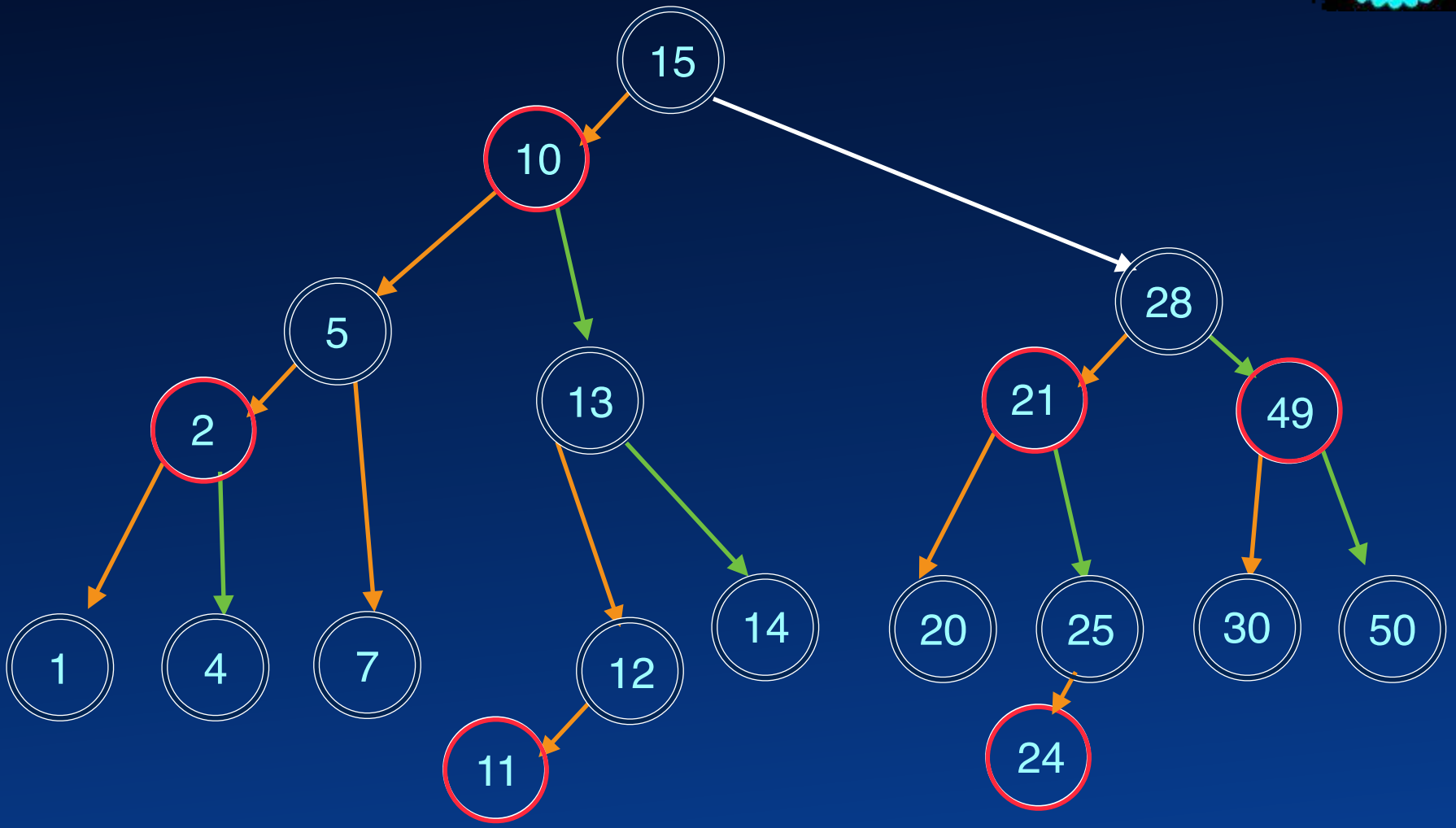


Insert @null in node ***n***

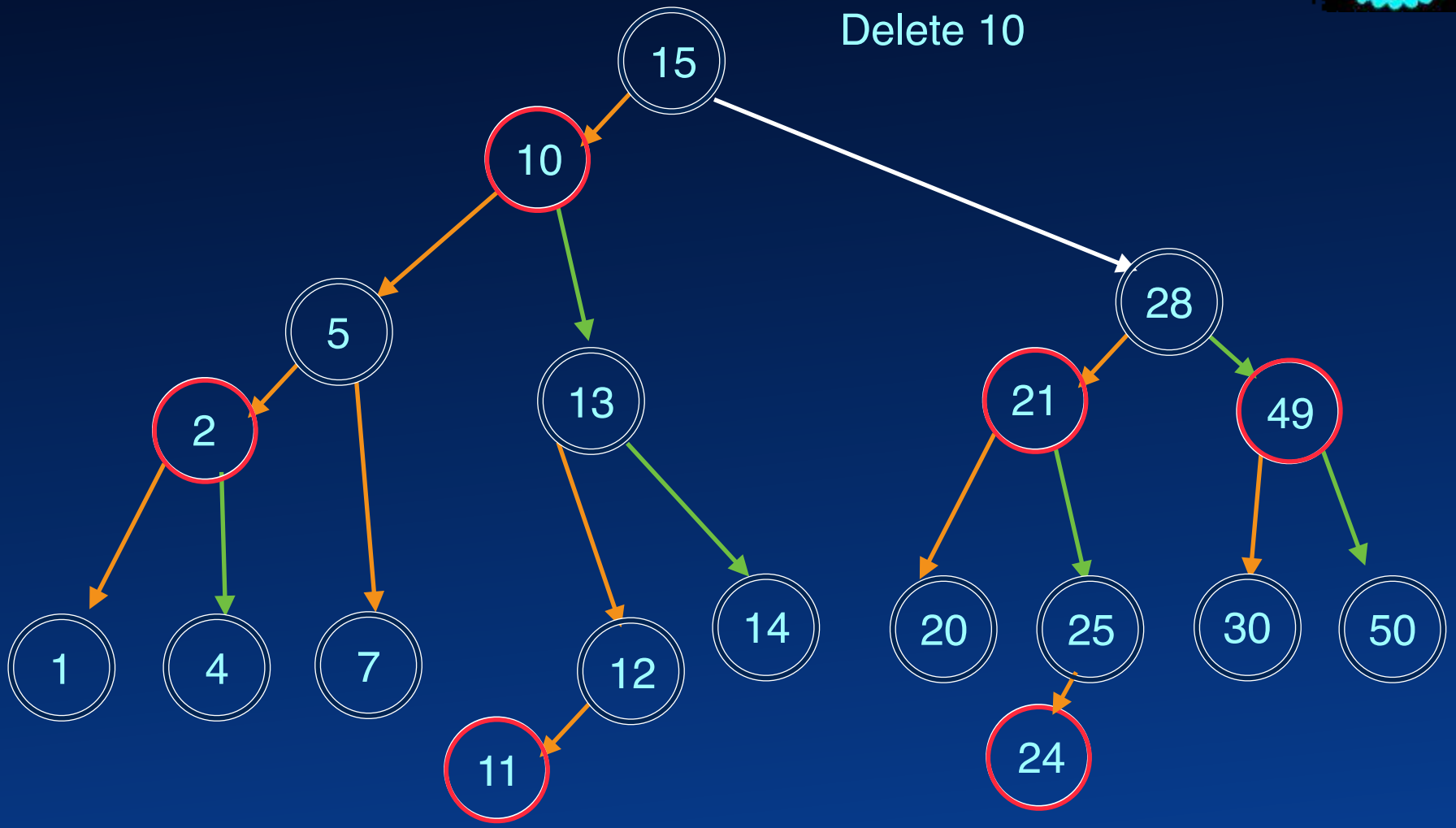
Red-Black Tree Insert



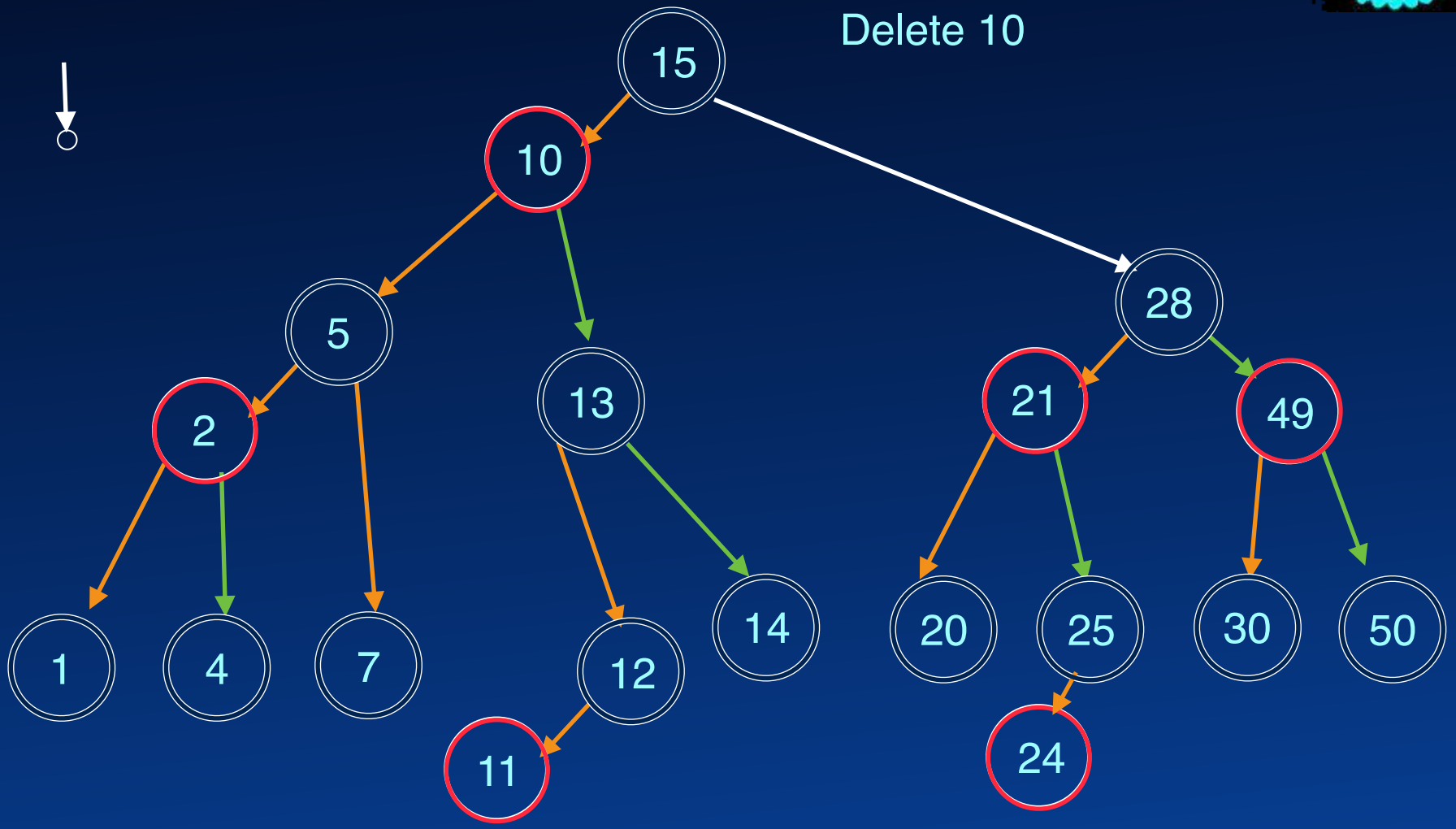
R-B Tree Deletion



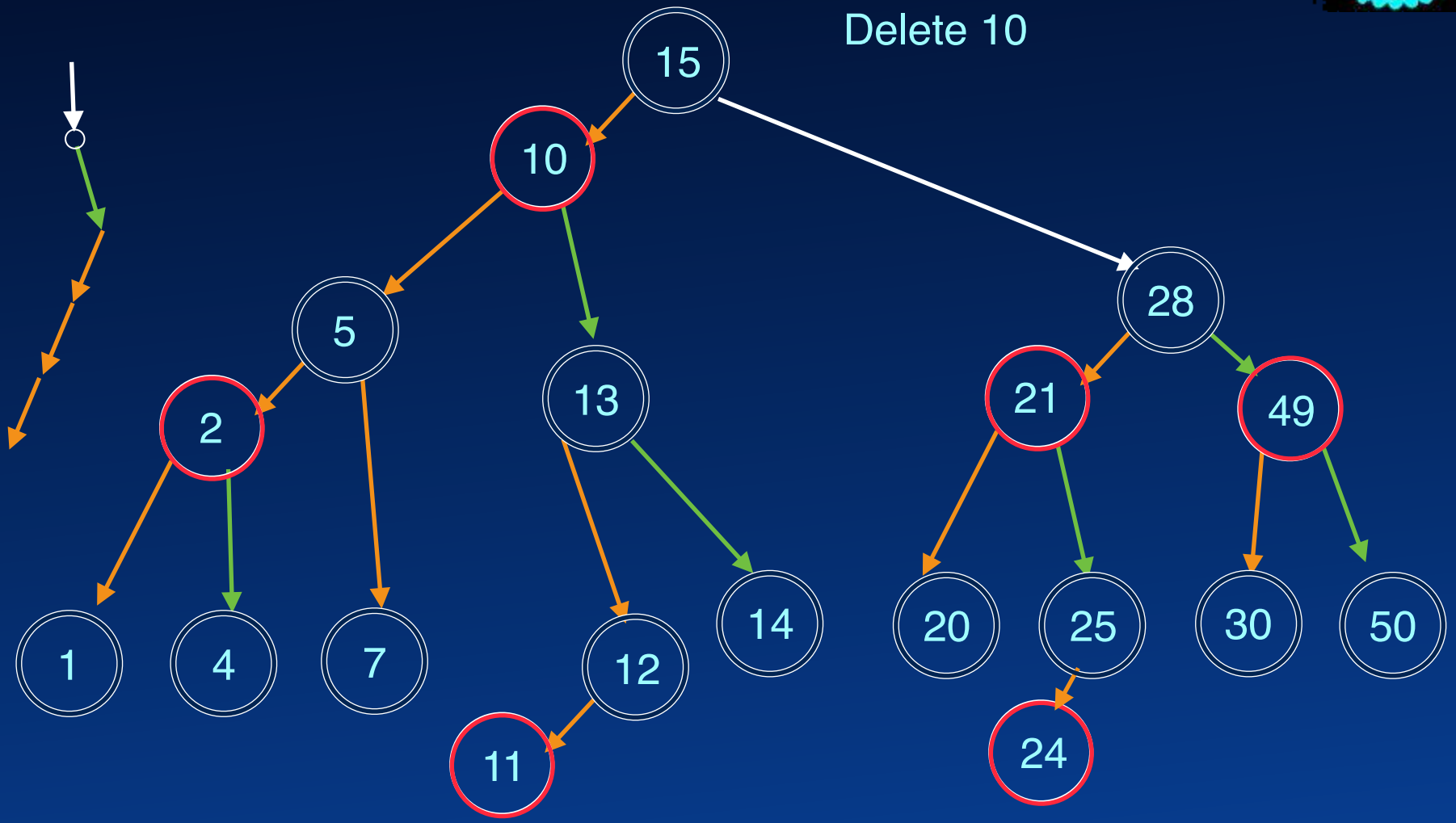
R-B Tree Deletion



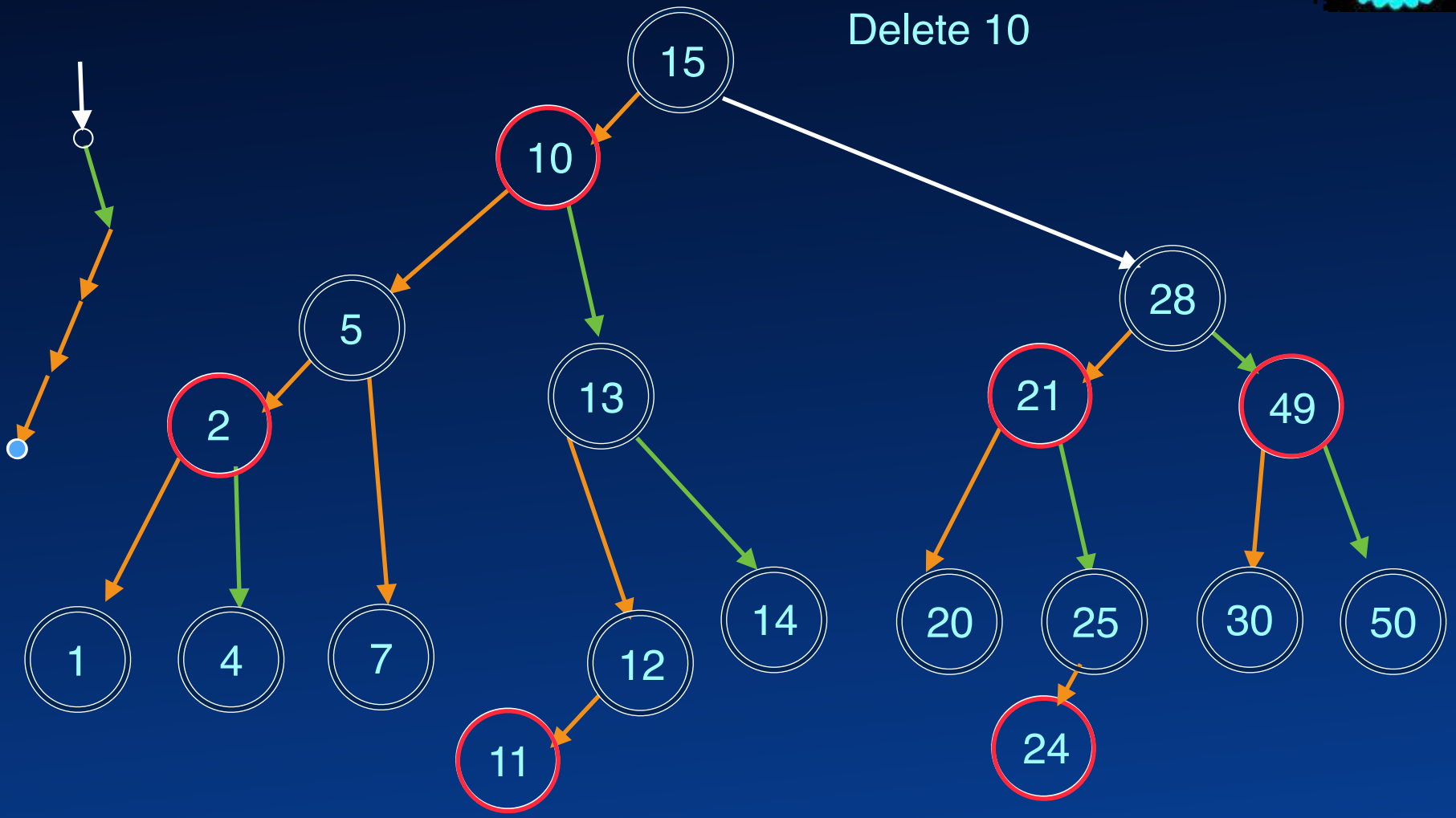
R-B Tree Deletion



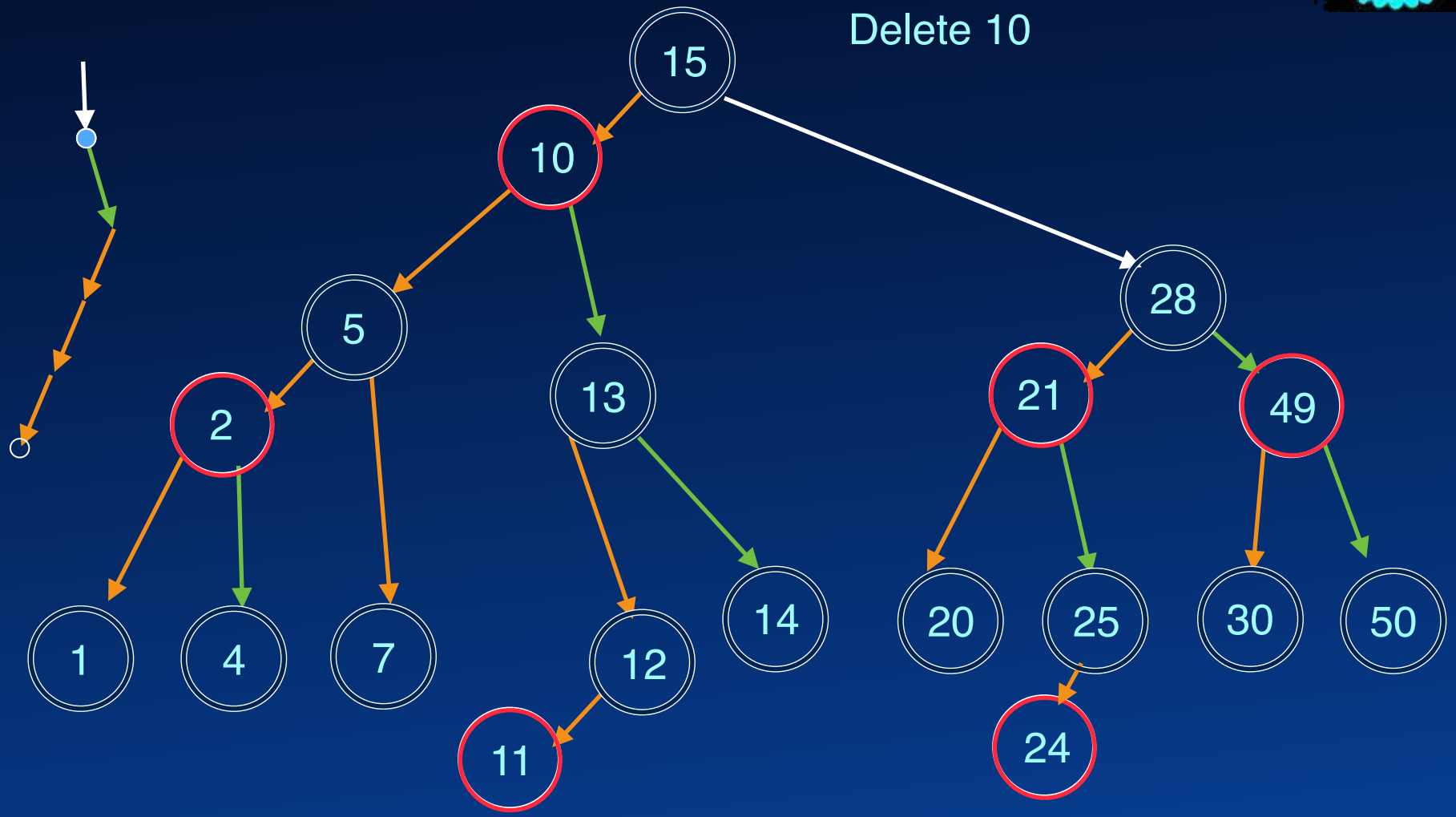
R-B Tree Deletion



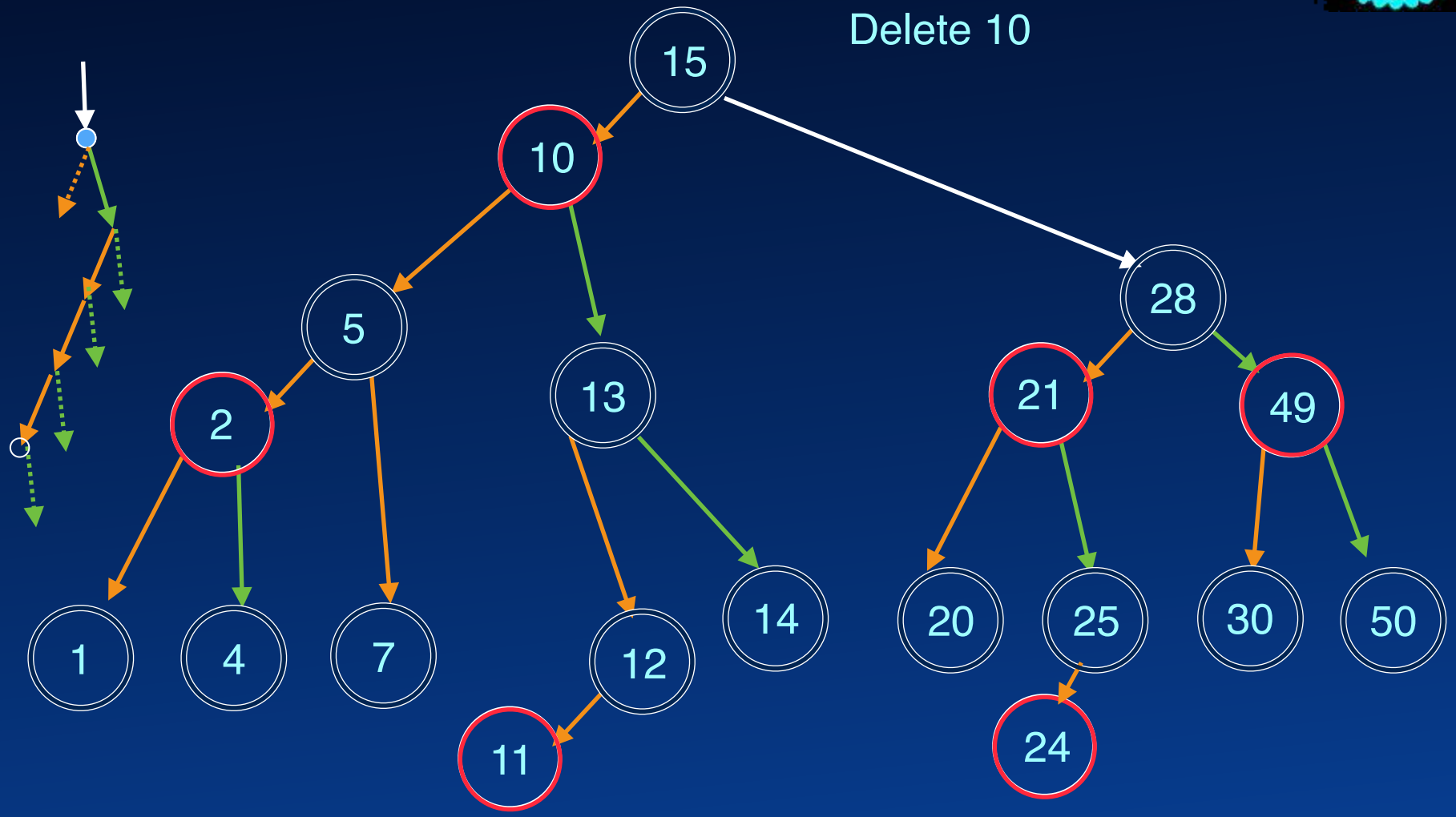
R-B Tree Deletion



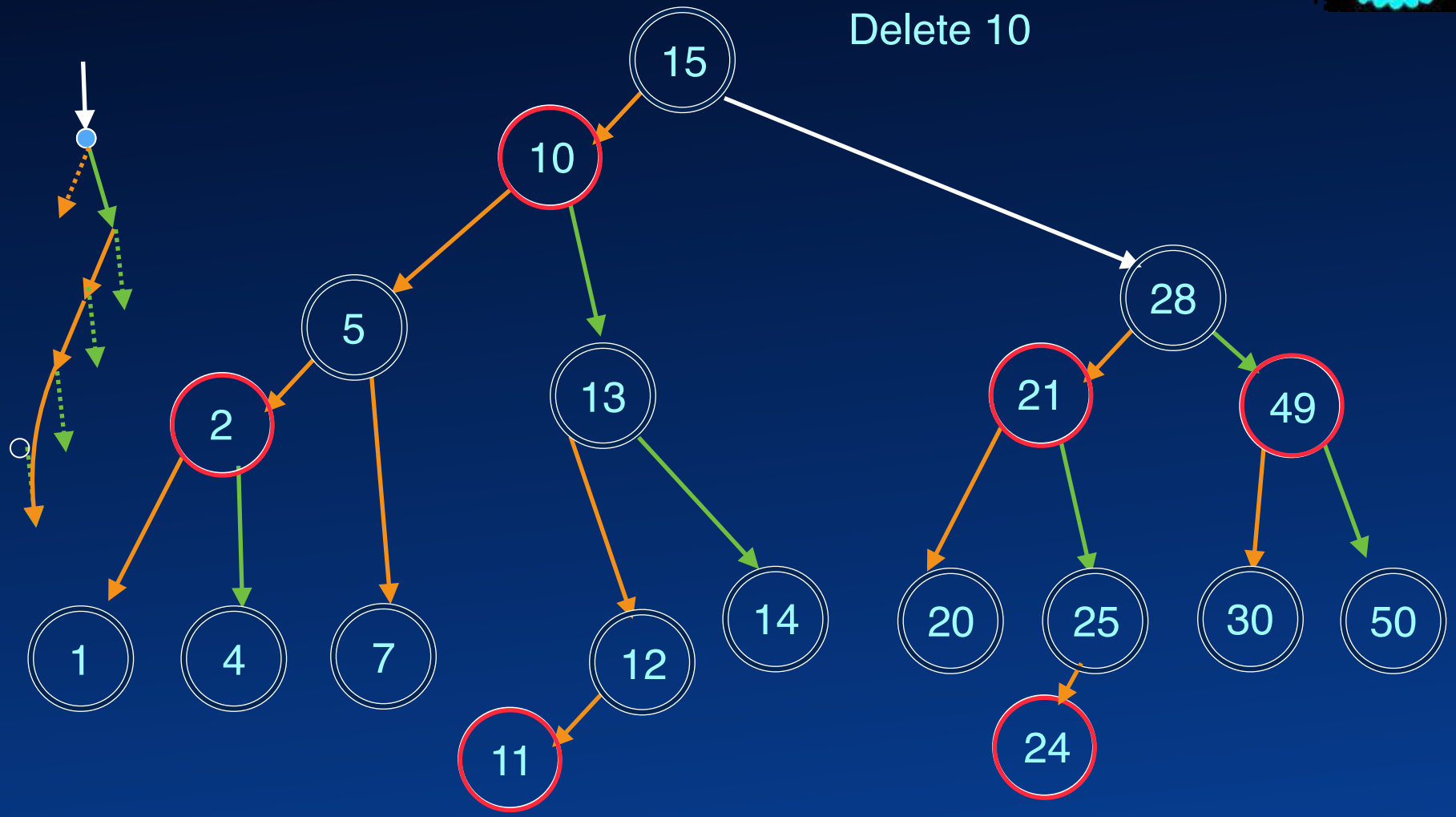
R-B Tree Deletion



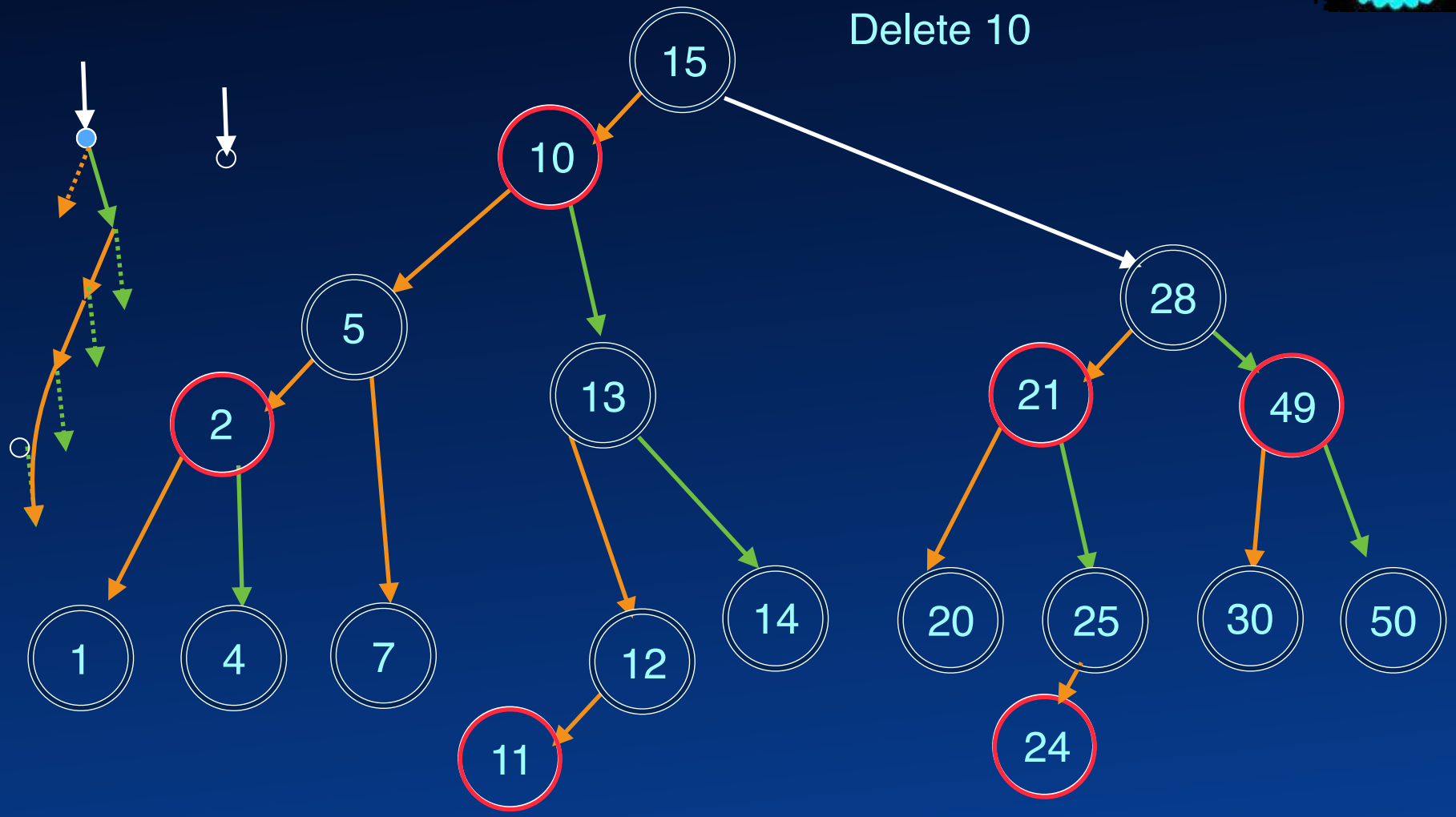
R-B Tree Deletion



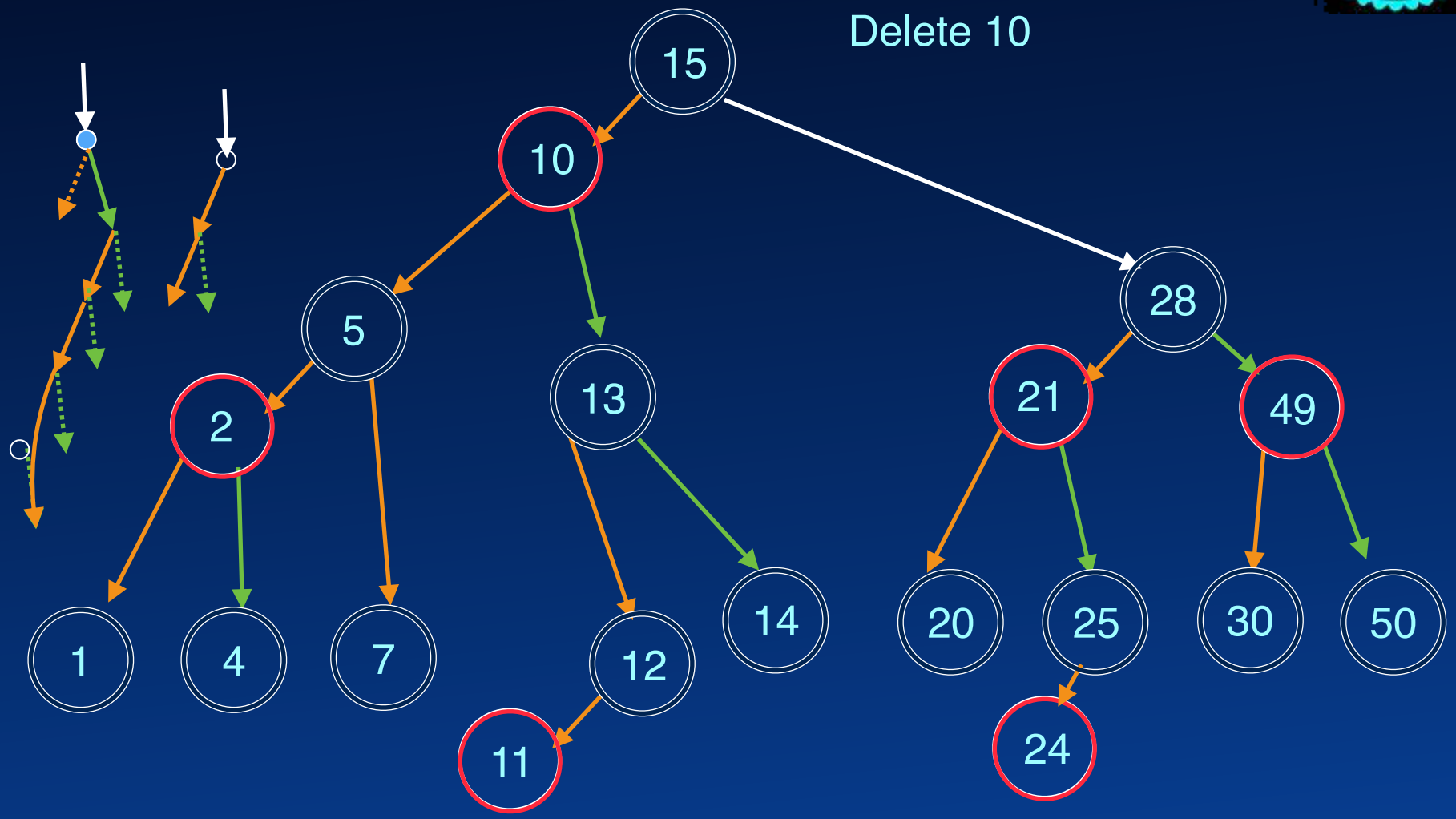
R-B Tree Deletion



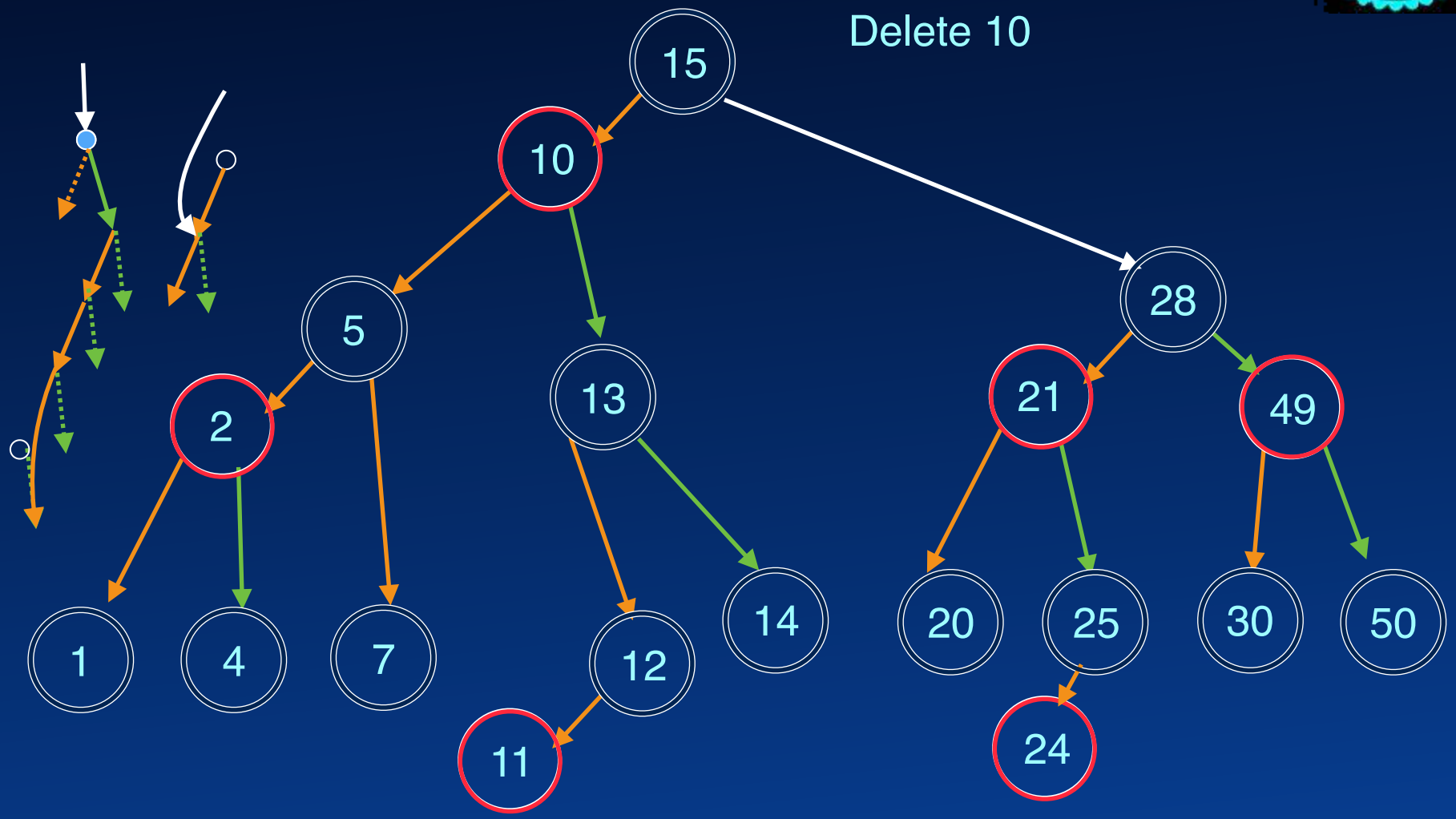
R-B Tree Deletion



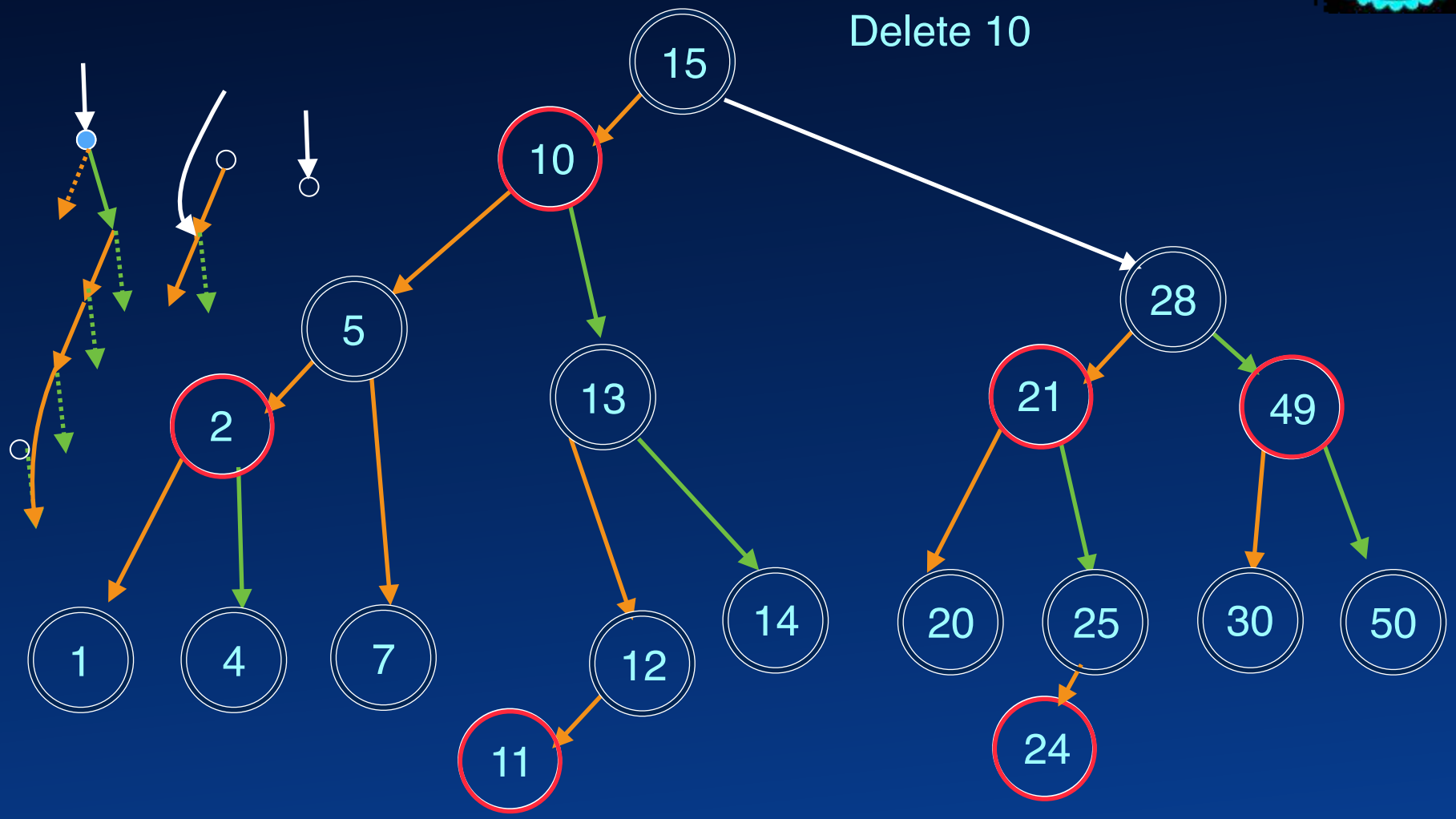
R-B Tree Deletion



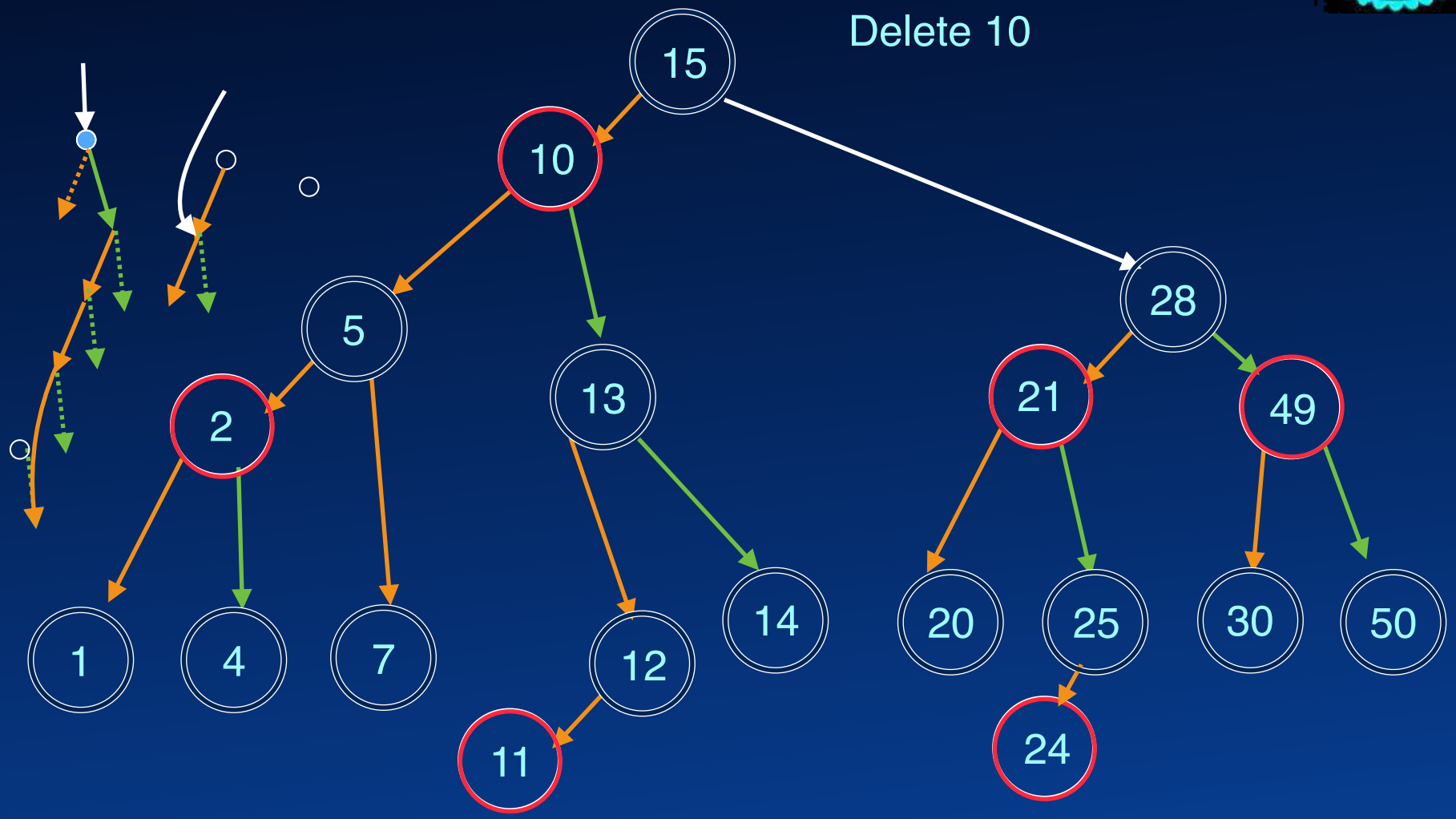
R-B Tree Deletion



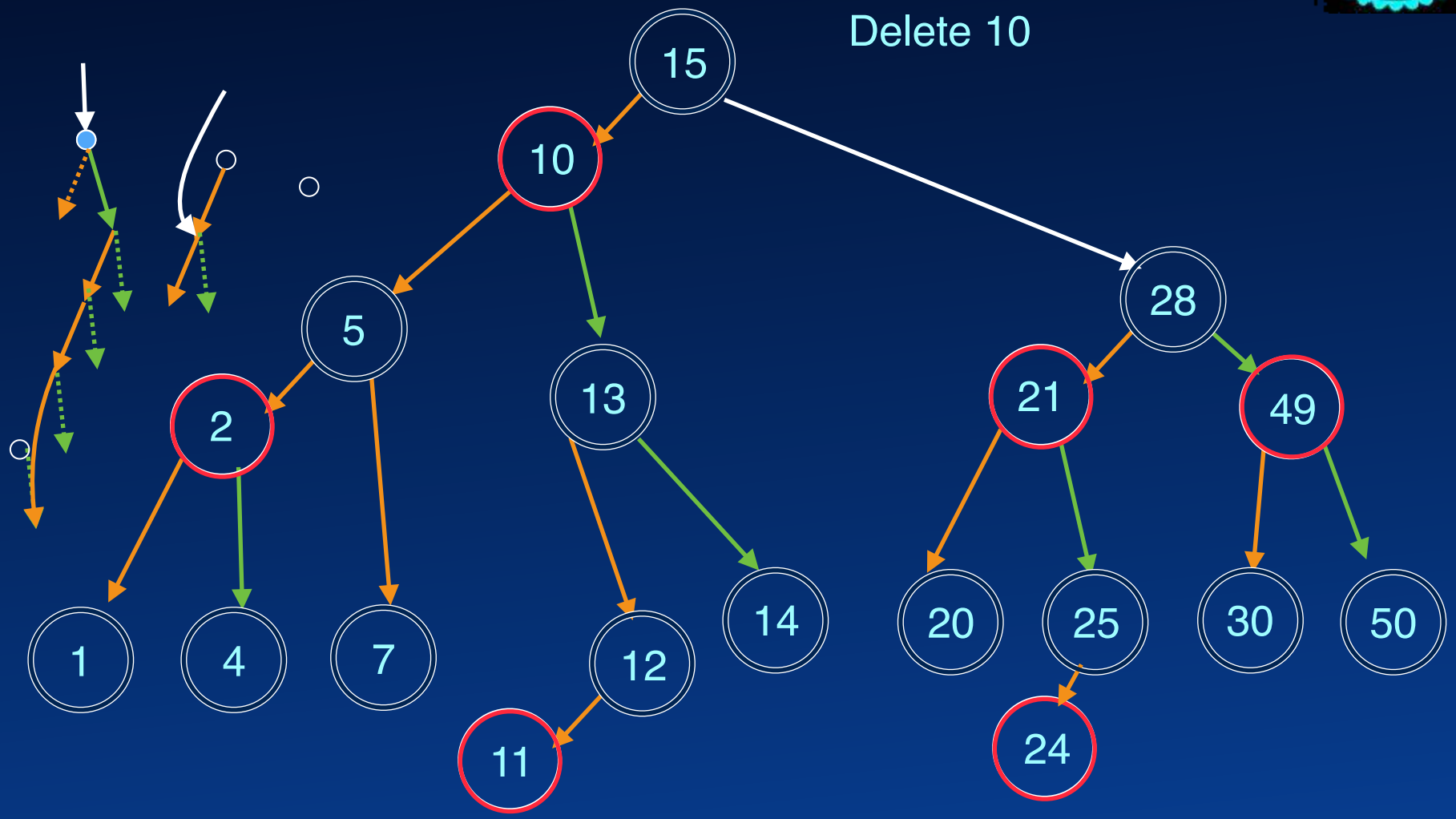
R-B Tree Deletion



R-B Tree Deletion

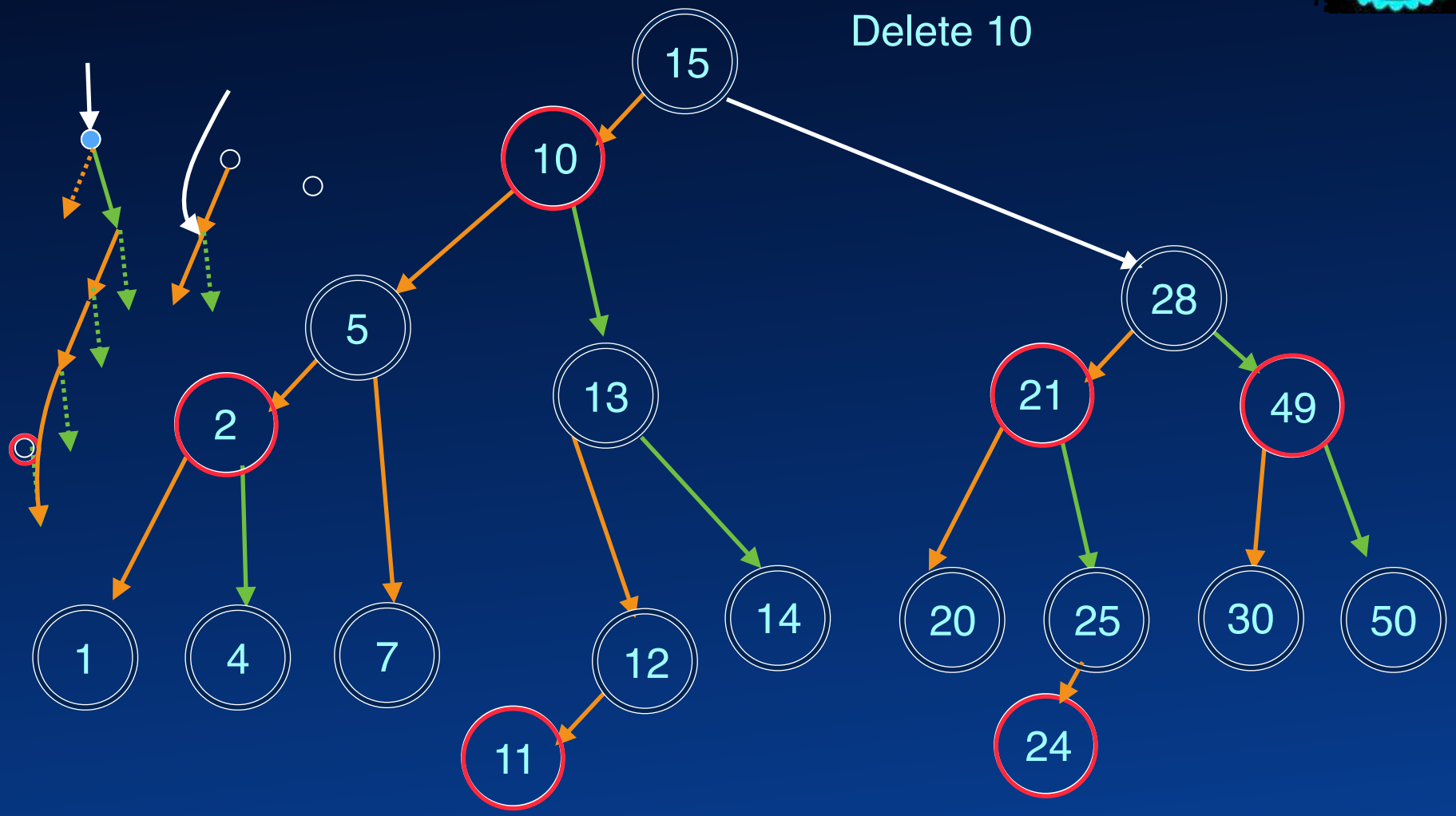


R-B Tree Deletion



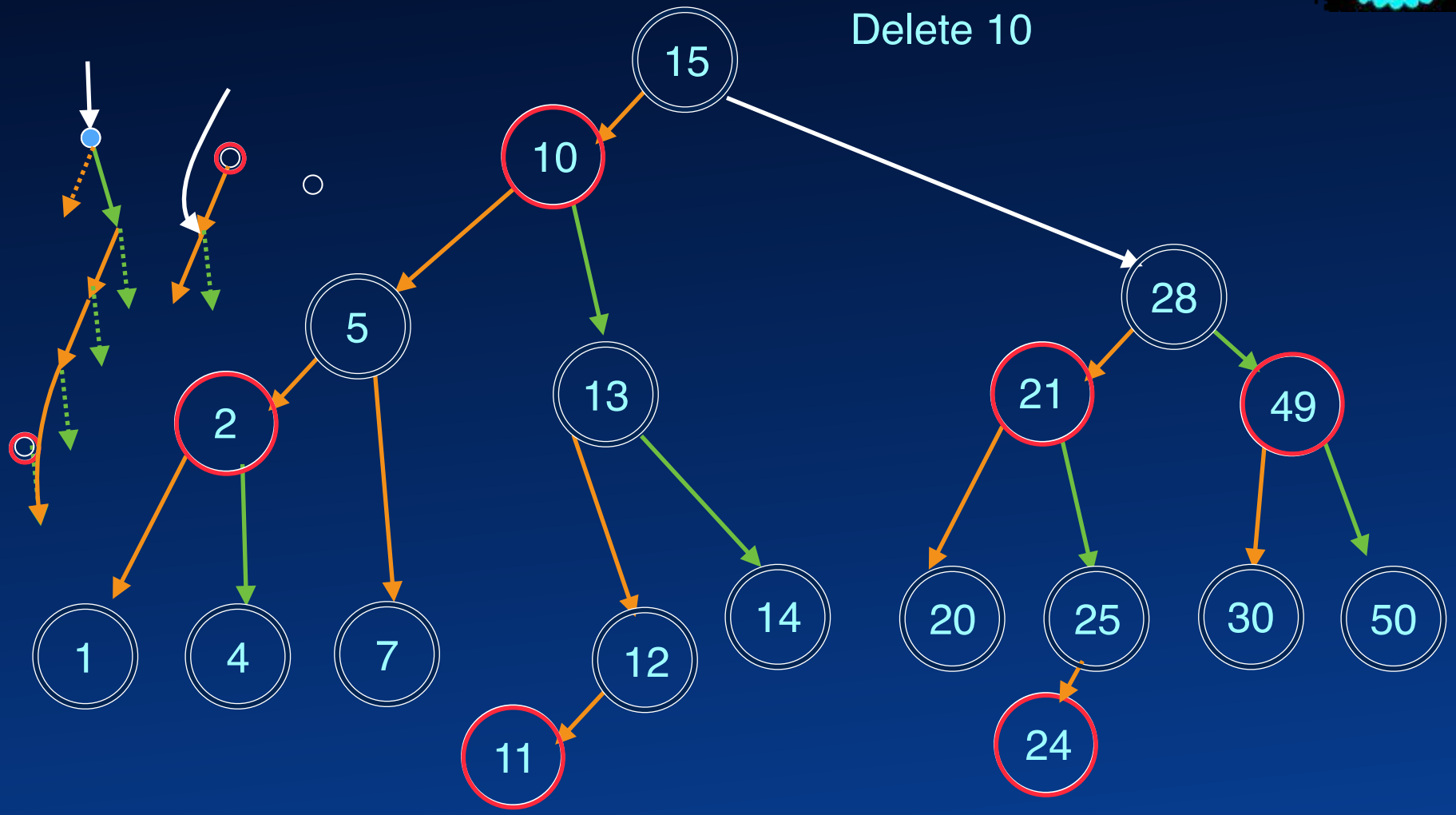
Expunge node ***n*** with @null in child.

R-B Tree Deletion



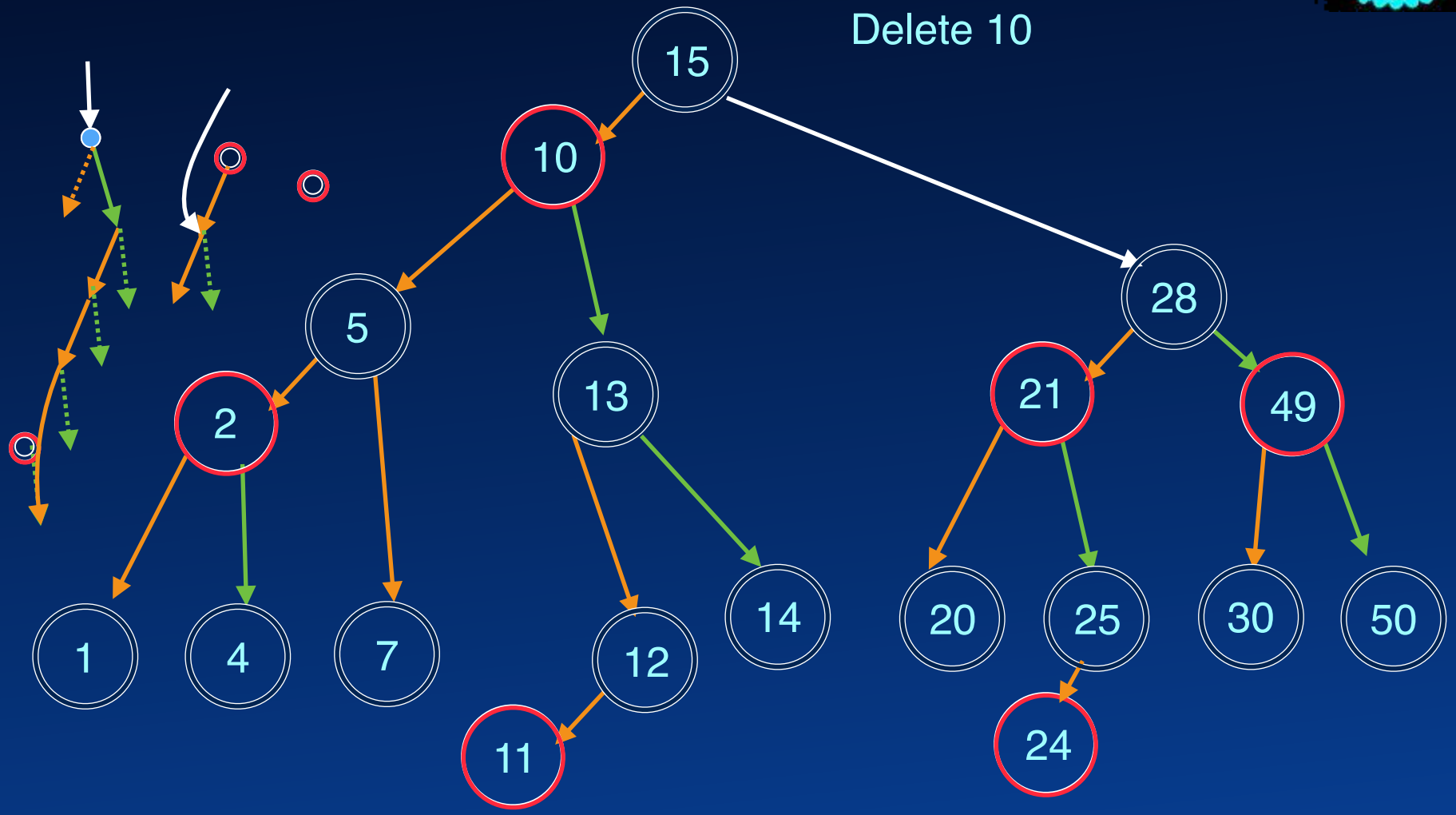
Expunge node ***n*** with @null in child.

R-B Tree Deletion



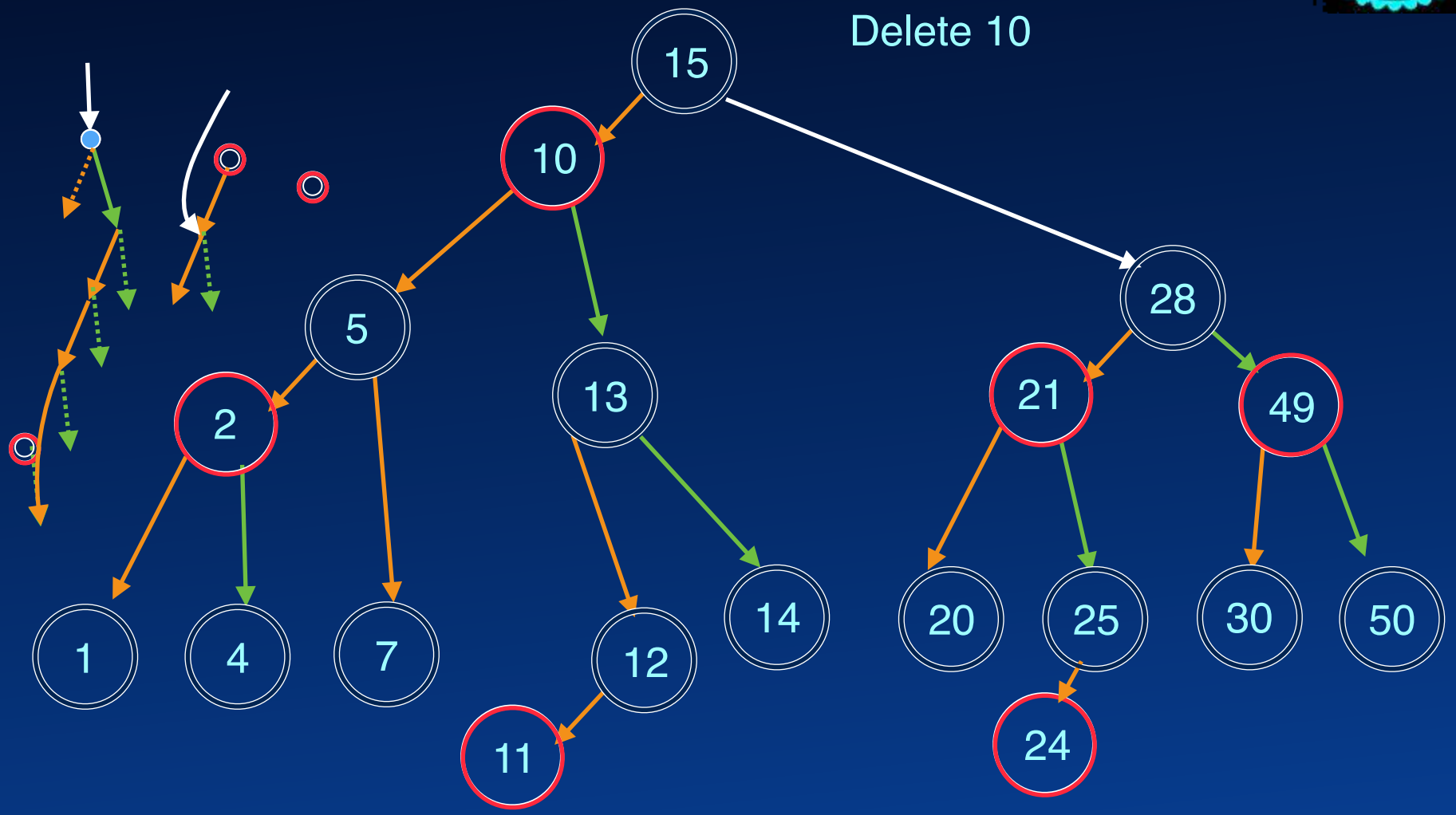
Expunge node ***n*** with @null in child.

R-B Tree Deletion



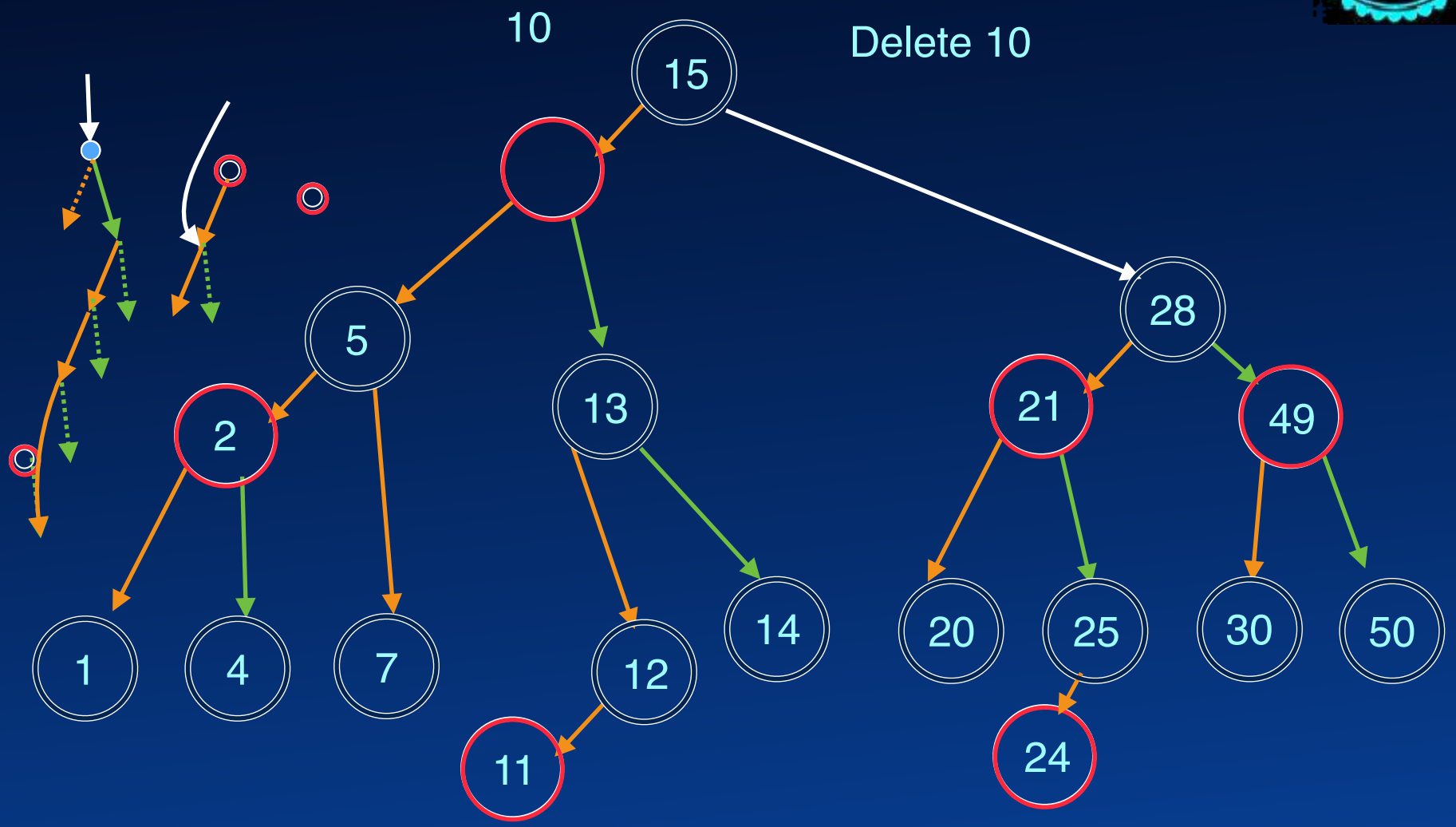
Expunge node ***n*** with @null in child.

R-B Tree Deletion



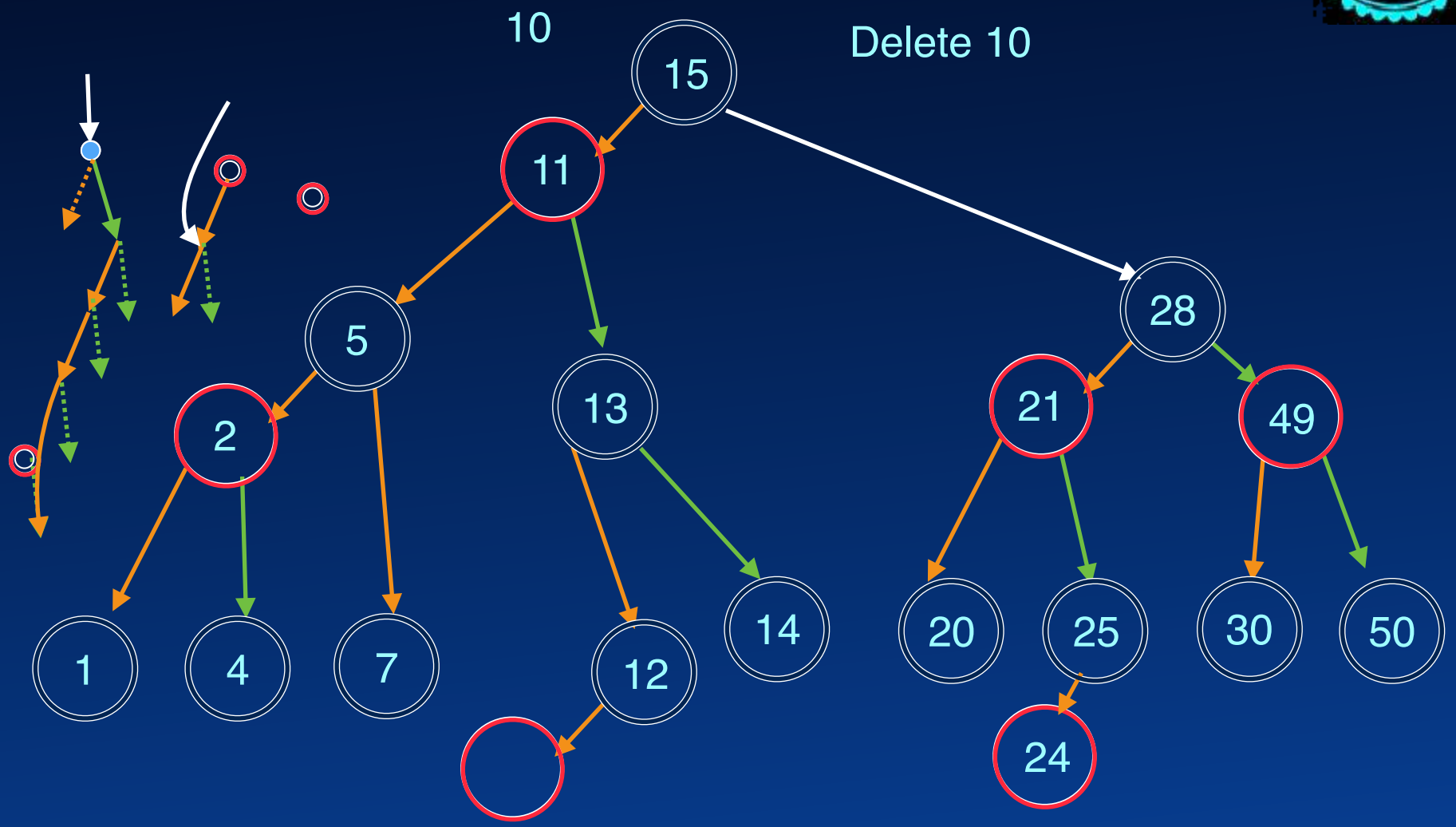
Expunge node ***n*** with @null in child. Simple if ***n*** is red.

R-B Tree Deletion



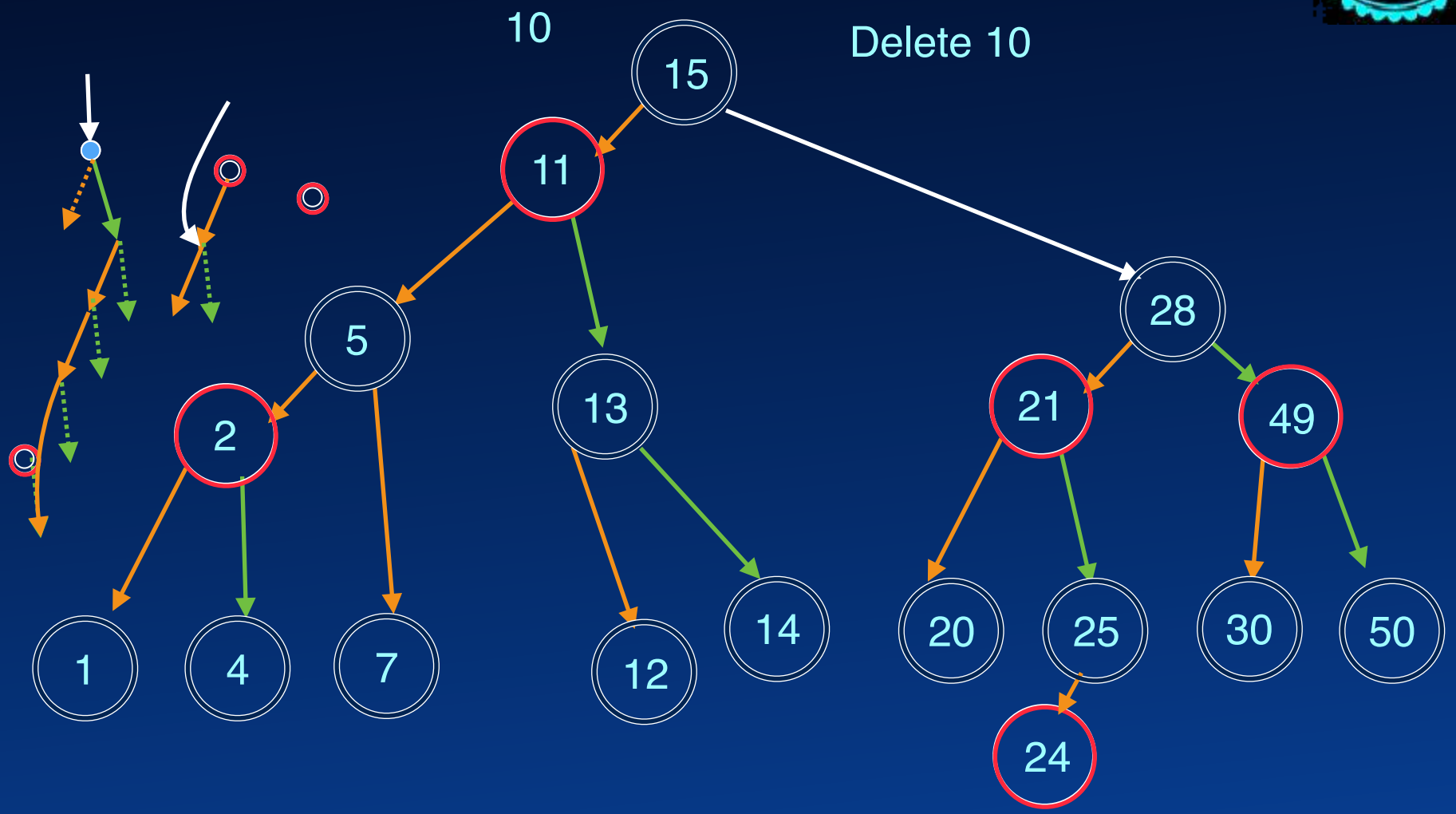
Expunge node ***n*** with @null in child. Simple if ***n*** is red.

R-B Tree Deletion



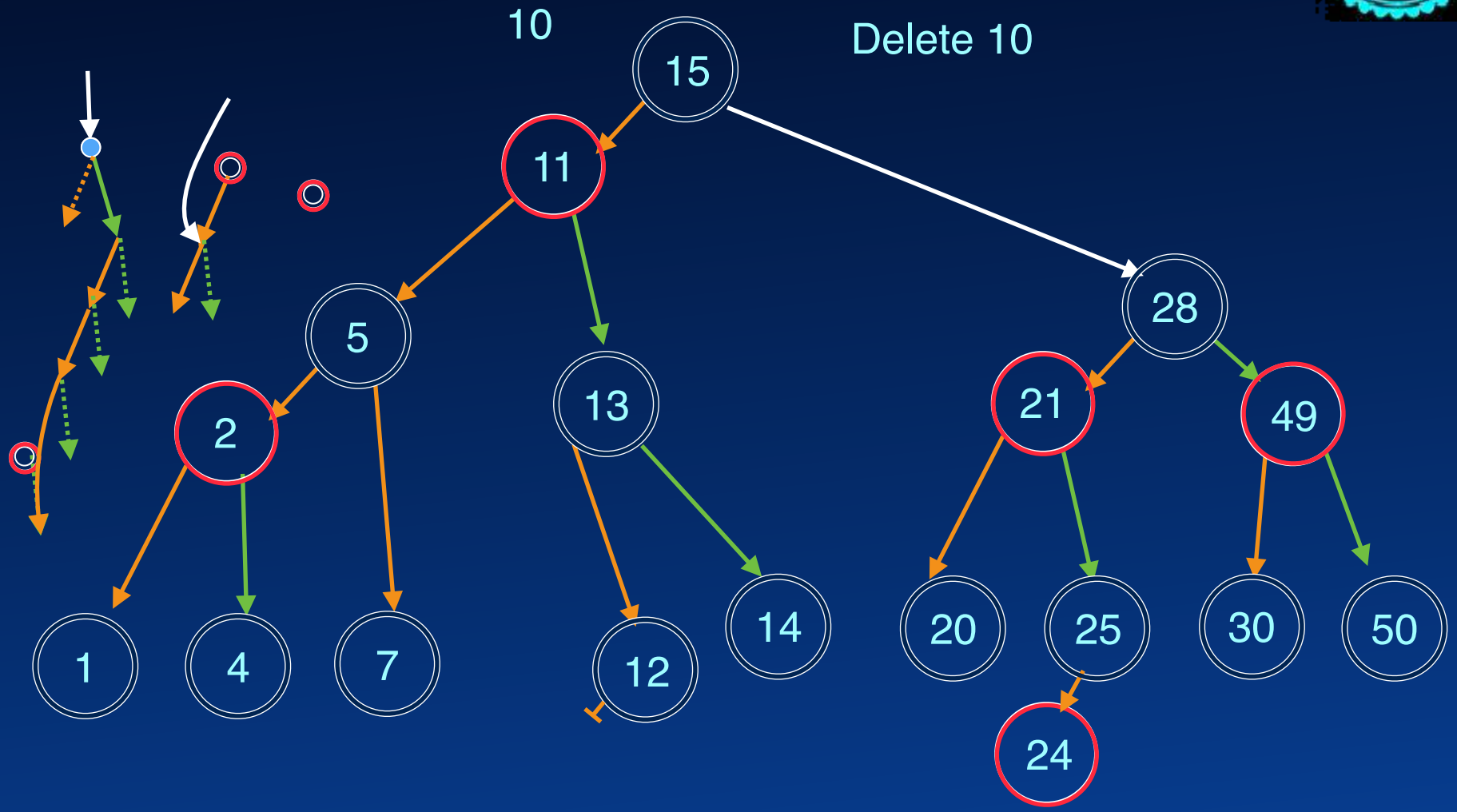
Expunge node ***n*** with @null in child. Simple if ***n*** is red.

R-B Tree Deletion



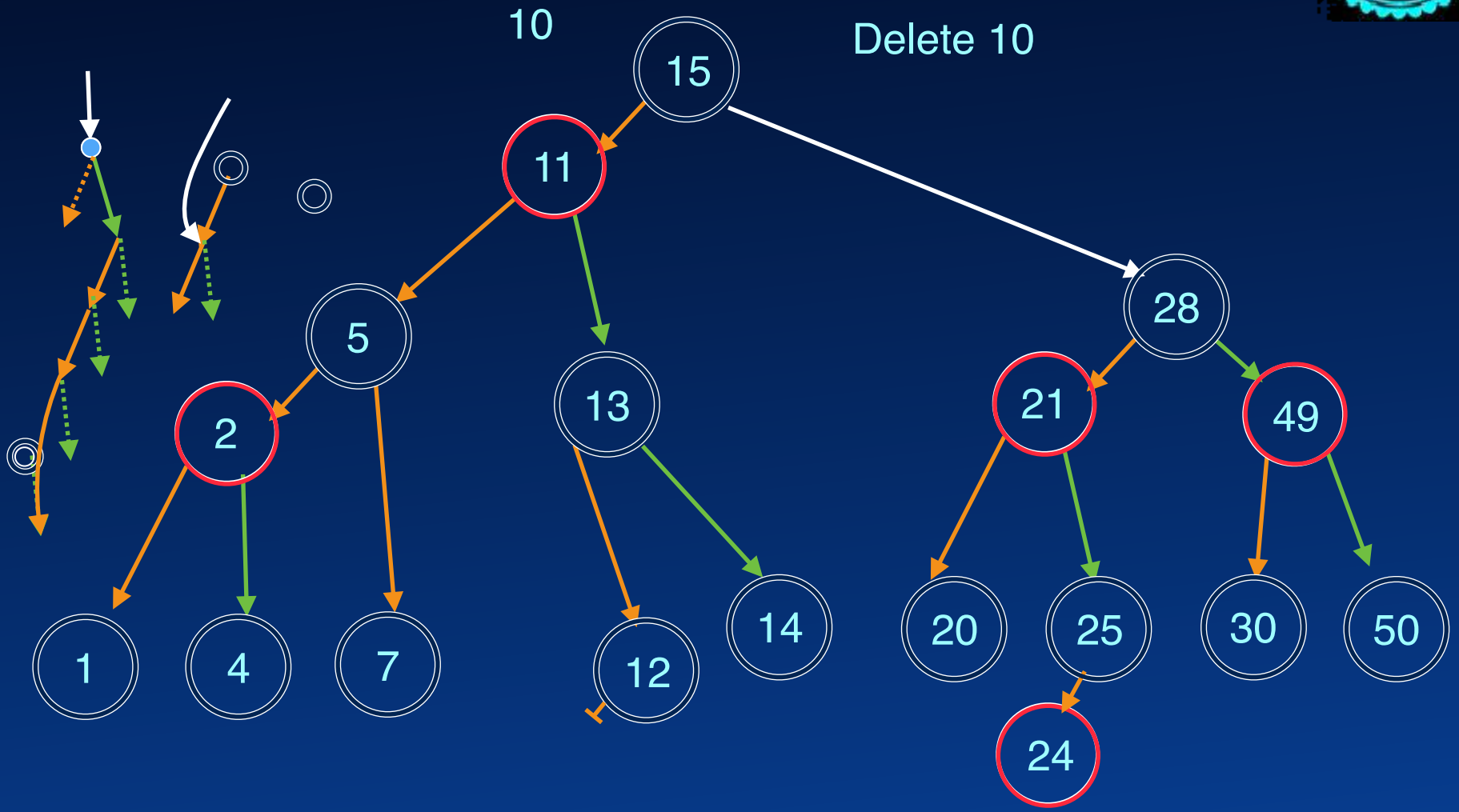
Expunge node ***n*** with @null in child. Simple if ***n*** is red.

R-B Tree Deletion



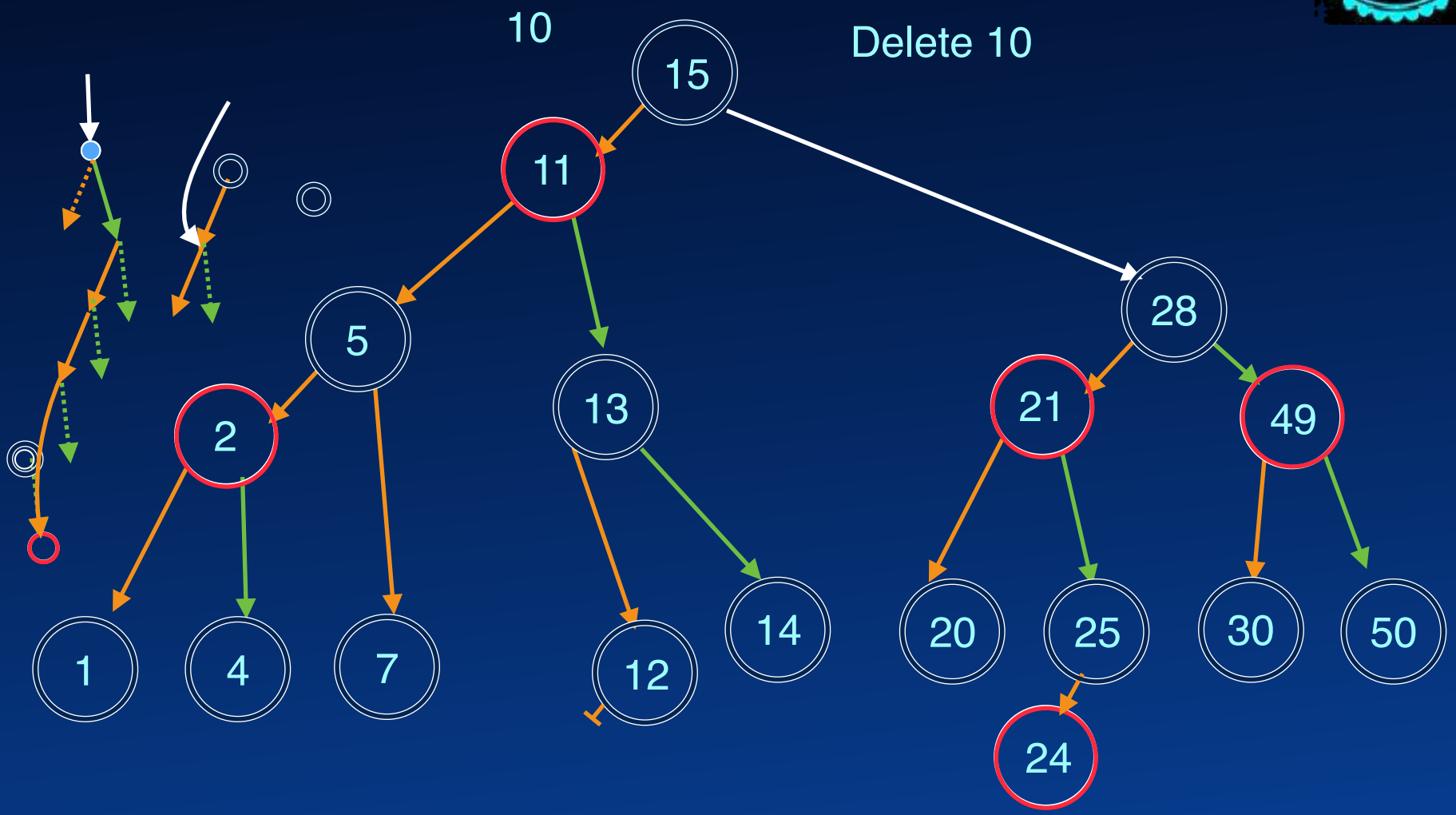
Expunge node ***n*** with @null in child. Simple if ***n*** is red.

R-B Tree Deletion



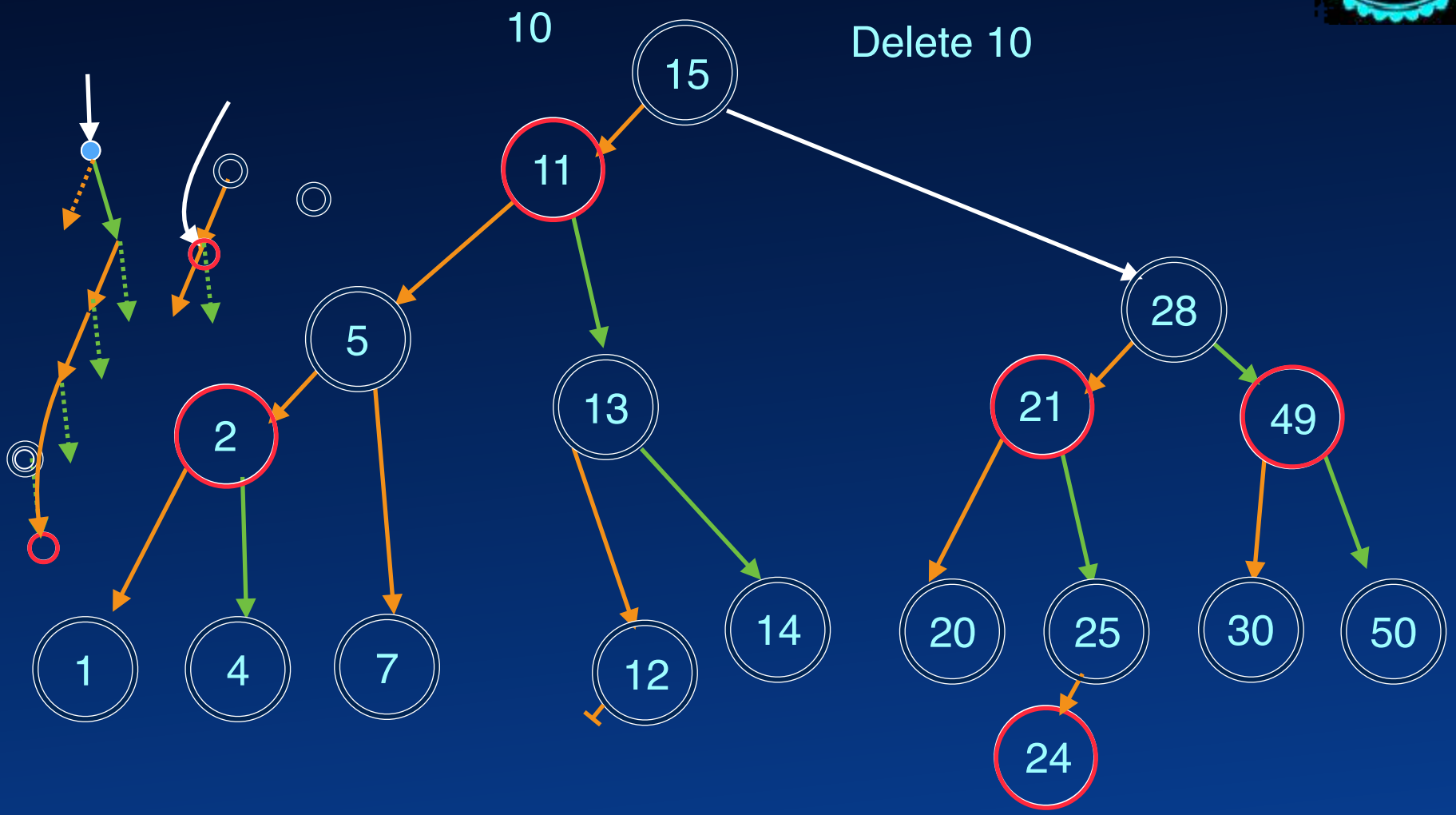
Expunge node ***n*** with @null in child. Simple if ***n*** is red.

R-B Tree Deletion



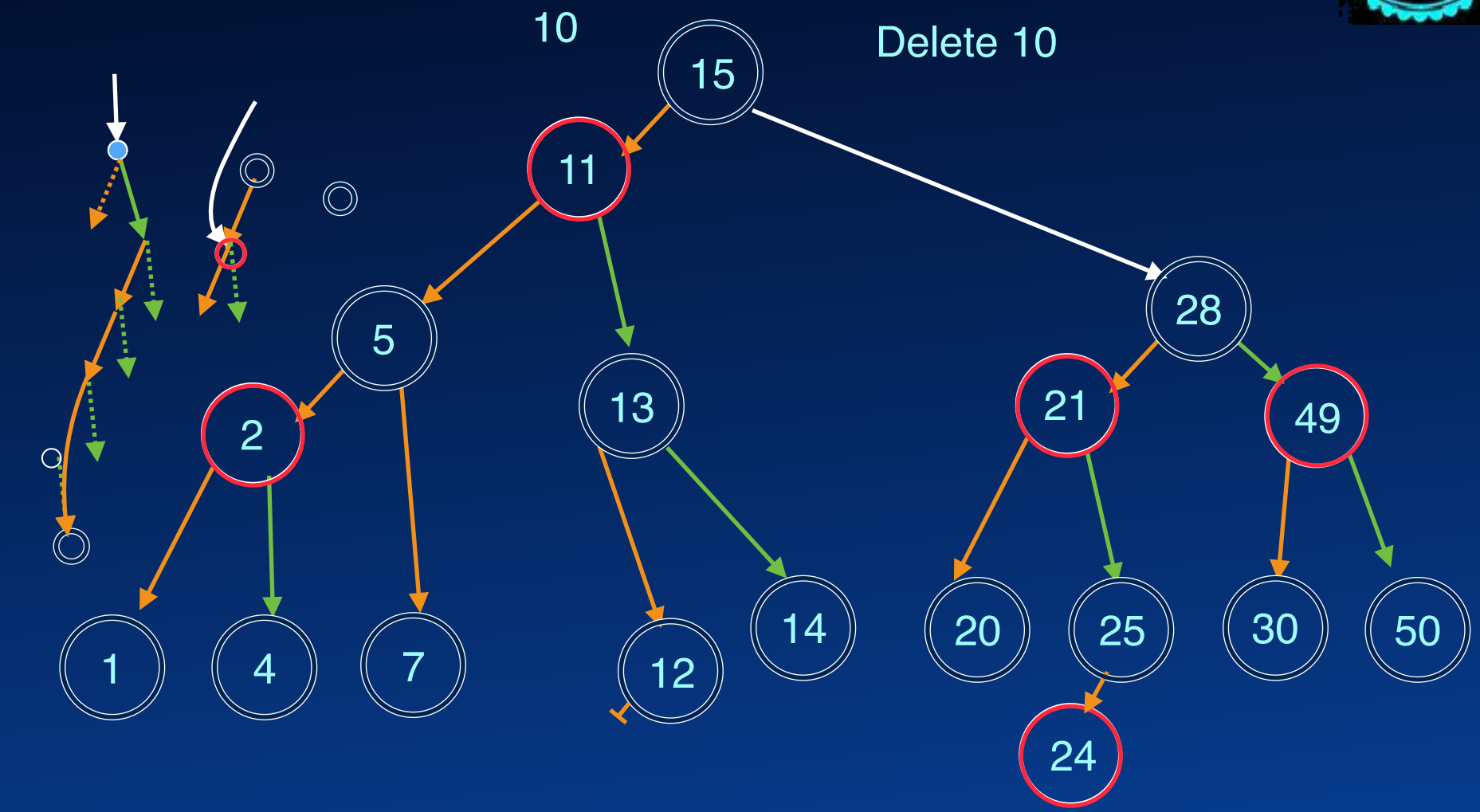
Expunge node ***n*** with @null in child. Simple if ***n*** is red.

R-B Tree Deletion



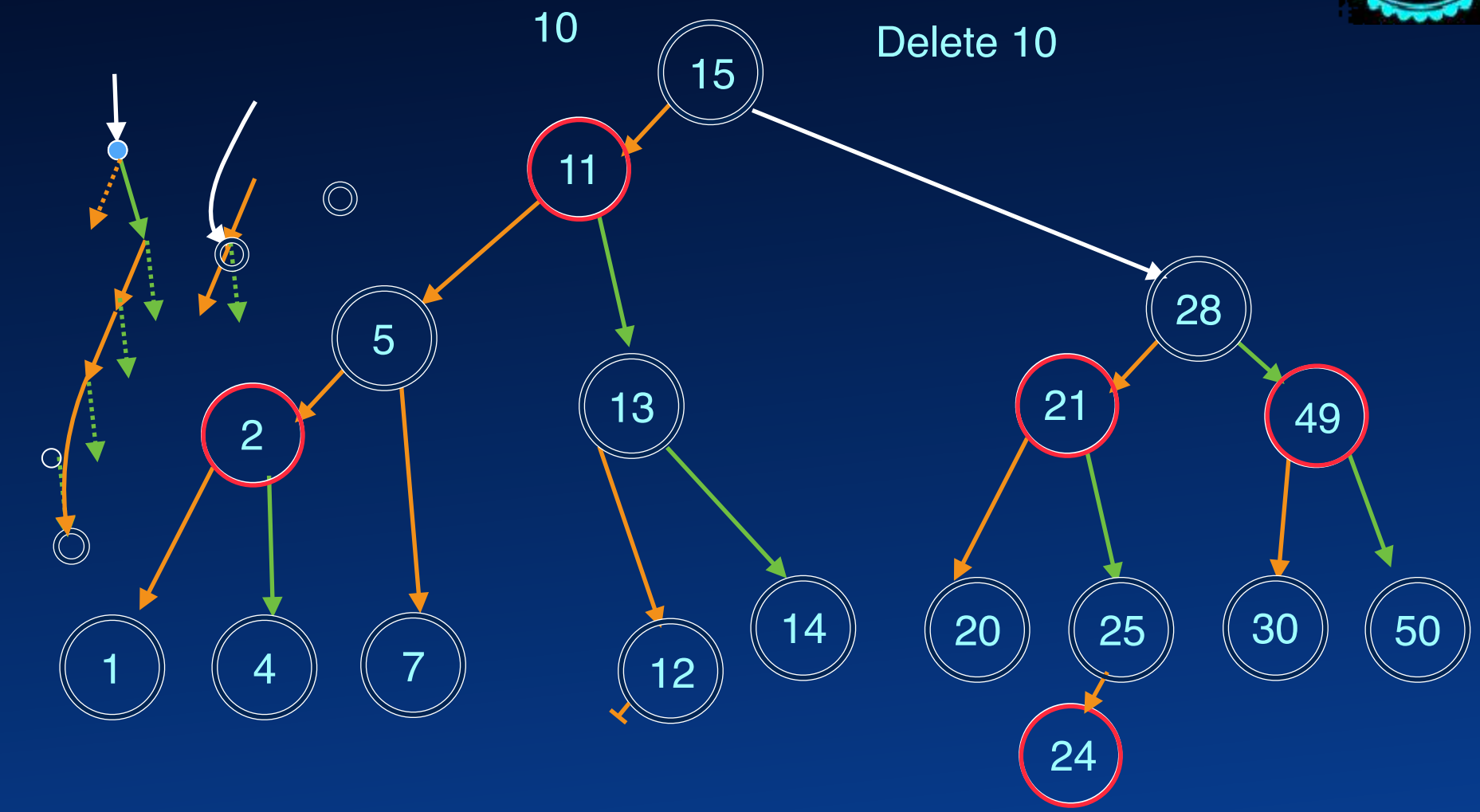
Expunge node ***n*** with @null in child. Simple if ***n*** is red.

R-B Tree Deletion



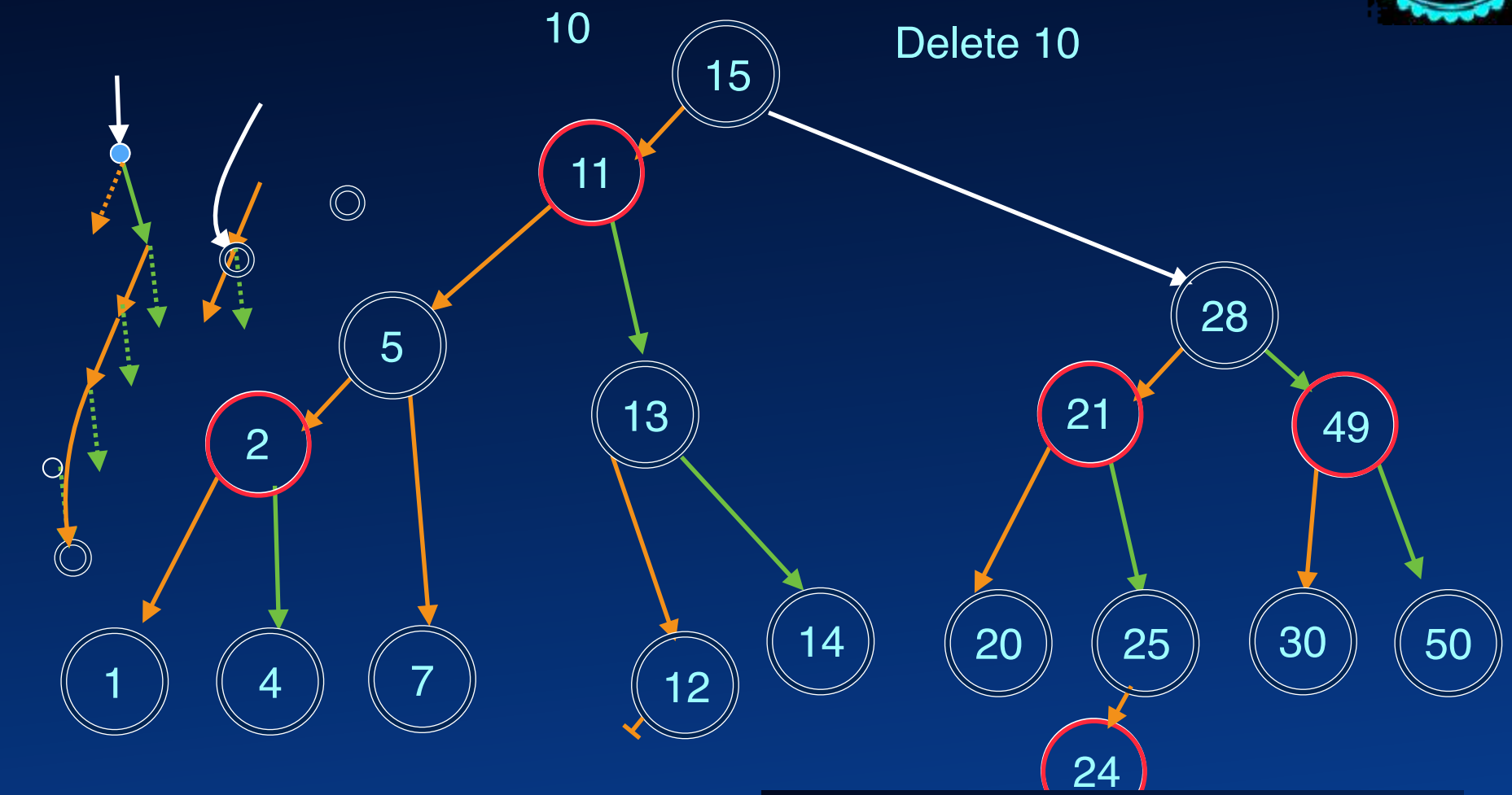
Expunge node ***n*** with @null in child. Simple if ***n*** is red.

R-B Tree Deletion



Expunge node ***n*** with @null in child. Simple if ***n*** is red.

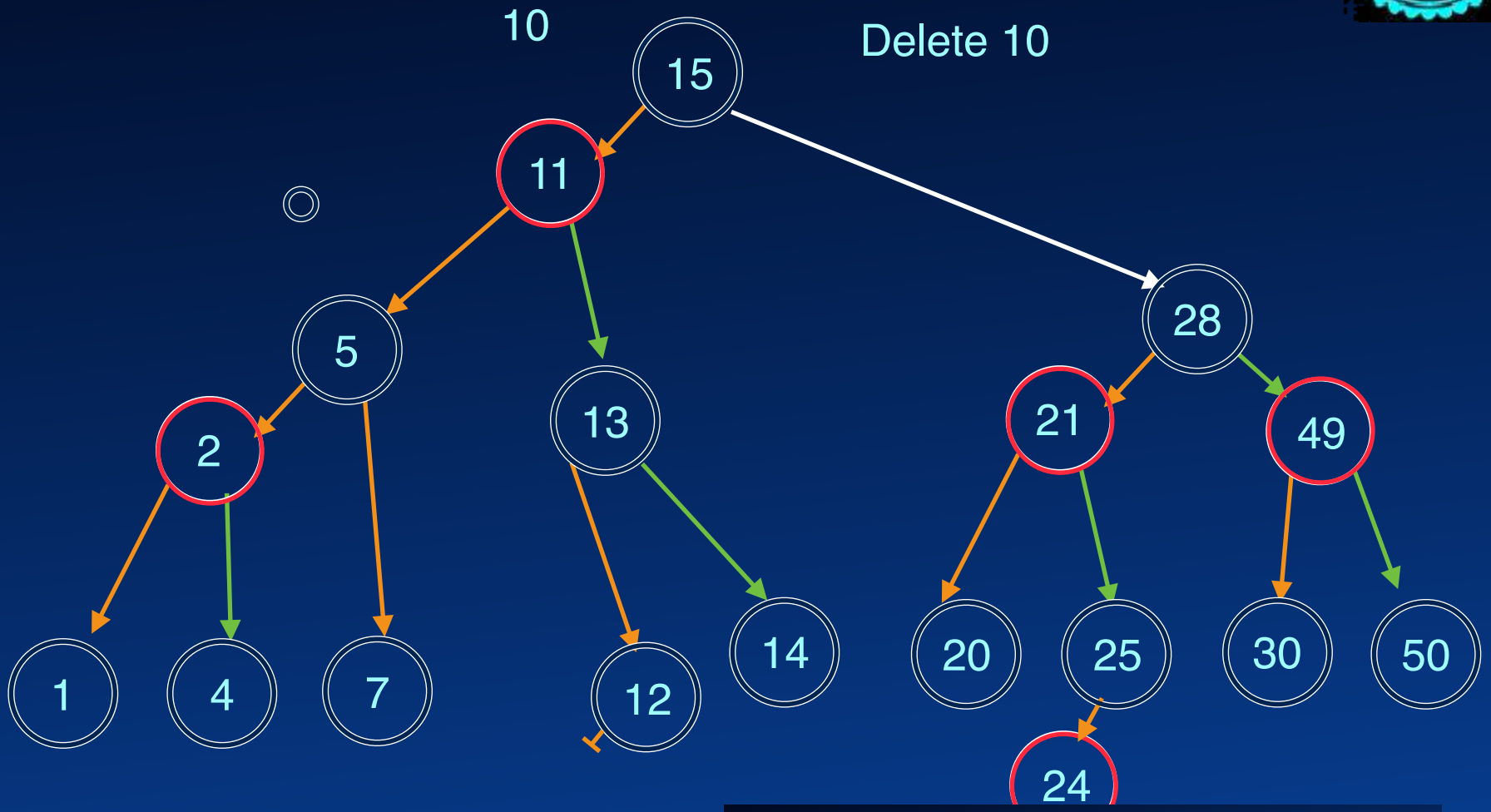
R-B Tree Deletion



Also simple if ***n*** 's child is red.

Expunge node ***n*** with @null in child. Simple if ***n*** is red.

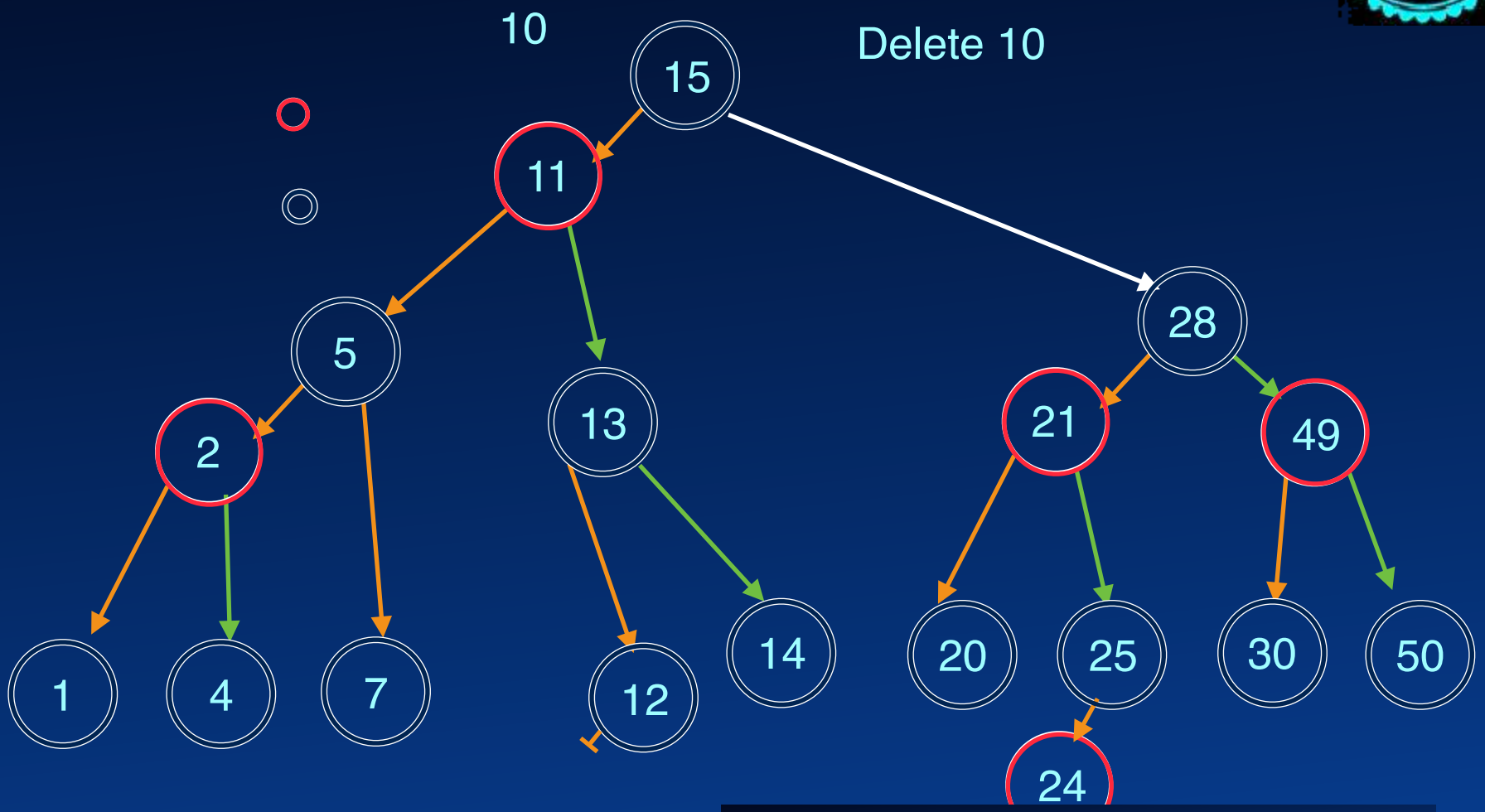
R-B Tree Deletion



Also simple if ***n*** 's child is red.

Expunge node ***n*** with @null in child. Simple if ***n*** is red.

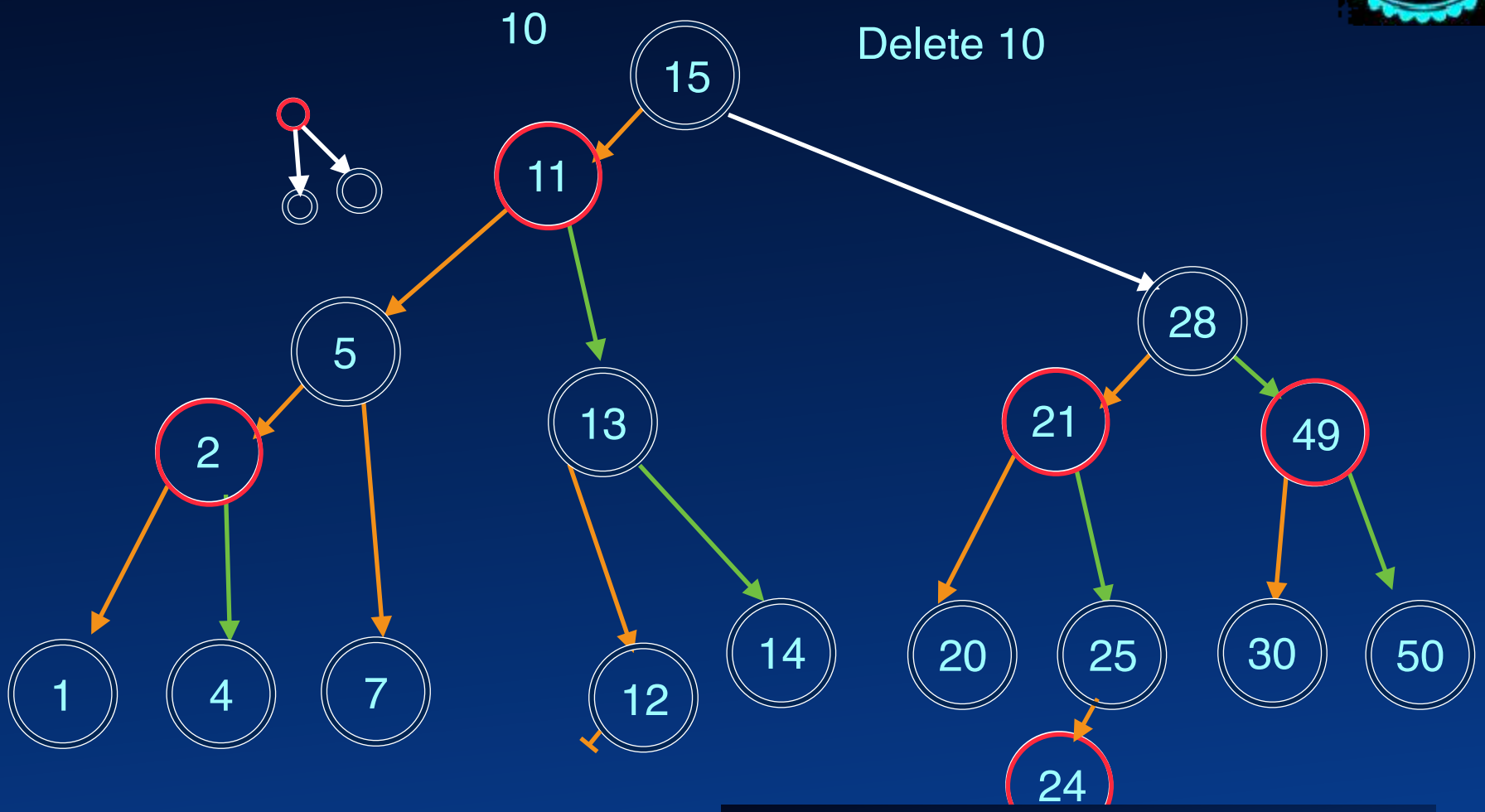
R-B Tree Deletion



Also simple if ***n*** 's child is red.

Expunge node ***n*** with @null in child. Simple if ***n*** is red.

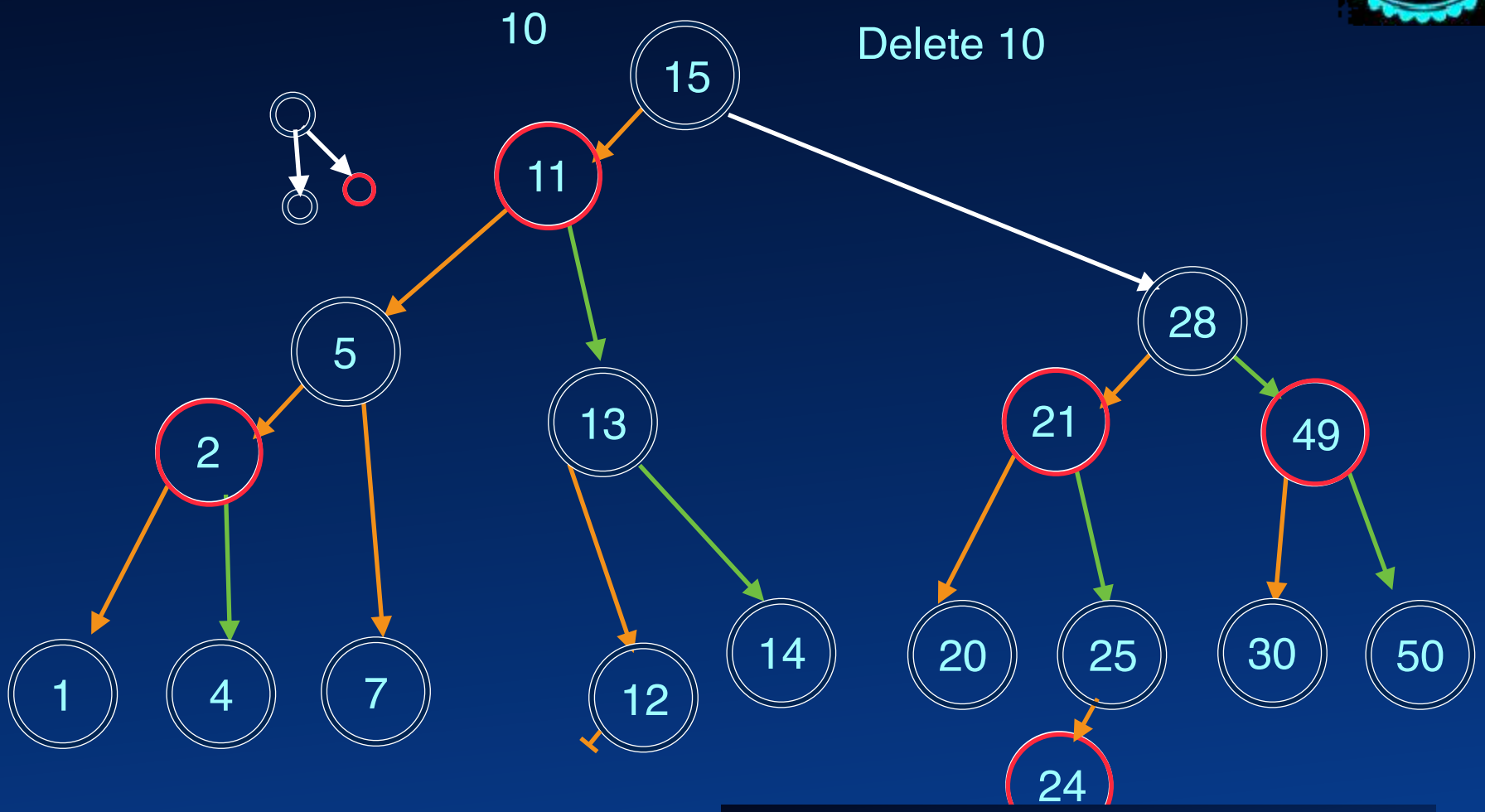
R-B Tree Deletion



Also simple if ***n*** 's child is red.

Expunge node ***n*** with @null in child. Simple if ***n*** is red.

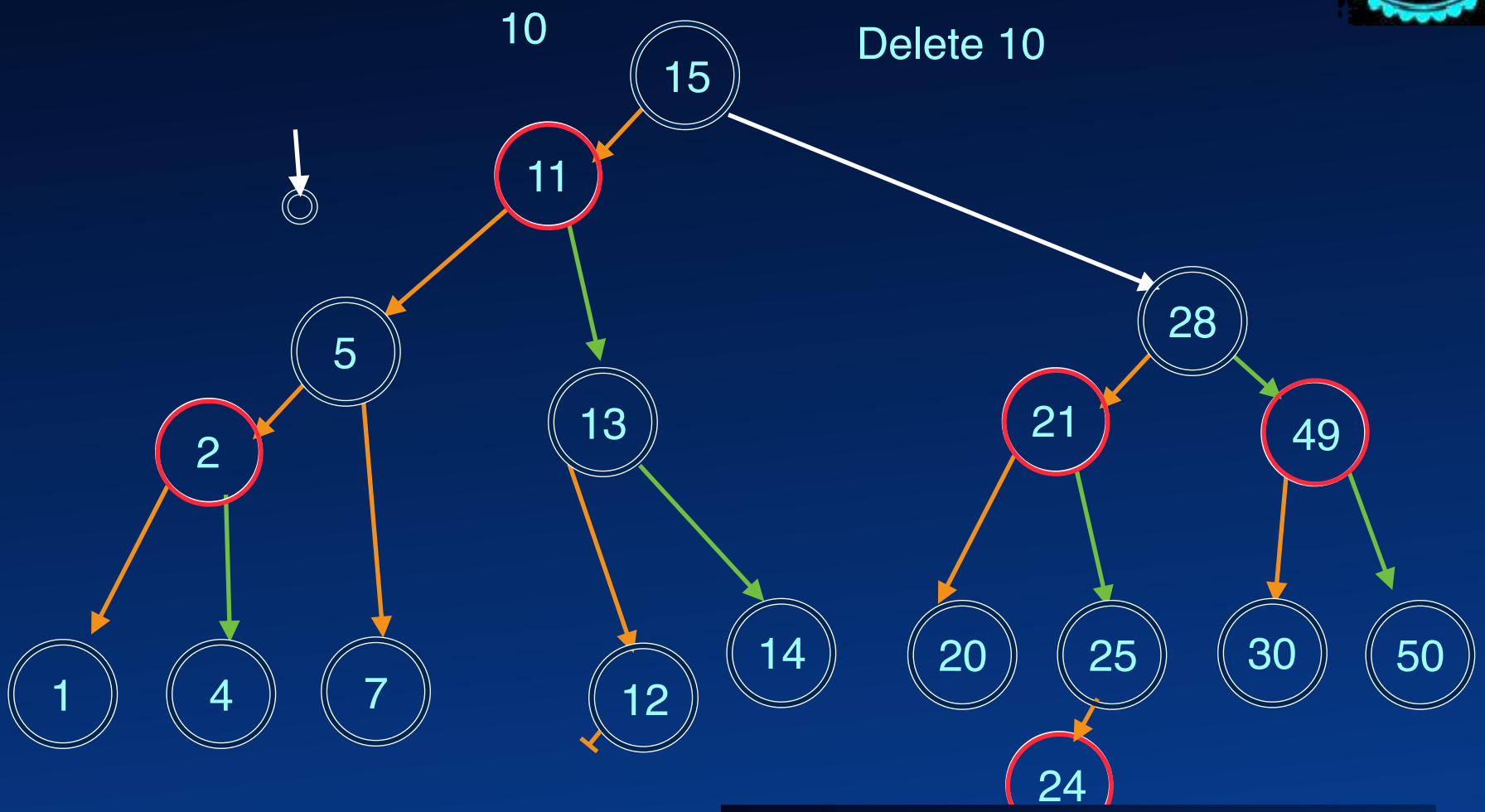
R-B Tree Deletion



Also simple if ***n*** 's child is red.

Expunge node ***n*** with @null in child. Simple if ***n*** is red.

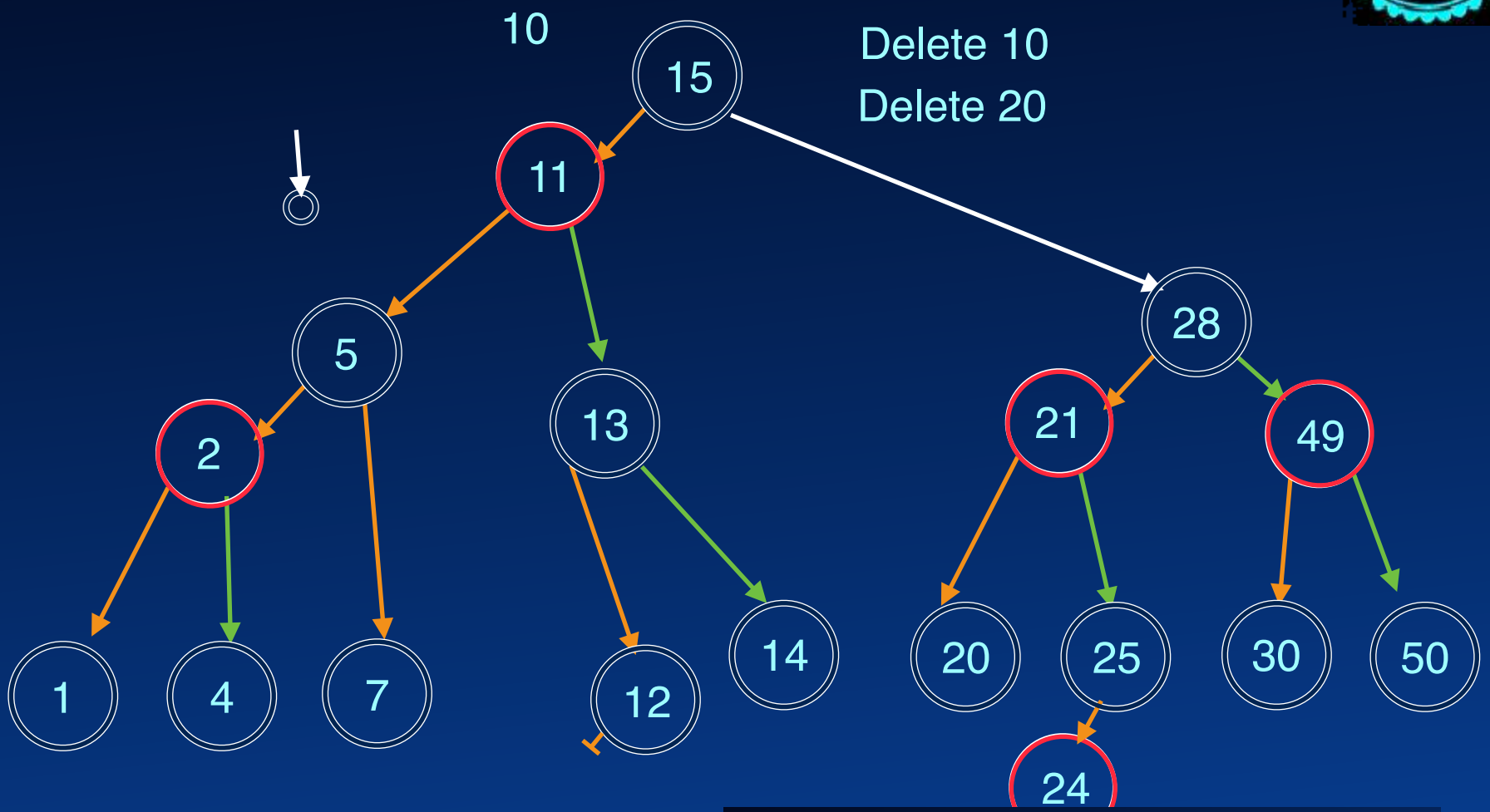
R-B Tree Deletion



Also simple if ***n*** 's child is red.

Expunge node ***n*** with @null in child. Simple if ***n*** is red.

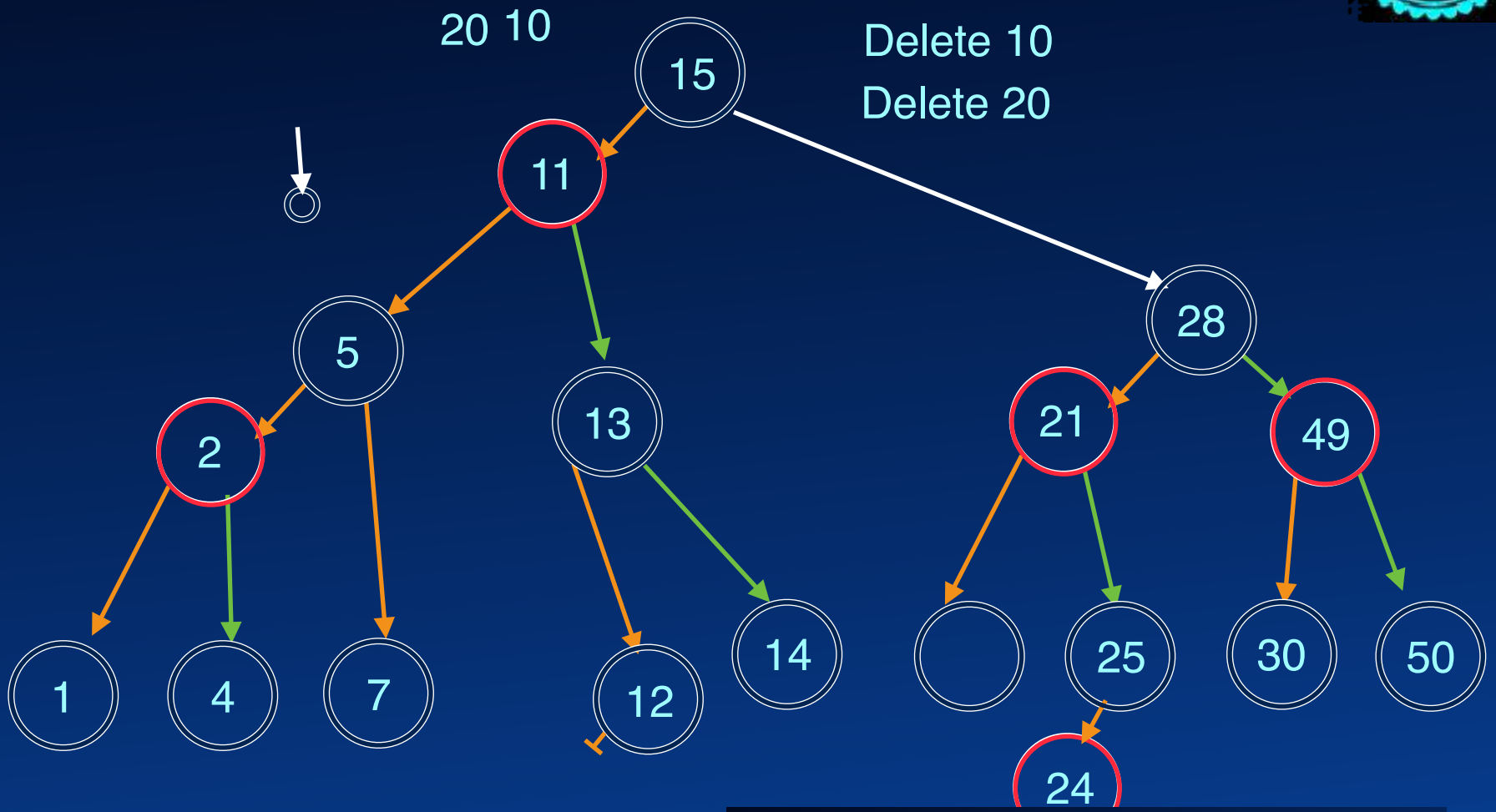
R-B Tree Deletion



Also simple if ***n*** 's child is red.

Expunge node ***n*** with @null in child. Simple if ***n*** is red.

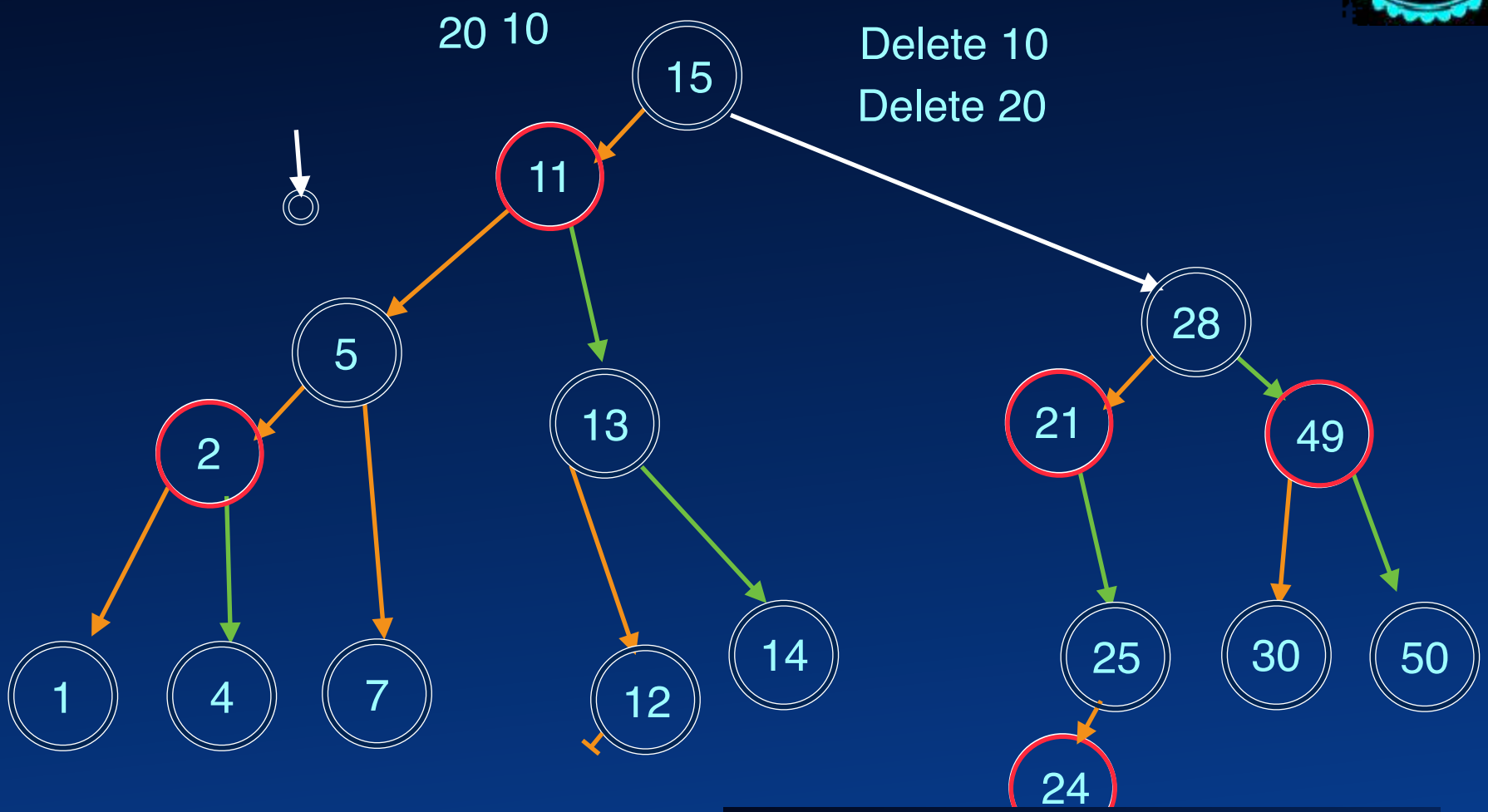
R-B Tree Deletion



Also simple if ***n*** 's child is red.

Expunge node ***n*** with @null in child. Simple if ***n*** is red.

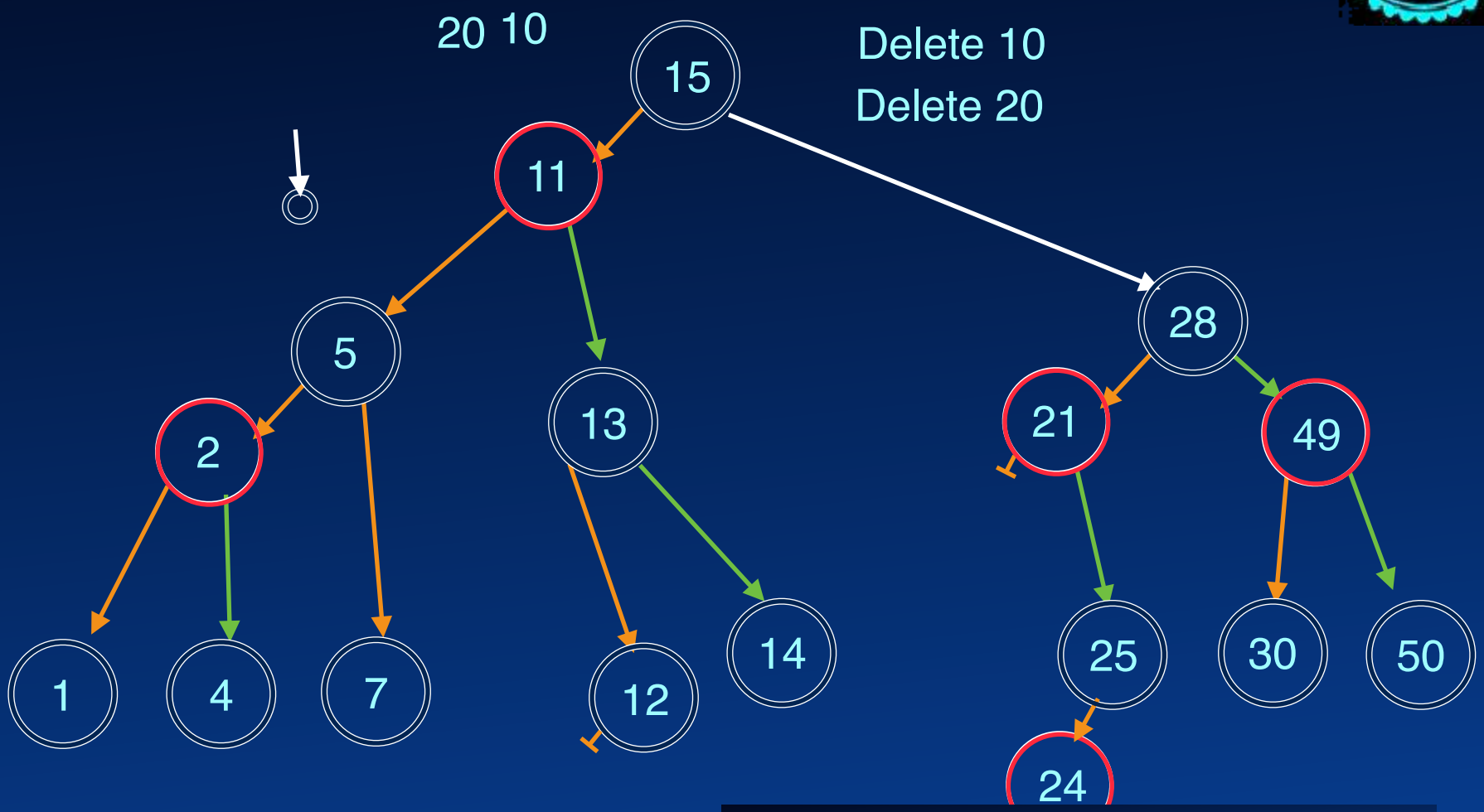
R-B Tree Deletion



Also simple if ***n*** 's child is red.

Expunge node ***n*** with @null in child. Simple if ***n*** is red.

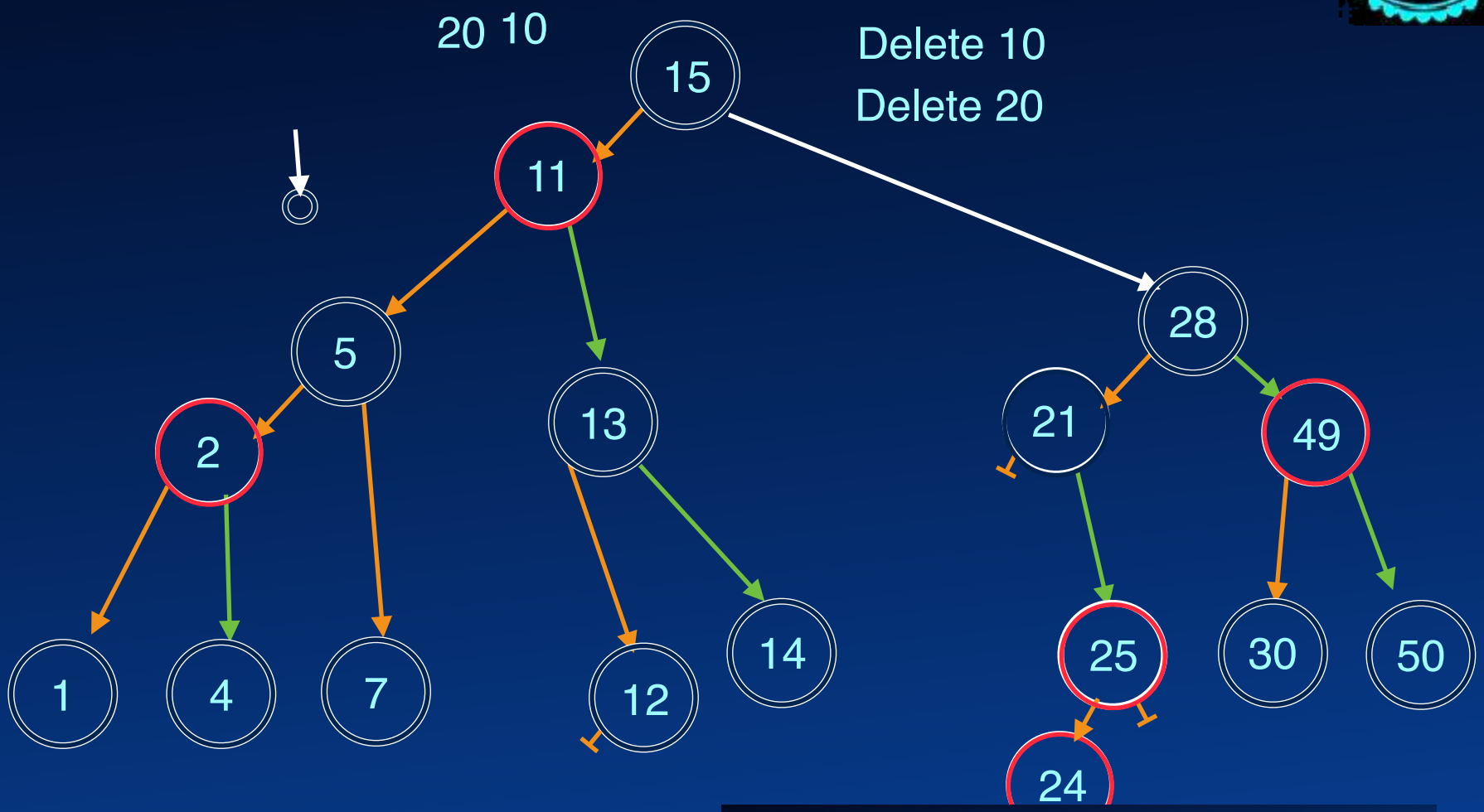
R-B Tree Deletion



Also simple if ***n*** 's child is red.

Expunge node ***n*** with @null in child. Simple if ***n*** is red.

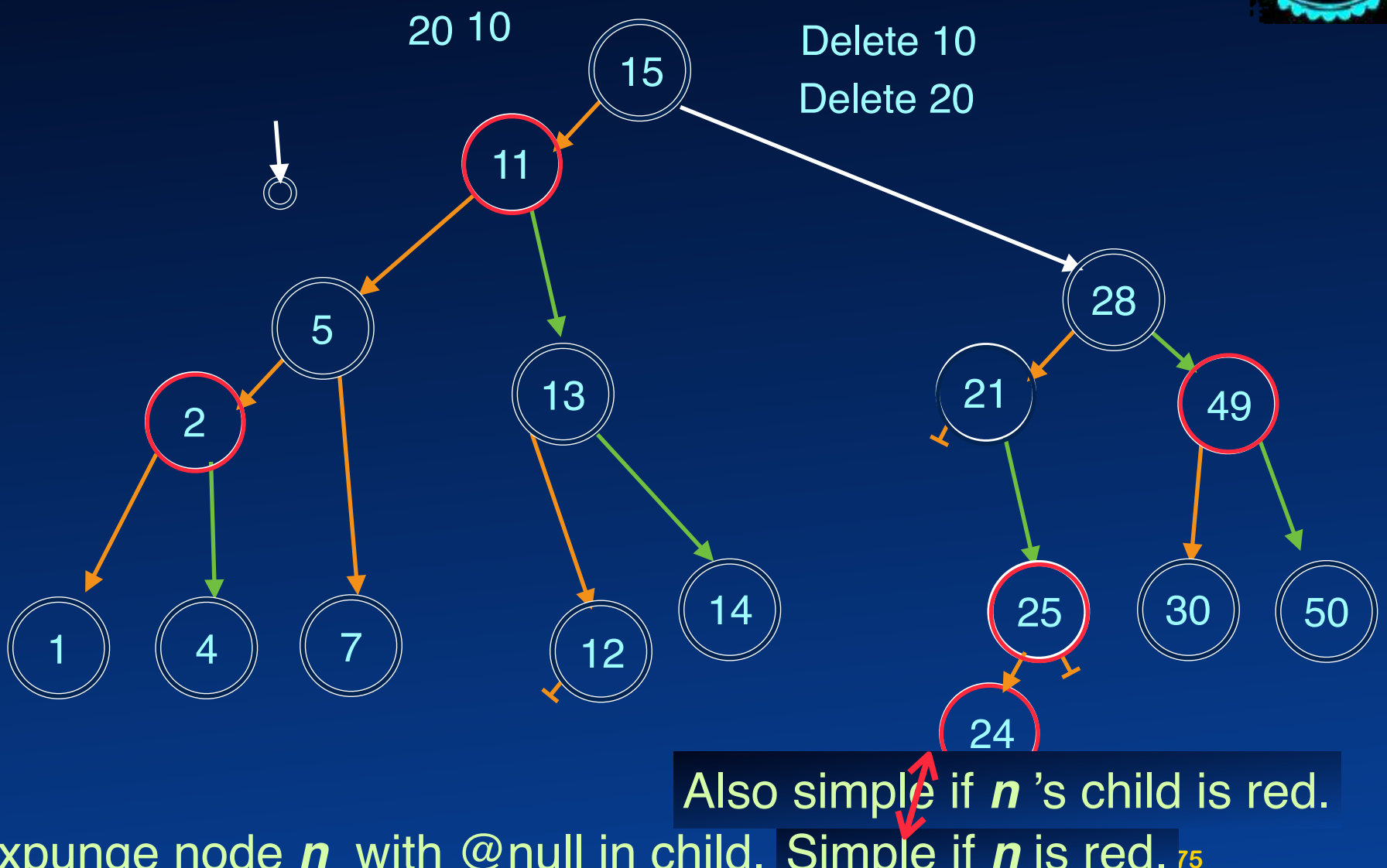
R-B Tree Deletion



Also simple if ***n*** 's child is red.

Expunge node ***n*** with @null in child. Simple if ***n*** is red.

R-B Tree Deletion

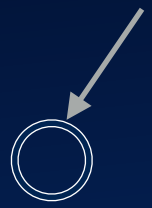


Also simple if ***n*** 's child is red.

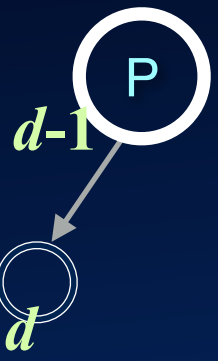
Expunge node ***n*** with @null in child. Simple if ***n*** is red.

Complex Delete Cases

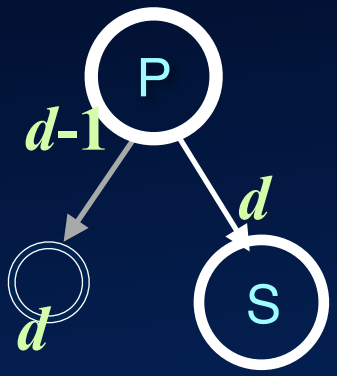
Complex Delete Cases



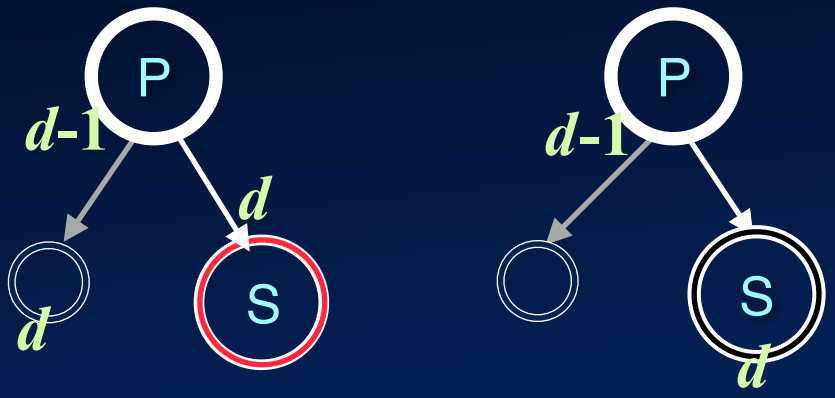
Complex Delete Cases



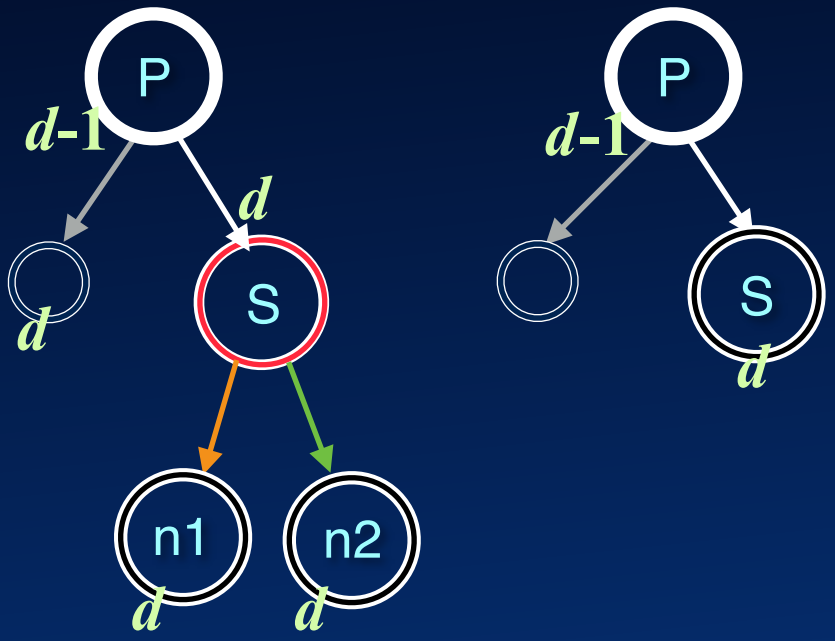
Complex Delete Cases



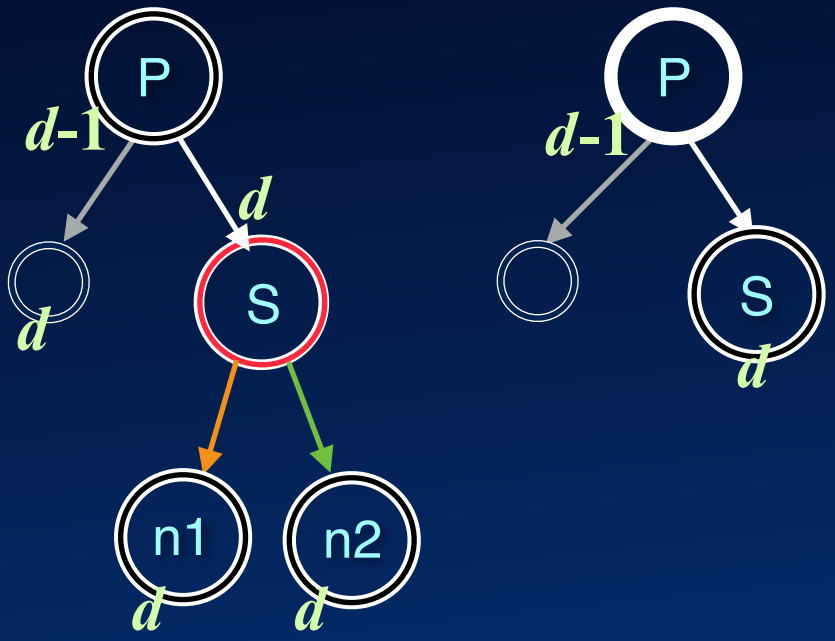
Complex Delete Cases



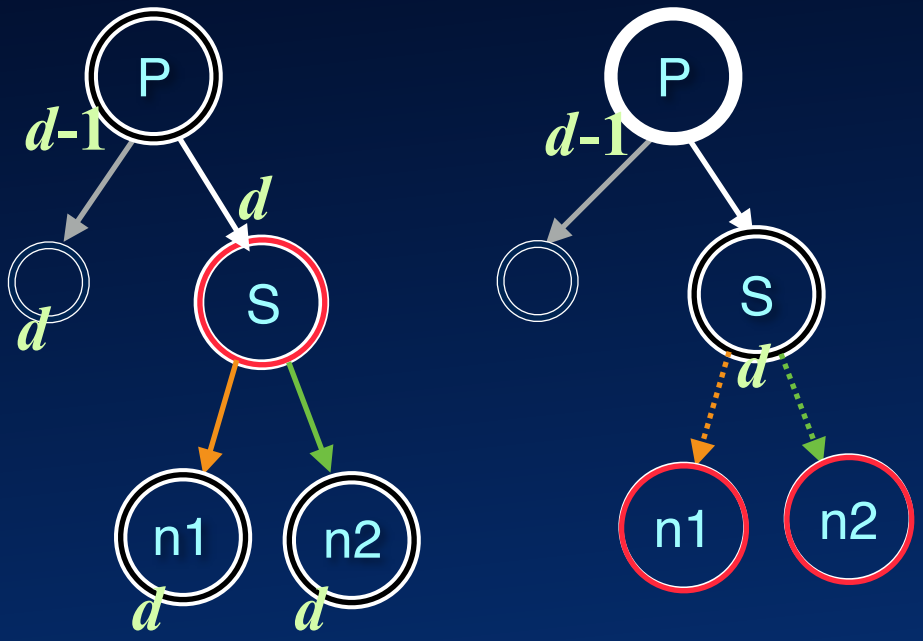
Complex Delete Cases



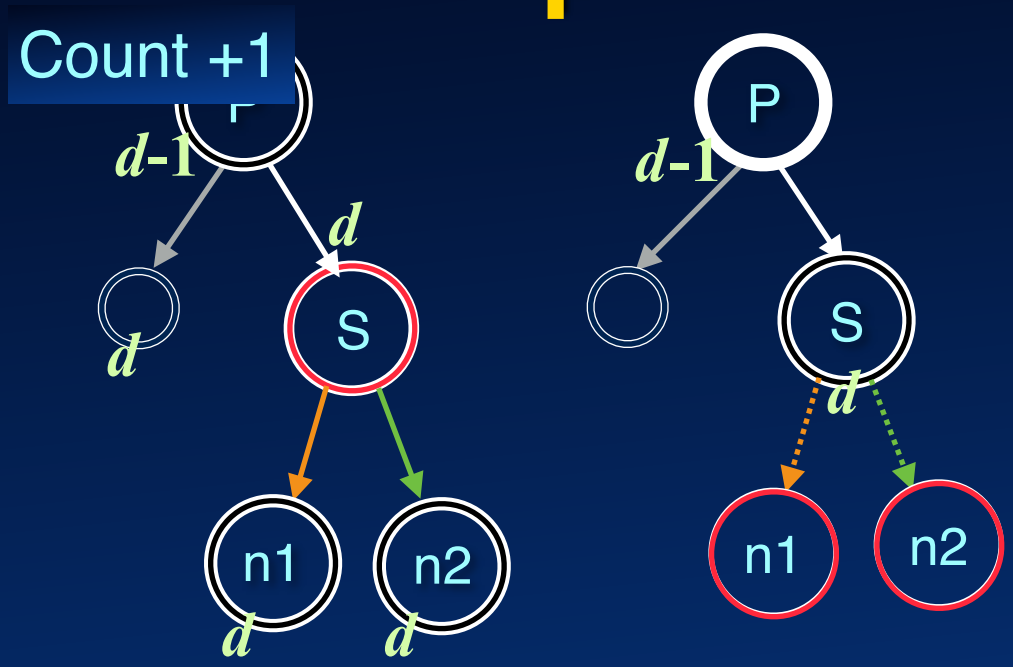
Complex Delete Cases



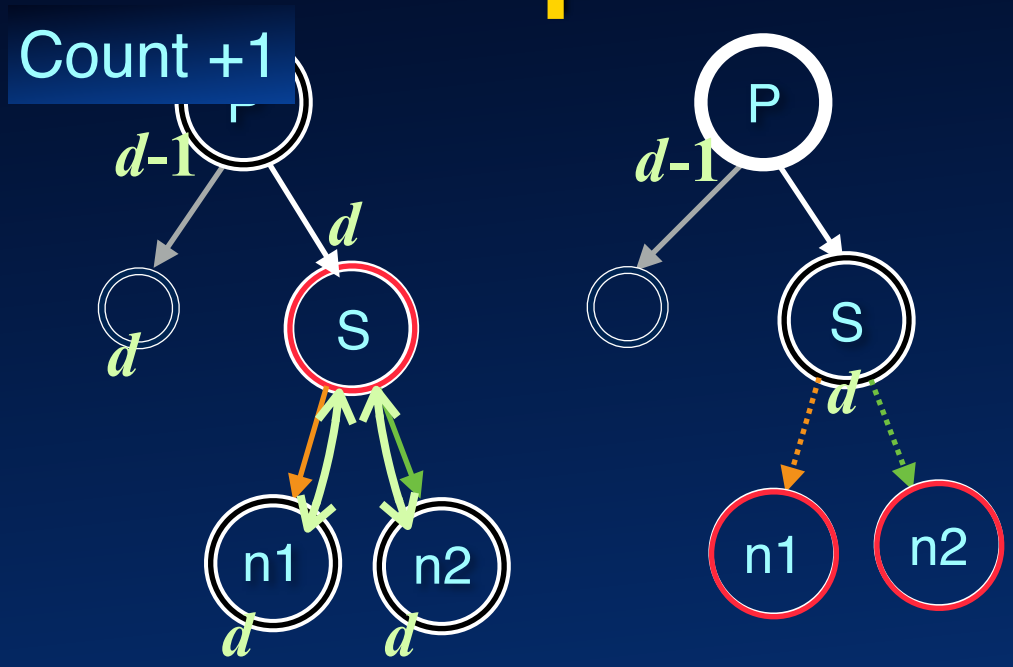
Complex Delete Cases



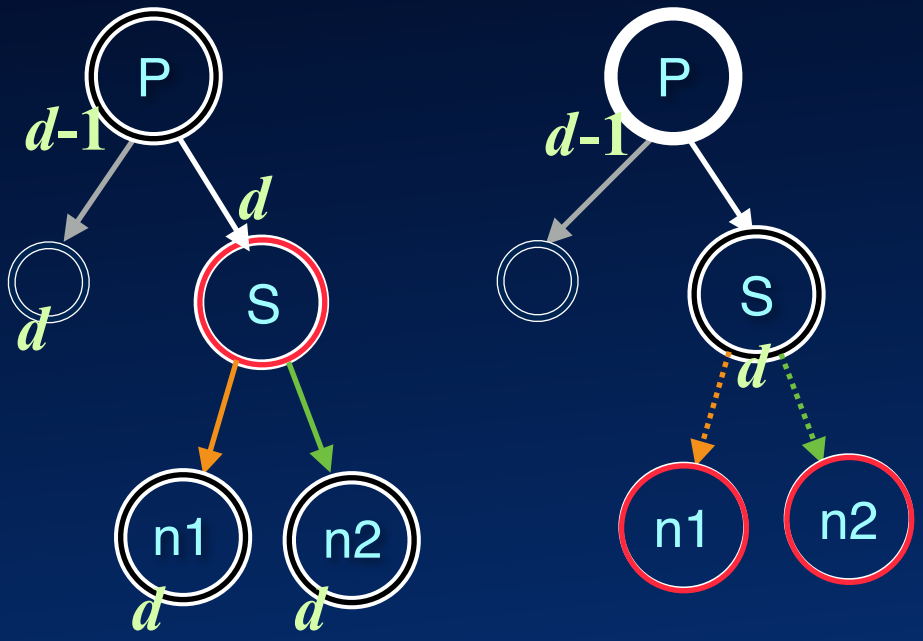
Complex Delete Cases



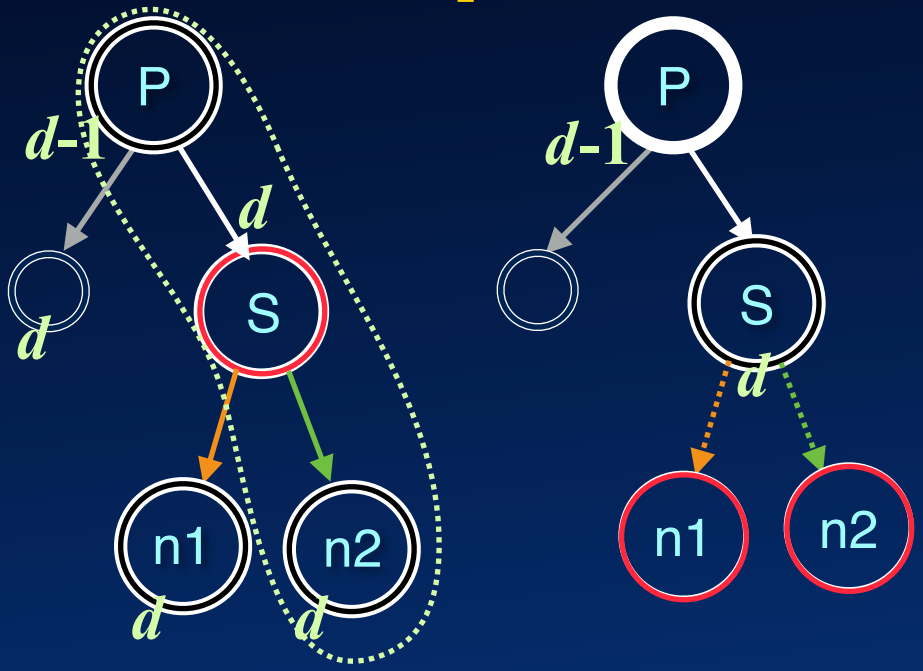
Complex Delete Cases



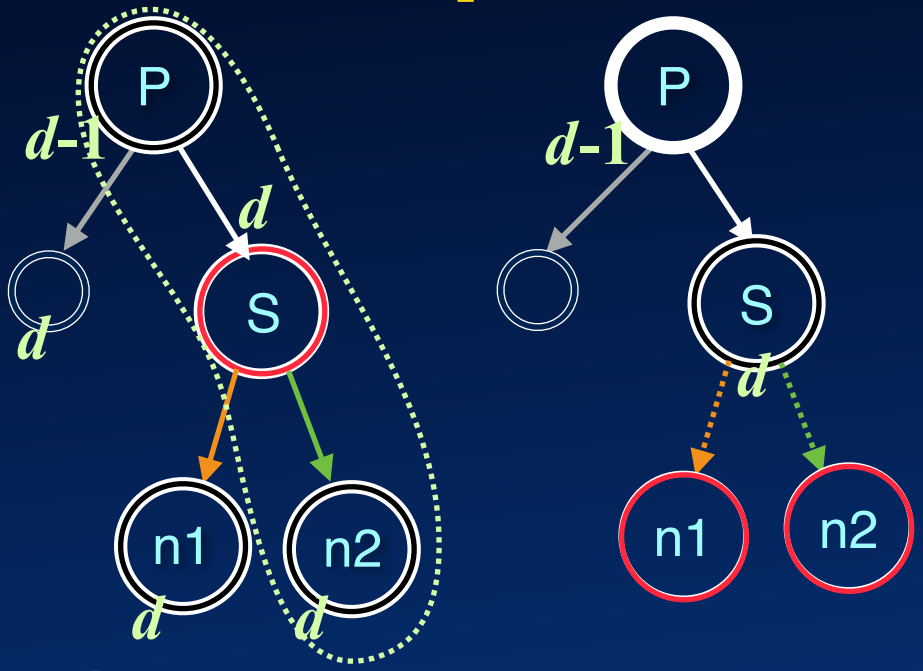
Complex Delete Cases

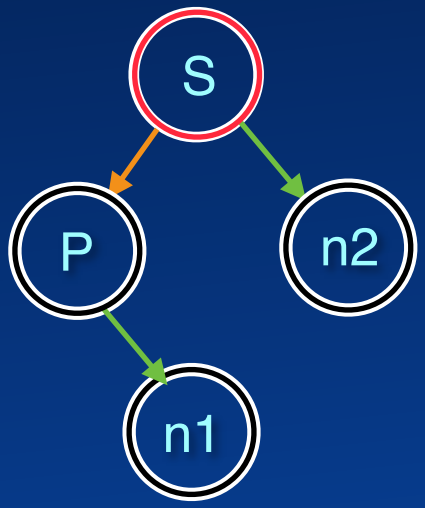


Complex Delete Cases

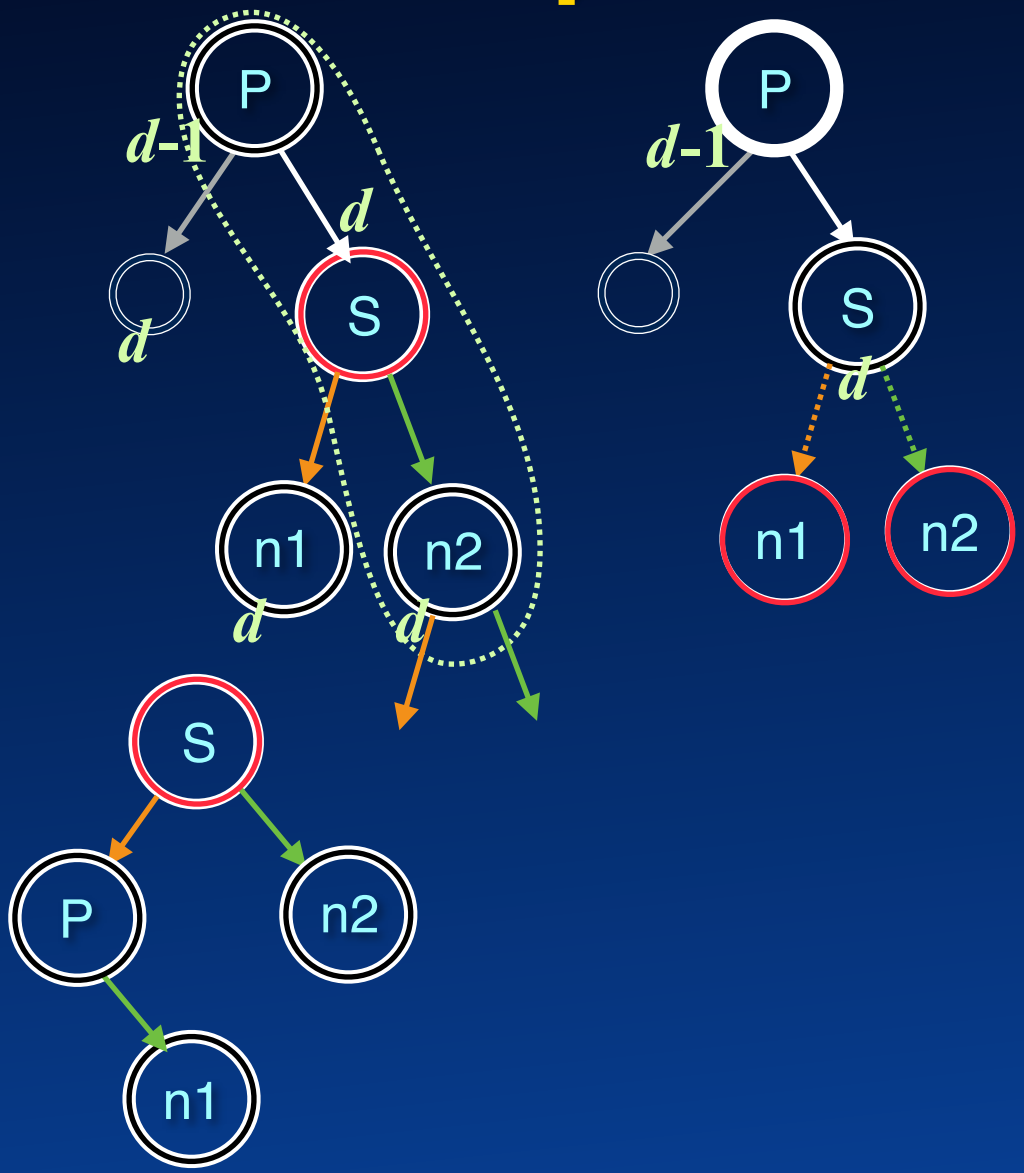


Complex Delete Cases

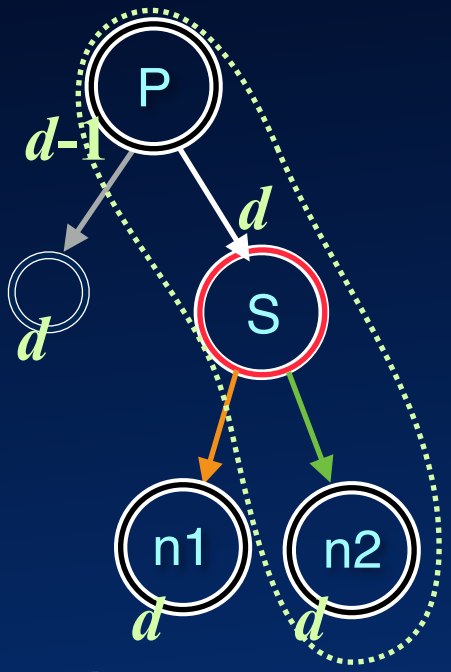


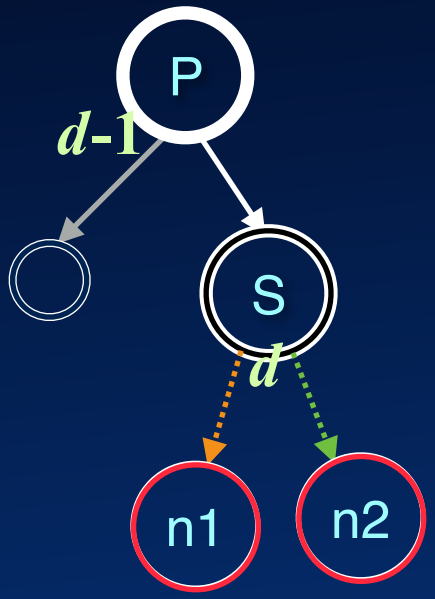


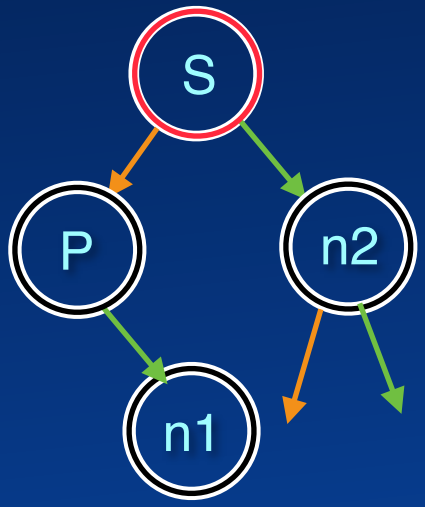
Complex Delete Cases



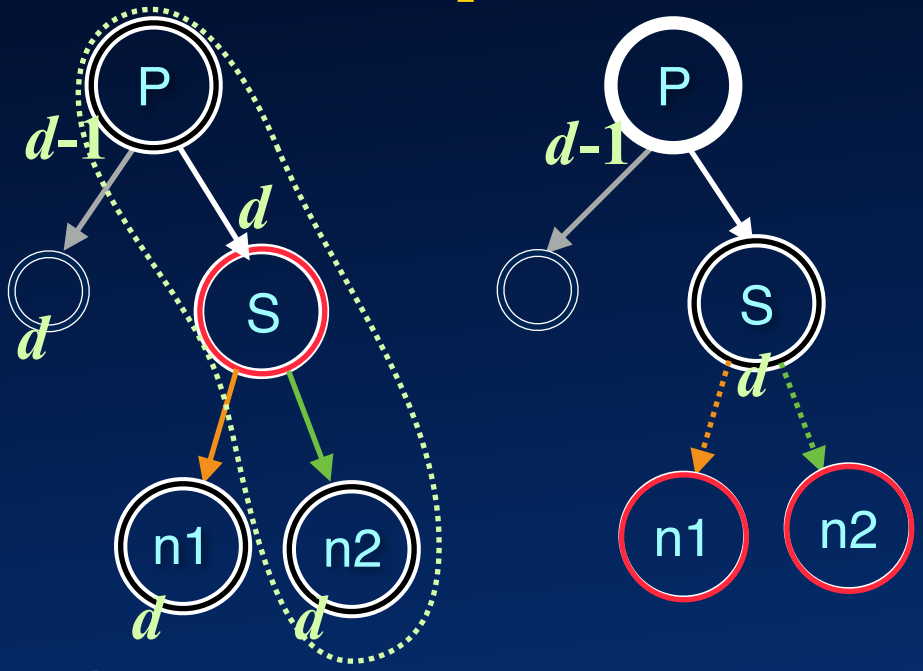
Complex Delete Cases

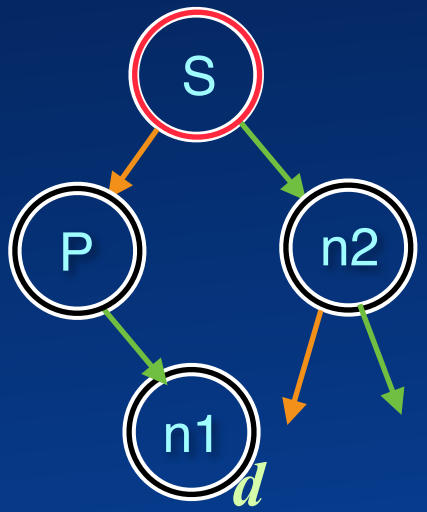




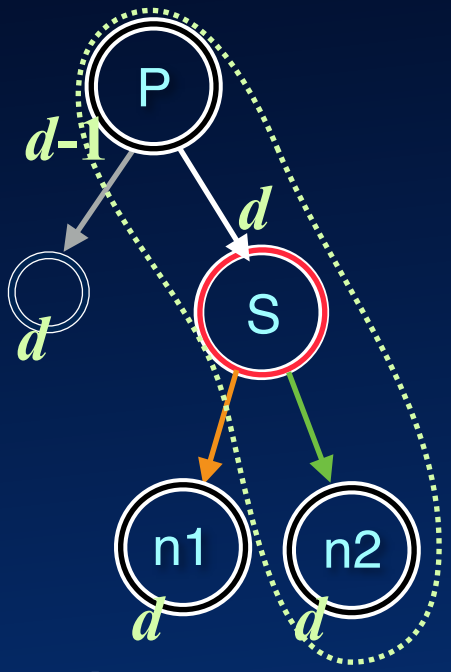


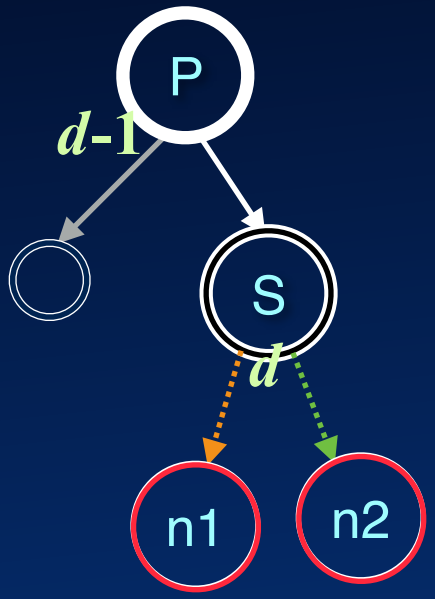
Complex Delete Cases

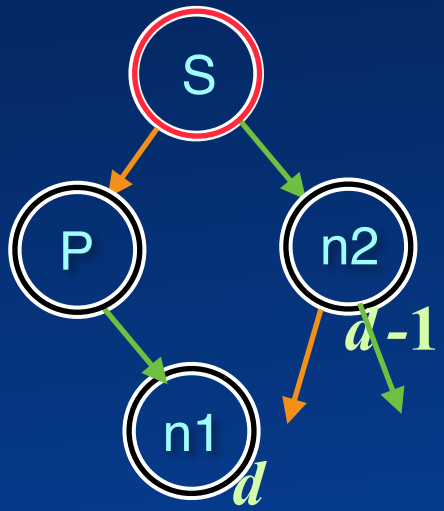




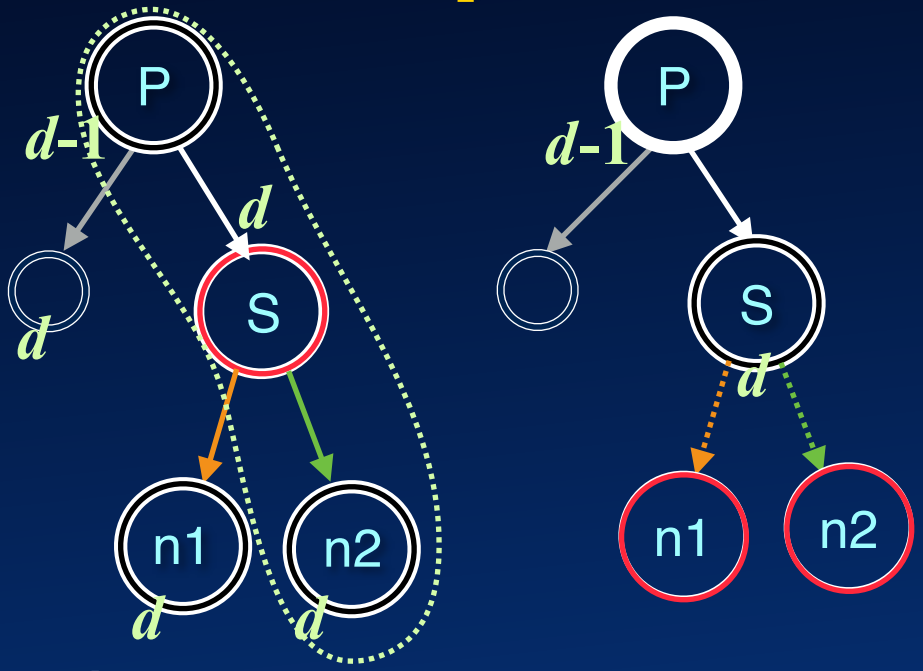
Complex Delete Cases

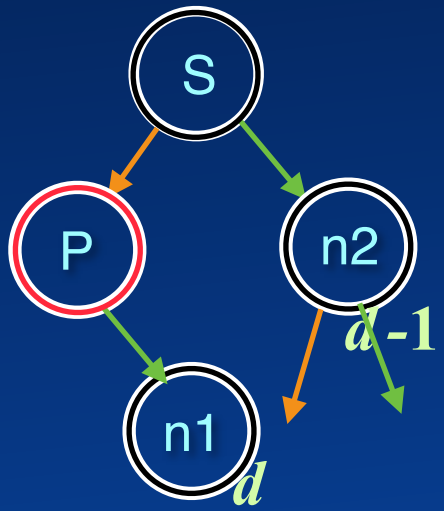




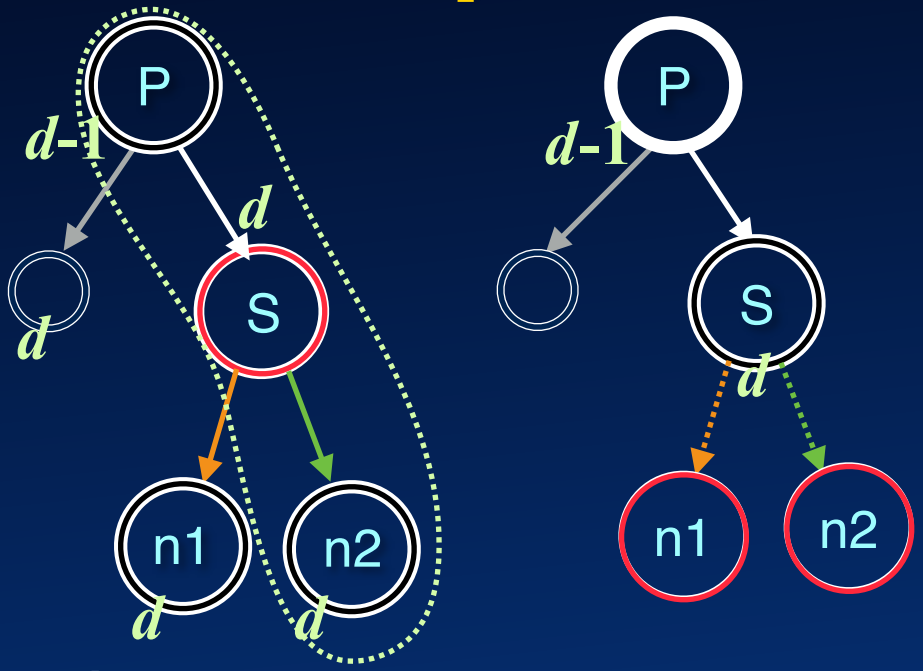


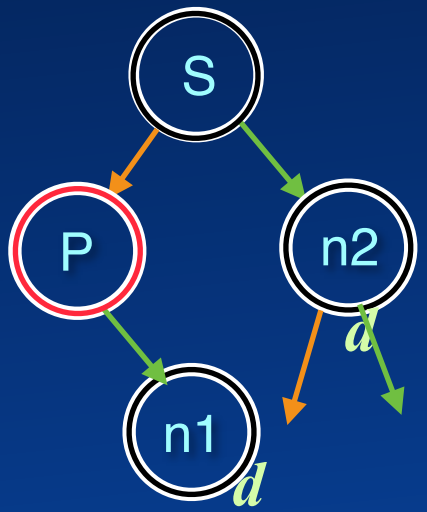
Complex Delete Cases



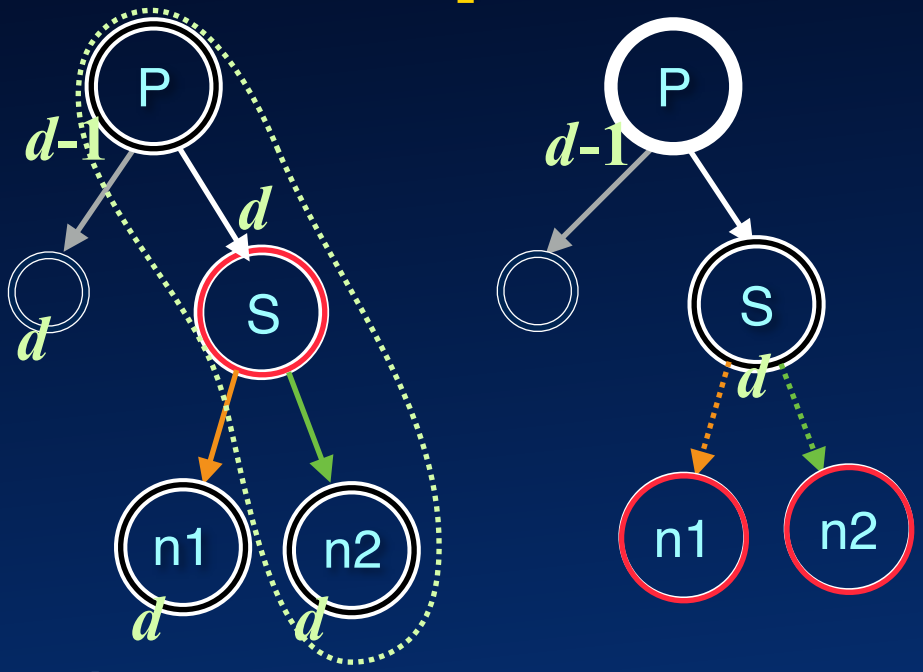


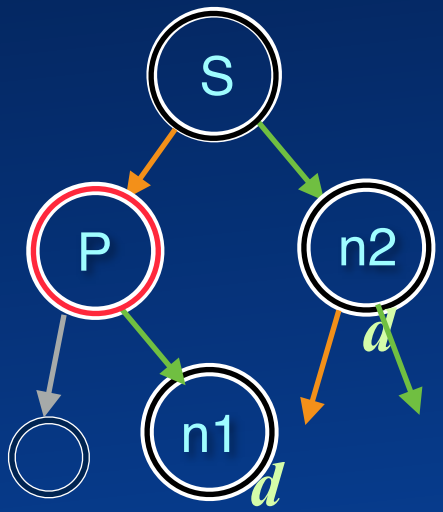
Complex Delete Cases



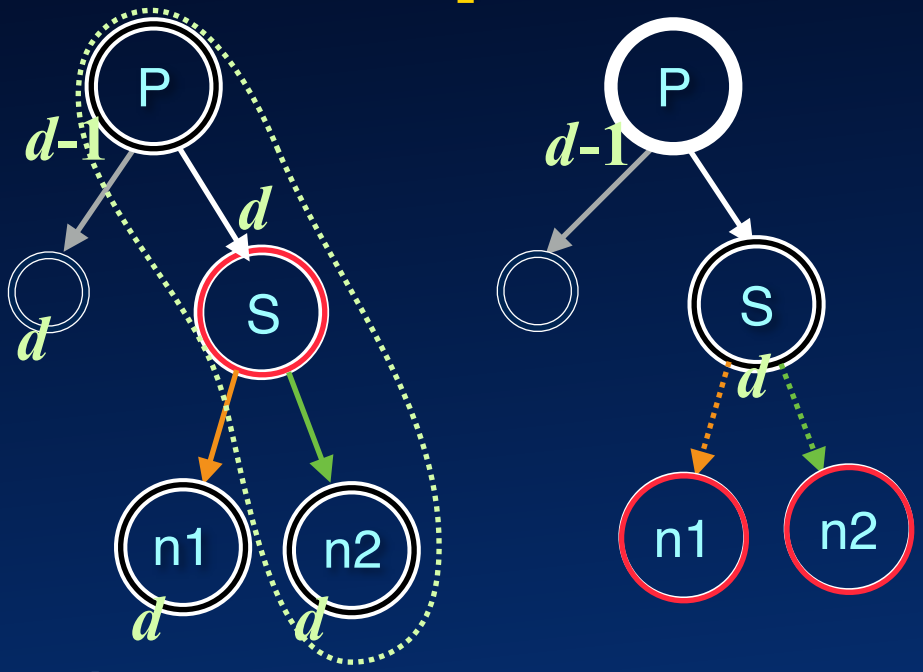


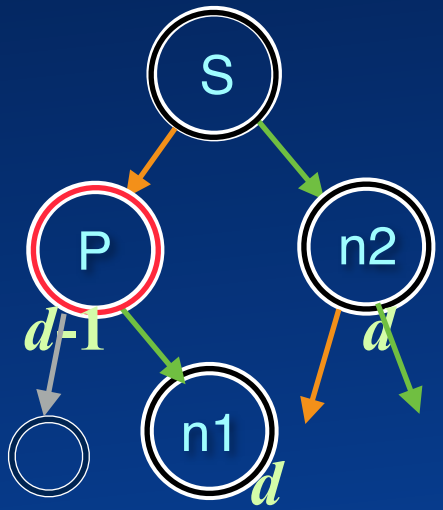
Complex Delete Cases



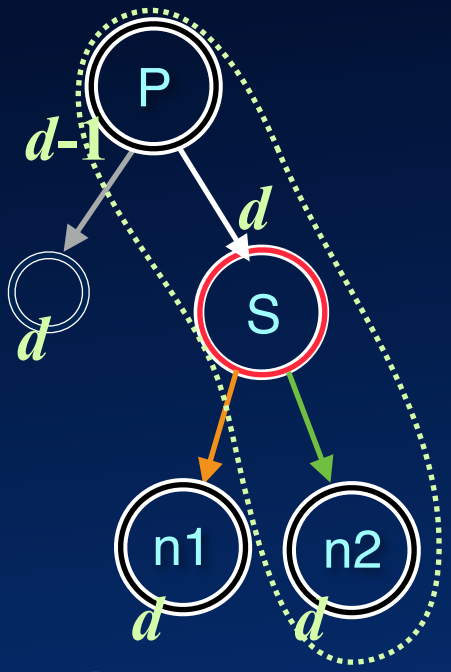


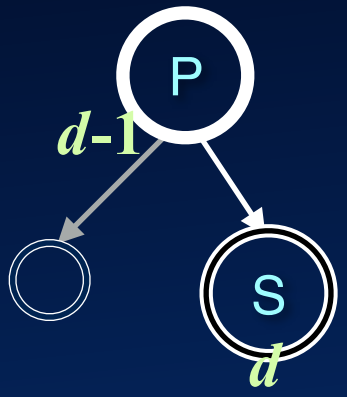
Complex Delete Cases

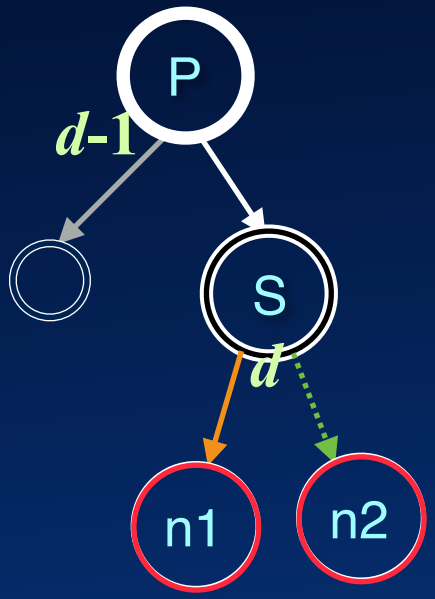


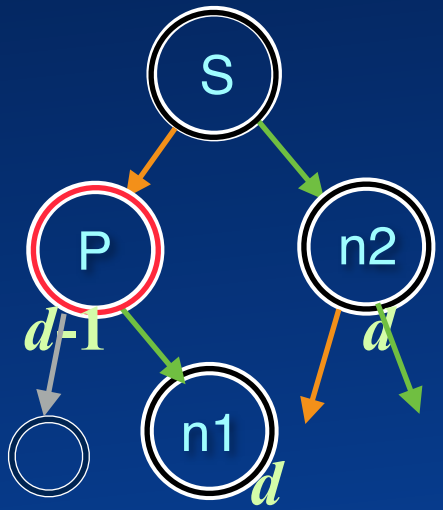


Complex Delete Cases

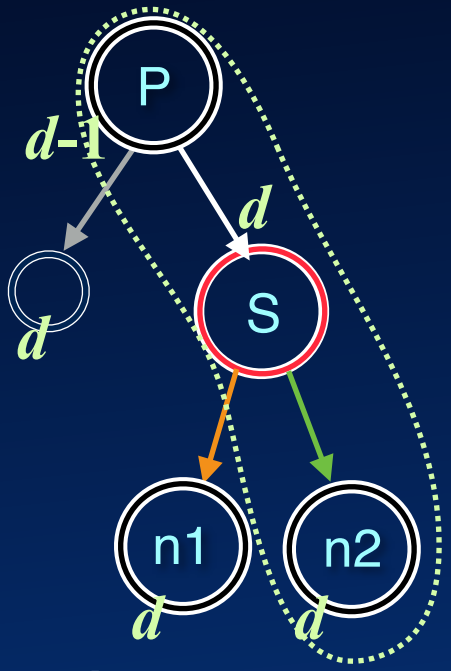


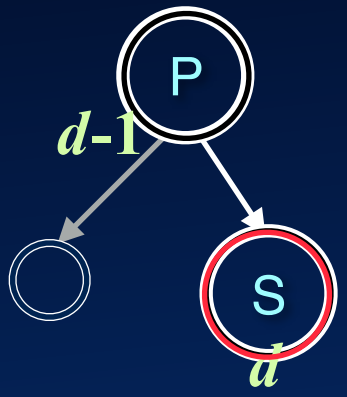


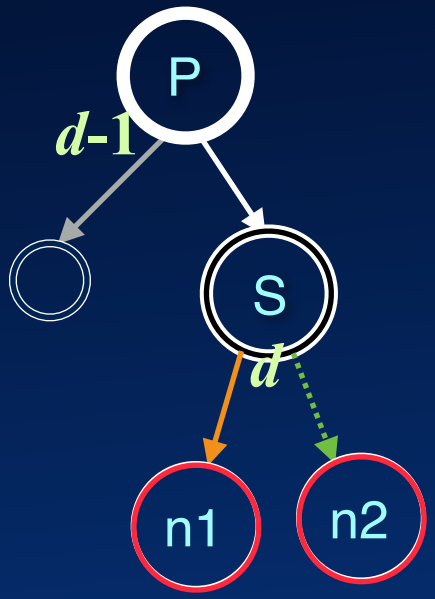


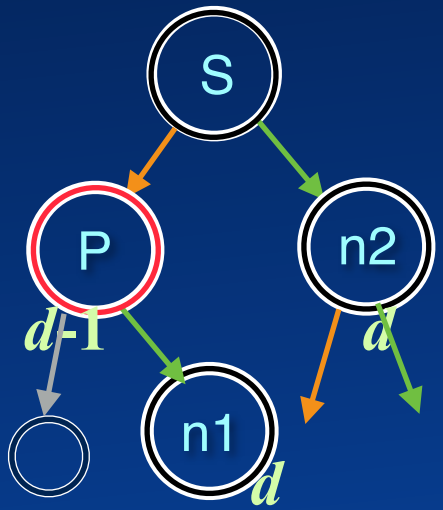


Complex Delete Cases

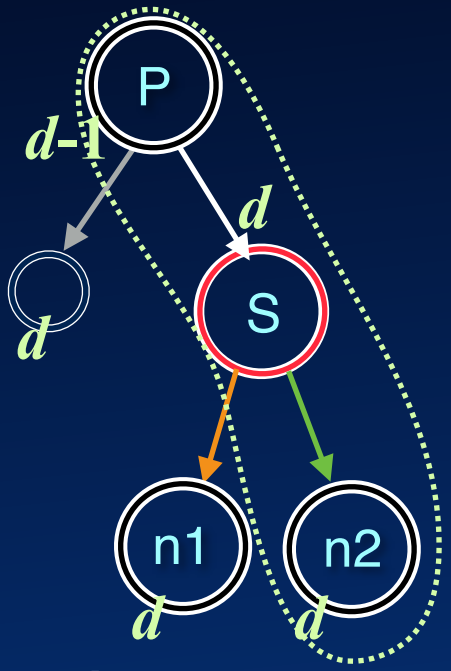


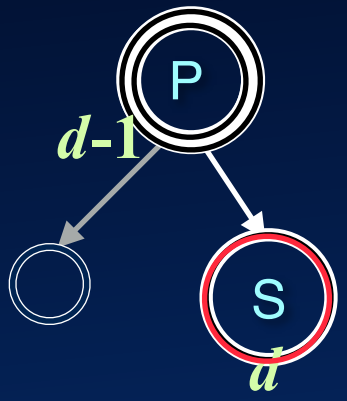


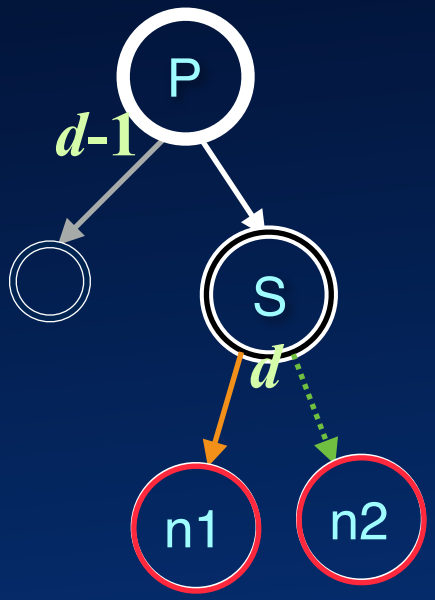


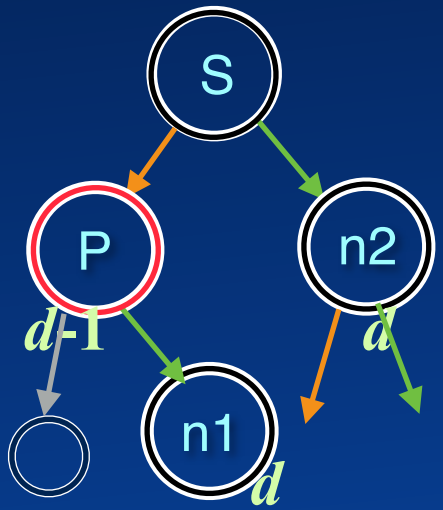


Complex Delete Cases

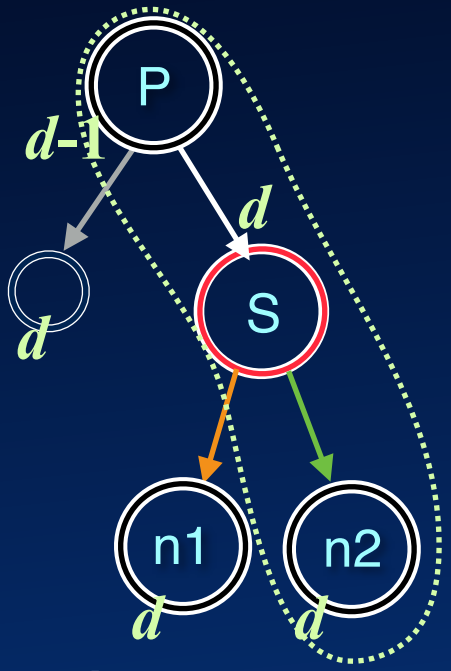


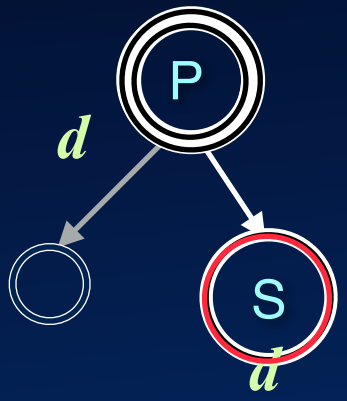


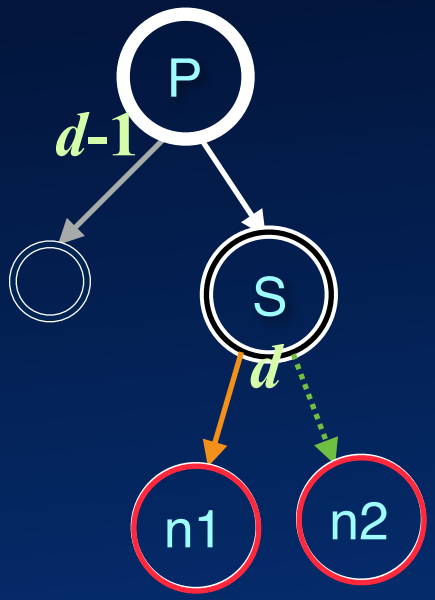


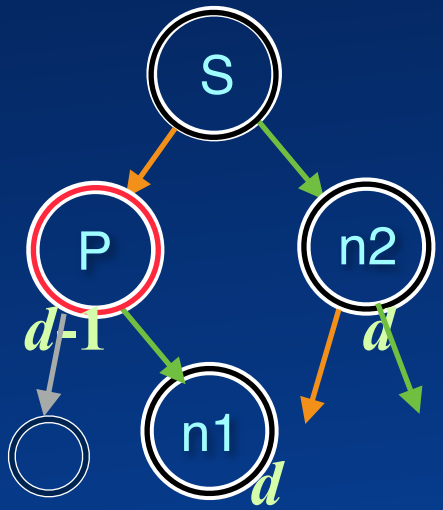


Complex Delete Cases

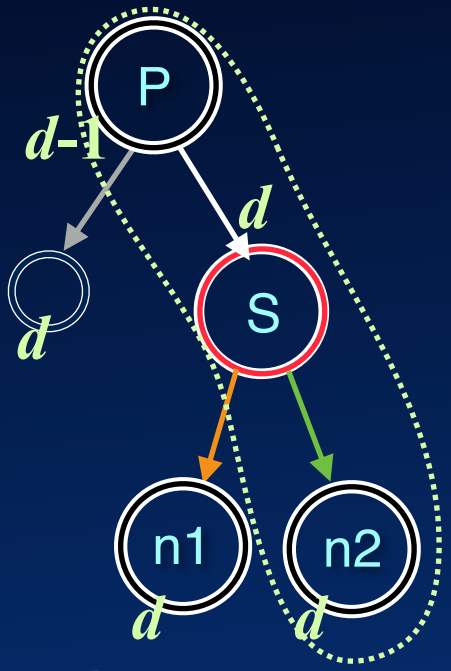


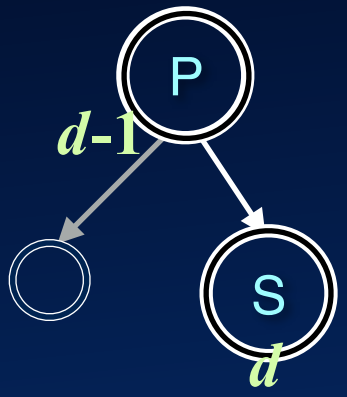


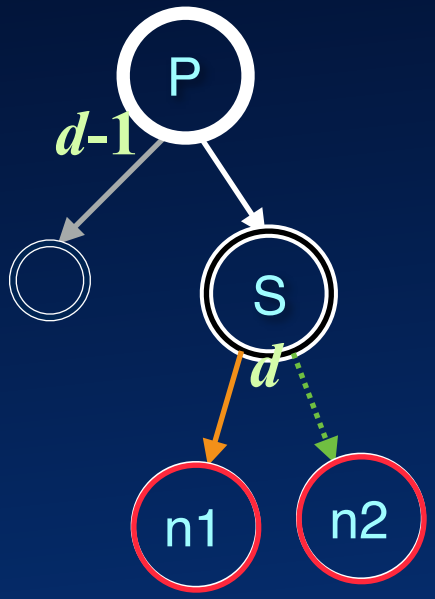


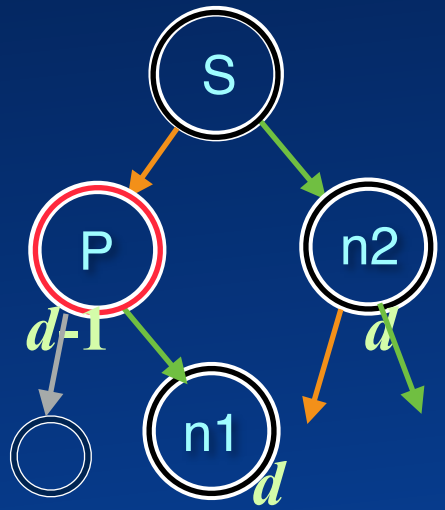


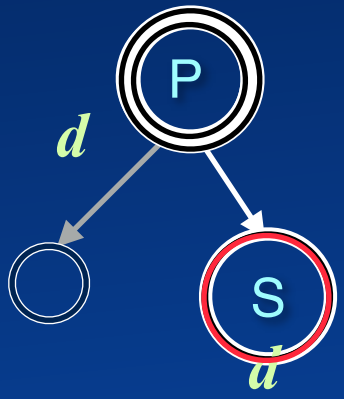
Complex Delete Cases



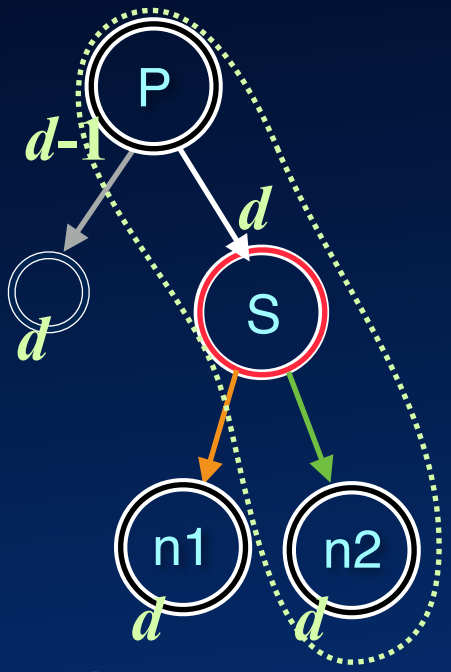


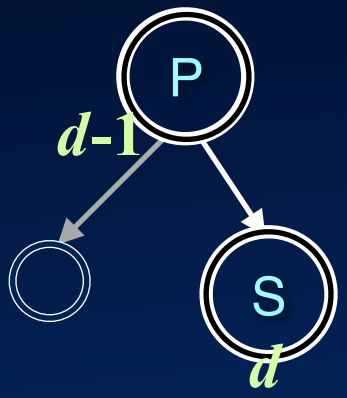


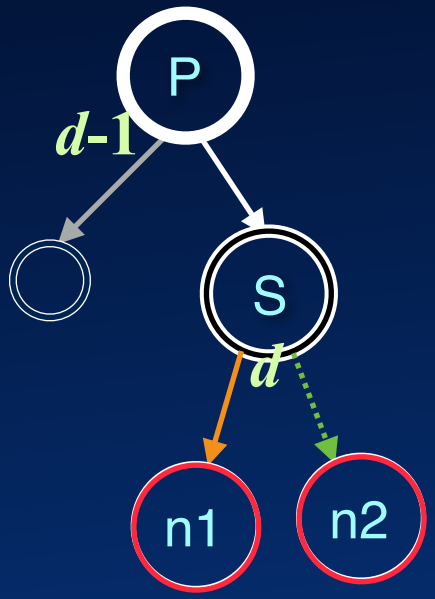


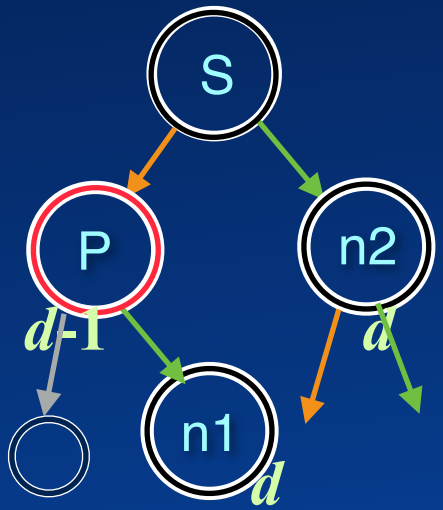


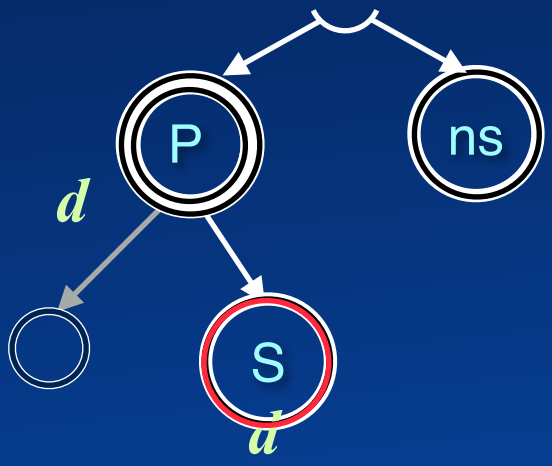
Complex Delete Cases



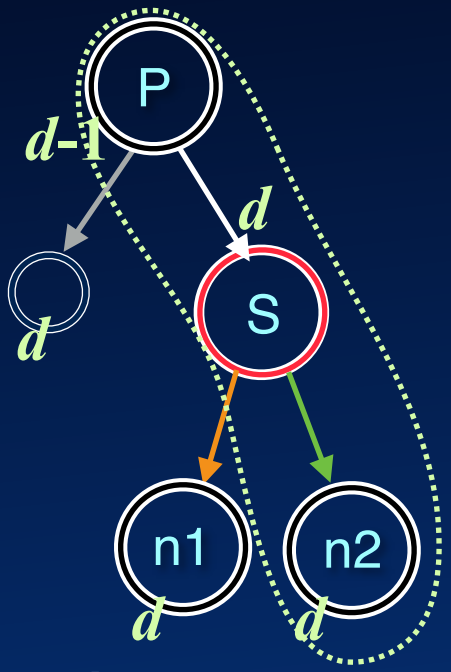


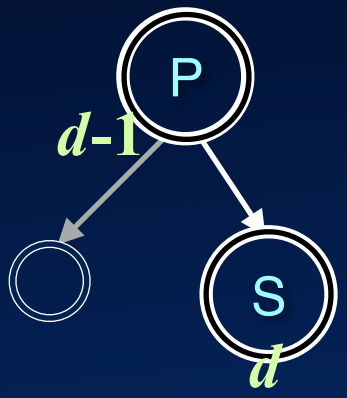


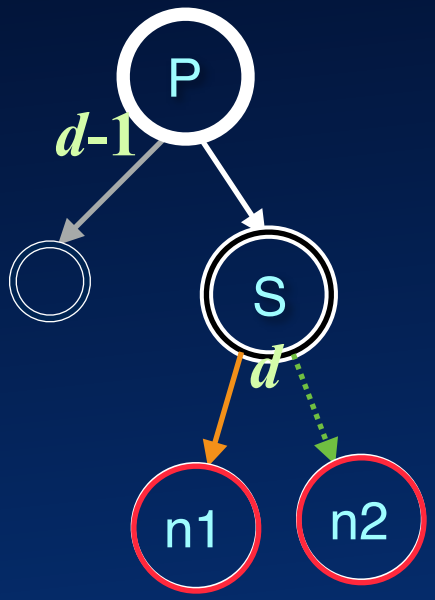


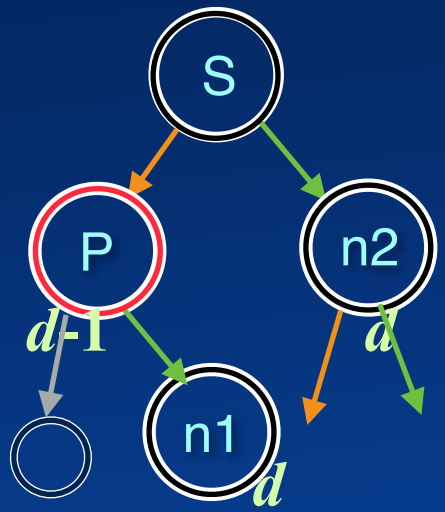


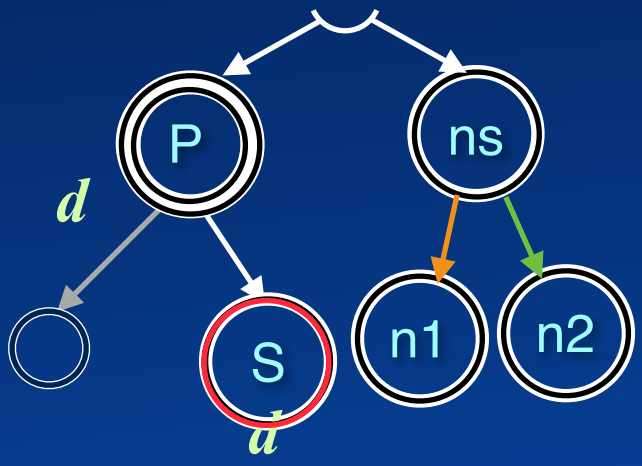
Complex Delete Cases



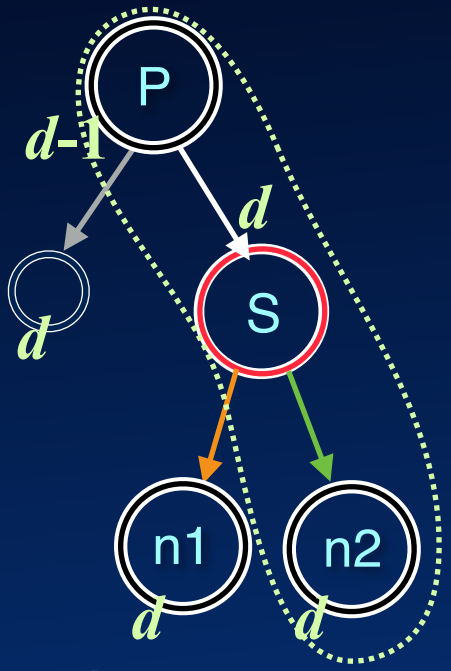


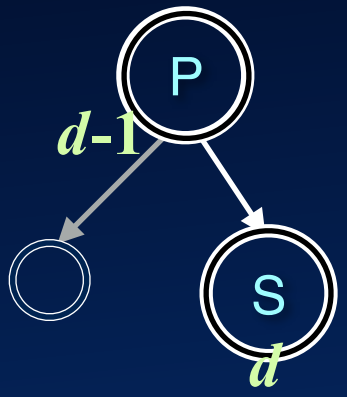


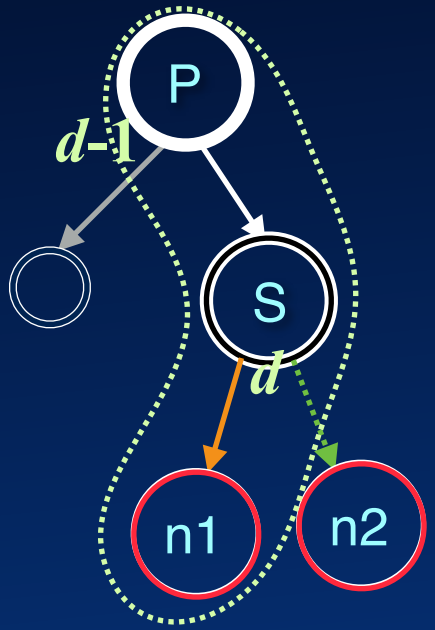


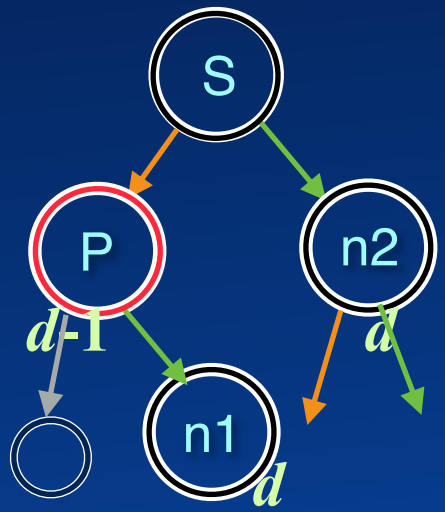


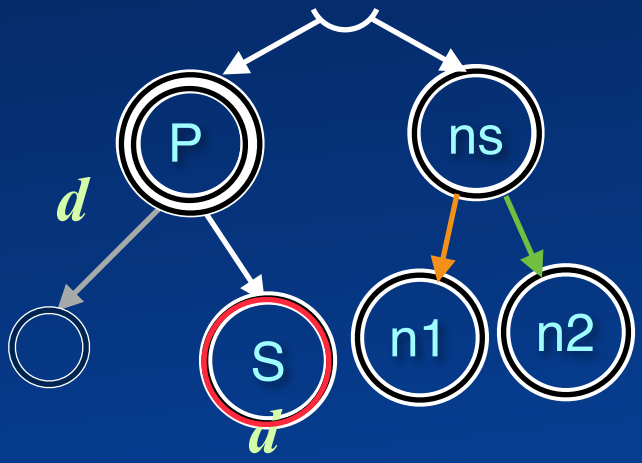
Complex Delete Cases



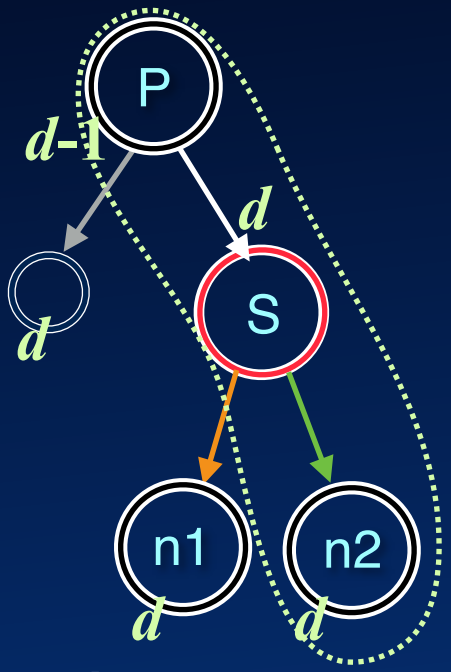


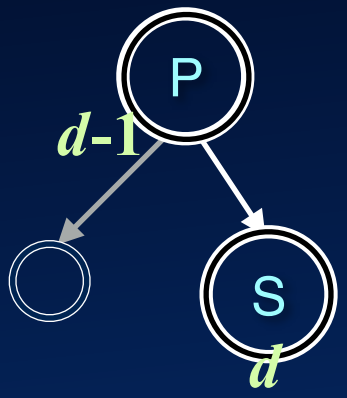


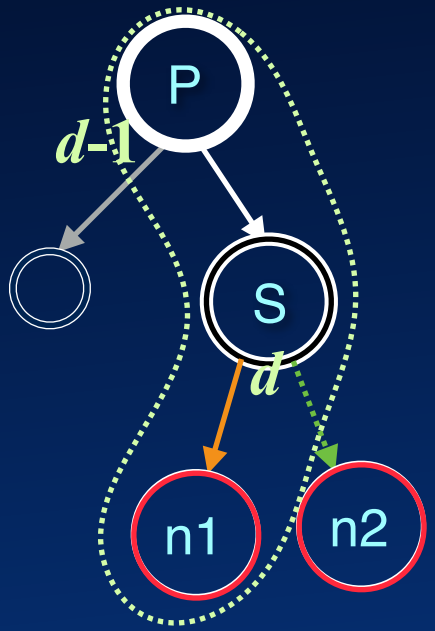


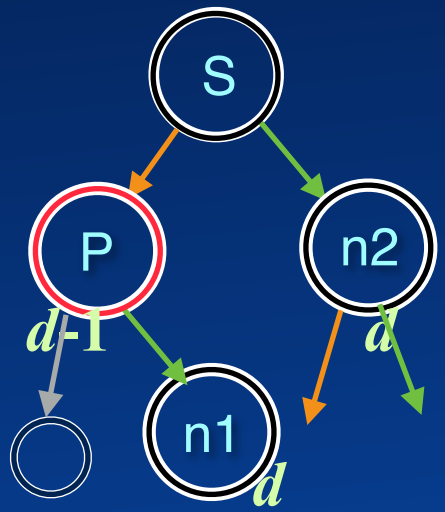


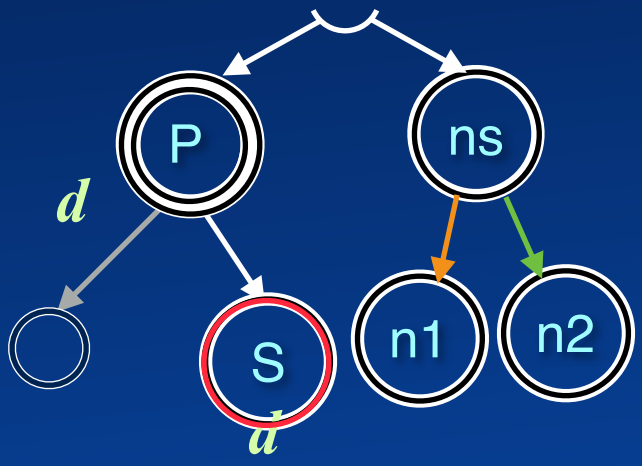
Complex Delete Cases

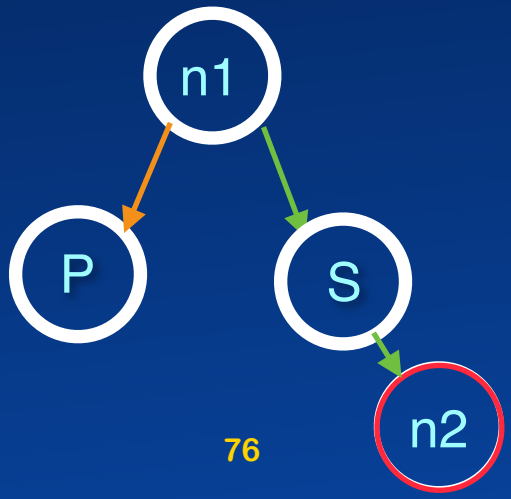




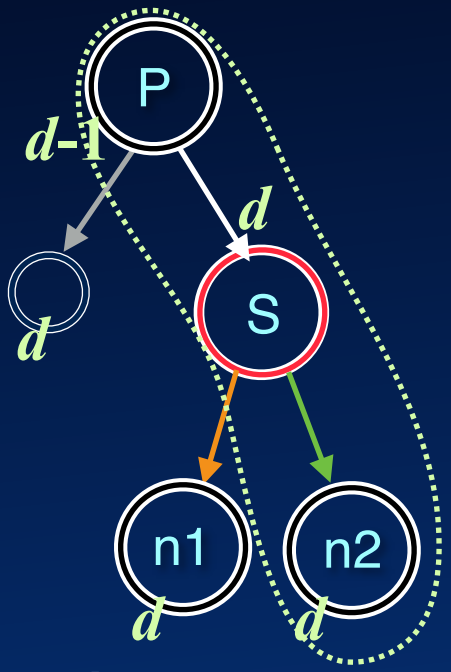


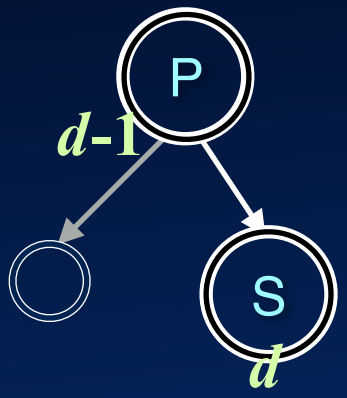


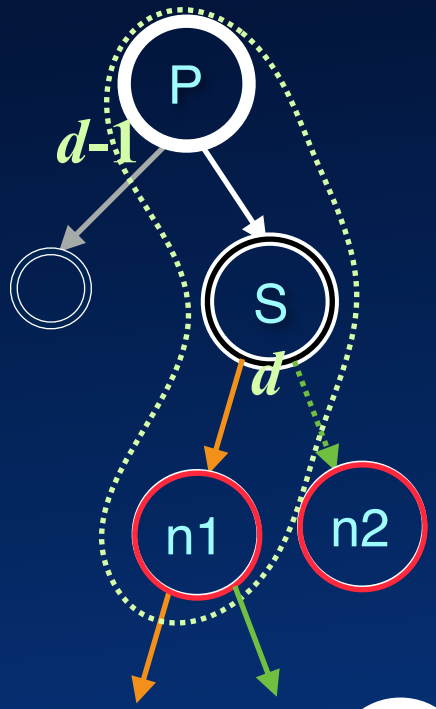


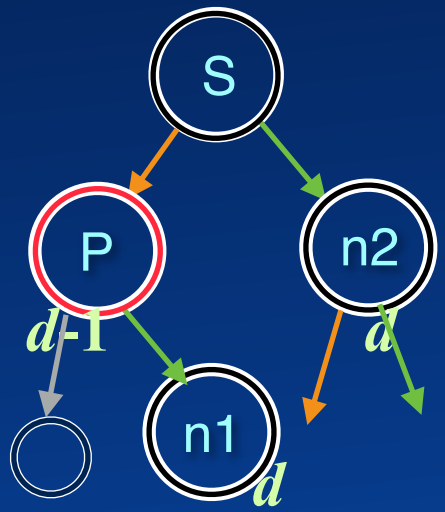


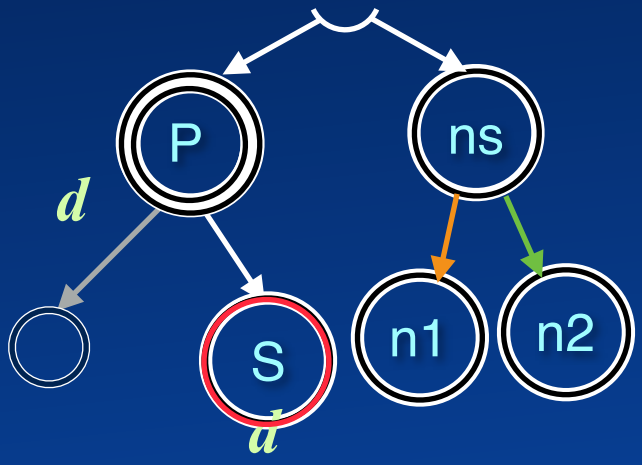
Complex Delete Cases

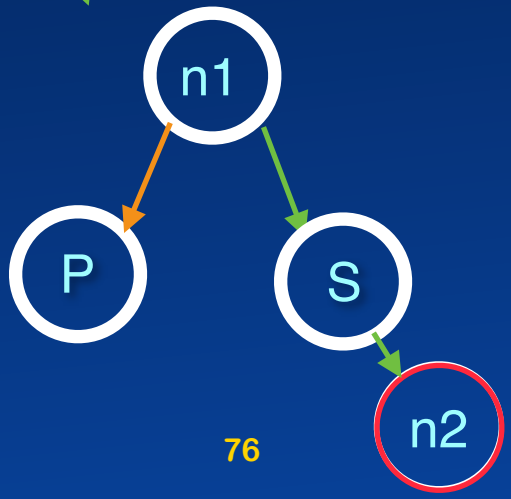




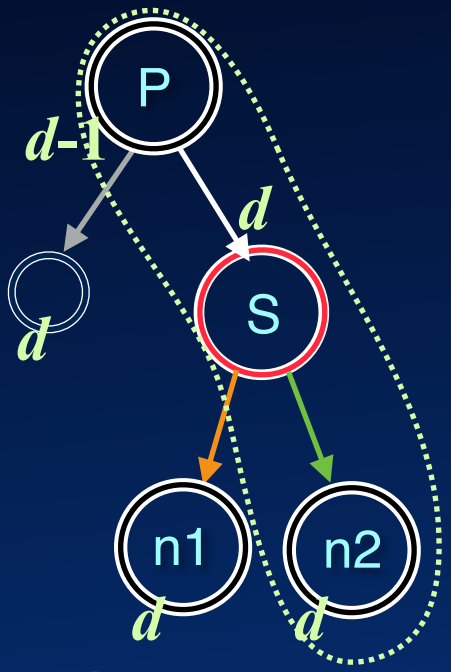


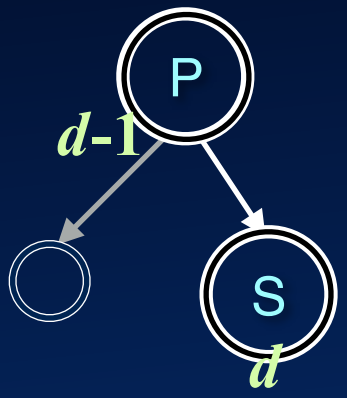


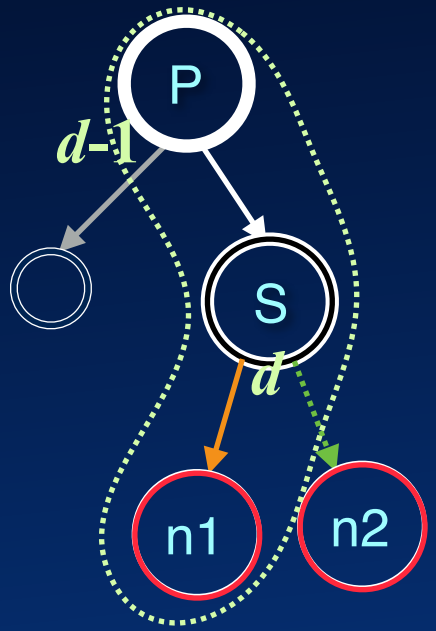


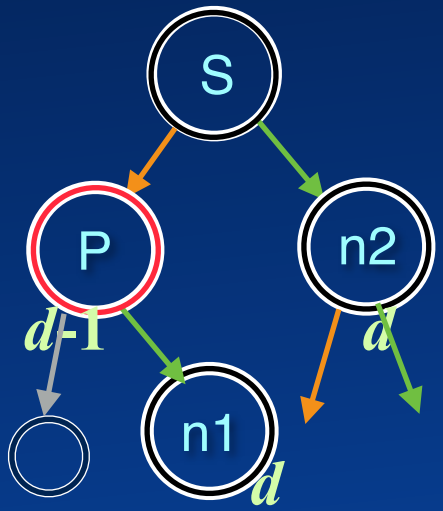


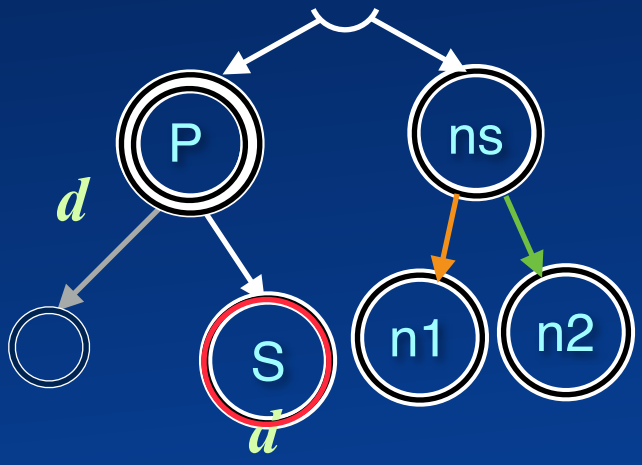
Complex Delete Cases

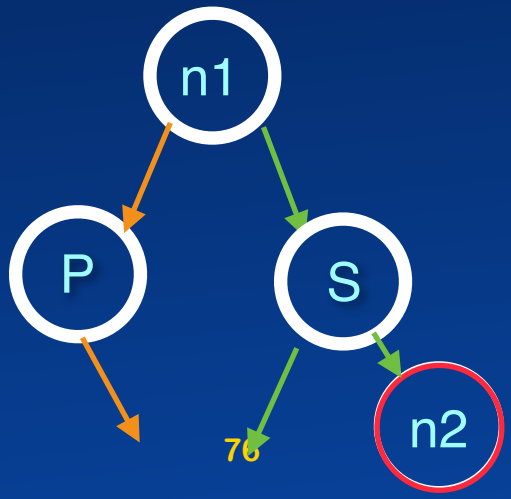




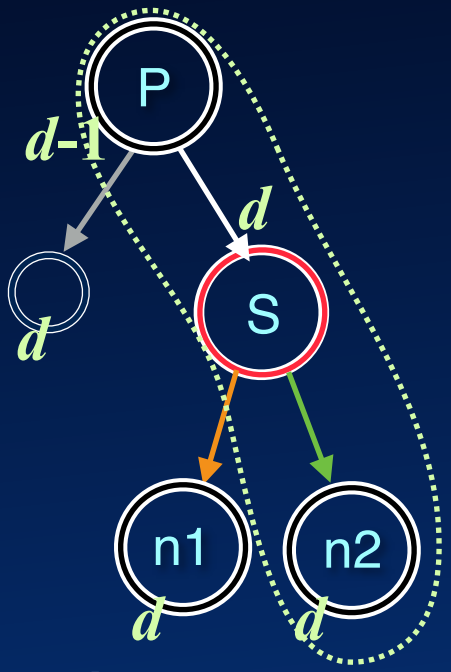


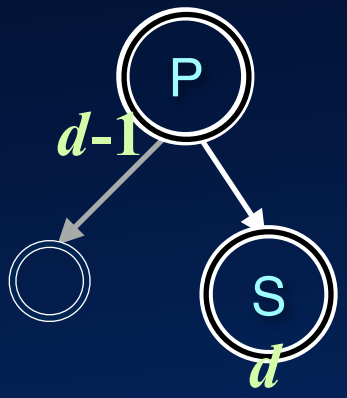


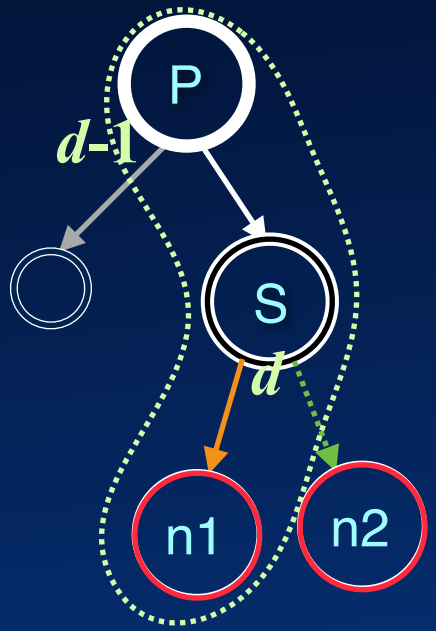


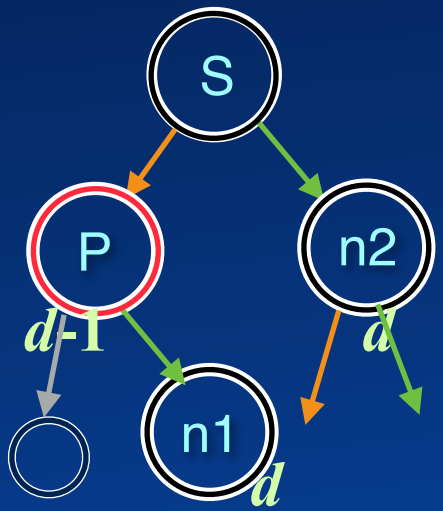


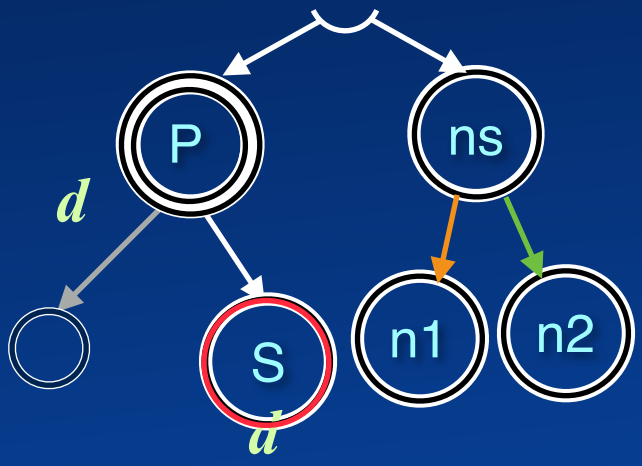
Complex Delete Cases

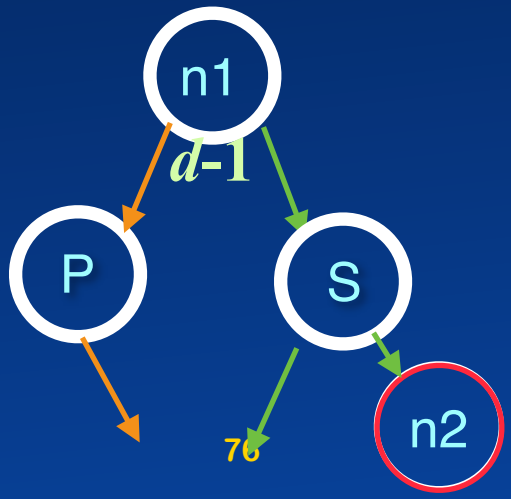




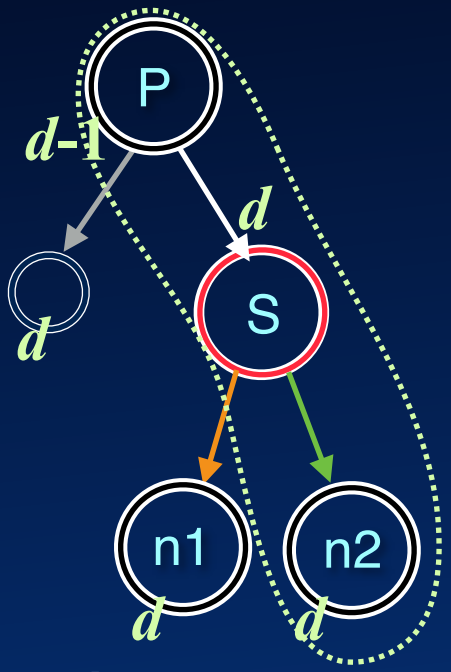


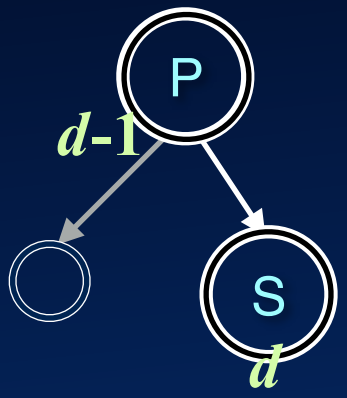


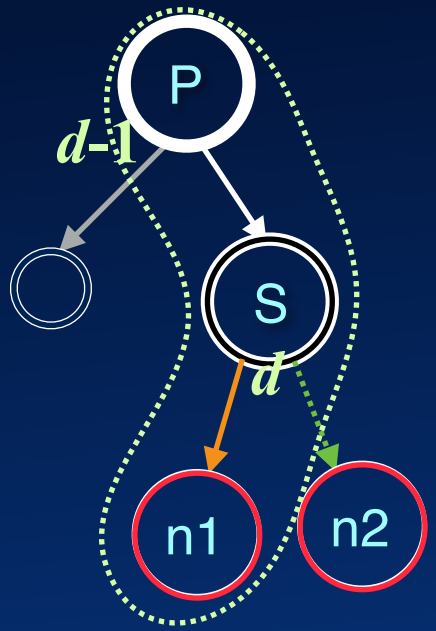


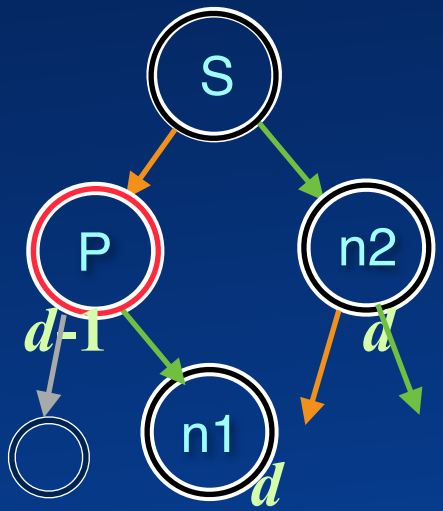


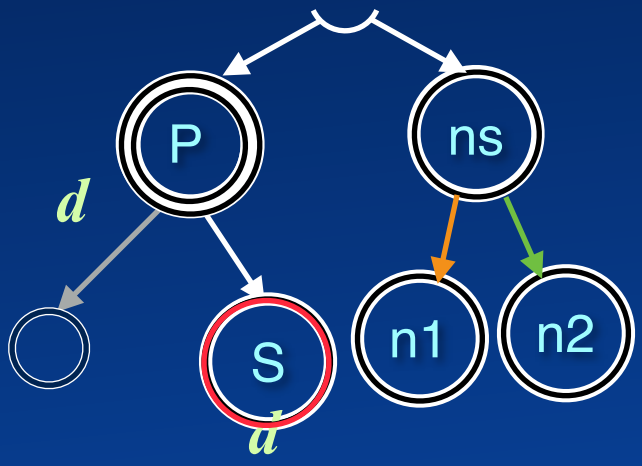
Complex Delete Cases

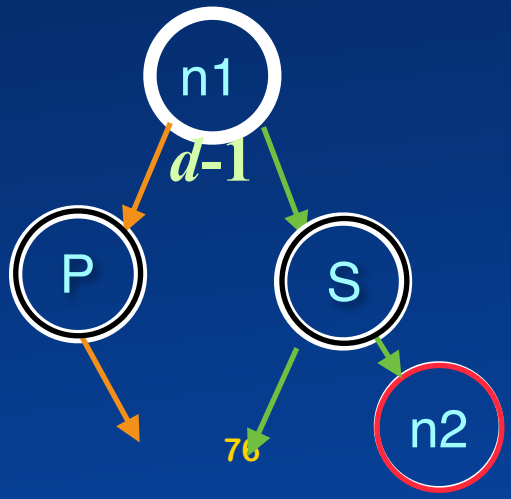




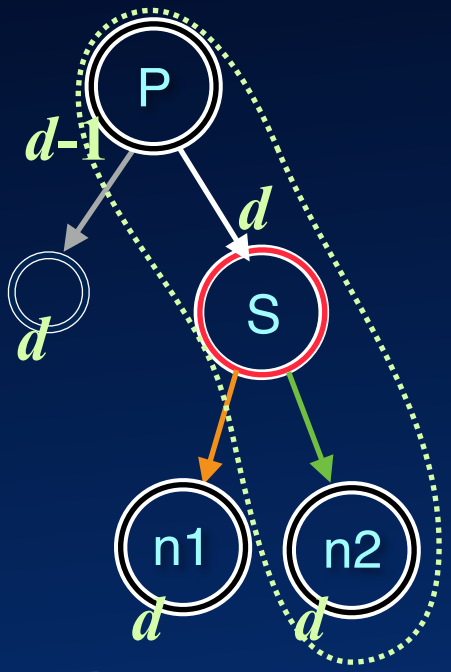


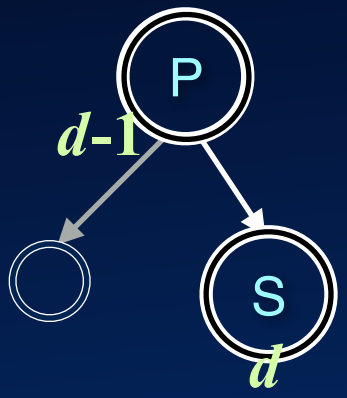


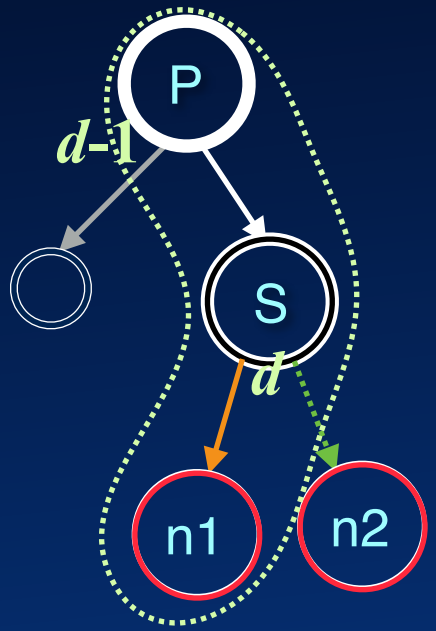


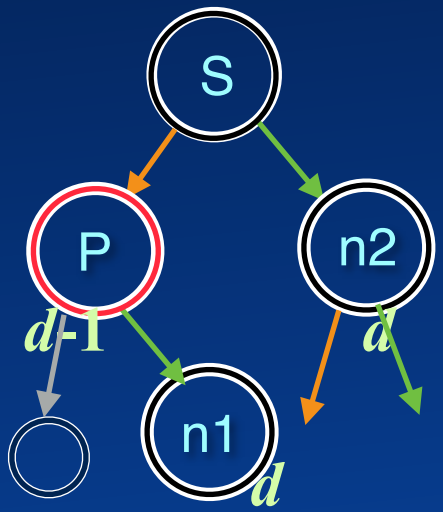


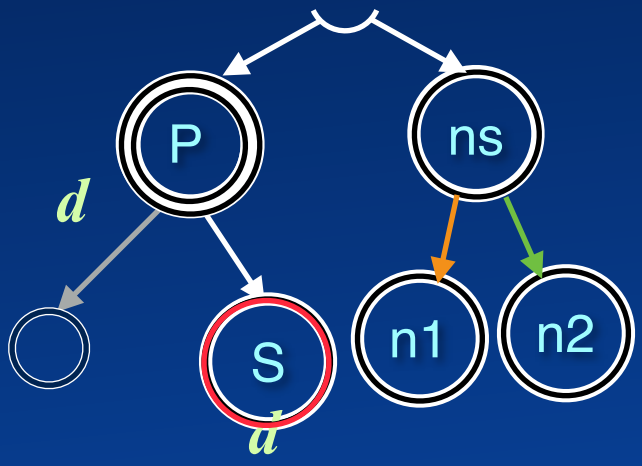
Complex Delete Cases

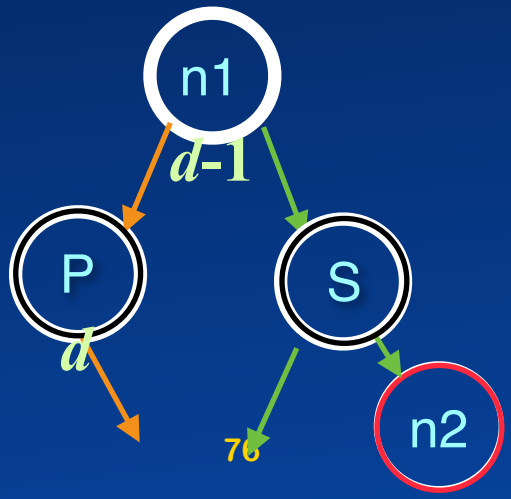




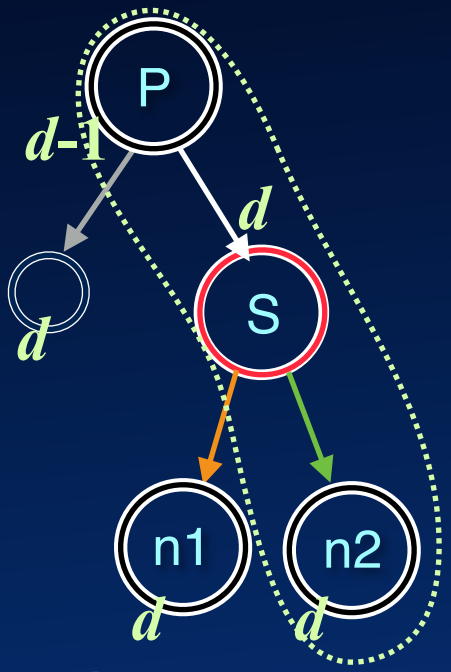


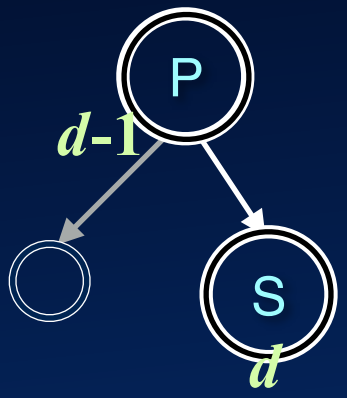


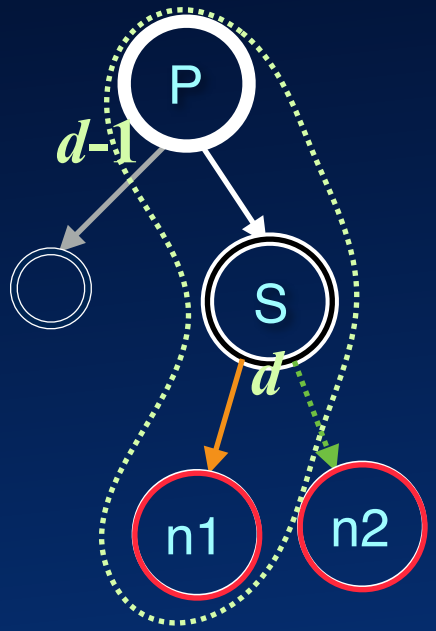


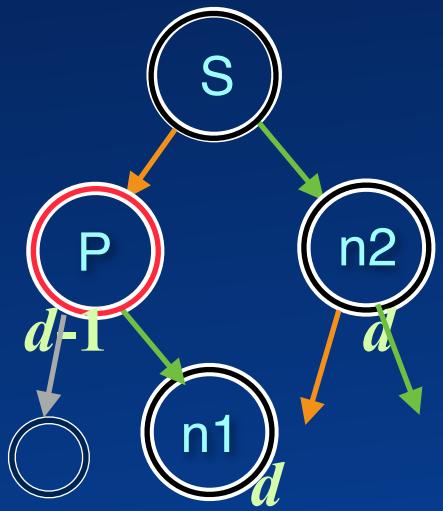


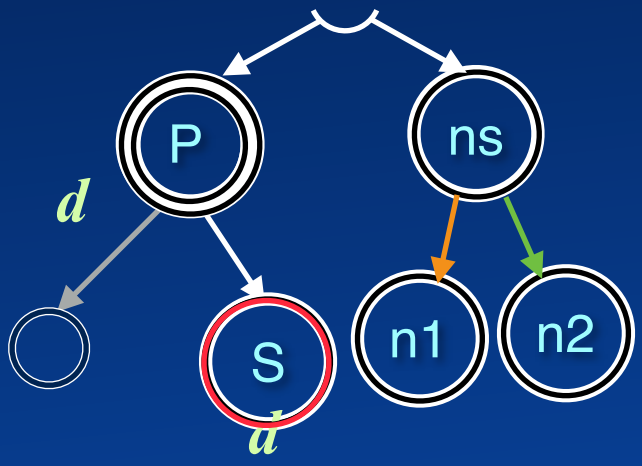
Complex Delete Cases

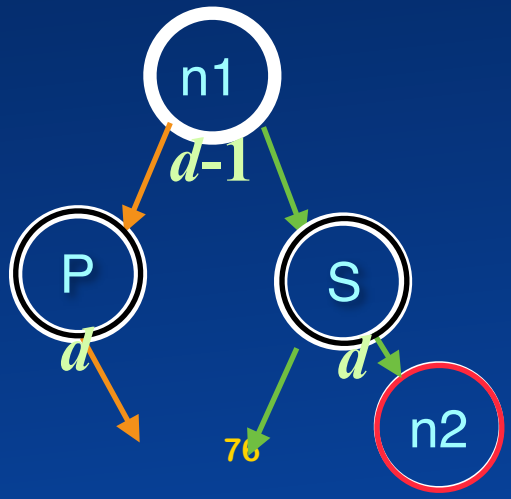




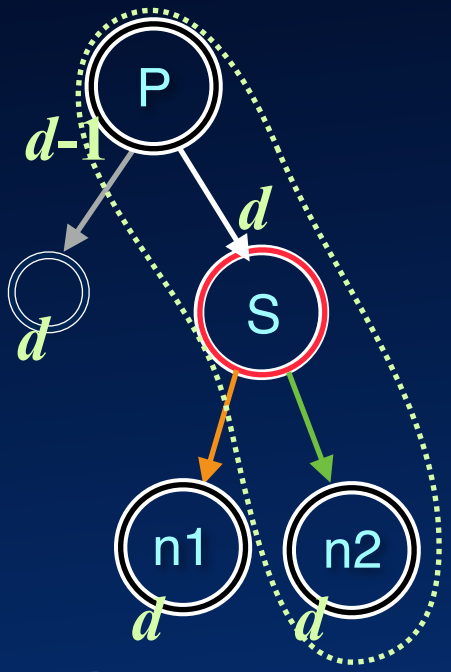


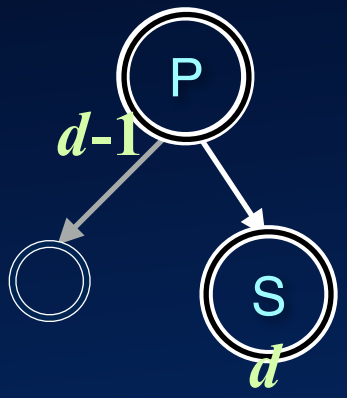


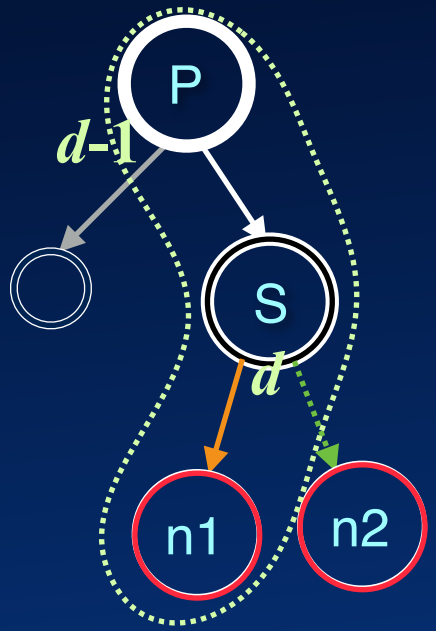


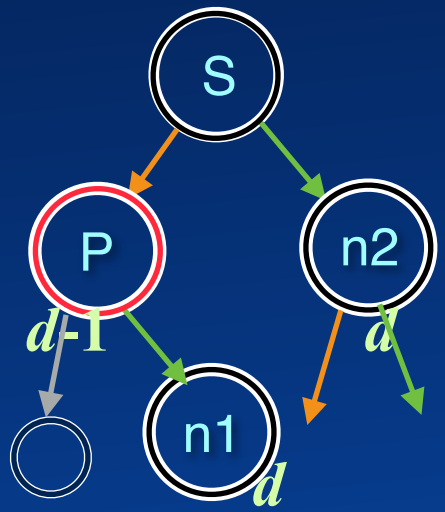


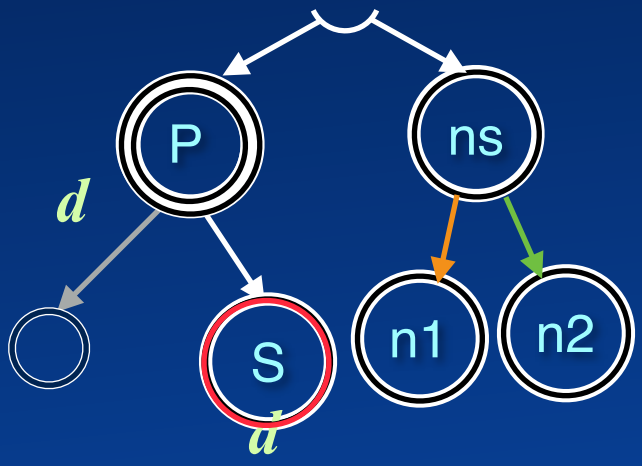
Complex Delete Cases

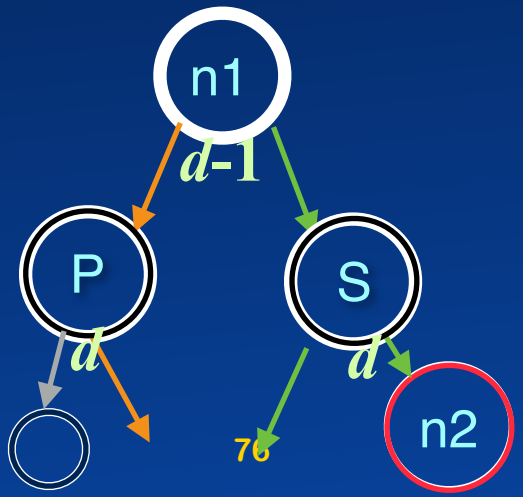








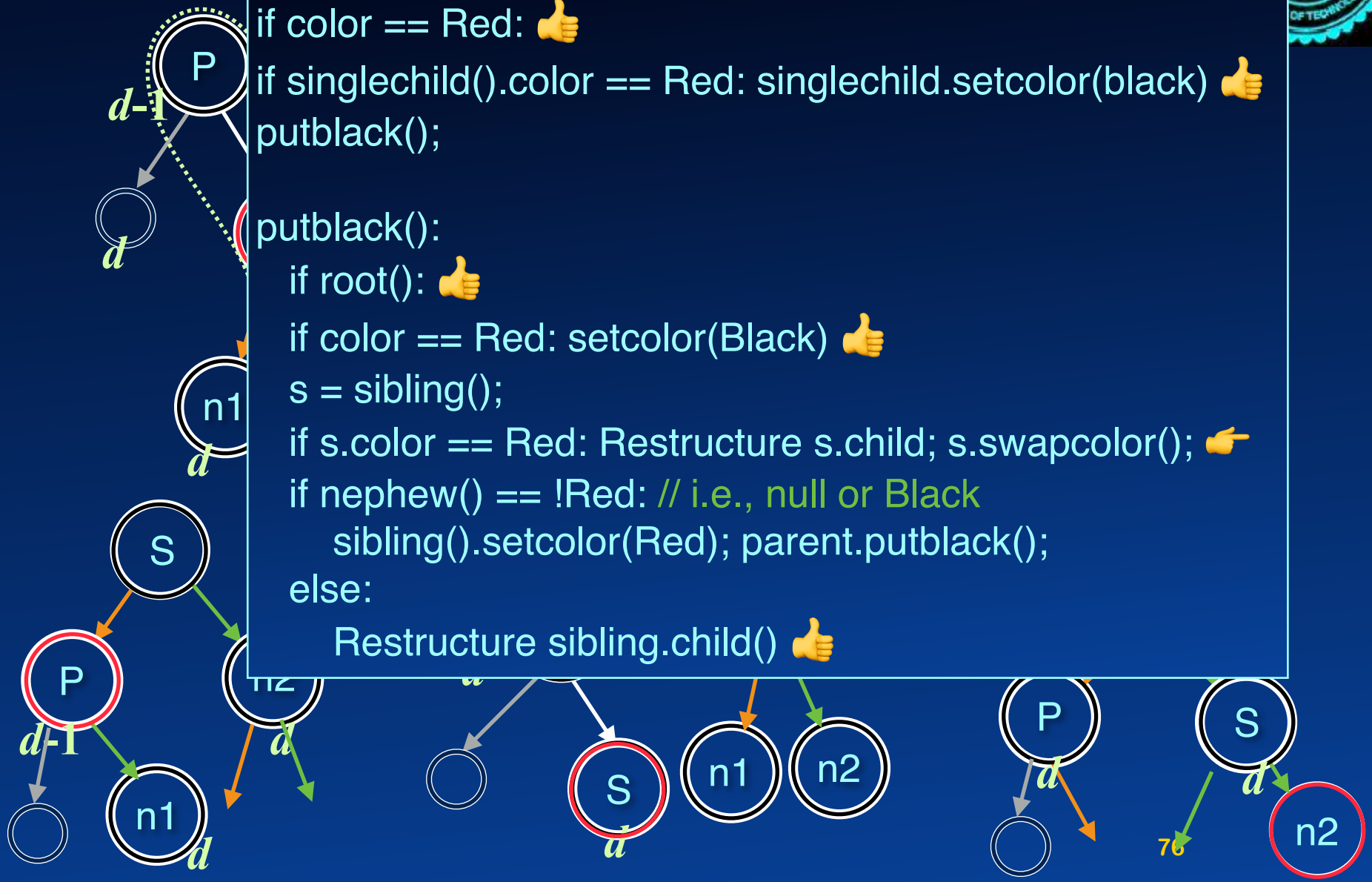




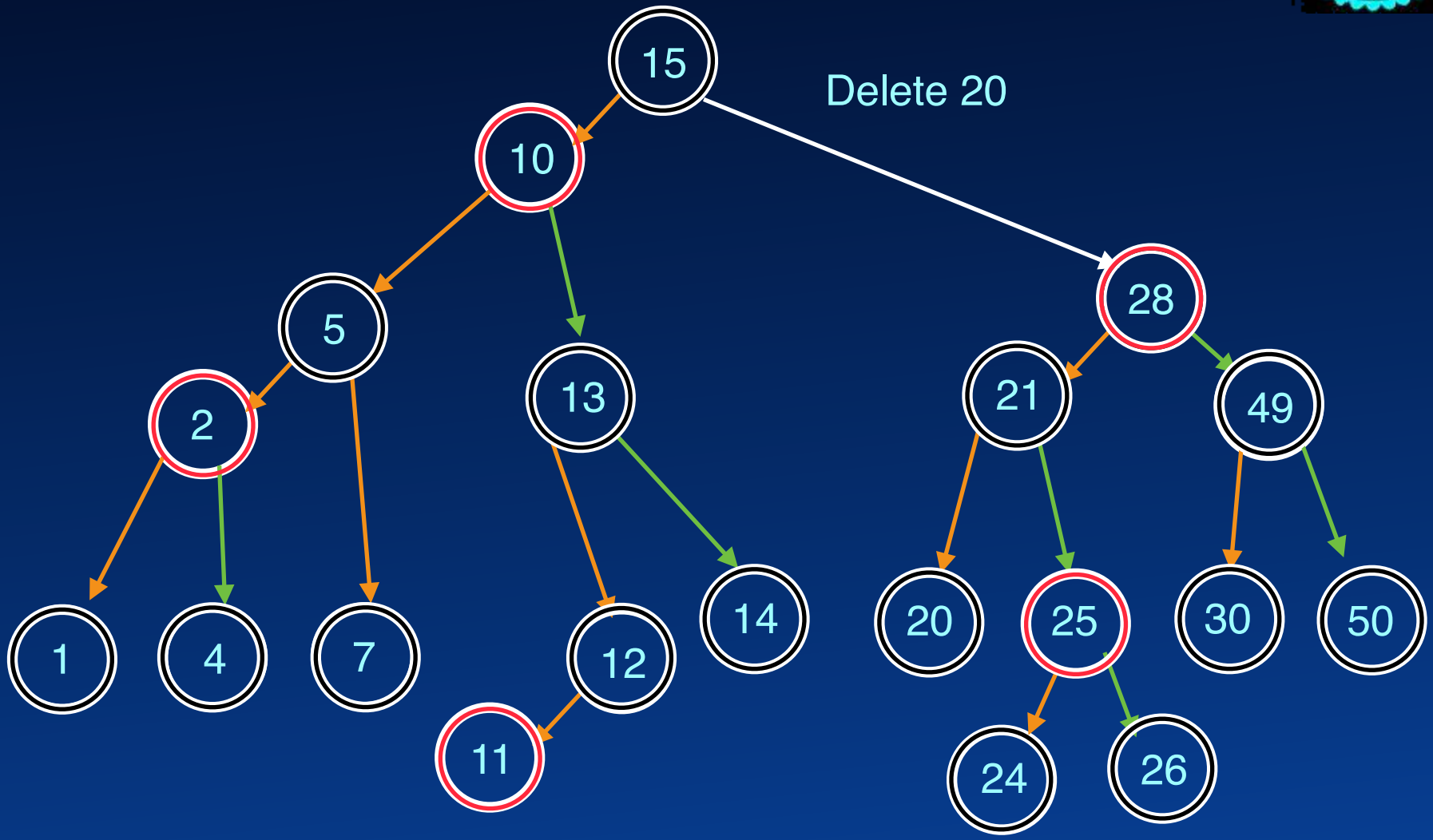
Complex Delete Cases

```
if color == Red: 👍
if singlechild().color == Red: singlechild.setcolor(black) 👍
putblack();

putblack():
   if root(): 👍
   if color == Red: setcolor(Black) 👍
   s = sibling();
   if s.color == Red: Restructure s.child; s.swapcolor(); 👉
   if nephew() == !Red: // i.e., null or Black
       sibling().setcolor(Red); parent.putblack();
   else:
       Restructure sibling.child() 👍
```

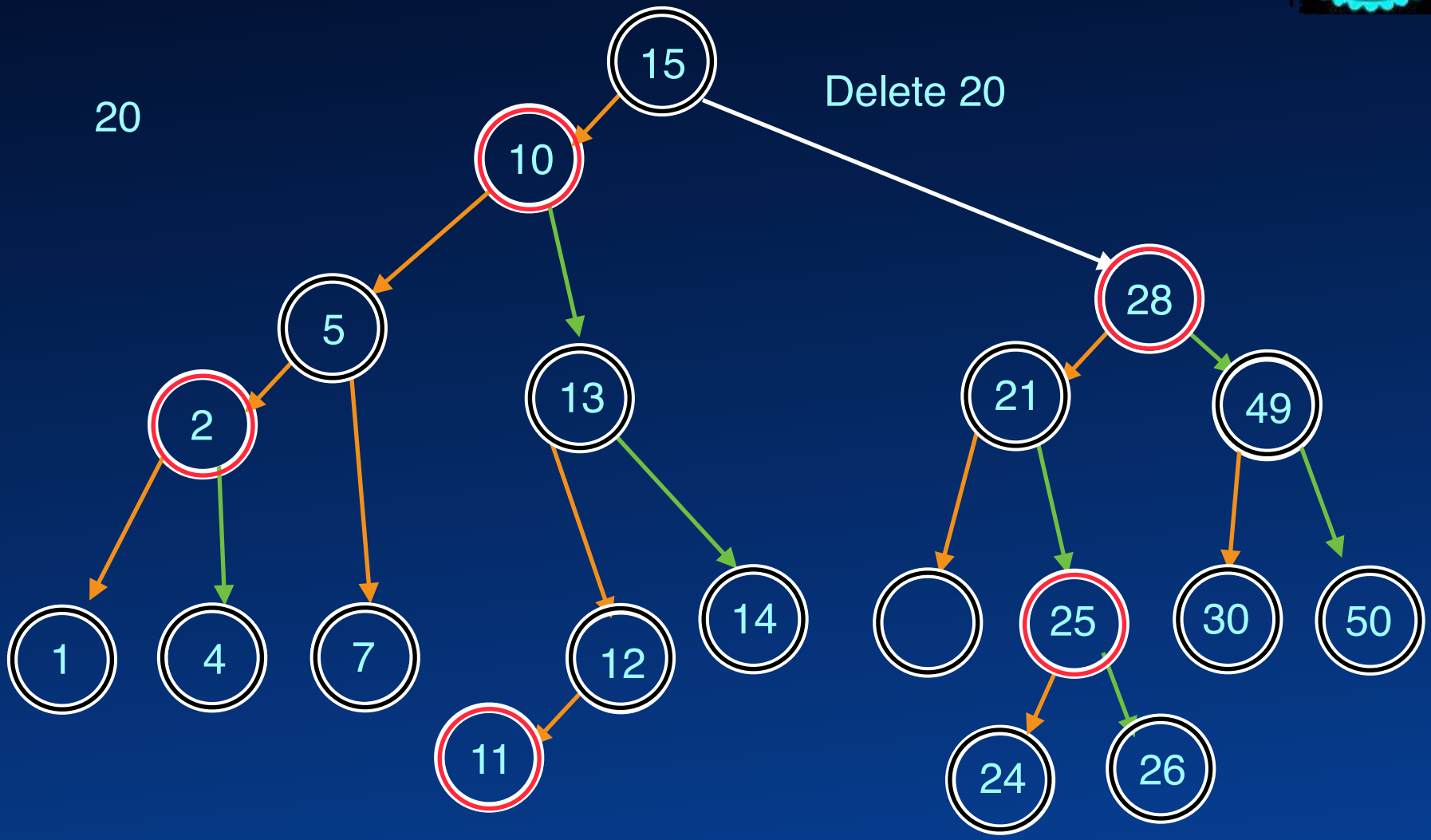


Red-Black Tree



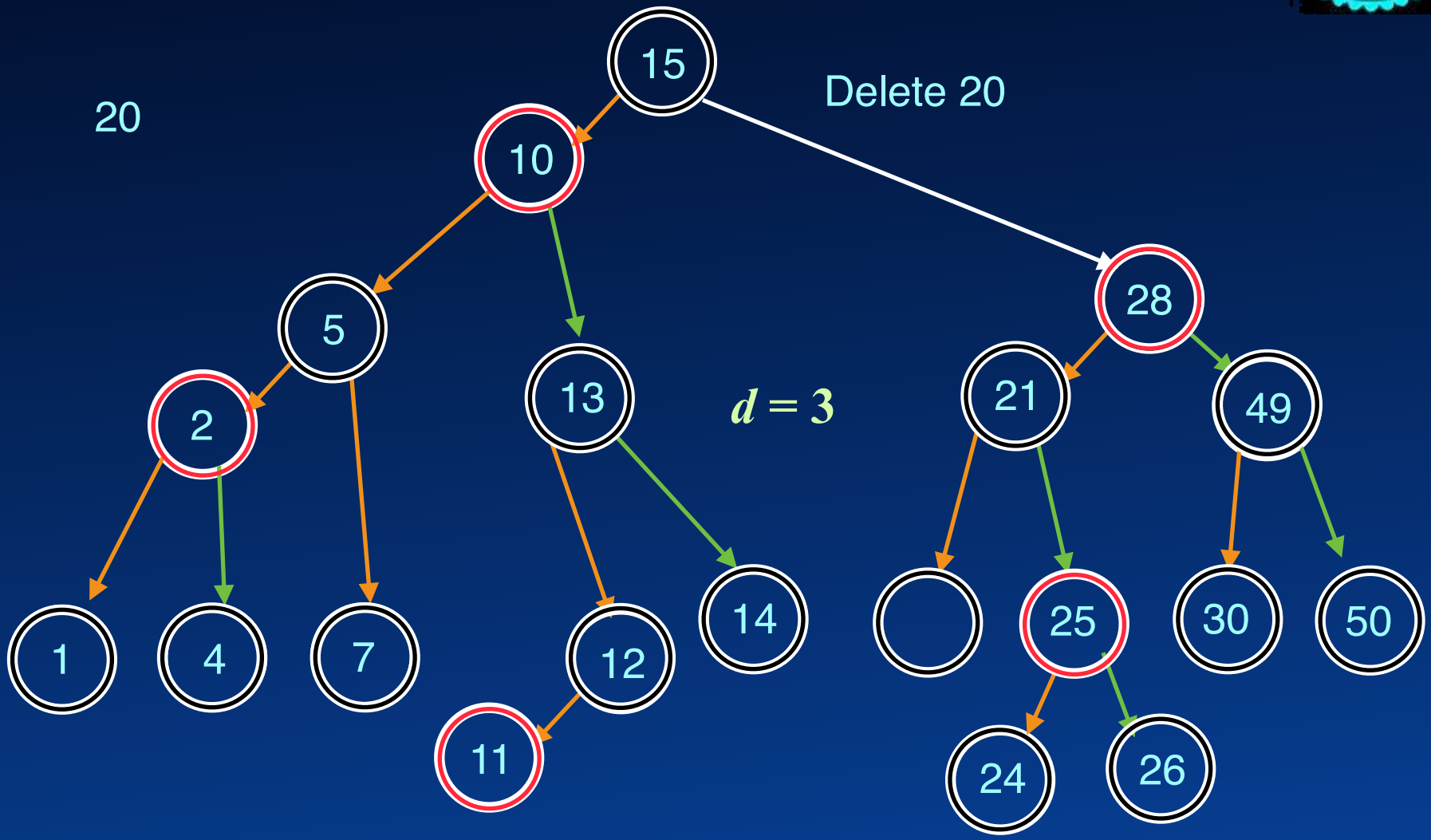
Expunge node with @null in node ***n***

Red-Black Tree



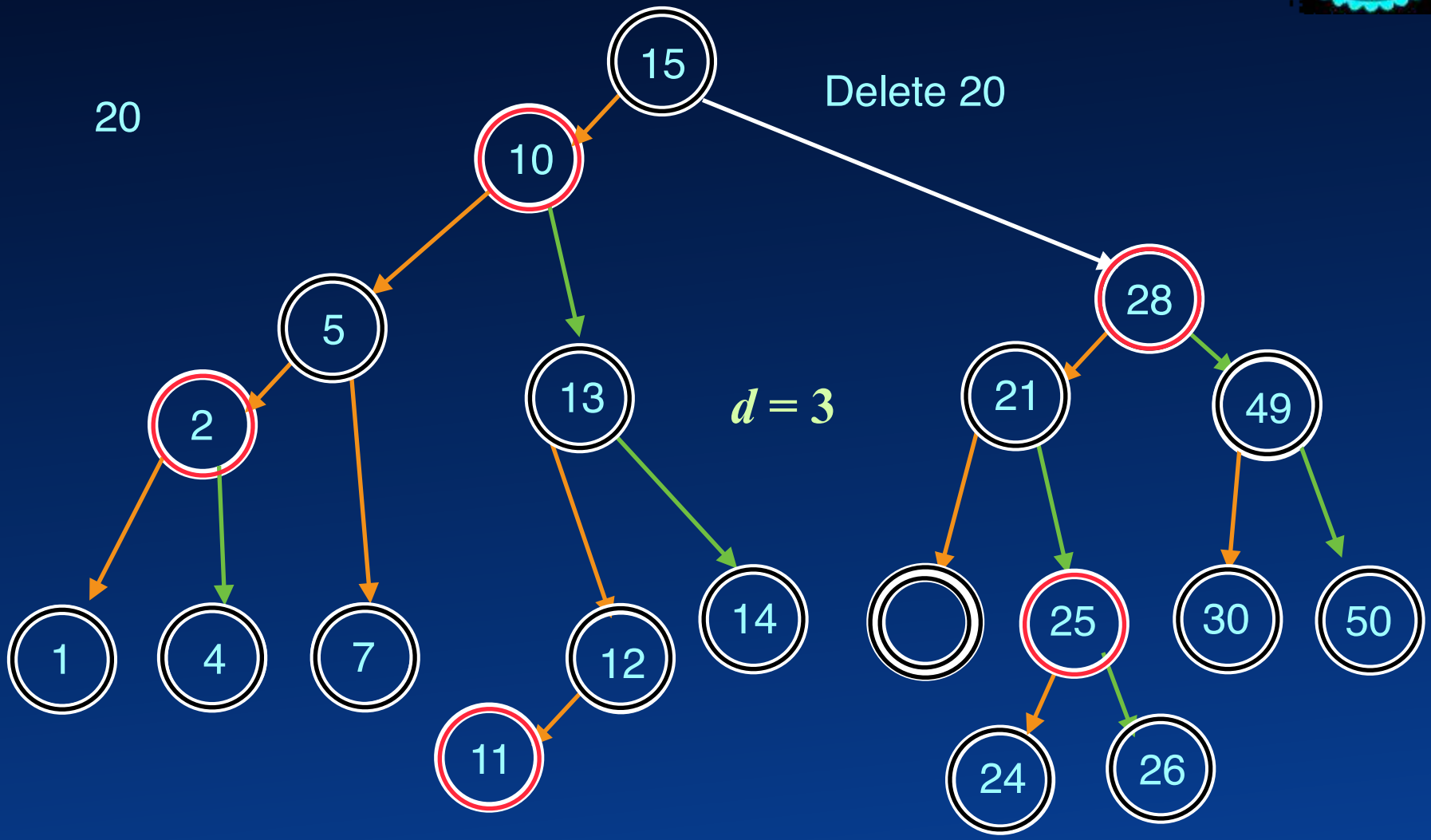
Expunge node with @null in node ***n***

Red-Black Tree



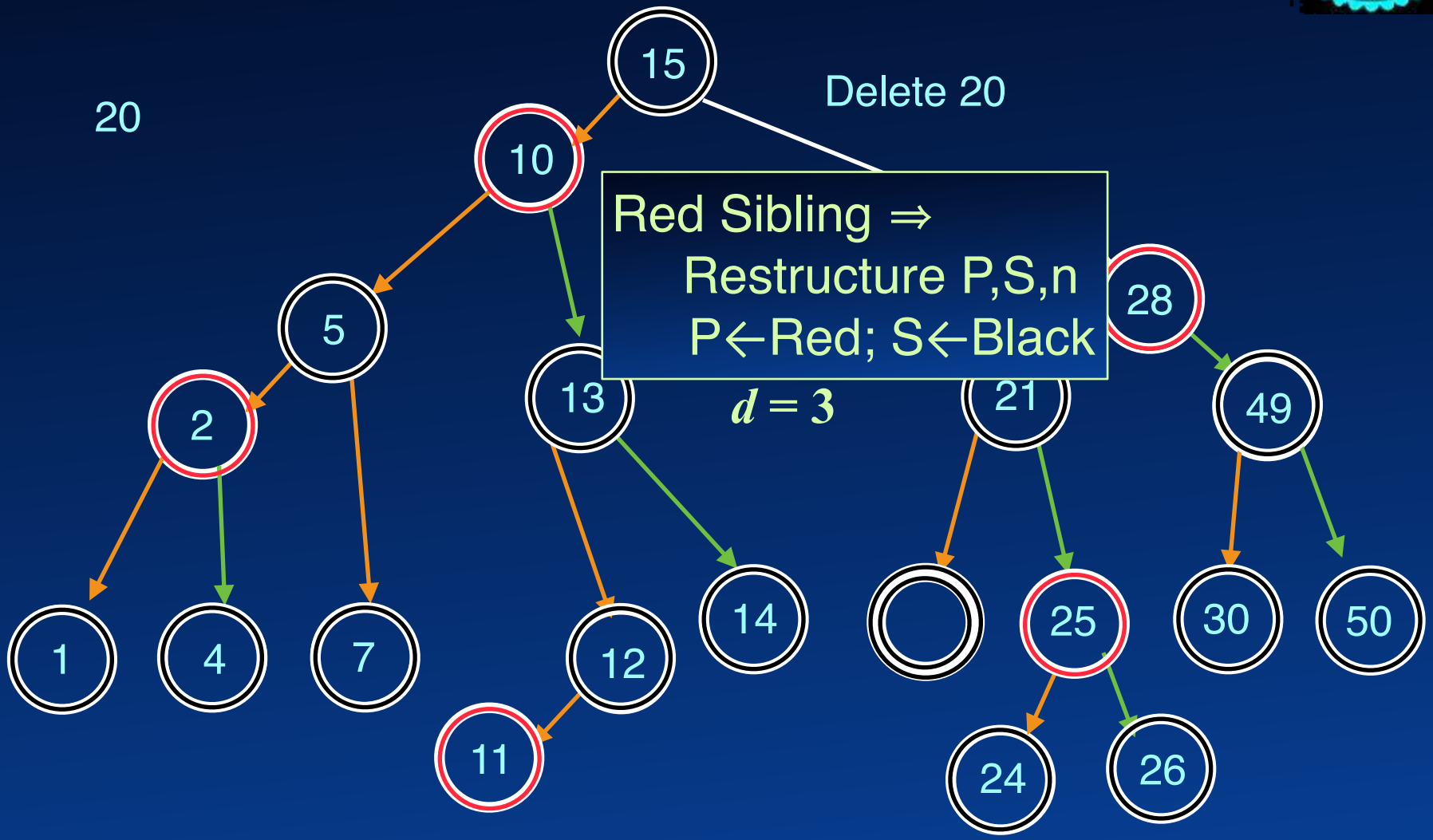
Expunge node with @null in node ***n***

Red-Black Tree



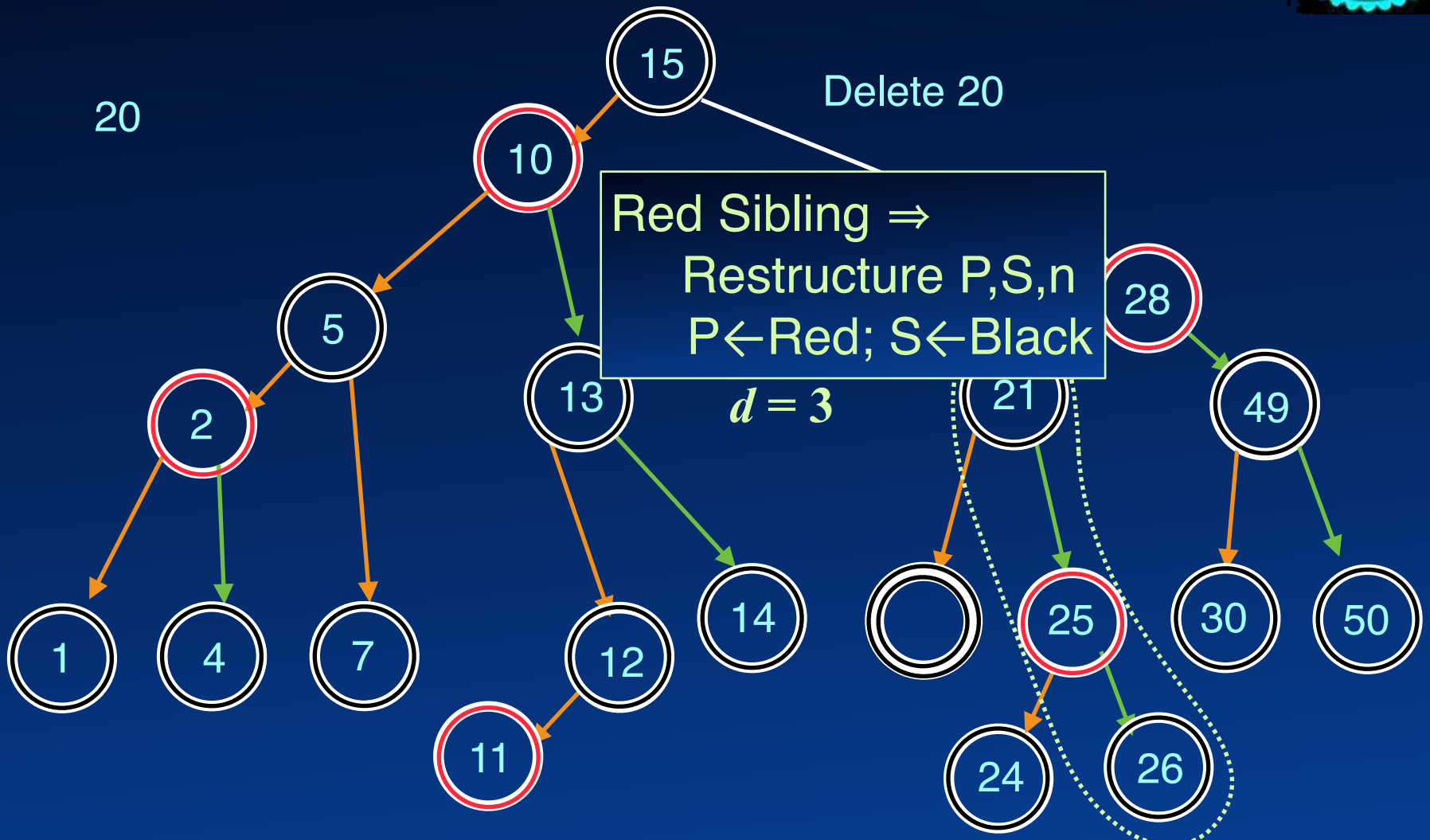
Expunge node with @null in node ***n***

Red-Black Tree



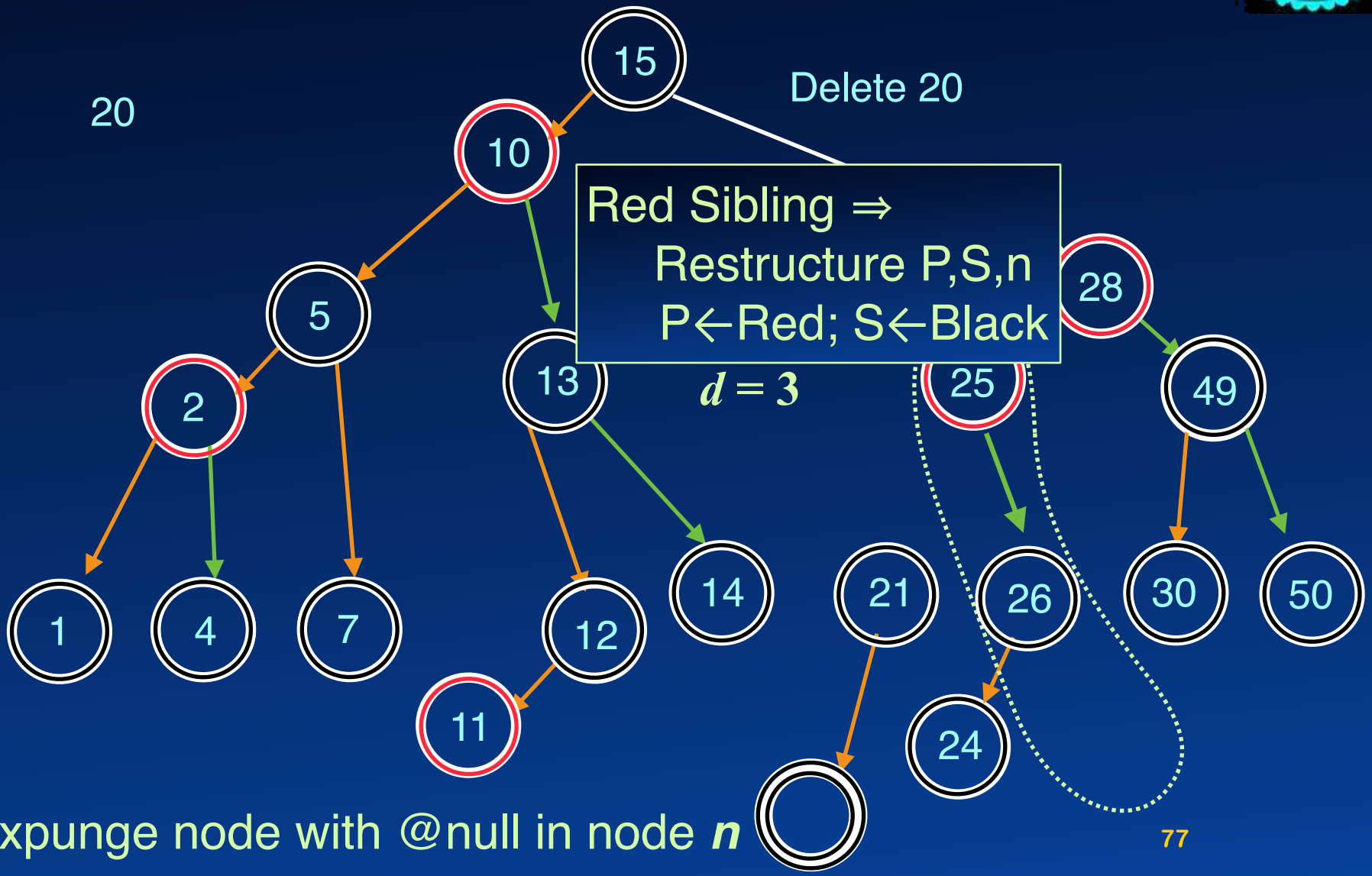
Expunge node with @null in node ***n***

Red-Black Tree



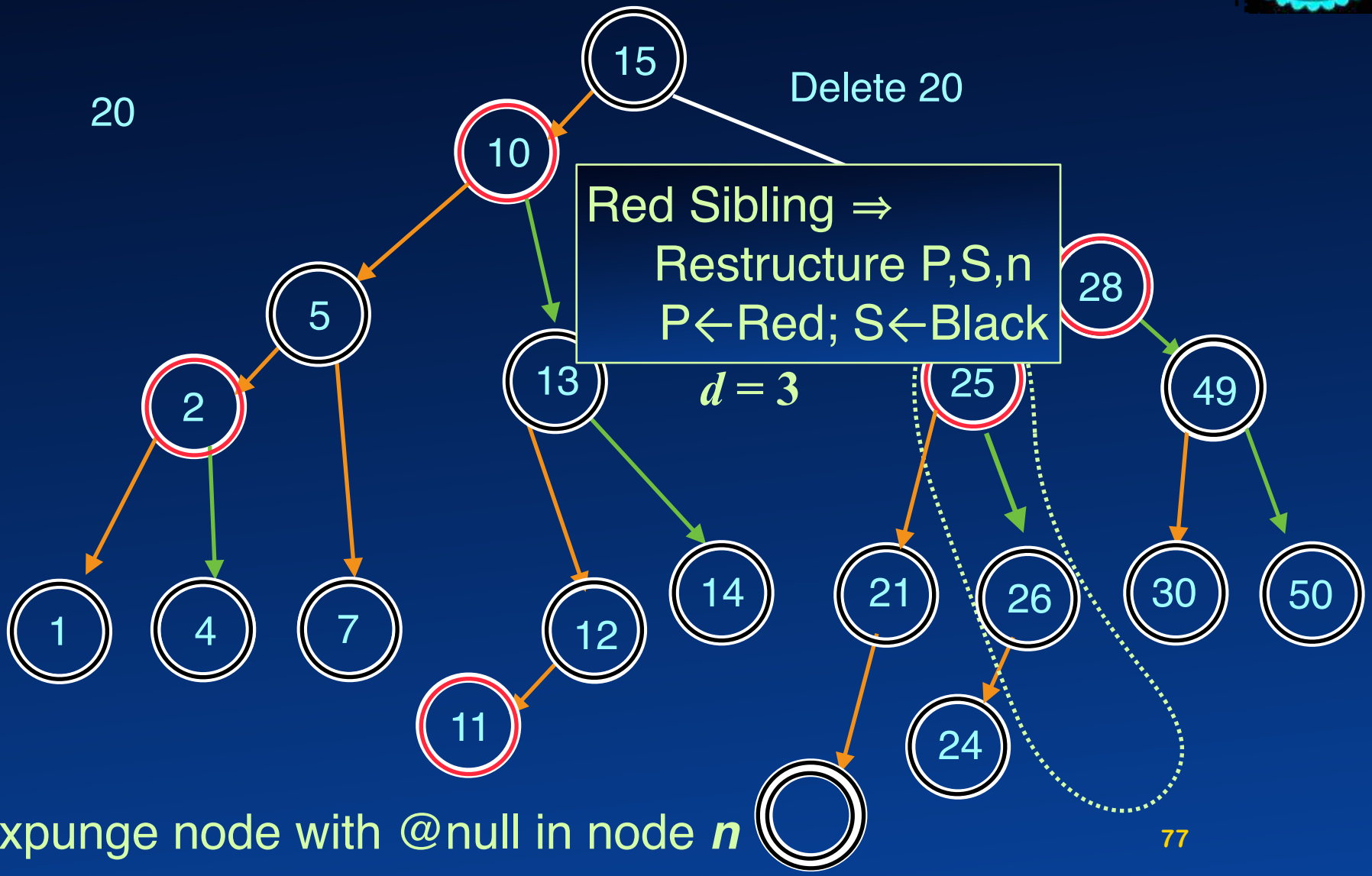
Expunge node with @null in node ***n***

Red-Black Tree



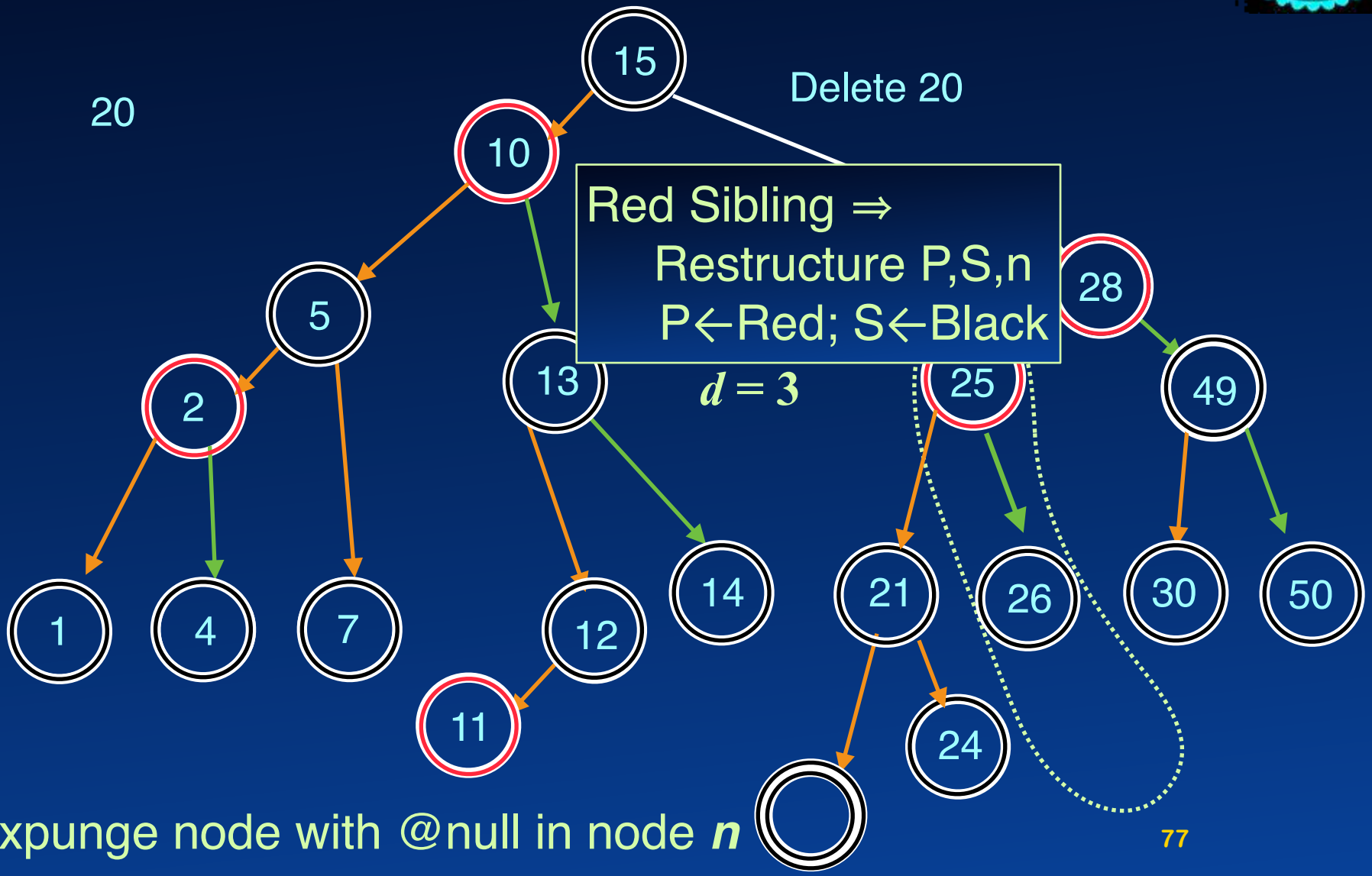
Expunge node with @null in node ***n***

Red-Black Tree



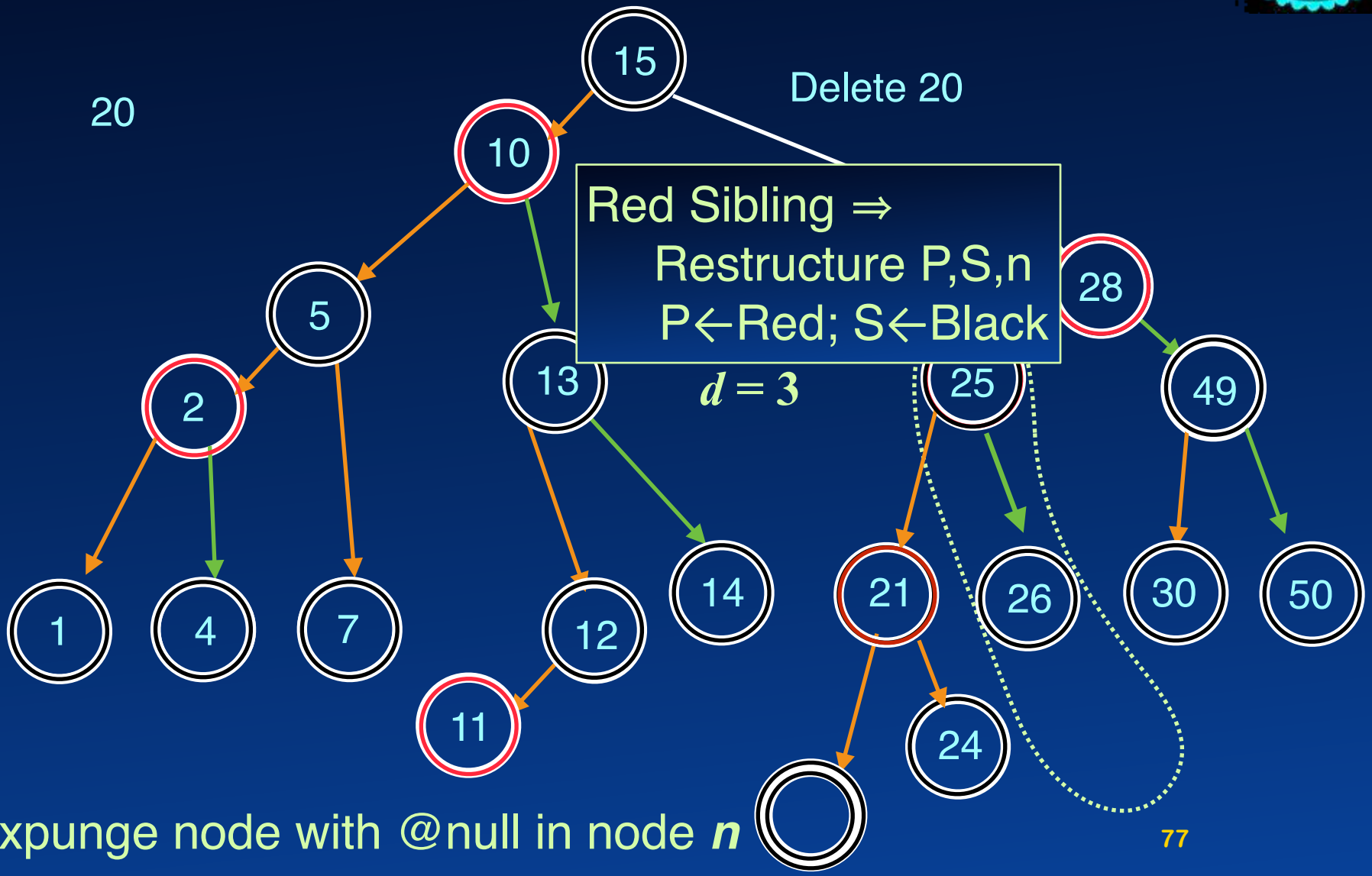
Expunge node with @null in node ***n***

Red-Black Tree



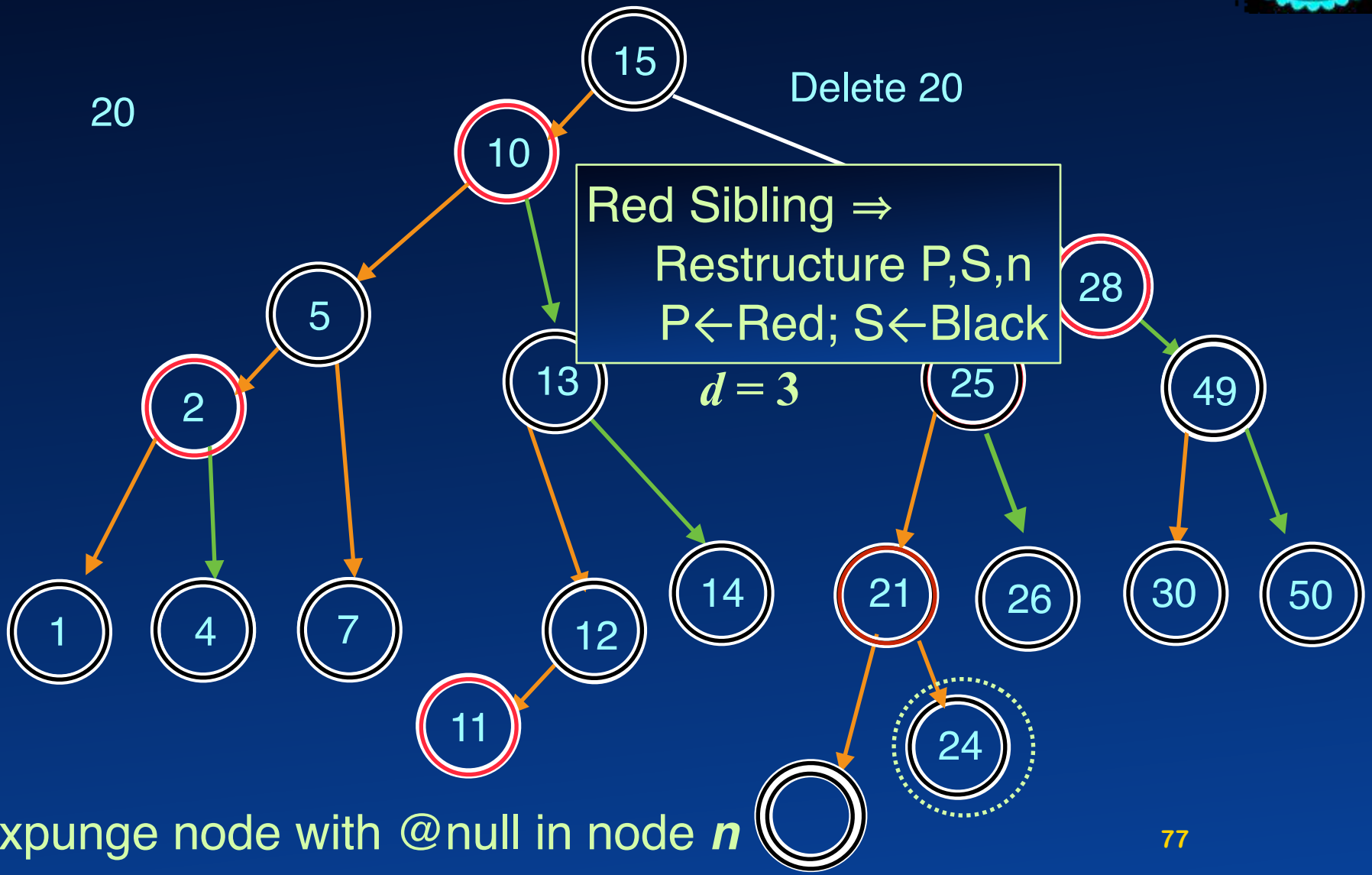
Expunge node with @null in node ***n***

Red-Black Tree



Expunge node with @null in node ***n***

Red-Black Tree



Expunge node with @null in node ***n***

Red-Black Tree

Black Sibling ⇒

No red nephew: Recolor S,P; Fix P.

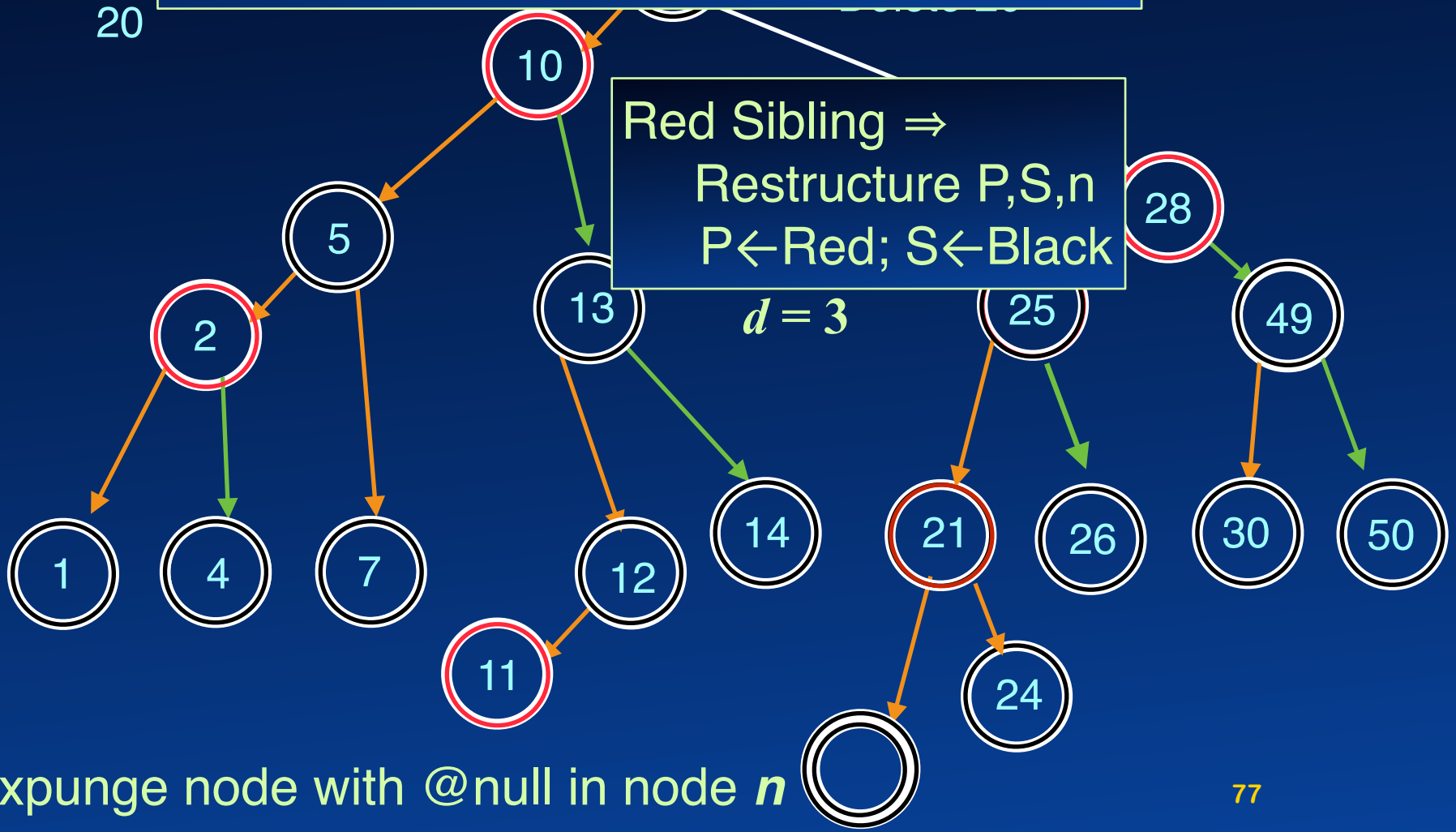
Red nephew: Restructure P,S,n; Done.

Red Sibling ⇒

Restructure P,S,n

P←Red; S←Black

$d = 3$



Expunge node with @null in node ***n***

Red-Black Tree

Black Sibling ⇒

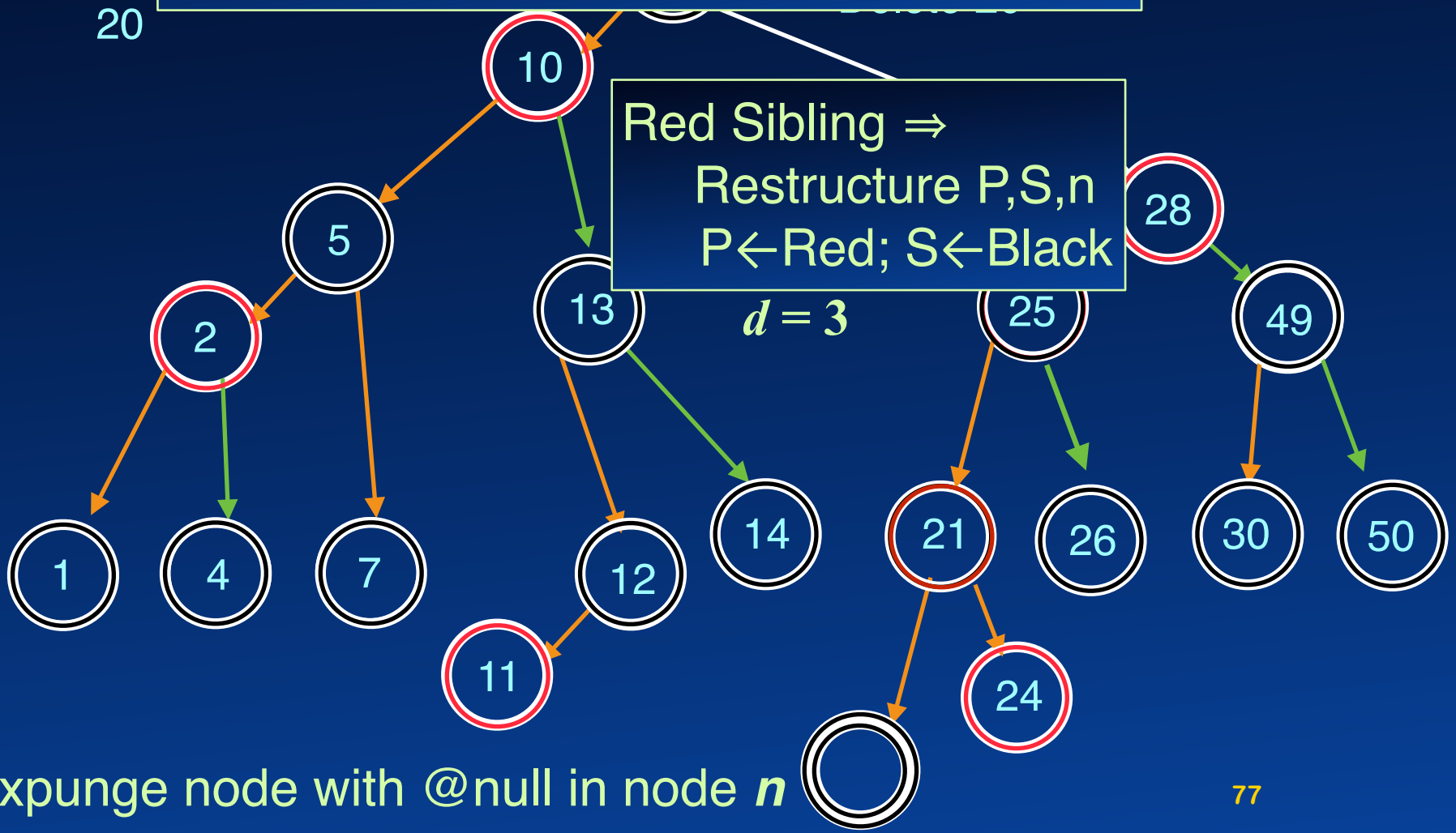
No red nephew: Recolor S,P; Fix P.

Red nephew: Restructure P,S,n; Done.

Red Sibling ⇒

Restructure P,S,n

P←Red; S←Black



$d = 3$

Expunge node with @null in node ***n***

Red-Black Tree

Black Sibling ⇒

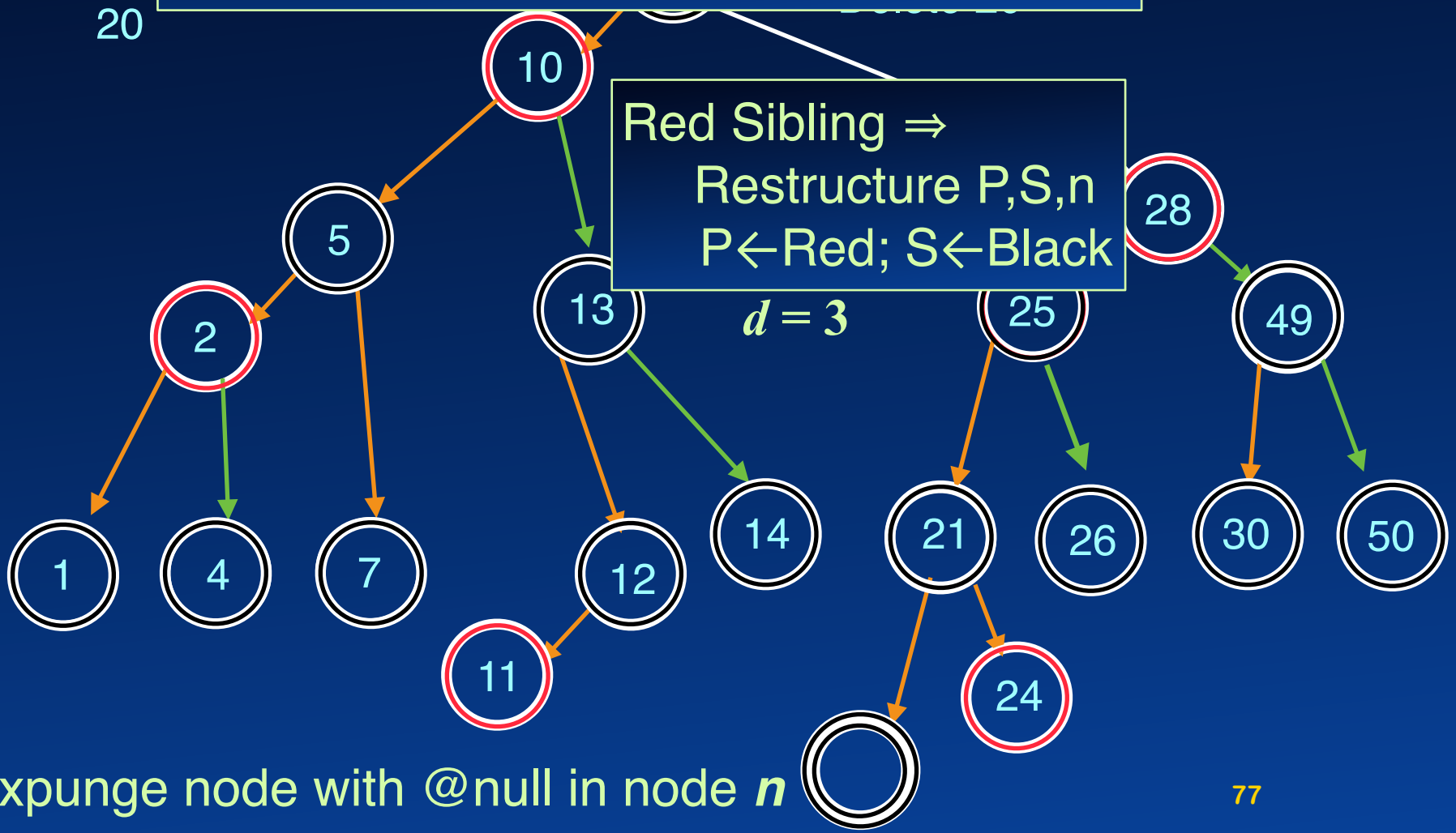
No red nephew: Recolor S,P; Fix P.

Red nephew: Restructure P,S,n; Done.

Red Sibling ⇒

Restructure P,S,n

P←Red; S←Black



Expunge node with @null in node ***n***

Red-Black Tree

Black Sibling ⇒

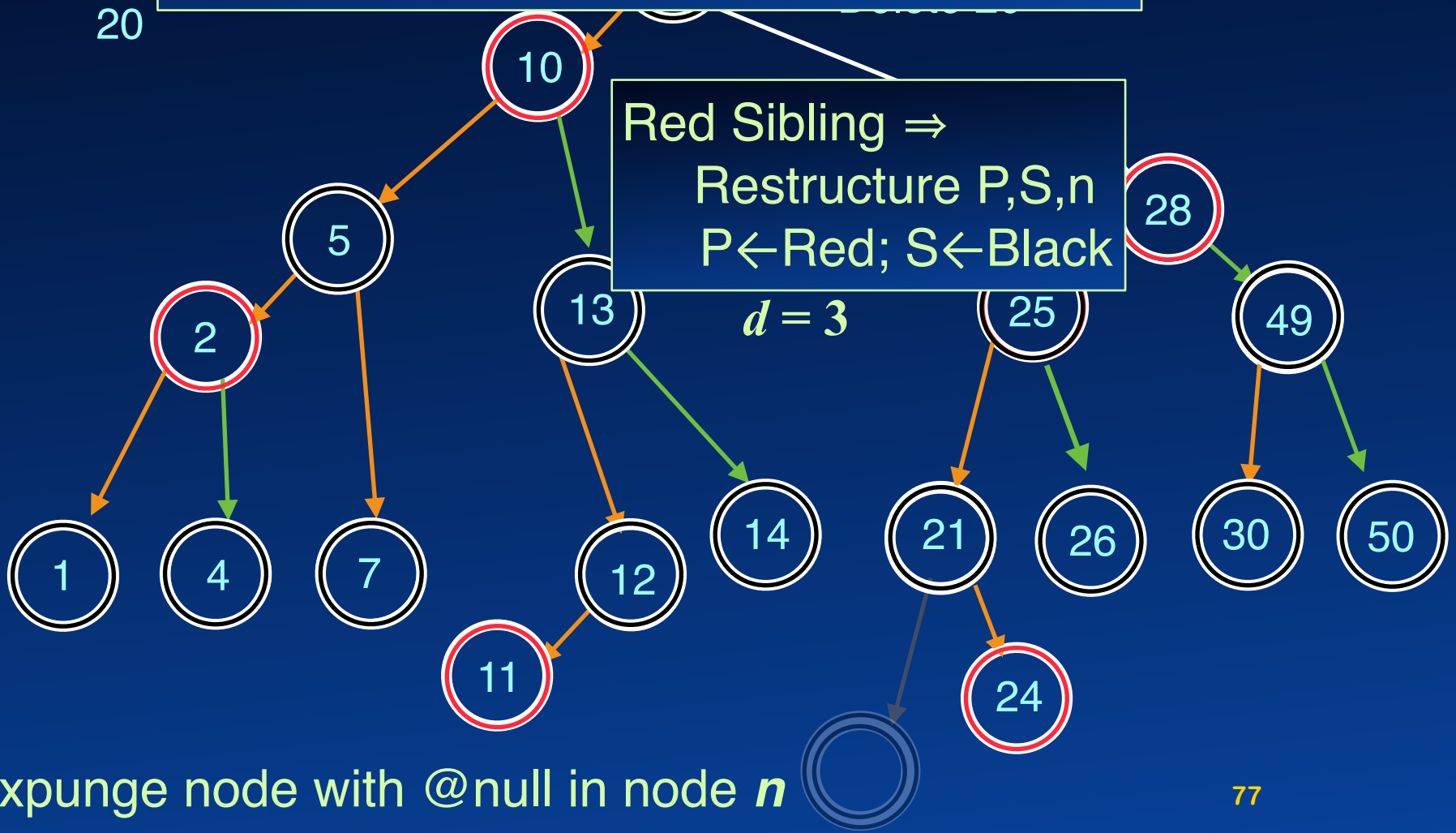
No red nephew: Recolor S,P; Fix P.

Red nephew: Restructure P,S,n; Done.

Red Sibling ⇒

Restructure P,S,n

P←Red; S←Black



$d = 3$

Expunge node with @null in node ***n***

Red-Black Tree

Black Sibling ⇒

No red nephew: Recolor S,P; Fix P.

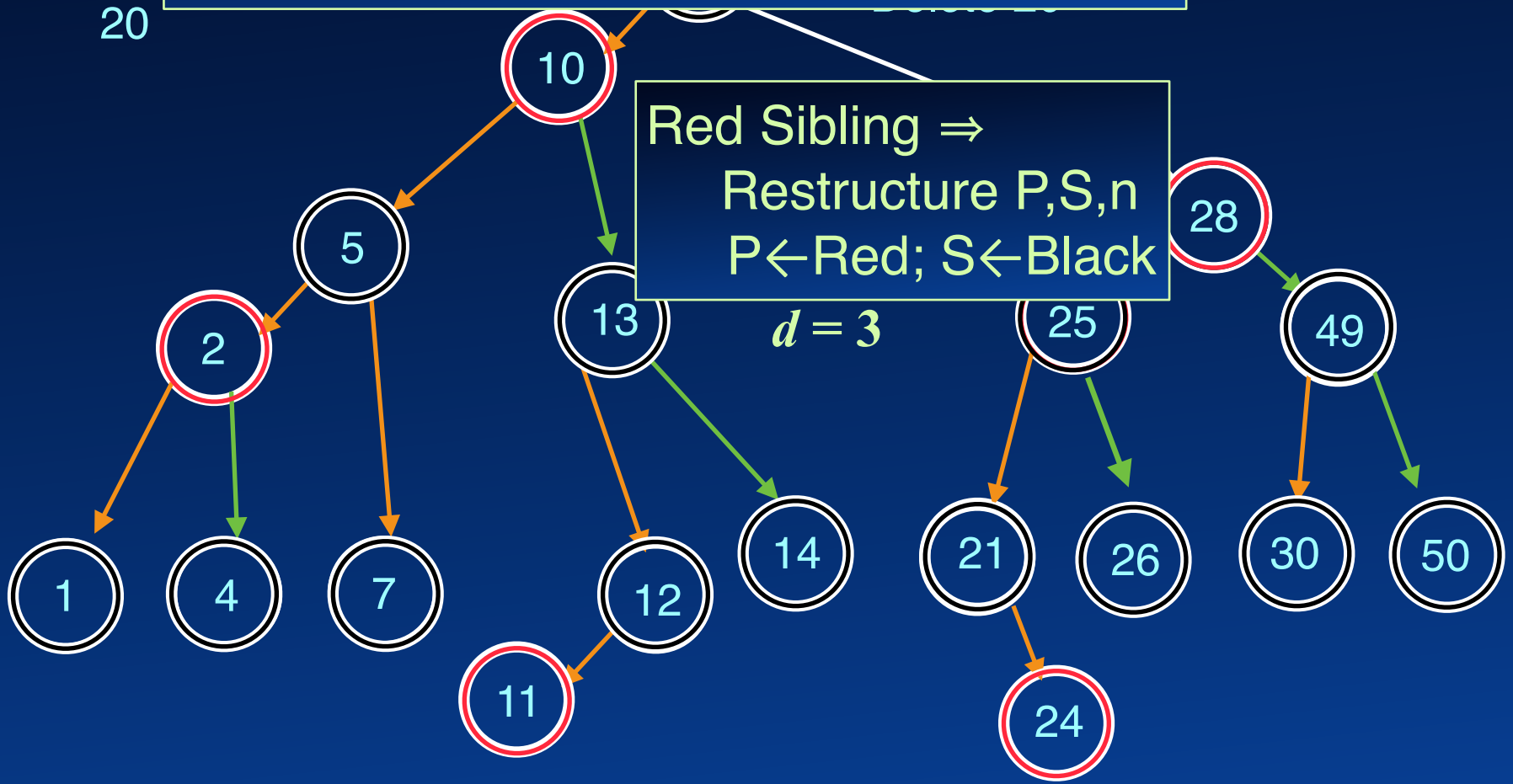
Red nephew: Restructure P,S,n; Done.

Red Sibling ⇒

Restructure P,S,n

P←Red; S←Black

$d = 3$



Expunge node with @null in node ***n***