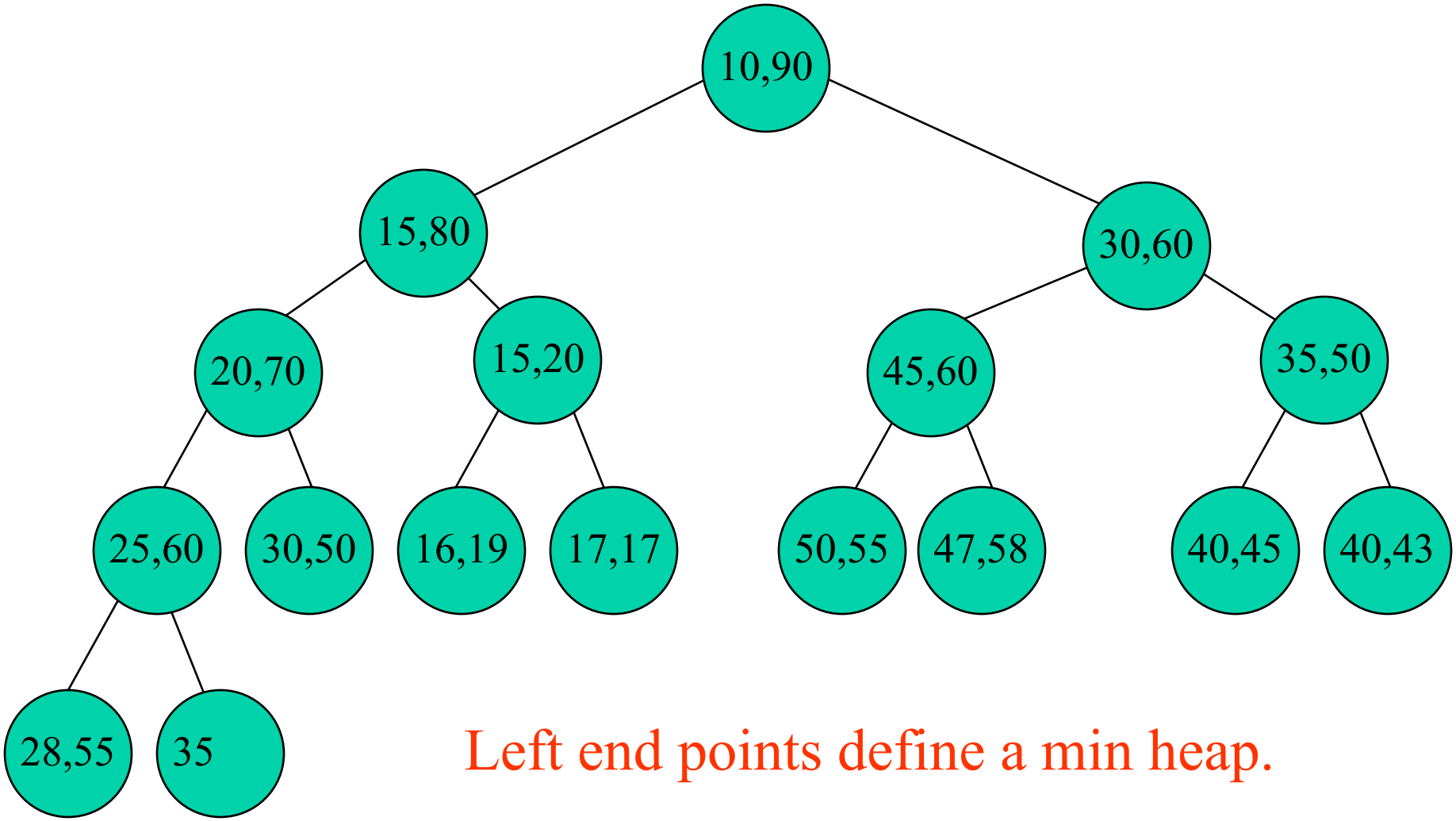
# Interval Heaps

- Complete binary tree.
- Each node (except possibly last one) has 2 elements.
- Last node has 1 or 2 elements.
- Let a and b be the elements in a node P, a <= b.
- [a, b] is the interval represented by P.
- The interval represented by a node that has just one element a is [a, a].
- The interval [c, d] is contained in interval [a, b] iff a <= c <= d <= b.
- In an interval heap each node's (except for root) interval is contained in that of its parent.

# Interval


- [c,d] is contained in [a,b]
- a <= c
- d <= b

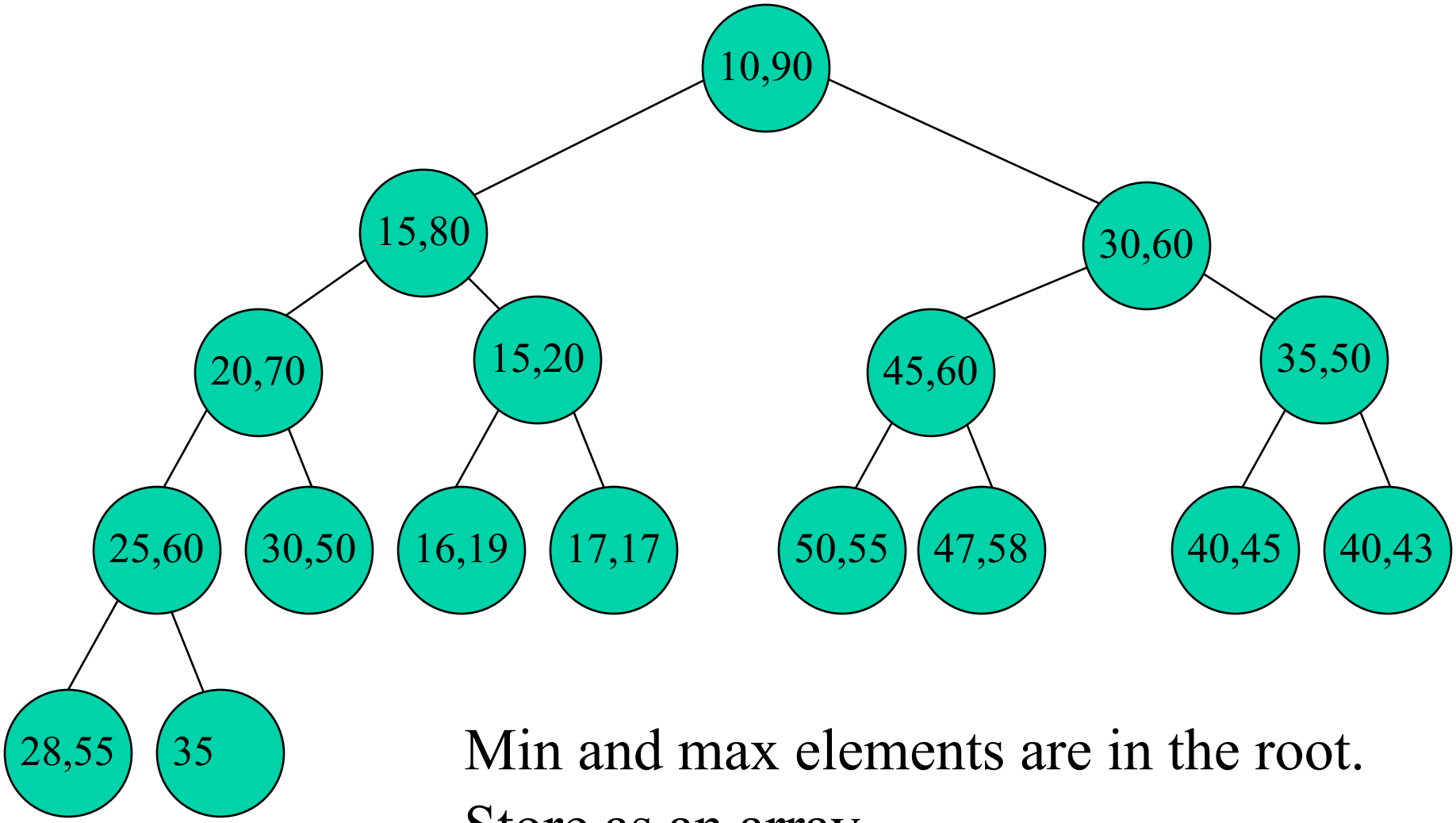
# Example Interval Heap




Left end points define a min heap.

Right end points define a max heap.

# Example Interval Heap

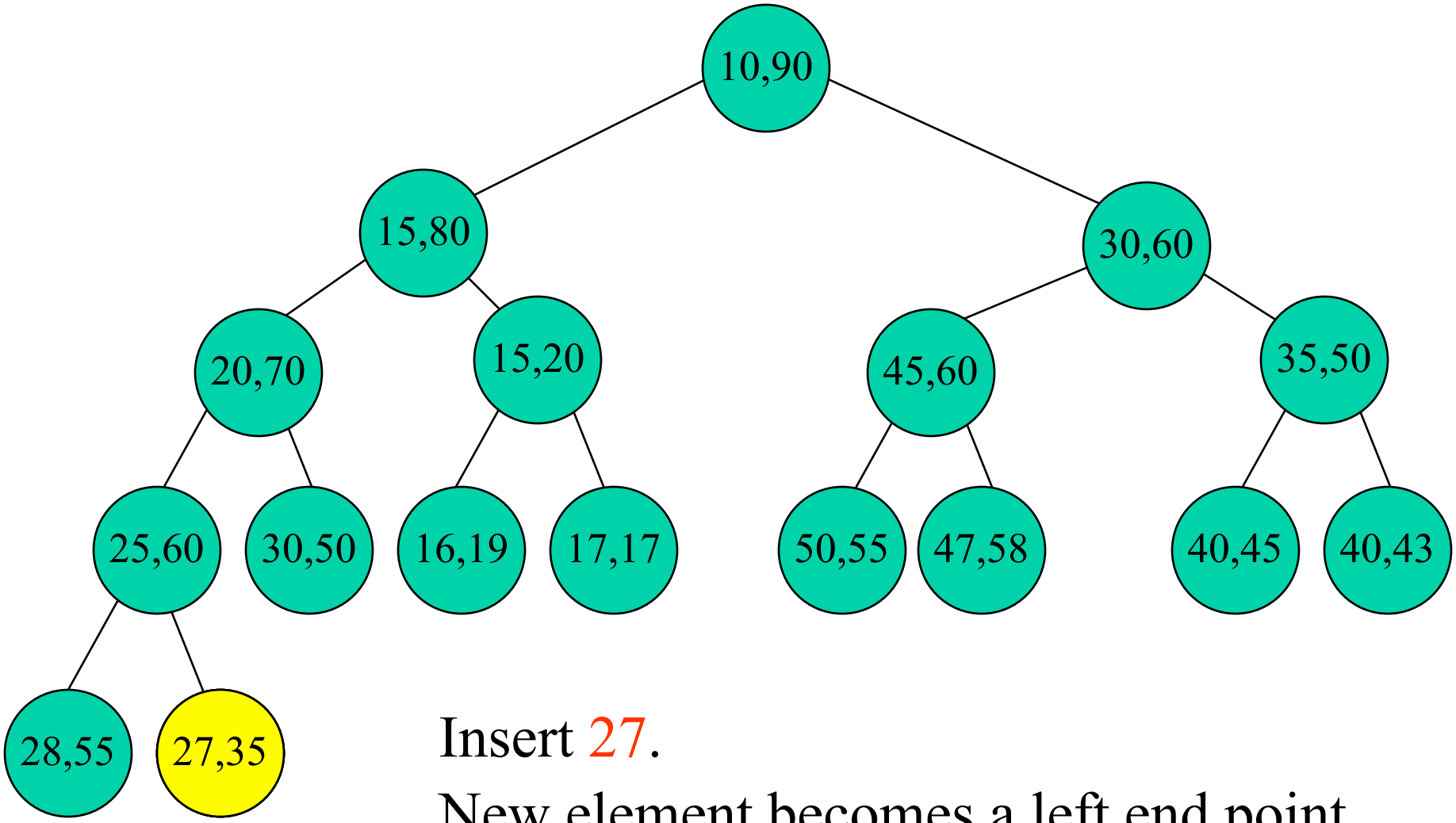



Min and max elements are in the root.

Store as an array.

Height is ~$\log_2 n$.

# Insert An Element

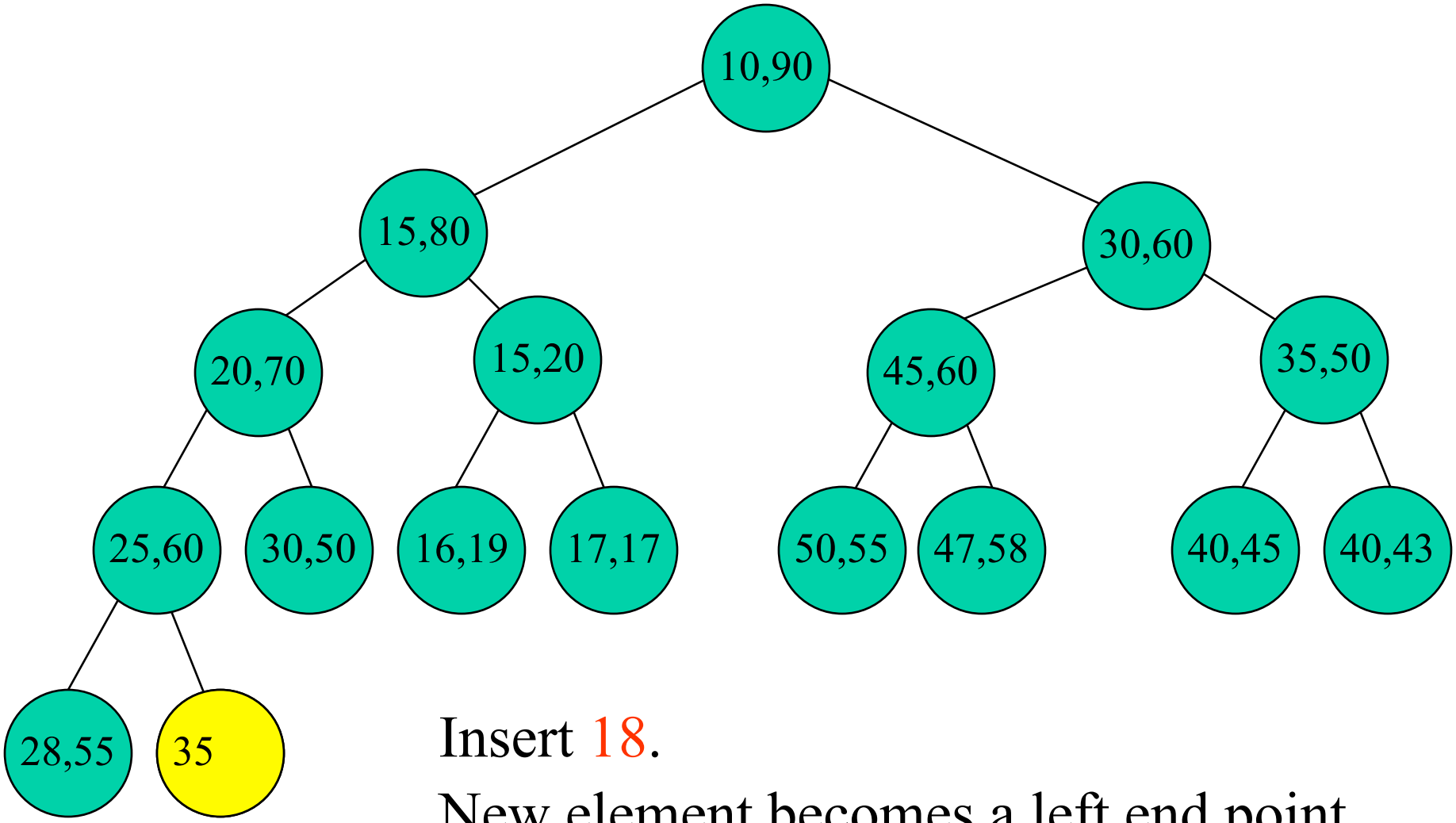



Insert 27.

New element becomes a left end point.

Insert new element into min heap.

# Another Insert

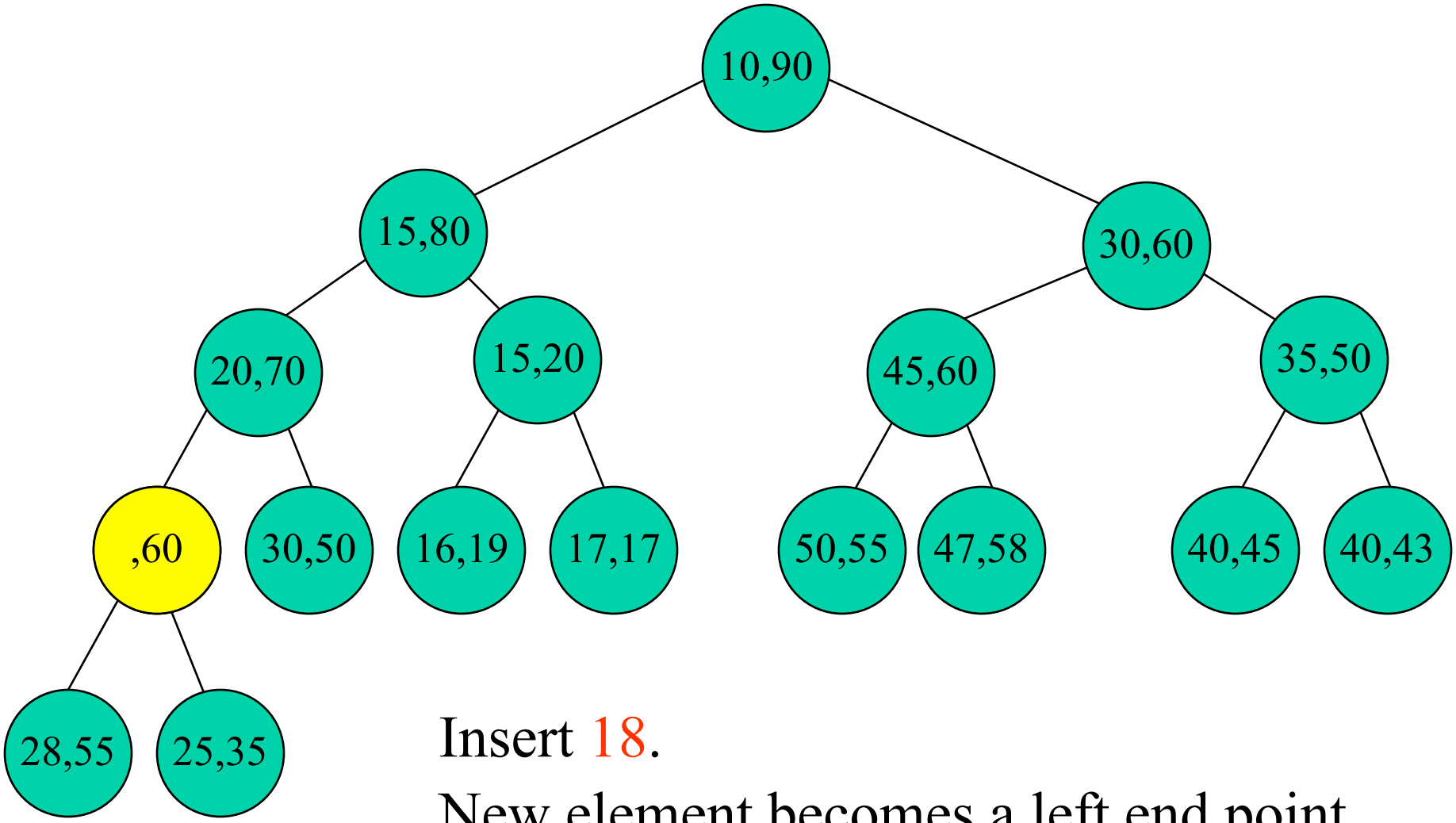



Insert 18.

New element becomes a left end point.

Insert new element into min heap.

# Another Insert

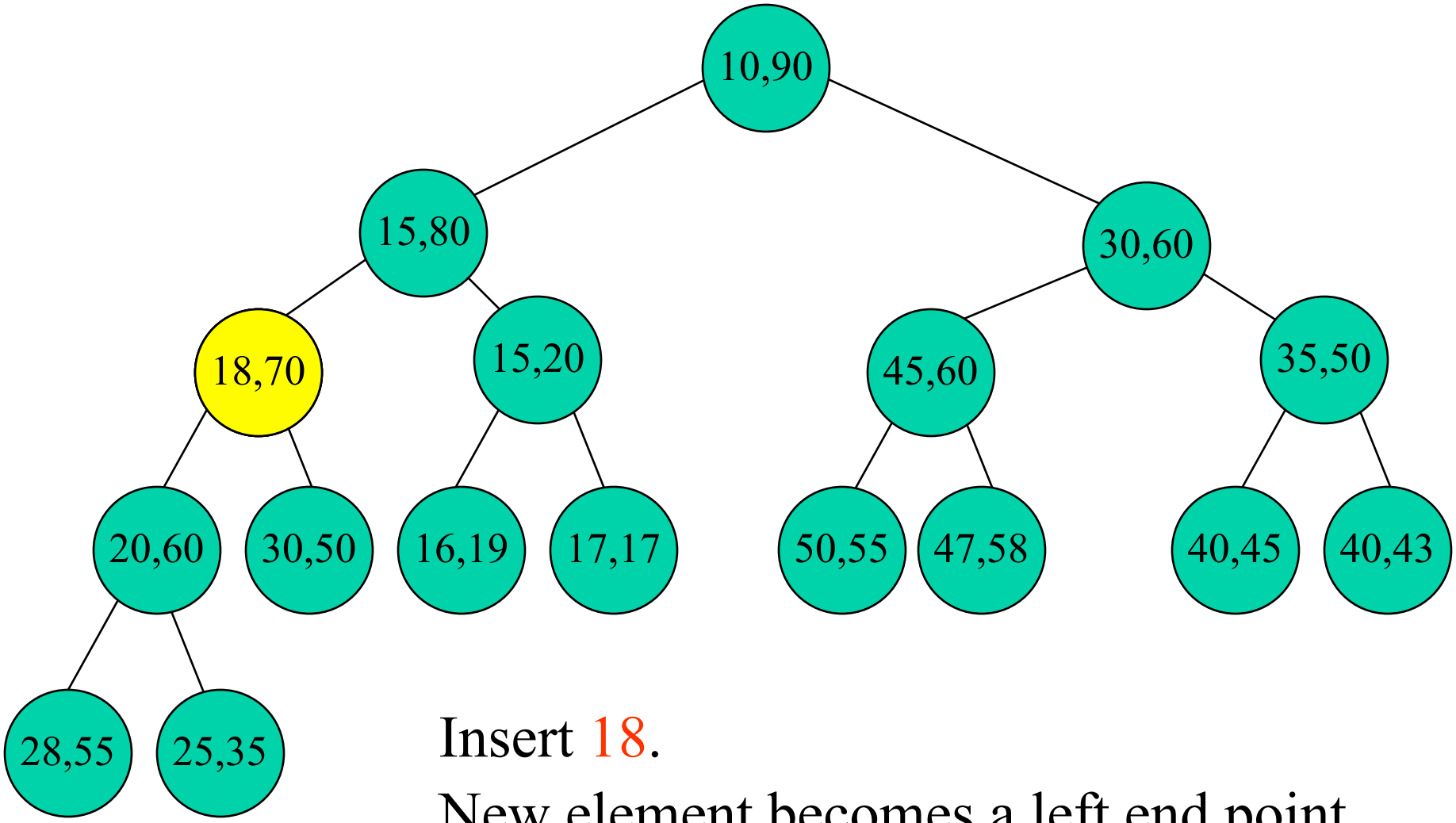



Insert 18.

New element becomes a left end point.

Insert new element into min heap.

# Another Insert

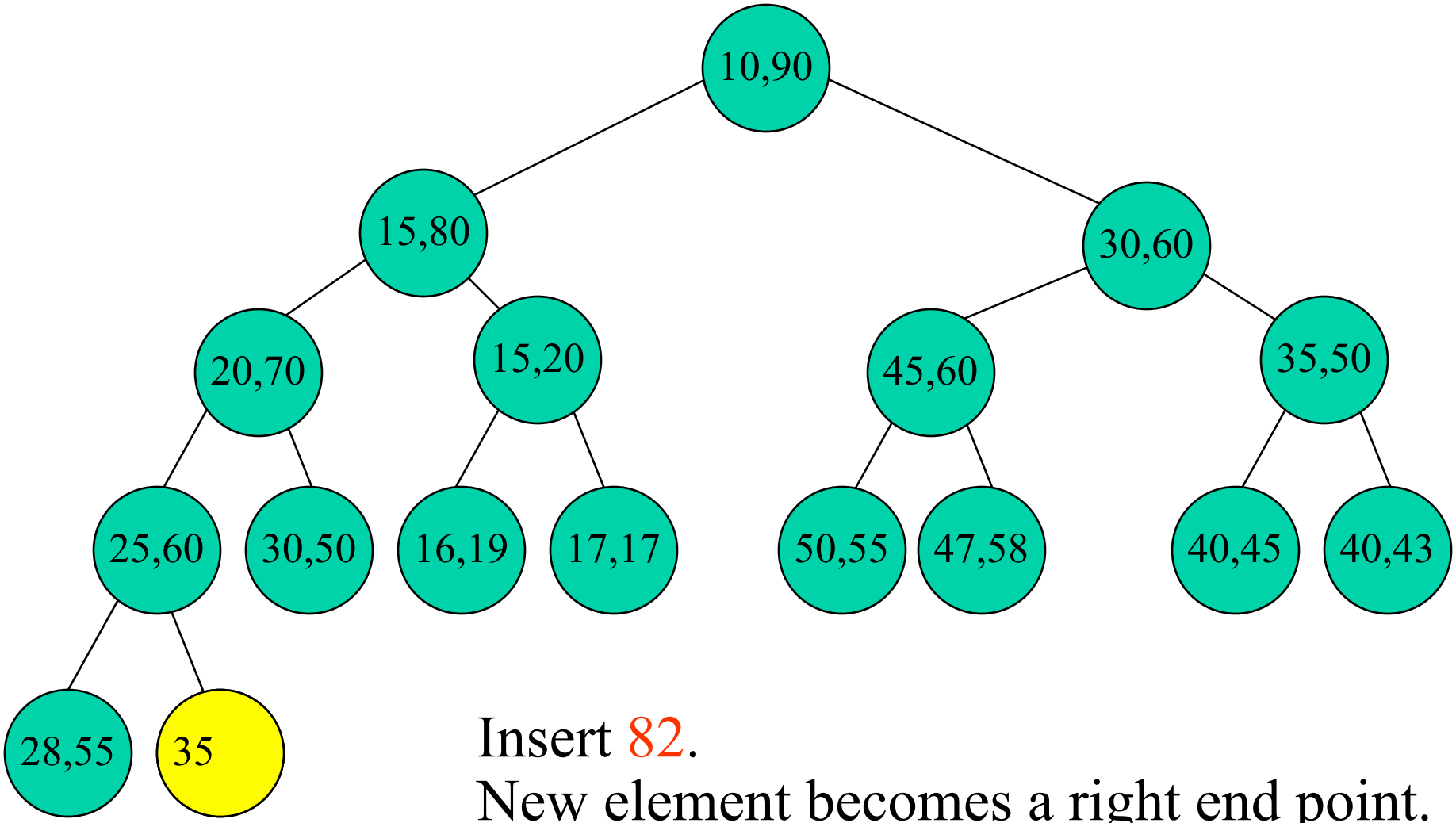



Insert 18.

New element becomes a left end point.

Insert new element into min heap.

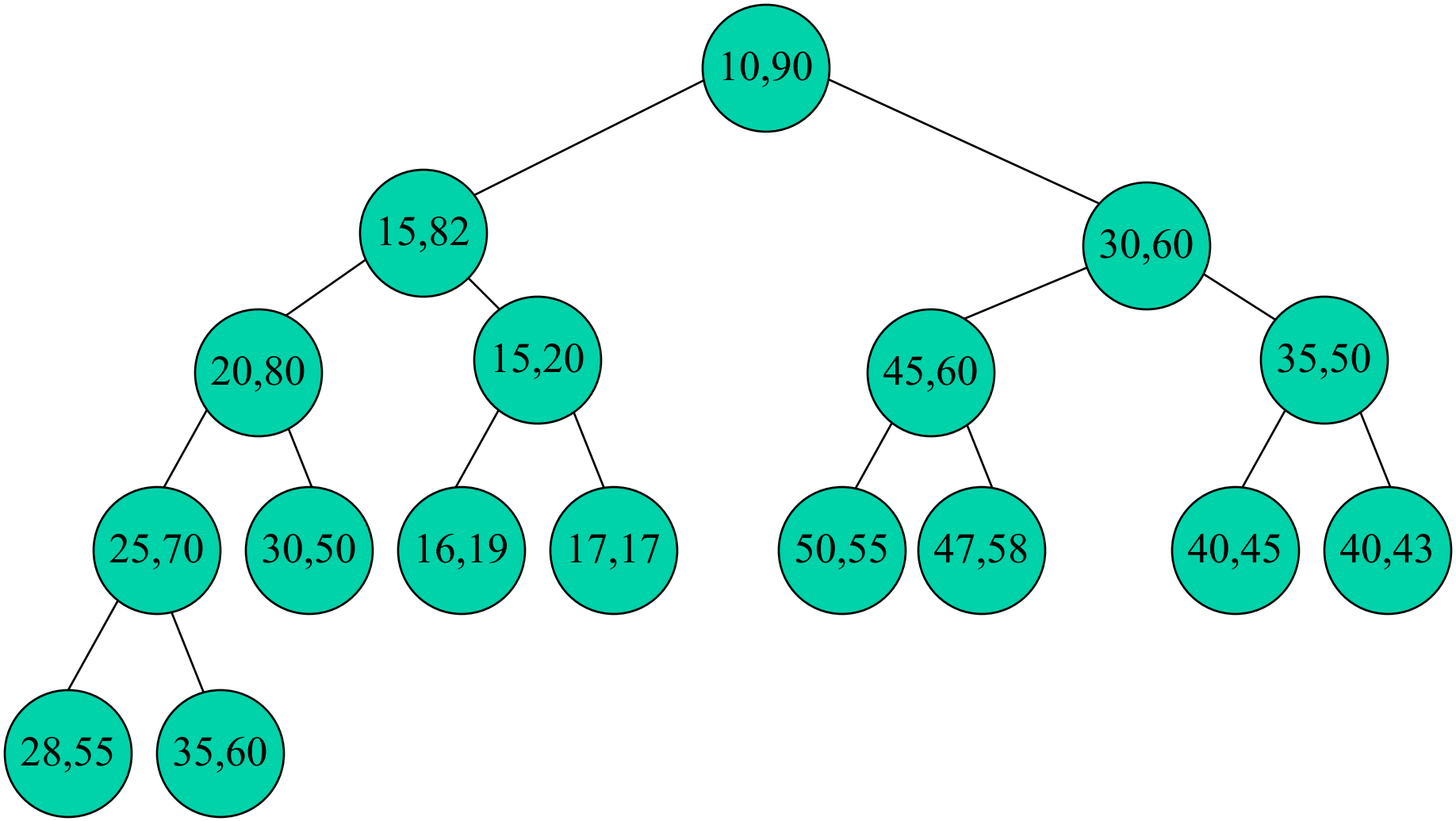
# Yet Another Insert



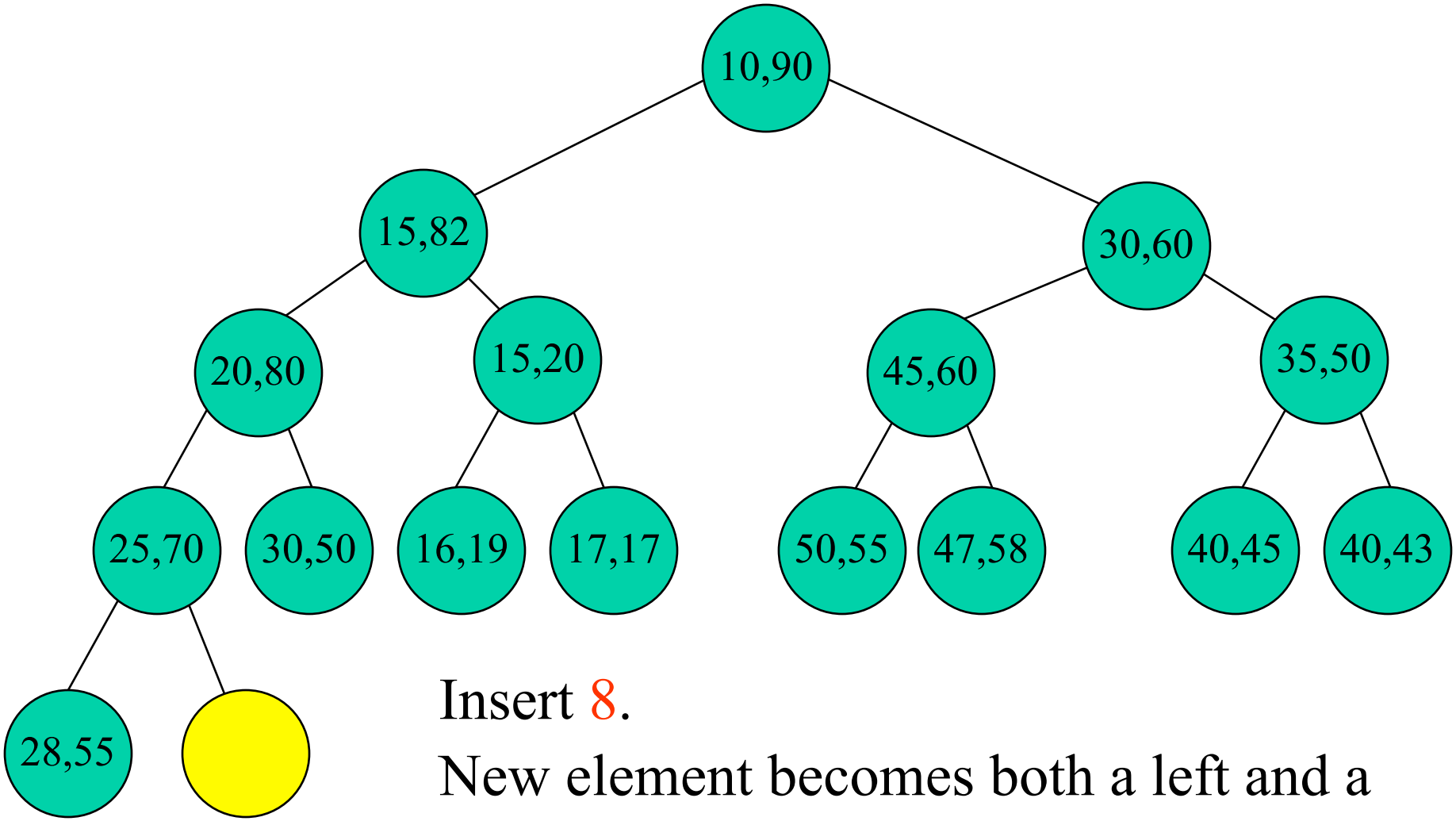

Insert 82.
New element becomes a right end point.

Insert new element into max heap.

# After 82 Inserted



# One More Insert Example

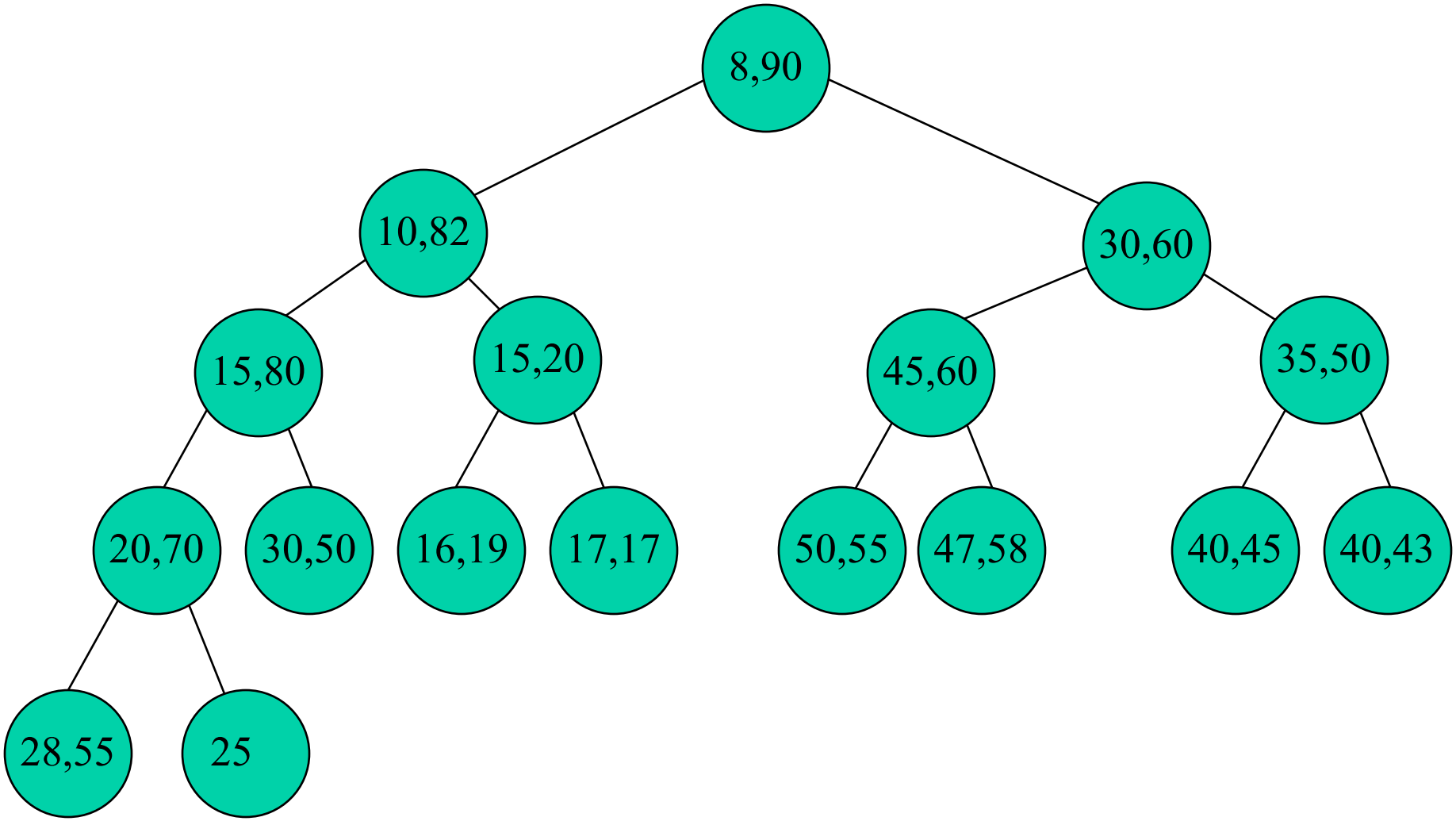



Insert 8.

New element becomes both a left and a right end point.

Insert new element into min heap.

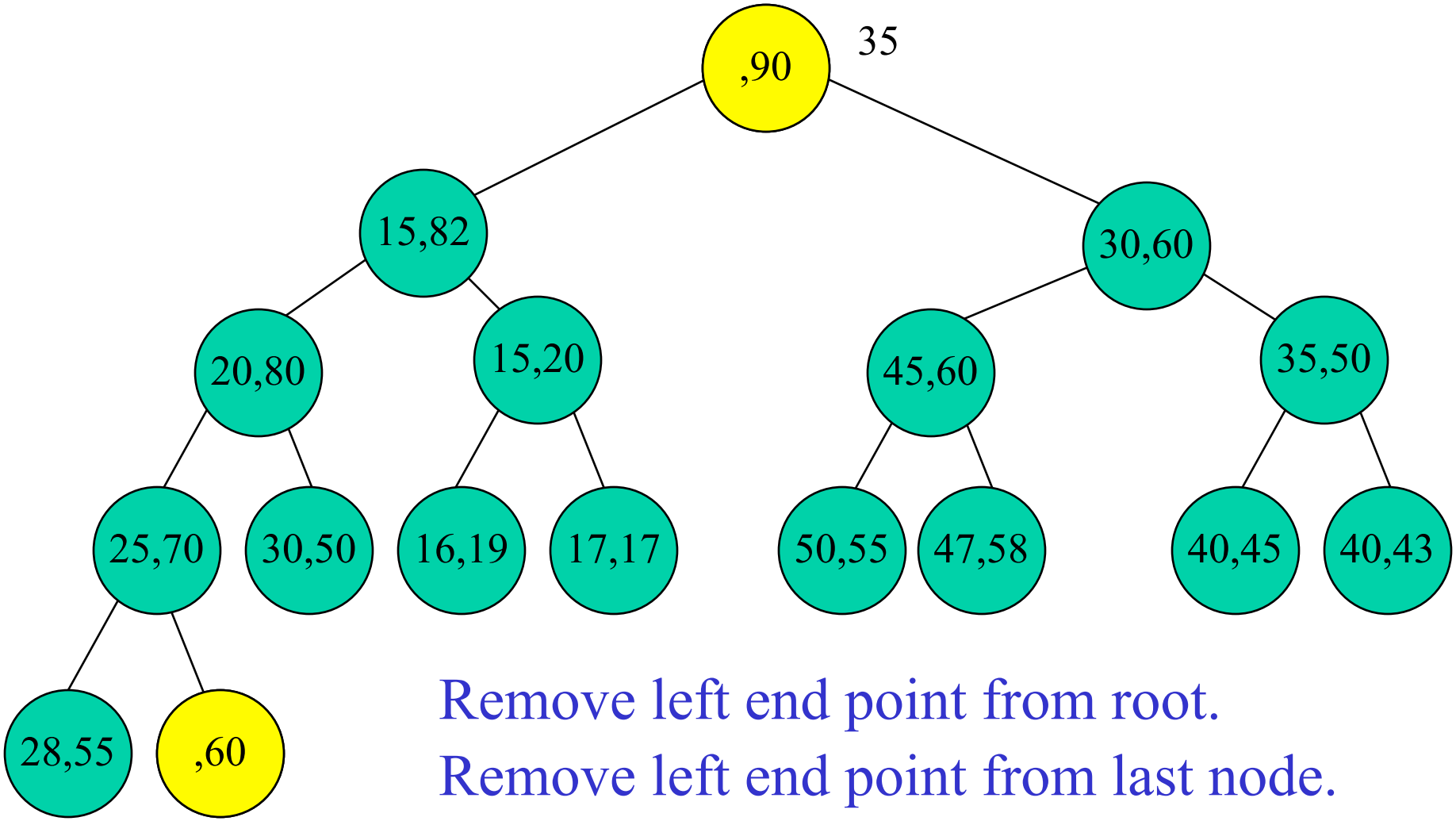
# After 8 Is Inserted



# Remove Min Element

- n = 0 => fail.
- n = 1 => heap becomes empty.
- n = 2 => only one node, take out left end point.
- n > 2 => not as simple.

# Remove Min Element Example



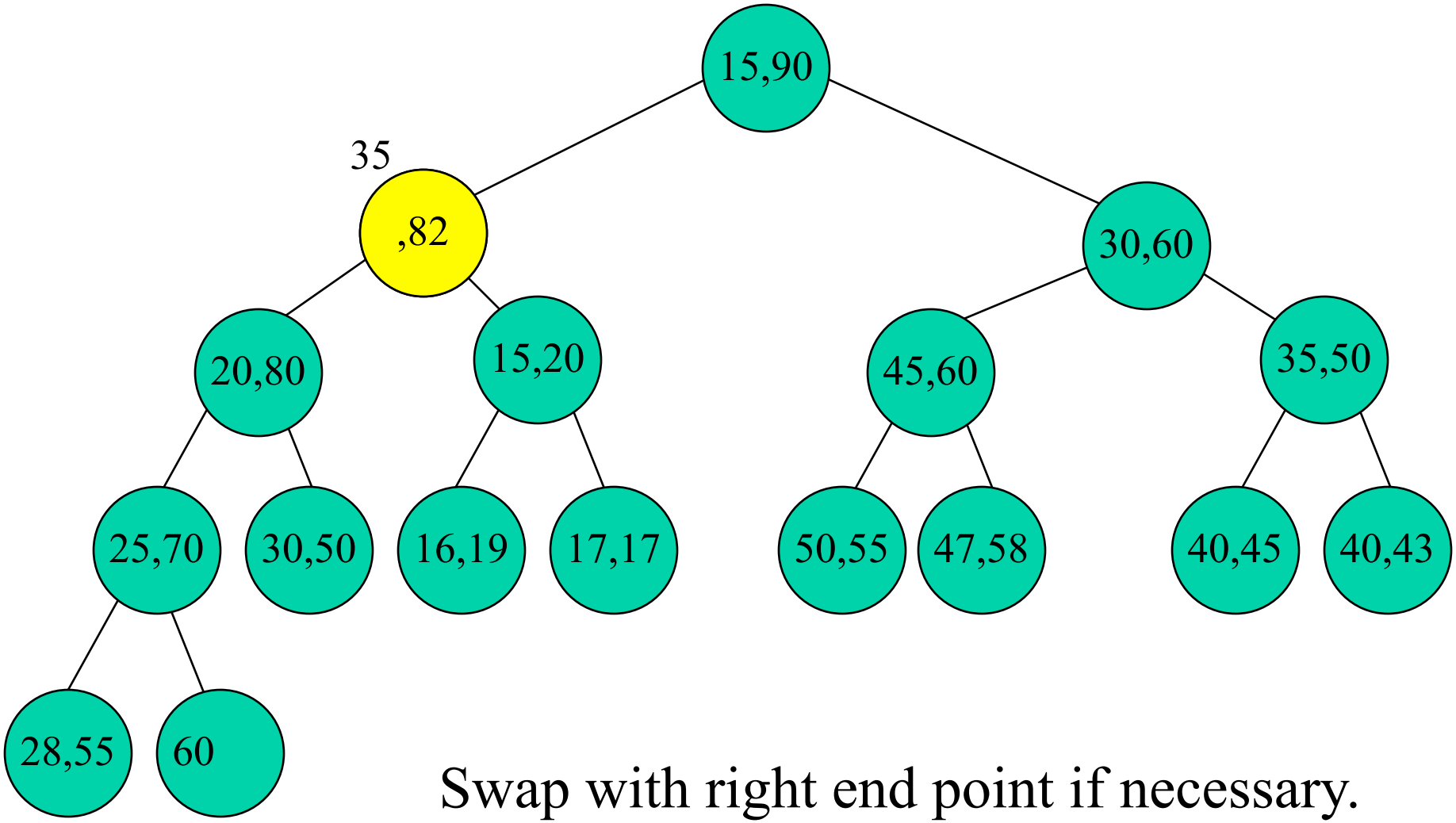

Remove left end point from root.

Remove left end point from last node.

Delete last node if now empty.

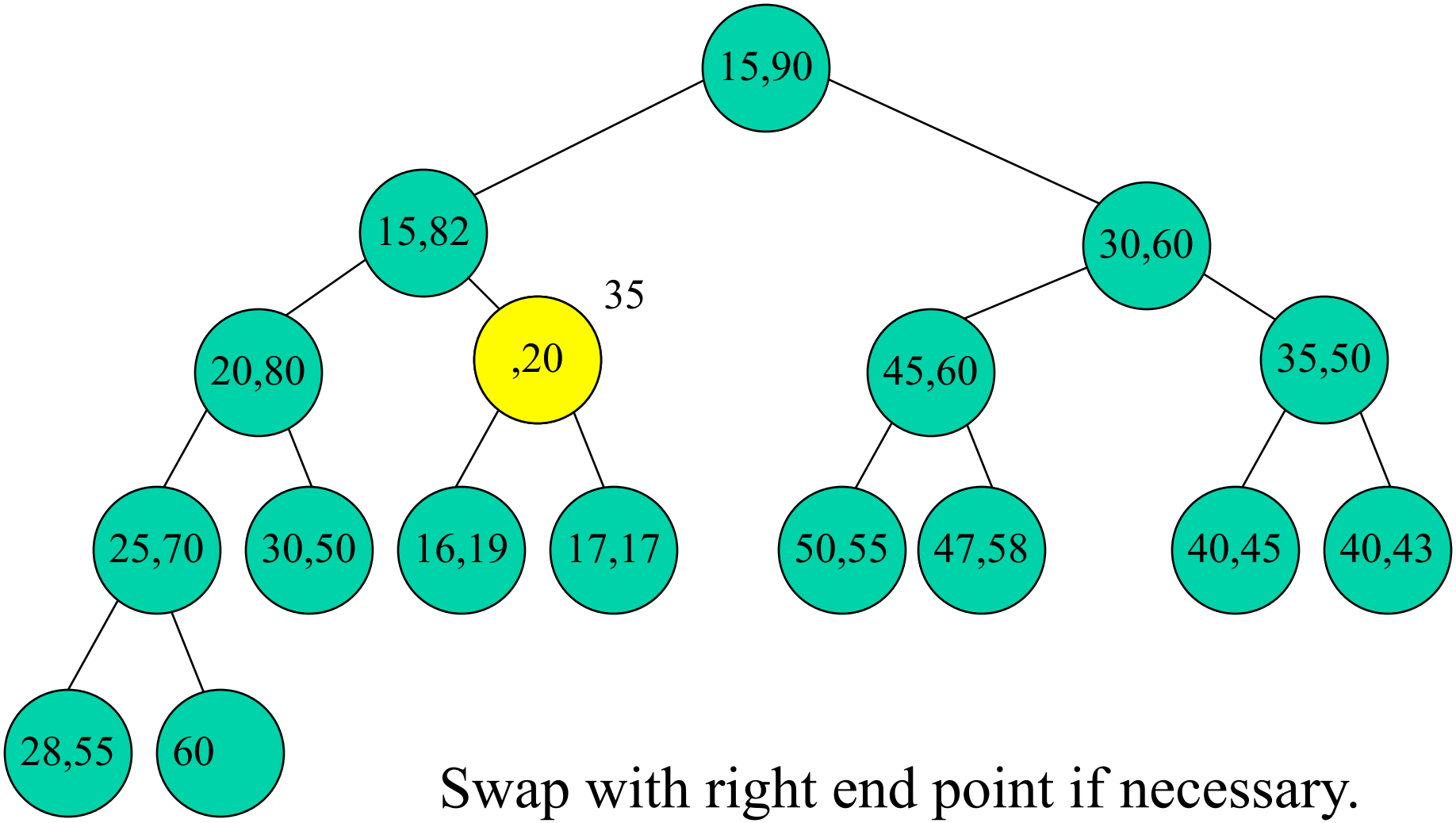
Reinsert into min heap, begin at root.

# Remove Min Element Example



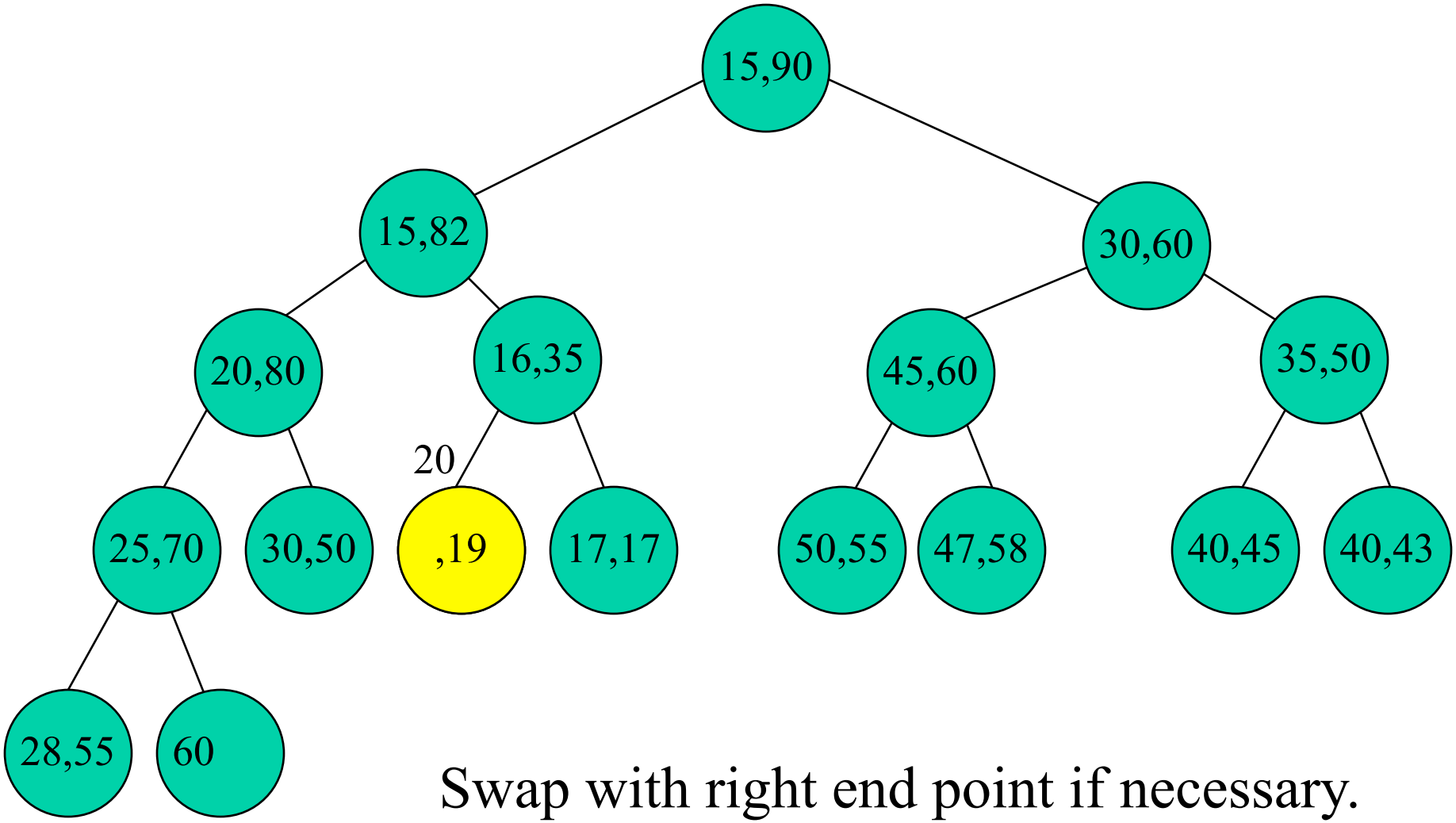

Swap with right end point if necessary.

# Remove Min Element Example



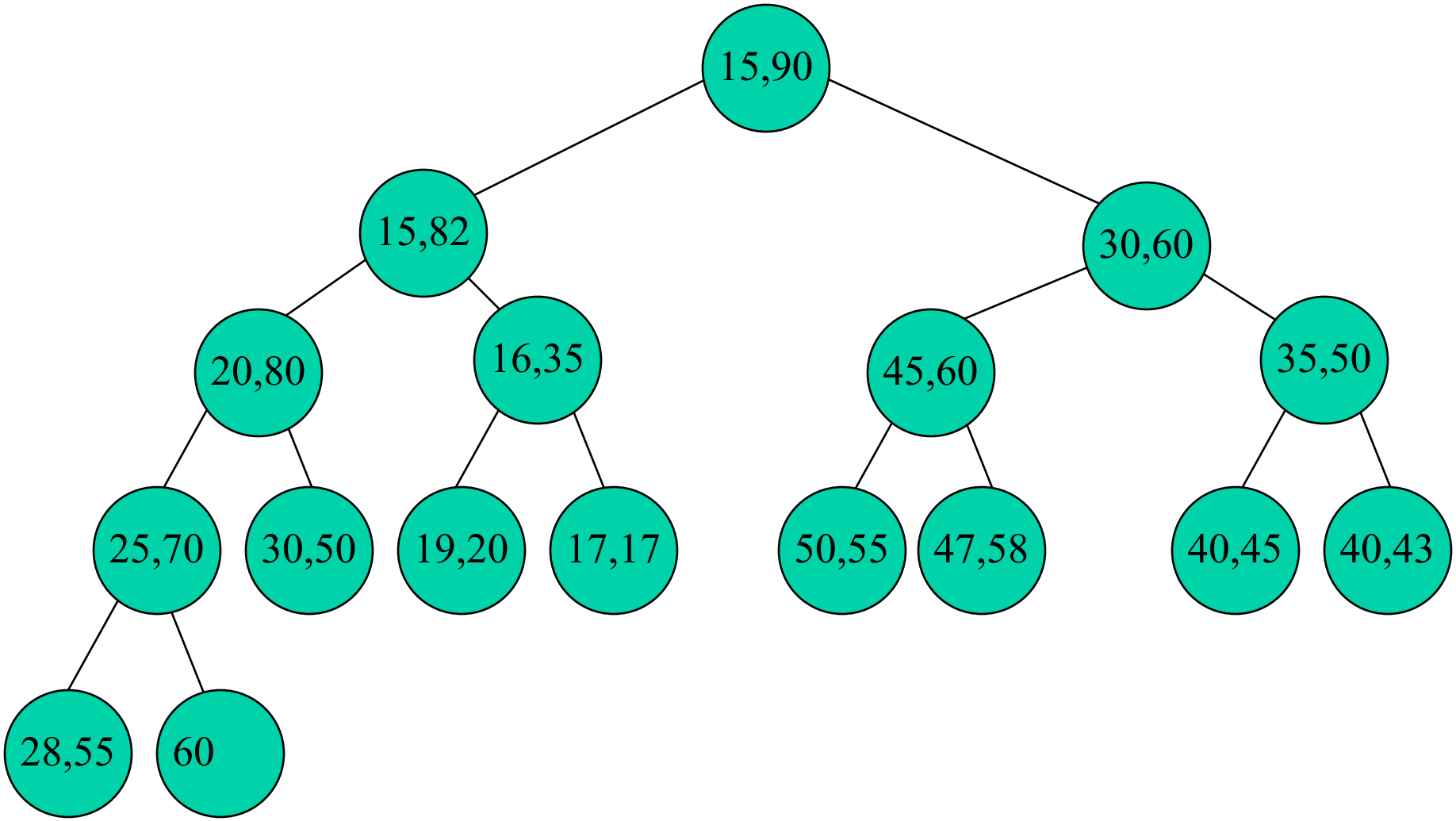

Swap with right end point if necessary.

# Remove Min Element Example

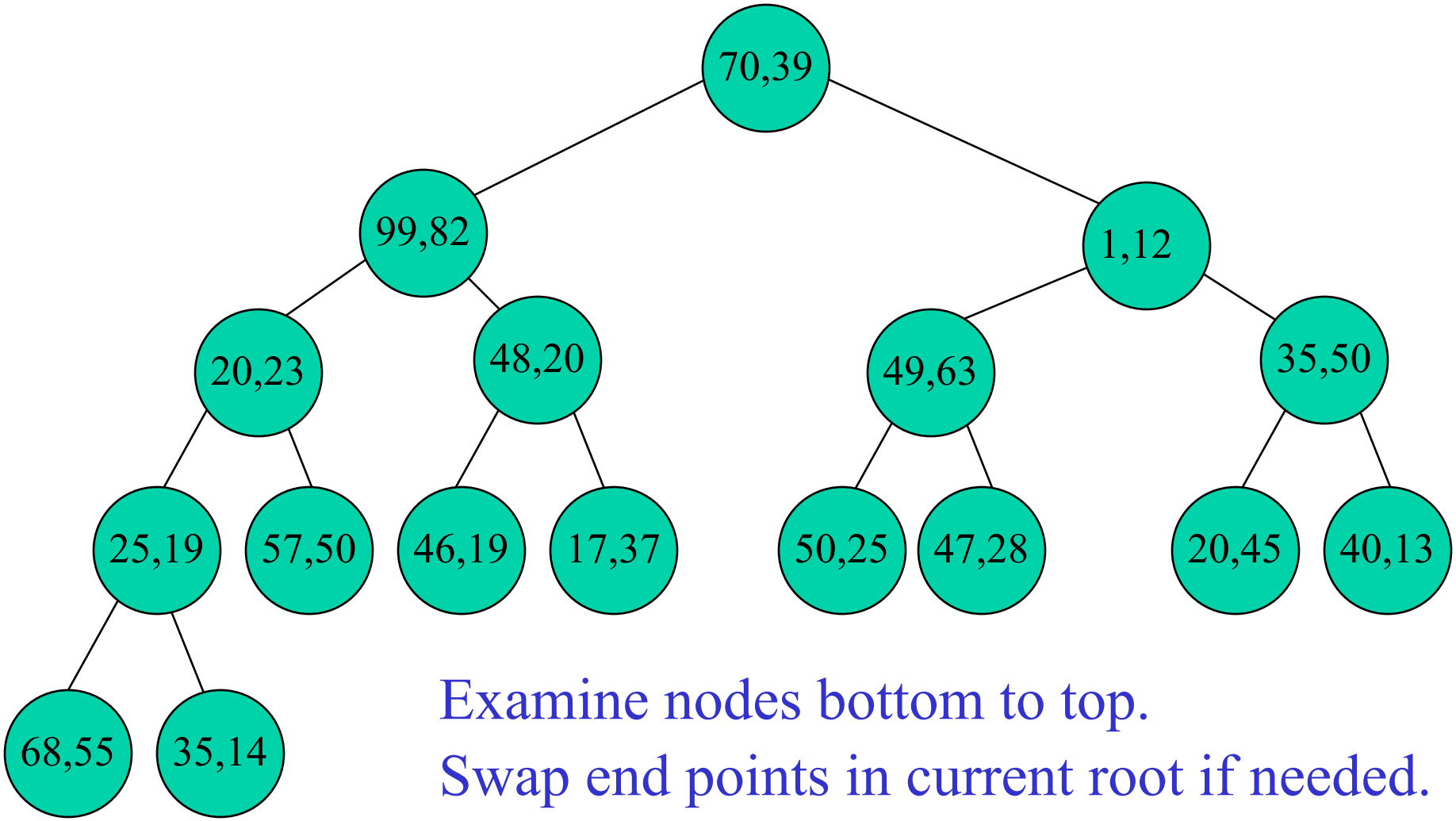



Swap with right end point if necessary.

# Remove Min Element Example



# Initialize




Examine nodes bottom to top.

Swap end points in current root if needed.

Reinsert left end point into min heap.

Reinsert right end point into max heap.