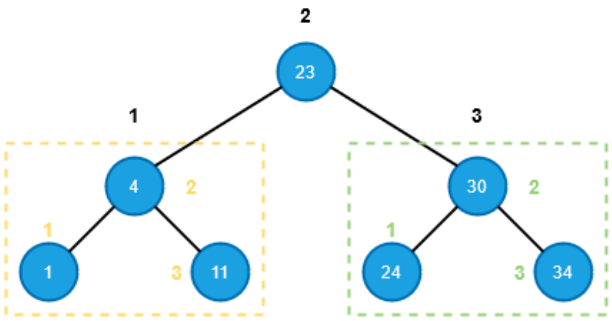
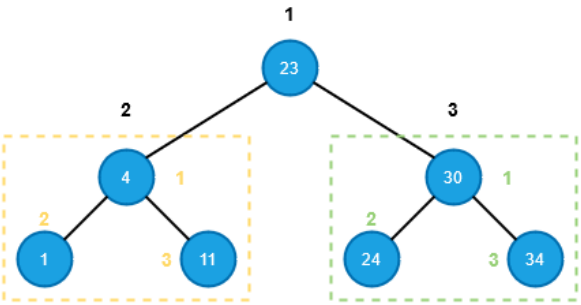
Binary Search Tree

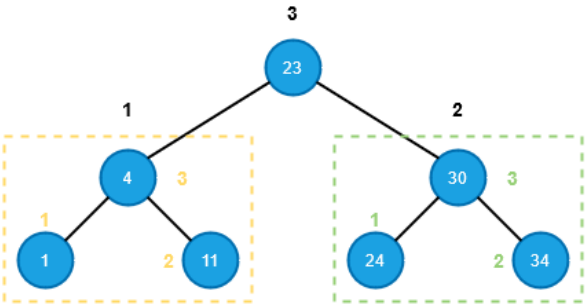
1. Tulis elemen-elemen yang ada di dalamnya (Inorder,Preorder,Postorder)



In-order output: 1, 4, 11, 23, 24, 30, 34



pre-order output: 23, 4, 1, 11, 30, 24, 34



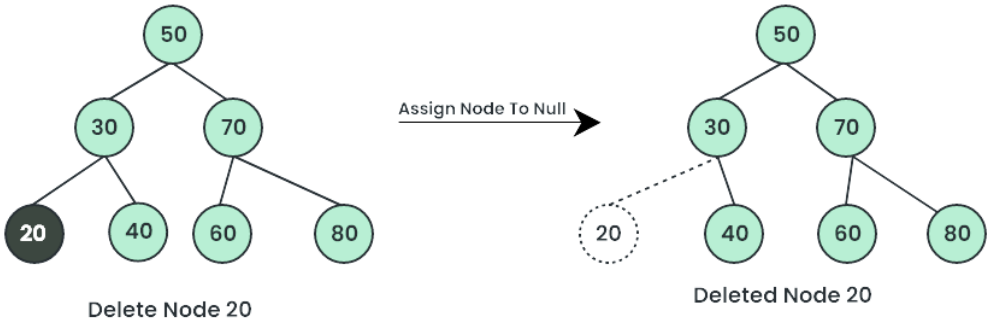
post-order output: 1, 11, 4, 24, 34, 30, 23

2. Tingginya berapa dan node-node penyusun (value).

3. Berapa leaf nodenya dan node-node penyusun (value).

4. Menghapus 2 node, dan gambarkan ulang tree.

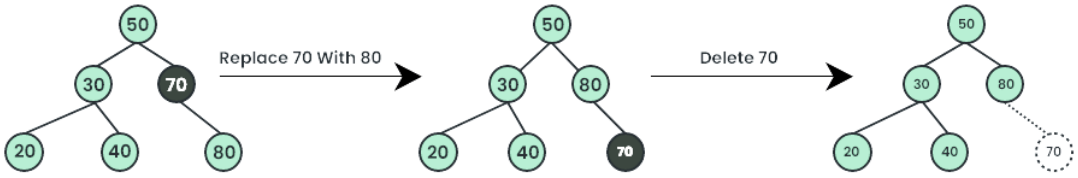
Case 1 : Delete A Leaf Node In BST



Delete Node 20

Deleted Node 20

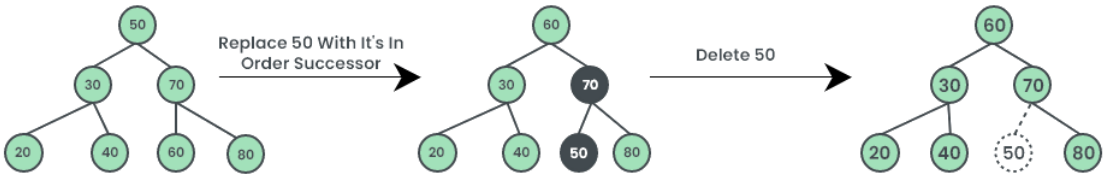
Case 2: Delete A Node With Single Child In BST



Delete Node 70

After Deletion

Case 3 : Delete A Node With Both Children In BST



Delete Node 50

After Deletion

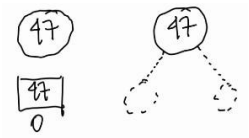
Heap

5. Eksekusi 1 input node berdasarkan algoritma

```
fungsi parent(idx)
    return ⌊(idx−1)/2⌋

fungsi leftChild(idx)
    return 2 * idx + 1

fungsi rightChild(idx)
    return 2 * idx + 2

fungsi swap(heap, idx, j)
    temp = 0
    temp = heap[idx]
    heap[idx] = heap[j]
    heap[j] = temp

fungsi heapifyDown(heap, idx)
    size = |heap|
    left = leftChild(idx)
    right = rightChild(idx)
    largest = idx

    if left < size and heap[left] > heap[largest]
       largest = left
    if right < size and heap[right] > heap[largest]
       largest = right
    if largest != idx
       swap(heap, idx, largest)
       heapifyDown(heap, largest)

fungsi heapifyUp(heap, idx)
    while (idx > 0) and (heap[idx] > heap[parent(idx)])
       parentIdx = parent(idx)
       swap(heap, idx, parentIdx)
       idx = parentIdx

fungsi insert(heap, value)
    heap[] = value
    heapifyUp(heap, |heap| - 1)

fungsi extractMax(heap)
    if len(heap) == 0
       cetak("Heap is empty. Cannot extract maximum element.")
       return None

    maxValue = heap[0]
    heap[0] = heap[-1] # -1 adalah indeks terakhir pada array
    pop(heap)          # menghapus elemen terakhir di array heap
    heapifyDown(heap, 0)

    return maxValue

fungsi printHeap(heap)
    cetak("Heap elements:", end=" ")

    for (i=0; i < size(heap); i++)
       cetak(heap[i])
```

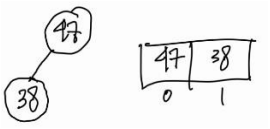
Eksekusi:

```
heap = []
insert(heap, 47)

36. fungsi insert([], 47)
37.    heap[] = 47       #heap = [47] #|heap| = 1
38.    heapifyUp([47], 0)
         30. fungsi heapifyUp([47], 0)   #heap[0] = 47
         31.    while 0 > 0 and 47 > heap[parent(0)]
                                            1. fungsi parent(0)
                                            2.   return -1
         31.    while (0 > 0) #False. Sisanya tidak perlu dieksekusi
heap = [47]
```



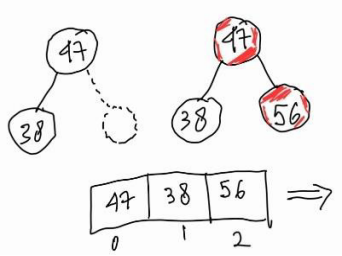
insert(heap, 38)

```
32. fungsi insert([47], 38)
33.    heap[] = 38       #|heap| = 2 #heap[47, 38]
34.    heapifyUp([47, 38], 1)
          29. fungsi heapifyUp([47, 38], 1)      #heap[1] = 38
          30.    while 1 > 0 and 38 > heap[parent(1)]
                                          1. fungsi parent(1)
                                          2. return 0
          29.    while 1 > 0 and 38 > 47 #False
heap = [47, 38]
```

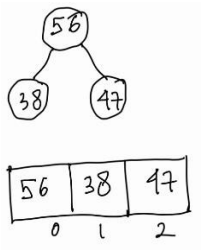


insert(heap, 56)

```
36. fungsi insert([47, 38], 56)
37.    heap[] = 56       #heap[47, 38, 56] #|heap| = 3
```

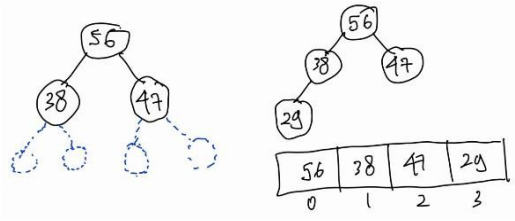



```
38.    heapifyUp([47, 38, 56], 2)
          30. fungsi heapifyUp([47, 38, 56], 2) #heap[2] = 56
          31.    while 2 > 0 and 56 > heap[parent(2)]
                                          1. fungsi parent(2)
                                          2.    return 0       #heap[0] = 47
          31.    while 2 > 0 and 56 > 47 #True
          32.       parentIdx = 0
          33.       swap([47, 38, 56], 2, 0)
                       10. fungsi swap([47, 38, 56], 2, 0) #heap[2] = 56 #heap[0] = 47
                       11.    temp = 0
                       12.    temp = 56
                       13.    heap[2] = 47       #heap = [47, 38, 47]
                       14.    heap[0] = 56       #heap = [56, 38, 47] #heap[0] = 56 #heap[2] = 47
          34.       idx = 0
          31.    while 0 > 0 #False. Sisanya tidak perlu dieksekusi
heap = [56, 38, 47]
```

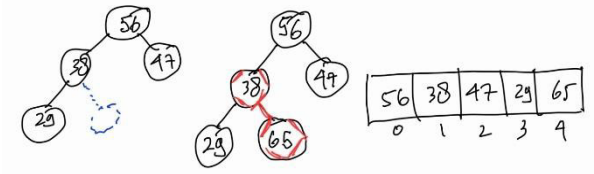



```
insert(heap, 29)
36. fungsi insert([56, 38, 47], 29)
```

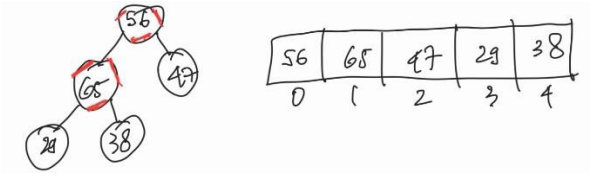
```
37.    heap[] = 29       #heap = [56, 38, 47, 29] #|heap| = 4
```




```
38.    heapifiyUp([56, 38, 47, 29], 3)
          30. fungsi heapifyUp([56, 38, 47, 29], 3)              #heap[3] = 29
          31.    while (3 > 0) and (29 > heap[parent(3)])
                                              1. fungsi parent(3)
                                              2.    return 1      #heap[1] = 38
          31.    while (3 > 0) and (29 > 38)     #False
heap = [56, 38, 47, 29]

insert(heap, 65)
36. fungsi insert([56, 38, 47, 29], 65)
37.    heap[] = 65       #heap = [56, 38, 47, 29, 65]     #|heap| = 5
```

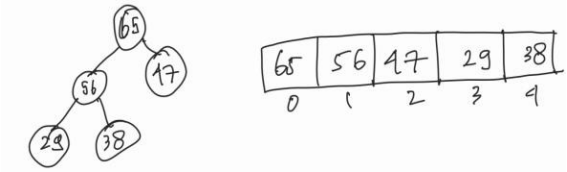



```
38.    heapifyUp([56, 38, 47, 29, 65], 4)
          30. heapifyUp([56, 38, 47, 29, 65], 4) #heap[4] = 65
          31.    while 4 > 0 and 65 > heap[parent(4)]
                                              1. fungsi parent(4)
                                              2.    return 1
          31.    while (4 > 0) and (65 > 38)     #True
          32.       parentIdx = 1
          33.       swap([56, 38, 47, 29, 65], 4, 1)
                       10. fungsi swap([56, 38, 47, 29, 65], 4, 1)
                       11.    temp = 0           #heap[4] = 65
                       12.    temp = 65          #heap[1] = 38
                       13.    heap[4] = 38       #heap = [56, 38, 47, 29, 38]
                       14.    heap[1] = 65       #heap = [56, 65, 47, 29, 38]
```

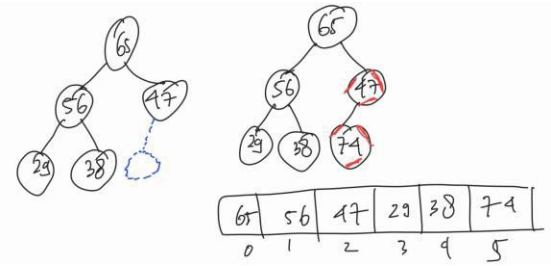



```
          34.       idx = 1      #heap[1] = 65
          31.    while (1 > 0) and (65 > heap[parent(1)])
                                              1. fungsi parent(1)
                                              2.    return 0   #heap[0] = 56
          31.    while (1 > 0) and (65 > 56)     #True
          32.       parentIdx = 0
          33.       swap([56, 65, 47, 29, 38], 1, 0)
                       10. fungsi swap([56, 65, 47, 29, 38], 1, 0)
                       11.    temp = 0           #heap[1] = 65
                       12.    temp = 65          #heap[0] = 56
                       13.    heap[1] = 56       #heap = [56, 56, 47, 29, 38]
                       14.    heap[0] = 65       #heap = [65, 56, 47, 29, 38]
          34.       idx = 0      #heap[0] = 65
          31.    while (0 > 0)   #False. Sisanya tidak perlu dieksekusi
```

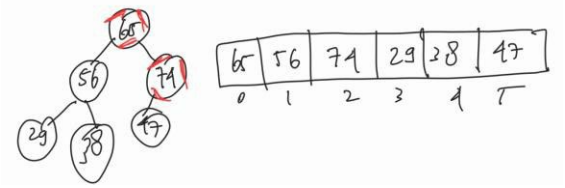
heap = [65, 56, 47, 29, 38]



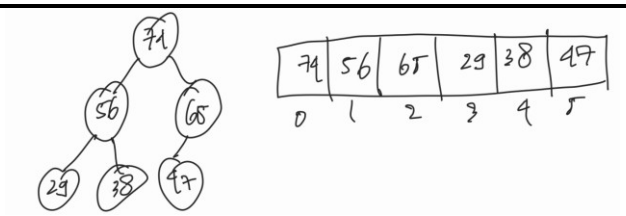

```
insert(heap, 74)
36. fungsi insert([65, 56, 47, 29, 38], 74)
37.    heap[] = 74       #heap = [65, 56, 47, 29, 38, 74] #|heap| = 6
```



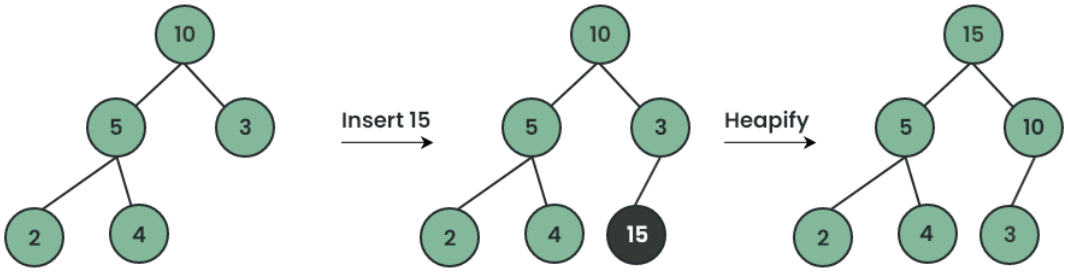

```
38.    heapifyUp([65, 56, 47, 29, 38, 74], 5)
          30. fungsi heapifyUp([65, 56, 47, 29, 38, 74], 5)       #heap[5] = 74
          31.    while (5 > 0) and (74 > heap[parent(5)])
                                                  1. fungsi parent(5)
                                                  2.    return 2   #heap[2] = 47
          31.    while (5 > 0) and (74 > 47)      #True
          32.       parentIdx = 2
          33.       swap([65, 56, 47, 29, 38, 74], 5, 2)
                       10. fungsi swap([65, 56, 47, 29, 38, 74], 5, 2)
                       11.    temp = 0           #heap[5] = 74
                       12.    temp = 74          #heap[2] = 47
                       13.    heap[5] = 47       #heap = [65, 56, 47, 29, 38, 47]
                       14.    heap[2] = 74       #heap = [65, 56, 74, 29, 38, 47]
```




```
          34.       idx = 2      #heap[2] = 74
          31.    while (2 > 0) and (74 > heap[parent(2)])
                                                  1. fungsi parent(2)
                                                  2.    return 0   #heap[0] = 65
          31.    while (2 > 0) and (74 > 65)      #True
          32.       parentIdx = 0
          33.       swap([65, 56, 74, 29, 38, 47], 2, 0)
                       10. fungsi swap([65, 56, 74, 29, 38, 47], 2, 0)
                       11.    temp = 0           #heap[2] = 74
                       12.    temp = 74          #heap[0] = 65
                       13.    heap[2] = 65       #heap = [65, 56, 65, 29, 38, 47]
                       14.    heap[0] = 74       #heap = [74, 56, 65, 29, 38, 47]
          34.       idx = 0
          31.    while (0 > 0)   #False. Sisanya tidak perlu dieksekusi
```



6. Gambarkan heap setelah dimasukkan 1 input node



7. Gambarkan setelah hapus heap

Suppose the Heap is a Max-Heap as:

```
    10
  /   \
 5     3
/ \
2  4
```

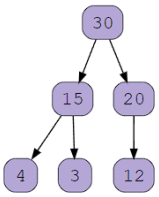
The element to be deleted is root, i.e. 10.
Process:
The last element is 4.
Step 1: Replace the last element with root, and delete it.

```
    4
  /   \
 5     3
/
2
```

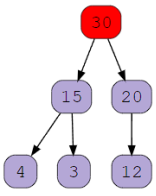
Step 2: Heapify root.
Final Heap:

```
    5
  /   \
 4     3
/
2
```

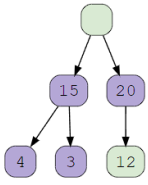
Consider this max heap



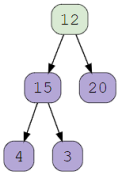
Let's delete root node



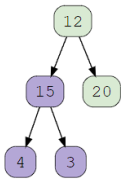
Replace last node with deleted node



This is not max heap. Let's heapify it



Swap 12 and 20



Max heap is: