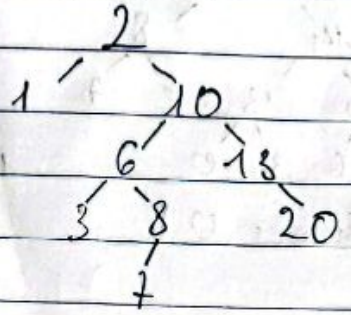
Memo No.
Date / /

Bài 2:

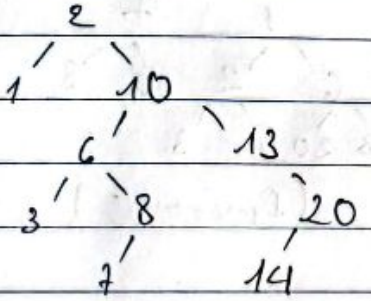
List of integer number: 2, 1, 10, 6, 3, 8, 7, 13, 20

- Binary search tree:

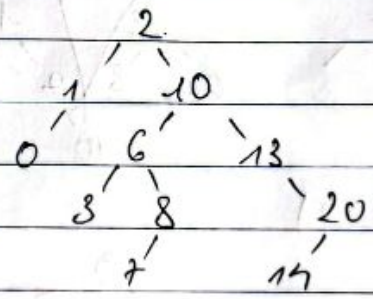


- Insert 14, 0, 35:

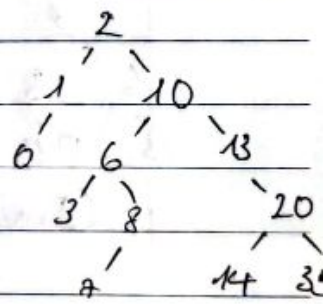
+ Insert 14:



+ Insert 0

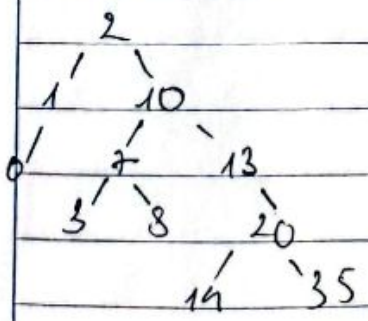


+ Insert 35:

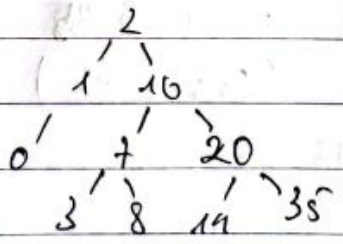


- Delete 6, 13, 35:

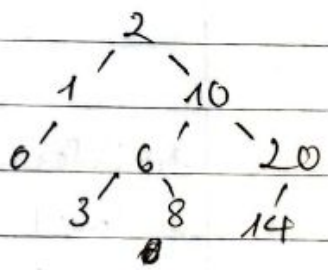
+ Delete 6:



+ Delete 13:



+ Delete 35:

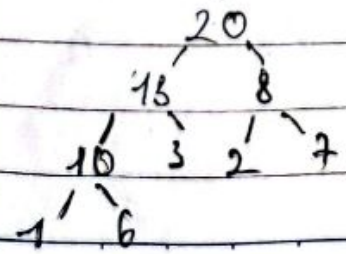


Bài 3:

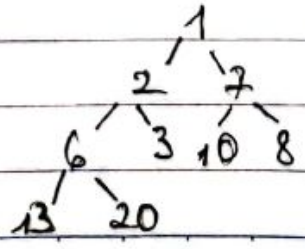
List of integer number: 2, 1, 10, 6, 3, 8, 7, 13, 20

- Heap tree

+ Max heap:



+ Min heap:

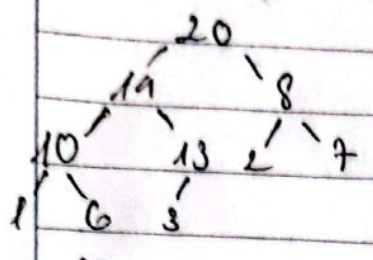


HẢI TIẾN

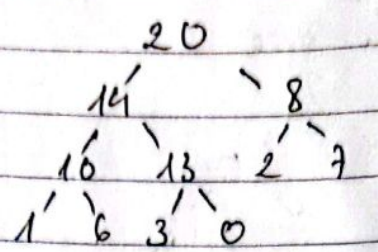
Memo No.
Date / /

- Insert 14, 0, 5:

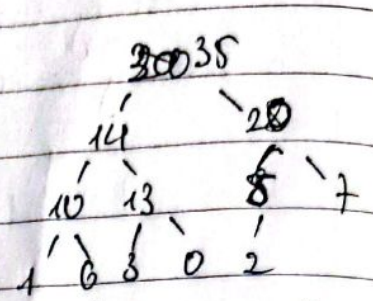
+ Max heap:



(Insert 14)

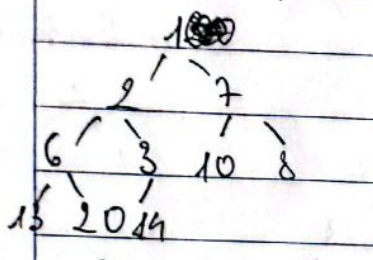


(Insert 0)

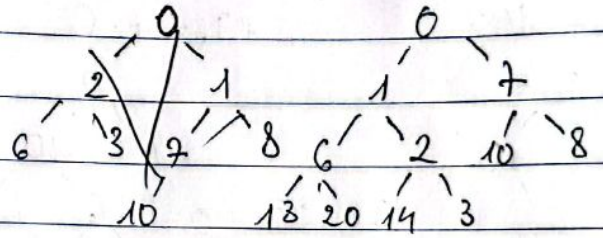


(Insert 5)

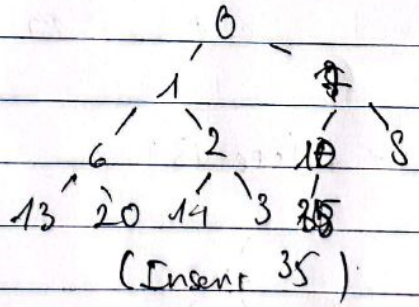
+ Min heap



(Insert 14)



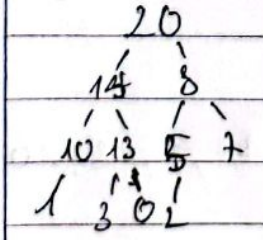
(Insert 0)

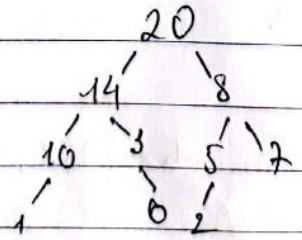


(Insert 35)

- Delete 6, 13, 35:

+ Max heap



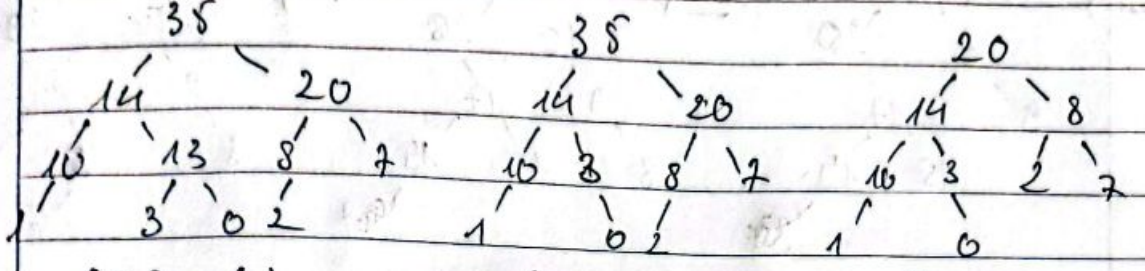


HẢI TIẾN

Memo No.
Date / /

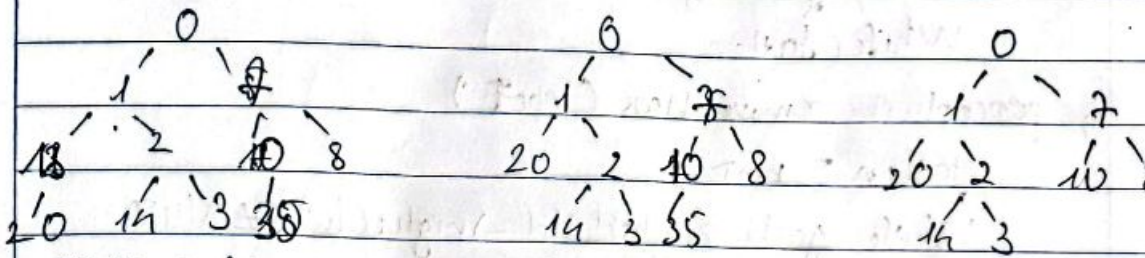
- Delete 6, 13, 35:

+ Max heap:



(Delete 6) (Delete 13) (Delete 35)

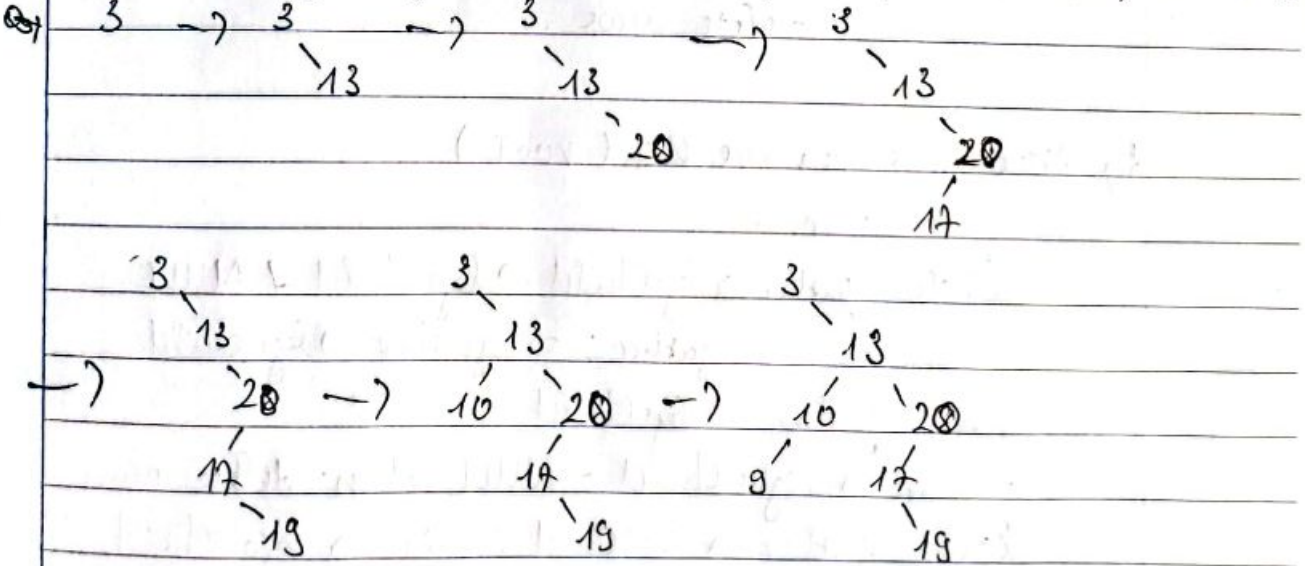
+ Min heap:



(Delete 6) (Delete 13) (Delete 35)

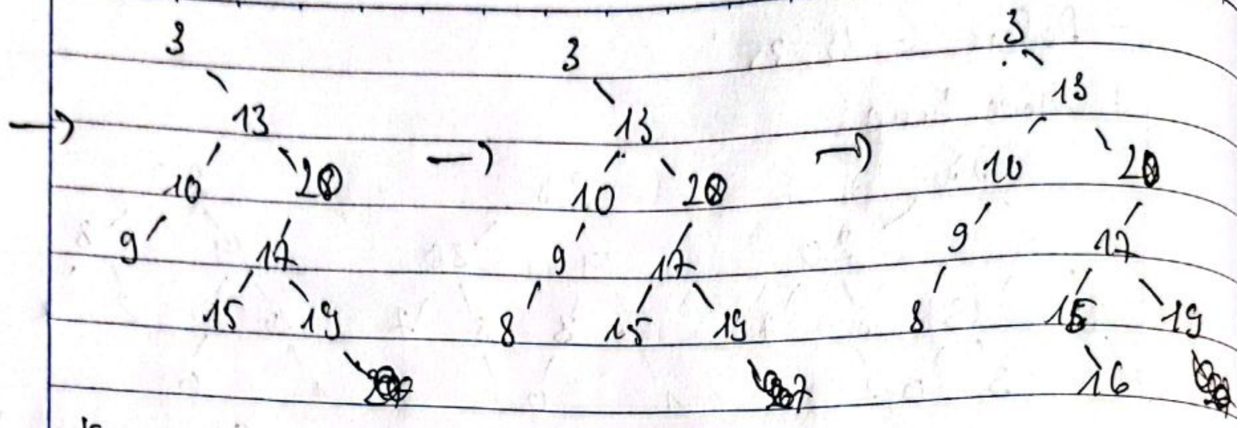
Bài 4:

List of integer number: 3, 13, 20, 17, 19, 10, 9, 27, 8, 16 (15)



HẢI TIẾN

Memo No.
Date / /



b,

~~Void deleteMax (Node* root) {~~
~~Node* fatherOfDeletedNode = root ;~~
~~While (father~~

+, procedure removeMax (root)
    father = root
    While father->rightChild->rightChild ≠ NULL
            father := father->rightChild
    max = father -> rightChild
    if max -> leftChild = NULL then delete max
    else father -> rightChild = max -> leftChild
            delete max .

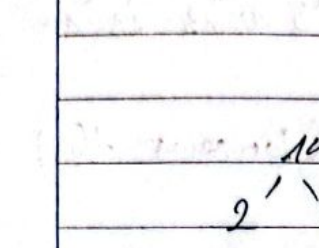
+, procedure removeMin(root)
    father = root
    While father-> leftChild -> leftChild ≠ NULL
            father := father -> left child
    min = father -> leftChild
    if min -> rightChild = NULL then delete min
    else father -> leftChild = min -> rightChild
            delete min

Memo No.
Date / /

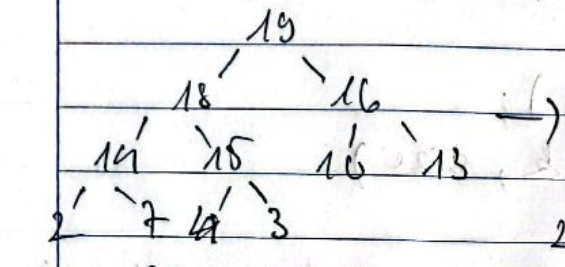
Bai 5:

List of integer number: 16, 2, 10, 7, 4, 13, 19, 14, 18, 15

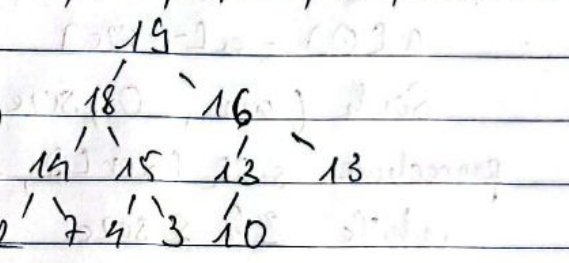
- Heap tree from S2:



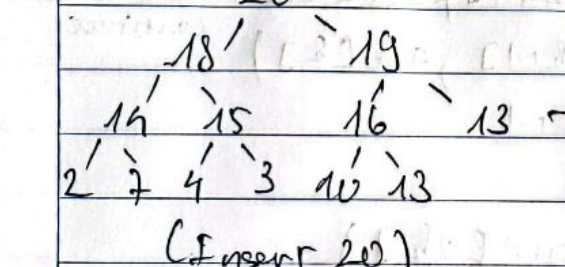
- S1: 3, 13, 20, 17, 19, 10, 9, 15, 8, 16



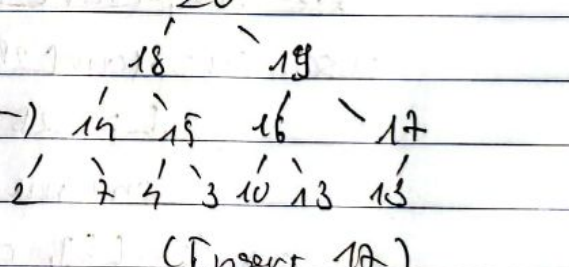
(Insert 3)



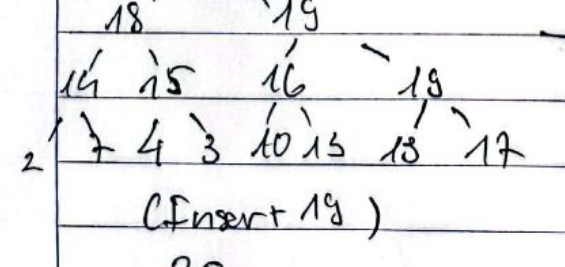
(Insert 13)



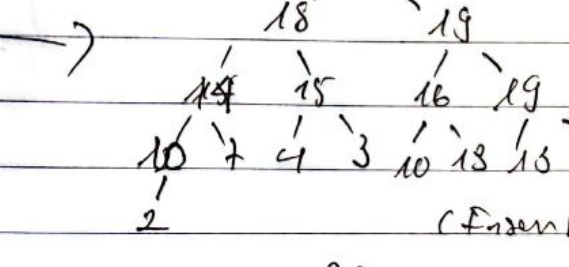
(Insert 20)



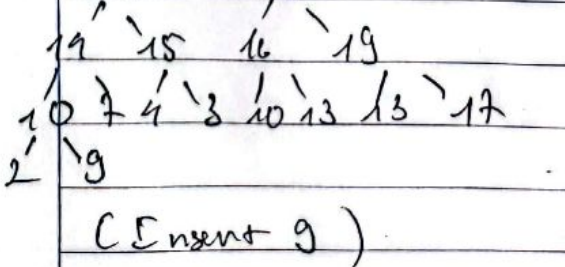
(Insert 17)



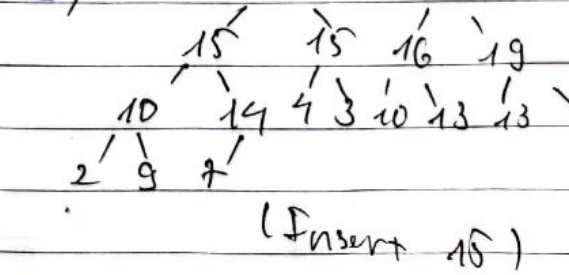
(Insert 19)



(Insert 10)



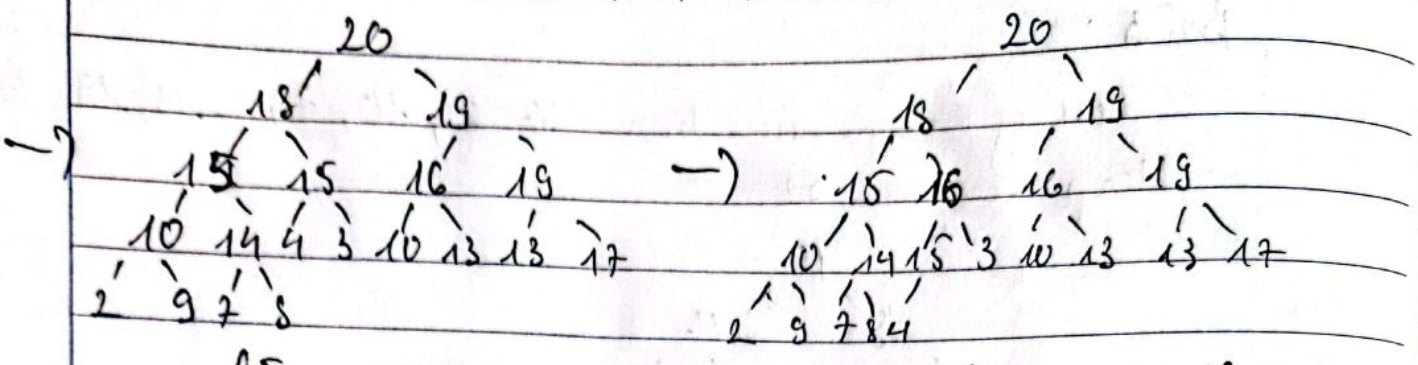
(Insert 9)



(Insert 15)

HẢI TIẾN

Memo No.
Date / /



(Insert 8) (Insert 16)

- procedure removeMax(arr[], &size)
  a[0] = a[--size]
  Sink(arr, 0, size);
  procedure sink(arr[], k, size)
  while 2k ≤ size
  if arr[2k] > arr[2k+1] then swap(arr[k], arr[2k])
  ~~else swap(arr[2k+1], arr[k])~~ k := 2k
  continue
  else swap(arr[2k+1], arr[k])
  k := 2k+1
  continue
  swap(arr[k], arr[2k])
  k := 2k