PROBLEMS USING HEAPS

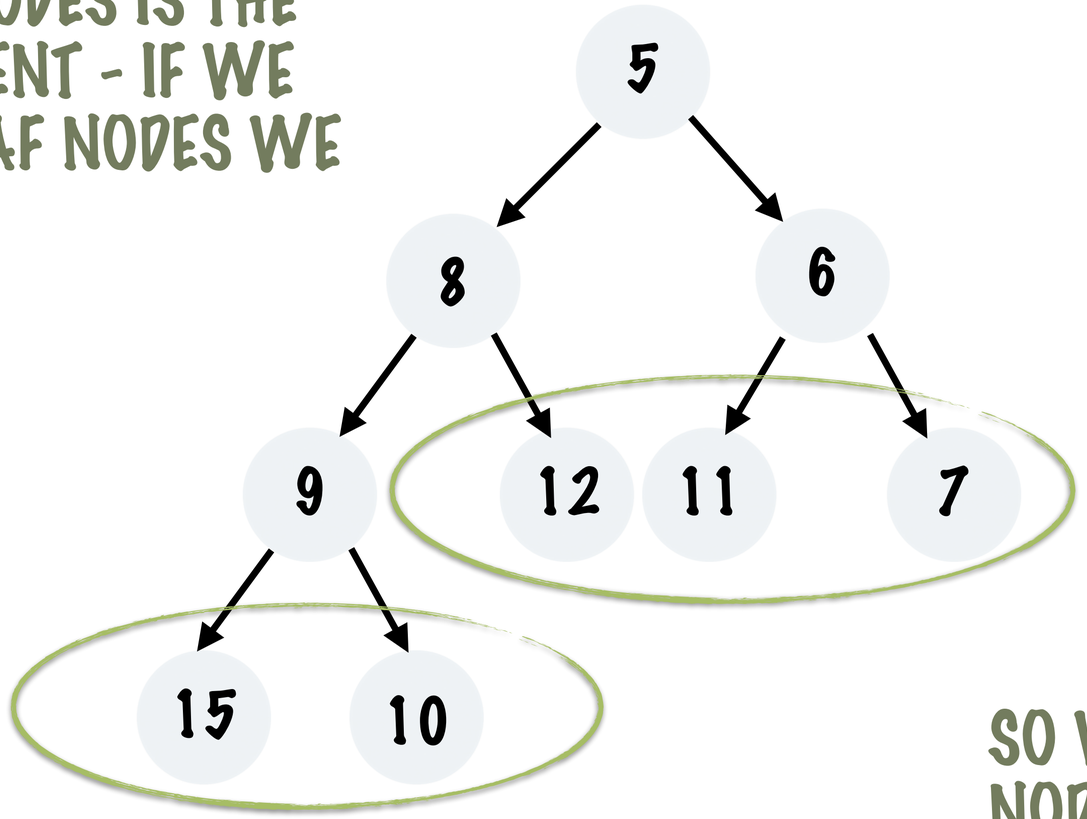
PROBLEMS USING HEAPS

LET'S PRACTICE USING HEAPS
TO SOLVE REAL PROBLEMS!

FIND THE MAXIMUM ELEMENT IN A MINIMUM HEAP

FIND THE MAXIMUM ELEMENT IN A MINIMUM HEAP

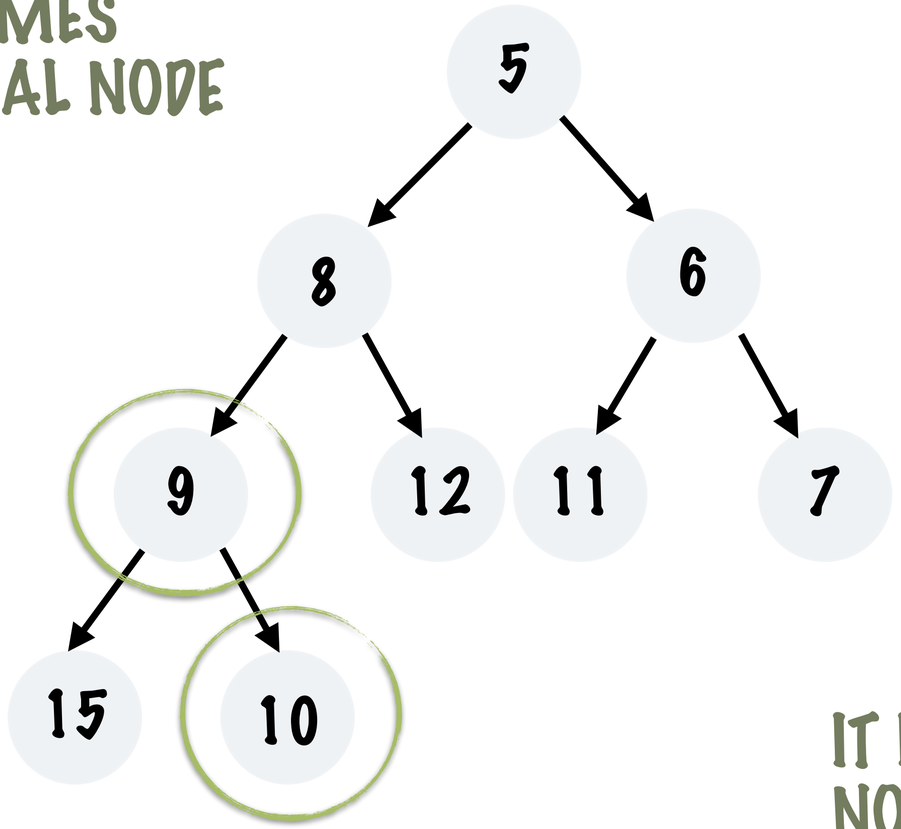
ONE OF THE LEAF NODES IS THE MAXIMUM ELEMENT - IF WE ONLY SCAN THE LEAF NODES WE WILL FIND IT



SO WHICH IS THE FIRST LEAF NODE?

FIND THE MAXIMUM ELEMENT IN A MINIMUM HEAP

THE FIRST LEAF NODE COMES AFTER THE LAST INTERNAL NODE



IT IS THE PARENT OF THE LAST NODE IN THE HEAP

GET MAX ELEMENT IN MIN HEAP

```
public static int getMaximum(MinHeap<Integer> minHeap) {
    int lastIndex = minHeap.getCount() - 1;
    int lastParentIndex = minHeap.getParentIndex(lastIndex);

    int firstChildIndex = lastParentIndex + 1;

    int maxElement = minHeap.getElementAtIndex(firstChildIndex);
    for (int i = firstChildIndex; i <= lastIndex; i++) {
        if (maxElement < minHeap.getElementAtIndex(i)) {
            maxElement = minHeap.getElementAtIndex(i);
        }
    }

    return maxElement;
}
```

GET THE LAST LEAF NODE IN THE HEAP - PRESENT AT THE LAST INDEX OF THE ARRAY

FIND THE PARENT OF THE VERY LAST INDEX, THIS IS THE LAST INTERNAL NODE

ITERATE THROUGH ALL THE LEAF NODES STARTING AT THE INDEX AFTER THE INDEX OF THE LAST PARENT NODE

RETURN THE MAXIMUM ELEMENT - THIS IS NOW A SIMPLE SCAN

FIND THE K LARGEST ELEMENTS IN A STREAM

FIND THE K LARGEST ELEMENTS IN A STREAM

USE A MINIMUM HEAP WITH SIZE K TO STORE ELEMENTS AS THEY COME IN

THE TOP OF THE HEAP WILL HAVE THE SMALLEST OF THE K ELEMENTS STORED IN THE HEAP

IF THE NEW ELEMENT IN THE STREAM IS LARGER THAN THE MINIMUM - ADD IT TO THE HEAP

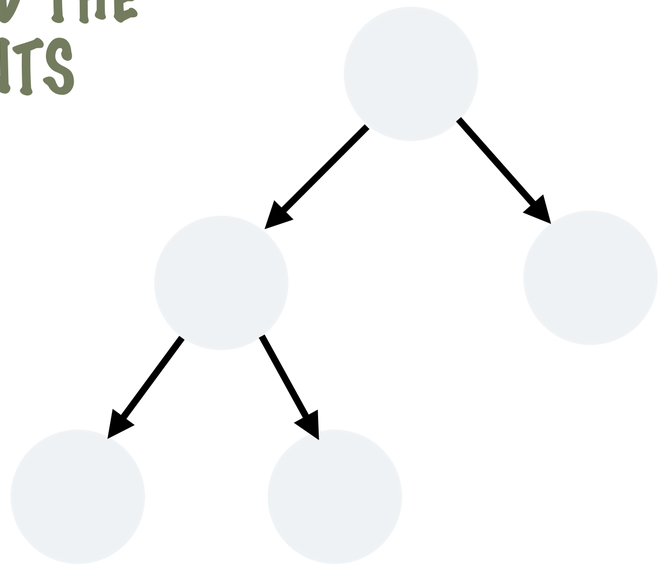
THE REMAINING ELEMENTS CAN BE IGNORED

THE HEAP WILL ALWAYS HAVE THE LARGEST K ELEMENTS FROM THE STREAM

FIND THE K LARGEST ELEMENTS IN A STREAM

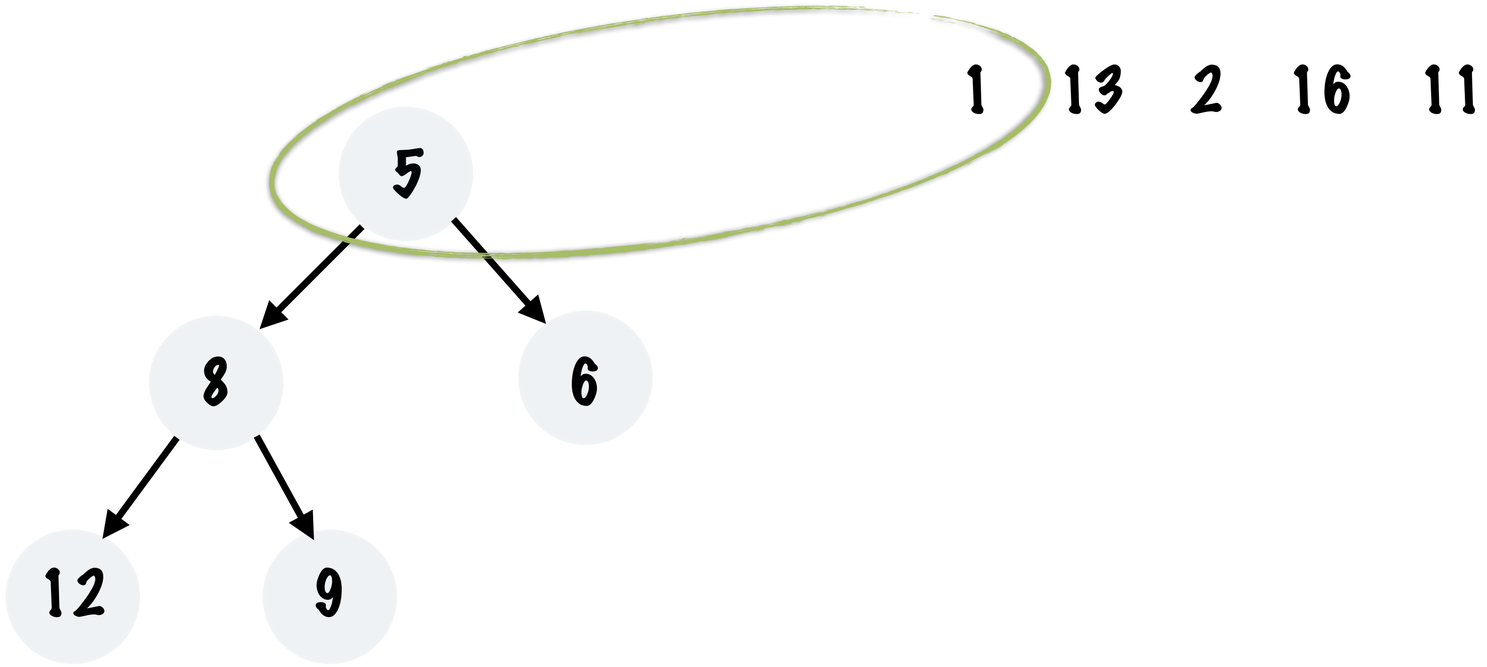
SAY WE WANT TO FIND THE FIVE LARGEST ELEMENTS

5 8 6 12 9



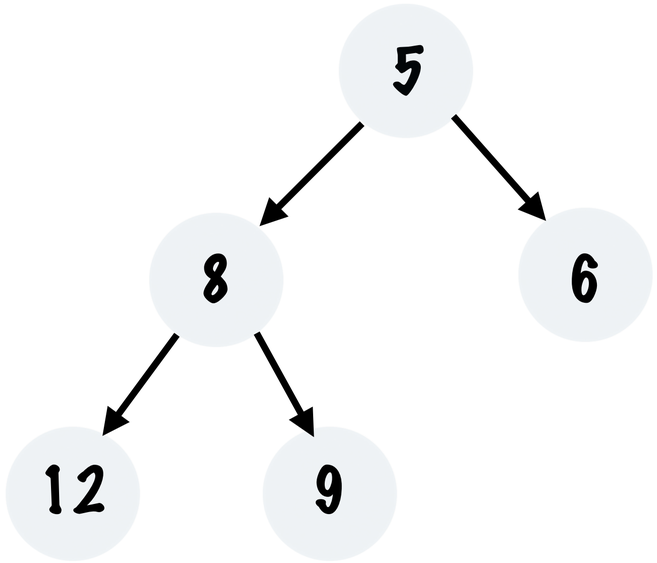
THE FIRST 5 ELEMENTS IN THE STREAM ARE JUST ADDED TO THE HEAP

FIND THE K LARGEST ELEMENTS IN A STREAM

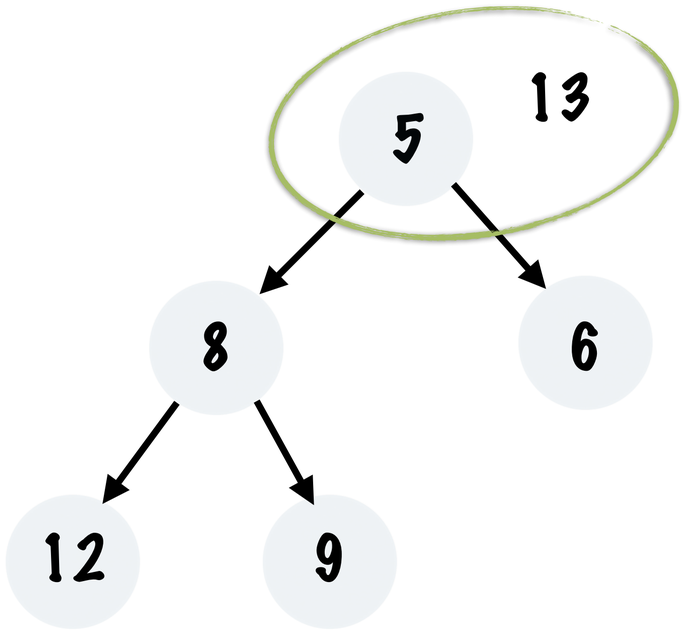


FIND THE K LARGEST ELEMENTS IN A STREAM

13 2 16 11



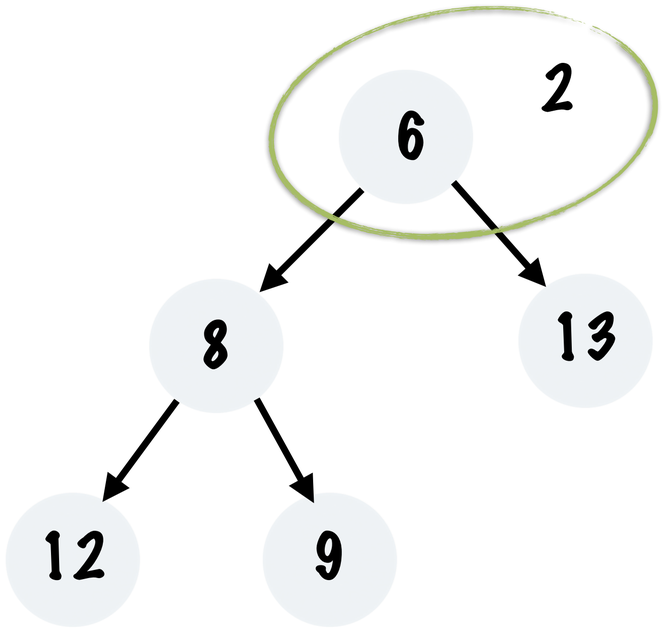
FIND THE K LARGEST ELEMENTS IN A STREAM



2 16 11

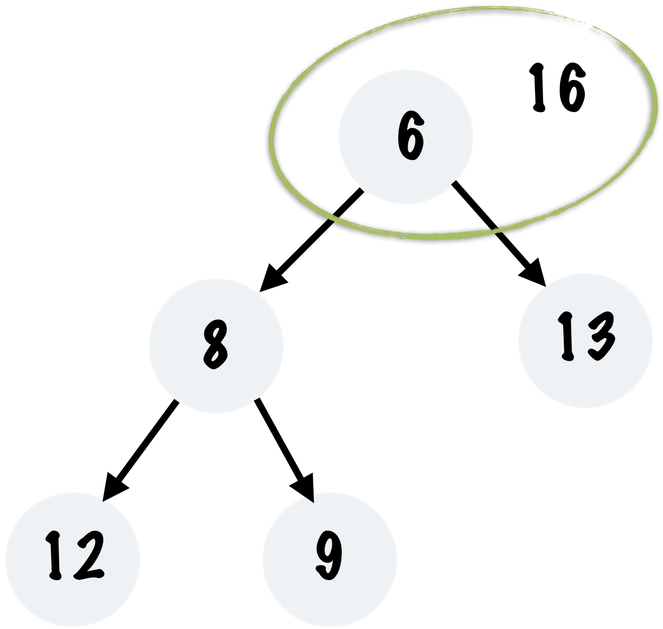
FIND THE K LARGEST ELEMENTS IN A STREAM

5



16 11

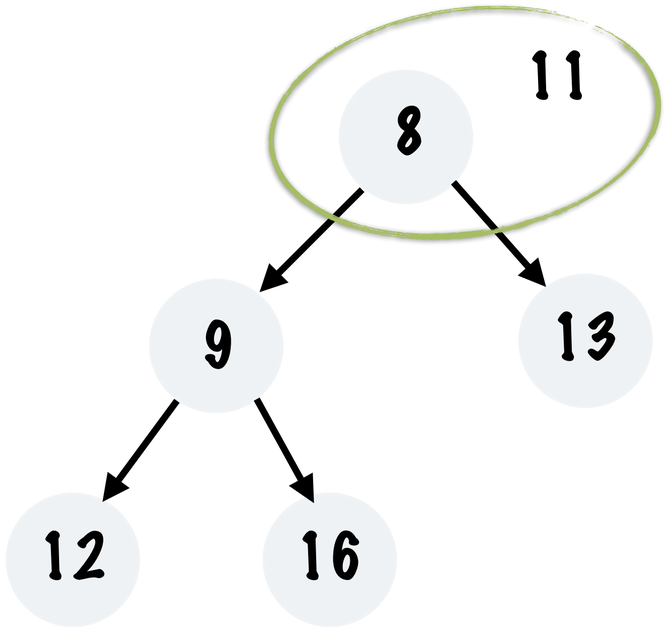
FIND THE K LARGEST ELEMENTS IN A STREAM



11

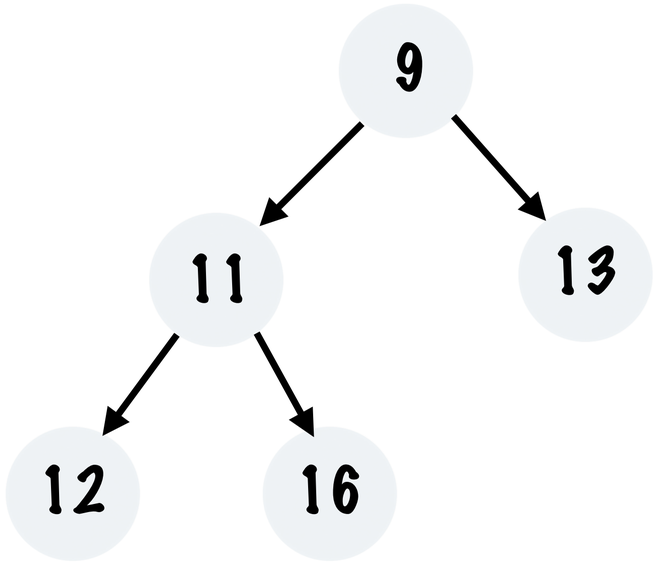
FIND THE K LARGEST ELEMENTS IN A STREAM

6



FIND THE K LARGEST ELEMENTS IN A STREAM

8



FIND THE K LARGEST ELEMENTS

```
public static void printMaximumKElements(int k)
        throws MinHeap.HeapEmptyException, MinHeap.HeapFullException {
    MinHeap<Integer> minHeap = new MinHeap<>(Integer.class, k);

    for (int number : randomNumberArray) {
        if (minHeap.isEmpty()) {
            minHeap.insert(number);
        } else if (!minHeap.isFull() || minHeap.getHighestPriority() < number) {
            if (minHeap.isFull()) {
                minHeap.removeHighestPriority();
            }
            minHeap.insert(number);
        }
    }

    minHeap.printHeapArray();
}
```

SPECIFY K AS AN ARGUMENT TO THIS FUNCTION

SET UP THE MIN HEAP WHICH WILL HOLD THE LARGEST K ELEMENTS, K IS THE CAPACITY OF THE HEAP

INSTEAD OF A STREAM JUST ITERATE THROUGH AN ARRAY WITH RANDOM NUMBERS

IF THE HEAP IS ALREADY FULL REMOVE THE SMALLEST ELEMENT FROM THE HEAP

IF THE HEAP IS NOT FULL OR THE INCOMING ELEMENT IS GREATER THAN THE K SMALLEST ELEMENT THEN ADD THIS ELEMENT TO THE HEAP