

Heapsort

PARENT(i)
 return $\lfloor i/2 \rfloor$

LEFT(i)
 return $2i$

RIGHT(i)
 return $2i + 1$

Max-heap property is $A[\text{PARENT}(i)] \geq A[i]$ for every node i other than the root.

MAX-HEAPIFY(A, i)
 $l \leftarrow \text{LEFT}(i)$
 $r \leftarrow \text{RIGHT}(i)$
if $l \leq \text{heap-size}[A]$ and $A[l] > A[i]$
 then $\text{largest} \leftarrow l$
 else $\text{largest} \leftarrow i$
if $r \leq \text{heap-size}[A]$ and $A[r] > A[\text{largest}]$
 then $\text{largest} \leftarrow r$
if $\text{largest} \neq i$
 then exchange $A[i] \leftrightarrow A[\text{largest}]$
 MAX-HEAPIFY($A, \text{largest}$)

```

BUILD-MAX-HEAP( $A$ )
   $heap-size[A] \leftarrow length[A]$ 
  for  $i \leftarrow \lfloor length[A]/2 \rfloor$  downto 1
    do MAX-HEAPIFY( $A, i$ )

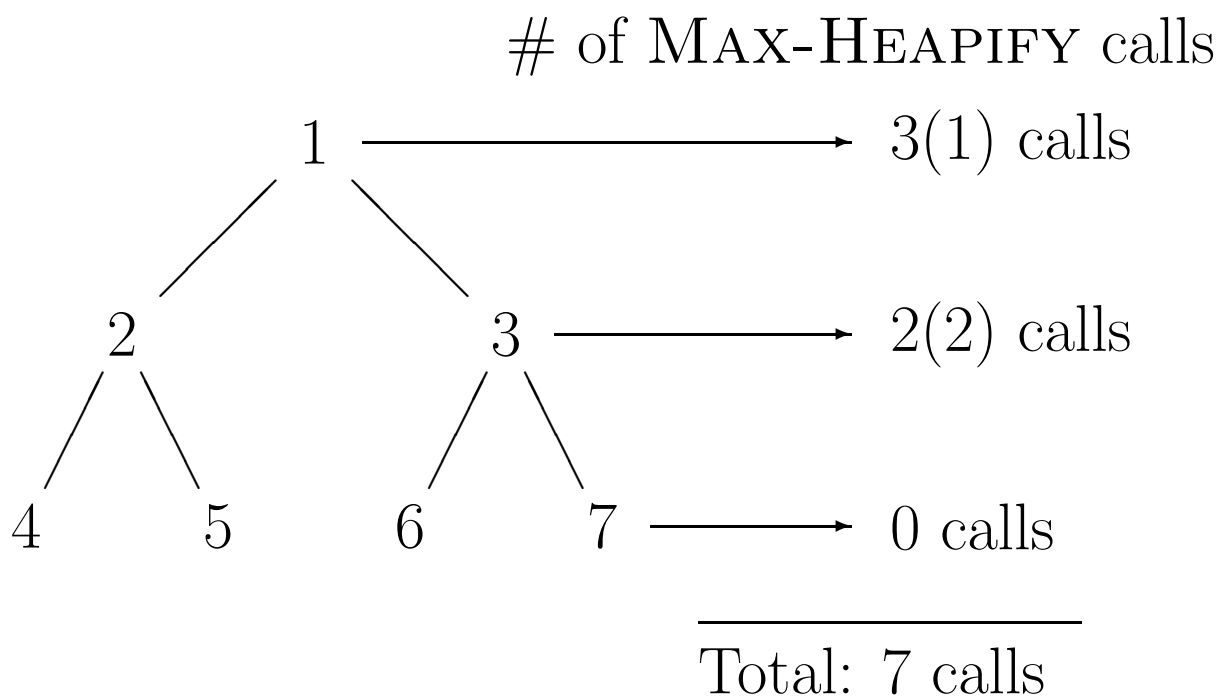
```

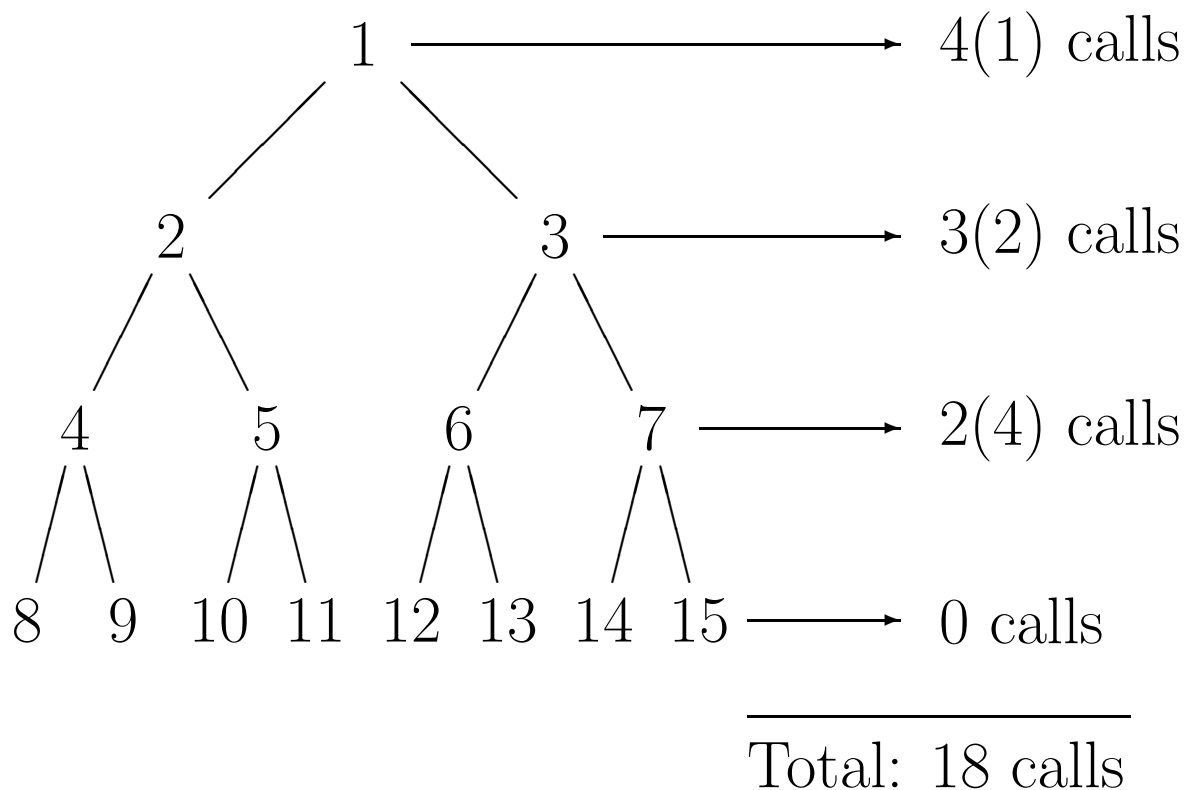
```

HEAPSORT( $A$ )
  BUILD-MAX-HEAP( $A$ )
  for  $i \leftarrow length[A]$  downto 2
    do exchange  $A[1] \leftrightarrow A[i]$ 
       $heap-size[A] \leftarrow heap-size[A] - 1$ 
      MAX-HEAPIFY( $A, 1$ )

```

BUILD-MAX-HEAP Behavior





```

HEAP-MAXIMUM( $A$ )
  return  $A[1]$ 

```

```

HEAP-EXTRACT-MAX( $A$ )
  if  $heap-size[A] < 1$ 
    then error "heap underflow"
   $max \leftarrow A[1]$ 
   $A[1] \leftarrow A[heap-size[A]]$ 
   $heap-size[A] \leftarrow heap-size[A] - 1$ 
  MAX-HEAPIFY( $A, 1$ )
  return  $max$ 

```

HEAP-INCREASE-KEY(A, i, key)

if $key < A[i]$

then error “new key is smaller than current key”

$A[i] \leftarrow key$

while $i > 1$ and $A[\text{PARENT}(i)] < key$

do exchange $A[i] \leftrightarrow A[\text{PARENT}(i)]$

$i \leftarrow \text{PARENT}(i)$

MAX-HEAP-INSERT(A, key)

$heap\text{-}size[A] \leftarrow heap\text{-}size[A] + 1$

$A[heap\text{-}size[A]] \leftarrow -\infty$

HEAP-INCREASE-KEY($A, heap\text{-}size[A], key$)