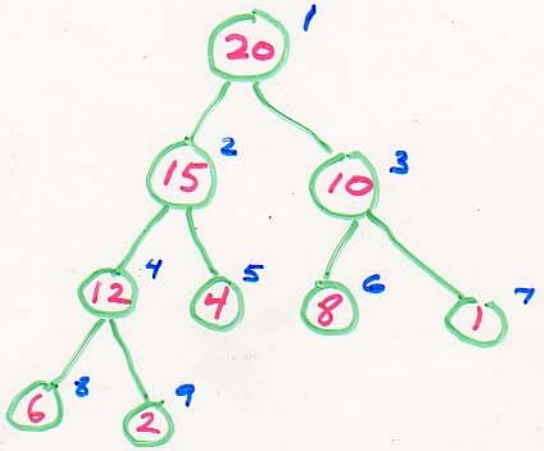
CS 223 Lec 2

Heaps

- useful for sorting
- and for implementing priority queues

heaps are binary trees

every node has a key value.



max-heaps

(min-heaps)

Max Heap Property

$\forall$ nodes x in the heap:

$$\text{Key}(\text{parent}(x)) \geq \text{Key}(x)$$

Min Heap Property

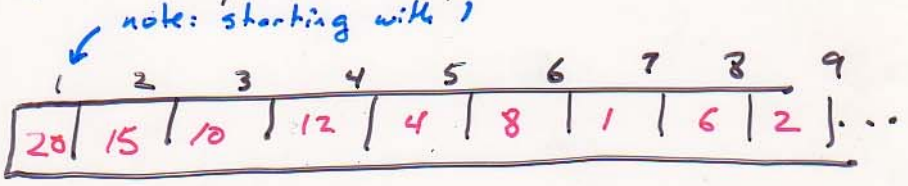
$\forall$ nodes x in the heap:

$$Key(parent(x)) \leq Key(x)$$

Storing Heaps in Arrays

example heap:

A:



$parent(i) = \lfloor i/2 \rfloor$ (floor notation)

$left(i) = 2i$

$right(i) = 2i + 1$

Heap Properties

max heap: $A[parent(i)] \geq A[i]$
for all $i \geq 1$.

min heap: $\forall i \geq 1$, $A[\text{parent}(i)] \leq A[i]$

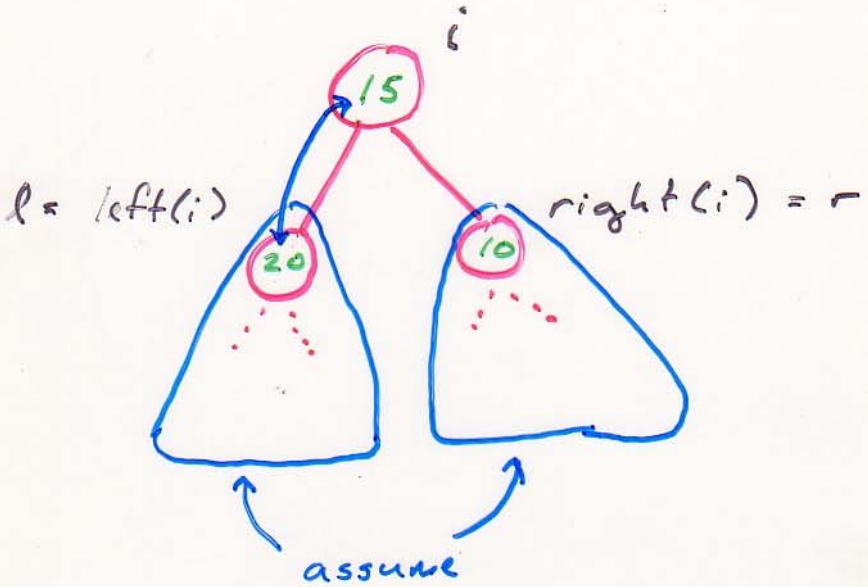
Building a Heap

- assume heap is stored in array A

BUILD-MAX-HEAP

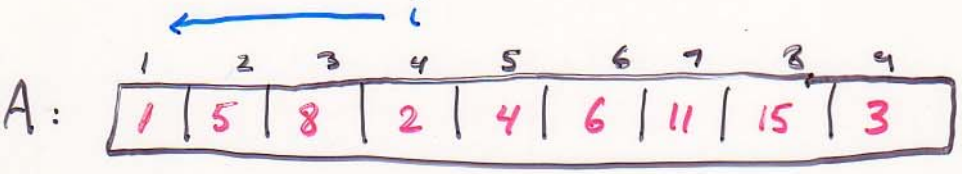
1. heap-size(A) ← length(A)
2. for $i \leftarrow \lfloor \text{length}(A)/2 \rfloor$ down to 1
3. do MAX-HEAPIFY(A, i)

MAX-HEAPIFY (A, i)




```
1.  l ← left(i)
2.  r ← right(i)
3.  if l ≤ heap-size(A) and A(l) > A(i)
4.  then largest ← l
5.  else largest ← i
6.  if r ≤ heap-size(A) and A(r) > A(largest)
7.  then largest ← r
8.  if largest ≠ i
9.  then { exchange A(i) ↔ A(largest)
10.  MAX-HEAPIFY(A, largest) }
```

Example of building a heap



run BUILD-MAX-HEAP...

...

MAX-HEAPIFY (A, 4) ✓

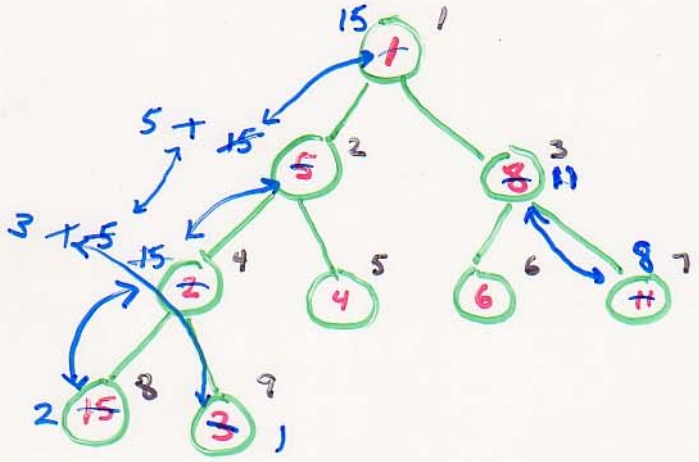
" (A, 3) ✓

" (A, 2) → M-H(A, 4) ✓

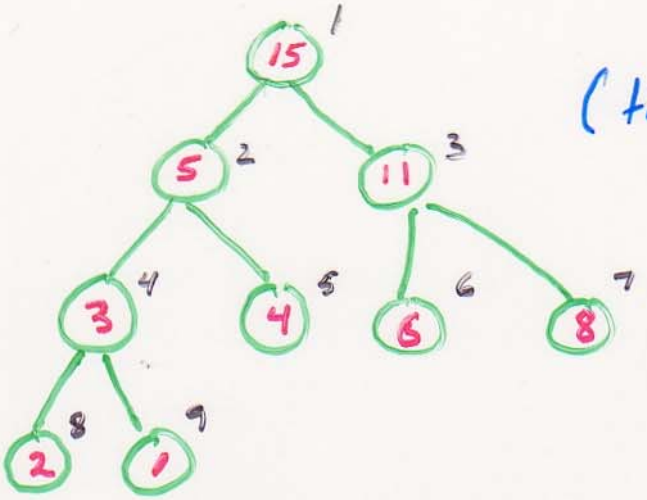
" (A, 1) ✓

→ M-H(A,2) → M-H(A,4)

MAX-HEAPIFY (A, 4)



... afterwards we have:



(this is a heap)

A: | 15 | 5 | 11 | 3 | 4 | 6 | 8 | 2 | 1 |

Heapsort

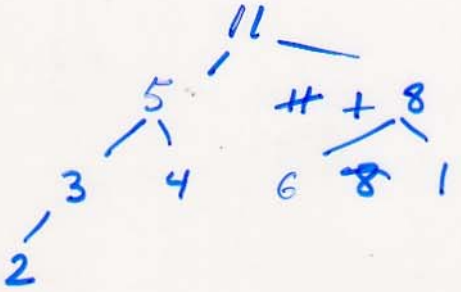
HEAP-SORT(A)

1. BUILD-MAX-HEAP(A)
2. for $i \leftarrow$ length(A) down to 2
3. do { exchange $A(1) \leftrightarrow A(i)$
4. heap-size(A) $\leftarrow$ heap-size(A) -1
5. MAX-HEAPIFY(A, 1) }

running heap-sort on previous example...

$i = 9$

A | 1 | 5 | 11 | 3 | 4 | 6 | 8 | 2 | 15



$i = 8$

A | 2 | 5 | 8 | 3 | 4 | 6 | 1 | 11 | 15

⋮

$i = 2$

A | 1 | 2 | 3 | 4 | 5 | 6 | 8 | 11 | 15