

Heaps Analysis

CS 225 - Fall 2026

Mattox Beckman

Objectives

Demonstrate the time complexity of insert and delete

Implement the heapify function

Implement heap sort

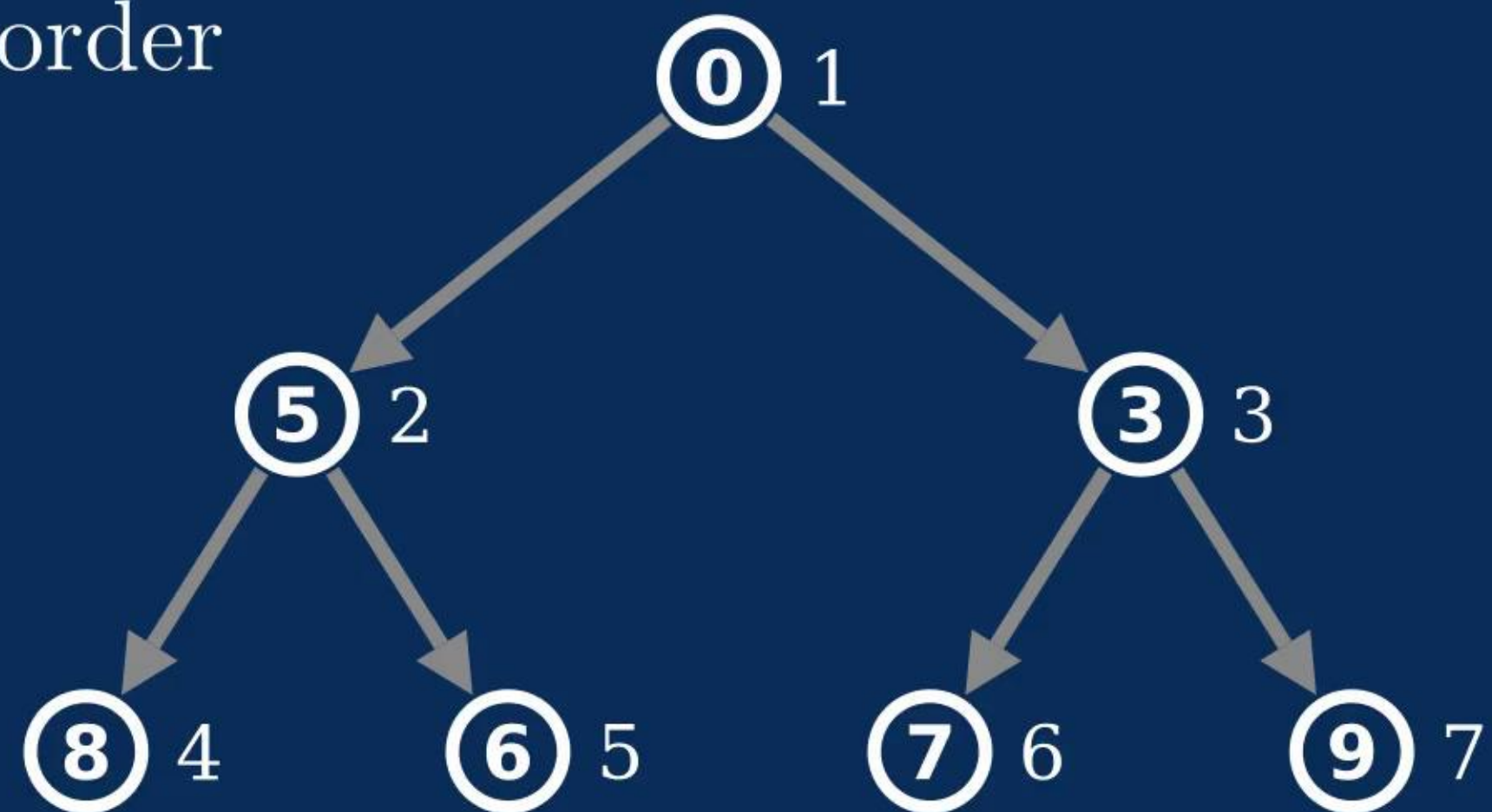
Nodes are placed in complete tree order

- Filled level by level
- Filled from left to right

Node priority higher than children

Nodes are numbered

We use an array for storage

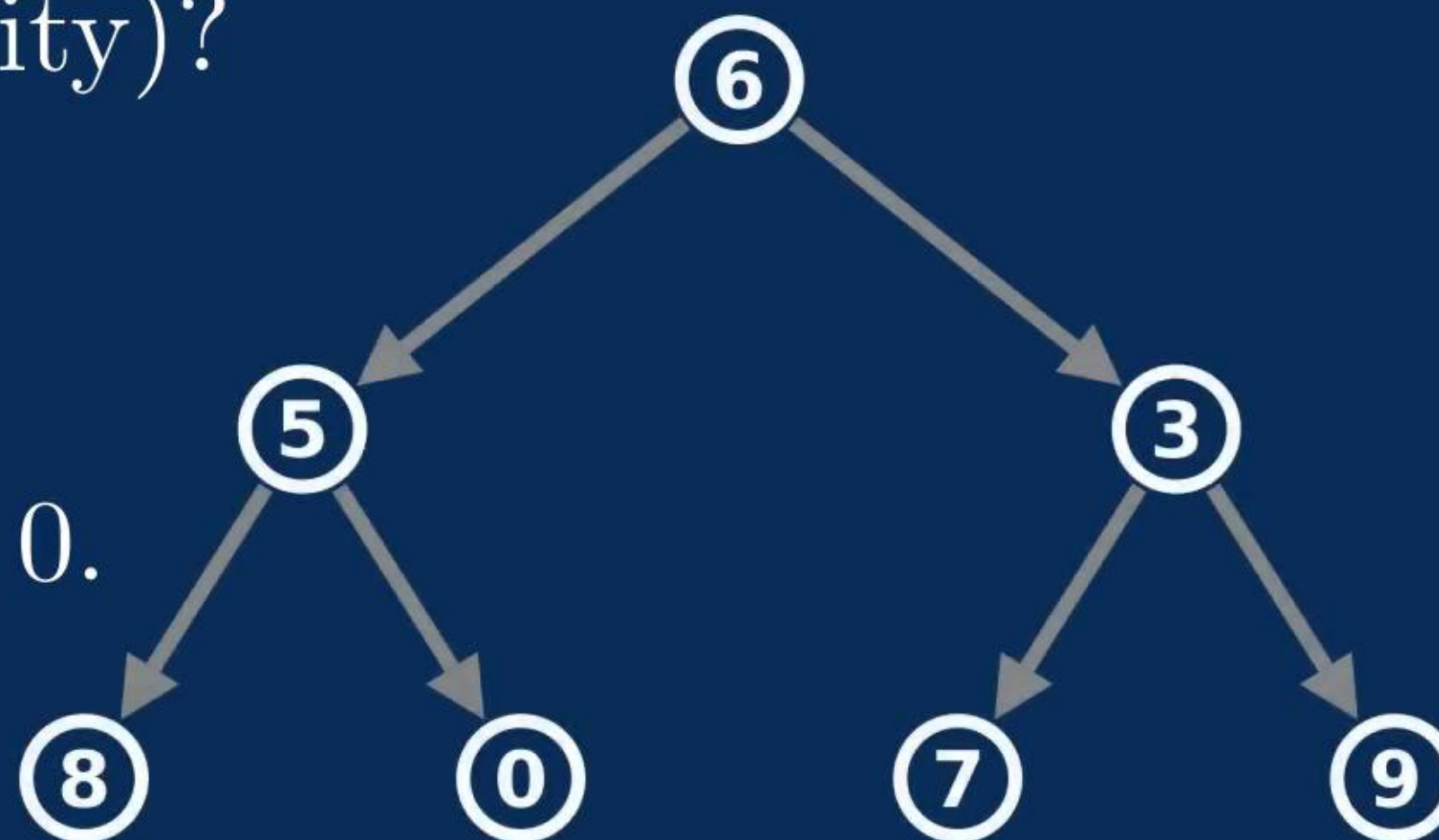


Node n and parent p

Is $n < p$ (higher priority)?

- $\text{swap}(n, p)$
- recurse up.

Let's do this on node 0.

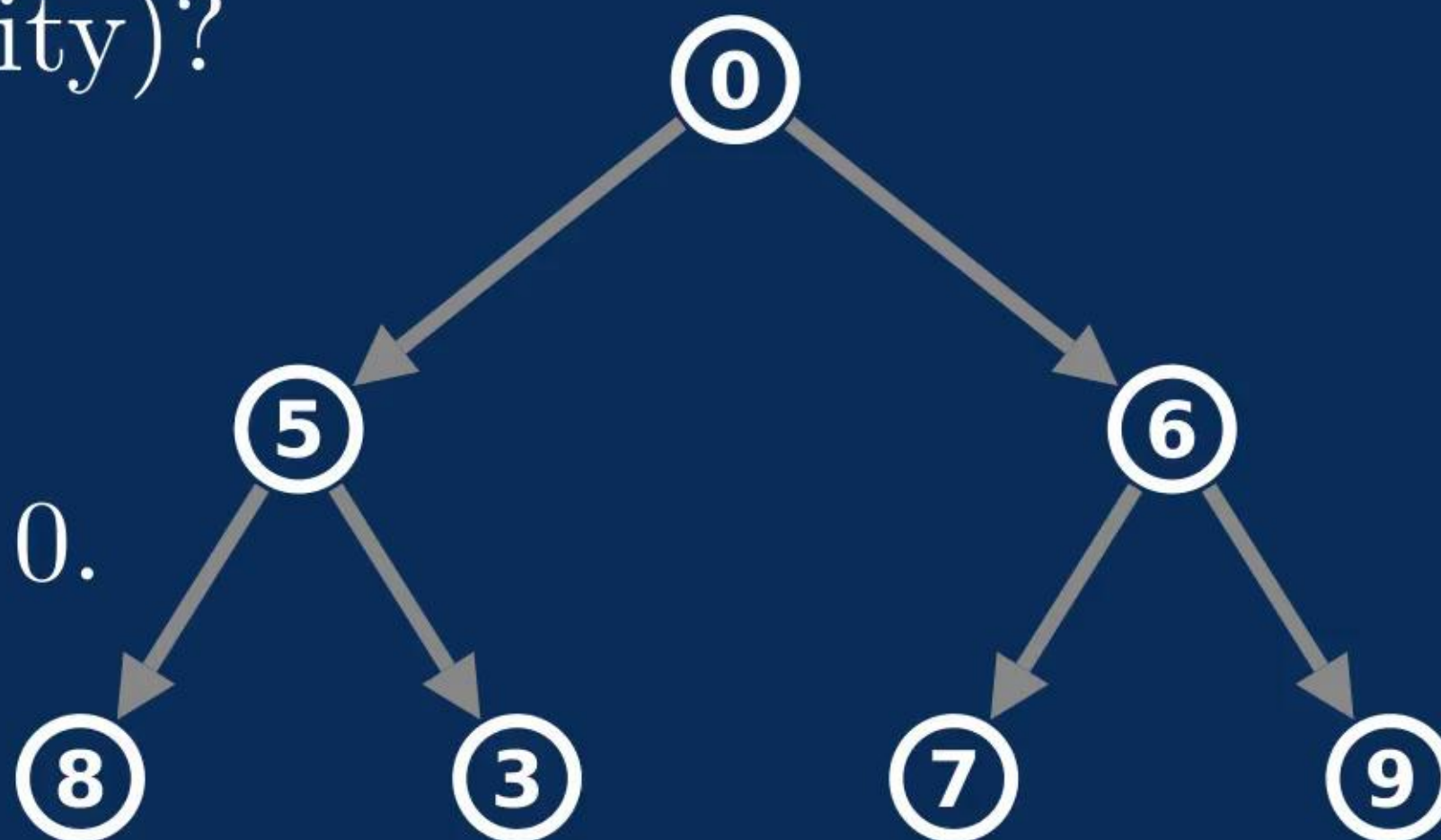


Node n and parent p

Is $n < p$ (higher priority)?

- $\text{swap}(n, p)$
- recurse up.

Let's do this on node 0.



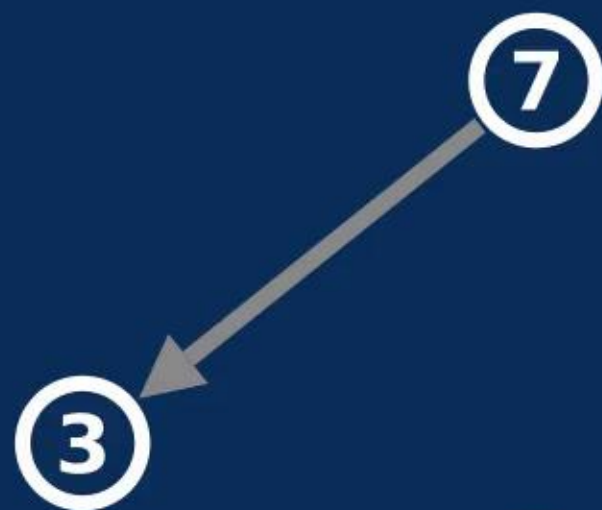
If a child is smaller, swap with smallest child

Before



If a child is smaller, swap with smallest child

Before



After



If a child is smaller, swap with smallest child

Before



After



If a child is smaller, swap with smallest child

Before

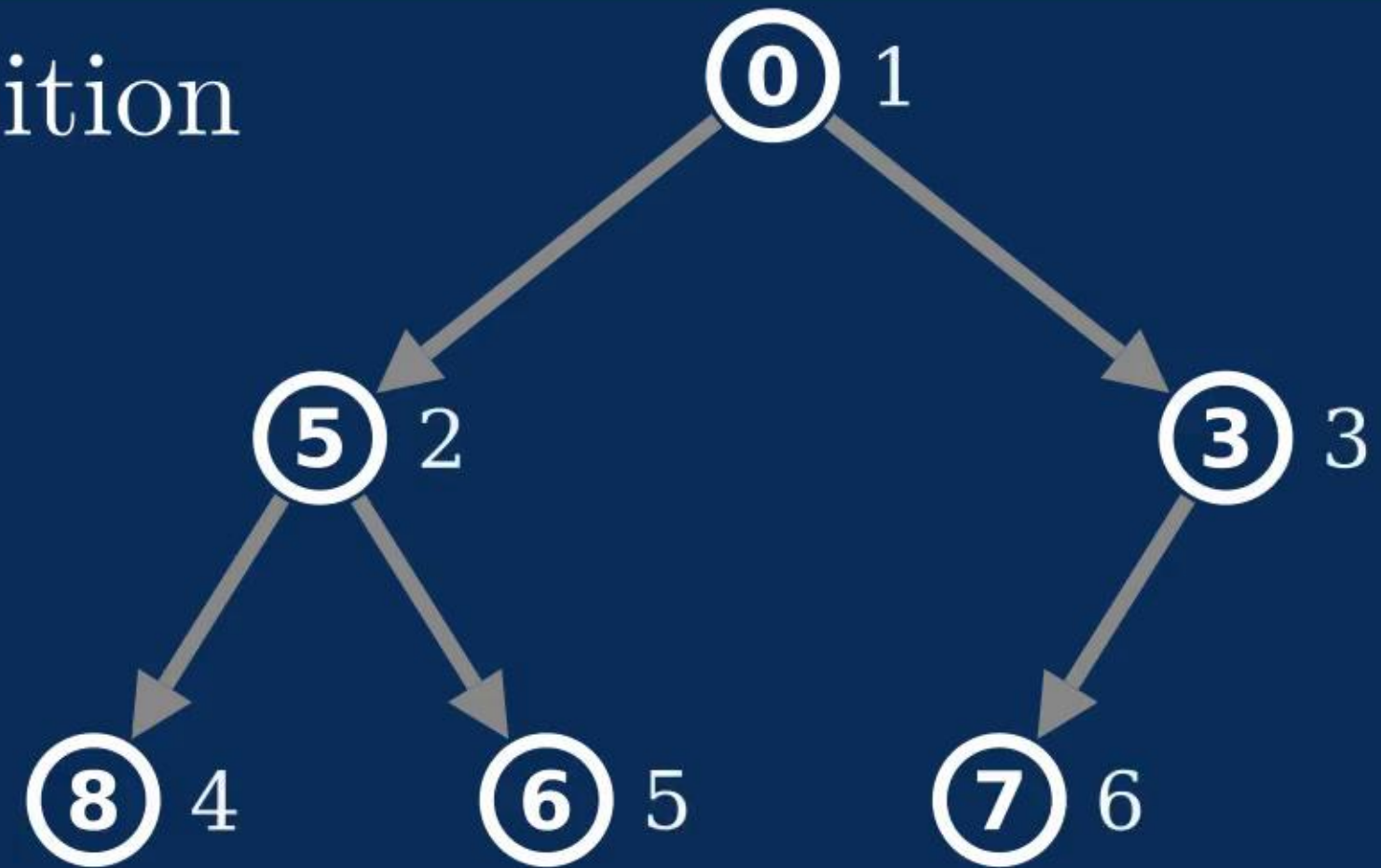


After



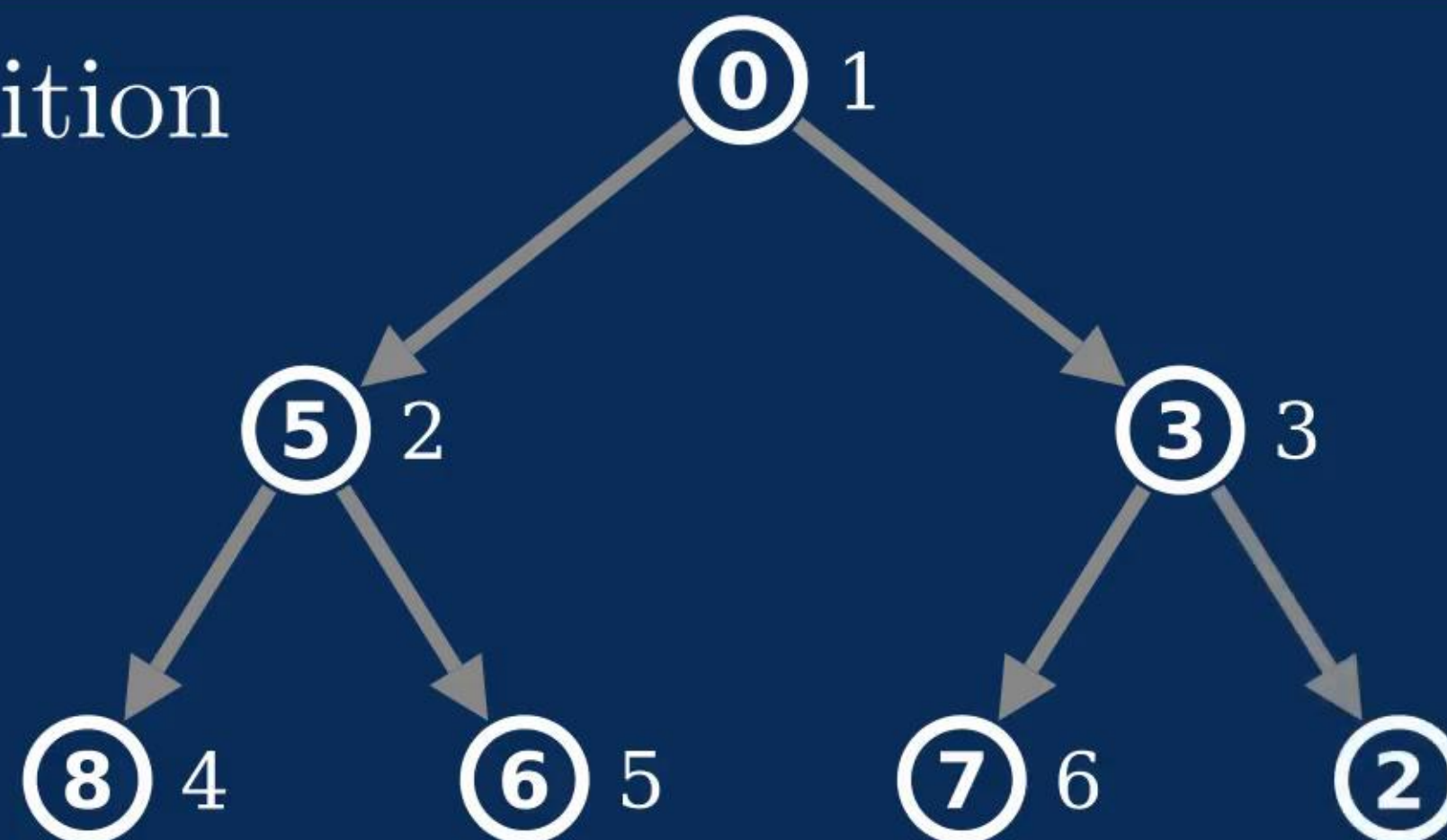
Place new element in last position

Call percolate up on it.



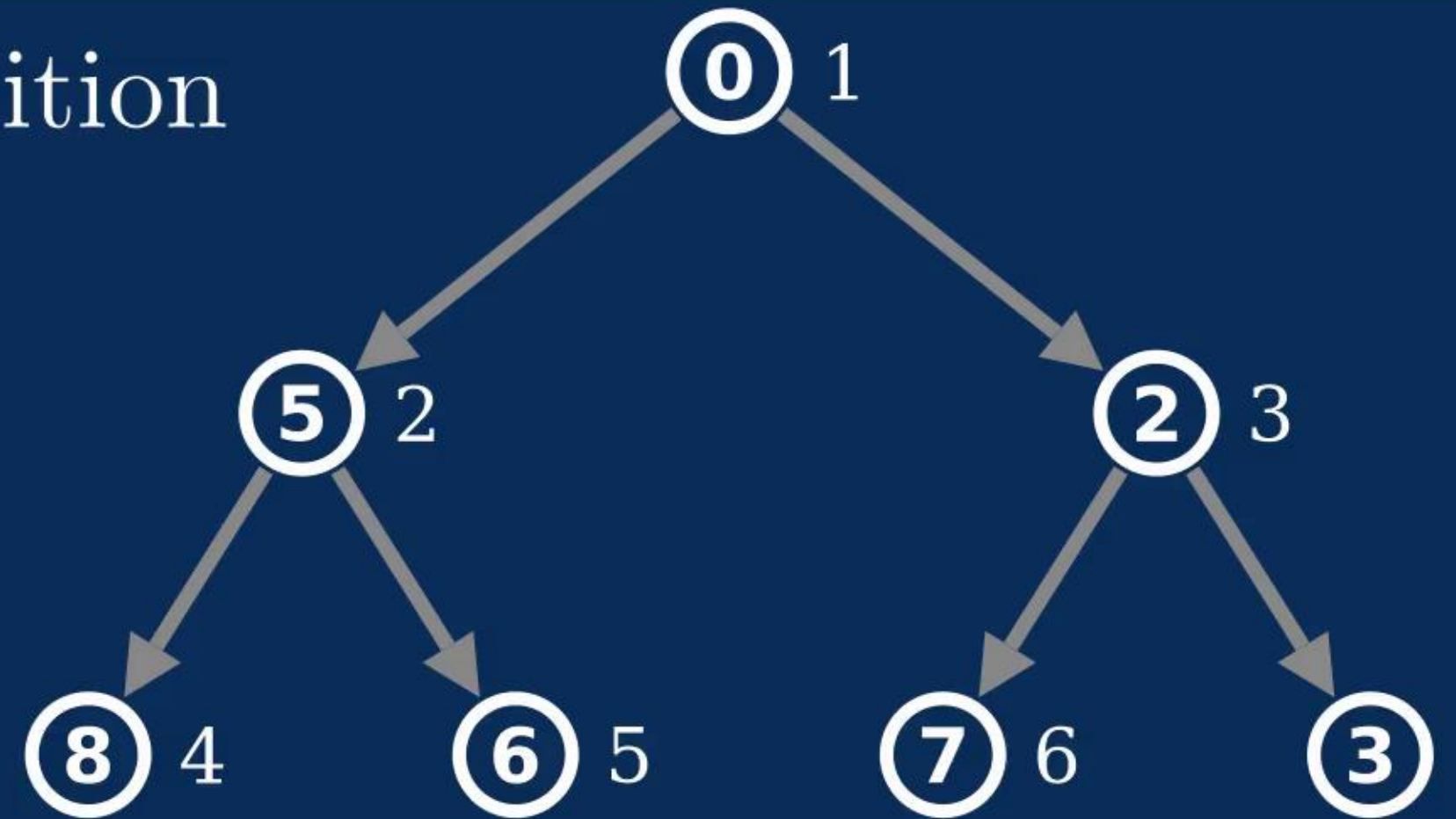
Place new element in last position

Call percolate up on it.



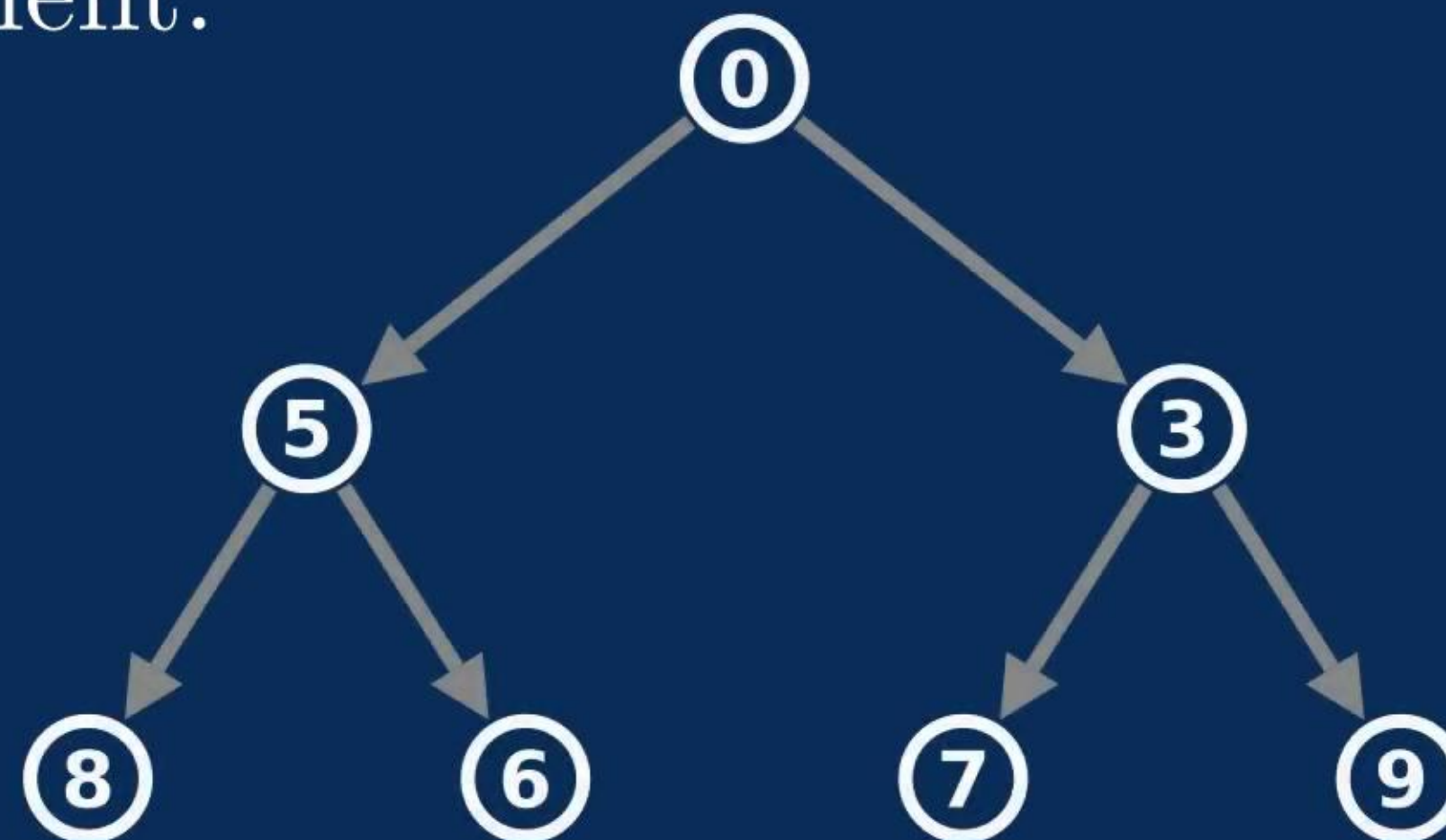
Place new element in last position

Call percolate up on it.



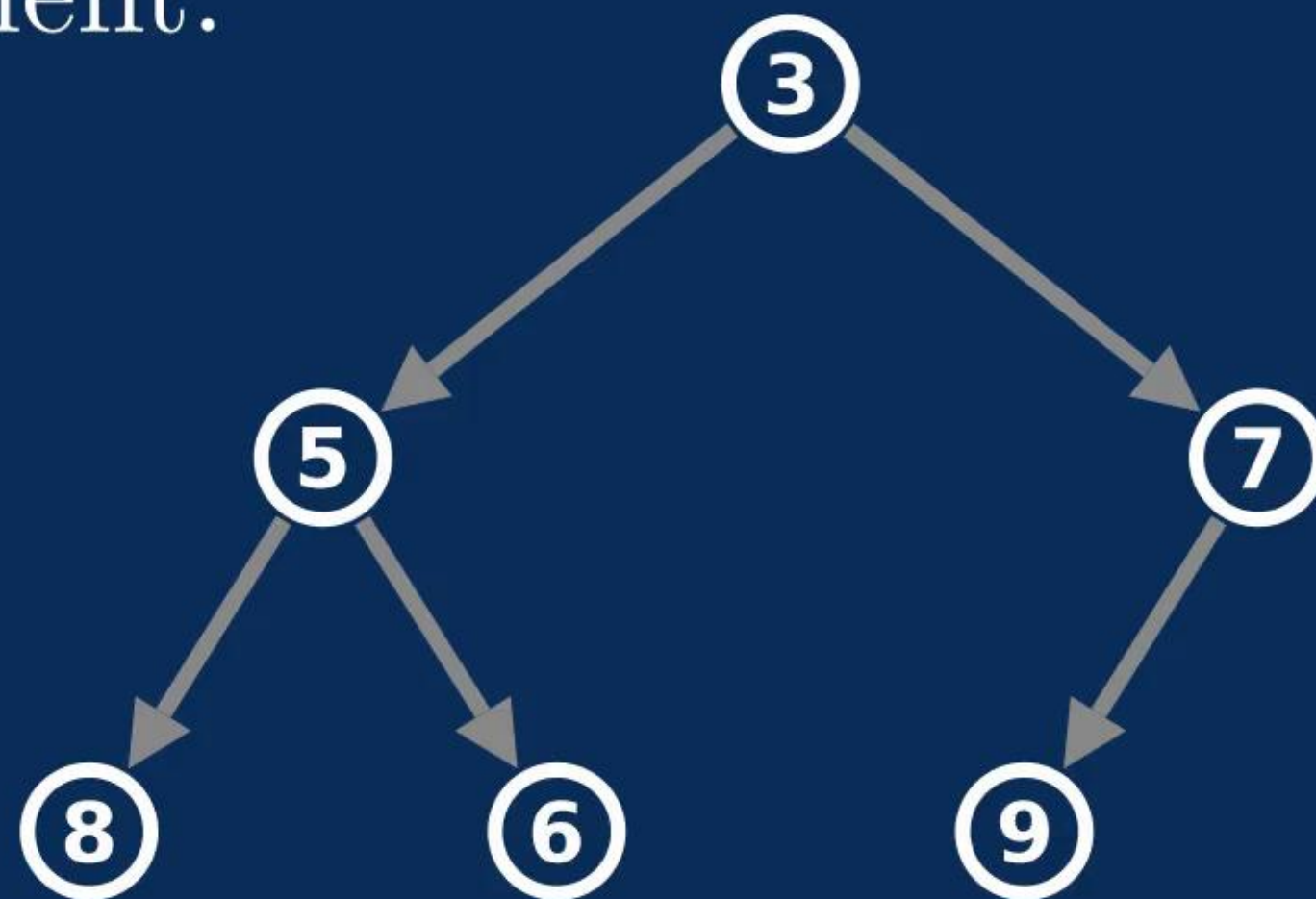
Overwrite root with last element.

Call `percolate down` on it.



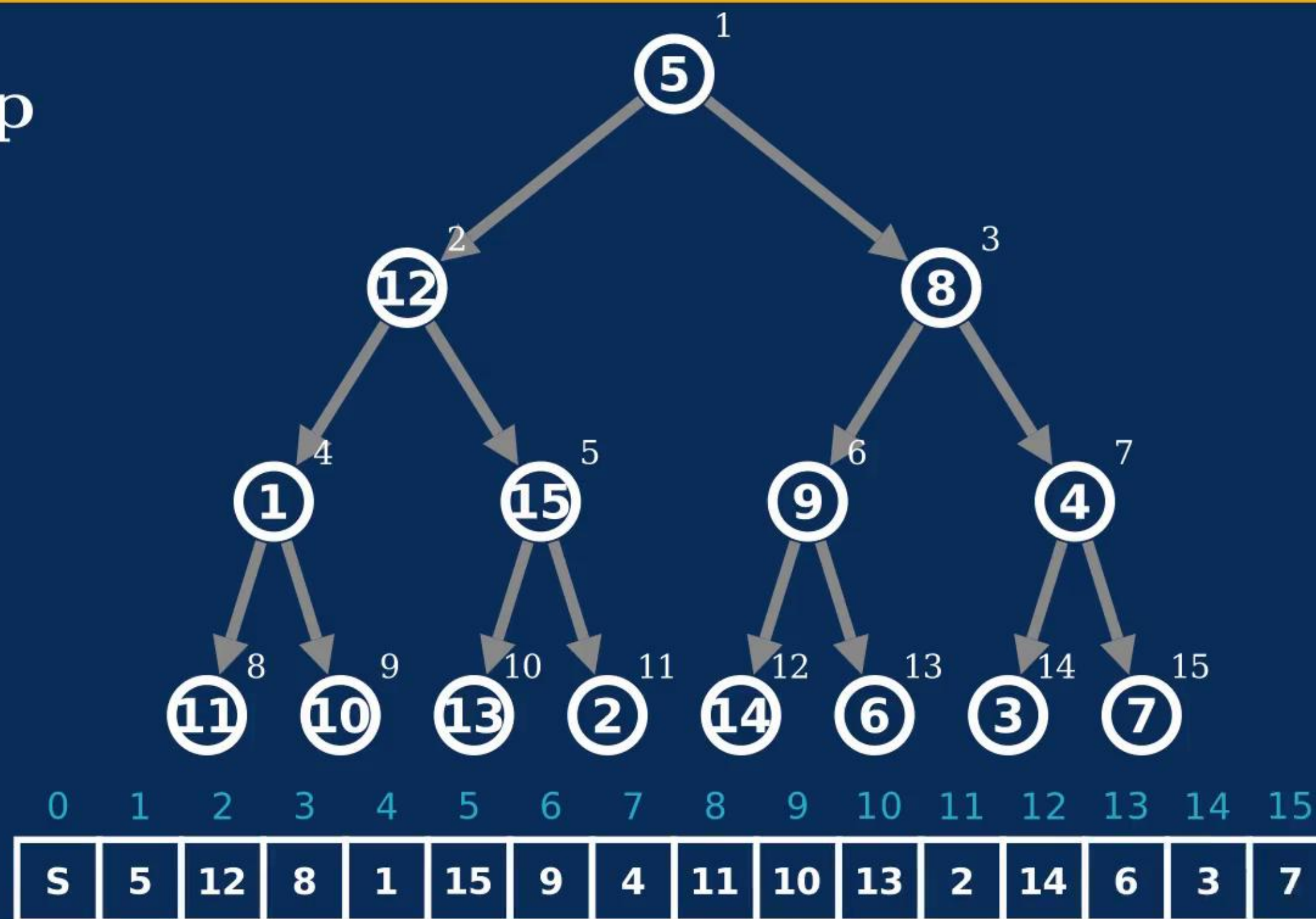
Overwrite root with last element.

Call percolate down on it.



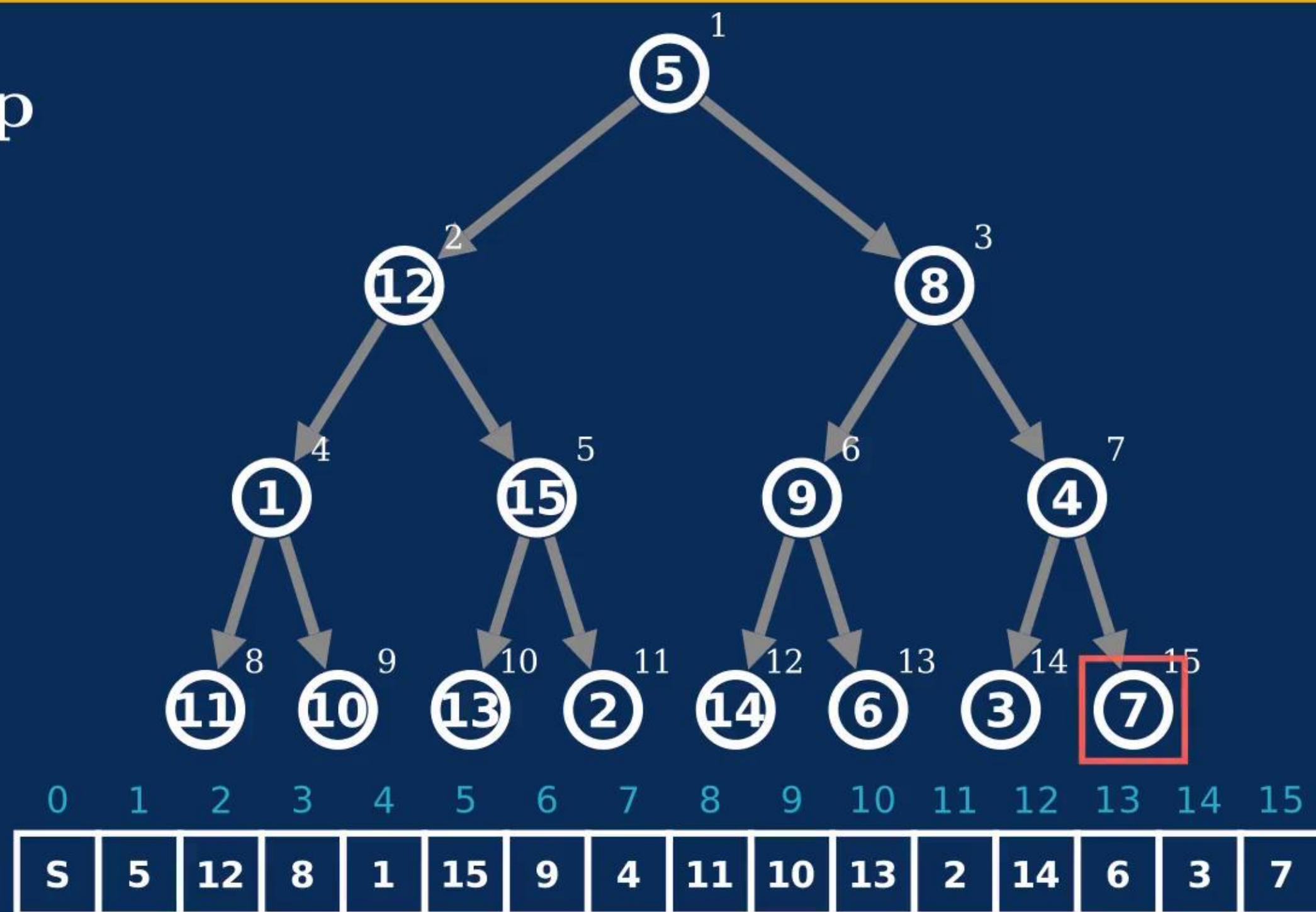
Creating a Heap

How to start?



Creating a Heap

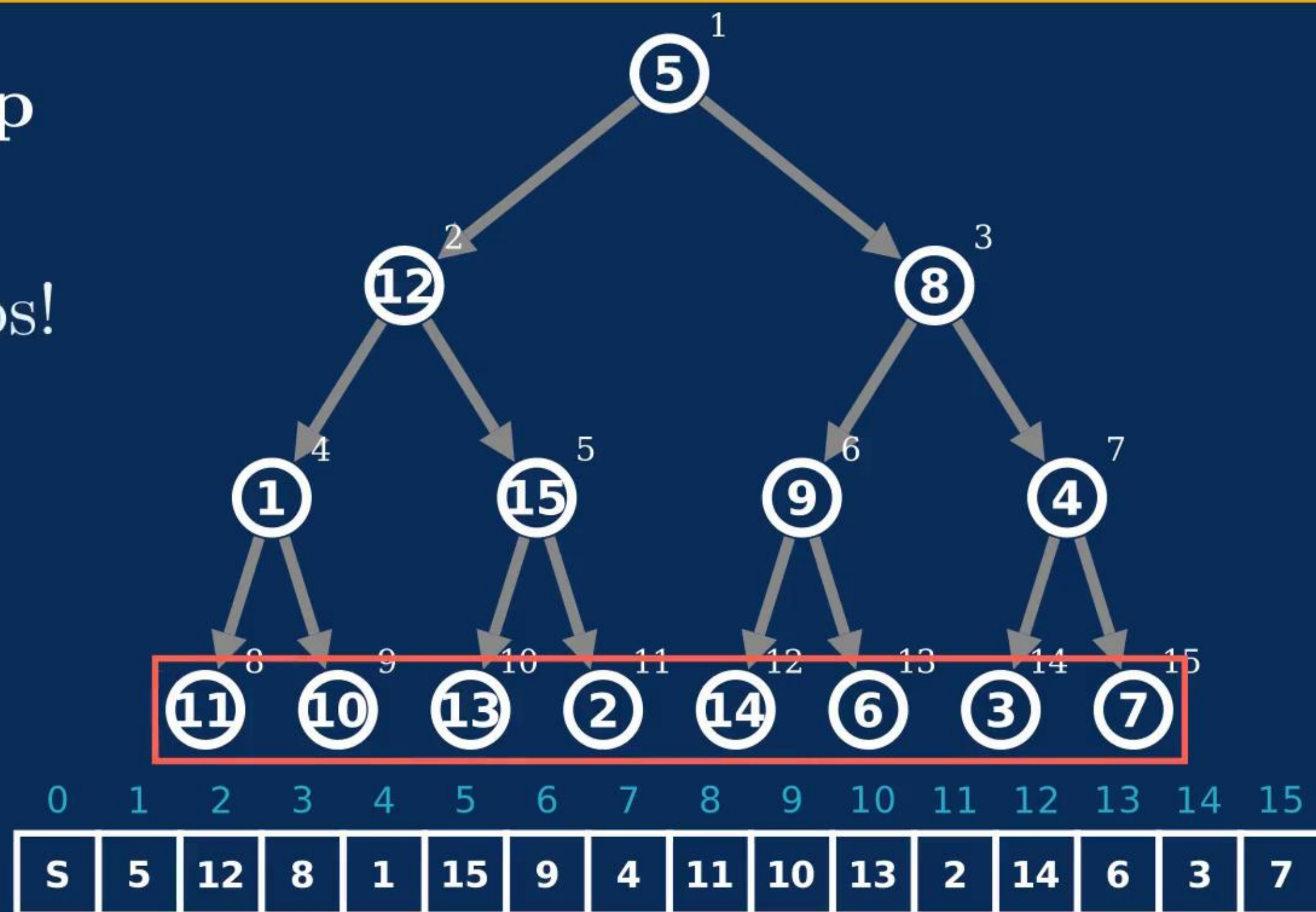
How to start?



Creating a Heap

How to start?

All these are heaps!

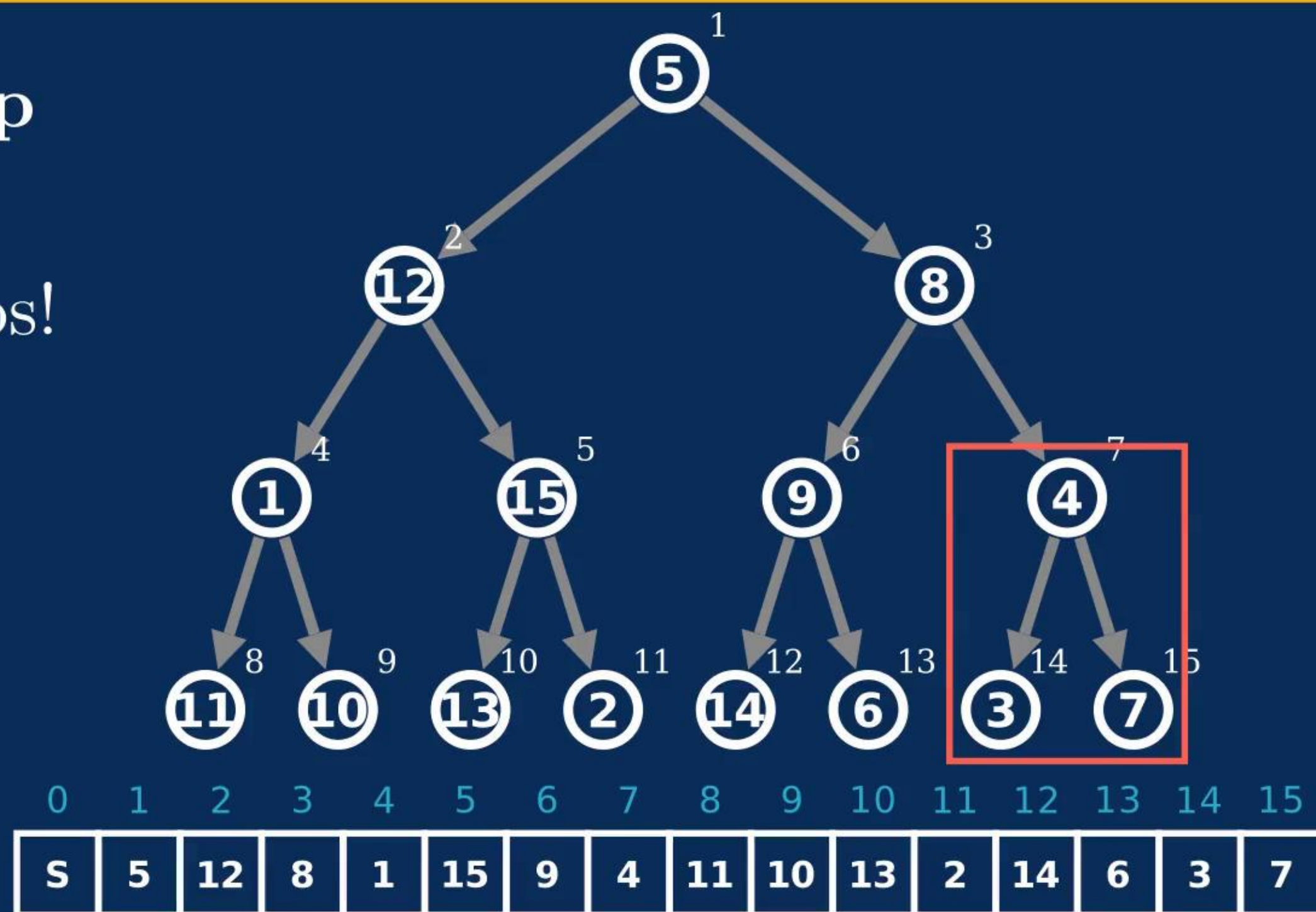


Creating a Heap

How to start?

All these are heaps!

How about this?



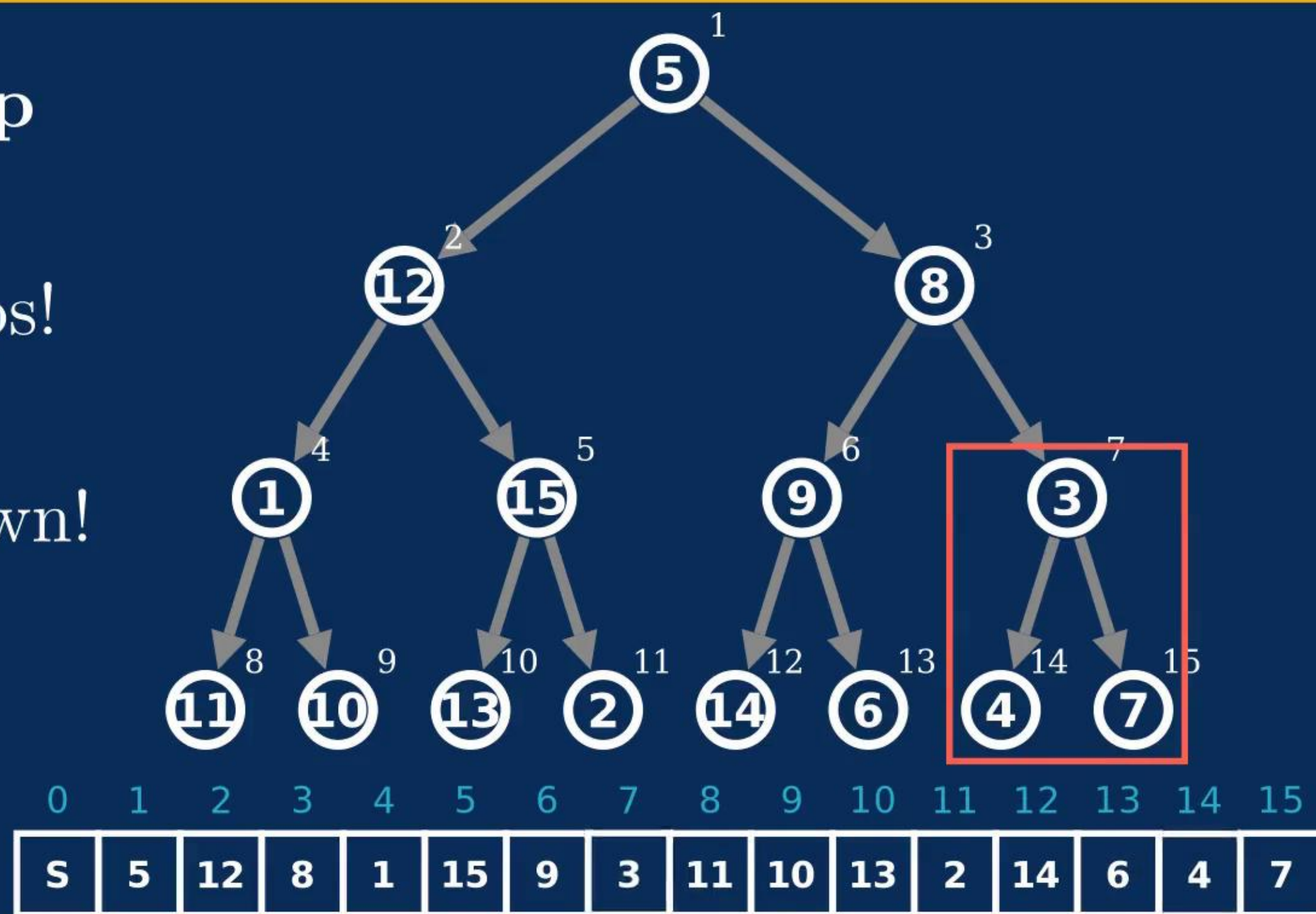
Creating a Heap

How to start?

All these are heaps!

How about this?

Just percolate down!



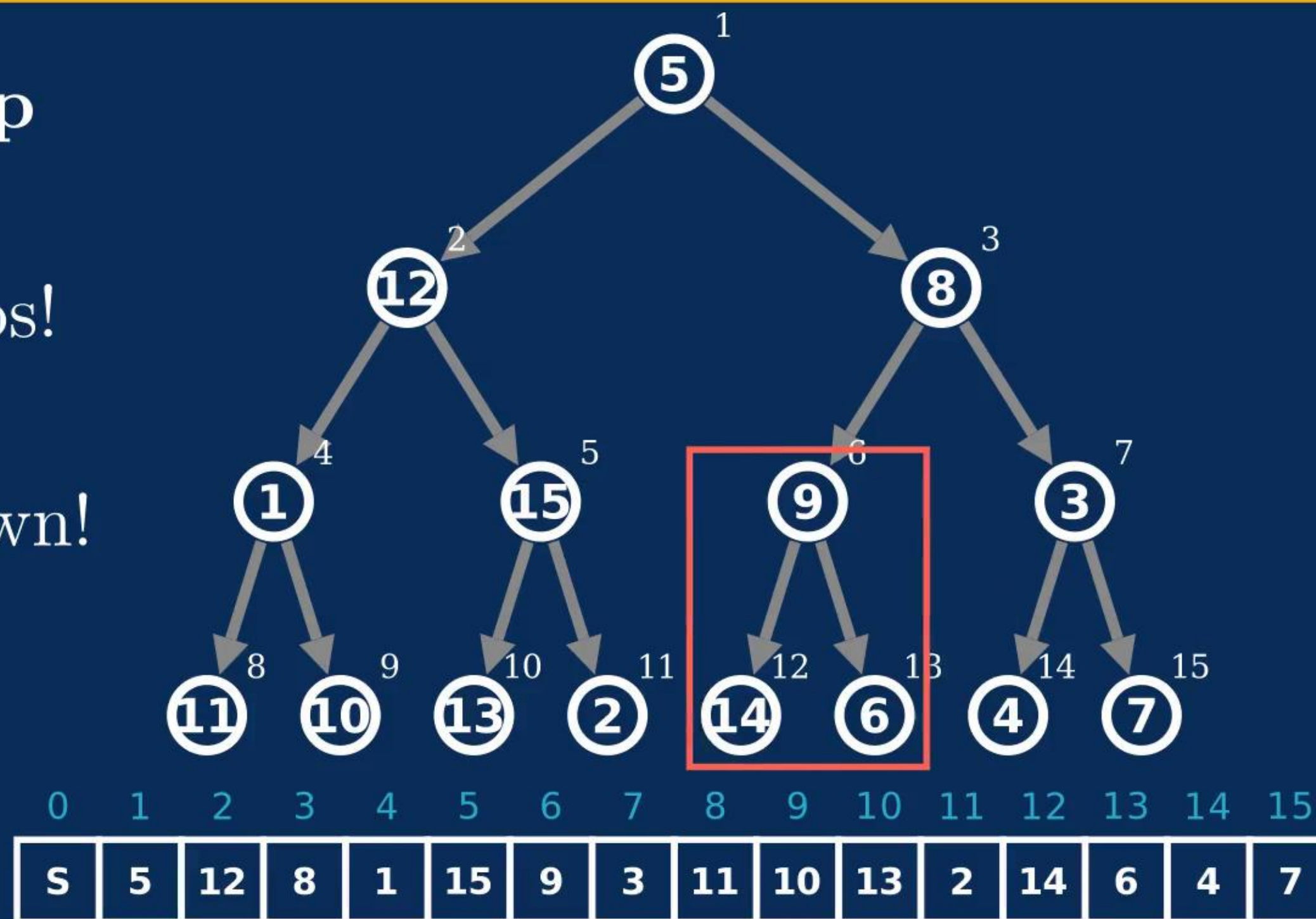
Creating a Heap

How to start?

All these are heaps!

How about this?

Just percolate down!



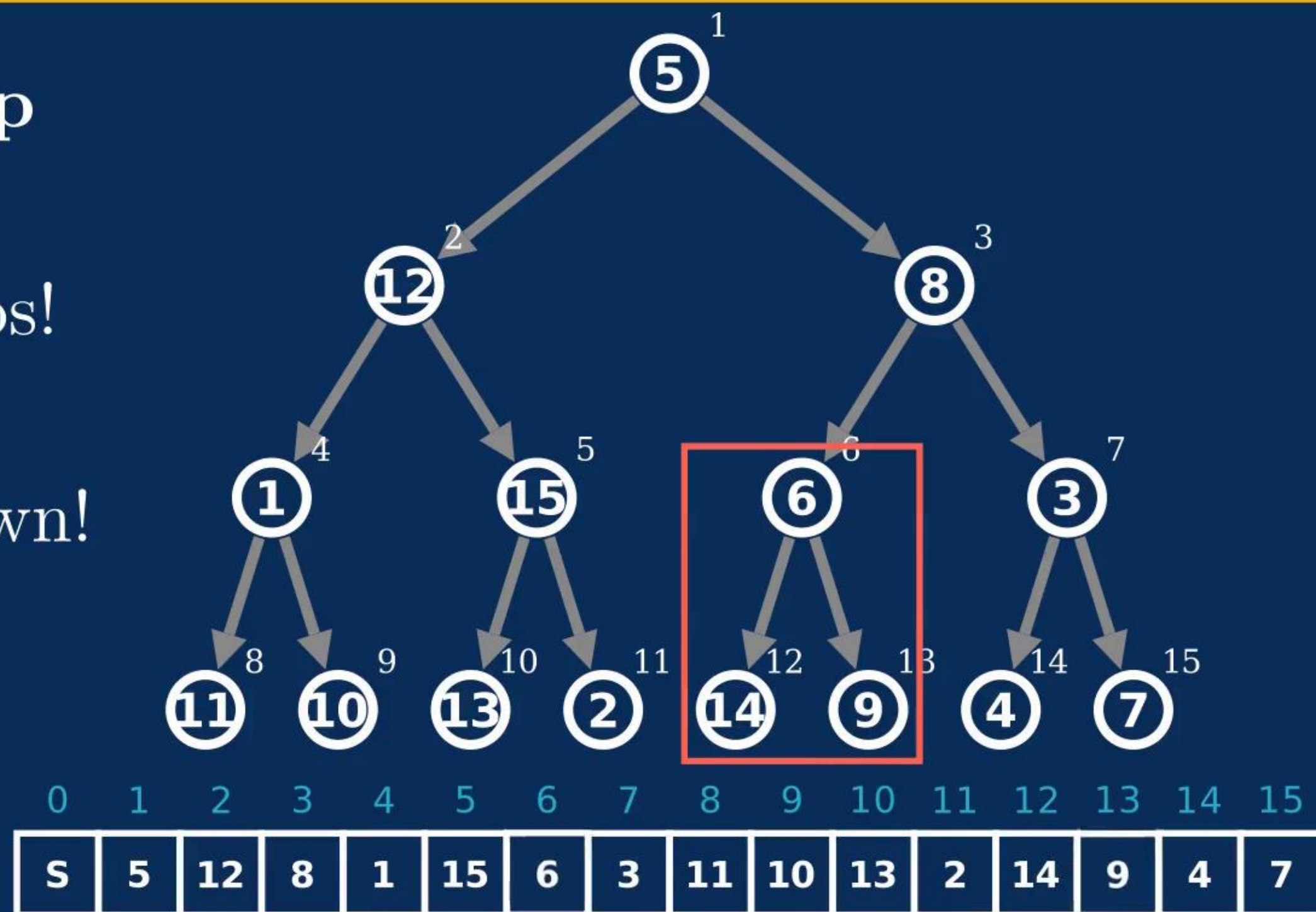
Creating a Heap

How to start?

All these are heaps!

How about this?

Just percolate down!



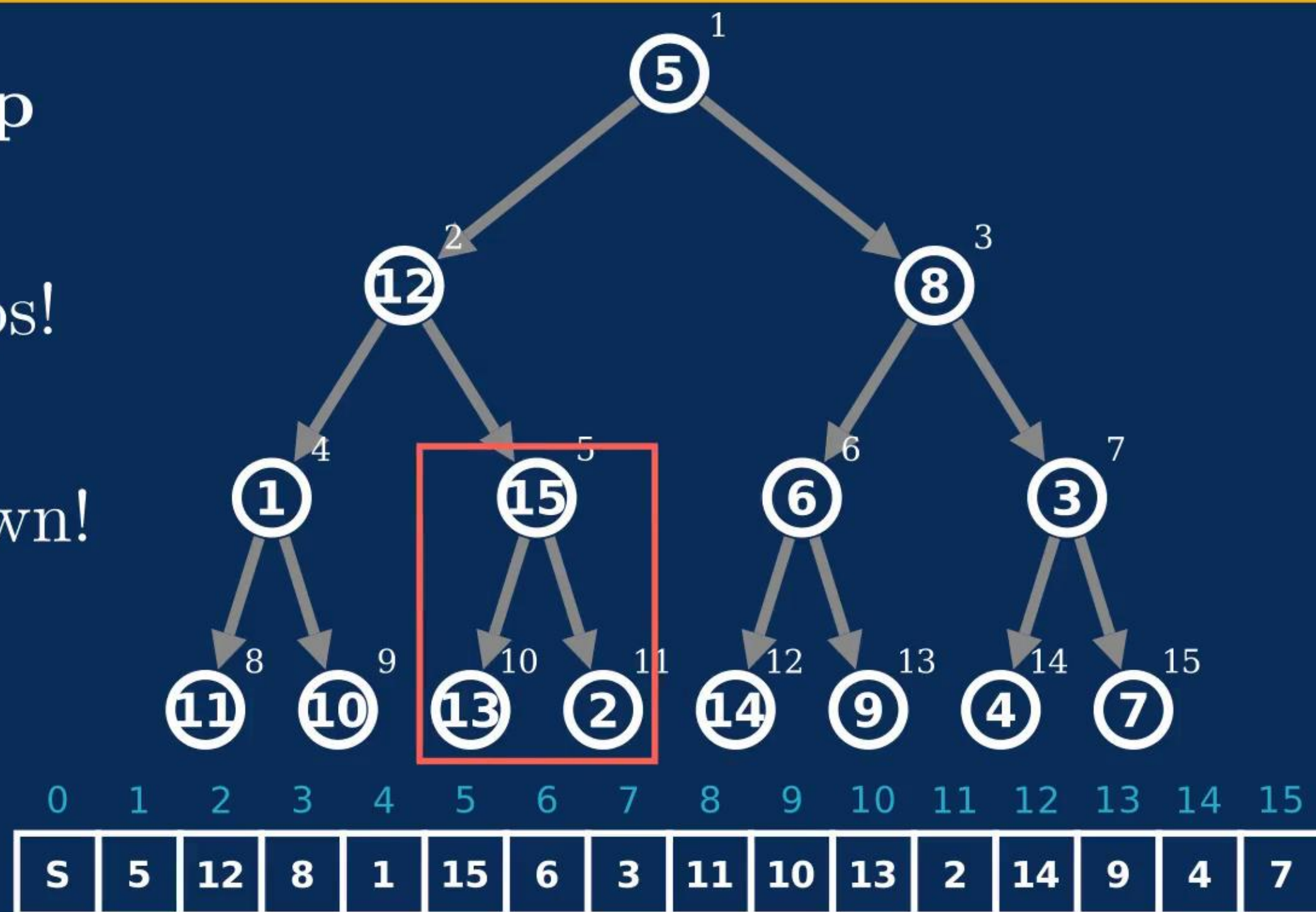
Creating a Heap

How to start?

All these are heaps!

How about this?

Just percolate down!



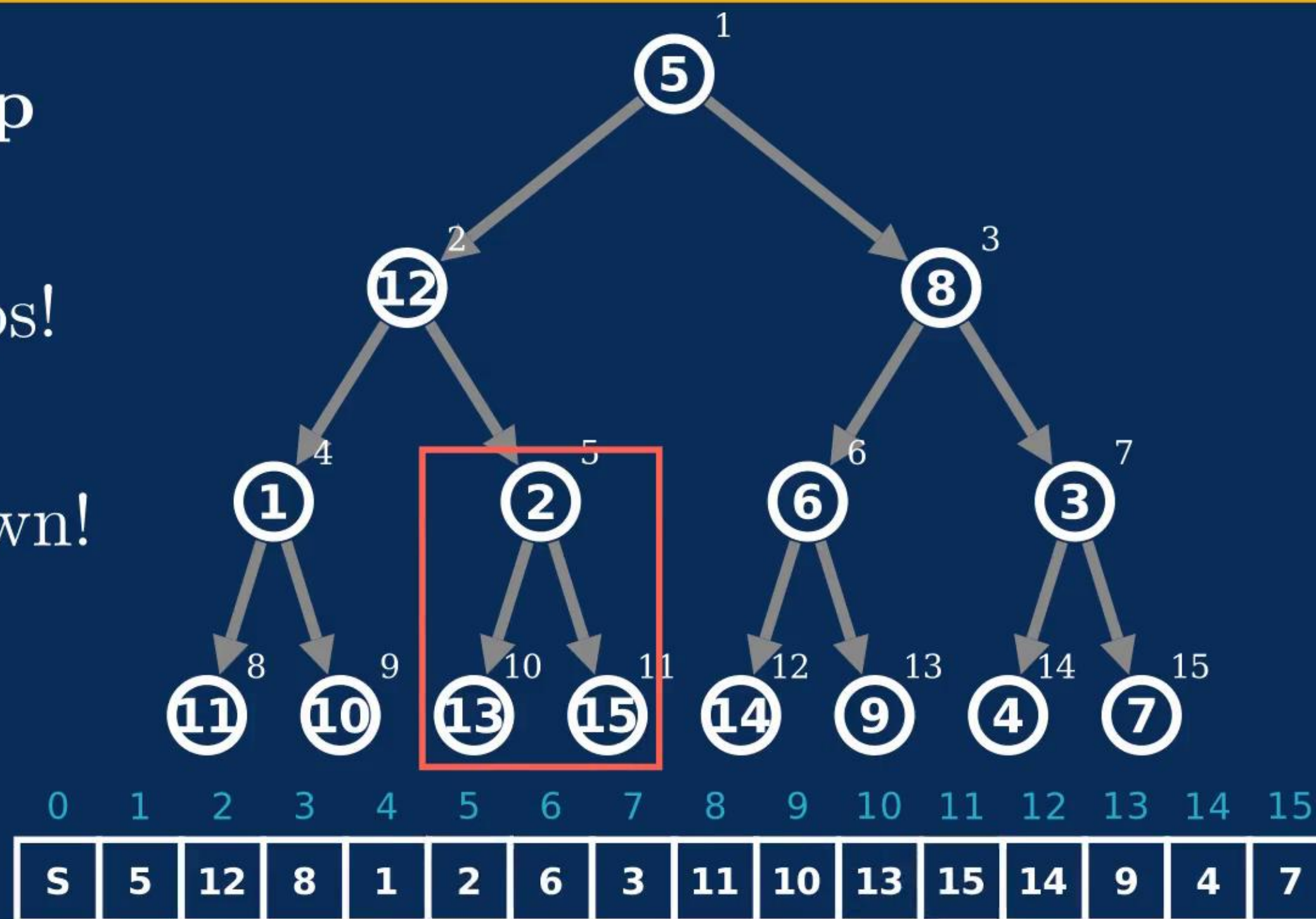
Creating a Heap

How to start?

All these are heaps!

How about this?

Just percolate down!



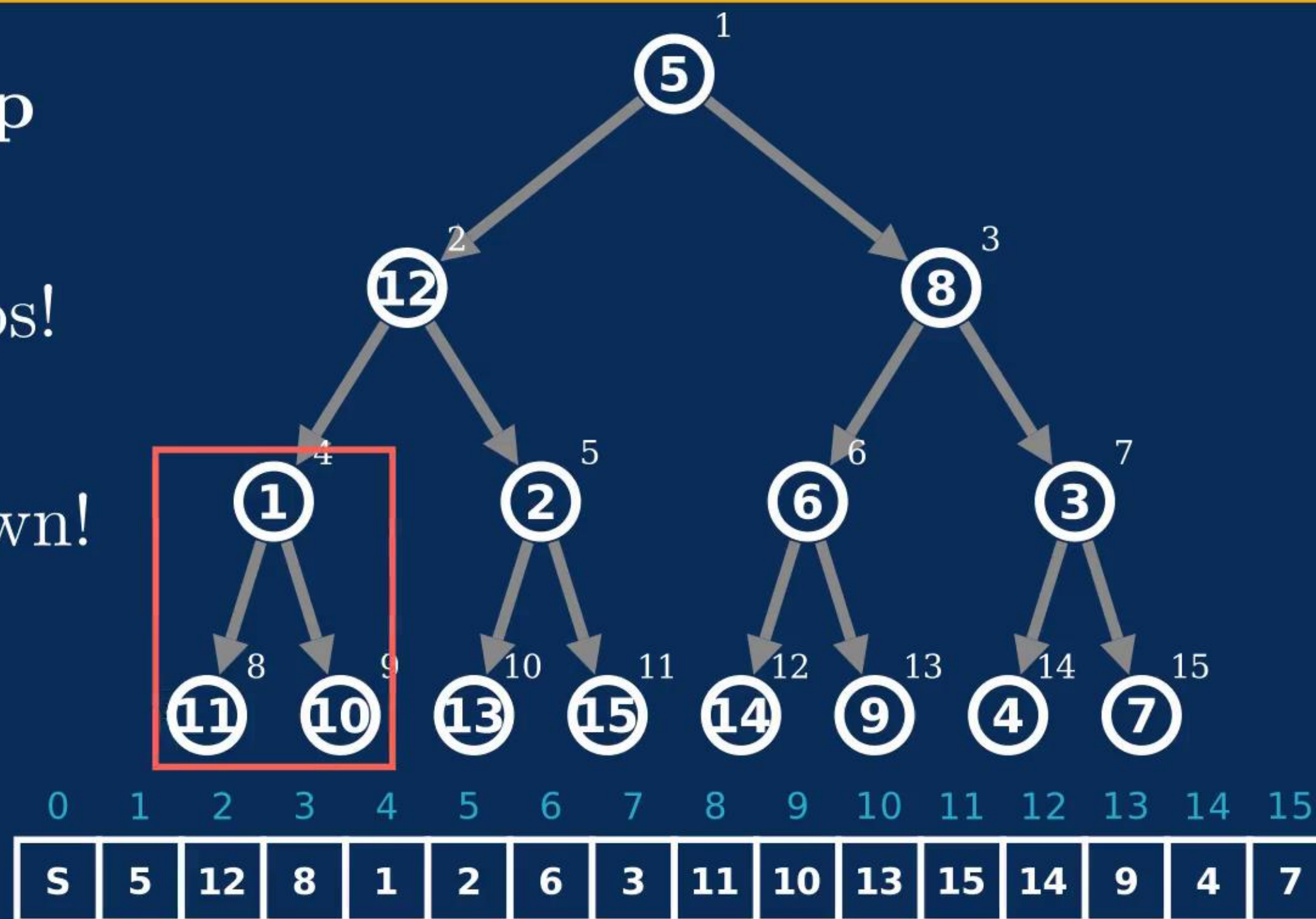
Creating a Heap

How to start?

All these are heaps!

How about this?

Just percolate down!



Creating a Heap

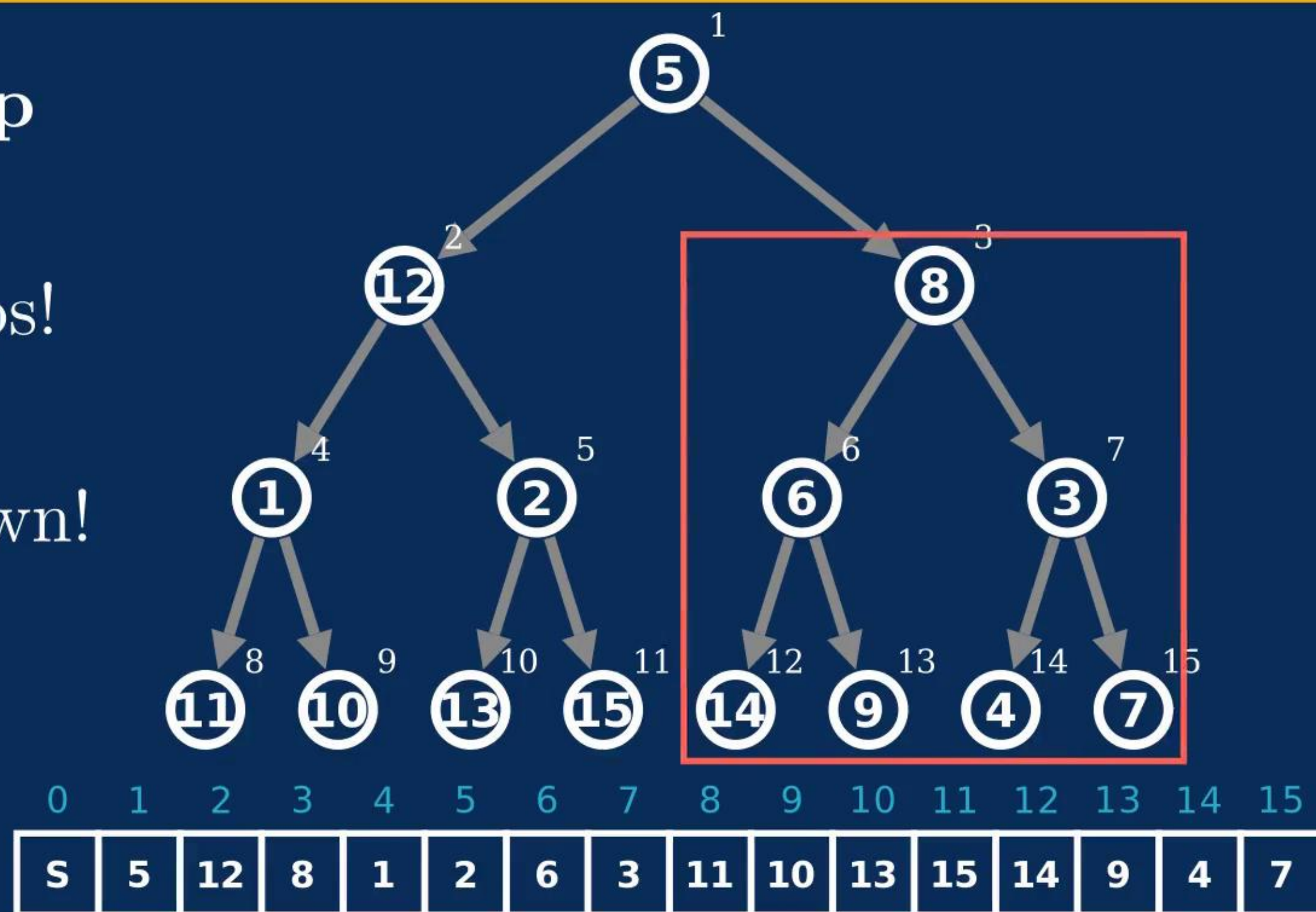
How to start?

All these are heaps!

How about this?

Just percolate down!

Next row...



Creating a Heap

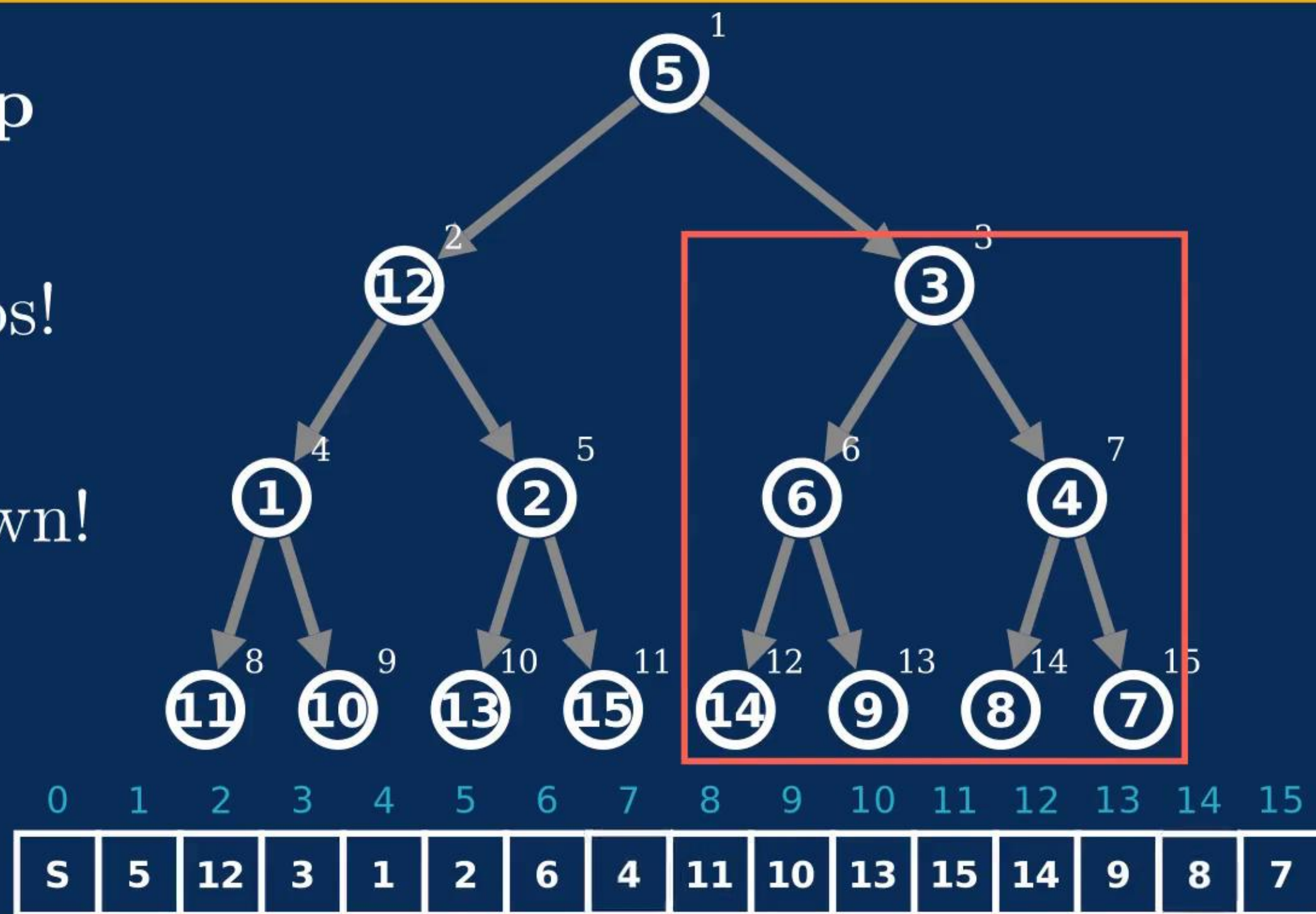
How to start?

All these are heaps!

How about this?

Just percolate down!

Next row...



Creating a Heap

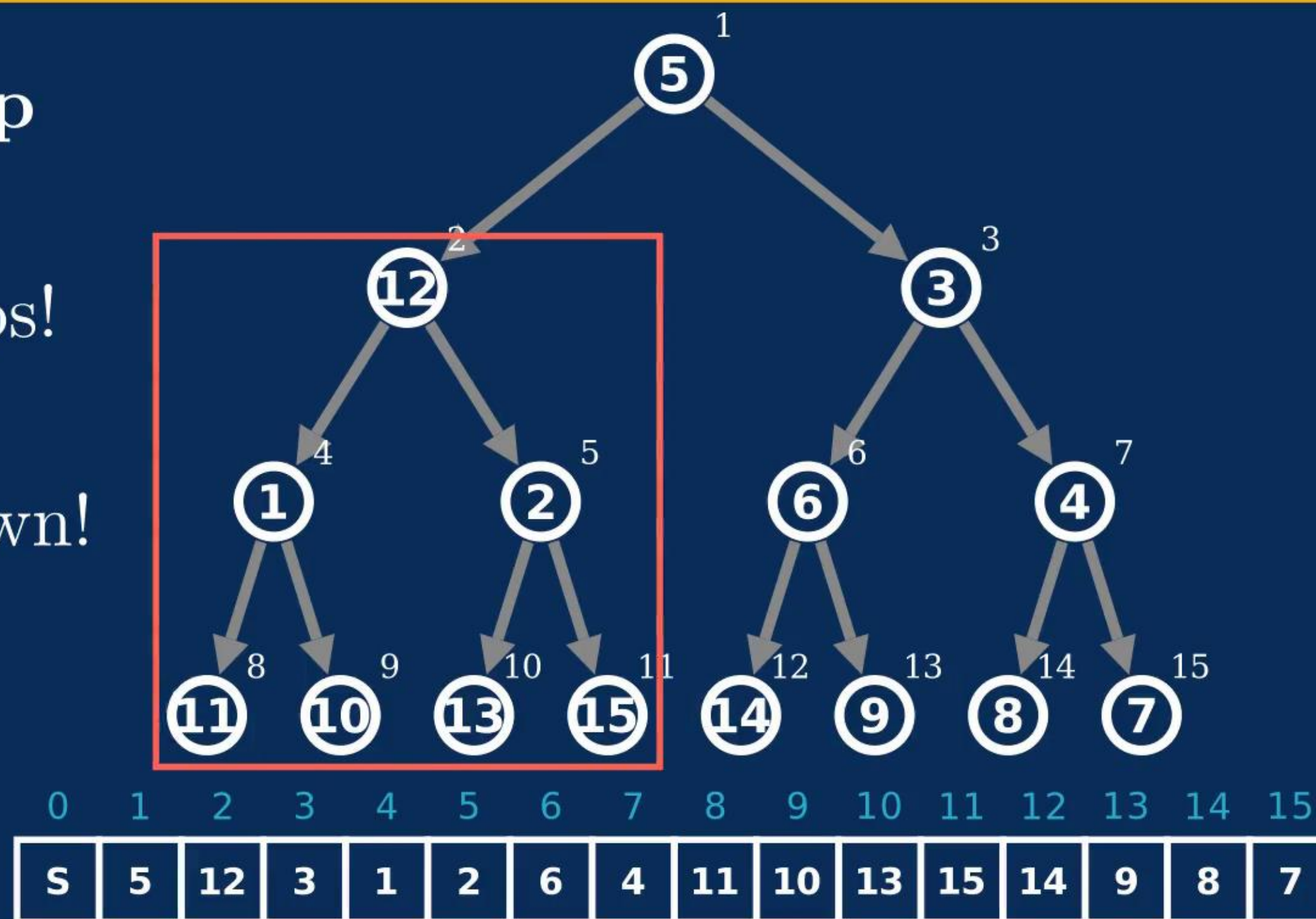
How to start?

All these are heaps!

How about this?

Just percolate down!

Next row...



Creating a Heap

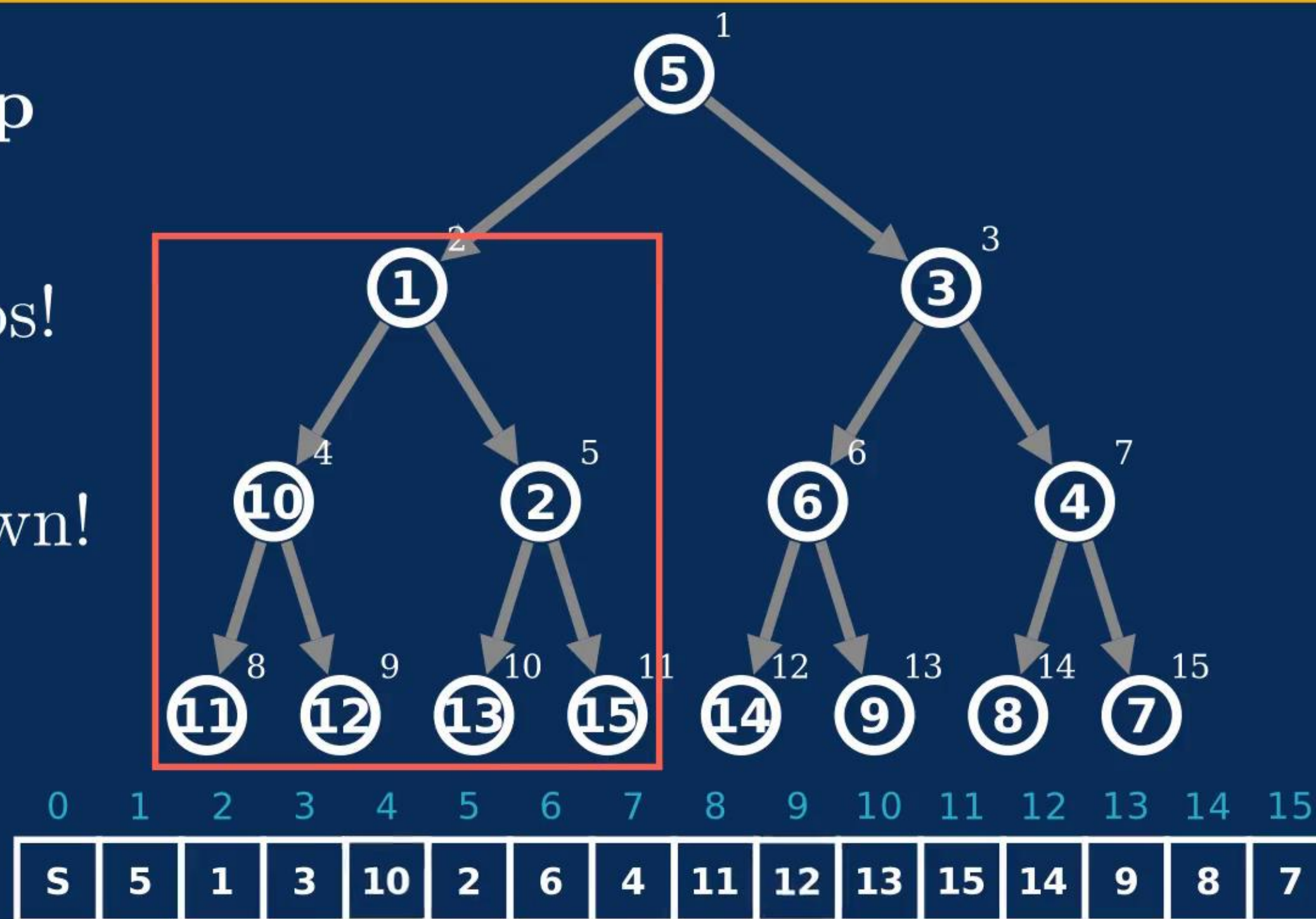
How to start?

All these are heaps!

How about this?

Just percolate down!

Next row...



Creating a Heap

How to start?

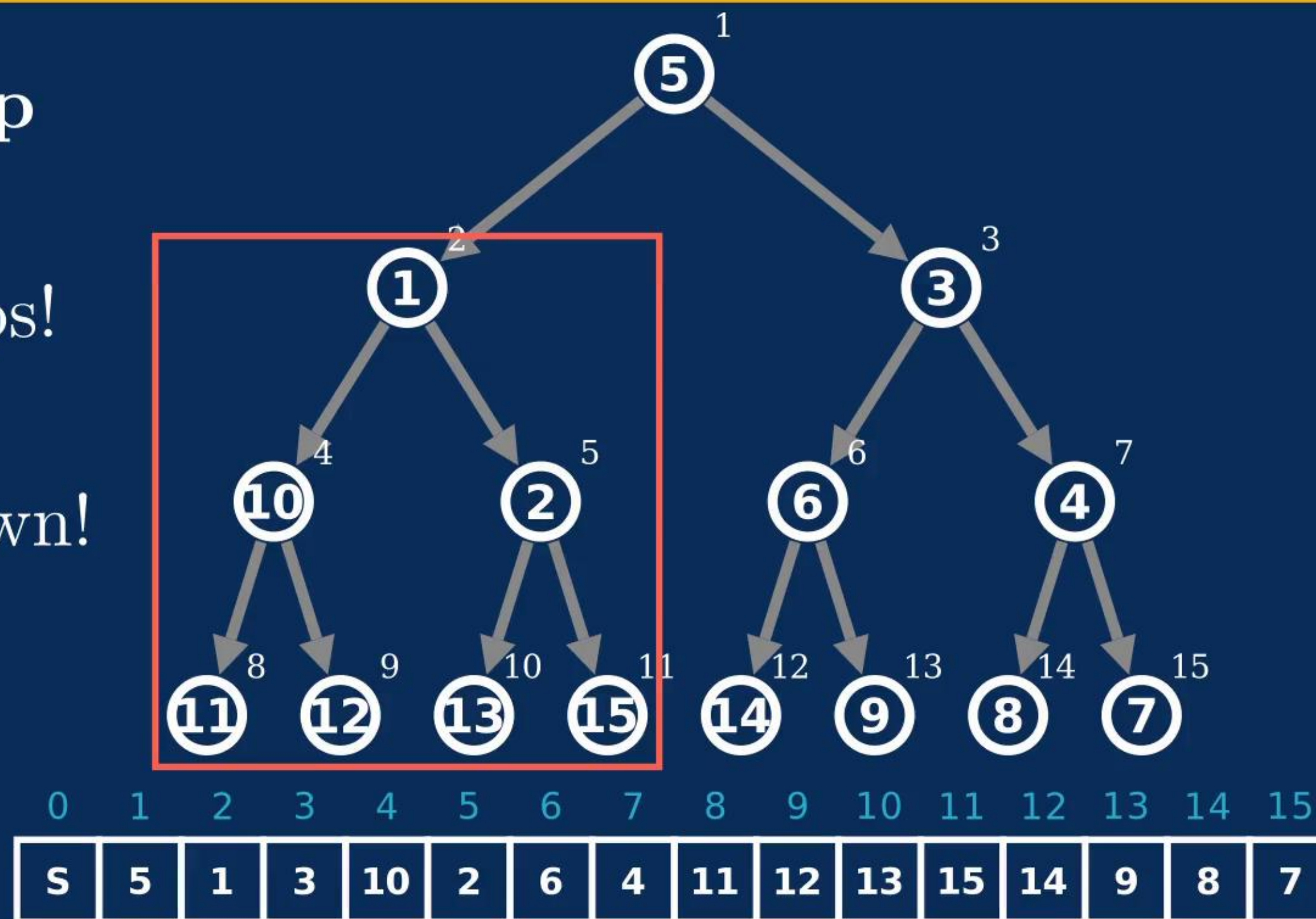
All these are heaps!

How about this?

Just percolate down!

Next row...

Root...



Creating a Heap

How to start?

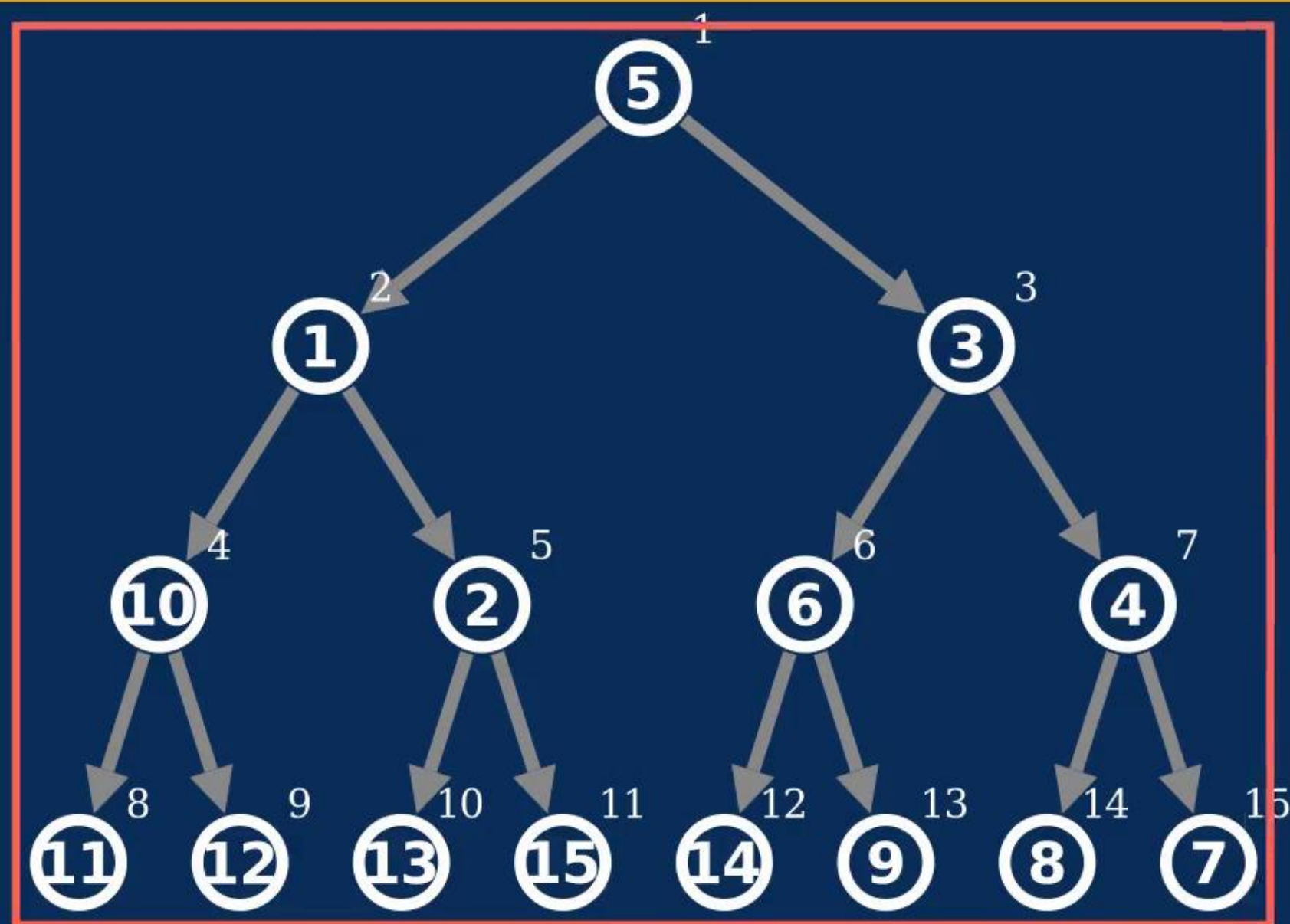
All these are heaps!

How about this?

Just percolate down!

Next row...

Root...



0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
S	5	1	3	10	2	6	4	11	12	13	15	14	9	8	7

Creating a Heap

How to start?

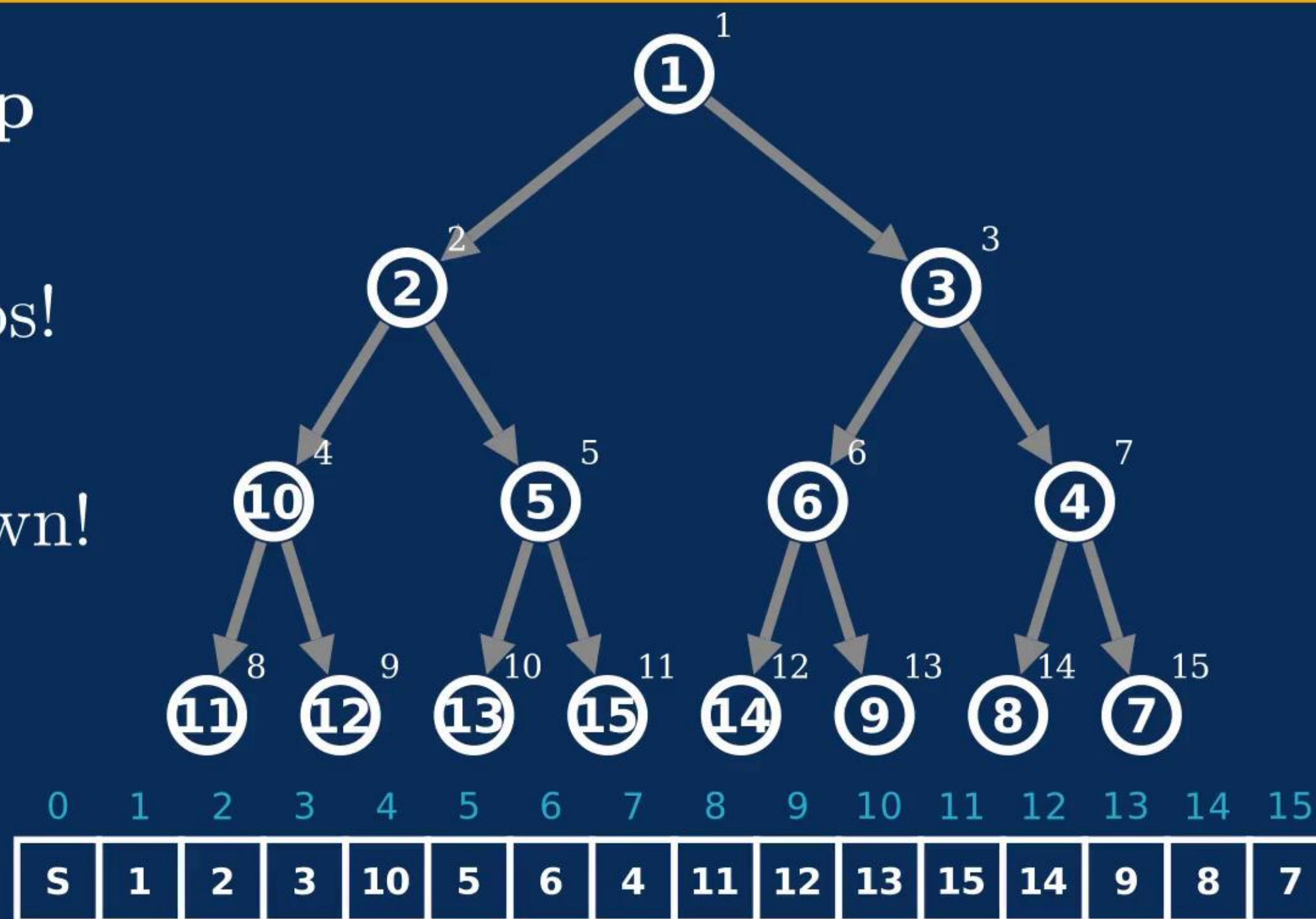
All these are heaps!

How about this?

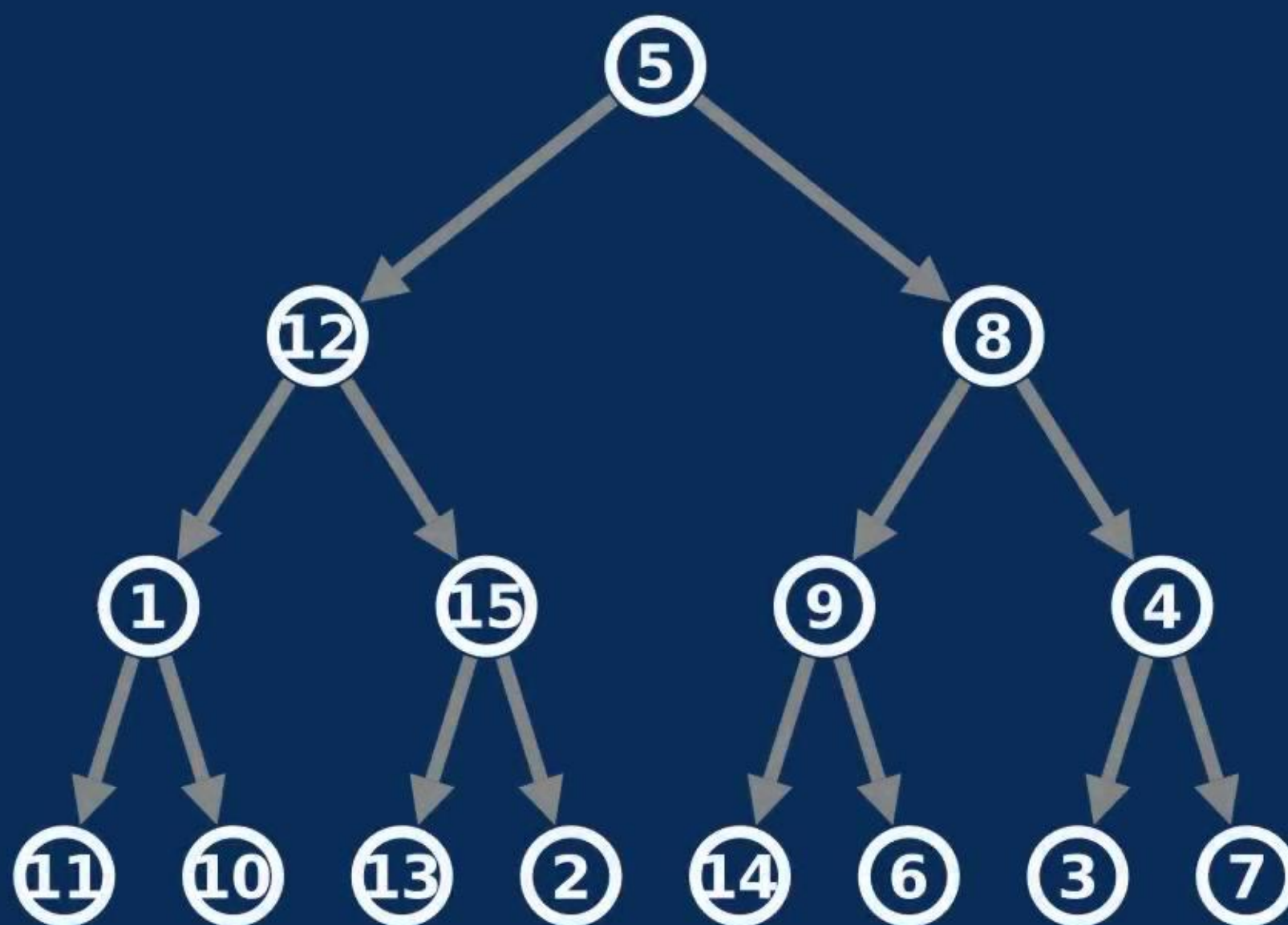
Just percolate down!

Next row...

Root...

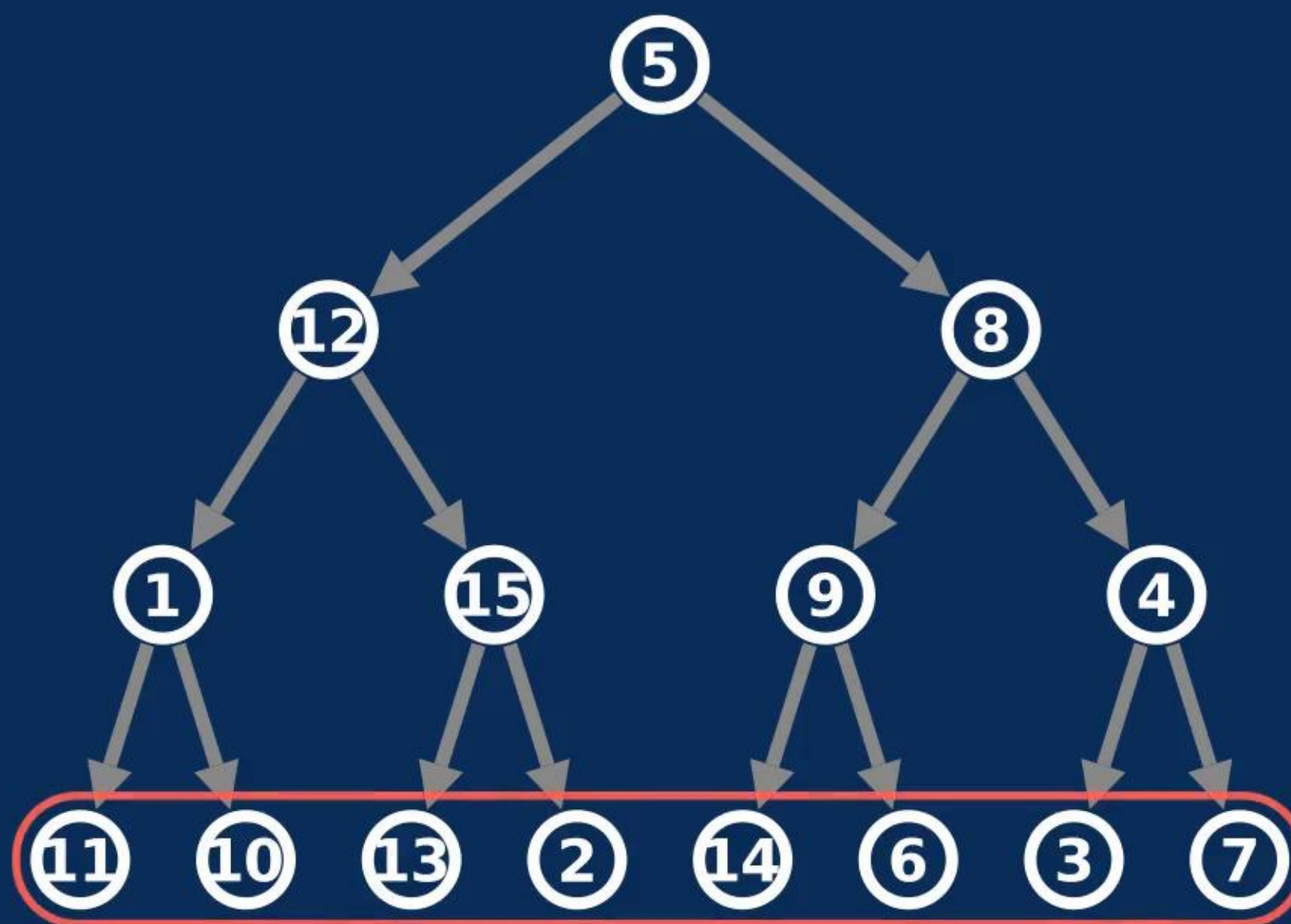


How long does this take?



How long does this take?

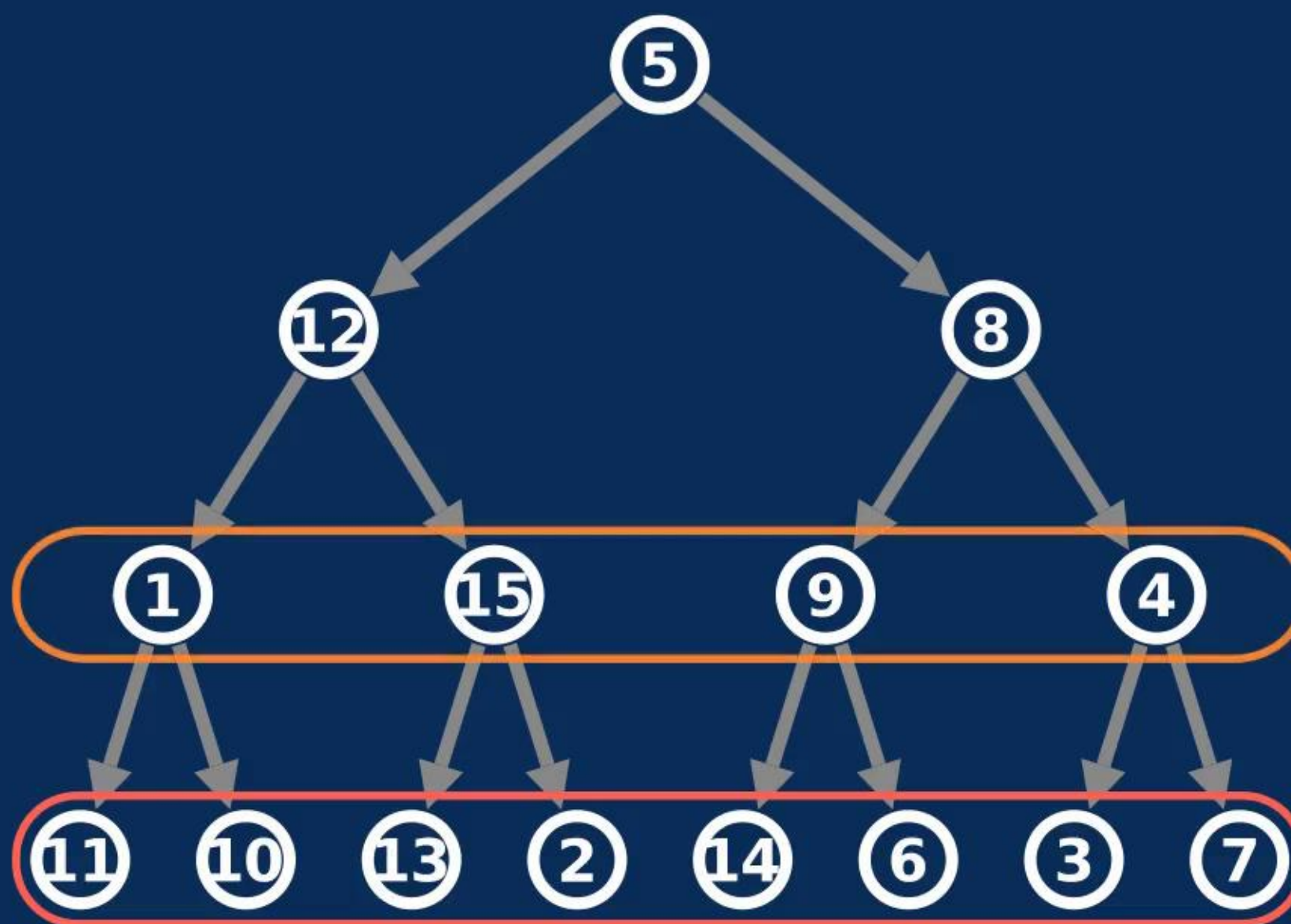
$h = 3$ (leaves): Free!



How long does this take?

$h = 3$ (leaves): Free!

$h = 2$: 1 each $\times 4$.

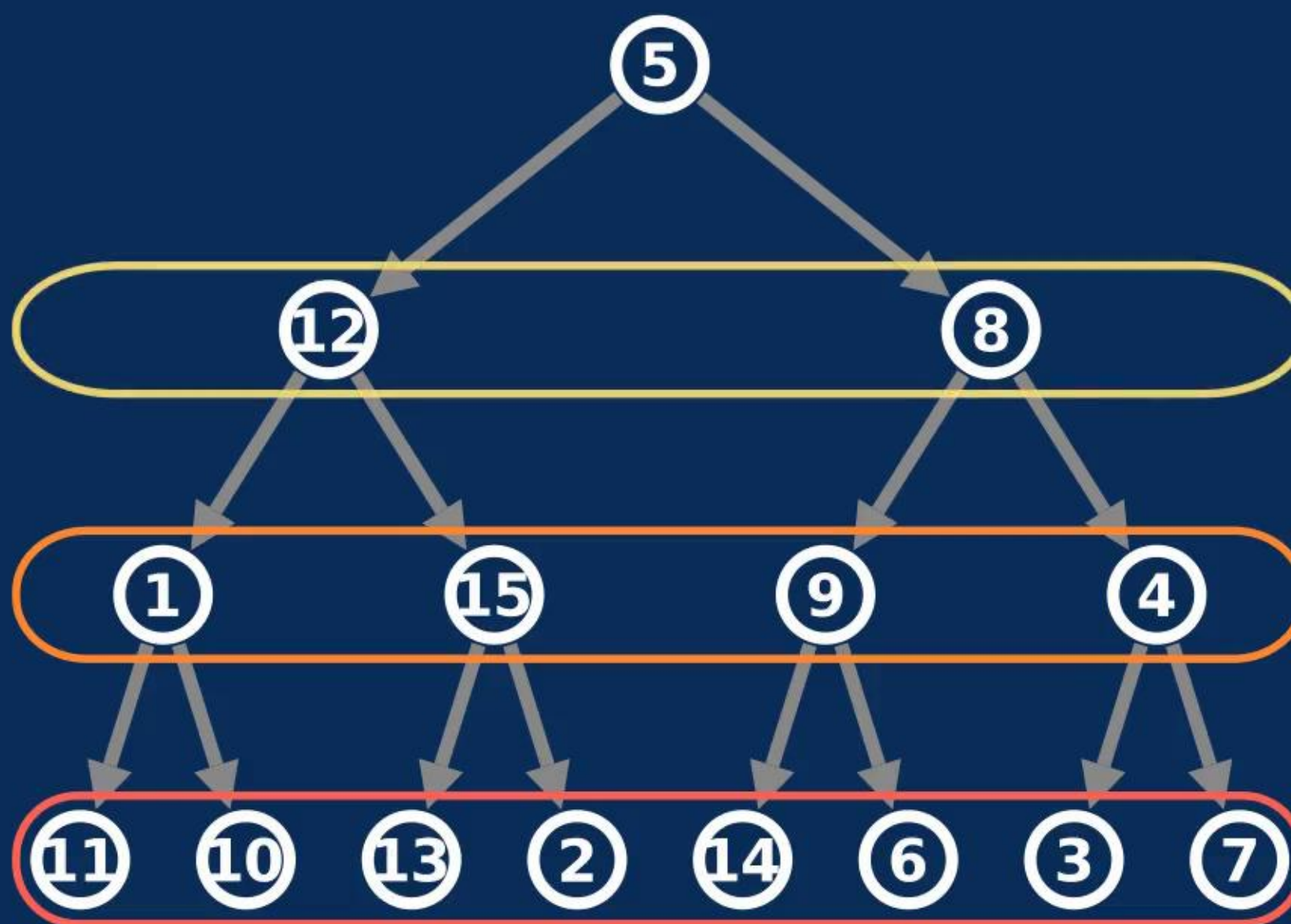


How long does this take?

$h = 3$ (leaves): Free!

$h = 2$: 1 each $\times 4$.

$h = 1$: 2 each $\times 2$.



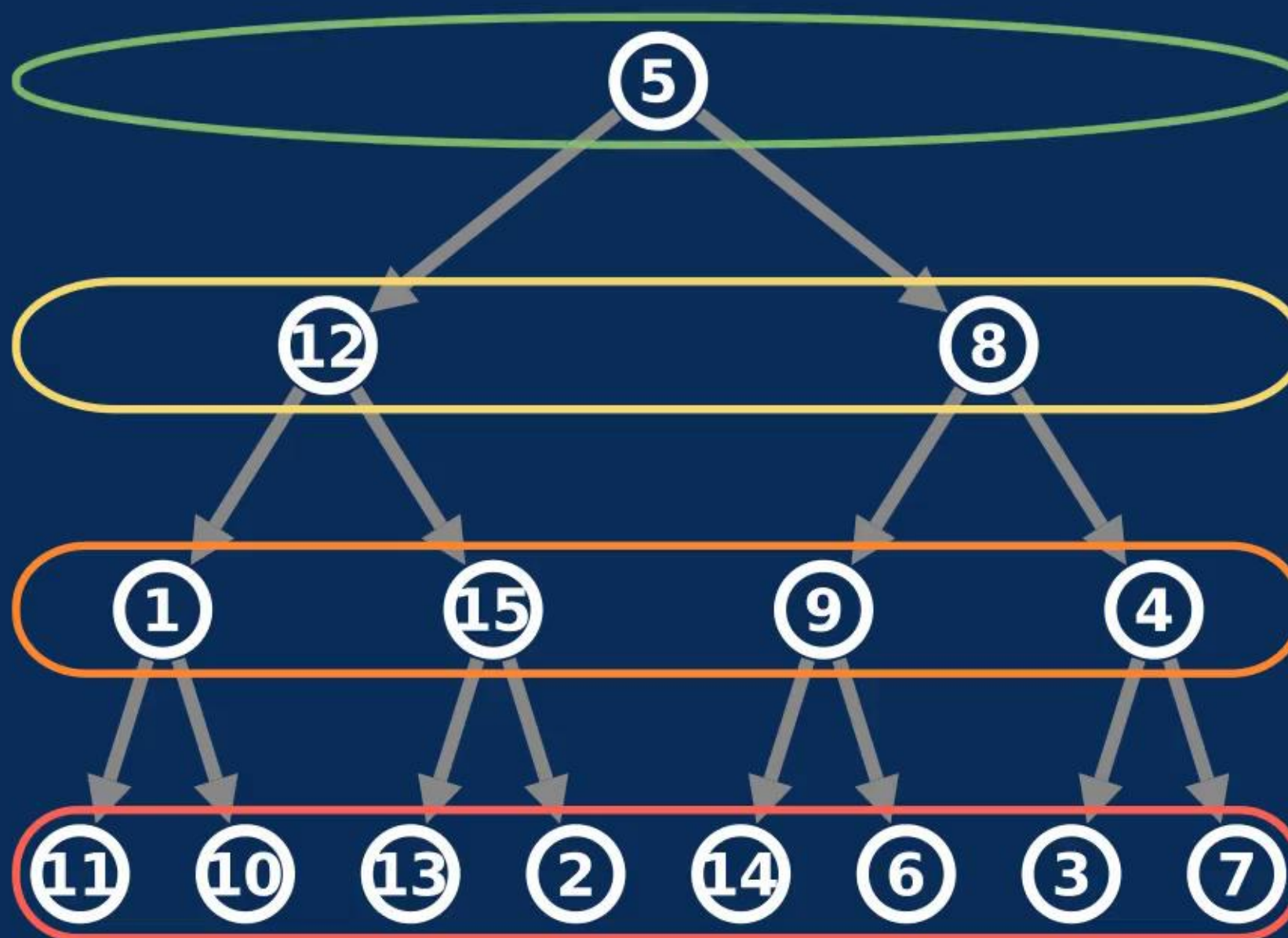
How long does this take?

$h = 3$ (leaves): Free!

$h = 2$: 1 each $\times 4$.

$h = 1$: 2 each $\times 2$.

$h = 0$: (root): 3.



How long does this take?

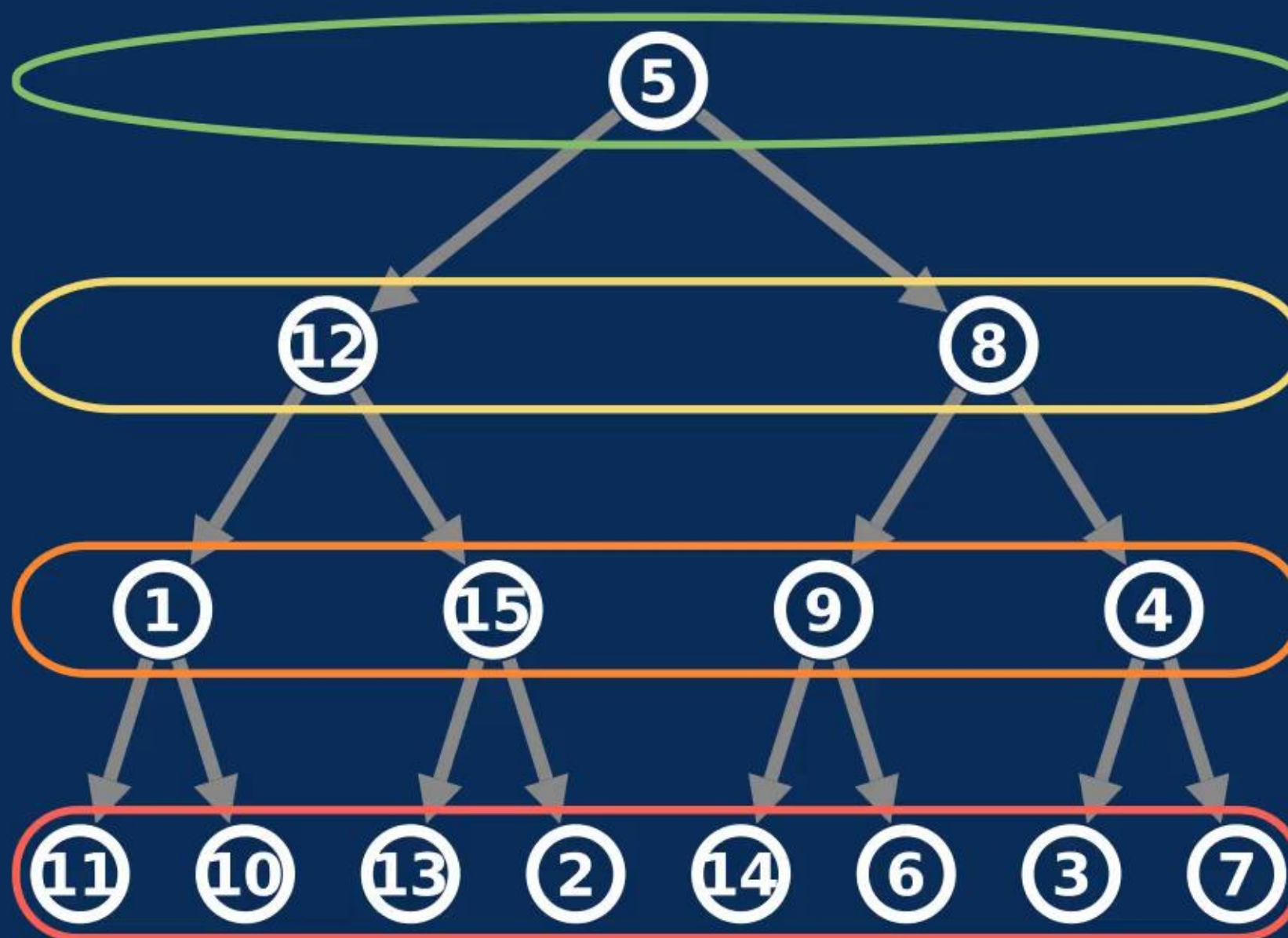
$h = 3$ (leaves): Free!

$h = 2$: 1 each $\times 4$.

$h = 1$: 2 each $\times 2$.

$h = 0$: (root): 3.

$$\text{Total} = \sum_{i=0}^h i \times n/2^{i+1}$$



How long does this take?

$h = 3$ (leaves): Free!

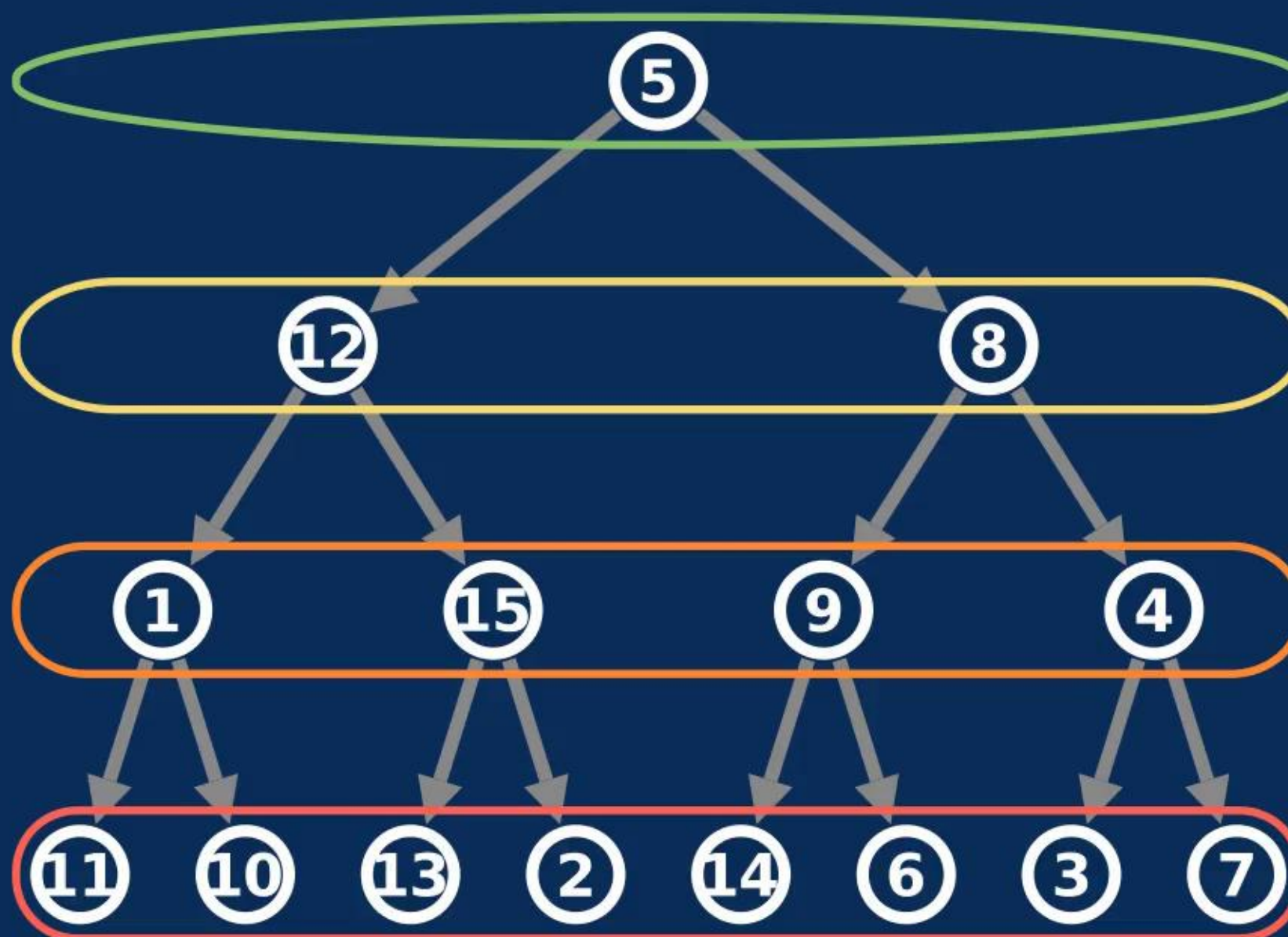
$h = 2$: 1 each $\times 4$.

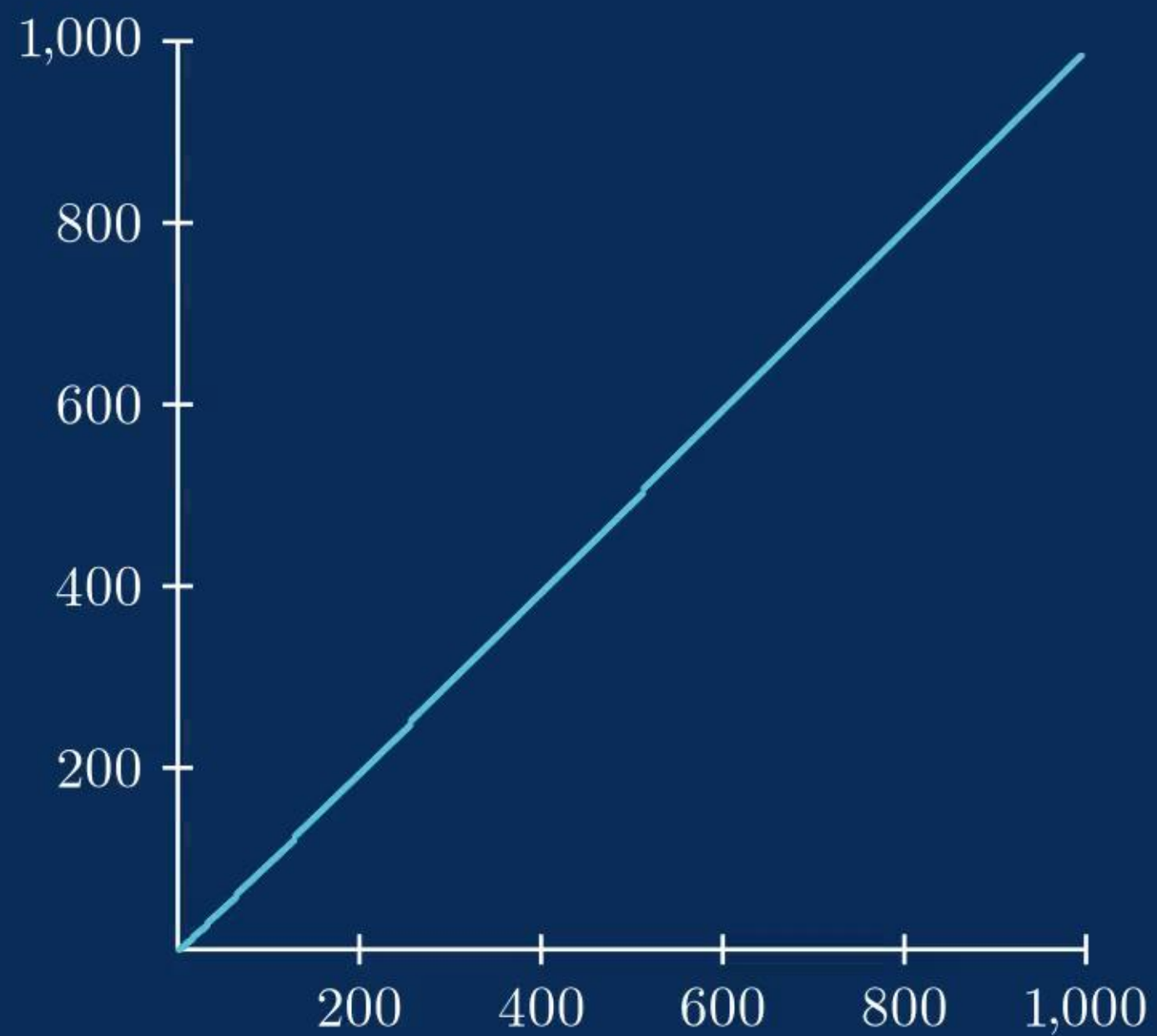
$h = 1$: 2 each $\times 2$.

$h = 0$: (root): 3.

$$\text{Total} = \sum_{i=0}^h i \times n/2^{i+1}$$

For this: 11 swaps.





One final trick: We can use a heap to sort things!

One final trick: We can use a heap to sort things!

How do we do it? And how long does it take?

One final trick: We can use a heap to sort things!

How do we do it? And how long does it take?

- Call Heapify on an array: $\mathcal{O}(n)$

One final trick: We can use a heap to sort things!

How do we do it? And how long does it take?

- Call Heapify on an array: $\mathcal{O}(n)$
- Remove n nodes: $n \times \log n$

One final trick: We can use a heap to sort things!

How do we do it? And how long does it take?

- Call Heapify on an array: $\mathcal{O}(n)$
- Remove n nodes: $n \times \log n$
- Total time: $\mathcal{O}(n \log n)$

One final trick: We can use a heap to sort things!

How do we do it? And how long does it take?

- Call Heapify on an array: $\mathcal{O}(n)$
- Remove n nodes: $n \times \log n$
- Total time: $\mathcal{O}(n \log n)$

Next time: Disjoint sets!

One final trick: We can use a heap to sort things!

How do we do it? And how long does it take?

- Call Heapify on an array: $\mathcal{O}(n)$
- Remove n nodes: $n \times \log n$
- Total time: $\mathcal{O}(n \log n)$

Next time: Disjoint sets!