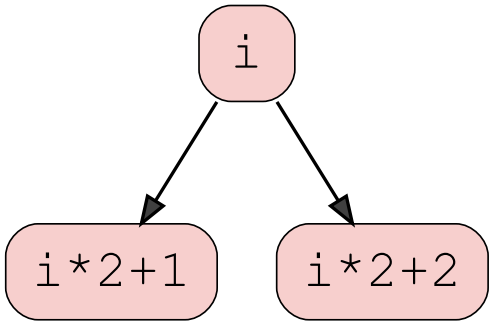
Heap Sort (Implementation)

(Reading time: 3 minutes)

To make a heap out of an array, we first need to go over all the items in the array, from right to left. This is necessary, as we start at the leaves. It receives the array we want to sort, and invokes the function that makes a heap on every element.

```
function makeHeap(arr) {
  const n = arr.length;
  for (let i = n - 1; i >= 0; i--) {
    // Heapify function.
  }
}
```

The heapify function should receive the array, the length of the array, and the current index. In order to make the heap, this order is very important to remember:



If we have a *maximum heap*, the node with the highest value should be on top. Next, we have the left node, and the right node.

```
function maxHeapify(arr, n, i) {
   let max = i;
```

```
   const left = 2 * i + 1;
   const right = 2 * i + 2;
}
```

By default, we make `i` the maximum value. However, if the value of i is not the maximum, we need to swap elements.

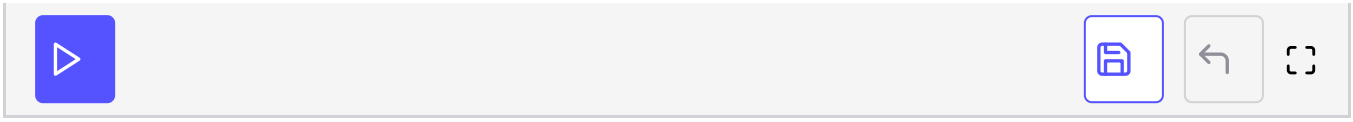
If there is a **left** or **right** value, and if the maximum value is smaller than the **left** or **right** value, we make the left or right value the maximum value. However, we're not actually swapping yet! We're just redeclaring the `max` variable! If the value of the `max` variable changed, the last if-statement returns true, and we swap the elements. The `maxHeapify` function gets called again, only this time for the values based on the **maximum** value. This way, we go through the array, and *maxHeapify* the entire array!

```
if (left < n && arr[max] < arr[left]) {
   max = left;
}

if (right < n && arr[max] < arr[right]) {
   max = right;
}

if (max !== i) {
   [arr[i], arr[max]] = [arr[max], arr[i]]
   maxHeapify(arr, n, max)
}
```

The entire function would look like:

```
function maxHeapify(arr, n, i) {
   let max = i;
   const left = 2 * i + 1;
   const right = 2 * i + 2;

   if (left < n && arr[max] < arr[left]) {
      max = left;
   }

   if (right < n && arr[max] < arr[right]) {
      max = right;
   }

   if (max !== i) {
      [arr[i], arr[max]] = [arr[max], arr[i]]
      maxHeapify(arr, n, max)
   }
}

function makeHeap(arr) {
  const n = arr.length;
  for (let i = n - 1; i >= 0; i--) {
     maxHeapify(arr, n, i);
```

```
  }
  return arr;
}
```



In the next lesson, I will talk about the time complexity of this algorithm.