Parent and Child Elements

7 min

Great work so far! Our MinHeap adds elements to the internal heap, keeps a running count, and has the beginnings of .bubbleUp().

Before we dive into the logic for .bubbleUp(), let's review how heaps track elements. We use an

Preview: Docs Loading link description

array

for storing internal elements, but we're modeling it on a

Preview: Docs Loading link description

binary

tree, where every parent element has up to two child elements.

Child and parent elements are determined by their relative indices within the internal heap. By doing some arithmetic on an element's

Preview: Docs Loading link description

index

, we can determine the indices for parent and child elements (if they exist).

- Parent: (index / 2), rounded down
- Left Child: index * 2
- Right Child: (index * 2) + 1

These calculations are important for the efficiency of the heap, but they're not necessary to memorize, so we have provided three helper functions: getParent(), getLeft(), and getRight() in **MinHeap.js**.

These helpers take an index as the sole

Preview: Docs Loading link description

parameter

and return the corresponding parent or child index.

```
console.log(myHeap.heap)
// returns [null, 10, 13, 21, 61, 22, 23, 99]

getParent(4); // returns (4 / 2) == 2

getLeft(3);   // returns (3 * 2) == 6
```

getRight(3); // returns (3 * 2) + 1 == 7

Instructions

1. Checkpoint 1 Passed

1.

In **script.js**, test out the three helper functions above. A sample populated MinHeap has been provided for you. Replace null with the correct way to access the values of the parent, left child and right child indices.

Hint

To access the value of an element in MinHeap based on its index, do the following:

```
const index = 7;
const value = minHeap.heap[index];
```

script.js

```
const { MinHeap, getParent, getLeft, getRight } = require('./MinHeap');


// instantiate MinHeap and assign to minHeap

const minHeap = new MinHeap();


// sample content of minHeap

minHeap.heap = [ null, 10, 13, 21, 61, 22, 23, 99 ];


// display content of minHeap

console.log(minHeap.heap);


// display the current value, its parent value, left child value and right child value

// replace null with the correct way to access the values of the parent, left child and right child

const current = 3;

const currentValue = minHeap.heap[current];

console.log(`Current value of ${current} is ${currentValue}`);
```

```
console.log(`Parent value of ${currentValue} is ${minHeap.heap[getParent(current)]}`);
console.log(`Left child value of ${currentValue} is ${minHeap.heap[getLeft(current)]}`);
console.log(`Right child value of ${currentValue} is ${minHeap.heap[getRight(current)]}`);
```

MinHeap.js

```
class MinHeap {
  constructor() {
    this.heap = [ null ];
    this.size = 0;
  }

  add(value) {
    this.heap.push(value);
    this.size++;
    this.bubbleUp();
    console.log(this.heap);
  }

  bubbleUp() {
    console.log('Bubble Up');
  }
}

const getParent = current => Math.floor((current / 2));
const getLeft = current => current * 2;
const getRight = current => current * 2 + 1;

module.exports = {
  MinHeap,
```

```
  getParent,
  getLeft,
  getRight
};
```

>> Out

```
[ null, 10, 13, 21, 61, 22, 23, 99 ]
Current value of 3 is 21
Parent value of 21 is 10
Left child value of 21 is 23
Right child value of 21 is 99
```