Min Heap (Implementation)

We'll implement a min heap in this lesson!

We'll cover the following

Implementation

Now that we have discussed all the functions of a Min-Heap, we've implemented them in the following code executable. Run and test the code on multiple outputs to see if it returns the elements in the correct order every time? Try it!

MinHeap.py

```
class MinHeap:
    def __init__(self):
        self.heap = []

    def insert(self, val):
        self.heap.append(val)
        self.__percolateUp(len(self.heap)-1)

    def getMin(self):
        if self.heap:
            return self.heap[0]
        return None

    def removeMin(self):
        if len(self.heap) > 1:
            min = self.heap[0]
            self.heap[0] = self.heap[-1]
            del self.heap[-1]
            self.__minHeapify(0)
            return min
        elif len(self.heap == 1):
            min = self.heap[0]
            del self.heap[0]
            return min
        else:
            return None

    def __percolateUp(self, index):
        parent = (index-1)//2
        if index <= 0:
            return
        elif self.heap[parent] > self.heap[index]:
            tmp = self.heap[parent]
            self.heap[parent] = self.heap[index]
            self.heap[index] = tmp
            self.__percolateUp(parent)

    def   minHeapify(self, index):
```

```
        left = (index * 2) + 1
        right = (index * 2) + 2
        smallest = index
        if len(self.heap) > left and self.heap[smallest] > self.heap[left]:
            smallest = left
        if len(self.heap) > right and self.heap[smallest] > self.heap[right]:
            smallest = right
        if smallest != index:
            tmp = self.heap[smallest]
            self.heap[smallest] = self.heap[index]
            self.heap[index] = tmp
            self.__minHeapify(smallest)

    def buildHeap(self, arr):
        self.heap = arr
        for i in range(len(arr)-1, -1, -1):
            self.__minHeapify(i)


heap = MinHeap()
heap.insert(12)
heap.insert(10)
heap.insert(-10)
heap.insert(100)

print(heap.getMin())
print(heap.removeMin())
print(heap.getMin())
heap.insert(-100)
print(heap.getMin())

```

Output 0.155s

```
-10
-10
10
-100
```

Explanation

The code above for min heaps is an exact reflection of the code for max heaps! It's a good exercise to try and figure out what changed. However, if you have any confusion about it, leave a question on the community forum and we'll get back to you!

And now that we have covered both the implementations, let's try to solve some practice questions using the Heap data structure in the next few lessons!