



Min Heap (Implementation)

We'll implement a min heap in this lesson!

We'll cover the following ^

- Implementation
- Explanation

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

```
1 class MinHeap:
2     def __init__(self):
3         self.heap = []
4
5     def insert(self, val):
6         self.heap.append(val)
7         self.__percolateUp(len(self.heap)-1)
8
9     def getMin(self):
10        if self.heap:
11            return self.heap[0]
12        return None
13
14    def removeMin(self):
15        if len(self.heap) > 0:
```



```
15         if len(self.heap) > 1:
16             min = self.heap[0]
17             self.heap[0] = self.heap[-1]
18             del self.heap[-1]
19             self.__minHeapify(0)
20             return min
21         elif len(self.heap) == 1:
22             min = self.heap[0]
23             del self.heap[0]
24             return min
25         else:
26             return None
27
28     def __percolateUp(self, index):
```



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!

Interviewing soon? We've partnered with Hired so that companies apply to you instead of you applying to them. [See how](#) ⓘ

[← Back](#)[Next](#)



 Report an Issue

