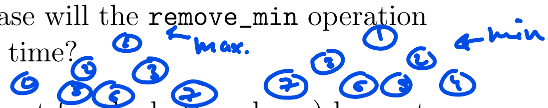
2.1. Let the binary heap contain numbers from 1 to 1000, once each. What is the smallest number that can be at the lowest level in the heap? 512

2.2. Let the binary heap contain n elements. How many leaves does the corresponding tree have? n - floor(log2(n))

2.3. Let the heap contain numbers from 1 to n, once each. In which case will the `remove_min` operation work for the minimum time, and in which case for the maximum time?



2.4. Let the heap tree be organized in such a way that each node (except for the bottom layer) has not two children, but three. What indices will the children of the node i have in this case? $3i, 3i+1, 3i+2$

2.5. Add operation `change_key(node, value)` to the binary heap, which changes the key of the given node in $O(\log n)$ time. if (h[node] < value) sift-down(node); else sift_up(node);

2.6. How to make a data structure out of two binary heaps that can simultaneously find and remove both the maximum and the minimum elements?

2.7. Based on the binary heaps, make a data structure that can find and remove the median element ($n/2$ element in sorted order).

2.8. Peter wanted to build a heap in $O(n)$ time, but he did it not quite right:

```
for i = 0 .. n - 1:
    sift_down(i)
```

Show that this algorithm sometimes does not work.