9.1. How many vertices of rank r are there in a binomial tree of rank k?

9.2. Develop a binomial heap merging algorithm that goes from large trees to smaller ones, with the same time complexity.

9.3. One item is added to the binomial heap of size n. How to quickly figure out how many binomial tree merges will happen?

9.4. Show that the amortized cost of the `insert` operation in binomial heap is $O(1)$ (assuming the amortized cost of the remaining operations is $O(\log n)$).

9.5. How can the binomial heap be modified so that `insert` runs in true $O(1)$? (google the Knuth counter)

9.6. Add a key increment operation to the binomial heap, in $O(\log n)$. Is it possible to make the key increment operation faster than $O(\log n)$?

9.7. Prove that the maximum rank of a tree in the Fibonacci heap is $O(\ logn)$.

9.8. Show that for any n there is a sequence of operations that results in the Fibonacci heap containing a tree that is a chain of n elements.