1. *Empirical analysis*. The following table gives approximate running times for a program with $N$ inputs, for various values of $N$.

<table>
<thead>
<tr>
<th>N</th>
<th>time</th>
</tr>
</thead>
<tbody>
<tr>
<td>1000</td>
<td>10 seconds</td>
</tr>
<tr>
<td>2000</td>
<td>40 seconds</td>
</tr>
<tr>
<td>5000</td>
<td>~4 minutes</td>
</tr>
</tbody>
</table>

Predict its running time (in minutes) for $N = 10,000$ and give a formula that estimates the running time as a function of $N$. 

2. **Worst-case input for weighted quick-union.** A binomial tree is defined recursively: a binomial tree of order 0 consists of a single node; a binomial tree of order $h$ is a tree obtained from two binomial trees of order $h-1$, by linking the root of one to the other. Below are binomial trees of order 0, 1, 2, 3, and 4.

![Binomial Trees](image)

(a) How many nodes are in a binomial tree of order $h$?

(b) And what is the height of a binomial tree of order $h$?

(c) What is the minimum number of `union()` operations (using the weighted quick-union algorithm) that produces a binomial tree of order $h = 3$.

(d) What is the worst case number of array accesses of `find()` on a binomial tree, as a function of its number of nodes $N$?

```java
public int find(int p) {
    while (p != id[p])
        p = id[p];
    return p;
}
```