A **Binomial tree** $B_k$ of degree $k$ is defined recursively as follows.

$$B_0 = \emptyset$$

$$B_k = B_{k-1} \cup B_k$$

Note that the order of the child nodes are significant.

**Properties of a Binomial Tree**

1. $B_k$ has $2^k$ nodes.
2. $B_k$ has height $k$.
3. $B_k$ has $\binom{k}{i}$ at depth $i = 0, \ldots, k$.
4. Root of $B_k$ has degree $k$ which is maximum over all nodes.
5. The subtree rooted at the $i$-th child of the root of $B_k$ is a binomial tree $B_i$.

$\Rightarrow$ Maximum degree in an $n$-node binomial tree is $\lg n$.

**A Binomial Heap** is a set of binomial trees such that

1. each binomial tree is heap-ordered,
2. there is at most one $B_k$ for a given $k$.

$\Rightarrow$ The binomial heap for $n$ items contains one binomial tree for each 1-bit in the binary representation of $n$. If the $i$-th bit is set, then the corresponding binomial tree is $B_i$, where the least significant bit occurs at $i = 0$.

**Example.** The binomial heap for 13 items (binary 1101) is shown below.

The root nodes of all the binomial trees in the binomial heap are chained together in a doubly-linked circular list called the *root list*. The children of each node are also chained together in a doubly-linked circular list to facilitate merging.

**Operation**

- **Minimum($H$)** Iterate through the root list to find the minimum root. Root list has at most $\lg n$ nodes.
- **Union($H_1, H_2$)** Merge the two root list in order of root degrees. Iterate through merged root list and merge the binomial trees analogous to binary addition. Merging two binomial trees take $O(1)$ time. There are $O(\lg n_1 + \lg n_2) = O(\lg(n_1 + n_2))$ trees to merge.
- **Insert($H, x$)** Make $x$ a single node binomial heap and union with $H$.
- **ExtractMin($H$)** Find Minimum($H$), remove minimum root, make its children into a new binomial heap, and union with $H$.
- **DecreaseKey($H, x, k$)** Update $x.key$ to $k$, and bubble $x$ up to maintain heap order.
- **Delete($H, x$)** Decrease key of $x$ to $-\infty$, and extract minimum.

<table>
<thead>
<tr>
<th>Operation</th>
<th>Binary Heaps</th>
<th>Binomial Heaps</th>
<th>Fibonacci Heaps</th>
</tr>
</thead>
<tbody>
<tr>
<td>Insert</td>
<td>$O(\lg n)$</td>
<td>$O(\lg n), O(1)^A$</td>
<td>$O(1)$</td>
</tr>
<tr>
<td>Minimum</td>
<td>$O(1)$</td>
<td>$O(\lg n)$</td>
<td>$O(1)$</td>
</tr>
<tr>
<td>ExtractMin</td>
<td>$O(\lg n)$</td>
<td>$O(\lg n)$</td>
<td>$O(\lg n)^A$</td>
</tr>
<tr>
<td>Delete</td>
<td>$O(\lg n)$</td>
<td>$O(\lg n)$</td>
<td>$O(\lg n)^A$</td>
</tr>
<tr>
<td>DecreaseKey</td>
<td>$O(\lg n)$</td>
<td>$O(\lg n)$</td>
<td>$O(1)^A$</td>
</tr>
<tr>
<td>Union</td>
<td>$O(n)$</td>
<td>$O(\lg n), O(1)^A$</td>
<td>$O(1)$</td>
</tr>
</tbody>
</table>