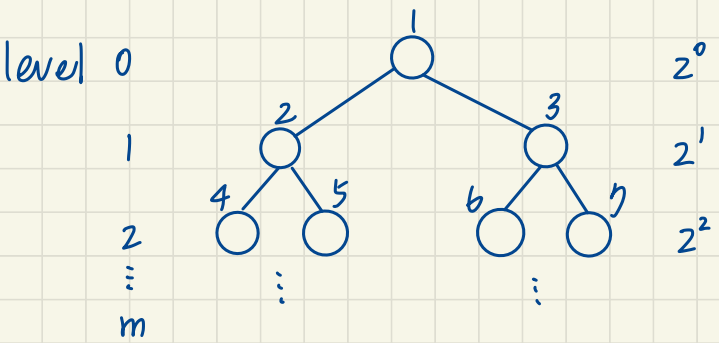
4) The worst-case number of comparisons performed during an insertion into a max heap can be reduced to O(loglog *n*) by performing a binary search on the path from the new leaf to the root. This does not affect the number of data moves though. Write an insertion algorithm that uses this strategy. Based on your experiments, what can you say about the value of this strategy over the one used in following program?



$$n = 2^0 + 2^1 + \dots + 2^m = \frac{2^{m+1}-1}{2-1} = 2^{m+1}-1$$

$$\Rightarrow \log n = \log(2^{m+1}-1)$$

$$\Rightarrow m = \log n$$

change the search method into binary search.

$$\Rightarrow \#\text{ of search} = \lfloor \log m \rfloor = \lfloor \log \log n \rfloor \therefore O(\log \log n).$$

$\Downarrow$

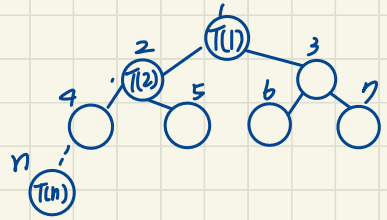
the origin strategy is compare each elements

$$\#\text{ of search} = m = \log n. \therefore O(\log n)$$

To sum up. this strategy reduce the complexity of search from $O(\log n)$ to $O(\log \log n)$. however, this doesn't affect the number of move though

$\therefore$ total complexity reduce to $O(\log \log n)$

pseudocode.



Input: the insertion data

Output: new max heap

1 compare the insertion data from $\frac{n}{2}$. and next every time always devide 2. $\lfloor \frac{n}{2} \rfloor$

if $(T(n) < T(\frac{n}{2}))$

compare $T(n)$ and $T(\frac{3}{4}n)$ and so on

if $(T(n) > T(\frac{n}{2}))$

compare $T(n)$ and $T(\frac{1}{4}n)$ and so on.

2 while $(T(n) \geq T(X-1)$ & $T(n) < T(X+1))$

change the insertion data to the place final place X.