Big-O Notation

a)
```
int sum=0;
for (int k=n; k>0; k/=2)
for(int i=0; i<k; i++)
  sum++;
```

$O(n \log(n))$

b)
```
int sum=0;
for (int i=1; i<n; i*=2)
for (int j=0; j<i; j++)
  sum++;
```

$O(\log(n))$

c)
```
int sum=0;
for(int i=1; i<n; *=2)
for(int j=0; j<n; j++)
  sum++;
```

$O(n \log(n))$

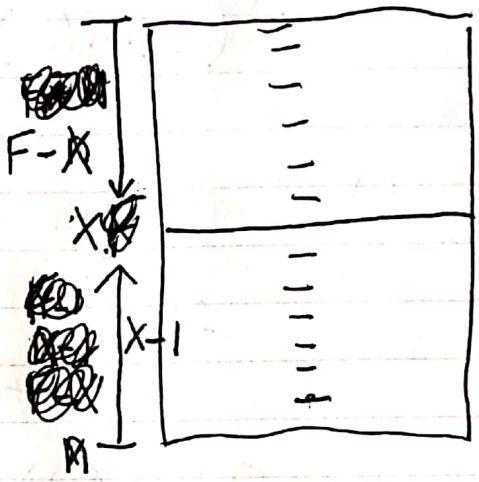
Egg Throwing

F = __ and uses $O(\log(n))$

$N(e,t) = F$

e = eggs
t = tries
F = floor that break
X-1 = safe
F-X = broken
N = number of...

If egg breaks:
(e-1) → One less egg
(t-1) → One less try
$N(e-1,t-1) = n-1 \Rightarrow (1)$



If egg doesn't break
(e) → same amount of eggs
(t-1) → one less try

$N(e,t-1) = F-n \Rightarrow 2$

If we add 1 and 2 together we can get of answer

$N(e-1,t-1) + N(e,t-1)$

$= (n-1) + (F-n) \Rightarrow F-1 = N(e,t)-1$

$N(e,t) = 1 + N(e-1,t-1) + N(e,t-1)$

$\Rightarrow 2(\log(F))$ using this we can reduce the number of broken eggs