Solution to Problem 2 of handout for Section 3

In the base case, we just return, so the work is constant. In non-base cases, we iterate over the nested loops and then make 2 recurrsive calls.

$$T(n) = \begin{cases} c_0 & n = 0 \\ 2T(n/2) + c_2 \underbrace{\frac{n^2 - n}{2}} + c_1 & n > 0 \end{cases}$$

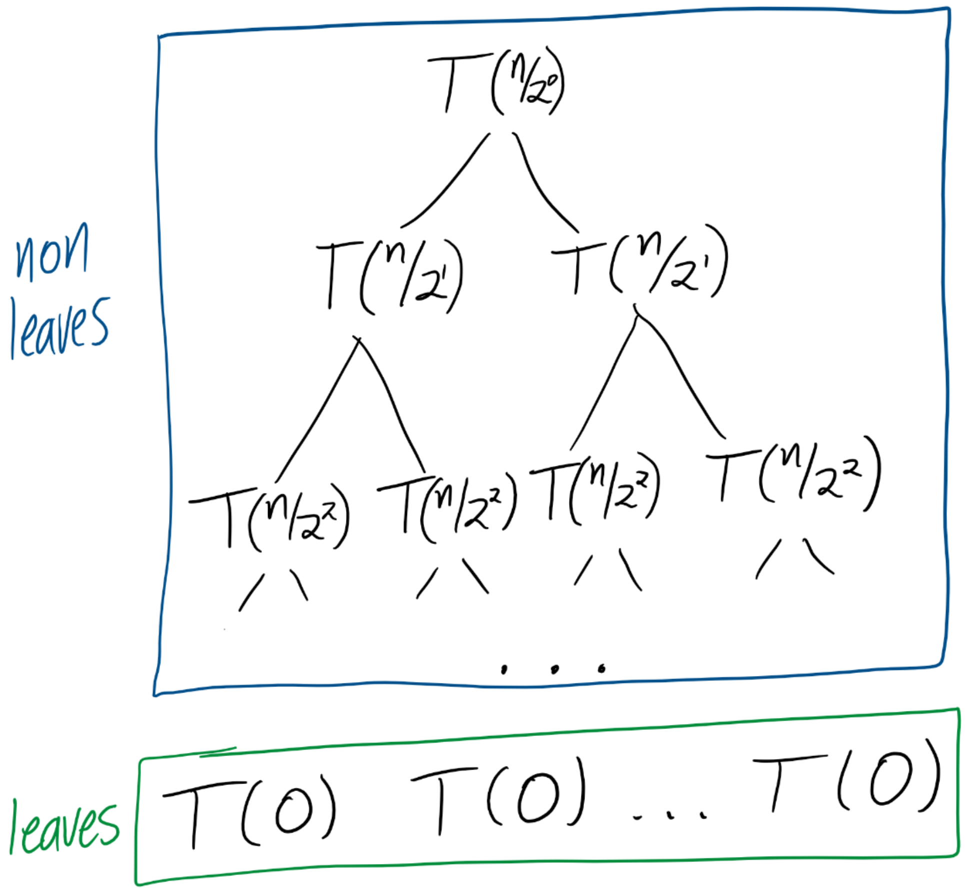
"otherwise" also works as we are not expecting negative input (n usually represents size of input)

See summations sheet

$$\boxed{\sum_{i=0}^{n-1} i = \frac{n(n-1)}{2}}$$

outer loop

inner loop



Level	number of nodes	work at each node
0	$1=2^0$	$\frac{n^2-n}{2}c_2+c_1$
1	$2=2^1$	$\frac{(n/2)^2-(n/2)}{2}c_2+c_1$
2	$4=2^2$	$\frac{(n/4)^2-(n/4)}{2}c_2+c_1$
	⋮	⋮
$\lfloor \lg(n) \rfloor$	$2^{\lfloor \lg(n) \rfloor}$	c_0

so we get:

because level lg(n) is leaves, this adds up non-leaf work

$$\underbrace{\sum_{i=0}^{\lg(n)-1}}_{\text{each level of the tree}} \underbrace{2^i}_{\text{number of nodes at that level}} \underbrace{\left(\frac{(n/2^i)^2-(n/2^i)}{2}c_2+c_1\right)}_{\text{work at each node at that level}} + \underbrace{c_0 \cdot 2^{\lg(n)}}_{\text{leaf work}}$$

but we're looking for Big-O, not exact closed form, so we can make life a little easier by turning $\left(\frac{(n/2^i)^2 - (n/2^i)}{2} c_2 + c_1\right)$ into $(n/2^i)^2$.

only keep this

$$\sum_{i=0}^{\lg(n)-1} 2^i \left(\frac{n}{2^i}\right)^2 + c_0 2^{\lg(n)}$$

$$= \sum_{i=0}^{\lg(n)-1} 2^i \cdot \frac{n^2}{2^{2i}} + c_0 n$$

$$= \sum_{i=0}^{\lg(n)-1} n^2 \cdot \frac{2^i}{4^i} + c_0 n$$

$$= n^2 \sum_{i=0}^{\lg(n)-1} \left(\frac{2}{4}\right)^i + c_0 n$$

Infinite Geometric Series is easier to deal with than Finite Geometric Series

$$\leq n^2 \boxed{\sum_{i=0}^{\infty} \left(\frac{1}{2}\right)^i} + c_0 n$$

we can do this because we want Big-O, for which we need to show less than or equal to

$$= n^2\left(\frac{1}{1-\frac{1}{2}}\right) + c_0 n$$

$$= n^2\left(\frac{1}{\frac{1}{2}}\right) + c_0 n$$

$$= n^2 \cdot 2 + c_0 n$$

this is all we care about for Big-O,

so it's $O(n^2)$!