Tyler Huynh

Q6 S16

<u>Lines of code:</u>

line 2 = 1 step to print ("hello")
3 = 2 steps for the assignment of n and calling the function len.
4 = 1 step for $n \leq 1$, 1 step for returning L (Base Case)
5 = 1 step for assignment, 1 step to split list
6 = same a line 5, 2 steps
7 = This line is calling recursively the list that has $n/2$, such that $T(\frac{n}{2})$.
8 = This line is calling recursively the list that has $n/2$, such that $T(\frac{n}{2})$.
9 = This line has 2 steps as 1 is for the assignment, and another is for the appending of L_1 and L_2.
10 = 1 step to return L_1

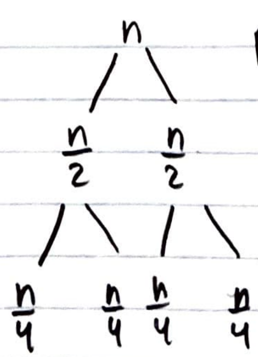
<u>Total runtime complexity of T(n):</u>

Base Case = $\Theta(1)$
Recursive Case = $2T(\frac{n}{2}) + \Theta(1)$

* The other steps result in being $\Theta(1)$, because it is a constant that is not determined by the size of the list.

$$T(n) \begin{cases} \Theta(1) & n \leq 1 \\ 2T(\frac{n}{2}) + \Theta(1) & n > 1 \end{cases}$$

<u>Finding how many times "hello" will be printed:</u>



print "hello" 2^0

2 print "hello" 2^1

4 print "hello" 2^2

k will represent the number of levels.

$$\frac{n}{2^k} \leq 1$$

$$n \leq 2^k$$

$$\log_2 n \leq k$$

For each level of k, hello will be printed 2^k times.

$$\sum_{i=0}^{\log_2 n} 2^i \quad \text{Geometric series}$$

$$= \frac{1-2^{\log_2 n+1}}{1-2}$$

$$= \frac{1-(2^{\log_2 n})(2)}{-1}$$

$$= \frac{1-(2n)}{-1}$$

$$= -1+2n$$

$$= 2n-1$$

"hello" will be printed $\underline{\underline{2n-1}}$ times.