Ahmed Yousry Elghoneimy 700324

Question 1

(b) for the naïve interative Method involves looping n times performing multiplication n times to compute a^n. so the time complexity is ϴ(n)

In divide and conquer we can use recursion as

$$T(n)=T(n/2) +O (1)$$

as the problem divides each step and a constant amout of work is preformed at each step that represent O (1) as a=1 as we call function once with the same prameter but we multiply it twice with the result

If we want to solve recurrence relation, we can use recuuence tree method:

Level	Tree	nodes	costatlevel
0	c	1	c
1	c/2	1	c/2
2	c/4	4	c/4
.h	O(1) O(1) O(1) O(1) O(1) O(1)	n	c/2^h

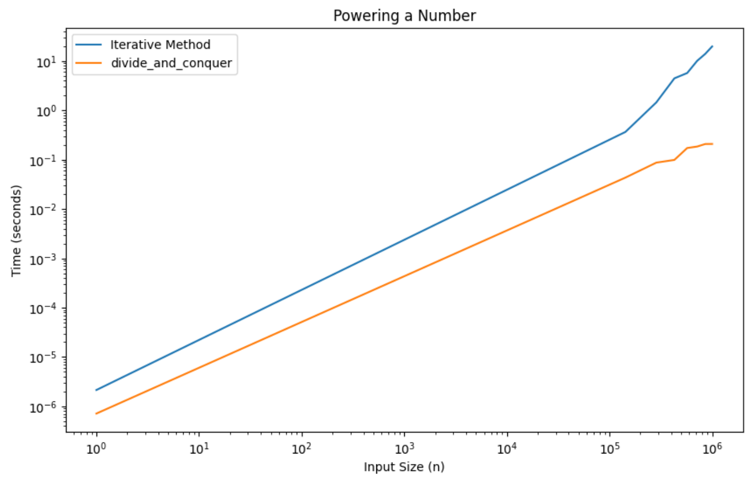
Numer of level h = log2 (n)

Cost of all leaves = $\sum_{i=0}^{log2(n)-1} a^i\, f(n/b) + log2(n)$ = $\sum_{i=0}^{log2(n)-1}$ $c/2^\wedge i$ + log2(n) =

By using gemotric series = 2c(1-(1/n)) + log2(n) as log2(n) is higher than (-1/n) as decay of (-1/n) faster

So the ϴ(log(n))

c-



I ran the graph function on Colab

As you can see the the effect on time when we use divide and conquer approach as we increase size of n iterative Method is O(n) and divide and conquer is O(log(n))

(d) as illustrated in the graph the shape of in divide and conquer the shape of function tends to be logrithmic (log(n)) as we and it is less than the iterative Method as it appears to be O(n)

Question 2

Merge sort =T(n)= 2T(n/2)+O(n)

O(n) for merge

Level	Tree	nodes	costatlevel
0	n	1	n
1	n/2 n/2	2	2(n/2)
2	n/4 n/4 n/4 n/4	4	
.h	O(1) O(1) O(1) O(1) O(1) O(1)	n	2^i(n/2^i)

Numer of level h = log2 (n)

Cost of all leaves = $\sum_{i=0}^{log2(n)-1} a^i\ f(n/b) + log2(n)$ = $\sum_{i=0}^{log2(n)-1}\ 2\hat{\ }i(n/2\hat{\ }i)$ + nlog2(n) =

By using gemotric series it will ends by Θ(n log2(n))

And for binary search T(n)= T(n/2)+O(1)

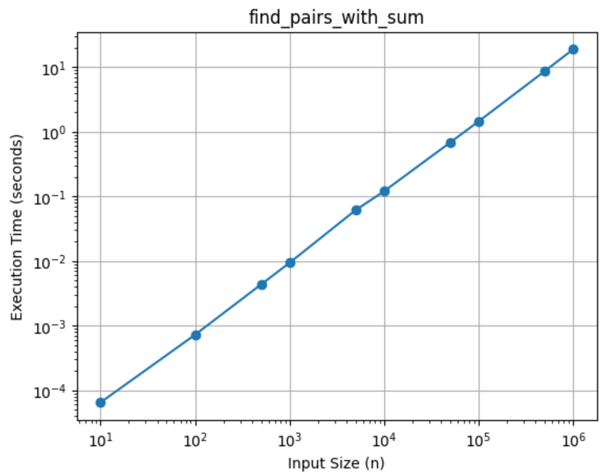
Numer of level h = log2 (n)

Cost of all leaves = $\sum_{i=0}^{log2(n)-1} a^i\ f(n/b) + log2(n)$ = $\sum_{i=0}^{log2(n)-1}\ c/2\hat{\ }i$ + log2(n) =

By using gemotric series = 2c(1-(1/n)) + log2(n) as log2(n) is higher than (-1/n) as decay of (-1/n) faster

So the Θ (log(n))

So over all Θ(nlog(n))



(c) as illustrated in the graph the shape of in divide and conquer the shape of function tends to be n log(n) as the calculated before