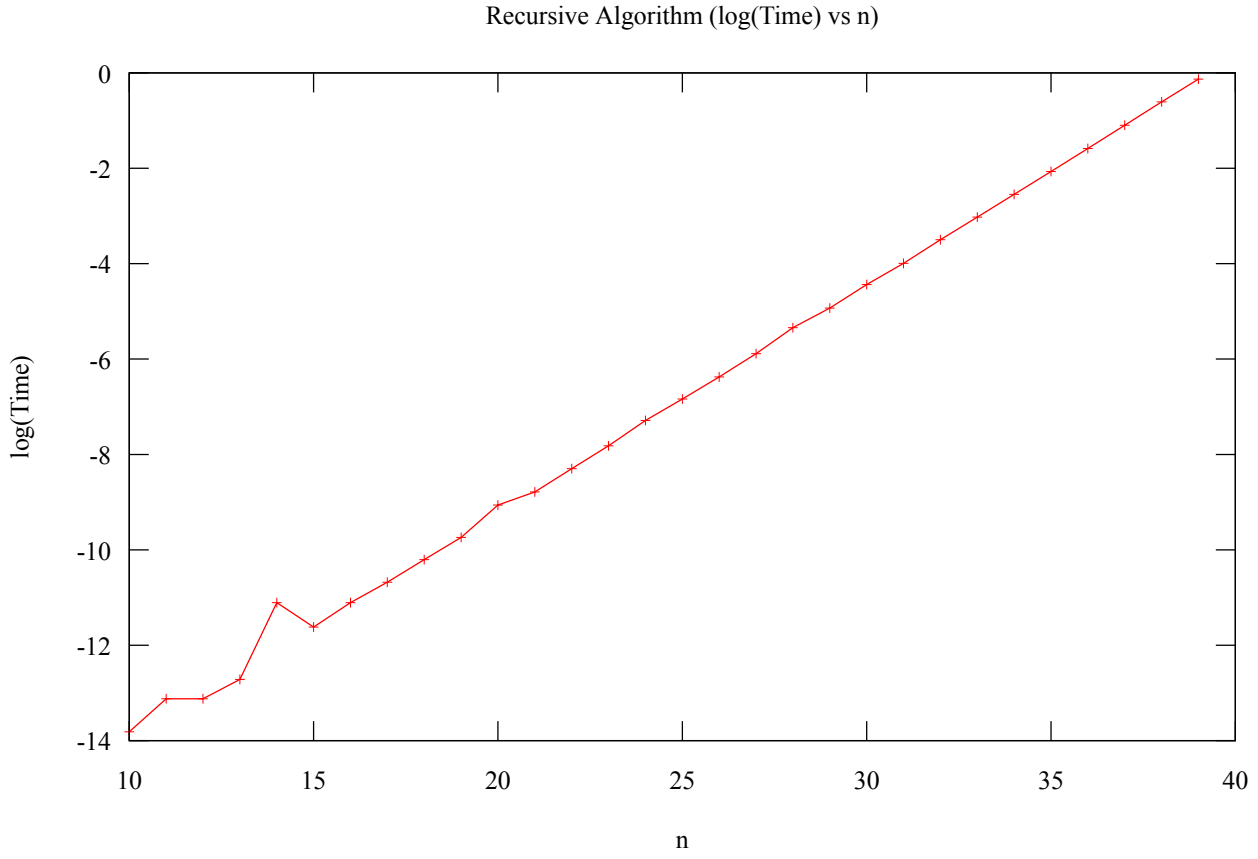
1.

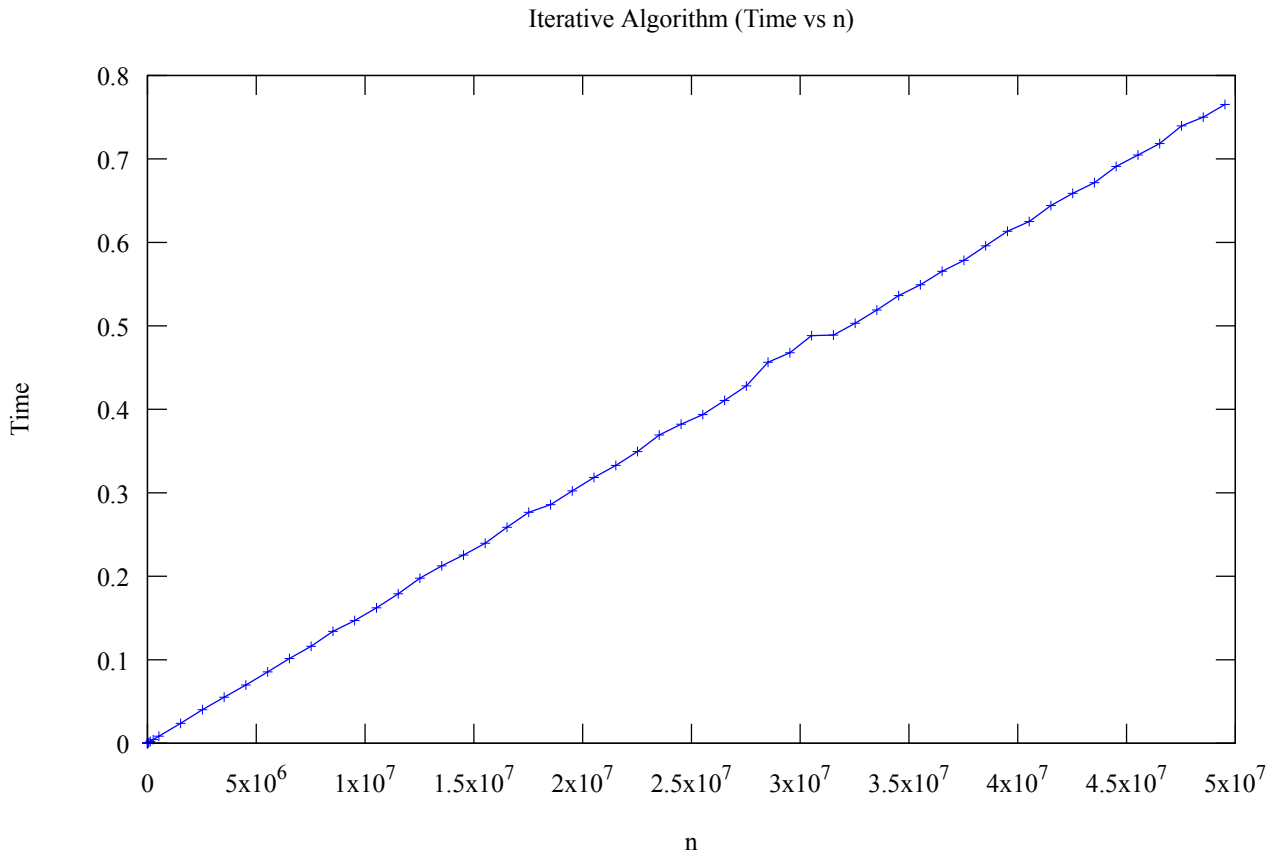
Time taken to compute G(n) mod m (in seconds)	10^{-5}	10^{-4}	10^{-3}	10^{-2}	10^{-1}	1	10
Max value of Recursive Algorithm	15	21	25	30	40	45	50
Max value of Iterative Algorithm	550	7210	65536	554288	6494288	64524288	591234288
Max value of Matrix Method	26955000000000001	$>10^{18}$	$>10^{18}$	$>10^{18}$	$>10^{18}$	$>10^{18}$	$>10^{18}$

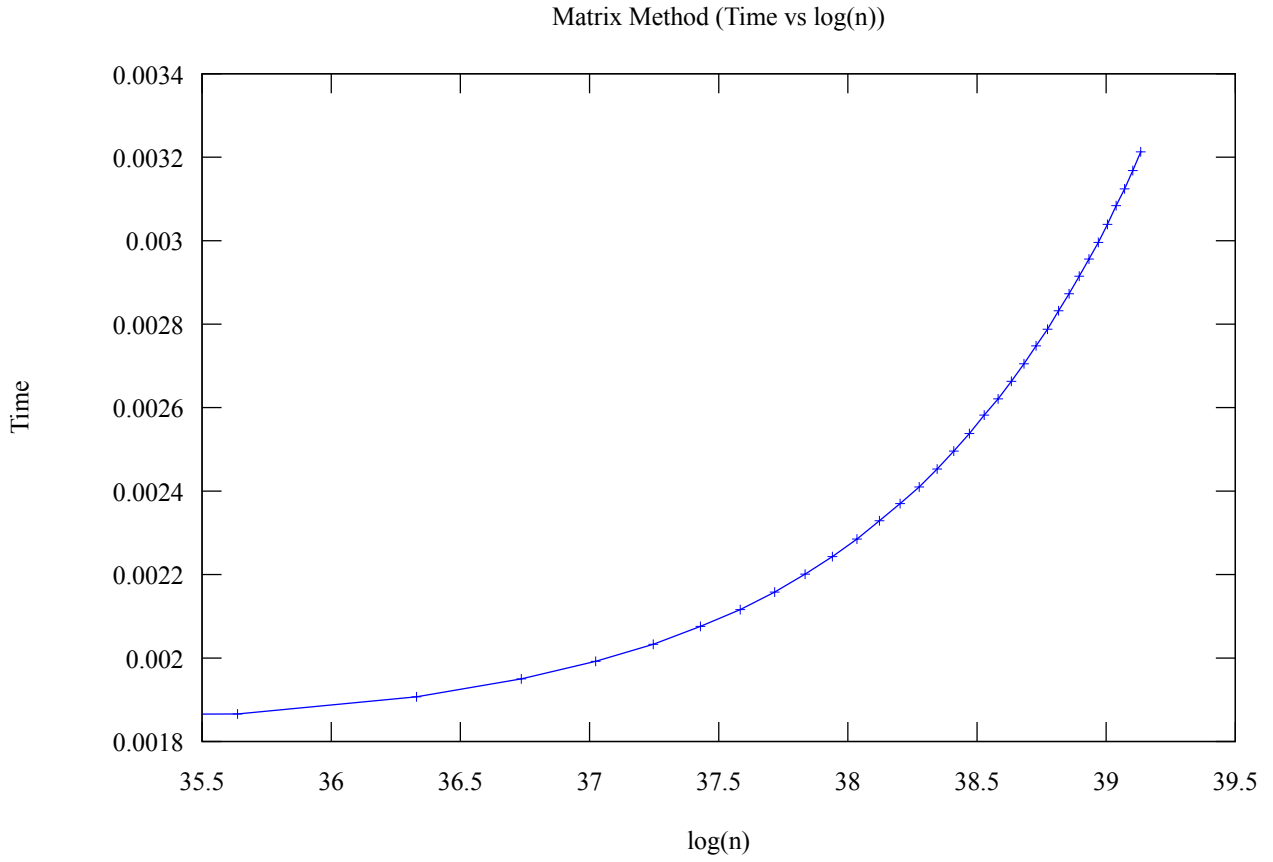
2. PLOTS OF ALGORITHMS



Values Used for Arguments -

```
a = 1
b = 1
c = 0
m = 1000000007
```





3.

Since the recursive algorithm calls for both G(n-1) and G(n-2) in its recursive step, many computations get repeated, and thus the complexity also increases. The complexity of the recursive algorith is $O(2^n)$ and hence, it's execution time is very large, and log of execution time is proportional to n.

The iterative algorithm just iterates over and keeps on adding the previous two values (with some factor), and thus is a linear algorithm. It's complexity is O(n) and is, therefore more efficient than the recursive approach, but it still doesn't work for large values of n ($>10^8$)

The matrix method is the fastest among these with a complexity of O(log(n)). Therefore, log(n) is proportional to execution time. Therefore, it can easily compute G(n) for very large n ($>10^{18}$) is less than one second.

Hence order of efficiency is
Recursive < Iterative < Matrix