GATE 2006, Question Number 15

$\boxed{n/2^k > 0}$

Complexity

$n/2^k = 1$

$\boxed{n = 2^k}$ $\boxed{1 > 0}$

$i = n,\ j = 0$

$\boxed{\text{Series}}$

- $i = n/2$ $\quad j = 0 + n/2 \leftarrow$ (I) iteration
- $i = n/2^2$ $\quad j = 0 + n/2 + n/2^2$ – (II) iteration
- $i = n/2^3$ $\quad j = 0 + n/2 + n/2^2 + n/2^3$ – III iteration

Last iteration $\leftarrow$ $i = n/2^k$ $\quad j = n/2 + n/2^2 + n/2^3 + \cdots + n/2^k$

GATE 2006, Question Number 15

value approximated $\underline{\Theta(n)}$

2^K

$j = n/2 + n/2^2 + n/2^3 + n/2^4 + \dots + n/2^k$

$= n\left[1/2 + 1/2^2 + 1/2^3 + \dots + 1/2^k\right]$

$= n\left[\dfrac{\cancel{1/2}\,(1 - 1/2^k)}{\cancel{1 - 1/2}}\right]$

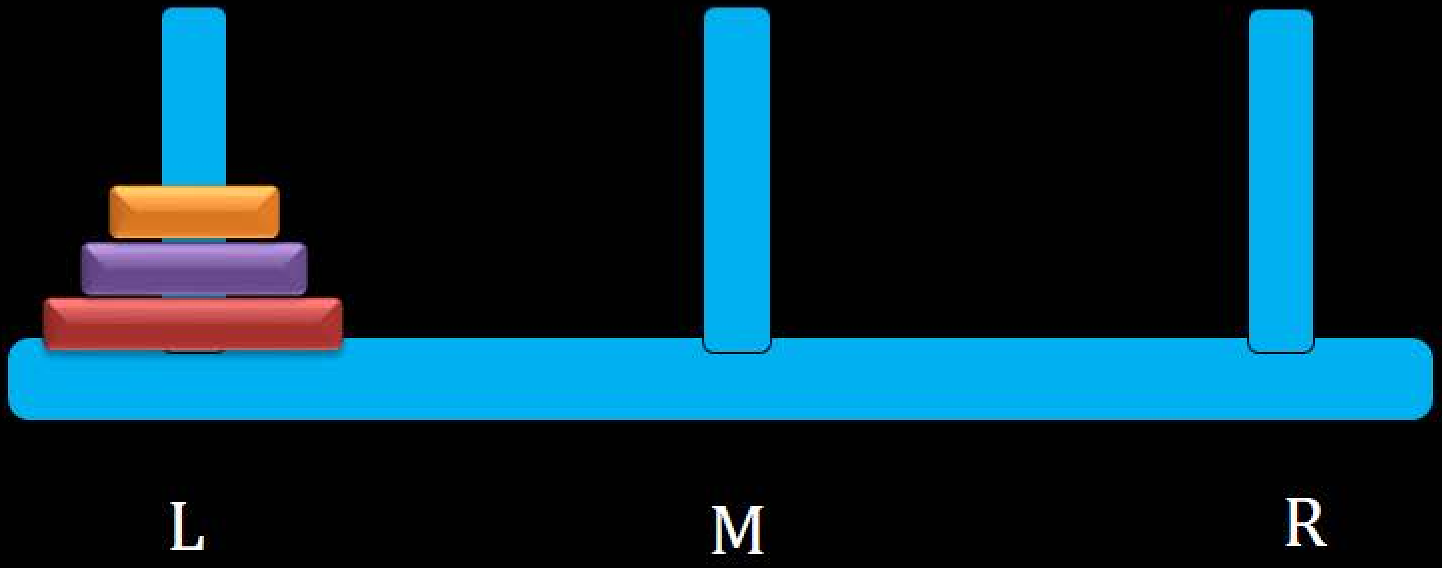
$= n\left(1 - 1/2^k\right)$

$= n(1 - 1/n) = n - 1 = \boxed{\Theta(n)}$

$\dfrac{a(1-r^n)}{1-r}$

Reduction to base condition

$\dfrac{n = 2^k}{n/2^k = 1}$



Recurrence Relation

Number of Steps Taken By An Algorithm

	Total steps	
	$n = 0$	$n > 1$
`Algorithm Rsum (a,n){`		
`if (n <= 0) then`		
`   return 0.0;`		
`else`		
`   return a+ RSum(a, n - 1) +a`		
`}`		

Number of Steps Taken By An Algorithm

```
Algorithm Rsum (a,n){
if (n <= 0) then
   return 0.0;
else
   return a+RSum(a, n - 1)
}
```

Recursive Function

```
long power(long x, long n){
   if (n == 0)
      return 1;
   else
      return x * power(x, n-1);
}
```

How many times is this executed?

Recurrence Relation and Base Condition

Recurrence Relation and Base Condition

$T(n)$ = Time required to solve a problem of size n

Recurrence relations are used to determine the running time of recursive programs – recurrence relations themselves are recursive

$T(0)$ = time to solve problem of size 0 – Base Case

$T(n)$ = time to solve problem of size n – Recursive Case

Solving Recurrences

- Recurrence relations, such as $T(n) = 2T(\lfloor n/2 \rfloor) + n$. Typically these reflect the runtime of recursive algorithms.
- For example, the recurrence above would correspond to an algorithm that made two recursive calls on subproblems of size $\lfloor n/2 \rfloor$, and then did n units of additional work.

Recurrence Relation

```
long power(long x, long n){
if (n == 0)
     return 1;
else
     return x * power(x, n-1);
}
```

$T(0) = 1$

$T(n) = 1 + T(n-1)$

Recurrence Relation

- $T(0) = 1$
- $T(n) = 1 + T(n - 1)$

Guess the k^{th} term

$$T(n-k)+k$$

Reducing to base condition

$$T(n-k)+k$$

Reducing to base Condition

$T(0) = 1$

$T(n) = T(n-k) + k$, for all k

Guess the k^{th} term

If we set $k = n$,

we have: $T(n) = T(n - n) + n \times 1$

$= T(0) + n \times 1$

$= 1 + n \times 1 \in \Theta(n)$

Solving Recurrence Relation

Substitution method

Solving Recurrence Relation

Substitution method

- Get the base condition and Recurrence relation for the for the program
- Substitute the recurrence for next term
- Guess the k^{th} term
- Reduce to base condition
- Solve the series if generated
- Get the final answer or bound which ever required.

Different Method of Solving Recurrence Relation

Different Method of Solving Recurrence Relation

- Substitution method
- Recursion tree
- Master Method

Question

Solve the following Recurrence Relation

$$T(n) = T(n-1) + c$$

$$T(1) = 1$$

Question

Solve the following Recurrence Relation

$$T(n) = T(n-1) + n$$

$$T(1) = 1$$

Question

Solve the following Recurrence Relation

$$T(n) = T(n-1) + \frac{1}{n}$$

$$T(1) = 1$$

Is this Recurrence Relation Represents complexity of fact

$$T(n) = \begin{cases} 1 & n = 1 \\ T(n-1) * n, & n > 1 \end{cases}$$

false

$$T(n-1) = (n-1) * T(n-2)$$

$$T(n) = T(n-2) * (n-1) * n$$

$$n! = \Theta(n!)$$

or

$$O(n^n)$$

```
int fact (int n) {
    if (n == 1)
        return 1
    else
        return n*fact(
}
```

con

const

factoral

Question

Solve the following Recurrence Relation

$$T(n) = T(n-1) + \frac{1}{n} + a$$

$$T(1) = 1$$

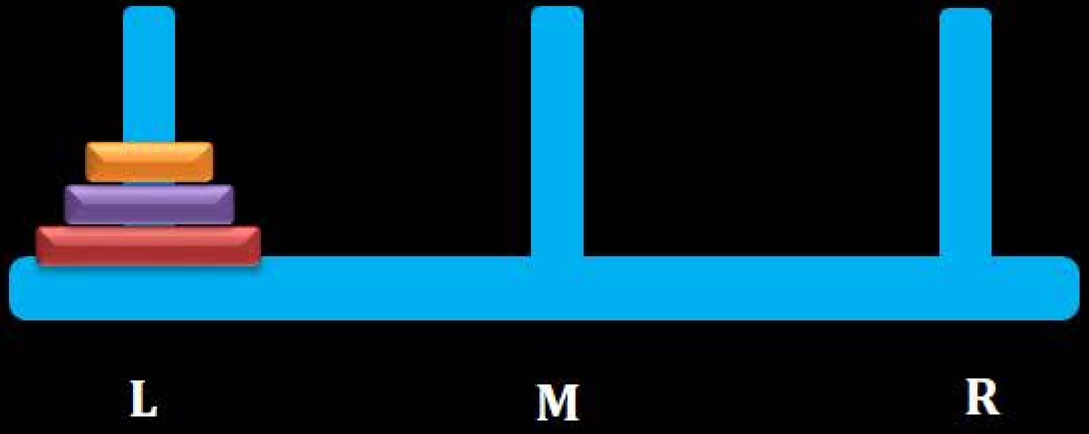
Homework :

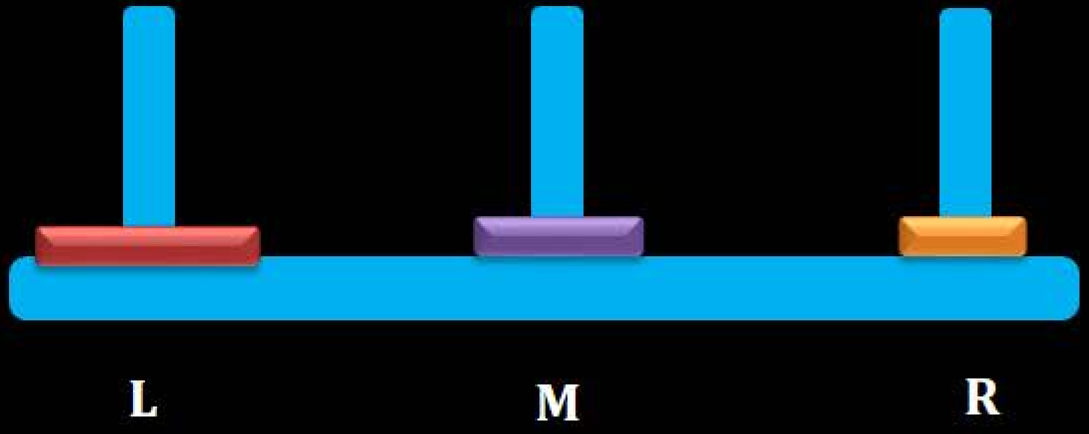
$$T(n) = T(n-1) + \log n \quad , \; n > 1$$

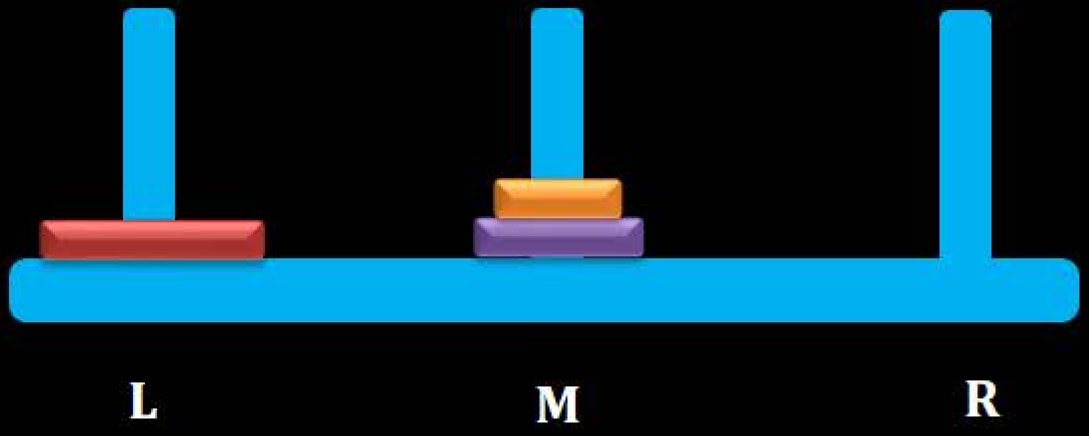
$$T(1) = 1$$

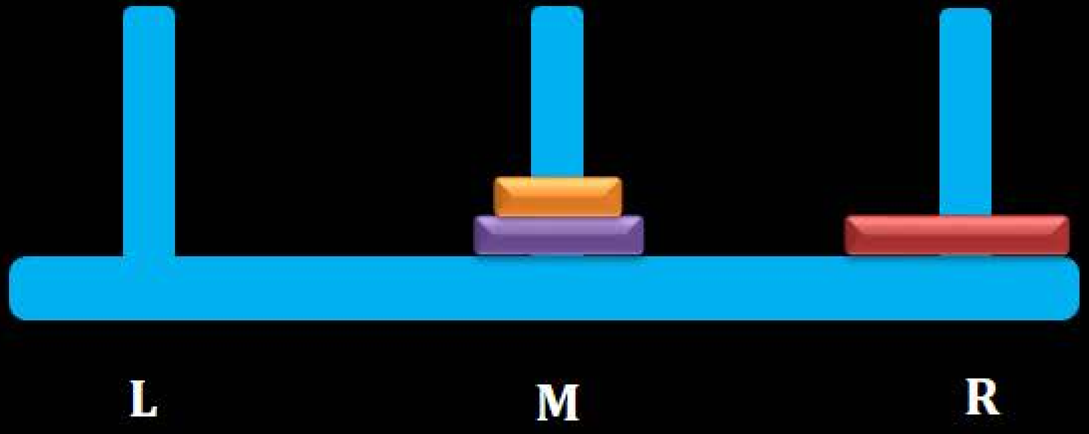
Tower of Hanoi

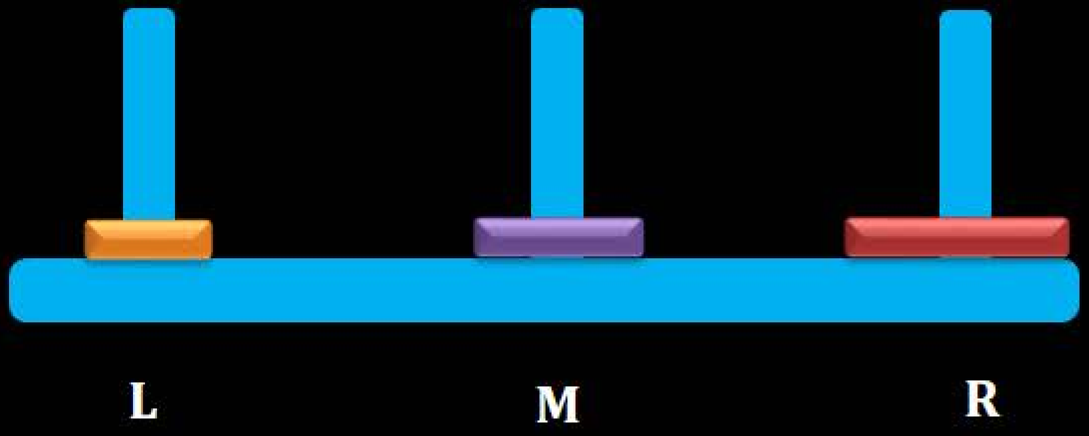
```
Algorithm TowersOfHanoi(n, L, M, R)
// Move the top n disks from tower x to tower y.
{
      if (n >= 1) then {
            TowersOfHanoi(n – 1, L, R, M);
            write ("move top disk from tower", L, "to top of tower", R);
            TowersOfHanoi(n – 1,M, L, R);
      }
}
```

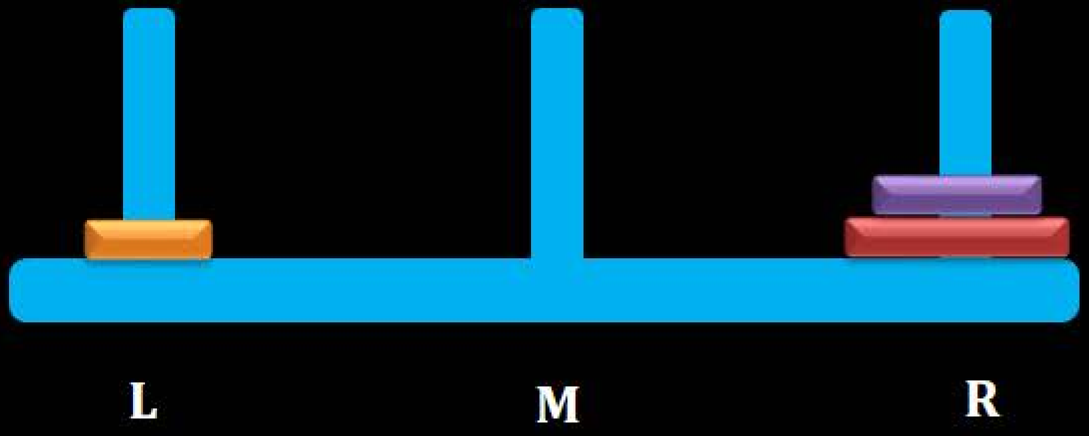

L
M
R


L
M
R


L
M
R


L
M
R


L
M
R


L
M
R

Question

Consider the following Recurrence Relation

$$T(n) = 2T(n-1) + 1$$

$$T(1) = 1$$

Question

Consider the following Recurrence Relation

$T(n) = T(n-1) + \log n$

$T(1) = 1$

Question

Consider the following Recurrence Relation

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

$$T(1) = 1$$

Question

Consider the following Recurrence Relation

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

$$T(1) = 1$$