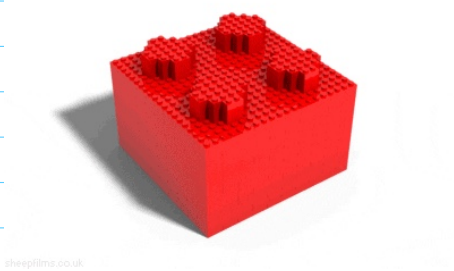
Recursoin?

function calleng itself / subproblem



Topics:
- master theorem
- Quizes
- kth symbol easy
- kth symbol hard
- Towers of hanoi

wikipedia

optional, check online resources if interested!

Master Theorem:

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

if → $T(n) = a\,T(n/b) + O(n^d)$

$a \geq 1$, $b > 1$, $d \geq 0$

a	b	d
1	2	0

$b^d = 2^0 = 1 = a$

solutions:

① $b^d > a \rightarrow TC = O(n^d)$

② $b^d = a \rightarrow TC = O(n^d \log_b(n))$

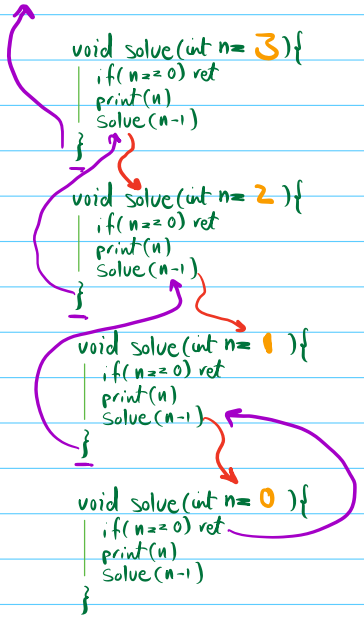
③ $b^d < a \rightarrow TC = O(n^{\log_b(a)})$ (power)

$TC: O(n^1 \times \log_2(n))$

Quiz

Quiz output 3 2 1

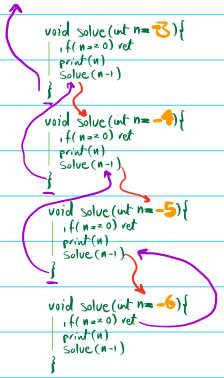
```
void solve(int n= 3 ){
    if( n==0) ret
    print(n)
    Solve(n-1)
}
```



Quiz output -3 -4 -5 ...

```
void solve(int n= -3 ){
    if( n==0) ret
    print(n)
    Solve(n-1)
}
```

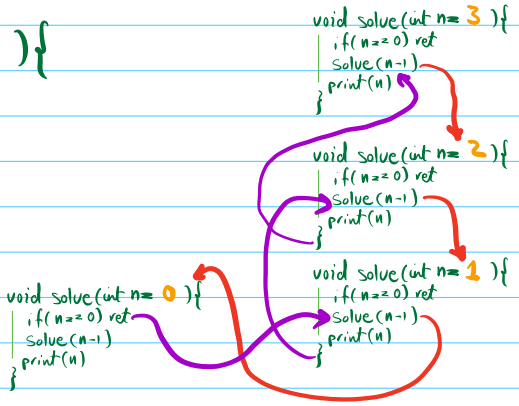
Stack overflow



Quiz

```
void solve(int n= 3 ){
    if( n==0) ret
    Solve(n-1)
    print(n)
}
```

output 1 2 3



P1 kth symbol (easy!)

Problem Description

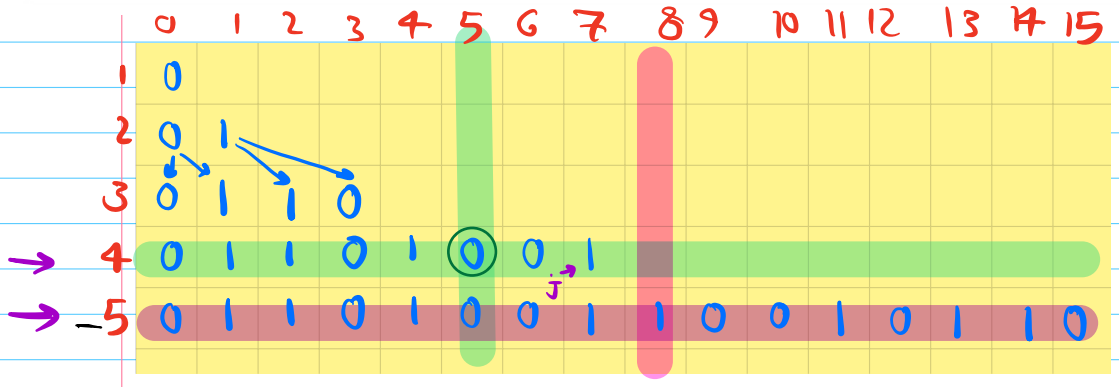
On the first row, we write a `0`. Now in every subsequent row, we look at the previous row and replace each occurrence of `0` with `01`, and each occurrence of `1` with `10`.

Given row number A and index B, return the B^{th} indexed symbol in row A. (The values of B are 0-indexed.).

Problem Constraints

1 <= A <= 20

0 <= B < 2^{A-1}



	A	B	
ex1	4	5	→ 0 ← ans
ex2	5	8	→ 1 ← ans

row# col# 0-based

TC
< $O(A2^A)$

```
int kthNum(int A, int B){
    curRow = ArrayList<int>; curRow.append(0);
    for(i=2; i<=A; i++){
        nextRow = ArrayList<int>();
        for(j=0; j<curRow.len; j++){
            if(curRow[j]==0){ nextRow.append(0)
                              nextRow.append(1)
                            }
            else{ nextRow.append(1)
                  nextRow.append(0)
                }
        }
        curRow = nextRow
    }
  ret curRow[B]
}
```

$2^{A-1} + 2^{A-2}$

$O(2^A)$

TC

```
ArrayList<int> AthRow(int A){
    if(A==1) ret newArrayList(0);
    List<int> prevRow = AthRow(A-1)
    List<int> r = new Arrlist();
    for(i=0; i<prevRow.len; i++){
        if(prevRow[i]==0){ r.append(0)
                           r.append(1)
                         }
        else{ r.append(1)
              r.append(0)
            }
    }
    ret r;
}
```

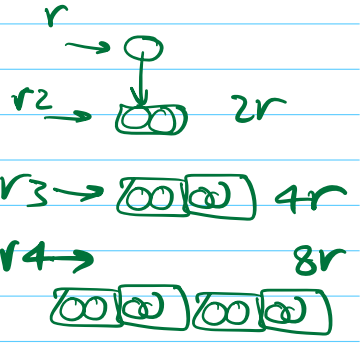
SC

$O(A + \{2^{A-1} + 2^{A-2}\})$

stack

$(A+2^A)$

$O(2^A)$



$1 + 2 + 4 + 8 + 16 + \cdots + 2^{A-1}$:

P2 kth symbol (hard)

Problem Description

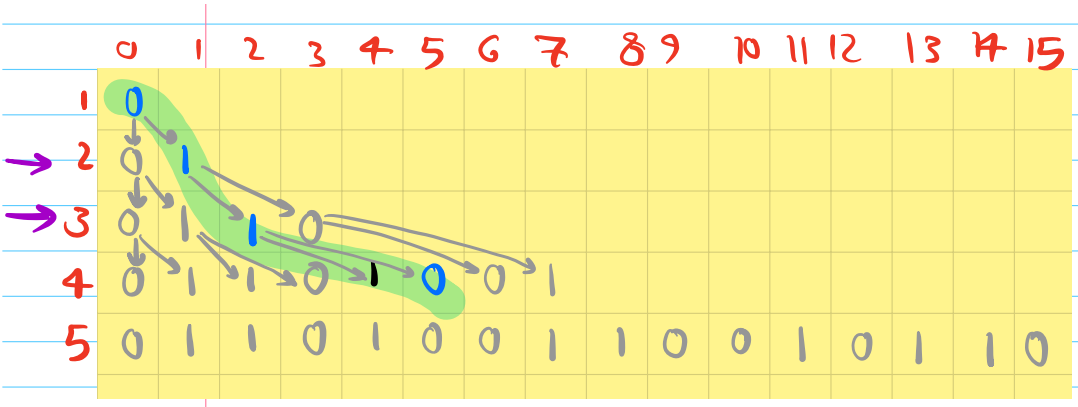
On the first row, we write a `0`. Now in every subsequent row, we look at the previous row and replace each occurrence of `0` with `01`, and each occurrence of `1` with `10`.

Given row number A and index B, return the B^{th} indexed symbol in row A. (The values of B are 0-indexed.).

Problem Constraints

$1 <= A <= 10^5$

$0 <= B <= \min(2^{A-1} - 1, 10^{18})$



TC: O(A) → only the green path

SC: O(A)

```
int kthNum2(int A, long B){
    if(A==1) ret 0;
    P = kthNum2(row-1, B/2)
    if(B%2==0){
        ret P
    }
    ret 1-P
}

main → ret kthNum2(A,B)
```

A = 4, B = 5

4-1=3 ⌊5/2⌋=2

row → row-1

Col → ⌊Col/2⌋

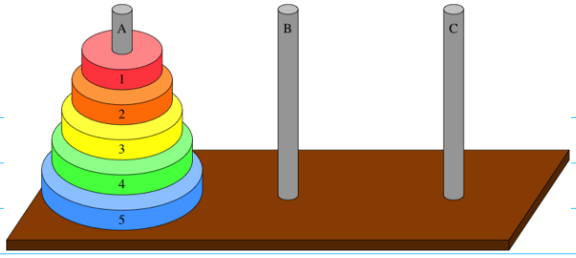
parent info

if parent is (1)
  if(B%2==0) → 1
  else → 0
if parent is 0
  if(B%2==0) → 0
  else → 1

ret (parent + B%2)%2

row	col 0
A	B
4	5
3	2 (1)
2	1 (0)
1	0

Towers of Hanoi



P1 three towers A, B, C

n discs on tower A;

different sizes, smaller on top of larger

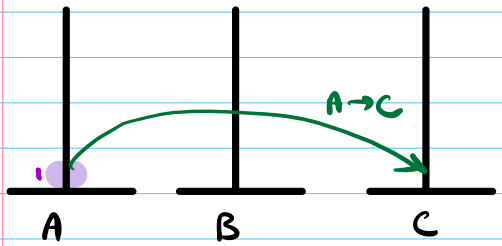
Goal: Move all discs from tower A to C.

Constraints: 1- we can move one disc at a time

2- Large disc cannot be placed on small disc

- Print the moves s.t. all the discs move from A to C in minimum steps.

n=1

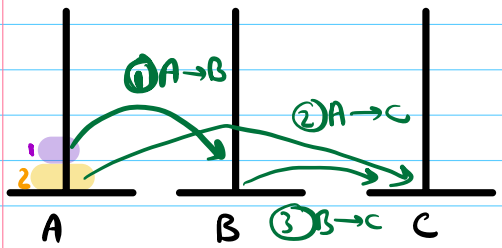


output:

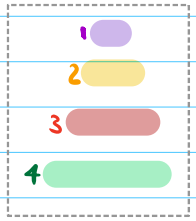
A→C

n=2

helper rod



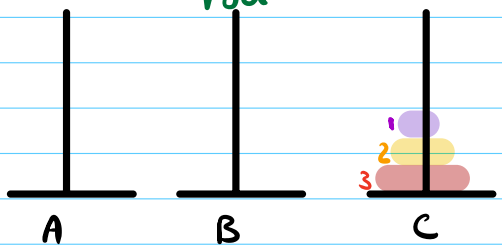
output:



disc bag!

n=3

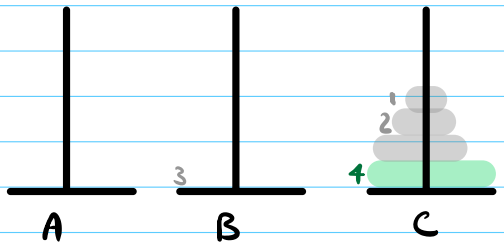
helper rod



output:

7 step

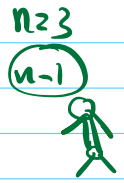
n=4



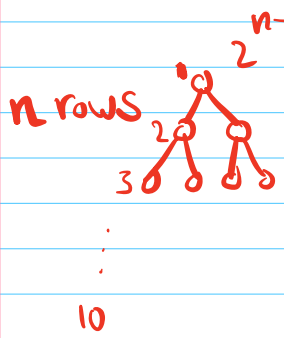
char → rod labels

int #disc Source dest helper

```
void tower(n, A, C, B){
    if(n==1){print(1, A→C); ret}
    tower(n-1, A, B, C)
    print (n, A→C)
    tower(n-1, B, C, A)
}
```



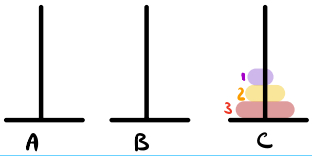
$$T(n) = T(n-1) + 1 + T(n-1)$$
$$= 2T(n-1) + 1$$
$$= 4T(n-2) + 1 + 1$$
$$= 8T(n-3) + 1 + 1 + 1$$
$$\vdots$$

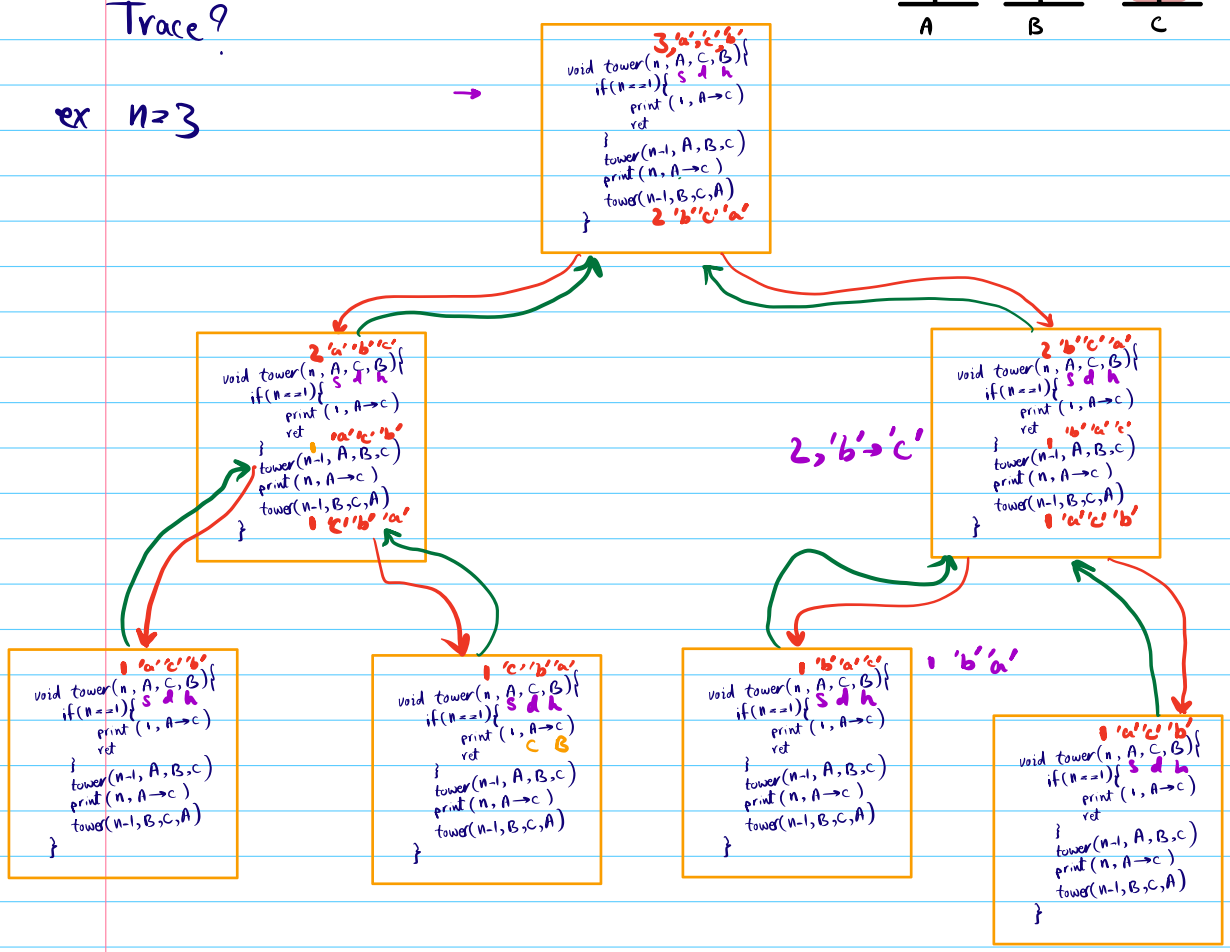


$TC: O(2^n)$

Trace 9

ex n=3





TC

optional

SC → HW