1) Drop in PSP

pow(a, n)

pow (a, n, p)

TC for recursive codes

SC for recursive codes

Q1) Given a, n find a^n using recursion $\boxed{n > 0}$

a	n	a^n
2	5	$2^5 = 32$
3	4	$3^4 = 81$
5	3	$5^3 = 125$

Assumption: Given a, n calc & return a^n

$A^N = A \times A^{N-1}$ pow(A,N) = pow(A, N-1) $\times$ A

Base case:

Main logic

```
int pow(a, n) {
    if (n==1) { return a }    // if (n==0) { return 1 }
    return pow(a, n-1) * a
}
```

pow(a, 5) → pow(a, 4) → pow(a, 3) → pow(a, 2) → pow(a, 1)

pow(a, N) TC : O(N)

$a^8 = a^7 \times a^1 \longrightarrow a^8 = a^4 \times a^4$

$a^{10} = a^5 \times a^5$

$a^{13} = a^6 \times a^6 \times a$

$a^{15} = a^7 \times a^7 \times a$

pow(a,n)

$a^N \longrightarrow a^{N/2} \times a^{N/2}$

pow(a, $\frac{n}{2}$)

$a^N \xrightarrow{\text{N is odd}} a^{N/2} \times a^{N/2} \times a$

$\frac{N-1}{2}$

N is odd

$N = 13 \qquad \frac{N-1}{2} = 6 \qquad \frac{N}{2} = 6$

$\boxed{\text{Fast exponentiation}}$

math.pow(a,n)

Main logic:

```
int pow(a, n) {
    if (n == 1) { return a }
    ha = pow(a, n/2)
    if (N & 1 == 1) {        # N is odd
        return ha * ha * a
    }
    else {
        return ha * ha
    }
}
```

a ** 1000

a^{1000}

```
int pow(a, n) {      n=23    a=2
1 if (n==1) { return a }
2 ha = pow(a, n/2)           2048
3 if (N & 1 == 1) {          # N is odd
     return ha*ha*a
  }                          2048*2048*2 → output
4 else {
     return ha*ha
  }
}
```

```
int pow(a, n) {      n=11    a=2
1 if (n==1) { return a }
2 ha = pow(a, n/2)     32
3 if (N & 1 == 1) {          # N is odd
     return ha*ha*a          32*32*2
  }
4 else {
     return ha*ha
  }
}
```

a^2

```
int pow(a, n) {      n=5    a=2
1 if (n==1) { return a }
2 ha = pow(a, n/2)     a^2=4
3 if (N & 1 == 1) {          # N is odd
     return ha*ha*a          4x4x2=32
  }
4 else {
     return ha*ha
  }
}
```

```
int pow(a, n) {      n=2    a=2
1 if (n==1) { return a }
2 ha = pow(a, n/2)     a
3 if (N & 1 == 1) {          # N is odd
     return ha*ha*a
  }
4 else {
     return ha*ha
  }
}
```

```
int pow(a, n) {      n=1    a=2
1 if (n==1) { return a }
2 ha = pow(a, n/2)
3 if (N & 1 == 1) {          # N is odd
     return ha*ha*a
  }
4 else {
     return ha*ha
  }
}
```

$N \to \frac{N}{2} \to \frac{N}{4} \ \ldots \ 1$

$\boxed{TC: O(\log N)}$

Q3) Given a, n, m calculate $a^n \% m$ $\to [0, m-1]$

$1 \leq a \leq 10^9$

$1 \leq n \leq 10^9$ HW: Update this method to take

$1 \leq m \leq 10^9$ care of overflow

10^{1000}

```
int pow(a, n) {
    if (n == 1) { return a }
    long ha = pow(a, n/2)
    long hp = ((ha%m) * ha%m)%m
    if (N & 1 == 1) {          # N is odd
        return (hp%m * a%m)%m
                      10^9
    }
    else {
        return hp
    }
}
```

1) Modular arithmatic class revise

$\boxed{9:53 - 10:08}$

1) Do a dry run

Substitution method for TC calculation

TC for recursive codes using recursive solution

N = 1 $f(1)$

```
int sum(N){
    if (N == 1) { return 1 }
    return sum(N-1) + N
}
```

TC $\rightarrow$ $f(N)$

$f(N) = O(1) + f(N-1)$

1) Find recursive time equation
2) Find general solution after k substitution
3) Equate unknw ith base condition and find TC

$f(N) = f(N-1) + 1$ Recursive time equation

$f(N) = \boxed{f(N-1)} + 1$

$f(N) = f(N-1) + 1$

substitute $N \rightarrow N-1$

$\boxed{f(N-1)} = f(N-2) + 1$

$f(N) = (f(N-2)+1) + 1$

$f(N) = \boxed{f(N-2)} + 2$

$f(N) = f(N-1) + 1$

$N \rightarrow N-2$

$\boxed{f(N-2)} = f(N-3) + 1$

$$f(N) = (f(N-3)+1)+2$$

$$f(N) = f(N-3)+3$$

$\vdots$

$$\boxed{f(N)} = \boxed{f(N-k)} + k$$

final solution

unknown term

$$f(N) = f(1) + N - 1$$

$$f(N) = 1 + N - 1$$

$$\boxed{f(N) = N}$$

$$f(1) = 1$$

$$f(N-k) = f(1)$$

$$N - k = 1$$

$$k = N - 1$$

$f(1) = 1$

```
int  pow(a, n) {
    if (n == 1) { return a }
    ha = pow(a, n/2)
    if (N & 1 == 1) {      # N is odd
        return ha*ha*a
    }
    else {
        return ha * ha
    }
}
```

step 1)

TC $\longrightarrow f(N)$

$$\boxed{f(N) = O(1) + f(N/2)}$$

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

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

$N \rightarrow N/2$

$$f(N/2) = f(N/4) + 1$$

$$f(N) = \left(f\left(\frac{N}{4}\right) + 1\right) + 1 \quad = f\left(\frac{N}{4}\right) + 2$$

$$f(N) = f\left(\frac{N}{4}\right) + 2$$

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

$N \rightarrow N/4$

$$f\left(\frac{N}{4}\right) = f\left(\frac{N}{8}\right) + 1$$

$f(N) = f\left(\frac{N}{8}\right) + 3$

1) $f(N) = f\left(\frac{N}{2}\right) + 1$

2) $f(N) = f\left(\frac{N}{4}\right) + 2$

3) $f(N) = f\left(\frac{N}{8}\right) + 3$

$f(N) = f\left(\frac{N}{2^k}\right) + k$

$f(N) = f(1) + \log_2 N$

$f(N) = 1 + \log_2 N$

$f(N) = O(\log_2 N)$

$f\left(\frac{N}{2^k}\right) = f(1)$

$\frac{N}{2^k} = 1$

$N = 2^k$

Taking $\log_2$ both sides

$\log_2 N = \log_2 2^k$

$\boxed{\log_2 N = k}$

$\log_x x^n = n$

7   2min   8   4min

$A \rightarrow 2$ $B \rightarrow 4$

$A * B$

~~4~~, 6, ~~8~~

Time equation:

$$\boxed{f(N) = 1 + f\left(\frac{N}{2}\right) + f\left(\frac{N}{2}\right)}$$

```
int pow(a, n) {
    if (n == 1) { return a }
    ha = pow(a, n/2) x pow(a, n/2)
    if (N & 1 == 1) {        # N is odd
        return ha x a
    }
    else {
        return a
    }
}
```

$$f(N) = 2f\left(\frac{N}{2}\right) + 1 \qquad f(N) = 2f\left(\frac{N}{2}\right) + 1$$

$N \to N/2$

$$f\left(\frac{N}{2}\right) = 2f\left(\frac{N}{4}\right) + 1$$

$$f(N) = 2\left[2f\left(\frac{N}{4}\right) + 1\right] + 1$$

$$f(N) = 4f\left(\frac{N}{4}\right) + 3 \qquad f(N) = 2f\left(\frac{N}{2}\right) + 1$$

$N \to N/4$

$$f\left(\frac{N}{4}\right) = 2f\left(\frac{N}{8}\right) + 1$$

$$f(N) = 4\left[2f\left(\frac{N}{8}\right) + 1\right] + 3$$

$f(N) = 8f\left(\frac{N}{8}\right) + 7$

$k=1$

$f(N) = 2f\left(\frac{N}{2}\right) + (2-1)$

$k=2$

$\boxed{f(N) = 2^k f\left(\frac{N}{2^k}\right) + (2^k - 1)}$

$f(N) = 4f\left(\frac{N}{4}\right) + (4-1)$

$k=3$

$f(N) = 8f\left(\frac{N}{8}\right) + (8-1)$

$k=4$

$f(N) = 16f\left(\frac{N}{16}\right) + (16-1)$

$f(N) = Nf(1) + N - 1$

$= 2N - 1$

$f(1) = O(1)$

$f(N) = O(N)$

$f\left(\frac{N}{2^k}\right) = 1$

$N = 2^k$

HW

$f(N) = 4f\left(\frac{N}{2}\right) + 1 \qquad f(1) = N$

$f(N) = f(N-1) + N \qquad f(0) = 1$

$f(N) = f(N-1) + f(N-2) + 1$

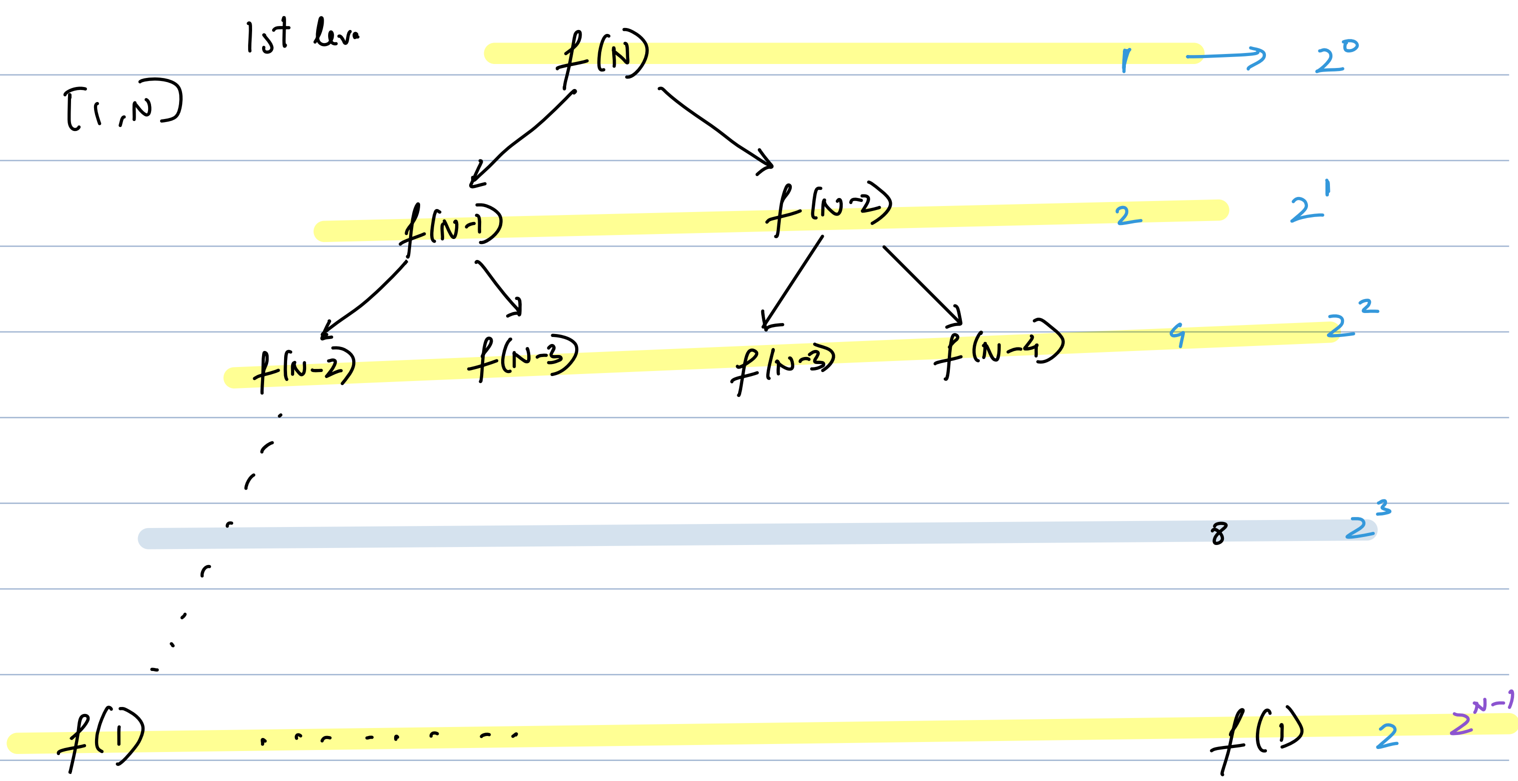
Right side more than one recursive term

substitution fails

```
int fib(N){
    if(N<=1){return N}
    return fib(N-1) ± fib(N-2)
}
```

for (i=0; i<N; i++)

Time equation: $f(N) = f(N-1) + f(N-2) + 1$



N levels

TC: (Total No. of funtion calls) × Time taken for every funtion call

$1 + 2 + 4 + 8 \dots 2^{N-1}$

sum of GP $\quad a\left(\frac{r^n - 1}{r-1}\right)$

$r = 2 \quad n = N$

$a = 1$

$\boxed{TC: O(2^N)}$

Space complexity

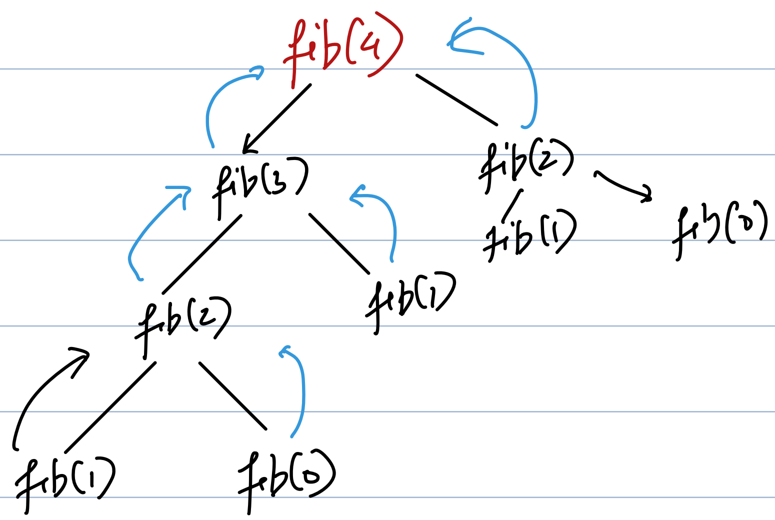
Dry run

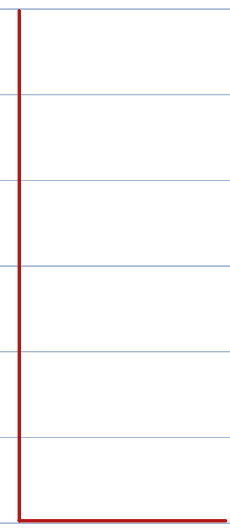
Max no. of entries in function call stack

fib(4)

TC: $O(2^N)$

SC: $O(2^N)$





size = 4

$\boxed{SC: O(N)}$

Print string in reverse order using recursion

0	1	2	3	4	5
P	R	E	R	N	A

[0 : i]

Assumption : rev will print the string in rev order.

```
void rev(s, i) {
    if (i == -1) { return }
    print(s[i])
    rev(s, i-1)
}
```

$\boxed{i = len-1}$