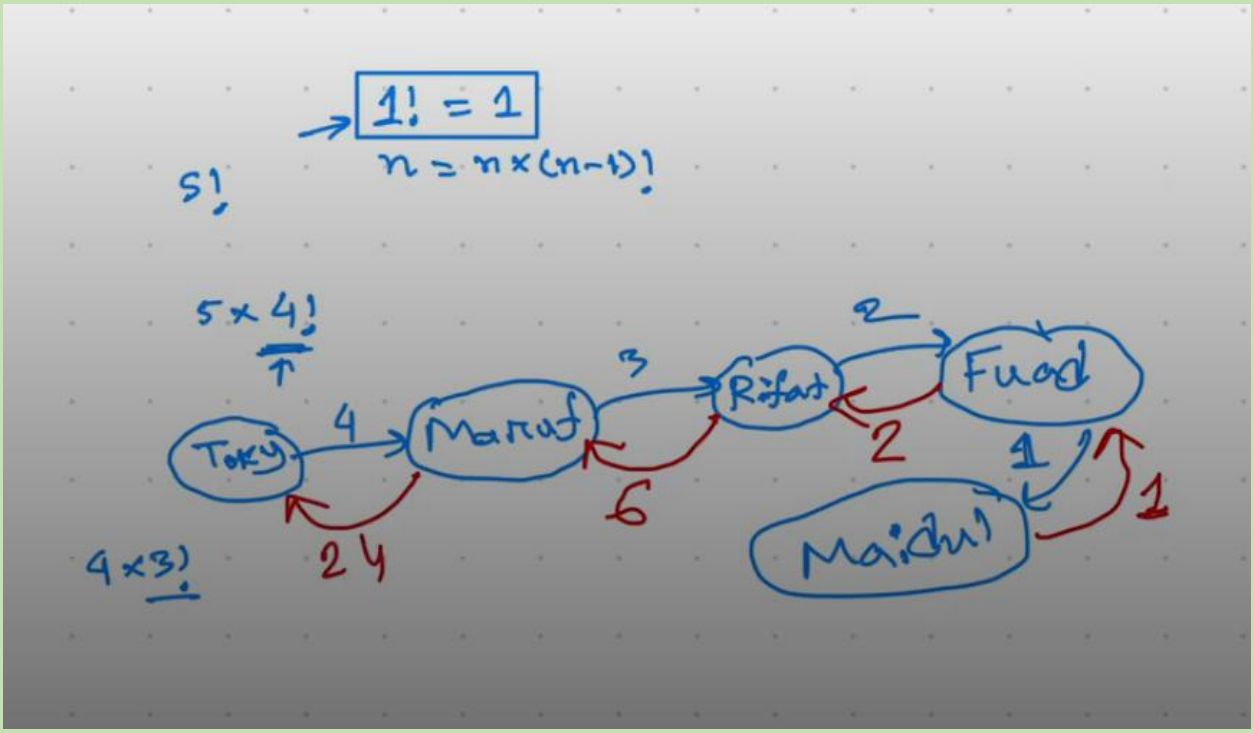
Recursion

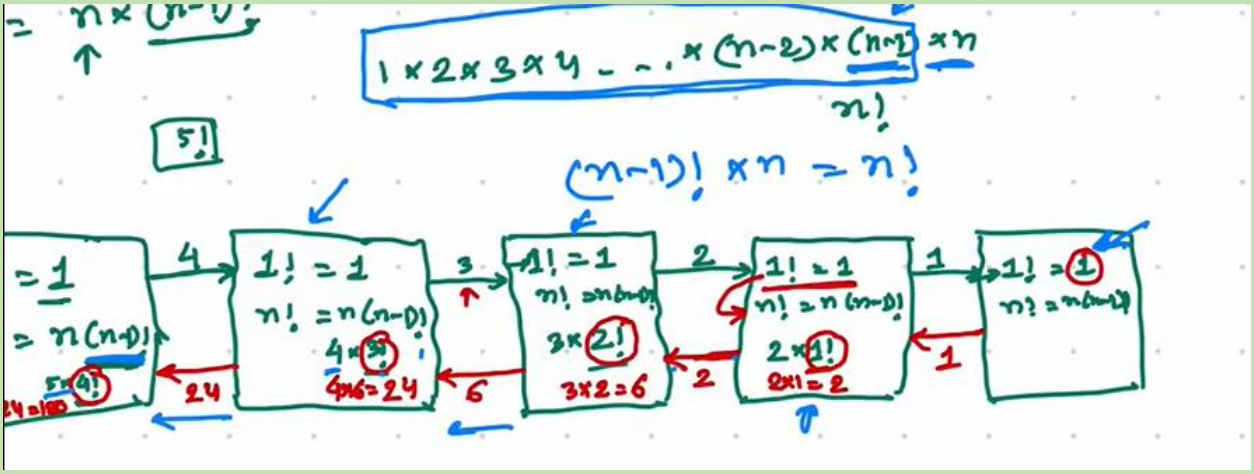
3!

↓

1 x 2 x 3

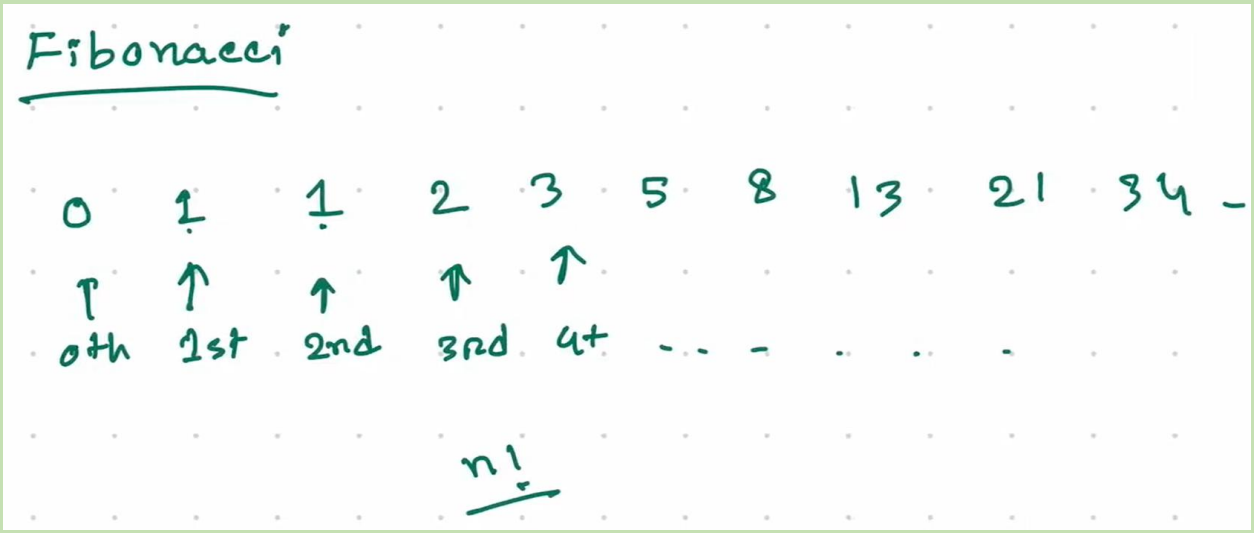
machine

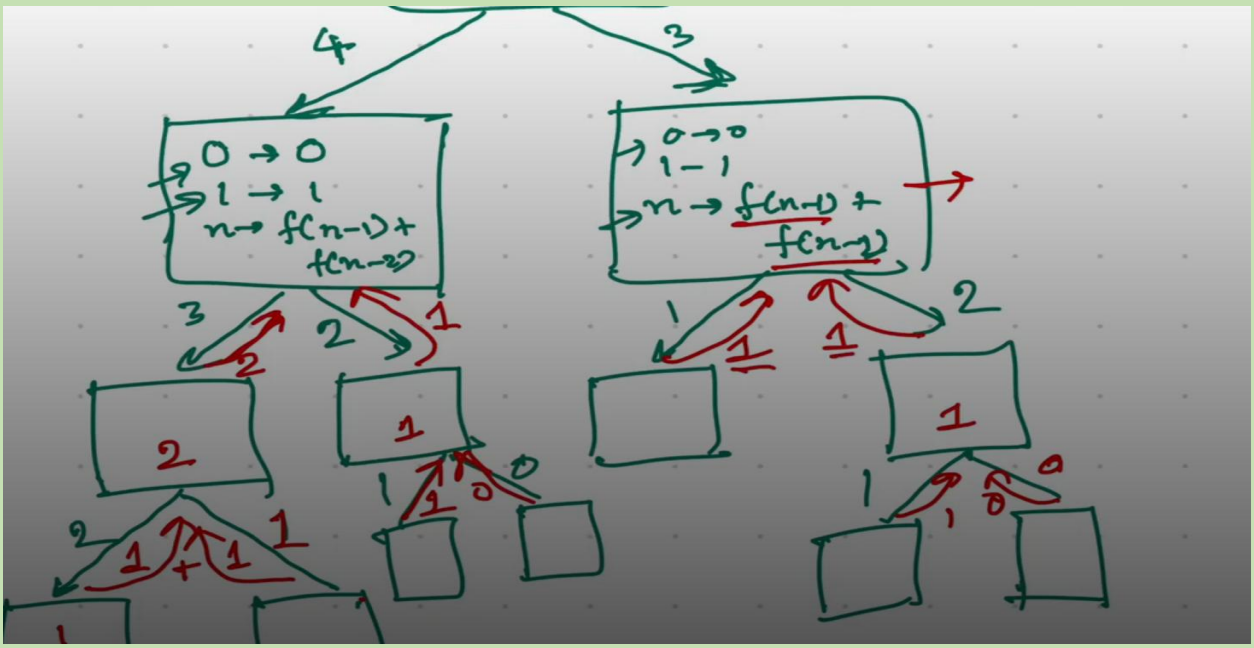




Implementation of this problem:

```
3
4  int fact(int n){
5      if(n == 1) return 1;
6      return n * fact(n-1); // new function create new memory on stack
7  }
8
9
```

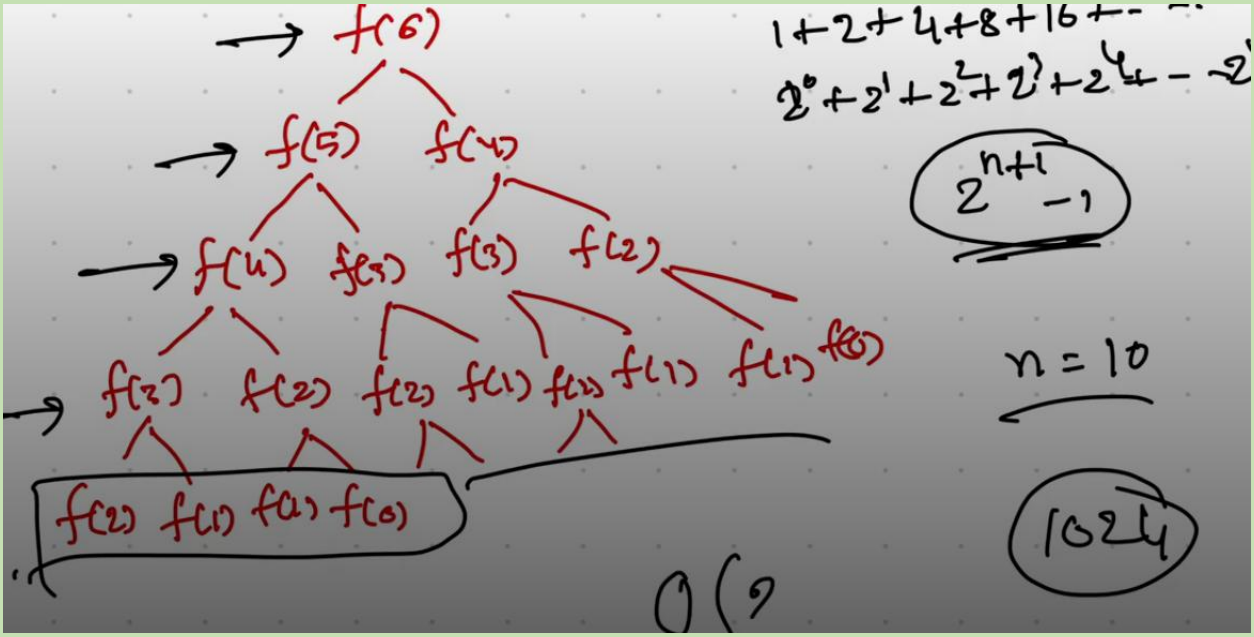




Implementation:

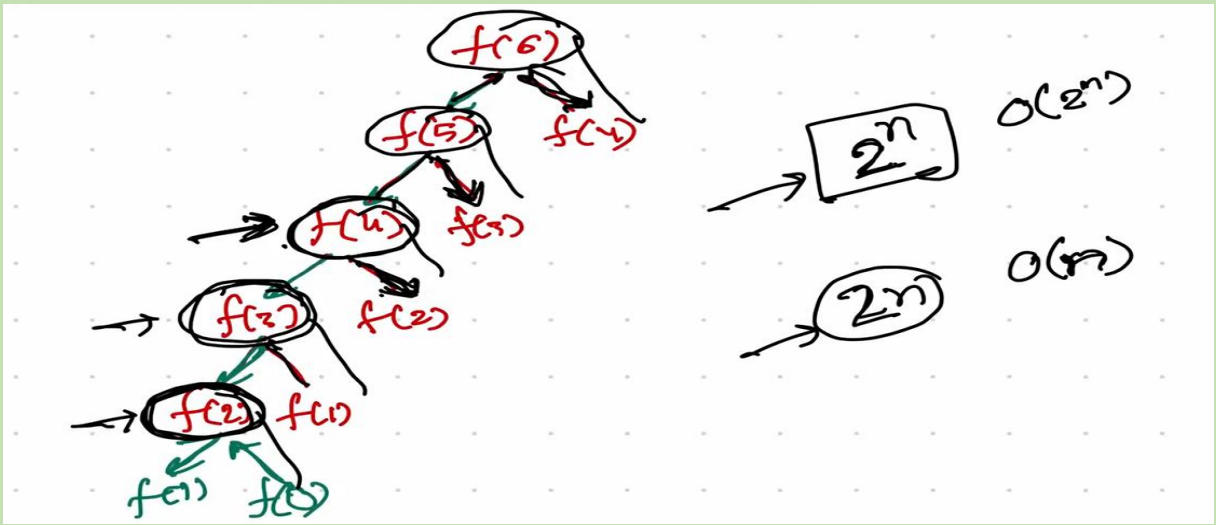
```
4
5  int fib(int n){
6      if(n == 0 || n == 1) return n;
7      //if(n <= 1) return n;
8      return fib(n - 1) + fib(n - 2);
9  }
10
```

```
time complexity: O(2^n) // exponential
```



Ekhne kicu function bar bar call hocce, like : f(4), f(3), f(2) and more……jeta complexity baraia diche.Ekbar ber kore store kore rakhle, then again proyojon hole oy store kora value ta use korle complexity reduce kora possible, let's see..

Recursion Tree :



Every function is executing independently, no related to each other, every function allocated new memory.

Optimal Solution will be like this:

```
k to add a breakpoint
int fib(int n){
    if(n <= 1) return n;

    if(mark[n] == 1) return mem[n]; // if already calculated then return the value from memory
    int res = fib(n-1) + fib(n-2);

    mem[n] = res; // store the value in memory
    mark[n] = 1;  // mark as calculated

    return res;

}
```

Complexity: O(n)

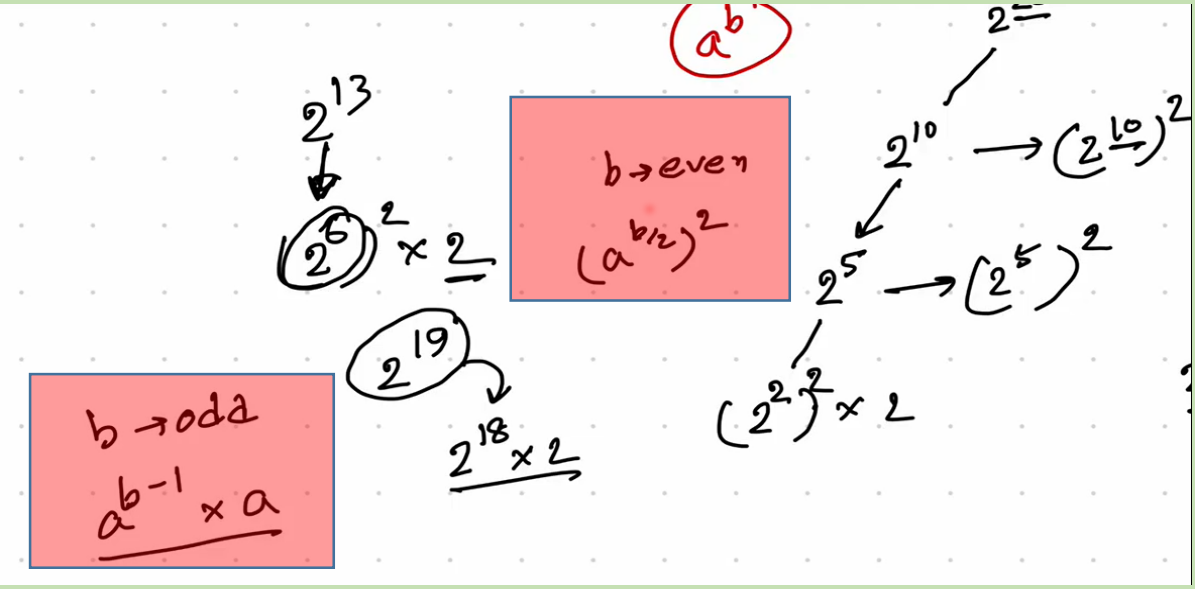
Question: given a, b, find out the a^b

Bruteforce solution, without pow function.

```
int pw_calculatation(int a, int b){
    if(b == 0) return 1;
    int ans = 1;
    for(int i = 1; i <=b; i++)
    {
        ans *=a;
    }
    return ans;
}
```

```
//Complexity: O(b)
```

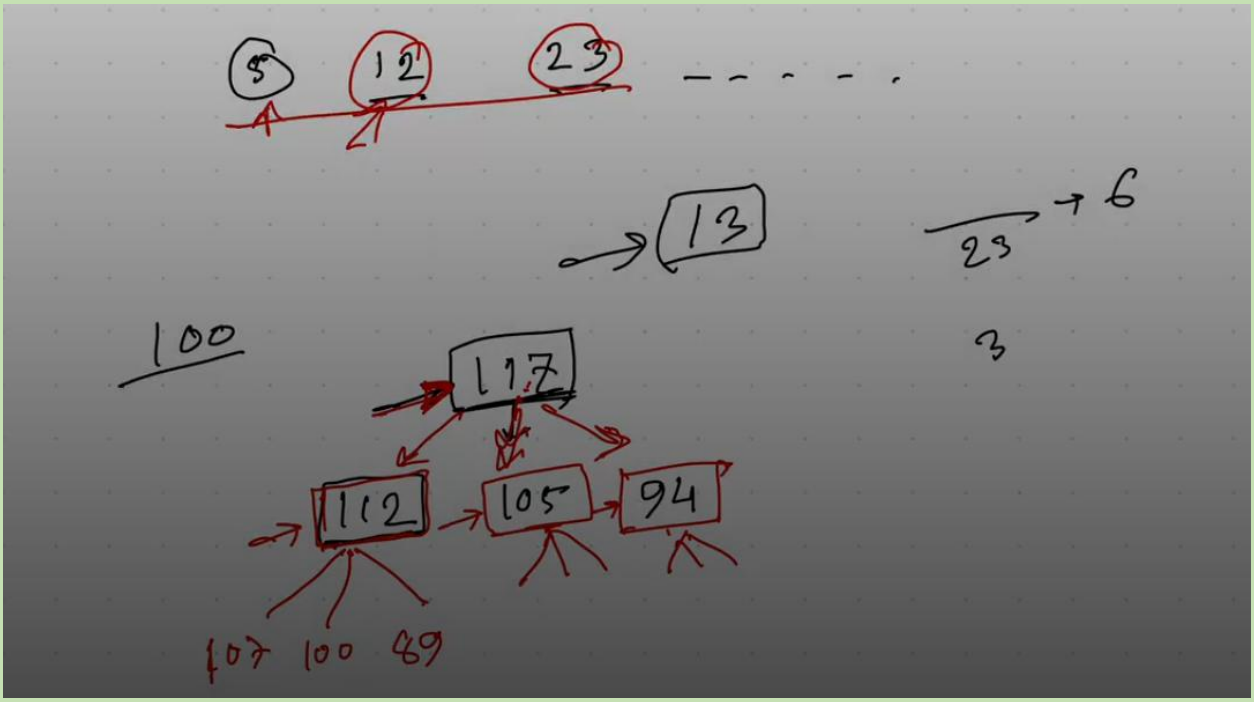
Recursive solution visualization:




```
int power(int a, int b)
{
    if(b == 0) return 1;

    if(b % 2 == 0)
    {
        int x = power(a, b/2);
        return x*x;
    }
    else
    {
        return power(a, b-1)*a;
    }
}
```

Another problem:



Sol:

```
vector<int> coins = {5, 12, 23};

int isPossible(int n)
{
    if(n < 0) return 0;
    if(n == 0) return 1;

    for(int coin: coins)
    {
        if(isPossible(n - coin)) return 1;
    }

    return 0;
}
```

Optimal:

```cpp
#include<bits/stdc++.h>
using namespace std;

vector<int> coins = {5, 12, 23};
int dp[100005];

int isPossible(int n)
{
    if(n < 0) return 0;
    if(n == 0) return 1;

    if(dp[n] != -1) return dp[n];
    int res = 0;
    for(int coin: coins)
    {
        if(isPossible(n - coin)) res = 1;
    }


    return dp[n] = res;
}

int main()
{
    memset(dp, -1, sizeof(dp)); // used to initialize the dp array with -1

    int n;
    cin >> n;
    cout<< isPossible(n) << endl;
    return 0;
}
```

Question: given an integer, return its all digits like 123 , return 1, 2, 3 seperately

```
void print_digits(int n){
    if(n==0) return;

    int last_digit = n%10;

    cout << last_digit << endl;

    print_digits(n/10);
}

int main(){
    int n;
    cin >> n;

    print_digits(n);

    return 0;
}
```

Optimal:

```
#include <bits/stdc++.h>
using namespace std;

void print_digits(int n){
    if(n==0) return;

    int last_digit = n%10;

    print_digits(n/10);

    cout << last_digit << endl;
}

int main(){
    int n;
    cin >> n;

    print_digits(n);

    return 0;
}
```