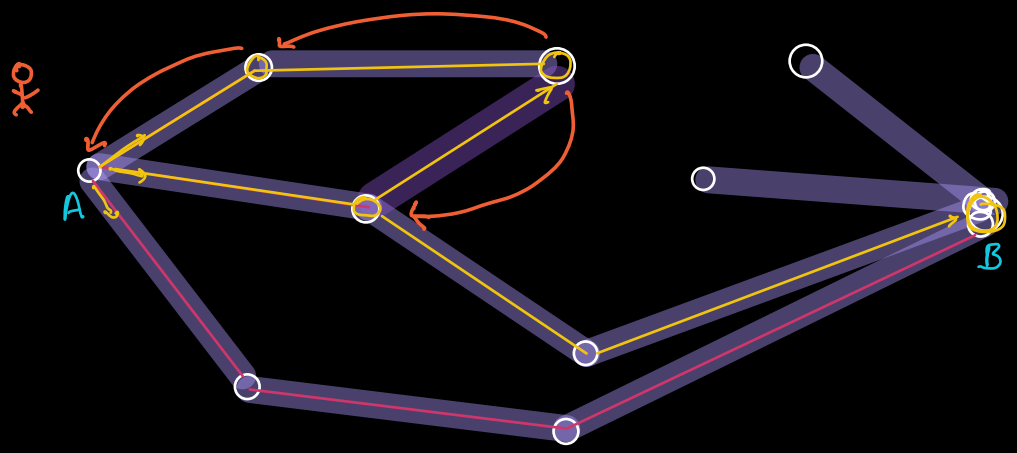
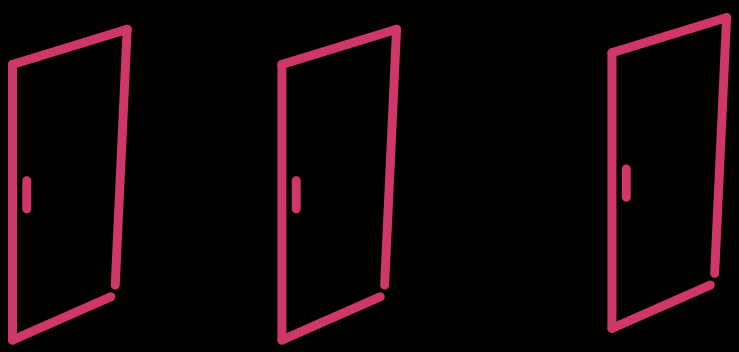




Open
Check
Close ⟶ Backtrack

Backtracking → Trying out all possibilities.

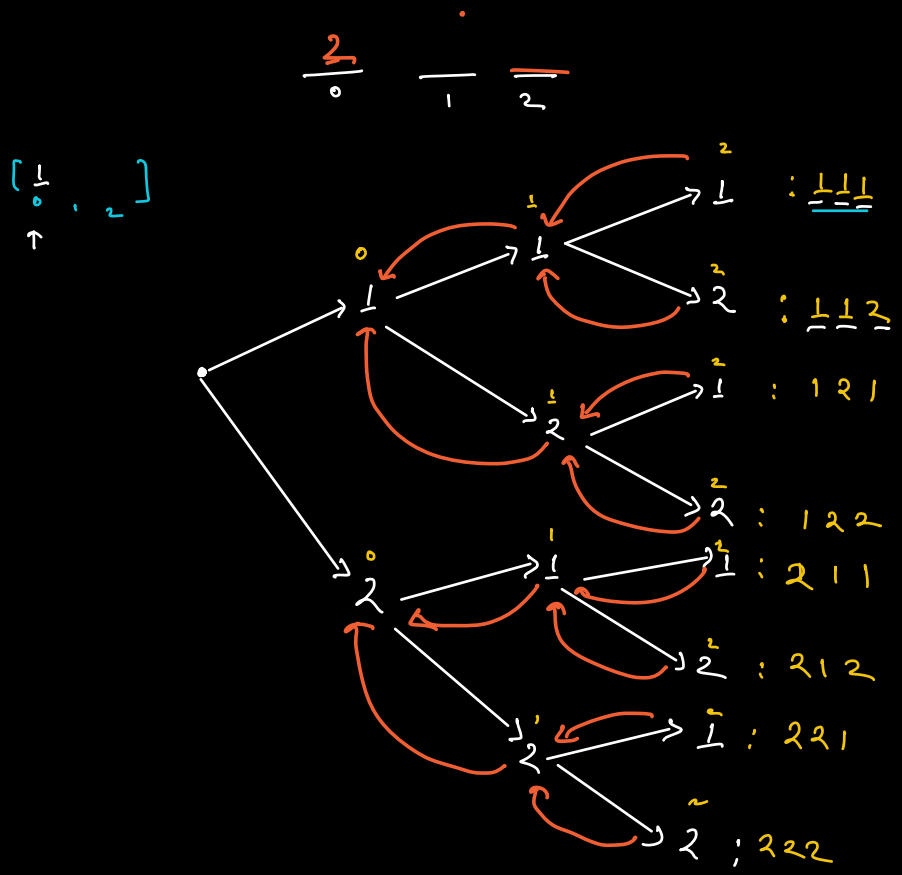
Q. Print all N digit numbers using {1 & 2}

↑ Input

N = 1 — 1, 2

N = 2 — — : 1 1, 1 2, 2 1, 2 2

N = 3 — — — : 1 1 1, 1 1 2, 1 2 1, 1 2 2, 2 1 1, 2 1 2, 2 2 1, 2 2 2

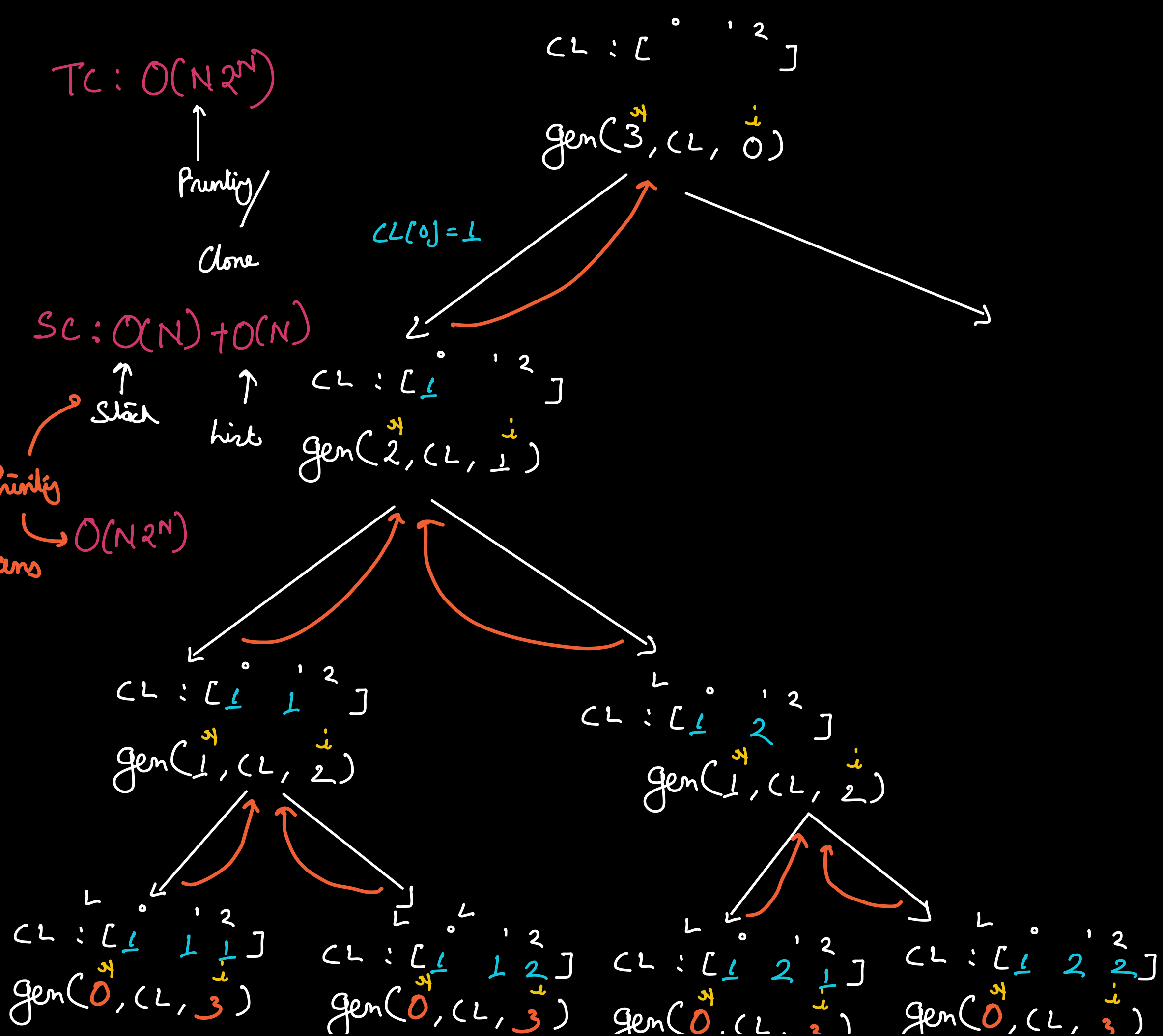


```
void generate(N, currList, index) {      // currList: size → N
    if(N == 0){
        Print currList;      ⟹ O(N)
        return;
    }
    currList[index] = 1;
    generate(N-1, currList, index+1);
    currList[index] = 2;
    generate(N-1, currList, index+1);
}
```

TC: $O(N2^N)$ ↑ Printing / Clone

SC: $O(N) + O(N)$
↑ Stack ↑ List

Printing ↪ $O(N2^N)$ ans



Q Generate/Print all N digit numbers using {1, 2, 3, 4, 5}

```
void generate(curArr[], N, index) {
    if (N == index) {
        print curArr. ⇒ O(N)
        ret;
    }
    curArr[index] = 1;
    generate(curArr, N, index+1);
    curArr[index] = 2;
    generate(curArr, N, index+1);
    curArr[index] = 3;
    generate(curArr, N, index+1);
    curArr[index] = 4;
    generate(curArr, N, index+1);
    curArr[index] = 5;
    generate(curArr, N, index+1);
}
```

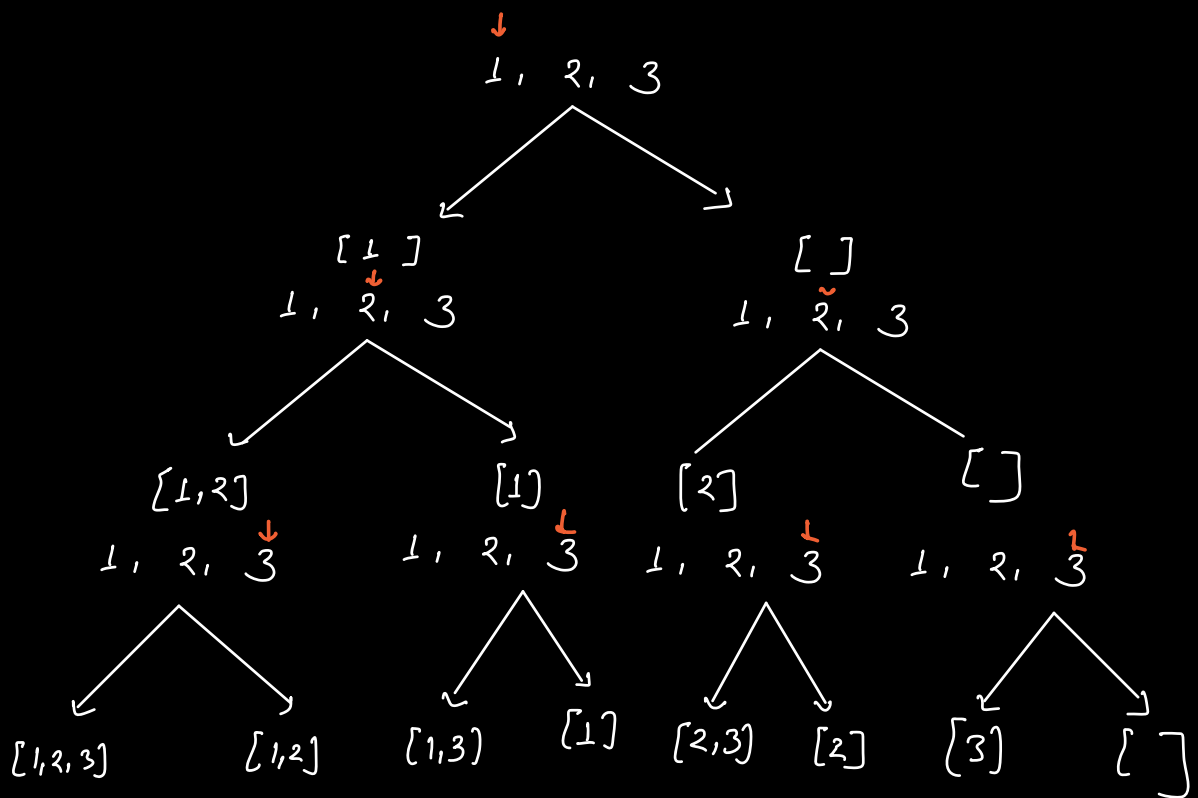
```
for(i=1; i<=5; i++){
    curArr[index] = i;
    generate(curArr, N, index+1);
}
```

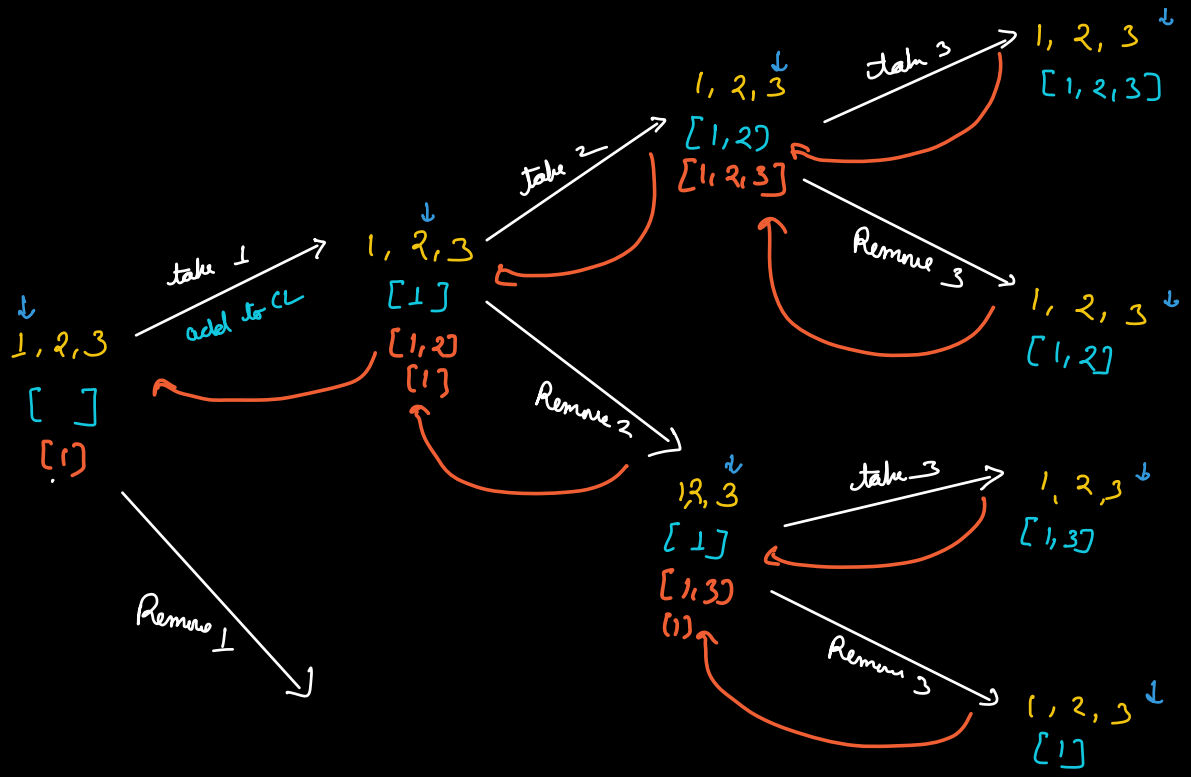
TC : $O(N\,5^N)$

SC → HW

Q Generate all subsets of an array.

A: [1, 2, 3] ⟶ [[ ], [1], [1, 2], [1, 2, 3], [1, 3], [2], [2, 3], [3]]






```
Print All Subsets (curList<>, curIndex, A[], list<list<>> ans){
    if (curIndex == A.length) {
        ans.add(copy of curList); ⇒ O(N)
        ret;
    }
    curList.add(A[curIndex]);
    PrintAllSubset(curList, curIndex+1, A, ans);
    curList.pop();
    PrintAllSubset(curList, curIndex+1, A, ans);
}
```

TC : $O(N\,2^N)$

SC : $O(N)$ ⟶ Excluding output

Q Count the no. of subsets with sum = K.

MS
Amazon
Intuit

A : [5, 2, 7] K = 7

Interviewbit
Leetcode
Pramp

```
int countSubsetSum(A[], K, index, curSum) {
    if (index == A.length) {
        if (curSum == K)
            ret 1;
        else ret 0;
    }
    curSum = curSum + A[index];
    int x = countSubsetSum(A, K, index+1, curSum);
    curSum = curSum - A[index];
    int y = countSubsetSum(A, K, index+1, curSum);
    ret x+y;
}
```

[2, 0, 0, 0]
K = 2

Q Given an array. Print all the permutations.
(with distinct no)

A : 1, 2, 4
N

0	1	2
1,	2,	4
1,	4,	2
2,	1,	4
2,	4,	1
4,	1,	2
4,	2,	1

N!