

N Bit binary Number

* Find all N Bit Binary Number having more than equal to 1 than 0 for any prefix

N=4

⑥

1101 ✓

111 ✓

110

1000 ✗

1001

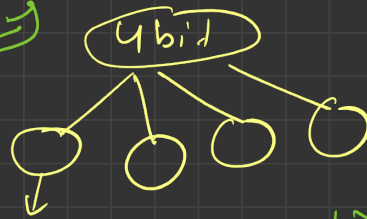
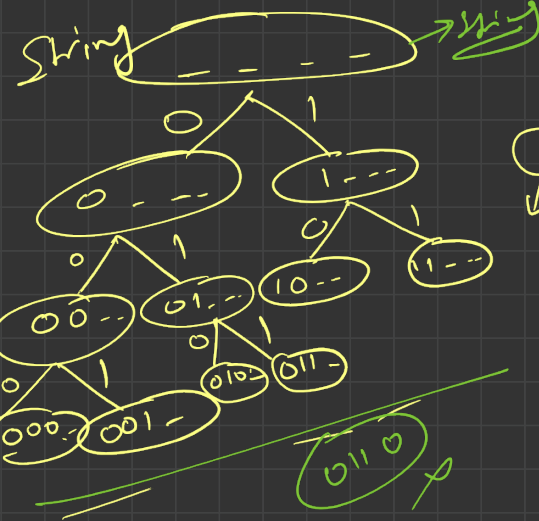
1 ✓

10 ✓

100 ✗

100

1001 ✓



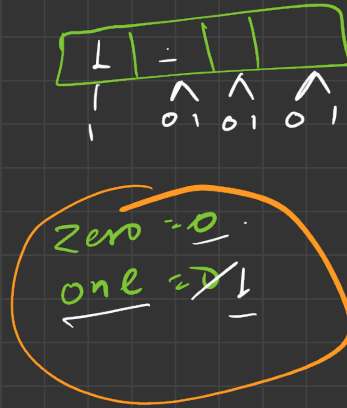
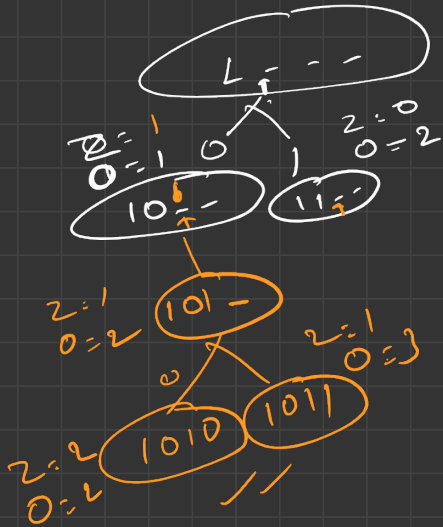
1 > 0
+ 0 0 1 x

Zero: ~~01~~
one: ~~123~~

↓ ↓ ↓ ↓
1 1 0 1

1101

0 → 1 2
1 → 1



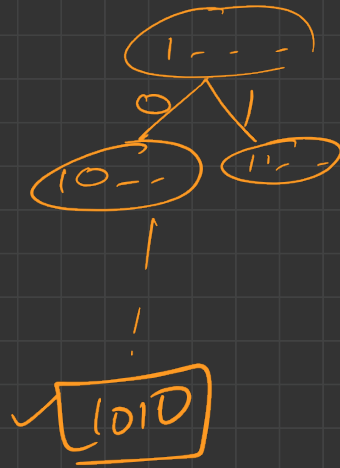
4bit

`if(zero <= one) {`

vector <string> ans
N ←

String temp

zero
one



Find (int n, vector<string> &ans, string &temp, int zero, int one){

if (temp.size() == n) {
 ans.push-back(temp);
 return;
}

✓✓ if (zero < one) {

temp.push-back('0');

~~Find~~ Find (n, ans, temp, zero+1, one)

→ temp.pop-back();
}

→ ✓✓ temp.push-back('1')

✓✓ Find (n, ans, temp, zero, one+1)

✓✓ temp.pop-back();

}

Handwritten diagram illustrating the recursive steps for calculating the number of ways to reach a target sum n using numbers 1 and 2. The diagram shows a tree structure of states, where each node represents a state with variables $temp$, $zero$, one , and n .

- Root node: $temp=1$, $zero=0$, $one=0$, $n=4$
- Node 1: $temp=10$, $zero=0$, $one=1$, $n=4$
- Node 2: $temp=101$, $zero=1$, $one=1$, $n=4$
- Node 3: $temp=1011$, $zero=2$, $one=2$, $n=4$
- Node 4: $temp=10110$, $zero=2$, $one=3$, $n=4$
- Node 5: $temp=10111$, $zero=3$, $one=3$, $n=4$

The final answer is shown as $ans = 1010$ and 1011 .

