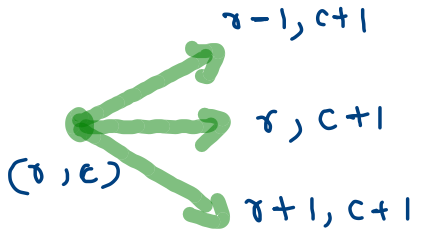
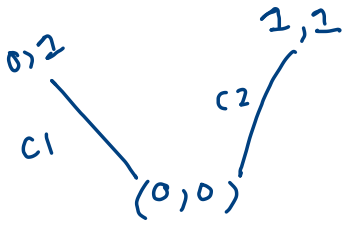
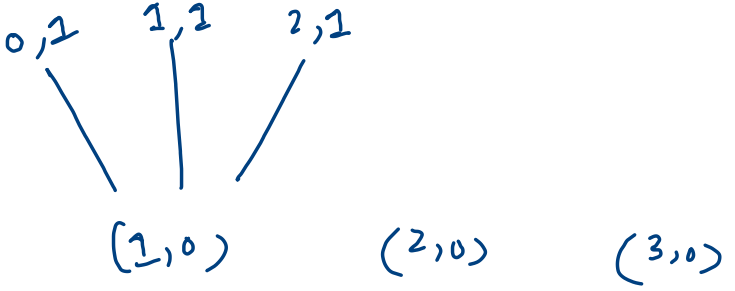
2	1	3	4	5
9	0	2	6	7
3	1	0	6	0
4	8	3	9	7

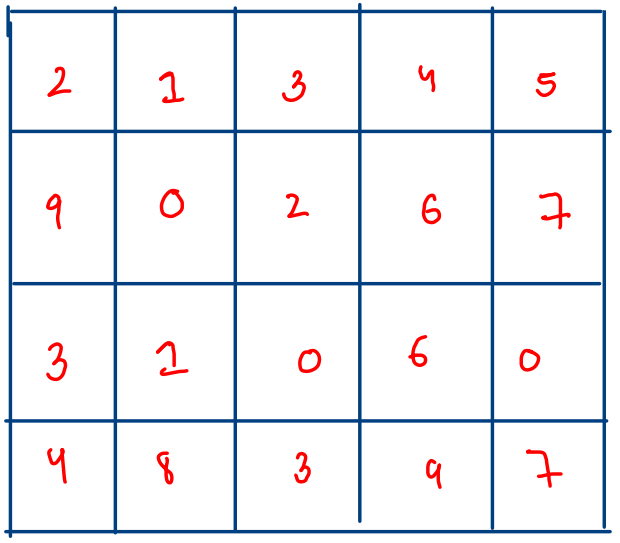




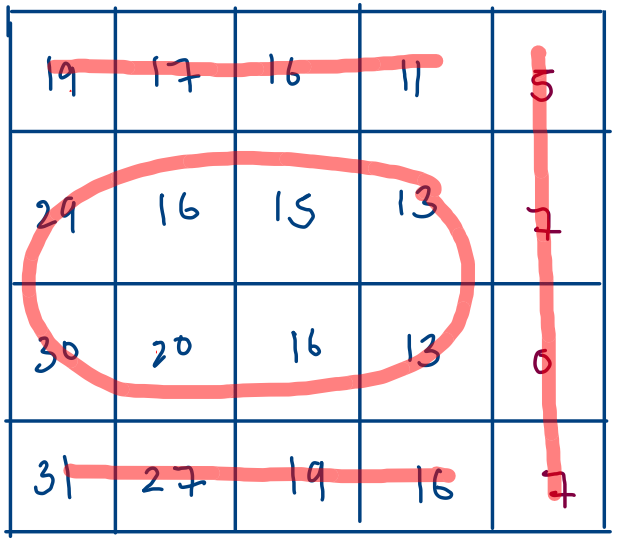


ans-> max(c1, c2) + gold[i][j]

m1 m2



gold



dp

$dp[i][j] = \max(dp(i,j+1))$ $dp[i][j] \rightarrow$

$(dp(i-1,j+1)$?

$(dp(i+1,j+1))$

4 2 1

4 1 2

4 3

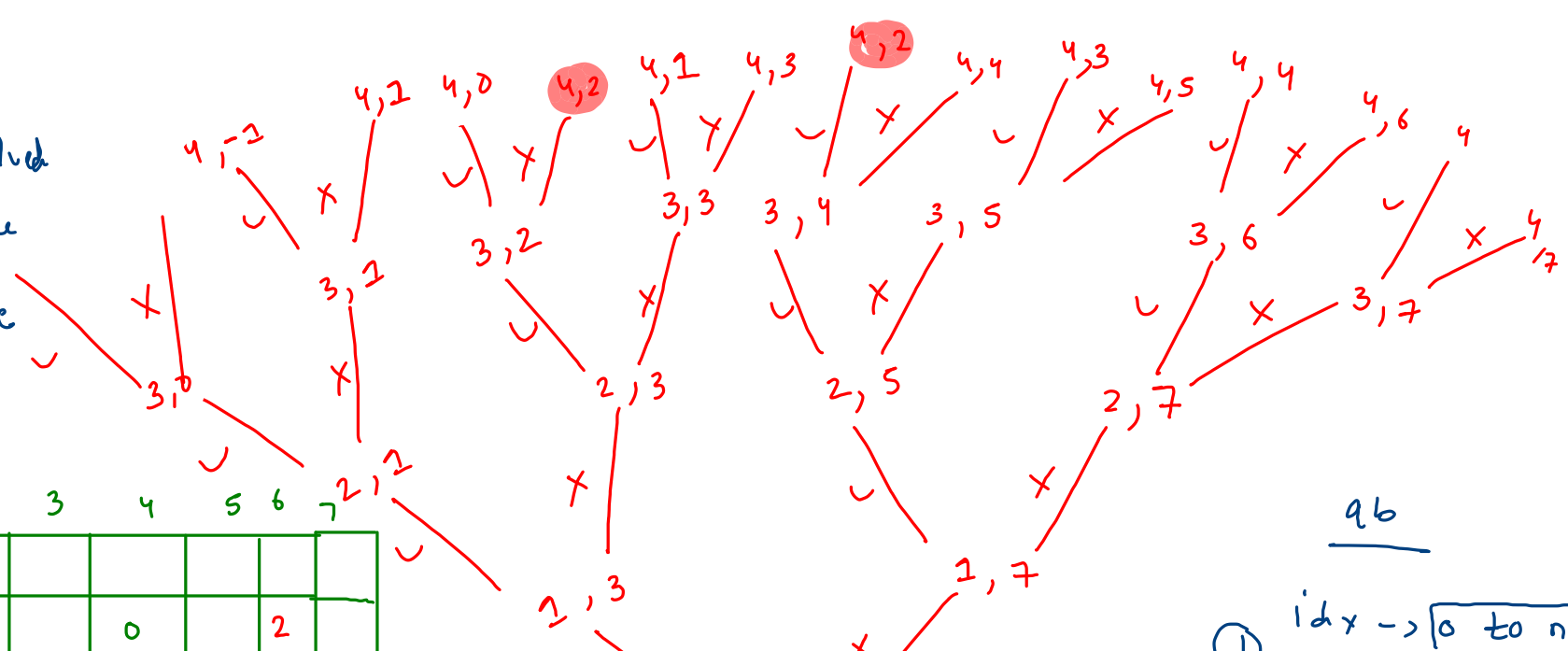
2 2 3

4_0 2_1 1_2 2_3 3_4 target = 7

0 -> unsolved

1 -> true

2 -> false



	0	1	2	3	4	5	6	7
0								
1			1		0		2	
2			0					
3				1	0			
4								

qb

① idx -> 0 to n-1

size -> n

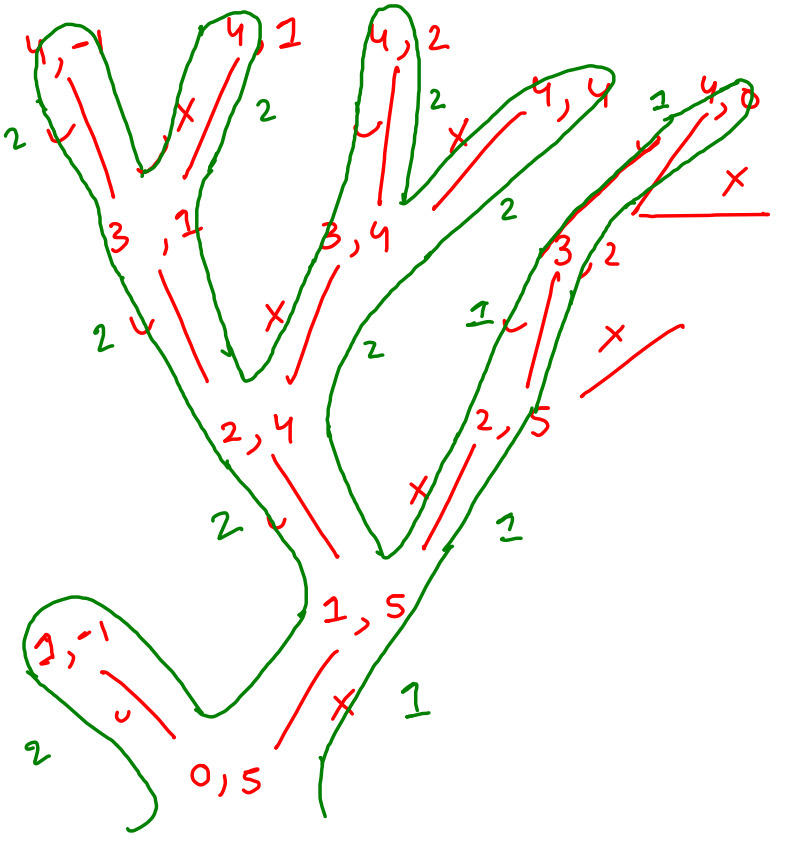
② target -> 0 to tar

size -> target + 1

	0	1	2	3	4	5
0						1
1						1
2					2	1
3		2	1		2	

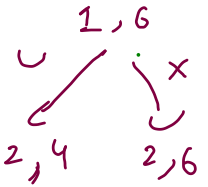
target = 5

6 0 1 1 3 2 2 3

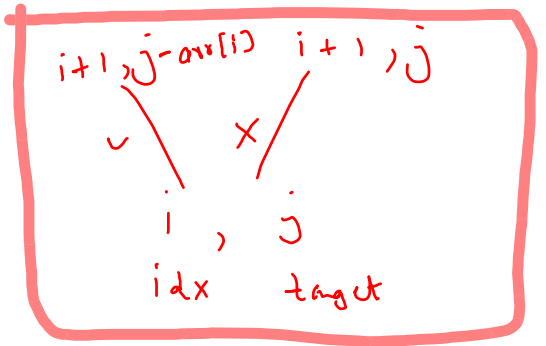



```java
public static int targetSumSubset_mem(int[]arr,int idx,int target,int[][]qb) {

    if(idx == arr.length) {
        if(target == 0) {
            return 1;
        }
        else {
            return 2;
        }
    }

    if(target == 0) {
        return 1;
    }

    if(target < 0) {
        return 2;
    }

    if(qb[idx][target] != 0) {
        return qb[idx][target];
    }

    int inc = targetSumSubset_mem(arr,idx+1,target-arr[idx],qb);

    if(inc == 1) {
        qb[idx][target] = 1;
        return 1;
    }

    int exc = targetSumSubset_mem(arr,idx+1,target,qb);

    int myans = (exc == 1) ? 1 : 2;
    qb[idx][target] = myans;

    return myans;
}
```

	0	1	2	3	4	5	6	7	8	9	10
4_0	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
2_1	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
7_2	✓	✓	X	✓	✓	X	X	✓	✓	X	✓
1_3	✓	✓	X	✓	✓	X	X	X	X	X	X
3_4	✓	X	X	✓	X	X	X	X	X	X	X



$$dp[i][j] = dp[i+1][j],\ dp[i+1][j-arr[i]]$$



Coin Change perm

coins = [2, 3, 5]

coins -> infinite

amt -> 7

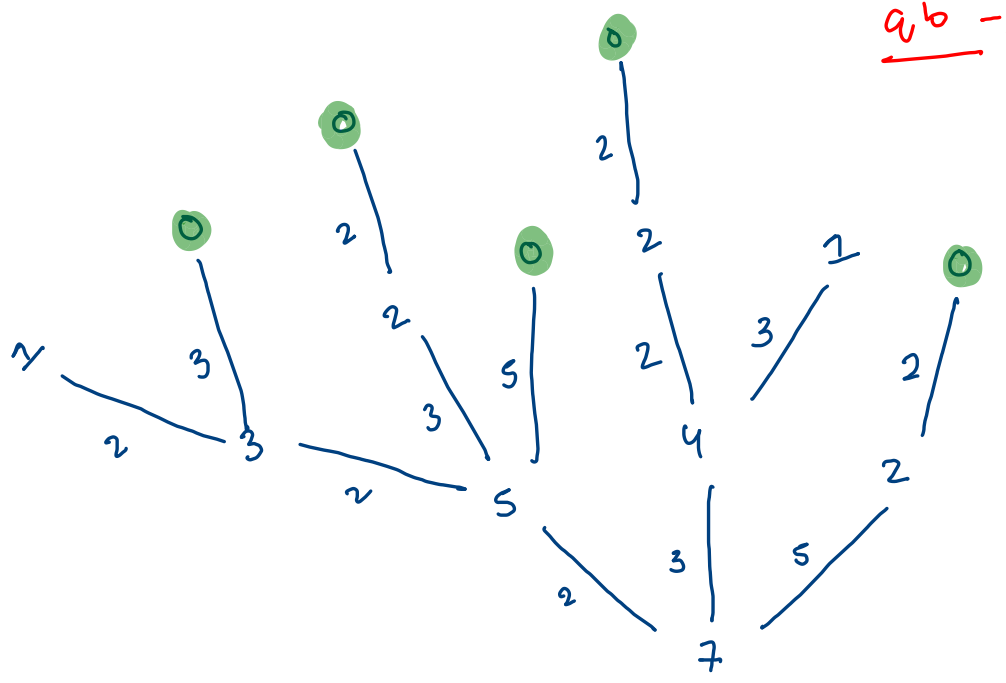
2 5

5 2

2 2 3

2 3 2

3 2 2



qb -> ld

recursion

memoisation

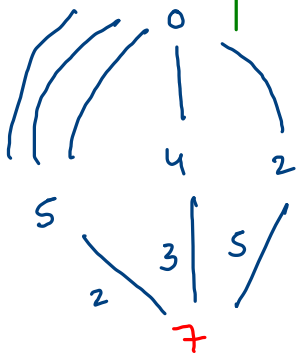
```
for(int i=0; i < coins.length;i++) {
    int rem_amt = amt-coins[i];

    if(rem_amt >= 0) {
        int temp = CCP(coins,rem_amt);
        ways += temp;
    }
}
```

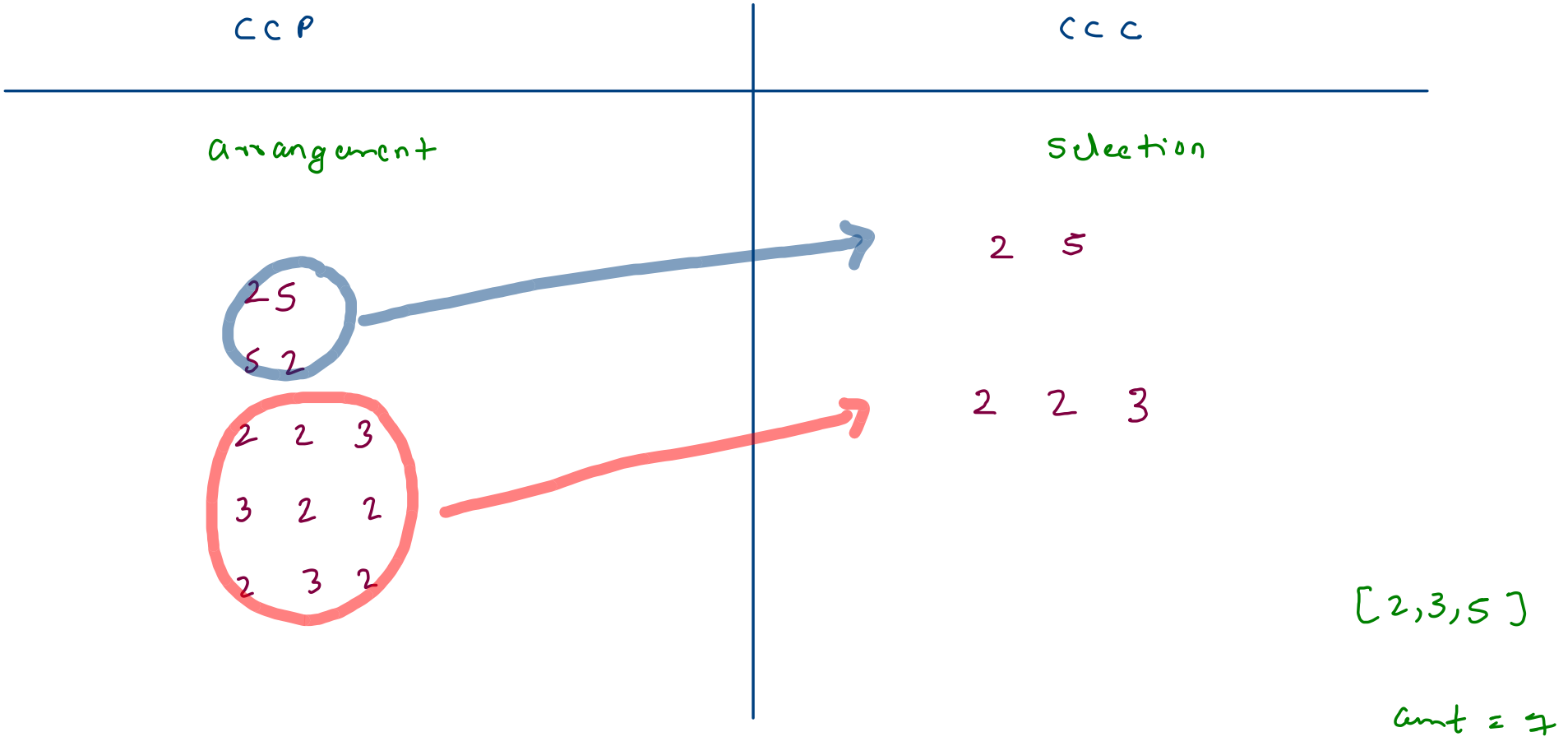
dp[i] -> ways to pay i amt.

coins = [2, 3, 5]

amt = 7.

1	0	1	1	1	3	2	5
0	1	2	3	4	5	6	7
.		2.	3.	22.	32. 23. 5.	222. 33.	322. 232 52. 223. 25.




CCP
CCC
Arrangement
Selection
2 5
5 2
2 2 3
3 2 2
2 3 2
2 5
2 2 3
[2,3,5]
cnt = 7