Recursion Problems

Question: Print all subsequence of an array

arr = [3,1,2]

Subsequences →
{∅}
3
1
2
3,1
1,2
3,2
3,1,2

→ 8 subsequences

How can we take elements from array to create the subsequences?

V → take
X → Not take

For → {∅} [X 3, X 1, X 2]

For → {3} [✓ 3, X 1, X 2]

For → {1} [X 3, ✓ 1, X 2]

For → {2} [X 3, X 1, ✓ 2]

For → {3,1} [✓ 3, ✓ 1, X 2]

For {1,2} → [X 3, ✓ 1, ✓ 2]

For, {3,2} → [✓ 3, X 1, ✓ 2]

For, {3,1,2} → [✓ 3, ✓ 1, ✓ 2]

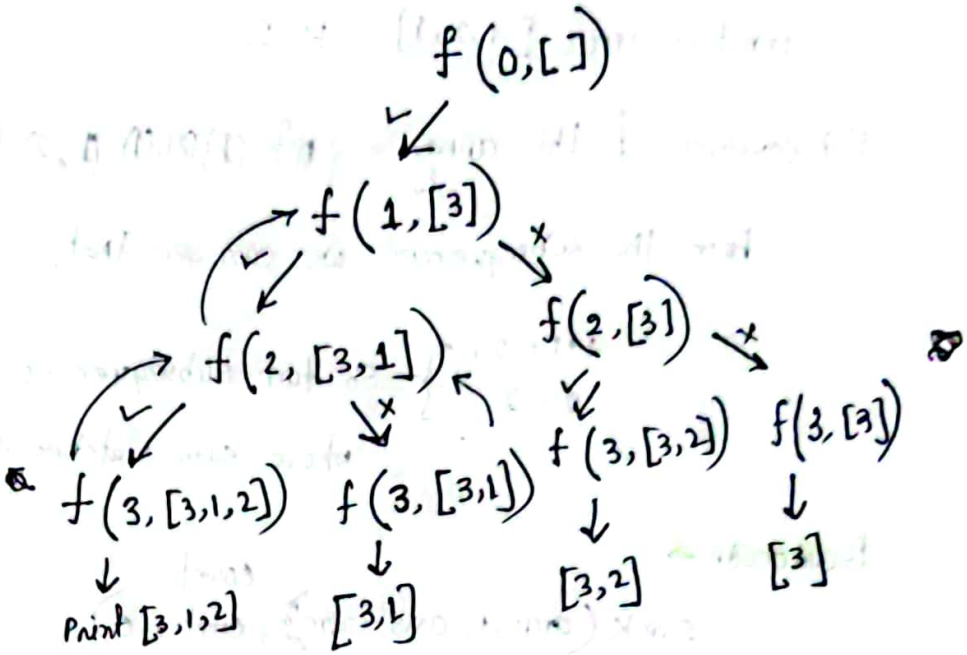
Pseudocode:

input = [3,1,2]

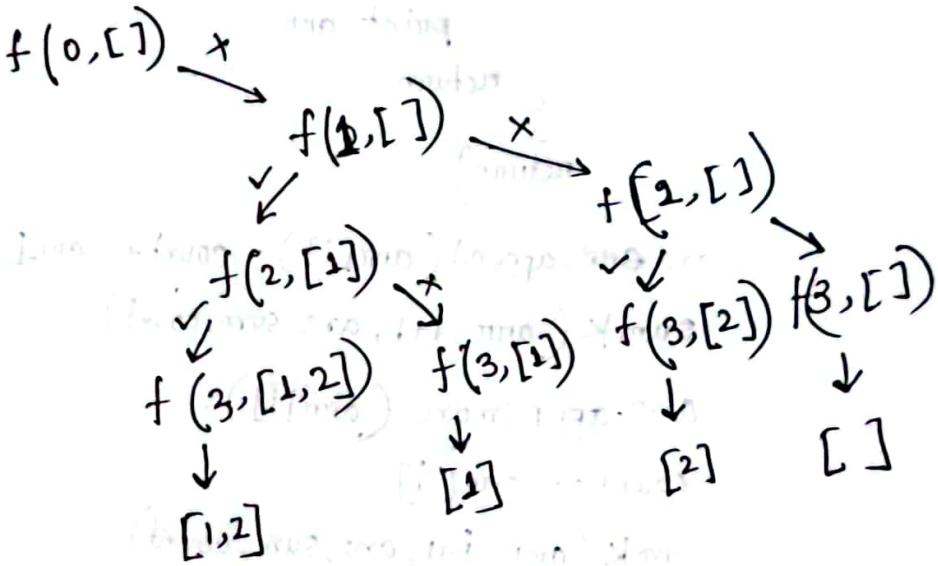
```
Subsequence (input, index, ans) {:
    if (index ≥ len(input))
        Print(ans)
        return
    ans.append(input[index]) → Adding element (Take)
    Subsequences(input, index+1, ans)
    ans.remove(input[index]) → Removing element (Not take)
    Subsequences(input, index+1, ans)
```

Recursion Tree of Subsequences of an array

Left Recursion



Right Recursion:



Question: Print all the subsequence of an array who sum is k.

input → arr [1,2,1] , K=2

Subsequences of this array = {∅}, (1), (2), (1), (1,2), (2,1), (1,2,1)
(1,1)

From the subsequences, we can see that,

1+1 = 2,
2 = 2 } So two subsequences are available that whose sum matches to K.

Pseudocode →

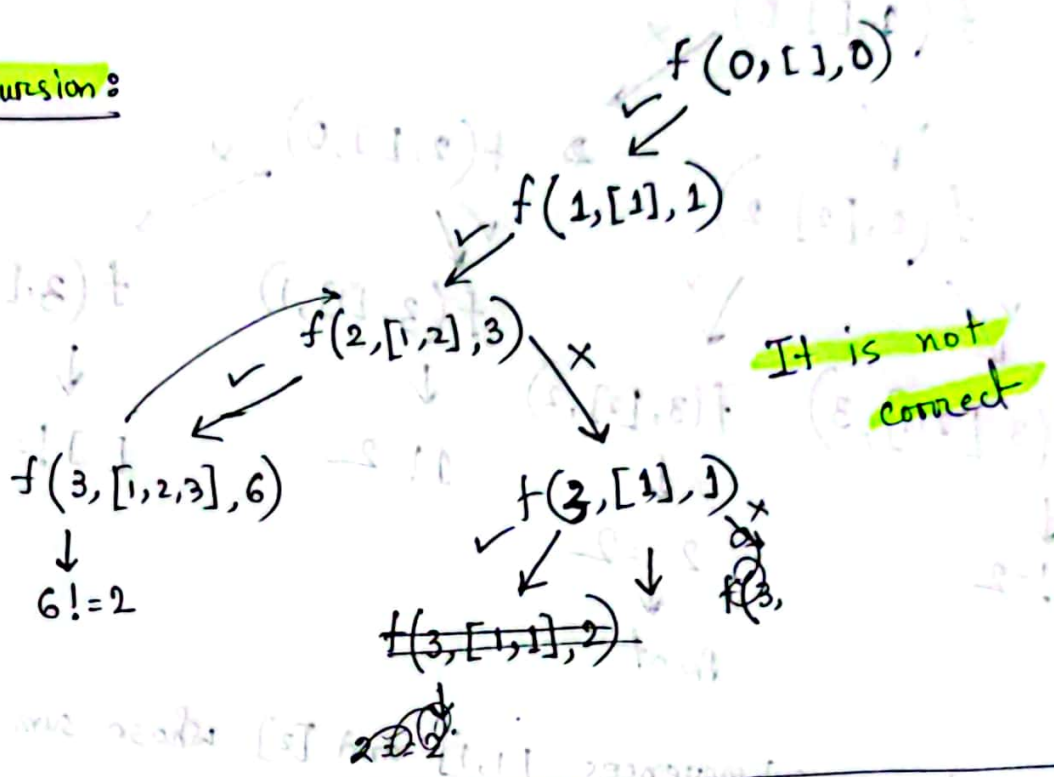
```
sumK(arr, i, ans, sum, count = 0)
{ if (i == len(arr)):
  { if sum == count:
      print arr
    return
  }
  return}

ans arr.append(arr[i]); count += arr[i]   [take]
sumK(arr, i+1, ans, sum, count)
ans.appr remove(arr[i]);
count -= arr[i]
sumK(arr, i+1, ans, sum, count)
```

Recursion Tree of Sum K from subsequence array

Array → [1,2,1] , ~~Sum~~ K = 2

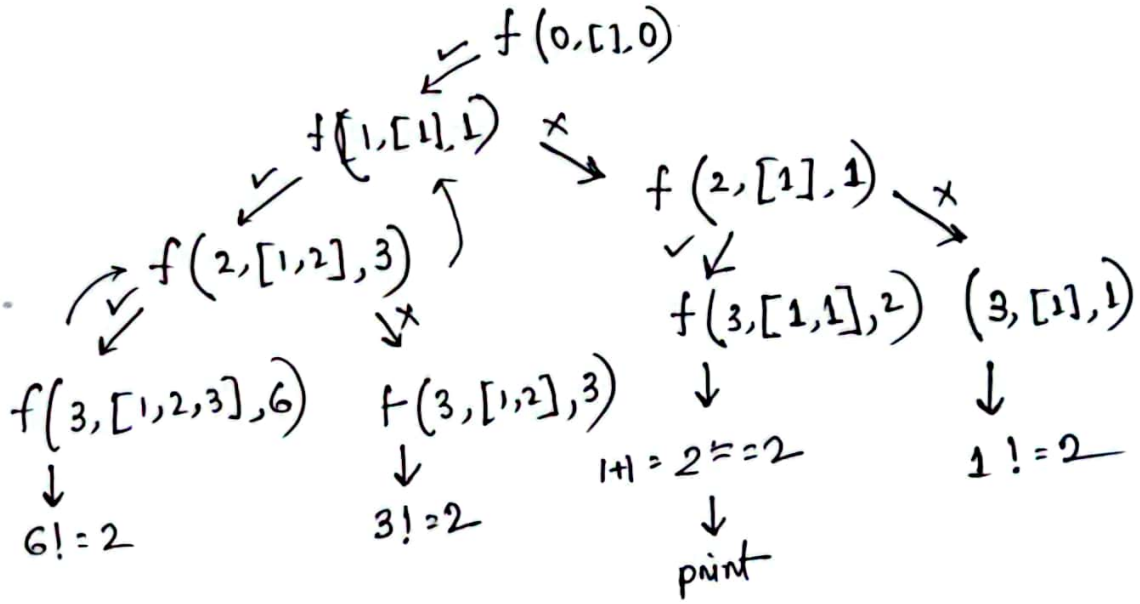
f (index, ans, count)

Left Recursion:



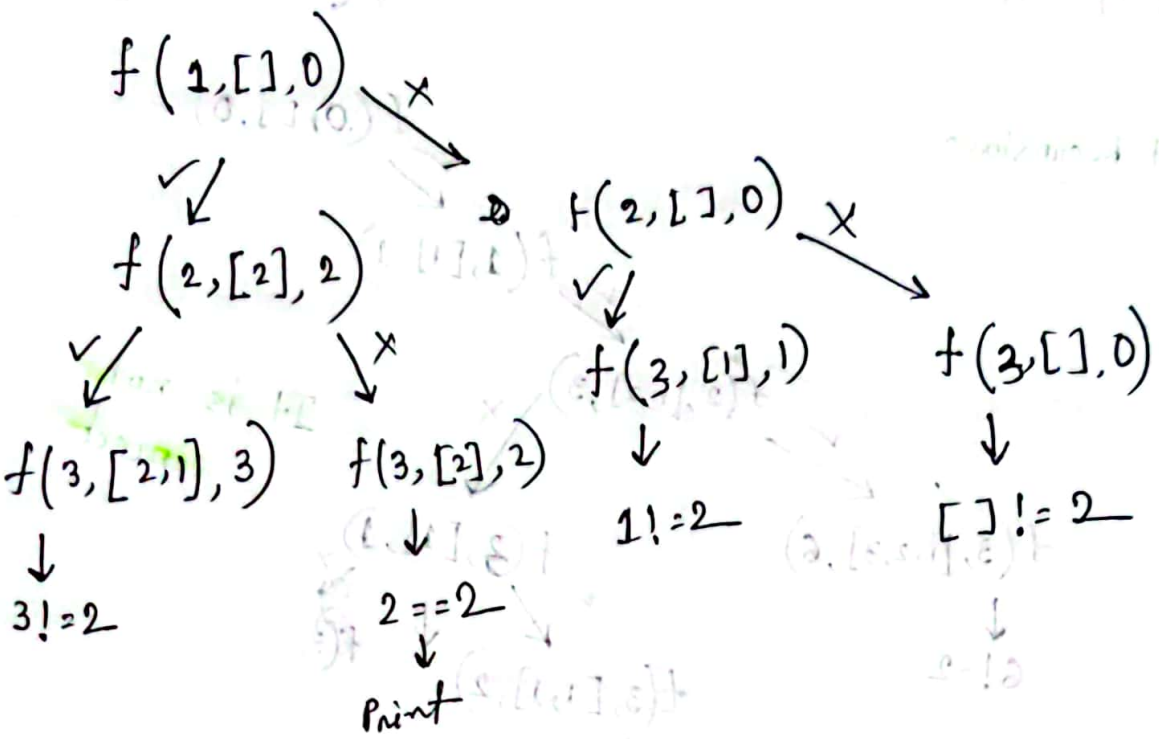
Left Recursion:

f (index, ans, count)



Right Recursion:

f(index, ans, count)



So, we got our subsequences [1,1] and [2] whose sum is K=2