

## < ≡ L6. Recursion on Subsequences | Printing Subsequences

Print all subsequences

To exit full screen, press Esc



TUF



0:02 / 25:00



Print all Subsequences

→ a contiguous / non-contiguous  
sequences, which follows  
the order.

$n = 3$

arr → { 3, 1, 2 }

3 2 1

{ } → ✓  
3 ✓  
1 ✓  
2 ✓  
3 1 ✓  
1 2 ✓  
3 2 ✓  
3 1 2 ✓

→ 8



<

Print all subsequences

→ a contiguous / non-contiguous  
sequences, which follows  
the order.

$n = 3$

arr → { 3, 1, 2 }

3 2 1

{ } → ✓  
3 ✓  
1 ✓  
2 ✓  
3 1 ✓  
1 2 ✓  
3 2 ✓  
3 1 2 ✓

→ 8

## L6. Recursion on Subsequences | Printing Subsequences

$\{3, 1, 2\}$

$\{3, 2\}$

✓  
 $\{3$

✗

✓  
 $2\}$



take / not take

index

✗ ✓ ✓  
\_ \_ \_

$\{1, 2\}$



TUF

$\checkmark$   $\times$   $\checkmark$   
 $\underline{\quad}$   $\underline{\quad}$   $\underline{\quad}$   
 $\{3$   $2\}$

mden

$\times$   $\checkmark$   $\checkmark$   
 $\underline{\quad}$   $\underline{\quad}$   $\underline{\quad}$   
 $[1, 2]$

$\checkmark$   $\checkmark$   $\times$   
 $\underline{\quad}$   $\underline{\quad}$   $\underline{\quad}$   
 $\{3, 1\}$

$\checkmark$   $\checkmark$   $\checkmark$   
 $\underline{\quad}$   $\underline{\quad}$   $\underline{\quad}$   
 $\{3, 1, 2\}$

$\times$   $\times$   $\times$   
 $\underline{\quad}$   $\underline{\quad}$   $\underline{\quad}$   
 $\{\}$



<

arr  $\rightarrow$  [3, 1, 2]  
0 1 2

```
f(ind, arr)
{
    if (ind >= n)
        print(arr)
        return;
    arr.add
```



return;

q1.add(arr[i]);

f(ind + 1, q1);

q1.remove(arr[i]);

f(i,



<

0 1 2

```
f(ind, s)
{
    if (ind >= n)
        print(s)
        return;
    s.add(arr[i]);
    f(ind+1, s);
    s.remove(arr[i]);
    f(ind+1, s);
}
```





< 0 1 2

```
f(ind, s)
```

```
{
```

```
    if (ind >= n)
        print(s)
        return;
```

```
    s.add(arr[i]);
```

```
    f(ind+1, s); → take
```

```
    s.remove(arr[i]);
```

```
    f(ind+1, s); → not take
```

```
}
```



```

f(ind, s)
{
    if (ind >= n)
        print(s)
        return;
    s.add(arr[ind]);
    f(ind+1, s); // take
    s.remove(arr[ind]);
    f(ind+1, s); // not take
}

main()
{
    arr = {3, 1, 2}
    f(0, s)
}

```



$f(1, [3])$

$y() \times$

$[3,1] [3].add(arr[1])$   
 $f(2, [3,1])$

$[3,1,2] [3,1].add(arr[2])$   
 $f(3, [3,1,2])$

< X

{ }

arr  $\rightarrow$   $\begin{bmatrix} 3 & 1 & 2 \end{bmatrix}$   
          0     1     2

x

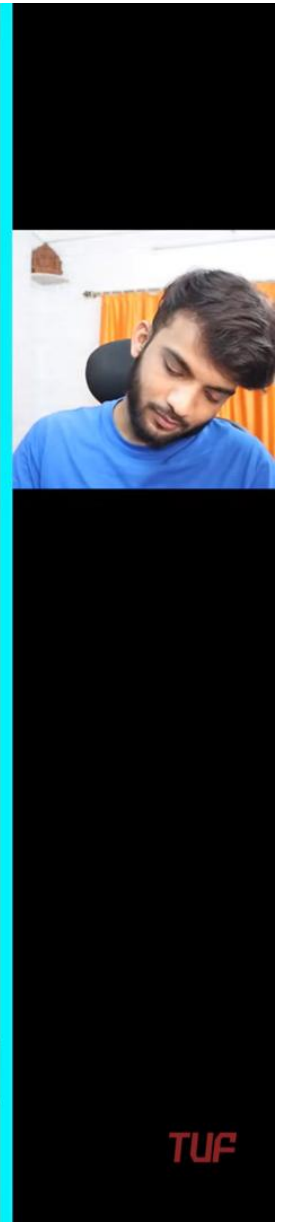
$f(1, [3])$   
  {  
  }  $is() \times$

arr[i]);  
);  $\rightarrow$  take  
arr[i]);  
);  $\rightarrow$  not take

$[3, 1]$   $[3].add(arr[i])$   
           $f(2, [3, 1])$

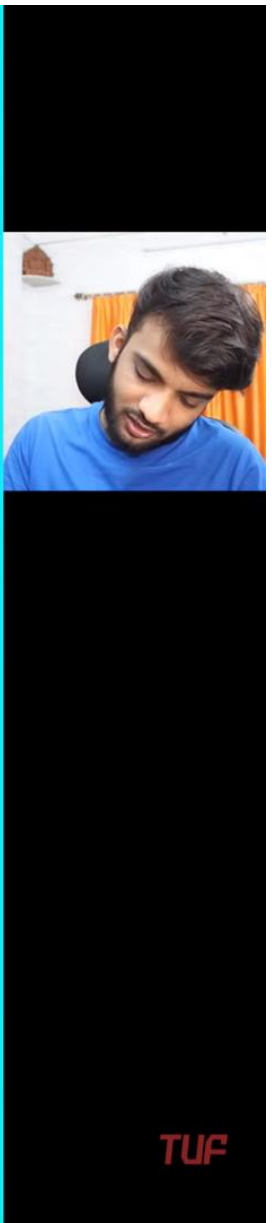
$f(2, [3, 1])$   
  {  
  }  $is() \times$

$[3, 1].add(arr[2])$



$f(3, 1]$   
 $f(3, 1] \cdot \text{add} [\text{arr}[2]]$   
 $f(3, \{3, 1, 2\})$   
 $f(3, 1] \leftarrow \{3, 1, 2\} \cdot \text{remove} (\text{arr}[2])$   
 $f(3, \{3, 1\})$   
 $f(3)$

ground  
 return.



Output

$(2, 1, 2)$   
 $(3, 1)$

✓  $(3, 1, 2)$   
↗

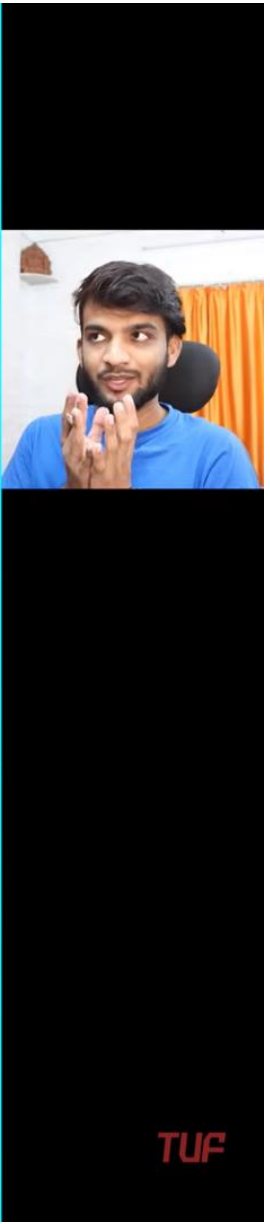
$\underline{3}, 1 = \xrightarrow{x} (3, 1)$

→  $f(2, \boxed{[3, 1]})$   
   $\{$   
   $\hat{y} \quad x$

→  $f(3, \underline{[3, 1, 2]})$   
   $\{$   
   $\hat{y} \text{ (mod } \geq n)$   
   $\checkmark$



$f(1, [3])$   
 $\{ \}$   
 $xy() \times$   
 $[3,1] [3].add(arr[1])$   
 $f(2, [3,1])$   
 $[3,1,2]$   
 $[3,1].add(arr[2])$   
 $f(3, [3,1,2])$   
 $[3,1,2].remove(arr[2])$   
 $f(3, [3,1])$









```

<
{
  f(ind+1, s); // take
}
main()
{
  arr = {3, 1, 2}
  f(0, [])
}

```

$f(2, [3])$

$f(3, [3, 1])$

$f(2, [3])$

$i \text{ (ind)} \geq 3$  ✓  
 print  
 arr

$\underline{3}$

take → 3, 1

→ [3]

✓ (3, 2)

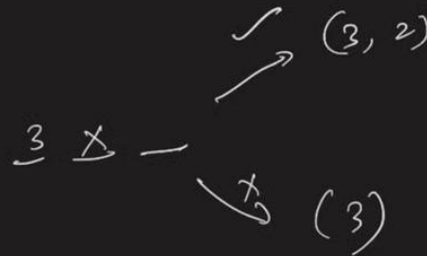
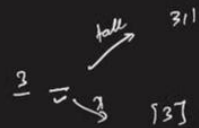
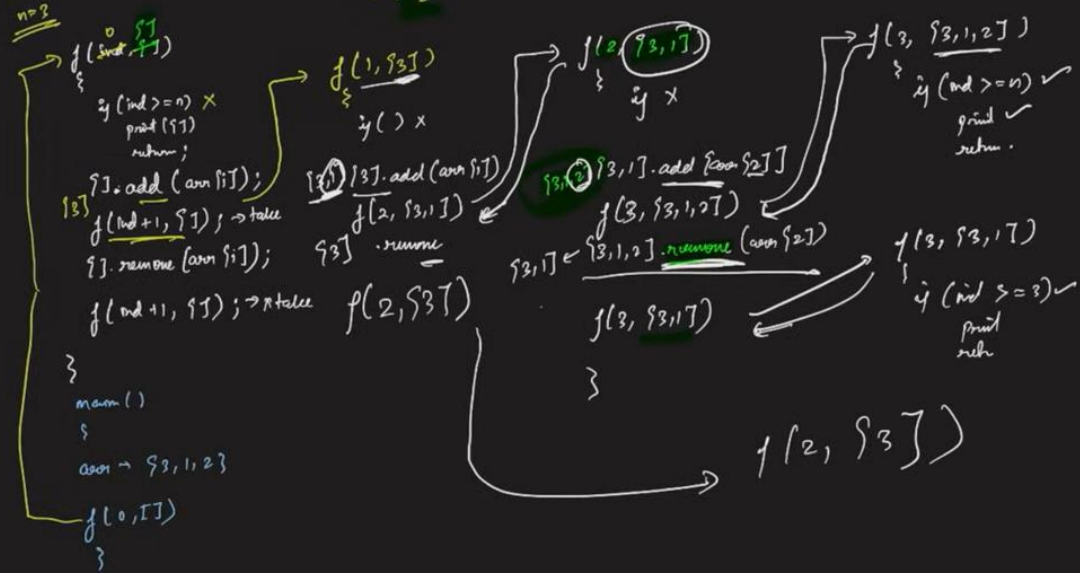
$\underline{3} \quad \underline{X} \quad \text{—}$

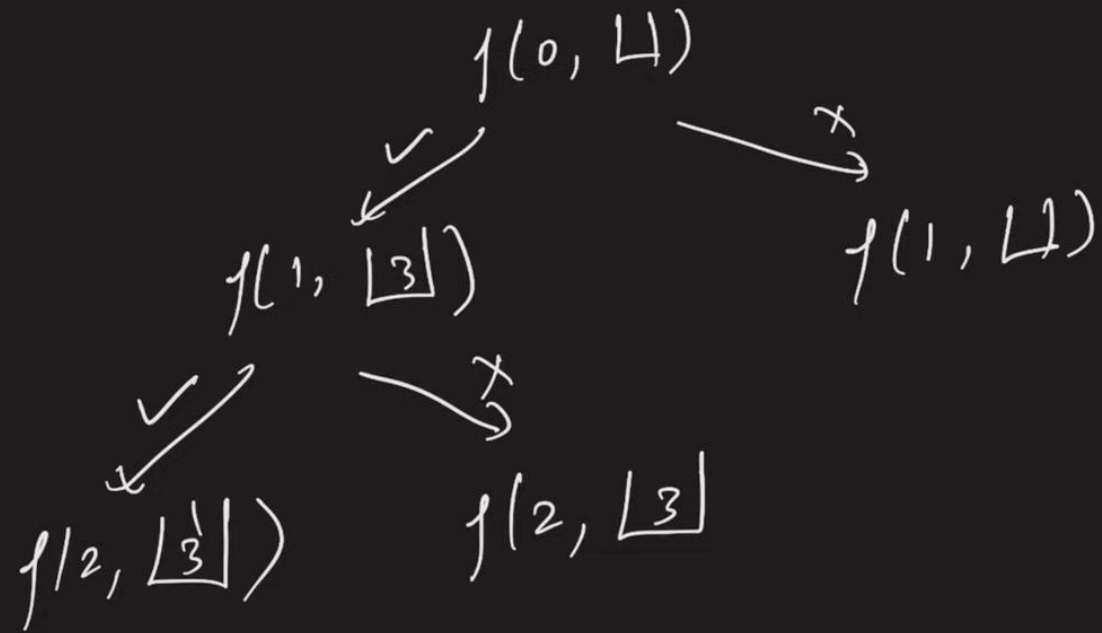
✗ (3)

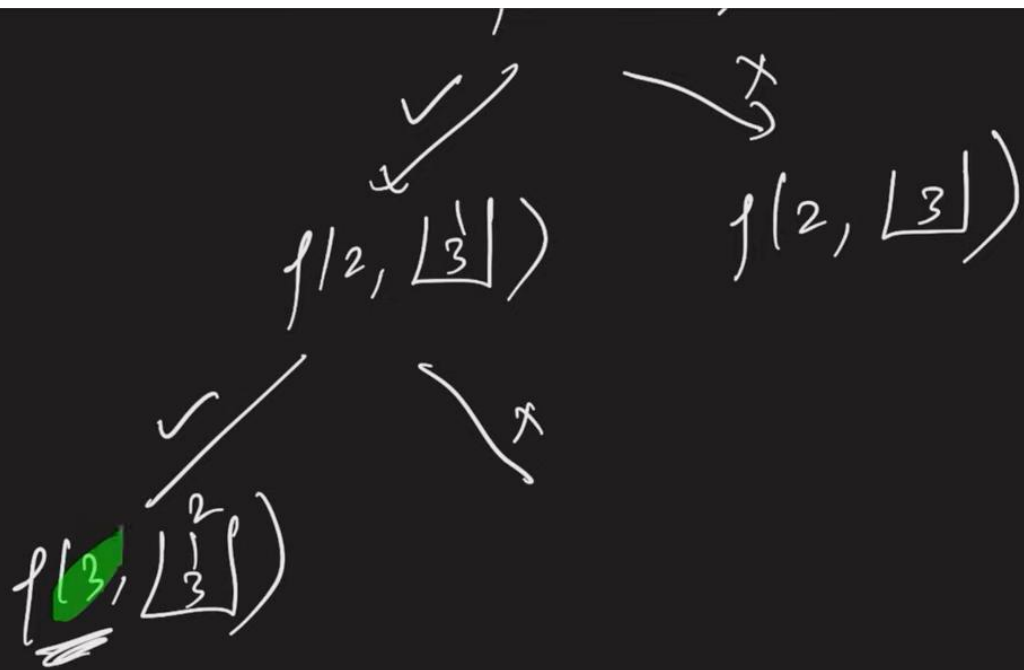


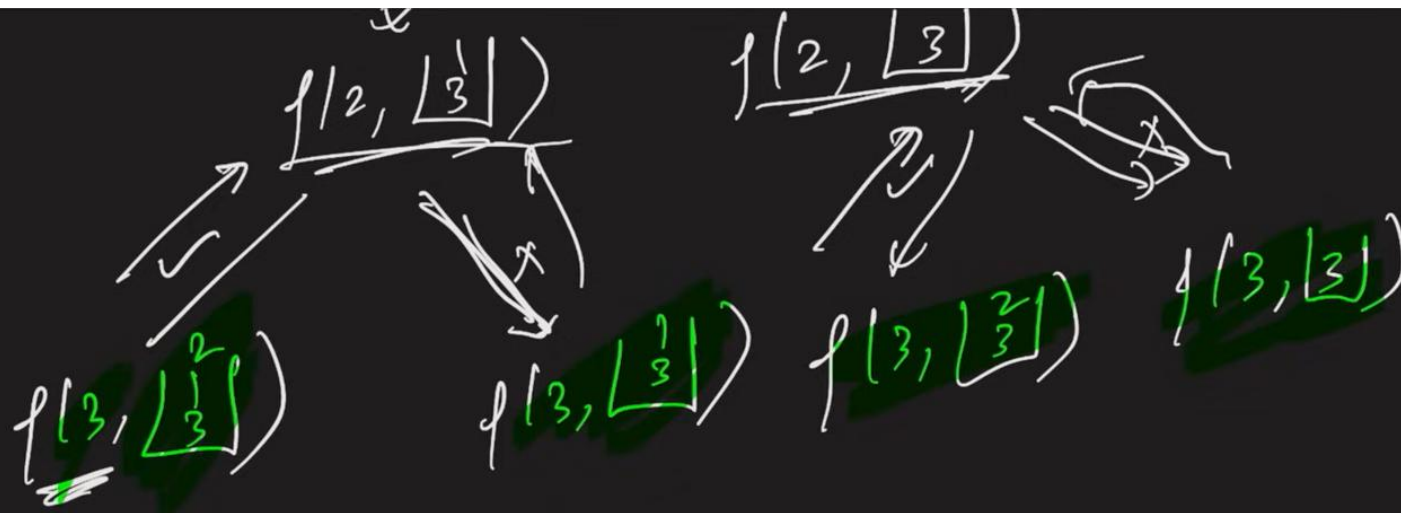
x x x { }

arr → {0, 1, 2}









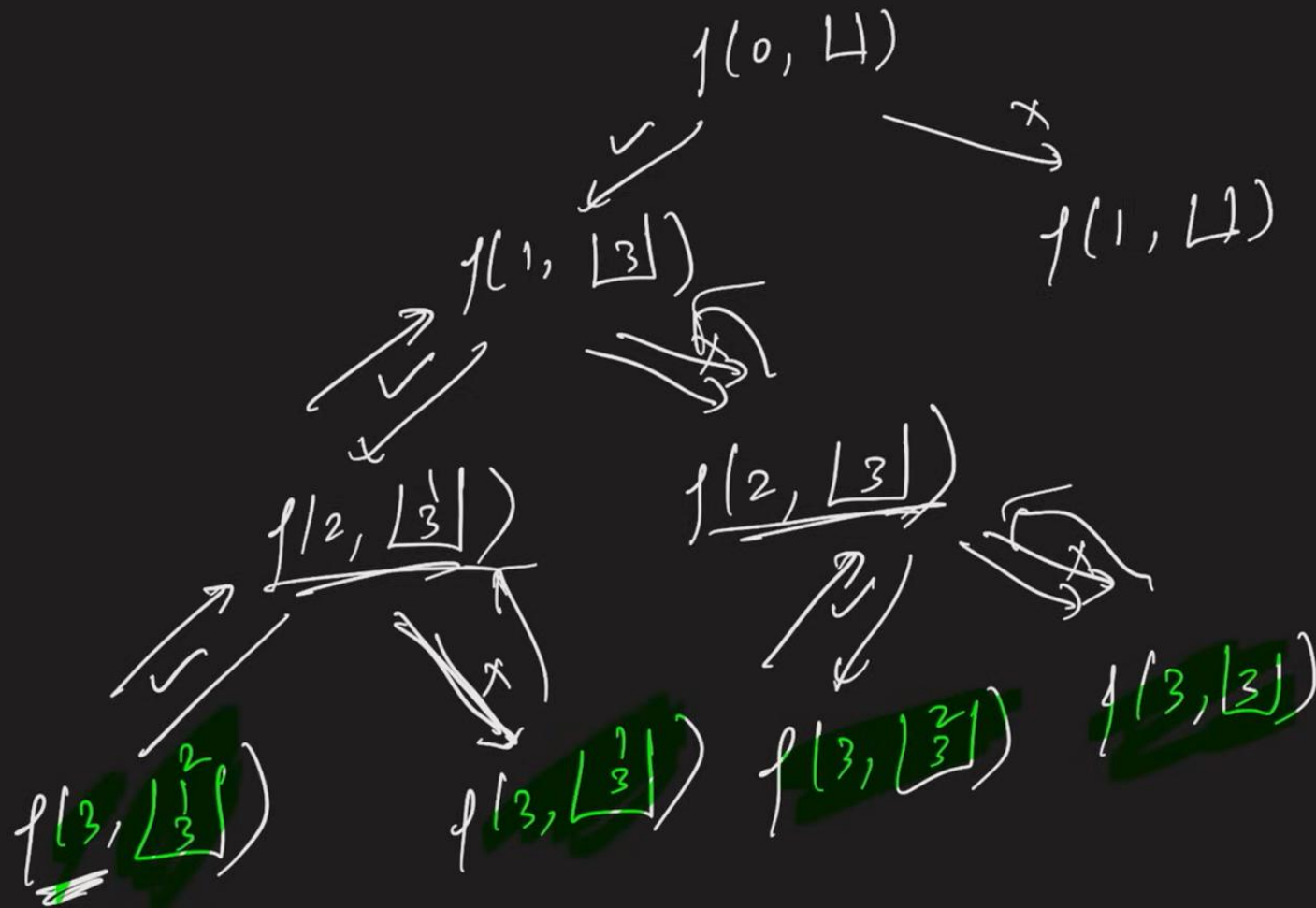
Handwritten sequence of numbers:

```

2 1 2
3 1
3 2
3
  
```

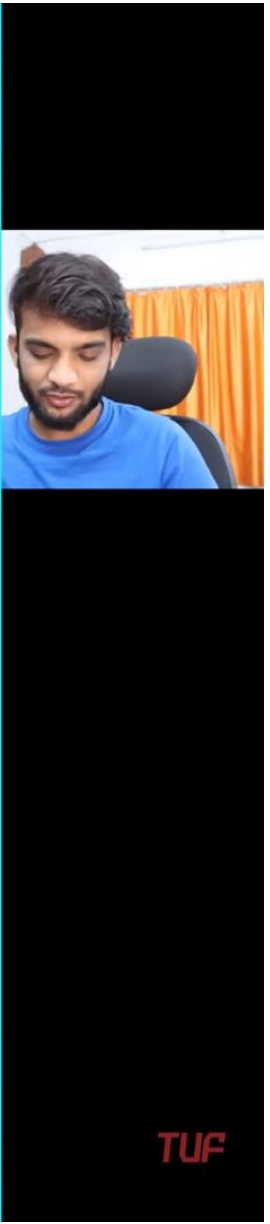
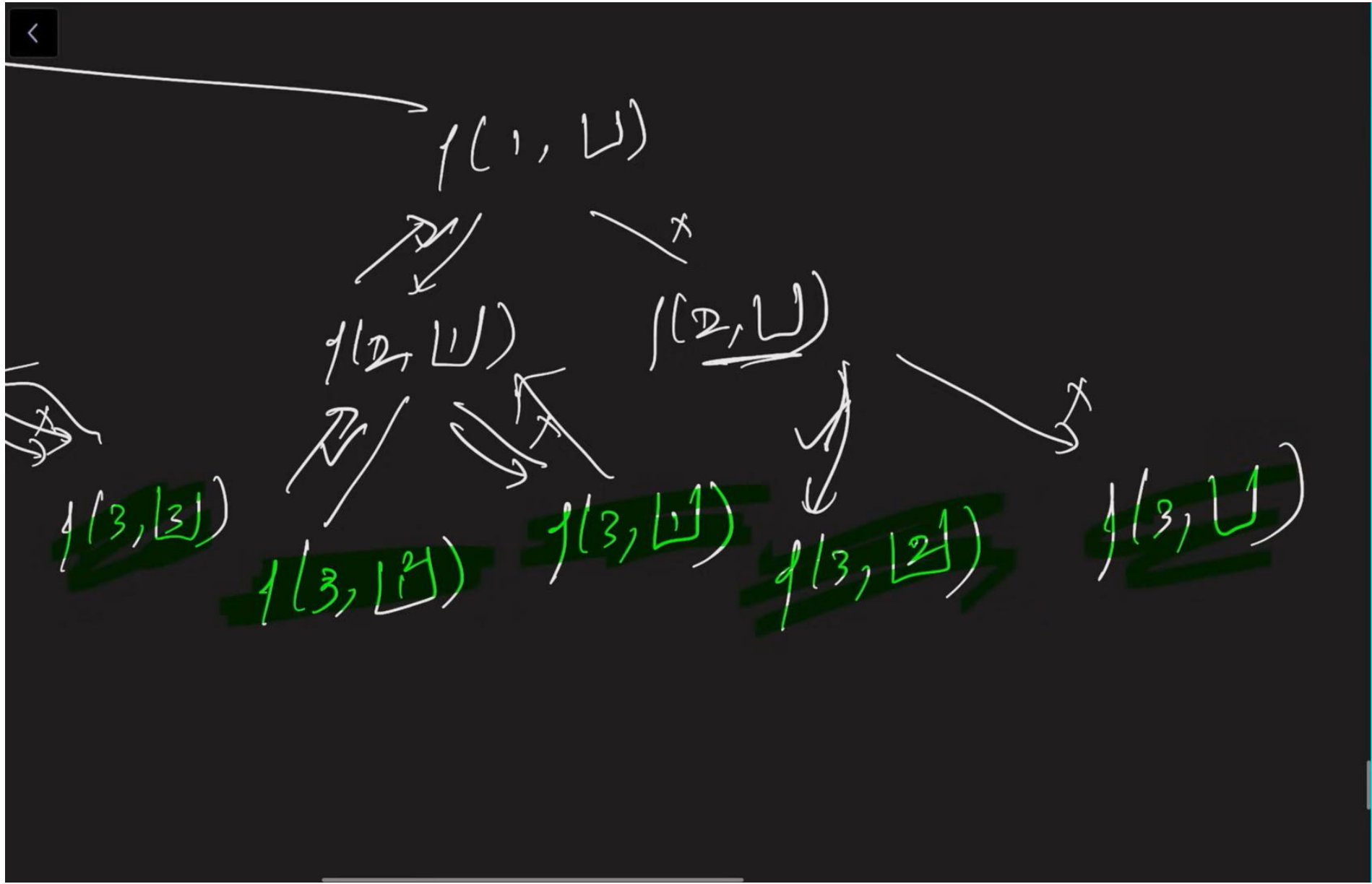
The numbers are grouped by a large right-facing curly bracket on the right side.





2 1 2 }  
3 1 }

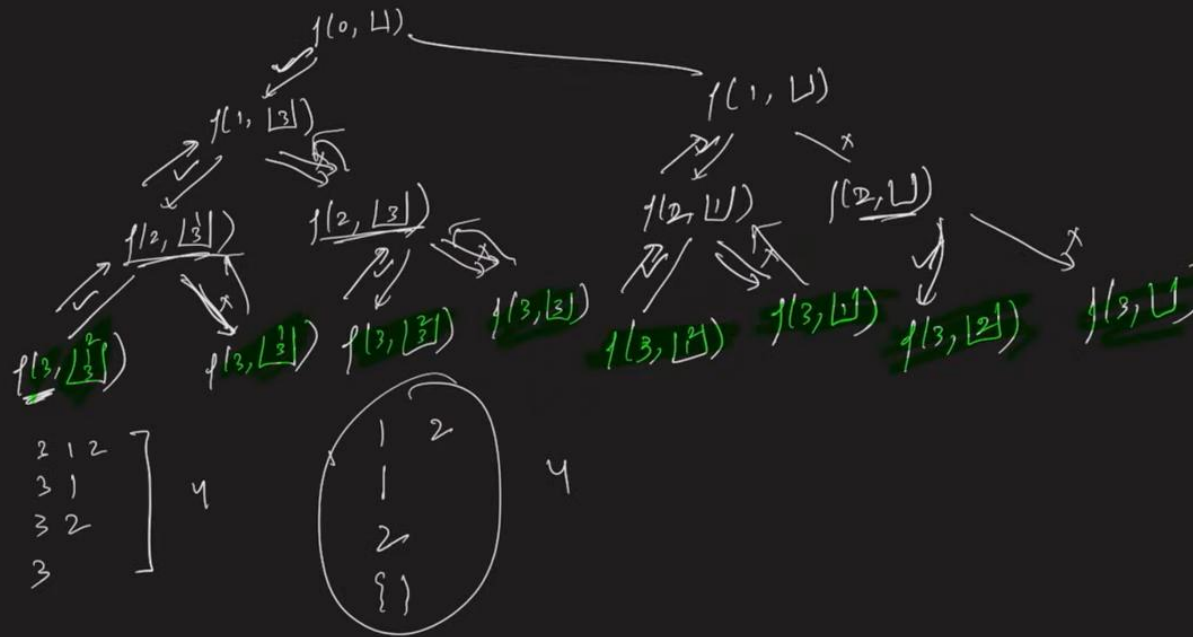


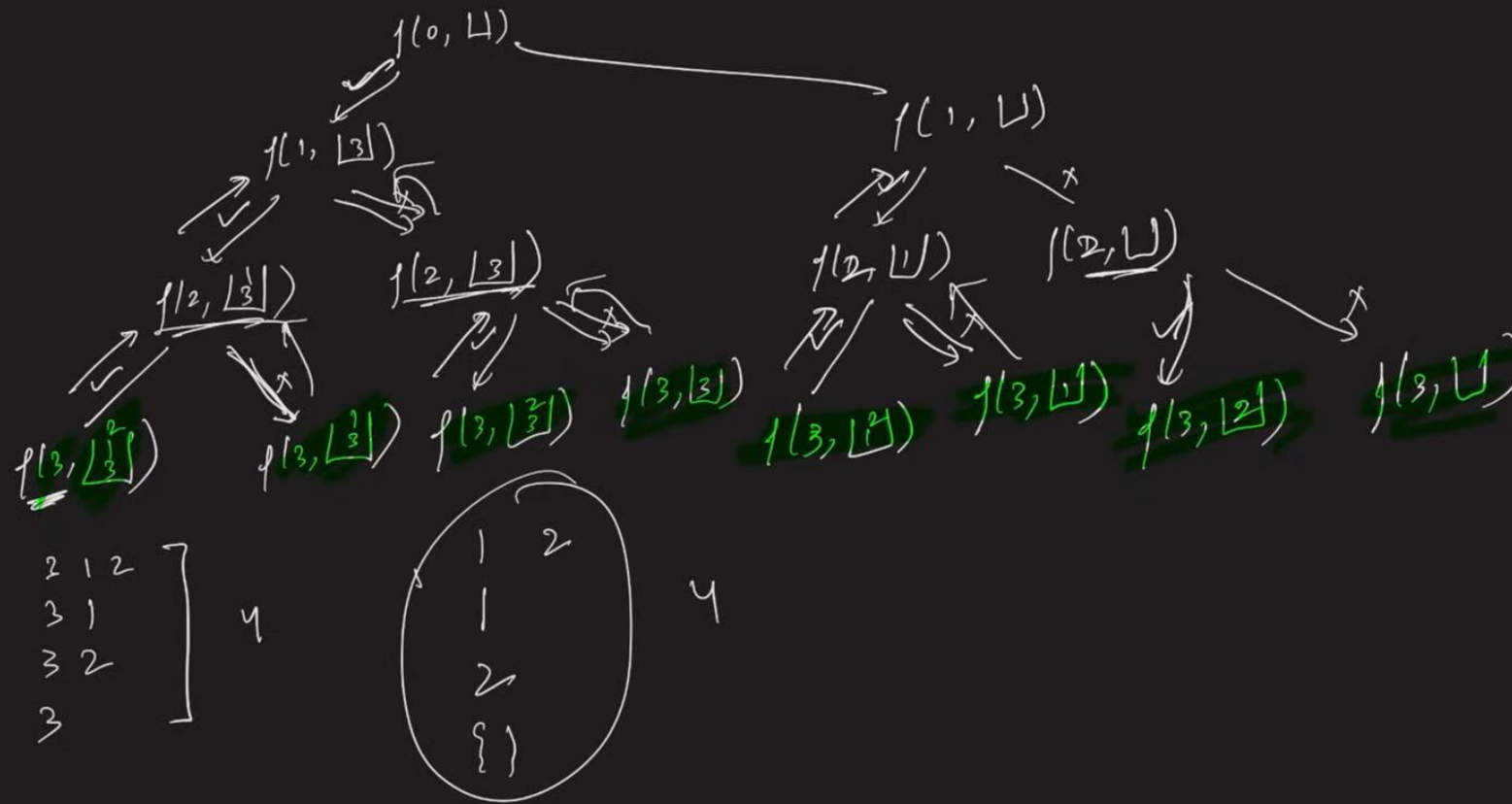




<

given  
 $f(1)$  →







arr  $\rightarrow$  [3, 1, 2]

n=3

f(ind, arr)

if (ind >= n)  $\times$   
print(arr)  
return;

arr.add(arr[ind]);

f(ind+1, arr);  $\rightarrow$  take

~~arr.remove(arr[ind]);~~

f(ind+1, arr);  $\rightarrow$  not take

}

main()

{

arr  $\rightarrow$  [3, 1, 2]

f(0, arr)

}

f(1, [3])

if ()  $\times$

[3].add(arr[1])

f(2, [3, 1])

[3].remove

f(2, [3])

f(2, [3, 1])

if  $\times$

[3, 1].add(arr[2])

f(3, [3, 1, 2])

[3, 1]  $\leftarrow$  [3, 1, 2].remove(arr[2])

f(3, [3, 1])

}

f(3, [3, 1, 2])

if (ind >= n)  $\checkmark$   
print  $\checkmark$   
return.

f(3, [3, 1])

if (ind >= 3)  $\checkmark$   
print  
return

f(2, [3])

example to remember

3  $\xrightarrow{\text{take}}$  3, 1  
3  $\xrightarrow{\text{not take}}$  3

(3, 2)

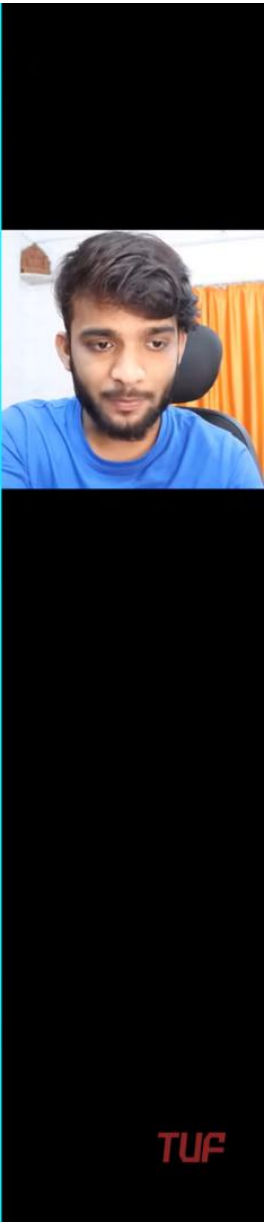
TUF

```
code.cpp  stdc++.h  input.txt
1  #include<bits/stdc++.h>
2  using namespace std;
3  void printF(int ind, vector<int> &ds, int arr[], int n) {
4      if(ind == n) {
5          for(auto it : ds) {
6              cout << it << " ";
7          }
8          cout << endl;
9          return;
10     }
11     // take or pick the particular index into the subsequence
12     ds.push_back(arr[ind]);
13 }
14 int main() {
15     #ifndef ONLINE_JUDGE
16     freopen("input.txt", "r", stdin);
17     freopen("output.txt", "w", stdout);
18     #endif
19     int arr[] = {3, 1, 2};
20     int n = 3;
21     vector<int> ds;
22     printF(0, ds, arr, n);
23
24     return 0;
25 }
```

```
input.txt
1 5
2 1 2 3 4 5
```

```
output.txt
1 3
```

Finished in 1.6s]

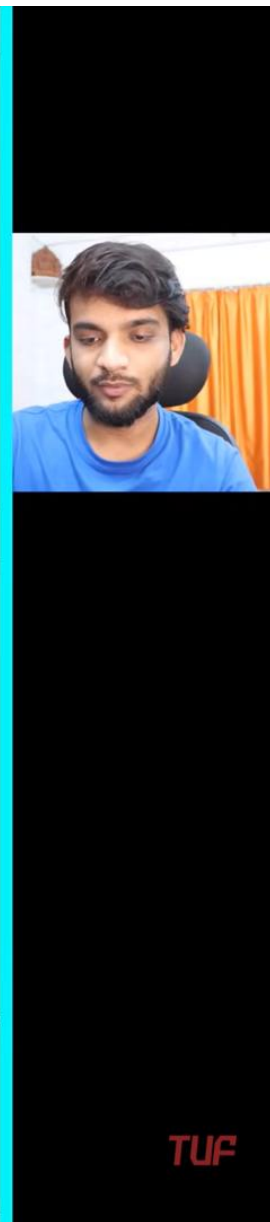


```
code.cpp  x  stdc++.h  x
1  #include<bits/stdc++.h>
2  using namespace std;
3  void printF(int ind, vector<int> &ds, int arr[], int n) {
4      if(ind == n) {
5          for(auto it : ds) {
6              cout << it << " ";
7          }
8          cout << endl;
9          return;
10     }
11     // take or pick the particular index into the subsequence
12     ds.push_back(arr[ind]);
13     printF(ind+1, ds, arr, n);
14     ds.pop_back();
15
16     // not pick, or not take condition, this elemnt is not added to your subsequence
17     printF(ind+1, ds, arr, n);
18 }
19 int main() {
20     #ifndef ONLINE_JUDGE
21     freopen("input.txt", "r", stdin);
22     freopen("output.txt", "w", stdout);
23     #endif
24     int arr[] = {3, 1, 2};
25     int n = 3;
26     vector<int> ds;
27     printF(0, ds, arr, n);
28
29     return 0;
30 }
```

```
input.txt  x
1  5
2  1 2 3 4 5

output.txt  x
1  3 1 2
2  3 1
3  3 2
4  3
5  1 2
6  1
7  2
8
9
```

Finished in 2.0s]

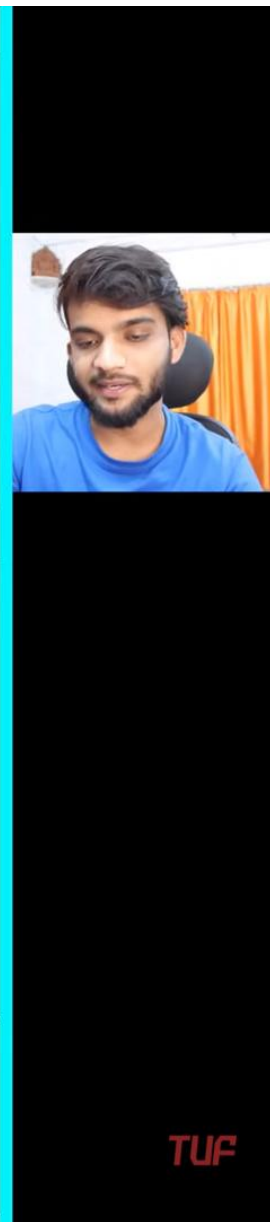


```
code.cpp  stdc++.h  input.txt
1  #include<bits/stdc++.h>
2  using namespace std;
3  void printF(int ind, vector<int> &ds, int arr[], int n) {
4      if(ind == n) {
5          for(auto it : ds) {
6              cout << it << " ";
7          }
8          if(ds.size() == 0) {
9              cout << "{}";
10         }
11         cout << endl;
12         return;
13     }
14     // take or pick the particular index into the subsequence
15     ds.push_back(arr[ind]);
16     printF(ind+1, ds, arr, n);
17     ds.pop_back();
18
19     // not pick, or not take condition, this elemnt is not added to your subsequence
20     printF(ind+1, ds, arr, n);
21 }
22 int main() {
23     #ifndef ONLINE_JUDGE
24     freopen("input.txt", "r", stdin);
25     freopen("output.txt", "w", stdout);
26     #endif
27     int arr[] = {3, 1, 2};
28     int n = 3;
29     vector<int> ds;
30     printF(0, ds, arr, n);
31 }
```

```
input.txt
1 5
2 1 2 3 4 5
```

```
output.txt
1 3 1 2
2 3 1
3 3 2
4 3
5 1 2
6 1
7 2
8 {}
9
```

Finished in 1.1s



TUF

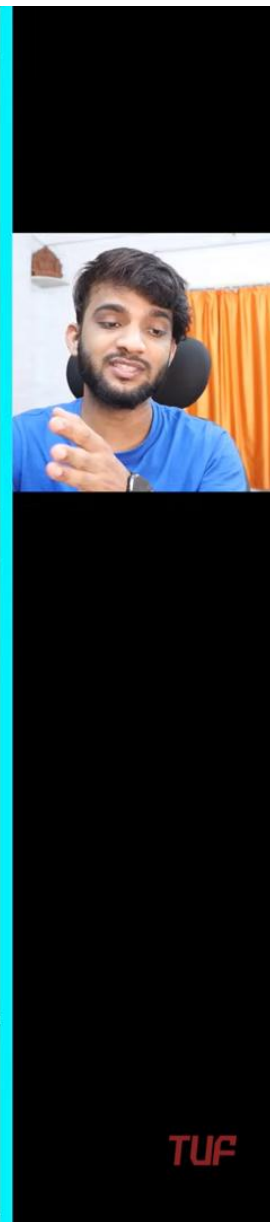


```
code.cpp  x  stdc++.h  x
7      }
8      if(ds.size() == 0) {
9          cout << "{}";
10     }
11     cout << endl;
12     return;
13 }
14
15 // not pick, or not take condition, this elemnt is not added to your subsequenc
16 printf(ind+1, ds, arr, n);
17
18
19 // take or pick the particular index into the subsequence
20 ds.push_back(arr[ind]);
21 printf(ind+1, ds, arr, n);
22 ds.pop_back();
23
24 }
25
26 int main() {
27     #ifndef ONLINE_JUDGE
28     freopen("input.txt", "r", stdin);
29     freopen("output.txt", "w", stdout);
30     #endif
31     int arr[] = {3, 1, 2};
32     int n = 3;
33     vector<int> ds;
34     printf(0, ds, arr, n);
35
36     return 0;
37 }
```

```
input.txt  x
1 5
2 1 2 3 4 5

output.txt  x
1 {}
2 2
3 1
4 1 2
5 3
6 3 2
7 3 1
8 3 1 2
9
```

Finished in 1.3s]

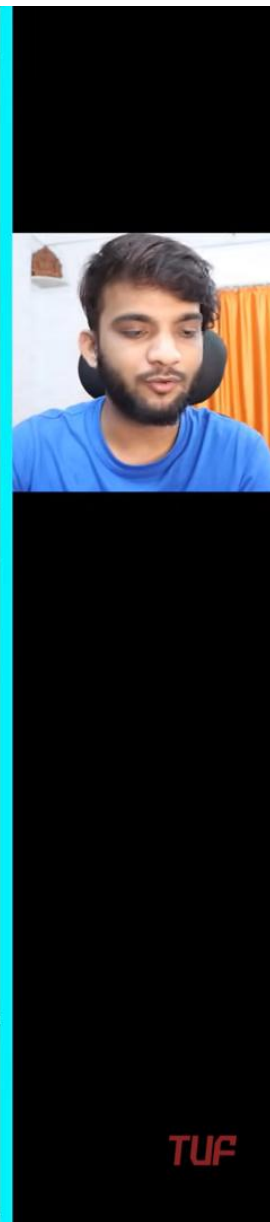


```
code.cpp  x  stdc++.h  x
7      }
8      if(ds.size() == 0) {
9          cout << "{}";
10     }
11     cout << endl;
12     return;
13 }
14
15 // not pick, or not take condition, this element is not added to your subsequence
16 printf(ind+1, ds, arr, n);
17
18
19 // take or pick the particular index into the subsequence
20 ds.push_back(arr[ind]);
21 printf(ind+1, ds, arr, n);
22 ds.pop_back();
23
24
25 }
26 int main() {
27     #ifndef ONLINE_JUDGE
28     freopen("input.txt", "r", stdin);
29     freopen("output.txt", "w", stdout);
30     #endif
31     int arr[] = {3, 1, 2};
32     int n = 3;
33     vector<int> ds;
34     printf(0, ds, arr, n);
35
36     return 0;
37 }
```

```
input.txt  x
1 5
2 1 2 3 4 5

output.txt  x
1 {}
2 2
3 1
4 1 2
5 3
6 3 2
7 3 1
8 3 1 2
9
```

Finished in 1.3s





```
code.cpp      stdc++.h
1  #include<bits/stdc++.h>
2  using namespace std;
3  void printF(int ind, vector<int> &ds, int arr[], int n) {
4      if(ind == n) {
5          for(auto it : ds) {
6              cout << it << " ";
7          }
8          if(ds.size() == 0) {
9              cout << "{}";
10         }
11         cout << endl;
12         return;
13     }
14
15     // not pick, or not take condition, this elemnt is not added to your subsequence
16     printF(ind+1, ds, arr, n);
17
18     // take or pick the particular index into the subsequence
19     ds.push_back(arr[ind]);
20     printF(ind+1, ds, arr, n);
21     ds.pop_back();
22 }
23
24 // 2^n
25 int main() {
26     #ifndef ONLINE_JUDGE
27     freopen("input.txt", "r", stdin);
28     freopen("output.txt", "w", stdout);
29     #endif
30     int arr[] = {3, 1, 2};
31     int n = 3;
32 }
```

$$2^n \times n$$

input.txt

```
1 5
2 1 2 3 4 5
```

output.txt

```
1 {}
2 2
3 1
4 1 2
5 3
6 3 2
7 3 1
8 3 1 2
9
```

Finished in 1.3s



```
code.cpp      stdc++.h
1 #include<bits/stdc++.h>
2 using namespace std;
3 void printF(int ind, vector<int> &ds, int arr[], int n) {
4     if(ind == n) {
5         for(auto it : ds) {
6             cout << it << " ";
7         }
8         if(ds.size() == 0) {
9             cout << "{}";
10        }
11        cout << endl;
12        return;
13    }
14
15    // not pick, or not take condition, this elemnt is not added to your subsequence
16    printF(ind+1, ds, arr, n);
17
18    // take or pick the particular index into the subsequence
19    ds.push_back(arr[ind]);
20    printF(ind+1, ds, arr, n);
21    ds.pop_back();
22 }
23
24 // 22222
25 int main() {
26     #ifndef ONLINE_JUDGE
27     freopen("input.txt", "r", stdin);
28     freopen("output.txt", "w", stdout);
29     #endif
30     int arr[] = {3, 1, 2};
31     int n = 3;
32 }
```

$$2^n \times n$$

$$O(N)$$

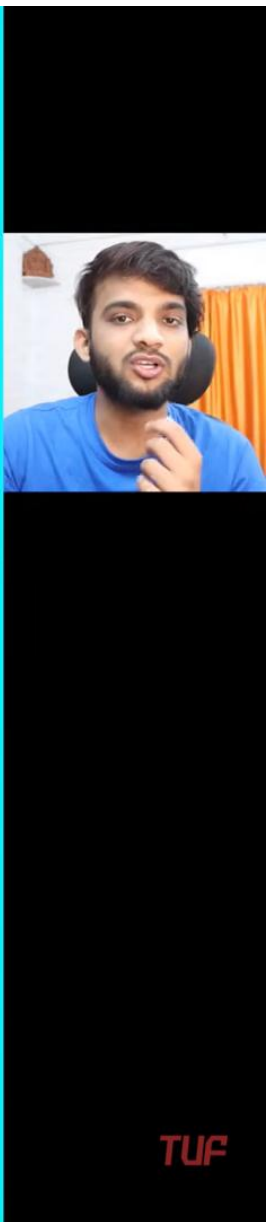
input.txt

```
1 5
2 1 2 3 4 5
```

output.txt

```
1 {}
2 2
3 1
4 1 2
5 3
6 3 2
7 3 1
8 3 1 2
9
```

Finished in 1.3s]



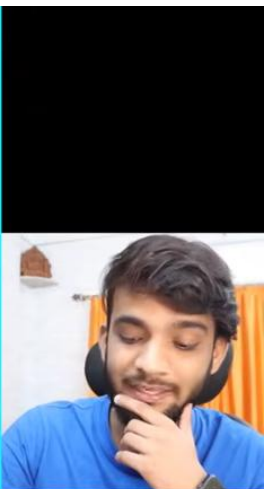
```
code.cpp      stdc++.h
1 #include<bits/stdc++.h>
2 using namespace std;
3 void printF(int ind, vector<int> &ds, int arr[], int n) {
4     if(ind == n) {
5         for(auto it : ds) {
6             cout << it << " ";
7         }
8         if(ds.size() == 0) {
9             cout << "{}";
10        }
11        cout << endl;
12        return;
13    }
14
15    // not pick, or not take condition, this elemnt is not added to your subsequenc
16    printF(ind+1, ds, arr, n);
17
18    // take or pick the particular index into the subsequence
19    ds.push_back(arr[ind]);
20    printF(ind+1, ds, arr, n);
21    ds.pop_back();
22 }
23
24 // 22222
25 int main() {
26     #ifndef ONLINE_JUDGE
27     freopen("input.txt", "r", stdin);
28     freopen("output.txt", "w", stdout);
29     #endif
30     int arr[] = {3, 1, 2};
31     int n = 3;
32 }
```

TC  $2^n \times n$

SC  $\rightarrow O(1)$

```
input.txt
1 5
2 1 2 3 4 5

output.txt
1 {}
2 2
3 1
4 1 2
5 3
6 3 2
7 3 1
8 3 1 2
9
```



Finished in 1.3s]