

L6. Recursion on Subsequences | Printing Subsequences

Print all subsequences

To exit full screen, press Esc



TUF

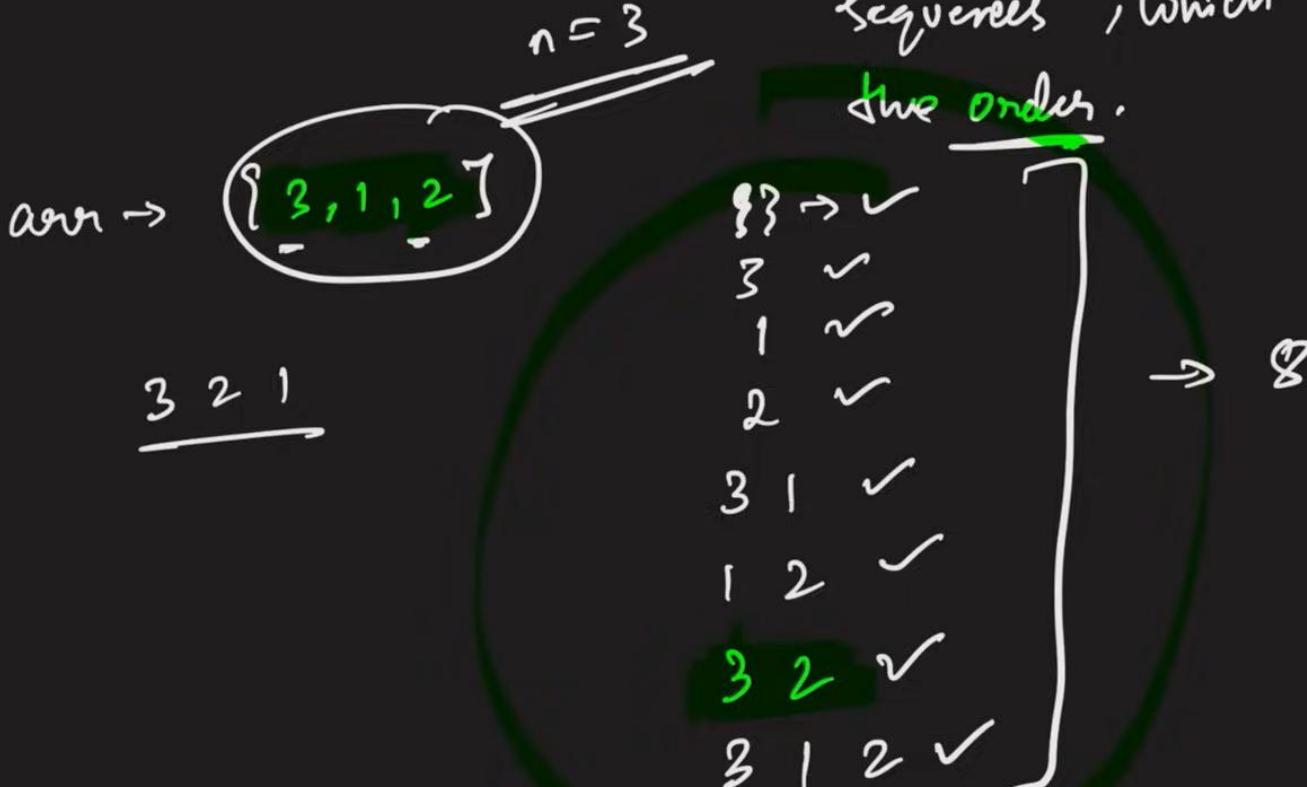


◀ ▶ ⏸ 0:02 / 25:00

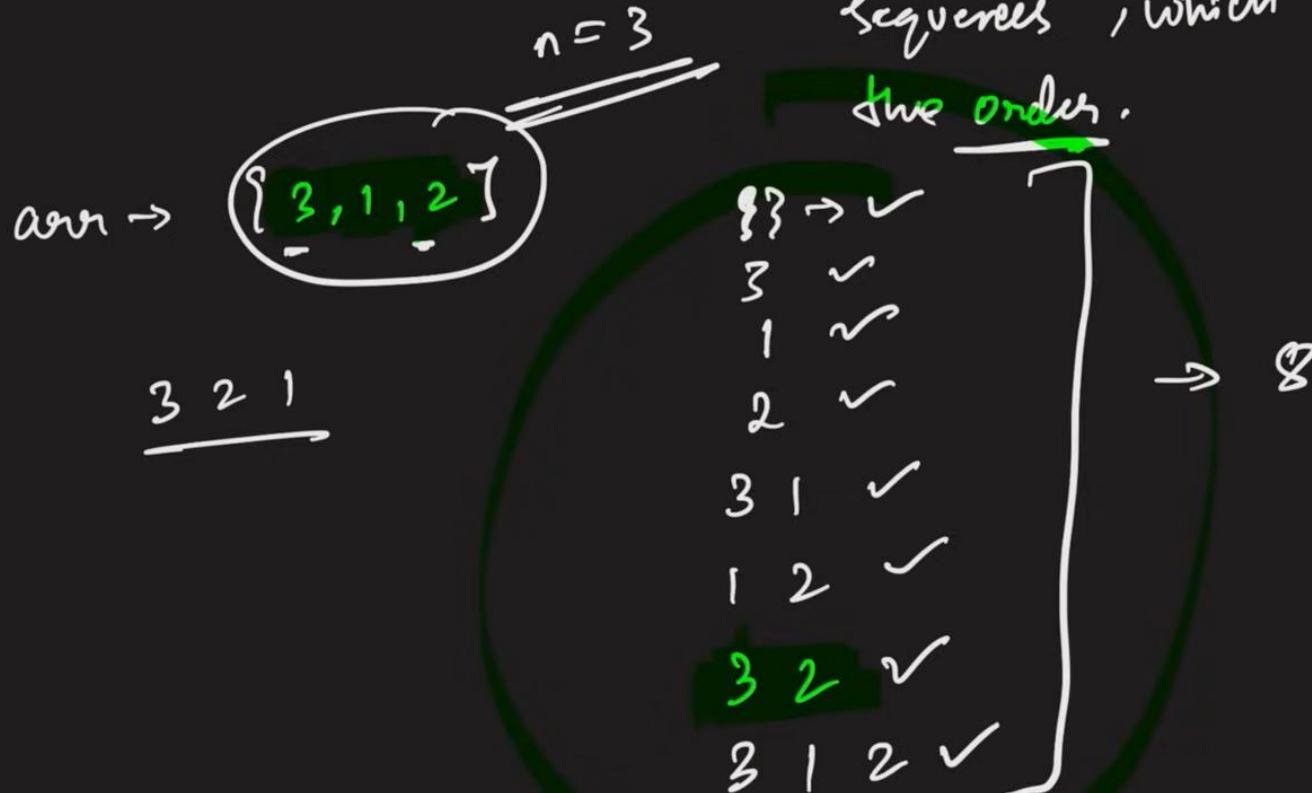


Point all Subsequences

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



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



L6. Recursion on Subsequences | Printing Subsequences



{3, 1, 2}

{3, 2}

✓ ~~✗~~ ✓
— — 2]

take / not take

index

✗ ✓ ✓
— — —

{1, 2}

-3 ~~1~~ 2] Index

~~1~~ ✓ ✓ $[1, 2]$

✓ ~~✓~~ ~~1~~ $\{3, 1\}$

~~1~~ ~~✓~~ ~~✓~~ $\{3, 1, 2\}$

~~1~~ ~~✓~~ ~~✓~~ $\{\}$

$\text{arr} \rightarrow [3, 1, 2]$

0 1 2

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



```
' return ;  
q].add (arr [i]);  
j (ind + 1, q]);  
q].remove (arr [i]);  
  
f (
```



0 1 2

```
f(ind, S)
{
    if(ind >= n)
        print(S)
        return;
    S.add(avr Si];
    f(ind+1, S);
    S.remove(avr Si]);
    f(ind+1, S);
}
```



TUF

0 1 2

f(ind, { })
{

if (ind >= n)
 print({ })
 return;

{ }.add(avr[i]);

f(ind + 1, { }); \rightarrow take

{ }.remove(avr[i]);

f(ind + 1, { }); \rightarrow not take

2



TUF

```
f(ind, arr)
{
    if (ind >= n)
        print (arr)
        return;
    arr.add (arr[i]);
    f(ind+1, arr);  $\rightarrow$  take
    arr.remove (arr[i]);
    f(ind+1, arr);  $\rightarrow$  not take
}
main()
{
    arr  $\rightarrow$  {3, 1, 2}
}
f(0, arr)
```



$f(1, s_3])$

$y()x$

$s_3, s_1] s_3]. add(\text{arr } s_1)$

$f[2, s_3, s_1]$

$f \{ \cdot \cdot \cdot \}$
 $y \times$

$s_3, s_1, s_2] s_3, s_1]. add(s_2)$

$f(3, s_3, s_1, s_2)$



< Σ $\{ \}$
 x $y(x)$
 $\{ s_i \};$ $f(1, s_3)$
 $); \rightarrow \text{take}$ $\{ s_3, 1 \} \xrightarrow{\quad} f[2, s_3, 1]$
 $\{ s_i \};$ $f(2, s_3, 1) \xrightarrow{\quad} f(2, s_3, 1]$
 $); \rightarrow \text{take}$
 $f(2, s_3, 1] \xrightarrow{\quad} f(2, s_3, 1]$
 $\{ s_3, 1 \} \cdot \text{add}(\text{area } s_1)$
 $f(2, s_3, 1)$
 $\{ s_3, 1 \} \cdot \text{add}(\text{area } s_2)$



$\langle \dots \rangle$
return $S_1 \rangle$

$\langle \dots \rangle$
return $\{S_3, 1\} \cdot \underline{\text{add}} \left[\text{over } S_2 \right]$

$\{S_3, 1, 2\}$

$f(3, \{S_3, 1, 2\})$

$\{S_3, 1\} \leftarrow \{S_3, 1, 2\} \cdot \underline{\text{remove}} \left[\text{over } S_2 \right]$

$f(3, \{S_3, 1\})$

return
return .





Output

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

$\nearrow \quad (3, 1, 2)$
 \searrow

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

$\rightarrow f(2, \{3, 1\})$
 $\left\{ \begin{matrix} y \\ x \end{matrix} \right.$

$\rightarrow f(3, \underline{\{3, 1, 2\}})$
 $\left\{ \begin{matrix} y \\ (m >= n) \\ \dots \end{matrix} \right.$

TUF



0 1 2

)
= n) X
(
m;
_ (arr s_i);
, s_i); → take
me [arr s_i];

1, s_i); → n take

$f(1, s_3)$
 \downarrow
 $y() X$

$f(2, \{s_3, 1\})$
 \downarrow
 $y X$

$\{s_3, 1\} \{s_3\}.add(\text{arr } s_1)$
 $f(2, \{s_3, 1\})$

$s_3, 1 \circled{2} \{s_3, 1\}.add(\text{arr } s_2)$
 $f(3, \{s_3, 1, 2\})$

$\{s_3, 1\} \leftarrow \{s_3, 1, 2\}.remove(\text{arr } s_2)$

$f(3, \{s_3, 1\})$

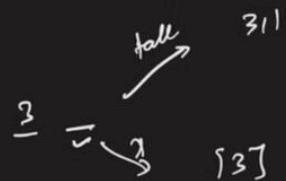
}

```

<--> f(2, [1, 2])
      if(ind >= n) ✗
      print(s1)
      return;
[2]. s1.add(ann s1);
f(ind+1, s1); → tail
s1.remove(ann s1);
f(ind+1, s1); → tail
}

main()
{
    ann → [3, 1, 2]
    f(0, [])
}

```



```

<-- f(0, [ ])
    |---- f(1, [ ])  ← i = 0
        |---- i >= n → X
        |---- print([ ]) → X
        |---- return;
        |---- g[ ].add([ ]); → tail
        |---- f(ind+1, g[ ]); → tail
        |---- g[ ].remove([ ]); → tail
        |---- f(ind+1, g[ ]); → tail
    |---- f(2, [ ])  ← i = 1
        |---- i >= n → X
        |---- print([ ]) → X
        |---- return;
    |---- f(3, [ ])  ← i = 2
        |---- i >= n → X
        |---- print([ ]) → X
        |---- return;
    |---- f(2, [ ])  ← i = 3
        |---- i >= n → X
        |---- print([ ]) → X
        |---- return;
}
main()
$ 
aaon → [3, 1, 2]
f(0, [ ])

```

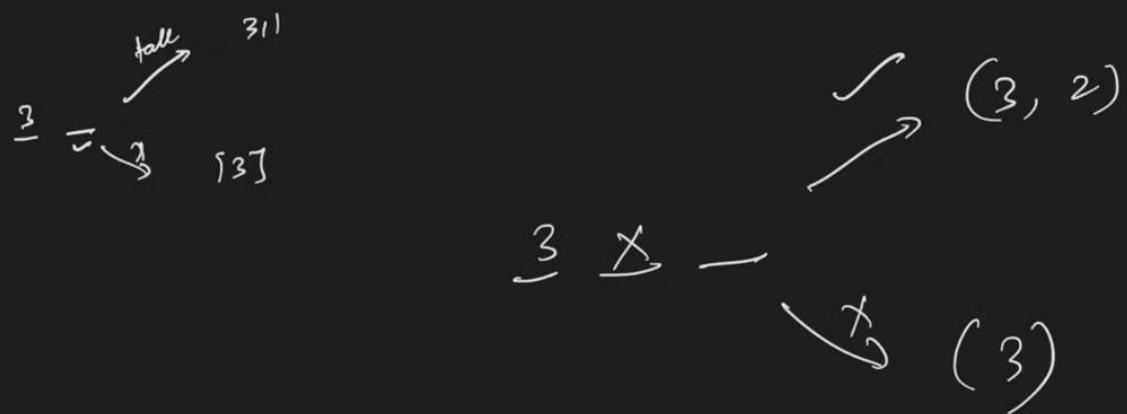
$\begin{matrix} 3 \\ \diagdown \\ 3 \end{matrix} \xrightarrow{\text{tall}} [3]^{[1]}$

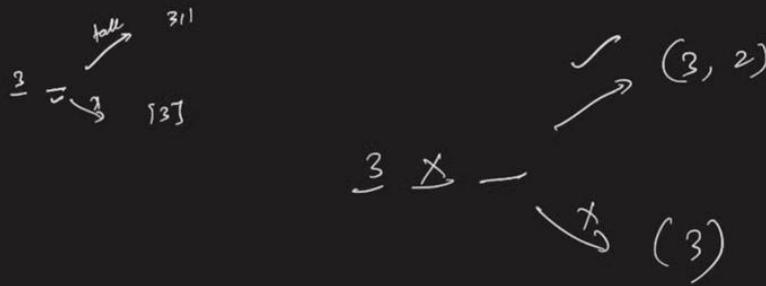
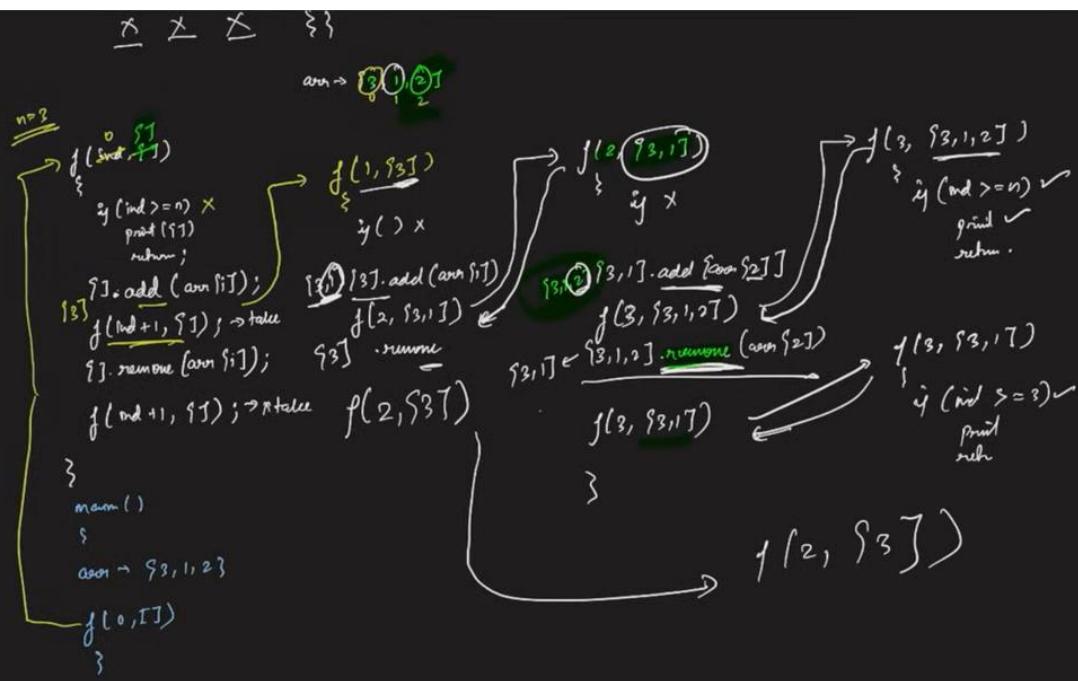
```

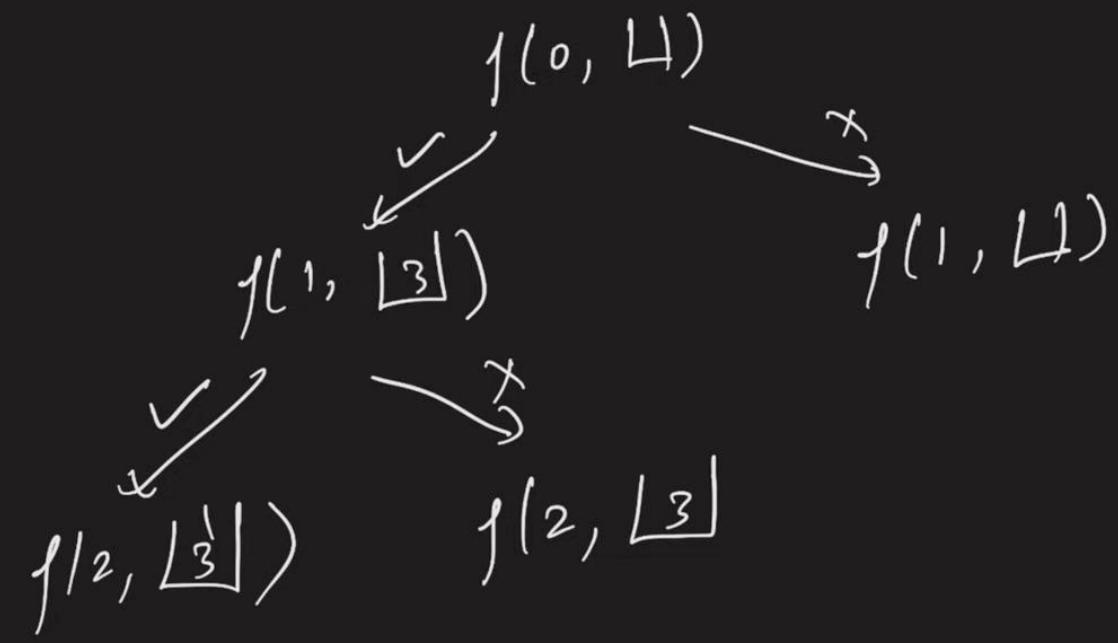
< } f(m+1, S); // take f(2, S[3])
} main()
$ arr → S[3, 1, 2]
f(0, I)
}

```

↑ (mid >= 3) ✓
print
return





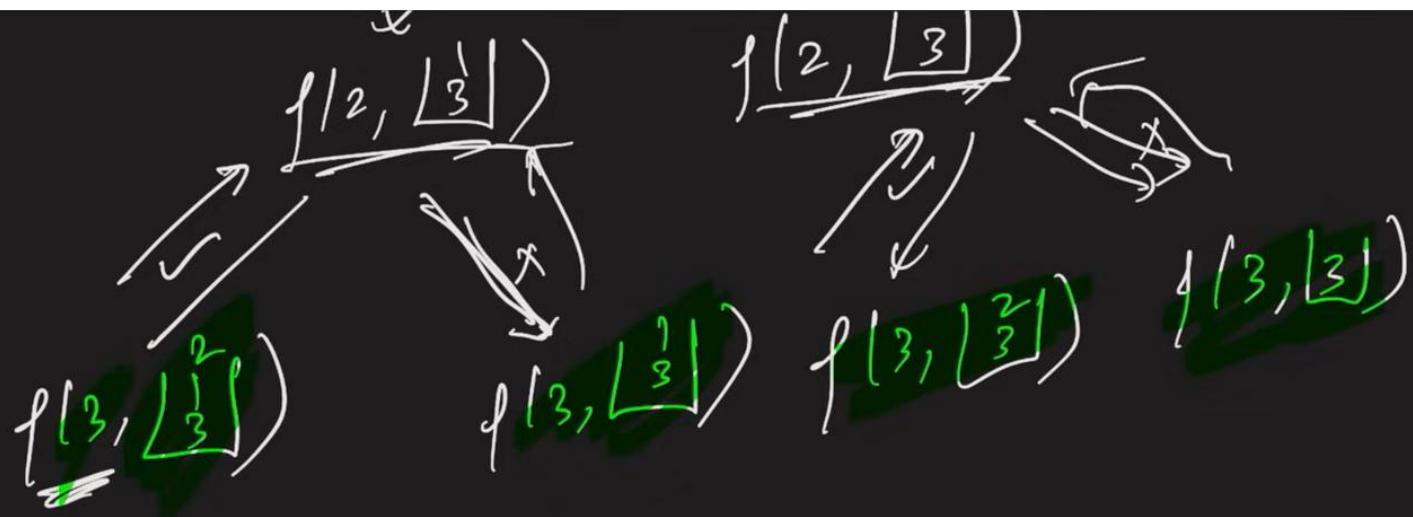


TUF

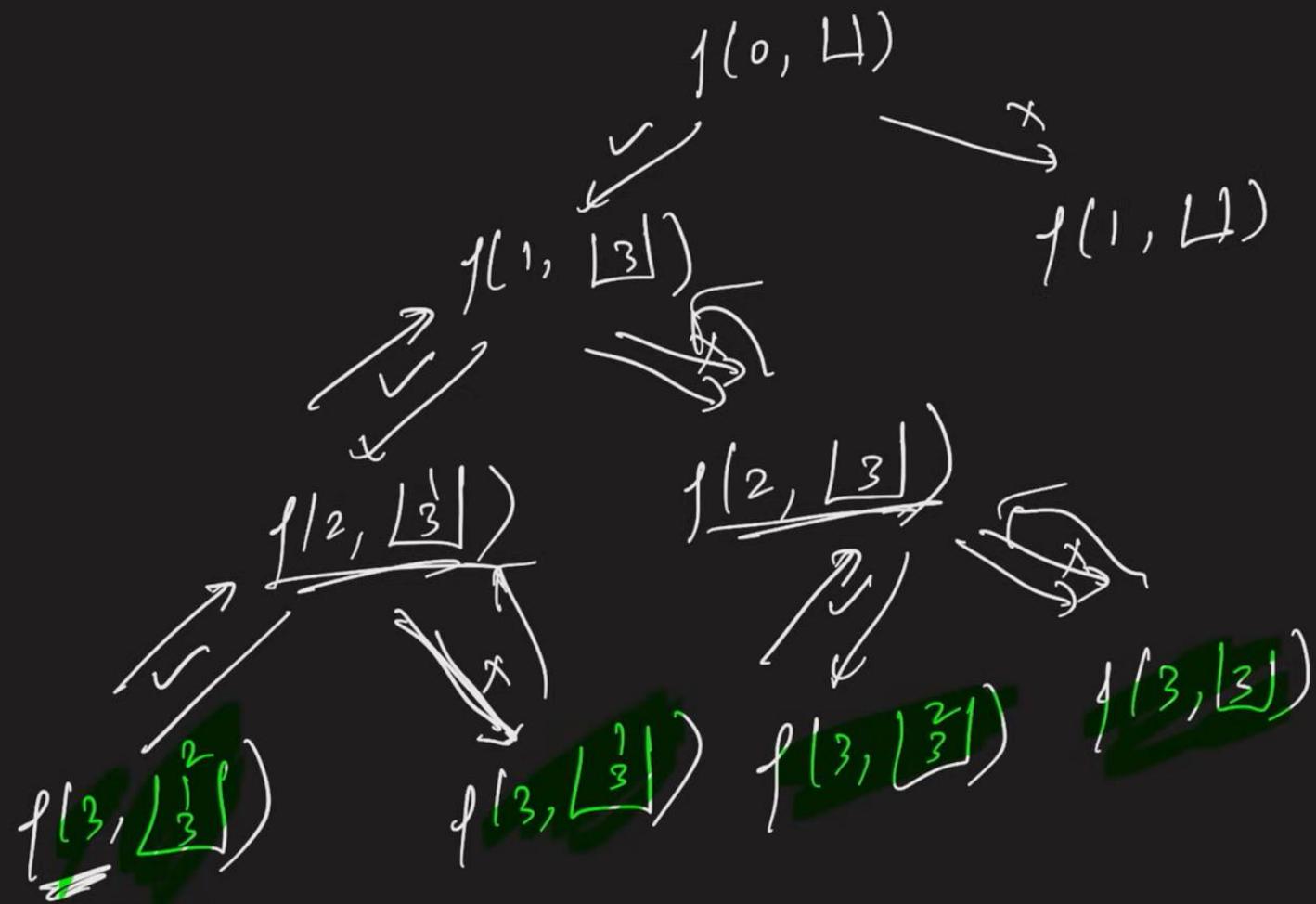
$\left| f(2, \lfloor \frac{1}{3} \rfloor) \right\rangle$

$\left| f(2, \lfloor \frac{2}{3} \rfloor) \right\rangle$

$\left| f(2, \lfloor \frac{1}{3} \rfloor) \right\rangle$



2	1	2]
3	1		
3	2		
3			

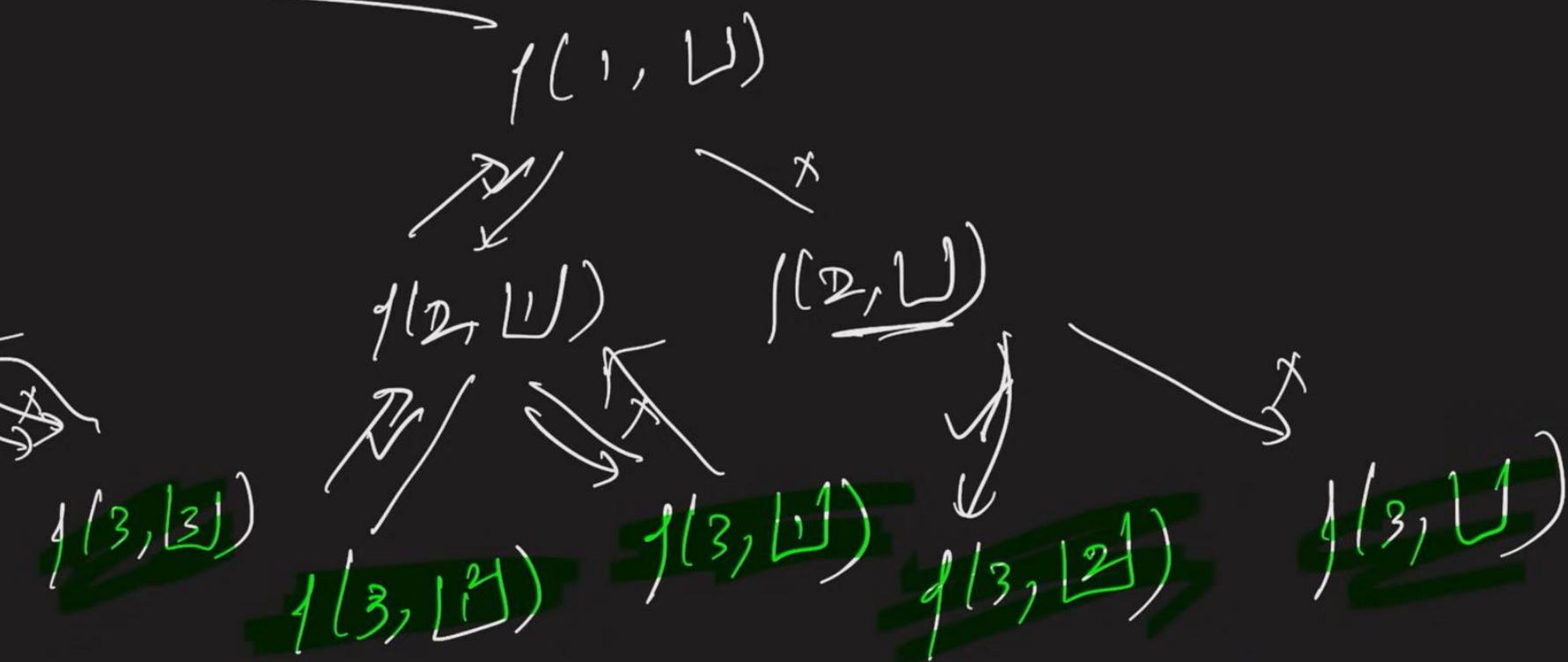


2 1 2 7
3 1

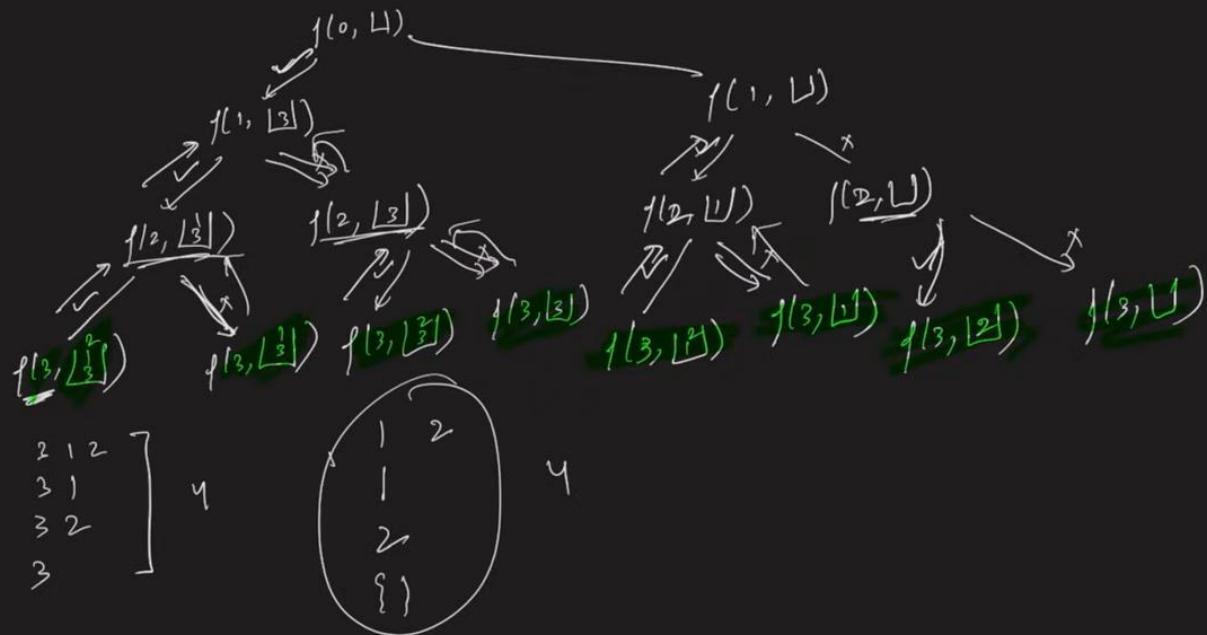
TUF

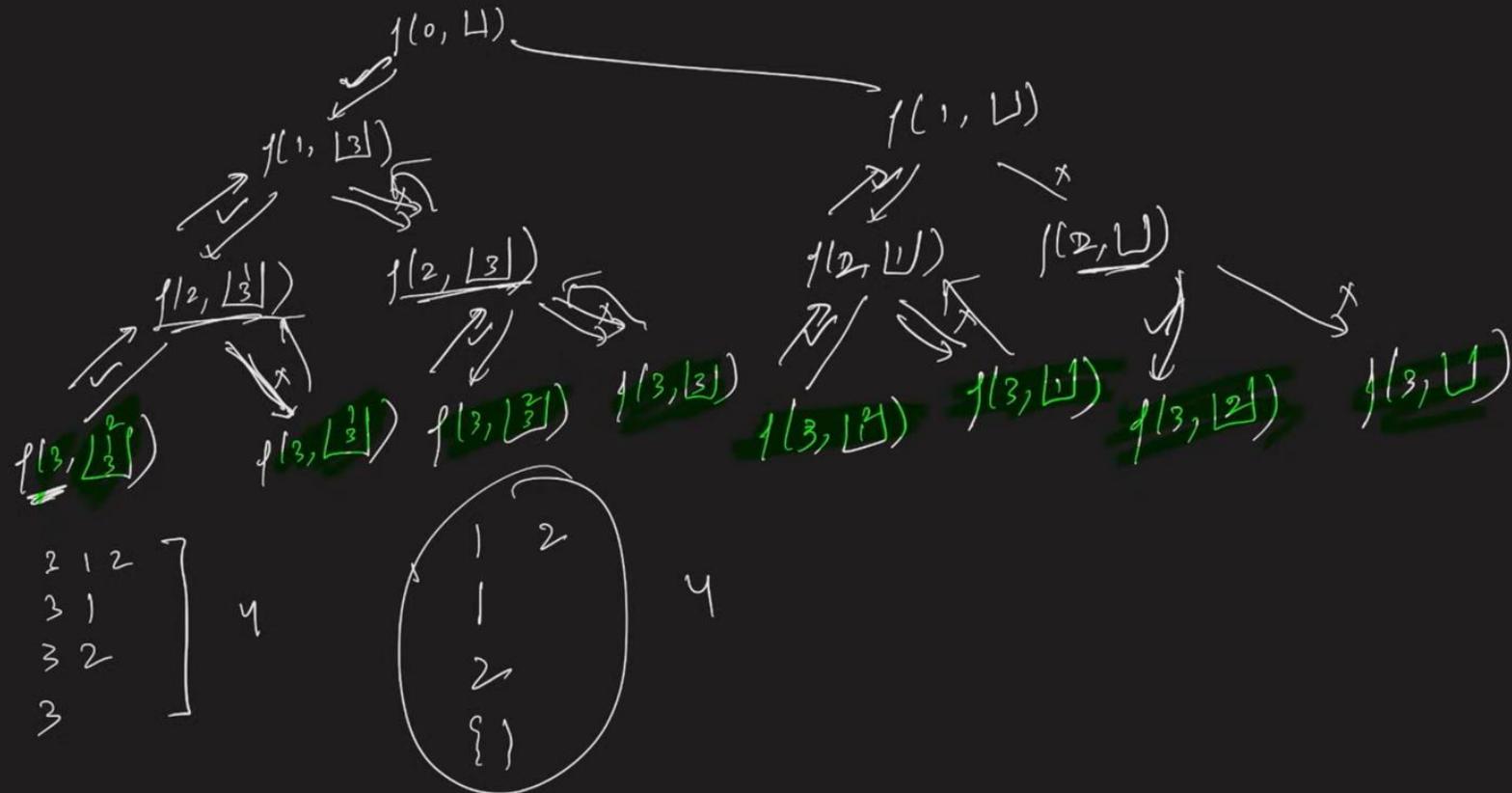


TUF



reun
f()






```
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         cout << endl;
9         return;
10    }
11    // take or pick the particular index into the subsequence
12    ds.push_back()
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 }
```

Finished in 1.6s]



TUF

The image shows a developer working on a computer, with a video feed of their face visible on the right side of the screen. The developer is wearing a blue t-shirt and has a beard. The main focus is on the left side of the screen, which displays a code editor and a terminal window.

Code Editor:

```
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         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 element 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 }
```

Terminal Output:

```
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
```

Text at the bottom:

Finished in 2.0s]

TUF

```
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    // 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 }
```

Finished in 1.1s]

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	

TUF



A screenshot of a code editor and terminal window. The code editor shows a C++ file named `code.cpp` with the following content:

```
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}
26int 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}
```

The terminal window shows the input file `input.txt` containing the numbers 1 through 5, and the output file `output.txt` showing all possible subsequences of the array [3, 1, 2].

input.txt content:

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

output.txt content:

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

Text at the bottom left: `Finished in 1.3s]`

Text at the bottom right: `TUF`

The image shows a developer's workspace during a video call. On the left, a code editor displays a C++ file named `code.cpp`. The code implements a recursive function `printF` to generate all subsequences of an array. It uses a vector `ds` to store the current subsequence and prints it when the size of `ds` reaches the input size `n`. The developer has opened an `input.txt` file containing the input array [1, 2, 3, 4, 5] and an `output.txt` file where the generated subsequences are being printed. The developer's terminal at the bottom shows the command `Finished in 1.3s]`. On the right, a video feed of the developer, a young man with a beard wearing a blue t-shirt, is visible. The TUF logo is in the bottom right corner.

```
code.cpp
stdc++.h
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
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 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 vector<int> ds;
33 printF(0, ds, arr, n);
34
35
36 return 0;
37 }
```

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]

TUF

```
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 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 // 2 2 2 2
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 }
```

Finished in 1.3s]

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	

TUF

```
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 element 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 // 2 2 2 2
24 int main() {
25     #ifndef ONLINE_JUDGE
26         freopen("input.txt", "r", stdin);
27         freopen("output.txt", "w", stdout);
28     #endif
29     int arr[] = {3, 1, 2};
30     int n = 3;
31 }
```

Finished in 1.3s]

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	

TUF

```
code.cpp stdc++.h input.txt output.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
15    // not pick, or not take condition, this element 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 // 2^n subsequences
24 int main() {
25     #ifndef ONLINE_JUDGE
26         freopen("input.txt", "r", stdin);
27         freopen("output.txt", "w", stdout);
28     #endif
29     int arr[] = {3, 1, 2};
30     int n = 3;
31 }
```

Finished in 1.3s]

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	

TUF