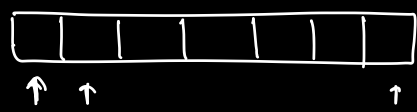
Subarray → Contiguous part of an array.



$N + (N-1) + (N-2) + \dots + 1$

$= \frac{N \times (N+1)}{2}$

3, 6, 5, 7, 1, 2, 10

5, 1, 2

3, 2, 10

5, 10

Subsequences

Subsequence

Any sq that can be generated by deleting 0 or more elements from an array.

3	6	5	7	1	2	10		
✗	✗	✗	✓	✗	✓	✓		→ [7, 2, 10]
✗	✓	✗	✗	✓	✗	✓	✗	→ [6, 1, 2]
✗	✗	✗	✗	✗	✓	✗	✗	→ [2]
✗	✗	✗	✗	✗	✗	✗	✗	→ [ ]
✓	✓	✓	✓	✓	✓	✓	✓	

Quiz

1, 2, 3, 4, 5

1, 2, 3, 4, 5

4

2, 3, 5

5, 4, 3 ✗

A : [-1, 4, 3, 9]

	Sub array	Sub seq
[-1, 4]	✓	✓
[4, 3, 9]	✓	✓
[4, 9]	✗	✓
[-1, 3, 9]	✗	✓
[3]	✓	✓
[9, 3]	✗	✗

All subarrays are sub-sequences.
But vice-versa is not true.

A : 4 , -1 , 2 $\xrightarrow{\text{Sort}}$ -1 , 2 , 4

{ }	{ }
[4]	[4]
[-1]	[-1]
[2]	[2]
[4, -1]	[-1, 4]
[4, 2]	[2, 4]
[-1, 2]	[-1, 2]
[4, -1, 2]	[-1, 2, 4]

Subsets : Same as subsequence but

- Order does not matter ⇐
- Only unique values.

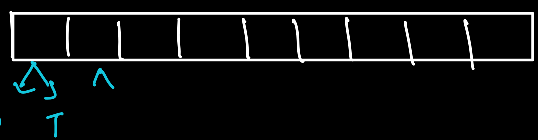
A : 4 , -1 , 2 $\xrightarrow{\text{Sort}}$ -1 , 2 , 4

{ }	{ }
{4}	{-1}
{-1}	{2}
{2}	{4}
{4, -1}	{-1, 2}
{4, 2}	{-1, 4}
{-1, 2}	{2, 4}
{4, -1, 2}	{-1, 2, 4}

In an array of distinct elements.

No of Subsets = No of sub-seq.

No of sub sq. in array of size N



$2 \times 2 \times - - - -$

2^N

$a_0 \quad a_1 \quad a_2$

$\{ \quad \}$

a_0

$a_0 \quad a_1$

$a_0 \quad a_2$

a_1

$a_1 \quad a_2$

a_2

$a_1 \quad a_1 \quad a_2$

No of subset $= 2^n$

Q Given an array of size N (distinct elements). Check if there exists a subset with sum K.

$$A: \overset{0}{3},\ \overset{1}{-1},\ \overset{2}{0},\ \overset{3}{6},\ \overset{4}{2},\ \overset{5}{-3},\ \overset{6}{5}$$

$K = 10$

{5, 3, 2}

{-1, 6, 5}

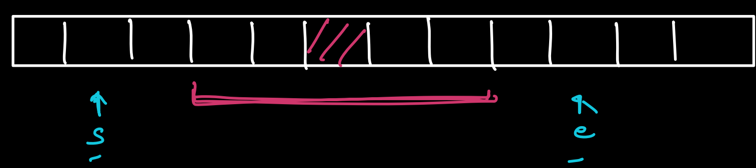
{3, -1, 6, 2}

⋮

True

$K = 20$

False



Iterate over all possible subsets

2^N

$$\overset{0}{-2},\ \overset{1}{6},\ \overset{2}{4}$$

		Index 2	1	0	
0	→	0	0	0	{ } ⇒
1	→	0	0	1	{-2} ⇒ →
2	→	0	1	0	{6} →
3	→	0	1	1	{-2, 6} ⇒ →
4	→	1	0	0	{4} →
5	→	1	0	1	{-2, 4} →
6	→	1	1	0	{6, 4} →
7	→	1	1	1	{-2, 6, 4}

Bit Mask

```
for ( i=0;  i< 2^N ;  i ++ ){          ⟹  2^N
     // for i  check which all bits are set
     sum = 0;
     for( j=0;  j<N ;  j++) {           ⟹  N
          if (CheckBit ( i, j)) {       ⟹  O(1)
               sum += A[j];
          }
     }
     if ( sum == K)
          ret true;
}
ret false;
```

TC : $O(N 2^N)$ $\xrightarrow{\text{Backtracking}}$ $O(2^N)$ $\xrightarrow[\text{Programming}]{\text{Dynamic}}$ $O(N*K)$

SC (Extra) : $O(1)$

Q Given an array of distinct elements.
Find the sum of all subset sums.

[-2, 6, 4] ⟶ 32

{ } ⇒ 0
$\boxed{\{-2\} \Rightarrow -2}$
{6} → 6
$\boxed{\{-2, 6\} \to 4}$
{4} → 4
$\boxed{\{-2, 4\} \to 2}$
{6, 4} → 10
$\boxed{\{-2, 6, 4\} \to 8}$

32

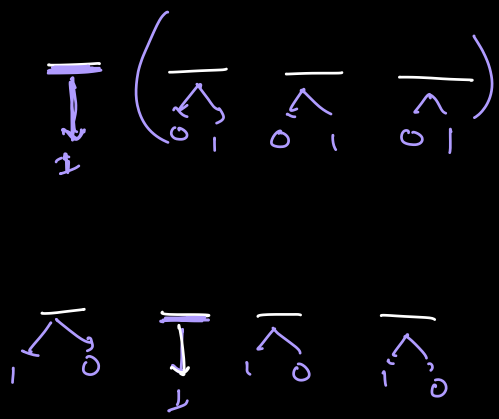
$-2 \times 4 + 6 \times 4 + 4 \times 4$
$= 32$

App 1 Iterate over all subsets

$O(N 2^N)$

App 2 Contribution tech.

0 1   0 1   0 1   0 1 ⇒ 2^4



$\Rightarrow 2^3$

$\Rightarrow 2^3$

$-2, \; 6, \; 4$

6,	-2	4
6, -2	-2, 6	4, -2
6, 4	-2, 4	4, 6
6, -2, 4	-2, 6, 4	4, -2, 6

1, -2, 6, 4

1	1, -2, 6	1, -2, 6, 4
1, -2	1, -2, 4	
1, 6	1, 6, 4	
1, 4		

No. of subsets in which any i^{th} element will be present $= 2^{(N-1)}$

```
sum = 0;
for (i=0; i<N; i++){
  sum = sum + (A[i] x 2^(N-1));
}
ret sum;
```

TC: O(N)

$1 << (N-1)$

No. of subsets in which A[i] is present.

$A[i] << (N-1)$

Return ans % P

$\underbrace{Pow(a, N, P)}_{(\log N)} \Rightarrow a^N \% P$

zeta

Q Given an array of N distinct elements.

Cal (Sum of all subset sum) / 2^N.

$a_0 \quad a_1 \quad a_2 \quad \dots \quad a_n$

Sum of all subset $= 2^{N-1} \times a_0 + 2^{N-1} a_1 + \dots 2^{N-1} a_N$

$= 2^{N-1} (a_0 + a_1 + \dots a_N)$

$$\frac{\text{Sum of all subset}}{2^N} = \frac{a_0 + a_1 + a_2 + \dots a_N}{2}$$

Q

Facebook

Given an array. Find the sum of max of every subsequence.

A : 3, 1, -4

	max
[ ] →	0
[3] →	3
[1] →	1
[-4] →	-4
[3, 1] →	3
[3, -4] →	3
[1, -4] →	1
[3, 1, -4] →	3
	10

App 1

Iterate over all sub seq.

$O(N \, 2^N)$

App : 2 Contribution tech.

→ Count cont of every element in the final sum

↓

No of subsq in which the element is max.

$$\begin{array}{ccccc} 0 & 1 & 2 & 3 & 4 \\ [\,1, & 2, & 3, & 4, & 5\,] \\ \uparrow & \uparrow & \uparrow & \uparrow & \uparrow \\ 2^0 & 2^1 & 2^2 & 2^3 & 2^4 \end{array} \Rightarrow \text{Sorted order}$$

A : 3, 1, -4 $\xrightarrow{\text{Sort}}$ 1, 3, -4

	Max
[ ]	0
[3]	3
[1]	1
[-4]	-4
[3, 1]	3
[3, -4]	3
[1, -4]	1
[3, 1, -4]	3

Max / Min / Sum / Prod $\longrightarrow$

[ ]	0
[1]	1
[3]	3
[-4]	-4
[1, 3]	3
[1, -4]	1
[3, -4]	3
[1, 3, -4]	3

→ Sort the array

→ Contribution of ith element = $A[i] \times 2^i$

```
MaxSum = 0;
for (i = 0; i < N; i++) {
    MaxSum += (A[i] x 2^i);
}
ret MaxSum;
```

TC : $O(N \log N) + O(N)$

↑ Sort ↑ Cal ans

$\approx O(N \log N)$