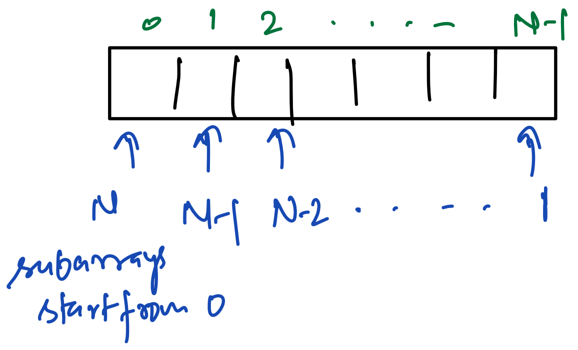
Subsequences & Subsets

Content

- → Revise Subarray
- → Subsequences vs Subsets
- → Check subset with given sum
- → Sum of all subsets
- → Sum of max of all subsets

Subarray : Continous part of an array

\# subarray: [s, e]



$$\text{total subarrays} = (1+2+3...+N)$$

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

Subsequence : Sequence generated by deleting 0 or more elements from your array.

	0	1	2	3	4	5	6	7	
a[8] :	3	-2	0	1	8	7	4	9	
	×	✓	✓	×	✓	×	✓	×	{-2, 0, 8, 4}
	✓	✓	✓	✓	×	×	×	×	{3 -2 0 1}
	✓	✓	✓	✓	✓	✓	✓	✓	{ All elements }
	×	×	×	×	×	×	×	×	{ } → empty subsequence
	×	×	✓	✓	✓	×	✓	✓	{0 1 8 4 9}

{0 1 8 4 9}

≠

{1 0 8 4 9}

order matters in a subsequence

order is based on element index.

Subarrays vs Subsequences

	0	1	2	3	4
a[5] :	-3	0	1	2	6

	Subarray	Subsequence
[1 2 6]	✓	✓
[-3 1 2]	✗	✓
[0 1 2]	✓	✓

[-3 1 6]	✗	✓
[6 1 0]	✗	✗

All subarrays are subsequences TRUE

All subsequences are subarrays FALSE

Sorting in subsequences

$$a[3] = \begin{array}{ccc} 0 & 1 & 2 \\ 3 & -2 & 1 \end{array} \xrightarrow{\text{sort}} \begin{array}{ccc} 0 & 1 & 2 \\ -2 & 1 & 3 \end{array}$$

All subsequences	All subsequences
{}	{}
{3}	{3}
{-2}	{-2}
{1}	{1}
{3, -2} max= 3 min=-2 sum= 1	{-2, 3} max=3 min=-2 sum=1
{-2, 1}	{-2, 1}
{3, 1}	{1, 3}
{3, -2, 1}	{-2, 1, 3}

Conclusion: If we sort, data subsequences will also change.

Subsets: Exactly same as subsequence. Just order doesn't matter.

$$a[3] = \overset{0}{3} \quad \overset{1}{-2} \quad \overset{2}{1} \quad \xrightarrow{\text{sort}} \quad \overset{0}{-2} \quad \overset{1}{1} \quad \overset{2}{3}$$

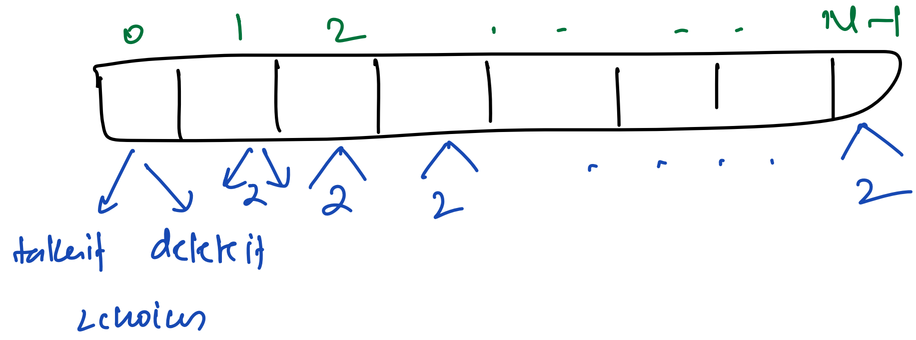
All subsequences	All subsequences
{}	{}
{3}	{3}
{-2}	{-2}
{1}	{1}
{3, -2}	{-2, 3}
{-2, 1}	{-2, 1}
{3, 1}	{1, 3}
{3, -2, 1}	{-2, 1, 3}

Note: Sorting doesn't change subsets in an array.

Count number of subsequences ?

↳ by deleting 0 or more

// Given N elements ?



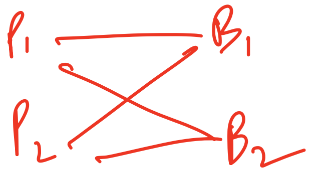
$= \underbrace{2\times2\times2 \cdots \times 2}_{n \text{ times}}$

$= 2^N$

How 2^n ?

2 pens , 2 books

P_1, P_2 B_1, B_2



$\Rightarrow 4$

a[2] = [1 3]

✗ ✗ ⇒ {}

✗ ✓ ⇒ {3}

✓ ✗ ⇒ {1}

✓ ✓ ⇒ {1, 3}

Given N = 2^N subsequences

Given N distinct elements = 2^N subsets

eg

$\begin{array}{ccc} 0 & 1 & 2 \\ 1 & 2 & 2 \end{array}$

Subsequences :

{} {1} $\overset{1}{\{2\}}$ $\overset{2}{\{2\}}$

{1,2} {1,2} {2,2}

{1,2,2}

$\begin{array}{ccc} 0 & 1 & 2 \\ 1 & 2 & 2 \end{array}$

Subsets :

{} {1} {2} {2}

{1,2} {1,2} {2,2}

{1,2,2}

same subset

If elements are repeating, subsets will change.

Question 1

Given N distinct elements → 2^N subsets

Check if there exists a subset with sum = K?

↳ Bool (T/F)

eg $a[7] = \begin{array}{ccccccc} 0 & 1 & 2 & 3 & 4 & 5 & 6 \\ 3 & -1 & 0 & 6 & 2 & -3 & 5 \end{array}$

$K=10$: $\left.\begin{array}{l} \{-1,6,5\} \\ \{3,2,5\} \\ \{6,2,-1,3\} \end{array}\right\}$ return True

$K=20$: return false

Idea: $a[3] = \begin{array}{cccc} & 0 & 1 & 2 \\ \big[& 3 & -2 & 1 \big] \end{array}$ → [8 subsets] = $[0, 2^3-1]$

i	2	1	0		subsets	sum
0 :	0	0	0	→	{ }	= 0
1 :	0	0	1	→	{3}	= 3
2 :	0	1	0	→	{-2}	= -2
3 :	0	1	1	→	{3,-2}	= 1
4 :	1	0	0	→	{1}	= 1
5 :	1	0	1	→	{3,1}	= 4
6 :	1	1	0	→	{-2,1}	= -1
7 :	1	1	1	→	{3,-2,1}	= 2

Code

```
for (i=0; i< $2^N$; ++i) {
    // i represents 1 subset
    sum = 0
    for (j=0; j<N; ++j) {
        if (checkBit(i,j)) {
            sum += a[j]
        }
    }
    if (sum == K)
        return true
}
return false
```

TODO

TC : $O(2^N \times N)$

SC : $O(1)$

2^N = 1<<N

Question 2

Given N distinct elements, find sum of [subset sum].

eg [3 1 4]

{} → 0
{3} → 3
{1} → 1
{4} → 4
{3 1} → 4
{3 4} → 7
{1 4} → 5
{3 1 4} → 8

ans = 32

Idea1: for every subset, iterate & get sum

TC: $O(2^N \times N)$ SC: $O(1)$

Idea 2: Contribution technique

$= 3\times(4) + 1\times(4) + 4\times(4)$

$= 12 + 4 + 16 = 32$

→ In how many subsets each element is present?

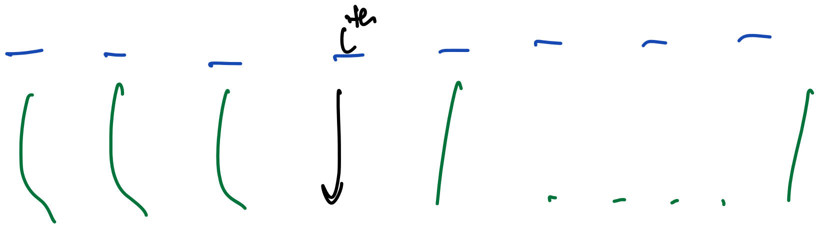
$$\begin{array}{lcccc} & 0 & 1 & 2 & 3 \\ a[i]: & 3 & 2 & 6 & 8 \end{array}$$

subsets: $\{1 * 2 * 2 * 2\} = 8$

eg

{3}
{3, 2}
{3, 6}
{3, 8}
{3, 2, 6}
{3, 2, 8}
{3, 6, 8}
{3, 2, 6, 8}

for N elements :



$2 \times 2 \times 2 \times 1 \times 2 \times \ldots 2 = 2^{N-1}$

ith index elements will be present in 2^{N-1} subsets.

Code

```
ans = 0
for (i=0; i<n; ++i) {
    ans += a[i] * (2^{N-1})
                  ↳ (1<<(n-1))
}
return ans
```

TC: O(N)

SC: O(1)

Given N distinct elements : $1 <= N <= 10^5$ ⇒ 2^{N-1} not possible

calculate (sum of all subset sums) / 2^N

$$\text{sum of subset sum} = (a_0 * 2^{N-1} + a_1 \times 2^{N-1} + \ldots + a_{n-1} * 2^{N-1}) / 2^N$$

$$= 2^{N-1} (a_0 + a_1 + \ldots + a_{n-1}) / 2^N$$

$$= (a_0 + a_1 + \ldots + a_{n-1}) / 2$$

= (sum of array elements)/2

Question3

Given an array, find the sum of max of every subsequence.

eg Arr : 3 1 -4

{} : 0
{3} : 3
{1} : 1
{-4} : -4
{3, 1} : 3
{3, -4} : 3
{1, -4} : 1
{3, 1, -4} : 3

Sum = 10

Idea1: for every subsequence get max & add it to the sum.

TC: $O(2^N * N)$ SC: $O(1)$

Idea2: Contribution technique

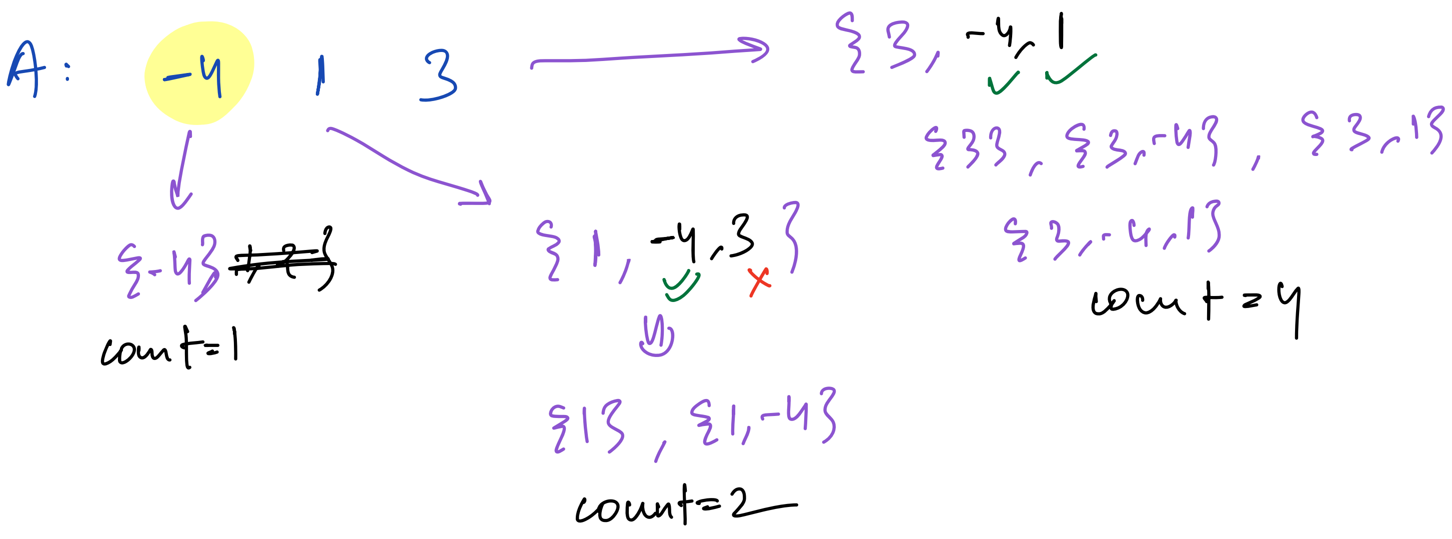
$= 3*(4) + 1*(2) + -4*(1)$

$= 12 + 2 - 4 = 10$

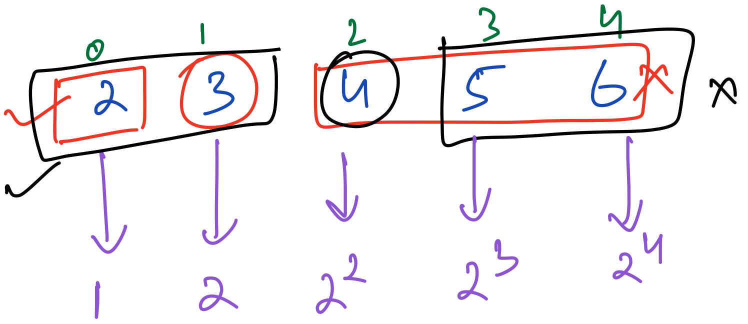
Can we sort the array?

→ But it changes the subsequences right?

→ However, it doesn't change max/min/sum of subsequences.



eg [3 2 6 4 5]



$ans = 2 \times 1 + 3 \times 2 + 4 \times 2^2 + 5 \times 2^3 + 6 \times 2^4$

$= 2 + 6 + 16 + 40 + 96 = 160$

Code

```
// sort in ascending order → NlogN

sum =0
for (i=0; i<n; ++i) {
    sum += a[i] * (1<<i)
}
return sum
```

TC: O(NlogN +N)

: O(NlogN)

SC: O(1)

TODO

sum of (max of every subsequence) ✓

sum of (min of every subsequence)

sum of (max-min of every subsequence)

$$(a_0 - b_0) + (a_1 - b_1) + (a_2 - b_2) \; . \; . \; . \; - + (a_{n-1} - b_{n-1})$$

$$\Rightarrow (a_0 + a_1 + \; . \; . \; . \; + a_{n-1}) - (b_0 + b_1 + \; . \; . \; . \; + b_{n-1})$$