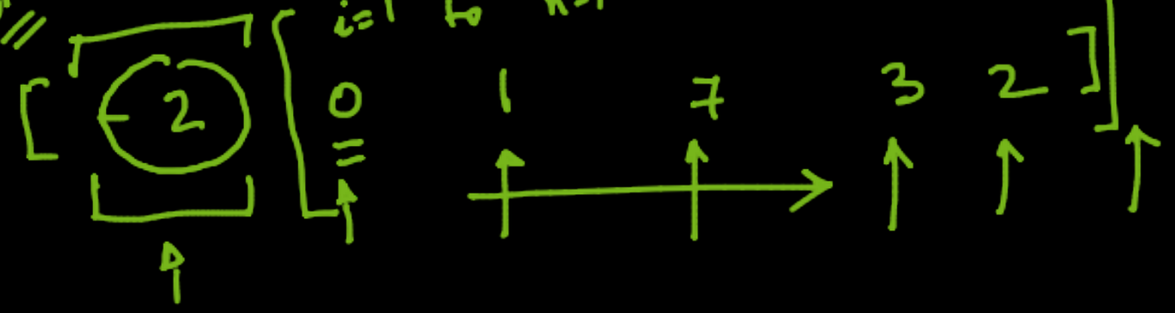
A = [ ] → copy

maxValue = A[0]

Index out of Bound



maxValue = -2 ~~0~~ ~~1~~ 7

↑

↳ INT_MIN

↳ INT_MAX

maxValue = ~~-INF~~ ~~-2~~ ~~0~~ ~~1~~ 7

n = 5

Bubble Sort

idx 0 1 2 3 4

arr[] [5 4 3 2 1]

1st pass

[5 4 3 2 1]

i = 0 =

4 5 3 2 1

n - i - 1 =

4 3 5 2 1

5 - 0 - 1 = 4

4 3 2 5 1

=

[4 3 2 1][5]

2nd pass

[4 3 2 1] [5]

i = 1 =

3 4 2 1

[n - i - 1]

3 2 4 1

5 - 1 - 1

= 3

[3 2 1][4 5]

3rd pass

[3 2 1] [4 5]

i = 2 =

2 3 1

n - i - 1

[2 1] [3 4 5]

5 - 2 - 1

= 2

(n-1)th pass

[2 1] [3 4 5]

i = 3 =

5 - 3 - 1

= 1

[1] [2 3 4 5]

[1 2 3 4 5]

stop

$(n-1)$ passes

$i=0$	$n-i-1$	$n-1$	break (Best)
$i=1$		$n-2$	
$i=2$		$n-3$	(worst)
⋮		⋮	
$i=n-2$		1	

$$= [(1+2+3+\dots n-3+n-2+n-1)+n]n$$

$$= (1+2+3+\dots n) - n$$

$$= \frac{n(n+1)}{2} - n$$

$$\text{worst} = \boxed{\frac{n(n-1)}{2}}$$

1 2 3 4 5

$(n-1)$ steps

$O \rightarrow$ worst $\quad \Theta \rightarrow$ avg.

$\Omega \rightarrow$ best

```
for (i=0; i< n-1; i++)
i=0  int count=0
   for(j=0; j< n-1-i; j++){
      if(a[j] > a[j+1]) {
         swap(a[j], a[j+1])
         count++;
      }
      if(count==0) break
   }
```

⇒ sorted $\Theta\left(\frac{n + \boxed{n^2}}{2}\right) \Rightarrow \Theta(n^2)$

Best

[1 2 3 4 5] ⇒

```
for (int i=0; i<n-1; i++) {
   for (int j=0; j<n-i-1; j++){
      ⇔ → n(n-1)/2
   }
   break
}
```

1 line $(n-1)$

$n=5, \quad n-1=4$

for(i=0; i<4; i++)

i=0, i=1, i=2, i=3 stop

1 2 3 4 5

$(n-1)$ steps (Best)

$\frac{n(n-1)}{2}$ steps (worst)

5 4 3 2)

i/p → n ⇒ 1 → 1^2 sec

2 → 2^2 sec

time complexity

$$\Rightarrow \left[\frac{n(n-1)}{2}\right] = \frac{n^2 - n}{2}$$

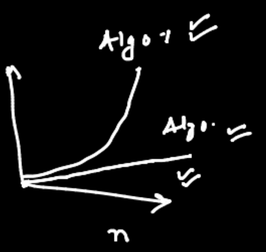
$\Rightarrow \boxed{O(n^2)} \rightarrow$ worst

$\boxed{\Omega(n)} \rightarrow$ best

worst $= O(n^2)$ ← quadratic

[Big O]

$O(n) \rightarrow$ Linear



Best case

$(n-1)$ steps

$\Omega(n)$

Best $=$

$x = 5$ $y = 10$

1. ⇒ `int temp = x; x = y; y = temp;` swap

2. ⇒ builtin swap(x, y)

3. ⇒ $x = (x+y) - (y=x)$

$= 15 - 5$

$x = 10$ $y = 5$

$x = 5;$

4) $x = x \wedge y$ → ①

$y = x \wedge y = (x \wedge y) \wedge y$ → ②

$x = x \wedge y = (x \wedge y) \wedge x$

$x = y$

swap

$x = 1$

swap(x, x)

$z = x + y = 5 + 10 = 15$

$y = x - y = 15 - 5 = 10$

$x = x - y = 10 - 5 = 5$

temp = x

x = x

x = temp

temp = 1

x = 1

$x = 1$ ⇒

$x = x \wedge x = 0$

$x = 0 \wedge x = x$

$x = x \wedge x = 0$

Integers

+ve $x = 5$ $y = 10$

Slower

Out-dated $= x = x \wedge y$ —①

$= y = x \wedge y = (x \wedge y) \wedge y$ $x \wedge 0$

↳ ② $= x$

$= x = x \wedge y = (x \wedge y) \wedge x$ $y \wedge 0$

$= y$

$x \wedge x = 0$ ←

$x \wedge 0 = x$ ←

Swap (built in)

⇒ Selection Sort

[n = 5]

← unsorted →

arr = [5 4 3 2]

idx 0 1 2 3 4

1st pass

i = 0

⇒ min_idx = 2

swap (a[i], a[min_idx])

swap (a[0], a[2])

after 1st pass [1] [4 5 3 2]

[1 2] [5 3 4]

← sorted → ← unsorted →

3rd pass

i = 2

swap (a[2], a[3])

[1 2 3] [5 4]

4th pass

i = 3

swap (a[3], a[4])

[1 2 3 4] [5] ↳ sorted

idx 0 1 2 3 4

arr [5 4 1 3 2]

unsorted

i = 0

min idx = ~~0~~ ~~1~~ 2

swap (a[0], a[2])

after 1st pass [1] [4 5 3 2]

1st pass

5 ≤ 5 ✓

4 < 5 ✓

2 < 4 ✓

3 < 1 ✗

2 < 1 ✗

2nd pass

⇒ i = 1 [1] [4 5 3 2]

swap (a[1], a[4])

after 2nd → [1 2] [5 3 4]

idx 0 1 2 3 4

arr [5 4 1 3 2]

sorted

1st pass

i = 0

min_idx = i = ~~0~~ ~~1~~ 2

5 < 5 ✗

4 < 5 ✓

1 < 4 ✓

3 < 1 ✗

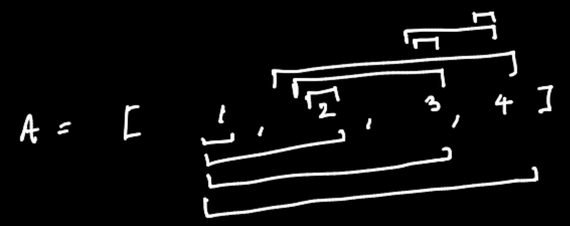
2 < 1 ✗

swap (a[i], a[min_idx])

swap (a[0], a[2])

after 1st pass

[1 4 5 3 2]



⇒ Sub-array (contiguous)

* Sub-seq (subset)

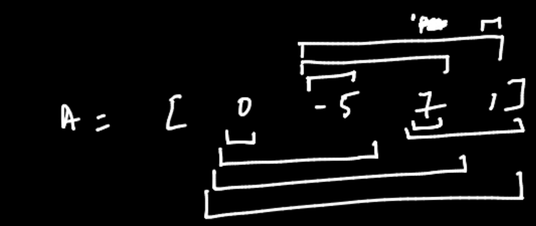
A = [1 2 3 4]

{ 1 3 } → subseq

↳ order is main

{ 3, 1 } X

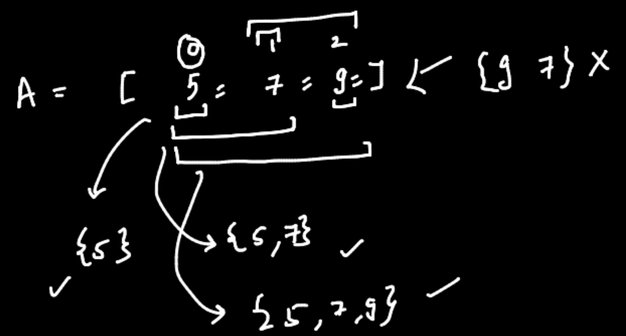
→ sub array is a subseq



✓ order maintained ✓

subseq (✓X) contiguous ✓ sub array

=



{9 7} X

x

{}

=

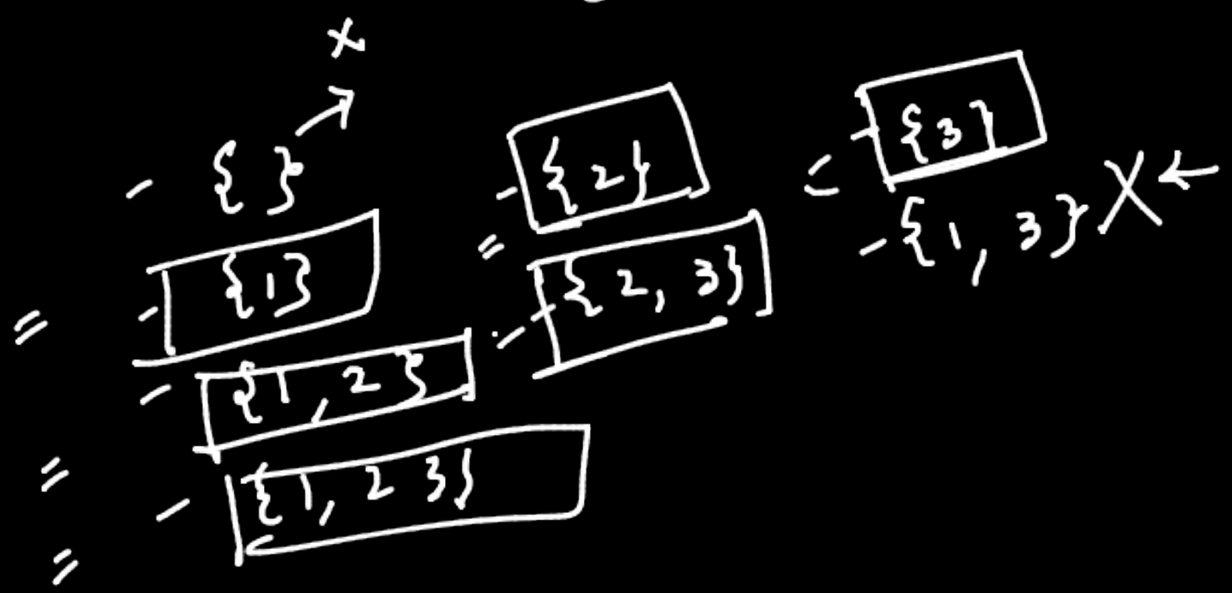
order maintained

✓ x contiguous/consecutive

≠ subseq / subset

$n = 3 \Rightarrow 2^3 = 8$

{ 1 2 3 }



$n=6 \Rightarrow \frac{6(6-1)}{2}=15$

maxSum = [INT_MIN] 3 / 5 ... ⇒ 8

max sub-array sum

A[ ] = { -3, 4, -2, 1, 5, -3 } (indices 0, 1, 2, 3, 4, 5)

max sub array is 8

i=0: {-3} = -3; {-3, 4} = 1; {-3, 4, -2} = -1; {-3, 4, -2, 1} = 0; {-3, 4, -2, 1, 5} = 5; {-3, 4, -2, 1, 5, -3} = 2

i=1: {4} = 4; {4, -2} = 2; {4, -2, 1} = 3; {4, -2, 1, 5} = 8; {4, -2, 1, 5, -3} = 5

i=2: {-2} = -2; {-2, 1} = -1; {-2, 1, 5} = 4; {-2, 1, 5, -3} = 1

i=3: {1} = 1; {1, 5} = 6; {1, 5, -3} = 3

i=4: {5} = 5; {5, -3} = 2

i=5: {-3} = -3

$\frac{n(n+1)}{2} = \frac{6(6+1)}{2} = 21$

$n=6$

$\frac{n(n+1)}{2} = \frac{6(6+1)}{2} = 21$

$\frac{n(n-1)}{2} + n$

$^nC_2 + {}^nC_1 = \frac{n(n+1)}{2}$

$^nC_2 \rightarrow$ missing out single index sub arr $i=j$

$^nC_2 + n$ ($^nC_1$)

Array → N how many sub-array will it contain?

$^nC_2 + {}^nC_1 = \frac{n(n-1)}{2} + n = \frac{n(n+1)}{2}$

Every sub-array can be uniq. identified by its start idx & end idx

$n \rightarrow n$ indices [0 to n-1]

(i, j) $\binom{n}{2} = {}^nC_2 = \frac{n!}{2!\cdot(n-2)!} = \frac{n \times (n-1) \times (n-2)!}{2!\,(n-2)!}$

$= \frac{n(n-1)}{2}$ ← total # of sub array

$(i, i) \Rightarrow$