

Maximum positivity.

[9 3 11 -2 1 2]

Find a subarray with max^m length such that it doesn't contain any -ve elements.

Ans: [9 3 11]

Idea: Generate all subarrays, and keep track of max length subarray with no -ve values.

[9 3 11 -2 1 2]
i j

starts with 9.	starts with 3	starts with 11	starts with -2	starts with 1	starts with 2
[9]	[3]	[11]	[]	[1]	[2]
[9 3]	[3 11]			[1 2]	
[9 3 11]					

Idea 2: My ans is from [l..r], Both l and r must lie on the side of any negative number in the array.

[9 3 11 -2 1 2]
j
i

[9]
[9 3]
[9 3 11]
[1]
[1 2]

```
def maxPositivity(arr, N)
```

```
{
```

```
    i=0, j=0, start=0, end=0.
```

```
    while (i < N & j < N)
```

```
    {
```

```
        if (arr[j] > 0)
```

```
            j=j+1
```

```
        else
```

```
            if (j-i) > (end-start+1)
```

```
                start = i
```

```
                end = j
```

```
            i=j+1
```

```
            j=j+1
```

```
    }
```

```
    return arr[start:end]
```

```
}
```

→ This will store final subarray.

TC: $O(N)$

SC: $O(1)$

→ To make sure we written the first subarray, in case of overlap.

Q2: Sort the permutation. [All the elements in the array will be from $1-N$].

The nos that can be sorted among themselves independent of others should be colored the same.

Ex: 1

A = [4, 3, 5, 1, 2]

1	2	3	4	5
4	3	5	1	2
8	3			5
2				

5 groups.

(clubbing 4 & 1) → 4.

(clubbing 3 & 5) → 3.

(clubbing 5 & 2) → 2.

Ex: 2.

A = [1, 2, 4, 3, 5, 6]

Observation: Ideal position for $A[i]$ is $(i+1)^{\text{th}}$ index, If its not in that position then this leads to a new group.

Idea: Assume that there are 'N' independent groups, and you can decrementing the value of 'N' as you club the nos (swap the nos).

Code:

```
def sortThePermutation(arr, n)
    ans = n
    for i in range(0, n)
        if (A[i] != i+1)
            while (A[i] != i+1)
                ans = ans - 1
                x = arr[i]
                A[i] = A[x-1]
                A[x-1] = x
    return ans
```

Tc: $O(N)$.
Sc: $O(1)$.

Reason for Tc.

- (i) looking at ans.
- (ii) At the max, you've n swaps.

Simply swap $A[i]$ with $A[A[i]-1]$.

Q3: Bus Dilemma.

$B \rightarrow$ max no. of people that bus can accommodate.



5.



2



1



-3

Possible values for no. of people in the bus initially, in-order for the above data to be valid.

Ex:

(i)	Can we've	0	people in the bus	\rightarrow	2, 3, 0	✓
	" "	1	" " " "	\rightarrow	3, 4, 1	✓
	" "	2	" " " "	\rightarrow	4, 5, 2	✓
	" "	3	" " " "	\rightarrow	5, 6	✗.

(ii) No. of people can't be negative initially, start with 0.

The possible values lies from $[0 \dots B]$.

Idea: Try checking for every value from $[0 \dots B]$ as shown above,
The running sum you get shouldn't exceed B , and shouldn't be -ve.

Idea 2: Try using prefix sum.

Arr: [2 1 -3]

Pf: [2 3 0]. , $B=5$.

Pfmax & Pfmin

$\begin{bmatrix} 0 \\ 1 \\ 2 \\ 3 \\ 4 \\ 5 \end{bmatrix}$

(i) Running sum shouldn't be more than B .

The chosen value must not exceed $[0 \rightarrow B - \text{Pfmax}]$.

$[a \ b] = b - a + 1$.

$$[B - P_{fmax} + 1] \rightarrow \text{ans.}$$

(ii) Running sum shouldn't be -ve.

$$[-2, 1] \quad , \quad B = 4.$$

$$\text{If } [-2, 1].$$

$$\text{If } P_{fmin} < 0,$$

$$\text{ans} = \text{ans} - P_{fmin}.$$

0
1
2
3
4.

The solⁿ lies from $[0 \text{ to } B - P_{fmax}]$.

$$\text{Total elements} = B - P_{fmax} + 1.$$

Corners.

(i) If $(-P_{fmin}) < B$, can there be a solⁿ? No. return 0.

$$[-4, -5] \quad , \quad B = 3.$$

(ii) If $P_{fmin} < 0$, remove P_{fmin} elements.

$$\text{ans} = \text{ans} + P_{fmin}.$$

(iii) If $P_{fmax} < 0$, remove P_{fmax} elements, $\Rightarrow \text{ans} = \text{ans} + P_{fmax}$.

$$\begin{array}{l} [-3, -7] \quad , \quad B = 20. \\ \text{If } [-3, -10] \end{array} \quad / \quad \begin{array}{l} \text{ans} = [10 \dots 20]. \\ = 11. \end{array}$$

```
def busDilemma (arr, n)
```

```
{
```

```
    // create pf and populate it.
```

```
    // find Pfmax & Pfmin
```

```
    if (-Pfmin < B) → should've been >.
```

```
        return 0
```

```
    ans = B - Pfmax + 1
```

```
    if (Pfmin < 0)
```

```
        ans = ans + Pfmin
```

```
    if (Pfmax < 0)
```

```
        ans = ans + Pfmax
```

```
    return ans → should've been (ans > 0? ans: 0) since ans can be negative.
```

```
}
```

Tc: $O(N)$.

Sc: $O(N)$.