Striver SDE Sheet - Challenge Note

① Problem (Set Matrix Zeros)

(If cell has a matrix zero, set that cells entire row and col zero)

Brute:

(i,j)

→ loop over every cell, for every cell having '0' mark all the row 'i' & col (j) with -1

→ Then mark every '-1' cells with zero in second loop

T.C = O(M*N) S.C = O(1)

Better:

	0	-1	0	-1
0	1	1	1	1
-1	1	0	1	0
0	1	1	1	1

row array

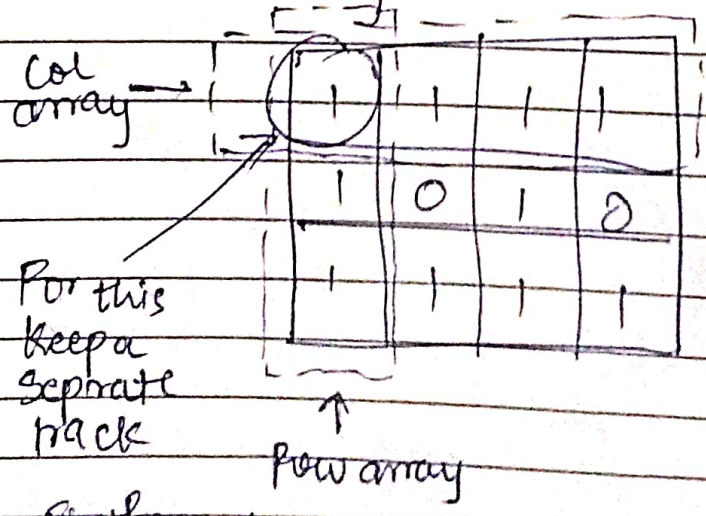
col array

1	0	1	0
0	0	0	0
1	0	1	0

T.C = O(M+N) S.C = O(M+N)

Best approach:

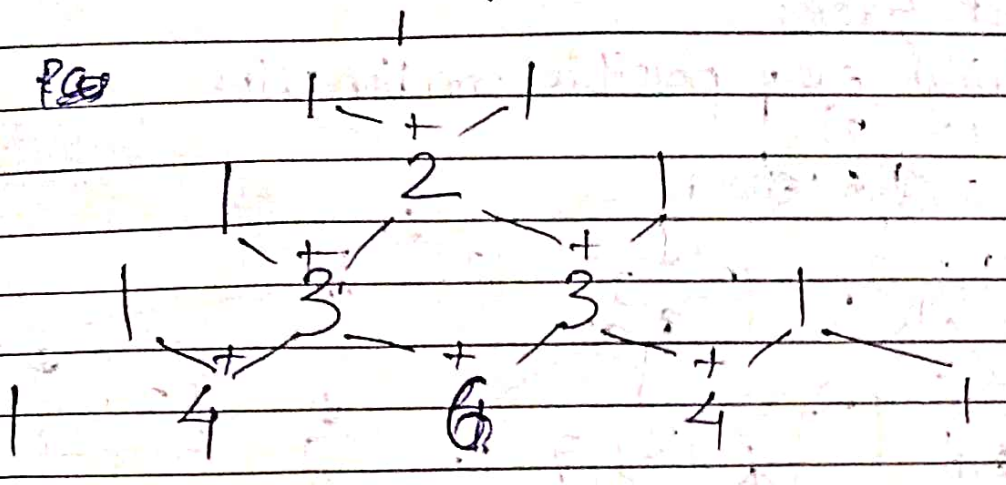
take row array and col array inside



firstRow = -1

firstCol = -1

(2) Pascal's Triangle:



1st approach:

Use Formula:

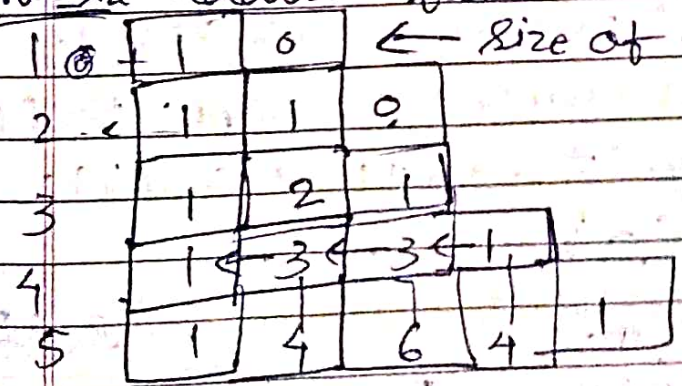
→ $nCr = n! / (r! * (n-r)!)$

→ Shortcut: $10C_3 = \frac{10 \times 9 \times 8}{1 \times 2 \times 3}$

T.C = O(N*R)

2nd approach

row size addition from above row



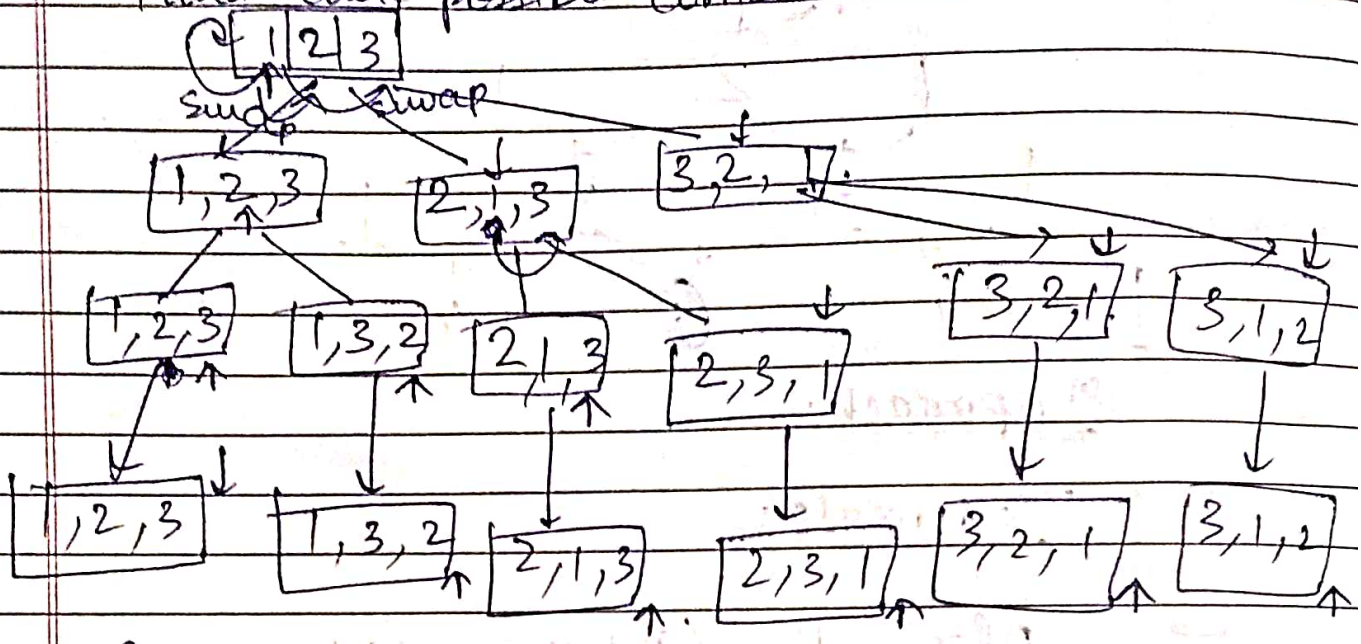
← Size of col = row no.

no. of rows = 'n'

(3) Next lexographical combination:

Brute force:

Find every possible combination



→ Store all the combination

→ compare every combination with given & output the next combination

Best approach:

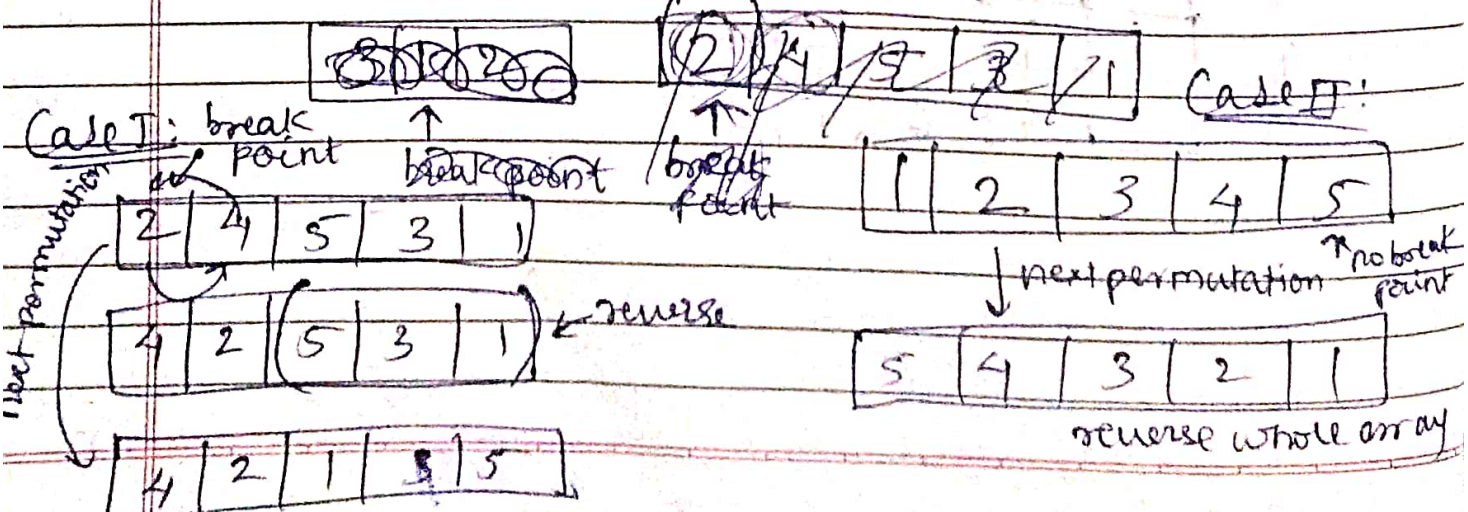
Use dictionary order approach:

→ Find break point A[i] < A[i+1] ← index

→ If no break point reverse whole array

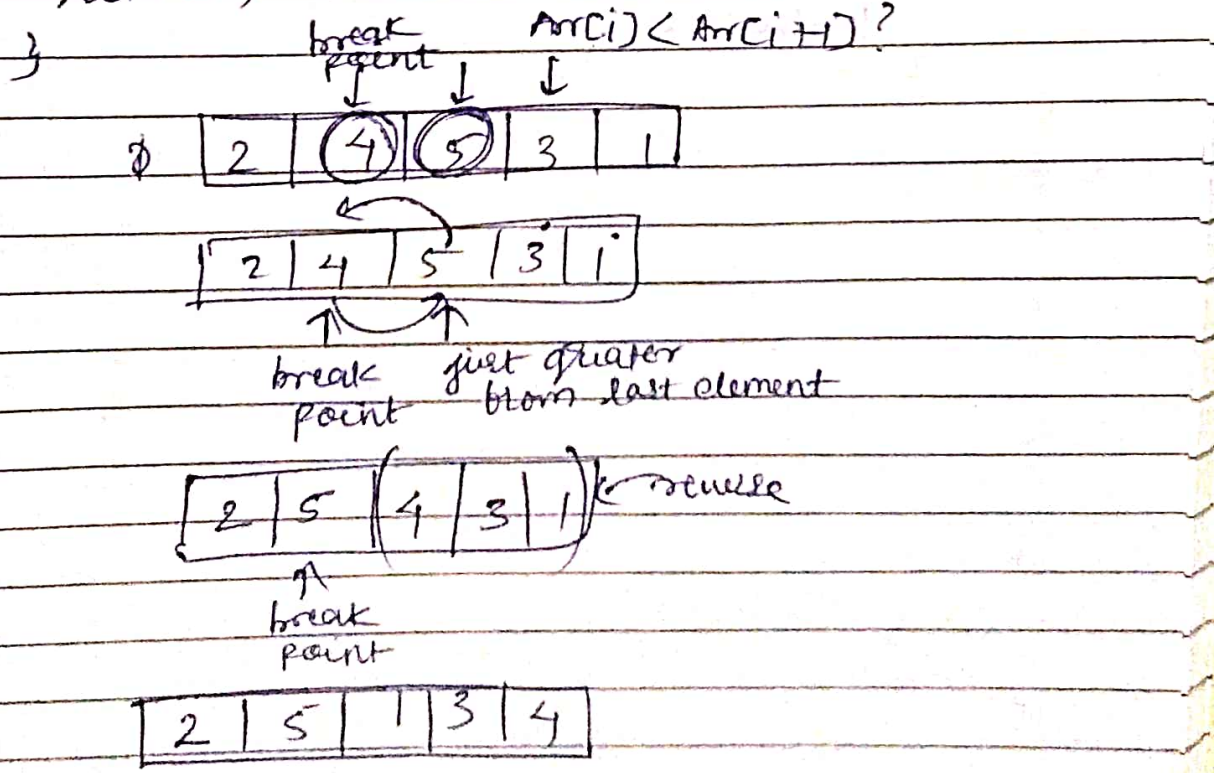
→ Find next greater element & swap with A[index]

→ reverse (A.begin() + index+1, A.end());



Best approach:

```
→ (i= nums.size() -2; i>=0; i--)
    if (nums[i] < nums[i+1])
        break;
    if (i== -1)
    { reverse( nums.begin(), nums.end());
      return;
    }
    for(j= nums.size() -1; j>i; j--)
    { if (nums[j] > nums[i])
        break;
    }
    swap (nums[i], nums[j]);
    reverse ( nums.begin() +i+1, nums.end());
    return;
}
```



④ Maximum Subarray Sum: (Kadane's algo)

Brute force approach

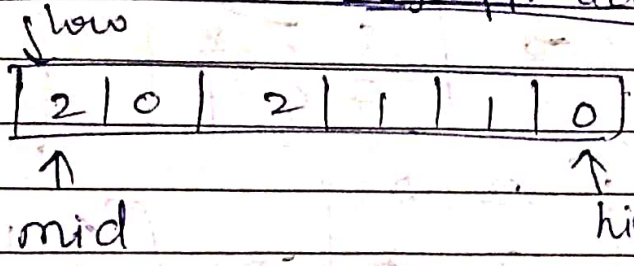
```
for(i=0 → n)
{ for(j=i → n)
  { int sum=0; }
  for(k=i → j)
  {sum += arr[k];}
}
```

Better approach:
(Kadane's algo)

```
for(i=0 → n)
{
 (localSum > maxi)
  { maxi = localSum
  }
  if(localSum < 0)
  localSum = 0;
}
```

(5) Sort (0, 1, 2)

Dutch National Flag approach:




```
if(arr[mid] == 0)
    swap(arr[low], arr[mid])
if(arr[mid] == 1)
    swap(arr[mid], arr[high]) mid++;
if(arr[mid] == 2)
    swap(arr[mid], arr[high])
```

(6) Stock Buy & Sell:

Brute approach:

```
for(i=0 → n)
{ for(j=i+1 → n)
  { if(arr[j] > arr[i])
      maxPro = max(arr[j]-arr[i], maxPro);
  }
}
```

Best approach:

```
for(i=0; i<n)
{ minPrice = min(minPrice, prices[i]);
  maxProfit = max(maxProfit, prices[i]-
                  minPrice);
}
```