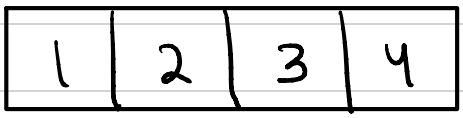
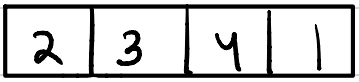
ROTATE ARRAY BY K PLACES

✡ In this question, if the input is



and K = 1, our output should be



Pseudocode (K = 1) :

```
shiftByOne(arr, N) {
    t = arr[0],
    for(int i = 1 → N) {
        arr[i-1] = arr[i]
    }
    arr[N-1] = t
}
```

What if K > 1
If K is equal to the length, array will stay normal. Hence K = K % N

For Brute force approach, we will perform the single rotation step K times. Time complexity becomes O(NK). Better solution is to take a temporary array, store the first K elements. Then shift remaining

N - K elements by K places to the left. Finally, add back the temporarily stored elements at the end.

Pseudocode :

```
shiftLeftByK(arr, N, K) {
    K = K % N
    temp[K]
    for(int i = 0 → K) {
        temp[i] = arr[i]
    }
    for(int i = K → N) {
        arr[i - K] = arr[i]
    }
    for(int i = N - K → N) {
        arr[i] = temp[i - N + K]
    }
}
```

Optimal solution is to first reverse array from 0 to K, then reverse from K to N and then finally reverse the full array.

Pseudocode :

```
shiftLeftByK(arr, N, K) {
    K = K % N
    reverse(arr, 0, K)
    reverse(arr, K+1, N)
    reverse(arr, 0, N)
}
```