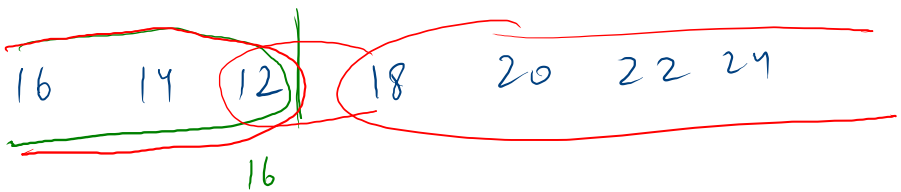
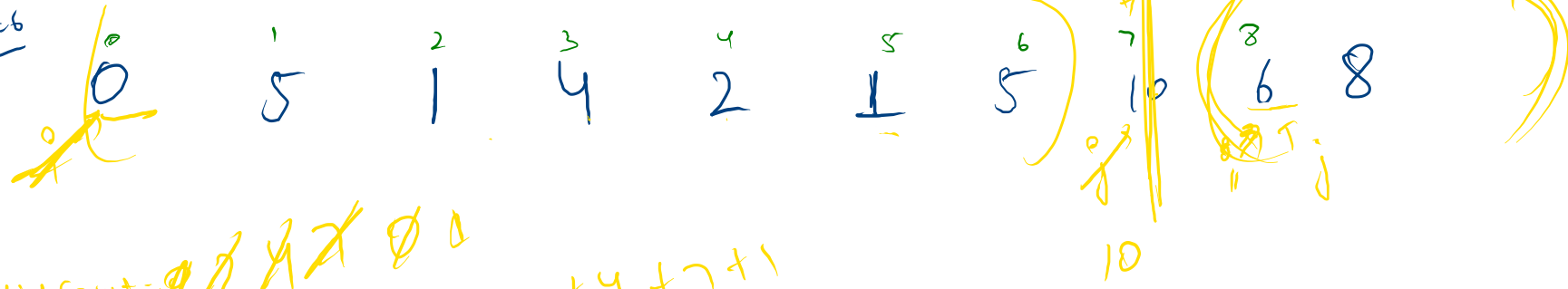
	0	1	2	3	4	5	6	7	8
arr:	22	18	20	26	24	30	32	34	28
prefixmax:	22	22	22	26	26	30	32	34	34
suffixmin:	18	18	20	24	24	28	28	28	28

```java
class Solution {
    public int maxChunksToSorted(int[] arr) {
        int n = arr.length;
        int[]prefixmax = new int[n];
        int[] suffixmin = new int[n];

        prefixmax[0] = arr[0];
        suffixmin[n-1] = arr[n-1];

        for(int i = 1;i<n;i++){
            prefixmax[i] = Math.max(prefixmax[i-1],arr[i]);
        }

        for(int i =n-2;i>=0;i--){
            suffixmin[i] = Math.min(suffixmin[i+1],arr[i]);
        }

        int ans = 0;

        for(int i =0;i<n-1;i++){

            if(prefixmax[i] <= suffixmin[i+1]){
                ans++;
            }
        }

        ans++;
        return ans;
```



12 14 16 18 20 22 24

L=3 , R=6



prevvalidCount = 0 2 4 4 4 4 7 7

ans = 0+0+2+2+4+4+4+7+7

```
int i =0;
int prevvalidcount=0;
int ans=0;


for(int j =0;j<arr.length;j++){

    if(arr[j]>=L && arr[j]<=R){
        ans += (j-i+1);
        prevvalidcount = j-i+1;

    }else if(arr[j] < L){
        ans+=prevvalidcount;

    }else{

        i = j+1;
        prevvalidcount=0;
    }

}


return ans;
}
```

wiggle sort

{ 3 5 1 6 2 4 }

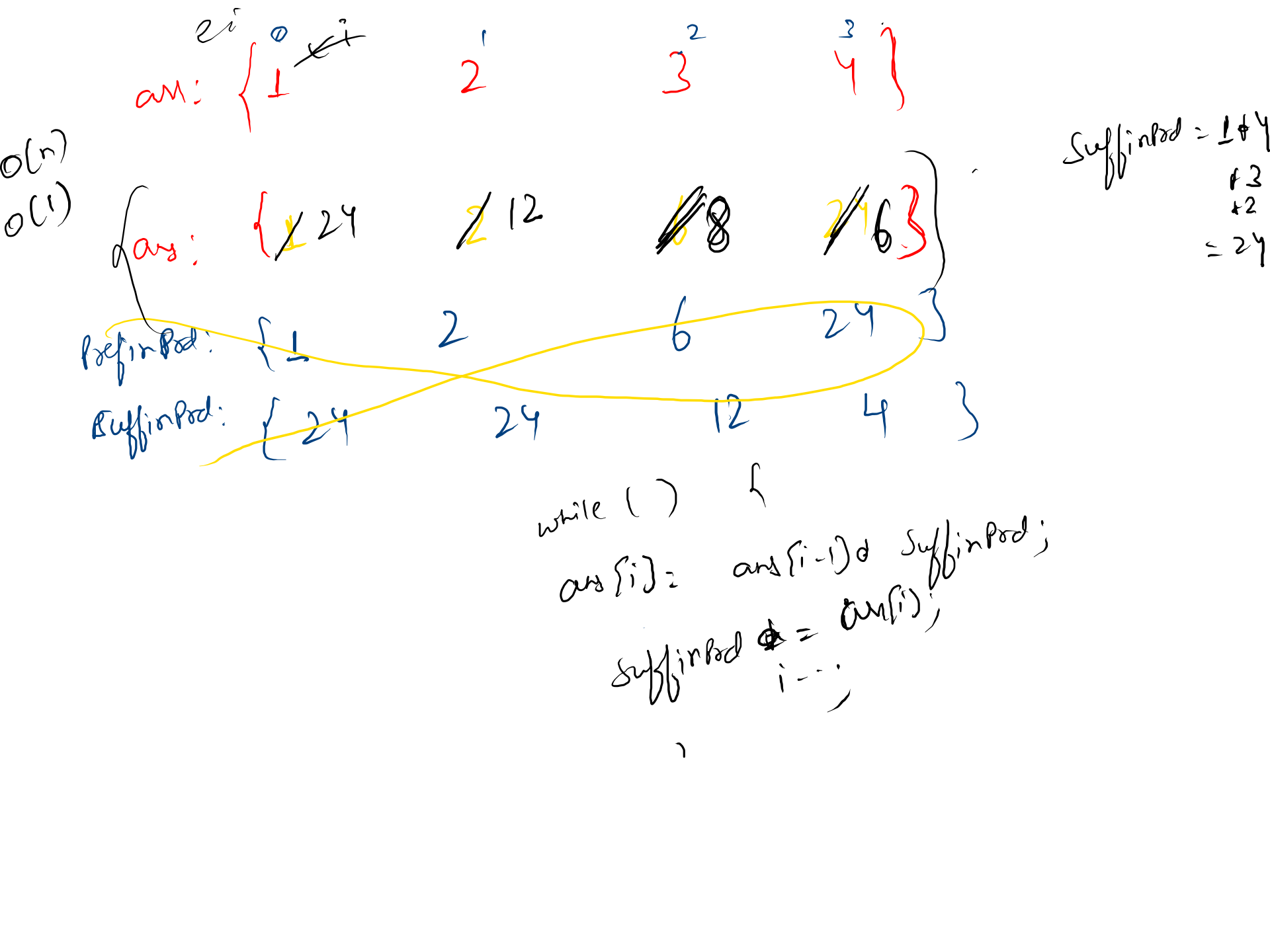
0 1 2 3 4 5

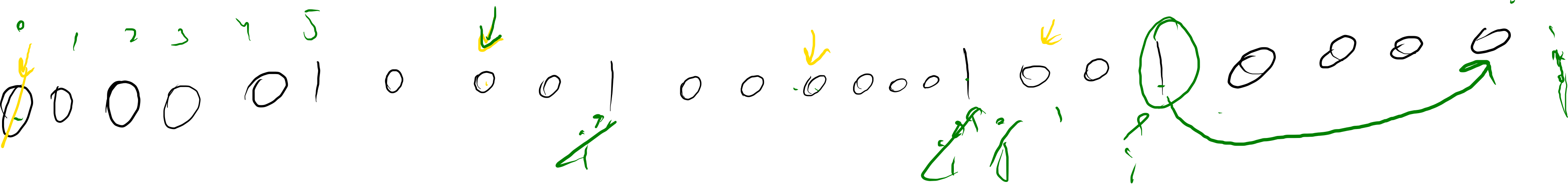
3 <= 5 >= 1 <= 6 >= 2 <= 4

{ nums[0] <= nums[1] >= nums[2] <= nums[3] >= nums[4] <= nums[5] }

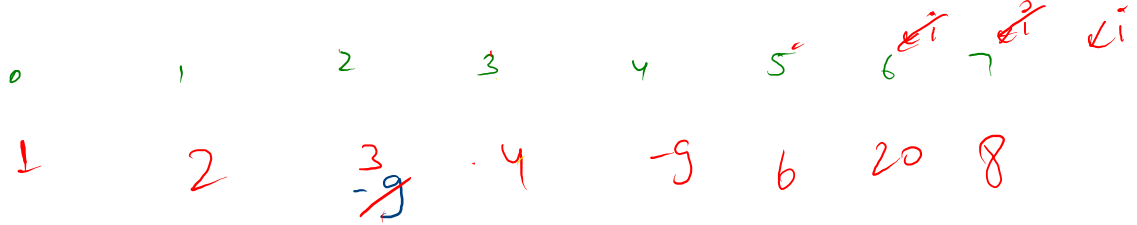
1 < 2 (6)

6 > 2


ans: { 1 2 3 4 }
O(n)
O(1)
ans: { 24 12 8 6 }
PrefinPod: { 1 2 6 24 }
SuffinPod: { 24 24 12 4 }
SuffinPod = 1*4*3*2 = 24
while ( ) {
ans[i] = ans[i-1] * SuffinPod;
SuffinPod *= ans[i];
i--;
}



Ans: ~~0~~ 5



```
for(int i =0;i<nums.length;i++){
    if(i+1 == nums[i]){
        continue;
    }

    if(nums[i]< 1 || nums[i] > n){
        continue;
    }


    int idx1 = i;
    int idx2 = nums[i] - 1;

    if(nums[idx2] == nums[idx1]){
        continue;
    }

    int temp = nums[idx1];
    nums[idx1] = nums[idx2];
    nums[idx2] = temp;
    i--;

}
```

Complexity → O(n)

Swapping

(1, 8)

~~As range → (1, n+1)~~

Checkrange → (1, n)

O(n) + O(n) ⇒ O(2n) ≅ O(n)

O(n) → time

O(1) → space