i	1	2	3	4	5	......	m
s	1	1+2	1+2+3	1+2+3+4	1+2+3+4+5		$\boxed{1+2+\dots m}$

```
i=0
s=0
while( s <= n){
    i++;
    s = i+s
}
```

for O(n)

logn

$\sqrt{n}$

$1+2+3 \dots m \le n$

$\frac{m(m+1)}{2} \le n$

$\rightarrow \boxed{m^2+m \le 2n}$

$m^2+m \le 2n$

$m^2+m^2 \le 2n$

$2m^2 \le 2n$

$m^2 \le n$

$m \le \sqrt{n}$

$\boxed{m = \sqrt{n}}$

$O(\sqrt{n})$

```
for( i=1, j=1; i< =n ;j++){
    if( j == n){
        i++;
        j=0;
    }
}
```

}k

n

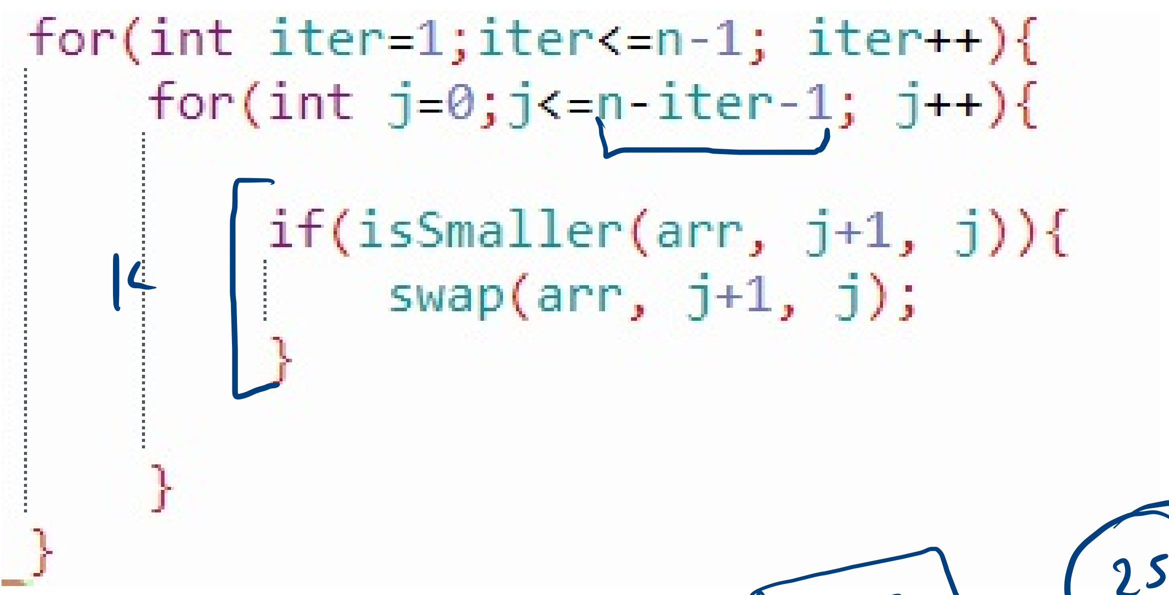
i	j	work
1	1 2 3 4 5 ..... n	n k
2	0 1 2 3 4 5 . — n	n k
3	0 1 2 3 4 5 . — n	
4	0 1 2 3 4 - — . n	n k
⋮		
n	0 1 2 3 . . . n	n k

nk + nk + nk (n times)

$n(nk)$

$\longrightarrow O(n^2)$

$O(n^2) = \Omega(n^2)$ n



```
for(int iter=1;iter<=n-1; iter++){
    for(int j=0;j<=n-iter-1; j++){
        if(isSmaller(arr, j+1, j)){
            swap(arr, j+1, j);
        }
    }
}
```

k

iter	j	work
1	0 ..... (n-2)	(n-1) k
2	0 .... (n-3)	(n-2) k
3	0 .... (n-4)	(n-3) k
n-1	0 0	1 k

[1 2 3 4 5] $\boxed{n=5}$

[5 4 3 2 1] n=5

25

25

$(n-1)k + (n-2)k + (n-3)k + \ldots\ldots 1k$

$k[(n-1) + (n-2) + (n-3) \ldots\ldots 1]$

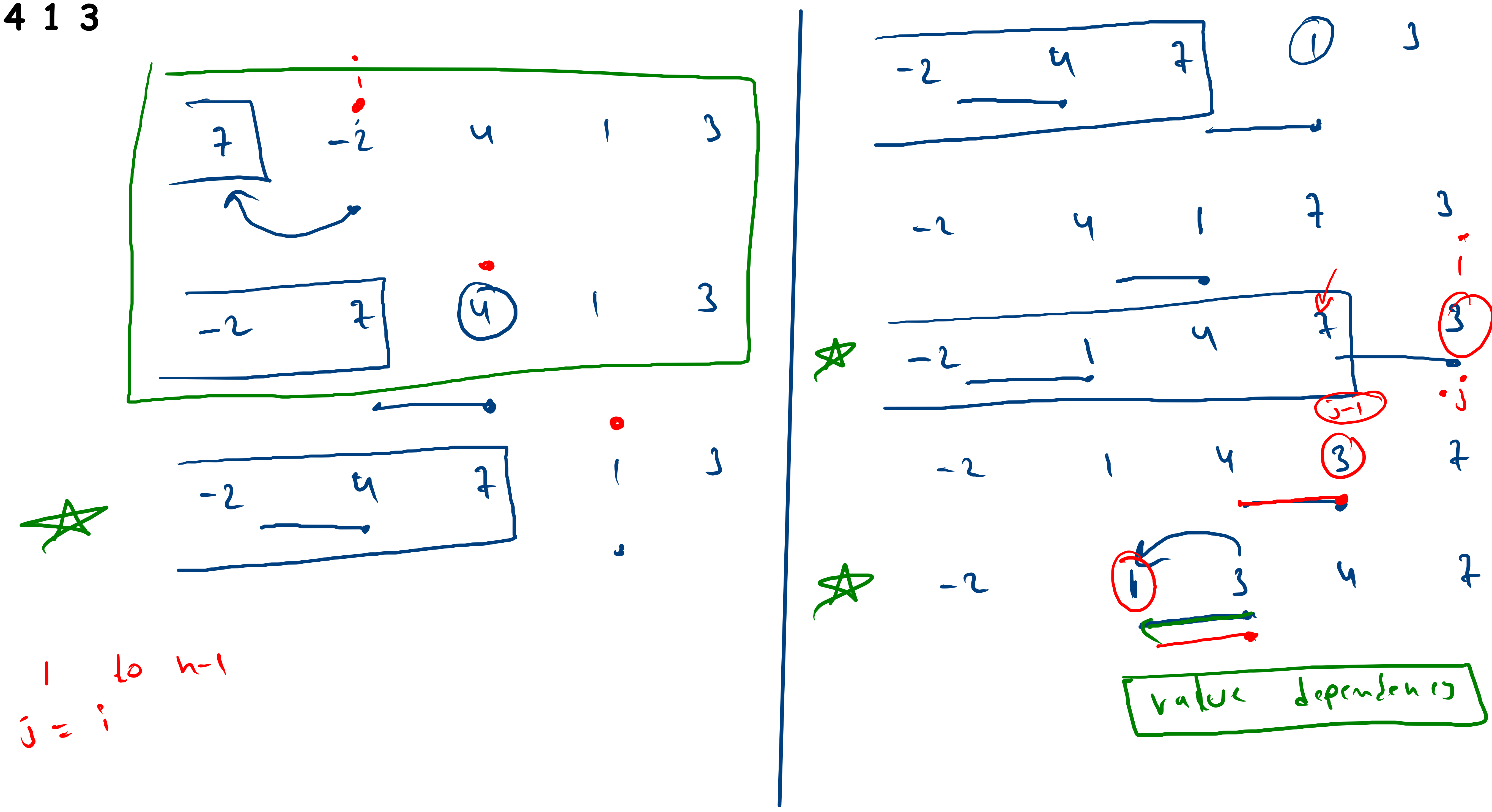
$f(n) = k\left[\frac{(n-1)n}{2}\right]$

$c_1 g(n) = \frac{k}{2}[n^2 - n]$

$= \frac{k}{2}(n^2)$

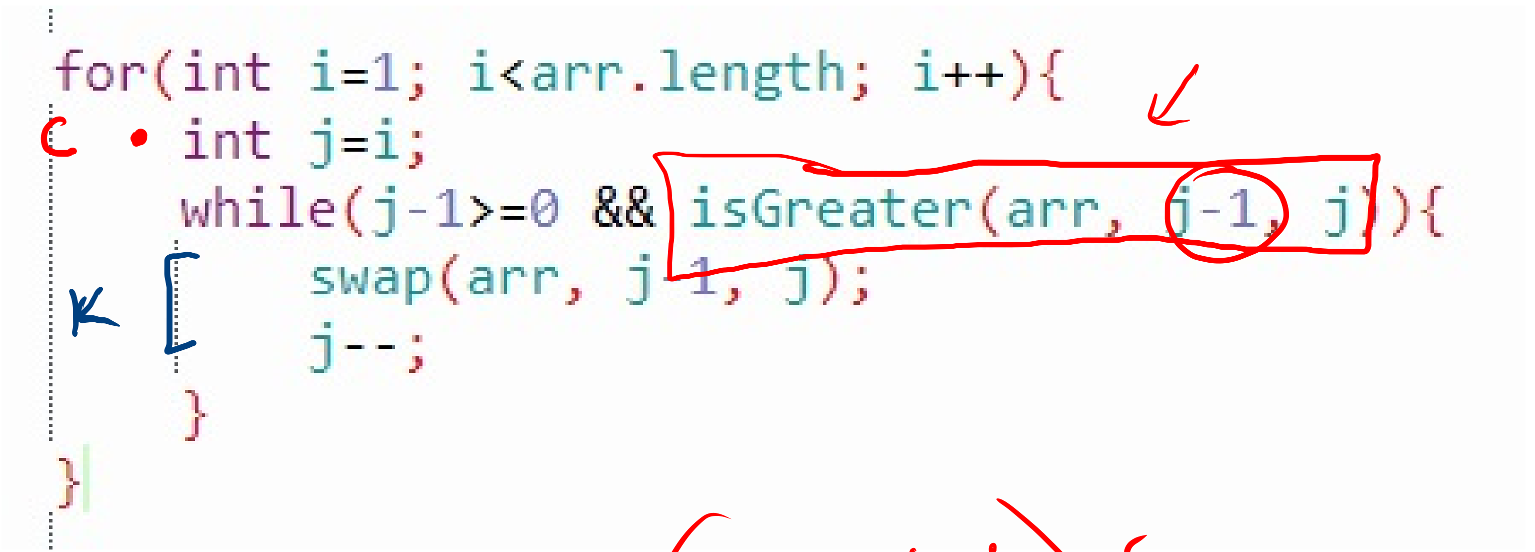
$\rightarrow O(n^2) \Leftarrow$

7 -2 4 1 3



i → 1 to n-1

j = i



```
for(int i=1; i<arr.length; i++){
    int j=i;
    while(j-1>=0 && isGreater(arr, j-1, j)){
        swap(arr, j-1, j);
        j--;
    }
}
```

C

K

$(1 \cdots n-1)\, c$

$(n-1)\, c$

$nc - c$

$\Omega(n)$

$[1 \quad 2 \quad 3 \quad 4 \quad 5]$

$O(n)$

$[5 \quad 4 \quad 3 \quad 2 \quad 1]$

$O(n^2)$

4 5 3 2 1

4 3 5 2 1

3 4 5 2 1

3 4 2 5 1

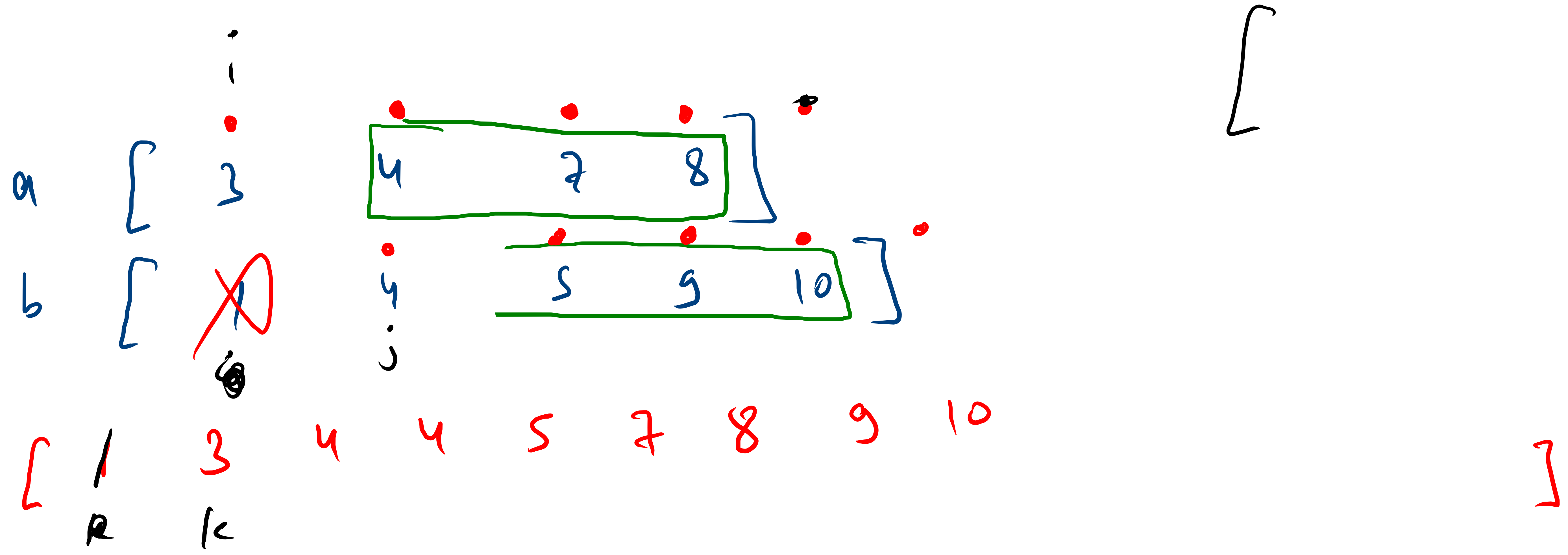
3 2 4 5 1

2 3 4 5 1

i	worst
1	1k
2	2k
3	3k
⋮	
n-1	(n-1)k

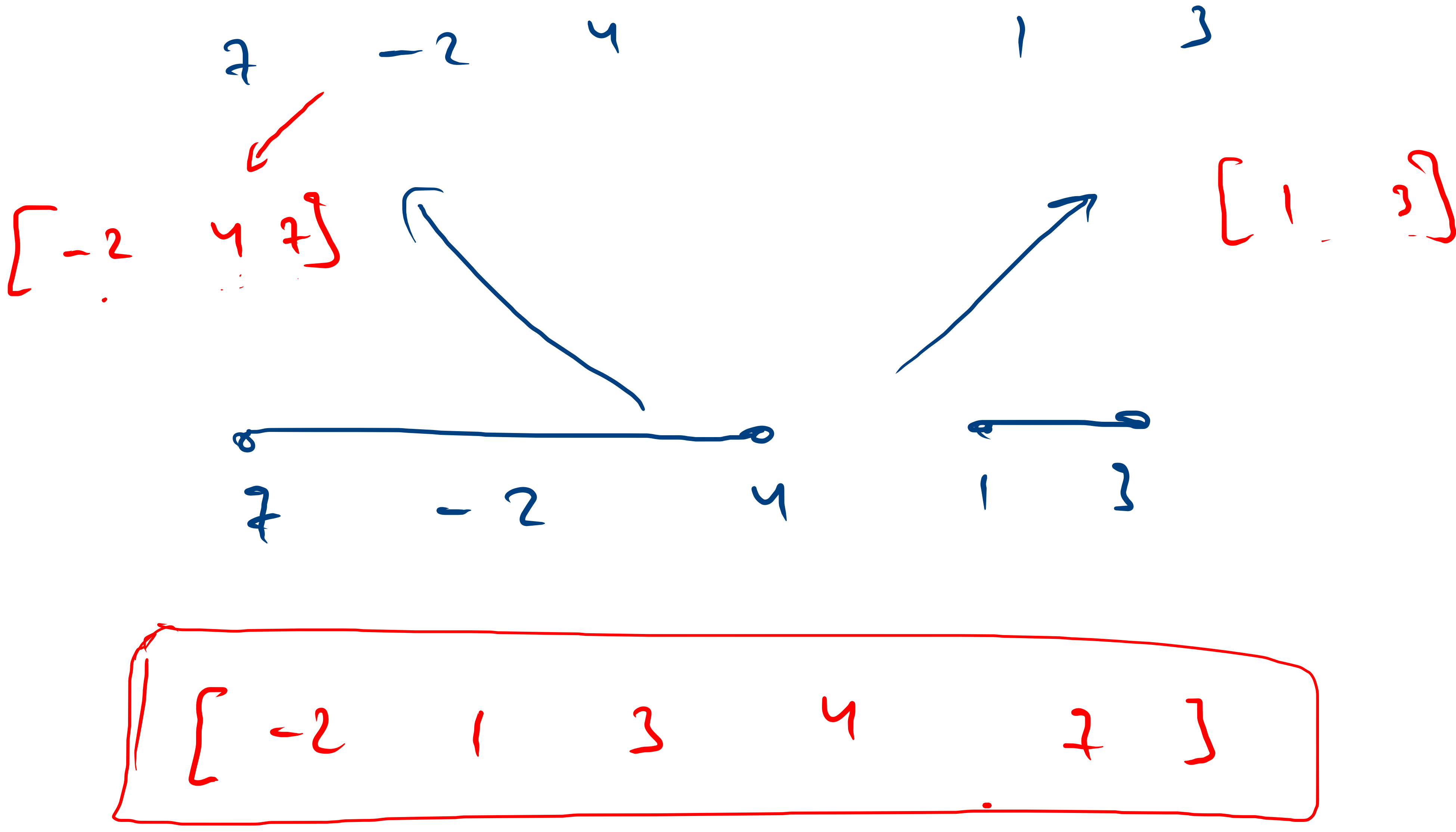
$$\frac{(n-1)\,n}{2}$$

$$\boxed{O(n^2)}$$

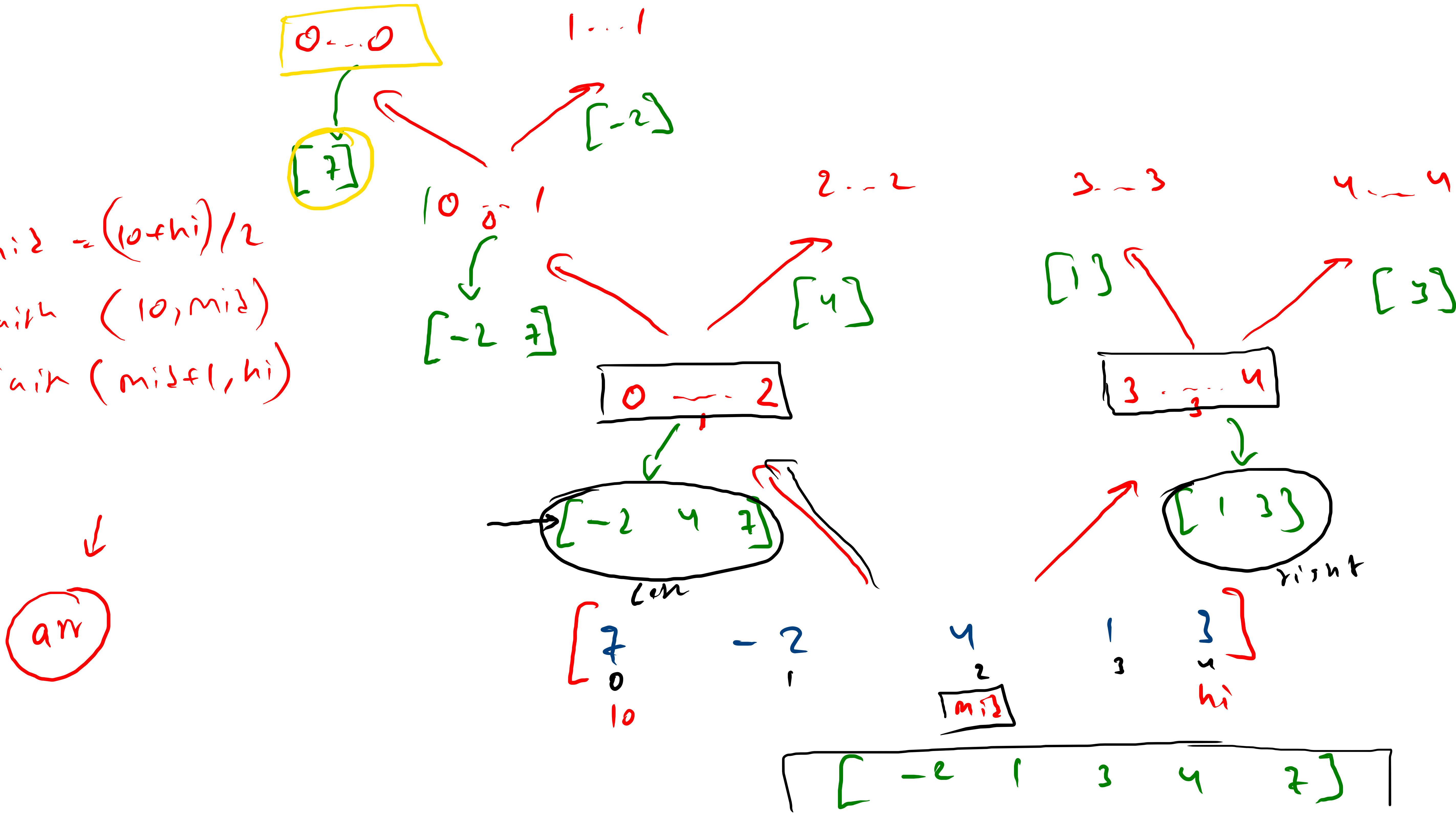


ans [1 3 4 4 5 7 8 9 10]

7 -2 4 1 3



```
mid = (lo+hi)/2
fair (lo, mid)
fair (mid+1, hi)
```



```
sum = 0;
for( i=0; i <= n; i++){
     sum = sum + 1
}
```

i

$1 \; [\; 0$

$n \; [\; 1, 2, 3, \ldots, n$

$(n+1)k$

$O(nk)$

k k k k

```
sum ( int n){
     if(n==0)return 0;
     return n + sum(n-1)
}
```

k

$(n+1)k$

$\boxed{O(n)}$

$sum(n) = 0 + 1 + 2 + 3 \ldots\ldots n$

$sum(n) = n + sum(n-1)$ $\leftarrow$

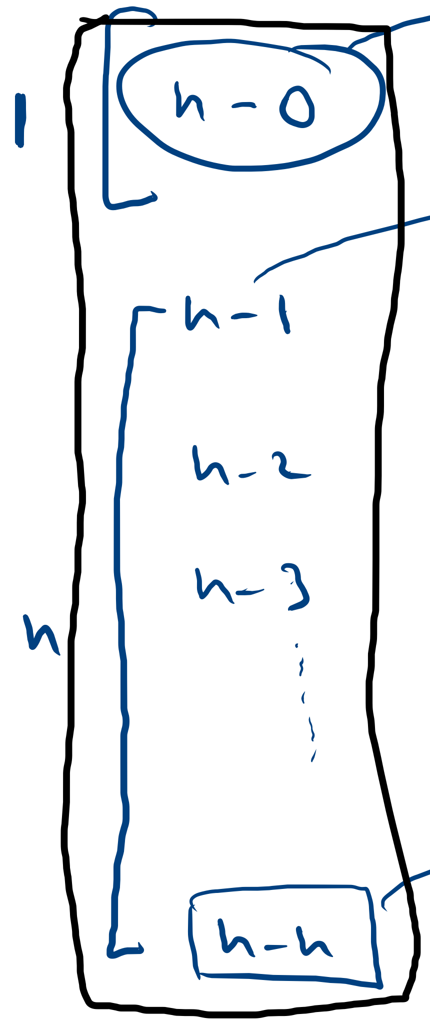
```
sum ( int n){
    if(n==0)return 0;
    return n + sum(n-1)
}
```

recurrence relation

$f(n)$

n+1

① $$\boxed{T(n) = T(n-1) + k}$$



$T(n) = \cancel{T(n-1)} + k$

$\cancel{T(n-1)} = \cancel{T(n-2)} + k$

$\cancel{T(n-2)} = \cancel{T(n-3)} + k$

$\vdots$

$\cancel{T(0)} = k$

$$\boxed{T(n) = (n+1)k}$$

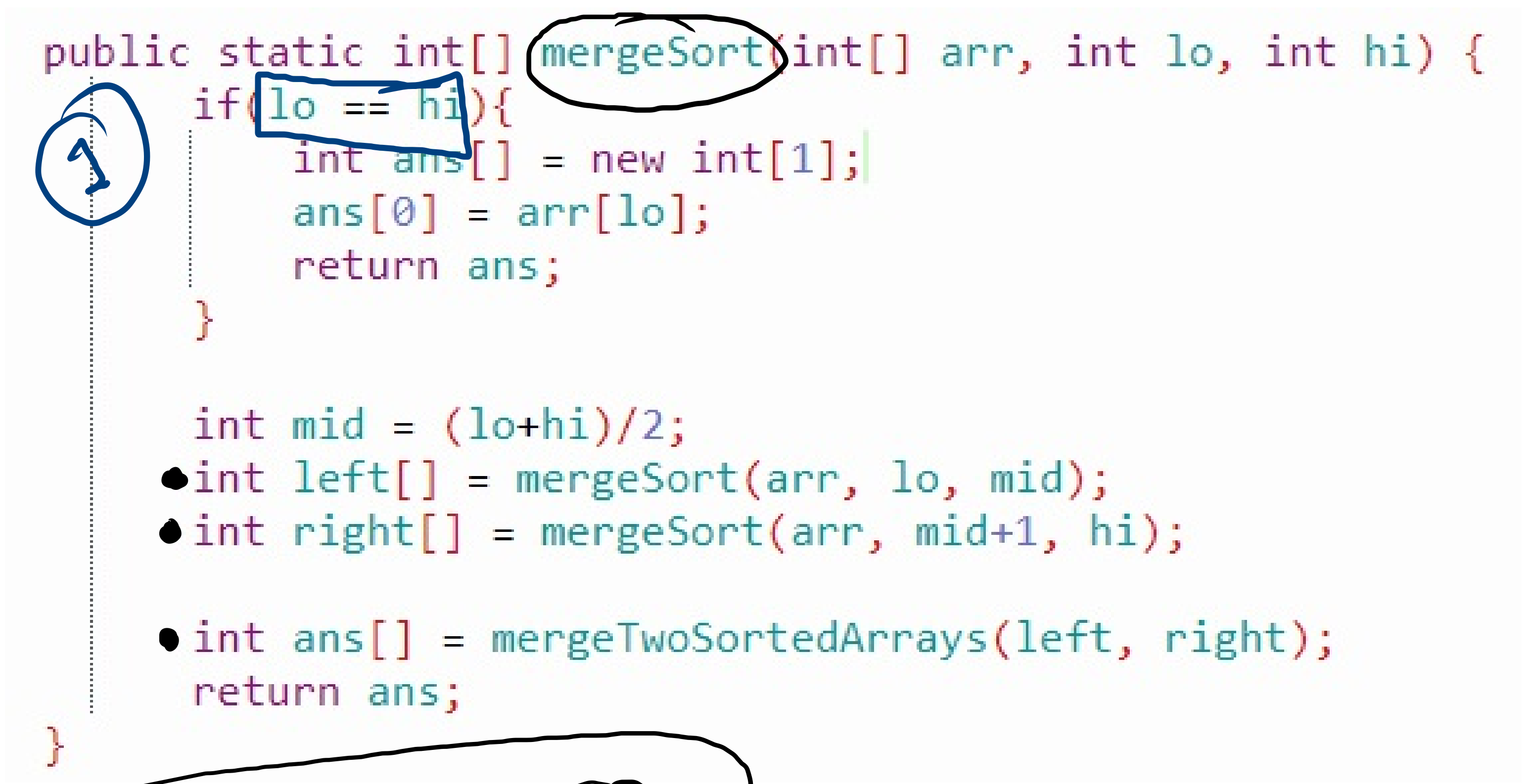
$(n+1)k$

$nk + k$

$2nk$

$2k \times n$

$O(n)$



```
public static int[] mergeSort(int[] arr, int lo, int hi) {
    if(lo == hi){
        int ans[] = new int[1];
        ans[0] = arr[lo];
        return ans;
    }

    int mid = (lo+hi)/2;
    int left[] = mergeSort(arr, lo, mid);
    int right[] = mergeSort(arr, mid+1, hi);

    int ans[] = mergeTwoSortedArrays(left, right);
    return ans;
}
```

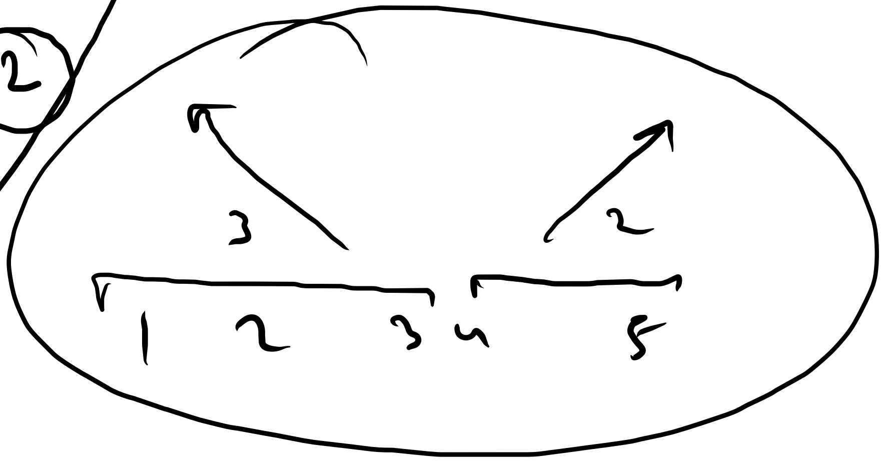
$$\boxed{T(n) = \underbrace{T\left(\frac{n}{2}\right)}_{\text{left}} + \underbrace{T\left(\frac{n}{2}\right)}_{\text{right}} + n}$$

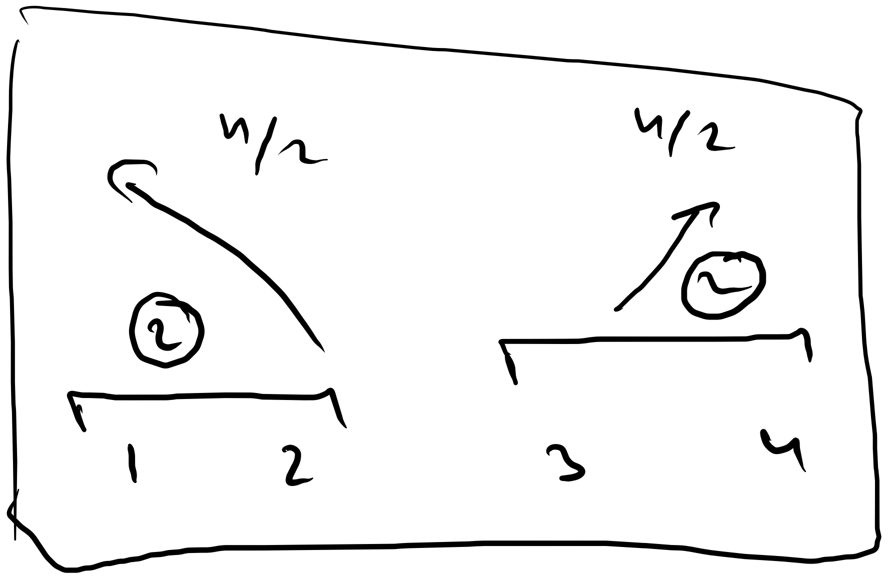
$$\boxed{T(n) = 2T\left(\frac{n}{2}\right) + n}$$

$a[\cdot, \cdot, \cdot, \dots]$ l_1

$b[\cdot, \cdot, \cdot, \dots]$ l_2

$O(l_1 + l_2)$





$n=4$

$$\boxed{T(n) = 2T\left(\frac{n}{2}\right) + n}$$

work

$$\begin{array}{l|l|rcl|l}
1 & \frac{n}{2^0} & T(n) & = & \cancel{2T\left(\frac{n}{2}\right)} + n & \\
m & n/2^1 & \cancel{2T\left(\frac{n}{2}\right)} & = & \cancel{4T\left(\frac{n}{4}\right)} + n & \\
 & n/2^2 & \cancel{4T\left(\frac{n}{4}\right)} & = & \cancel{8T\left(\frac{n}{8}\right)} + n & \\
 & n/2^3 & \cancel{8T\left(\frac{n}{8}\right)} & = & \cancel{16T\left(\frac{n}{16}\right)} + n & \\
 & n/2^4 & \cancel{16} & \vdots & \cancel{n} & \\
 & \vdots & \cancel{n} & \vdots & & \\
 & \frac{n}{2^m} & \cancel{X\,T(1)} & = & n &
\end{array}$$

$$\frac{n}{2^m} = 1$$

$$n = 2^m$$

$$\log_2 n = \log_2 2^m$$

$$\log_2 n = m \log_2 2$$

$$\boxed{m = \log_2(n)}$$

$$m+1$$

$$T(n) = (m+1)n$$

$$T(n) = \boxed{n\log(n)} + \boxed{n}$$

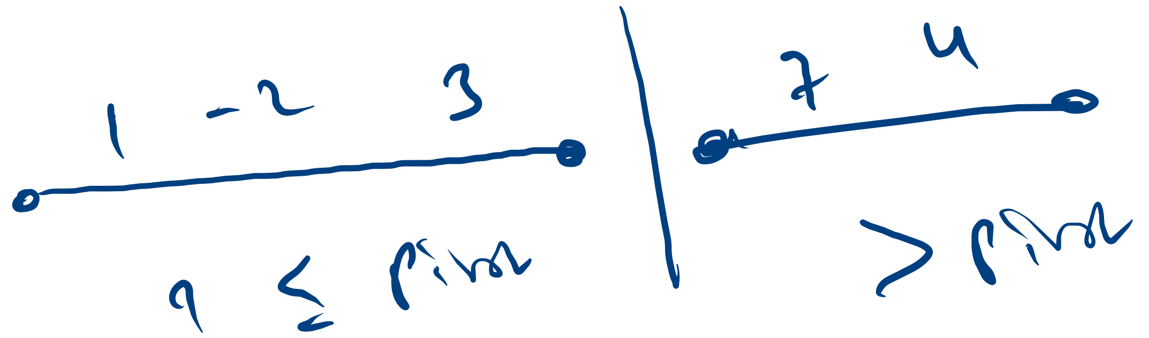
$$\Theta\left(n \log(n)\right)$$

$\boxed{\text{Pivot} \rightarrow 3}$

Piv $\rightarrow \infty$

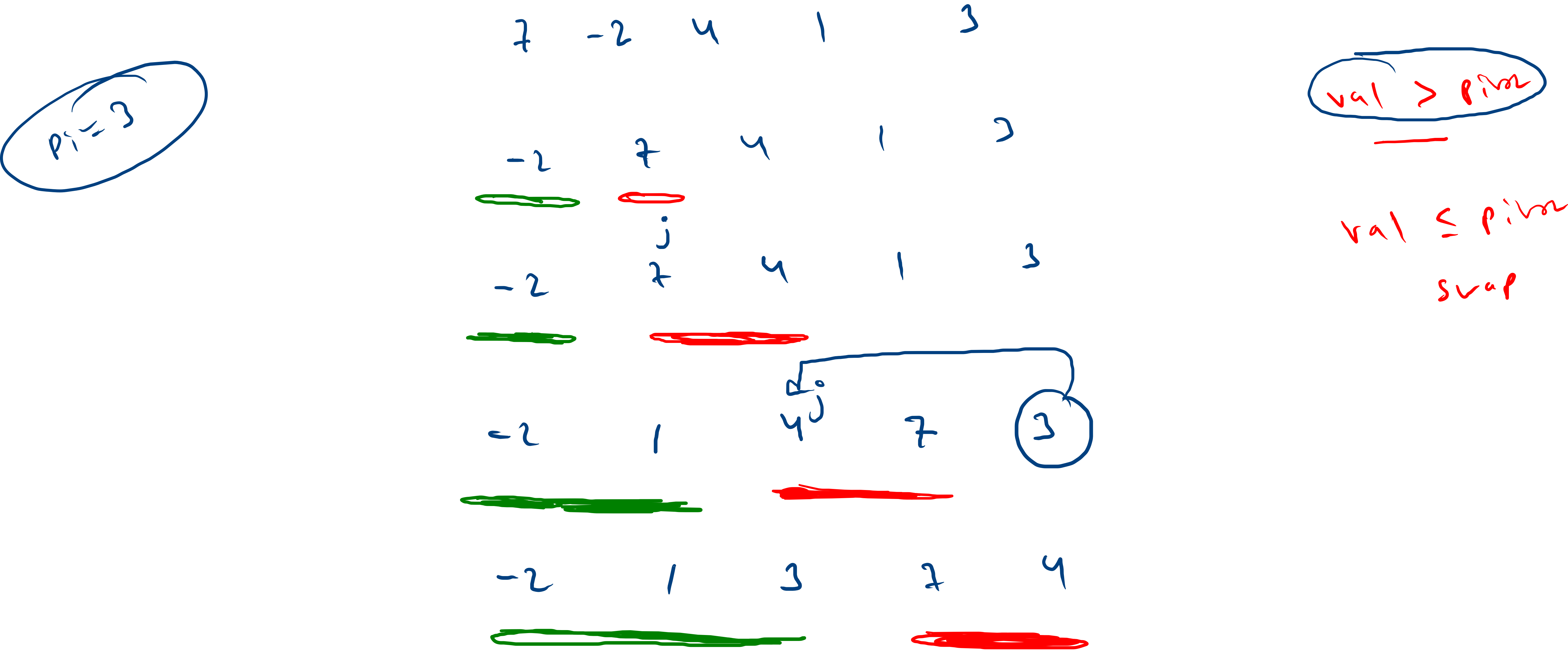
Time	$O(n)$	$O(n)$
space	$O(n)$	$O(1)$

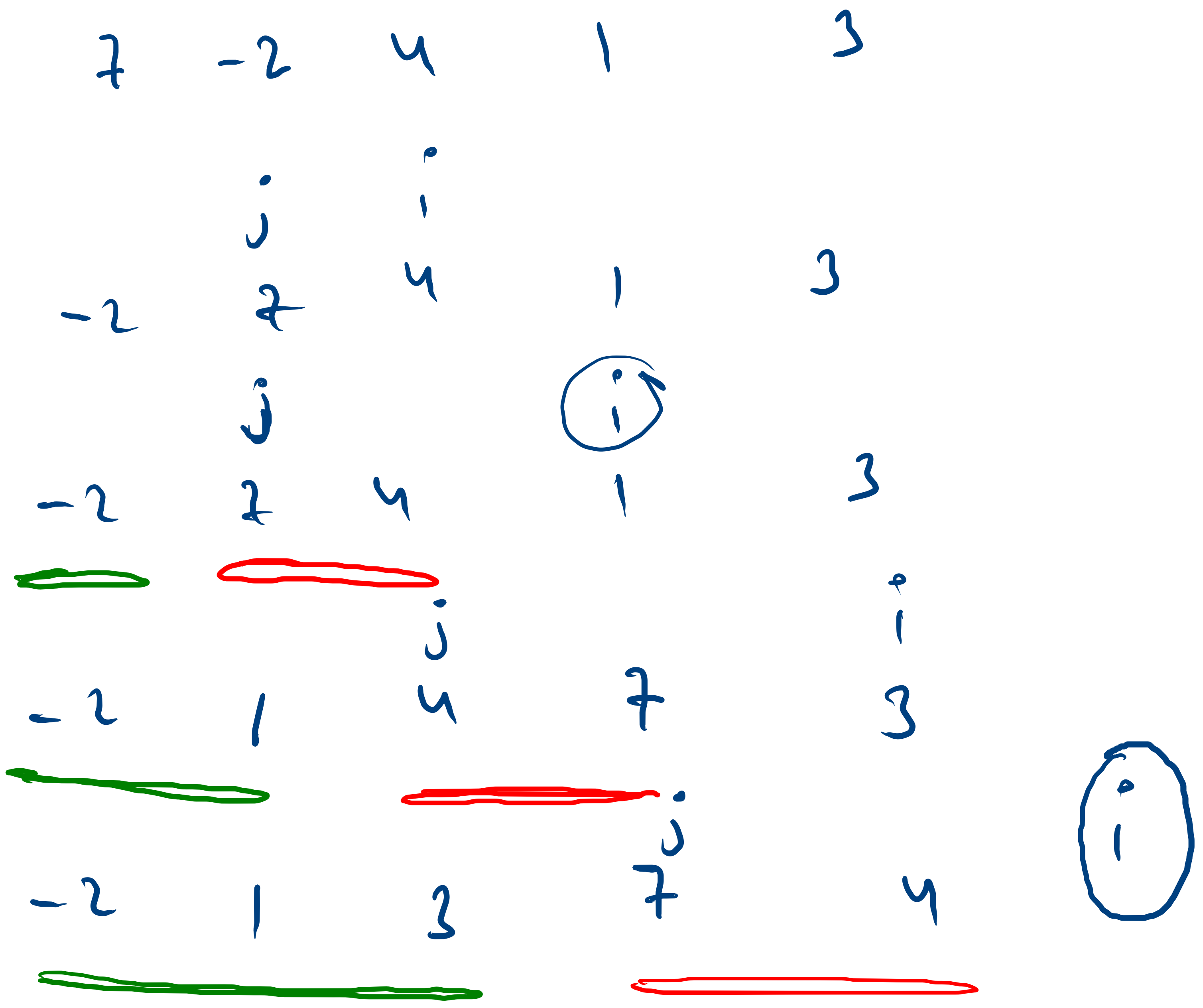
7 -2 4 1 3



≤ 3 $\quad$ > 3

7 -2 4 1 3 $\boxed{3}$


pi = 3
7 -2 4 1 3
-2 7 4 1 3
j
-2 7 4 1 3
-2 1 4 7 3
j
-2 1 3 7 4
val > pivot
val ≤ pivot
swap

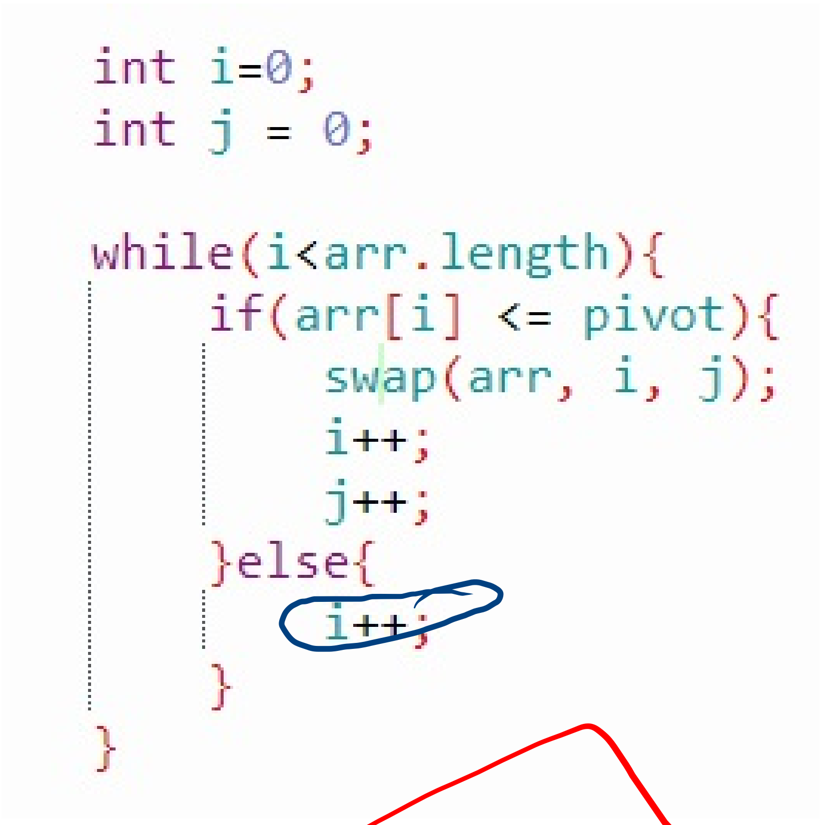


```
→ i = 0
  j = 0

val > pivot
    i++

val ≤ pivot
    swap(i, j)
    j++
    i++
```

$\boxed{\text{Pivot} = 3}$



```
int i=0;
int j = 0;

while(i<arr.length){
    if(arr[i] <= pivot){
        swap(arr, i, j);
        i++;
        j++;
    }else{
        i++;
    }
}
```

j   i

3   2   1   2   4   6   5   7

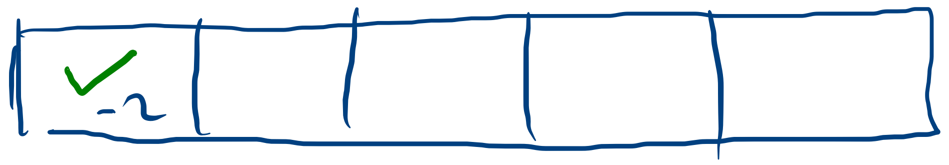
$\boxed{\text{H.W}}$

complexity

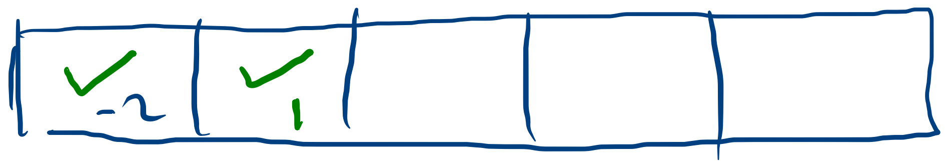
pivot = arr[lastindex]

$7 \quad -2 \quad 4 \quad 1 \quad 3 \longrightarrow \boxed{-2} \quad 1 \quad 3 \quad 4 \quad 7$

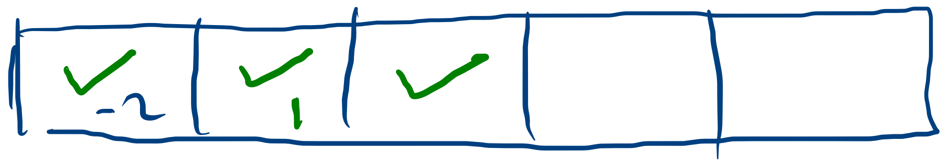
iter 1



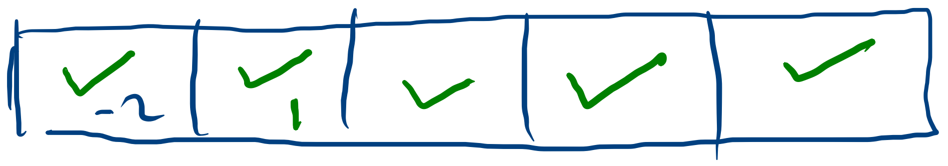
iter 2



iter 3



iter 4



↓

7 -2 4 1 3

min

-2 7 4 1 3

min

-2 1 4 7 3

min

-2 1 3 7 4

-2 1 3 4 7