5-) Otherfunction —— O(1)

```
Some function(arr[], arr_len) {
    for (i=0; i<arr_len-1; i++)  —— O(arr_len)
    {
        min_idx = i  —— O(1)
        for (j = i+1; j<arr_len; j++)  —— Σ_{i=0}^{n-1} n-i-1
            if(arr[j] < arr[min_idx]) — O(1)
                min_idx = j — O(1)
        Otherfunction(arr[min_idx], arr[i]) — O(1)
    }
```

3

Outter loop

execute arr_len times

Nested loop

i=0	executed	arr_len+1	times
i=1	executed	arr_len-1	times
i=2	executed	arr_len-2	times
⋮		⋮	
i=arr_len-2	executed	1	times

$n+(n-1)+(n-2)+\dots+1$

$= \frac{n\cdot(n+1)}{2}$ (n+1)

$= \frac{n^2+n}{2}$ len + 1

$\Theta(arr\_len^2)$

$T^{(n)}_{Best} = T^{(n)}_{Average} = T^{(n)}_{worst} = O(n^2) = \Omega(n^2) = \Theta(n^2)$

6-) for i=0 to arr_len

execute arr_len time

for(j=i to arr_len)

arr_len	$a\frac{n_2(n+1)}{2}$
arr_len-1	
arr_len-2	
⋮	
1	

for (j=1 to n%i) in Otherfunction

When i=1 n%1=1

In best case n can divide all i values so this loop constant

$T^{(n)}_{best} = \Theta(n^2)$

In worst case

i	n%i
1	0
2	1
3	2
4	3
⋮	⋮
n	n-1

$\frac{n-1 \cdot n}{2} = \frac{n^2-n}{2}$ so total complexity

$T^{(n)}_{worst} = T^{(n)}_{Ave} = \Theta(n^4)$

I didn't take for i to b loop because otherfunction always called with b=2 so this is constant time complexity loop

7-)

```
for(i=0; i<=arr_len-1; i++)  —— n time
    Otherfunction(arr, i)  —— Σ_{i=0}^{n} log n = log 1 + log 2 + log 3 + ... log n
                                                 = log(n!)
```

```
for(j=1; j<n; j=j*2)
```

j goes 1 2 4 8 16 ... n

So this for loop time complexity log n

$T^{(n)}_{Best} = T^{(n)}_{worst} = T^{(n)}_{Ave} = O(n \log n)$