1) Sel Sort

```
From i to length of Array
    From j=i+1 to length of Array
        if A[i] > A[j]
            swap A[i] and A[j]
```

Invariants

1) The outer loop will contain sorted elements from least to greatest

2) The inner loop value A[j] is larger then any value in the outer loop

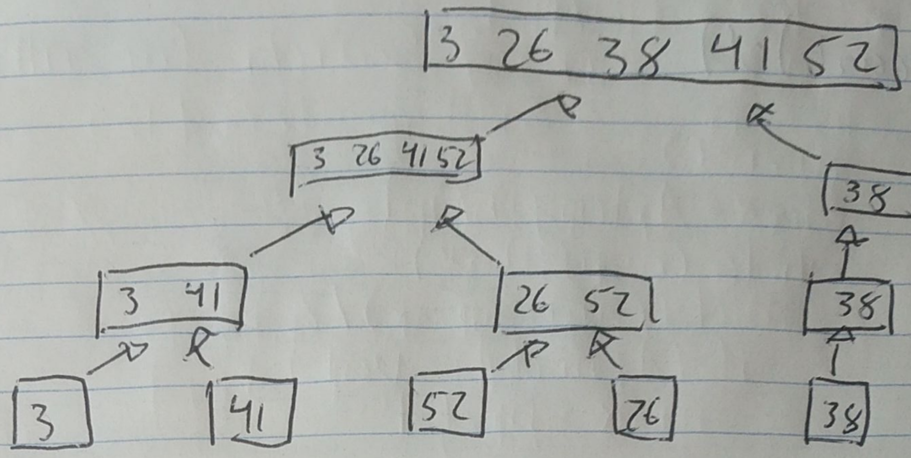
Why not $n-1$

The final element will be automatically sorted because each element will be in the correct spot already

Run times

Both best and worst case is $\Theta(n^2)$ becase both sorted and unsorted portion need to be looped through

2)



3) If there is only one element or even zero then there is nothing to ~~sep~~ sort. thus $\Theta(1)$

The time for all elements besides n can be represented by $T(n-1)$

The time for the nth element can be represented as $\Theta(n)$

$$T(n) = \begin{cases} \Theta(1) & n \leq 1 \\ T(n-1) + \Theta(n) & n > 1 \end{cases}$$

5) a) i 2 3 8 6 1

j 2 3 8 6 1

$$\boxed{(2,1), (3,1), (8,6), (8,1), (6,1)}$$

b) The array that will have the most inversions is

$\{n, \ldots, 3, 2, 1\}$

It will have $(n-1)!$ inversions