1 Selection Sort

0: [12, 7, 5, 13, 6, 2, 4]

i, min — j — Search new min

1: [12, 7, 5, 13, 6, 2, 4]

i — SWAP — min — j

2: [2, 7, 5, 13, 6, 12, 4]

i, min — j

3: [2, 7, 5, 13, 6, 12, 4]

i, min — j — Search new min

4: [2, 7, 5, 13, 6, 12, 4]

i — SWAP — j, min

5: [2, 4, 5, 13, 6, 12, 7]

i, min — j

K-Steps

K: [2, 4, 5, 6, 7, 12, 13]

i — j

Invariant: Selection sort

For any given value of i in the main loop of selection sort, at the beginning of the iteration, the following invariants hold:

1. $array[1..i-1]$ is sorted
2. For any x in $array[1..i-1]$ and y in $array[i..n]$, $x \leq y$

LI: arr [1...0] is sorted

```
for (i = 1, i <= n; i++):
    min = i
```

LI: arr [1... i-1] is sorted & $x \in$ arr [1... i-1],

for (j = (i+1), j <= n, j++)

$y \in arr[i \ldots n]$
where $x <= y$

:

if (arr[j] < arr[min]):
min = arr[j]

swap (arr[i], arr[min])

LI: arr[1... i] is sorted & $x \in arr[1 \ldots i]$, $y \in arr[i+1 \ldots n]$ where $x <= y$.

LI: arr[1... i] is sorted where i = n when loop terminates, so arr[1... n] is sorted.

Code:

```
def selectionSort(arr, n):

    for i in range(n):
        min = i
        for j in range( i+1, n ):
            if arr[ j ] < arr[min]:
                min = j

        arr[ i ], arr[min] = arr[min], arr[ i ]

    return arr
```

Time:

$$T(n) = n + (n-1) + (n-2) + \ldots + 2 + 1$$

$$T(n) = n\left(\frac{n-1}{2}\right) = \frac{n^2 - n}{2}$$

Best / Worst-Case: $O(n^2)$

Space: $O(1)$ In-Place: Yes, as aux space is $O(1)$.

Stable: No, because selection sort swaps non-adjacent elements which changes the relative order of elements with the same value.

Online: No, because if any new elements are added to the input, the algo. would have to start all over.