Selection Sort Correctness

```
function selectionSort (arr: Array) {
  for (let i = 0; i < arr.length - 1; i++) {
    let minIndex = i;
    for (let j = i + 1; j < arr.length; j++) {
      if (arr[j] < arr[minIndex]) {
        minIndex = j;
      }
    }
    if (minIndex !== i) {
      let temp = arr[i];
      arr[i] = arr[minIndex];
      arr[minIndex] = temp;
    }
  }
  return arr;
}
```

The Selection Sort algorithm is a simple and intuitive sorting algorithm that works by repeatedly finding the minimum element from the unsorted part of the array and putting it at the beginning of the array. It then repeats this process for the remaining unsorted part of the array until the entire array is sorted. The algorithm has a time complexity of **O(n^2)**, which makes it inefficient for large lists. However, it has the advantage of being easy to understand and implement, and it performs well on small lists or lists that are already partially sorted.

The correctness of the Selection Sort algorithm can be proven by induction. The base case is trivial: a list of length 1 is already sorted. For the inductive step, assume that the algorithm correctly sorts a list of length n. We need to show that it also correctly sorts a list of length n+1. The algorithm works by finding the minimum element in the unsorted part of the array and swapping it with the first element of the unsorted part. This ensures that the first element of the unsorted part is always the minimum element. After n iterations, the first n elements of the array are sorted, and the (n+1)th element is the minimum element of the remaining unsorted part. Therefore, the algorithm correctly sorts a list of length n+1.