**<Selection Sort(선택 정렬)>**

1. Find the smallest card. Swap it with the first card.
2. Find the second-smallest card. Swap it with the second card.
3. Find the third-smallest card. Swap it with the third card.
4. Repeat finding the next-smallest card, and swapping it into the correct position until the array is sorted.

**Finding the index of the minimum element in a subarray**

One of the steps in selection sort is to find the next-smallest card to put into its correct location. For example, if the array initially has values [13, 19, 18, 4, 10], we first need to find the index of the smallest value in the array. Since 4 is the smallest value, the index of the smallest value is 3.

Selection sort would swap the value at index 3 with the value at index 0, giving [4, 19, 18, 13, 10]. Now we need to find the index of the second-smallest value to swap into index 1.

It might be tricky to write code that found the index of the second-smallest value in an array. I'm sure you could do it, but there's a better way. Notice that since the smallest value has already been swapped into index 0, what we really want is to find the smallest value in the part of the array that starts at index 1. We call a section of an array a subarray, so that in this case, we want the index of the smallest value in the subarray that starts at index 1. For our example, if the full array is [4, 19, 18, 13, 10], then the smallest value in the subarray starting at index 1 is 10, and it has index 4 in the original array. So index 4 is the location of the second-smallest element of the full array.

1. In the first call of indexOfMinimum, it has to look at every value in the array, and so the loop body in indexOfMinimum runs 8 times.
2. In the second call of indexOfMinimum, it has to look at every value in the subarray from indices 1 to 7, and so the loop body in indexOfMinimum runs 7 times.
3. In the third call, it looks at the subarray from indices 2 to 7; the loop body runs 6 times.
4. In the fourth call, it looks at the subarray from indices 3 to 7; the loop body runs 5 times.
5. …
6. In the eighth and final call of indexOfMinimum, the loop body runs just 1 time.

If we total up the number of times the loop body of indexOfMinimum runs, we get 8 + 7 + 6 + 5 + 4 + 3 + 2 + 1 = 36 times.