Selection Sort

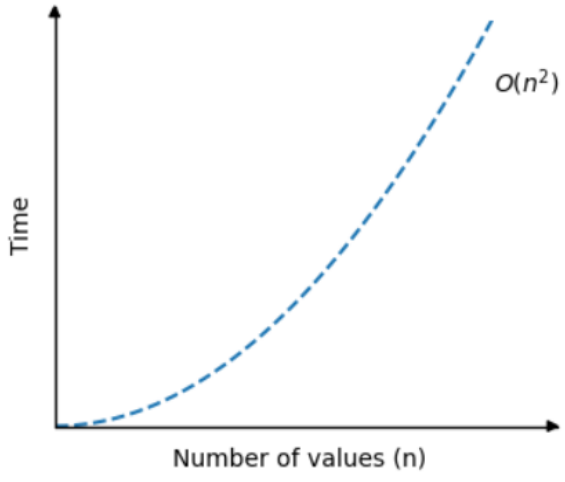
The selection sort algorithm finds the lowest value in an array and moves it to the front of the array. The algorithm looks through the array again and again, moving the next lowest values to the front, until the array is sorted.

Manual run through

Step 1: start with an unsorted array.
[7, 12, 9, 11, 3]
Step 2: Go through the array, one value at a time. Which value is the lowest? 3, right?
[7, 12, 9, 11, 3]
Step 3: move the lowest value 3 to the front of the array.
[3, 7, 12, 9, 11]
Step 4: look through the rest of the value, starting with 7. 7 is the lowest value, and already at the front of the array, so we don't need to move it.
[3, 7, 12, 9, 11]
Step 5: look through the rest of the array: 12, 9 and 11. 9 is the lowest value.
[3, 7, 12, 9, 11]
Step 6: move 9 to the front.
[3, 7, 9, 12, 11]
Step 7: looking at 12 and 11, 11 is the lowest.
[3, 7, 9, 12, 11]
Step 8: move it to the front.
[3, 7, 9, 11, 12]
Finally, the array is sorted.

Selection sort time complexity

Selection Sort sorts an array of **n** values. On average, about n/2 elements are compared to find the lowest value in each loop. And Selection Sort must run the loop to find the lowest value approximately **n** times. We get time complexity: **$O(n.n/2) = O(n^2)$**
The time complexity for the Selection Sort algorithm can be displayed in a graph like this:



[code]