Insertion-Sort/Execution-Counter Worksheet

Assume array A is indexed from 1 to n.

INSERTION_SORT(A, n)
1. **for** $j \leftarrow 2$ **to** n **do**
2. $\quad key \leftarrow A[j]$
3. $\quad i \leftarrow j - 1$;
4. $\quad$ **while** $i > 0$ **and** $A[i] > key$ **do**
5. $\quad\quad A[i+1] \leftarrow A[i]$
6. $\quad\quad i \leftarrow i - 1$
7. $\quad A[i+1] \leftarrow key$

Instance 1 : [4, 3, 2, 1]
Instance 2 : [1, 4, 2, 3]
Instance 3 : [5, 4, 3, 2, 1]
Instance 4 : [1, 2, 3, 4]

	# Times Executed			
Line No	Instance 1	Instance 2	Instance 3	Instance 4
L1	3	3	4	3
L2	3	3	4	3
L3	3	3	4	3
L4	8	7	16	3
L5	6	2	12	0
L6	6	2	12	0
L7	3	3	4	3
Total	32	23	56	15

List any observations.

In instance 1, the list is in reverse order and has $2n^2$ operations, while the instance that I created was the same size and already sorted, and had $n^2 - 1$ steps (according to my calculations).

More generally speaking, the closer to being sorted an input was, the fewer instructions it took, as long as the size was held constant. However, even with a small increase in the length of the input the number of times each line was executed jumped dramatically.