Pior caso:

Insertion Sort	**Custo**	**Vezes**
for j ← 2 **to** comprimento [A]	c1	n
do chave ← A[j]	c2	n - 1
i ← j – 1	c3	n - 1
while i > 0 **e** chave < A[i]	c4	$\sum_2^n t$
do A[i+1] ← A[i]	c5	$\sum_2^n t - 1$
i ← i – 1	c6	$\sum_2^n t - 1$
A[i + 1] ← chave	c7	n – 1

T(n) = c1 * n + c2 * (n – 1) + c3 * (n - 1) + c4 * $\sum_2^n t$ + (c5 + c6) * $\sum_2^n (t-1)$ + c7 * (n - 1)

Temos que:

$\sum_2^n t = \frac{n*(n+1)}{2} - 1$

$\sum_2^n t - 1 = \frac{n*(n-1)}{2}$

Substituindo:

T(n) = c1 * n + c2 * (n – 1) + c3 * (n - 1) + c4 *($\frac{n^2}{2} + \frac{n}{2} - 1$) + (c5 + c6) * ($\frac{n^2}{2} - \frac{n}{2}$) + c7 * (n - 1)

Organizando:

T(n) = $\frac{c4+c5+c6}{2}$ * n^2 + (c1 + c2 + c3 + $\frac{c4}{2}$ - $\frac{c5}{2}$ - $\frac{c6}{2}$ + c7) * n – (c2 + c3 + c4 + c7)

Melhor caso:

Insertion Sort	**Custo**	**Vezes**
for j ← 2 **to** comprimento [A]	c1	n
do chave ← A[j]	c2	n - 1
i ← j – 1	c3	n - 1
while i > 0 **e** chave < A[i]	c4	n - 1
do A[i+1] ← A[i]	c5	0
i ← i – 1	c6	0
A[i + 1] ← chave	c7	n – 1

T(n) = (c1 + c2 + c3 + c4 + c7) * n – (c2 + c3 + c4 + c7)

A primeira análise considera o pior caso do Insertion Sort, que é quando a entrada está na ordem inversa da que deseja-se ordenar. Para esse caso, o Insertion se comporta como uma função quadrática, podendo ser expresso na notação grande-O como O(n^2). Porém se a entrada estiver totalmente ordenada, teremos o melhor caso e então o comportamento passa a ser linear, como pode ser conferido na segunda análise. Dessa forma, podemos expressar na notação grande-O como O(n).

Selection Sort	**Custo**	**Vezes**
for i ← 1 **to** comprimento [A] - 1	c1	n
menor ← i	c2	n - 1
for j ← i + 1 **to** comprimento [A]	c3	$\sum_1^n t$
if A[j] < A[menor]	c4	$\sum_1^{n-1} t$
menor ← j	c5	$\sum_1^{n-1} t$
if menor != i	c6	n - 1
swap(A[menor] , A[i])	c7	n - 1

T(n) = c1 * n + c2 * (n – 1) + c3 * $\sum_1^n t$ + (c4 + c5) * $\sum_1^{n-1} t$ + c6 * (n – 1) + c7 * (n - 1)

Organizando, temos:

T(n) = c1 * n + (c2 + c6 + c7) * (n – 1) + c3 * $\sum_1^n t$ + (c4 + c5) * $\sum_1^{n-1} t$

Temos que:

$\sum_1^n t = \frac{(1+n)*n}{2} = \frac{n^2}{2} + \frac{n}{2}$

$\sum_1^{n-1} t = \frac{(1+n-1)*(n-1)}{2} = \frac{n^2}{2} - \frac{n}{2}$

Substituindo:

T(n) = c1* n + (c2 + c6 + c7) * (n - 1) + c3 * $(\frac{n^2}{2} + \frac{n}{2})$ + (c4 + c5) * $\frac{n^2}{2} - \frac{n}{2}$

Organizando, temos:

T(n) = $\frac{(c3+ c4+c5)}{2}$ * n^2 + (c1 + c2 + $\frac{c3}{2} - \frac{c4}{2} - \frac{c5}{2} + c6 + c7$) * n - (c2 + c6 + c7) * 1

Podemos expressar a T(n) acima como $an^2 + bn + c$, portanto temos uma função quadrática. Na notação grande-O, não é importante os fatores constantes. Dessa forma, temos que o resultado da execução do algoritmo Selection Sort é O(n^2). Diferente do insertion sort, o selection sort irá apresentar esse comportamento independentemente da ordenação prévia da entrada.