**Линейный поиск**

Лучший случай (элемент на 0 месте)

int i = 0; // c1 \* 1  
while (i < list.Count && list[i] != searchElement) // c2 \* 1  
{  
 i++;  
}  
  
return i; // c3 \* 1

Tл(n) = c1 + c2 + c3 = C = Ω(1)

Худший случай (элемента нет в массиве)

int i = 0; // c1 \* 1  
while (i < list.Count && list[i] != searchElement) // c2 \* (n + 1)  
{  
 i++;  
}  
  
return i; // c3 \* 1

Tх(n) = c1 + c2\*(n+1) + c3 = c1 + c2\*n + c2 + c3 = c2\*n + c1 + c2 + c3 = A\*n + B = O(n)

**Линейный поиск с барьером**

Лучший случай (элемент на 0 месте)

list.Add(searchElement); // c1 \* 1  
  
int i = 0; // c2 \* 1  
while (list[i] != searchElement) // c3 \* 1  
{  
 i++;  
}  
  
return i; // c4 \* 1

Tл(n) = c1 + c2 + c3 + c4 = C = Ω(1)

Худший случай (элемента нет в массиве)

list.Add(searchElement); // c1 \* 1  
  
int i = 0; // c2 \* 1  
while (list[i] != searchElement) // c3 \* (n + 1)  
{  
 i++;  
}  
  
return i; // c4 \* 1

Tх(n) = c1 + c2 + c3\*(n+1) + c4 = c1 + c2 + c3\*n + c3 + c4 = c3\*n + c1 + c2 + c3 + c4 = A\*n + B = O(n)

**Сортировка методом простого обмена**

for (var i = 0; i < list.Count - 1; i++) // c1 \* n  
{  
 for (var j = 0; j < list.Count - i - 1; j++) // c2 \* (n+(n-1)+(n-2)+…+1)  
 {  
 if (list[j] > list[j + 1]) // c3 \* ((n-1)+(n-2)+(n-3)+…+1)  
 {  
 (list[j], list[j + 1]) = (list[j + 1], list[j]);  
 }  
 }  
}  
  
return list;

T(n) =

= Ω(n2) = O(n2)

Порядок роста Θ(n2)

**Сортировка методом простого выбора**

for (var i = 0; i < list.Count - 1; i++) // c1 \* n  
{  
 var minI = i; // c2 \* (n – 1)  
   
 for (var j = i + 1; j < list.Count; j++) // c3 \* (n+(n-1)+(n-2)+…+1)  
 {  
 if (list[j] < list[minI]) // c4 \* ((n-1)+(n-2)+(n-3)+…+1)  
 {  
 minI = j;  
 }  
 }  
  
 if (minI != i) // c5 \* (n - 1)  
 {  
 (list[i], list[minI]) = (list[minI], list[i]);  
 }  
}  
  
return list;

T(n) =

= Ω(n2) = O(n2)

Порядок роста Θ(n2)