

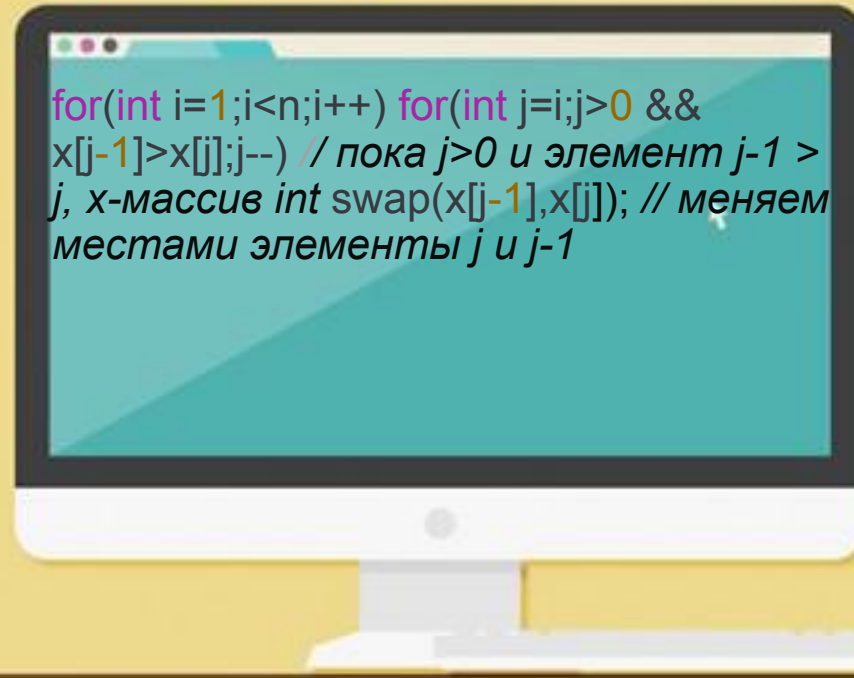
# Сортировка вставка







```
for(int i=1;i<n;i++) for(int j=i;j>0 &&  
x[j-1]>x[j];j--) // пока j>0 и элемент j-1 >  
j, x-массив int swap(x[j-1],x[j]); // меняем  
местами элементы j и j-1
```



Сортировка вставками имеет сложность  $n^2$ , количество сравнений вычисляется по формуле  $n*(n-1)$   
Для доказательства был написан следующий код:

```
void Sort(int* arr,int n){  
    int counter=0;  
    for(int i=1;i<n;i++){  
        for(int j=i; j>0 && arr[j-1]>arr[j];j--){  
            counter++;  
            int tmp=arr[j-1];  
            arr[j-1]=arr[j];  
            arr[j]=tmp;  
        }  
    }  
    cout<<counter<<endl;  
}
```

Количество перестановок для 100 элементов:

cmd. Command Prompt

```
C:\Users\ART\Google Диск\sources\sort>a  
Number of elements:100  
Number of Swaps:4950  
  
C:\Users\ART\Google Диск\sources\sort>
```

Таблица 1 — Сортировка вставками

шаг	отсортированная часть массива							тек. элемент	вставка
1	3							3	false
2	3	3						7	false
3	3	3	7					1	true
4	1	3	3	7				2	true
5	1	2	3	3	7			5	true
6	1	2	3	3	5	7		0	true
—	0	1	2	3	3	5	7	—	—

```

1 // insertion_sort.cpp: определяет точку входа для консольного приложения.
2
3 #include "stdafx.h"
4 #include <iostream>
5 #include <ctime>
6 #include <iomanip>
7 using namespace std;
8
9 void insertionSort(int *, int); // прототип функции сортировки вставками
10
11 int main(int argc, char* argv[])
12 {
13     srand(time(NULL));
14     setlocale(LC_ALL, "rus");
15     cout << "Введите размер массива: ";
16     int size_array; // длина массива
17     cin >> size_array;
18
19     int *sorted_array = new int [size_array]; // одномерный динамический массив
20     for (int counter = 0; counter < size_array; counter++)
21     {
22         sorted_array[counter] = rand() % 100; // заполняем массив случайными числами
23         cout << setw(2) << sorted_array[counter] << " "; // вывод массива на экран
24     }
25     cout << "n";
26
27     insertionSort(sorted_array, size_array); // вызов функции сортировки вставками
28
29     for (int counter = 0; counter < size_array; counter++)
30     {
31         cout << setw(2) << sorted_array[counter] << " "; // печать отсортированного массива
32     }
33     cout << "n";
34     delete [] sorted_array; // высвобождаем память
35     system("pause");
36     return 0;
37 }
38
39 void insertionSort(int *arrayPtr, int length) // сортировка вставками
40 {
41     int temp, // временная переменная для хранения значения элемента сортируемого массива
42         item; // индекс предыдущего элемента
43     for (int counter = 1; counter < length; counter++)
44     {
45         temp = arrayPtr[counter]; // инициализируем временную переменную текущим значением элемента массива
46         item = counter-1; // запоминаем индекс предыдущего элемента массива
47         while(item >= 0 && arrayPtr[item] > temp) // пока индекс не равен 0 и предыдущий элемент массива
48             {
49                 arrayPtr[item + 1] = arrayPtr[item]; // перестановка элементов массива
50                 arrayPtr[item] = temp;
51                 item--;
52             }

```

CppStudio.com

```

Введите размер массива: 20
9 83 89 40 4 81 28 86 39 68 83 89 48 25 9 15 77 87 37 40
4 9 9 15 25 28 37 39 40 40 48 68 77 81 83 83 86 87 89 89
Для продолжения нажмите любую клавишу . . .

```