**Использование стандартной библиотеки С++ <algorithm>**

В стандартной библиотеке С++ имеется поразительно много функций, работающих с классом vector, но определенных не как функции-члены класса, а как набор обобщенных алгоритмов. Вот их неполный перечень:  
алгоритмы поиска: find(), find\_if(), search(), binary\_search(), count(), count\_if();  
алгоритмы сортировки и упорядочения: sort(), partial\_sort(), merge(), partition(), rotate(), reverse(), random\_shuffle();  
алгоритмы удаления: unique(), remove();  
численные алгоритмы: accumulate(), partial\_sum(), inner\_product(), adjacent\_difference();  
алгоритмы генерации и изменения последовательности: generate(), fill(), transform(), copy(), for\_each();  
алгоритмы сравнения: equal(), min(), max().  
В число параметров этих обобщенных алгоритмов входит итераторная пара, задающая диапазон элементов вектора, к которым применяется алгоритм. Скажем, чтобы упорядочить все элементы некоторого вектора ivec, достаточно написать следующее:

sort ( ivec.begin(), ivec.end() );

Чтобы применить алгоритм sort() только к первой половине вектора, мы напишем:

sort ( ivec.begin(), ivec.begin() + ivec.size()/2 );

Роль итераторной пары может играть и пара указателей на элементы встроенного массива. Пусть, например, нам дан массив:

int ia[7] = { 10, 7, 9, 5, 3, 7, 1 };

Упорядочить весь массив можно вызовом алгоритма sort():

sort ( ia, ia+7 );

Так можно упорядочить первые четыре элемента:

sort ( ia, ia+4 );

Для использования алгоритмов в программу необходимо включить заголовочный файл

#include <algorithm>

Ниже приведен пример программы, использующей разнообразные алгоритмы в применении к объекту типа vector:

#include <vector>

#include <algorithm>

#include <iostream>

int ia[ 10 ] = {  
 51, 23, 7, 88, 41, 98, 12, 103, 37, 6  
};

int main()  
{  
 vector< int > vec( ia, ia+10 );  
 vector<int>::iterator it = vec.begin(), end\_it = vec.end();

cout << "Начальный массив: ";  
  
 for ( ; it != end\_it; ++ it ) cout << \*it << ' ';  
 cout << "\n";

// сортировка массива  
 sort( vec.begin(), vec.end() );

cout << "упорядоченный массив: ";  
 it = vec.begin(); end\_it = vec.end();  
 for ( ; it != end\_it; ++ it ) cout << \*it << ' ';  
 cout << "\n\n";

int search\_value;  
 cout << "Введите значение для поиска: ";  
 cin >> search\_value;

// поиск элемента  
 vector<int>::iterator found;  
 found = find( vec.begin(), vec.end(), search\_value );

if ( found != vec.end() )  
 cout << "значение найдено!\n\n";  
 else cout << "значение найдено!\n\n";  
 // инвертирование массива  
 reverse( vec.begin(), vec.end() );

cout << "инвертированный массив: ";  
 it = vec.begin(); end\_it = vec.end();

for ( ; it != end\_it; ++ it ) cout << \*it << ' ';  
 cout << endl;

}