transform()

```
template <class InIter, class OutIter, class Func>
        OutIter transform(InIter start, InIter end,
                                OutIter result, Func unaryfunc);
template <class InIter1, class InIter2, class OutIter, class Func>
        OutIter transform(InIter1 start1, InIter1 end1, InIter2 start2,
                                OutIter result, Func binaryfunc);
```

Описание

Применяет функцию к диапазону элементов и сохраняет результат в последовательности

Алгоритм transform() применяет функцию к диапазону элементов и сохраняет результат в последовательности, заданной параметром result. В первой форме диапазон задается параметрами start и end. Применяемая функция задается параметром unaryfunc. Она принимает значение элемента в качестве параметра и должна возвратить преобразованное значение.

Во второй форме алгоритма преобразование применяется с использованием бинарной функции, которая принимает значение элемента из последовательности, предназначенного для преобразования, в качестве первого параметра и элемент из второй последовательности в качестве второго параметра.

Обе версии возвращают итератор, указывающий на конец результирующей последовательности.

Совет программисту

Одним из самых интересных алгоритмов является алгоритм transform(), поскольку он модифицирует каждый элемент из заданного диапазона в соответствии с предоставленной вами функцией. Например, в следующей программе используется простая функция преобразования xform(), предназначенная для возведения в квадрат содержимого списка. Обратите внимание, что результирующая последовательность сохраняется в том же списке, в котором содержится и исходная.

```
// Пример использования алгоритма преобразования transform().
#include <iostream>
#include <list>
#include <algorithm>
using namespace std;

// Простая функция преобразования,
int xform(int i) {
        return i*i; // квадрат исходного значения
}

int main()
```

```
{
        list<int> xl;

        int i;

        //Помещаем значения в список.
        for(i=0; i<10; i++) xl.push_back(i);

        cout << "Исходное содержимое списка xl: ";
        list<int>::iterator p = xl.begin();
        while(p != xl.end()){
                cout << *p << " ";
                p++;
        }

        cout << endl;

        // Преобразуем список xl.
        p = transform(xl.begin() , xl.end(), xl.begin(), xform);

        cout << "Преобразованное содержимое списка xl:";

        p = xl.begin();
        while(p!= xl.end()) {
                cout << *p << " ";
                p++;
        }

        return 0;
}
```

Ниже представлен результат работы этой программы.

```
Исходной содержимое списка xl:          0 1 2 3 4 5 6 7 8 9
Преобразованное содержимое списка xl:   0 1 4 9 16 25 36 49 64 81
```

Как видите, каждый элемент в списке xl возведен в квадрат.