C++
Information
Tutorials
Reference
Articles
Forum

Reference
C library:
Containers:
Input/Output:
Multi-threading:
Other:
<algorithm>
<bitset>
<chrono>
<codecvt>
<complex>
<exception>
<functional>
<initializer_list>
<iterator>
<limits>
<locale>
<memory>
<new>
<numeric>
<random>
<ratio>
<regex>
<stdexcept>
<string>
<system_error>
<tuple>
<typeindex>
<typeinfo>
<type_traits>
<utility>
<valarray>

<algorithm>
adjacent_find
all_of
any_of
binary_search
copy
copy_backward
copy_if
copy_n
count
count_if
equal
equal_range
fill
fill_n
find
find_end
find_first_of
find_if
find_if_not
for_each
generate
generate_n
includes
inplace_merge
is_heap
is_heap_until
is_partitioned
is_permutation
is_sorted
is_sorted_until
iter_swap
lexicographical_compare
lower_bound
make_heap
max
max_element
merge
min
minmax
minmax_element
min_element
mismatch
move
move_backward

function template

std::is_permutation

<algorithm>

```
            template <class ForwardIterator1, class ForwardIterator2>
equality (1)    bool is_permutation (ForwardIterator1 first1, ForwardIterator1 last1,
                                     ForwardIterator2 first2);
            template <class ForwardIterator1, class ForwardIterator2, class BinaryPredicate>
predicate (2)   bool is_permutation (ForwardIterator1 first1, ForwardIterator1 last1,
                                     ForwardIterator2 first2, BinaryPredicate pred);
```

Test whether range is permutation of another

Compares the elements in the range `[first1,last1)` with those in the range beginning at *first2*, and returns `true` if all of the elements in both ranges match, even in a different order.

The elements are compared using `operator==` (or *pred*, in version *(2)*).

The behavior of this function template is equivalent to:

```
template <class InputIterator1, class InputIterator2>
  bool is_permutation (InputIterator1 first1, InputIterator1 last1,
                       InputIterator2 first2)
{
  std::tie (first1,first2) = std::mismatch (first1,last1,first2);
  if (first1==last1) return true;
  InputIterator2 last2 = first2; std::advance (last2,std::distance(first1,last1));
  for (InputIterator1 it1=first1; it1!=last1; ++it1) {
    if (std::find(first1,it1,*it1)==it1) {
      auto n = std::count (first2,last2,*it1);
      if (n==0 || std::count (it1,last1,*it1)!=n) return false;
    }
  }
  return true;
}
```

Parameters

`first1, last1`
Input iterators to the initial and final positions of the first sequence. The range used is `[first1,last1)`, which contains all the elements between *first1* and *last1*, including the element pointed by *first1* but not the element pointed by *last1*.

`first2`
Input iterator to the initial position of the second sequence.
The function considers as many elements of this sequence as those in the range `[first1,last1)`.
If this sequence is shorter, it causes *undefined behavior*.

`pred`
Binary function that accepts two elements as argument (one of each of the two sequences, in the same order), and returns a value convertible to `bool`. The value returned indicates whether the elements are considered to match in the context of this function.
The function shall not modify any of its arguments.
This can either be a function pointer or a function object.

`InputIterator1` and `InputIterator2` shall point to the same type.

Return value

`true` if all the elements in the range `[first1,last1)` compare equal to those of the range starting at *first2* in any order, and `false` otherwise.

Example

```
// is_permutation example
#include <iostream>     // std::cout
#include <algorithm>    // std::is_permutation
#include <array>        // std::array

int main () {
  std::array<int,5> foo = {1,2,3,4,5};
  std::array<int,5> bar = {3,1,4,5,2};

  if ( std::is_permutation (foo.begin(), foo.end(), bar.begin()) )
    std::cout << "foo and bar contain the same elements.\n";

  return 0;
}
```

Output:

```
foo and bar contain the same elements.
```

Complexity

If both sequence are equal (with the elements in the same order), linear in the distance between *first1* and *last1*.
Otherwise, up to quadratic: Performs at most N^2 element comparisons until the result is determined (where *N* is the distance between *first1* and *last1*).

Data races

Some (or all) of the objects in both ranges are accessed (possibly multiple times each).

Exceptions

Throws if any of the element comparisons (or *pred*) throws, or if any of the operations on iterators throws.
Note that invalid parameters cause *undefined behavior*.

See also

equal	Test whether the elements in two ranges are equal (function template)
mismatch	Return first position where two ranges differ (function template)
next_permutation	Transform range to next permutation (function template)
prev_permutation	Transform range to previous permutation (function template)