**一. 变动性算法**

|  |  |
| --- | --- |
| copy() | 从第一个元素开始，复制某段区间 |
| copy \_backward() | 从最后一个元素开始，复制某段区间 |
| transform() | 变动(并复制)元素，将两个区间的元素合并 |
| merge() | 合并两个区间 |
| swap\_ranges() | 交换两区间内的元素 |
| fill() | 以给定值替换每一个元素 |
| fill\_n() | 以给定值替换n个元素 |
| generate() | 以某项操作的结果替换每一个元素 |
| generate\_n() | 以某项操作的结果替换n个元素 |
| replace() | 将具有某特定值的元素替换为另一个值 |
| replace()\_if() | 将符合某准则的元素替换为另一个值 |
| replace\_copy() | 复制整个区间，同时并将具有某特定值的元素替换为另一个值 |
| replace\_copy\_if() | 复制整个区间，同时并将符合某种准则的元素替换为另一个值 |

**二. 复制元素**

**[cpp]** [view plaincopyprint?](http://blog.csdn.net/lwbeyond/article/details/7329032)

1. //将源区间[sourceBeg, sourceEnd]中所有元素复制到以destBeg为起点的目标区间去
2. //返回目标区间内最后一个被复制元素的下一个位置
3. //destBeg或destEnd不可处于[sourceBeg, sourceEnd]区间内
4. OutputIterator
5. copy (InputIterator sourceBeg,
6. InputIterator sourceEnd,
7. OutputIterator destBeg)
8. //将源区间[sourceBeg, sourceEnd]中所有元素复制到以destEnd为终点的目标区间去
9. BidirectionalIteratorl
10. copy\_backward (BidirectionalIterator1 sourceBeg,
11. BidirectionalIterator1 source End,
12. BidirectionalIterator2 destEnd)

//将源区间[sourceBeg, sourceEnd]中所有元素复制到以destBeg为起点的目标区间去

//返回目标区间内最后一个被复制元素的下一个位置

//destBeg或destEnd不可处于[sourceBeg, sourceEnd]区间内

OutputIterator

copy (InputIterator sourceBeg,

InputIterator sourceEnd,

OutputIterator destBeg)

//将源区间[sourceBeg, sourceEnd]中所有元素复制到以destEnd为终点的目标区间去

BidirectionalIteratorl

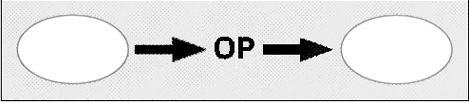
copy\_backward (BidirectionalIterator1 sourceBeg,

BidirectionalIterator1 source End,

BidirectionalIterator2 destEnd)

**三. 转换和结合元素**

**1. 转换元素**



**[cpp]** [view plaincopyprint?](http://blog.csdn.net/lwbeyond/article/details/7329032)

1. //针对源区间[sourceBeg, sourceEnd)中的每一个元素调用op(elem)
2. //并将结果写到以destBeg起始的目标区间内
3. //返回目标区间内的"最后一个被转换元素"的下一位置
4. //sourceBeg与destBeg可以相同
5. OutputIterator
6. transform (InputIterator sourceBeg, InputIterator sourceEnd,
7. Output Iterator destEeg,
8. UnaryFunc op)

//针对源区间[sourceBeg, sourceEnd)中的每一个元素调用op(elem)

//并将结果写到以destBeg起始的目标区间内

//返回目标区间内的"最后一个被转换元素"的下一位置

//sourceBeg与destBeg可以相同

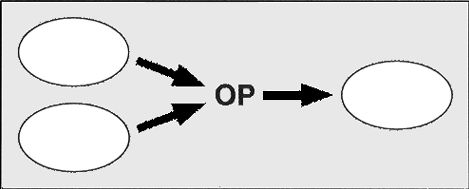
OutputIterator

transform (InputIterator sourceBeg, InputIterator sourceEnd,

Output Iterator destEeg,

UnaryFunc op)

**2. 将两序列的元素加以结合**



**[cpp]** [view plaincopyprint?](http://blog.csdn.net/lwbeyond/article/details/7329032)

1. //针对第一个源区间[source1Beg, source1End)以及从source2Beg开始的第二个源区间元素
2. //调用op(source1Elem, source2Elem),并将结果写入destBeg起始的目标区间内
3. OutputIterator
4. transform (InputIterator1 source1Beg, InputIterator1 source1End,
5. InputIterator2 source2Beg,
6. OutputIterator destBeg,
7. BinaryFunc op)

//针对第一个源区间[source1Beg, source1End)以及从source2Beg开始的第二个源区间元素

//调用op(source1Elem, source2Elem),并将结果写入destBeg起始的目标区间内

OutputIterator

transform (InputIterator1 source1Beg, InputIterator1 source1End,

InputIterator2 source2Beg,

OutputIterator destBeg,

BinaryFunc op)

**四. 互换元素内容**

**[cpp]** [view plaincopyprint?](http://blog.csdn.net/lwbeyond/article/details/7329032)

1. //将区间[beg1, end1)内的元素和从beg2开始的区间内的对应元素互换
2. //返回第二个区间中最后一个被交换元素的下一位置
3. ForwardIterator2
4. swap\_ranges (ForwardIterator1 beg1, ForwardIterator1 end1,
5. ForwardIterator2 beg2)

//将区间[beg1, end1)内的元素和从beg2开始的区间内的对应元素互换

//返回第二个区间中最后一个被交换元素的下一位置

ForwardIterator2

swap\_ranges (ForwardIterator1 beg1, ForwardIterator1 end1,

ForwardIterator2 beg2)

**五. 赋于新值**

**1. 赋于完全相同的数值**

**[cpp]** [view plaincopyprint?](http://blog.csdn.net/lwbeyond/article/details/7329032)

1. //将区间[beg, end)内的每一个元素都赋于新值newValue
2. void
3. fill (ForwardIterator beg, ForwardIterator end,
4. const T& newValue)
5. //将从beg开始的前num个元素赋于新值newValue
6. void
7. fill\_n (OutputIterator beg,
8. Size num,
9. const T& newValue)

//将区间[beg, end)内的每一个元素都赋于新值newValue

void

fill (ForwardIterator beg, ForwardIterator end,

const T& newValue)

//将从beg开始的前num个元素赋于新值newValue

void

fill\_n (OutputIterator beg,

Size num,

const T& newValue)

**2. 赋于新产生的数值**

**[cpp]** [view plaincopyprint?](http://blog.csdn.net/lwbeyond/article/details/7329032)

1. //op()产生新值，并赋值给区间[beg, end)内的每个元素
2. void
3. generate (ForwardIterator beg, ForwardIterator end,
4. Func op)
5. //op()产生新值，并赋值给以beg起始的区间内的前num个元素
6. void
7. generate\_n (OutputIterator beg,
8. Size num,
9. Func op)

//op()产生新值，并赋值给区间[beg, end)内的每个元素

void

generate (ForwardIterator beg, ForwardIterator end,

Func op)

//op()产生新值，并赋值给以beg起始的区间内的前num个元素

void

generate\_n (OutputIterator beg,

Size num,

Func op)

**六. 替换元素**

**1. 替换序列内的元素**

**[cpp]** [view plaincopyprint?](http://blog.csdn.net/lwbeyond/article/details/7329032)

1. //将区间[beg, end)内每一个与 oldValue 相等的元素替换为 newValue
2. void
3. replace (Forwardlterator beg, ForwardIterator end,
4. const T& oldValue,
5. const T& newValue)
6. //将区间[beg, end)内每一个令op(elem)为true的元素替换为 newValue
7. void
8. replace\_if (ForwardIterator beg, ForwardIterator end,
9. UnaryPredicate op,
10. const T& newValue)

//将区间[beg, end)内每一个与 oldValue 相等的元素替换为 newValue

void

replace (Forwardlterator beg, ForwardIterator end,

const T& oldValue,

const T& newValue)

//将区间[beg, end)内每一个令op(elem)为true的元素替换为 newValue

void

replace\_if (ForwardIterator beg, ForwardIterator end,

UnaryPredicate op,

const T& newValue)

**2. 复制并替换元素**

**[cpp]** [view plaincopyprint?](http://blog.csdn.net/lwbeyond/article/details/7329032)

1. //将源区间[beg, end)中的元素复制到以 destBeg 为起点的目标区间去
2. //同时将其中与 oldValue 相等的所有元素替换为 newValue
3. OutputIterator
4. replace\_copy (InputIterator sourceBeg, InputIterator sourceEnd,
5. OutputIterator destBeg,
6. const T& oldValue, const T& newValue)
7. //将源区间[beg, end)中的元素复制到以 destBeg 为起点的目标区间去
8. //同时将其中使p(elem)为true的所有元素替换为 newValue
9. OutputIterator
10. replace\_copy\_if (InputIterator sourceBeg, InputIterator sourceEnd,
11. OutputIterator destBeg,
12. UnaryPredicate op, const T& newValue)