

# List Operations

## sort()

- The generic sort() algorithm requires random access to container elements
- This is not supported for list and forward\_list

```
sort(l.begin(), l.end());
```

// Does not compile

- Instead, we have to use the member function

```
l.sort();
```

// OK

## merge()

- merge() will remove elements from the argument list and merge them into the "this" list
- The list will be sorted in ascending order, provided both lists were sorted before the operation

```
list<int> l2{1, 7, 12, 24};
```

```
list<int> l3{9, 3, 14};
```

```
l2.sort();           // Make sure lists are sorted
```

```
l3.sort();
```

```
l2.merge(l3);        // l2 now contains 1, 3, 7, 9, 12, 14, 24 and l3 is empty
```

## splice()

- splice() moves elements from another list into the "this" list just before a given iterator

```
list<int> l4{1, 12, 6, 24};
```

```
list<int> l5{9, 3, 14};
```

```
auto p = l4.begin();
```

```
++p;
```

```
// p is an iterator to the second element of l4 (with value 12)
```

```
l4.splice(p, l5);
```

```
// l4 now contains 1, 9, 3, 14, 12, 6, 24 and l5 is empty
```

- There are also versions which move a single element or a range of elements from the argument list