List Operations Solutions

List Operations

- Do the C++ Standard implementations of linked lists support random access?
  - No
  - To access a list element, we have to iterate through every element until we reach the one we require
  - The time taken to access an element will depend on how far it is from the starting point

List Operations

- Give some examples of generic standard algorithm functions that have been re-implemented as member functions of std::list
  - sort(), remove()
- Write a simple program that uses some of these member functions

List Member Operations

- Are there any advantages to using these member function versions?
    - The member functions can be optimized for std::list, or take advantage of some of its features
    - For example, remove() can be implemented as a couple of pointer operations, instead of moving the element to the end of the container

Element Moving Operations

- Describe the merge() and splice() member functions of std::list
  - merge() moves the elements from its argument into the list
  - If both lists were sorted, the result will be sorted
  - splice() moves the elements from its argument into the list, just before a given iterator
- Write a simple program that uses these functions
- Change your program so that it uses std::forward_list. Do you need to make any other changes?
  - std::forward_list has splice_after() instead of splice()
  - This splices the elements in after the iterator, instead of before it