CSCI-1200 Data Structures — Fall 2020
Lab 5 — Reversing Data Many Different Ways: STL Vectors vs. STL Lists

Extra Credit Checkpoint 0: Big O Notation

estimate: 20 minutes

Checkpoint 0, a team exercise, will be available at the start of Wednesday's lab.

https://submitty.cs.rpi.edu/courses/f20/csci1200/course_materials

Checkpoint 1: Reverse with STL Vector Swaps

estimate: 10-20 minutes

Download this starter code:
http://www.cs.rpi.edu/academics/courses/fall20/csci1200/labs/05_lists_iterators/checkpoint1.cpp

Complete the function `reverse` that reverses the contents of an STL `vector` of integers. For example, if the contents of the vector are in increasing order before the call to `reverse_vector`, then they will be in decreasing order afterwards. For this checkpoint, use **indexing/subscripting/`[]`** on the vector, not iterators (or pointers). *You may not use a second vector or array or list.*

The trick is to step through the vector one location at a time, swapping values between the first half of the vector and the second half. As examples, the value at location 0 and the value at location `size()-1` must be swapped, and the value at location 1 and the value at location `size()-2` must be swapped.

Make sure your code works with even and odd length vectors. Also add a few more tests to the main function to make sure your code will work for the special cases of an empty vector and vectors of size 1 and 2.

Checkpoint 2: Reverse with STL List Swaps

estimate: 15-30 minutes

Copy your code from Checkpoint 1 to a new file. Then, convert this code to use STL Lists instead of STL vectors. Start by replacing '`vector`' with '`list`' everywhere. And you'll need to replace your subscripting with iterators. You may want to use a straightforward concept we did not discuss in lecture: a reverse iterator. A reverse iterator is designed to step through a list from the back to the front. An example will make the main properties clear:

```
std::list<int> a;
unsigned int i;
for ( i=1; i<10; ++i ) a.push_back( i*i );

std::list<int>::reverse_iterator ri;
for( ri = a.rbegin(); ri != a.rend(); ++ri )
  cout << *ri << endl;
```

This code will print out the values $81, 64, 49, \ldots, 1$, in order, on separate lines. Observe the type for the reverse iterator, the use of the functions `rbegin` and `rend` to provide iterators that delimit the bounds on the reverse iterator, and the use of the `++` operator to take one step backwards through the list. It is very important to realize that `rbegin` and `end` are NOT the same thing! One of the challenges here will be determining when to stop (when you've reached the halfway point in the list). You may use an integer counter variable to help you do this.

For this checkpoint you should not use `erase`, or `insert`, or the `push` or `pop` functions.

Note, you'll probably need to add the keyword `typename` in front of your templated iterator types to unconfuse the compiler.

```
typename std::list<T>::iterator itr = data.begin();
```

Write two versions of the function, one using the reverse iterator, and one using only the forward iterator.

To complete this checkpoint, show a TA your debugged functions to reverse STL vectors and STL lists by element swapping. Be sure to ask your TA/mentors any questions you have about regular vs. reverse iterators for lists and vectors.

Checkpoint 3

estimate: 30-40 minutes

Checkpoint 3, a team exercise, will be available at the start of Wednesday's lab.

https://submitty.cs.rpi.edu/courses/f20/csci1200/course_materials