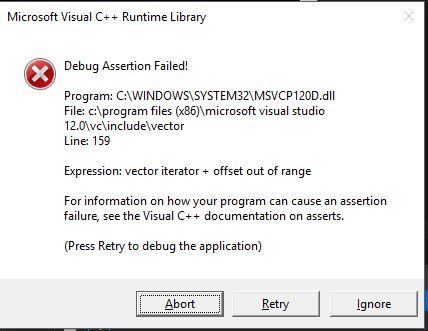
Within the magicSort function, there only exists one line of output to print progress. I added a cin.ignore() line to the end of the if statement to better watch what was happening. By adding this line of code and seeing each iterations of the for loop inside magicSort, we can see that there is an implementation of the bubble sort algorithm happening. Each successive iteration raises the current entry up to its most desirable location.

While our “Freaky” line 20 should not appear to work at first glance, it does indeed work, but only when compiled through the command line. I received the following error when the code was compiled through MSVC.

There were some libraries that I needed to include in order to run the code on my g++ command line as well, including #include <cstdlib>, #include <cstdio>. Once that was completed, the code ran without the out of range error. Something that surprised me was that while the g++ compiler treated the iterator as random-acess iterator (which supports + and -) and can accept an offset, causing the function to access the proper element of it’s base. That is, accoriding to cplusplus.com (<http://www.cplusplus.com/reference/iterator/reverse_iterator/base/>) the base iterator then becomes the element next to the one the reverse\_iterator is pointing at (which is always offset by -1). This forces our iterator to work as a bidirectional iterator without being declared as one.

If we look at the fact that bubble sort which runs in O(n2) time at its worst case, then it would be safe rational to state that this algorithm also runs in at least O(n2). If this is nested inside the for statement, which would give us the appearance of O(n), then our algorithm runs in O(n\*n2) or O(n3).