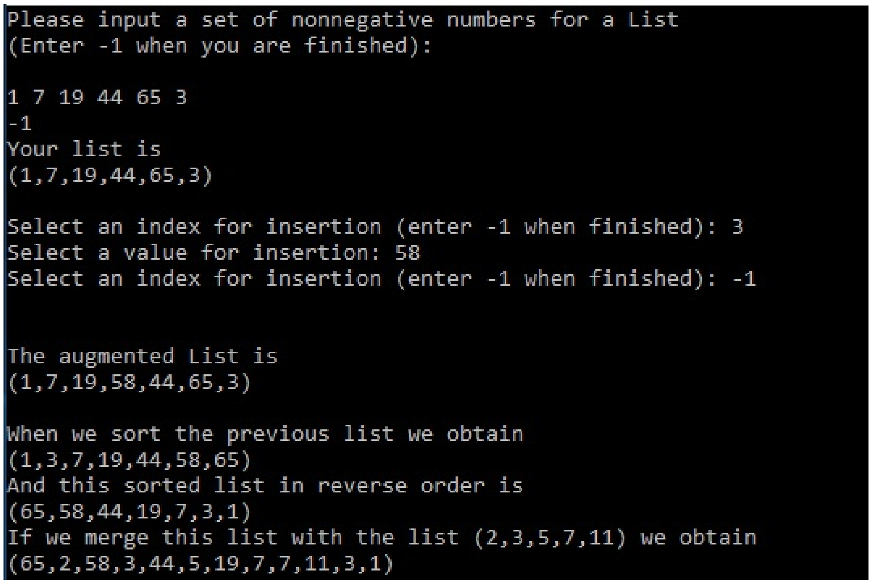
Problem 1:

Using good coding practices, design your own Linked List of `int` types. It will consist of classes `List`, `Node`, and `Iterator` granting friendship and defining functions as needed. Write constructors, destructors, and all necessary member functions such as `insert`, `erase`, increment and decrement operators, `operator*` to dereference, as well as `operator==` and `operator!=` to check whether two iterators are pointing to the same element. You will then use this list to write the following member functions:

(1) `List::reverse` to reverse your nodes
(2) `List::push_front` to add a value to the beginning of a list
(3) `List::sort` to sort the elements of a linked list (without copying them into a vector or another data type)
(4) `List::merge` which accepts another `List` object and merges the two lists into one, alternating elements from each list such that merging 1 7 12 and 8 3 11 2 2 1 yields the list 1 8 7 3 12 11 2 2 1.

Write a main function to test your list such that it follows the output shown in Figure 1.



```
Please input a set of nonnegative numbers for a List
(Enter -1 when you are finished):

1 7 19 44 65 3
-1
Your list is
(1,7,19,44,65,3)

Select an index for insertion (enter -1 when finished): 3
Select a value for insertion: 58
Select an index for insertion (enter -1 when finished): -1


The augmented List is
(1,7,19,58,44,65,3)

When we sort the previous list we obtain
(1,3,7,19,44,58,65)
And this sorted list in reverse order is
(65,58,44,19,7,3,1)
If we merge this list with the list (2,3,5,7,11) we obtain
(65,2,58,3,44,5,19,7,7,11,3,1)
```

Figure 1: sample output.

Good Coding Practices:

- think about cross-platform. Don't use Windows or Mac only commands. For example, pause == cin.get() twice, write many \n vs. system(clear) or system('cls).
- passing objects by reference & or const & when possible
- using field initializer list when possible in all constructors

Instructions for submission:

- Name your files exactly `hw5.cpp`, `List.h`, `List.cpp`, `Node.h`, `Node.cpp`, `Iterator.h`, and `Iterator.cpp`.
- You may not use `#include "stdafx.h"`.
- Add code description in the comment at the beginning of the file.
  A sample description may look like:

```
/*
    PIC 10B 2A, Homework 1
    Purpose: Tic-tac-toe game
    Author: Hanqin Cai
    Date: 10/10/2019
*/
```

- Submit your header files and source codes to BruinLearn in separate files. Only .h and .cpp files should be uploaded.