Lab 03 - Linked Lists

Instructions:

- **The lab requires completing a few tasks.**
- **Your submissions must be submitted to the GitHub repository in your Lab03 directory.**
- **You can only use the libraries *iostream*, *string*, *sstream*, *stdexcept*, and the header file `"Util.h"`.**
- **Cheating of any kind is prohibited and will not be tolerated.**
- **Violating or failing to follow any of the rules will result in an automatic zero (0) for the lab.**

TO ACKNOWLEDGE THAT YOU HAVE READ AND UNDERSTOOD THE INSTRUCTIONS ABOVE, AT THE BEGINNING OF YOUR SUBMISSION(S), ADD A COMMENT THAT CONSISTS OF YOUR NAME AND THE DATE. PROVIDE THE NAME OF YOUR TEAMMATE IF YOU ARE WORKING IN A GROUP.

Grading

Task	Maximum Points	Points Earned
1	2.5	
2	2.5	
Total	5.0	

Task 1

Create a header file named `"InsertMethod.h"` **that define the void function** `InsertList()` **whose header is**

```
template <typename T>
void InsertList(Node<T>* lhs,Node<T>* rhs,unsigned int idx)
```

It inserts the list rhs **between the node of** lhs **whose index is** idx **and its next link if neither** lhs **nor** rhs **is an empty list and** idx **is a valid index of** lhs **[i.e.** $0 \leq idx < n$ **where** n **is the size of** lhs**]; otherwise, it does nothing.**

Hint: Having a reference to the tail of rhs **will be necessary.**

Example:
After the caller `InsertList(a,b,1)` **where** $a = [1, 2, 3, 4]$ **and** $b = [7, 8, 5]$**,** a **will be** $[1, 2, 7, 8, 5, 3, 4]$**.**

Task 2

Create a header file named `"RemoveMethod.h"` **that define the void function** `RemovetList()` **whose header is**

```
template <typename T>
void RemoveList(Node<T>*& rt,unsigned int i,unsigned int j)
```

It removes nodes from rt **that are inclusively between the nodes of** rt **whose indices are** i **and** j **if** rt **is not an empty list and both** i **and** j **are valid indices of** rt**; otherwise, it does nothing. You must deallocate the removed portion of the list.**

Hint: Order the indices first and consider what happens if the head is being removed.

Example:
After the caller `RemoveList(a,3,1)` **where** $a = [0, 1, 2, 3, 4, 5, 6, 7, 8, 9]$**,** a **will be** $[0, 4, 5, 6, 7, 8, 9]$**.**

Extra Credit

- **Define an Boolean function named** `NotContained()` **whose header is**

```
bool NotContained(const Node<int>* a,const Node<int>* b)
```

It returns true if either every element in a **is not in** b **or exactly one of the lists is an empty list; otherwise, it returns false.**

- **Define a Boolean function named** `Same()` **whose header is**

```
bool Same(const Node<int>* a,const Node<int>* b)
```

It returns true either if a **and** b **are both empty lists or if** a **and** b **share all the same values which are not necessarily in the same order; otherwise, it returns false.**