Background:
In previous assignments, you have stored data in arrays. An array is useful when the size of your input set is known at the beginning of your application. After pointers were introduced, it became possible to declare arrays at runtime—i.e., prompt the user for the size of an array.

To add or remove elements from an array, the array must be resized. After the array has been resized, the elements must be rearranged. In an array, this is an expensive operation. To remove one element, all subsequent elements must be shifted.

A linked list is a collection of nodes wherein each node points to the next node in the list. The last node in a linked list points to NULL. Linked lists allow nodes to be added and removed. When manipulating a linked list, all operations involve updating pointers.

Description:
In this assignment, you will create a linked list. You have been provided with three structs. The LinkedList contains a head pointer, tail pointer, and num_nodes counter. The pointers point to the beginning and end of the linked list respectively. The num_nodes is simply a counter for the number of nodes in the list.

The Node struct contains a pointer to data. Data refers to the information contained within the node. The next pointer refers to the next node in the linked list.

Instructions:
1. Preliminary Instructions
   - Read this prompt completely.
   - Read the provided template—including all comments.
2. Structs
   - Familiarize yourself with the provided structs.
   - The LinkedList and Node structs are described in the Description section.
   - The Record struct contains:
     - X and y attributes to store position.
     - Depth to store ocean depth data.
3. Function: main
   - Complete the provided while loop.
     - Read the x,y and depth values into the to_add pointer
     - Make no further modifications to the while loop.
   - Complete the second while loop
     - Modify the body of the loop to find the minimum depth reading.
     - Modify the body of the loop to find the maximum depth reading.
     - Hardcoding these values will result in a deduction of 16 points.
4. **Function**: main (continued)
   - Complete the third while loop
     - Output each node in the Linked list.
     - You output must match the provided sample output.
   - Output the maximum depth.
   - Output the minimum depth.

5. **Function**: addNode
   - The function body has been provided.
   - The code to add the first node to the linked list has been provided.
   - Complete the else block.
     - Add the new node to the linked list
     - Update the tail pointer.
Sample Output:

**A9-solution.cpp**

Ocean Depth Records

Enter Reading #1: 0 0 -10
Would you like to add another Record? (Y/N): Y
Enter Reading #2: 0 10 -20
Would you like to add another Record? (Y/N): Y
Enter Reading #3: 10 10 -30
Would you like to add another Record? (Y/N): Y
Enter Reading #4: 10 20 -40
Would you like to add another Record? (Y/N): Y
Enter Reading #5: 10 30 -5
Would you like to add another Record? (Y/N): N

Summary: Depth Readings

Record 1
Position:  ( 0.00 , 0.00 )
Depth:  -10.00

Record 2
Position:  ( 0.00 , 10.00 )
Depth:  -20.00

Record 3
Position:  ( 10.00 , 10.00 )
Depth:  -30.00

Record 4
Position:  ( 10.00 , 20.00 )
Depth:  -40.00

Record 5
Position:  ( 10.00 , 30.00 )
Depth:  -5.00

Summary: Depth Reading Extrema

Lowest Depth:  -40.00
Highest Depth:  -10.00