1. For the first problem, I first created a while loop that keeps creating nodes until the user pressed CTRL+D. Each time there is an integer inputted in, I allocate memory for a new struct node and then add it on to the tail variable which holds the last node inserted into my linked list. After this, I first display each node starting from the head using a temp variable called curr. Then I create a new linked list from the previous head, and then display each of those again. To reverse the linked list, I use a temp variable called dummy which acts as the head of the reversed linked list. I again use the curr variable to traverse the original linked list, but now I add the current node behind the dummy node and then set the current node to be the dummy node.
2. For this problem, I again first read in the linked list from the user util CTRL+D is pressed and then I pass in the address to the head node to my bubblesort function which takes as input a pointer to a pointer to a struct node. This way I can easily swap the nodes' locations because I would already have the address which holds the location of the node itself. After this, I created logic in my bubblesort function to keep sorting until we reach a point where the last swapped element is seen.