**Enqueue:**

First, memory is allocated for the new node and initialized to a given value. Then, depending on whether it's the first element of the queue, proper adjustments will be made.

**Dequeue:**

First, the program determines whether the queue is empty or not. If it is empty, -1 is returned to show that an error has occurred. If it’s not empty, the node pointed to by head is updated to the second node in the queue, and the first node is deleted. After that, “prev” is updated to NULL.

**printHeadToTail:**

The program originally initializes the loop variable “i” as 1 and curr to point to the same node as head. Then, in a while loop that continues as long as curr != NULL, the elements of each node are printed out, and lines are changed after printing every 20 elements. The values of “i” and curr are also updated in the loop.

**getSize:**

The program originally initializes cnt  as 0 and curr to point to the same node as head. Then, in a while loop that continues as long as curr != NULL, cnt is incremented by one after each iteration, and curr is also updated to point to the next node in the queue.

**Trial(in main program):**

A random value for the number of elements that should be enqueued(in) in the trial is first generated. Then, the program generates another value(out), which is the number of elements that should be dequeued, and this number is smaller than the previously generated value. After that, the program enqueues random values between 1 and 99 into the queue for “in” times ,  and dequeues elements from the queue for “out” times. After each enqueue or dequeue operation, **getSize** is used to calculate the number of elements in the queue.