Task 2 Report – Treatment Queue

In Task 2, a treatment request system is implemented using a Queue data structure. The main goal of this task is to understand how FIFO (First In First Out) logic works in real life scenarios such as hospital treatment order. Each treatment request is added to the queue when a patient wants treatment and processed in the same order.

The queue is implemented by using a linked list. The enqueue operation adds a new treatment request to the end of the queue, and the dequeue operation removes the first request from the front. Both enqueue and dequeue operations work in **O(1)** time because no shifting is needed and only pointer updates are done. The printQueue method runs in **O(n)** time since it prints all elements in the queue.

Queue is suitable for managing treatment requests because patients should be treated in the order they arrive. This ensures fairness and prevents skipping patients. If a Stack was used instead of a Queue, the last patient who arrived would be treated first. This behavior is not correct for a hospital system because it can cause earlier patients to wait longer unfairly.

In terms of time complexity, Stack and Queue operations are similar and both have **O(1)** push/pop or enqueue/dequeue operations. However, their behavior is very different. Queue preserves arrival order while Stack reverses it.

Overall, Queue is the correct data structure for treatment requests because it matches the real-world process of hospital treatment scheduling.