

## ❖ Circular Queue:

- Explanation- The problem requires designing a circular queue with a fixed size that efficiently supports `enqueue`, `dequeue`, `isFull`, and `isEmpty` operations. The circular structure allows for efficient use of space by reusing empty slots created by dequeued elements, ensuring constant time complexity for all operations. This implementation is useful for scenarios requiring a limited-size buffer, such as scheduling tasks or handling resources in a constrained environment.
- Time Complexity- Enqueue:  $O(1)$   
Dequeue:  $O(1)$
- Space Complexity-  $O(n)$
- Flowchart-

[Start]

|

v

[Input Capacity]

|

v

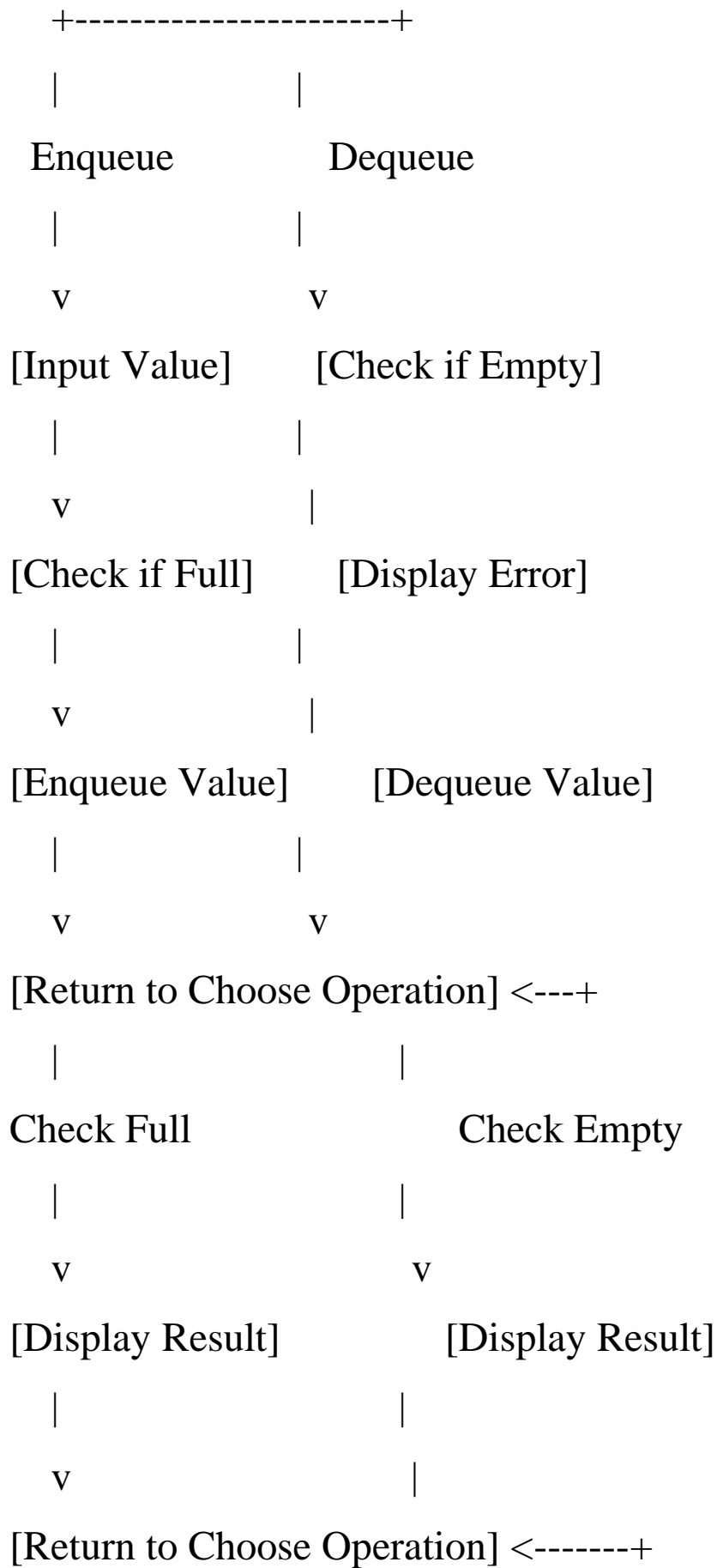
[Initialize Queue]

|

v

[Choose Operation]

|



|

Exit

|

v

[End]