**Code: -**

class LinearQueue:

    def \_\_init\_\_(self, capacity):

*self*.capacity = capacity  *# Size of the queue*

*self*.queue = [None] \* capacity  *# Array to hold the queue elements*

*self*.front = -1  *# Front points to the first element of the queue*

*self*.rear = -1  *# Rear points to the last element of the queue*

    def is\_empty(self):

        return *self*.front == -1  *# Queue is empty if front is -1*

    def is\_full(self):

        return *self*.rear == *self*.capacity - 1  *# Queue is full if rear is at capacity - 1*

    def enqueue(self, value):

        if *self*.is\_full():

            print("Queue is full!")

            return

        if *self*.is\_empty():

*self*.front = 0  *# When the first element is inserted*

*self*.rear += 1

*self*.queue[*self*.rear] = value  *# Insert the value at the rear*

    def dequeue(self):

        if *self*.is\_empty():

            print("Queue is empty!")

            return None

        dequeued\_value = *self*.queue[*self*.front]

        if *self*.front == *self*.rear:  *# If the front is the last element, reset the queue*

*self*.front = *self*.rear = -1

        else:

*self*.front += 1  *# Move front to the next element*

        return dequeued\_value

    def peek(self):

        if *self*.is\_empty():

            print("Queue is empty!")

            return None

        return *self*.queue[*self*.front]

    def display(self):

        if *self*.is\_empty():

            print("Queue is empty!")

            return

        print("Queue elements:")

        for i in range(*self*.front, *self*.rear + 1):

            print(*self*.queue[i], end=" ")

*# Example usage*

queue = LinearQueue(5)

queue.enqueue(10)

queue.enqueue(20)

queue.enqueue(30)

queue.enqueue(40)

queue.enqueue(50)

print("Initial Queue:")

queue.display()

print("\nDequeue operation:", queue.dequeue())

print("\nQueue after Dequeue:")

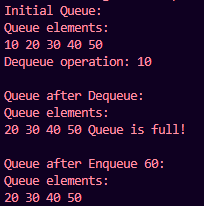
queue.display()

queue.enqueue(60)

print("\nQueue after Enqueue 60:")

queue.display()

**OUTPUT:**

****