006 Queue Constructor.mp4

In [4]:
```
class Node:
    def __init__(self, value):
        self.value = value
        self.next = None

class Queue:
    def __init__(self,value):
        new_node = Node(value)
        self.first = new_node
        self.last = new_node
        self.length = 1

    def print_queue(self):
        temp = self.first
        while temp is not None:
            print(temp.value)
            temp = temp.next

my_queue = Queue(10)

my_queue.print_queue()
```

```
10
```

007 Queue Enqueue.mp4

In [6]:
```
class Node:
    def __init__(self, value):
        self.value = value
        self.next = None

class Queue:
    def __init__(self,value):
```

```
        new_node = Node(value)
        self.first = new_node
        self.last = new_node
        self.length = 1

    def print_queue(self):
        temp = self.first
        while temp!= None:
            print(temp.value)
            temp = temp.next

    def enqueue(self, value):
        new_node = Node(value)

        if self.length == 0:
            self.first  = new_node
            self.last = new_node
        else:
            self.last.next  = new_node
            self.last       = new_node
        self.length +=1
        return True

my_queue = Queue(10)
my_queue.enqueue(20)

my_queue.print_queue()

print(f'----------------')

my_queue.enqueue(30)
my_queue.print_queue()
```

```
10
20
----------------
10
20
30
```

008 Queue Dequeue.mp4

In [11]:
```
class Node:
    def __init__(self, value):
        self.value= value
        self.next = None

class Queue:
    def __init__(self, value):
        new_node = Node(value)
        self.first = new_node
        self.last = new_node
        self.length = 1

    def print_queue(self):
        temp = self.first
        while temp != None:
            print(temp.value)
            temp = temp.next

    def enqueue(self,value):
        new_node = Node(value)
        if self.first is None:
            self.first = new_node
            self.last = new_node
        else:
            self.last.next = new_node
            self.last = new_node
        self.length +=1
        return True

    def dequeue(self):
        if self.first is None:
            return None
        else:
            temp = self.first
            self.first = self.first.next
            temp.next = None
```

```
        return temp


my_queue = Queue(10)
my_queue.enqueue(20)

my_queue.print_queue()

print(f'------Queue from last----------')

my_queue.enqueue(30)
my_queue.print_queue()


print(f'--------Dequeue from front--------')

my_queue.dequeue()
my_queue.print_queue()
```

```
10
20
------Queue from last----------
10
20
30
--------Dequeue from front--------
20
30
```

In [ ]: