```
#NAME: RADHIKA OMKAR DATAR
#id:2021sc04408
#Lets implement a queue using stacks(array implementation) using this method first

class Queue():
    def __init__(self):
        self.s1 = []
        self.s2 = []


    def peek(self):
        if len(self.s1) == 0:
            print("Queue empty")
        else:
            return self.s1[len(self.s1)-1]


    def enqueue(self, data):
        for i in range(len(self.s1)):
            item = self.s1.pop()
            self.s2.append(item)
        self.s1.append(data)
        for i in range(len(self.s2)):
            item = self.s2.pop()
            self.s1.append(item)
        return

    def dequeue(self):
        if len(self.s1)==0:
            print("Queue Empty")
            return
        else:
            return self.s1.pop()

    def print_queue(self):
        if len(self.s1) == 0:
            print("Queue Empty")
            return
        for i in range(len(self.s1) - 1,0,-1):
            print(f'{self.s1[i]} <<-- ',end='')
        print(self.s1[0])
        return


my_queue = Queue()
my_queue.enqueue(2)
my_queue.enqueue(5)
my_queue.enqueue(0)
my_queue.print_queue()
#2 <<-- 5 <<-- 0

my_queue.dequeue()
my_queue.print_queue()
#5 <<-- 0
```

```
print(my_queue.peek())
#5
my_queue.enqueue(9)
my_queue.print_queue()
#5 <<-- 0 <<-- 9

my_queue.dequeue()
my_queue.dequeue()
my_queue.dequeue()
my_queue.print_queue()
#Queue Empty


'''
For the second method, we can make the dequeue operation costly just like we did with the enqueue operation above
.
For enqueueing, we will simply push in s1.
For dequeueing, we will pop all but last element of s1 and push it onto s2. Then we will pop the last element of s1,
Which is the element we want to dequeue. After that we pop out all items of s2 and push it back onto s1.
This makes the dequeue uperation O(n) while enqueue and peek remain O(1)
'''
```