Day 9

Reversing the first k element of Queue:

```
from queue import Queue

def reversequeue(k, queue):
    if (k > Queue.qsize() or Queue.empty()
                                ==True):
        return
    if (k <=0):
        return
    Stack = []

    for i in range(k):
        Stack.append(Queue.queue[0])
        Queue.pop()
        Queue.get()

    while (len(Stack) !=0):
        Queue.put(Stack[-1])
        Stack.pop()
    for i in range(Queue.qsize() - k)
        Queue.put(Queue.queue[0])
        Queue.get()
```

```
def print(Queue):
    while (not Queue.empty()):
        print(Queue.queue[0], end = " ")
        Queue.get()
```

§ Standard operation:

* enqueue(x) : Add an item x to rear of queue.

* dequeue() : Remove an item from front of queue.

* size() : Returns number of element in queue.

* front() : finds front item.

Example:

Input : Q = [10, 20, 30, 40, 50, 60, 70, 80, 90, 100]
k = 5

Output :

Q = [50, 40, 30, 20, 10, 60, 70, 80, 90, 100]

Sorting a Queue without extra space:

```
From queue import Queue

def minindex(q, sortedIndex):
    min_index = -1
    min_val = 999999999999
    n = q.qsize()
    for i in range(n):
        curr = q.queue[0]
        q.get()
        if (curr <= min_val and i <= sortedIndex):
            min_index = i
            min_val = curr
        q.put(curr)
    return min_index

def insertMinToRear(q, min_index):
    min_val = None
    n = q.qsize()
    for i in range(n):
        curr = q.queue[0]
        q.get()
```

```
        if (i != min_index):
            q.put(curr)

        else:
            min_val = curr

    q.put(min_val)

def sortqueue(q)
    for i in range(1, q.qsize() +1):
        min_index = minindex(q, q.qsize() -i)
        insertmintorear(q, min_index)
```

Input :

11 5 4 21

Output :

4 5 11 21

interleave the first half of the queue with second half.

```
From queue import Queue
def interleave(q):
    if (q.qsize % 2 !=0):
        print("even number of integer")

    s = []
    half_size = int(q.qsize() / 2)
    for i in range(halfsize):
        s.append(q.queue[0])
        q.get()
    while len(s) !=0:
        q.put(s[-1])
        s.pop()

    for i in range(halfsize):
        q.put(q.queue[0])
        q.get()
```

```
while len(s) !=0 :
    q.put(s[-1])
    s.pop()
    q.put(q.queue[0])
    q.get()
```

input :

1 2 3 4

output :

1 3 2 4