# Circular queue implementation with fixed size

1.Initialize: - size - front pointer(exist) - rear pionter(entry) - queue 2.Enqueue: - queue is full (rear+1 == front) - first element (rear = front =0),then append - entering anywhere except at the begining(rear = rear +1%size),then append 3.Dequeue:

- rear =front=-1(queue empty)
- rear=front(only one element remain)(pop it by moving the front & rear to -1)
- dequeue the element pointed by front pointer(front means forst element)&increment front pointer dequeue = moving the front pointer to next value

```
class circularQ:
    def __init__(self,size):
        self.size=size
        self.front=self.rear = -1
        self.queue=[None] *size
    def Enqueue(self,data):
        if (self.rear + 1) % self.size == self.front:
            return("queue is full")
        elif self.rear == -1:
            self.front = self.rear=0
            self.queue[self.rear]=data
        else:
            self.rear=(self.rear+1)%self.size
            self.queue[self.rear]=data
    def dequeue(self):
        if self.front==-1:
            print("queue is empty")
        elif self.front == self.rear:
            self.front =self.rear = -1
        else:
            self.front = (self.front+1)%self.size
    def display(self):
        if self.front == -1:
            print("queue is empty")
            return
        i = self.front
        while True:
            print(self.queue[i])
            if i==self.rear:
                break
            i = (i+1) %self.size


 C =circularQ(4)

C.Enqueue(23)
```

```
C.Enqueue(45)
C.Enqueue(56)
C.Enqueue(98)
C.Enqueue(90)
```

```
'queue is full'
```

```
C.dequeue()
C.dequeue()
C.dequeue()
C.dequeue()
C.dequeue()
```

```
queue is empty
```

```
C.Enqueue(45)
C.Enqueue(56)
C.Enqueue(98)
C.Enqueue(90)
C.display()
```

```
45
56
98
90
```

def difference of sum(n,m).the function accepts two integers n,m as arguments find the sum of all numbers in range from 1 to m(both inclusive)that are not divisible by n.return difference between sum of divisible by n with sum of numbers divisible by

- n.input: n=4,=20 output:90(4+8+12+16+20=60) (1+2+3+5+6+7+9+10+11+13+14+15+17+18+19=150)(150-60=90)

```
a=[1,2,3,4,5,6,7,8,910,11,12,13,14,15,16,17,18,19,20]
div=[]
nondiv=[]
for a in range(len(a)):
    if a%4 == 0:
        div.append(a)
    else:
        nondiv.append(a)
print("div",div)
print("nondiv",nondiv)
```

```
div [0, 4, 8, 12, 16]
nondiv [1, 2, 3, 5, 6, 7, 9, 10, 11, 13, 14, 15, 17, 18]
```

```
def dif(n,m):
    divi = 0
    ndivi = 0
    for i in range(1,m+1):
```

```
        if i % 4 == 0:
            divi = divi+i
        else:
            ndivi = ndivi+1
    if ndivi > divi:
        print(ndivi-div)
    else:
        print(divi-ndivi)

d=dif(3,20)
```

```
45
```

you are give a function.checkpassword(char,n) ;the function accepts string str of size n as an argument .implement the function which return 1 if give string str is valid password else 0. str is a valid password if it staisfies the below conditions.

- at least 4 cha

```
def function(char,n):
    for i in range(n):
        if char[i]==4:
```