Understanding the problem statement

Given an array, we have to form the largest number possible which is multiple of 3. Suppose we have arr = [6,3,1,9]. Largest number possible is 9631,but it is not multiple of 3. Largest multiple of 3 is 963. ANS: 963

Algorithm

1. First sort the arr in decreasing order. 2. Find the sum of all digits. After finding sum there are three cases a. If sum of digits is divisible by 3 then that means number is divisible 3. Therefore return the number as it is. b. If the sum of digits gives remainder 1 then : b1. Remove the smallest number which gives remainder 1 when divided by 3. b2. If there is no number which gives remainder 1 when divided by 3 then remove two smallest numbers which gives remainder 2 when divided by 3. b3. If both the above cases are not possible, then return 0 c. If the sum of digits gives remainder 2 then : c1. Remove the smallest number which gives remainder 2 when divided by 3. c2. If there is no number which gives remainder 2 when divided by 3 then remove two smallest numbers which gives remainder 1 when divided by 3. c3. If both the above cases are not possible, then return 0 NOTE : Actual Implementation may be little different but the overlying idea of algorithm is same.

Implementation

In [3]:
```
class Queue:
    def __init__(self,maxsize):
        self.queue = [0]*maxsize
        self.front = 0
        self.rear = 0
        self.maxsize = maxsize

    def enqueue(self,item):
        self.rear = (self.rear+1)%self.maxsize
        if (self.front == self.rear):
            print("Overflow")
            if self.rear == 0:
                self.rear = self.maxsize - 1
            else:
                self.rear = self.rear - 1
            return
        else:
            self.queue[self.rear] = item
            return

    def dequeue(self):
        if (self.front == self.rear):
            print("Underflow")
            return -1
        else:
            self.front = (self.front+1)%self.maxsize
            item = self.queue[self.front]
            return item

    def isempty(self):
        if self.front == self.rear:
            return 1
        return 0

    def display(self):
        print(self.queue)
```

In [7]:
```
def maketemp(q0,q1,q2):
    temp = []
    while(not q0.isempty()):
```

```
        temp.append(q0.dequeue())
    while(not q1.isempty()):
        temp.append(q1.dequeue())
    while(not q2.isempty()):
        temp.append(q2.dequeue())
    return temp

def maxMultipleof3(arr,size):
    arr = sorted(arr)
    q0 = Queue(maxsize=size)
    q1 = Queue(maxsize=size)
    q2 = Queue(maxsize=size)
    arr_sum = 0
    for i in range(size):
        arr_sum += arr[i]
        if arr[i]%3 == 0:
            q0.enqueue(arr[i])
        elif arr[i]%3 == 1:
            q1.enqueue(arr[i])
        else:
            q2.enqueue(arr[i])
    if arr_sum%3 == 1:
        if not q1.isempty():
            q1.dequeue()
        else:
            if not q2.isempty():
                q2.dequeue()
            else:
                return 0
            if not q2.isempty():
                q2.dequeue()
            else:
                return 0
    elif arr_sum%3 == 2:
        if not q2.isempty():
            q2.dequeue()
        else:
            if not q1.isempty():
                q1.dequeue()
            else:
                return 0
            if not q1.isempty():
                q1.dequeue()
            else:
                return 0
    temp = maketemp(q0,q1,q2)
    temp = sorted(temp,reverse=True)
    return temp
```

In [8]:
```
arr = [9,1,6,3]
size = 4
maxMultipleof3(arr,size)
# ans
# [9,6,3]
```

Out[8]: [9, 6, 3]

In [9]:
```
arr = [8, 1, 7, 6, 0]
size = 5
maxMultipleof3(arr,size)
# ans
# [8, 7, 6, 0]
```

Out[9]: [8, 7, 6, 0]

In [ ]: