```
public class MyFIFO {
 private final Object[] array;
 private int head; //=0 //premier emplacement non libre ou sera enleve l'elt
private int tail; //=0 //emplacement libre ou sera le prochain ajoute

public MyFIFO(int maxCap) {
if(maxCap<1) throw new IllegalArgumentException();
array = new Object[maxCap]; //initial capacity, to be increased if full
 }

public void offer(Object o) {
if(head==tail && array[tail]!=null) throw new IllegalStateException("full stack");
array[tail++]=Objects.requireNonNull(o);
if(tail==array.length) tail = 0; //circulaire ! ou ne pas remettre _azero et utiliser des modulo...
}

public Object poll() {
if(head==tail && array[head]==null) throw new IllegalStateException("empty stack");
Object o = array[head];
array[head]=null;
head++;
if(head==array.length) head=0;
return o;
 }

 @Override
 public String toString() {
StringBuilder sb = new StringBuilder();
sb.append("[");
int idx = head;
int myTail = tail;
 //attention cas file pleine
 if(tail<head || (tail==head && array[head]!=null)) myTail += array.length;

 for(int i=head ; i<myTail ; ++i) {
 if(i==myTail-1) {
sb.append(array[idx]);
 }
 else {
 sb.append(array[idx]).append(",");
 }
 idx++;
 if(idx==array.length)idx=0;
 }

 return sb.append(']').toString();
 }

 public int size() {
int mytail=tail;
```

```
if(tail<head) {
mytail+=array.length;
}
return mytail-head;
}

public boolean isEmpty() {
return (head==tail && array[head]==null);
}
```