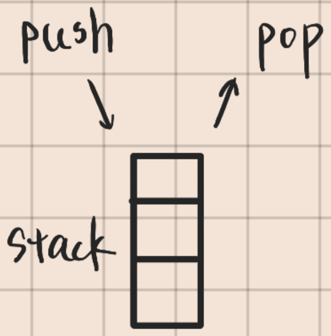
Grow Rate	Asymtopic Notation
Linear = Single loop ⟶	$O(n)$
Quadratic = loop insid loop ⟶	$O(n^2)$
Constant ⟶	$O(1)$
Cubic = loop^3 ⟶	$O(n^3)$
Exponential ⟶	$O(n^n)$

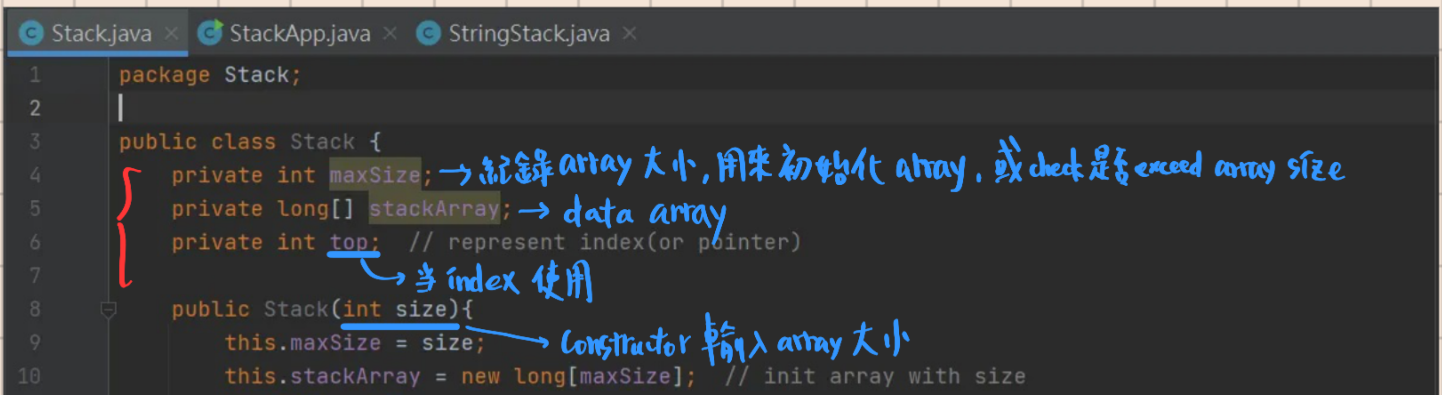
n = input size

ADT : Abstract Data Type

Stack Data Structure

LIFO : last in first out





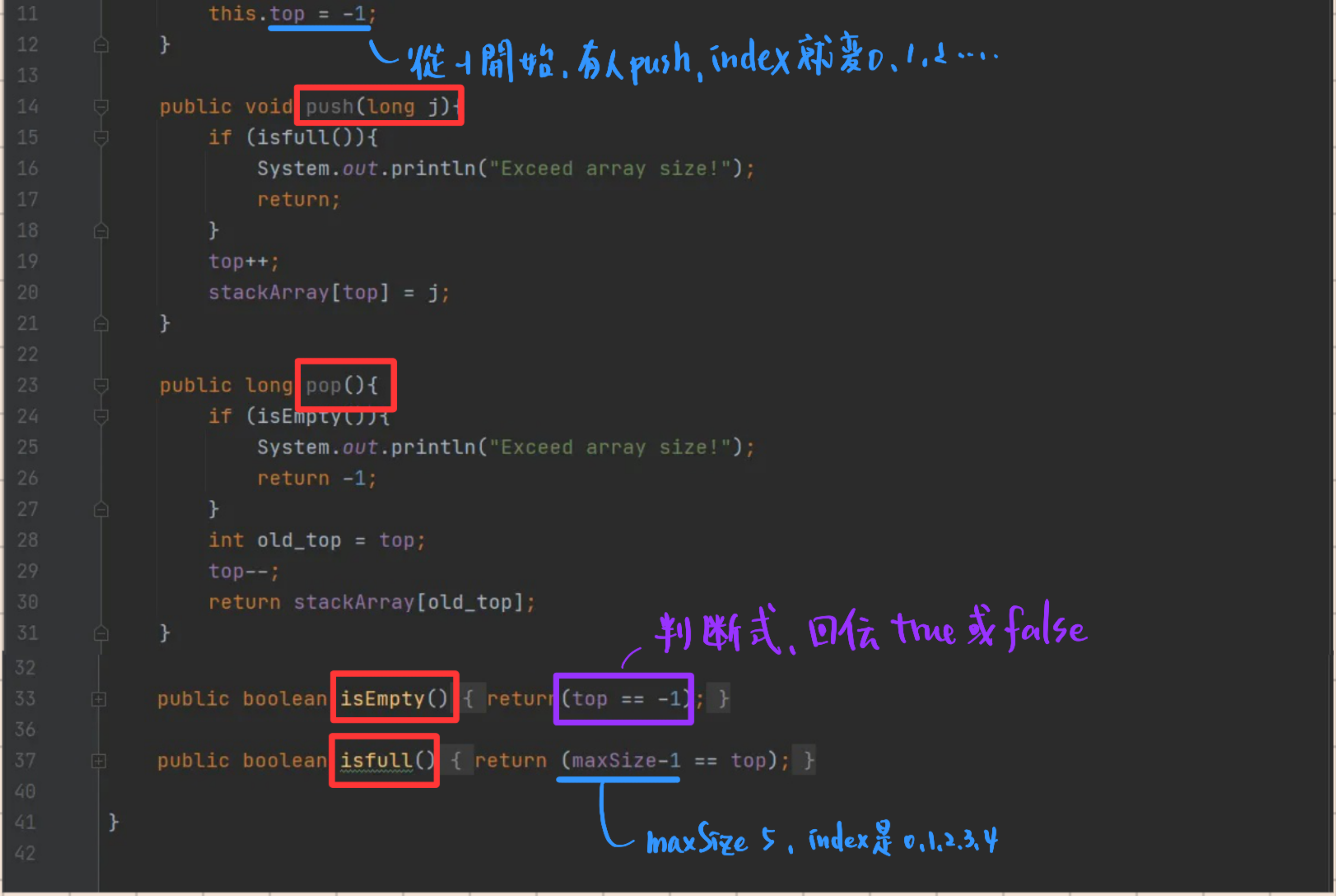
```java
package Stack;

public class Stack {
    private int maxSize;
    private long[] stackArray;
    private int top;  // represent index(or pointer)

    public Stack(int size){
        this.maxSize = size;
        this.stackArray = new long[maxSize];  // init array with size
```

maxSize → 紀錄 array 大小, 用來初始化 array, 或 check 是否 exceed array size

stackArray → data array

top → 當 index 使用

int size → Constructor 輸入 array 大小



```
        this.top = -1;
    }

    public void push(long j){
        if (isfull()){
            System.out.println("Exceed array size!");
            return;
        }
        top++;
        stackArray[top] = j;
    }

    public long pop(){
        if (isEmpty()){
            System.out.println("Exceed array size!");
            return -1;
        }
        int old_top = top;
        top--;
        return stackArray[old_top];
    }

    public boolean isEmpty() { return (top == -1); }

    public boolean isfull() { return (maxSize-1 == top); }

}
```

從-1開始，有人push，index就變0、1、2……

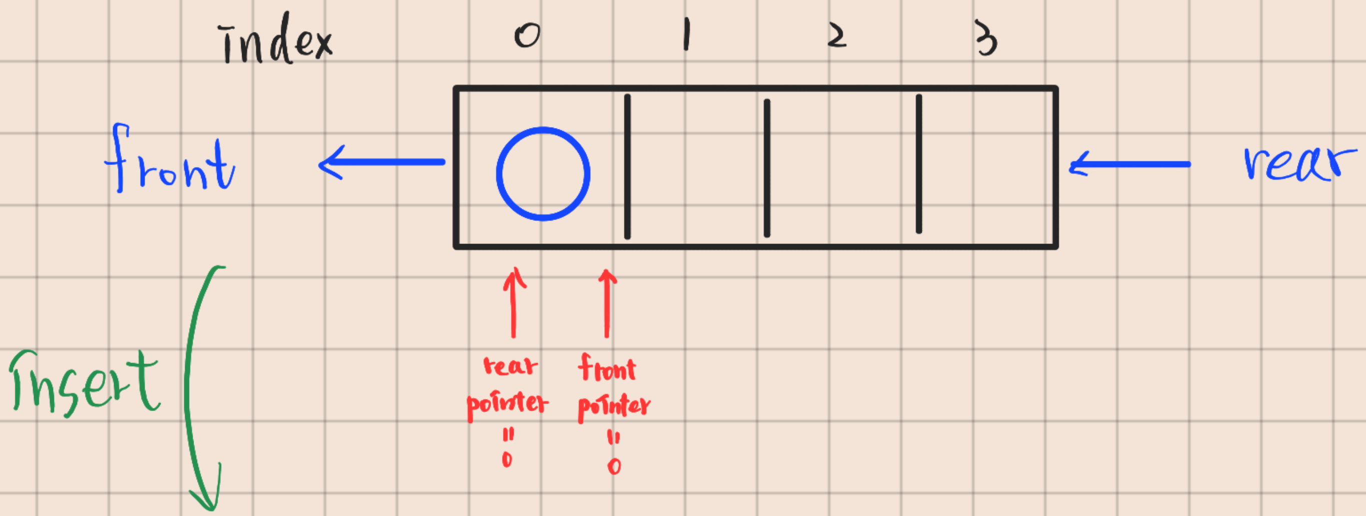
判斷式，回傳 true 或 false

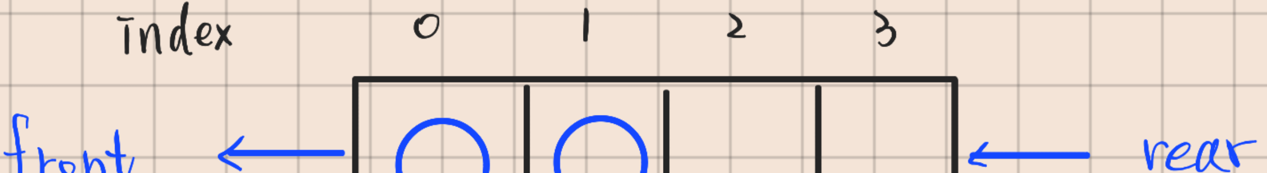
maxSize 5，index是 0,1,2,3,4

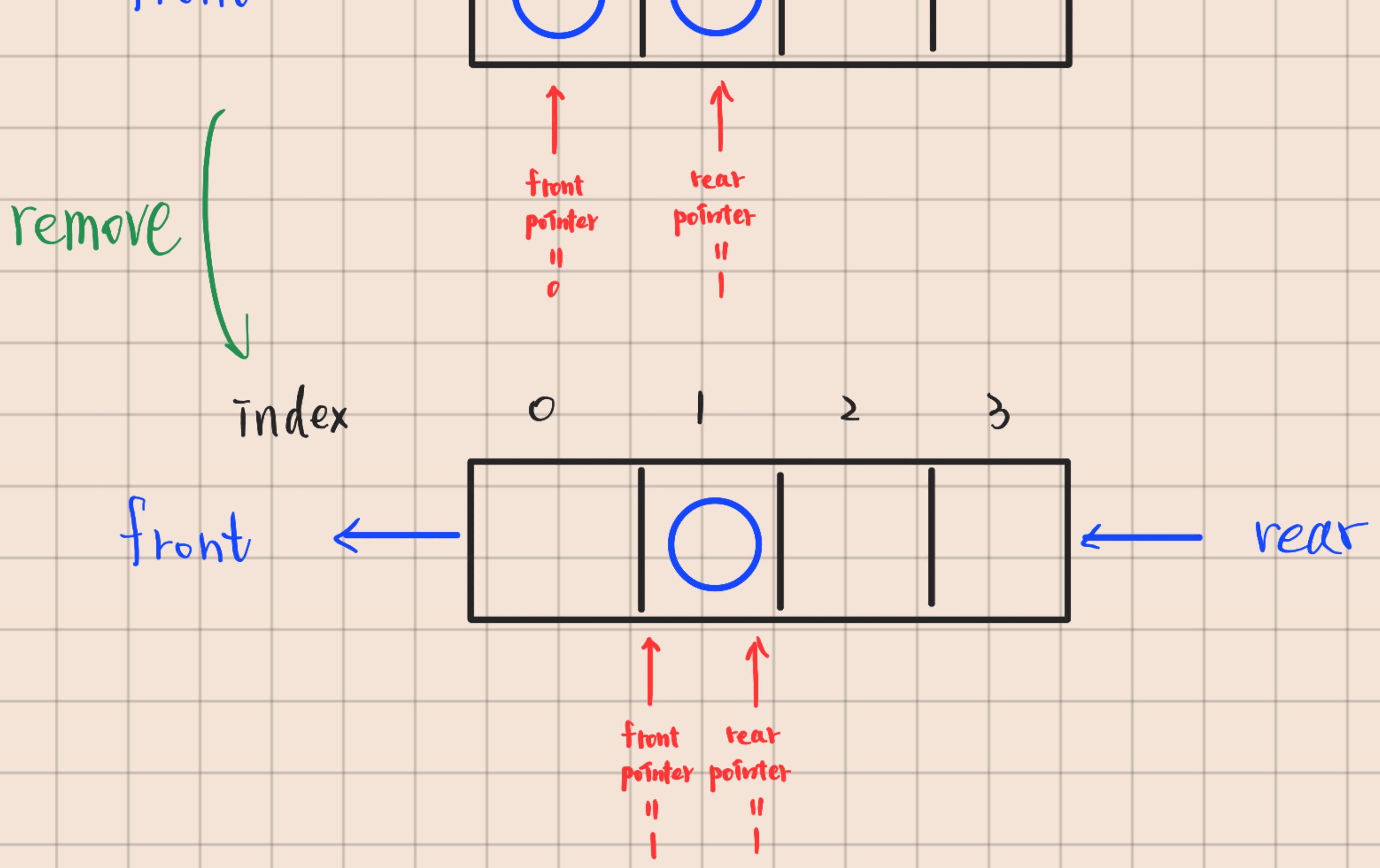
Queue Data Sturcture

FIFO: first in first out

Rear In
Front Out



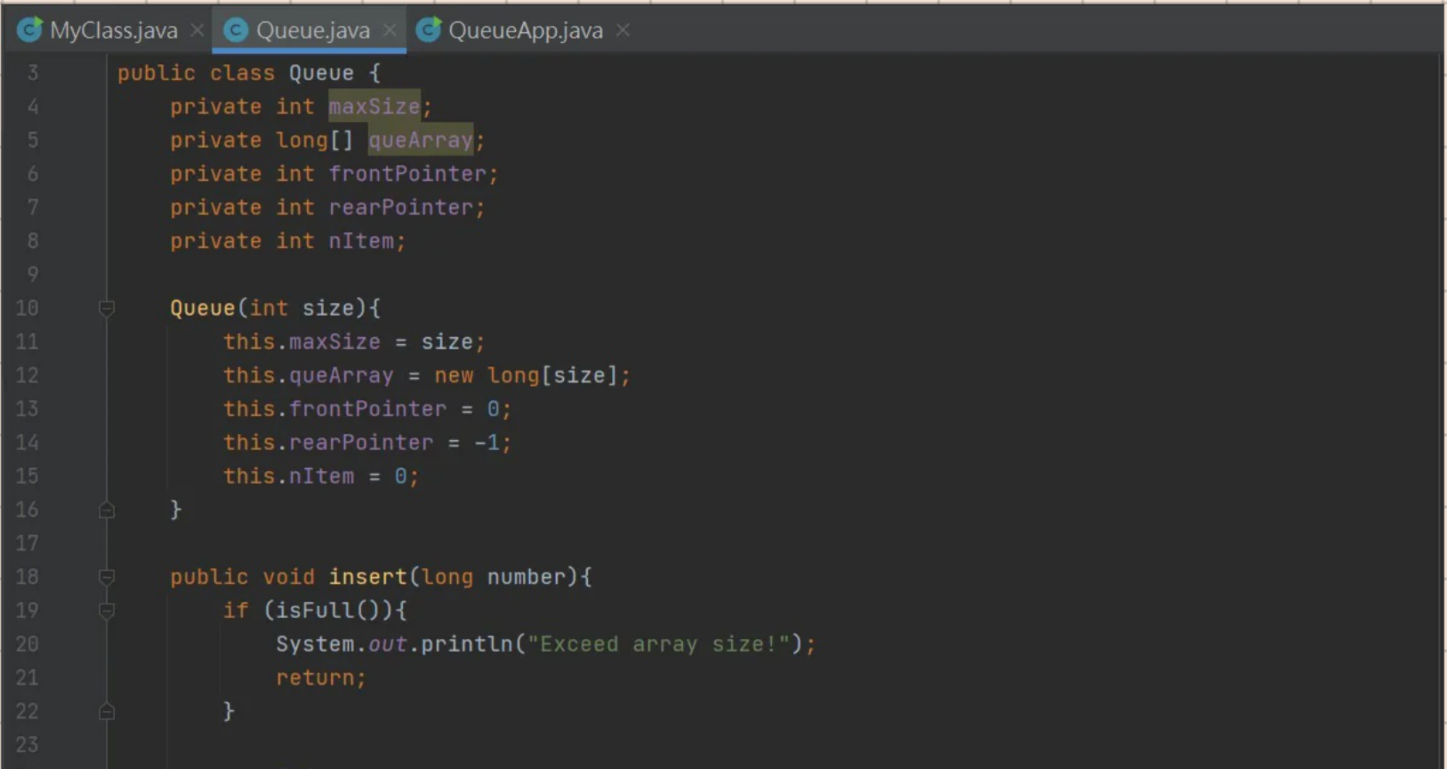




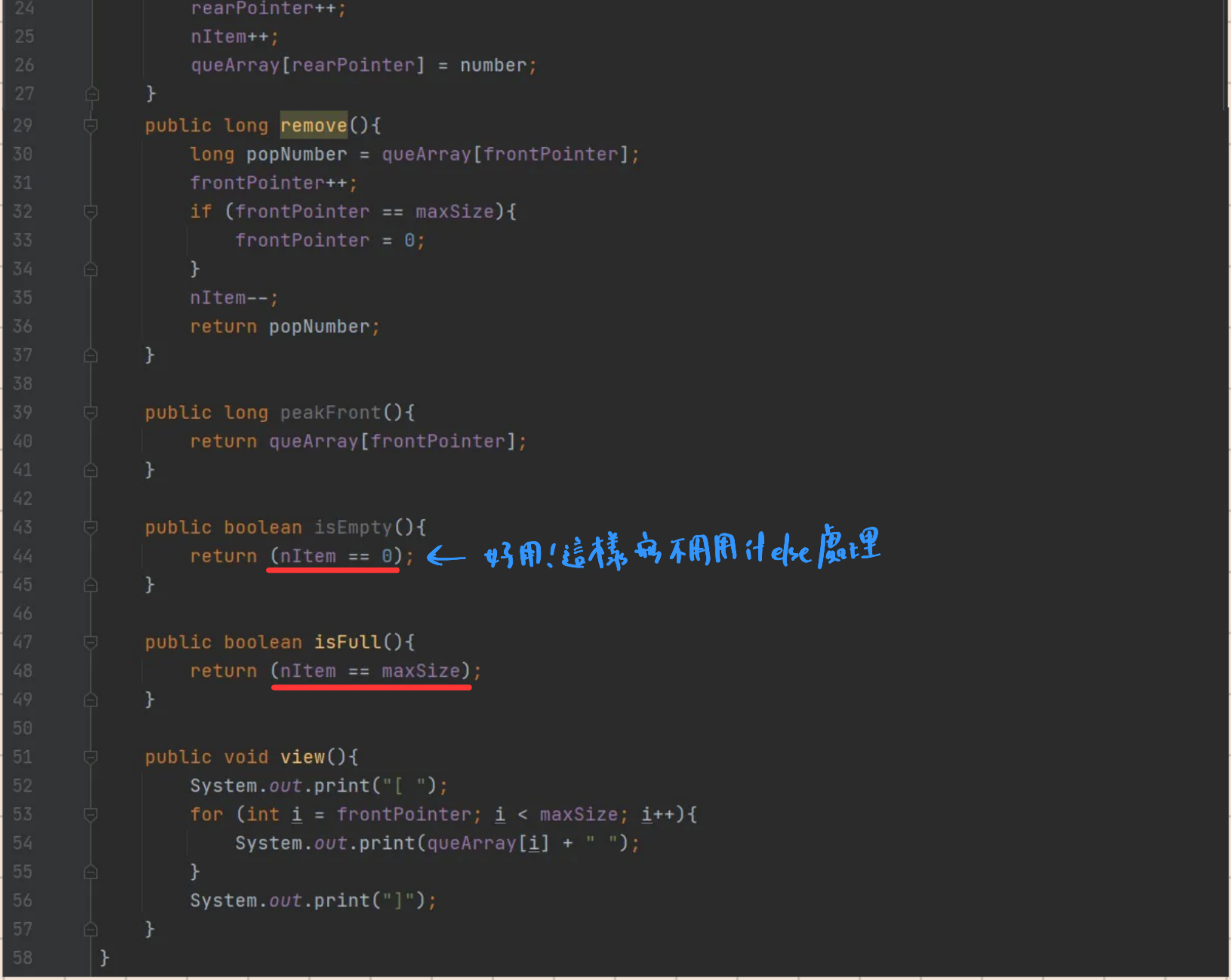
結論： 初始化 $\begin{cases} \text{front pointer} = 0 \\ \text{rear pointer} = -1 \end{cases}$

insert ⇒ rear pointer ++

remove ⇒ front pointer ++



```java
public class Queue {
    private int maxSize;
    private long[] queArray;
    private int frontPointer;
    private int rearPointer;
    private int nItem;

    Queue(int size){
        this.maxSize = size;
        this.queArray = new long[size];
        this.frontPointer = 0;
        this.rearPointer = -1;
        this.nItem = 0;
    }

    public void insert(long number){
        if (isFull()){
            System.out.println("Exceed array size!");
            return;
        }

```



```java
        rearPointer++;
        nItem++;
        queArray[rearPointer] = number;
    }
    public long remove(){
        long popNumber = queArray[frontPointer];
        frontPointer++;
        if (frontPointer == maxSize){
            frontPointer = 0;
        }
        nItem--;
        return popNumber;
    }

    public long peakFront(){
        return queArray[frontPointer];
    }

    public boolean isEmpty(){
        return (nItem == 0);
    }

    public boolean isFull(){
        return (nItem == maxSize);
    }

    public void view(){
        System.out.print("[ ");
        for (int i = frontPointer; i < maxSize; i++){
            System.out.print(queArray[i] + " ");
        }
        System.out.print("]");
    }
}
```

← 好用!這樣寫不用用if else處理