Pseudocódigo practica 2

Ejercicio A

```
public class SushiMonitor_01 {

        int emptySeats = 5
        boolean groupEating = false

        Condition canEnter

        public void enter(int i) {

                while(emptySeats <= 0 || groupEating) {

                        if (emptySeats <= 0 && !groupEating)
                                groupEating = true

                        waitC(canEnter)
                }

                decrease emptySeats
        }

        public void exit(int i) {
                increase emptySeats

                if (emptySeats >= 5)
                        groupEating = false

                signalC(canEnter)
        }
}
```

Ejercicio B

```
public class SushiMonitor_02 {

	int emptySeats = 5
	boolean groupEating = false
	LinkedHashSet waitingQueue; // ordered and unique elements

	Condition canEnter

	public void enter(int i) {

		waitingQueue.enqueue(i)

		while(emptySeats <= 0 || groupEating || waitingQueue.first != i) {
			if (emptySeats <= 0 && !groupEating)
				groupEating = true
			signalC(canEnter)
			waitC(canEnter)
		}


		waitingQueue.pop()

		decrease emptySeats
	}

	public void exit(int i) {

		increase emptySeats

		if (emptySeats >= 5)
			groupEating = false

		signalC(canEnter)
	}
}
```

Ejercicio C

```
public class SushiMonitor_03 {

        int emptySeats = 5
        boolean groupEating = false
        LinkedHashSet waitingQueue; // ordered and unique elements
        Int waitingVips = 0

        Condition canEnter

        // Mismo codigo que en el apartado A pero con cambios marcados en verde
        public void enterVIP (int i) {
                waitingVips ++;
                while(emptySeats <= 0) {
                        waitC(canEnter)
                }

                decrease emptySeats
                groupEating = false
                waitingVips --;
        }

        // Mismo codigo que en el apartado A
        public void exitVIP (int i) {
                increase emptySeats

                if (emptySeats >= 5)
                        groupEating = false

                signalC(canEnter)
        }

        public void enter(int i) {
                waitingQueue.enqueue(i)

                while(waitingVips > 0 || emptySeats <= 0 || groupEating ||
waitingQueue.first != i) {
                        if (emptySeats <= 0 && !groupEating)
                                groupEating = true
                        signalC(canEnter)
                        waitC(canEnter)
                }


                waitingQueue.pop()

                decrease emptySeats
        }

        public void exit(int i) {

                increase emptySeats
```

```
                if (emptySeats >= 5)
                        groupEating = false

                signalC(canEnter)
        }
}
```