2. Pseudocode Algorithm B

Pseudocode Algorithm B: Priority Queue

Algorithm B: Priority Queue

Input: Order ใหม่
Output: Order ที่ถูกเลือกตาม Priority

1. Create `PriorityQueue`
2. รับ `Order` ใหม่
3. กำหนด `Priority` ให้กับ Order
4. เพิ่ม `Order` เข้า `PriorityQueue`
5. ตรวจสอบ `Priority` ของ Order ใน Queue
6. ถ้า `Queue` ว่าง
   - Return `false`
7. ถ้า `Queue` ไม่ว่าง
   - เลือก `Order` ที่มี Priority สูงสุด
   - ถ้า Priority เท่ากัน ให้ดู `Order Time`
   - นำ `Order` ออกจาก Queue ด้วย `poll()`
   - ประมวลผล `Order`
   - บันทึก `Waiting Time`
8. ตรวจสอบว่า `Order` ต่อไปอยู่ใน Queue
9. คืนค่า `Order`
10. End

ระดับ Priority

Express > Normal > Large

เหตุผลของการทำแบบนี้

- Express ต้องได้รับความสำคัญสูงสุด
- Normal หรือ Large สามารถรอได้
- ถ้า Priority เท่ากันสามารถใช้ `Order Time` ตัดสินได้
- ลดเวลารอของ Order ที่เร่งด่วน

ข้อจำกัด

- โครงสร้างซับซ้อนกว่า Algorithm A
- Normal หรือ Large อาจรอนาน
- อาจเกิดปัญหา Starvation ถ้ามี Express เข้ามาตลอด
- ต้องจัดการ Priority และ Order Time ให้ถูกต้อง

ตัวอย่างลำดับการทำงาน

```
02 (Express)
      ↓
04 (Express)
      ↓
01 (Normal)
      ↓
03 (Normal)
      ↓
05 (Large)
```

ดังนั้นลำดับคือ

`02 → 04 → 01 → 03 → 05`