Priority Task Execution

ระบบปฏิบัติการหนึ่งได้จัดลำดับการประมวลผลงานที่เข้ามาตามลำดับความสำคัญของงาน (priority ที่มีค่าต่ำจะได้ประมวลผลก่อน) กำหนดให้มีคิวงานจำนวน 4 คิวงาน แต่ละงานอ้างอิงด้วยเลข 2 หลัก

การประมวลผล (Execute) ในแต่ละรอบจะใช้เวลา 2 วินาที ถ้างานใดใช้เวลาประมวลผลมากกว่า 2 วินาที งานนั้นๆ จะยังคงค้างอยู่ในคิวเพื่อรอการประมวลผลในรอบถัดไปโดยเก็บเวลาที่ยังคงค้างอยู่ และให้ปรับ priority ให้มีค่าเพิ่มขึ้น 1 ค่า (ลดความสำคัญลง) แล้วไปต่อท้ายคิวที่มี priority เท่ากัน

กรณีงานที่เข้าสู่ระบบมากกว่าขนาดของคิว ให้พิจารณาจำนวนงานในลำดับแรกก่อน (เท่ากับขนาดของคิว) จากนั้นเมื่อประมวลผลจนมีที่ว่างในคิวจึงทำการพิจารณางานที่เหลือเข้าสู่คิวด้วย priority ของงาน

งานของคุณ

จงเขียนโปรแกรมเพื่อจัดการงานในคิวด้วย Circular Priority Queue เพื่อดำเนินการประมวลผลนี้

ข้อมูลนำเข้า

บรรทัดแรก ประกอบด้วยเลข 2 จำนวน คือ จำนวนงาน n งาน ที่รับเข้ามาเพื่อรอการประมวลผล และจำนวนรอบที่ต้องการให้ประมวลผล (r) โดยที่ $1 \leq n \leq 8$ และ $r \geq 0$

n บรรทัดถัดมา คือ งานที่จำทำการประมวลผล ประกอบด้วย 3 ค่า คือ task_ID, Priority, exe_Time

ข้อมูลส่งออก

Task_ID ตัวเลข 2 หลัก

$0 \leq$ Priority ≤ 5

$1 \leq$ exe_Time ≤ 10

<u>กรณีที่ 1</u> จำนวนรอบในการประมวลผลที่กำหนดในข้อมูลนำเข้า r > จำนวนรอบสูงสุดที่ต้องใช้ในการประมวลผลจนเสร็จ

ให้แสดงจำนวนรอบสูงสุดที่ต้องใช้ในการประมวลผลจนเสร็จ

<u>กรณีที่ 2</u> จำนวนรอบในการประมวลผลที่กำหนดในข้อมูลนำเข้า r ≤ จำนวนรอบสูงสุดที่ต้องใช้ในการประมวลผลจนเสร็จ

ให้แสดงจำนวนงานที่เหลืออยู่ในคิว และจำนวนครั้งที่เหลือสำหรับการประมวลผลที่จะเคลียร์งานที่ค้างอยู่ในคิว

ตัวอย่างที่ 1

ข้อมูลนำเข้า			ข้อมูลส่งออก	
4	3		2	3
10	1	5		
30	3	1		
09	2	1		
12	2	2		

ตัวอย่างที่ 2

ข้อมูลนำเข้า			ข้อมูลส่งออก
4	7		6
10	1	5	
30	3	1	
09	2	1	
12	2	2	

ตัวอย่างที่ 3

ข้อมูลนำเข้า			ข้อมูลส่งออก	
7	3		4	9
10	1	5		
30	3	1		
09	2	1		
12	2	2		
44	3	4		
55	2	3		
66	1	1		