**การจำลองการทำงานของโครงสร้างข้อมูล Queue**

ให้เขียนโปรแกรมเพื่อทำการจำลองการทำงานของ Queue โดยใช้ Pointer Based Implementation สำหรับข้อมูลที่แต่ละโหนดใน Queue เก็บกำหนดให้เป็นเลขจำนวนเต็ม 1 ตัว ส่วนการเลือกทำ Operation จะให้รับข้อมูลเข้าเป็นจำนวนเต็มระหว่าง 1-7 โดยที่

|  |  |
| --- | --- |
| Operation | หน้าที่ |
| 1 | เรียก operation enqueue เพื่อทำการเพิ่มข้อมูลเข้าไปใน Queue  เมื่อเลือก operation แล้วจะให้รับ parameter 1 ตัวเป็นข้อมูลที่ต้องการเพิ่ม  การแสดงผลหลังจากเรียก operation : ไม่มี |
| 2 | เรียก operation dequeue เพื่อนำข้อมูลตัวที่เพิ่มเข้าไปนานที่สุดออกจาก Queue การแสดงผลหลังจากเรียก operation : ให้แสดงค่าข้อมูลตัวที่เพิ่มนานสุด ถ้าไม่มีให้แสดง no |
| 3 | เรียก operation getFront เพื่อสอบถามข้อมูลตัวที่เพิ่มเข้าไปนานที่สุดใน Queue  การแสดงผลหลังจากเรียก operation : ให้แสดงค่าข้อมูลตัวที่เพิ่มนานสุด ถ้าไม่มีให้แสดง no |
| 4 | เรียก operation clearQueue เป็นการเคลียร์ค่า Queue ให้เป็น Queue ว่างโดยลบโหนดออกจาก Queue  การแสดงผลหลังจากเรียก operation :  ให้แสดงค่าข้อมูลตัวที่เพิ่มนานสุดจนหมดทุกตัว แต่ละตัวคั่นด้วย space ถ้าไม่มีให้แสดง no |
| 5 | เรียก operation size เพื่อสอบถามจำนวนข้อมูลที่ Queue เก็บ  การแสดงผล : ให้แสดงจำนวนข้อมูลที่เก็บ |
| 6 | เรียก operation isEmpty เพื่อสอบถามว่า Queue ว่างหรือไม่  การแสดงผลหลังจากเรียก operation :  ถ้า Queue ว่าง แสดงผลเป็น yes ถ้า Queue ไม่ว่างแสดงผลเป็น no |
| 7 | จบการทำงานของโปรแกรม (แต่ละข้อมูลเข้าจะมี 7 เป็นข้อมูลสุดท้าย) |

**ข้อมูลเข้า**

จะเป็นเลขจำนวนเต็ม 1 ตัวต่อหนึ่งบรรทัด ซึ่งเป็นการเรียกใช้งาน operation ต่างๆ และเรียกจบการทำงานด้วย 7 เป็นข้อมูลสุดท้ายเสมอ

**ข้อมูลออก**

แต่ละบรรทัดเป็นการแสดงผลตามการดำเนินการที่ได้รับ

**ตัวอย่าง**

|  |  |  |
| --- | --- | --- |
| ข้อมูลเข้า | ข้อมูลเข้า | ข้อมูลเข้า |
| 1 2  3  4  5  6  7 | 1  2  1  3  1  4  2  5  6  7 | 1  1  1  2  1  3  1  4  1  5  1  6  4  3  2  7 |
| ข้อมูลออก | ข้อมูลออก | ข้อมูลออก |
| 2  2  0  yes | 2  2  no | 1 2 3 4 5 6  no  no |