โครงสร้างข้อมูลแบบ คิว

QUEUE

นิยามของคิว

คิวเป็นโครงสร้างข้อมูลแบบหนึ่งซึ่งมีลักษณะที่ว่า ข้อมูลที่นำเข้าไปเก็บก่อนจะถูกนำออกมาทำงานก่อน ส่วนข้อมูลที่เข้าไปเก็บทีหลังก็จะถูกนำออกมาใช้งานทีหลัง ขึ้นอยู่กับลำดับการเก็บข้อมูล จะเรียกลักษณะการทำงานแบบนี้ว่า **เข้าก่อนออกก่อน หรือ First In First Out (FIFO)**

โครงสร้างข้อมูลแบบนี้เป็นโครงสร้างที่ปรากฏอยู่โดยทั่วๆ ไป โดยเฉพาะอย่างยิ่งในระบบปฏิบัติการคอมพิวเตอร์ ในระบบคมนาคม รวมทั้งในระบบการทดลองดาวเทียมด้วย

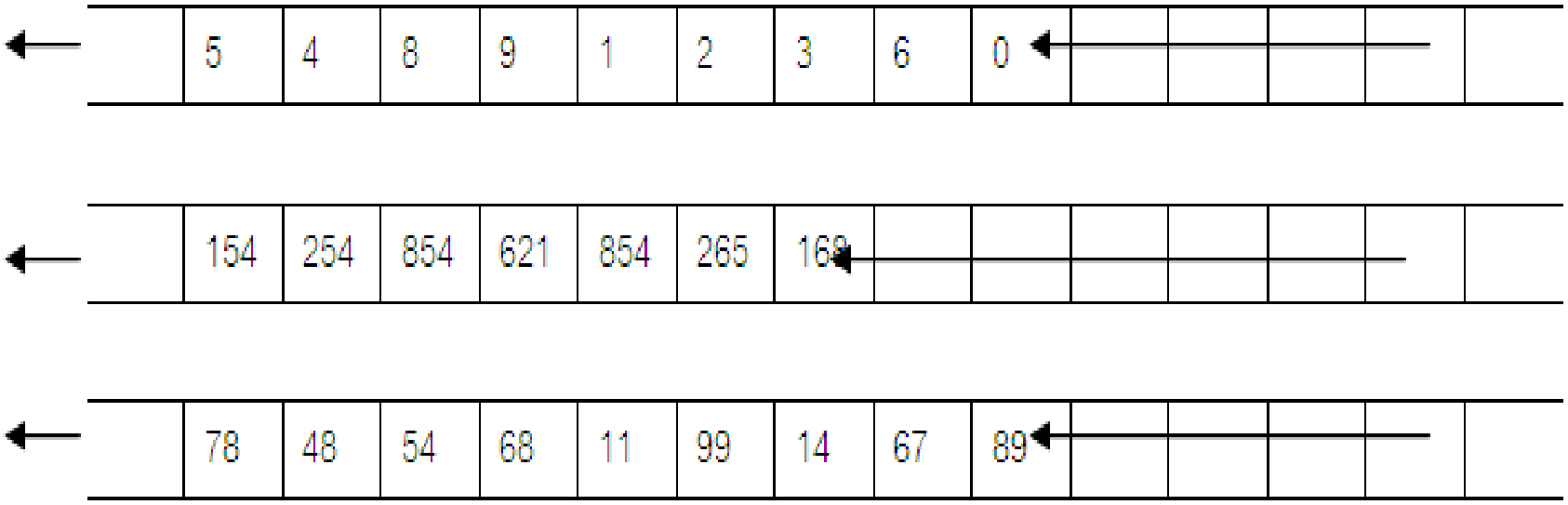
นิยามของคิว

ลักษณะของโครงสร้างแบบคิวจะเหมือนกับการเข้าแถวรอคอย ไม่ว่าจะเป็นการรอคอยอะไรก็ตาม หรือจะเรียกสั้นๆ ว่า เข้าคิว ก็ได้ ด้วยคุณสมบัติที่เด่นชัดของการทำงานของโครงสร้างข้อมูลแบบคิวนี้ ว่าสิ่งใดที่เข้าก่อนย่อมต้องได้รับการทำงานก่อน

คิวจะมีโครงสร้างแบบเชิงเส้นเหมือน stack แต่แตกต่างตรงที่ queue มีตัวชี้ 2 ตัวคือ หัว(Head) และหาง(Tail) โดยการใส่ข้อมูลเข้าและนำข้อมูลออก

นิยามของคิว

การนำข้อมูลสู่คิว จะเหมือนกับการยืนต่อแถวคอยอยู่



การกระทำ(Operation) โครงสร้างข้อมูลแบบ Queue

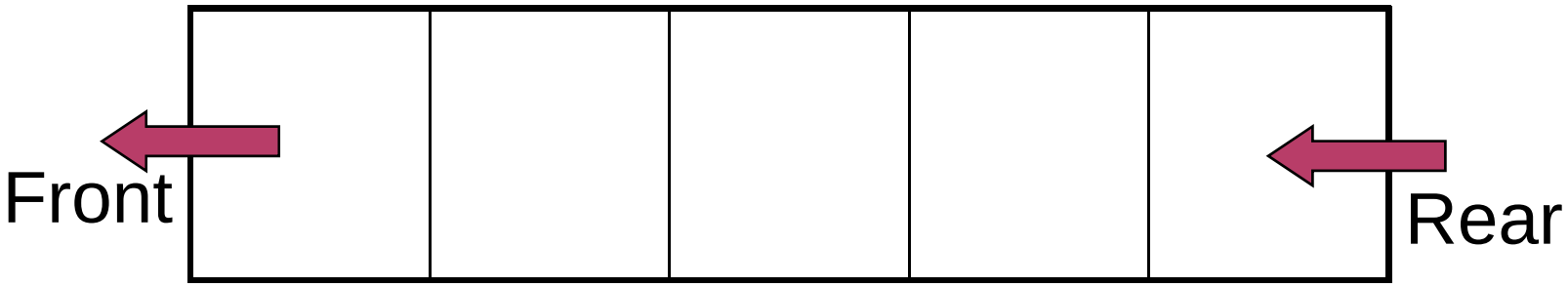
การกระทำ(Operation) ที่เกี่ยวข้องกับโครงสร้างข้อมูลแบบ Queue ประกอบด้วย

- การสร้างคิว(Create)
- การนำสมาชิกข้อมูลเข้าคิว(Enqueue)
- การนำสมาชิกข้อมูลออกจากคิว(Dequeue)
- การทดสอบว่าคิวว่างหรือไม่(Empty)
- การทดสอบว่าคิวเต็มหรือไม่(Full)
- การทำให้คิวเป็นคิวว่าง(Clear)

ลักษณะของโครงสร้างข้อมูลแบบคิว

- คล้ายกับโครงสร้างข้อมูลแบบ Stack
- แต่จะต่างตรงที่ข้อมูลที่เข้าและออกจะกระทำที่ปลายคนละด้านกัน
- ข้อมูลที่เข้ามาใหม่จะเพิ่มเข้ามาด้านหลัง (Rear)
- ข้อมูลที่จะนำออกก่อนจะอยู่ด้านหน้า (Front)
- ลักษณะนี้เรียกว่า เข้ามาก่อน ออกไปก่อน (FIFO หรือ FCFS)

ตัวอย่าง

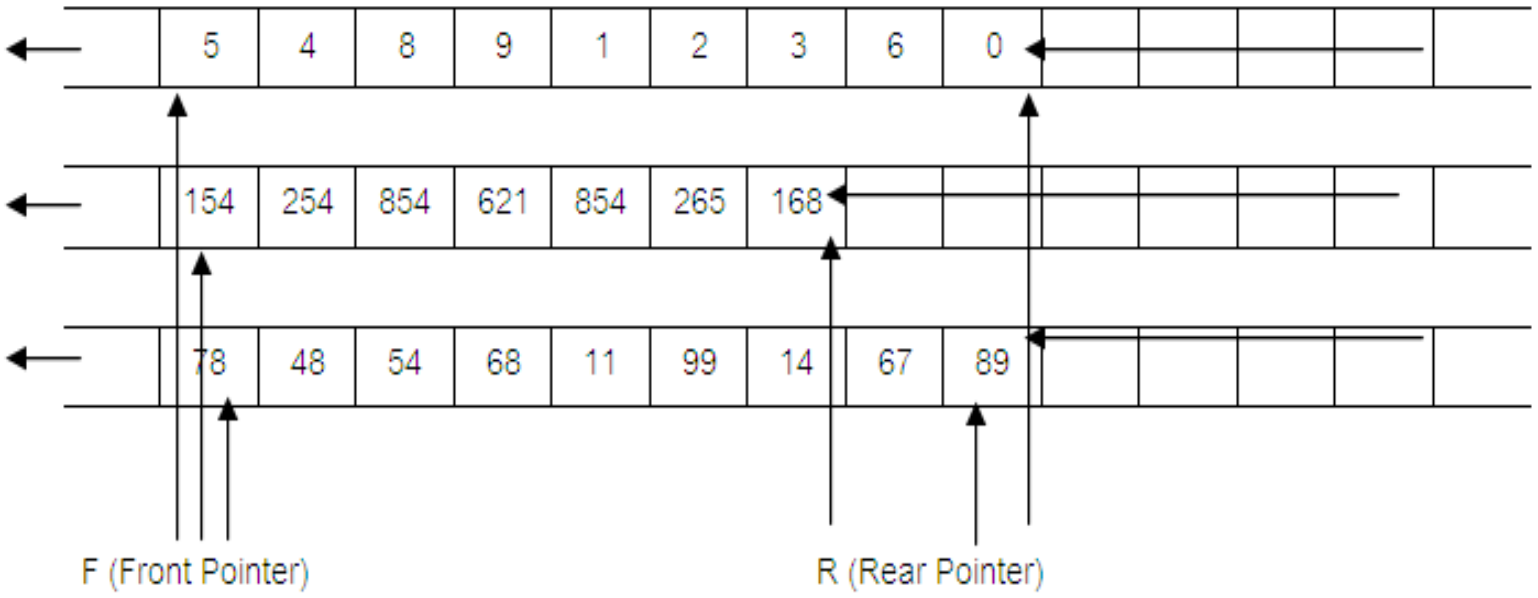


การสร้างคิว (Queue)

คิวที่อยู่ในคอมพิวเตอร์สามารถจัดเก็บได้หลายลักษณะ แต่โดยทั่วไปแล้วจะใช้การจัดเก็บแบบลิงค์ลิสท์เดี่ยวหรือจัดเก็บโดยใช้อาร์เรย์

ก่อนที่จะทำการสร้างคิวจะต้องทำความเข้าใจถึงโครงสร้างของคิว ซึ่งประกอบไปด้วย ตัวคิว ซึ่งในที่นี้ขอแทนด้วยอาร์เรย์ และจะต้องมีตัวชี้อีก 2 ตัว ได้แก่ ตัวชี้ F (Front Pointer) ชี้ไปที่สมาชิกตัวแรก และตัวชี้ R (Rear Pointer) ชี้ไปที่สมาชิกตัวสุดท้ายของคิว โดยที่เวลาข้อมูลจะเข้าสู่คิวจะเข้าทาง R ส่วนเวลาที่ข้อมูลจะออกจากคิวจะออกทาง F

การสร้างคิว (Queue)



การสร้างคิว (Queue)

เมื่อเริ่มต้นสร้างคิว คิวนั้นไม่มีค่าใดๆ ยังว่างเปล่า F และ R จะมีค่าเป็น 0 ทั้งคู่ คือไม่ได้ชี้ไปที่สมาชิกตัวใด การนำข้อมูลเข้าสู่คิวเรียกว่า การ Insertion ส่วนการนำข้อมูลออกจากคิวเรียกว่า การ Deletion

การนำสมาชิกข้อมูลเข้าคิว(Enqueue)

- การนำข้อมูลใหม่เข้ามาแถวคอย จะเพิ่มเข้ามาด้านหลัง
- และจะนำเข้ามาเรื่อย ๆ จนเต็ม หรือเรียกว่า แถวคอยเต็ม (Queue Overflow)

เป็นการนำข้อมูลเข้าสู่คิว โดยการที่จะนำข้อมูลเข้าสู่คิวนั้นจะแบ่งออกเป็น 2 กรณี คือ

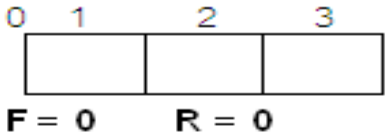
1. การนำข้อมูลเข้าไปในคิวว่าง โดยจะต้องดำเนินการให้พอยน์เตอร์ทั้ง 2 คือ F และ R ชี้ไปยังช่องแรกหรือตำแหน่งที่จะเก็บข้อมูลแรก

2. การนำข้อมูลเข้าไปในคิวต่อจากข้อมูลเดิม จะต้องจัดการให้พอยน์เตอร์ R ชี้ไปยังช่องหรือตำแหน่งของข้อมูลที่นำเข้าไป ส่วนพอยน์เตอร์ F ยังคงชี้ไปยังช่องหรือตำแหน่งของข้อมูลที่นำเข้าไปเป็นข้อมูล

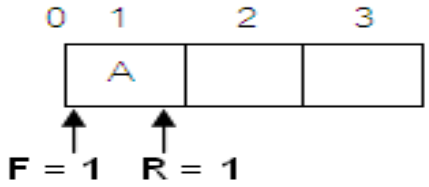
การนำสมาชิกข้อมูลเข้าคิว(Enqueue)

จากการ Insertion ข้อมูลเข้าไปในคิวแล้ว ถ้าหากทำการ Insertion ข้อมูลจน**พอยน์เตอร์ R อยู่ที่ช่องสุดท้ายแล้ว จะไม่สามารถทำการ Insertion ข้อมูลลงคิวได้อีก** เนื่องจากตัวที่เก็บข้อมูลคิวเต็มทำให้ไม่สามารถที่จะรับข้อมูลอื่นๆ อีกได้ ฉะนั้นจะเกิด Error ขึ้น ซึ่ง Error นี้เรียกว่า Overflow ขึ้น เช่น ถ้าจอง Array ขนาด 3 ช่องชื่อ Q

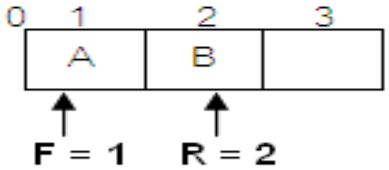
1. เมื่อคิวว่าง (Empty Queue)



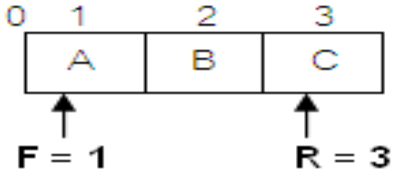
2. INSERT (Q , A)



3. INSERT (Q , A)



4. INSERT (Q , C)



หลังจากนี้จะเพิ่มข้อมูลเข้าคิวอีกไม่ได้ เนื่องจากคิวเต็มคือ R = 3

การนำสมาชิกข้อมูลออกจากคิว(Dequeue)

- ข้อมูลที่จะนำออกก่อนจะเป็นข้อมูลที่อยู่ด้านหน้า
- สามารถนำข้อมูลออกเรื่อย ๆ จนไม่มีข้อมูล หรือเรียกว่า แถวคอยว่าง (Queue Underflow)

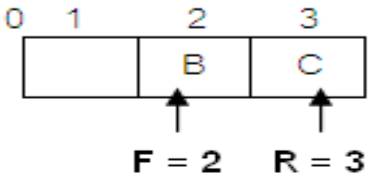
เป็นการนำข้อมูลที่เก็บอยู่ในคิวออกจากคิว โดยการเมื่อทำการ Deletion ข้อมูลนั้นออกจากคิวแล้ว จะต้องมีการจัดการให้ตัวชี้คิว F ชี้ไปยังช่องหรือตำแหน่งต่อจากข้อมูลที่จะได้ทำการ Deletion ไปแล้ว ส่วนพอยน์เตอร์ R ชี้ไปยังช่องข้อมูลสุดท้ายเหมือนเดิม

การ Deletion ข้อมูลนี้จะทำการนำข้อมูลในส่วนของข้อมูลตัวแรกสุดที่เข้าสู่คิวออกไปทำงานตามต้องการ แต่การ Deletion ข้อมูลนี้จะ**ไม่สามารถ Deletion ข้อมูลออกจากคิวที่ว่างเปล่าหรือไม่มีข้อมูลได้ (F = 0)** ถ้าเกิดกรณีเช่นนี้จะเกิด Error ที่เรียกว่า Underflow ขึ้น ฉะนั้นก่อนที่จะทำการ Deletion ควรที่จะต้องมีการตรวจสอบว่าคิวว่างหรือไม่ เพื่อไม่ให้เกิด Error นี้ขึ้น

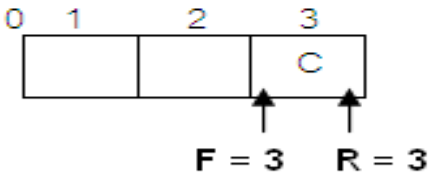
การนำสมาชิกข้อมูลออกจากคิว(Dequeue)

จากตัวอย่างข้างบน เมื่อทำการ Deletion

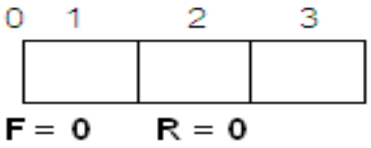
5. DELETE (Q)



6. DELETE (Q)



7. DELETE (Q)



การสร้างคิวด้วย Array

หมายถึง การแทนที่ข้อมูลของคิวด้วย array ซึ่งเป็นการจัดสรรเนื้อที่หน่วยความจำแบบ static นั่นคือ มีการกำหนดขนาดของคิวล่วงหน้าว่ามีขนาดเท่าใด และจะมีการจัดสรรเนื้อที่หน่วยความจำให้เลย

การแทนโครงสร้างข้อมูลQueue ด้วย Array

- จะมีการกระทำทั้งหมด 4 ขั้นตอนคือ
  - การสร้าง
  - การนำข้อมูลเข้า
  - การนำข้อมูลออก
  - การแสดงข้อมูล

การสร้าง

- เป็นขั้นตอนแรกของโครงสร้างข้อมูลแบบแถวคอย คือก่อนที่จะเก็บข้อมูลต่าง ๆ จะต้องทำการสร้างที่เก็บข้อมูลก่อน
- มีการกำหนดตัวชี้ 2 ตัวคือ F และ R
- ตัวอย่างภาษาซี เช่น

```
int  Queue[4];

int  F,R;
```

การสร้างคิวด้วย Link List

หมายถึง การแทนที่ข้อมูลของคิวด้วย Link list ซึ่งเป็นการจัดสรรเนื้อที่หน่วยความจำแบบ Dynamic นั่นคือ หน่วยความจำจะถูกจัดสรรเมื่อมีการของใช้จริงๆ ระหว่างการประมวลผลโปรแกรมผ่านตัวแปรชนิด Pointer

การแทนที่โครงสร้างข้อมูลแบบแถวคอยด้วยรายการโยง Link List

- มีทั้งหมด 4 ขั้นตอนดังนี้
  - การสร้าง
  - การนำข้อมูลเข้า
  - การนำข้อมูลออก
  - การแสดงข้อมูล

การสร้าง

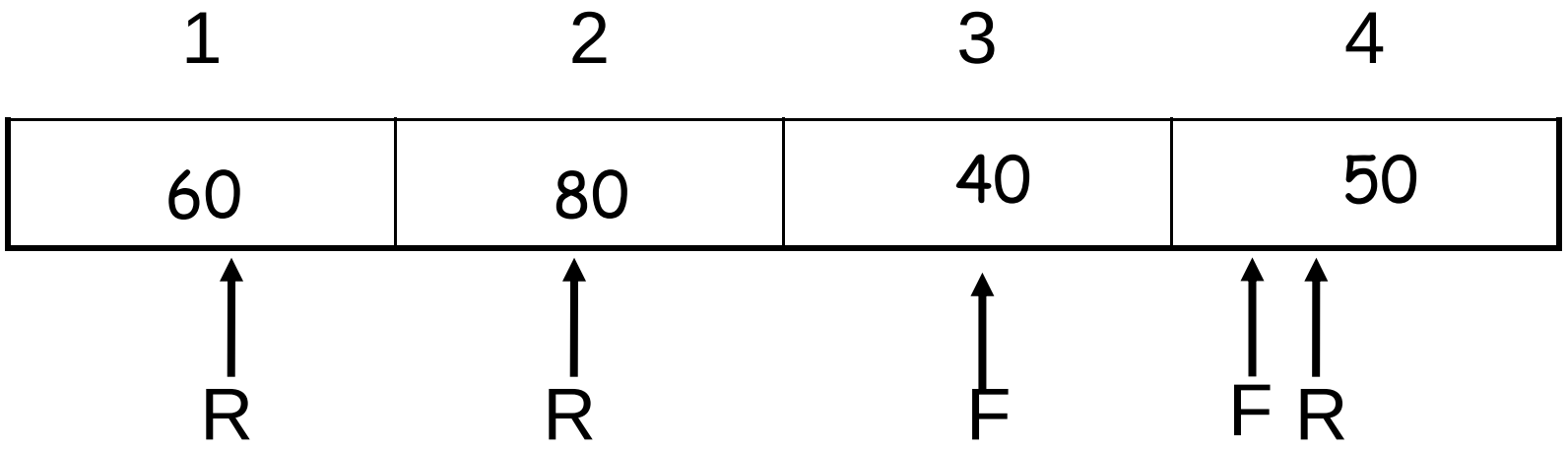
- การสร้างรายการโยงจะกำหนดตัวชี้ 2 ตัวคือ F และ R ดังนี้

```
typedef struct node
{  int data;
    struct node *next;
} node;
node *F, *R;
```

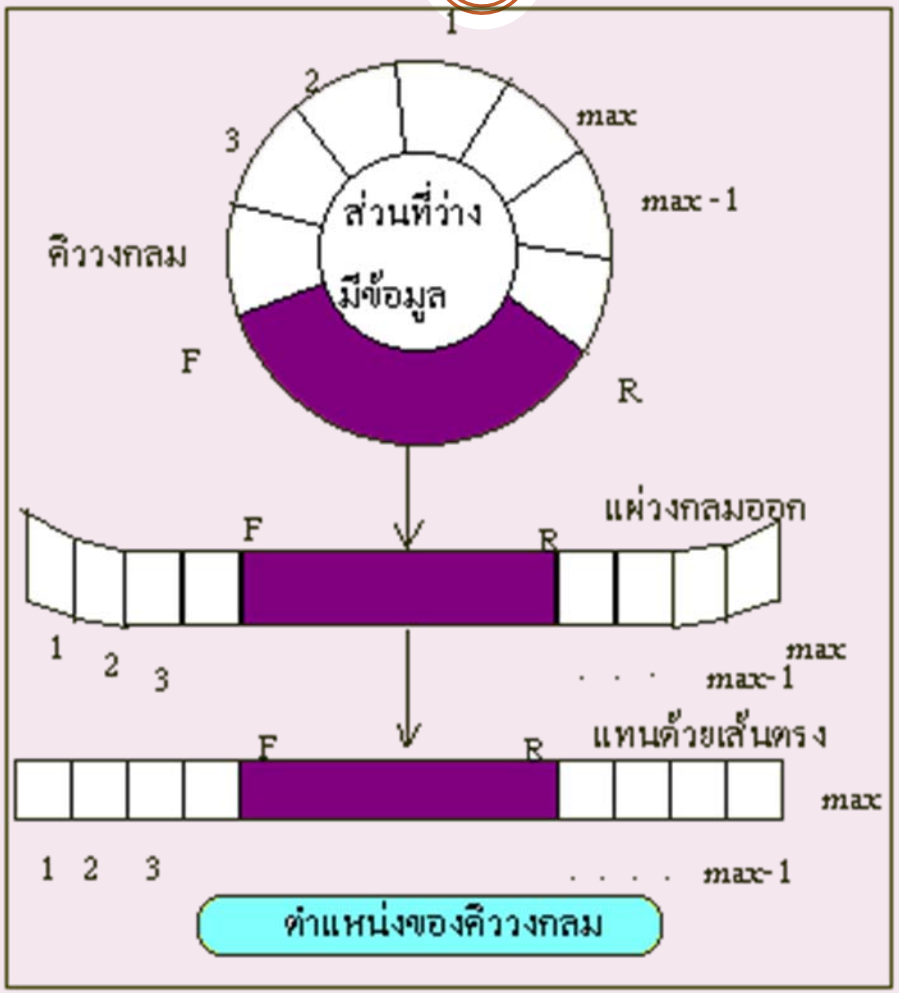
คิววงกลม (Circular Queue)

หลักการของคิววงกลมคือการนำหัวคิวและหางคิวมาเชื่อมต่อกันเป็นวง ทำให้สามารถเพิ่มข้อมูลต่อกันไปเรื่อย ๆ ได้ โดยไม่จำกัดอยู่ที่ขนาดของอะเรย์ (Array Size) โดยอาจจินตนาการถึงคิววงกลมได้ดังภาพ

โครงสร้างแถวคอยแบบวงกลม



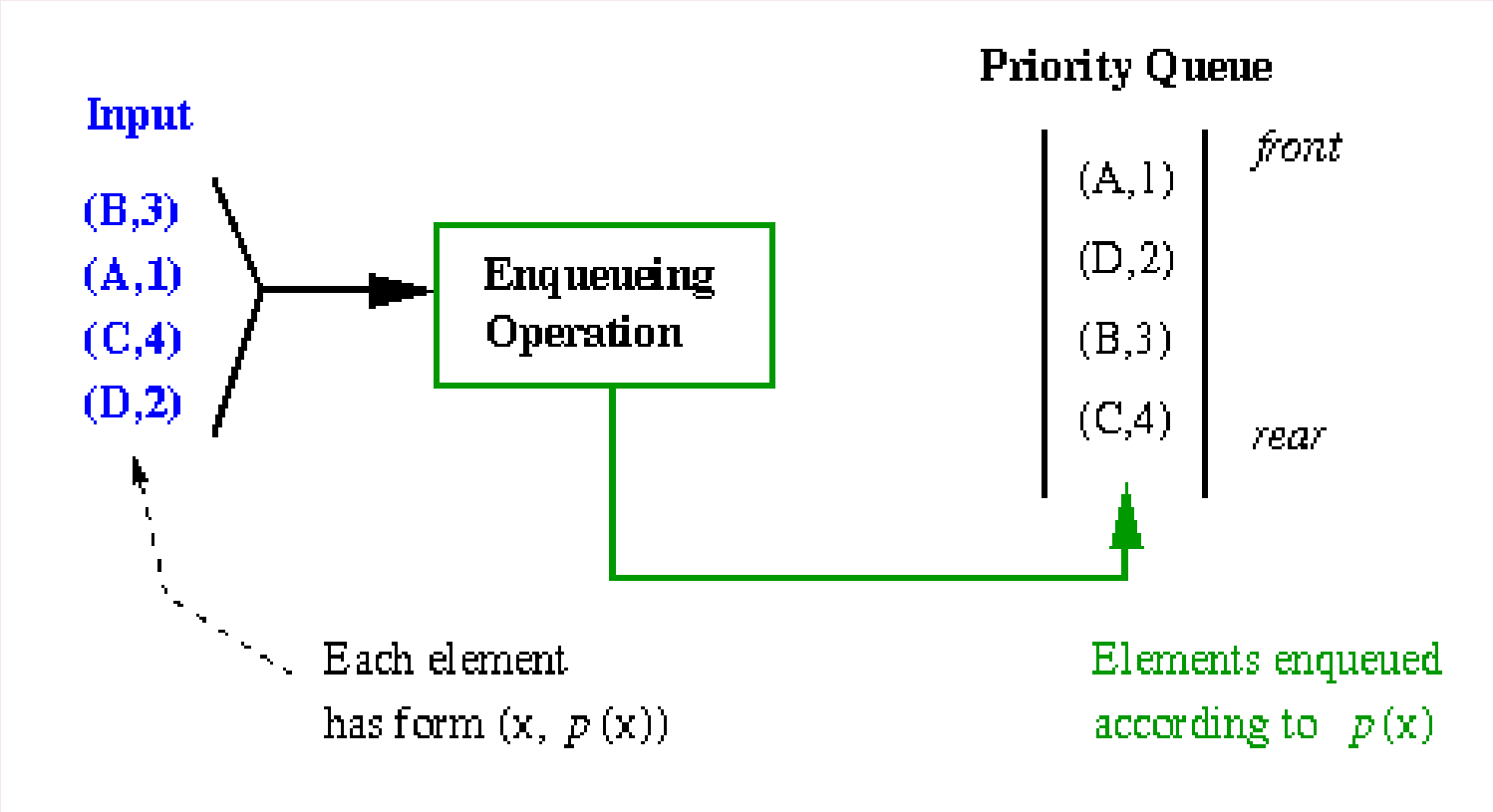
คิววงกลม (Circular Queue)



คิวที่มีอภิสิทธิ์ (Priority Queue)

- คิวที่มีอภิสิทธิ์เป็นคิวที่ประยุกต์ใช้ในระบบปฏิบัติการ โดยหลักในการนำข้อมูลออกเหมือนคิวทั่วไป แต่การนำข้อมูลเข้าจะพิจารณาจากค่าอภิสิทธิ์หรือค่าของข้อมูลเอง ขึ้นอยู่กับการประยุกต์ใช้

คิวที่มีอภิสิทธิ์ (Priority Queue)



การประยุกต์ใช้คิว

- การทำงานของระบบคอมพิวเตอร์ในโลกยุคปัจจุบันจะเป็นการทำงานในลักษณะเครือข่าย และมีการใช้ทรัพยากรร่วมกัน การที่ระบบปฏิบัติการจะจัดสรรทรัพยากรเพื่อให้บริการแก่ผู้ใช้ หรืองานต่าง ๆ อย่างมีประสิทธิภาพ สมดุล และยุติธรรม ชนิดข้อมูลแบบคิวมีความสำคัญมากในการดำเนินงานของระบบปฏิบัติการ ตัวอย่างเช่น ในการใช้ทรัพยากร CPU จากเครื่อง Server หรือการใช้ทรัพยากร System Printer สำหรับผู้ใช้หลายคน ระบบปฏิบัติการจะใช้ชนิดข้อมูลแบบคิวในการจัดระเบียบผู้ใช้ (http://sot.swu.ac.th/cp341/lesson06/ms2t1.htm)

การประยุกต์ใช้คิว