หน่วยที่ 5

โครงสร้างข้อมูลแบบลิงค์ลิสต์

หัวข้อเรื่อง

1. โครงสร้างข้อมูลแบบลิงค์ลิสต์
2. ลิงค์ลิสต์เดี่ยว
3. การสร้างโหนด
4. การแทรกโหนดในลิงค์ลิสต์
5. การท่องเข้าไปในลิงค์ลิสต์
6. การลบโหนดออกจากลิงค์ลิสต์
7. ลิงค์ลิสต์วงกลม
8. ลิงค์ลิสต์คู่

สาระสำคัญ

โครงสร้างข้อมูลแบบลิงค์ลิสต์ เป็นโครงสร้างการจัดเก็บข้อมูลโดยกำหนดลักษณะของหน่วยข้อมูลเป็นโหนดที่เชื่อมโยงกันด้วยพอยน์เตอร์ ภายในโหนดจะแบ่งเป็น 2 ส่วน คือ ส่วนของการจัดเก็บข้อมูล และส่วนที่ใช้ในการจัดเก็บแอดเดรสหรือตำแหน่งของข้อมูลตัวถัดไป โครงสร้างข้อมูลแบบลิงค์ลิสต์จะไม่มีการจองหน่วยความจำไว้ก่อนเหมือนอาร์เรย์ แต่จะจองหน่วยความจำให้กับแต่ละโหนดก็ต่อเมื่อจะทำการเพิ่มโหนดเข้าไปในลิงค์ลิสต์เท่านั้น และเมื่อมีการลบโหนดออกจากลิงค์ลิสต์ก็จะทำการคืนพื้นที่ของโหนดที่ถูกลบกลับคืนสู่หน่วยความจำของระบบ ทำให้พื้นที่ของหน่วยความจำถูกนำมาใช้อย่างคุ้มค่า

จุดประสงค์เชิงพฤติกรรม

1. อธิบายโครงสร้างข้อมูลแบบลิงค์ลิสต์ได้
2. อธิบายลิงค์ลิสต์เดี่ยวได้
3. เขียนคำสั่งการสร้างโหนดได้
4. อธิบายหลักการการแทรกโหนดในลิงค์ลิสต์ได้
5. อธิบายการท่องเข้าไปในลิงค์ลิสต์ได้
6. อธิบายหลักการลบโหนดออกจากลิงค์ลิสต์ได้
7. อธิบายหลักการของลิงค์ลิสต์วงกลมได้
8. อธิบายหลักการของลิงค์ลิสต์คู่ได้
9. บอกความหมายของศัพท์ที่ใช้ในโครงสร้างข้อมูลแบบลิงค์ลิสต์ได้

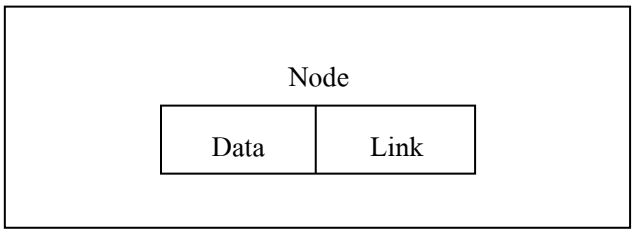
โครงสร้างข้อมูลแบบอาร์เรย์มีข้อจำกัดในเรื่องของการกำหนดขนาดของอาร์เรย์ซึ่งจะต้องกำหนดไว้ล่วงหน้า ถ้าขนาดของอาร์เรย์ที่กำหนดไว้ไม่เพียงพอกับจำนวนข้อมูล ก็จะทำให้ไม่

สามารถจัดเก็บข้อมูลในอาร์เรย์ได้หมด ดังนั้นจึงได้นำเอาโครงสร้างข้อมูลแบบลิงค์ลิสต์ (Linked List Structure) เข้ามาเพื่อแก้ปัญหานี้ เนื่องจากโครงสร้างข้อมูลแบบลิงค์ลิสต์ไม่ต้องทำการจองพื้นที่ไว้ล่วงหน้าเหมือนกับอาร์เรย์ แต่จะทำการจองพื้นที่ก็ต่อเมื่อต้องการเพิ่มข้อมูลเข้าไปในลิงค์ลิสต์เท่านั้น ซึ่งถือว่าเป็นการใช้พื้นที่แบบไดนามิก (Dynamic) คือ เป็นแบบยืดหยุ่น โครงสร้างข้อมูลแบบลิงค์ลิสต์สามารถทำการเพิ่มและลบข้อมูลได้ทันที และพื้นที่เก็บข้อมูลที่ถูกลบไปแล้วสามารถนำกลับมาใช้ได้อีก ทำให้ลิงค์ลิสต์ไม่มีวันเต็มตราบใดที่หน่วยความจำยังมีพื้นที่ให้จัดสรรได้อยู่ ดังนั้นโครงสร้างข้อมูลแบบลิงค์ลิสต์จึงถูกนำไปใช้ในกรณีที่ข้อมูลมีปริมาณไม่แน่นอน

โครงสร้างข้อมูลแบบลิงค์ลิสต์

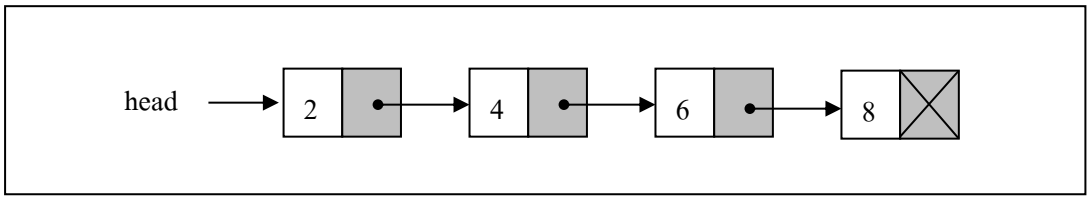
โครงสร้างข้อมูลแบบลิงค์ลิสต์ เป็นการนำเอาข้อมูลแต่ละรายการมาจัดเรียงต่อกันและมีการเชื่อมโยงกันด้วยพอยน์เตอร์ ข้อมูลแต่ละรายการในลิงค์ลิสต์จะเรียกว่า โหนด (Node) แต่ละโหนดจะประกอบด้วย 2 ฟิลด์ คือ

1. ฟิลด์ข้อมูล (Data Field) เป็นส่วนที่ใช้เก็บข้อมูล
2. ฟิลด์ตัวชี้ (Link Field หรือ Pointer) เป็นส่วนที่เก็บแอดเดรส (Address) หรือตำแหน่งของข้อมูลตัวถัดไปในหน่วยความจำหลัก



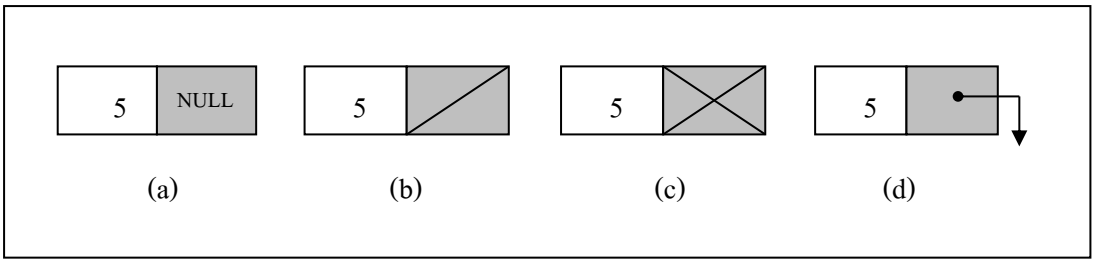
รูปที่ 5.1 โครงสร้างโหนด

ในการเชื่อมโยงไปยังส่วนอื่นๆ ของลิงค์ลิสต์นั้น จะใช้รูปแบบของพอยน์เตอร์เป็นตัวชี้ไปยังโหนดต่อๆ ไป ดังนั้น โครงสร้างข้อมูลแบบลิงค์ลิสต์จึงมีรูปแบบการจัดเก็บข้อมูลที่ประกอบไปด้วยโหนดหลายๆ โหนดมาจัดเรียงต่อกันและทำการเชื่อมโยงโหนดด้วยพอยน์เตอร์ ทำให้โครงสร้างข้อมูลแบบลิงค์ลิสต์มีลักษณะดังรูปที่ 5.2



รูปที่ 5.2 โครงสร้างข้อมูลแบบลิงค์ลิสต์

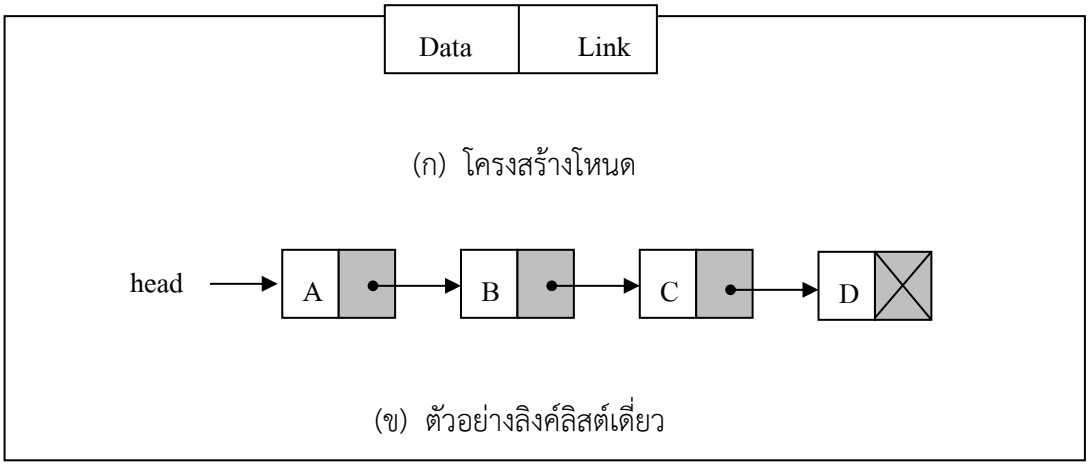
จากรูปที่ 5.2 จะเห็นว่ามี head เป็นพอยน์เตอร์ชี้ที่โหนดแรกของลิงค์ลิสต์ แสดงให้เห็นว่าการเข้าถึงลิงค์ลิสต์จะต้องมีพอยน์เตอร์ชี้ไปยังโหนดแรกของลิงค์ลิสต์เสมอ เพื่อบอกให้ทราบว่าโหนดแรกอยู่ที่ใด เมื่อทราบตำแหน่งของโหนดแรกแล้วจะสามารถหาโหนดถัดไปได้ เนื่องจากแต่ละโหนดจะมีฟิลด์ตัวชี้สำหรับชี้ไปยังโหนดถัดไป สำหรับโหนดสุดท้ายจะไม่มีการชี้ไปยังโหนดอื่นอีก ในฟิลด์ตัวชี้จึงกำหนดให้เป็น NULL เพื่อบอกการสิ้นสุดของลิงค์ลิสต์นั้น สัญลักษณ์ที่บ่งบอกว่าเป็นโหนดสุดท้ายมีหลายแบบ ดังแสดงในรูปที่ 5.3



รูปที่ 5.3 สัญลักษณ์ของโหนดสุดท้าย

ลิงค์ลิสต์เดี่ยว

ลิงค์ลิสต์เดี่ยว (Singly Linked List) เป็นลิงค์ลิสต์ที่โหนดแต่ละโหนดมีโครงสร้างภายในประกอบด้วย ฟิลด์ข้อมูล (Data Field) และฟิลด์ตัวชี้ (Link Field หรือ Pointer) เพียง 1 ฟิลด์ สำหรับชี้ตำแหน่งของโหนดถัดไป จึงทำให้ลิงค์ลิสต์ประเภทนี้มีทิศทางเดียว นั่นคือ การเข้าไปดำเนินการใดๆ กับลิงค์ลิสต์จะต้องเริ่มเดินจากโหนดแรกไปยังโหนดสุดท้ายจะเดินย้อนกลับไม่ได้



รูปที่ 5.4 โครงสร้างของลิงค์ลิสต์เดี่ยวภายในแต่ละโหนดมีฟิลด์ตัวชี้ (Link Field) เพียง 1 ฟิลด์

จากรูปที่ 5.4 (ข) เป็นลิงค์ลิสต์เดี่ยวที่มี head เป็นพอยน์เตอร์ชี้อยู่ที่โหนดแรก และโหนดแต่ละโหนดจะเชื่อมโยงถึงกันด้วยพอยน์เตอร์ ซึ่งถ้าไม่มีพอยน์เตอร์เชื่อมก็จะทำให้โหนดขาดออกจากกัน ดังนั้นในโปรแกรมการสร้างลิงค์ลิสต์เราจะต้องทำการกำหนดโครงสร้างโหนดขึ้นมาก่อน

โดยกำหนดเป็นแบบเรคอร์ดเพื่อใช้เก็บข้อมูลและพอยน์เตอร์เอาไว้ในชุดเดียวกัน และพอยน์เตอร์ที่ใช้ในการชี้ตำแหน่งของโหนดจะต้องเป็นพอยน์เตอร์ชนิดเรคอร์ด

การประกาศตัวแปรที่มีโครงสร้างแบบลิงค์ลิสต์เดี่ยวด้วยภาษาซี

```
#include <stdio.h>
 struct node
    {
          int data;
          node *next;
     };
    node  *head;
```

การประกาศตัวแปรข้างต้นได้กำหนดตัวแปร head เป็นตัวแปรพอยน์เตอร์ชนิดเรคอร์ด ซึ่งจะประกอบด้วยฟิลด์ 2 ฟิลด์ คือ

1. ฟิลด์ข้อมูล (Data Field) ในที่นี้คือ data
2. ฟิลด์ตัวชี้ (Link Field หรือ Pointer) ในที่นี้คือ next

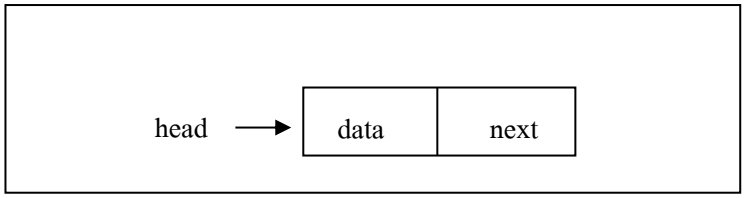
ในโปรแกรม เราจะอ้างชื่อตัวแปร head เพื่อจัดการกับข้อมูลในลิงค์ลิสต์

การสร้างโหนด

การสร้างโหนด จะใช้คำสั่ง new สำหรับจองหน่วยความจำ ดังนี้

```
head = new node;
```

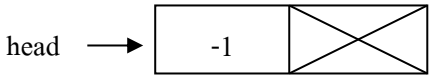
ผลคือ จะได้โหนดใหม่ที่มีพอยน์เตอร์ head ชี้อยู่ ซึ่งโหนดใหม่นี้จะถูกเรียกว่าโหนด head แสดงได้ดังรูปที่ 5.5



รูปที่ 5.5 โหนด head

จากรูปที่ 5.5 โหนด head ประกอบด้วยฟิลด์ 2 ฟิลด์ คือฟิลด์ data และฟิลด์ next การกำหนดค่าให้แต่ละฟิลด์ ของโหนด head ทำได้ดังนี้

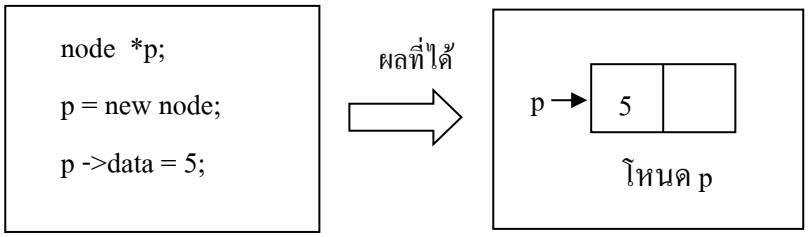
```
head -> data = -1;     /*  นำ -1 ไปเก็บในฟิลด์ data  */
head -> next = NU LL;   /* นำค่า  NULL  ไปเก็บในฟิลด์  next  */
```



การกำหนดให้ head -> data เป็น -1 เพื่อเป็นการบอกว่า head คือโหนดแรกนั่นเอง และกำหนดให้พอยน์เตอร์ของโหนด head เก็บค่า NULL เพื่อบอกให้รู้ว่า ขณะนี้ยังไม่มีโหนดใดๆ ในลิงค์ลิสต์ หรืออาจกำหนดให้ head = NULL เลยก็ได้กรณีที่ยังไม่มีโหนดในลิงค์ลิสต์ และเมื่อต้องการสร้างโหนดใหม่เพื่อเพิ่มข้อมูลในลิงค์ลิสต์ สามารถทำได้ตามขั้นตอนดังนี้

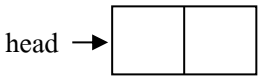
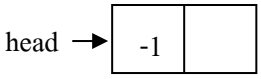
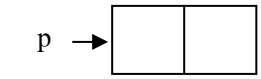
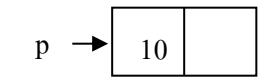
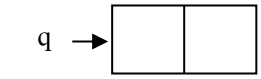
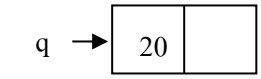
1. กำหนดตัวแปรพอยน์เตอร์เป็นชนิดเรคอร์ด
2. ใช้คำสั่ง new เพื่อสร้างโหนดใหม่และจองหน่วยความจำให้กับโหนดนั้น
3. เก็บค่าที่ต้องการเอาไว้ในฟิลด์ข้อมูลของโหนด

จากขั้นตอนข้างต้นสามารถนำมาเขียนชุดคำสั่งได้ดังนี้

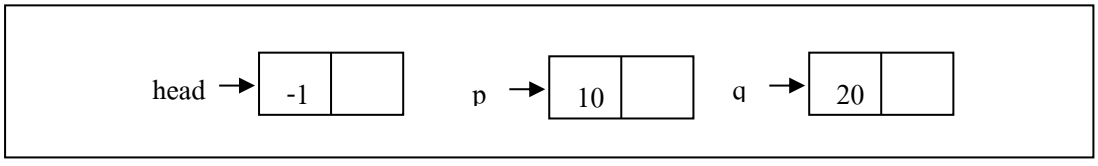


รูปที่ 5.6 ภาพโหนด p ซึ่งเป็นผลลัพธ์จากการใช้คำสั่ง

ตัวอย่างที่ 5.1 การสร้างโหนด

คำสั่ง	**ภาพของหน่วยความจำ**
node *head;	
head = new node;	
head ->data = -1;	
node *p;	
p = new node;	
p ->data = 10;	
node *q;	
q = new node;	
q ->data = 20;	

จากตัวอย่างที่ 5.1 เป็นการใช้คำสั่ง new สร้างโหนดใหม่ขึ้นมา 3 โหนด คือ โหนด head โหนด p และ โหนด q (จะเรียกชื่อโหนดตามชื่อพอยน์เตอร์ที่ชี้) เมื่อนำข้อมูลไปเก็บแต่ละโหนดแล้วตอนนี้จะมีโหนดในหน่วยความจำดังรูปที่ 5.7



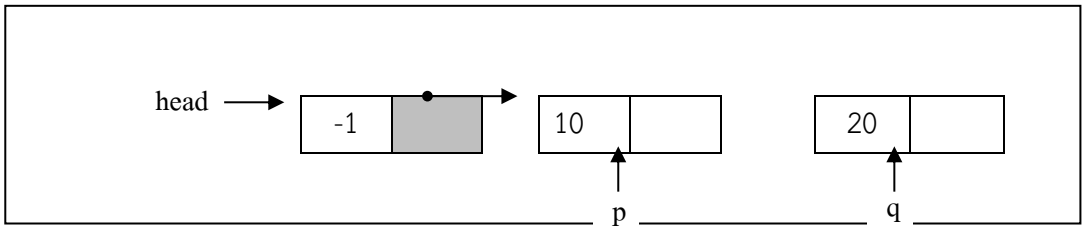
รูปที่ 5.7 ภาพรวมของโหนดในหน่วยความจำ

วิธีการเชื่อมโยงโหนดเข้าด้วยกัน

วิธีการเชื่อมโยงโหนดเข้าด้วยกัน จะใช้คำสั่งดังนี้

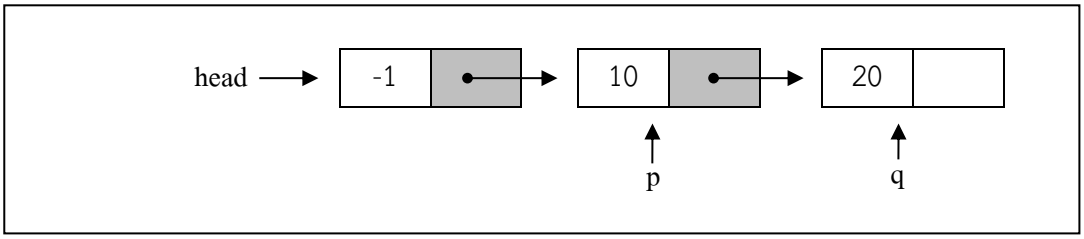
```
head->next = p;
```

มีผลให้โหนด head และโหนด p เชื่อมต่อกัน แต่โหนด q ยังไม่ได้เชื่อมกับโหนดใด ดังรูปที่ 5.8



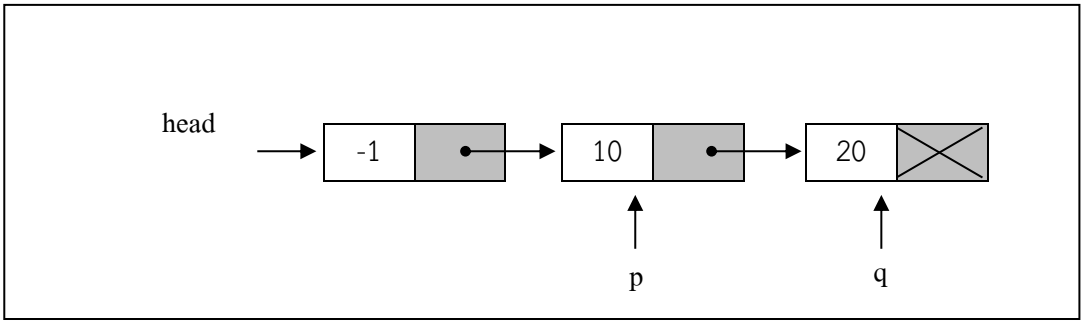
รูปที่ 5.8 แสดงการเชื่อมโหนด head และโหนด p เข้าด้วยกัน

และถ้าใช้คำสั่ง p->next = q จะมีผลให้โหนด p เชื่อมต่อกับโหนด q เกิดเป็นลิงค์ลิสต์ที่เชื่อมโยงสามโหนดเข้าด้วยกันดังรูปที่ 5.9



รูปที่ 5.9 ลิงค์ลิสต์ที่เกิดจากการเชื่อมโหนดสามโหนดเข้าด้วยกัน

จากรูปที่ 5.9 โหนด q เป็นโหนดสุดท้ายในลิงค์ลิสต์ ดังนั้นจะต้องกำหนดให้ฟิลด์ next ของโหนด q มีค่าเป็น NULL โดยใช้คำสั่ง q->next = NULL ผลคือจะได้ลิงค์ลิสต์เดี่ยวที่สมบูรณ์ดังรูปที่ 5.10



รูปที่ 5.10 ลิงค์ลิสต์เดี่ยวที่ประกอบด้วยโหนดสามโหนด

การแทรกโหนดในลิงค์ลิสต์

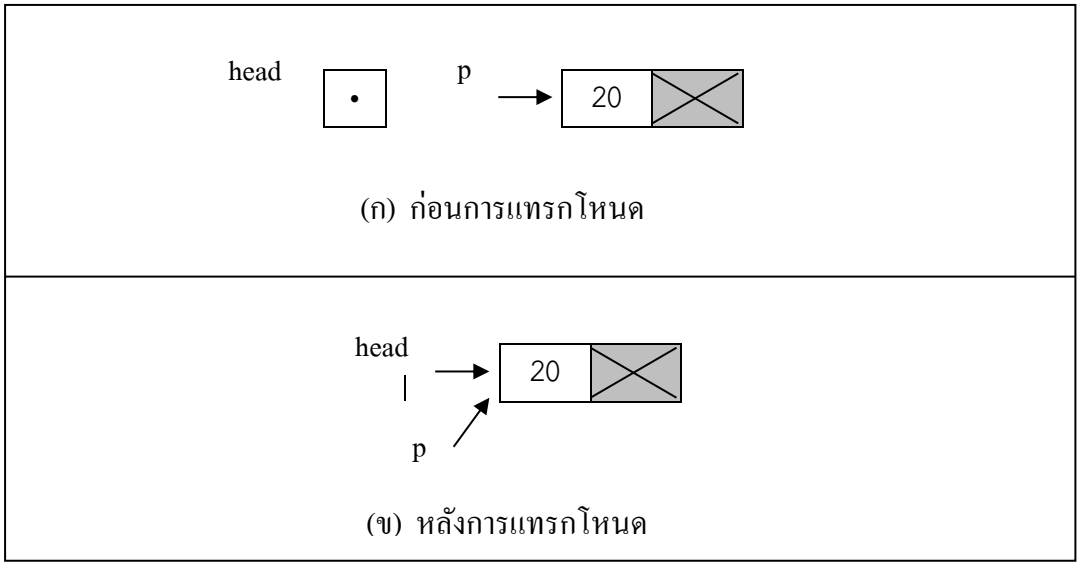
การแทรกโหนดในลิงค์ลิสต์ (Insert Node) เป็นการเพิ่มโหนดใหม่เข้าไปในลิงค์ลิสต์ ซึ่งสามารถทำได้ 4 รูปแบบ ดังนี้

1. การแทรกโหนดในลิงค์ลิสต์ว่าง
2. การแทรกโหนดที่ตำแหน่งแรกของลิงค์ลิสต์
3. การแทรกโหนดตรงส่วนกลางของลิงค์ลิสต์
4. การแทรกโหนดที่ท้ายลิงค์ลิสต์

1. การแทรกโหนดในลิงค์ลิสต์ว่าง

การแทรกโหนดในลิงค์ลิสต์ว่าง เป็นการแทรกโหนดเพิ่มเข้าไปในลิงค์ลิสต์ในขณะ ที่ลิงค์ลิสต์ว่างเปล่า นั่นหมายถึงเป็นการแทรกโหนดแรกเข้าไป ซึ่งขณะนั้นพอยน์เตอร์ head มีค่าเป็น NULL เนื่องจากเป็นลิงค์ลิสต์ว่าง หลังจากแทรกโหนดแรกเข้าไปพอยน์เตอร์ head จะมาชี้ที่โหนดแรกนี้ สามารถเขียนชุดคำสั่งได้ดังนี้

```
head = NULL;      /* ในขณะที่ลิงค์ลิสต์ว่างให้พอยน์เตอร์ head มีค่าเป็น NULL */
p = new node;     /* สร้างโหนดใหม่โดยให้พอยน์เตอร์ p ชี้โหนดใหม่ */
p ->data = 20;    /* นำข้อมูลไปเก็บในฟิลด์ data ของโหนด p */
p->next = head;   /* ให้ฟิลด์ next เก็บค่า NULL เนื่องยังไม่มีการเชื่อมต่อโหนด*/
head = p;         /* ให้ head มาชี้ที่โหนด p ซึ่งเป็นโหนดแรกในลิงค์ลิสต์ */
```

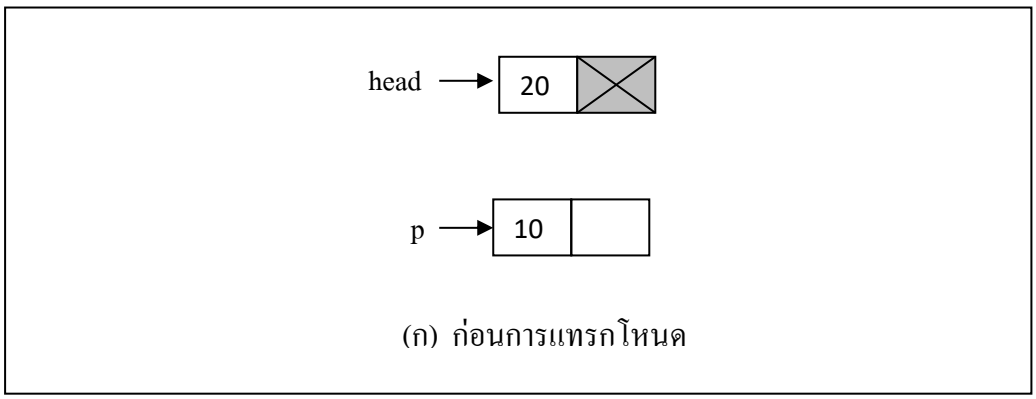


รูปที่ 5.11 การแทรกโหนดในลิงค์ลิสต์ว่าง

2. การแทรกโหนดที่ตำแหน่งแรกของลิงค์ลิสต์

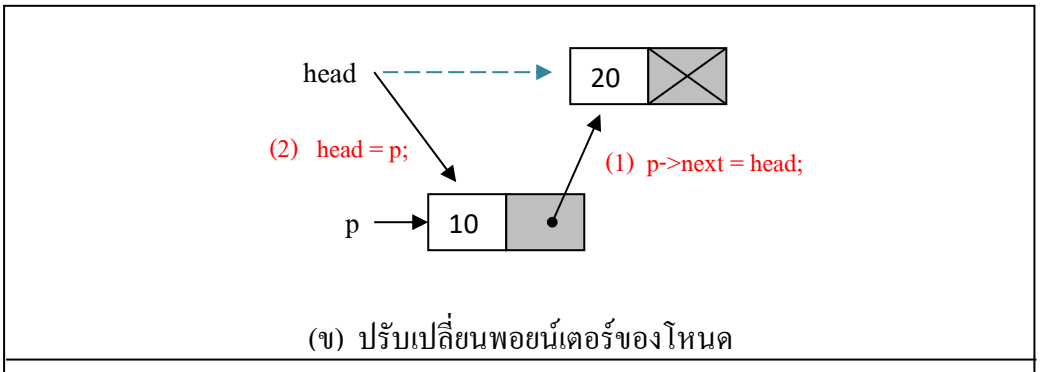
การแทรกโหนดที่ตำแหน่งแรกของลิงค์ลิสต์ จะมีผลทำให้โหนดซึ่งเคยอยู่ลำดับแรกเลื่อนมาต่อท้ายโหนดใหม่ที่แรกเข้าไป สามารถเขียนชุดคำสั่งได้ดังนี้

```
p = new node;     /* สร้างโหนดใหม่โดยให้พอยน์เตอร์ p ชี้โหนดใหม่ */
p->data = 10;     /* นำข้อมูลไปเก็บในโหนด */
p->next = head;   /* ให้พอยน์เตอร์ของโหนด p ชี้ไปยังโหนดที่ head ชี้อยู่ */
head = p;         /* เปลี่ยนให้ head มาชี้ที่โหนดใหม่ */
```

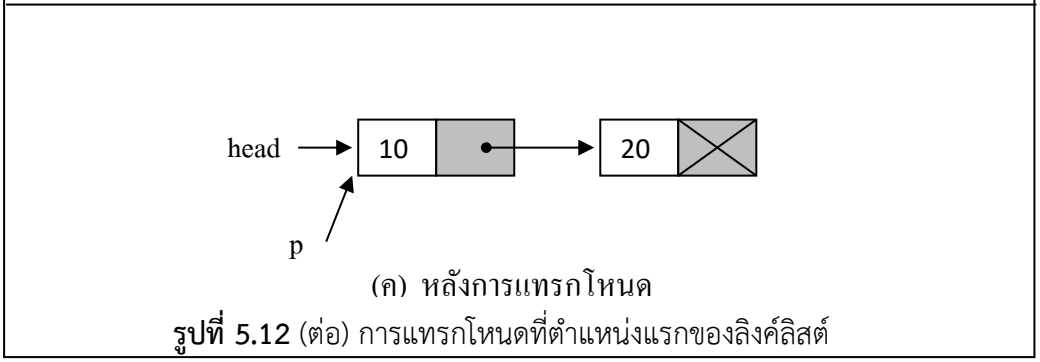


(ก) ก่อนการแทรกโหนด

รูปที่ 5.12 การแทรกโหนดที่ตำแหน่งแรกของลิงค์ลิสต์



(ข) ปรับเปลี่ยนพอยน์เตอร์ของโหนด



(ค) หลังการแทรกโหนด

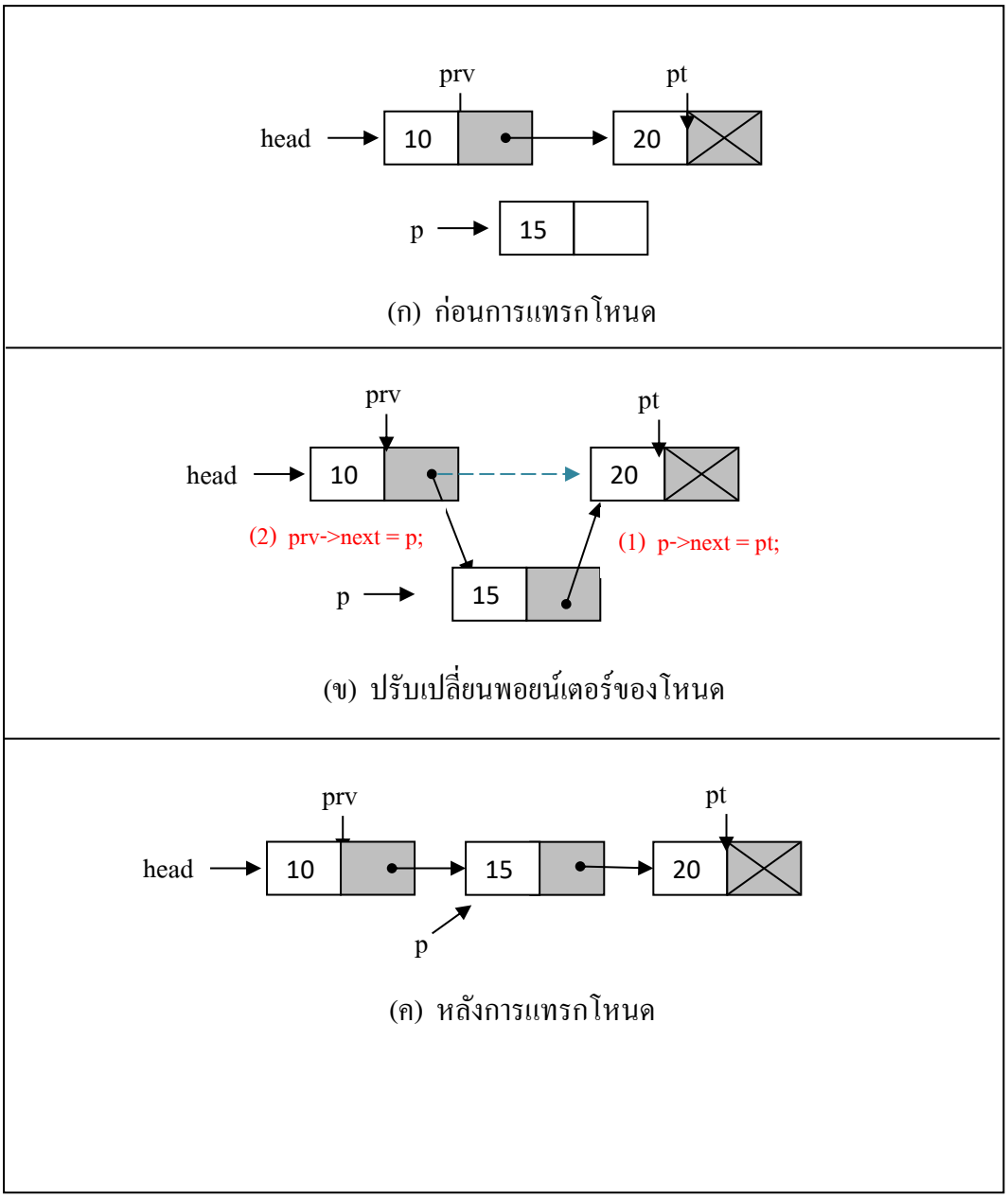
รูปที่ 5.12 (ต่อ) การแทรกโหนดที่ตำแหน่งแรกของลิงค์ลิสต์

3. การแทรกโหนดตรงส่วนกลางของลิงค์ลิสต์

การแทรกโหนดตรงส่วนกลางของลิงค์ลิสต์ เป็นการเอาโหนดใหม่เข้าไปแทรกระหว่างโหนดสองโหนดที่อยู่ในลิงค์ลิสต์ โดยเริ่มต้นจะต้องรู้ตำแหน่งโหนดก่อนหน้าที่จะแทรกเสียก่อน

เพราะโหนดก่อนหน้าจะมีฟิลด์ next เป็นพอยน์เตอร์ที่ใช้เชื่อมโยงไปยังโหนดถัดไป ดังนั้นวิธีการนี้จึงต้องมีพอยน์เตอร์เพิ่มมาอีก 2 ตัว คือ prv และ pt สำหรับท่องเข้าไปในลิงค์ลิสต์เพื่อหาตำแหน่งของโหนดสองโหนดก่อนที่จะแทรกโหนดใหม่ซึ่งมี p ชี้อยู่ระหว่างสองโหนดนั้น สามารถเขียนชุดคำสั่งได้ดังนี้

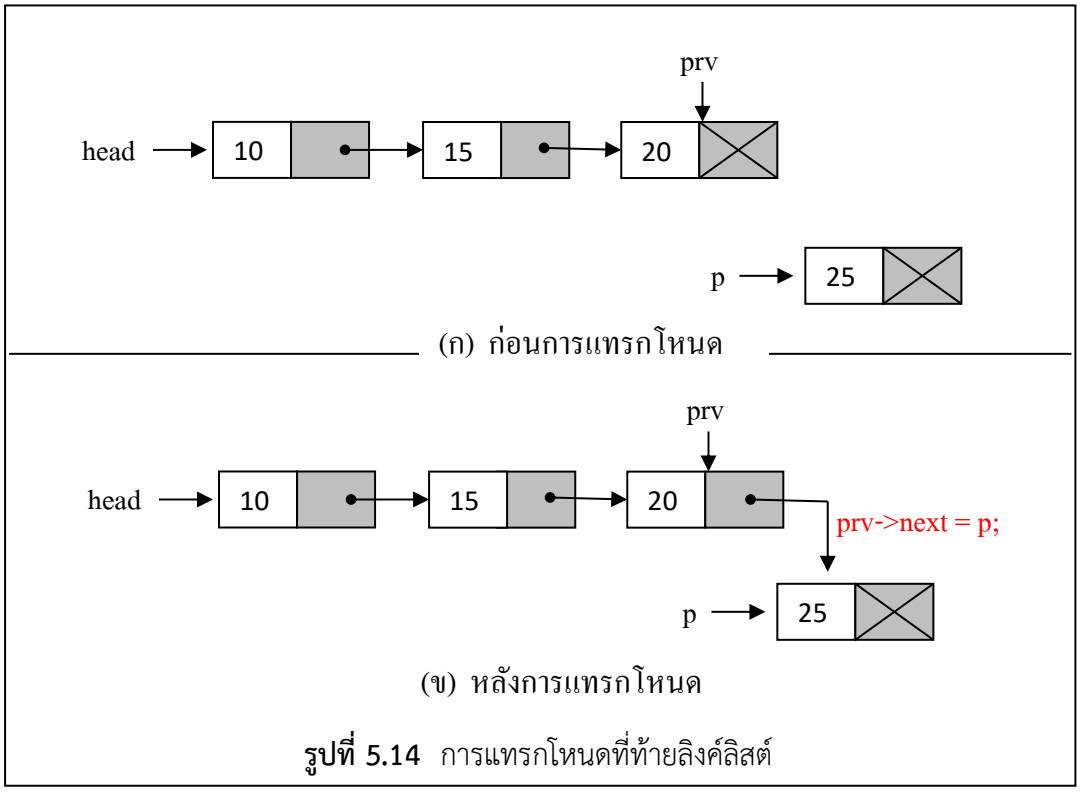
```
p = new node;          /* สร้างโหนดใหม่โดยให้พอยน์เตอร์ p ชี้โหนดใหม่ */
p->data = 15;          /* นำข้อมูลไปเก็บในฟิลด์ data ของโหนด p */
p->next = pt;          /* ให้พอยน์เตอร์ของโหนด p ชี้ไปยังโหนดที่พอยน์เตอร์
pt ชี้อยู่ */
prv->next = p;         /* เปลี่ยนให้พอยน์เตอร์ของโหนด prv มาชี้ที่โหนดใหม่
*/
```



4. การแทรกโหนดที่ท้ายลิงค์ลิสต์

การแทรกโหนดที่ท้ายลิงค์ลิสต์ จำเป็นต้องมีพอยน์เตอร์ prv ท่องเข้าไปที่ปลายสุดของลิงค์ลิสต์ เพื่อให้พอยน์เตอร์ prv เข้าไปชี้อยู่ที่โหนดสุดท้าย หลังจากนั้นจึงทำการเชื่อมโยงโหนดสุดท้ายกับโหนดใหม่เข้าด้วยกัน สามารถเขียนขั้นตอนของคำสั่งดังนี้

```
p = new node;          /* สร้างโหนดใหม่โดยให้พอยน์เตอร์ p ชี้โหนดใหม่ */
  p->data = 25;        /* นำข้อมูลไปเก็บในฟิลด์ data ของโหนด p */
p->next = NULL;        /* ให้พอยน์เตอร์ของโหนด p ชี้ไปยัง NULL */
prv->next = p;         /* เปลี่ยนให้พอยน์เตอร์ของโหนด prv มาชี้ที่โหนดใหม่
*/
```



รูปที่ 5.14 การแทรกโหนดที่ท้ายลิงค์ลิสต์

จากรายละเอียดการแทรกโหนดเข้าไปในลิงค์ลิสต์ในรูปแบบต่างๆ ไม่ว่าจะเป็นการแทรกในขณะที่ลิงค์ลิสต์ว่าง การแทรกที่ตำแหน่งแรก การแทรกตรงกลาง และการแทรกที่ท้ายลิงค์ลิสต์จะใช้คำสั่งไม่เหมือนกัน โปรแกรมที่เขียนขึ้นจึงจำเป็นต้องมีเงื่อนไขเพื่อใช้ตรวจสอบให้ได้ว่าเป็นการแทรกลักษณะใด และเพื่อให้เข้าใจขั้นตอนการทำงาน การแทรกโหนดในลิงค์ลิสต์ยิ่งขึ้น ขอให้พิจารณาโปรแกรมที่ 5.1 การแทรกโหนดในลิงค์ลิสต์เพื่อให้ลิงค์ลิสต์เรียงลำดับ

โปรแกรมที่ 5.1 การแทรกโหนดในลิงค์ลิสต์เพื่อให้ลิงค์ลิสต์เรียงลำดับ

```
#include  <stdio.h>
#include <conio.h>
struct node
{
   int data;
   node *next;
};
node  *head;

void addnode(int x)
{
    node  *p,*prv,*pt;
     p = new node;
     p->data = x;
     p->next = NULL;
     prv = NULL;
     pt = head;
     while ((pt != NULL)&&(pt->data<
x))
     {
           prv = pt;
           pt = pt->next;
     }
     if (prv == NULL)
     {
            p->next = pt;
            head = p;
      }
      else
       {
           p->next = pt;
           prv->next = p;
       }
}

void main()
{
  clrscr();
  addnode(5);
  addnode(6);
  addnode(4);
  addnode(10);
  addnode(8);

   printf("Data  in  List \n ");
  node *p;
  for (p=head;p!=NULL;p=p->next)
           printf("%d  ",p->data);

 getch();

  }
```

ผลรัน

```
Data  in  List
4  5  6  8  10
```

การท่องเข้าไปในลิงค์ลิสต์

การท่องเข้าไปในลิงค์ลิสต์ หมายถึง การเข้าถึงโหนดทีละโหนดที่อยู่ในลิงค์ลิสต์นั้นโดยเริ่มต้นตั้งแต่โหนดแรกไปจนถึงโหนดสุดท้าย โดยทั่วไปเมื่อมีการจัดการกับลิงค์ลิสต์เรียบร้อยแล้วหากต้องการพิมพ์ค่าที่เก็บในลิงค์ลิสต์ทั้งหมดออกมา เราสามารถท่องเข้าไปในลิงค์ลิสต์เพื่อพิมพ์ค่าที่เก็บในโหนดทีละโหนดได้ หรือถ้าจะหาว่ามีข่าวสารที่ต้องการอยู่ในลิงค์ลิสต์นี้หรือไม่ เราก็สามารถรู้ได้โดยการค้นหาแบบลำดับ (Sequential Search) โดยตั้งต้นจาก head และค้นหาไปเรื่อยๆ ตามลำดับ หรือเป็นการท่องเข้าไปเพื่อหาผลรวมของคะแนนดิบของนักศึกษาทั้งหมดแล้วนำมาคิดเป็นคะแนนเฉลี่ย เป็นต้น

ในการท่องเข้าไปในลิงค์ลิสต์จะต้องทำการกำหนดพอยน์เตอร์ขึ้นมาเพื่อท่องเข้าไป ซึ่งเราจะไม่ใช้ head เนื่องจากถ้าเลื่อน head แล้วจะทำให้กลับไปที่ต้นลิงค์ลิสต์ไม่ได้ และจะมีการกำหนดลูปเพื่อให้พอยน์เตอร์เคลื่อนที่ไปยังโหนดถัดไป ซึ่งสามารถเขียนคำสั่งการท่องเข้าไปในลิงค์ลิสต์ได้ดังนี้

```
p = head;     /* ให้พอยน์เตอร์ p ชี้โหนดแรก  */
while (p->next != NULL)  /* วนลูปในขณะที่พอยน์เตอร์ของโหนด p ยังไม่เท่ากับ NULL  */
        p = p->next;   /* เปลี่ยนให้พอยน์เตอร์ p  ชี้ไปยังโหนดถัดไป */
```

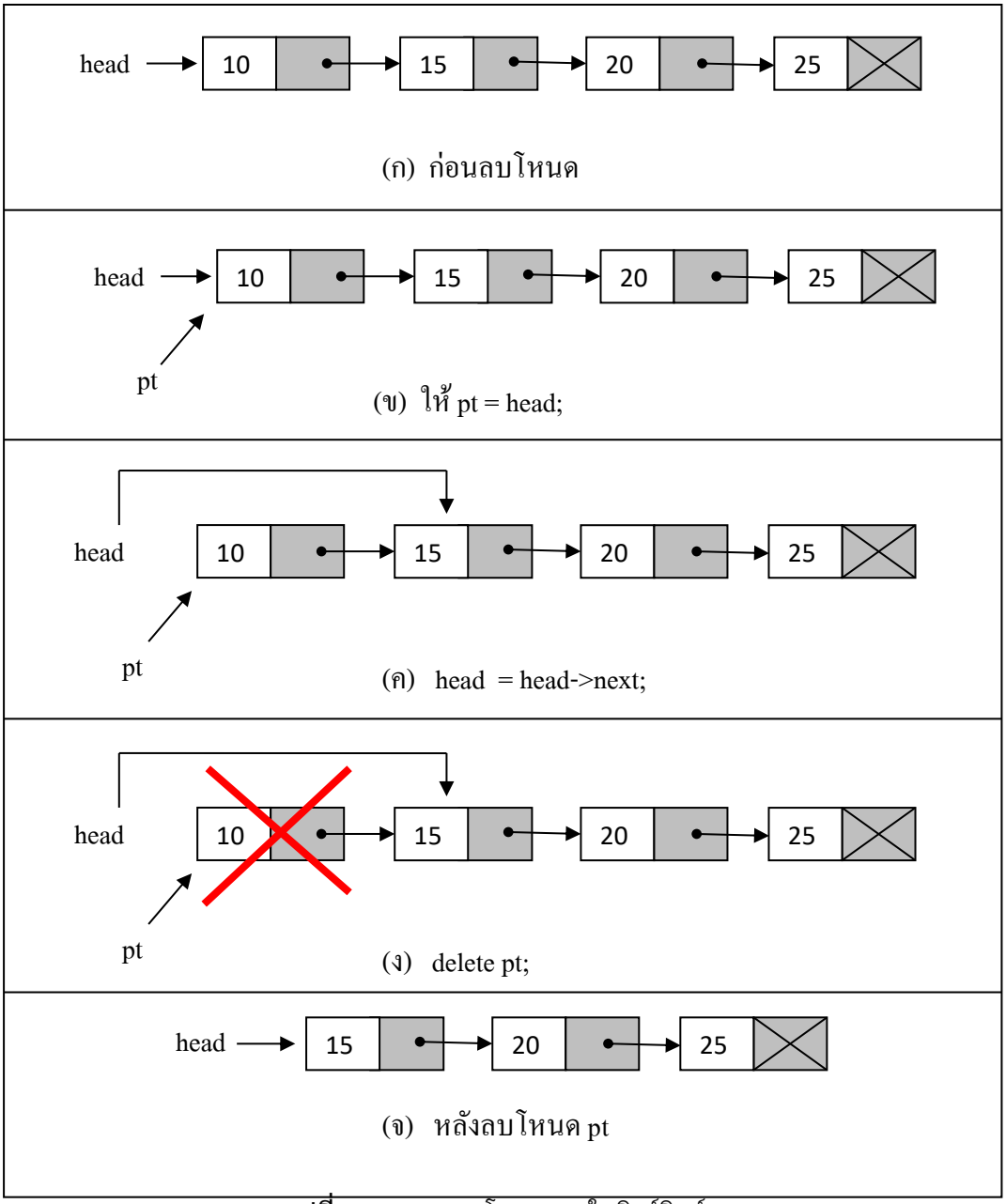
การลบโหนดออกจากลิงค์ลิสต์

การลบโหนดออกจากลิงค์ลิสต์ จะเป็นการลบทั้งโหนดและข้อมูลออกไป และพื้นที่หน่วยความจำของโหนดที่ถูกลบจะถูกส่งคืนแก่หน่วยความจำระบบเพื่อให้นำไปใช้งานอื่นต่อไป ในการลบโหนดออกจากลิงค์ลิสต์มีหลักการง่ายๆ คือ ทำการค้นหาโหนดที่ต้องการลบ เมื่อพบแล้วให้ทำการเปลี่ยนพอยน์เตอร์ของโหนดก่อนหน้าโหนดที่ต้องการลบให้ชี้ไปยังพอยน์เตอร์ของโหนดที่ต้องการลบ และคืนค่าโหนดที่ต้องการลบแก่หน่วยความจำระบบ การลบโหนดออกจากลิงค์ลิสต์มีรายละเอียดดังนี้

การลบโหนดแรกในลิงค์ลิสต์

การลบโหนดแรกในลิงค์ลิสต์ให้ทำการย้าย head ให้ชี้ไปยังโหนดที่อยู่ถัดจากโหนดแรกก่อนแล้วทำการลบโหนดแรก โดยใช้ชุดคำสั่งดังนี้

```
pt = head;                  /* ให้พอยน์เตอร์ pt ชี้ไปยังโหนดที่พอยน์เตอร์ head ชี้อยู่
*/
head = head->next;     /* ให้พอยน์เตอร์ head ชี้ไปยังโหนดถัดไป  */
delete pt;                  /*  ลบโหนด pt  */
```



รูปที่ 5.15 การลบโหนดแรกในลิงค์ลิสต์

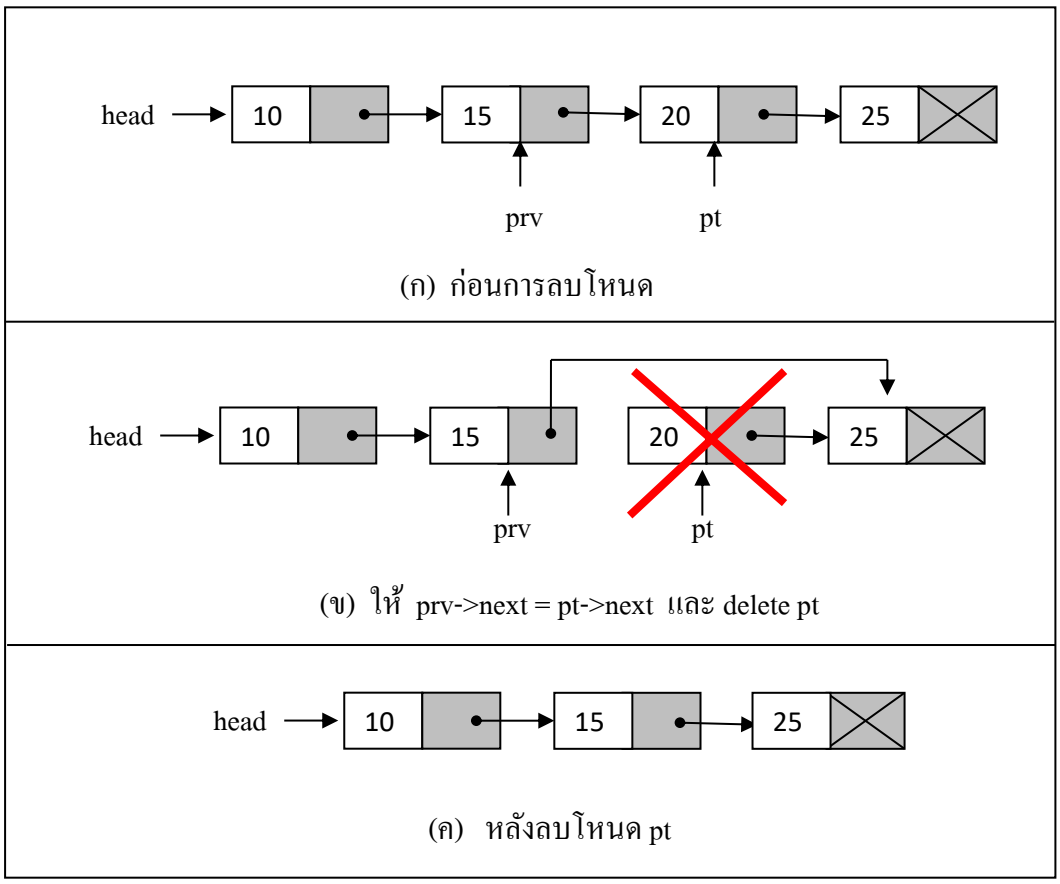
อย่างไรก็ตามถ้าโหนดแรกนั้นเป็นเพียงโหนดเดียวที่อยู่ในลิงค์ลิสต์ หลังจากที่ถูกลบออกไปพอยน์เตอร์ head ก็จะถูกกำหนดให้เป็น NULL ซึ่งถือว่าเป็นลิงลิสต์ว่างไปโดยปริยาย

การลบโหนดโดยทั่วไป

การลบโหนดโดยทั่วไป จะประกอบด้วย การลบโหนดที่อยู่ภายในลิงค์ลิสต์และการลบโหนดสุดท้ายในลิงค์ลิสต์ โดยทั้งสองกรณีสามารถใช้ชุดคำสั่งเดียวกันได้ ในการลบโหนดขั้นตอนแรกจะต้องรู้ตำแหน่งโหนดที่จะลบเสียก่อน ซึ่งจะใช้วิธีการให้พอยน์เตอร์ คือ prv และ pt ท่องเข้าไปในลิงค์ลิสต์ตามติดกันไปเพื่อหาโหนดที่ต้องการลบ เมื่อพบแล้วก็จะทำการปรับเปลี่ยนพอยน์เตอร์ที่ชี้โหนดเสียใหม่

กรณีการลบโหนดที่อยู่ภายในลิงค์ลิสต์ สามารถเขียนขั้นตอนคำสั่งได้ดังนี้

```
prv->next =pt->next;  /* ให้พอยน์เตอร์ของโหนด prv ชี้ไปยังโหนดที่พอยน์เตอร์ของ
                         โหนด pt ชี้อยู่ */
delete pt;            /* ลบโหนด pt */
```

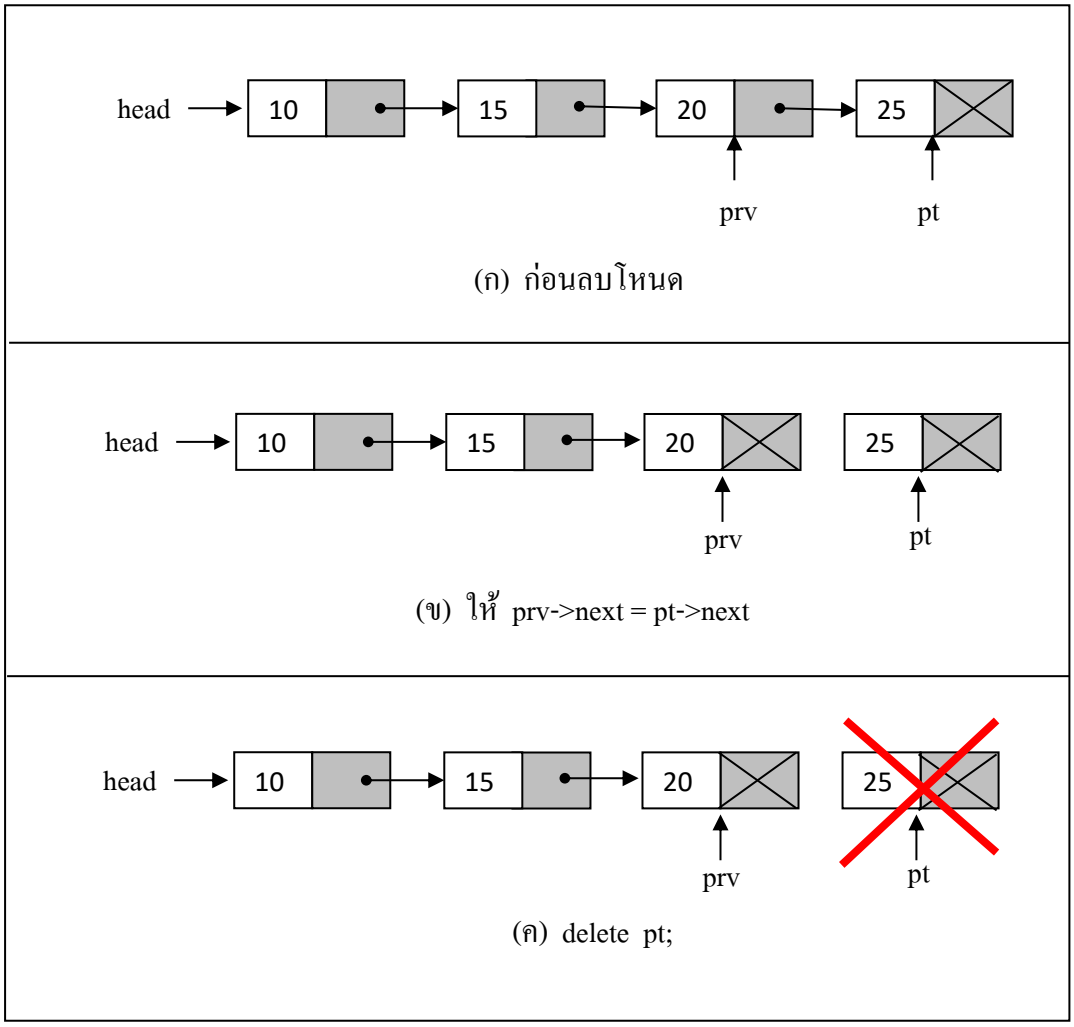


รูปที่ 5.16 การลบโหนดที่อยู่ภายในลิงค์ลิสต์

กรณีการลบโหนดสุดท้ายในลิงค์ลิสต์ สามารถเขียนขั้นตอนคำสั่งได้ดังนี้

```
prv->next =pt->next;  /* ให้พอยน์เตอร์ของโหนด prv ชี้ไปยังโหนดที่พอยน์เตอร์
                         ของโหนด pt ชี้อยู่ */
```

```
delete  pt;                    /*  ลบโหนด pt   */
```



รูปที่ 5.17 การลบโหนดสุดท้ายในลิงค์ลิสต์

โปรแกรมที่ 5.2 การแทรกโหนดและการลบโหนดในลิงค์ลิสค์

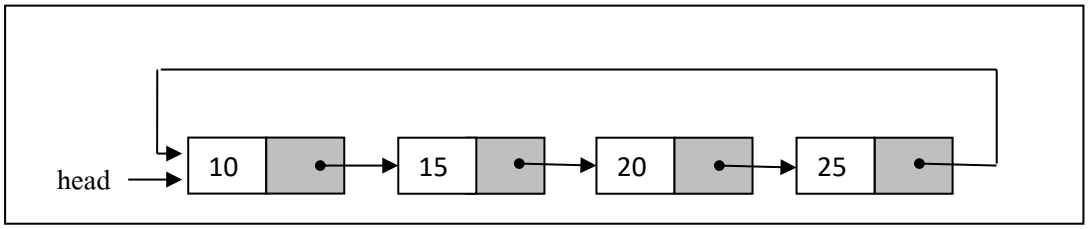
```
#include  <stdio.h>
#include <conio.h>
struct node
{
   int data;
   node *next;
};
node  *head;

void addnode(int x)
{
   node  *p,*prv,*pt;

     p = new node;
     p->data = x;
     p->next = NULL;
     prv = NULL;
     pt = head;
     while ((pt != NULL)&&(pt->data< x))
     {
           prv = pt;
           pt = pt->next;
     }
     if (prv == NULL)
     {
            p->next = pt;
            head = p;
     }
     else
     {
            p->next = pt;
            prv->next = p;
     }
}

void  removenode(int deldata)
{
   node  *pt, *prv;

   if (head == NULL)
       printf("list is Empty\n");
   else
   {
       prv = NULL;
       pt  = head;
       while ((pt != NULL)&&(pt->data !=
deldata))
       {
            prv = pt;
            pt  = pt->next;
       }
       if (pt != NULL)
       {
            if (prv == NULL)
                 head = head->next;
```

โปรแกรม (ต่อ)

```
            else
                prv->next = pt->next;
            printf("data = %d\n",pt->data);
            delete pt;

        }
        else
            printf("No Data \n");
    }
}

void main()
{
  clrscr();
  addnode(5);
  addnode(6);
  addnode(4);
  addnode(10);
  addnode(8);
  printf("Insert Nodes \n  ");
  printf("Data in List =>  ");
  node *p;
  for (p=head;p!=NULL;p=p->next)
     printf("%d  ",p->data);

  printf("\n\n Delete Nodes\n",p->data);
  removenode(4);
  removenode(10);
  removenode(3);
  removenode(6);

  printf("\nCurrent Data in List\n");
  for (p=head;p!=NULL;p=p->next)
     printf("data = %d\n",p->data);
  getch();
}
```

ลิงค์ลิสต์แบบวงกลม

ลิงค์ลิสต์แบบวงกลม (Circular Linked List) จะใช้วิธีการเชื่อมโยงฟิลด์ next ของโหนดสุดท้ายซึ่งเดิมเก็บค่า NULL ให้เปลี่ยนไปชี้ยังโหนดแรก และด้วยการเชื่อมโยงดังกล่าวจึงทำให้ลิงค์ลิสต์แบบวงกลมสามารถเข้าถึงทุกๆ โหนดในลิงค์ลิสต์ได้โดยไม่จำเป็นต้องเริ่มต้นจากโหนดแรกของลิงค์ลิสต์เสมอไป แต่สามารถเริ่มต้นท่องจากโหนดใดๆ ในลิงค์ลิสต์ก็ได้

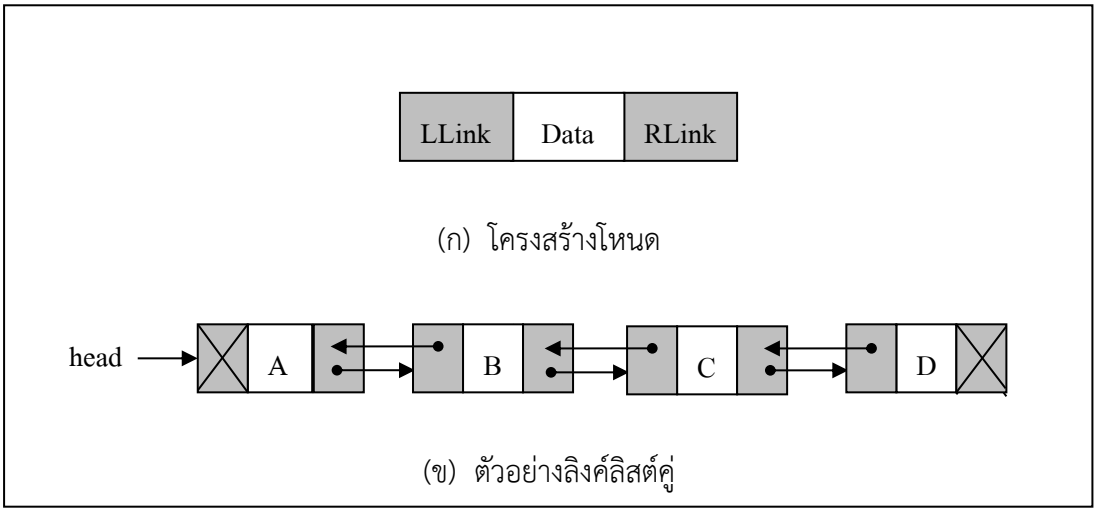


รูปที่ 5.18 ตัวอย่างลิงค์ลิสต์แบบวงกลม

ในการแทรกโหนดหรือลบโหนดออกจากลิงค์ลิสต์วงกลมจะมีรูปแบบเช่นเดียวกันกับลิงค์ลิสต์เดี่ยว ยกเว้นเพียงแต่พอยน์เตอร์ของโหนดสุดท้ายเท่านั้นที่จะชี้ไปยังโหนดแรก สำหรับโหนดสุดท้ายที่ได้รับการแทรกหรือลบออกไปก็จะต้องปรับเปลี่ยนพอยน์เตอร์ใหม่เพื่อเชื่อมโยงฟิลด์ไปยังโหนดแรกเสมอ

ลิงค์ลิสต์คู่

ลิงค์ลิสต์คู่ (Doubly Linked List) เป็นลิงค์ลิสต์ที่ภายในแต่ละโหนดประกอบด้วยฟิลด์ 3 ฟิลด์ คือ ฟิลด์ข้อมูล (Data Field) และ ฟิลด์ตัวชี้ (Link Field) จำนวน 2 ฟิลด์ สำหรับชี้ตำแหน่งของโหนดถัดไปและชี้ตำแหน่งของโหนดก่อนหน้า ทำให้ลิงค์ลิสต์ประเภทนี้มี 2 ทิศทาง นั่นคือสามารถเดินไปข้างหน้าและเดินย้อนกลับได้ หรือเรียกว่า เดินไปได้ทั้งซ้ายและขวา ดังแสดงในรูปที่ 5.19



รูปที่ 5.19 โครงสร้างของลิงค์ลิสต์คู่ภายในแต่ละโหนดมีฟิลด์ตัวชี้ 2 ฟิลด์

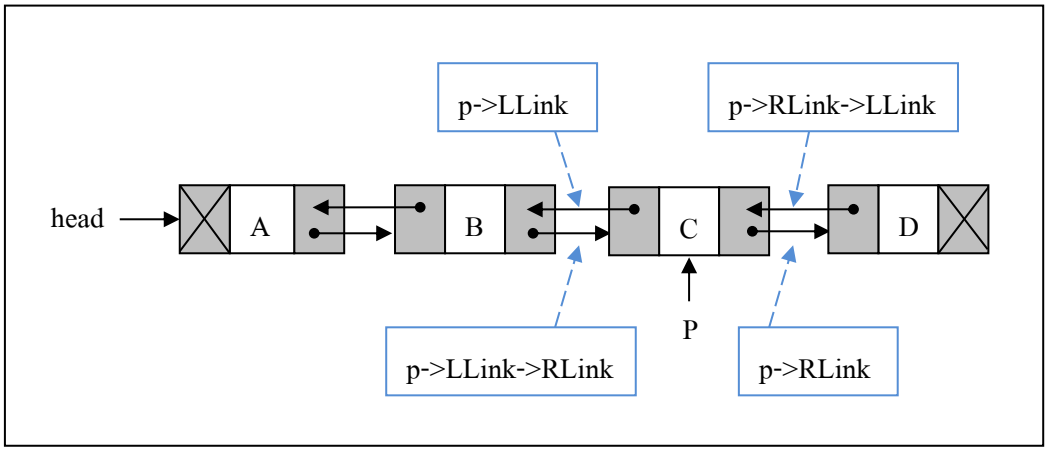
จากรูปที่ 5.19 แต่ละโหนดของลิงค์ลิสต์คู่จะบรรจุตัวชี้หรือพอยน์เตอร์อยู่สองตัว คือ LLink ที่ชี้ไปข้างหลังซึ่งก็คือโหนดก่อนหน้า ในขณะที่ RLink จะชี้ไปข้างหน้าซึ่งก็คือโหนดถัดไป สังเกตว่าฟิลด์ LLink ของโหนดแรกและฟิลด์ RLink ของโหนดสุดท้ายมีค่าเป็น NULL เนื่องจากไม่มีการเชื่อมโยงโหนด

การประกาศตัวแปรที่มีโครงสร้างแบบลิงค์ลิสต์คู่

```
struct node
{
        node  *LLink;
        int data;
        node *RLink;
 };
node  *head *p, *pt;
```

นั่นคือ กำหนดตัวแปร head, p และ pt เป็นตัวแปรพอยน์เตอร์ชนิดเรคอร์ดซึ่งประกอบด้วยฟิลด์ 3 ฟิลด์ คือ

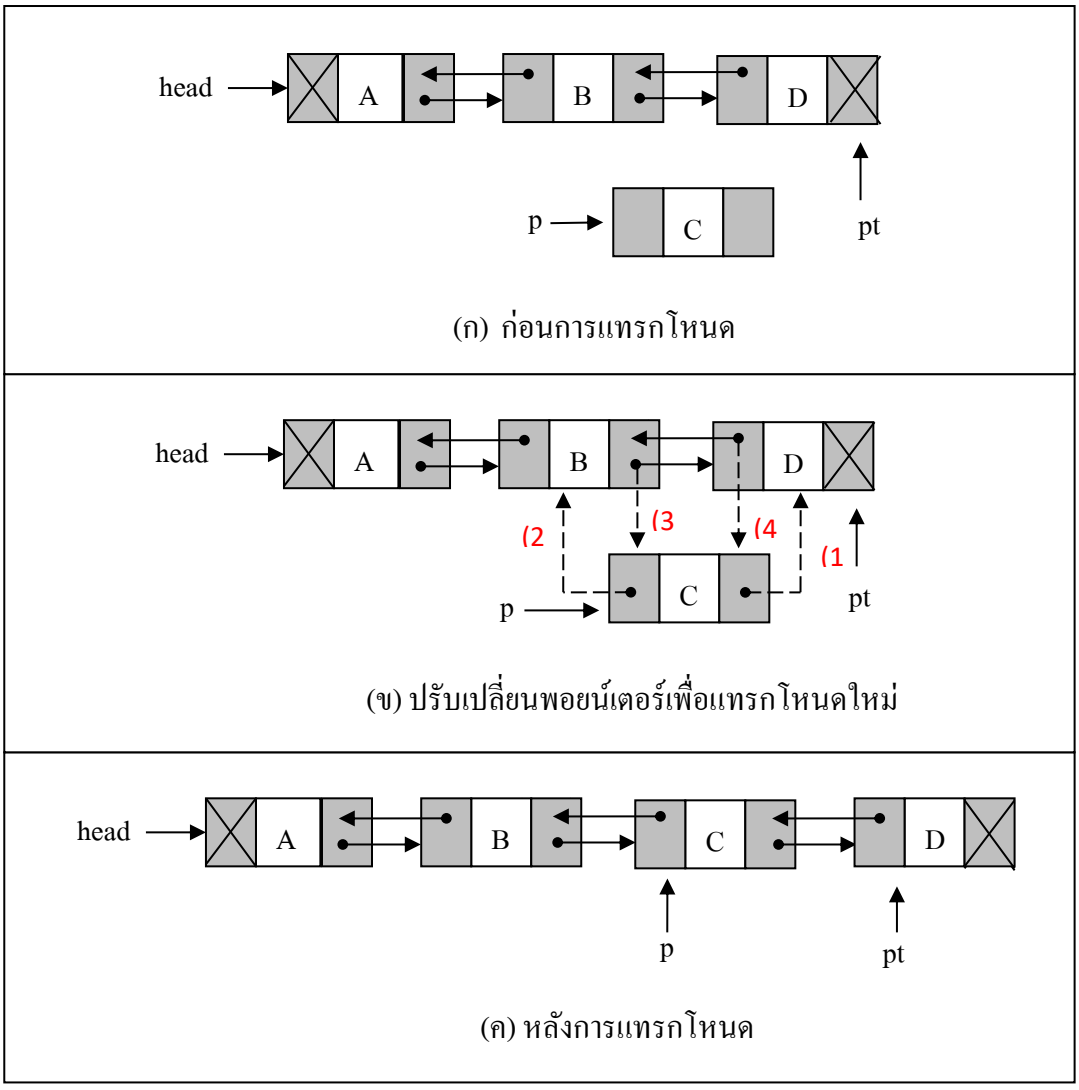
1. ฟิลด์ LLink เป็นฟิลด์ตัวชี้หรือพอยน์เตอร์สำหรับชี้โหนดก่อนหน้า
2. ฟิลด์ data เป็นฟิลด์สำหรับเก็บข้อมูล
3. ฟิลด์ RLink เป็นฟิลด์ตัวชี้หรือพอยน์เตอร์สำหรับชี้โหนดถัดไป



รูปที่ 5.20 ตัวอย่างลิงค์ลิสต์คู่และการเรียกชื่อพอยน์เตอร์

การแทรกโหนดในลิงค์ลิสต์คู่

การแทรกโหนดในลิงค์ลิสต์คู่ค่อนข้างซับซ้อนกว่าลิงค์ลิสต์เดี่ยวเพราะต้องพิจารณาพอยน์เตอร์หลายตัว ดังรูปที่ 5.21



รูปที่ 5.21 การแทรกโหนดในลิงค์ลิสค์คู่

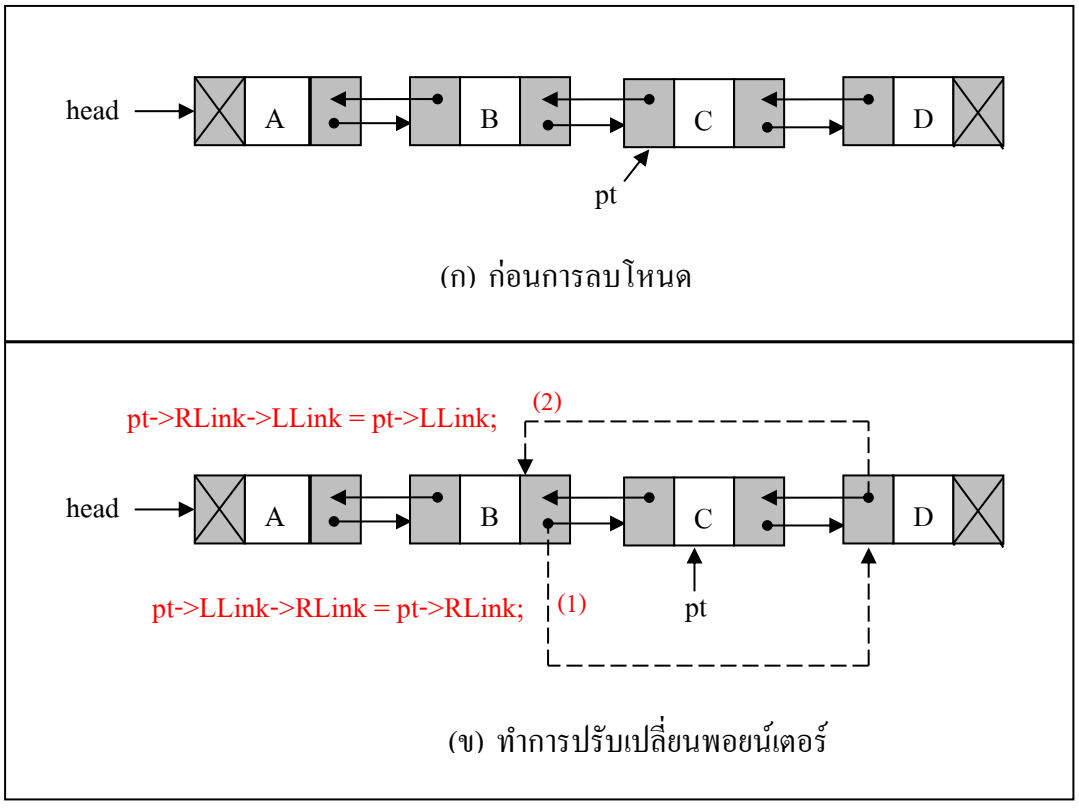
จากรูปที่ 5.21 การแทรกโหนดในลิงค์ลิสต์คู่สามารถใช้พอยน์เตอร์เพียงตัวเดียวคือ pt เพื่อท่องเข้าไปในลิงค์ลิสต์ และการแทรกโหนด p หน้าโหนด pt จะต้องใช้ 4 คำสั่งดังนี้

```
(1)  p->RLink = pt;
(2)  p-> LLink = pt->LLink;
(3)  pt->LLink->RLink = p;
(4)  pt->LLink = p;
```

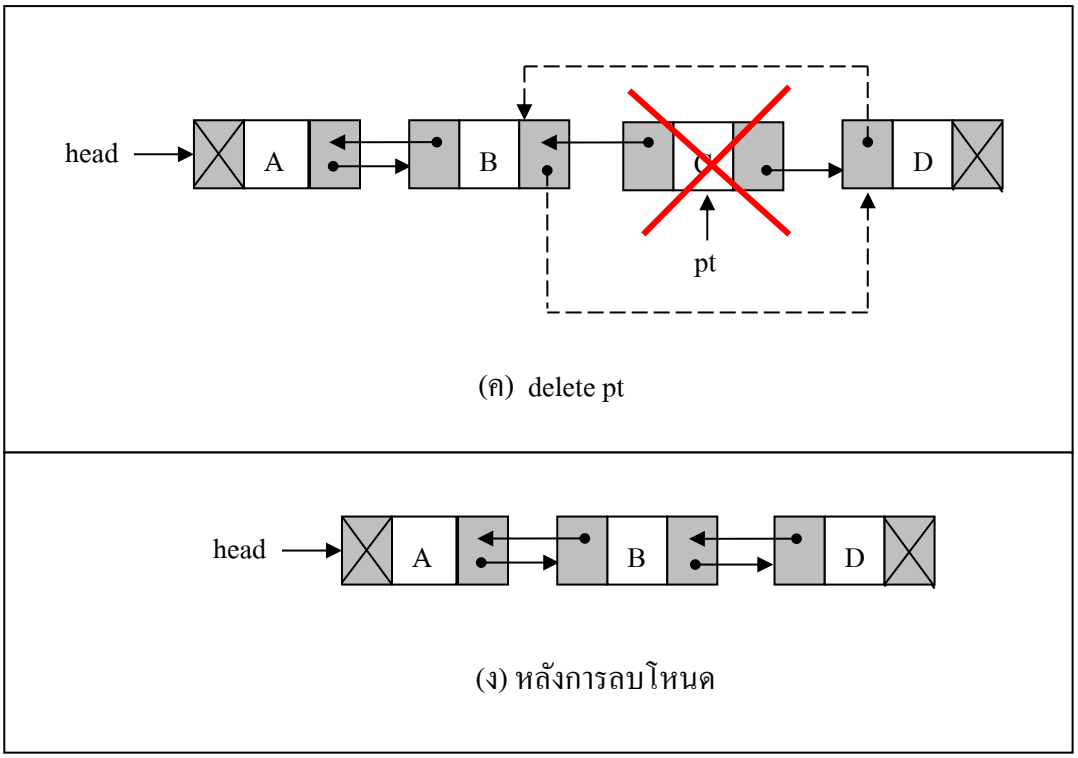
ลำดับของคำสั่งมีความสำคัญมาก จะต้องระมัดระวังเป็นพิเศษซึ่งจะสลับคำสั่งไม่ได้ เพราะจะมีผลทำให้เชื่อมโยงโหนดไม่ถูกต้อง

การลบโหนดในลิงค์ลิสต์คู่

การลบโหนดในลิงค์ลิสต์คู่เราสามารถใช้พอยน์เตอร์เพียงตัวเดียว คือ pt เหมือนกับการแทรกโหนดเพื่อท่องเข้าไปหาโหนดที่ต้องการลบ เมื่อค้นพบแล้วก็ให้ฟิลด์ LLink ของโหนดที่ต้องการลบคือโหนด pt เปลี่ยนค่าในฟิลด์ RLink ของโหนดก่อนหน้าให้ย้ายข้ามโหนดที่ไม่ต้องการไปชี้ยังโหนดถัดไป แสดงการลบโหนดดังรูปที่ 5.22



รูปที่ 5.22 การลบโหนดในลิงค์ลิสต์คู่



รูปที่ 5.22 (ต่อ) การลบโหนดในลิงค์ลิสต์คู่

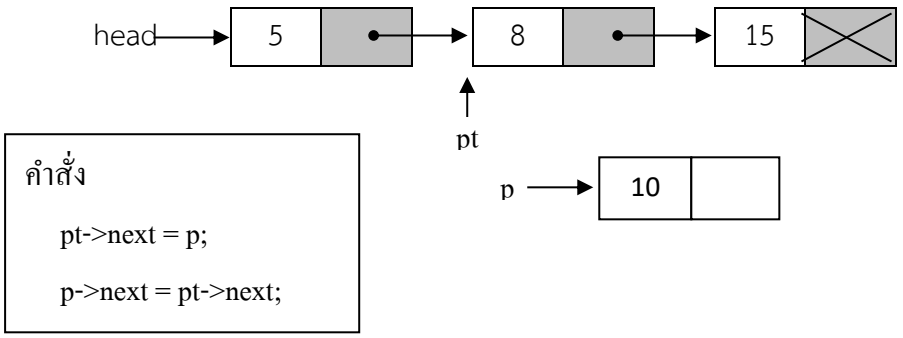
จากรูปที่ 5.22 สามารถเขียนขั้นตอนคำสั่งเพื่อลบโหนด pt ได้ดังนี้

```
(1) pt->LLink->RLink = pt->RLink;
(2) pt->RLink->LLink = pt->LLink;
(3) delete pt;
```

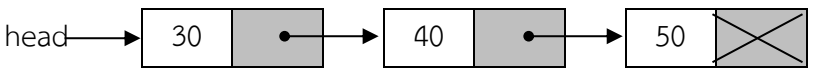
แบบฝึกหัดท้ายหน่วยที่ 5

ตอนที่ 1 จงตอบคำถามต่อไปนี้

1. จงอธิบายลักษณะโครงสร้างข้อมูลแบบลิงค์ลิสต์
2. ลิงค์ลิสต์เดี่ยวและลิงค์ลิสต์คู่เหมือนและแตกต่างกันอย่างไร
3. ลักษณะที่บ่งบอกว่าเป็นโหนดสุดท้ายในลิงค์ลิสต์มีลักษณะใดบ้าง
4. การท่องเข้าไปในลิงค์ลิสต์มีวัตถุประสงค์เพื่ออะไร
5. จงเขียนขั้นตอนคำสั่งการแทรกโหนดที่ตำแหน่งแรกของลิงค์ลิสต์ พร้อมวาดรูปประกอบ
6. จงเขียนขั้นตอนคำสั่งการแทรกโหนดที่ท้ายลิงค์ลิสต์ พร้อมวาดรูปประกอบ
7. จงเขียนขั้นตอนคำสั่งการลบโหนดตรงส่วนกลางของลิงค์ลิสต์ พร้อมวาดรูปประกอบ
8. โครงสร้างข้อมูลแบบลิงค์ลิสต์เหมาะกับงานลักษณะใด จงอธิบาย
9. เพราะเหตุใดจึงไม่ใช้พอยน์เตอร์ head ในการท่องเข้าไปในลิงค์ลิสต์
10. จากรูปลิงค์ลิสต์ ถ้าต้องการแทรกโหนด p ไว้ท้ายโหนด pt และใช้คำสั่งในการปรับเปลี่ยนพอยน์เตอร์ดังที่กำหนดจะเกิดผลอย่างไร จงอธิบายพร้อมกับวาดรูปประกอบ



11. จากผลลัพธ์ที่ได้ในข้อ 10 จะมีวิธีการแก้ไขอย่างไร
12. จงอธิบายลิงค์ลิสต์วงกลม
13. จงอธิบายลิงค์ลิสต์คู่
14. จากลิงค์ลิสต์ที่ข้อมูลมีการเรียงลำดับดังนี้



รูปของลิงค์ลิสต์จะเปลี่ยนแปลงไปอย่างไรภายหลังประมวลผลชุดคำสั่งต่อไปนี้

14.1
```
p = head->next;
p->data = 38;
```

14.2
```
q = new node;
q->data = 35;
p = head->next;
q->next = p;
head->next = p;
```

14.3
```
p = head->next;
q = p->next;
p->next = q->next;
delete q;
```

ตอนที่ 2 จงบอกความหมายของคำศัพท์ในโครงสร้างข้อมูลแบบลิงค์ลิสต์ต่อไปนี้

1. node
2. new
3. Data Field
4. Link Field
5. NULL
6. delete p;
7. LLink
8. RLink
9. Singly Linked List
10. Doubly Linked List
11. Circular Linked List

กิจกรรมฝึกปฏิบัติหน่วยที่ 5

จุดประสงค์

1. เพื่อให้นักศึกษามีความรู้ความเข้าใจเกี่ยวกับโครงสร้างข้อมูลแบบลิงค์ลิสต์
2. เพื่อทดสอบการทำงานของลิงค์ลิสต์
3. เพื่อให้เข้าใจขั้นตอนคำสั่งการแทรกและลบโหนดในลิงค์ลิสต์

กิจกรรรม

1. ให้นักศึกษาศึกษาตัวอย่างโปรแกรมที่ 5.2 โดยทำการป้อนโปรแกรมที่กำหนดโดยใช้ Editor ของภาษาซี ทำการคอมไพล์ รันโปรแกรม และเขียนผลลัพธ์ที่เกิดขึ้น
2. ให้นักศึกษาวาดรูปลิงค์ลิสต์ ภายหลังจากการประมวลผลคำสั่งในโปรแกรมเสร็จสิ้นแล้ว
3. ให้นักศึกษาเขียนคำอธิบายการทำงานของคำสั่งในโปรแกรมแต่ละบรรทัด
4. ให้นักศึกษาน้อมนำหลักปรัชญาเศรษฐกิจพอเพียงมาบูรณาการในการทำกิจกรรมครั้งนี้ โดยให้ทำกิจกรรมด้วยความรอบคอบ และดำเนินงานตามขั้นตอนด้วยความระมัดระวังเพื่อความปลอดภัย