ที่ 6 รายการโยง

บลำดับ
เับ

IT 204 127

6.6.3 การย้อนลำดับรายการ

6.7 รายการโยงแบบวงกลมและโหนดหัวเรื่อง

6.7.1 โหนดหัวเรื่อง

6.7.2 การลบโหนดเฉพาะ

6.8 รายการโยงคู่

6.8.1 แนวคิดเบื้องต้น

6.8.2 การนิยามรายการโยงคู่

6.8.3 การลบโหนด

6.8.4 การใส่โหนด

6.9 ตัวอย่างงานประยุกต์ของรายการ

6.9.1 พหุนาม

6.9.2 รายการหลายตัวโยงอย่างง่าย

6.9.3 แถวลำดับสพาซ

บทสรุป

แบบฝึกหัด

บทที่ 6
รายการโยง
(Linked Lists)

เนื้อหาในบทนี้จะพิจารณาการใช้รายการโยงเพื่อแทนที่โครงสร้างข้อมูลแบบเชิงเส้น : รายการเชิงเส้นทั่วไป

6.1 การแทนที่รายการโยง(Linked List Representation)

6.1.1 ปัญหาของการแทนที่แบบลำดับ (Problems with Sequential Representation)

ในบทที่แล้วเราได้แทนที่กองซ้อนและแถวคอยโดยเก็บไว้ในแถวลำดับซึ่งข้อจำกัดพื้นฐานของการแทนที่ภาวะนี้คอปริมาณของหน่วยเก็บซึ่งจัดสรรให้กับกองซ้อนหรือแถวคอยจะคงที่ เนื้อที่ถูกจัดสรรโดยไม่ต้องระมัดระวังว่ากองซ้อนหรือแถวคอยว่างหรือไม่และมีความเป็นไปได้ของการเกิดส่วนล้น(overflow) ถ้ากองซ้อนหรือแถวคอยมีขนาดใหญ่กว่าที่คาดไว้นี้คือปัญหาเมื่อแถวลำดับถูกใช้ให้เก็บรายการเชิงเส้นทั่วไป

อย่างไรก็ตามในกรณีทั่วไปมีขีดจำกัดมากกว่าจะเกิดอะไรขึ้นเมื่อสมาชิกตัวที่ 43 ชื่อ "IGOR" ถูกลบออกจากรายการ รายการเชิงเส้นที่มีข้อมูล 2492 ชื่อสมาชิกตัวที่ 1 จนถึงตัวที่ 42 ไม่ถูกกระทบแต่สมาชิกตัวที่ 44 ชื่อ "IVAN" จนถ้าสมาชิกตัวที่ 2492 ชื่อ "ZELDA" เชิงตรรกะแต่ละตัวจะเลื่อนขึ้นในแถวลำดับ

สมาชิกตัวที่ 44 กลายเป็นสมาชิกตัวที่ 43
สมาชิกตัวที่ 45 กลายเป็นสมาชิกตัวที่ 44
. . .
สมาชิกตัวที่ 2492 กลายเป็นสมาชิกตัวที่ 2491

ขณะนี่เกิดอะไรขึ้นเมื่อสมาชิกตัวใหม่ชื่อ "HORAITO" ใส่หลังสมาชิกตัวที่ 41 ชื่อ "HORACE" สมาชิกตัวที่ 42 ชื่อ "HOWARD" จนถึงสมาชิกตัวที่ 2491 ชื่อ "ZELDA" เชิงตรรกะต้องย้ายลงในแถวลำดับเพื่อทำให้เกิดช่องว่างสำหรับสมาชิกตัวที่ 42 ตัวใหม่ สมาชิกตัวใหม่ไม่สามารถนำไปไว้ตอนท้ายของรายการเพราะว่า การเรียงอันดับเชิงตรรกะของสมาชิกในรายการเชิงเส้นถูกส่งโดยการเรียงอันดับเชิงกายภาพของสมาชิกในแถวลำดับ

6.1.2 การแทนที่แบบไม่ใช่ลำดับ (Nonsequential Representation)

ผลเฉลยของปัญหาการเคลื่อนย้ายข้อมูลที่พบในการแทนที่แบบลำดับ คือใช้การแทนที่แบบไม่ใช่ลำดับแทนการ

การแทนที่แบบลำดับสะท้อนการเรียงอันดับเชิงตรรกะของสมาชิกในรายการเชิงเส้นในหน่วยเก็บเชิงกายภาพการเรียงอันดับเชิงตรรกะและการเรียงอันดับเชิงกายภาพ

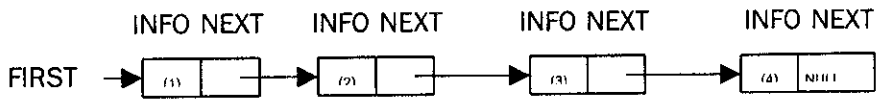
เหมือนกันสำหรับการแทนที่แบบไม่ใช้ลำดับการเรียงอันดับเชิงตรรกะของสมาชิกและการเรียงอันดับเชิงกายภาพของสมาชิกไม่จำเป็นต้องเหมือนกัน

การเรียงอันดับเชิงตรรกะหมายถึงการแทนที่โดยที่สมาชิกแต่ละตัวชี้ไปยังสมาชิกตัวถัดไปสมาชิกของรายการโยงซึ่งกันและกัน การเข้าถึงสมาชิกในอันดับเชิงตรรกะจะตามเส้นโยงเหล่านี้

(The logical ordering is represented by having each element point to the next element; the elements of the list are linked together. To access the elements in logical order. These links are followed.)

6.1.3 แนวคิดพื้นฐาน (Basic Concepts)

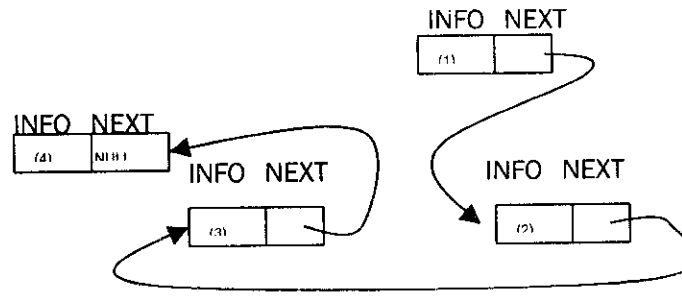
รูป 6-1 คือภาพวาดของรายการโยงแบบเชิงเส้นซึ่งมีสมาชิสี่ตัวแต่ละตัวเรียกว่าโหนด (node)



รูป 6-1 ตัวอย่าง linked list

มีพอยน์เตอร์หนึ่งตัวชื่อ FIRST ชี้ไปยังโหนดแรกของรายการโหนดแต่ละตัวมีการโยงไปยังโหนดถัดไปของรายการโหนดตัวสุดท้ายมีพอยน์เตอร์เป็น NULL ซึ่งแสดงว่าไม่มีโหนดถัดไป โหนดแต่ละตัวมีสองส่วน คือ มูลค่า(contents) ของข้อมูลและเขตพอยน์เตอร์ แต่ละโหนดคือโครงสร้างข้อมูลแบบระเบียน

โหนดรายการโยงไม่จำเป็นต้องประชิดกันเชิงกายภาพ รายการของโหนดสี่ตัวในรูป 6-1 อาจจะแสดงได้ด้วย รูป 6-2



รูป 6-2 ตัวอย่างรายการโยงซึ่งความหมายเหมือนกับรูป 6-1

รายการเชิงเส้นที่ไม่มีโหนดเรียกว่ารายการว่าง(empty list) การแทนที่รายการโยงของรายการว่างมีเพียงพอยน์เตอร์ตัวแกที่ว่าง การปฏิบัติการของการสร้างรายการว่างเป็นดังนี้

First := Null

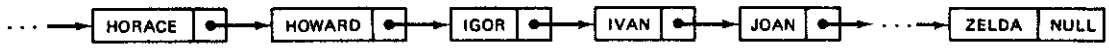
6.1.4 ข้อดี (Advantages)

ข้อจำกัดอีกประการหนึ่งของการใช้แถวลำดับเก็บแถวคอยหรือกองซ้อนคือ สมาชิกทุกตัวต้องเป็นชนิดเดียวกัน (homogeneous) ข้อจำกัดนี้จะละทิ้งไปโดยใช้โครงร่างการแทนที่แบบรายการโยง โหนดที่มีโครงสร้างข้อมูลแตกต่างกันสามารถโยงเข้าด้วยกัน

ข้อดีของการแทนที่แบบรายการโยงจะเห็นชัดเจนเมื่อพิจารณาปฏิบัติการของการใส่และการลบ ตัวอย่างเช่น ลบโหนดที่ 43 ชื่อ "IGOR" ออกจากรายการเชิงเส้นที่มีโหนด 2492 ตัว (รูป 6-3(a)) สิ่งที่จำเป็นมีเพียงเปลี่ยนพอยน์เตอร์ในโหนดก่อนหน้าโหนดที่ 42 ให้ชี้ไปยังโหนดถัดไป (โหนดที่ 44)ไปยังโหนดใหม่ และ ให้โหนดใหม่ชี้ไปยังโหนดที่ 42 (รูป 6-3(c)) ไม่มีการเคลื่อนย้ายข้อมูลแต่อย่างใด

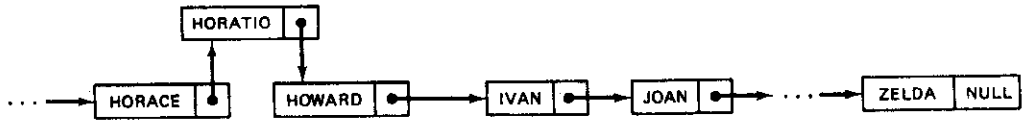
6.1.5 ค่าใช้จ่าย (Costs)

ค่าใช้จ่ายของการใช้การแทนที่แบบรายการโยงไม่ใช่การแทนที่แบบลำดับมีสองข้อ คือข้อแรกการแทนที่แบบรายการโยงจะต้องใช้เนื้อที่เพิ่มสำหับเขตพอยน์เตอร์ ข้อที่สองเวลาที่ใช้ในการค้นหาโหนดในการแทนที่แบบรายการโยงจะนานกว่ารายการที่เก็บในแถวลำดับ โปรดระลึกว่ามีวิธีการคำนวณซึ่งใช้หาตำแหน่งเริ่มต้นของสมาชิกตัวที่ N ในรายการโยง(สมมติว่าไม่ได้ใช้พอยน์เตอร์เพิ่ม) ต้องเยี่ยมโหนดก่อนหน้า N-1 สำหรับโหนดที่ N ไม่สามารถหาตำแหน่งได้โดยตรง



รูป 6-3 (a) ตัวอย่างรายการโยงเดี่ยว

รูป 6-3 (b) ลบ Igor ออกจากรายการโยงรูป



รูป 6-3 (c) ใส่ Horatio ในรายการโยงรูป 6-3 (b)

6.2 การปฏิบัติการพื้นฐานบนรายการโยง (Basic Operations on a Linked List)

ขณะนี้จะพิจารณาการปฏิบัติการพื้นฐานบนรายการโยงหลังจากนั้นจะดูว่าการประกาศรายการโยงและการจัดการรายการโยงในโปรแกรมจะทำอย่างไร

6.2.1 สัญกรณ์ (Notation)

ให้ P เป็นตัวแปรพอยน์เตอร์ (pionter variable) ค่าของ P คือเลขที่อยู่ (address) ตัวอย่างเช่น ตำแหน่งที่อยู่ของตัวแปรอื่น การปฏิบัติการที่ถูกต้องบนตัวแปรพอยน์เตอร์มีดังนี้

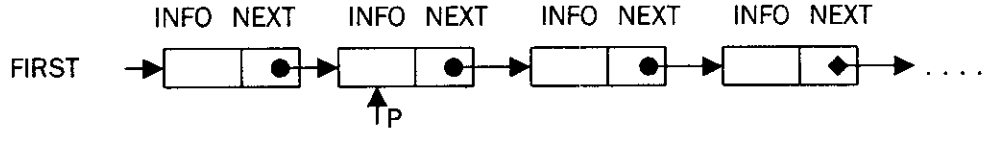
1. Test for null
2. Test for equality with another pointer variable
3. Set to null
4. Set to point to a node

การปฏิบัติการเหล่านี้เท่านั้นซึ่งสามารถกระทำได้โดยพอยน์เตอร์และกำหนดสัญกรณ์เป็นดังนี้

Node(P) โหนดถูกชี้โดย P
Info(P) ค่าในส่วน information ของโหนดซึ่งถูกชี้โดย P
Next(P) ค่าในส่วน link ของโหนดซึ่งถูกชี้โดย P

6.2.2 การลบโหนด (Removing a Node)

จงพิจารณารายการโยง รูป 6-4 (a)



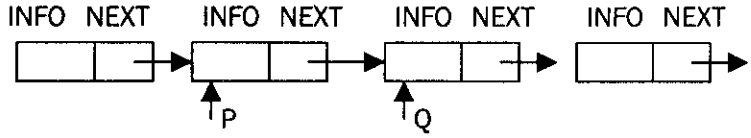
รูป 6-4 (a) ขั้นตอนในการลบโหนดออกจากรายการโยง

อัลกอริทึมเพื่อลบโหนดซึ่งอยู่หลังโหนดที่มีตัวชี้เป็น P ออกจากรายการโยงเราใช้ Q เป็นตัวแปรพอยน์เตอร์เพิ่มและขั้นแรกให้ Q ชี้ที่โหนดซึ่งต้องการจะลบออกจากรายการโยง

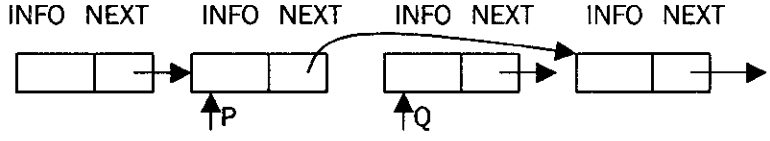
(a) q := Next (P)
(b) Next(P) := Next(Q)

(c) Free node Q's space for reuse.

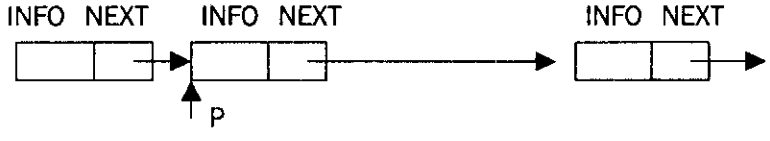
ขั้น (a) ให้โครงสร้างแสดงในรูป 6-4 (b)



ขั้น (b) โครงสร้างแสดงในรูป 6-4(c)



ขั้น (c) โครงสร้างแสดงในรูป 6-4 (d) การลบหนึ่งโหนดต้องสัมผัสสองโหนด โหนดซึ่งมีตัวชี้เป็น P และ Q ไม่มีโหนดอื่นใดถูกเข้าถึงหรือเปลี่ยนแปลง



รูป 6-4(d)

กิจกรรม

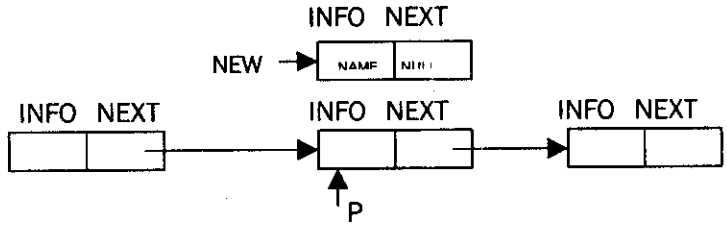
ลำดับของการปฏิบัติการนี้ส่งกลับ (returns) โหนดตัวที่ถูกลบให้นำกลับมาใช้ใหม่ได้ (reused) ยกเว้นถ้า P ชี้ที่โหนดตัวสุดท้ายของรายการจะต้องดัดแปลง (modified) อัลกอริทึมข้างต้นนี้อย่างไรเพื่อให้ครอบคลุมสถานะการณ์นั้น

. . . . . .

. . . . . .

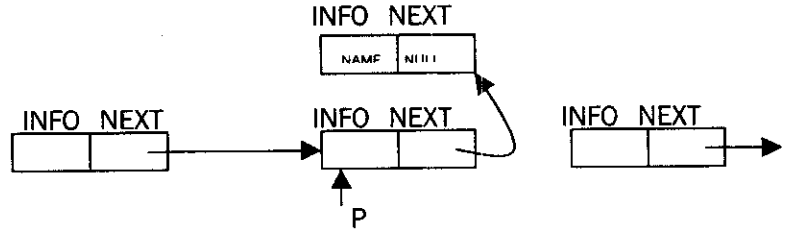
. . . . . .

หลังจากขั้น (a) และ (b) ให้โครงสร้างแสดงในรูป 6-5 (a)



รูป 6-5(a)ขั้นตอนการใส่หนึ่งโหนดในรายการโยง

หลังจากขั้น (c) และ (d) โครงสร้างแสดงในรูป 6-5 (b)



รูป 6-5(b) ขั้นตอนการใส่หนึ่งโหนดในรายการโยง

สิ่งสำคัญที่จะต้องรู้ไว้คือ links เช่น ตัวแปร P, Q , NEW และ NEXT ทั้งหมดนี้ชี้ไปที่โหนดไม่ใช่เป็นส่วน INFO ของโหนด ลูกศรในแผนภาพเหล่านี้ชี้ไปที่กล่อง (entire box) การใส่หนึ่งโหนดต้องสัมผัสสองโหนดคือโหนดตัวโหม่และอีกหนึ่งโหนดที่มีพอยน์เตอร์เป็น P นอกจากนี้แล้วไม่มีโหนดอื่นใดต้องเข้าถึงหรือเปลี่ยนแปลง

6.2.3 การใสโหนด (inserting a node)

อัลกอริทึมเพื่อใส่ contents ของตัวแปรชื่อ NAME เข้าไปในรายการโยง โดยที่ให้โหนดใหม่นี้อยู่หลังโหนดทีมีตัวชี้เป็น P อีกครั้งหนึ่งซึ่งเราจะใช้ตัวแปรเพิ่มสองตัวคือ Q และอีกตัวหนึ่งชื่อ NEW

(a) Get space for the new node, and make NEW point to it.

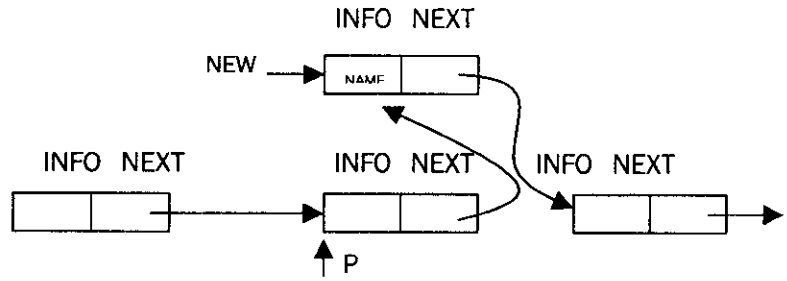
(b) Info(NEW) := NAME

(c) Q := Next(P)

(d) Next(P) := NEW

(e) Next(NEW) := Q

โครงสร้างแสดงในรูป 6-5(c)



รูป 6-5(c) ขั้นตอนการใส่หนึ่งโหนดในรายการโยง

6.3 การจัดการพื้นที่ว่าง (Manageng Available space)

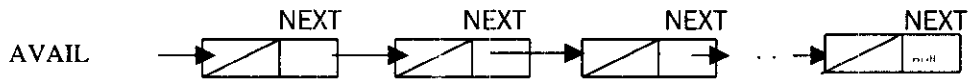
เราได้สมมติมาแล้วว่าฟังก์ชันสองชุด ฟังก์ชันหนึ่งสร้างพื้นที่โหนดและอีกฟังก์ชันหนึ่งส่งกลับโหนดที่ลบออกเพื่อนำกลับมาใช้อีก ซึ่งสองฟังก์ชันนี้ใช้ในอัลกอริทึมการใส่โหนดและอัลกอริทึมการลบโหนดขณะนี้จะพิจารณาฟังก์ชันเหล่านี้ในรายละเอียดที่มากขึ้น

6.3.1 ที่พักหน่วยเก็บ (The Storage Pool)

ที่พักหน่วยเก็บประกอบด้วยเนื้อที่ทั้งหมดซึ่งยังไม่ได้นำมาใช้ สมมติว่าเนื้อที่เหล่านี้ถูกจัดรูปแบบด้วยโครงสร้างของโหนดสำหรับรายการโยงที่จริงหนึ่งโปรแกรมนั้นอาจใช้รายการโยงหลายชุด โหนดอาจมีรูปแบบหลากหลายในกรณีซึ่งเรามีพื้นที่พักหน่วยเก็บของโหนดว่างหลายชุด และเพื่อให้ง่ายต่อการอธิบาย สมมติว่าขณะนี้เรามีโหนดเพียงหนึ่งชนิดเท่านั้น

อะไรคือโครงสร้างที่เหมาะสมสำหรับที่พักหน่วยเก็บเราคงไม่ต้องการใช้การแทนที่แบบลำดับเพราะว่าโหนดต่าง ๆ จะถูกลบออกค่อนข้างจะเป็นรายการเชิงสุ่มจากรายการโยงและเราต้องการนำเนื้อที่นั้นกลับมาใช้ใหม่เพราะฉะนั้นที่พักหน่วยเก็บจะถูกแทนที่โดยใช้โครงสร้างแบบโยงอย่างไรก็ตามเชิงตรรกะที่พักหน่วยเก็บเกือบจะเป็นรายการเชิงเส้น เราเรียกฟังก์ชั่นซึ่งส่งกลับตำแหน่งที่อยู่ของโหนดอิสระถัดไปในที่พักหน่วยเก็บว่า Getnode (we call the function that returns the location of the next free node in the storage pool Getnode.)

เริ่มแรกเมื่อรายการ 'user" ทั้งหมดว่างที่พักหน่วยเก็บจะประกอบด้วยเนื้อที่ว่างทั้งหมด เราจะใช้พอยน์เตอร์ Avail ชี้ไปที่โหนดตัวแรกบนรายการของเนื้อที่ว่าง (รูป 6-6)



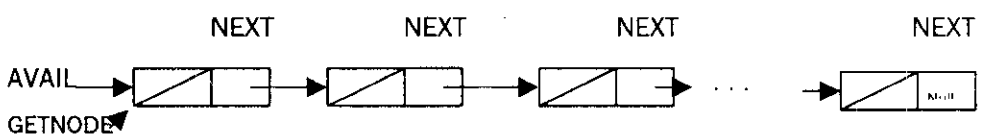
รูป 6-6 รายการเนื้อที่ว่าง (The list of available space)

6.3.2 การรับโหนดใหม่ (Getting a New Node)

เมื่อต้องการเนื้อที่โนหดให้เอามาจากรายการเนื้อที่ว่างนี้โดยเรียกฟังก์ชัน Getnode สำหรับเงื่อนไขยกเว้น (exception condition) กรณีไม่มีเนื้อที่ว่างเหลืออยู่เลยต้องถูกจัดการอย่างถูกต้อง

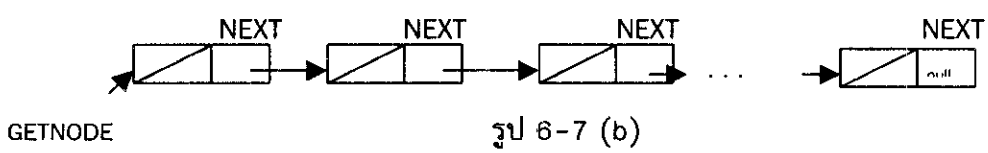
```
        If Avail = Null
        then out-of-free-space
(a)         else begin Getnode := Avail;
(b)                    Avail := Next(Avail);
(c)                    Next(Getnode) := Null
                end;
```

ขั้น (a) โครงสร้างแสดงในรูป 6-7(a)



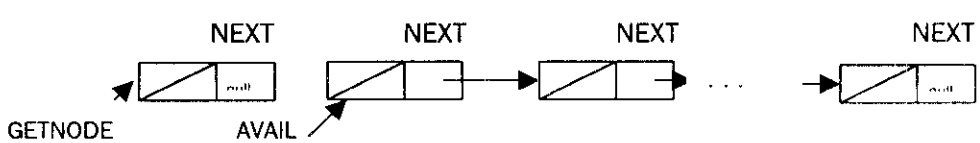
รูป 6-7(a)

ขั้น (b) โครงสร้างแสดงในรูป 6-7 (b)



รูป 6-7 (b)

ขั้น (c) โครงสร้างแสดง 6-7(c)



รูป 6-7 (c)

การรับโหนดหนึ่งตัวจากรายการเนื้อที่ว่างต้องสัมผัสเพียงหนึ่งโหนด(โหนดตัวที่อยู่หน้าสุดของรายการ) และการเปลี่ยนแปลงพอยน์เตอร์ Avail

6.3.3 การปล่อยโหนด (Freeing a Node)

เราเรียกโปรซีเดอร์ซึ่งส่งกลับโหนดซึ่งไม่ต้องใช้แล้วไปยังรายการพื้นที่ว่างว่า Freenode การกระทำนี้ทำให้พื้นที่ของโหนดตัวนั้นนำกลับมาใช้ได้อีก

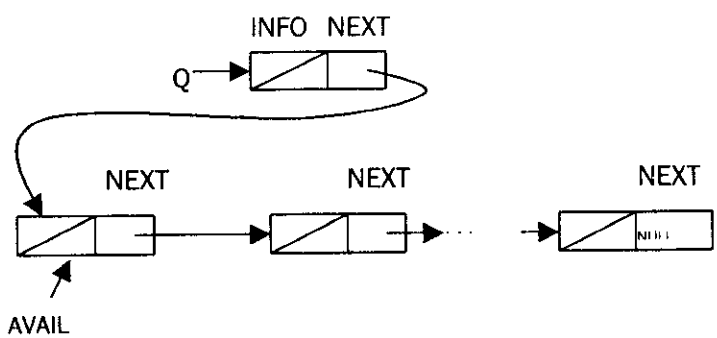
อาร์กิวเมนต์ของ Freenode คือ พอยน์เตอร์ไปยังโหนดซึ่งจะนำมาเก็บในที่พัก หน่วยเก็บโปรซีเดอร์ Freenode(Q) เขียนดังนี้

(a) Next(Q) := Avil

(b) Info(Q) := Null

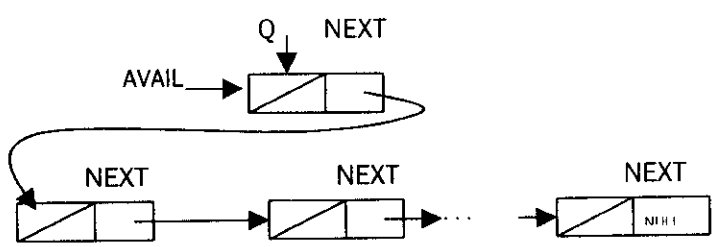
(c) Avail := Q

ขั้น (a) โครงสร้างแสดงในรูป 6-8 (a)



รูป 6-8(a) ขั้นตอนการปล่อยโหนดไปเก็บในรายการว่าง

หลังจากขั้น (b) และ (c) โครงสร้างแสดงในรูป 6-8 (b)



รูป 6-8 (b) ขั้นตอนการปล่อยโหนด ไปเก็บในรายการว่าง

หลังจากนั้นพอยน์เตอร์ Q จะถูกจัดการโดยรูทีนเรียก (invoking routine) ในลักษณะที่เหมาะสม Q จึงเป็นตัวแปรช่วย ส่วนของ Info ของโหนดซึ่งปล่อยเป็นอิสระดตัวใหม่จึงมี slash กำกับเหมือนกับส่วน Info ของโหนดอิสระอื่น ๆ ในแผนภาพ เพื่อแสดงว่ามันว่างและจะถูกเขียนใหม่เมื่อลบออกจากที่พักหน่วยเก็บโดยการเรียก Getnode ในภายหลัง

อัลกอริทึม Getnode และ Freenode กระทำกับโครงสร้างที่พักหน่วยเก็บซึ่งไม่เพียงแต่เป็นรายการเชิงเส้นเท่านั้นแต่เป็นกองซ้อนด้วย ที่พักหน่วยเก็บบางครั้งจึงถูกเรียกว่า Avail stack

กิจกรรม

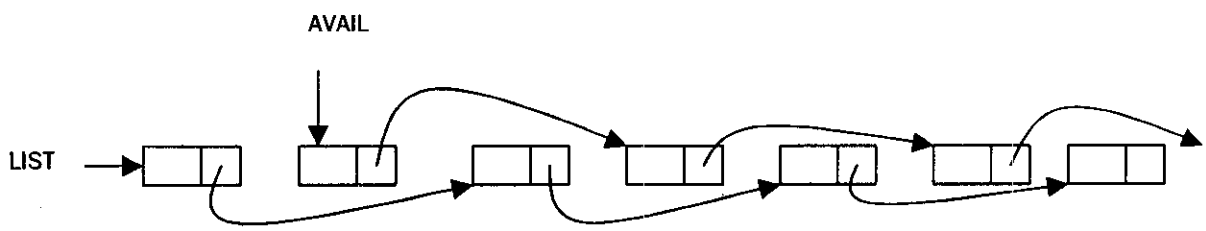
ถ้าที่พักหน่วยเก็บมีโครงสร้างเป็น Avail queue อัลกอริทึมจะเขียนแตกต่างกันอย่างไร?

. . . . . .

. . . . . .

. . . . . .

หลังจากลำดับของการปฏิบัติการใส่และการลบกระทำบนรายการผู้ใช้ รายการผู้ใช้และรายการว่างจะเป็นคู่แฝดกันในเชิงกายภาพ (รูป 6-9)



รูป 6-9 Avail list และ user's list

ในส่วนที่เหลือของการศึกษารายการโยงเรายังคงสมมติต่อไปว่ามีโปรซีเดอร์ Getnode และ Freenode อยู่

6.4 รายการโยงใน Pascal โดยใช้ตัวแปรพอยน์เตอร์ (Linked List In Pascal,Using Pointer Variable)

6.4.1 การนิยามรายการโยง (Defining Linked Lists)

ขณะนี้จะศึกษาเทคนิคการใช้รายการโยงในโปรแกรมในหัวข้อที่ผ่านมา เราใช้พอยน์เตอร์อย่างกว้างขวางภาษา Pascal มีข้อมูลชนิดพอยน์เตอร์ในตัว (built-in pointer) และสิ่งอำนวยความสะดวกสำหรับการควบคุมหน่วยเก็บแบบพลวัต (dynamic storage) หนึ่งโหนดถูกประกาศใน Pascal ดังนี้

```
type nodeptr = ↑ nodetype;
        nametype = packed array [1..10] of char;
        nodetype = record
                        info : nametype;
                        next : nodeptr
                   end;
```

สัญกรณ์ ↑ nodetype แสดงว่า nodeptr คือข้อมูลชนิดซึ่งชี้ไปยังสิ่งที่มีชนิดเป็น nodetype ใน Pascal ตัวแปรซึ่งประกาศเป็นพอยน์เตอร์มีการจำกัดให้เก็บแอสเดรสของโครงสร้างข้อมูลชนิดหนึ่งโดยเฉพาะ

เราประกาศ P ให้เป็น nodeptr และโหนดมีโครงสร้างเป็นชนิด nodetype

```
var p : nodeptr;
    node : nodetype;
```

และแทนที่ด้วย p↑ .info และ p↑ .next เป็นส่วน info และส่วน next ตามลำดับของโหนดที่มีตัวชี้เป็น p กรณีที่พอยน์เตอร์ p มีค่าเป็น null กำหนดโดย p := null

6.4.2 การจัดการเนื้อที่ (managing space)

verbs ต่อไปนี้โปรแกรมเมอร์ใช้เพื่อควบคุมสิ่งอำนวยความสะดวกการจัดการเนื้อที่ของ Pascal

new(p) จัดสรรเนื้อที่หนึ่งโหนดมีแบบชนิดข้อมูลเป็นอะไรก็ได้ ซึ่งมี P เป็นตัวชี้ P จึงเป็นเลขที่อยู่แอสเดรส(address)สิ่งนี้คือฟังก์ชัน getnode ของ Pascal

dispose(p) ให้อิสระกับวัตถุปัจจบันซึ่งมีตัวชี้เป็น p สิ่งนี้คือ โปรซีเดอร์ freenode ของ

Pascal

สิ่งแวดล้อมเวลาดำเนินงานของ Pascal จัดกระทำกับการจัดการพื้นที่ว่าง

(The PascAL run-time environment handles the free-space management.)

6.4.3 การลบโหนด (Removing a Node)

อัลกอริทึมลบโหนดซึ่งอยู่หลังโหนดที่มีตัวชี้เป็น p ออกจากรายการโยงเขียนด้วย Pascal ข้างล่างนี้เราใส่ content info ของ โหนดที่ถูกลบออกในตัวแปร out และสมมติว่าโหนดมีการประกาศเป็นส่วนกลาง (globally declared)

```
Procedure removaf(p:nodeptr, var  out :nametype);
    Var q : nodeptr;
    begin if (p ↑.next=nil)
        then UNDERFLOW-CONDITION
        else begin q:= p↑.next;
            p↑.next := q↑.next;
            out := q↑.info
            dispose(q)
            end;
    end;
```

มีเพียงสองโหนดในรายการโยงซึ่งถูกสัมผัส : โหนดซึ่งมีตัวชี้เป็น p และ q ส่วนโหนดอื่น ๆ ไม่ถูกเข้าถึงหรือไม่มีการเปลี่ยนแปลง

6.4.4 การใส่โหนด (Inserting a Node)

อัลกอริทึมใส่ contents ของตัวแปรให้เป็น in ในรายการโยงเพื่อให้โหนดใหม่นี้อยู่หลังโหนดทีมีตัวชี้เป็น p เขียนด้วยภาษา Pascal ดังนี้

```
Procedure insertaf(p:nodeptr, in:nametype);
    var q:nodeptr:
    begin new(q);
        q↑.info := in;
        q↑.info := p↑.next;
        p↑.next := q
    end;
```

มีเพียงสองโหนดเท่านั้นที่ถูกสัมผัส : โหนดตัวใหม่ที่มีตัวชี้เป็น q และโหนดอีกตัวหนึ่งที่มีตัวชี้เป็น p การตรวจสอบที่ปลอดภัย ก่อนเรียกอัลกอริทึมเหล่านี้โปรแกรมควรเชื่อมั่นว่าโหนดที่มีตัวชี้เป็น p มีอยู่จริงใน pascal ถ้า p = nil แสดงว่าไม่มีโหนดเช่นนั้น

6.5 รายการโยงใน COBOL และ Pascal โดยไม่ใช้ตัวแปรพอยน์เตอร์(Linked Lists in COBOL and Pascal, without Using Pointer Variables

6.5.1 การใช้แถวลำดับ (using an array)

เมื่อภาษาโปรแกรมที่ใช้ไม่มีข้อมูลชนิด pointer ในตัวการประกาศและการจัดการรายการโยงจะซับซ้อนมากขึ้น เรา simulate ตัวแปรพอยน์เตอร์โดยการเก็บรายการโยงในแถวลำดับ หลังจากนั้นค่าของตัวแปรพอยน์เตอร์จึงเป็นค่าของดรรชนีล่าง (subcript) รูป 6-10 คือแผนภาพของแถวของแถวลำดับซึ่งเก็บรายการโยงของชื่อ สมาชิกแต่ละตัวของแถวลำดับคือ หนึ่งระเบียนที่มีสองเขตข้อมูล : info และ Next เราใช้ค่า 0 แทน Null กองซ้อน Avail ใช้เนื้อที่แถวลำดับชุดนี้ร่วมกับ (รูป 6-11) การจัดการ links ระหว่างโหนดว่างทำให้การจัดการเนื้อที่ว่างง่ายกว่าที่สมาชิกแต่ละตัวมีพอยน์เตอร์ Null

	INFO	NEXT
1	IVAN	6
2	HOWARD	5
3		
4	HORACF	2
5	IGOR	1
6	JOAN	0
7		

First = 4

รูป 6-10 แถวลำดับเก็บรายการโยง

	INFO	NEXT
1	IVAN	6
2	HOWARD	5
3		7
4	HORACE	2
5	IGOR	1
6	JOAN	0
7		0

FIRST = 4

AVAIL = 3

รูป 6-11 แถวลำดับเก็บรายการ AVAIL และรายการโยงของผู้ใช้

รูป 6-12 (a) แสดงรายการโยงหลังจากลบสมาชิกที่มีตัวชี้เป็น P ออกไปเมื่อ p = 5 และหลังจากใส่ "HORATIO" ในลำดับเรียงตามตัวอักษรถูกต้อง รูป 6-12 (b)

	INFO	NEXT
1	IVAN	6
2	HOWARD	5
3		7
4	HORACE	2
5	IGOR	1
6	JOAN	0
7		0

FIRST = 4

AVAIL = 5

รูป 6-12(a) รายการโยงหลังจากลบ "IGOR"

	INFO	NEXT
1	IVAN	6
2	HOWARD	5
3		7
4	HORACE	2
5	IGOR	1
6	JOAN	0
7		0

FIRST = 4

AVAIL = 3

รูป 6-12(b) รายการโยงหลังจากลบ "IGOR" และในใส่ "HORATIO"

6.5.2 การนิยามรายการโยง (Defining a Linked List)

แถวลำดับซึ่งให้นิยามข้างล่างโปรดสังเกตว่าแถวลำดับเก็บทั้งรายการโยงของผู้ใช้และรายการโยงของเนื้อที่ว่าง

ใน COBOL :

```
01      SPACE-ARRAY.
        02 NODE OCCURS 500 TIMES.
              03 INFO          PIC  X(10).
              03 NEXTNODE      PIC  9(3).
        02 FIRSTNODE           PIC  9(3).
        02 AVAIL               PIC  9(3).
```

ในที่นี้เราใช้ NEXTNODE และ FIRSTNODE เพราะว่า NEXT และ FIRST เป็นคำสงวน (reserved words) ในภาษา COBOL

FIRSTNODE เก็บดรรชนีล่างของโหนดแรกในรายการของผู้ใช้ส่วน AVAIL เก็บดรรชนีล่างของโหนดแรกในรายการโยงเนื้อที่ว่าง

หน่วยเก็บแถวลำดับสามารถใช้ได้สำหรับรายการโยงใน Pascal ถึงแม้ว่าสิ่งนี้จะไม่ค่อยได้กระทำ เทคนิคตัวแปรพอยน์เตอร์ที่กล่าวมาแล้วในหัวข้อที่ผ่านมาถูกนำมาใช้

ใน Pascal หน่วยเก็บแถวลำดับสำหรับรายการโยงจะถูกประกาศดังนี้

```
type nodeptr = 0.. 500;
        nametype = packed array [1..10] of char;
        nodetype = record
                info : nametype;
                next : nodeptr
                end;
        var node : array[1..500] of nodetype
                first,avail : nodeptr;
```

ในทุกกรณีเมื่อเก็บรายการโยงในแถวลำดับเราใช้ค่า 0 แทน null

6.5.3 การลบโหนด (Removing a node)

อัลกอริทึมเพื่อลบโหนดซึ่งอยู่หลังจากโหนดที่มีตัวชี้เป็น P ออกจากรายการโยงจำเป็นต้องสัมผัสสองโหนด : โหนดตัวหนึ่งมีดรรชนีล่างเป็น P และอีกหนึ่งโหนดซึ่งอยู่ถัดจาก P ซึ่งมีดรรชนีล่างเป็น Q นอกจากนี้แล้วดรรชนีให้กับพื้นที่ว่างต้องเปลี่ยนแปลง

ใน COBOL :

```
REMOVAF.
        IF NEXTINODE(P) = 0
                underflow-condition
        ELSE COMPUTE Q = NEXTNODE(P)
                COMPUTE NEXTNODE(P) = NEXTNODE(Q)
                Output INFO(Q)
                COMPUTE NEXTNODE(Q) = AVAIL
                COMPUTE AVAIL = Q.
```

ใน Pascal:

```
procedure removaf(p:nodeptr);
var q: nodeptr;
        begin
                if (node[p].next = 0)
                then UNDERFLOW-CONDITION
                else begin
                        q := node[p].next;
                        node[p].next := node[p].next;
                        writeln(node[p].info);
                        node[q].next := avail;
                        avail := q
                       end;
         end;
```

6.5.4 การใส่โหนด (Insertion a Node)

อัลกอริทึมใส่ contents ของตัวแปร IN เข้าไปในรายการโยงเพื่อให้โหนดตัวนี้อยู่หลังโหนดที่มีตัวชี้เป็น P ต้องสัมผัสสองโหนด : โหนดตัวหนึ่งเอามาจากรายการพื้นที่ว่างซึ่งมีดรรชนีล่างเป็น Q และโนหดอีกหนึ่งตัวที่มีดรรชนีล่างเป็น P ดรรชนีของโหนดว่างตัวถัดไปต้องเปลี่ยนแปลง

ใน COBOL

```
INSERAF.
        IF AVAIL = 0
        THEN overflow-condition
        ELSE   COMPUTE Q = AVAIL
               COMPUTE AVAIL = NEXTNODE(AVAIL)
               MOVE IN TO INFO(Q)
               COMPUTE NEXTNODE(Q) = NEXTNODE(P)
               COMPUTE NEXTNODE(P) = Q.
```

ใน Pascal:

```
procedure inseraf(p:nodeptr, in : nametype);
var q : nodeptr;
begin  if (avail = 0)
       then OVERFLOW-CONDITION
       else begin q := avail;
              avail := node[avail].next;
              node[q].info := in;
              node[q].next := node[p].next;
              node[p].next := q
             end;
end;
```

6.6 การจัดการที่มากขึ้นของรายการโยงเดี่ยว

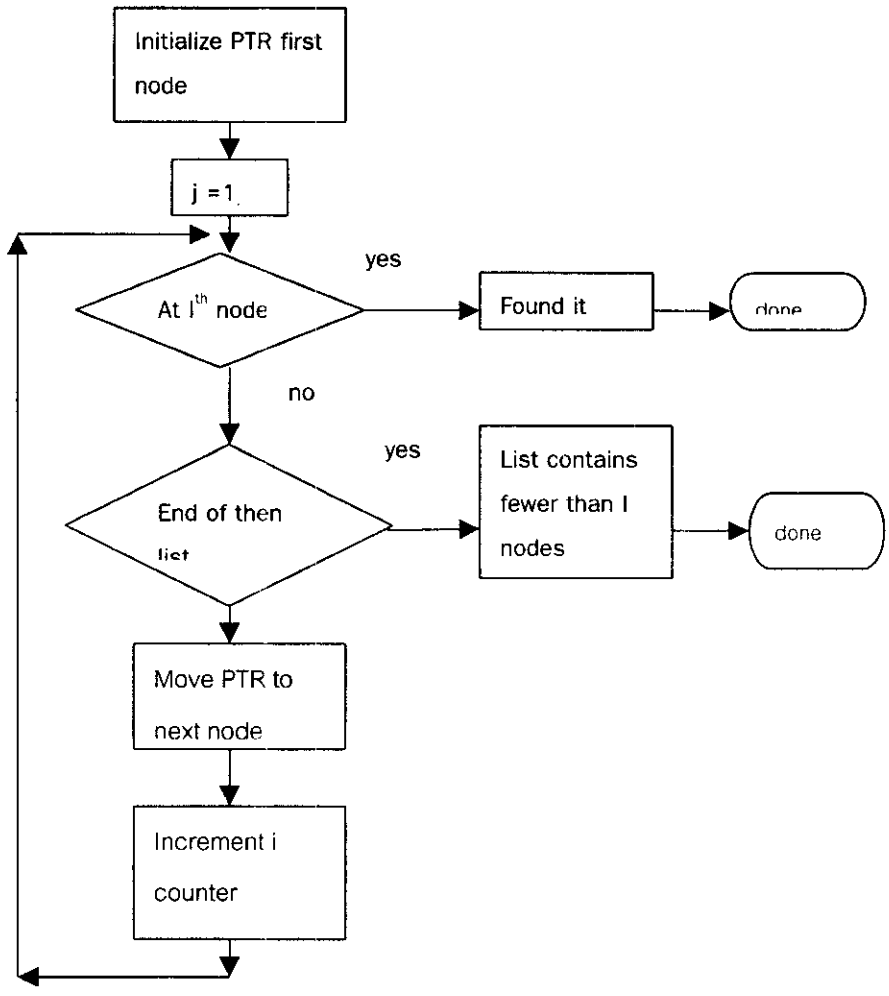
(Futher Manipulations of singly Linked Lists)

การปฏิบัติการพื้นฐานของการลบโหนดและการใส่โหนดในรายการโยงซึ่งรายละเอียดอยู่ในหัวข้อที่ผ่านมา ในที่นี้จะพัฒนาอัลกอริทึมเพื่อการจัดการที่เพิ่มขึ้นของรายการโยงและนำไปกสู่การอภิปรายของการ enhancements ร่วมกันกับรายการโยงเดี่ยว อัลกอริทึมแต่ละชุดถูกเข้ารหัส (encoded) ด้วยภาษา COBOL หรือ Pascal

6.6.1 การค้นหาโหนดเฉพาะ (Finding a Particular Node)

ขั้นแรกจะค้นหาโหนดที่ I ในรายการได้อย่างไร (first, how can you find the I^{th} node in a list)

ถ้ารายการถูกเก็บในแถวลำดับ การคำนวณโดยใช้เลขที่อยู่ฐาน (base location) ของแถวลำดับและขนาดของโหนดจะให้ผลลัพธ์เป็นแอดเดรสของโหนดที่ต้องการ ถ้ารายการถูกแทนที่ลักษณะโยงหลังจากนั้นต้องมีขั้นตอนผ่านโหนดต่าง ๆ นับจำนวนจนกระทั่งพบตำแหน่งโหนดตัวที่ I ซึ่งจำเป็นต้องตรวจสอบจริง ๆ อย่างน้อยที่สุด I โหนดในรายการอัลกอริทึมแสดงในรูป 6-13 การหาโหนดที่ I ต้องสัมผัส I โหนดยกเว้นกรณีรายการมีโหนดน้อยกว่า I ตัวดังนั้นรายการทั้งหมดต้องถูกสัมผัส



รูป 6-13 ตรรกะในการหาโหนดที่ I ของรายการโยง

ภาษา COBOL ทำให้เกิดผลในทางปฏิบัติของการค้นหาดังนี้โดยใช้การประกาศ SPACE-ARRAY จากหัวข้อที่แล้ว

```
COMPUTE PTR = FIRSTNODE.
PERFORM STEP-THRU-LIST VARYING J FROM 1 BY 1
                UNTIL J = I OR PTR= 0.
IF PTR = 0
        List has fewer than I nodes
ELSE output INFO(PTR).
```

เมื่อ

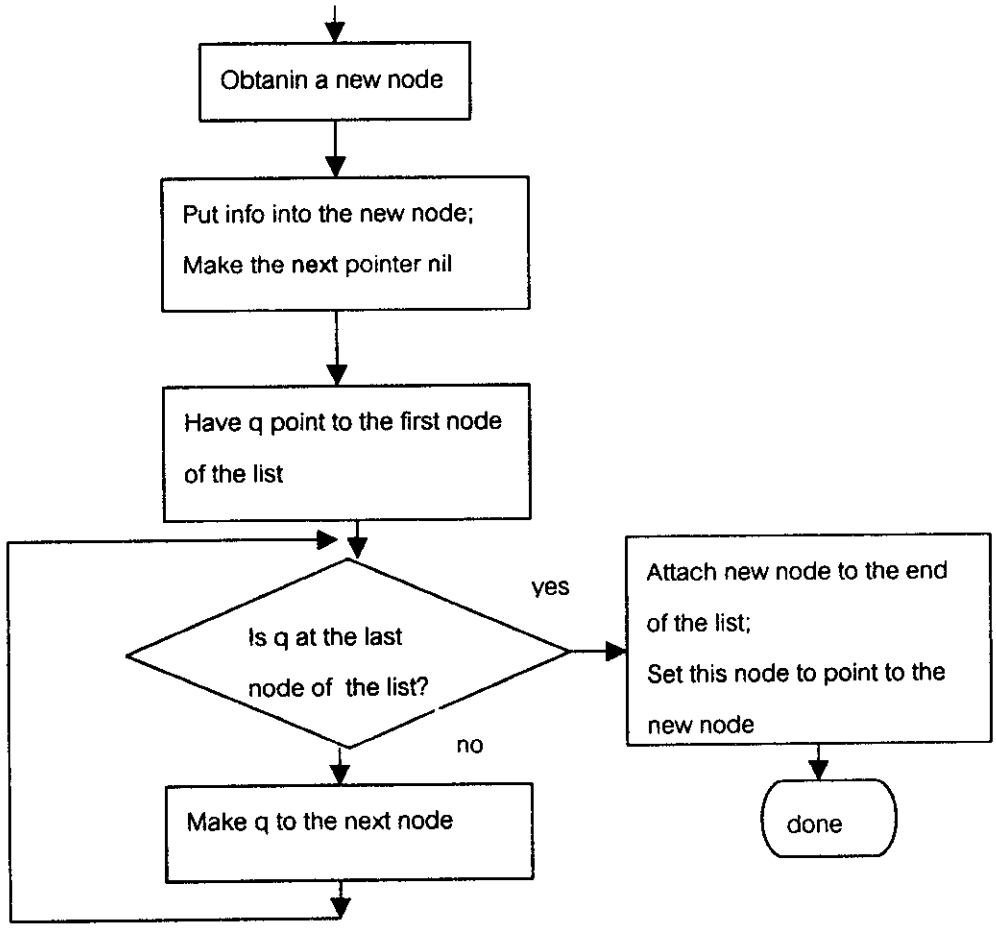
```
STEP-THRU-LIST.
        COMPUTE PTR = NEXTNODE(PTR).
```

COMPUTE PTR = NEXTNODE(PTR).

อัลกอริทึมของการนับจำนวนโหนดปัจจุบันในรายการโยงทิ้งให้เป็นแบบฝึกหัดท้ายบท

6.6.2 การใส่ที่ท้ายสุดของรายการ (Insertion at the End of a list)

อัลกอริทึมใส่โหนดที่ตอนต้นของรายการได้เขียนไปแล้วเมื่ออภิปรายหัวข้อกองซ้อนว่างและอัลกอริทึมการลบโหนดออกจากตอนต้นของรายการโยงเขียนไว้แล้วเช่นกันในตอนอภิปรายหัวข้อกองซ้อนว่างขณะนี้จะพิจารณาอัลกอริทึมใส่โหนดที่ตอนท้ายสุดของรายการโยงอังกอริทึมแสดงในรูป 6-14 โหนดทั้งหมดในรายการต้องถูกสัมผัสก่อนพบตอนจบรายการ ใช้การประกาศจากหัวข้อที่แล้ว



รูป 6-14 ตรรกะการใส่โหนดที่ตอนท้ายสุดของรายการโยง

```
Procedure inserend (first : nodeptr, in : nametype);
var newnode, q : nodeptr;
begin   new(newnode);
        Newnode↑.info := in;
        newnode↑.next := nil;
                    q := first;
        do while (q↑.next <> nil)
                q := q↑.next;
        q↑.next := newnode
end;
```

เราใช้ newnode เป็นชื่อพอยน์เตอร์ให้กับโหนดตัวใหม่เพราะว่า new เป็นคำสงวนรหัสข้องต้นนี้ทำงานได้ยกเว้นกรณีที่รายการถูกชี้โดย first เป็นค่าว่างตั้งแต่แรก
(This code works except in the case where the initial list pointex to by first is empty)

กิจกรรม จงปรับโปรซีเดอร์เพื่อให้สามารถทำงานได้อย่างถูกต้องสำหรับทุกกรณี

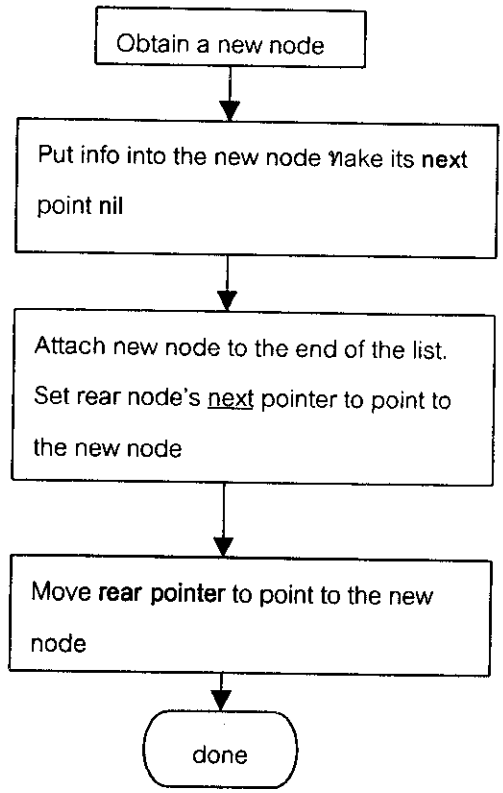
. . . . . .

. . . . . .

. . . . . .

ขณะนี้ได้เห็นแล้วว่ารายการโยงสามารถนำมาปฏิบัติให้เกิดผลได้อย่างไร ไม่เพียงแต่กับกองซ้อน แต่ยังเป็นแถวคอยด้วยเมื่อรายการโยงถูกนำมาใช้แทนแถวคอยควรจะมีพอยน์เตอร์โดยตรงไปยังตำแหน่ง rear ของแถวคอยนั้นเพื่อหลีกเลี่ยงการวนซ้ำ (looping) ผ่านโหนดทั้งหมด ในกรณีที่มีการค้นหาโหนดสุดท้าย

เราเรียกพอยน์เตอร์ตัวนี้ว่า rear ดังนั้นอัลกอริทึมการใส่โหนดแสดงในรูป 6-15 และตามด้วยรหัสภาษา Pascal เป็นดังนี้



รูป 6-15 ตรรกะการใส่โหนดที่ตอนท้ายสุดของรายการซึ่งมีพอยน์เตอร์เป็น rear

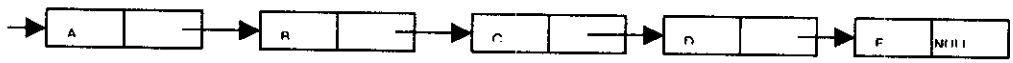
```
Procedure inseren(in : nametype, var rear : nodeptr);
var newnode : nodeptr;
begin new (newnode);
        newnode↑.info := in;
        newnode↑.next := nil;
        rear↑.next := newnode;
                rear := newnode
end;
```

มีเพียงสองโหนดเท่านั้นที่ต้องสัมผัส คือ newnode และโหนดอีกหนึ่งตัวซึ่งเป็นโหนดตัวสุดท้ายและมีพอยน์เตอร์เป็น rear

6.3.3 การย้อนลำดับรายการ (Reversing a List)

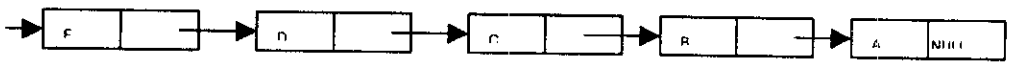
อัลกอริทึมเพื่อย้อนลำดับสมาชิกในรายการโยงเพื่อให้สมาชิกตัวสุดท้ายกลายเป็นสมาชิกใหม่ตัวที่หนึ่งและสมาชิกตัวที่หนึ่งกลายเป็นสมาชิกใหม่ตัวสุดท้ายเกี่ยวข้องกับการทำขั้นตอนผ่านทุกโหนดของรายการเพื่อเปลี่ยนแปลงพอยน์เตอร์ทั้งหมด

List

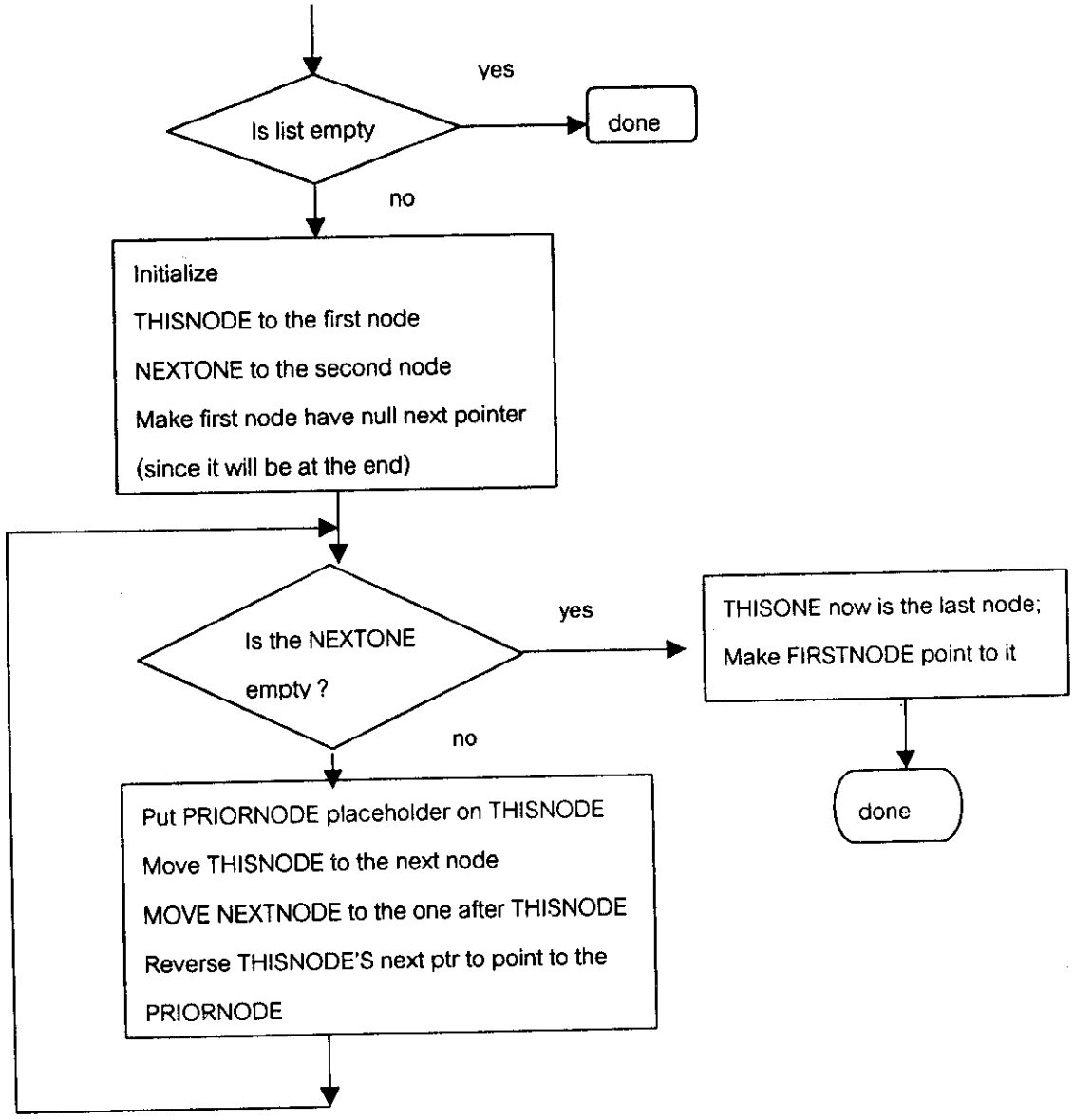


หลังจากย้อนลำดับรายการผลลัพธ์คือ

List



อัลกอริทึมแสดงในรูป 6-16



รูป 6-16 ตรรกะเพื่อย้อนลำดับโหนดของรายการโยง

ใน COBOL (โดยใช้ space-array)

```
01    AUX-POINTERS.
      02    THISNODE     PIC   9(3).
      02    PRIORNODE    PIC   9(3).
      02    NEXTON       PIC   9(3).


REVESE-LIST.
      IF    FIRSTNODE    NOT = 0
      THEN  COMPUTE      THISNODE = FIRSTNODE
            COMPUTE      NEXTONE = NEXTNODE(THISNODE)
            COMPUTE      NEXTNODE(THISNODE) = 0
            PERFORM      STEP-THIRU-LIST UNTIL(NEXTONE=0)
            COMPUTE      FIRSTNODE = THISNODE.
STEP-THRU-LIST.
      COMPUTE      PRIORNODE = THISNODE.
      COMPUTE      THISNODE = NEXTONE.
      COMPUTE      NEXTONE = NEXTNODE(THISNODE).
      COMPUTE      NEXTNODE(THISNODE) = PRIORNODE.
```

กิจกรรม

อัลกอริทึมอื่น ๆ ซึ่งจะเป็นประโยชน์ในการพัฒนาตนเองได้แก่

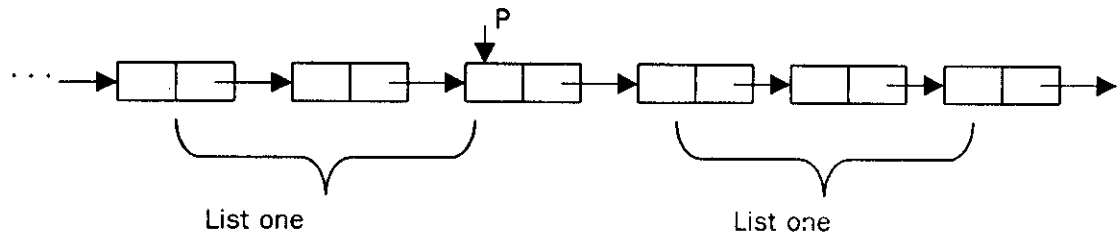
. การต่อรายการโยงสองชุดให้เป็นรายการโยงหนึ่งชุด(Concatenate two linked list into one linked list.)

. การแบ่งรายการโยงหนึ่งชุดให้เป็นรายการโยงสองชุดโดยให้สมาชิกตัวสุดท้ายในราย

การโยงชุดแรกคือสมาชิกที่มีตัวชี้เป็น P (รูป 6-17)

(Split a linked list into two linked lists so that the last element is the first one

is the element pointed to by P (Fig.6-17))



รูป 6-17 การแบ่งรายการโยงให้เป็นสองชุด

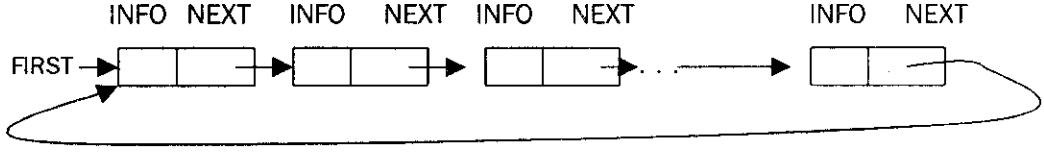
. การแบ่งรายการโยงหนึ่งชุดให้เป็นสองชุดโดยให้รายการโยงชุดที่หนึ่งมีเฉพาะสมาชิกที่ info < M และรายการโยงชุดที่สองประกอบด้วยสมาชิกที่ Info >= M เท่านั้น

6.7 รายการโยงแบบวงกลมและโหนดหัวเรื่อง (circularly Linked lists and Head Nodes)

ขณะนี้เรามีความเข้าใจการปฏิบัติการพื้นฐานเหล่านี้แล้ว เราสามารถสืบหาสิ่งสนับสนุนต่าง ๆซึ่งจะทำให้เอาชนะความยากที่มีหลากหลายในรายการโยงข้อบกพร่องอย่างหนึ่งซึ่งเราได้สังเกตเห็นแล้วคือเมื่อกำหนดให้ P เป็นพอยน์เตอร์ในรายการโยง มันเป็นไปไม่ได้ที่จะพบโหนดซึ่งอยู่หน้า Node(P)

การเปลี่ยนแปลงง่าย ๆ คือเปลี่ยนแปลงโครงสร้างข้อมูลซึ่งจะทำให้เราพบโหนดทุกตัวจากโหนดใด ๆ ก็ได้แทนที่จะเก็บ null pointer ใน next field ของโหนดตัวสุดท้ายของรายการให้โหนดตัวสุดท้ายชี้กลับไปยังตอนต้นของรายการ (รูป 6-18) โครงสร้างชนิดนี้เรียกว่า

รายการโยงแบบวงกลม (circularly linked list)



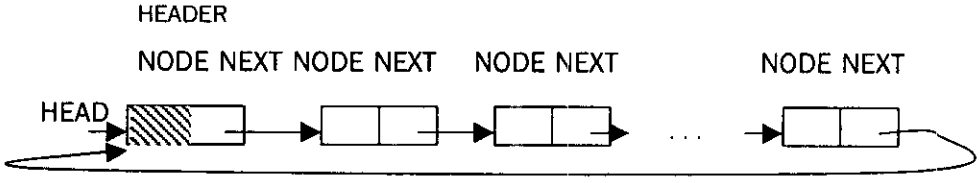
รูป 6-18 รายการโยงแบบวงกลม

กิจกรรม ให้นักศึกษาฝึกฝนตนเองถ้าการเปลี่ยนแปลงซึ่งโครงสร้างรายการ ประยุกต์กับอัลกอริทึมซึ่งนำเสนอในหัวข้อก่อนหน้านี้

6.7.1 โหนดหัวเรื่อง (Head nodes)

ขณะนี้ให้พิจารณาปัญหาของการค้นหาโหนดเฉพาะสมมติว่าเป็นโหนดที่ชื่อ "PETER" ในรายการโยงแบบวงกลม ถ้าโปรแกรมเมอร์ไม่ระมัดระวังมากพอเป็นการง่ายที่จะเกิดการวนซ้ำไม่รู้จบขณะผ่านขั้นตอนต่าง ๆ ในรายการ อย่าลืมว่าตอนท้ายสุดของรายการโยงแบบวงกลมไม่ใช่สมาชิกที่มี null pointer อีกต่อไป

วิธีหนึ่งของการแก้ปัญหานี้คือใส่โหนดหัวเรื่องที่ตอนต้นหรือตอนท้ายสุดของรายการ



รูป 6-20 รายการโยงแบบวงกลมที่มีโหนดหัวเรื่อง

โหนดหัวเรื่องแตกต่างจากโหนดอื่น ๆ หลายข้อดังนี้

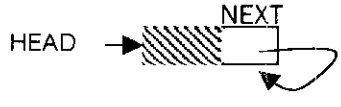
. โหนดตัวนี้อาจมีค่าพิเศษใน info field ของมันซึ่งเป็น invalid as data ในสมาชิกตัว

อื่น ๆ (ในกรณีนี้คือพื้นที่แรเงาใน info field ในรูปข้างต้น)

. อาจมี flag ซึ่งทำเครื่องหมายว่าเป็นโหนดหัวเรื่อง

นักออกแบบอัลกอริทึมอาจใส่ข้อมูลที่มีความหมาย (menaingful data) ในโหนดหัวเรื่อง ตัวอย่างเช่นโหนดหัวเรื่องมี content เป็นจำนวนโหนดในรายการ , คำอธิบายเช่นผู้ใช้ของรายการ (a user-oriented description of the list) , วันที่สร้างรายการ (a creation-date) หรือสารสนเทศอื่น ๆ (other information)

เมื่อรายการโยงแบบวงกลมที่มีหัวเรื่องเป็นรายการว่าง รายการนี้ยังคงมีโหนดหัวเรื่องดังแสดงในรูป 6-21



รูป 6-21 Empty circularty linked list with head node

นั่นคือ Next(Head) = Head

อัลกอริทึมสำหรับใส่ content ของตัวแปร NAME ที่อตนต้นของรายการคือ

(a) Get space for a new node, and make NEW point to it

(b) Put the content of the variable NAME into the INFO part of the new node.

(c) Make the new node point to the node that the head node used to pint to.

(d) Make Head point to the new node.

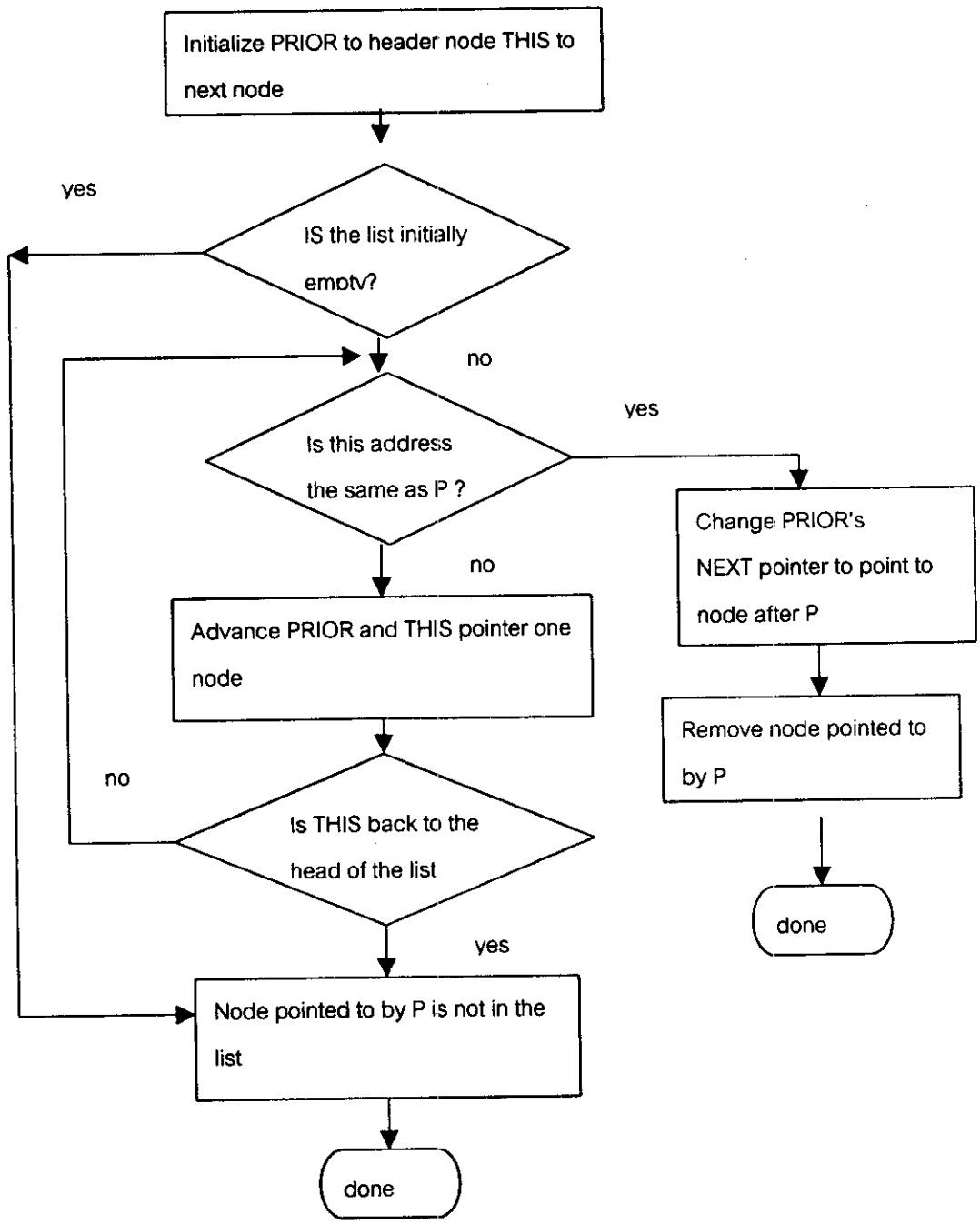
กิจกรรม การมีโหนดหัวเรื่องกระทบ (effect) กับอัลกอริทึมอื่น ๆ ที่นำแสนอในบทนี้หรือไม่ นักศึกษาควรมีข้อสังเกต indispensable value ของรูปภาพที่วาดเพื่อทำแบบจำลองการจัดกระทำพอยน์เตอร์ขณะที่พัฒนาและวิเคราะห์อัลกอริทึม

6.7.2 การลบโหนดเฉพาะ (Removing a Particular Node)

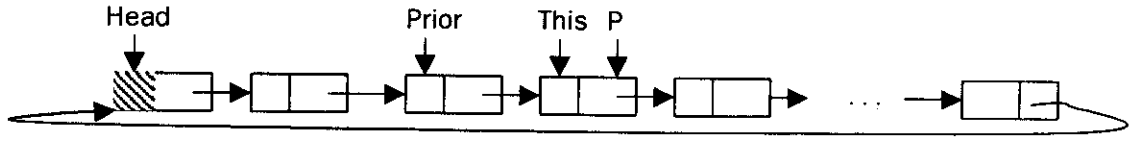
เราอาจมีข้อสังเกตแล้วว่าอัลกอริทึมลบโหนดนั้นทั้งหมดมีการกำหนดโหนดซึ่งอยู่หลังโหนดตัวที่ต้องการลบออกจากรายการเราจะลบโหดนที่มีตัวชี้เป็น P ได้อย่างไร เมื่อมีเฉพาะ links ไปยังโหนดถัดไปเท่านั้น สิ่งที่จำเป็นคือขั้นตอนผ่านโซ่ของโหนด เปรียบเทียบเลขที่อยู่จนกระทั่งพบโหนดตัวที่มีตำแหน่งเป็น P โดยต้องระวังไม่ให้ออกจากไปจากโหนดสุดท้ายของรายการไม่ใช่แบบวงกลม อัลกอริทึมในรูป 6-22 สิ่งจำเป็นไม่เพียงแต่เก็บการทดสอบเพื่อดูว่าเป็น โหนดที่มีตัวชี้เป็น THIS คือโหนดที่ต้องการลบ แต่ยังต้องเก็บ track คือโหนดก่อนหน้าเพื่อให้สามารถ link ไปยังโหนดหลังโหนดที่มีตัวชี้เป็น THIS ได้ ถ้า THIS เท่ากับ P (รูป 6-23) โดย

เฉลี่ยการลบโหนดที่มีตัวชี้เป็น P ต้องสัมผัสจำนวนโหนดครึ่งหนึ่งของโหนดในรายการ กรณีแย่ที่สุดซึ่งเกิดขึ้นเมื่อโหนดซึ่งมีตัวชี้เป็น P ไม่ได้อยู่ในรายการ โหนดทุกตัวในรายการต้องถูกสัมผัส

กิจกรรม จงเปรียบเทียบ performance นี้กับอัลกอริทึมการลบโหนดที่มีตัวชี้เป็น P ที่ได้นำเสนอสองชุดก่อนหน้าที่



รูป 6-22 ตรรกะสำหรับการลบโหนดที่มีพอยน์เตอร์ P ออกจากรายการโยงแบบวงกลม



รูป 6-23 รายการที่มีพอยน์เตอร์ช่วย (List with auxiliary pointers)

อัลกอริทึม เขียนด้วยรหัส Pascal ดังนี้

```
procedure removp (head : nodeptr, var p: nodeptr, out : nametype);
var prior, this : nodeptr;
        flag : 0..2;
begin
        prior := head;
        this := head↑.next;
        flag := 1; {search in progress}
        while flag = 1
                do begin
                        if (this = head)
                        then flag := 2; {have completed list, and not found
                                        node pointed to by P}
                        if (this = P)
                        then flag := 0 {have found node pointed to by P}
                        else begin prior := this;
                                this := this↑.next; {step to next node}
                             end;
                end;
if (flag> 0)
then NODE POINTED TO BY P IS NOT IN THE LIST
```

```
    else begin {removal of node pointed to by P}
            prior↑.next := p↑.next;
                    out := p↑.info;
                    dispose(p);
        end;
    end;
```

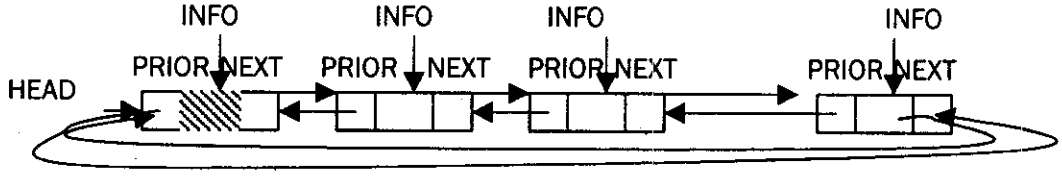
นักศึกษาจะพบแบบฝึกหัดที่เป็นประโยชน์เพื่อพัฒนาอัลกอริทึมข้างต้นสำหรับรายการโยงแบบไม่ใช่วงกลม และ/หรือรายการการโยงที่ไม่มีโหนดหัวเรื่อง

6.8 รายการโยงคู่ (Doubly linked list)

6.8.1 แนวคิดเบื้องต้น (Basic concepts)

เมื่อต้องการให้สามารถแวะผ่านรายการโยงแบบย้อนกลับ (backwords) หรือลบโหนดเฉพาะมีข้อดีในทางปฏิบัติคือให้รายการโยงคู่ไม่ใช่รายการโยงเดี่ยว

ในรายการโยงคู่โหนดแต่ละตัวไม่เพียงแต่มีพอยน์เตอร์ไปยังโหนดถัดไปเท่านั้นแต่ยังมีอีกหนึ่งพอยน์เตอร์ไปยังโหนดก่อนหน้า (รูป 6-24)



รูป 6-24 รายการโยงคู่

รายการโยงคู่มีหลากหลาย อาจจะไม่มีโหนดหัวเรื่องและ/หรือ ไม่เป็นรายการโยงแบบวงกลงมเราเรียกพอยน์เตอร์ไปยังโหนดถัดไปว่า Next (Node) เรียกพอยน์เตอร์ไปยังโหนดก่อนหน้าว่า Prior (Node) พอยน์เตอร์เหล่านี้อาจเรียกว่า right หรือ left pointers, successor และ predecessor pointers, s-link และ p-link pointers เป็นต้น

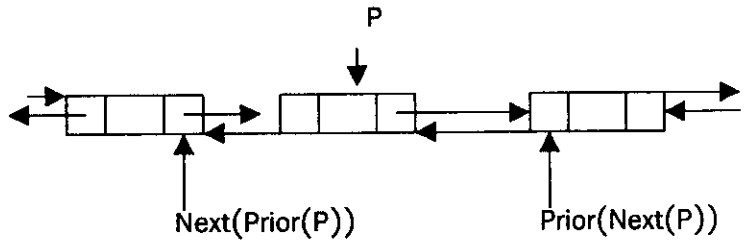
คุณสมบัติที่สำคัญของรายการโยงแบบวงกลมคือพอยน์เตอร์ P ใด ๆ ก็ตามในรายการ

$$Next\ (Prior\ (P)) = P$$

และ

$$P = Prior\ (Next(P))$$

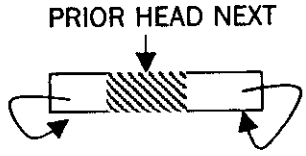
หนึ่งขั้นตอนย้อนกลับจากโหนด จากนั้นหนึ่งขั้นตอนไปข้างหน้าจะเป็นตำแหน่งโหนดเริ่มต้น (original node) ในทำนองเดียวกัน หนึ่งขั้นตอนไปยัง successor จากนั้น หนึ่งขั้นตอนย้อนกลับมายัง predecessor จะเป็นโหนดเริ่มต้น



รายการโยงคู่ว่าง(empty doubly linked list) แสดงให้เห็นในรูป 6-25 พอยน์เตอร์ Next และ Prior ในโหนดหัวเรื่องชี้ไปยังโหนดหัวเรื่องทั้งคู่และไม่มีโหนดอื่นในรายการนั่นคือ

Prior(Head) = Head

และ Next(Head) = Head



รูป 6-25 รายการโยงคู่ว่าง

6.8.2 การนิยามรายการโยงคู่ (Defining a Doubly Linked List)

โครงสร้างรายการโยงคู่ ประกาศใน Pascal โดยใช้ตัวแปรพอยน์เตอร์ดังนี้

```
type nodeptr = ↑.nodetype
     nodetype = record
                  prior : nodeptr;
                  info : nametype;
                  next : nodeptr
                end;
```

เนื่องจาก COBOL ไม่มีตัวแปรพอยน์เตอร์โปรแกรม COBOL จึงใช้หน่วยแถวลำดับสำหรับรายการโยงคู่ซึ่งเราได้เห็นมาแล้วในการอภิปรายหัวข้อการใช้แถวลำดับเก็บรายการโยงเดี่ยวรายการ Avail ของเนื้อที่ว่างให้ร่วมกับแถวลำดับของรายการโยงคู่ รายการAvail จึงเป็นรายการโยงคู่ด้วยโปรแกรมเมอร์ต้องจัดการดรรชนีที่เริ่มต้นรายการแต่ละชุด ในที่นี้มีตัวแปร HEAD และ AVAIL รหัสต่อไปนี้สมมติว่าจำนวนโหนดมากที่สุดคือ 500 โหนด

```
01    SPACE-ARRAY.
      02    NODE OCCURS 500 TIMES.
            03    PRIOR       PICTURE 9(3).
            03    INFO        PICTURE 9(3).
            03    NEXTNODE    PICTURE 9(3).
      02    HEAD              PICTURE 9(3).
      02    AVAIL             PICTURE 9(3).
```

รูป 6-26 แสดงให้เห็นแถวลำดับเก็บรายการโยงคู่

กิจกรรม

จงเปรียบเทียบตัวอย่างนี้กับรายการโยงเดี่ยวที่แสดงในรูป 6-12

	INFO	NEXT	PRIOR
1	IVAN	6	2
2	HOWARE	1	4
3		7	5
4	HORACE	2	0
5		3	0
6	JOAN	0	1
7		0	3

FIRST = 4

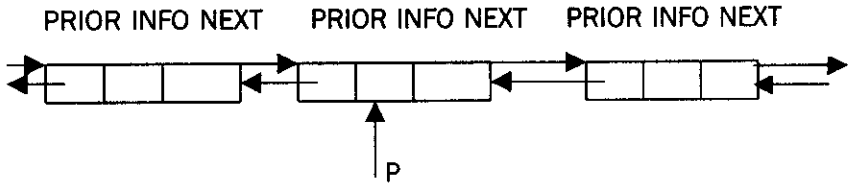
AVAIL = 5

รูป 6-26 แถวลำดับเก็บรายการโยงคู่

6.8.3 การลบโหนด (Removing a node)

ขณะนี้เราจะพิจารณาอัลกอริทึมสำหรับการลบโหนดและการใส่โหนดในรายการโยงคู่อีกครั้งหนึ่ง

ข้อแรกลบโหนดที่มีตัวชี้เป็น P ออกจากรายการโยงคู่นั่นคือพอยน์เตอร์ไปยังโหนดนั้นจากโหนดหลัง (successor node) และโหนดก่อน (predecessor node) ของมันต้องกำหนดใหม่ (รูป 6-27(a))



รูป 6-27(a) ขั้นตอนการลบโหนดออกจากรายการโยงคู่

โปรซีเดอร์คือ

(a) กำหนดพอยน์เตอร์ช่วยหนึ่งตัวให้กับโหนดตัวก่อนหน้าของ P และพอยน์เตอร์ช่วยอีกหนึ่งตัวให้กับโหนดตัวถัดไปของ P
(Set an anxiliary pointer to P's predecessor and one to its successor.)

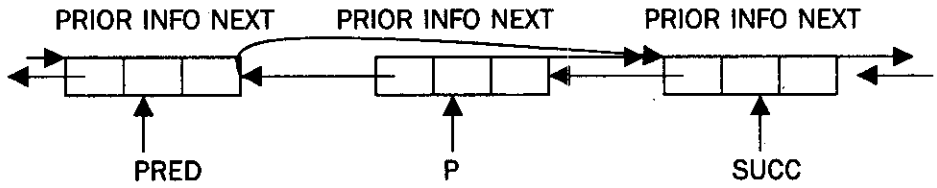
(b) เปลี่ยน Next pointer ของโหนดตัวก่อนหน้า P ให้ชี้ไปยังโหนดตัวถัดไปของ P
(Change the predecessor's Next pointer to point to P's successor, instead of to P)

(c) เปลี่ยน Prior pointer ของโหนดตัวถัดไปของ P ให้ชี้ไปยังโหนดตัวก่อนหน้า P
(Change the successor's Prior pointer to point to P's predecessor, instead of to P.)

(d) คืนเนื้อที่ซึ่งมีตัวชี้เป็น P (Free the space pointer to by P.)

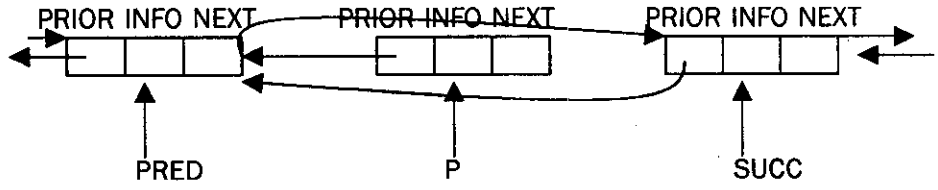
อัลกอริทึมนี้ต้องสัมผัสสามโหนด ตัวหนึ่งชี้โดย P , successor ของมันและ predecessor ของมันจงเปรียบเทียบ performance นี้กับอัลกอริทึมที่แสดงในรูป 6-22 ซึ่งลบโหนดที่มีตัวชี้เป็น P ออกจากรายการโยงเดี่ยว

ขั้น (a) และ (b)มีโครงสร้างซึ่งแสดงในรูป 2-27 (b)



รูป 6-27(b) ขั้นตอนการลบโหนดออกจากรายการโยงคู่

และหลังจากขั้น (c) มีโครงสร้างแสดในรูป 6-2(c)



ใน Pascal อัลกอริทึมเขียนเป็นรหัสดังนี้

```
procedure removp(var p: nodeptr,out : nametype);
var pred , succ : nodeptr;
begin   pred := p↑.prior;
        succ := p↑.next;
        pred↑.next := succ;
        succ↑.prior := pred;
        out := p↑.info;
        dispose(p)
end;
```

6.8.4 การใส่โหนด (Inserting a Node)

ให้ใส่ตัวแปร IN ในรายการโยงคู่เพื่อให้โหนดตัวนี้อยู่หลังโหนดที่มีพอยน์เตอร์เป็น P ขั้นแรกสร้างโหนดใหม่ในรายการจากนั้น reset พอยน์เตอร์ให้ถูกต้องสิ่งสำคัญของอัลกอริทึมนี้คือไม่มีการทำลาย (destroy) พอยน์เตอร์ซึ่งจำเป็นต้องใช้ในภายหลัง โปรซีเดอร์คือ

(a) เอาโหนดใหม่หนึ่งตัวและใส่ข้องมูลของผู้ใช้ในโหนดตัวนี้
(Get a new node and fill it with the user's data.)

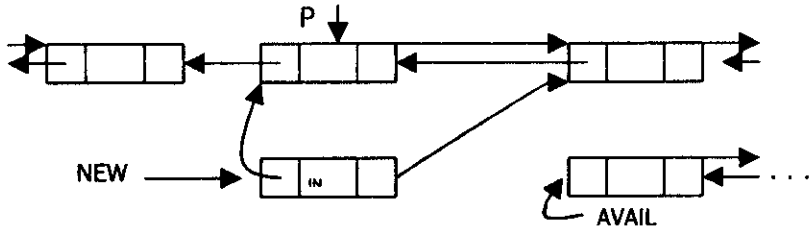
(b) ให้ Next พอยน์เตอร์ ของโหนดใหม่ชี้ไปยังโหนดหลังของ P และให้ พอยน์เตอร์ Prior ของโหนดใหม่ชี้ไปยัง P
(Set the new node's Next pointer to point to P's successor, and its Prior pointer to point to P.)

(c) เปลี่ยน พอยน์เตอร์ Next ของ P ให้ชี้ไปยังโหนดใหม่ (Change P's Next pointer to the new node.)

(d) เปลี่ยน พอยน์เตอร์ priorของ โหนดหลังของโหนดใหม่ ให้ชี้ไปยังโหนดใหม่
(Change the new node's successor's Prior pointer to point to the new node instead of to P.)

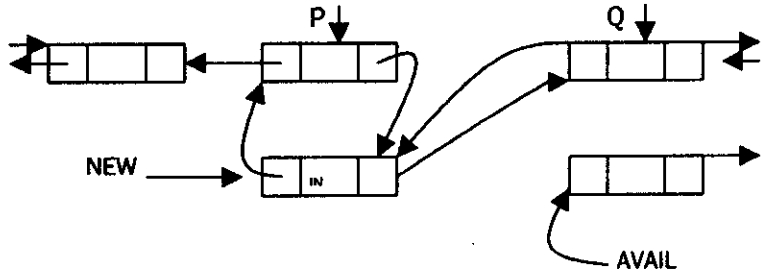
การทำให้เกิดผลในทางปฏิบัติขั้น (d) ต้องกำหนดพอยน์เตอร์ช่วยให้กับโหนดหลังไปของโหนดใหม่ ดังนั้นอัลกอริทึมจึงสัมผัสสามโหนดคือ โหนดตัวหนึ่งมีตัวชี้เป็น P โหนดหลังของตัว P และโหนดใหม่

ในทางตรงกันข้ามอัลกอริทึมใส่โหนดใหม่ หลังโหนดที่มีตัวชี้เป็น P ในรายการโยงเดี่ยวสัมผัสเพียงสองโหนดเท่านั้นโดยไม่จำเป็นต้องเข้าถึงโหนดตัวถัดไปของ P หลังจากขั้น (b) โครงสร้างแสดงในรูป 6-28(a)



รูป -28(a) ขั้นตอนการใส่โหนดในรายการโยงคู่

หลังจากขั้น(d)โครงสร้างแสดงในรูป



รูป 6-28(b) ขั้นตอนการใส่โหนดในรายการโยงคู่

ใน COBOL :

```
INSERAF.
        COMPUTE NEW = AVAIL.
        COMPUTE AVALI = NEXTNODE(AVAIL).
        MOVE IN TO INFO(NEW).
        COMPURTE NEXTNODE(NEW) = NEXTNODE(P).
        COMPUTE PRIOR(NEW) = P.
        COMPUTE NEXTNODE(P) = NEW.
        COMPUTE Q = NEXTNODE(NEW).
        COMPUTE PRIOR(Q) = NEW.
```

ให้นักศึกษาตรวจสอบว่าอัลกอริทึมนี้ทำงานได้ (works) ถึงแม้ว่าเมื่อเริ่มต้นรายการโยงคู่จะเป็นรายการว่างก็ตาม

กิจกรรม จงพัฒนาอัลกอริทึมที่ได้แนะนำมาแล้วในหัวข้อก่อนหน้านี้กับรายการโยงคู่ อัลกอริทึมเหล่านั้นควรจะถูกต้องอย่างทั่วไปเท่าที่จะเป็นไปได้ นั่นคืออัลกอริทึมหนึ่งชุดไม่ควรกระทำกับรายการว่างและมีอัลกอริทึมอีกหนึ่งชุดกระทำสิ่งเดียวกันกับรายการที่มีสมาชิกอัลกอริทึมโดยตัวมันเองควรจะจัดการกรณีต่าง ๆ ได้อย่างมากส่วน trade-off พื้นฐานระหว่าาการใช้รายการโยงเดี่ยวกับรายการโยงคู่คือ หน่วยเก็บพอยน์เตอร์(pointer stroage) และ maintenance costs กับความง่ายของการจัดการรายการ

6.9 ตัวอย่างงานประยุกต์ของรายการโยง (Example Applications of Linked Lists)

ขณะนี้จะพิจารณางานประยุกต์หลยอย่างซึ่งทำให้ใช้ได้ดีกับโครงสร้างข้อมูลรายการโยงตัวอย่างแรกใช้รายการโยงแทนพหุนาม (polynomials) ตัวอย่างที่สองใช้รายการโยงแทนข้อมูลสายอักขระเรียงลำดับตามตัวอักษร (alphabetically ordered string data) และตัวอย่างที่สามแสดงให้เห็นว่ารายการโยงสามารถนำมาใช้แทนแถวลำดับสพาซได้อย่างไร

6.9.1 พหุนาม (Polynomials)

พหุนามหมายถึงนิพจน์พีชคณิตมีรูปแบบดังนี้

$$a_n x^n + a_{n-1} x^{n-1} + \ldots + a_2 x^2 + a_1 x + a_0$$

เมื่อ a_i แต่ละตัวหมายถึงสัมประสิทธิ์ของเลขยกกำลังที่สมนัยกันของตัวแปร x
(Each a_i is a coefficient of the coresponding power of the x variable.)

ตัวอย่างเช่น พหุนาม

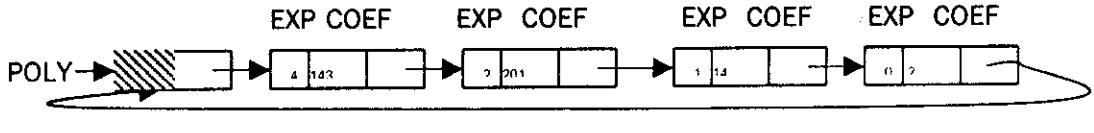
$$143x^4 + 201x^2 + 14x + 2$$

ในที่นี้ $a_4 = 143$, $a_3 = 0$, $a_2 = 201$, $a_1 = 14$ และ $a_0 = 2$
ให้พหุนามชื่อ POLY1

พหุหามถูกนำมาใช้ทั้งปัญหาทางวิทยาศาสตร์และปัญหาเชิงธุรกิจ ภาษาโปรแกรมซึ่งใช้งานทั่วไปเช่น Pascal, PL/1, COBOL และ FORTRAN ไม่มีข้อมูลชนิด buitlt-in หรือฟังก์ชันเพื่อคลุมแต่งพหุนามโดยตรง ดังนั้นวิธีปกติคือแทนพหุนามโดยใช้แถวลำดับหรือรายการโยง โครงสร้างโหนดที่เหมาะสม (ภาษา Pascal) คือ

```
type nodeptr = ↑.nodetype;
        nodetype = record
                        exp : integer;
                        coef : integer;
                        next : nodeptr
                   end;
```

เมื่อ exp หมายถึงเลขยกกำลังของตัวแปร (variabie's exponent)
coef หมายถึงสัมประสิทธิ์ที่สมนัยกัน (corresponding coeffeient)
และ next เชื่อมโหนดนี้กับโหนดถัดไป ในการแทนที่พหุนามเฉพาะเทอมที่มีสัมประสิทธิ์ไม่เท่ากับศูนย์เท่านั้นที่ต้องถูกแทนที่
เพื่อความสะดวกจึงเชื่อมโหนดแบบวงกลมโดยเรียงลำดับค่า exp ของเทอมต่าง ๆ จากมากไปหาน้อย (รูป 6-29)



รูป 6-29 ตัวอย่างการแทนที่พหุนาม

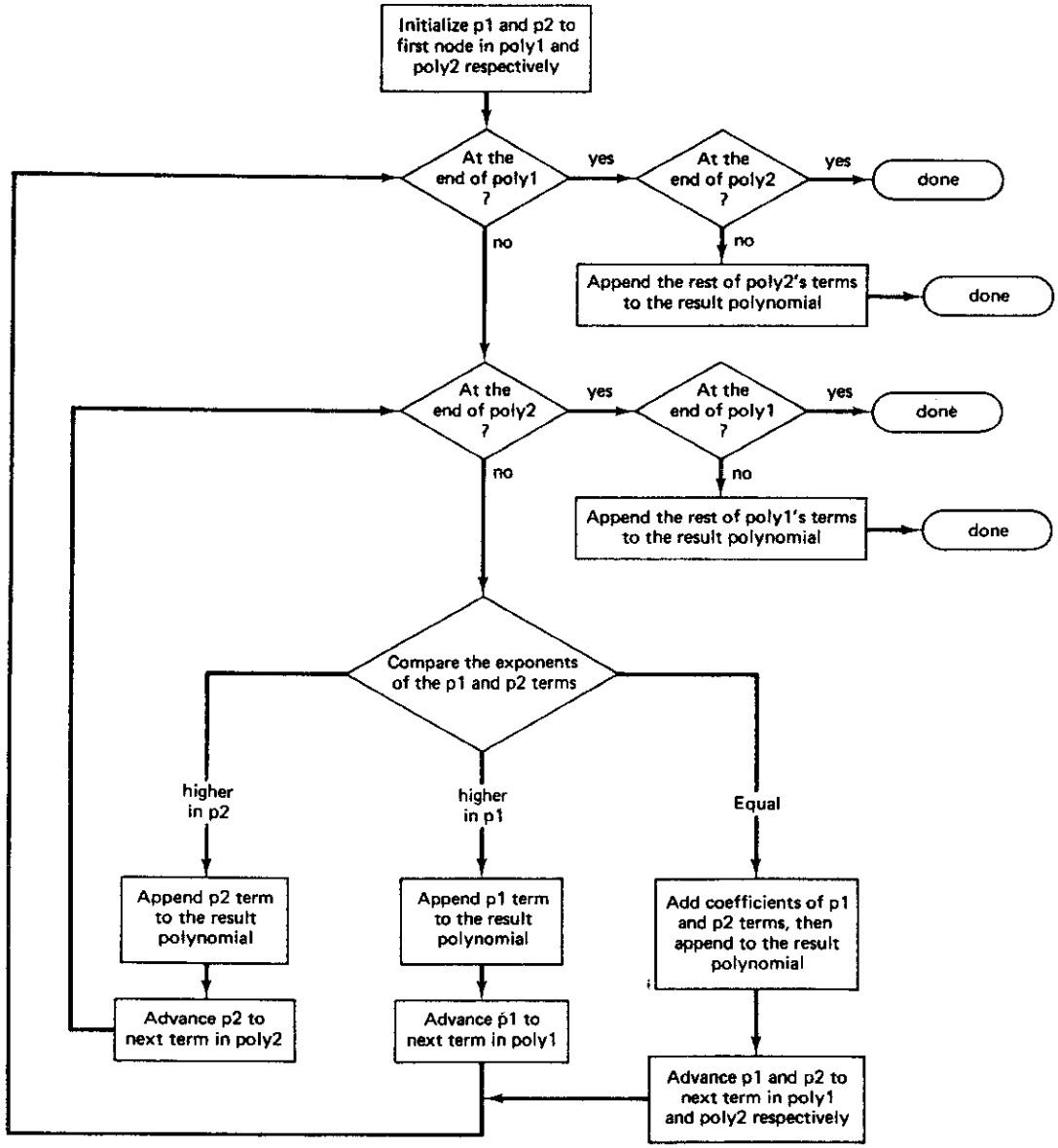
การบวก(หรือการลบ) พหุนามสองชุดต้องทำการบวก(หรือการลบ)สัมประสิทธิของเทอมที่มีเลขยกกำลังจับคู่กัน(matching exponents)

$$\begin{array}{rl} & 143x^4 \quad + 201^2 + 14x + 2 \\ + & \quad +312x^3 - 21x^2 + 42 \\ \hline \end{array}$$

ผลลัพธ์ เป็นดังนี้ $143x^4 + 312x^2 + 180x^2 + 14x^{+44}$

อัลกอริทึมเพื่อบวกพหุนาม POLY1 และ POLY2ให้เป็น POLYSUM อยู่ในรูป 6-30 หลังจากนั้นเป็น implementation ของอัลกอริทึมในภาษา Pascal โหนดหัวเรื่องถูกส่งไปยังโปรซีเดอร์ polyadd

poly1 และ Poly2 เป็นพอยน์เตอร์ของโหนดหัวเรื่องของพหุนาม POLY1 และ POLY2ตามลำดับ



รูป 6-30 ตรรกะการบวกพหุนามสองชุด

Polysum ชี้ที่โหนดหัวเรื่องของพหุนาม POLYSUM ซึ่งถูกสร้างโดยโปรซีเดอร์ พอยน์เตอร์ p1,p2 และ p3 เป็นแอดเดรสของเทอมกัจจุบันที่กำลังพิจารณาใน POLY1,POLY2 และ POLYSUM ตามลำดับ

ตัวชี้ macth นำมาใช้เพื่อแสดงถึงเงื่อนไขต่อไปนี้

exp1 high : เลขยกกำลังของ x ใน POLY1 ไม่ปรากฏใน POLY2

expmatch : เลขยกกำลังของ x ปรากฏใน POLY1 และ POLY2 ทั้งคู่

exp2high : เลขยกกำลังของ x ใน POLY2 ไม่ปรากฏใน POLY1

เครื่องหมาย end1 และ end2 นำมาใช้เพื่อให้สัญญาณว่าเป็นเทอมสุดท้ายของ POLY1 และ POLY2 ตามลำดับได้รับการประมวลผลแล้ว

ถ้า POLY1 จบ terminatesก่อน POLY2 เช่น

(POLY1) $14X^4 + 3X3$

+ (POLY2) $7X3 + X^4 + 3$

หลังจากนั้นเทอมต่าง ๆ ของ POLYSUM จาก exp = 2 เพียงแต่คัดลอก(copied) จาก POLY2 และเป็นจริงในทำนองกลับกัน (and vice versa) ถ้า POLY2 จบก่อน POLY1

โปรซีเดอร์ sumnode ใช้เพื่อสร้างโหนดต่าง ๆ ของพหุนามผลลัพธ์ POLYSUM พารามิเตอร์ส่งมายังพหุนามชุดนี้แสดงว่ามีเทอมที่กำลังถูกนำมาจาก POLY1 หรือ POLY2 และให้สัญญาณกับโปรซีเดอร์ว่าเทอมสุดท้ายของพหุนามนั้นได้รับการประมวลผลแล้ว

Nodeptr เป็นชนิดตัวแปรส่วนกลางซึ่งรู้จักแล้ว (Nodeptr is a globally known data type)

```
procedur polyadd (poly1, poly2 : nodeptr; var polysum : nodeptr);
type match = (edxp1 high , expmatch , exp2high);
var p1, p2 , psum : nodeptr;
        Expflag : match;
        End1 , end2 : 0. . 1;

procedure sumnode(poly : nodeptr ; var p , psum : nodeptr endflag : 0. . 1);
{sub procedure declaration}
var pnew : nodeptr;
begin new (pnew) ; {add new node to sumlist}
        psum ↑. next := pnew;
                psum := pnew;
        psum↑. exp := p ↑. exp;
        psum↑.coef := p ↑. coef;
                p := p ↑. next ; {move p to next node on poly}
        if p = poly then endflag := 1 {check if at end of poly}
end; {subnode}
begin end1 := 0; {main program block}
      end2 := 0;
      p1 := poly1 ↑. next;
      p2 := poly2 ↑. next;
      psum:= polysum;
if (p1 = poly1) then end1 := 1; {have reached end of poly1}
if (p2 = poly2) then end2 := 1: {have reached end of poly2}
```

```
while (end1 = 0 and end2 = 0 ) {there are still terms to add}
do begin if (p1↑.exp > p2) {pick term with greater exponent}
        then expflag := exp1 high{as next term to add}
        else if (p1 ↑. exp = p2↑. exp)
      then expflag := expmatch
      else expflag := exp2high;
      case expflag of
      exp1hieg : sumnode(poly1, p1 , psum , end1) : {add poly1 term}
      expmatc : begin sumnode (poly1, p1, psum , end1);
                                        {add poly1 term}
        psum↑ . coeff := psum↑. coef + p2↑. coef;
                                {add poly2 term}
              p2 := p2 ↑. next ;
              if p2 = poly2 then end2 := 1 {end of poly2}
              end;
        exp2high : sumnode(poly2 , p2 , psum , end 2 ) {add poly2 term}
        end;
        end;
              {have reached end of at least one poly}
        while (end1 = 0 ) {have reached end of poly2}
        do sumnode (poly2 , p2 , psum , end2 ) ; {add rest of poly1 terms}
        while (end2 = 0 ) { have reached end of poly1}
        do sumnode (poly2 , p2 , psum , end2);
                    {add rest of poly2 terms}
              psum↑. next := polysum
        end;
```

โปรแกรมนี้สามารถดัดแปร (modified) เพื่อจัดกระทำกับการลบของพหุนามสองชุดได้

กิจกรรม แบบฝึกหัดที่จะเป็นประโยชน์คือ ให้เขียนโปรแกรมบวกพหุนามสองชุดซึ่งแทนรายการโยงเก็บในแถวลำดับไม่ใช่การปฏิบัติการให้เกิดผลโดยใช้ตัวแปรพอยน์เตอร์ โปรดสังเกตว่าถ้ารายการโยงไม่ใช่แบบวงกลม อัลกอริทึมที่เขียนนั้นจะแตกต่างกันอย่างไร ถ้ารายการโยงไม่มีโหนดหัวเรื่องถ้าเป็นรายการโยงคู่ทำไมรายการโยงเดี่ยวจึงถูกเลือกเป็นโครงสร้างข้อมูลในที่นี้อัลกอริทึมสำหรับการคูณพหุนาม , การหารพหุนามคืออะไรจงพิจารณาปัญหาการคลุมแต่งพหุนามที่มีสองตัวแปรชื่อ x และ y อีกครั้งหนึ่งที่รายการโยงคือโครงสร้างข้อมูลที่เหมาะสมเพื่อแทนแบบชนิดข้อมูลนามธรรมนี้

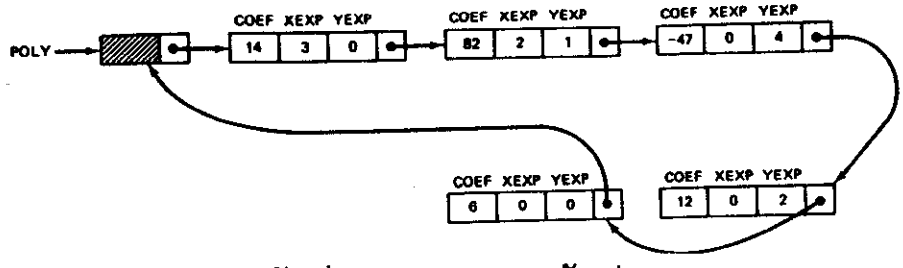
แต่ละเทอมถูกแทนด้วยโหนดชนิดข้างล่างนี้

```
type nodeptr = ↑ . nodetype;
     nodetype = record
                    coef : integer;
                    xexp : integer;
                    yexp : integer;
                    next : integer;
                end;
```

เฉพาะเทอมที่มีสัมประสิทธิ์ไม่เท่ากับศูนย์เท่านั้นซึ่งถูกแทนที่ อัลกอริทึมเพื่อคลุมแต่งการแทนที่พหุนามเหล่านี้ถูกทำให้ง่ายโดยให้เทอมต่าง ๆ โยงถึงกันในการเรียงอันดับเฉพาะอย่าง เช่น เรียงลำดับโดยเลขยกกำลัง x ก่อน แล้วจึงเรียงอันดับเลขยกกำลัง y

$$14x^3 + 82x^2y - 47y^4 + 12y4 + 6$$

แสดงในรูป 6-31



รูป 6-31 ตัวอย่างพหุนามแบบสองตัวแปร

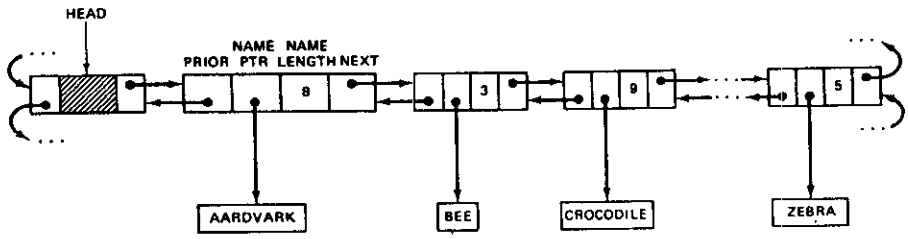
กิจกรรม

จงเขียนอัลกอริทึมบวกพหุนามแบบสองตัวแปร อัลกอริทึมนี้แตกต่างจากอัลกอริทึมซึ่งนำเสอนการบวกพหุนามแบบหนึ่งตัวแปรอย่างไร

โครงสร้างโหนดที่เหมาะสมสำหรับเทอมต่าง ๆ ของพหุนามแบบสามตัวแปรคืออะไร แบบ n ตัวแปรคืออะไร

6.9.2 รายการหลายตัวโยงอย่างง่าย (A simple multi-linked list)

ซึ่งจะอภิปรายรายละเอียดมากขึ้นในภายหลัง มีหลายกรณีซึ่งมีคาวมเหมาะสมที่จะให้รายการโยงชี้ เหนึ่งโหนดซึ่งมีความเหมาะสมที่จะใช้รายการโยงซึ่งหนึ่งโหนดมีพอยน์เตอร์มากกว่าสองตัว ตัวอย่างง่าย ๆ ของ โครงสร้างข้อมูลซึ่งหนึ่งโหนดมีพอยน์เตอร์สามตัวคือรายการโยงคู่ของข้อมูลสายอักขระความยาวแปรได้ (variable – length string data) ซึ่งโหนดแต่ละตัวเก็บพอยน์เตอร์ไปยังข้อมูลจริง (actual data) ไม่ใช่ค่าข้อมูล (data value)ดูรูป 6 - 32



รูป 6 - 32 ตัวอย่างรายการหลายตัวโยง

ข้อดีสำคัญของการใช้โครงสร้างข้อมูลในที่นี้คือ โหนดตัวแรกของรายการโยงคู่สามารถมีความยาวคงที่ โหนดแต่ละตัวมีค่าพอยน์เตอร์สามตัว และค่า integer หนึ่งตัว

การจัดการชิ้นความยาวคงที่ของหน่วยความจำที่สำคัวญคือมีความซับซ้อนน้อยกว่าการจัดการชิ้นความยาวแปรได้ของหน่วยความจำในที่นี้ยังคงมี contend เป็นชื่อจริงของสัตว์ ซึ่งความยาวผันแปร

6.9.3 แถวลำดับสพาซ (Sparse Arrays)

งานประยุกต์ปกติของรายการดยงคือการแทนที่แถวลำดับสพาซซึ่งเราได้อภิปรายการแทนที่แบบลำดับของแถวลำดับสพาซมาแล้วในตอนต้น (บทที่2)

การแทนที่แบบดยงของแถวลำดับสพาซมีศักยภาพที่สำคัญคือลดหน่วยเก็บที่จำเป็นต้องใช้และลดจำนวนของการคำนวนของการกระทำในการดำเนินงานของแวลำดับบางอย่าง

เฉพาะ entries ไม่ใช่ศูนย์ในแถวลำดับเท่านั้นซึ่งถูกแทนที่ ถ้าแถวลำดับไม่ใช่สพาซที่พอเพียงการแทนที่แบบโบงอาจจะไม่ใช่วิธีที่ดี การแทนที่แบบโยงที่เป็นไปได้อย่างหนึ่งของแถวลำดับสองมิติจะเป็นดังนี้ มีรายการดยงหนึ่งชุดสำหรับแต่ละแถว และมีรายโยงอีกหนึ่งชุดสำหรับแต่ละสดมภ์ของแถวลำดับ รายการดยงเหล่านี้แต่ละชุดมีโหนดหัวเรื่องและเป็นรายการโยงแบบวงกลมเป็นเพียงรายการโยงเดี่ยวเท่านั้น entry ไม่ใช่ศูนย์แต่ละตัวในแถวลำดับถูกแทนที่โดยโหนดของรูปแบบ(ใน pascal) ดังนี้

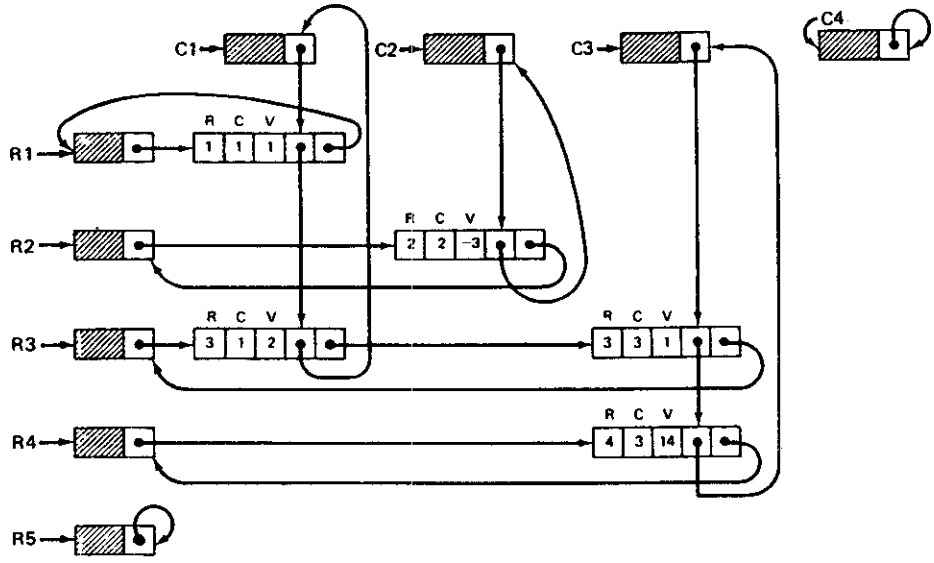
```
type nodeptr ↑ nodetype;
     nodetype = record
                    row : integer;
                    column : integer;
                    value : integer;
                    nextincol : nodeptr;
                    nextinrow : nodeptr
                end;
```

โหนดแต่ละตัวประกอบด้วยตัวแสดงแถวและสดมภ์ซึ่งเป็น nonzero entry ค่าของ entry , และพอยน์เตอร์อีกสองตัว พอยน์เตอร์ตัวหนึ่งชี้ไปยังโหนดถัดไปในแถวเดียวกัน และพอยน์เตอร์อีกตัวหนึ่ง ชี้ไปยัง โหนดถัดไปในสดมภ์เดียวกัน

ตัวอย่าง แถวลำดับขนาด 5 × 4 ข้างล่างนี้

1	0	0	0
0	-3	0	0
2	0	1	0
0	0	14	0
0	0	0	0

การแทนที่แสดงในรูป 6-33 โหนดแต่ละตัวซึ่งไม่ใช่โหนดหัวเรื่องเกี่ยวข้องกับรายการโยงสองชุดรายการโยงหนึ่งชุดสำหรับแถวของมัน และรายการโยงอีกหนึ่งชุดสำหรับสดมภ์ของมันในตัวอย่างนี้มี nonzero rows และ nonzero columns 3 สดมภ์, มี non-null list nonzero entries 5 ตัวในแถวลำดับและมี head nodes entries 9 ตัว และ 3 entry ซึ่งเป็นศูนย์ 15 ตัวไม่ต้องใช้เนื้อที่หน่วยเก็บ



รูป 6-33 การแทนที่แบบโยงของแถวลำดับสพาช

INFO NEXT

เทคนิคเฉพาะด้านสำหรับการแทนที่แถวลำดับนี้ค่อนข้างน่าพอใจเมื่อมีการปรับแถวลำดับให้เป็นปัจจุบันอย่างสม่ำเสมอ

โครงสร้างแบบโยงทำให้ง่ายต่อการทำการเปลี่ยนแปลงทั้ง entry ที่เป็นศูนย์ และไม่ใช่ศูนย์ (nonzero entry)

กิจกรรม

ให้นักศึกษาพัฒนาอัลกอริทึมบวกแถวลำดับสพาซสองชุดและกระทำการคูณเมทริกซ์สพาซสองชุด อะไรคือโครงสร้างที่เหมาะสมสำหรับการแทนแถวลำดับสพาซสาม-มิติ , แถวลำดับสพาซ n-มิติ

บทสรุป

รายการโยง หมายถึงวิธีแบบไม่ใช่ลำดับเพื่อแทนโครงสร้างข้อมูลแบบเชิงเส้น โหนดแต่ละตัวของรายการโยงจะประกอบด้วยพอยน์เตอร์อย่างน้อยที่สุดหนึ่งตัวซึ่งโยงโหนดตัวนั้นกับโหนดตัวถัดไปบนรายการ รายการโยงอาจเป็นแบบวงกลมในกรณีที่โหนดตัวสุดท้ายชี้ไปยังโหนดตัวแรก รายการโยงปกติปฏิบัติให้เกิดผลด้วยโหนดหัวเรื่องเพื่อให้อัลกอริทึมง่ายต่อการกระทำการใส่และการลบจากรายการโยงบ่อยครั้งจะใช้ตัวชี้ไปยังโหนดตัวก่อนหน้าเช่นเดียวกับ โหนดถัดไปรายการผลลัพธ์เรียกว่ารายการโยงคู่ (doubly linked list)

เมื่อกำหนดให้โหนดตัวหนึ่งสามารถมีส่วนร่วมพร้อมกันในรายการโยงมากกว่าหนึ่งชุดโครงสร้างนี้เรียกว่ารายการหลายตัวโยง (a multi-linked list) ซึ่งจะได้ศึกษาโครงสร้างเหล่านี้ในรายละเอียดมากขึ้นในภายหลัง

เราได้เห็นแล้วว่าการจะใช้รายการโยงเพื่อแทนที่ไม่เพียงแค่รายการเชิงเส้นทั่วไปเท่านั้นแต่รายการเชิงเส้นชนิดพิเศษอื่น ๆ ได้ด้วย โดยเฉพาะกองซ้อนและแถวคอย เราพิจารณาการแทนที่งานประยุกต์หลายชนิดที่ใช้รายการโยง

รายการโยงเดี่ยวเหมาะสมสำหรับการแทนที่พหุนามรายการดยงคู่สามารถนำมาใช้แทนสายอักขระคามยาวแปรได้ โครงสร้างรายการหลายตัวโยงแบบสองมิติสามารถนำมาใช้แทนแถวลำดับสพาซ ข้อดีพื้นฐานของการใช้รายการโยงคือมันไม่ต้องใช้หน่วยเก็บแบบลำดับ ดังนั้นจึงเป็นข้อดีที่ใช้รายการดยงเมื่อมีกิจกรรมการใส่และการลบที่สำคัญในโครงสร้างข้อมูลแบบอันดับเชิงตรรกะ จึงเหมาะสมเช่นกันที่ใช้รายการโยงเมื่อมีความจำเป็นที่สมาชิกเรียงอันดับเชิงตรรกะไม่ใช่แบบลำดับเชิงกายภาพ

ตัวอย่างเช่น การทำให้เกิดผลในทางปฏิบัติกองซ้อนของพื้นที่ว่างค่าใช้จ่ายหลักของการใช้

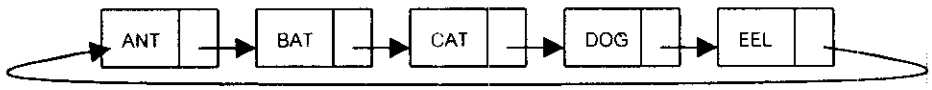
---

5. จงเขียนอัลกอริทึมส่งกลับรายการโยงแบบวงกลม T ไปยังที่พักโหนดว่าง
   (Write an algorithm to return a circular linked list T to the pool of avaliable node.)
6. ผลลัพธ์ของ Next(Prior(P)) ในรายการโยงคู่คืออะไร
   (What is the result of next(Prior(P)) in a doubly linked list?)
7. ให้ x เป็นพอยน์เตอร์ของโหนดใด ๆ ก็ได้ในรายการโยงเดี่ยวจงเขียนอัลกอริทึมซึ่งใส่ '1234' ในเขต info ของโหนดใหม่และให้ใส่โหนดตัวนี้หลังโหนดที่มีตัวชี้เป็น x
8. จงเขียนอัลกอริทึมใส่โหนด S เข้าไปในรายการโยงคู่ทันที่ก่อนโหนด x
   (Write an algorithm that will insert node S into a doubly linked list immediately before node x.)
9. จงเขียนอัลกอริทึมย้อนลำดับ รายการโยงเดี่ยวไม่ใช่แบบวงกลม, รายการโยงคู่

15. สพาซเมทริกซ์ข้างล่างนี้จะถูกแทนที่ด้วยรายการโยงอย่างไร
(How can the following sparse matrix be represented by linked lists?)

0	0	5	2
0	10	14	0
0	0	0	2
0	0	0	0

16. จงเขียนอัลกอริทึมเพื่อลบโหนดที่มีค่า 'cat' ออกจากรายการโยงข้างล่างนี้



17. จงเขียนอัลกอริทึมเพื่อแบ่ง (split) รายการโยงข้างต้นระหว่าง 'BAT' และ 'CAT' และให้ใส่โหนดหัวเรื่องจากรายการว่าง AVAIL ที่ตอนต้นของรายการใหม่

18. จงแสดงให้เห็นการใช้รายการโยงเพื่อแทนที่พหุนาม $5x^3 + 7x^2 + 9$ และ $3x^2 + 4x + 7$

19. จงแสดงให้เห็นว่าจะใช้รายการโยงเพื่อแทนที่พหุนามข้างล่างได้

$7x^3y - 3x^2y^2 + 5y + 12y^2 - 2$

$9x^3y^2 + 2x^2y^2 - 11xy + 3y^2$

20. จงเขียนอัลกอริทึมนับจำนวนโหนดในรายการโยง (Write an algorithm to count the number of nodes in alinked list.)

21. จงพิจารณาการใช้แถวลำดับเพื่อเก็บรายการโยงของสารสนเทศคงคลังของหนังสือ แถวลำดับนิยามในภาษา COBOL ดังนี้

```
01    BOOK-ARRAY.
      02    BOOK-NODE   OCCURS 15 ITMES.
            03    AUTHOR          PIC    X(20).
            03    NEXT-AUTHOR PIC    99.
            03    TITLE           PIC    X(30).
            03    NEXT-TITLE      PIC    99.
            03    STOCK-NO        PIC    9(5).
            03    NEXT-STOCK-NO PIC  99.
```


IT 204 181

03 STOCK-NO PIC 9(5).

```
        03    NEXT-AVAIL        PIC 99.
    02  HEADS.
        03    FIRST-AUTHOR      PIC 99.
        03    FIRST-TITLE       PIC 99.
        03    FIRST-STOCK-NO    PIC 99.
        03    LAST-STOCK-NO     PIC 99.
        03    AVAIL             PIC 99.
```

ข้อมูลถูกโยงเรียงลำดับตัวอักษรโดย last name ของผู้แต่ง , เรียงลำดับตัวอักษรโดย คำแรกของชื่อเรื่อง , และเรียงลำดับจากน้อยไปหามากโดย STOCK-NO จงแสดงให้เห็น contents ของ BOOK-ARRAY หลังจากเสร็จสิ้นแต่ละรายการเปลี่ยนแปลงต่อไปนี้

OPERATION	STOCK-NO	TITLE	AUTHOR
INSERT	53526	CREATIVE PHOTOGRAPHY	F.STOP FITZGERALD
INSERT	98374	NEWSPAPER DRIGAMI	LINUS TYPE
INSERT	14683	COOL HAND LUKE	HADDA DUWITT
INSERT	23764	THE LONGEST YARD	LOWEN MAUER
INSERT	49261	LA MARKE DE LA FRANCAISE	ASCENT AGU
INSERT	19822	BEER BASTED HOT DOGS	DR.FRANK ANNSTEIN
INSERT	76482	FOUR-WAY ROMANCE	QUAD LAMOORE
INSERT	17760	COMPUTER SIMULATION	ARITE ABACUS
INSERT	38641	BETSY WORE BLUE JEANS	DENN M.STRETCHER
INSERT	73920	FINGER LICKIN' GOOD	C.SANDERS
DELETE	14683		
DELETE	73482		

22. ข้อดีของการมีโหนดตัวหน้าบนรายการมีอะไรบ้างและมีข้อไม่ดีอะไรบ้าง
23. ข้อดีของรายการโยงเดี่ยวแบบวงกลมที่เหนือกว่ารายการโยงเดี่ยวแบบไม่ใช่วงกลมมีอะไรบ้าง มีข้อไม่ดีมีอะไรบ้าง
24. ข้อดีของรายการงโยงคู่แบบวงกลมที่เหนือกว่า รายการ โยงเดี่ยวแบบวงกลมมีอะไรบ้าง มีข้อไม่ดีมีอะไรบ้าง

25. จงพิจารณาแถวลำดับของโหนด 500 ตัวที่เก็บโดยรายการโยงคู่สมมติว่าโหนดว่างถูกโยงเข้าด้วยกันในรายการโยงเดี่ยว

(a) จงเขียนอัลกอทึมลบโหนดออกจากรายการโยงคู่

(b) จงเขียนอัลกอริทึมใส่โหนดเข้าไปในรายการโยงคู่

(c) จงหาค่าเฉลี่ยของจำนวนโหนดที่ต้องสัมผัสในอัลกอริทึมลบโหนด

(d) จงหาค่าเฉลี่ยของจำนวนโหนดที่ต้องสัมผัสในอัลกอริทึมใส่โหนด

(e) จงเปรียบเทียบคำตอบข้อ (a)-ข้อ(d)ในสถานะการณ์ซึ่งโหนดว่างเป็นรายการโยงคู่

183