Chapter 5 Linked List (II)

ผศ.ดร.สิลดา อินทรโสธรฉันท์

สาขาวิชาวิทยาการคอมพิวเตอร์ คณะวิทยาศาสตร์ มหาวิทยาลัยขอนแก่น

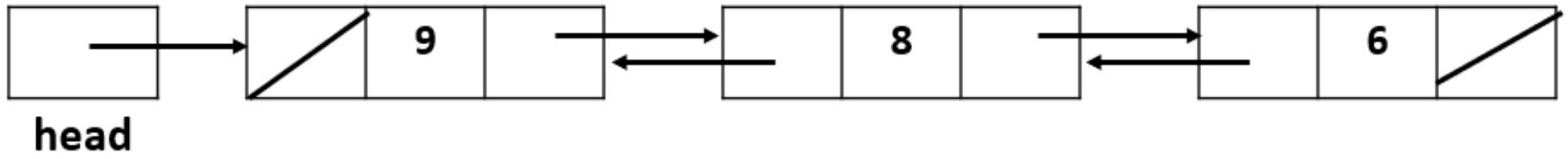
Doubly Linked – List

- แต่ละโหนดจะเก็บตำแหน่งของโหนดที่อยู่ก่อนและหลัง
- ทำให้โครงสร้างแต่ละโหนดประกอบด้วย 3 ส่วน คือ ส่วนเก็บข้อมูล (item) ตำแหน่งโหนดก่อนหน้าหรือลิงค์ทางซ้าย (Llink) และตำแหน่งโหนดถัดไปหรือลิงค์ทางขวา (Rlink)

Node

Doubly Linked – List

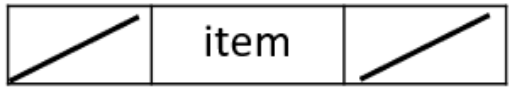
- จากโครงสร้างของโหนดสามารถนำมาเชื่อมต่อกันเป็นลิงค์ลิสต์ได้ดังภาพ



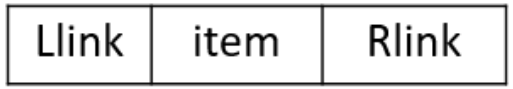
- จากภาพจะมีตัวแปร head ทำหน้าที่ในการเก็บตำแหน่งข้อมูลตัวแรกของลิงค์ลิสต์ไว้
- สมาชิกแต่ละตัวจะมีเส้นเชื่อมทั้งซ้ายและขวา เพื่อระบุตำแหน่งโหนดก่อนและหลังนั่นเอง

การสร้างโหนด

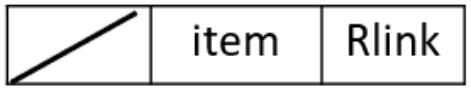
- จากลักษณะของโหนดจะมีความแตกต่างกันตามหน้าที่และตำแหน่งของโหนดในลิงค์ลิสต์ ซึ่งสามารถแบ่งเป็นหลักๆ 4 แบบคือ
  - โหนดที่เป็น head ที่ทำหน้าที่เก็บตำแหน่งของโหนด จะไม่มีการเก็บค่าข้อมูล
  - โหนดแรก จะมีค่าลิงค์ทางซ้ายเป็น NULL
  - โหนดสุดท้าย จะมีค่าลิงค์ทางขวาเป็น NULL
  - โหนดทั่วไป



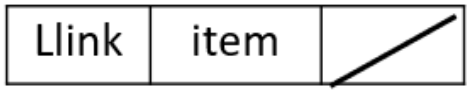
head



Node



โหนดแรก



โหนดสุดท้าย

Constructor ของคลาส DoubleLink

- จากโครงสร้างของโหนด มาใช้ในการกำหนด attribute ได้ดังนี้

```
private Object item;

private DoubleLink Rlink;

private DoubleLink Llink;
```

Constructor ของคลาส DoubleLink

- constructor สำหรับสร้าง head node

```
public DoubleLink() {
        Rlink = null;
        Llink = null;
}
```

Constructor ของคลาส DoubleLink

- constructor สำหรับสร้าง node ที่ไม่มีลิงค์ซ้ายและขวา

```
public DoubleLink (Object newItem) {
        item = newItem;
        Rlink = null;
        Llink = null;
}
```

Constructor ของคลาส DoubleLink

- constructor สำหรับสร้าง node ใดๆ

```
public DoubleLink(Object newItem, DoubleLink nextNode, DoubleLink prevNode){
        item = newItem;
        Rlink = nextNode;
        Llink = prevNode;
 }
```

เมธอดต่าง ๆ ของคลาส DoubleLink

- เมธอดในการ set และ get ค่าในส่วนของ item

```
public void setItem (Object newItem){
        item = newItem;
}
public Object getItem(){
        return item;
}
```

เมธอดต่าง ๆ ของคลาส DoubleLink

- เมธอดในการ set และ get ค่าในส่วนของ Rlink

```
public void setRlink(DoubleLink nextNode){
        Rlink = nextNode;
 }

public DoubleLink getRlink(){
        return Rlink;
}
```

เมธอดต่าง ๆ ของคลาส DoubleLink

- เมธอดในการ set และ get ค่าในส่วนของ Llink

```
public void setLlink(DoubleLink prevNode){
        Llink = prevNode;
}
public DoubleLink getLlink(){
        return Llink;
}
```

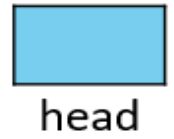
การดำเนินการบนลิงค์ลิสต์สองทาง

- การสร้างส่วนหัวของลิงค์ลิสต์ (การสร้างลิสต์)
- การค้นหาตำแหน่งที่ต้องการ
- การลบโหนด
- การแทรกโหนด

การสร้างส่วนหัวของลิงค์ลิสต์

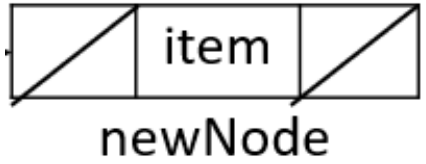
- การสร้างลิงค์ลิสต์จึงเริ่มด้วยส่วนหัวและการเพิ่มโหนด โดยมีขั้นตอนการทำงานดังนี้
- Step1: ทำการสร้างโหนด head

```
DoubleLink head = new DoubleLink();
```



- Step2: ทำการสร้างโหนดที่ต้องการเพิ่มเข้าสู่ลิงค์ลิสต์

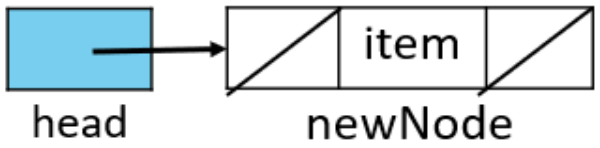
```
DoubleLink newNode = new DoubleLink(item);
```



การสร้างส่วนหัวของลิงค์ลิสต์

- Step3: ทำการกำหนดค่า head ให้ชี้ไปที่ตำแหน่งของโหนดที่เพิ่มเข้าไป

```
head = newNode;
```

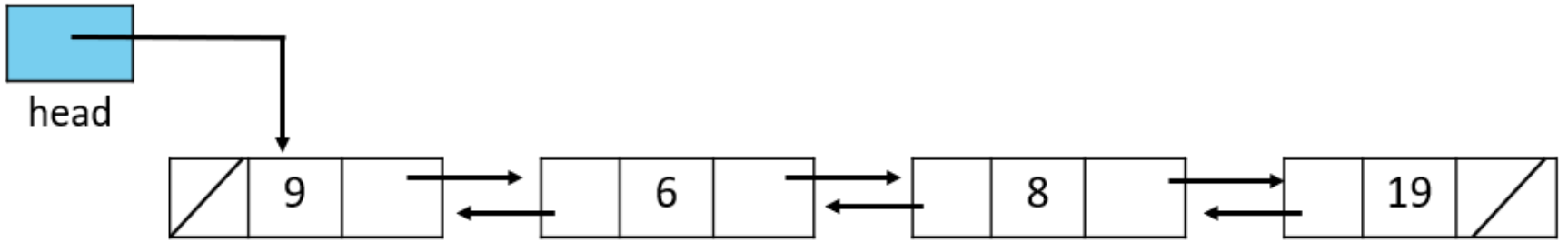


การค้นหาตำแหน่งที่ต้องการ

- ก่อนที่จะทำการลบหรือแทรกหนดในลิงค์ลิสต์ จะต้องค้นหาตำแหน่งโหนดที่ต้องการก่อน
- โดยขั้นตอนการค้นหาตำแหน่งโหนดจะมีตัวแปร 2 ตัว เพื่อใช้ในการปรับปรุงโครงสร้างลิงค์ลิสต์ คือ
  1. ตัวแปร curr ทำหน้าที่ค้นหาตำแหน่งโหนดที่ต้องการ
  2. ตัวแปร prev ทำหน้าที่อ้างอิงตำแหน่งโหนดที่อยู่ก่อนโหนดที่ค้นหา
  3. ตัวแปร next ทำหน้าที่อ้างอิงตำแหน่งโหนดที่อยู่ถัดจากโหนดที่ต้องการค้นหา
- การค้นหาข้อมูลในลิงค์ลิสต์จะเริ่มทำการค้นหาตั้งแต่ตำแหน่งแรกไปเรื่อย ๆ จนพบตำแหน่งที่ต้องการหรือถึงสมาชิกตัวสุดท้าย

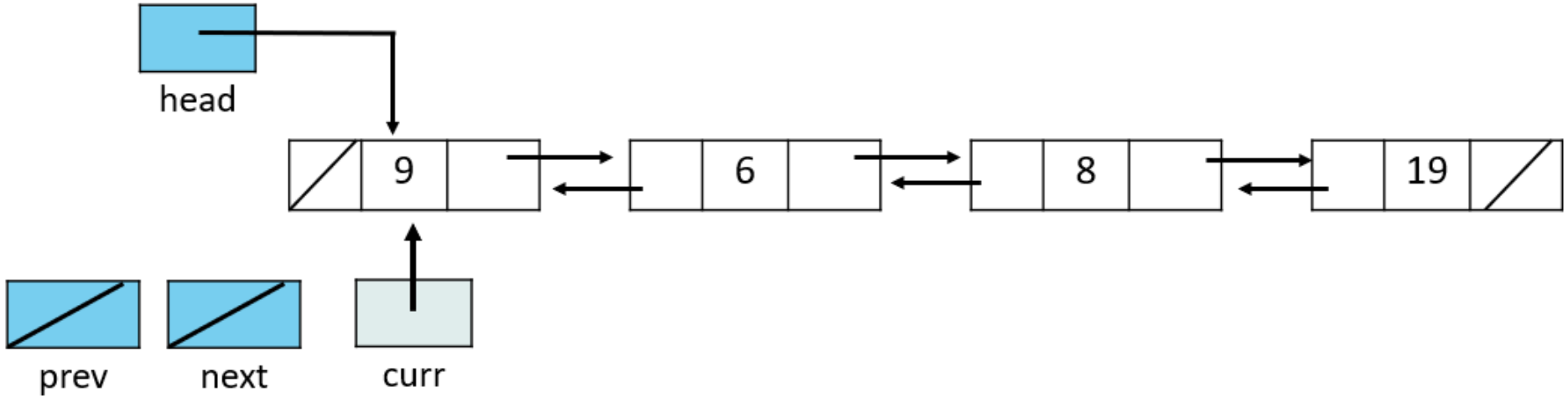
ข้อมูลที่ค้นหาอยู่ตำแหน่งแรก

- **ต้องการหาตำแหน่งที่เก็บข้อมูล 9** (ข้อมูลดังกล่าวอยู่ตำแหน่งแรก)



ข้อมูลที่ค้นหาอยู่ตำแหน่งแรก

- ในการค้นหาตัวแปร curr , prev และ next จะมีสถานะดังภาพ



ข้อมูลที่ค้นหาอยู่ตำแหน่งใดๆ ในลิงค์ลิสต์

- การค้นหาจะเริ่มจากตำแหน่งแรก เมื่อไม่พบก็จะทำการปรับค่าตัวแปร curr และตัวแปร prev
- โดยการอัพเดตค่าตัวแปร prev ให้ทำการเก็บตำแหน่งปัจจุบันไว้ แล้วทำการอัพเดตค่าตัวแปร curr ให้ไปยังโหนดถัดไปด้วยการเรียกเมธอด getRlink() ด้วยคำสั่ง

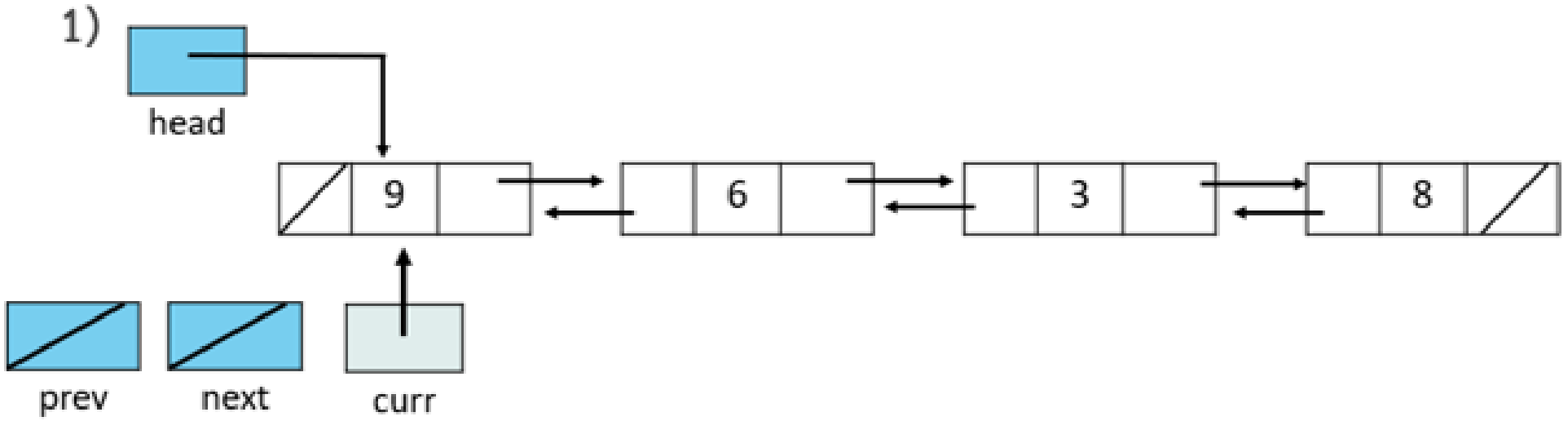
```
prev = curr;

curr = curr.getRlink();
```

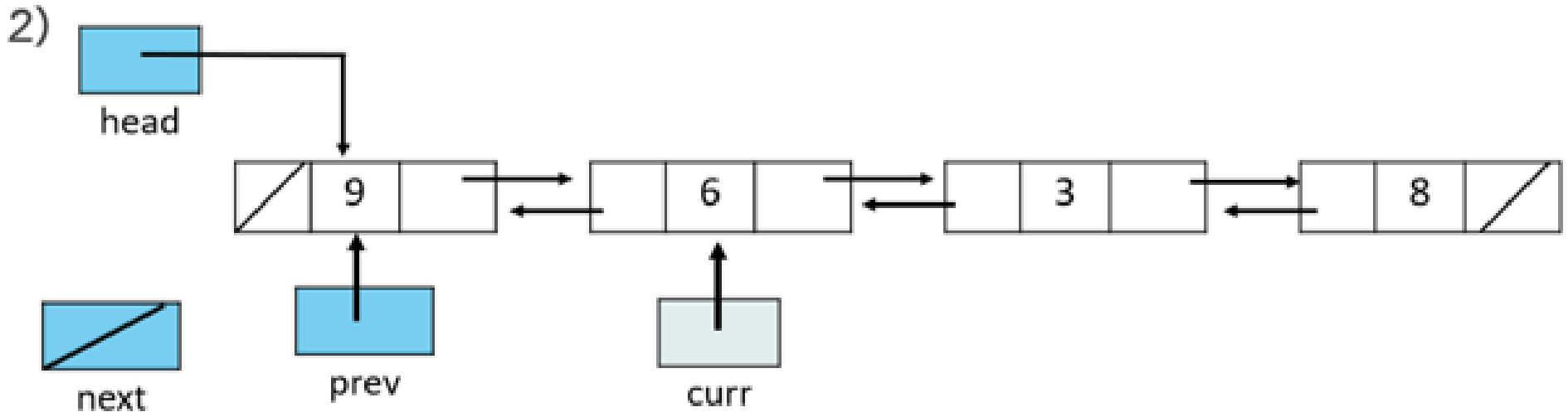
- โดยการทำงานทั้งสองขั้นตอนจะทำควบคู่กันไปจนกระทั่งพบตำแหน่งที่ต้องการ

ข้อมูลที่ค้นหาอยู่ตำแหน่งใดๆ ในลิงค์ลิสต์

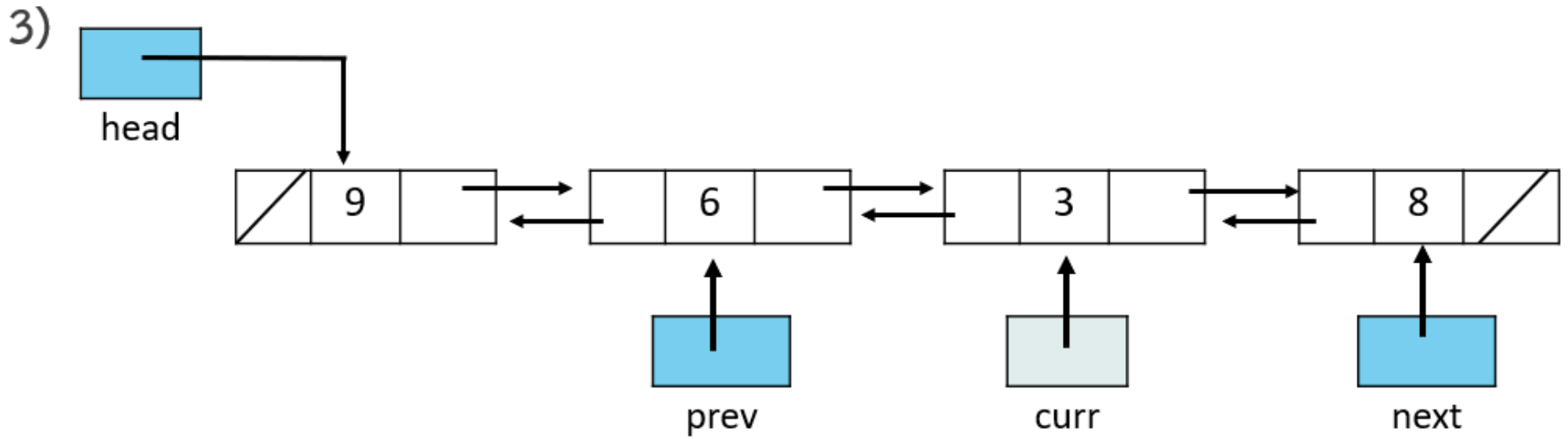
ต้องการค้นหาตำแหน่งที่เก็บข้อมูล 3



ข้อมูลที่ค้นหาอยู่ตำแหน่งใดๆ ในลิงค์ลิสต์



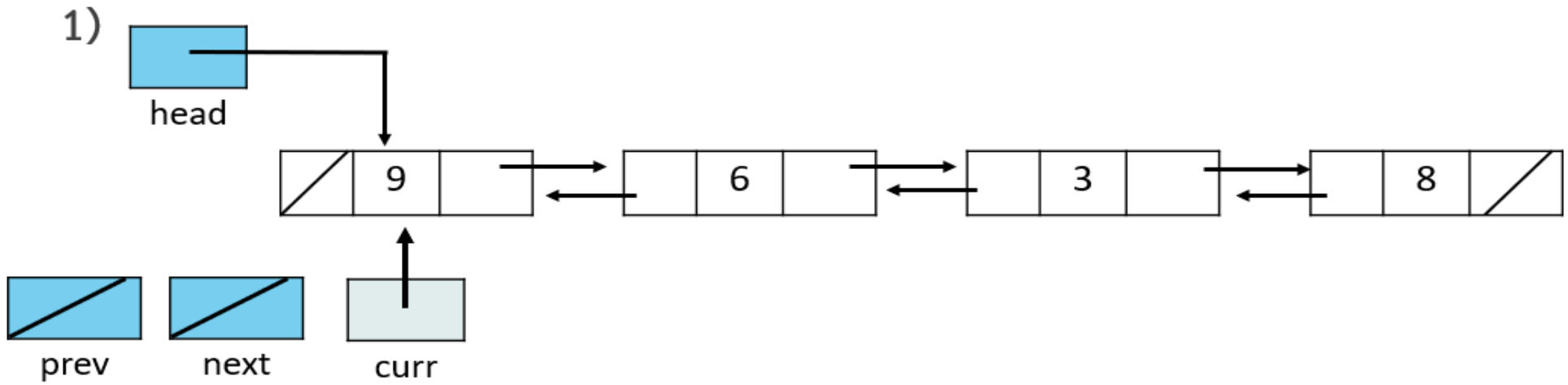
ข้อมูลที่ค้นหาอยู่ตำแหน่งใดๆ ในลิงค์ลิสต์



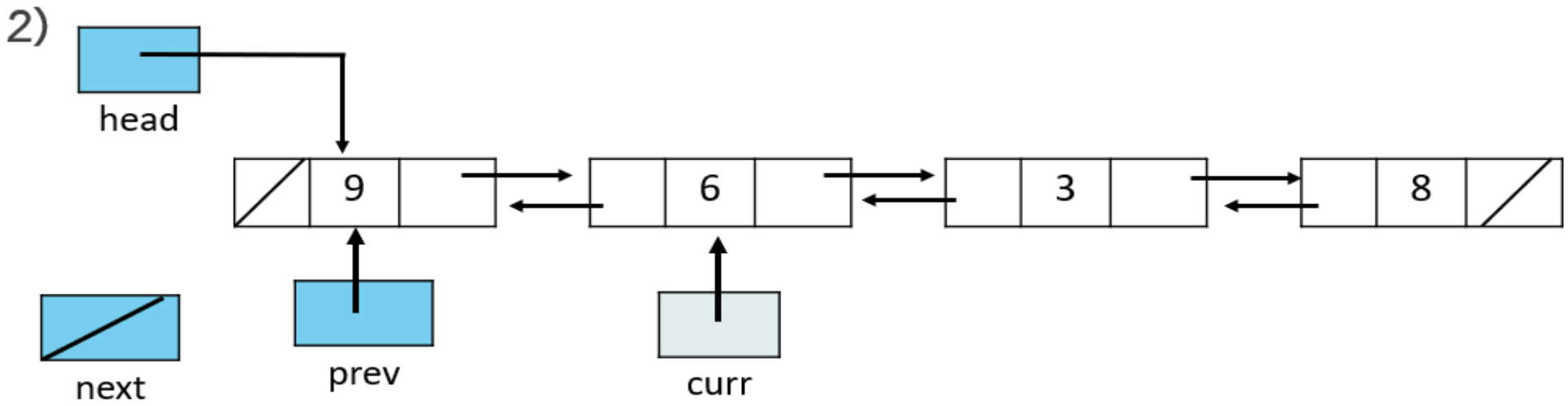
- กรณีที่ไม่พบข้อมูลเมื่อค้นหาจนถึงตำแหน่งสุดท้ายของลิงค์ลิสต์ เมื่อทำการอัพเดตค่าตัวแปรต่อ ตัวแปร curr จะมีค่าเป็น NULL

ข้อมูลที่ค้นหาอยู่ตำแหน่งใดๆ ในลิงค์ลิสต์

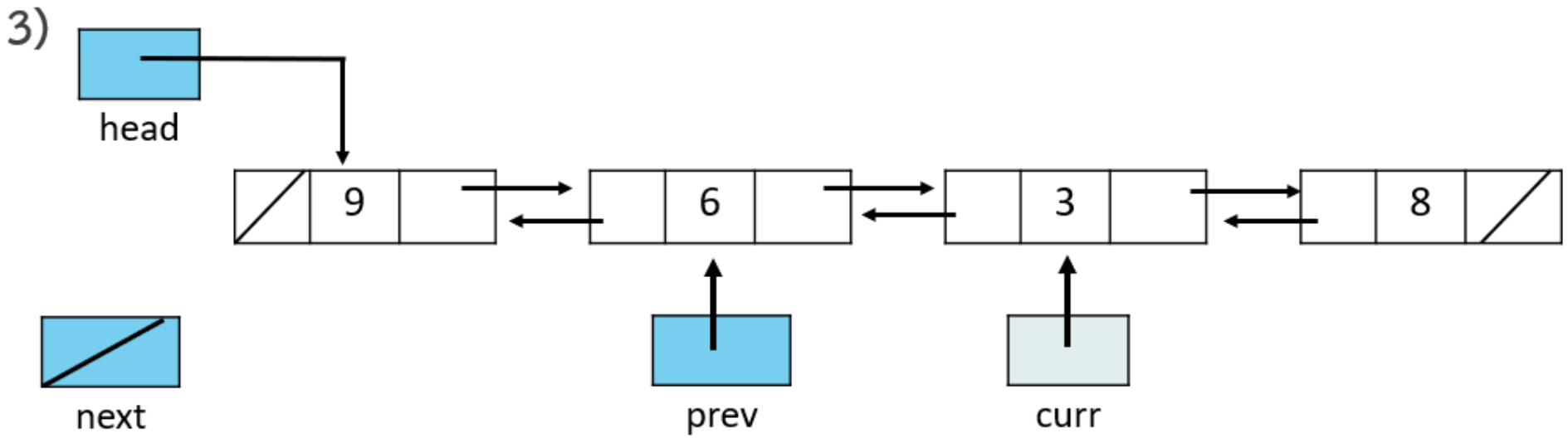
ต้องการค้นหาตำแหน่งที่เก็บข้อมูล 7 (กรณีที่ไม่พบข้อมูล)



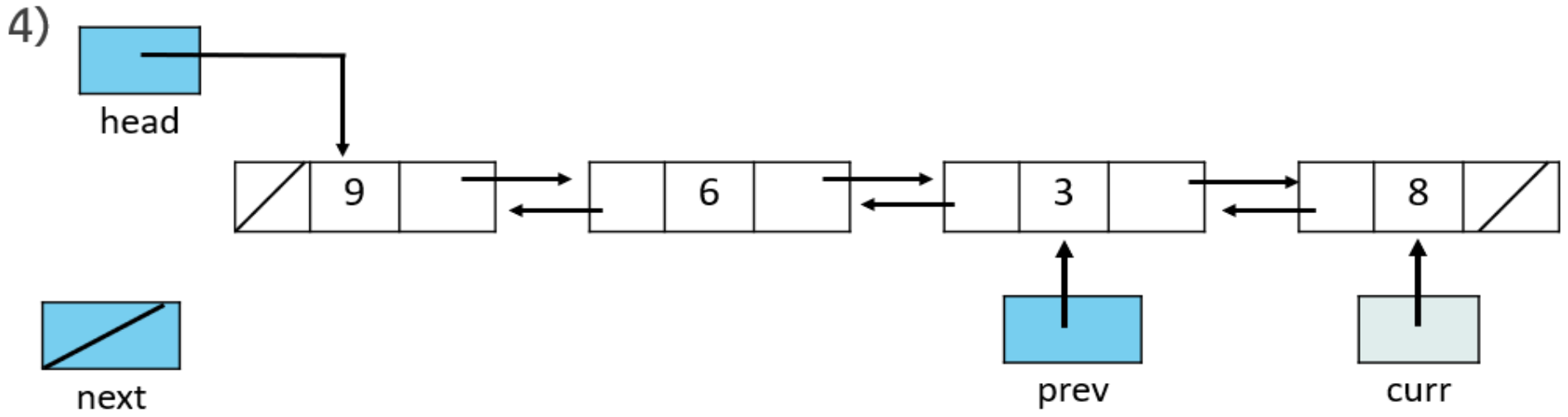
ข้อมูลที่ค้นหาอยู่ตำแหน่งใดๆ ในลิงค์ลิสต์



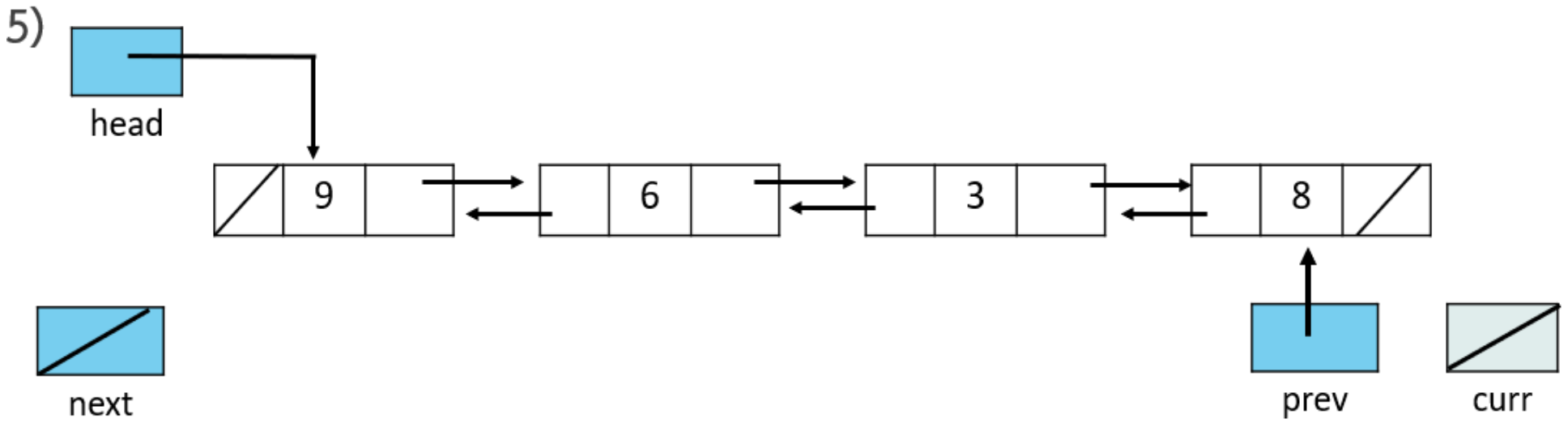
ข้อมูลที่ค้นหาอยู่ตำแหน่งใดๆ ในลิงค์ลิสต์



ข้อมูลที่ค้นหาอยู่ตำแหน่งใดๆ ในลิงค์ลิสต์



ข้อมูลที่ค้นหาอยู่ตำแหน่งใดๆ ในลิงค์ลิสต์



การลบโหนด

- การลบโหนดในลิงค์ลิสต์เป็นการยกเลิกการเชื่อมต่อไปยังโหนดที่ต้องการลบ
- เมื่อค้นหาโหนดที่ต้องการลบได้แล้วจะทำการปรับการเชื่อมโยงของโหนดที่เหลือให้ข้ามโหนดที่ลบ
- โดยตัวแปรที่จำเป็นในการค้นหาตำแหน่งคือตัวแปร curr , prev และ next

การลบโหนด

- ขั้นตอนการทำงานสามารถแยกตามตำแหน่งของการลบโหนดในลิงค์ลิสต์ได้เป็น 3 กรณี คือ
  - การลบโหนดที่อยู่ระหว่างข้อมูลที่มีอยู่ในลิงค์ลิสต์
  - ลบโหนดที่ตำแหน่งแรก
  - ลบโหนดที่ตำแหน่งสุดท้าย

การลบโหนดที่อยู่ระหว่างข้อมูลที่มีอยู่ในลิงค์ลิสต์

- เมื่อค้นหาข้อมูลโหนดที่ต้องการลบได้แล้ว จะทำการปรับค่าลิงค์ของโหนดที่เหลือ โดยมีขั้นตอนดังนี้

  Step 1: เปลี่ยนค่า Rlink ของโหนด prev ให้อ้างอิงไปยังโหนด next โดยใช้คำสั่ง

  ```
  prev.setRlink(next);
  ```

  Step2: เปลี่ยนค่า Llink ของโหนด next ให้อ้างอิงไปยังโหนด prev โดยใช้คำสั่ง

  ```
  next.setLlink(prev);
  ```

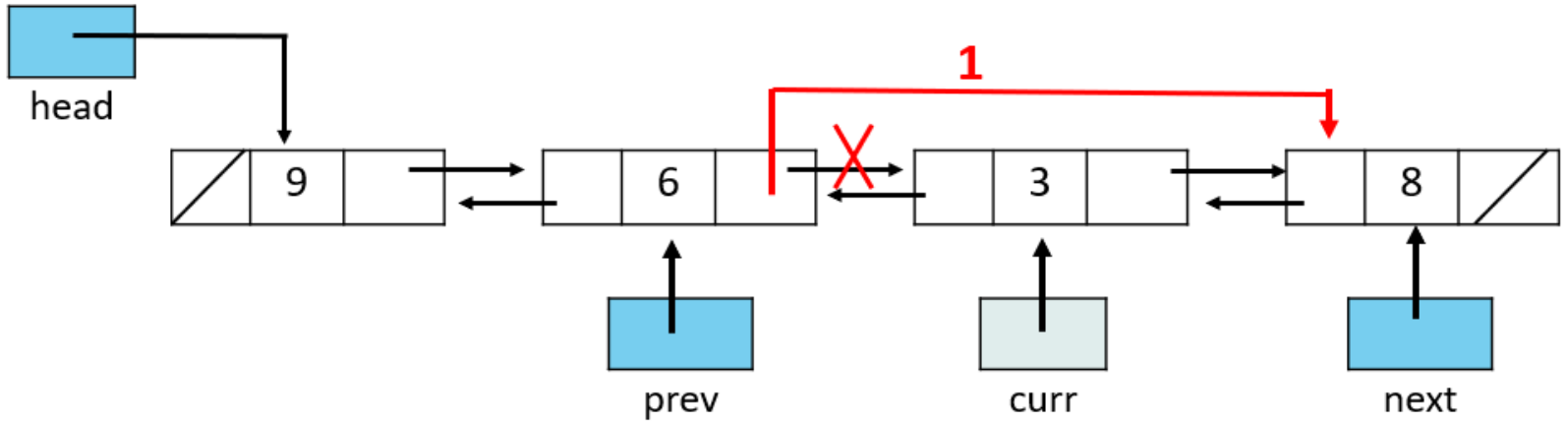
-

การลบโหนดที่อยู่ระหว่างข้อมูลที่มีอยู่ในลิงค์ลิสต์

ต้องการลบโหนดที่เก็บข้อมูล 3

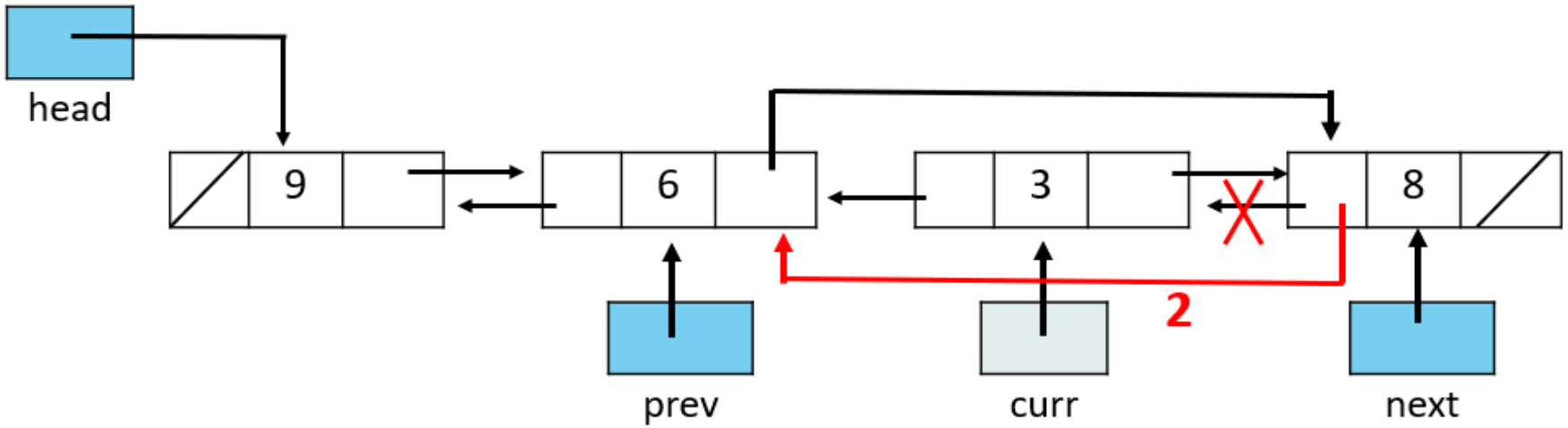
- เมื่อทำการค้นหาตำแหน่งที่ต้องการลบได้แล้ว ทำการปรับลิงค์ตาม step 1 โดยใช้คำสั่ง

prev.setRlink(next); จะปรากฏเส้นสีแดงดังรูป



การลบโหนดที่อยู่ระหว่างข้อมูลที่มีอยู่ในลิงค์ลิสต์

- Step2: เปลี่ยนค่า Llink ของโหนด next โดยใช้คำสั่ง next.setLlink(prev);



การลบโหนดที่อยู่ตำแหน่งแรกของลิงค์ลิสต์

- กรณีที่ทำการค้นหาโหนดที่ต้องการลบแล้วปรากฏว่าโหนดดังกล่าวอยู่ตำแหน่งแรกของลิงค์ลิสต์
- การลบโหนดจะมีขั้นตอนดังนี้

Step 1: อัพเดตค่าลิงค์ของ head เพื่อให้มีค่าเป็นโหนด next โดยใช้คำสั่ง

```
head = next;
```

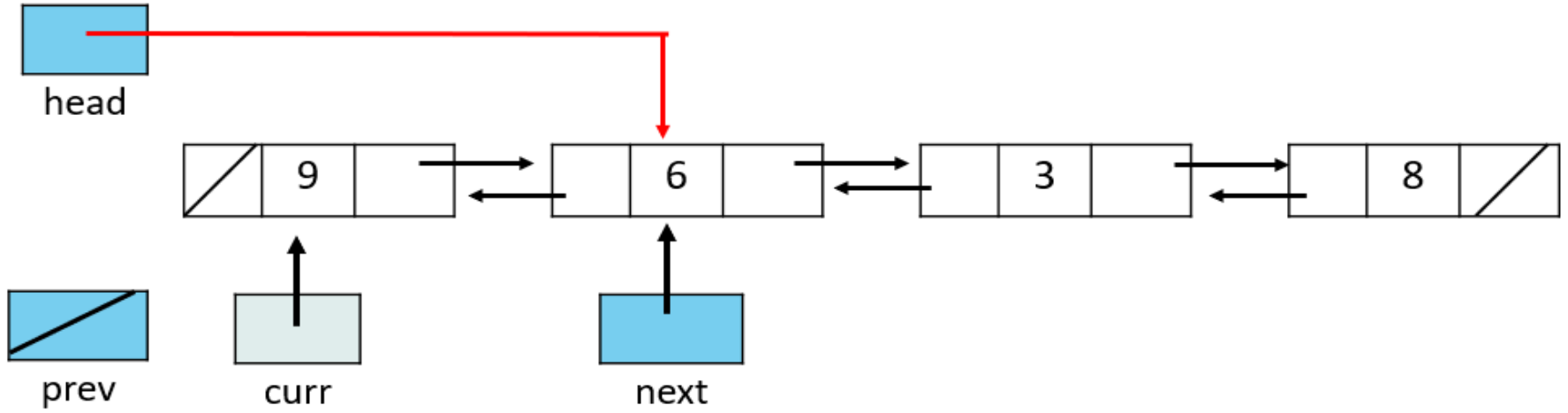
Step2: เปลี่ยนการอ้างอิง Llink ของโหนด next ให้เป็น NULL โดยใช้คำสั่ง

```
next.setLlink(null);
```

การลบโหนดที่อยู่ตำแหน่งแรกของลิงค์ลิสต์

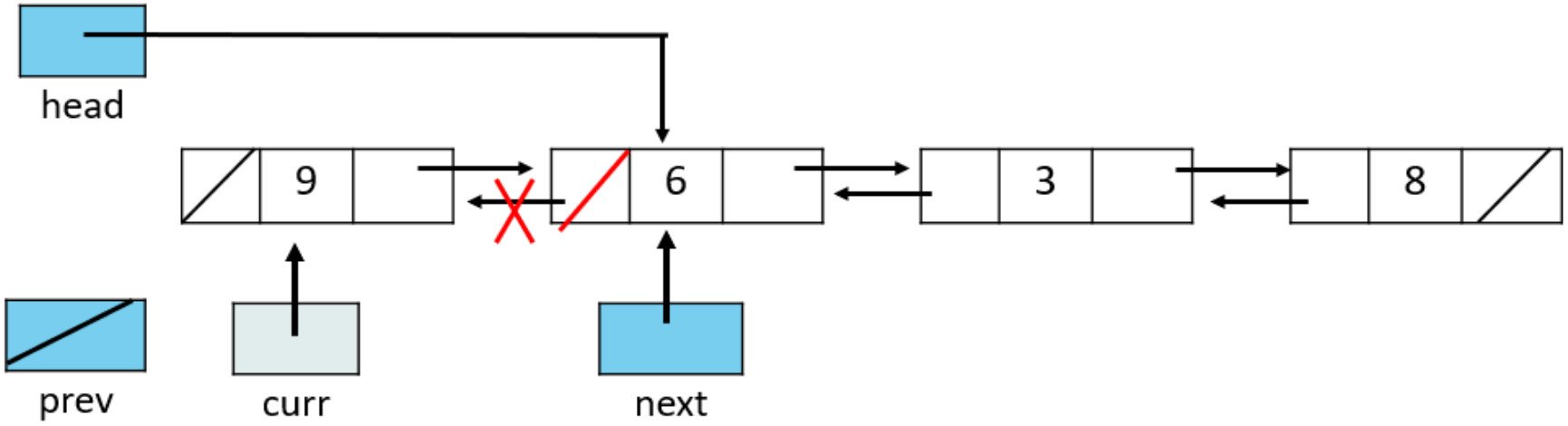
ต้องการลบโหนดที่เก็บข้อมูล 9

- เมื่อค้นหาตำแหน่งที่ต้องการได้แล้ว ตัวแปร curr จะอยู่ที่โหนดที่ต้องการลบ ส่วนตัวแปร next จะอยู่ตำแหน่งถัดไป ทำการปรับลิงค์ตาม step 1 โดยใช้คำสั่ง head = next; จะปรากฏเส้นสีแดงดังรูป



การลบโหนดที่อยู่ตำแหน่งแรกของลิงค์ลิสต์

- Step2: เปลี่ยนการอ้างอิง Llink ของโหนด next ให้เป็น NULL โดยใช้คำสั่ง next.setLlink(null);



การลบโหนดที่อยู่ตำแหน่งสุดท้ายของลิงค์ลิสต์

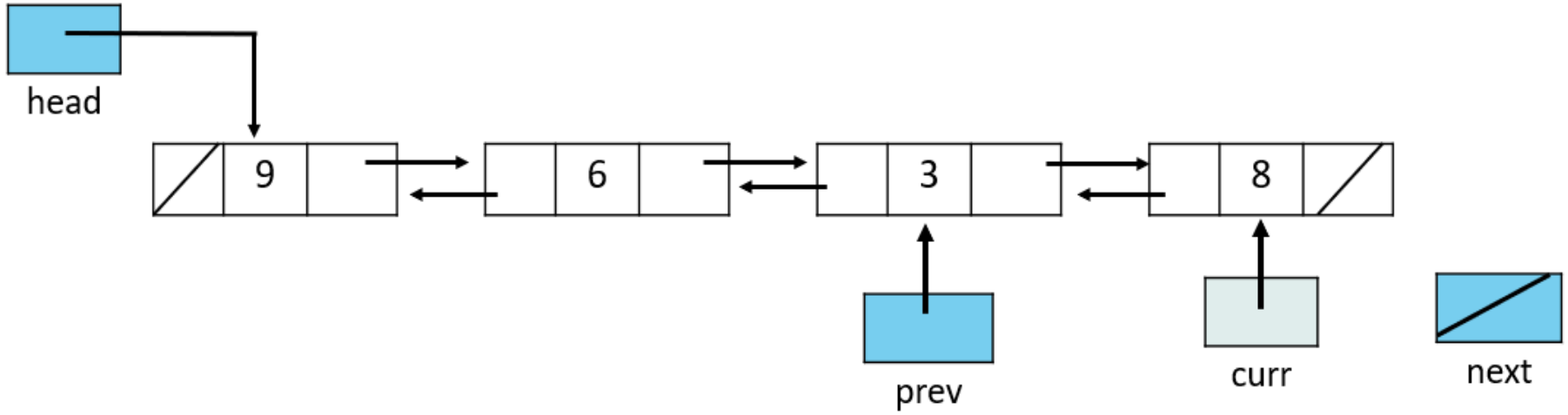
- เมื่อพบว่าโหนดที่ต้องการลบอยู่ตำแหน่งสุดท้ายของลิงค์ลิสต์ ในการอัพเดตโครงสร้างลิงค์ลิสต์จะเป็นการอัพเดตค่า Rlink ของโหนด prev ซึ่งเป็นโหนดที่อยู่ก่อนหน้าโหนดที่ต้องการลบให้มีค่าเป็น NULL เพื่อเปลี่ยนโหนด prev ให้เป็นโหนดสุดท้ายแทน โดยใช้คำสั่ง

```
prev.setRlink (null) ;
```

การลบโหนดที่อยู่ตำแหน่งสุดท้ายของลิงค์ลิสต์

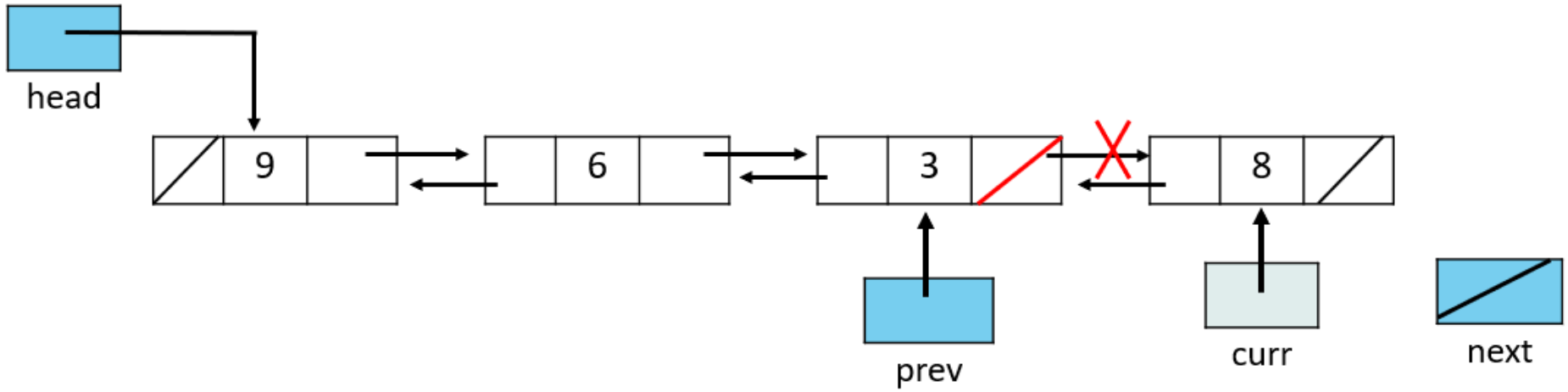
ต้องการลบโหนดที่ทำการเก็บข้อมูล 8

- เมื่อทำการค้นหาแล้วปรากฏว่าอยู่ที่โหนดสุดท้ายของลิงค์ลิสต์ ดังภาพ



การลบโหนดที่อยู่ตำแหน่งสุดท้ายของลิงค์ลิสต์

- ทำการอัพเดตค่า Rlink ของโหนด prev เป็น null เพื่อให้โหนด prev กลายเป็นโหนดสุดท้ายแทนดังภาพ



การแทรกโหนด

- เมื่อค้นหาตำแหน่งโหนดที่ต้องการได้แล้วจะทำการสร้างเส้นเชื่อมใหม่ระหว่างโหนดใหม่กับโครงสร้างเดิม โดยการอัพเดตค่าลิงค์ของโหนดที่เกี่ยวข้อง

- ในการแทรกโหนดจะมีตัวแปรช่วยในการทำงาน 2 ตัว คือ

  1. ตัวแปร curr ทำหน้าที่ค้นหาตำแหน่งโหนดที่ต้องการแทรก
  2. ตัวแปร prev ทำหน้าที่อ้างอิงตำแหน่งโหนดที่อยู่ก่อนหน้าโหนดที่ต้องการแทรก

การแทรกโหนด

- ขั้นตอนการทำงานสามารถแยกตามตำแหน่งของการแทรกโหนดในลิงค์ลิสต์ได้เป็น 3 กรณี คือ
  - การแทรกโหนดเข้าระหว่างข้อมูลที่มีอยู่ในลิงค์ลิสต์
  - การแทรกโหนดเข้าที่ตำแหน่งแรกของลิงค์ลิสต์
  - การเพิ่มโหนดในตำแหน่งสุดท้ายของลิงค์ลิสต์

การแทรกโหนดเข้าระหว่างข้อมูลที่มีอยู่ในลิงค์ลิสต์

- เมื่อค้นหาโหนดที่ต้องการได้แล้วจะทำการเชื่อม newNode เข้ากับลิงค์ลิสต์ และอัพเดตค่าลิงค์ของโหนดในลิงค์ลิสต์ใหม่ โดยมีขั้นตอนการดังนี้

  Step1: ทำการเชื่อม newNode เข้ากับโครงสร้างลิงค์ลิสต์ โดยใช้คำสั่ง

  ```
  newNode.setRlink(curr);

  newNode.setLlink(prev);
  ```

การแทรกโหนดเข้าระหว่างข้อมูลที่มีอยู่ในลิงค์ลิสต์

Step2: ทำการอัพเดตค่า Rlink ของโหนด prev ให้เชื่อมมายัง newNode โดยใช้คำสั่ง

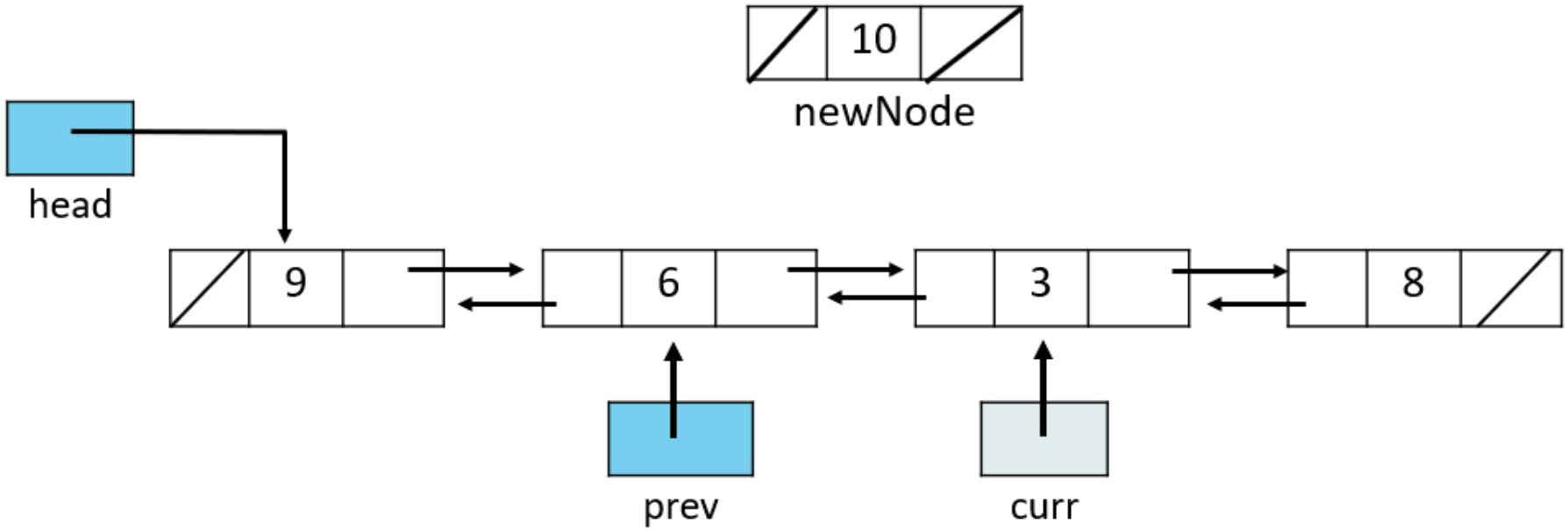
```
prev.setRlink(newNode);
```

Step3: ทำการอัพเดตค่า Llink ของโหนด curr ให้เชื่อมมายัง newNode โดยใช้คำสั่ง

```
curr.setLlink(newNode);
```

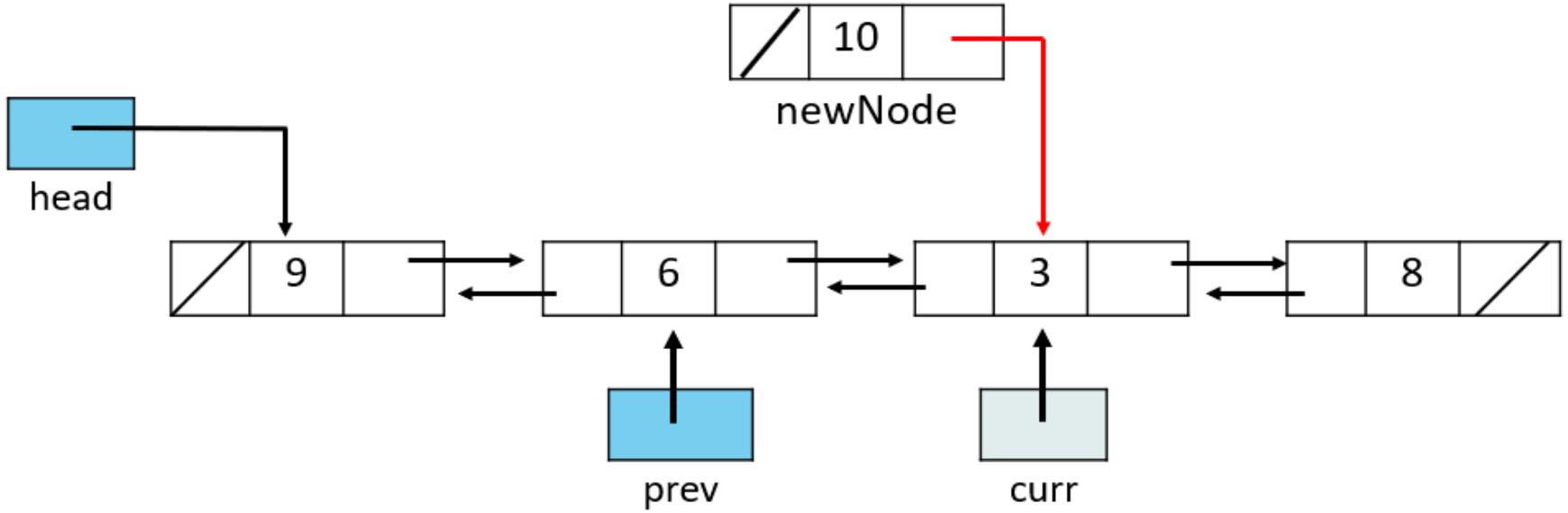
การแทรกโหนดเข้าระหว่างข้อมูลที่มีอยู่ในลิงค์ลิสต์

ต้องการแทรกโหนด newNode ที่มีค่าข้อมูล 10 ไว้ต่อจากโหนดที่เก็บ 6



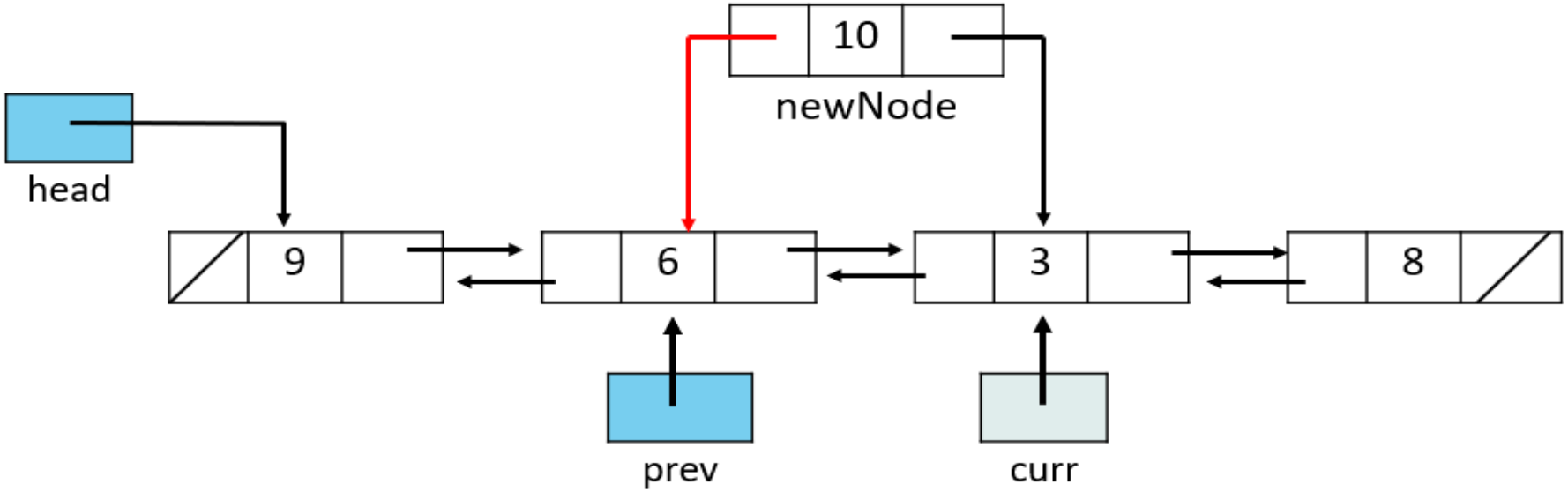
การแทรกโหนดเข้าระหว่างข้อมูลที่มีอยู่ในลิงค์ลิสต์

- Step1: ทำการเชื่อม newNode เข้ากับโครงสร้างลิงค์ลิสต์ โดยใช้คำสั่ง newNode.setRlink(curr);



การแทรกโหนดเข้าระหว่างข้อมูลที่มีอยู่ในลิงค์ลิสต์

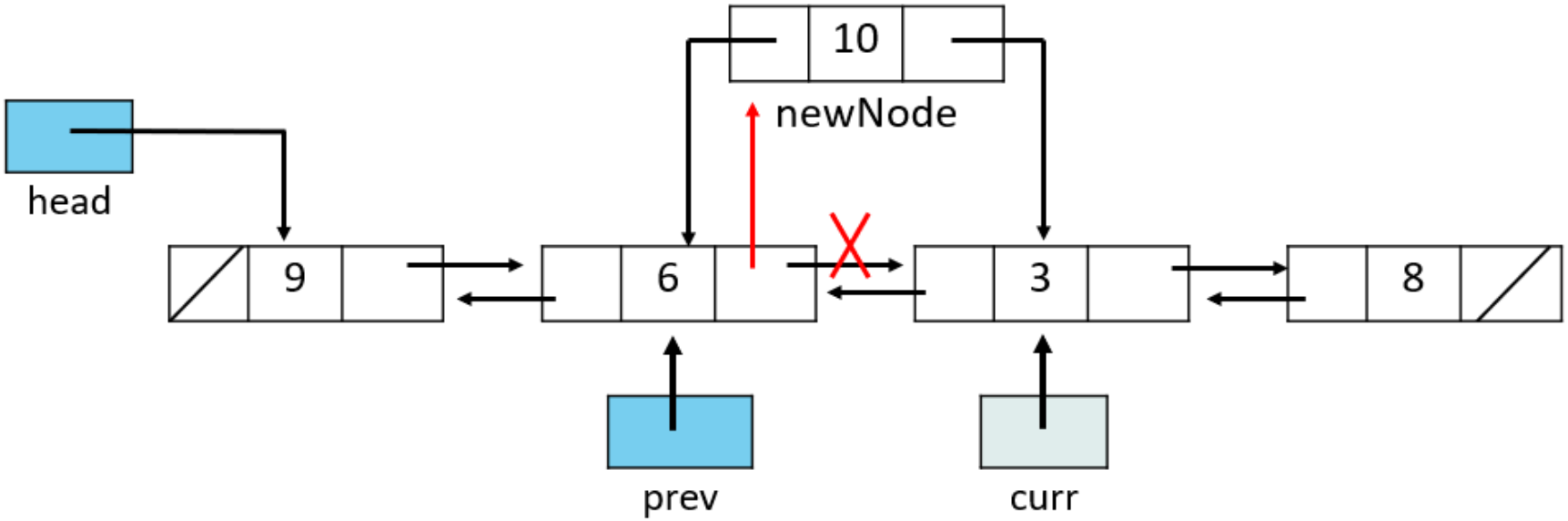
- Step1: ทำการเชื่อม newNode เข้ากับโครงสร้างลิงค์ลิสต์ โดยใช้คำสั่ง newNode.setLlink(prev);



การแทรกโหนดเข้าระหว่างข้อมูลที่มีอยู่ในลิงค์ลิสต์

- Step2: ทำการอัพเดตค่า Rlink ของโหนด prev ให้เชื่อมมายัง newNode โดยใช้คำสั่ง

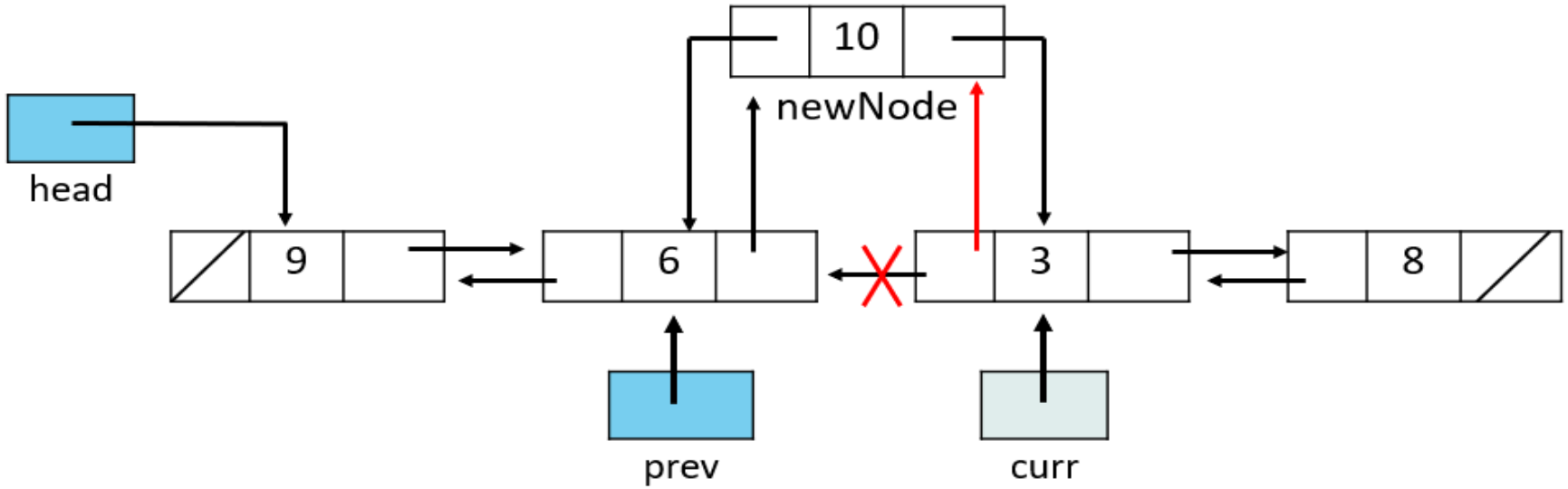
  prev.setRlink(newNode);



การแทรกโหนดเข้าระหว่างข้อมูลที่มีอยู่ในลิงค์ลิสต์

- Step3: ทำการอัพเดตค่า Llink ของโหนด curr ให้เชื่อมมายัง newNode โดยใช้คำสั่ง

curr.setLlink(newNode);



การแทรกโหนดเข้าที่ตำแหน่งแรกของลิงค์ลิสต์

- กรณีที่ต้องการแทรกโหนดเข้าไปที่ตำแหน่งแรกของลิงค์ลิสต์ ในการอัพเดตโครงสร้างลิงค์ลิสต์นอกจากการเชื่อมโหนดใหม่กับโหนดแรกแล้วยังต้องทำการอัพเดตค่า head เพื่อให้มีค่าเป็นโหนด newNode แทน

- โดยมีขั้นตอนการปรับโครงสร้างดังนี้

  Step1: ทำการเชื่อม newNode เข้ากับโหนดแรก โดยใช้คำสั่ง

  ```
  newNode.setRlink(curr);
  ```

การแทรกโหนดเข้าที่ตำแหน่งแรกของลิงค์ลิสต์

Step2: ทำการอัพเดตค่า Llink ของโหนด curr ให้เชื่อมมายัง newNode โดยใช้คำสั่ง

```
curr.setLlink(newNode);
```

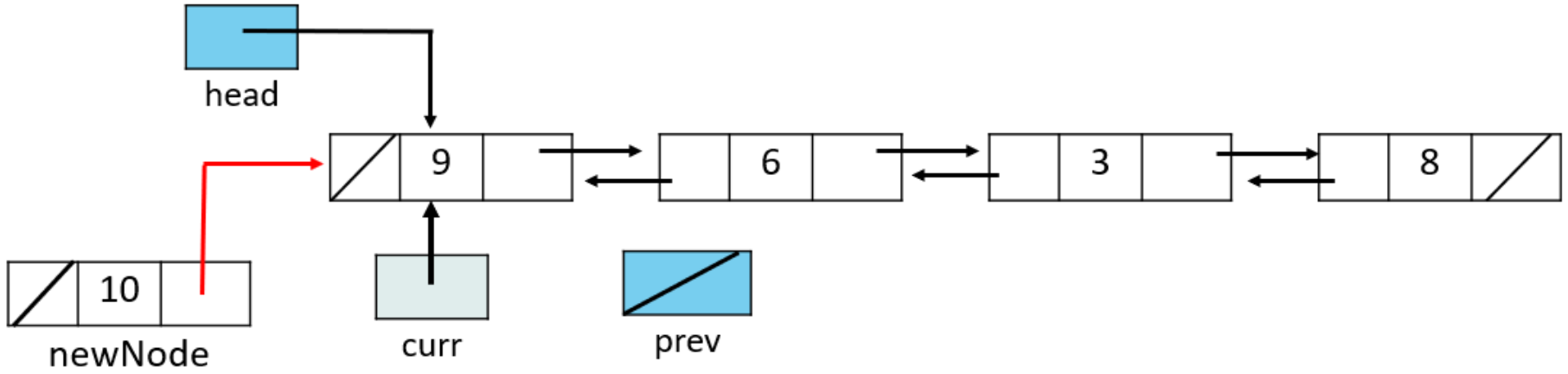
Step3: ทำการอัพเดตค่า head ให้ชี้มายัง newNode โดยใช้คำสั่ง

```
head = newNode;
```

การแทรกโหนดเข้าที่ตำแหน่งแรกของลิงค์ลิสต์

ต้องการแทรกโหนด newNode ที่มีค่าข้อมูล 10 ไว้ตำแหน่งแรก

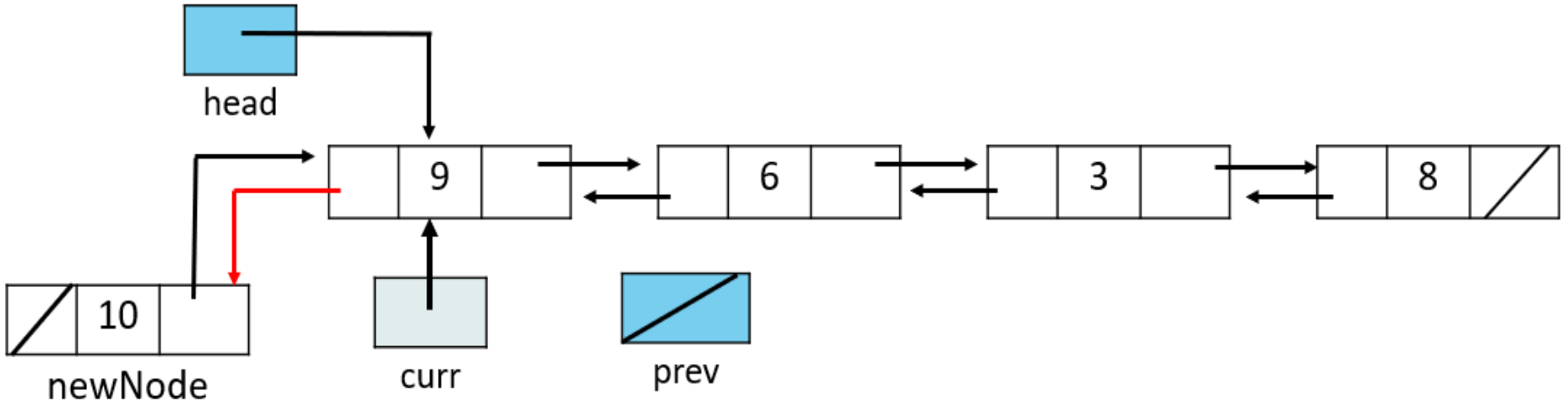
- Step1: ทำการเชื่อม newNode เข้ากับโหนดแรก โดยใช้คำสั่ง newNode.setRlink(curr);



การแทรกโหนดเข้าที่ตำแหน่งแรกของลิงค์ลิสต์

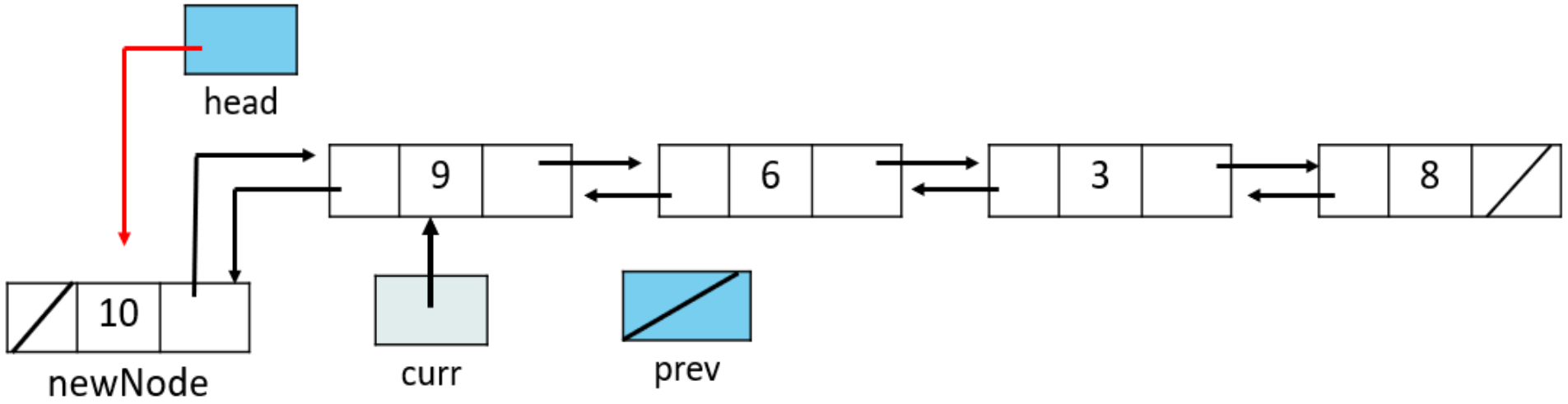
- Step2: ทำการอัพเดตค่า Llink ของโหนด curr ให้เชื่อมมายัง newNode โดยใช้คำสั่ง

curr.setLlink(newNode);



การแทรกโหนดเข้าที่ตำแหน่งแรกของลิงค์ลิสต์

- Step3: ทำการอัพเดตค่า head ให้ชี้มายัง newNode โดยใช้คำสั่ง head = newNode;



การเพิ่มโหนดเข้าที่ตำแหน่งสุดท้ายของลิงค์ลิสต์

- การเพิ่มโหนดในตำแหน่งสุดท้ายของลิงค์ลิสต์จะต้องทำการค้นหาโหนดสุดท้ายก่อน
- โดยจะใช้ตัวแปรช่วย 2 ตัวในการค้นหาตำแหน่งคือตัวแปร curr และ prev เพื่อใช้ในการเดินทางบนลิงค์ลิสต์เพื่อหาตำแหน่งโหนดสุดท้าย
- เมื่อพบตำแหน่งที่ต้องการตัวแปร prev จะทำหน้าที่ระบุตำแหน่งของโหนดสุดท้าย ดังนั้นการจะเพิ่มโหนดใหม่เข้าไปจะเป็นการเชื่อม newNode เข้ากับโหนด prev

การเพิ่มโหนดเข้าที่ตำแหน่งสุดท้ายของลิงค์ลิสต์

- โดยมีขั้นตอนการปรับโครงสร้างดังนี้

Step1: ทำการเชื่อม newNode เข้ากับโหนดสุดท้าย โดยใช้คำสั่ง

```
newNode.setLlink(prev);
```

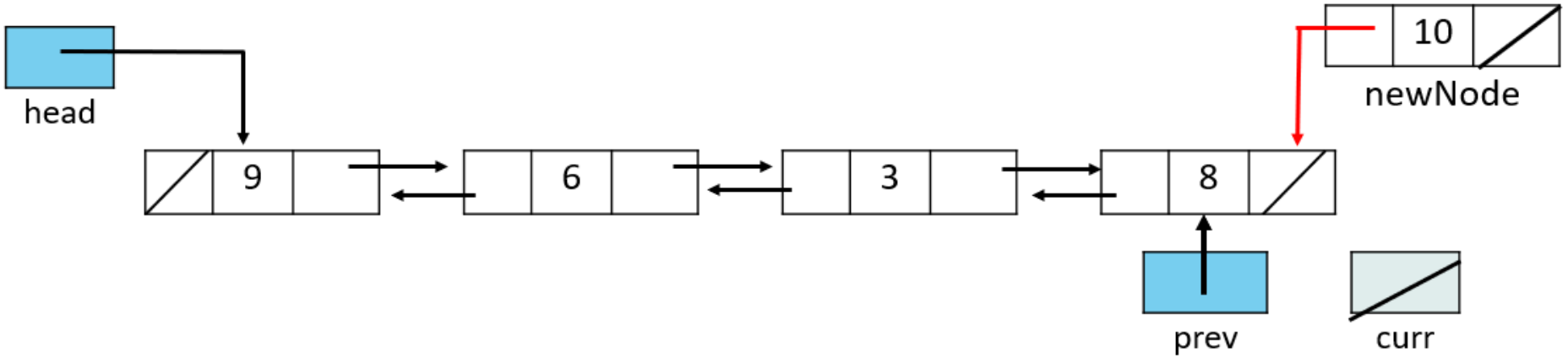
Step2: ทำการอัพเดตค่า Rlink ของโหนด prev ให้เชื่อมไปยังโหนด newNode โดยใช้คำสั่ง

```
prev.setRlink(newNode);
```

การเพิ่มโหนดเข้าที่ตำแหน่งสุดท้ายของลิงค์ลิสต์

ต้องการแทรกโหนด newNode ที่มีค่าข้อมูล 10 ต่อท้ายลิงค์ลิสต์

- Step1: ทำการเชื่อม newNode เข้ากับโหนดสุดท้าย โดยใช้คำสั่ง newNode.setLlink(prev);



การเพิ่มโหนดเข้าที่ตำแหน่งสุดท้ายของลิงค์ลิสต์

- Step2: ทำการอัพเดตค่า Rlink ของโหนด prev ให้เชื่อมไปยังโหนด newNode โดยใช้คำสั่ง prev.setRlink(newNode);