บทที่ 5 Linked List

บทเรียนย่อย

5.1 Linked Lists Operations and Concept

5.2 Linked Lists Component

5.3 Linked Lists Implementation

วัตถุประสงค์

- นิสิตมีความรู้ และความเข้าใจเกี่ยวกับแนวคิด และองค์ประกอบสำคัญต่าง ๆ ในการ จัดการโครงสร้างข้อมูลในรูปแบบของ Linked Lists
- นิสิตสามารถเขียนโปรแกรมเพื่อดำเนินการตามแนวคิดของ Linked Lists
- นิสิตสามารถนำแนวคิดของ Linked List มาประยุกต์ใช้งานในการพัฒนาโปรแกรม

บทที่ 5 Linked List

บทเรียนย่อย

5.1 Linked Lists Operations and Concept

5.2 Linked Lists Component

5.3 Linked Lists Implementation

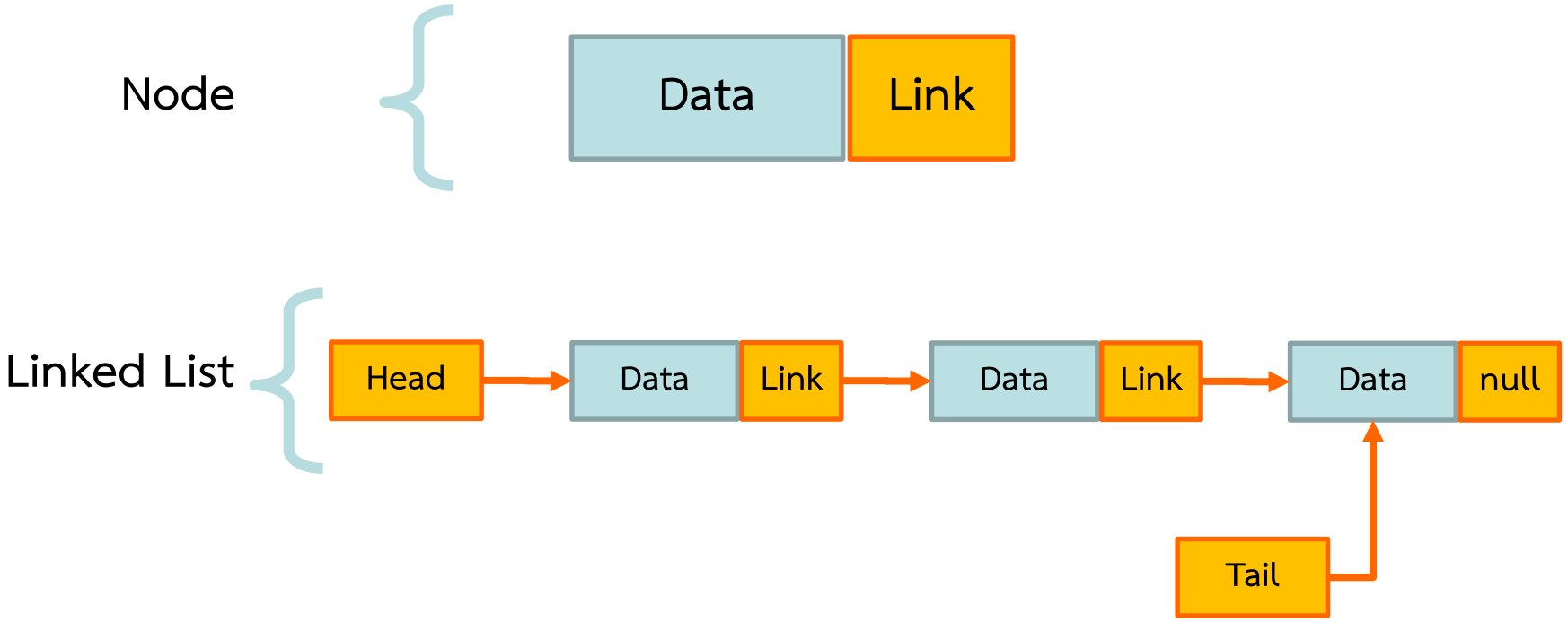
5.1 Linked Lists Operations and Concept

Linked Lists คือ โครงสร้างที่รวบรวมกลุ่มของข้อมูลที่ถูกเรียกว่า node โดยที่ทุก ๆ node จะเชื่อมต่อกัน (ยกเว้น node สุดท้าย) ผ่านที่อยู่ (Address) ของ node ตัวถัดไป ซึ่งใน Linked List จะต้องมีตัวชี้ (Pointer) เพื่อหา node ตัวแรก นิยมเรียกว่า Head หรือ First

Node คือ เป็นส่วนของข้อมูล โดยใน node นั้นจะประกอบไปด้วยอย่างน้อย สอง อย่างคือ

1. Data หรือ Information เป็นข้อมูลที่อยู่ใน node
2. Link ซึ่งทำหน้าที่เป็นตัวเชื่อม node โดยการเก็บข้อมูลที่อยู่ของ node ตัวถัดไป

5.1 Linked Lists Operations and Concept [2]



หมายเหตุ

= ข้อมูล

= ตัวชี้ (Pointer)

บทที่ 5 Linked List

บทเรียนย่อย

5.1 Linked Lists Operations and Concept

5.2 Linked Lists Component

5.3 Linked Lists Implementation

5.2 Linked Lists Component

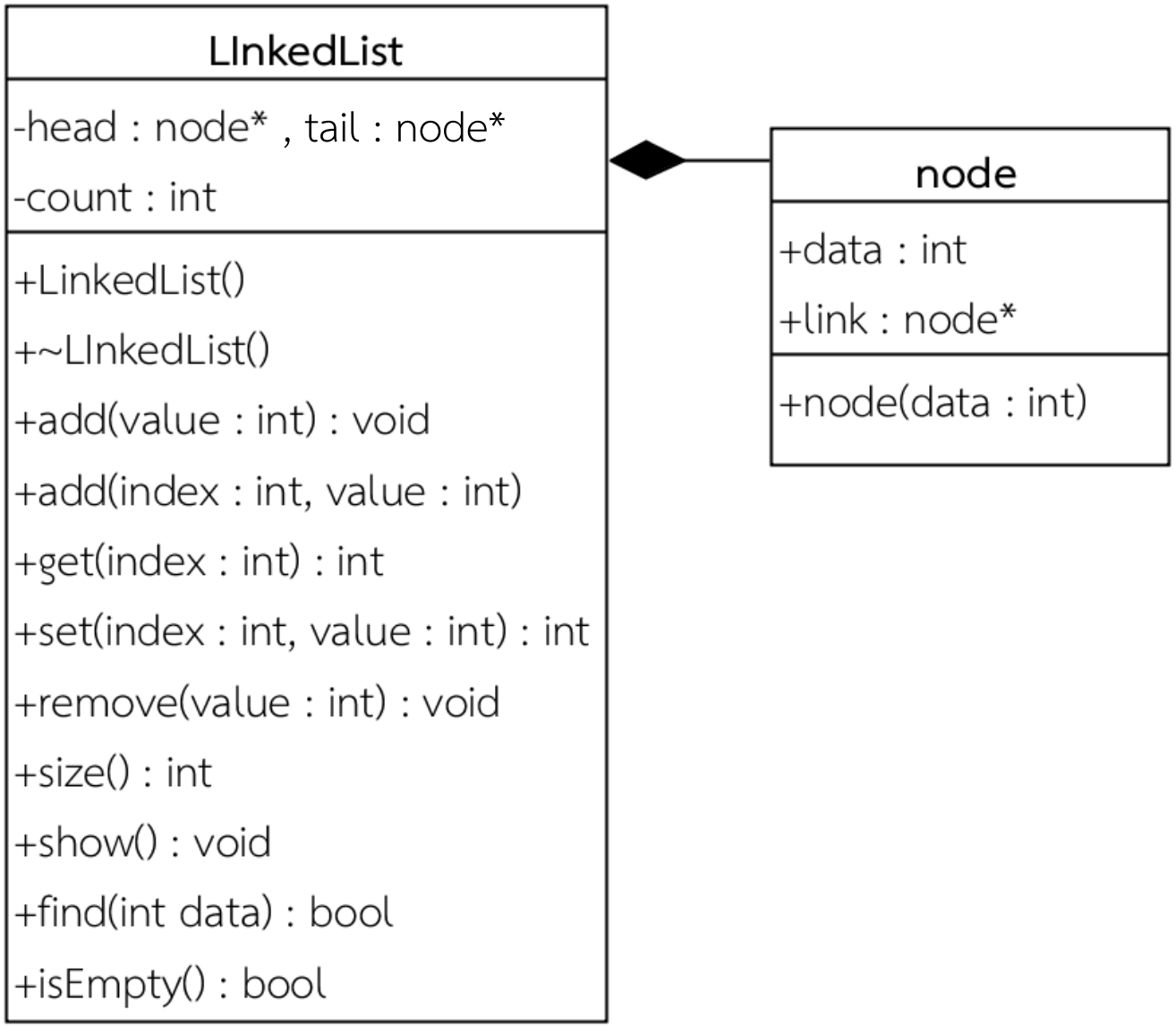
องค์ประกอบของ Linked Lists จะประกอบด้วย คุณสมบัติ (Property) และกระบวนการทำงาน (Method) เนื่องจากมีการสร้างขึ้นให้อยู่ในรูปแบบของคลาส (Class) ซึ่งมีรายละเอียดดังนี้

คุณสมบัติ (Property)	กระบวนการทำงาน (method)
class node	add
head	get
tail	set
count	remove
	show

5.2 Linked Lists Component [2]

คุณสมบัติ (Property)	กระบวนการทำงาน (method)
	size
	find
	isEmpty

Linked Lists Class



รายละเอียดคุณสมบัติของ Linked Lists Class

คุณสมบัติ (Property)	รายละเอียด
node	เป็น Class ที่เก็บข้อมูล และที่อยู่ของตัวถัดไป
count	ตัวแปรสำหรับการใช้นับจำนวนข้อมูลที่เก็บไว้ทั้งหมด
head	ตัวแปรสำหรับเก็บที่อยู่ของข้อมูลตัวแรก
tail	ตัวแปรสำหรับเก็บที่อยู่ของข้อมูลตัวสุดท้าย

รายละเอียดกระบวนการทำงานของ Linked Lists Class

กระบวนการทำงาน (method)	รายละเอียดการทำงาน
LinkedList()	Constructor สำหรับกำหนด ค่า head = nullprt; count = 0;
~LinkedList( )	Deconstructor สำหรับลบ Link ที่กำหนดขึ้นออกจากหน่วยความจำ
add(int value)	เพิ่มข้อมูลเข้า Linked List ในพื้นที่ว่างแบบต่อท้ายตามลำดับ
add(int index , int value)	เพิ่มข้อมูลเข้า Linked List แบบแทรกด้านหน้าในตำแหน่งที่ระบุ

รายละเอียดกระบวนการทำงานของ Linked Lists Class [2]

กระบวนการทำงาน (method)	รายละเอียดการทำงาน
int get(int index)	คืนค่าข้อมูลในตำแหน่งที่กำหนด จาก Linked List
set(int index , int value)	ปรับเปลี่ยนหรือแก้ไขค่า ในตำแหน่งที่กำหนด
remove(int value)	ลบข้อมูลออกจาก Linked List ตามค่าที่กำหนด

รายละเอียดกระบวนการทำงานของ Linked Lists Class [3]

กระบวนการทำงาน (method)	รายละเอียดการทำงาน
int size( )	คืนค่าจำนวนข้อมูลที่มีอยู่ใน Linked List
show( )	แสดงผลข้อมูลที่มีใน Linked List ทั้งหมด ผ่านทางหน้าจอ
bool find(int data)	ค้นหาค่าตามข้อมูลที่ระบุ
bool Empty()	ตรวจสอบค่าใน Linked List ว่ามีข้อมูลอยู่หรือไม่

บทที่ 5 Linked List

บทเรียนย่อย

5.1 Linked Lists Operations and Concept

5.2 Linked Lists Component

5.3 Linked Lists Implementation

5.3 Linked Lists Implementation

การสร้างคลาส Linked List ภาษา C++

```
class LinkedList {
    private :
     class node {
                ...
        };
        node *head;
        node *tail;
    public :
        LinkedList();
        ~LinkedList( );
                ...
};
```

5.3 Linked Lists Implementation [2]

การสร้างคลาส Linked List ภาษา C++

```
class LinkedList {
        ...
    add( int  value );
    add( int index , int  value );
    int get( int index );
    set( int index , int  value );
    remove( int value );
        ...
};
```

4.3 Linked Lists Implementation [3]

การสร้างคลาส Linked List ภาษา C++

```
class LinkedList {
        ...
    int size( );
    show( );
};
```

การดำเนินการใน Constructor และ Deconstructor

```
LinkedList :: LinkedList(){
    this->head = NULL;
    this->tail = NULL;
    this->count = 0;
}
```

การดำเนินการใน Constructor และ Deconstructor

```
LinkedList :: ~LinkedList( void ){
    for(int i = 1; i < this->count; i++){
        node * tmp = this->head;
        this->head = this->head->link;
        delete tmp;
        tmp = NULL;
    }
    this->head = NULL;
    this->tail = NULL;
}
```

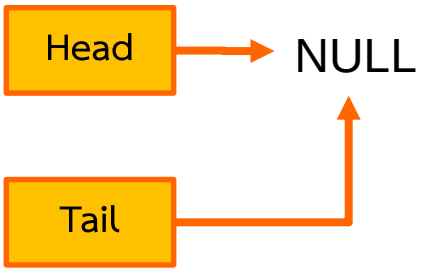
การเรียกใช้งานคลาส Linked List

```
int main(void){
    LinkedList *obj_linkedlist = new LinkedList();
    obj_linkedlist->add(5);
    obj_linkedlist->add(8);
    obj_linkedlist->show();

    LinkedList obj_linkedlist;
    obj_linkedlist.add(5);
    obj_linkedlist.add(8);
    obj_linkedlist.show();
}
```

แนวคิดการเพิ่มข้อมูล

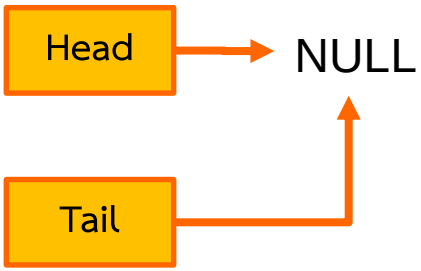
LinkedList obj_linkedlist;



count = 0

แนวคิดการเพิ่มข้อมูล [2]

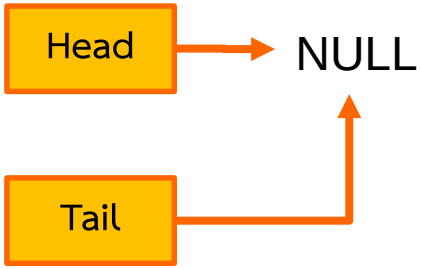
obj_linkedlist.add(5);



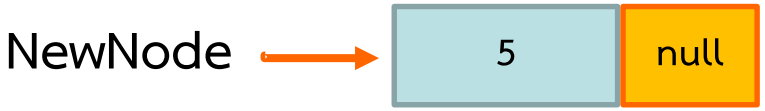
count = 0

แนวคิดการเพิ่มข้อมูล [3]

obj_linkedlist.add(5);



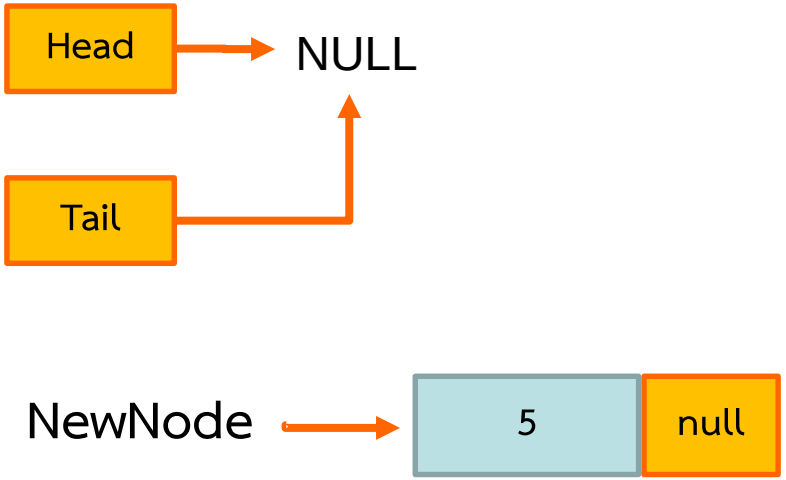
สร้าง new object จาก class node โดยกำหนดค่า data มีค่าเท่ากับ 5 และ link มีค่าเป็น NULL;



count = 0

แนวคิดการเพิ่มข้อมูล [4]

obj_linkedlist.add(5);



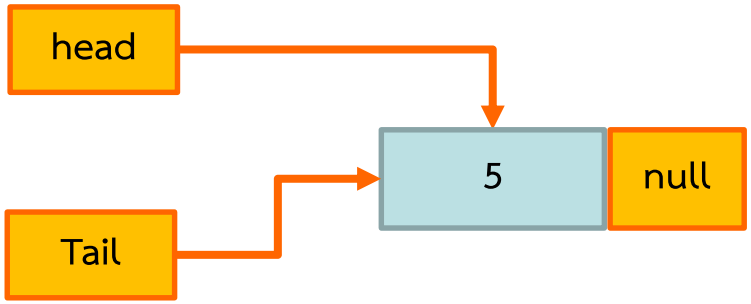
ตรวจสอบค่า head มีค่าเป็น null หรือไม่

ถ้าเป็น ทำการกำหนดค่า head และ tail ให้ เท่ากับ NewNode

ถ้าไม่เป็น ทำการนำ link ของ node ตัวสุดท้าย มากำหนดให้เท่ากับค่าของ NewNode และทำการเพิ่มค่า count

แนวคิดการเพิ่มข้อมูล [5]

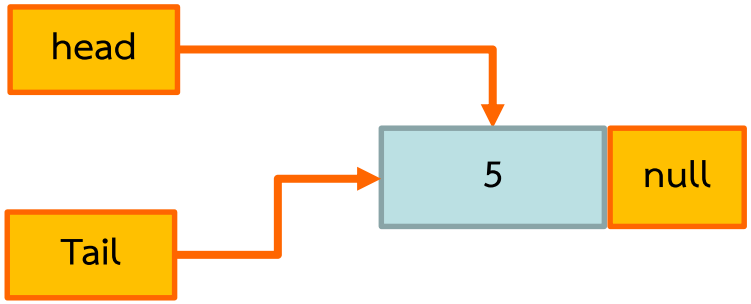
obj_linkedlist.add(5);



count = 1

แนวคิดการเพิ่มข้อมูล [6]

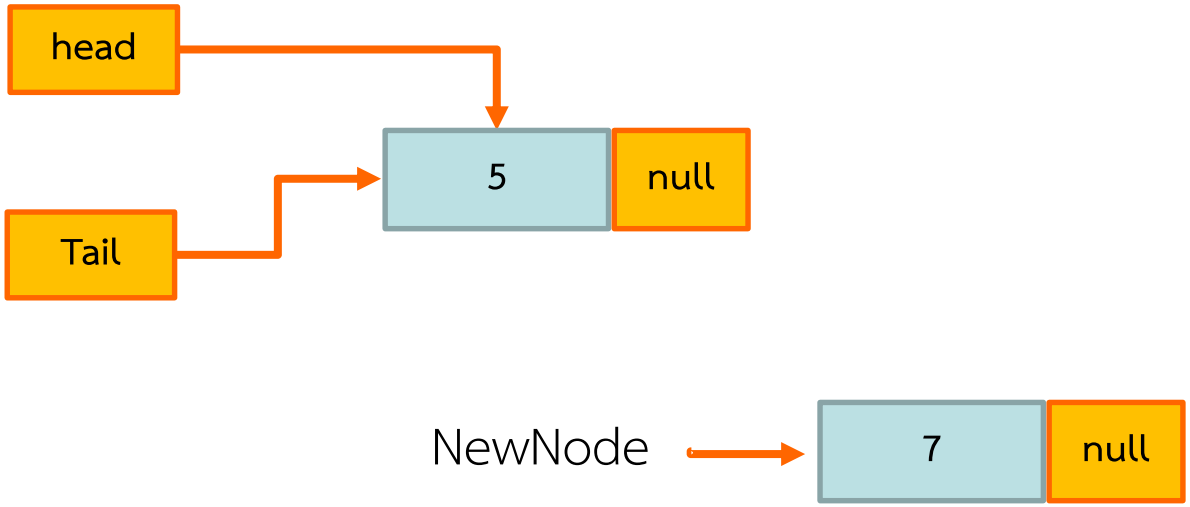
obj_linkedlist.add(7);



count = 1

แนวคิดการเพิ่มข้อมูล [7]

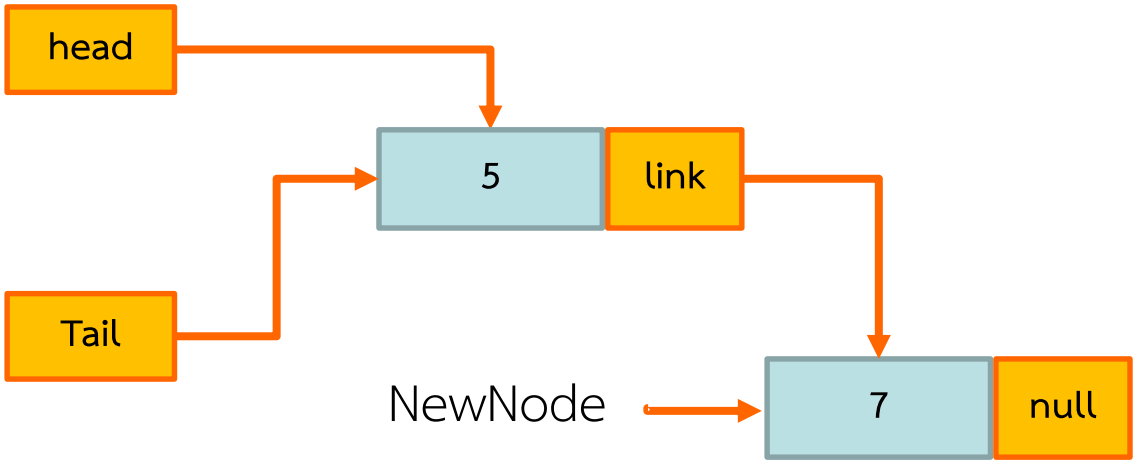
obj_linkedlist.add(7);



count = 1

แนวคิดการเพิ่มข้อมูล [8]

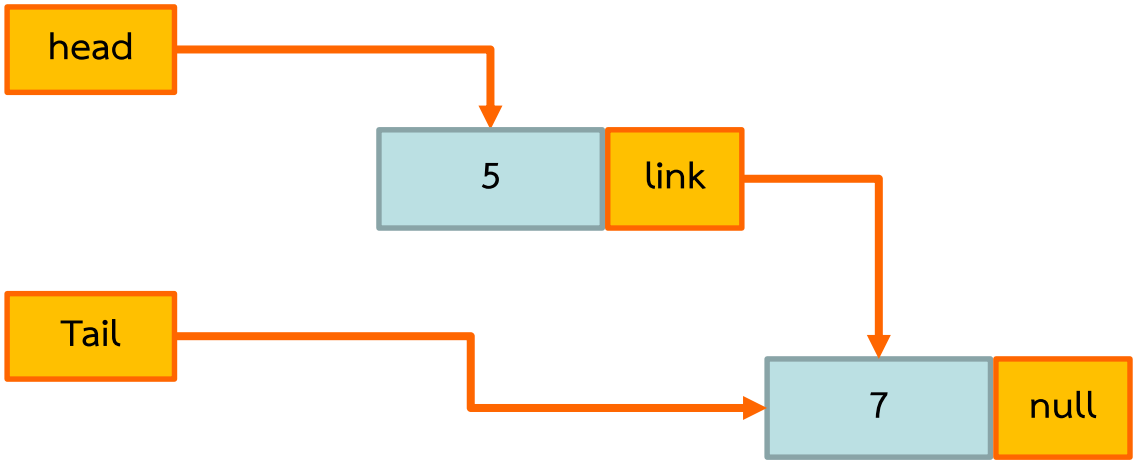
obj_linkedlist.add(7);



count = 2

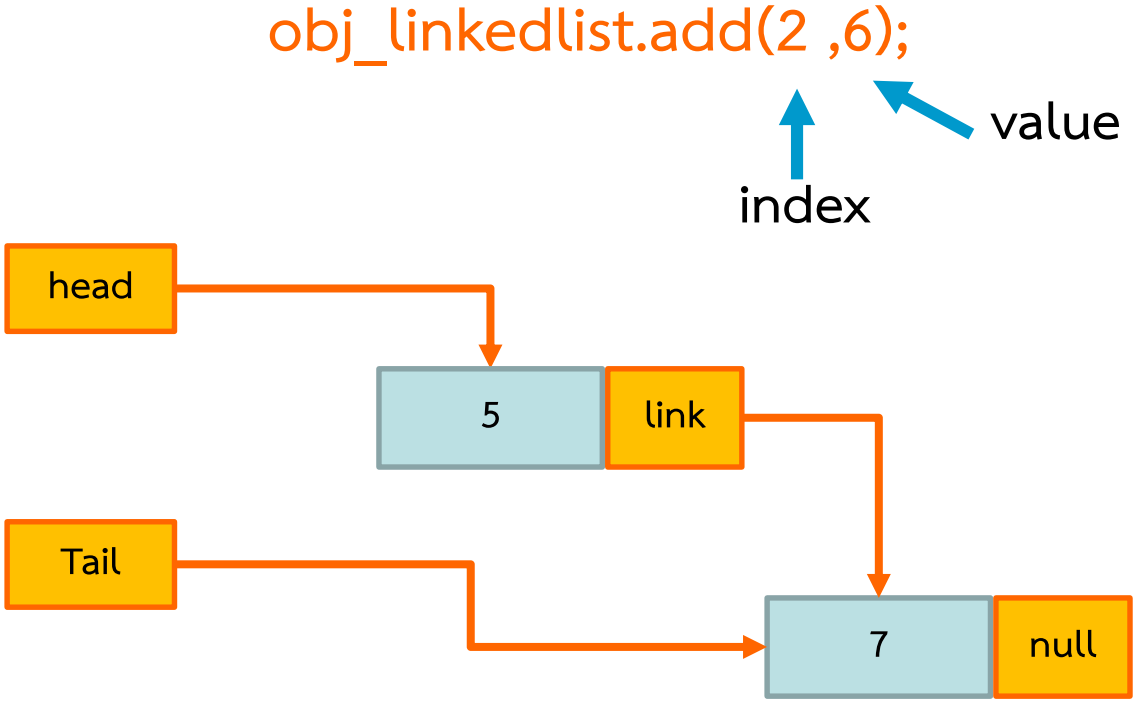
แนวคิดการเพิ่มข้อมูล [9]

obj_linkedlist.add(7);



count = 2

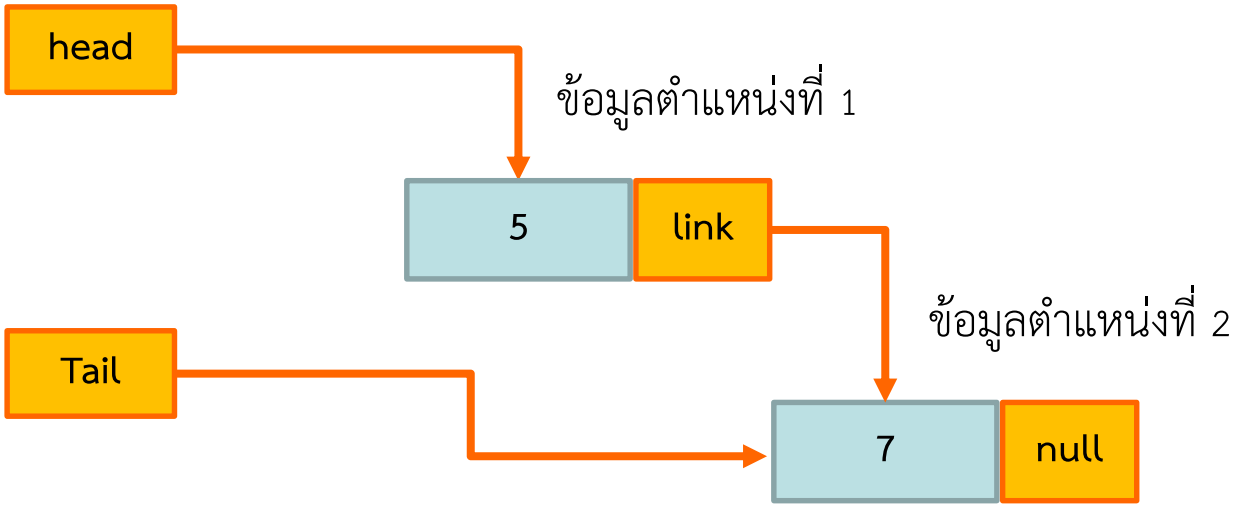
แนวคิดการเพิ่มข้อมูล กรณีระบุตำแหน่ง



count = 2

แนวคิดการเพิ่มข้อมูล กรณีระบุตำแหน่ง [2]

obj_linkedlist.add(2 ,6);

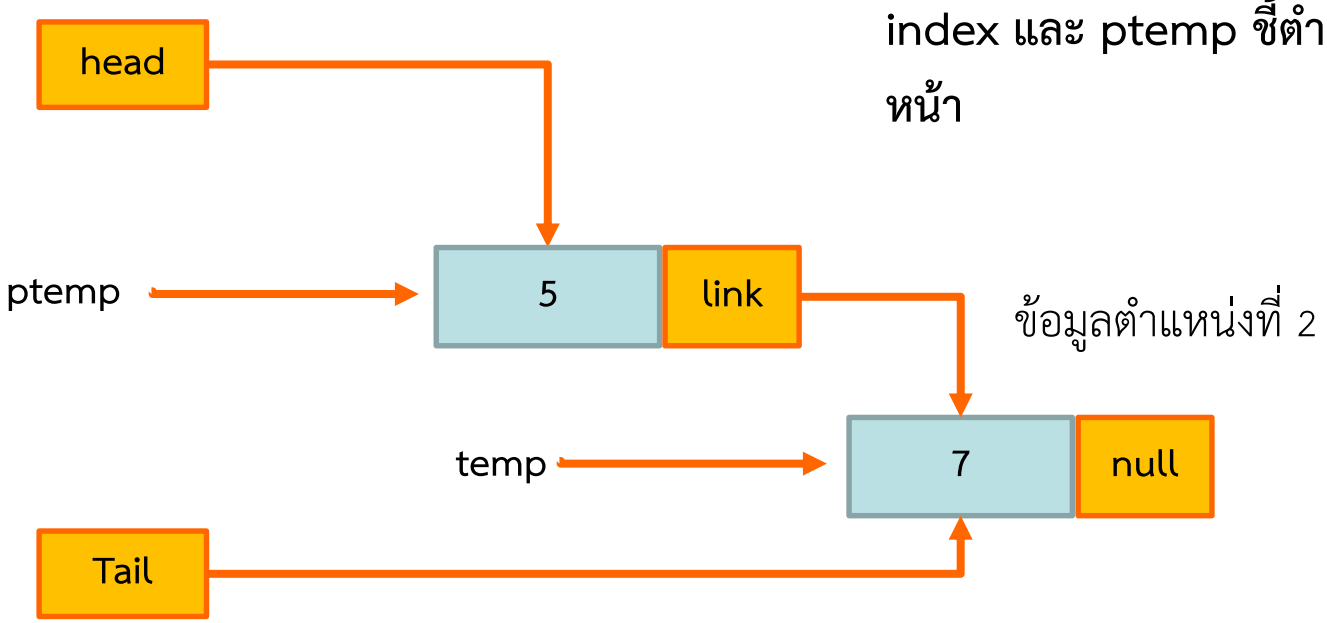


count = 2

แนวคิดการเพิ่มข้อมูล กรณีระบุตำแหน่ง [3]

obj_linkedlist.add(2 ,6);

สร้างตัวชี้ temp มาชี้ที่ค่าของตำแหน่ง index และ ptemp ชี้ตำแหน่งก่อนหน้า

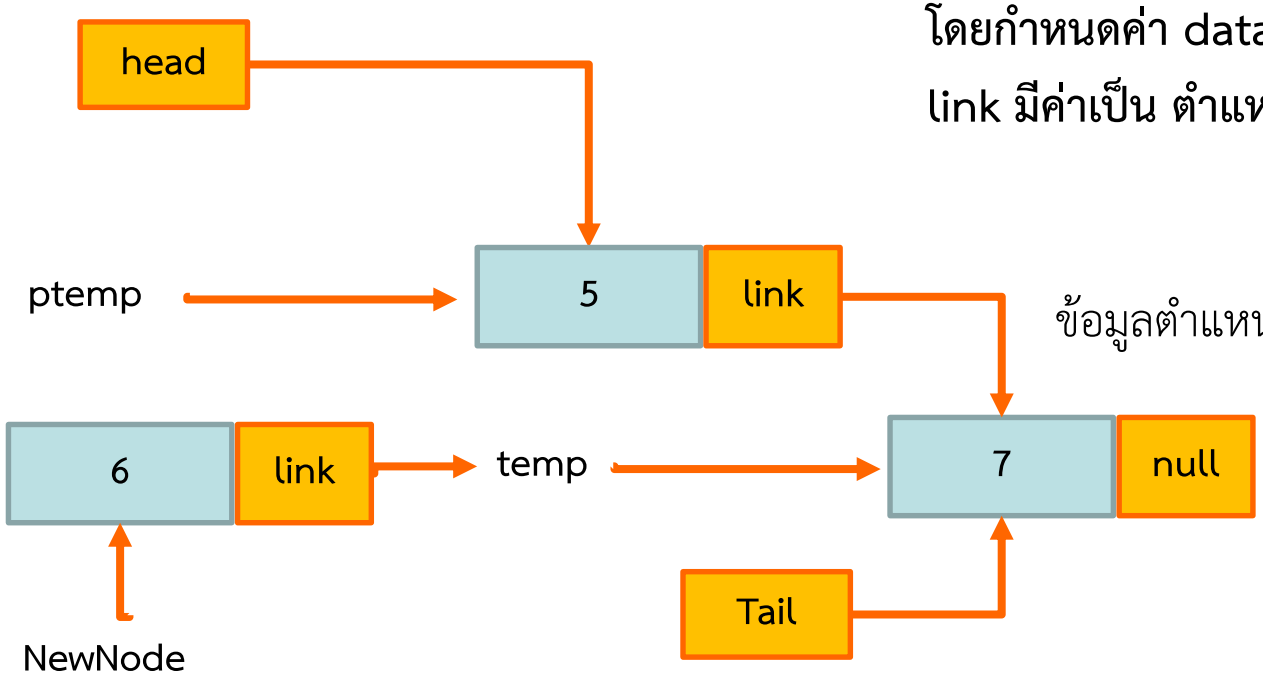


count = 2

แนวคิดการเพิ่มข้อมูล กรณีระบุตำแหน่ง [4]

obj_linkedlist.add(2 ,6);

สร้าง new object จาก class node โดยกำหนดค่า data มีค่าเท่ากับ 6 และ link มีค่าเป็น ตำแหน่งของ temp

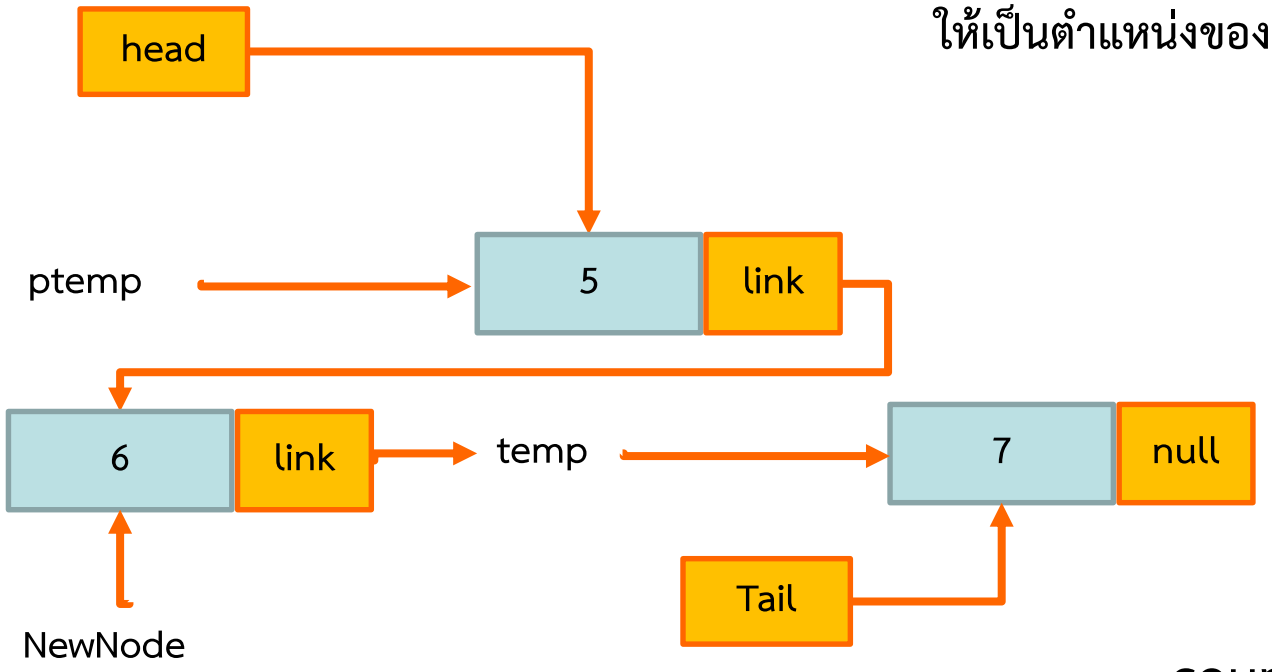


count = 2

แนวคิดการเพิ่มข้อมูล กรณีระบุตำแหน่ง [5]

obj_linkedlist.add(2 ,6);

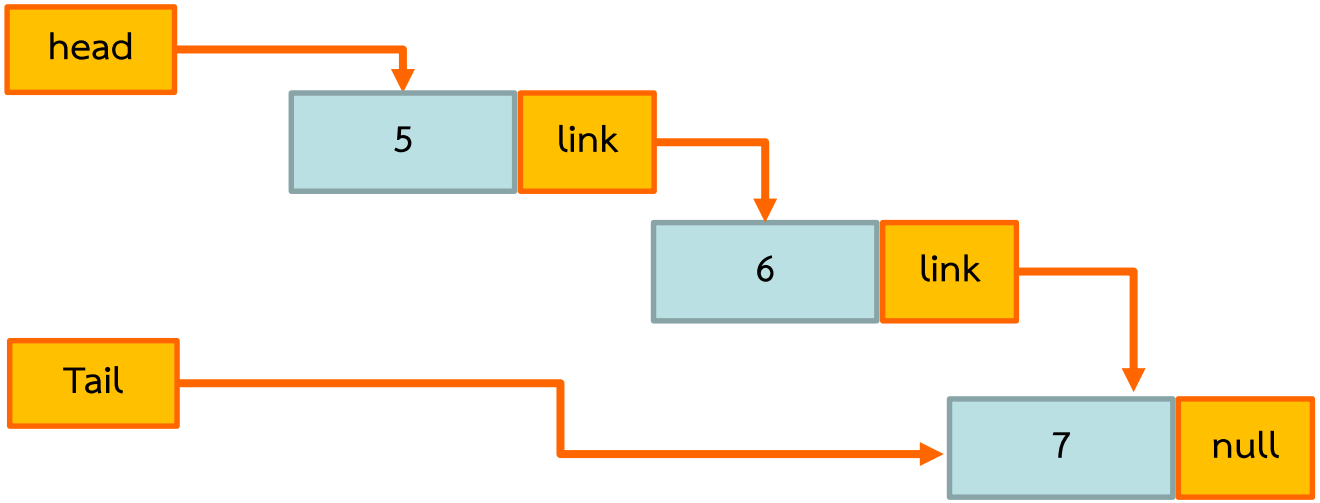
เปลี่ยน link ของตัวที่ ptemp ชี้
ให้เป็นตำแหน่งของ NewNode



count = 2

แนวคิดการเพิ่มข้อมูล กรณีระบุตำแหน่ง [6]

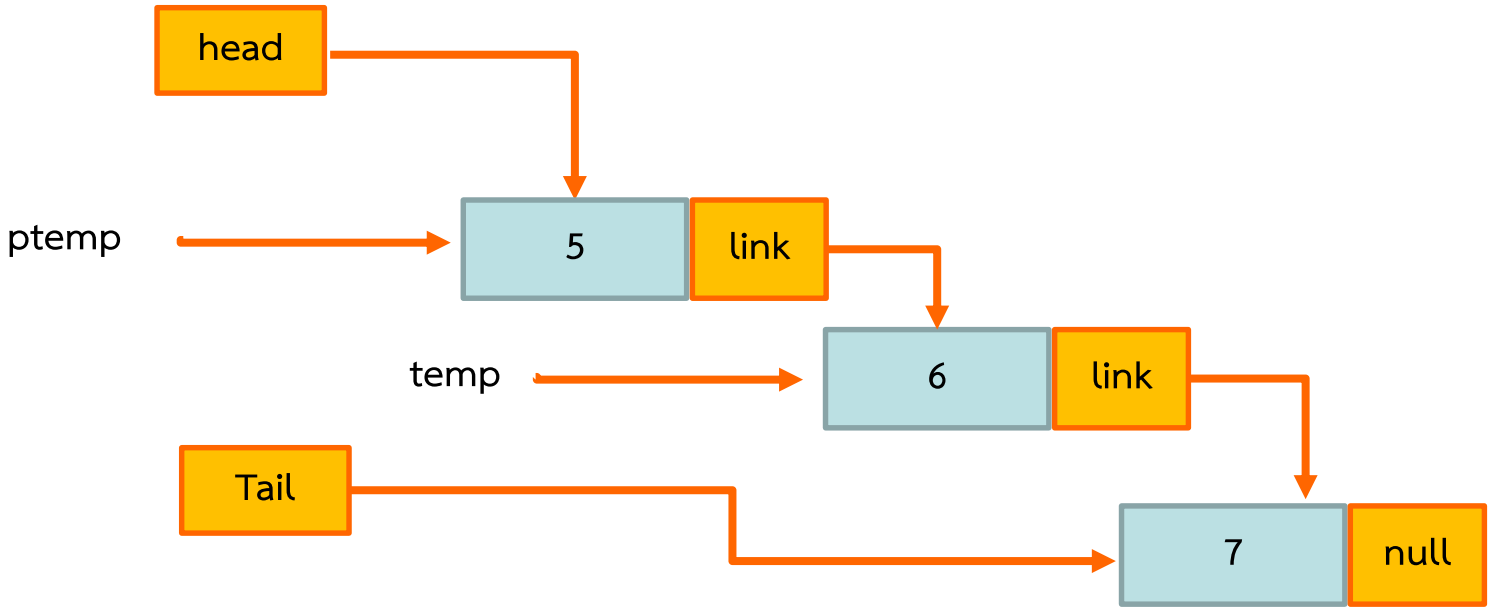
obj_linkedlist.add(2 ,6);



count = 3

แนวคิดการลบข้อมูล

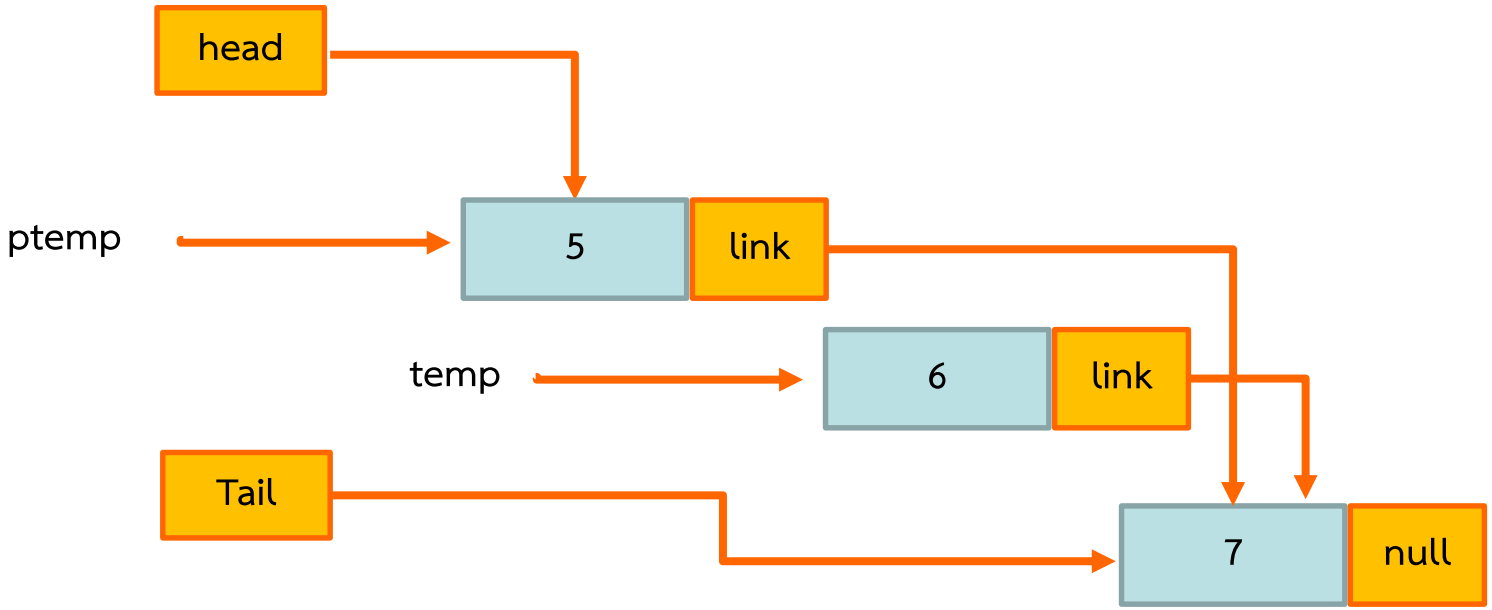
obj_linkedlist.remove(6);



count = 3

แนวคิดการลบข้อมูล [2]

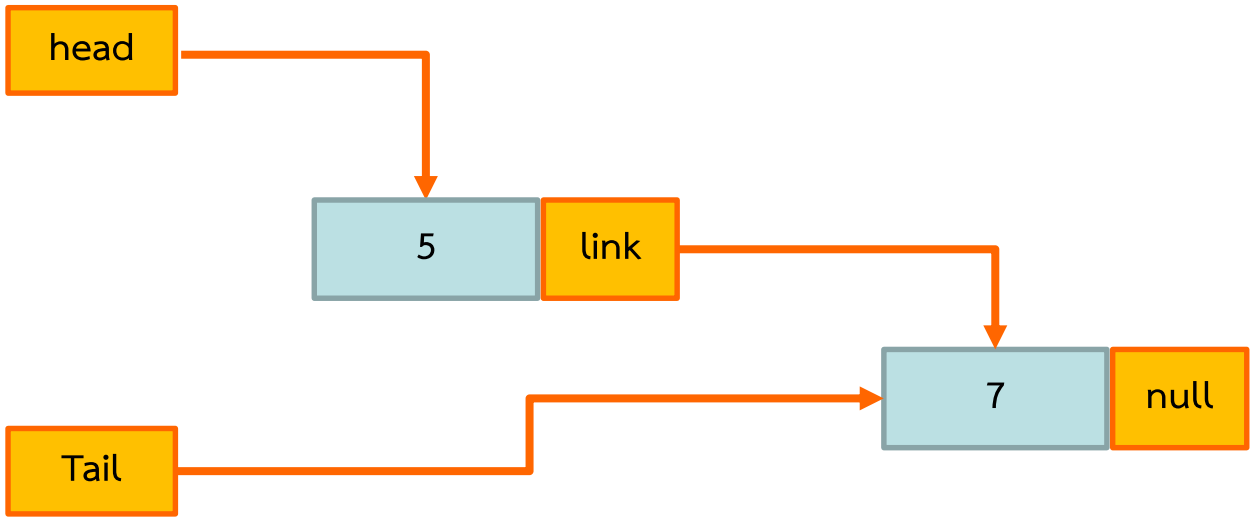
obj_linkedlist.remove(6);



count = 3

แนวคิดการลบข้อมูล [3]

obj_linkedlist.remove(6);



count = 2

แบบฝึกหัด

1. สร้าง คลาส LinkedList เพื่อจัดเก็บข้อมูลเลขจำนวนเต็ม โดยมีความสามารถในการจัดการข้อมูล ดังนี้
    - สามารถเพิ่มข้อมูลแบบปกติและแบบระบุตำแหน่ง
    - สามารถเปลี่ยนแปลง หรือ แก้ไขข้อมูล
    - สามารถลบข้อมูล
    - สามารถค้นหาข้อมูล
    - สามารถตรวจสอบจำนวนข้อมูลที่มีทั้งหมด
    - สามารถตรวจสอบพื้นที่ว่างในการเก็บข้อมูล
2. นำคลาสที่สร้างขึ้นไปทดสอบการใช้งานในฟังก์ชัน main โดยทำการสร้างเป็นลักษณะเมนูสำหรับทดลองทุกความสามารถที่มีในคลาส LinkedList ของตนเอง

Class ของแบบฝึกหัด