



List : Linked list

list - รายการ

สิ่งที่เก็บรายการ $\left\{ \begin{array}{l} \text{Array} \\ \text{Linked list} \end{array} \right.$

เมื่อเรียนจบแล้ว สิ่งที่คุณจะสามารถทำได้คือ

- 1) สามารถแยกแยะระหว่าง list และ linked list ได้
- 2) เข้าใจการทำงานของ linked list
- 3) สามารถเขียนโปรแกรมทำการเก็บข้อมูลลงใน linked list ได้
- 4) สามารถเขียนโปรแกรมทำการลบข้อมูลลงใน linked list ได้
- 5) สามารถเขียนโปรแกรมทำการค้นหาข้อมูลลงใน linked list ได้
- 6) สามารถเขียนโปรแกรมทำการนับจำนวนข้อมูลลงใน linked list ได้
- 7) สามารถนำ linked list ไปใช้ในการปัญหาทางการโปรแกรมได้
- 8) ทราบ BigO ทุก Operation



2.3 The List

รายการ

ข้อมูลที่เรียงกัน

A General list of the form $A_1, A_2, A_3, \dots, A_N$

list ใดๆ

ยกเว้น

list ว่าง

For any list except the empty list, we say that A_{i+1} follows (or succeeds) A_i ($i < N$) and that A_{i-1} precedes A_i ($i > 1$). The first element of the list is A_1 , and the last element A_i in a list is A_N

หน้าหน้า $\leftarrow i \leftarrow$ ตามหลัง

A_{i+1}, A_i, A_{i+1}

($i < N$)

เริ่มที่ 1 จนถึง N



2.3.1 List Operation

☐ insert

☐ remove

☐ printList

☐ makeEmpty

☐ find

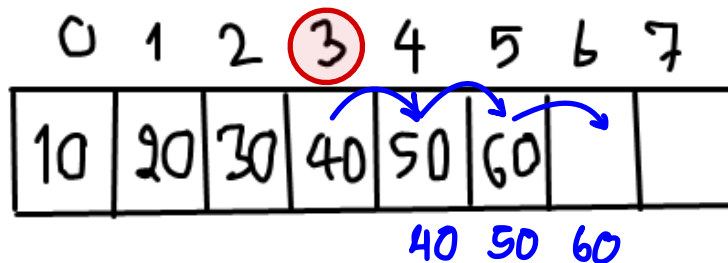
☐ count



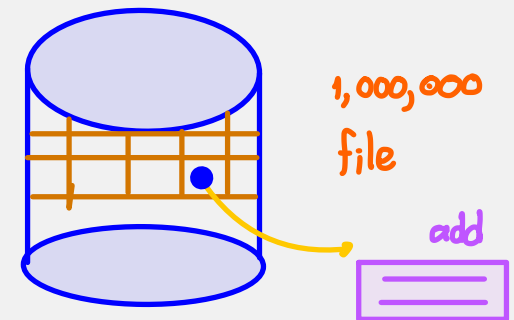
2.3.2 Array List

1) Insert

1. สร้าง Array โดยคาดคะเนว่า list จะมีข้อมูลมากที่สุดกี่ค่า
2. กำหนดตัวแปร size คือจำนวนข้อมูล pos(position) คือตำแหน่งที่ต้องการแทรก
3. การ insert จะทำโดยการหาตำแหน่งที่ต้องการแทรกข้อมูลก่อน เช่น ต้องการแทรก 35 ตำแหน่งที่จะแทรกคือ 3



ขยับ
หลังสุด



4. จะแทรกได้จะต้องขยับเลื่อนข้อมูลด้านหลังทุกตัวไป 1 ค่า ในช่วง pos-size
ต้องมี array ต้องพอ size
5. ทำการแทรก



03603212 : Module2-List, stack, Queue 5

```
...  
int main()  
{  int a[8]={10,20,30,40,50,60};  
    int newNumber,i,j,size,index;  
    size=5;  
    newNumber=35;  
    for(i=0;i<size;i++)  
    {  if(newNumber<a[i])  
        {      index=i;  
                break;  
        }  
    }  
    cout << index;  
}
```

$$\frac{n}{2} = \frac{1}{2} \times n = O(n)$$

$$\frac{n(n+1)}{2}$$

1. สร้าง Array โดยคาดคะเนว่า list จะมีข้อมูลมากที่สุดกี่ค่า
2. กำหนดตัวแปร size คือจำนวนข้อมูล pos(position) คือตำแหน่งที่ต้องการแทรก
3. การ insert จะทำโดยการหาตำแหน่งที่ต้องการแทรกข้อมูลก่อน เช่น

0	1	2	3	4	5	6	7
10	20	30	40	50	60		

index → size - 1

ต้องการแทรก 35 ตำแหน่งที่
จะแทรกคือ 3 ซึ่งเก็บอยู่ใน
ตัวแปร index



03603212 : Module2-List, stack, Queue 6

assign จากฟังก์ชันหน้า

$a[i+1] = a[i];$

```
for(i=size-1;i>=index;i--)
```

```
    a[i+1]=a[i]; i=5 ; a[5]
```

```
    size++;      i=4 ; a[4]
```

```
                i=3 ; a[3]
```

```
a[index]=newNumber;
```

0	1	2	3	4	5	6	7
10	20	30	40	50	60		

- ต้องเลื่อน ตั้งแต่ตัวที่ 3 - 5 หรือ
ตั้งแต่ index ถึง size ไป
ตำแหน่งที่ 4-6
ทุกตัวไป 1 ค่า ในช่วง pos-size
- ทำการแทรก



2) Delete

0	1	2	3	4	5	6	7
10	20	30	35	40	50	60	

1. หาดำแหน่งที่ต้องการลบใส่ตัวแปร index
2. ขยับข้อมูลตั้งแต่ลำดับ 4-6 คือ $\text{index}+1$ ถึง $\text{size} - 1$ เลื่อนมาด้านหน้า
3. ลดขนาด size



การบ้าน 2

1. จงเขียนโปรแกรมโดยใช้ Array ขนาด 10 ช่อง สร้างเป็น list โดยมีการทำงานตาม menu ดังนี้ ให้แยกการทำงานข้อ 1-3 ออกเป็นฟังก์ชัน

หมายเหตุ ให้ทดลอง insert 8 5 1 20 6 14 และลบ 8 20 1

=====Menu=====

+	1) Insert	+
+	2) Delete	+
+	3) Print	+
+	4) Exit	+

=====

Please choose >

ถ้าเลือกข้อ 1

Enter : 8

Output = 8 จากนั้นกลับไปเมนู

ถ้าเลือกข้อ 1

Enter : 5

Output = 5 8 จากนั้นกลับไปเมนู

ถ้าเลือกข้อ 2

Delete : 8

Output = 5 จากนั้นกลับไปเมนู



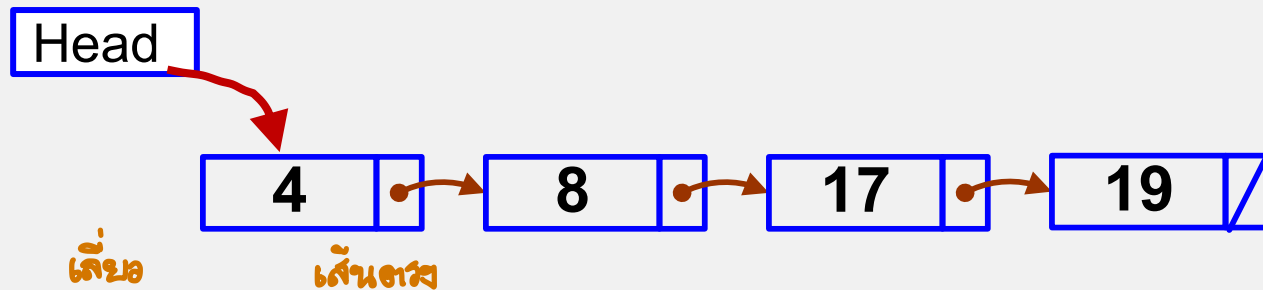
03603212 : Module1 – Introduction 9

ถ้าเลือกข้อ 3 สมมติว่ามีข้อมูล 10 20 30 40 50 60
จะแสดงข้อมูลดังนี้

Print : 10 20 30 40 50 60
Print first half : 10 20 30
Print second half : 40 50 60

หรือ

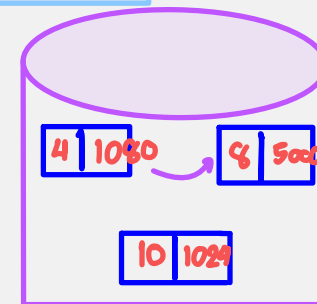
Print : 1 22 23 47 50
Print first half : 1 22
Print second half : 23 47 50



Avoid the linear cost of insertion and deletion of array.

The linked list consists of a series of nodes, which are not necessary adjacent in memory. Each node contains the element and a link to a node containing its successor. We call this the next link. The last cell's next link points to NULL.

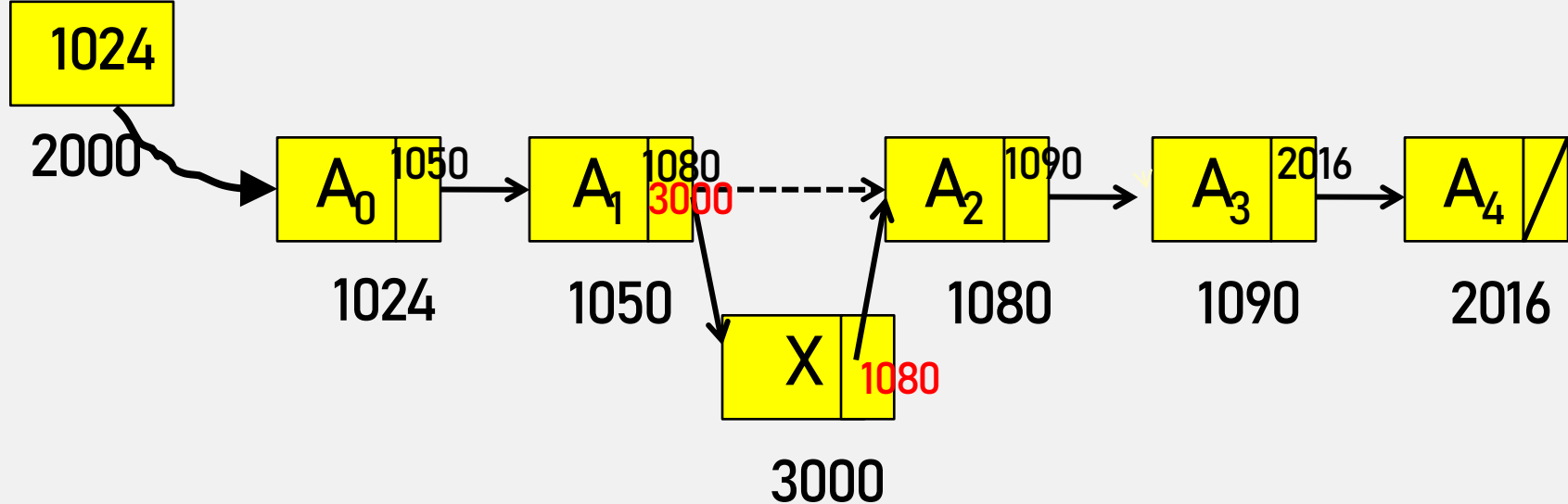
memory array ใช้อุปกรณ์



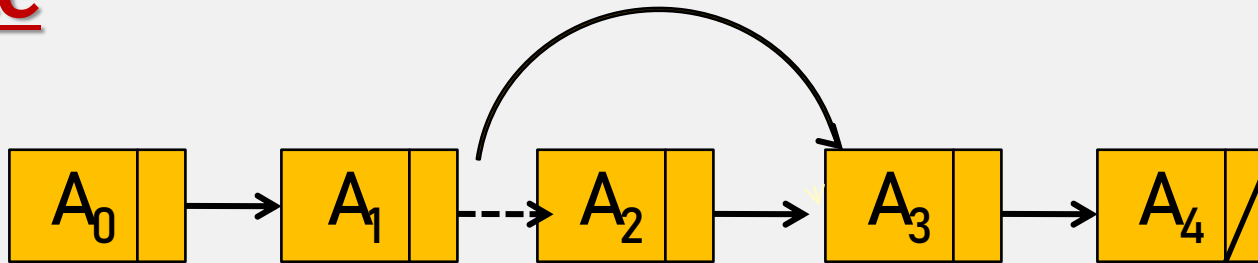
2.3.3 Linked List

Insert

Head ตัวชี้ Series

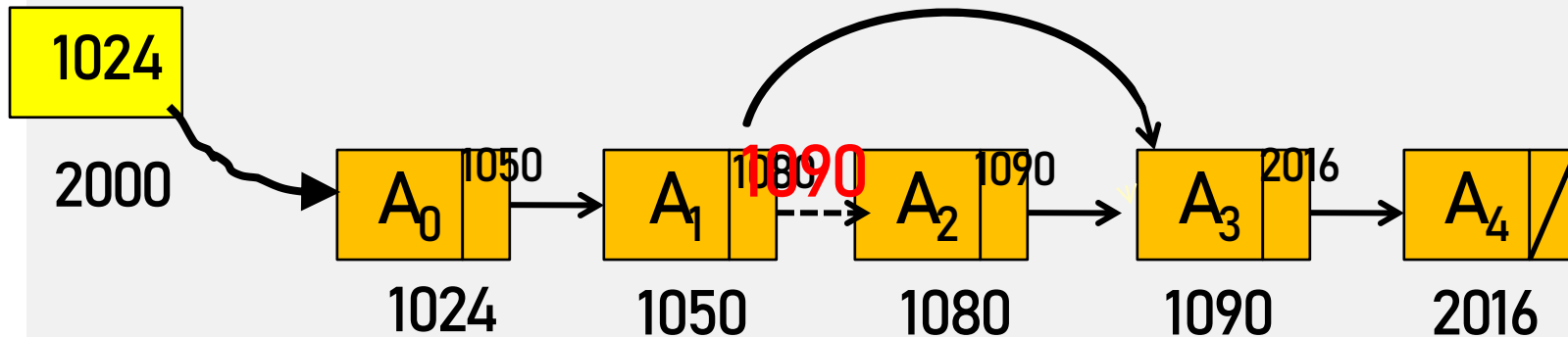


Delete



Delete

Head



ข้อแตกต่างระหว่าง Array list และ Linked list

1. การประกาศตัวแปร

Array list จะต้องประกาศตัวแปรก่อน จึงต้องคาดคะเนจำนวนข้อมูลไว้ ว่า list จะมีจำนวนกี่ตัว

Linked list ไม่จำเป็นจะต้องประกาศตัวแปรก่อน สามารถสร้าง node ขณะที่ run โปรแกรมได้

2. การ insert และ delete

Array list ทำได้ยากกว่า เพราะโครงสร้างไม่เหมาะสม
Linked list ทำได้ง่าย

ข้อแตกต่างระหว่าง Array list และ Linked list(ต่อ)

3. การเขียนโปรแกรม

Array list เขียนโปรแกรมโดยใช้การวน loop

Linked list เขียนโปรแกรมต้องใช้ pointer



1 การ Insert แยกกรณี

1. กรณีที่ไม่มีข้อมูล
2. กรณีที่มีข้อมูล
 - insert หน้าสุด
 - insert ตรงกลาง
 - insert ห้าย

เมื่อรับ input x
เข้ามากจะแยกความ
แตกต่างอย่างไร



03603212 : Module2-List, stack, Queue

3000

Head 2000

1024

8 Head = 3000

head เก็บ Value ไว้ก่อน

4

1024

8

1050

17

1080

19

2000

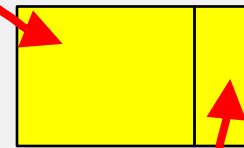
```
struct record{
```

```
    int value;
```

```
    struct record *next;
```

```
};
```

//กำหนด structure ของ linked list



1024

1024 → Value = 4

head → Value = 1024

head → next = 1050

1050 → value = 8



03603212 : Module2–List, stack, Queue

```
int menu()
{
    int choose;
    cout << "=====Menu =====\n";
    cout << " 1) Insert list\n";
    cout << " 2) Delete list\n";
    cout << " 3) Print list\n";
    cout << " 4) Exit\n";
    cout << " Please choose > ";
    cin >> choose;
    return choose;
}
```



```
struct record *insert(struct record *head,int data)
```

```
1 { struct record *node,*p;
```

```
2 if ( head == NULL ) *** T
```

```
3 { head=new struct record;
```

```
4     head-> value = data;
```

```
5     head-> next = NULL;
```

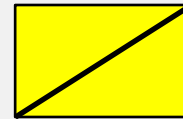
```
6 }
```

```
7 return head;
```

NULL

5

head



2000

head



2000

จบพื้นที่นี้



1024

return pointer

ให้เปลี่ยน * ไว้ข้างหน้า

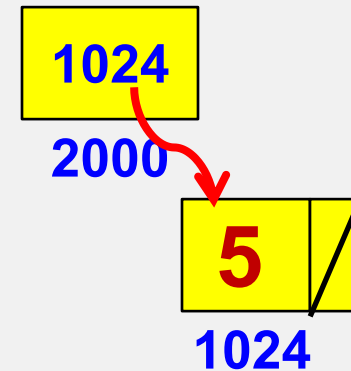


03603212 : Module2-List, stack, Queue

```
struct record *insert(struct record *  
1 { struct record *node,*p;  
2   if ( head == NULL )  
3   { head=new struct record;  
4     head-> value = data;  
5     head-> next = NULL;  
6   }
```

2. กรณีมีข้อมูลอยู่แล้ว
-Insert ด้านหน้า

head

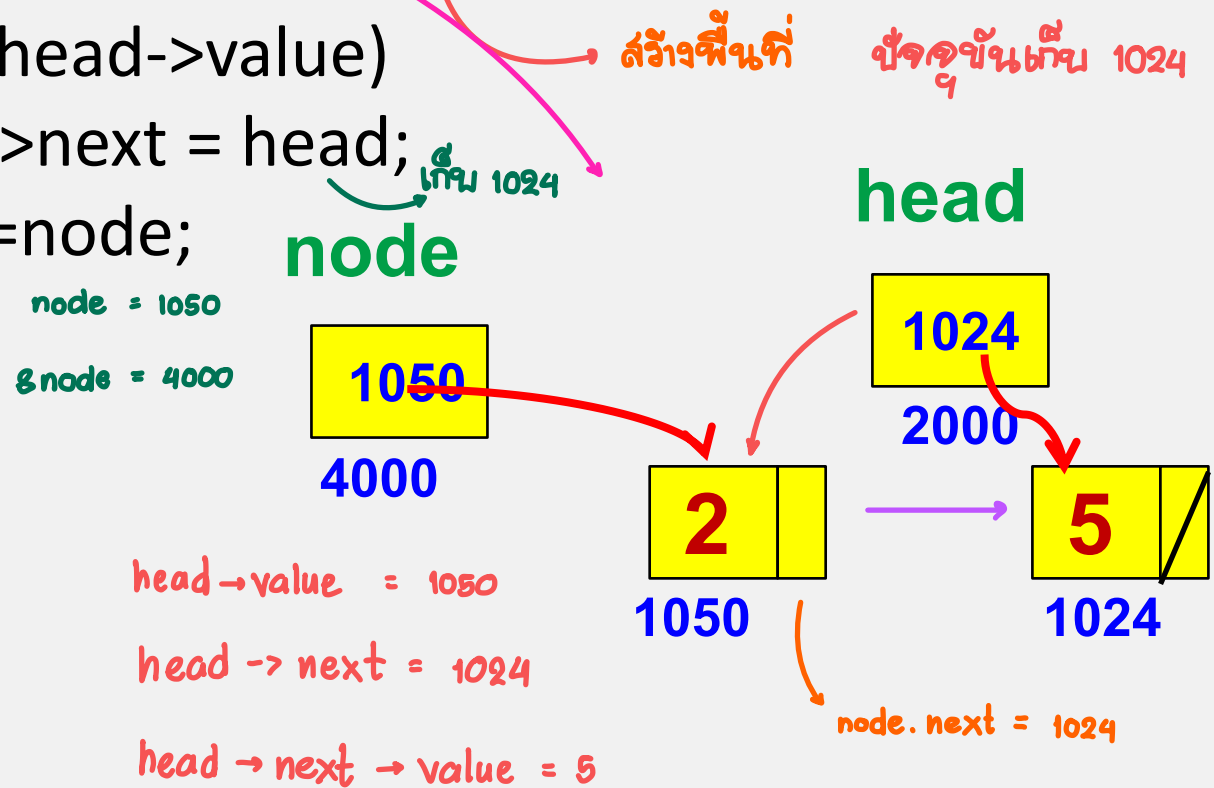




03603212 : Module2-List, stack, Queue

```
7 else    /**head !=NULL **/  
8 {      node=new struct record;  
9        node-> value = data;  
10       if( data < head->value)  
11       {   node->next = head;  
12           head=node;  
13       }  
14     }  
15 return head;
```

2. กรณีมีข้อมูลอยู่แล้ว
-Insert ด้านหน้า





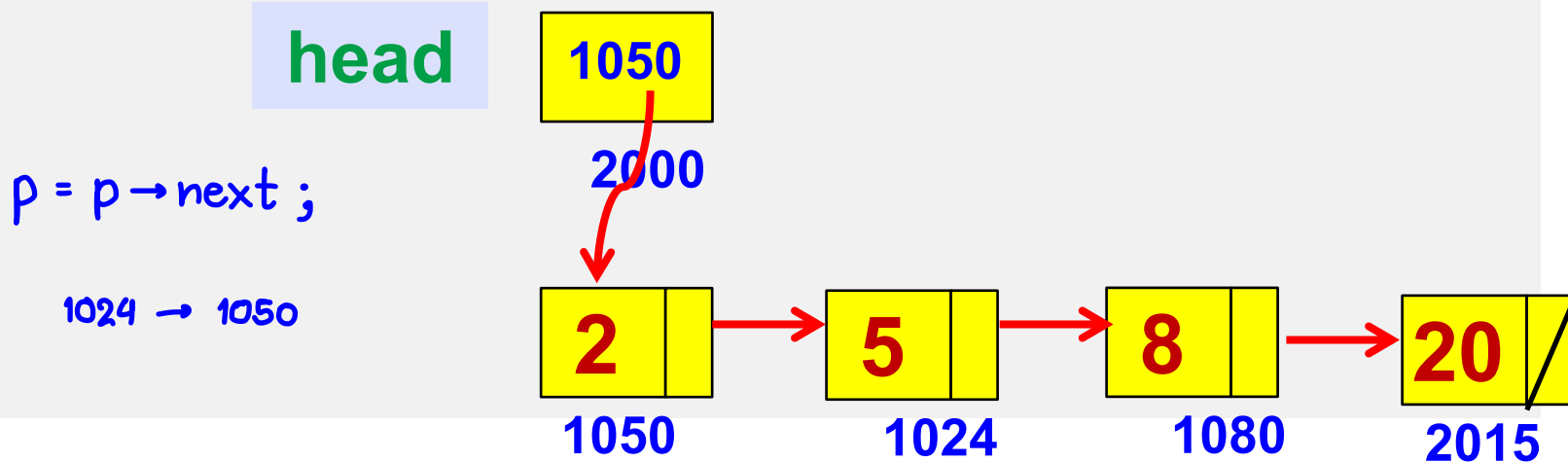
03603212 : Module2-List, stack, Queue

```
struct record *insert(struct record *head,int data)
```

```
1 { struct record *node,*p;  
2   if ( head == NULL )  
3   { head=new struct record;  
4     head-> value = data;  
5     head-> next = NULL;  
6   }
```

3. กรณีแทรกกลาง
หรือท้าย

9



```
7 else    /**head !=NULL **/  
8 {      node=new struct record;  
9        node-> value = data;  
10       if( data < head->value)  
11       {  
12         ...  
13       }  
        else  
        {  
          ...  
        }  
      }
```

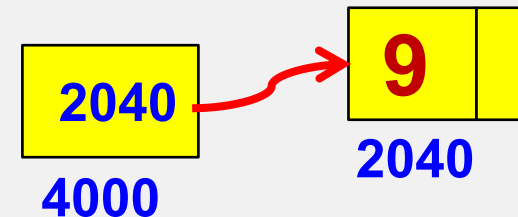
แทรกหน้า list กรณี 2 เรียบแล้ว ♥

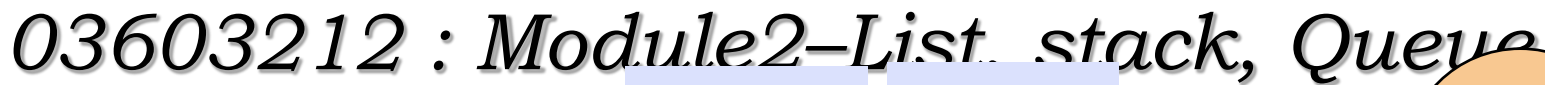
...

แทรกกลาง/ท้าย

3. กรณีแทรกกลาง

node





ထည့်သွင်းခြင်း insert

p

1050

5000

20

2015

କଟିକ

node

```
if( data < p->next->value) //แทรก
```

```
{ node->next=p->next; เก็บเงิน 2015
```

p->next = node; **2040** **พาทท้าย list**

```
node->next = p->next;
```

```
p->next = node;
```

```
node → next = node ;
```

```
p->next = NULL;
```

ଅର୍ଥ ୪

หน้า 20

កងឧក្រិដ្ឋ

14 else

```
15 { p=head;
```

```
16 while( p !=NULL) TT
```

17 {

```
18     if( data < p->next->value) //แทรก
```

```
19 { node->next=p->next; กันยายน 2015
```

20 p->next = ²⁰⁴⁰node; นำ node ที่ 1 ไป list

```
21      break;
```

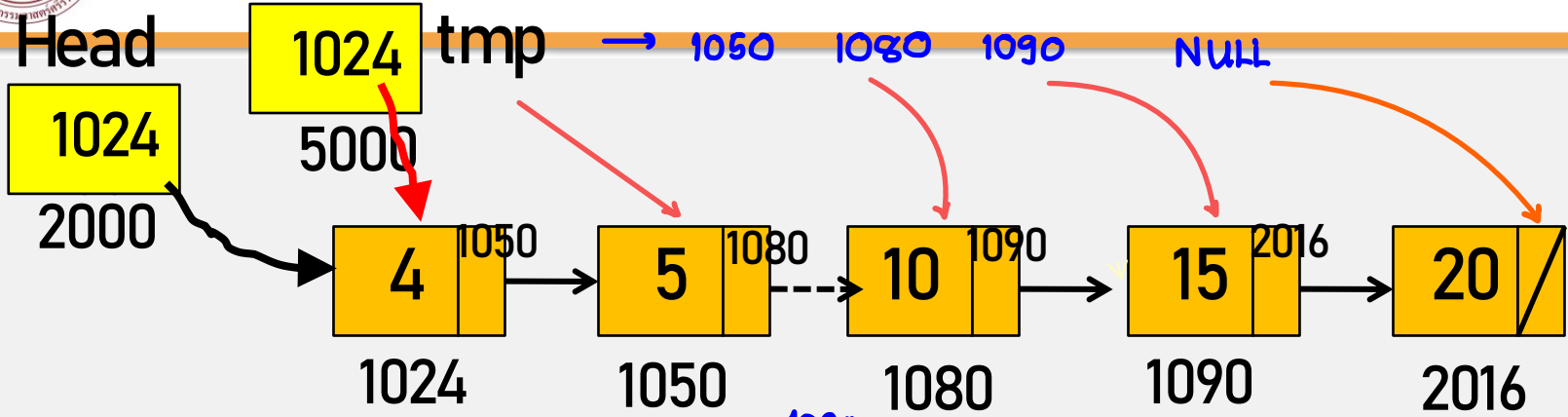
22 }

```
23     else p=p->next;
```

```
24    }//end while    } //end else    }/* end else head !=NULL */
```

```
25 return head; }
```


03603212 : Module2-List, stack, Queue



```
void print(struct record *head)
```

```
{
    cout << "\nPrint Listed : \n";
    struct record *tmp;
    tmp=head;
    while(tmp!=NULL)
    {
        cout << tmp->value << " ";
        tmp=tmp->next;
    }
}
```

4 5 10 15 20

int cout = 0;

T T T T T F

1, 2, 3