



Struktur Data

Meet 04



01

Linked List

Intro



List VS Linked List

11	3	23	7
0	1	2	3

List VS Linked List



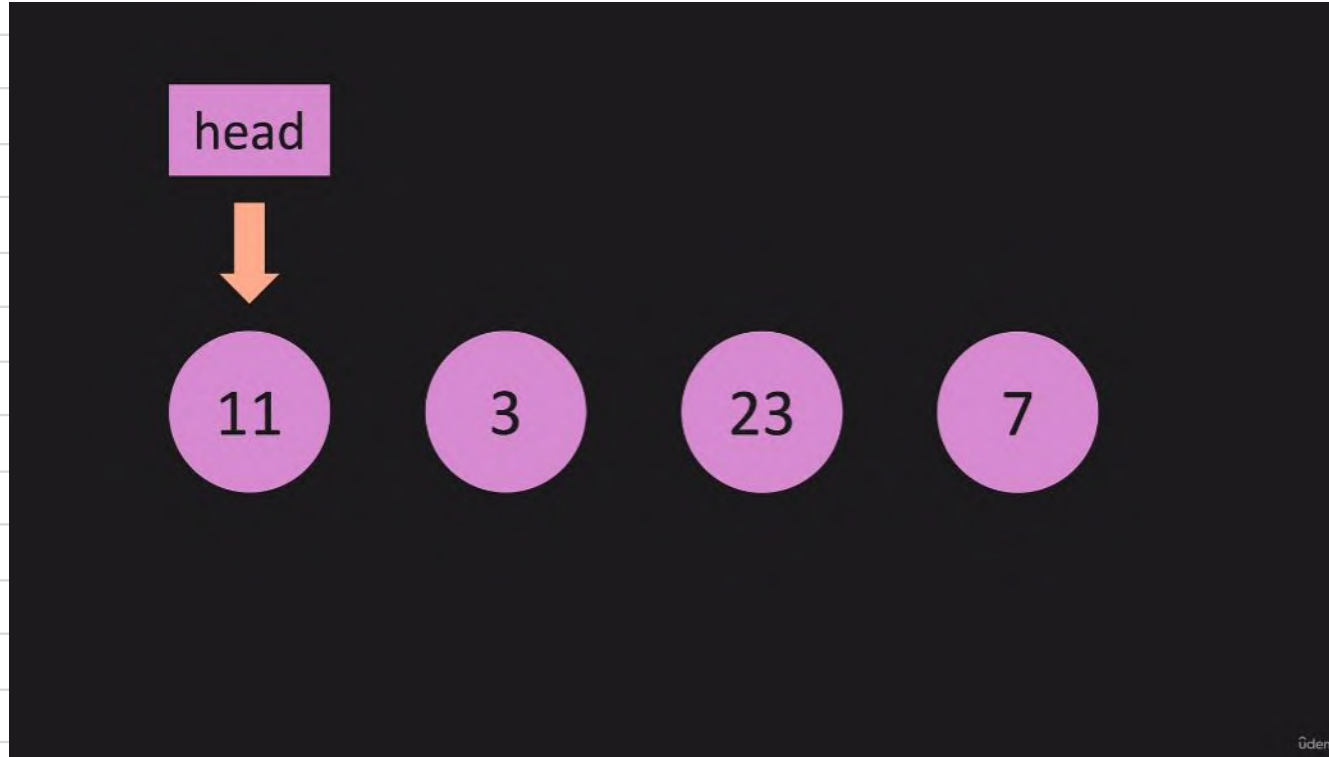
List VS Linked List



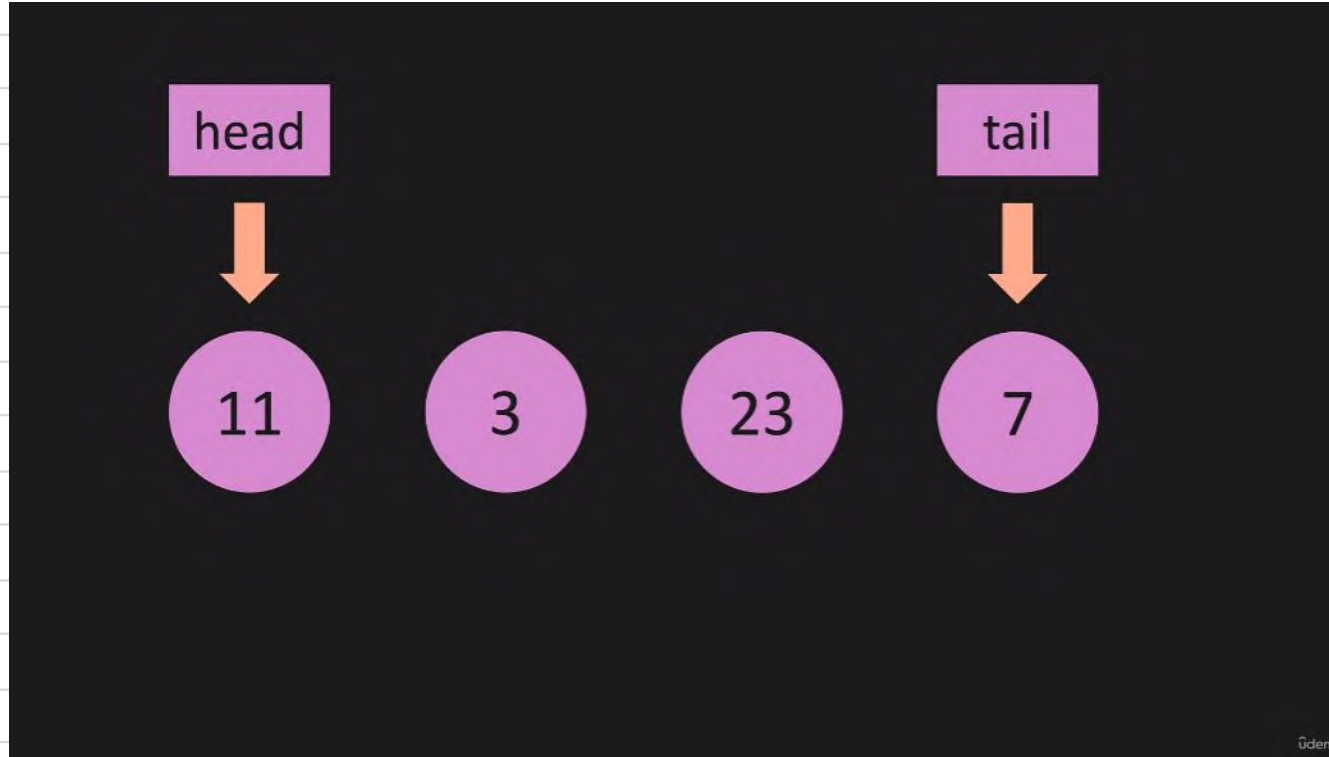
List VS Linked List



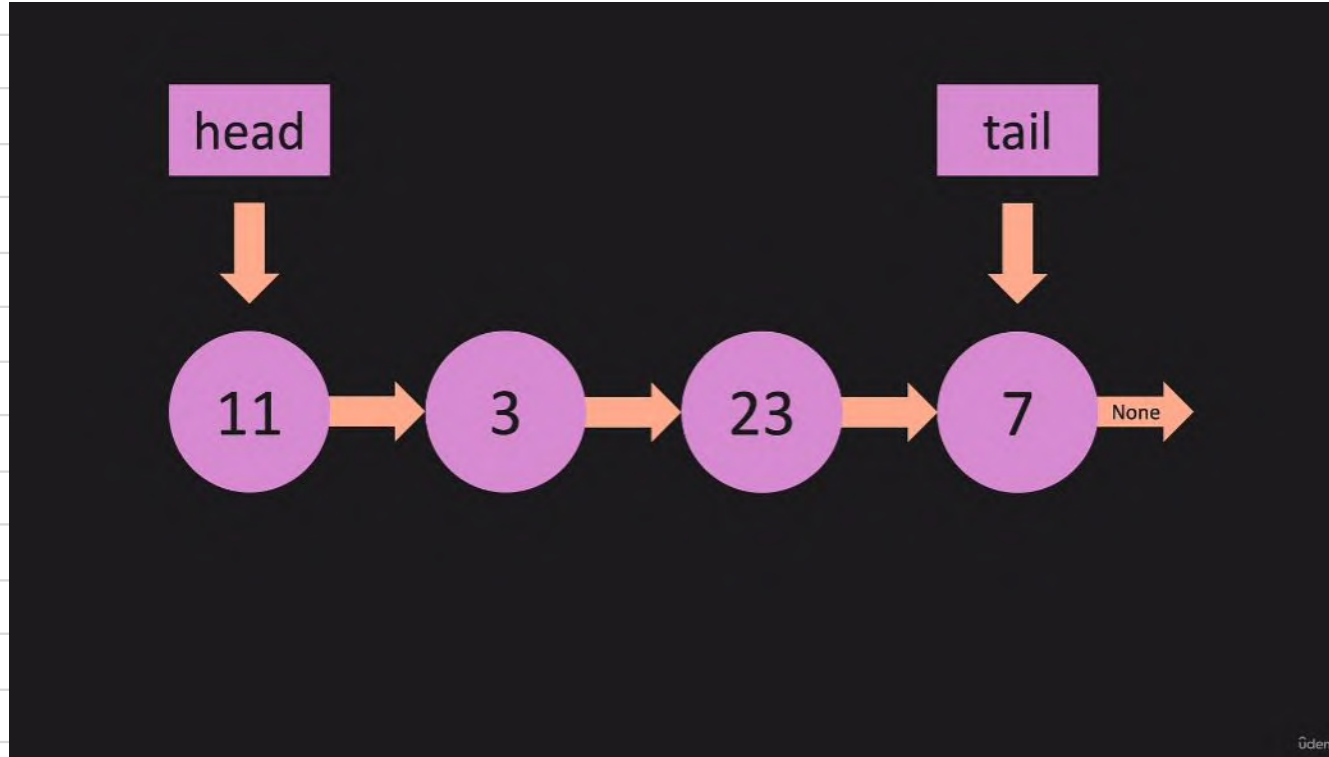
List VS Linked List



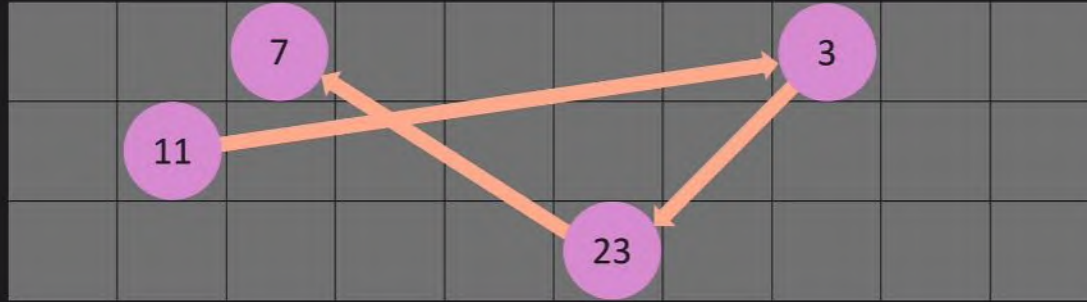
List VS Linked List



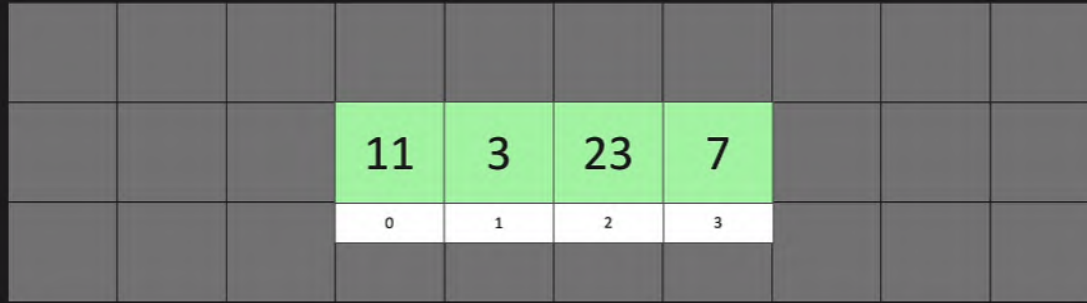
List VS Linked List



List VS Linked List



List VS Linked List



The diagram illustrates an array-based list structure. It consists of a 3x10 grid of cells. The middle row contains the values 11, 3, 23, and 7 in the 4th, 5th, 6th, and 7th columns, respectively. The bottom row contains the indices 0, 1, 2, and 3 in the same columns. The cells containing the values and indices are highlighted in light green. The rest of the grid is dark gray.

			11	3	23	7			
			0	1	2	3			



02

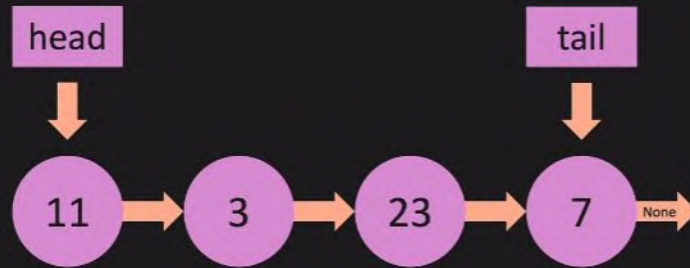
Linked List

Big O



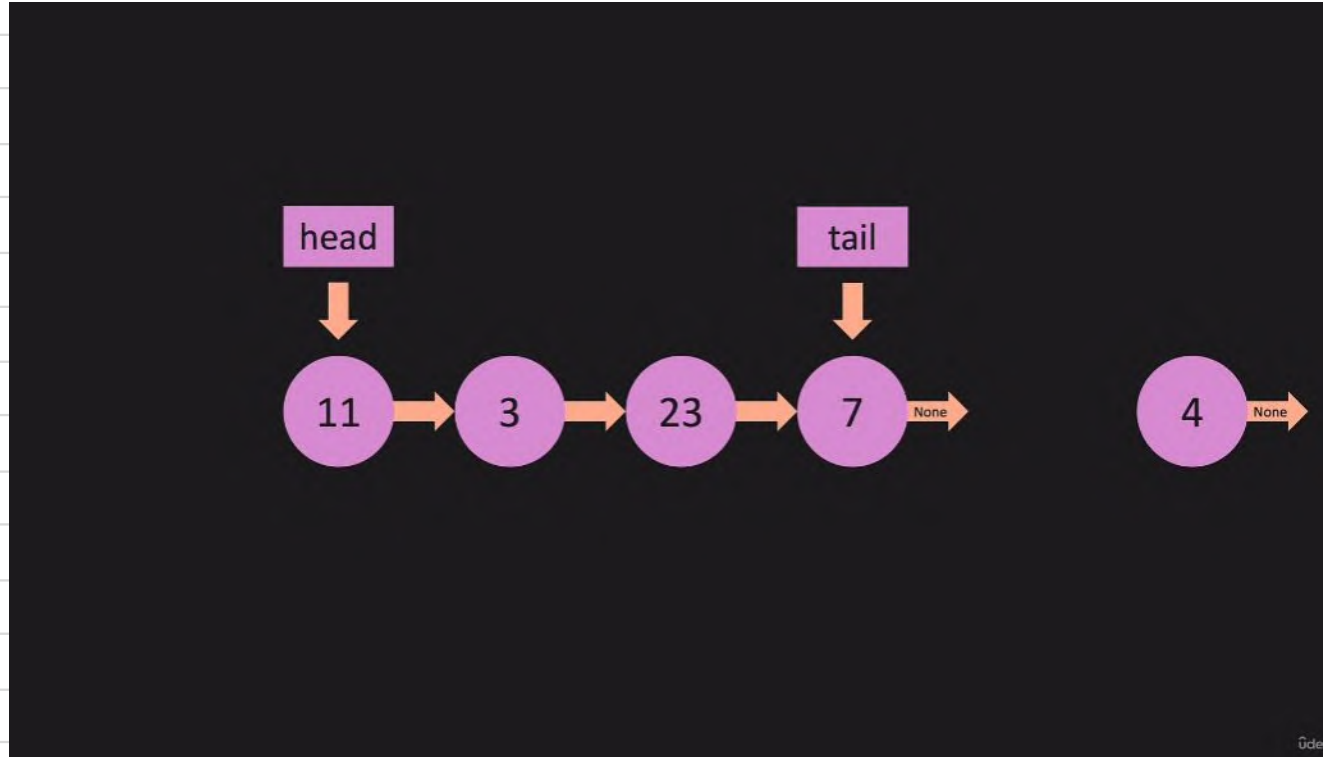
Diketahui suatu Linked List...

Add & Remove Node di Belakang



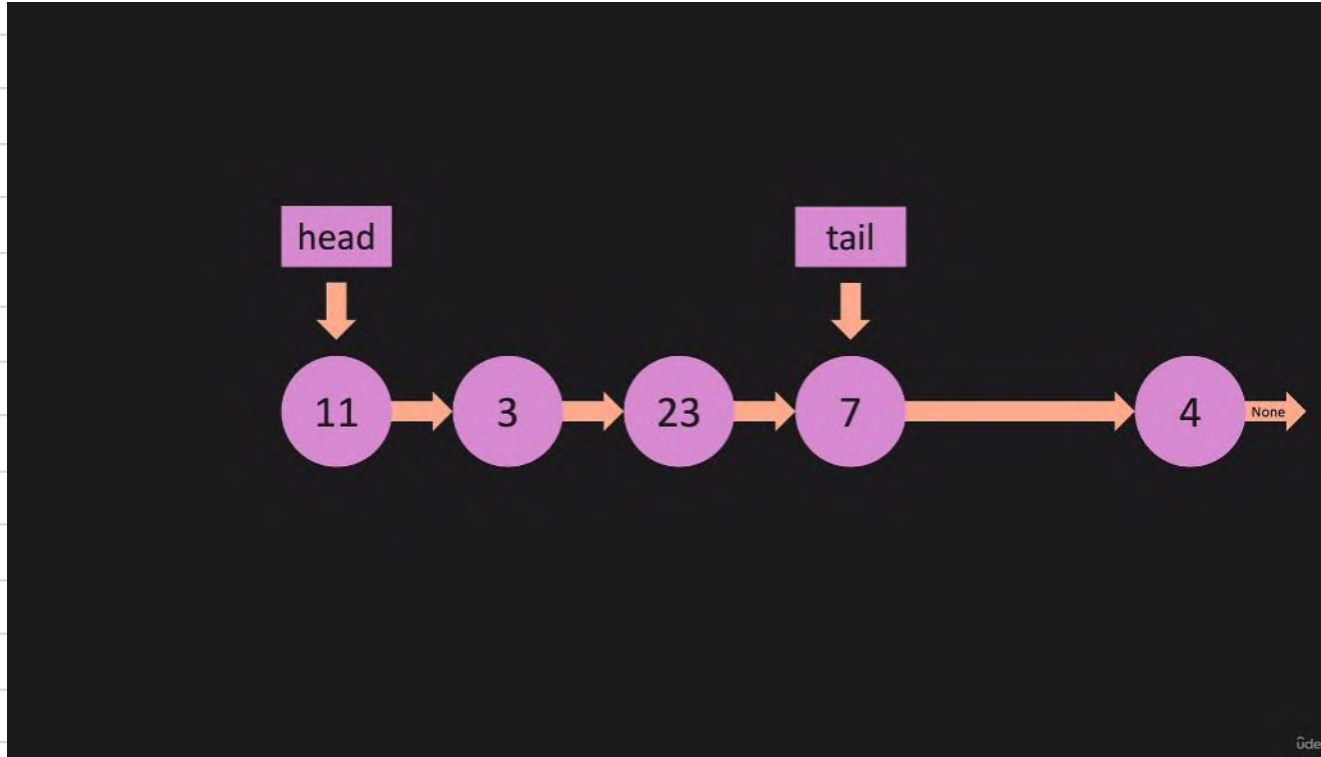
Add & Remove Node di Belakang

Bagaimana jika kita ingin menambahkan node "4" di belakang linked list



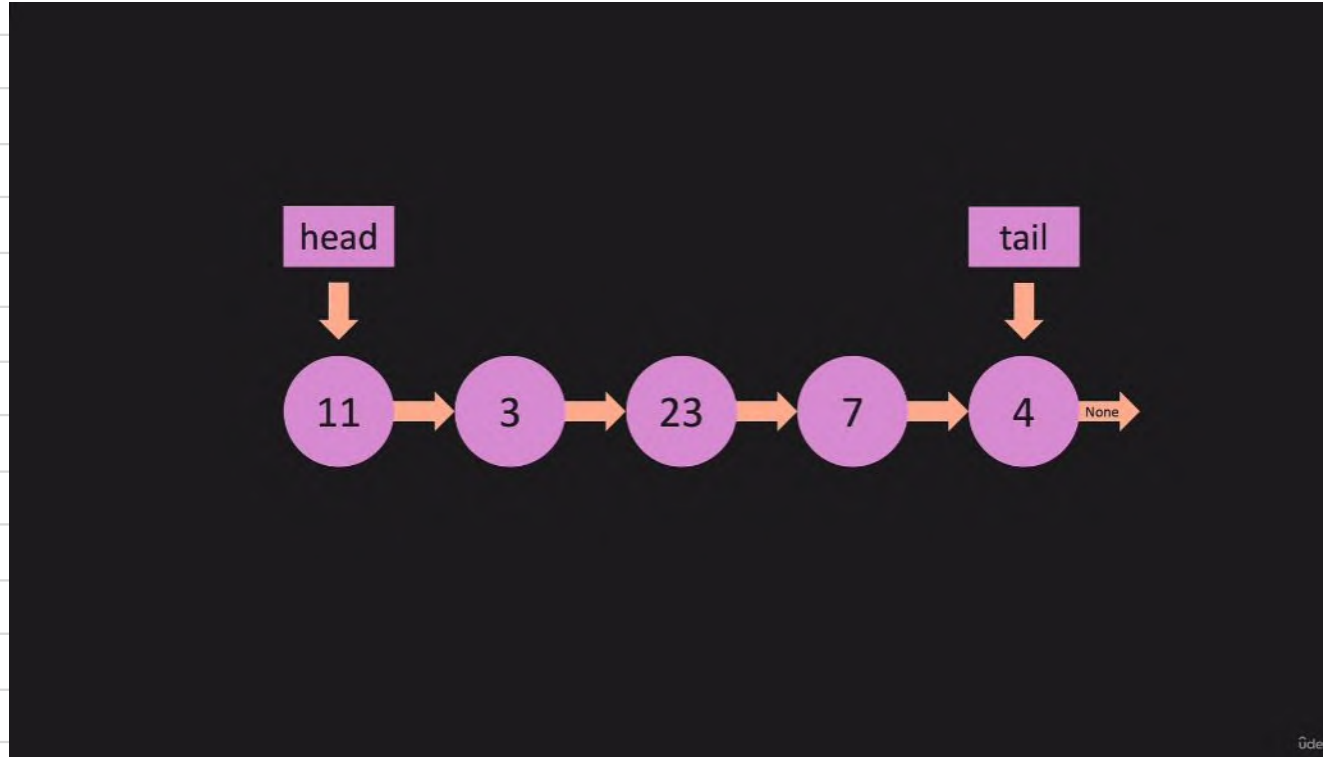
Add & Remove Node di Belakang

Bagaimana jika kita ingin menambahkan node "4" di belakang linked list



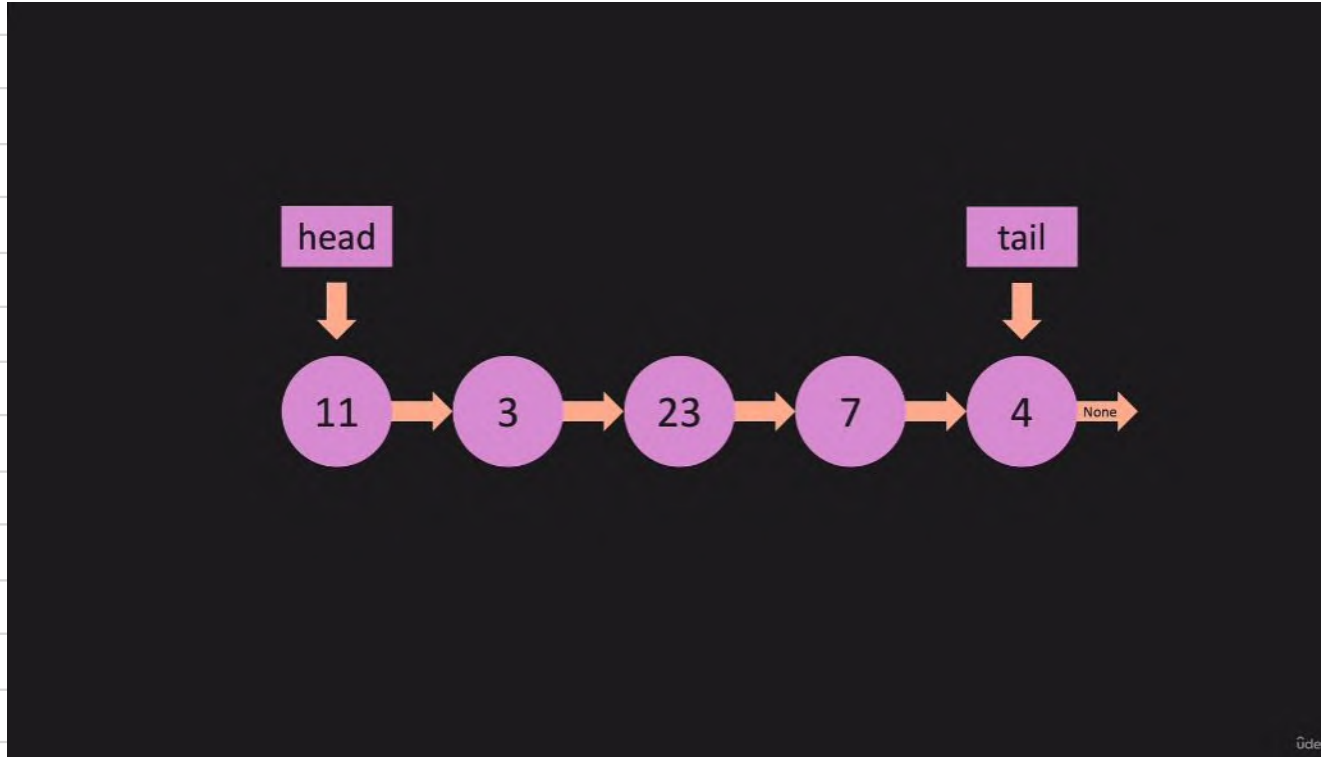
Add & Remove Node di Belakang

Bagaimana jika kita ingin menambahkan node "4" di belakang linked list

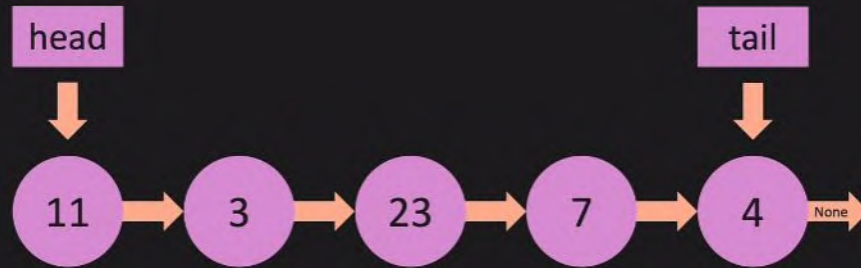


Add & Remove Node di Belakang

Apa notasi Big O untuk
Add Last ... ?



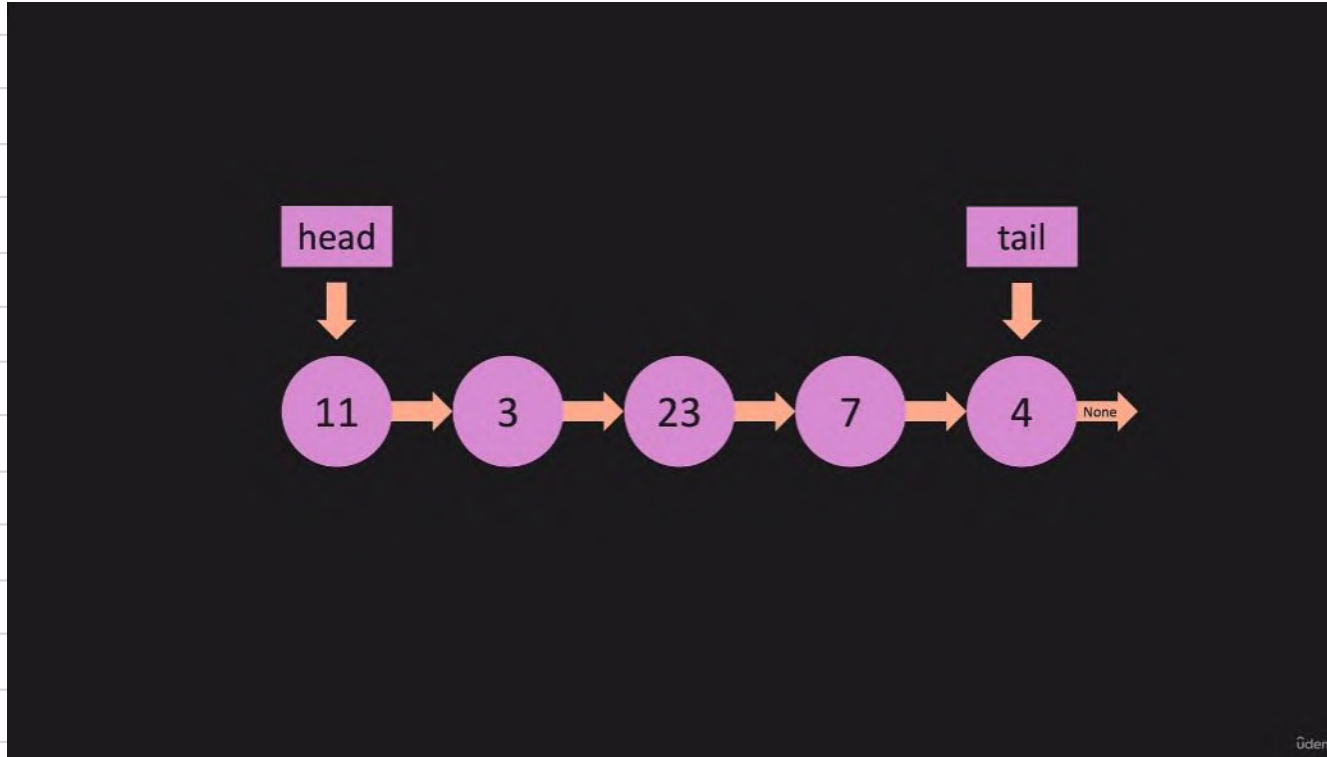
Add & Remove Node di Belakang



$O(1)$

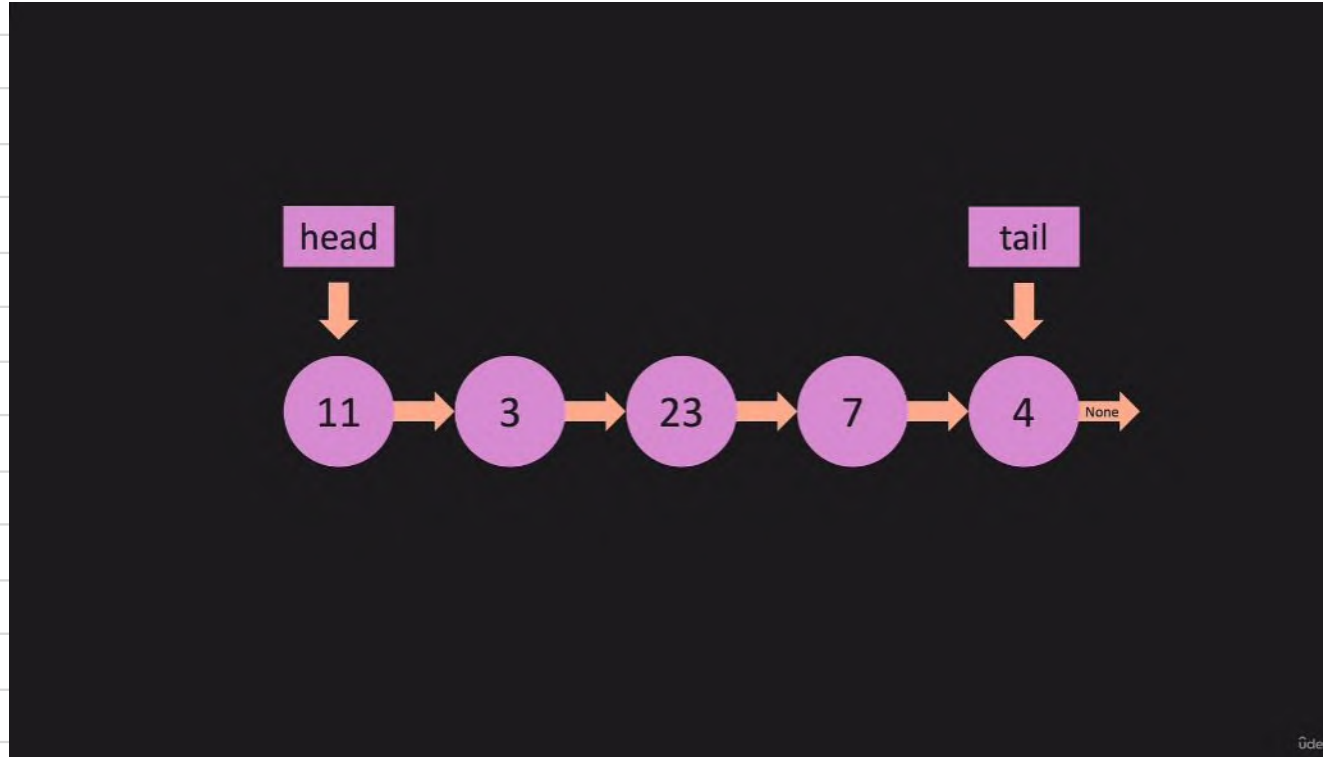
Diketahui suatu Linked List...

Add & Remove Node di Belakang



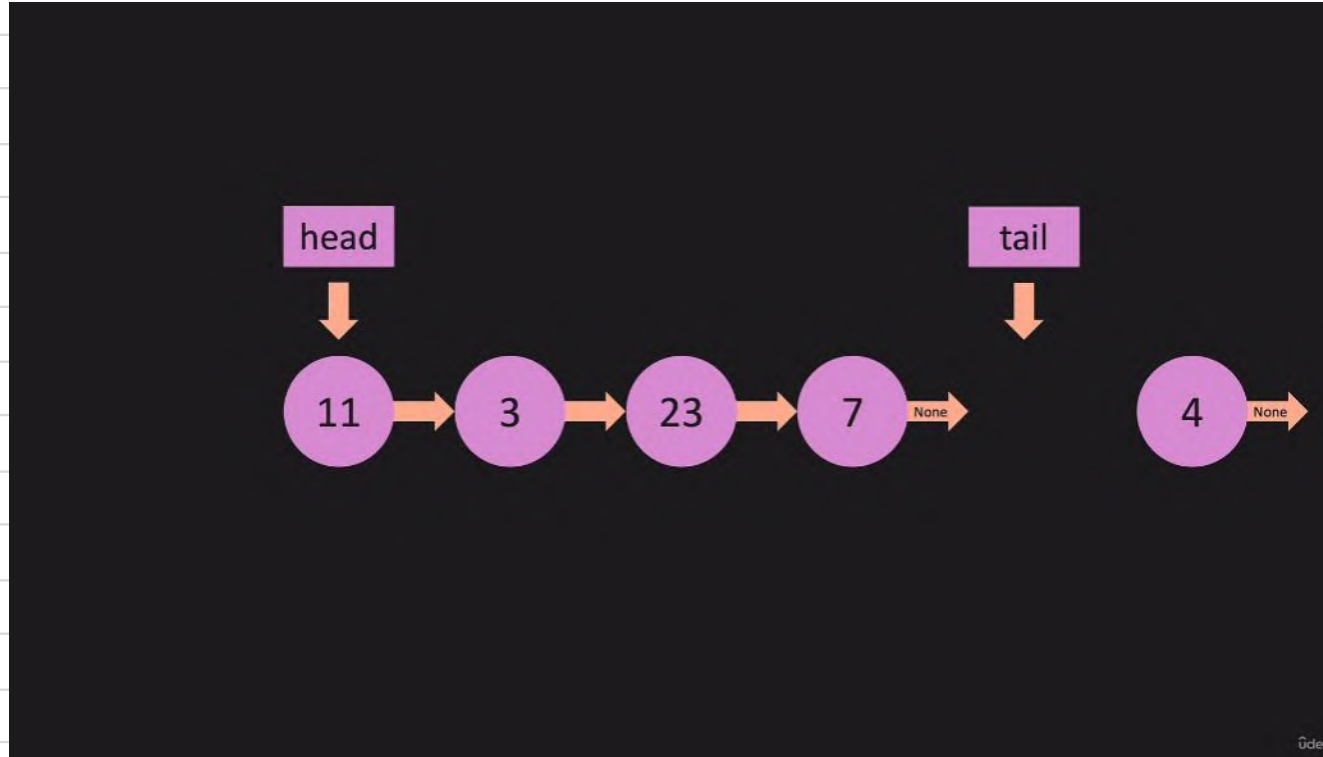
Add & Remove Node di Belakang

Bagaimana jika kita ingin menghapus node "4" dari linked list



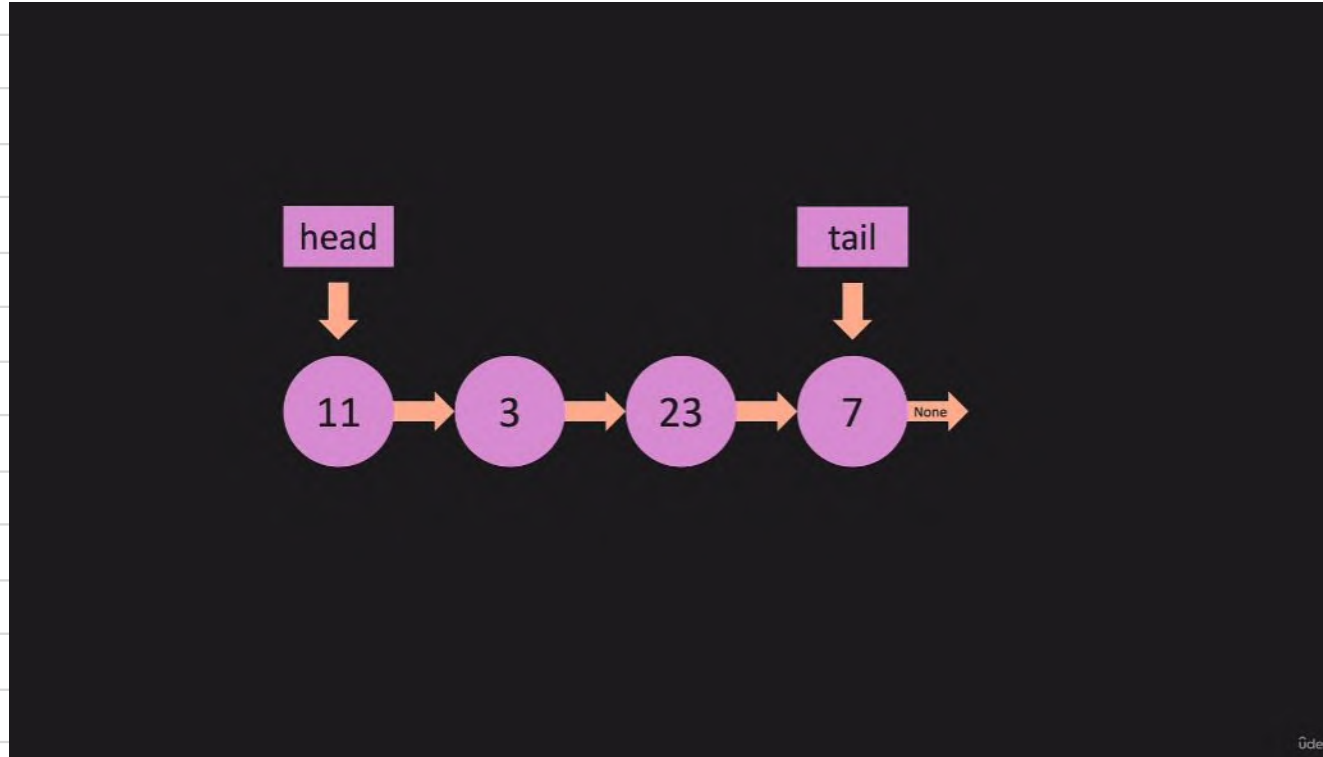
Add & Remove Node di Belakang

Bagaimana jika kita ingin menghapus node "4" dari linked list



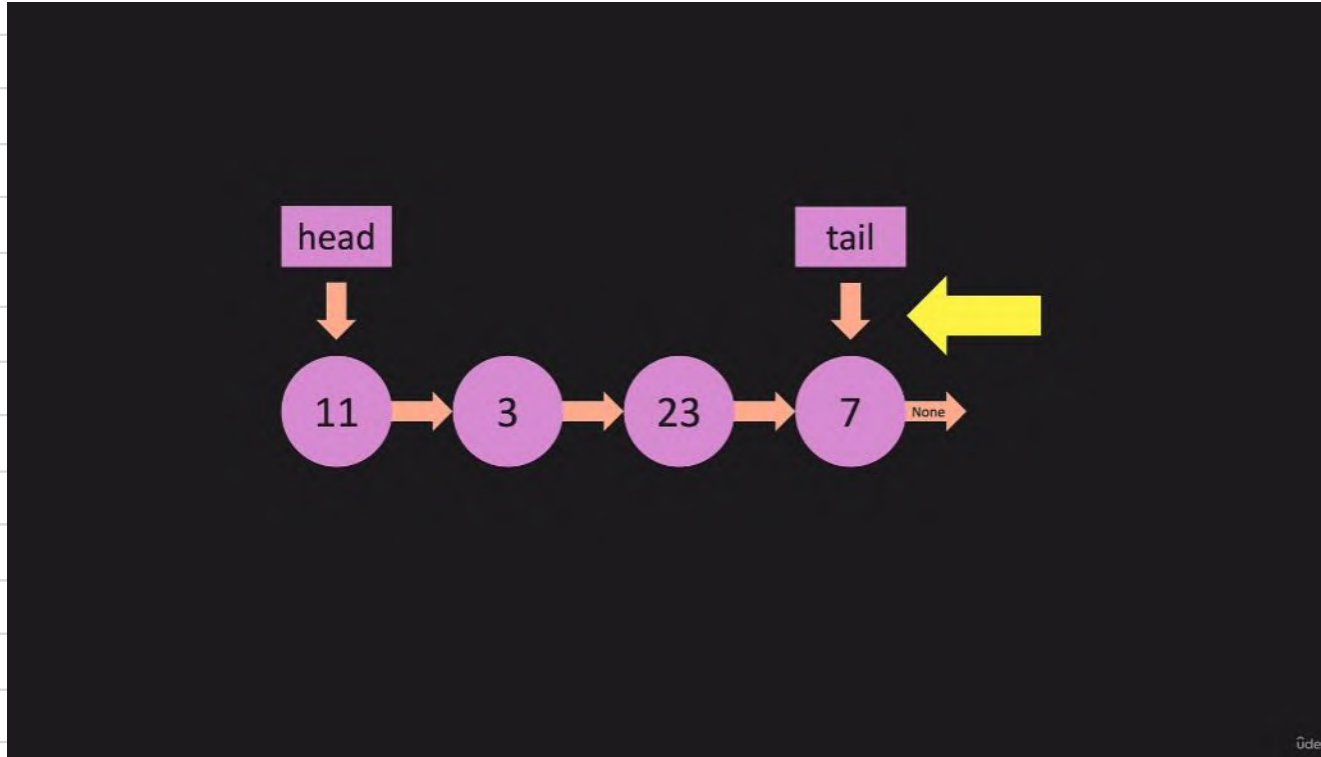
Add & Remove Node di Belakang

Bagaimana jika kita ingin menghapus node "4" dari linked list



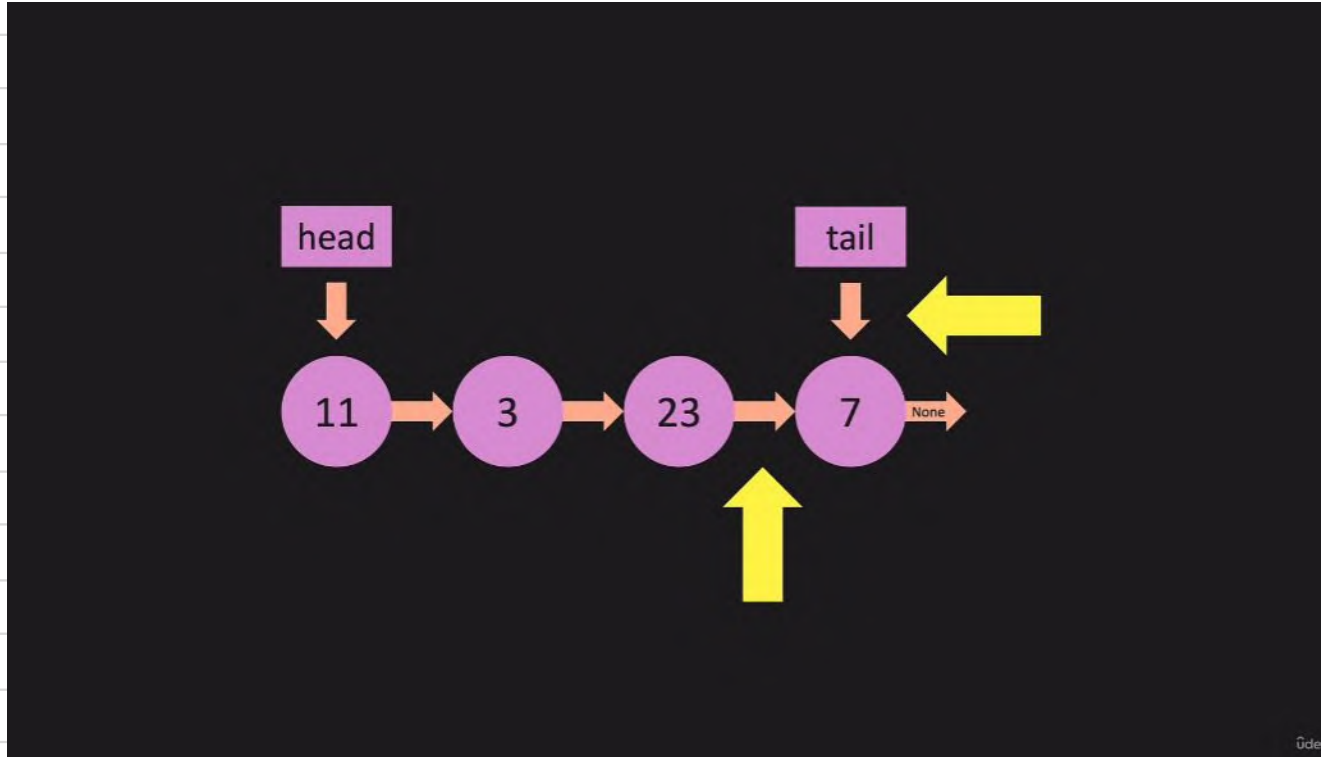
Add & Remove Node di Belakang

Bagaimana mengarahkan "tail" ke situ?



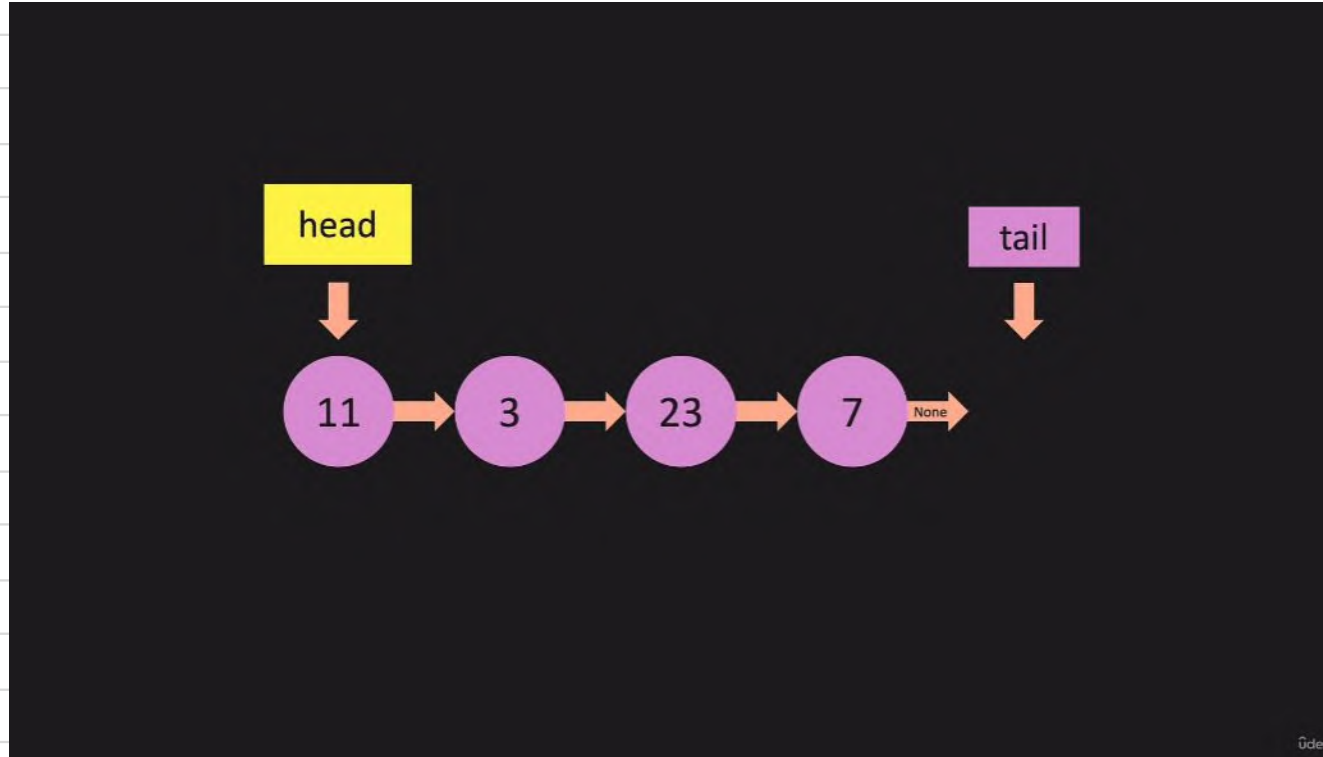
Add & Remove Node di Belakang

Bagaimana mengarahkan "tail" ke situ?



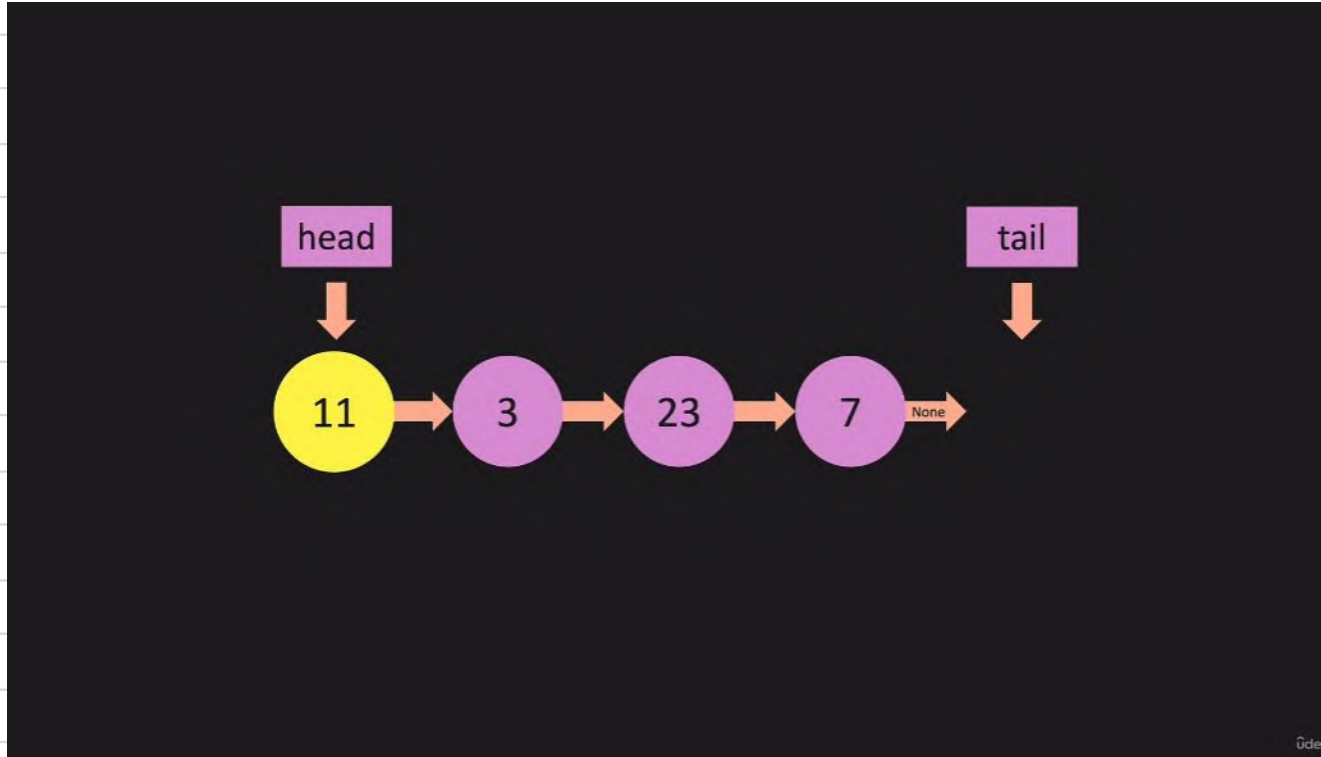
Add & Remove Node di Belakang

Bagaimana mengarahkan "tail" ke situ?



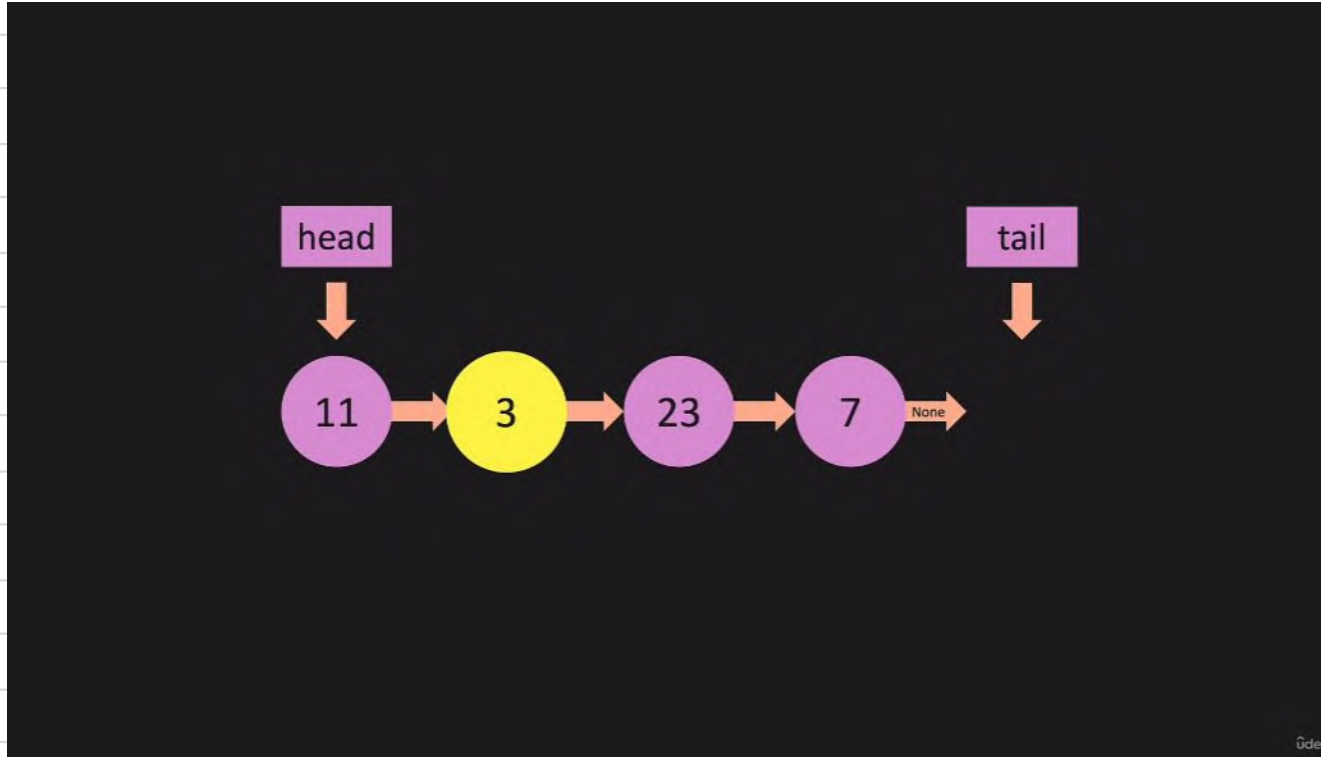
Add & Remove Node di Belakang

Bagaimana mengarahkan "tail" ke situ?



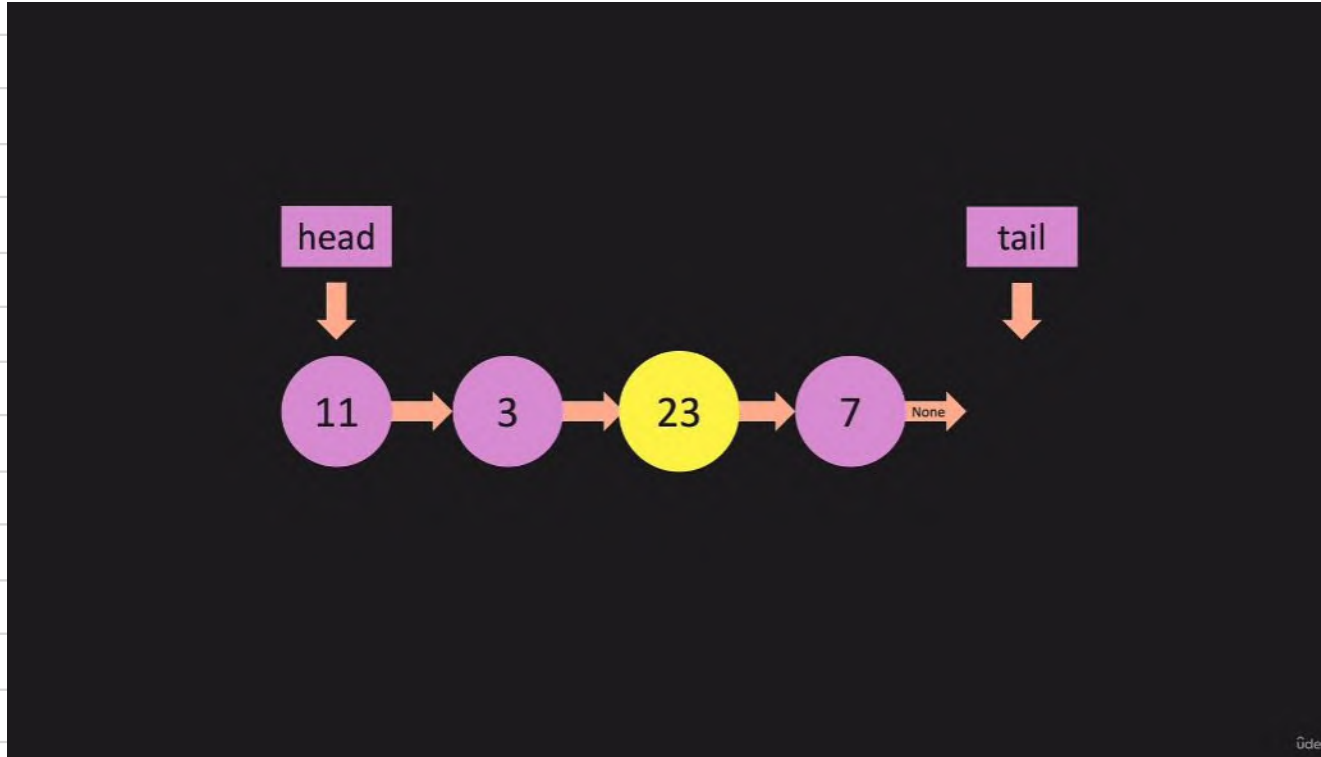
Add & Remove Node di Belakang

Bagaimana mengarahkan "tail" ke situ?



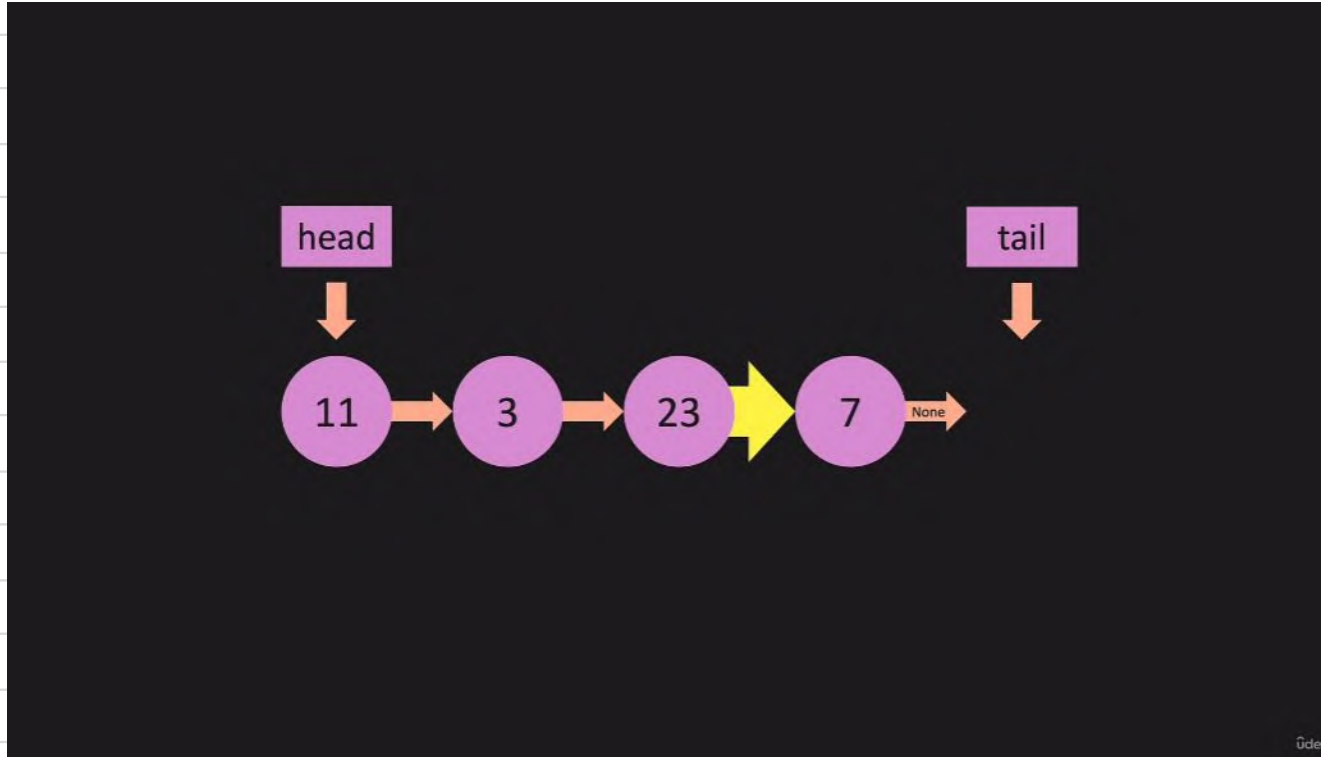
Add & Remove Node di Belakang

Bagaimana mengarahkan "tail" ke situ?



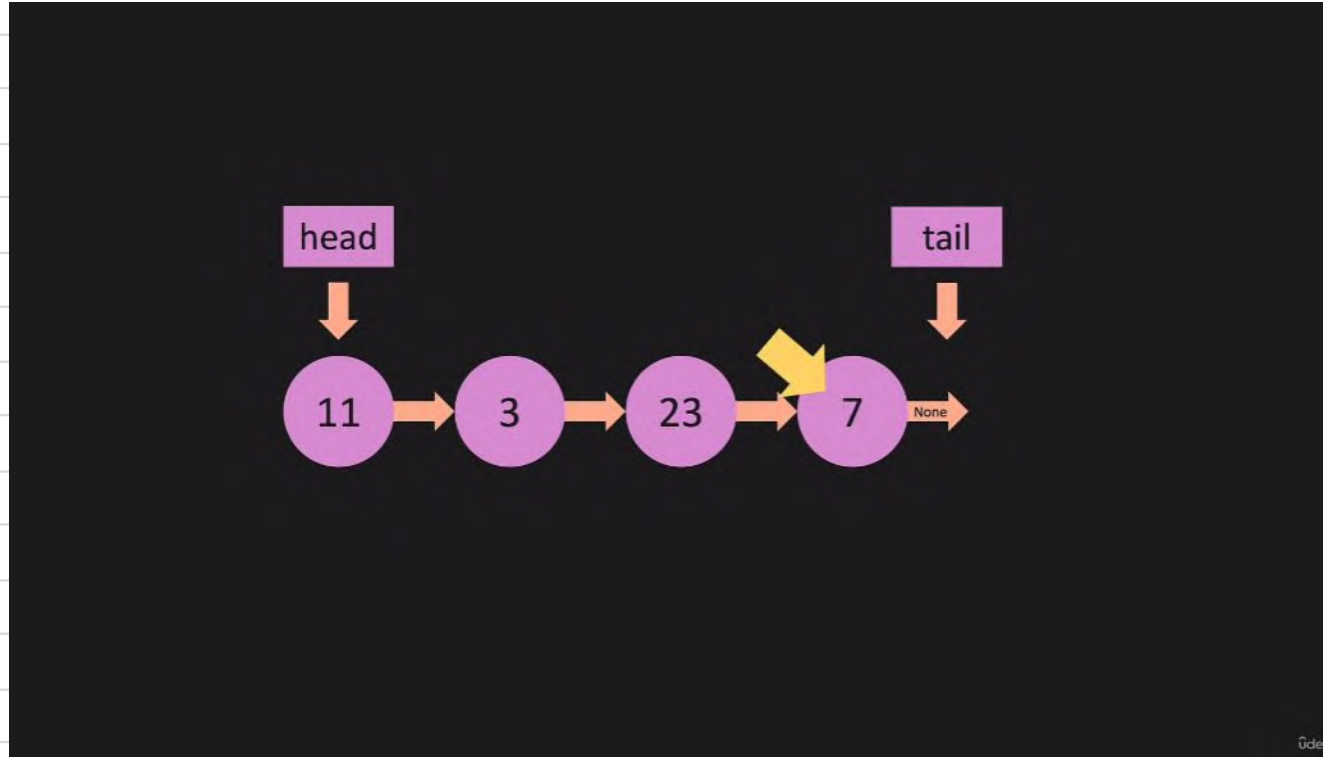
Add & Remove Node di Belakang

Bagaimana mengarahkan "tail" ke situ?



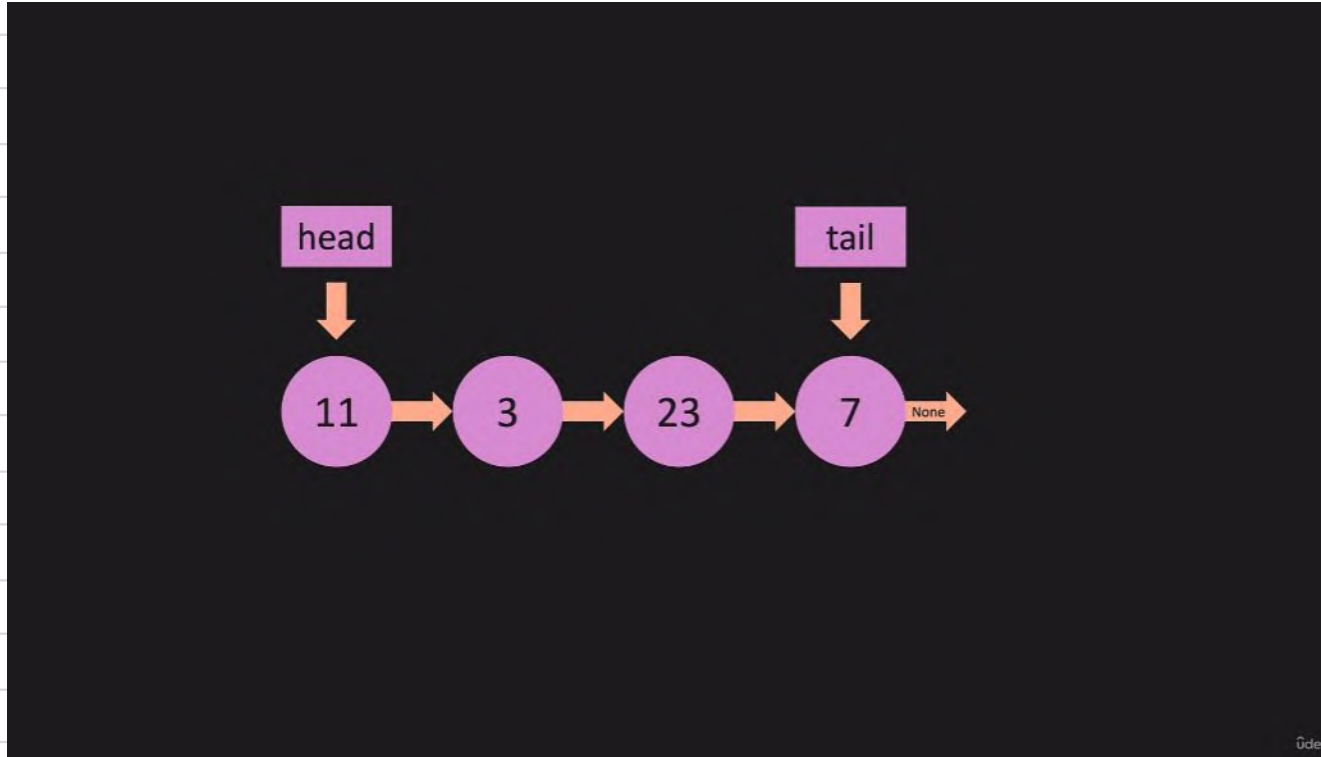
Add & Remove Node di Belakang

Bagaimana mengarahkan "tail" ke situ?



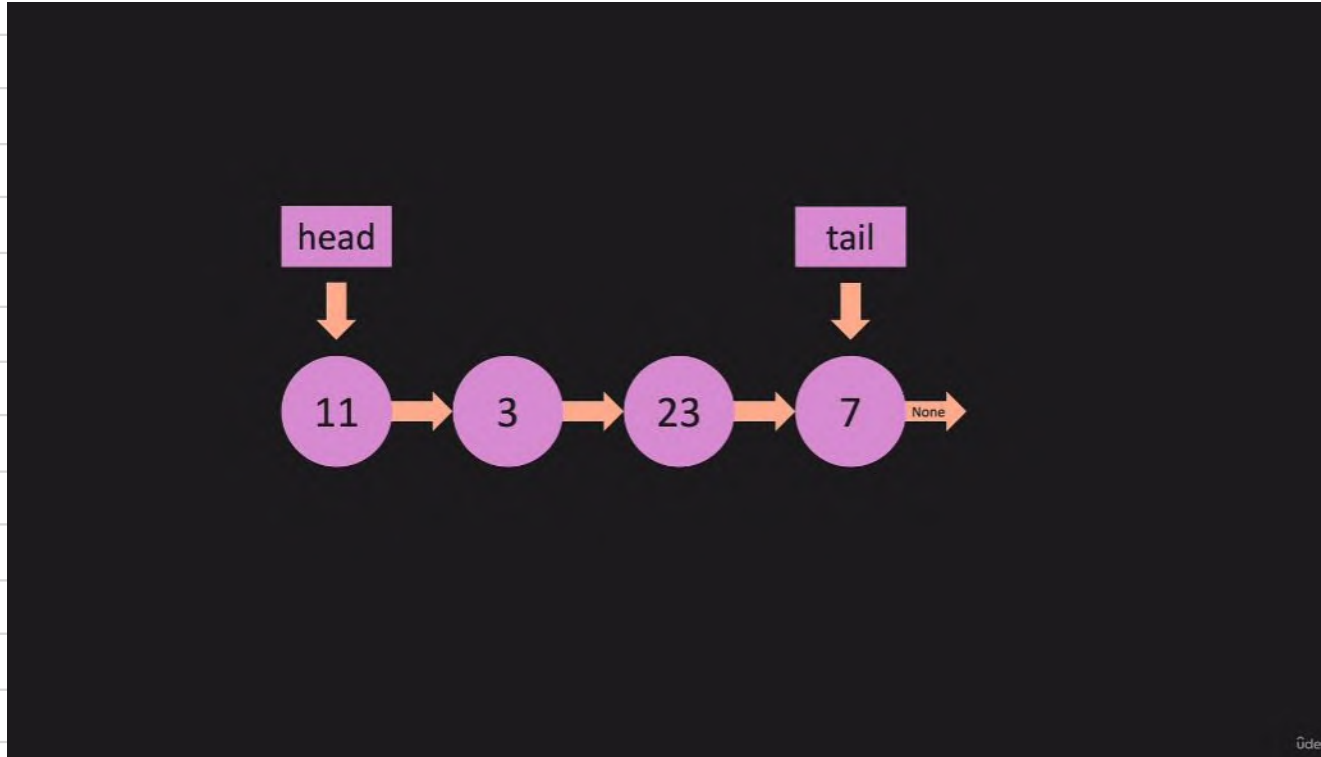
Add & Remove Node di Belakang

Bagaimana mengarahkan "tail" ke situ?



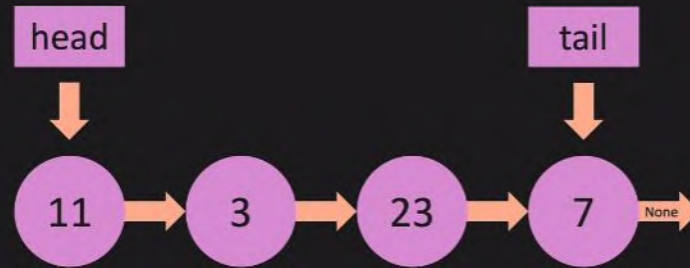
Add & Remove Node di Belakang

Apa notasi Big O untuk
Remove Last ... ?



Add & Remove Node di Belakang

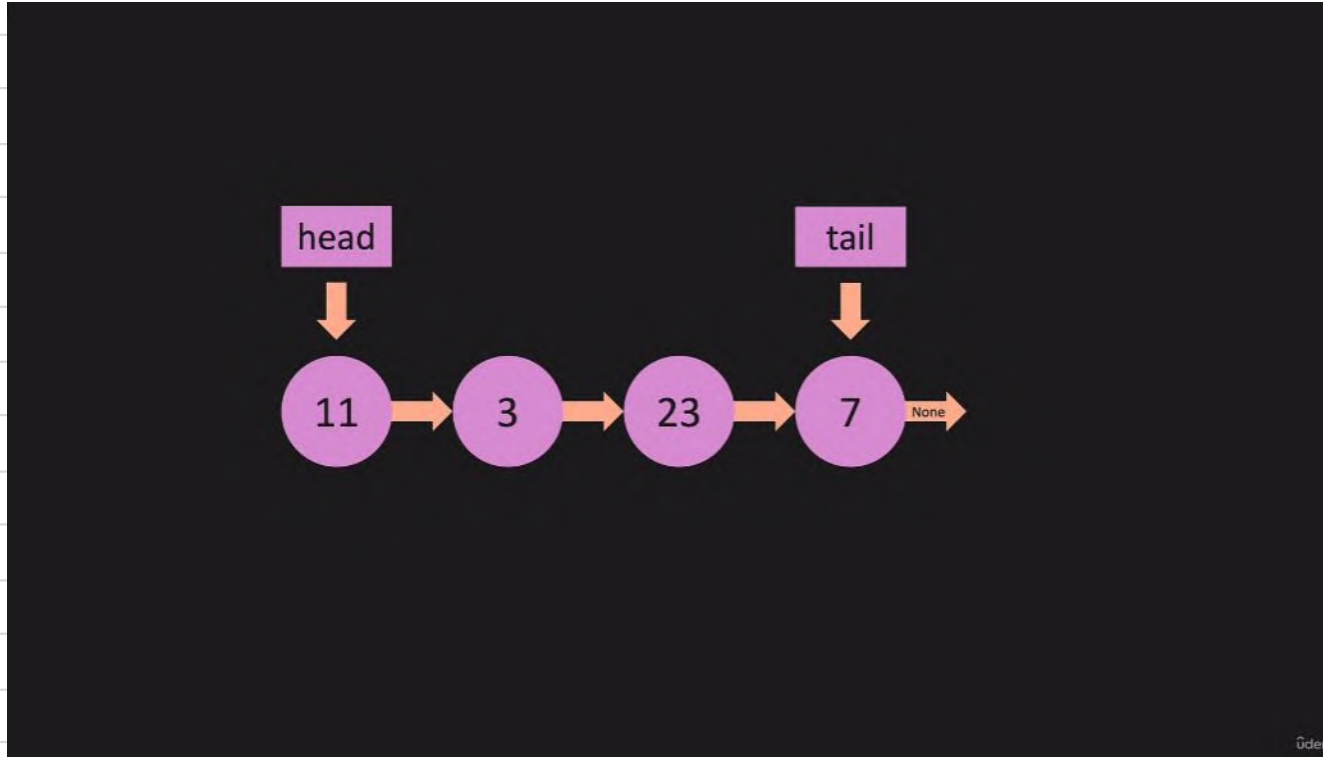
Apa notasi Big O untuk
Remove Last ... ?



$O(n)$

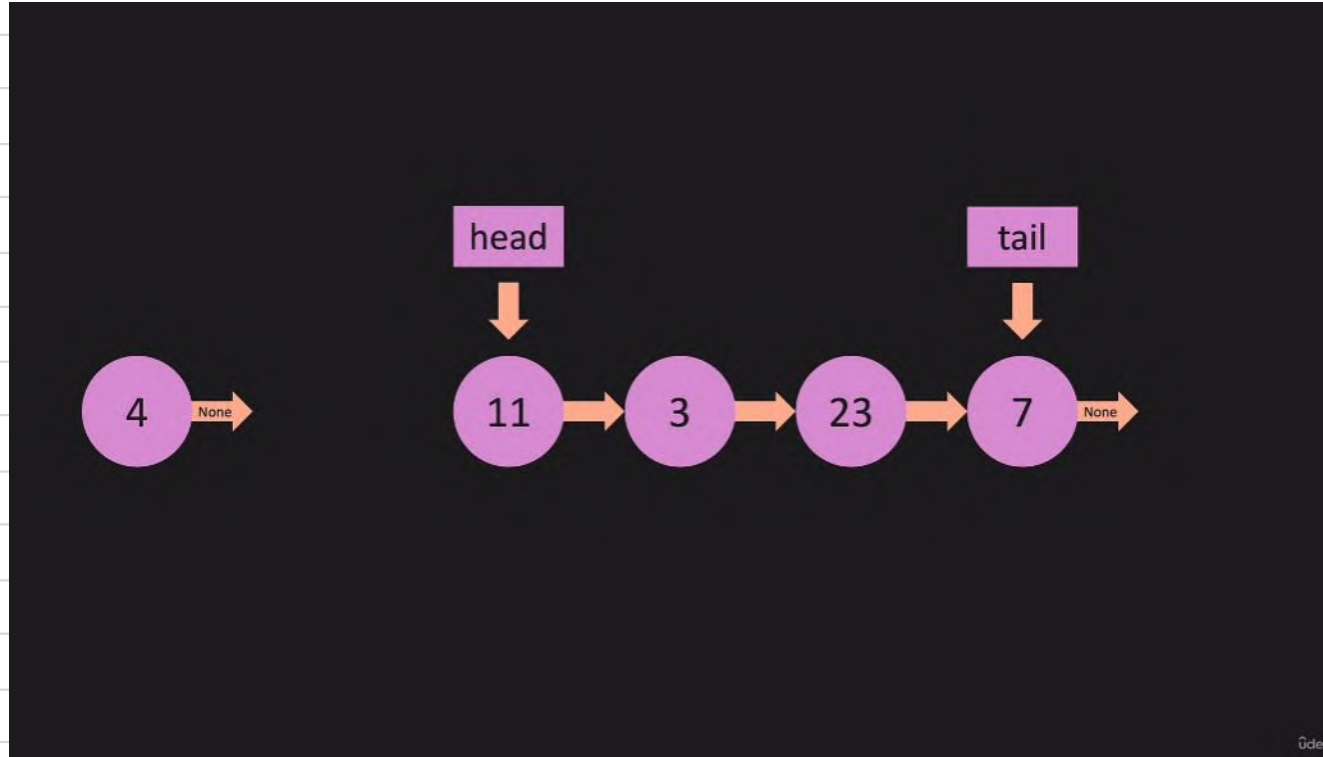
Diketahui suatu Linked List...

Add & Remove Node di Depan



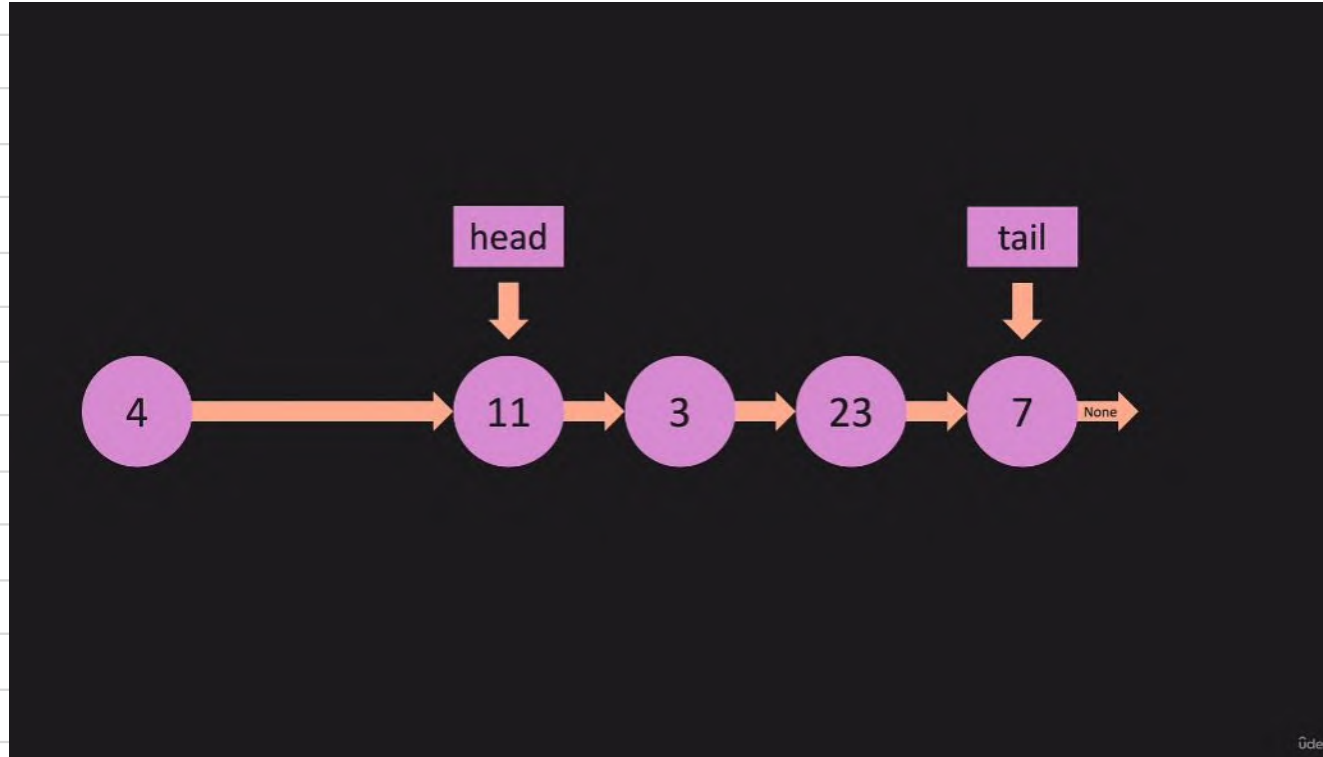
Add & Remove Node di Depan

Bagaimana jika kita ingin menambahkan node "4" di depan linked list



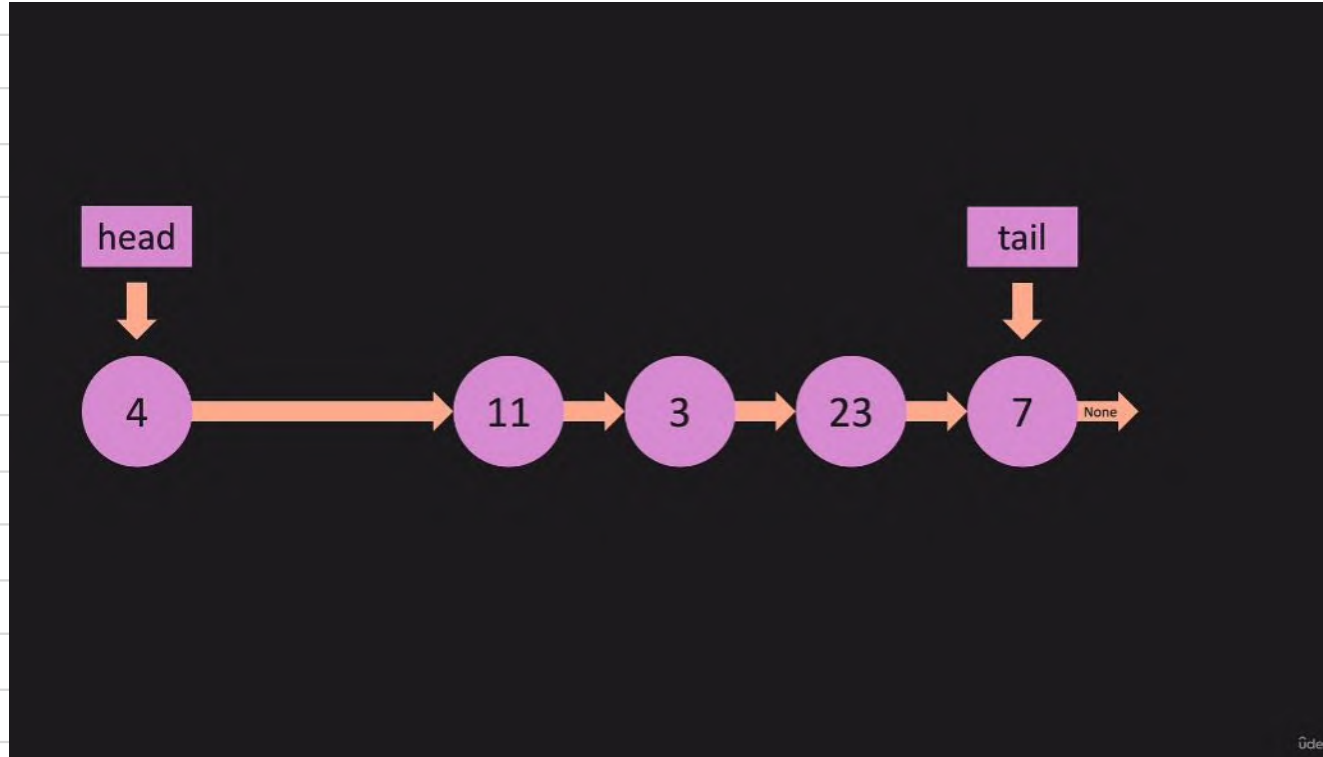
Add & Remove Node di Depan

Bagaimana jika kita ingin menambahkan node "4" di depan linked list



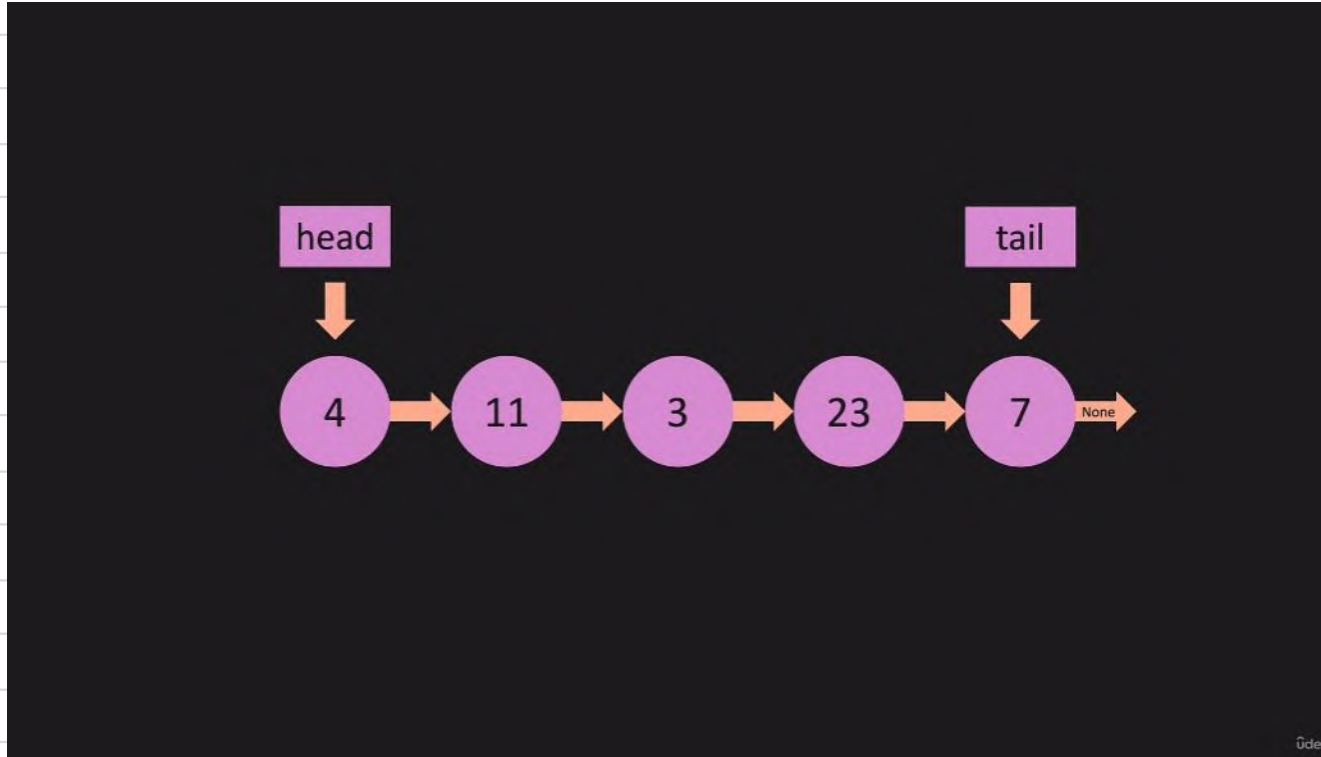
Add & Remove Node di Depan

Bagaimana jika kita ingin menambahkan node "4" di depan linked list



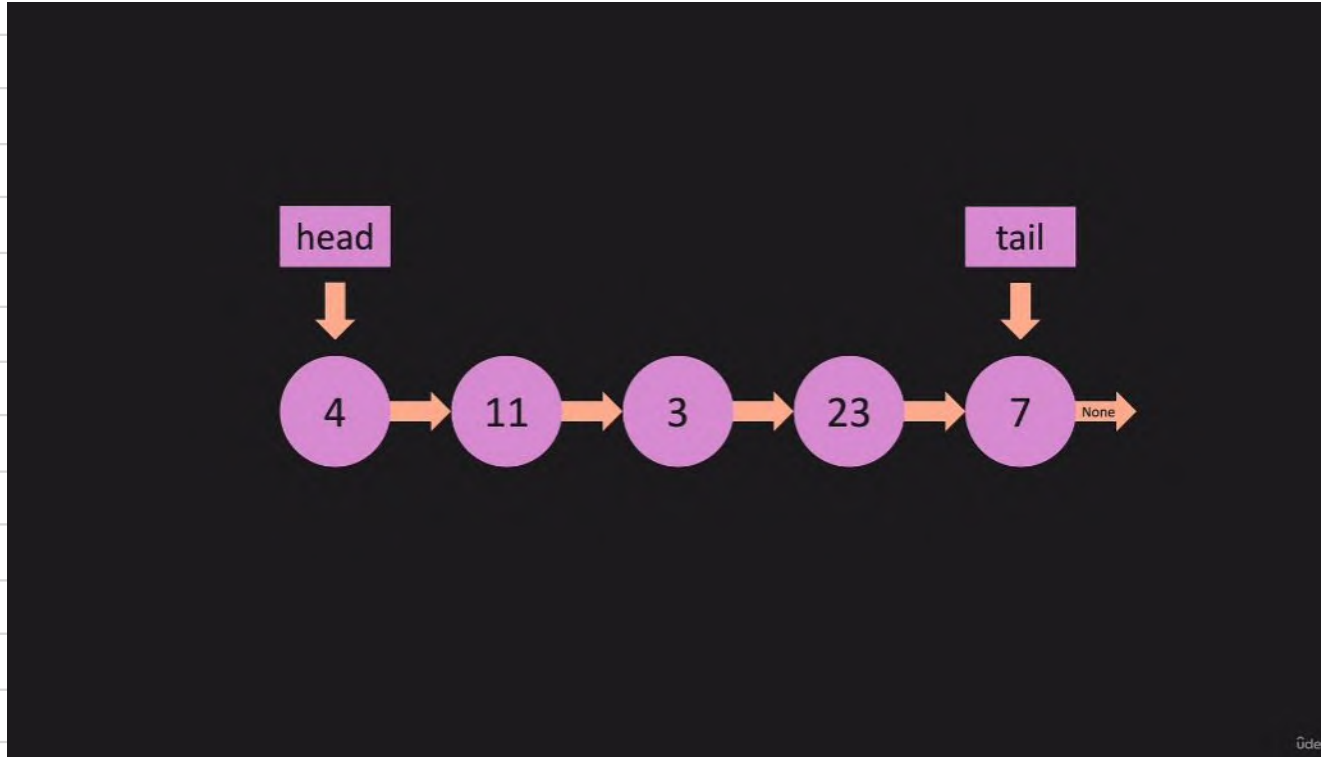
Add & Remove Node di Depan

Bagaimana jika kita ingin menambahkan node "4" di depan linked list



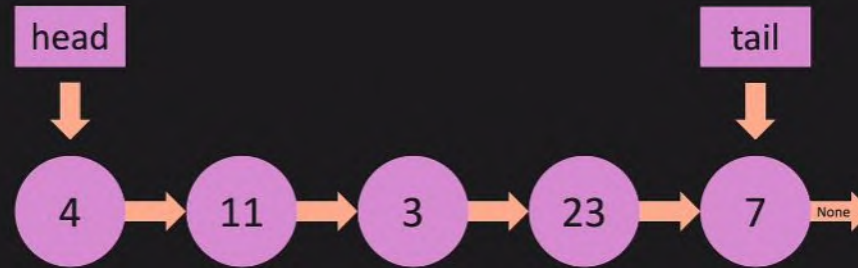
Add & Remove Node di Depan

Apa notasi Big O untuk
Add First ... ?



Add & Remove Node di Depan

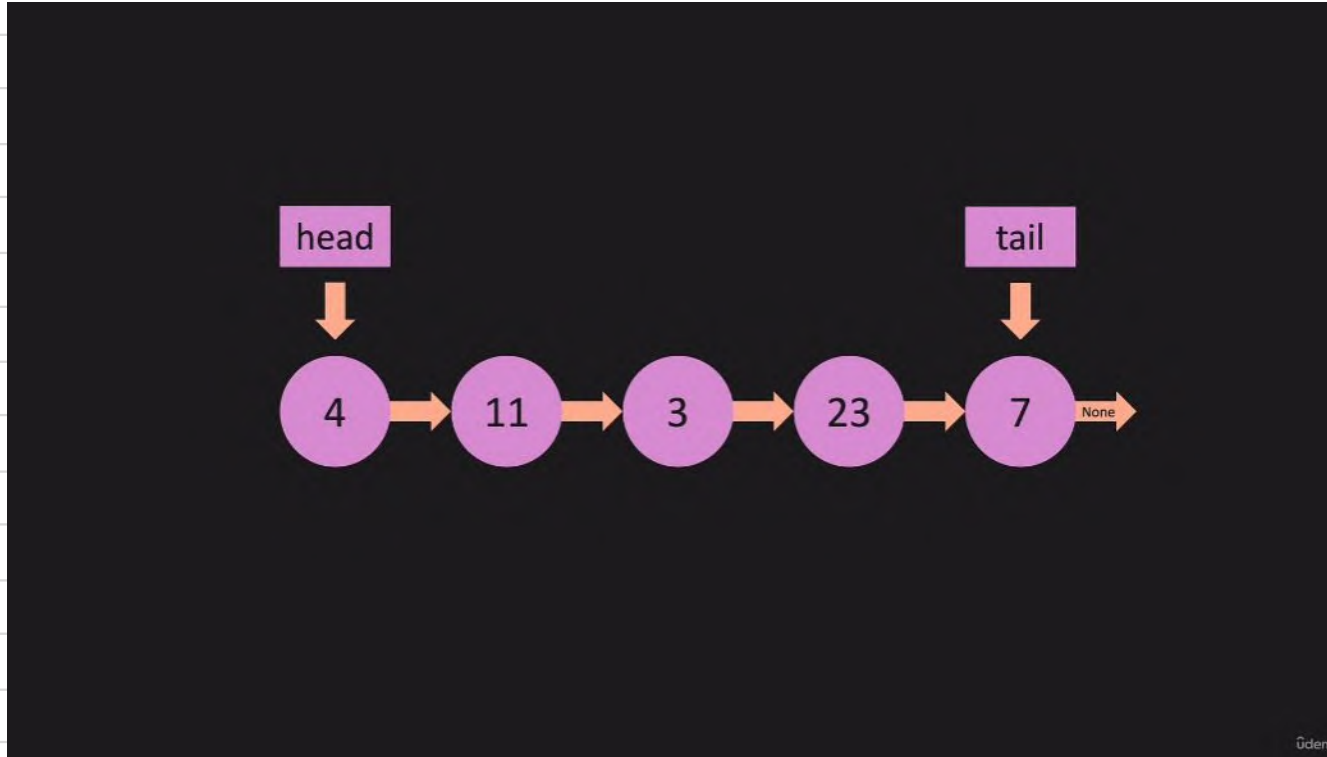
Apa notasi Big O untuk
Add First ... ?



$O(1)$

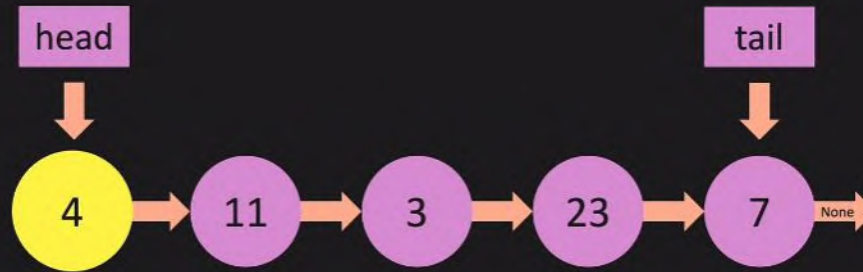
Diketahui suatu Linked List...

Add & Remove Node di Depan



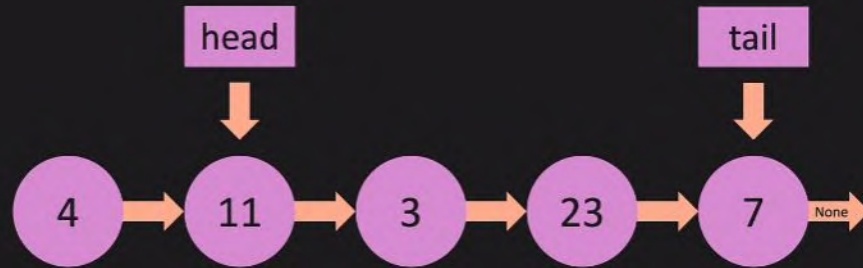
Add & Remove Node di Depan

Bagaimana jika kita ingin menghapus node "4" dari linked list



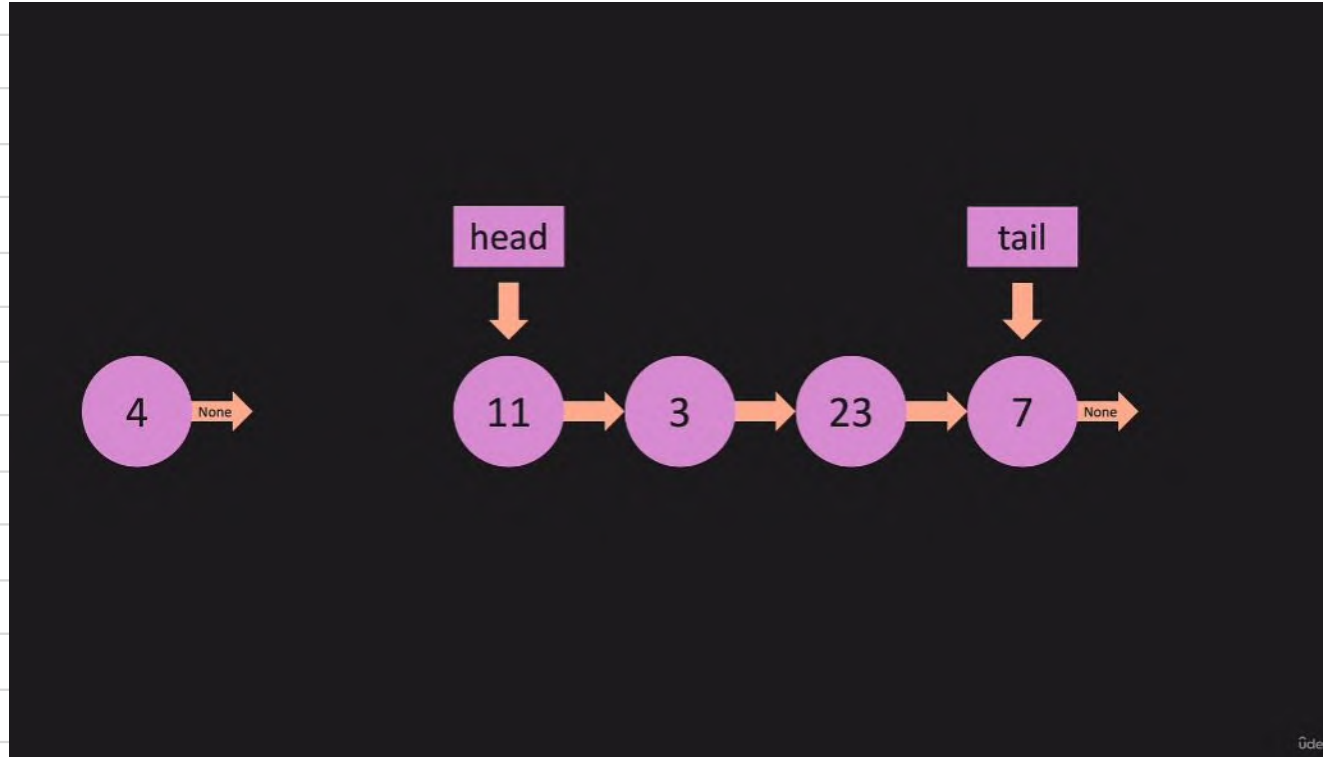
Add & Remove Node di Depan

Bagaimana jika kita ingin menghapus node "4" dari linked list



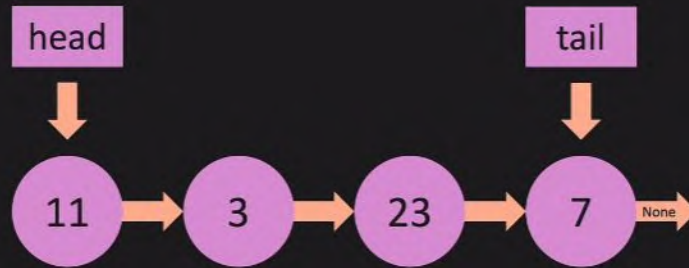
Add & Remove Node di Depan

Bagaimana jika kita ingin menghapus node "4" dari linked list



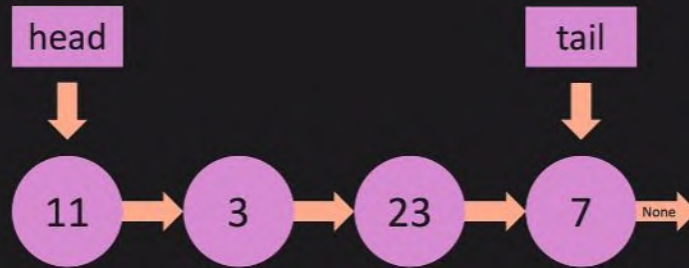
Add & Remove Node di Depan

Apa notasi Big O untuk
Remove First ... ?



Add & Remove Node di Depan

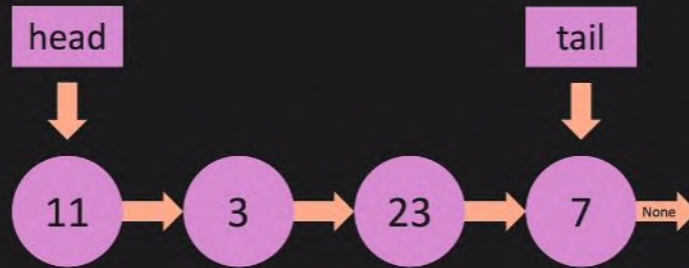
Apa notasi Big O untuk
Remove First ... ?



$O(1)$

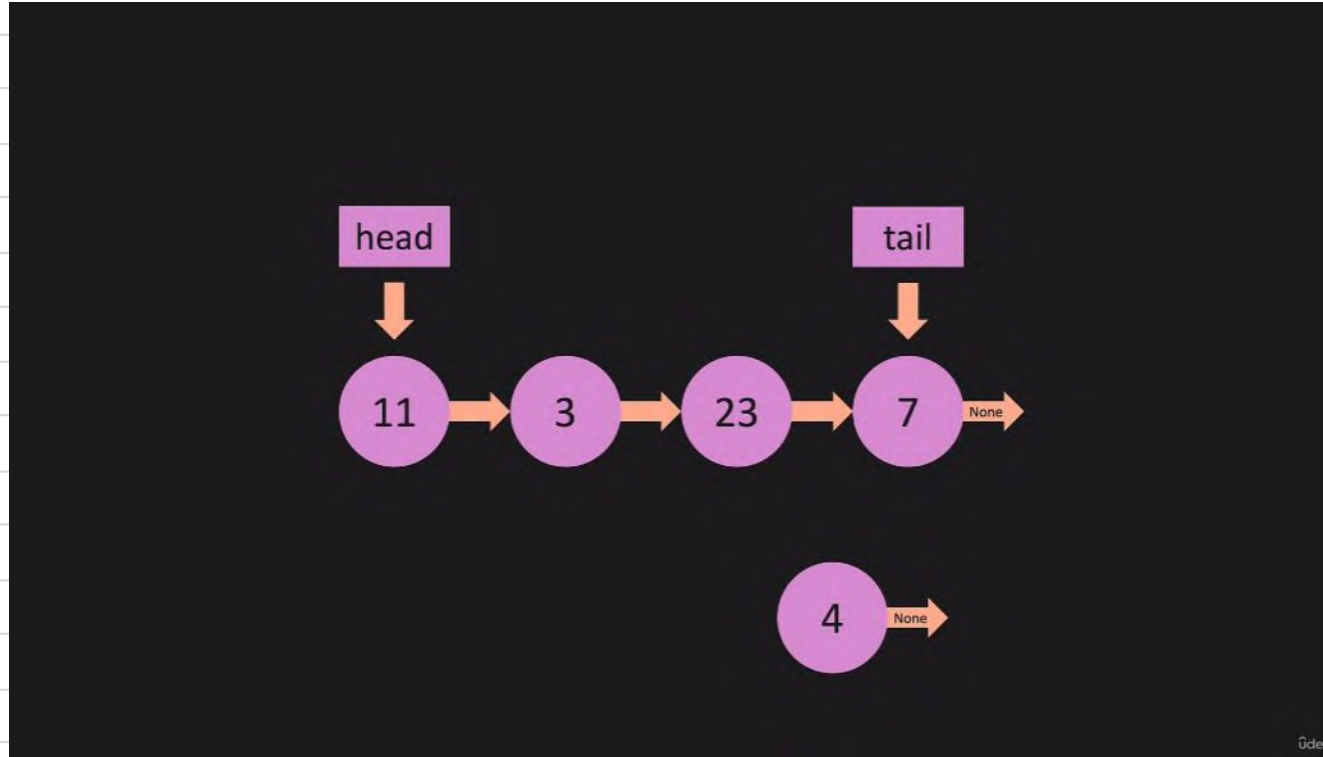
Diketahui suatu Linked List...

Add & Remove Node di Tengah



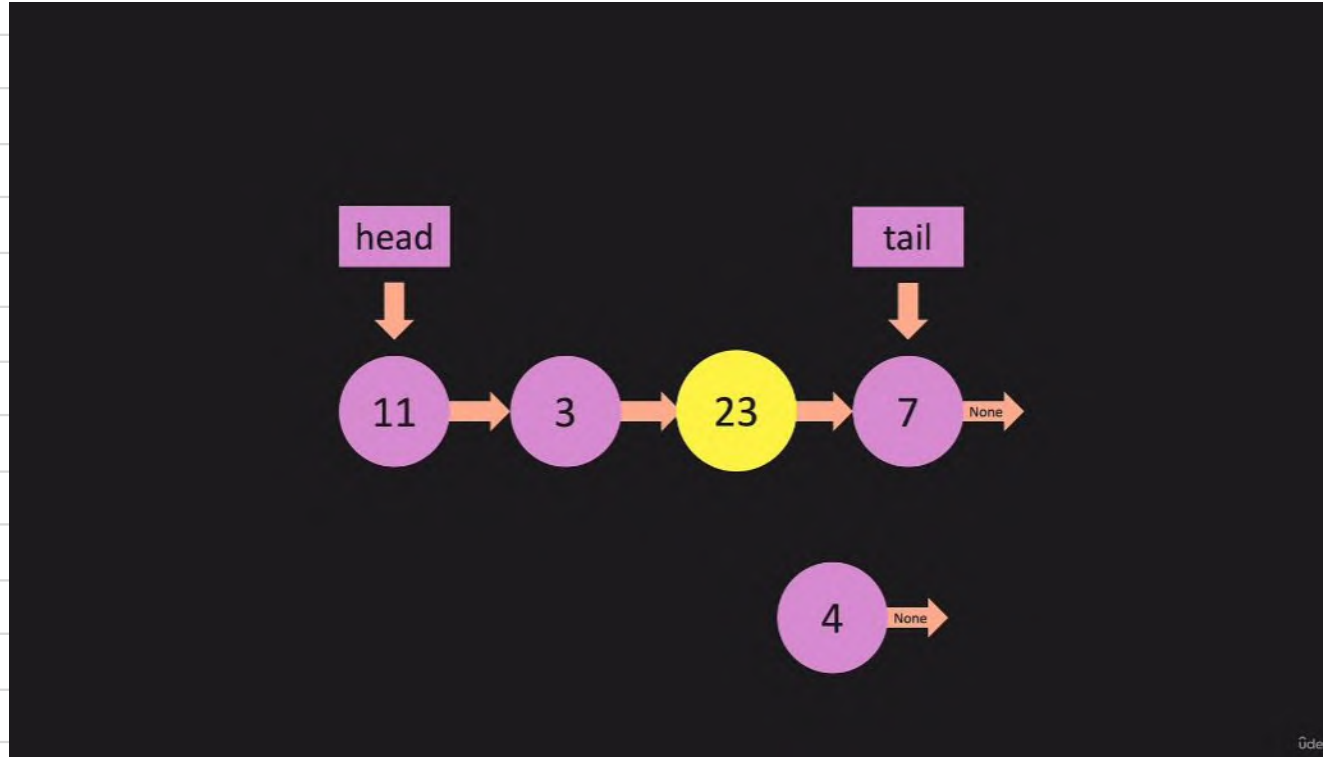
Add & Remove Node di Tengah

Bagaimana jika kita ingin menambahkan node "4" di tengah linked list



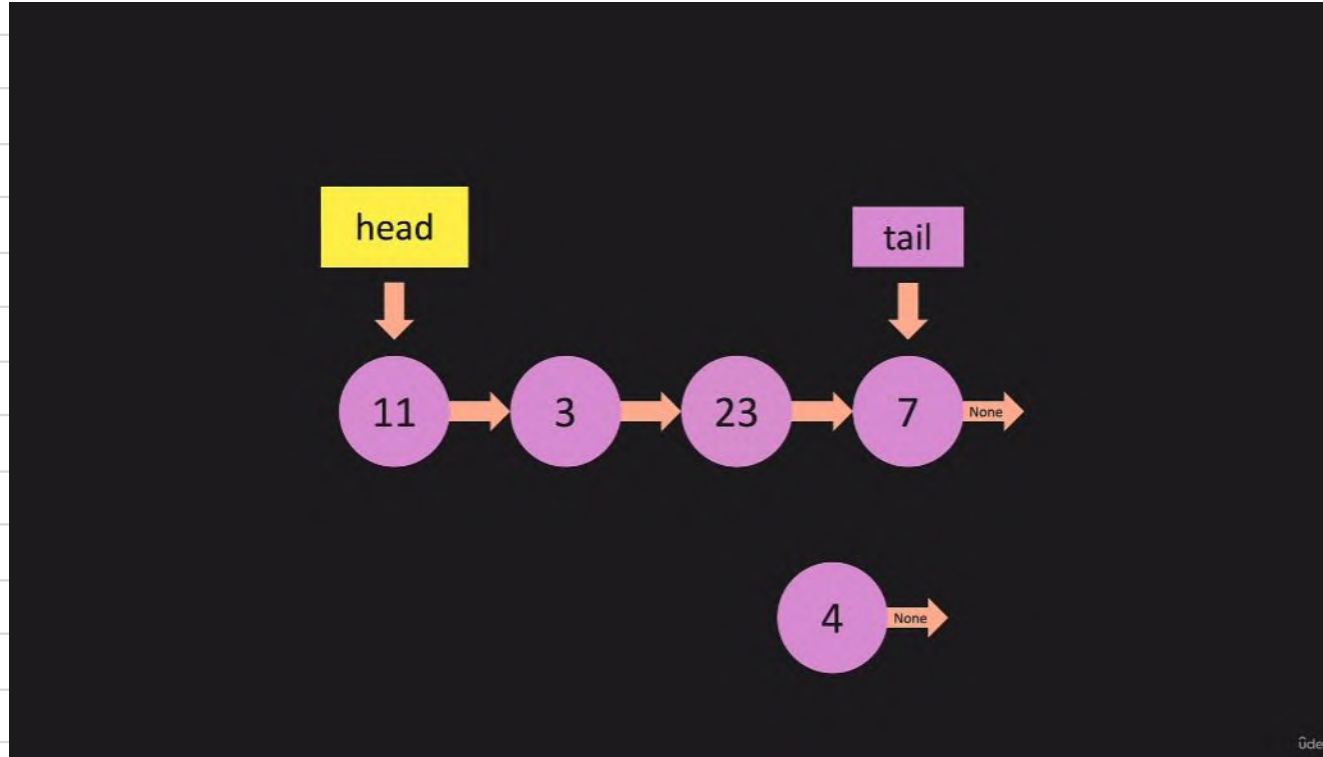
Add & Remove Node di Tengah

Bagaimana jika kita ingin menambahkan node "4" di setelah node "23"



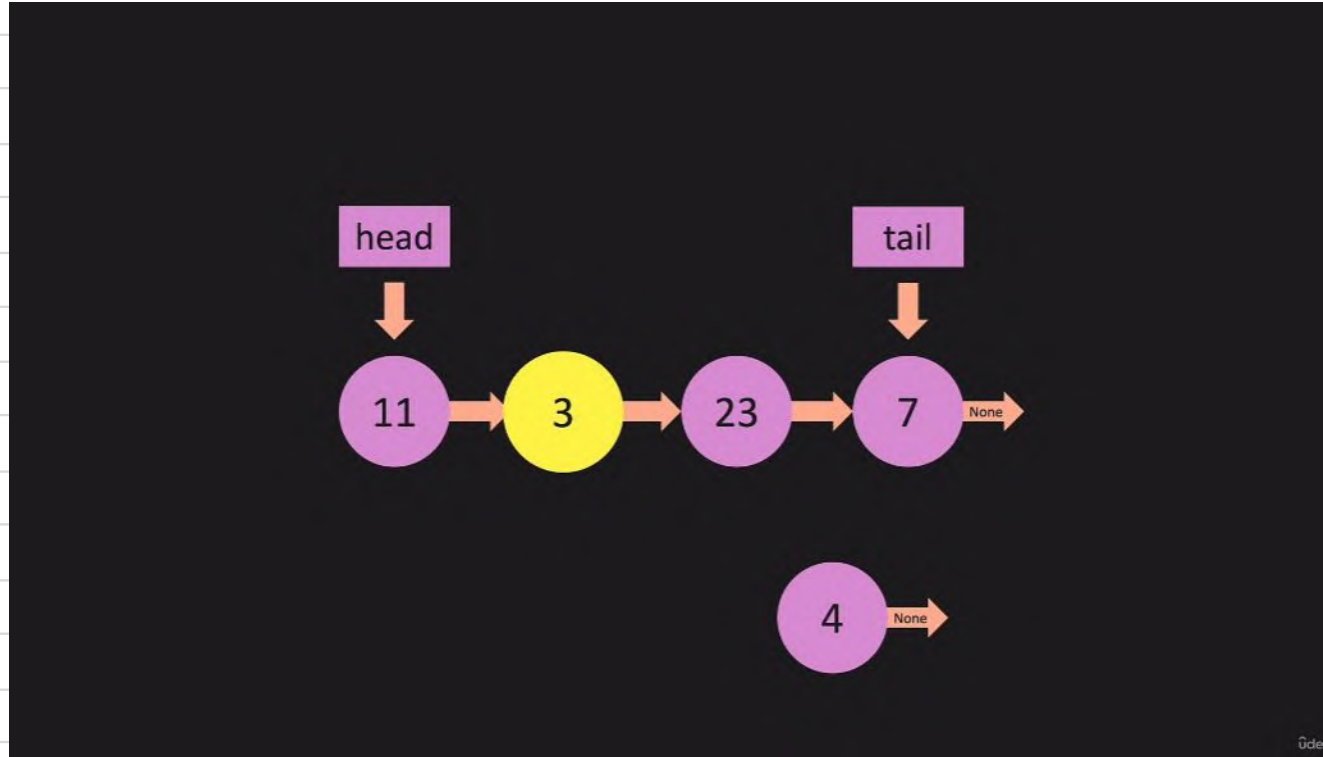
Add & Remove Node di Tengah

Bagaimana jika kita ingin menambahkan node "4" di setelah node "23"



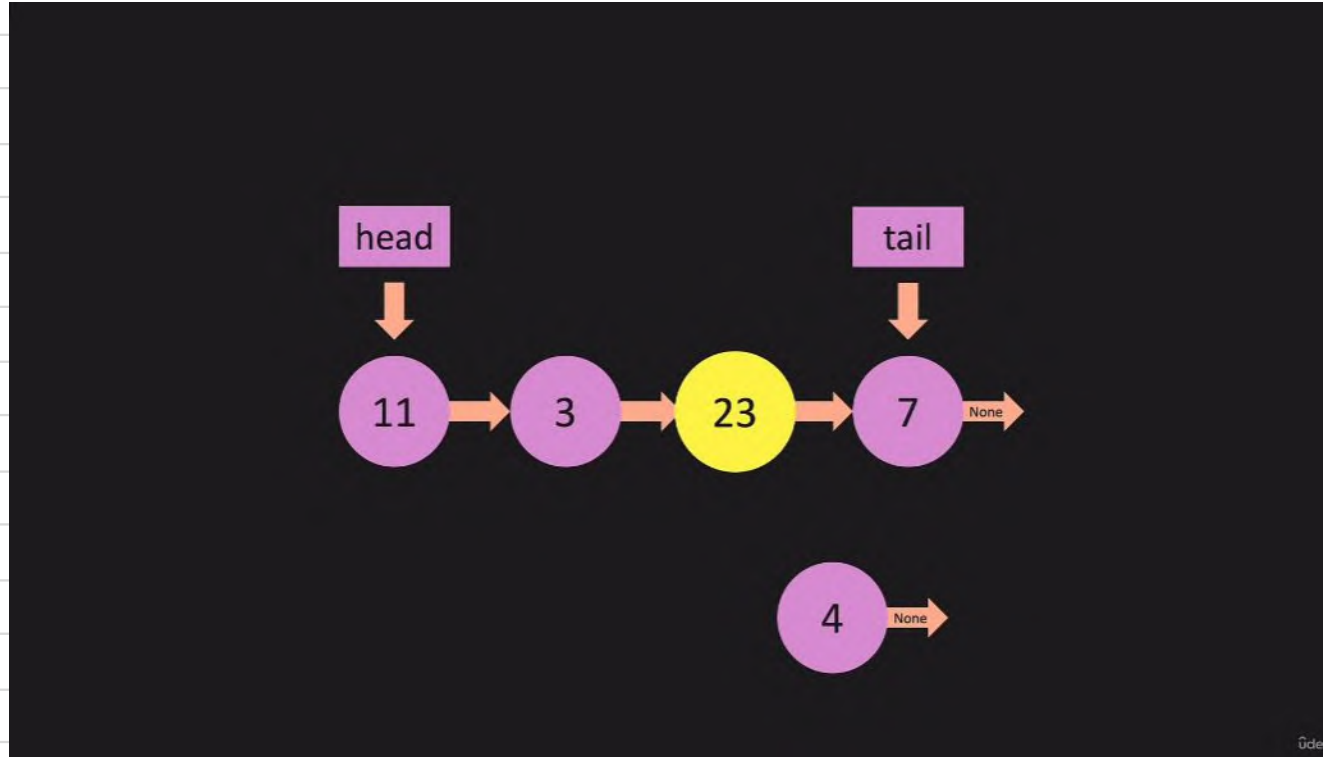
Add & Remove Node di Tengah

Bagaimana jika kita ingin menambahkan node "4" di setelah node "23"



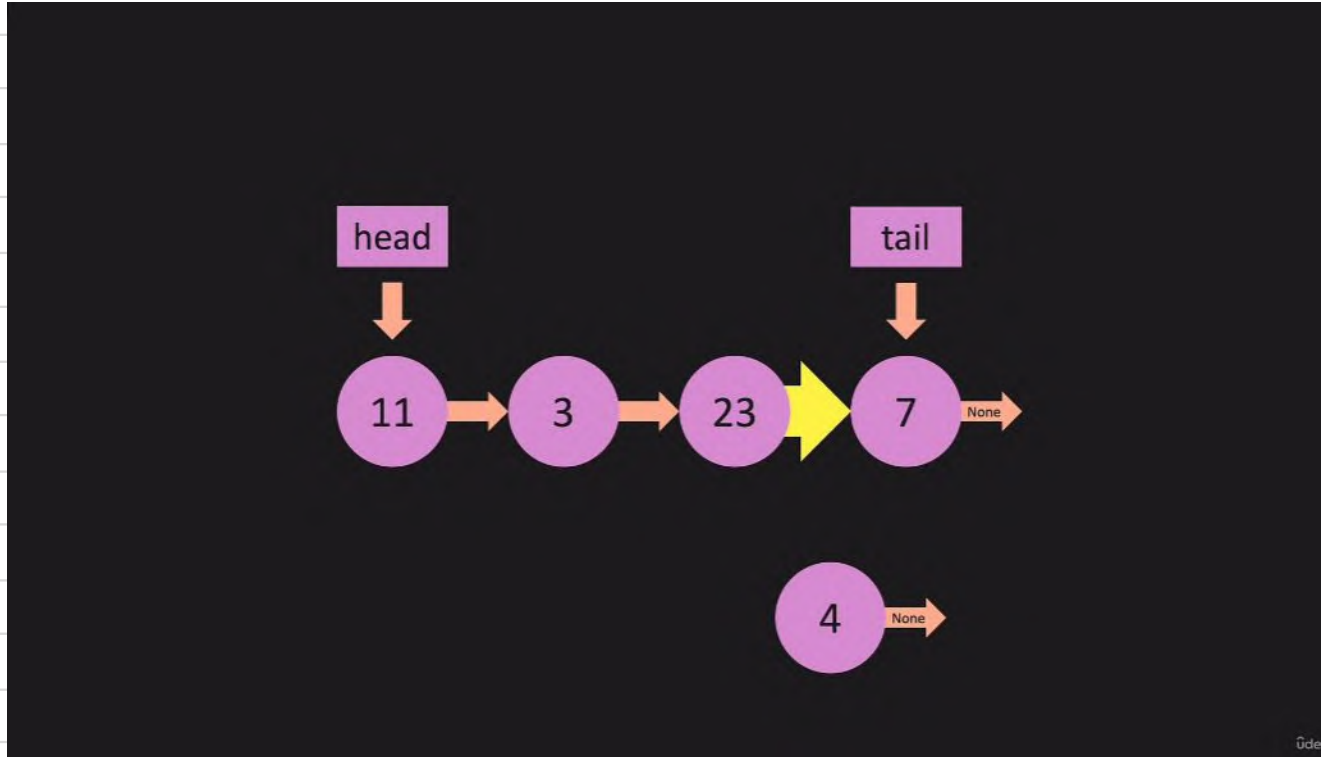
Add & Remove Node di Tengah

Bagaimana jika kita ingin menambahkan node "4" di setelah node "23"



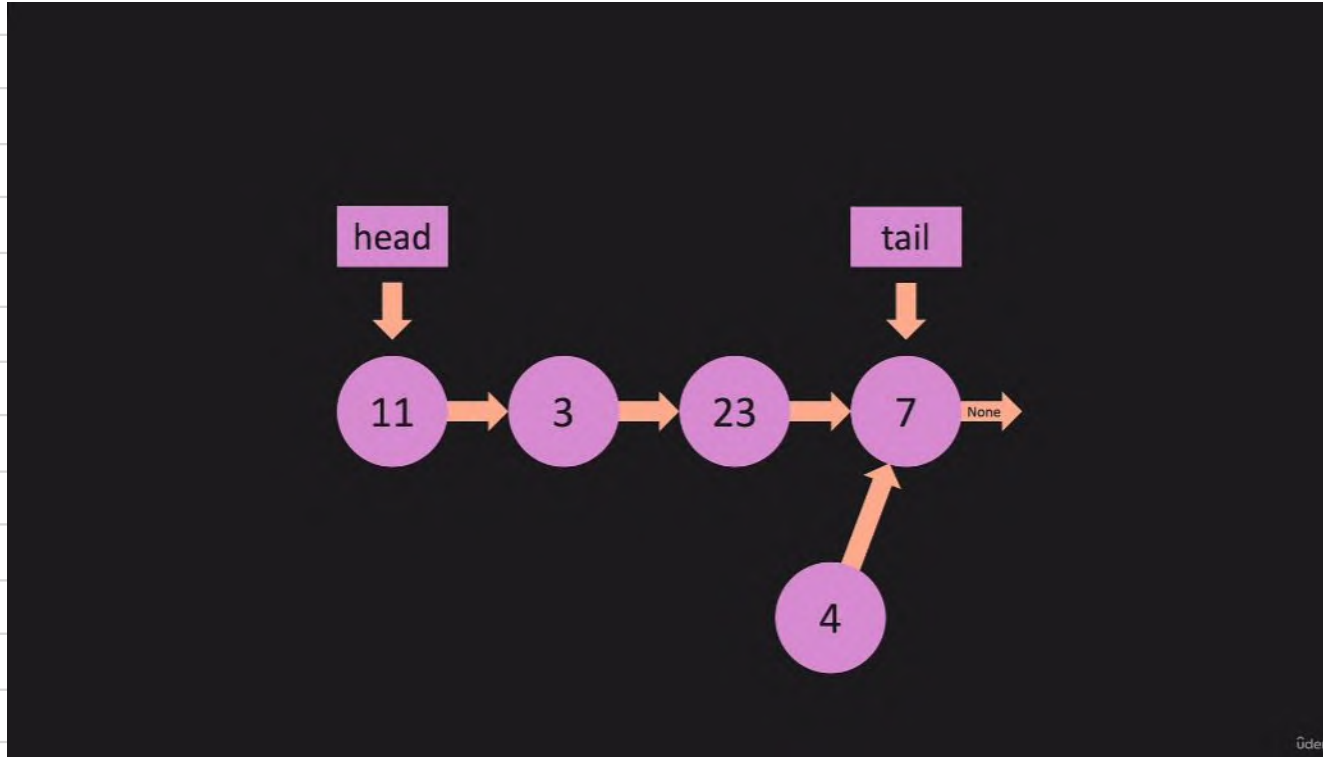
Add & Remove Node di Tengah

Bagaimana jika kita ingin menambahkan node "4" di setelah node "23"



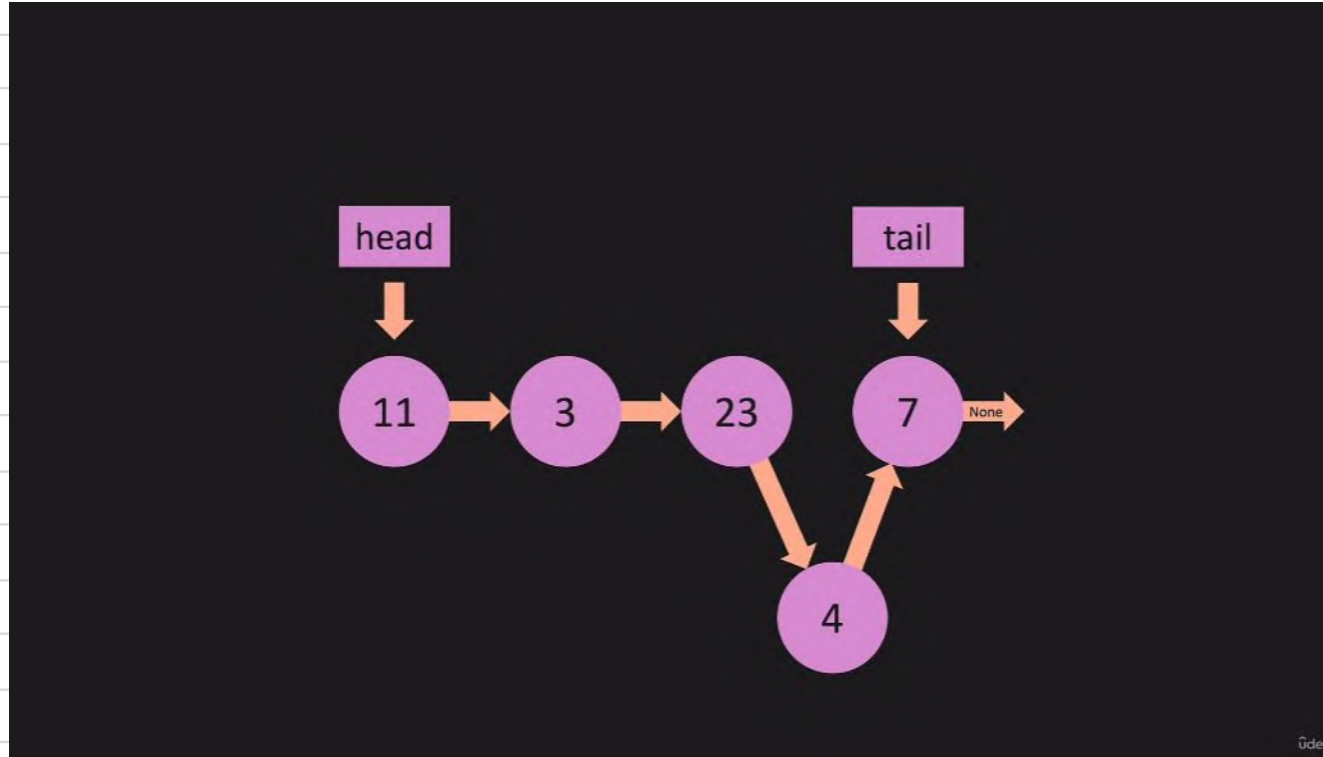
Add & Remove Node di Tengah

Bagaimana jika kita ingin menambahkan node "4" di setelah node "23"



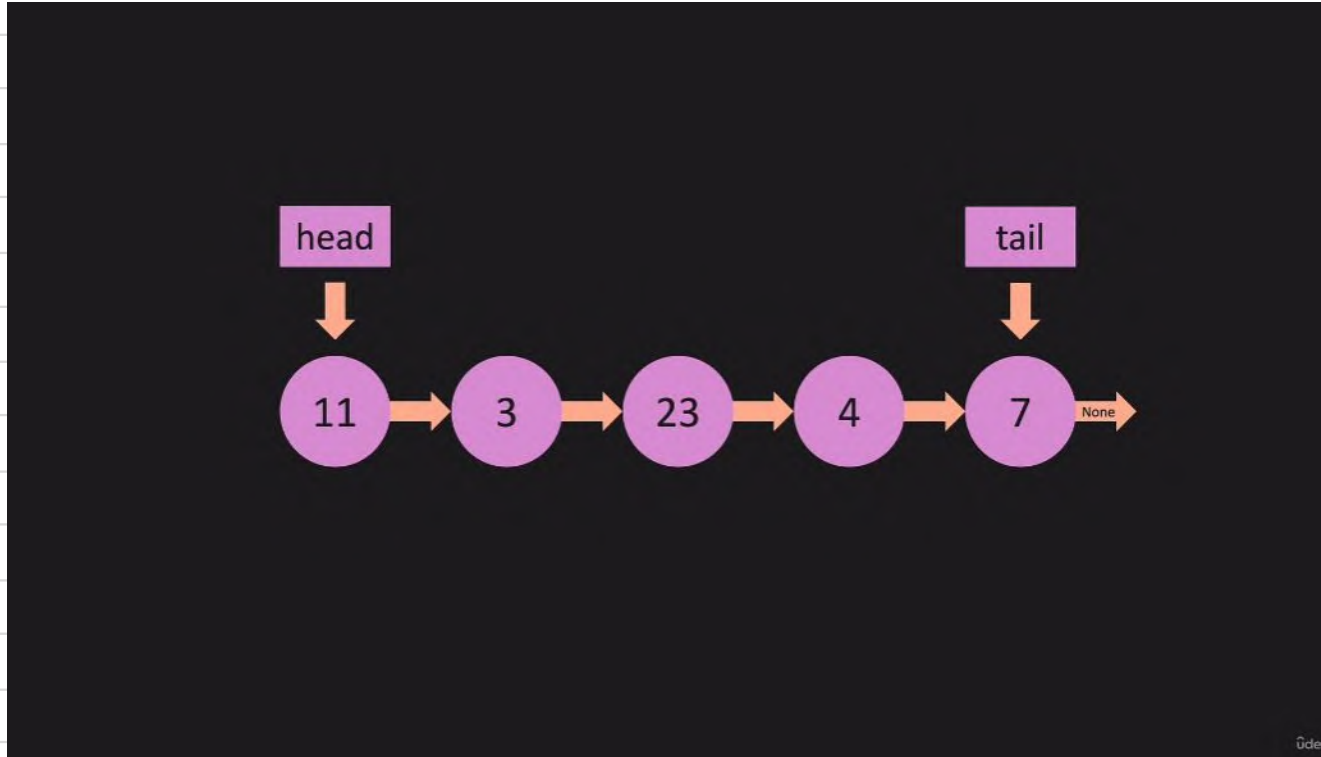
Add & Remove Node di Tengah

Bagaimana jika kita ingin menambahkan node "4" di setelah node "23"



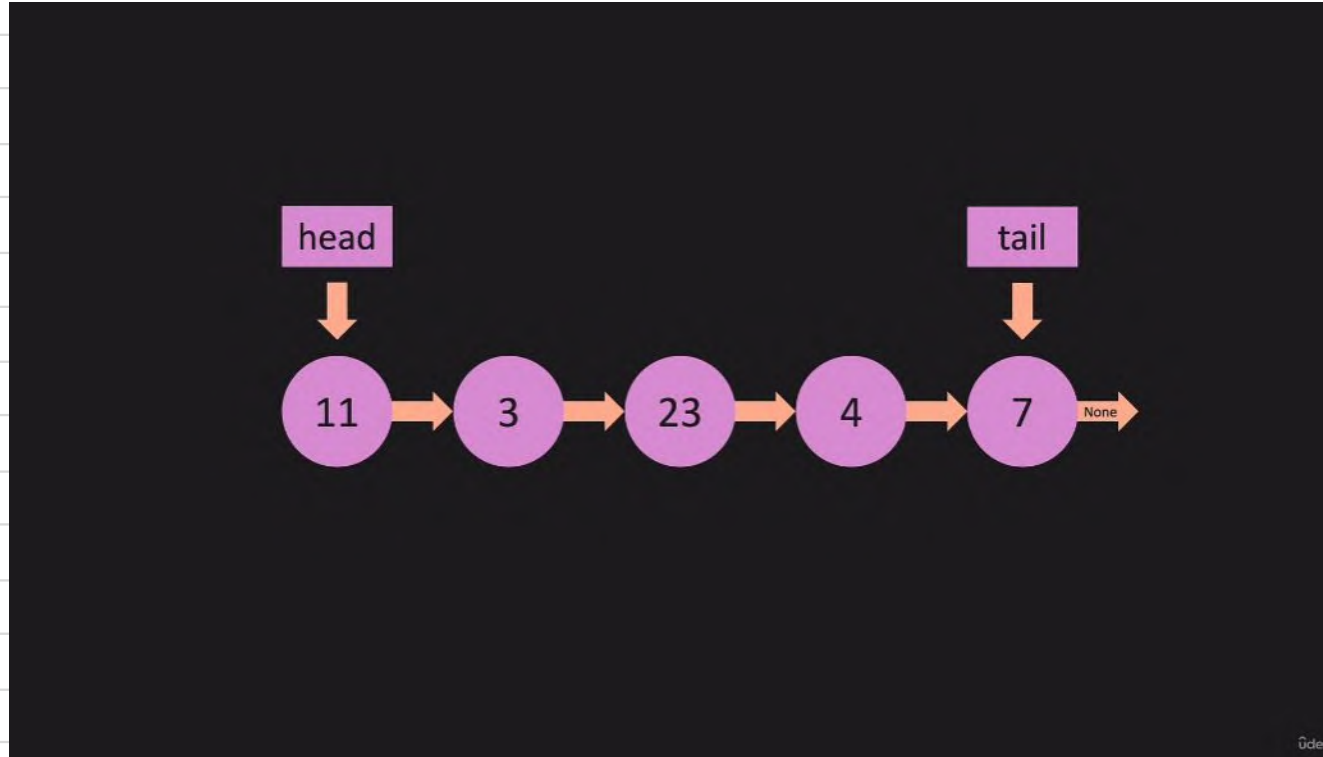
Add & Remove Node di Tengah

Bagaimana jika kita ingin menambahkan node "4" di setelah node "23"



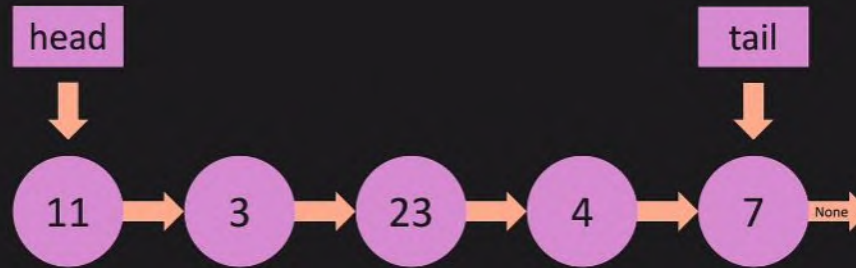
Add & Remove Node di Tengah

Apa notasi Big O untuk
Insert Node ... ?



Add & Remove Node di Tengah

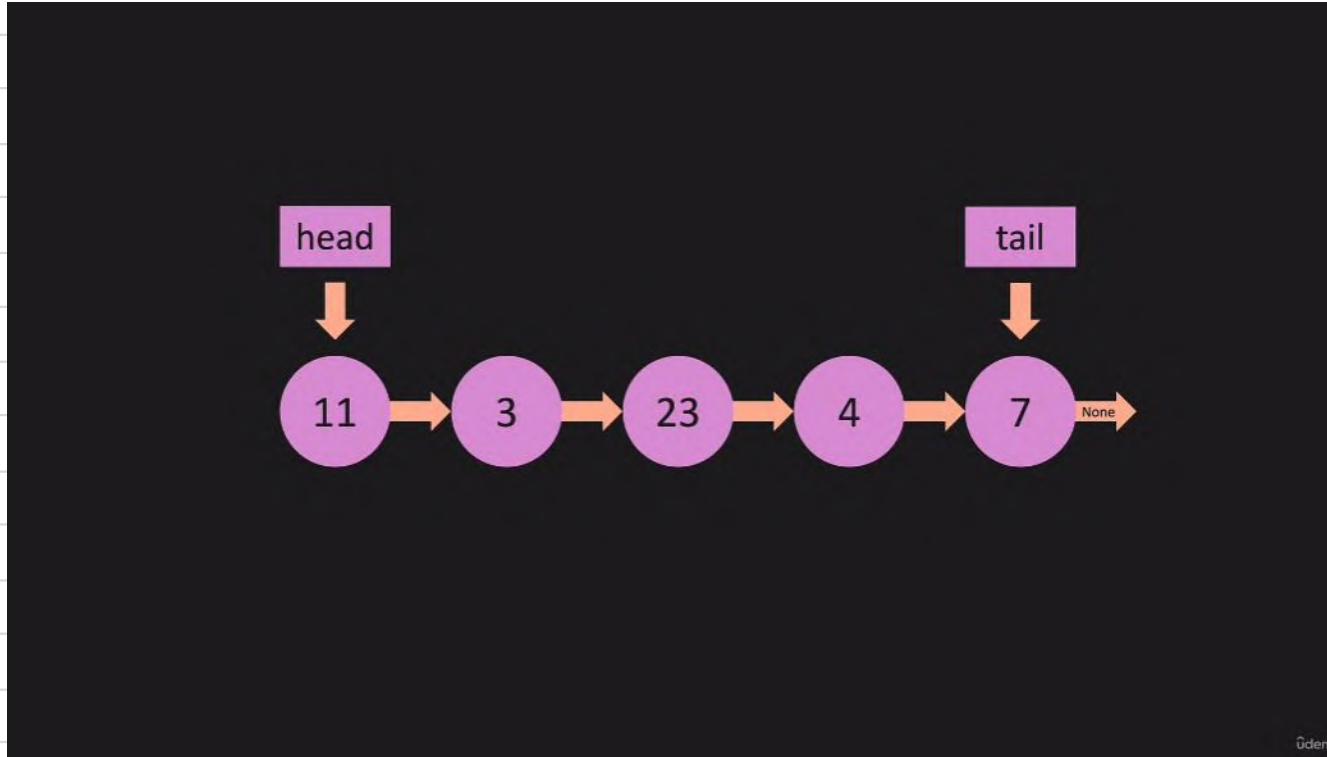
Apa notasi Big O untuk
Insert Node ... ?



$O(n)$

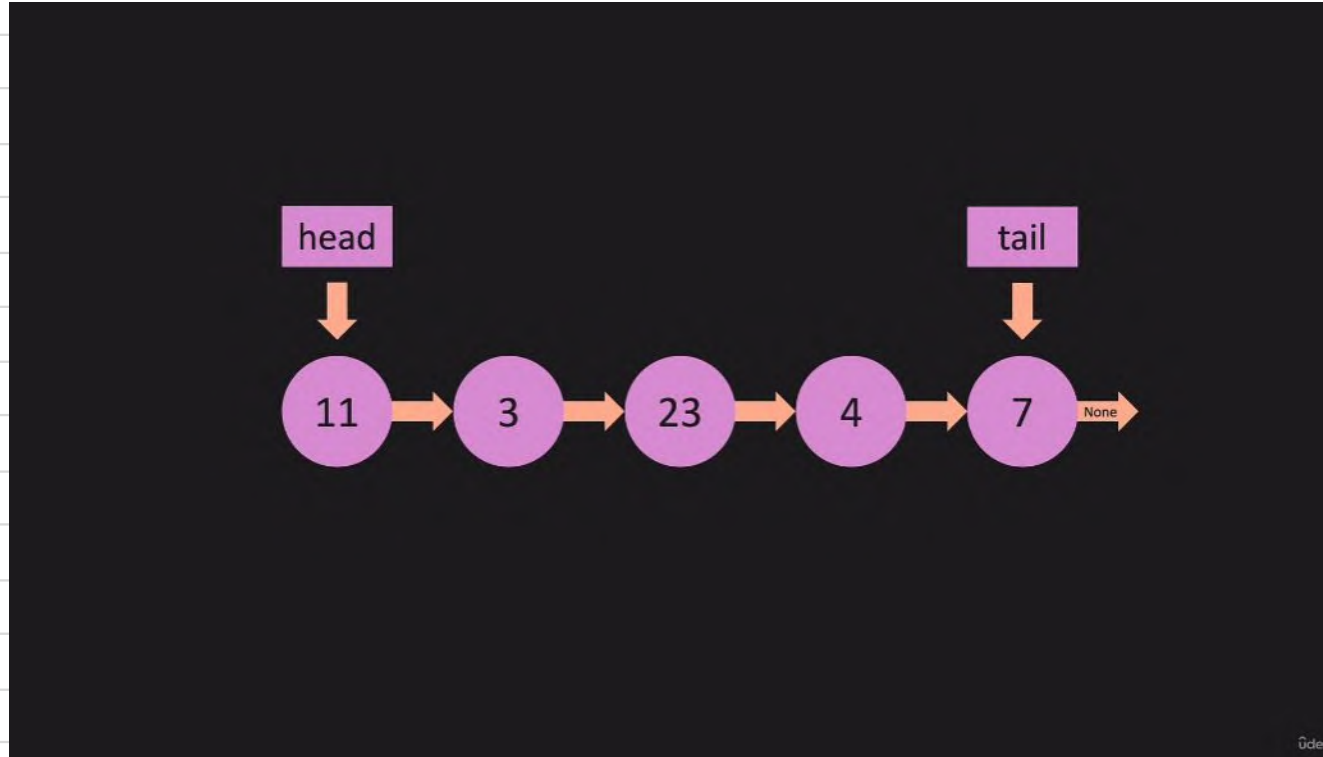
Diketahui suatu Linked List...

Add & Remove Node di Tengah



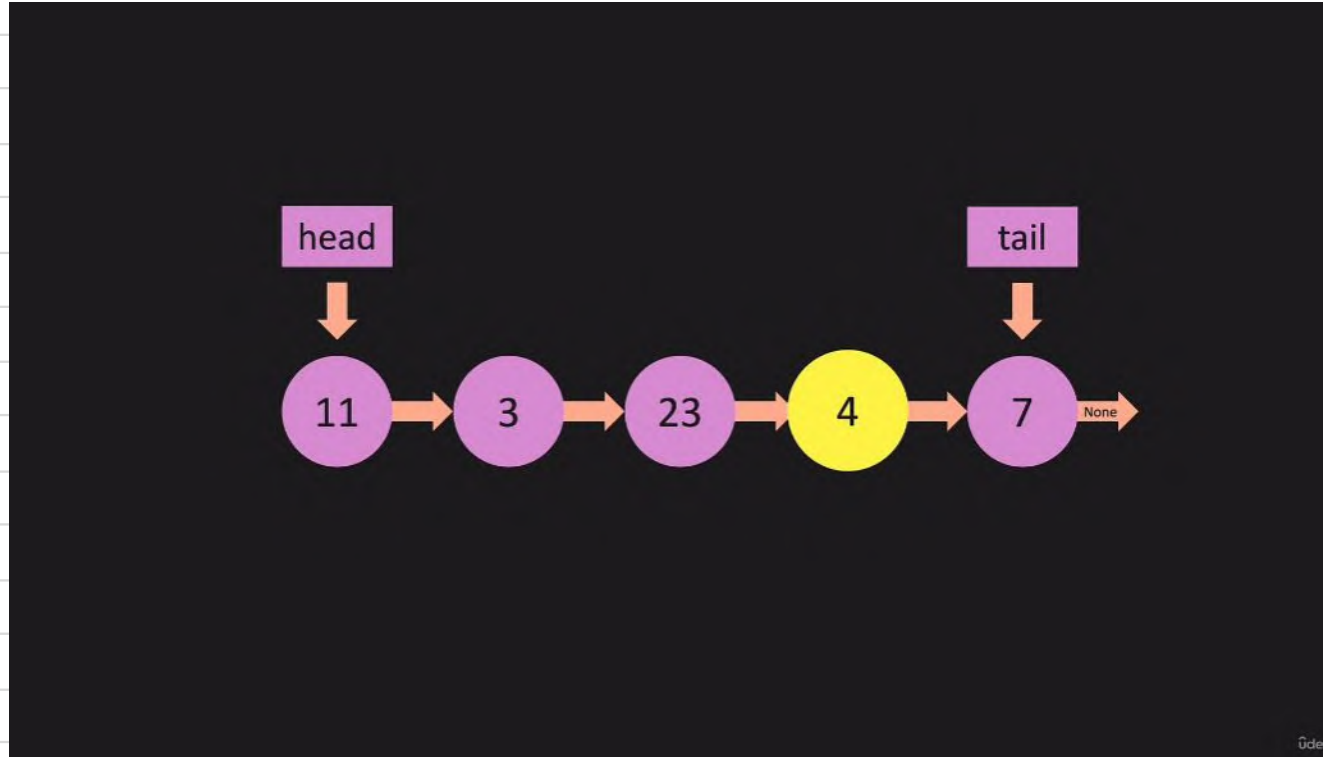
Add & Remove Node di Tengah

Bagaimana jika kita ingin menghapus node "4" dari linked list



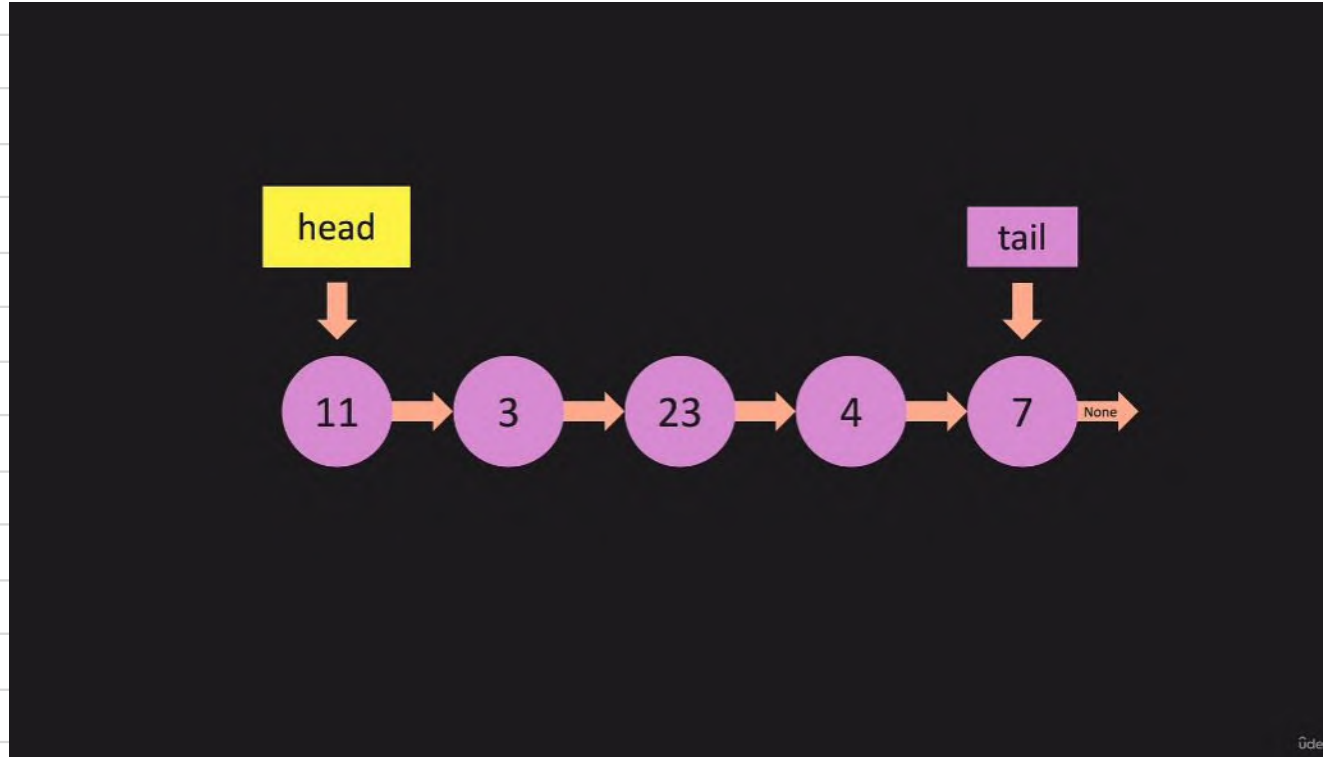
Add & Remove Node di Tengah

Bagaimana jika kita ingin menghapus node "4" dari linked list



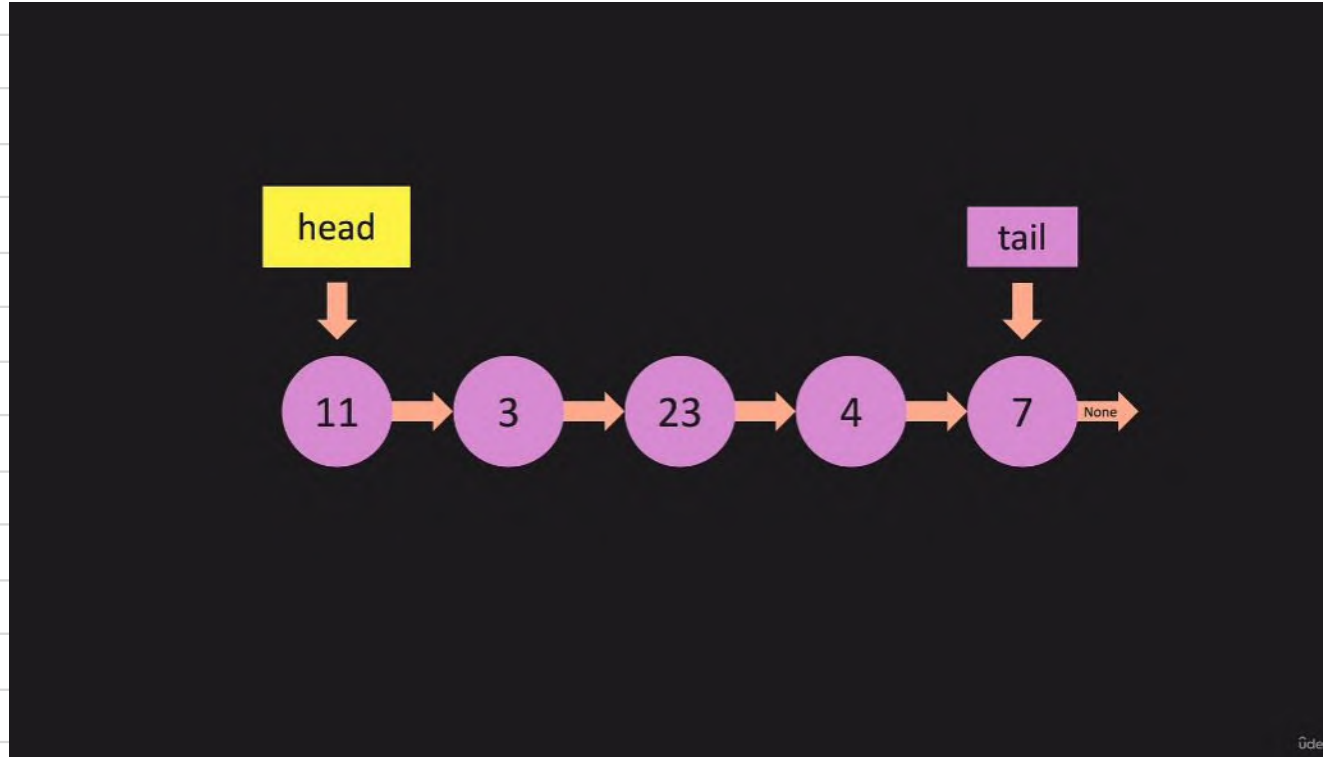
Add & Remove Node di Tengah

Bagaimana jika kita ingin menghapus node "4" dari linked list



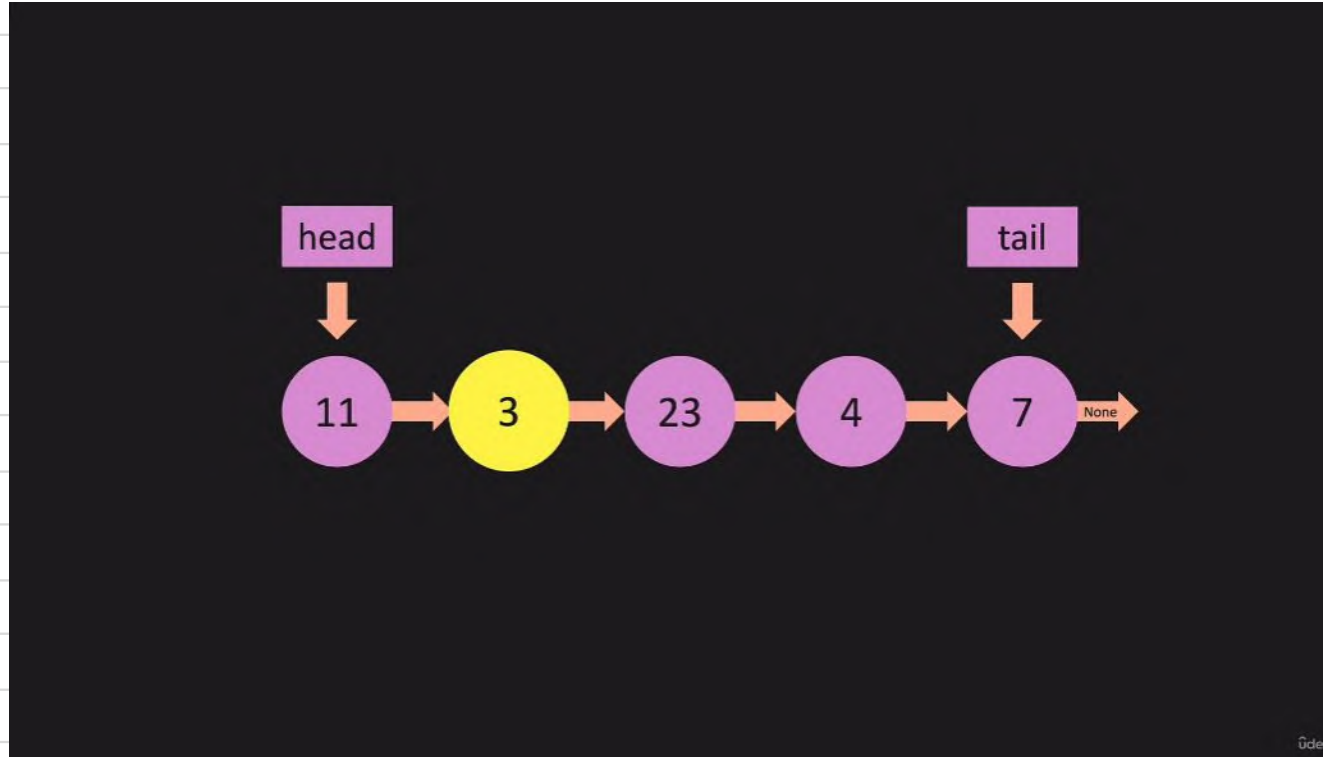
Add & Remove Node di Tengah

Bagaimana jika kita ingin menghapus node "4" dari linked list



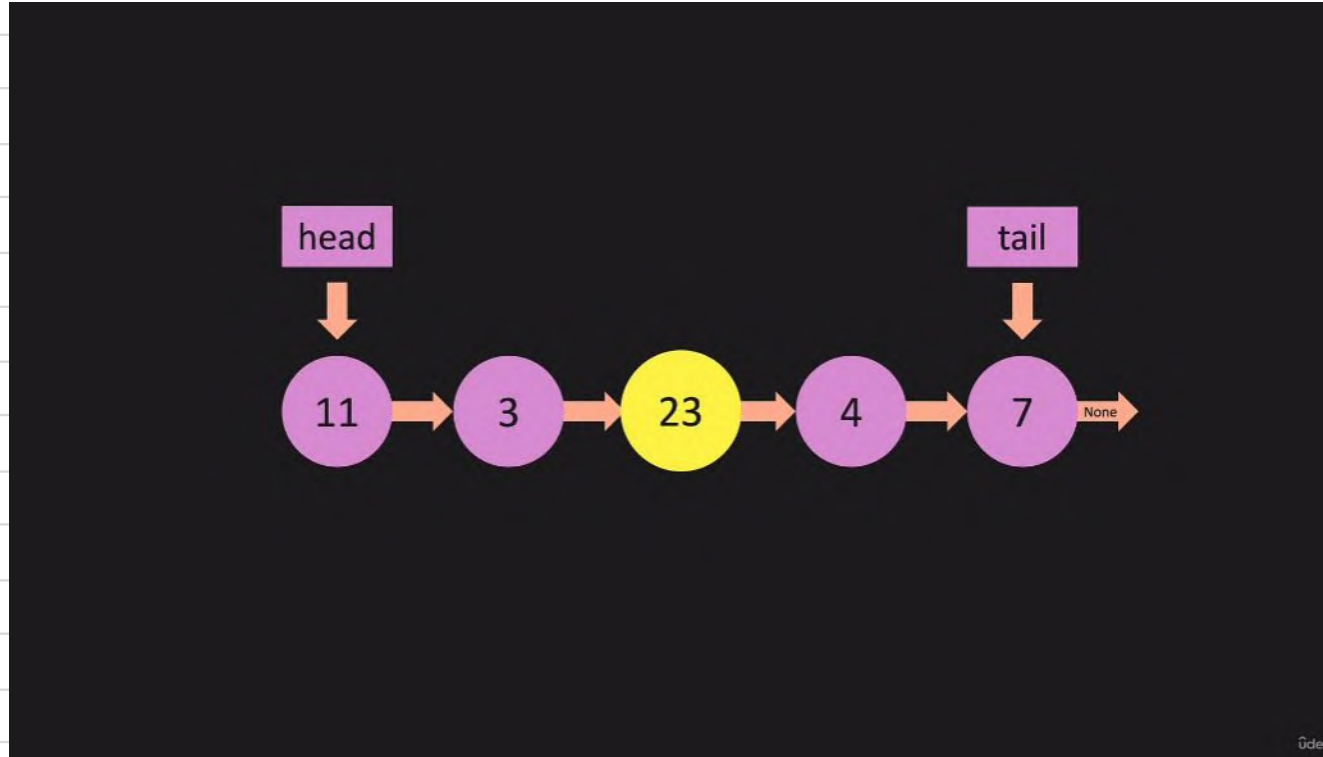
Add & Remove Node di Tengah

Bagaimana jika kita ingin menghapus node "4" dari linked list



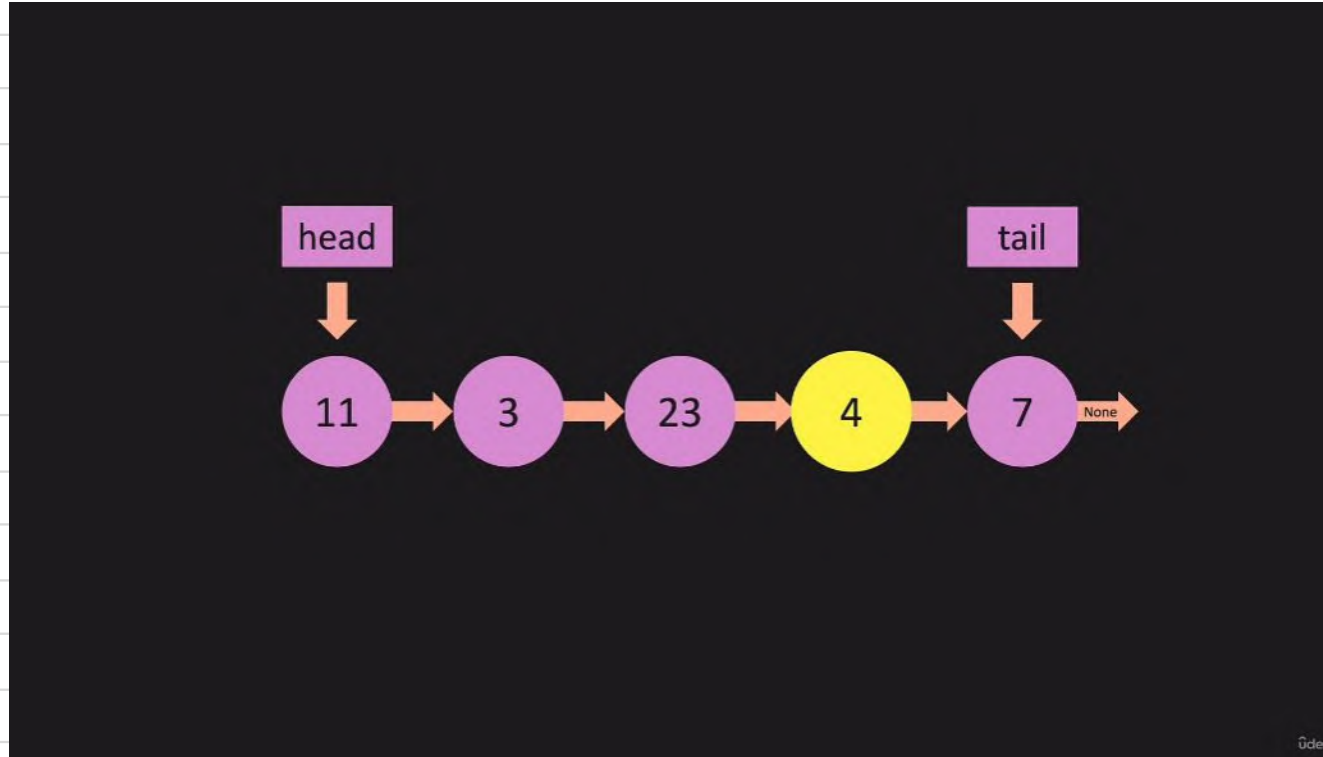
Add & Remove Node di Tengah

Bagaimana jika kita ingin menghapus node "4" dari linked list



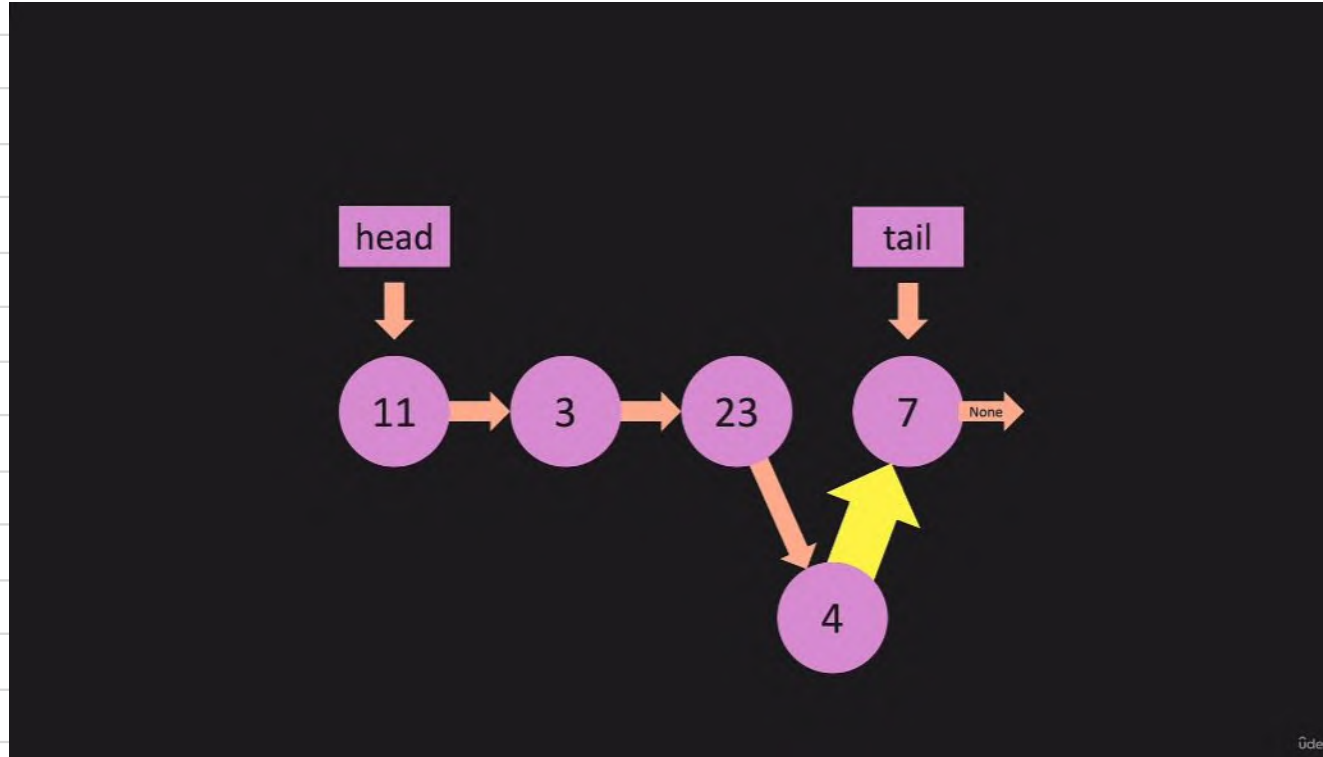
Add & Remove Node di Tengah

Bagaimana jika kita ingin menghapus node "4" dari linked list



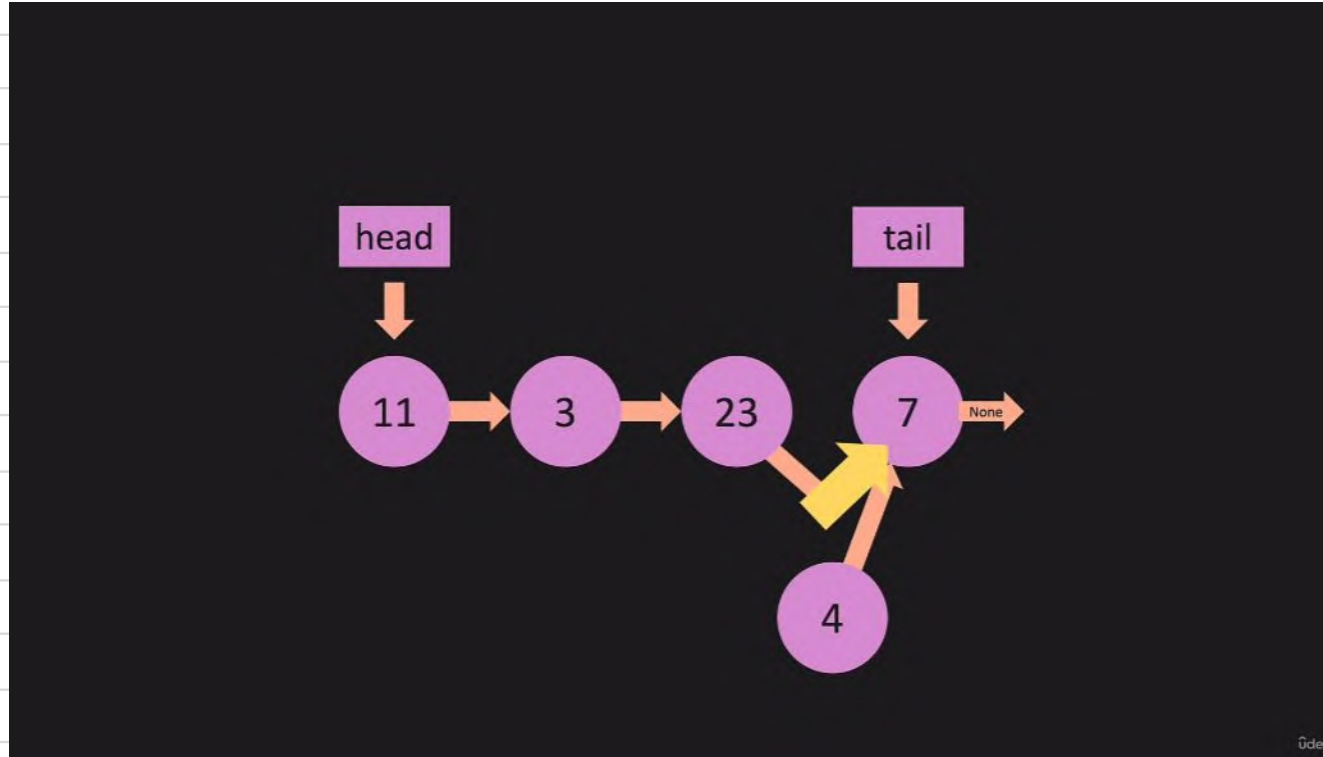
Add & Remove Node di Tengah

Bagaimana jika kita ingin menghapus node "4" dari linked list



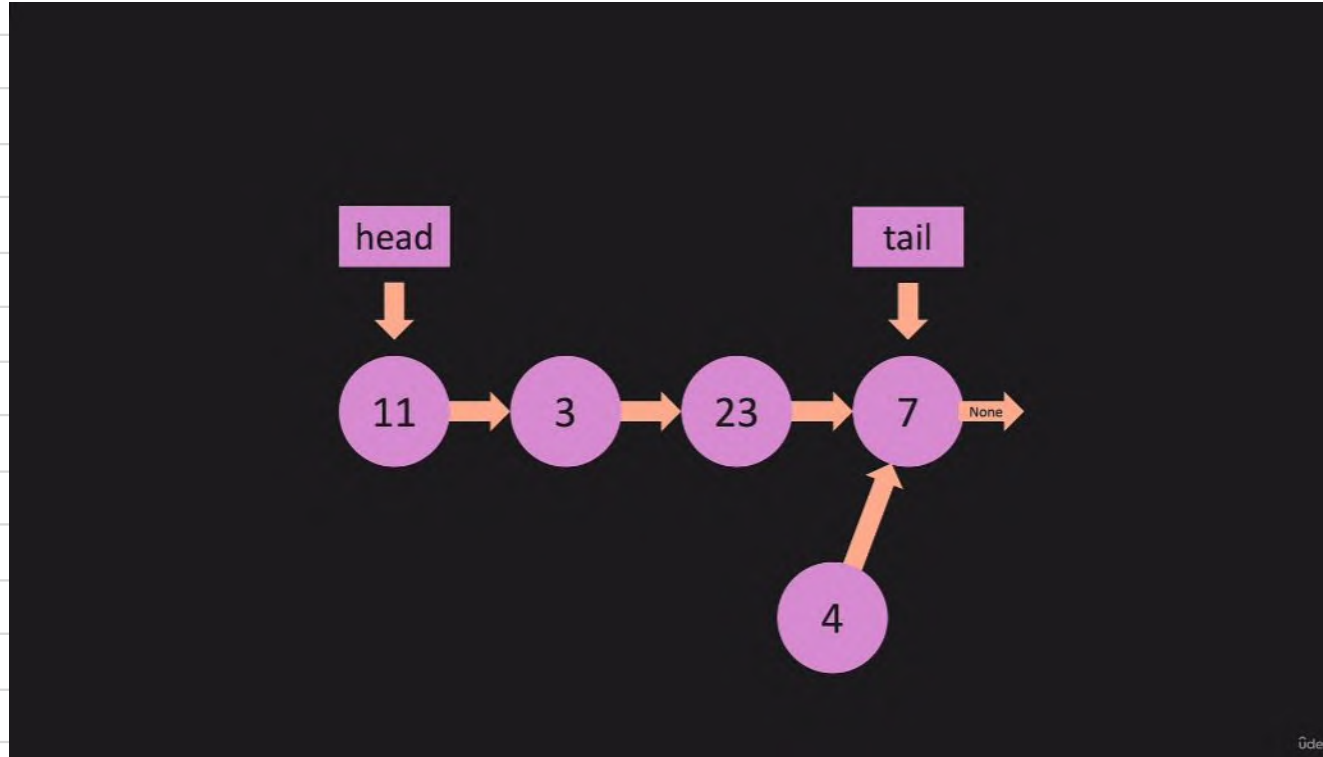
Add & Remove Node di Tengah

Bagaimana jika kita ingin menghapus node "4" dari linked list



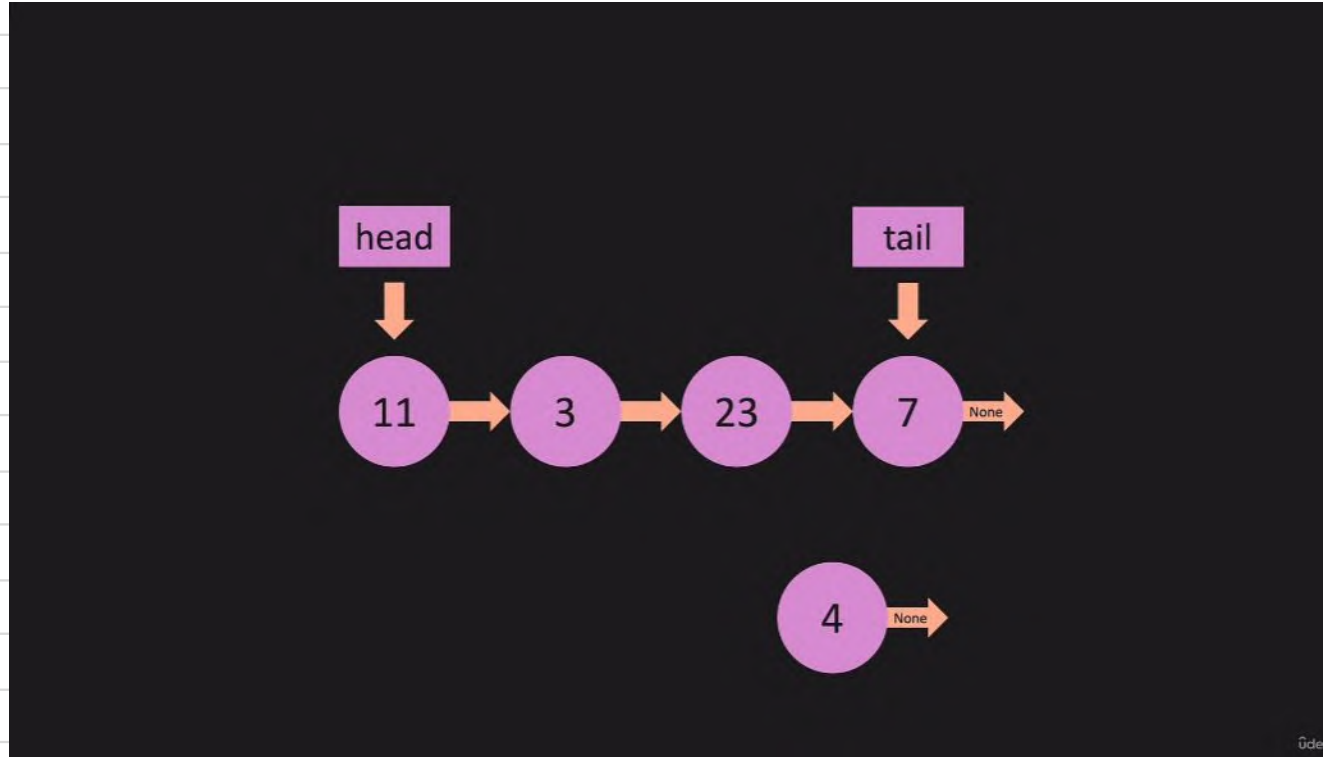
Add & Remove Node di Tengah

Bagaimana jika kita ingin menghapus node "4" dari linked list



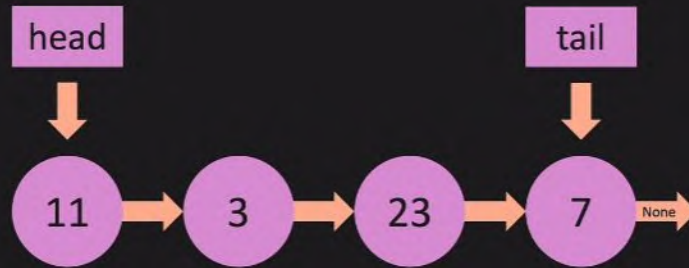
Add & Remove Node di Tengah

Bagaimana jika kita ingin menghapus node "4" dari linked list



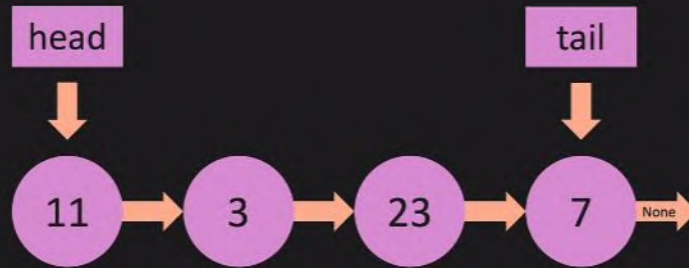
Add & Remove Node di Tengah

Bagaimana jika kita ingin menghapus node "4" dari linked list



Add & Remove Node di Tengah

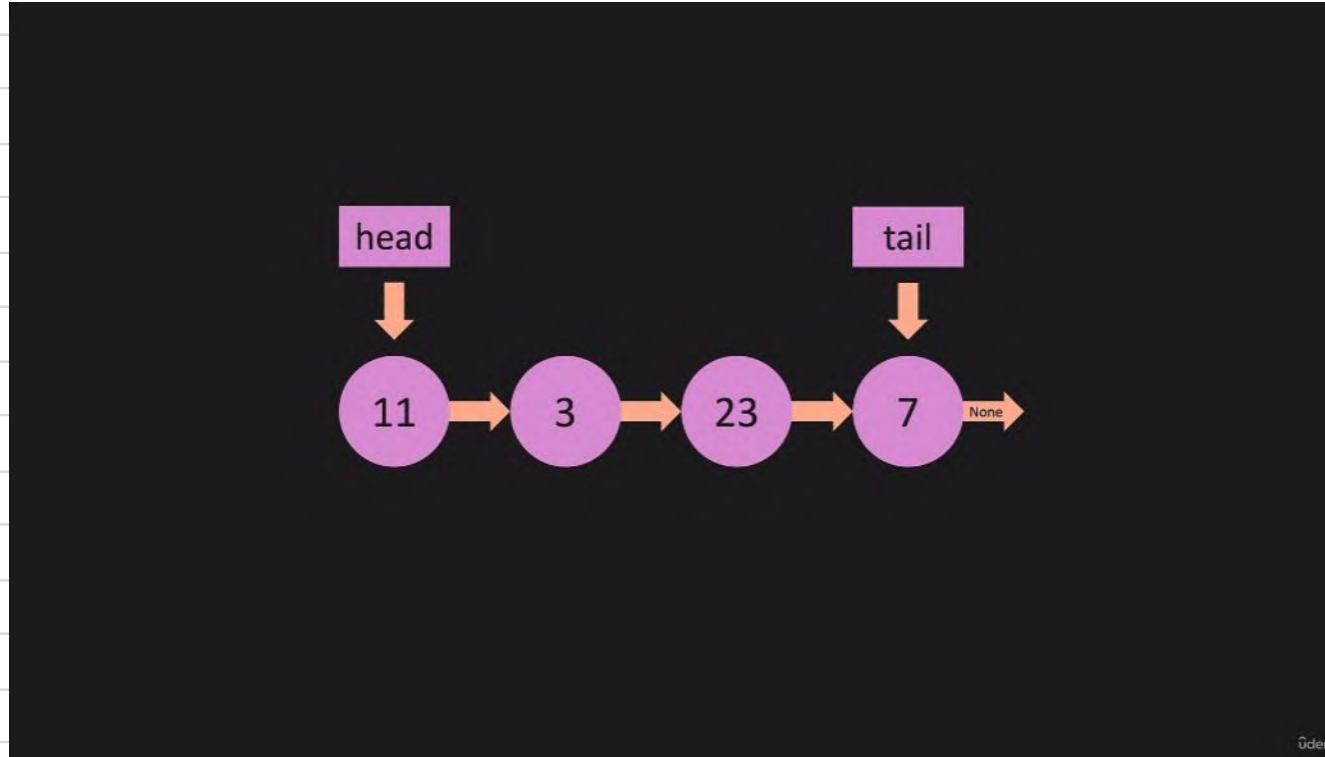
Apa notasi Big O untuk
Remove Node di Tengah... ?



$O(n)$

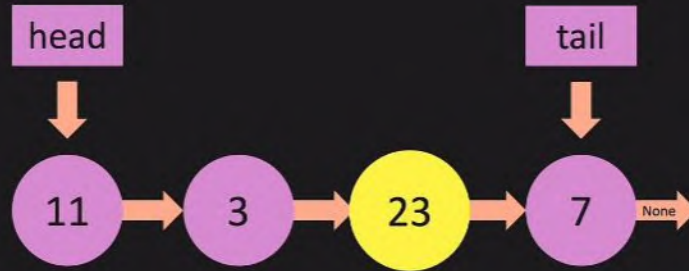
Diketahui suatu Linked List...

Search Node



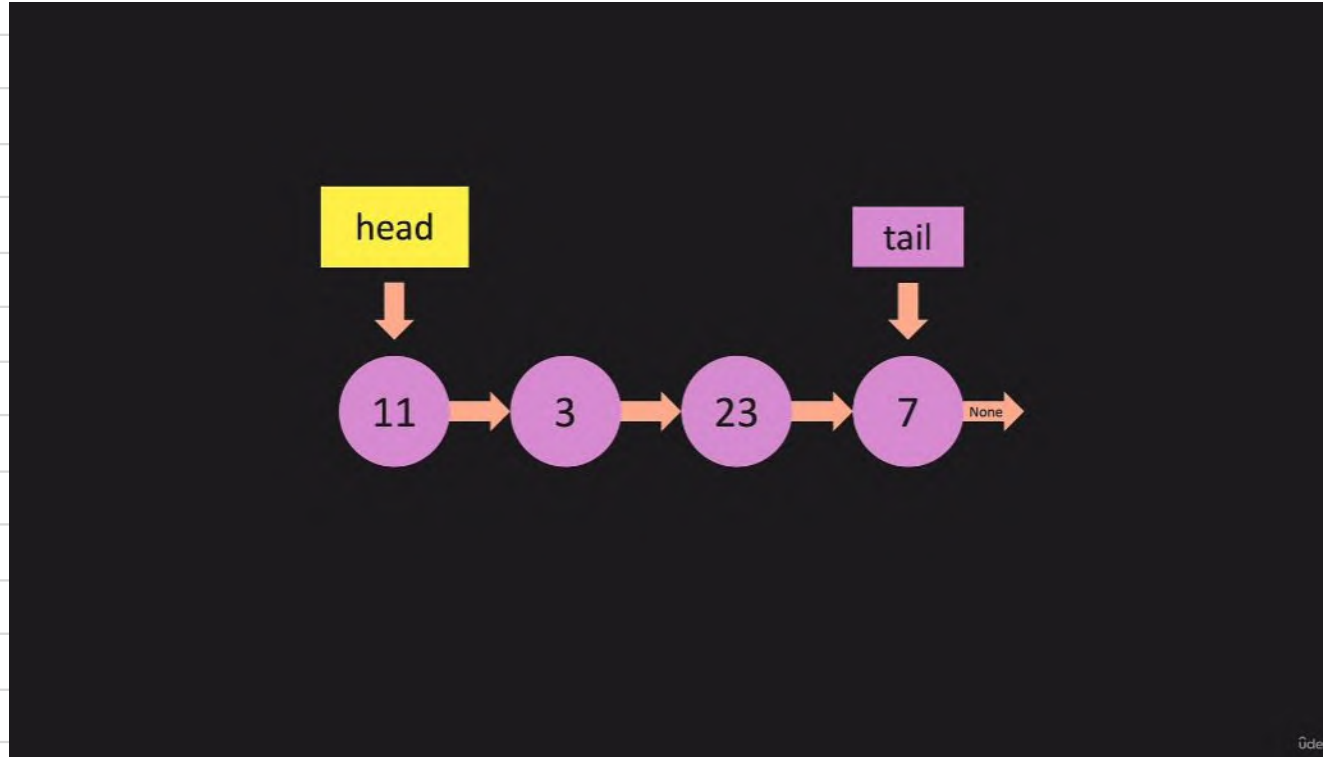
Search Node

Bagaimana jika kita ingin mencari node "23" di dalam linked list



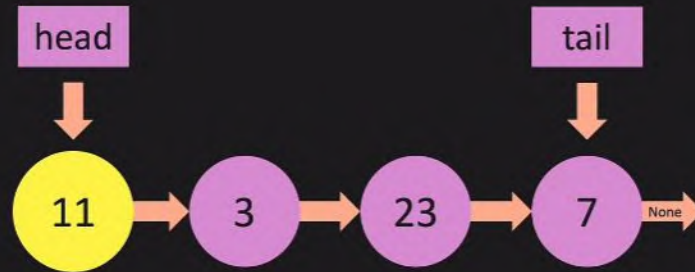
Search Node

Bagaimana jika kita ingin mencari node "23" di dalam linked list



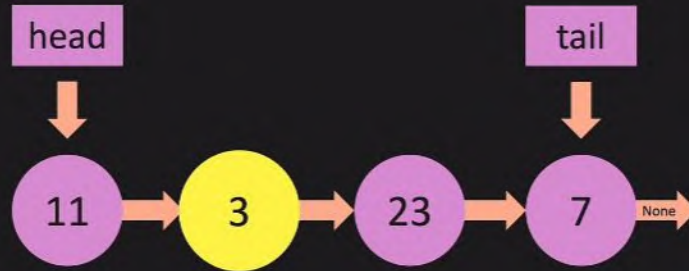
Search Node

Bagaimana jika kita ingin mencari node "23" di dalam linked list



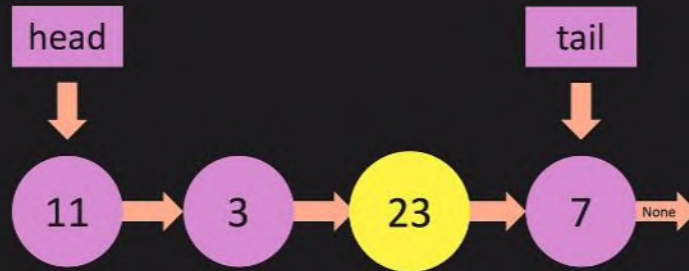
Search Node

Bagaimana jika kita ingin mencari node "23" di dalam linked list



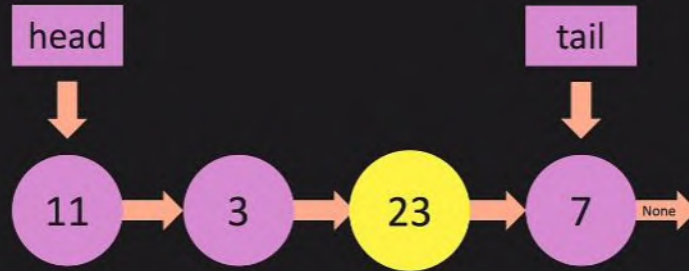
Search Node

Bagaimana jika kita ingin mencari node "23" di dalam linked list



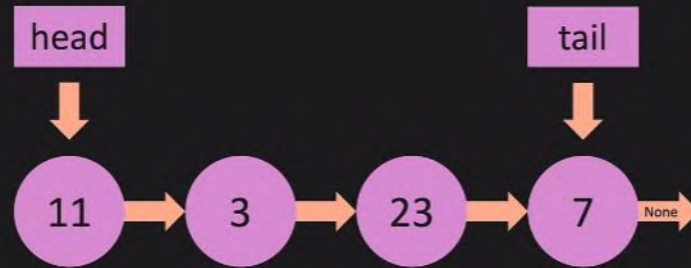
Search Node

Apa notasi Big O untuk
Mencari suatu node... ?



Search Node

Apa notasi Big O untuk
Mencari suatu node... ?



$O(n)$

List VS Linked List

	Linked Lists	Lists
Append	$O(1)$	$O(1)$
Pop	$O(n)$	$O(1)$
Prepend	$O(1)$	$O(n)$
Pop First	$O(1)$	$O(n)$
Insert	$O(n)$	$O(n)$
Remove	$O(n)$	$O(n)$
Lookup by Index	$O(n)$	$O(1)$
Lookup by Value	$O(n)$	$O(n)$

List VS Linked List

	Linked Lists	Lists
Append	$O(1)$	$O(1)$
Pop	$O(n)$	$O(1)$
Prepend	$O(1)$	$O(n)$
Pop First	$O(1)$	$O(n)$
Insert	$O(n)$	$O(n)$
Remove	$O(n)$	$O(n)$
Lookup by Index	$O(n)$	$O(1)$
Lookup by Value	$O(n)$	$O(n)$

List VS Linked List

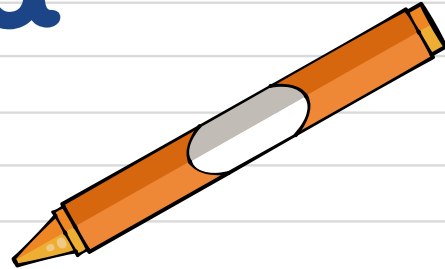
	Linked Lists	Lists
Append	$O(1)$	$O(1)$
Pop	$O(n)$	$O(1)$
Prepend	$O(1)$	$O(n)$
Pop First	$O(1)$	$O(n)$
Insert	$O(n)$	$O(n)$
Remove	$O(n)$	$O(n)$
Lookup by Index	$O(n)$	$O(1)$
Lookup by Value	$O(n)$	$O(n)$



03



Linked List Under The Hood

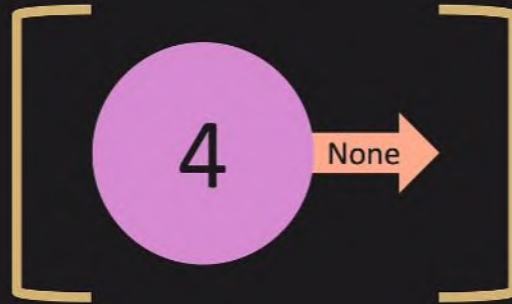


Apa itu Node...?

Apa itu Node...?

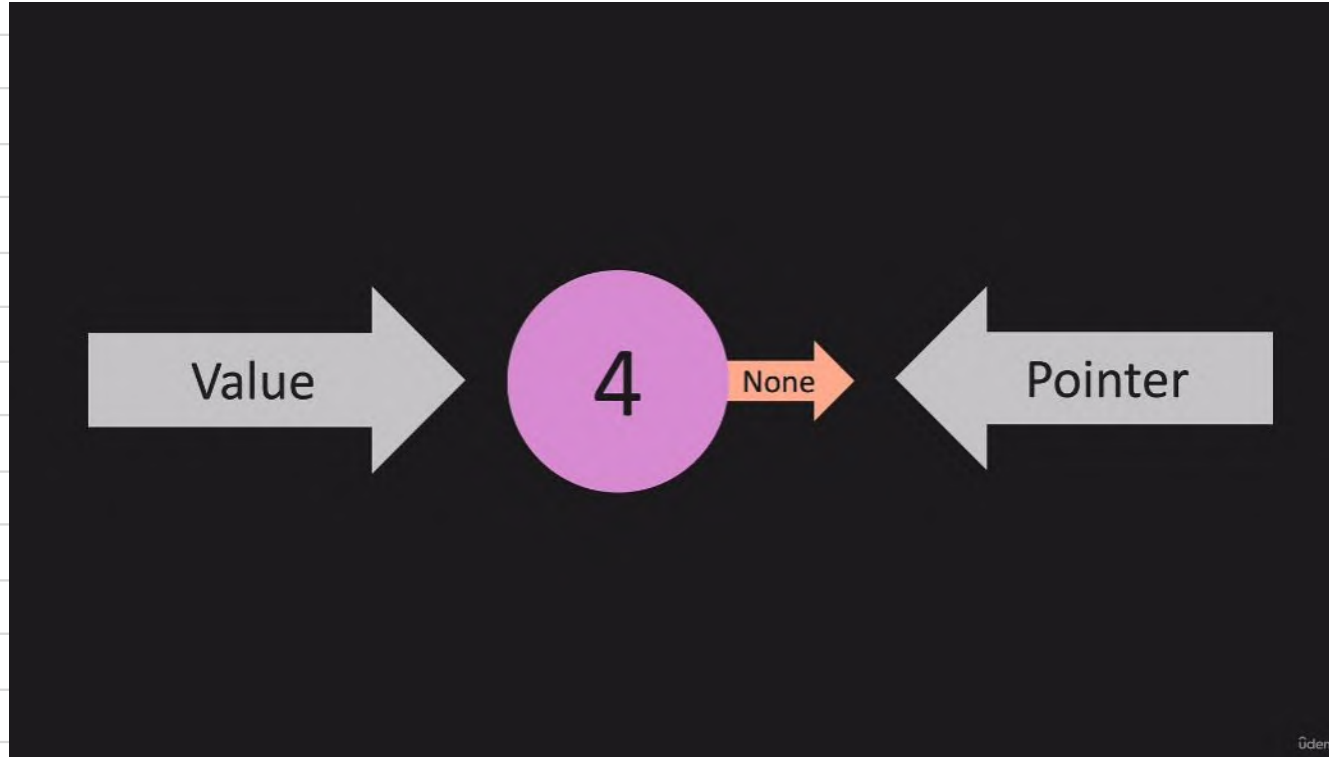


Apa itu Node...?



Node

Apa itu Node...?



Apa itu Node...?

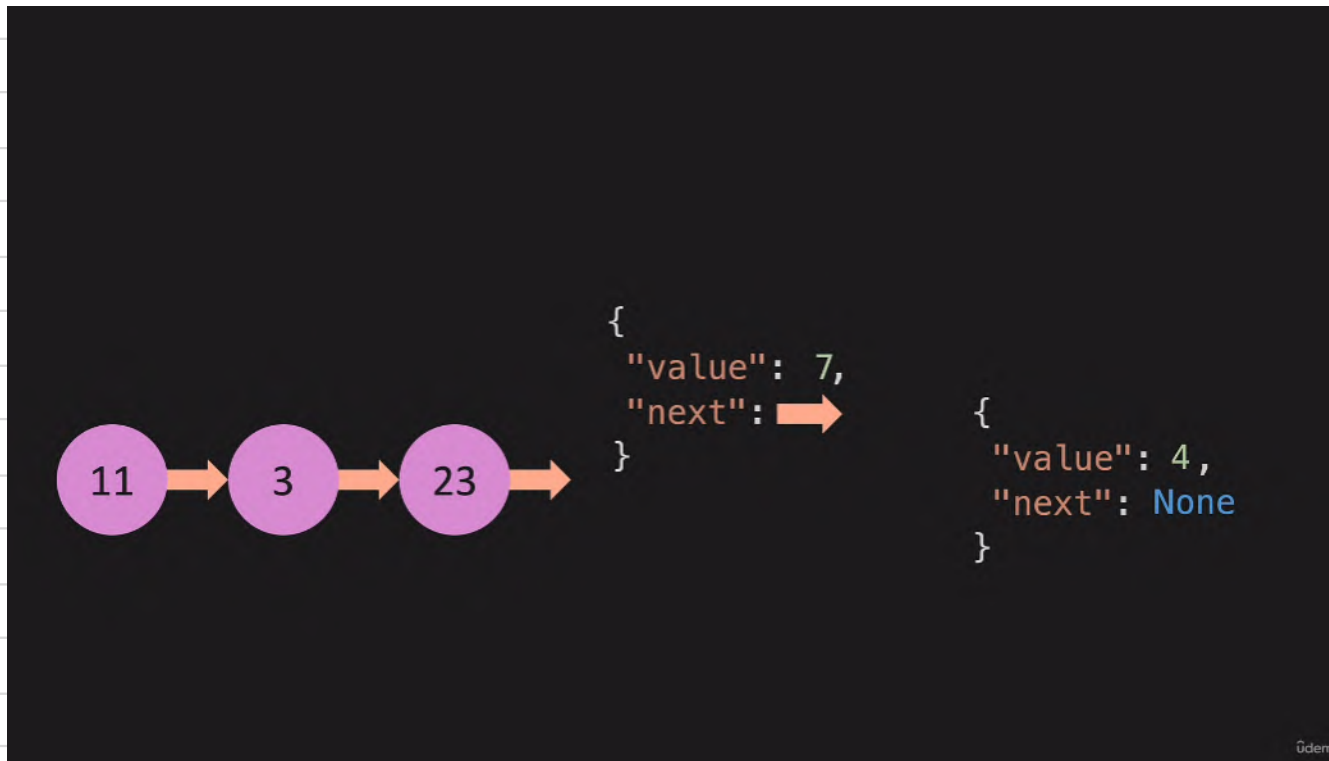
```
{  
  "value": 4,  
  "next": None  
}
```

Apa itu Node...?



```
{  
  "value": 4,  
  "next": None  
}
```

Apa itu Node...?



Apa itu Node...?

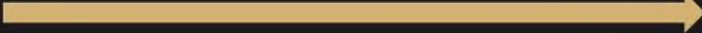


```
{  
  "value": 7,  
  "next": {  
    "value": 4,  
    "next": None  
  }  
}
```

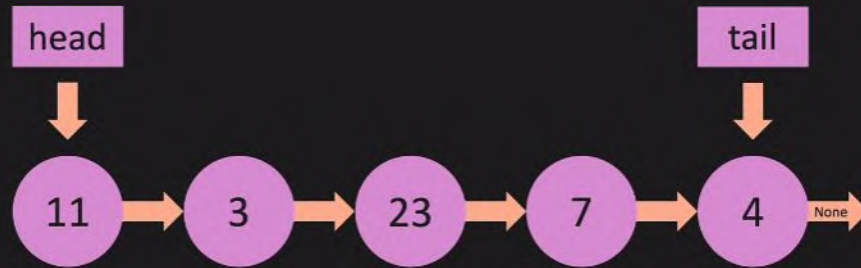
Apa itu Node...?

```
head: {  
  "value": 11,  
  "next": {  
    "value": 3,  
    "next": {  
      "value": 23,  
      "next": {  
        "value": 7,  
        "next": {  
          "value": 4,  
          "next": None  
        }  
      }  
    }  
  }  
}
```

Apa itu Node...?

```
head: {  
  "value": 11,  
  "next": {  
    "value": 3,  
    "next": {  
      "value": 23,  
      "next": {  
        "value": 7,  
        "next": {  
          "value": 4,  
          "next": None  
        }  
      }  
    }  
  }  
}  
  
tail: 
```

Apa itu Node...?

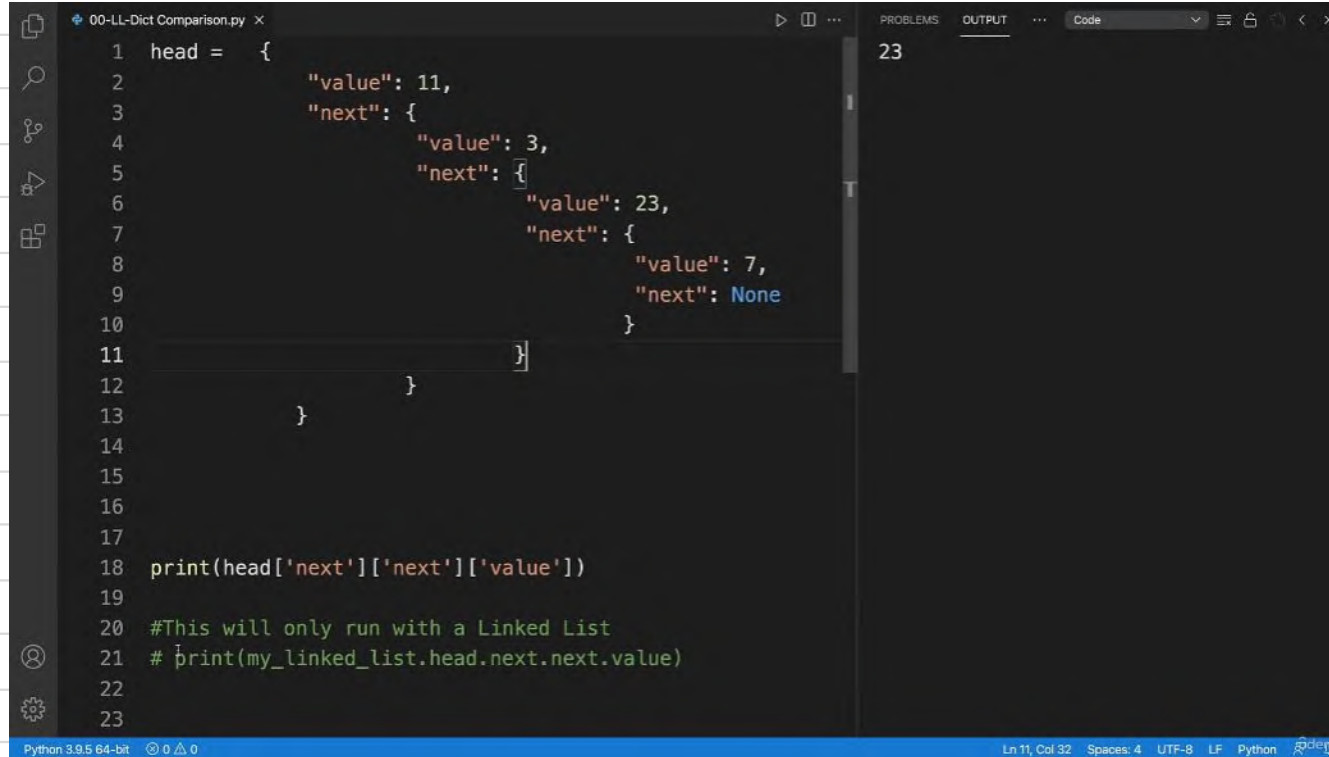


Apa itu Node...?

```
head = {  
    "value": 11,  
    "next": {  
        "value": 3,  
        "next": {  
            "value": 23,  
            "next": {  
                "value": 7,  
                "next": None  
            }  
        }  
    }  
}
```

```
print(head['next']['next']['value'])
```


Apa itu Node...?



```
00-LL-Dict Comparison.py x
1 head = {
2     "value": 11,
3     "next": {
4         "value": 3,
5         "next": {
6             "value": 23,
7             "next": {
8                 "value": 7,
9                 "next": None
10            }
11        }
12    }
13 }

14
15
16
17
18 print(head['next']['next']['value'])
19
20 #This will only run with a Linked List
21 # print(my_linked_list.head.next.next.value)
22
23
```

The screenshot shows a code editor with a Python file named '00-LL-Dict Comparison.py'. The code defines a dictionary 'head' representing a linked list. The 'head' dictionary has a 'value' of 11 and a 'next' key pointing to another dictionary. This second dictionary has a 'value' of 3 and a 'next' key pointing to a third dictionary. The third dictionary has a 'value' of 23 and a 'next' key pointing to a fourth dictionary. The fourth dictionary has a 'value' of 7 and a 'next' key pointing to 'None'. The code also includes a print statement to output the value of the third node (23) and two commented-out lines. The editor's interface includes a sidebar with icons for Explorer, Search, Source Control, Run and Debug, and Extensions. The top bar shows the file name and some editor controls. The bottom status bar indicates the Python version (3.9.5 64-bit) and the current line and column (Ln 11, Col 32).

Syntax sangat mirip

```
8                                     "value": 7,  
9                                     "next": None  
10                                    }  
11                                }  
12                            }  
13                        }  
14  
15  
16  
17  
18    print(head['next']['next']['value'])  
19  
20    #This will only run with a Linked List  
21    print(my_linked_list.head.next.next.value)  
22  
23
```

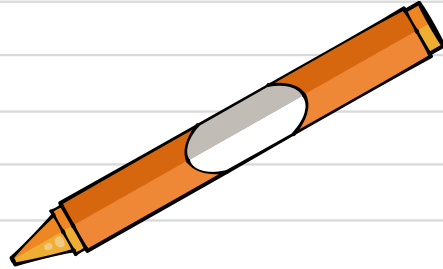
Python 3.9.5 64-bit 0 0 0



04



Linked List Constructor



Linked List: Class

```
class LinkedList:  
    def __init__(self, value):  
    def append(self, value):  
    def prepend(self, value):  
    def insert(self, index, value):
```

Linked List: Class

```
class LinkedList:  
    def __init__(self, value):  
    def append(self, value):  
    def prepend(self, value):  
    def insert(self, index, value)
```

Linked List: Class

```
class LinkedList:
    def __init__(self, value):
        create new Node
    def append(self, value):
        create new Node
        add Node to end
    def prepend(self, value):
        create new Node
        add Node to beginning
    def insert(self, index, value):
        create new Node
        insert Node
```

Linked List: Class

```
create new Node  
create new Node  
create new Node  
create new Node
```

```
class LinkedList:  
    def __init__(self, value):  
    def append(self, value):  
    def prepend(self, value):  
    def insert(self, index, value):
```

Ingat kembali Node



Ingat kembali Node

```
{  
  "value": 4,  
  "next": None  
}
```

Node: Class

```
class Node:
    def __init__(self, value):
        self.value = value
        self.next = None
```

Node: Class

```
class LinkedList:  
    def __init__(self, value):  
        new_node = Node(value)
```



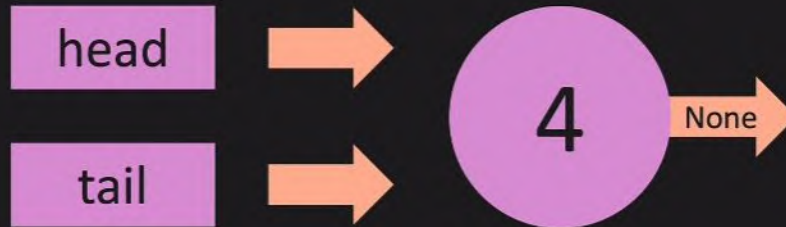
Node: Class

```
class LinkedList:  
    def __init__(self, value):  
        new_node = Node(value)  
        self.head = new_node
```



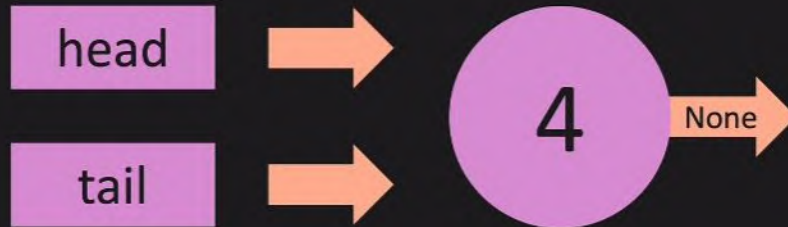
Node: Class

```
class LinkedList:  
    def __init__(self, value):  
        new_node = Node(value)  
        self.head = new_node  
        self.tail = new_node
```



Node: Class

```
class LinkedList:  
    def __init__(self, value):  
        new_node = Node(value)  
        self.head = new_node  
        self.tail = new_node  
        self.length = 1
```



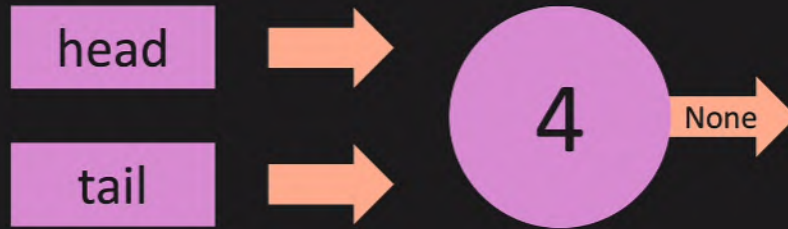
Node: Class

```
class LinkedList:
    def __init__(self, value):
        new_node = Node(value)
        self.head = new_node
        self.tail = new_node
        self.length = 1
```

```
my_linked_list = LinkedList(4)
```

Node: Class

```
my_linked_list = LinkedList(4)
```



Linked List : Constructor

```
class Node:
    def __init__(self, value):
        self.value = value
        self.next = None

class LinkedList:
    def __init__(self, value):
        new_node = Node(value)
        self.head = new_node
        self.tail = new_node
        self.length = 1

my_linked_list = LinkedList(4)

print(my_linked_list.head.value)
```

Linked List : Constructor

```
class Node:
    def __init__(self, value):
        self.value = value
        self.next = None

class LinkedList:
    def __init__(self, value):
        new_node = Node(value)
        self.head = new_node
        self.tail = new_node
        self.length = 1

my_linked_list = LinkedList(4)

print(my_linked_list.head.value)
```

4

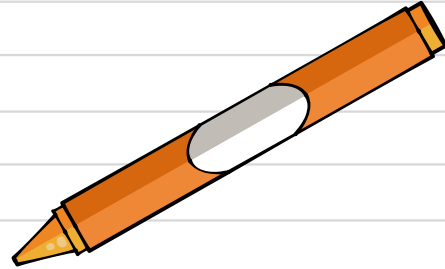


05



Linked List

Print List

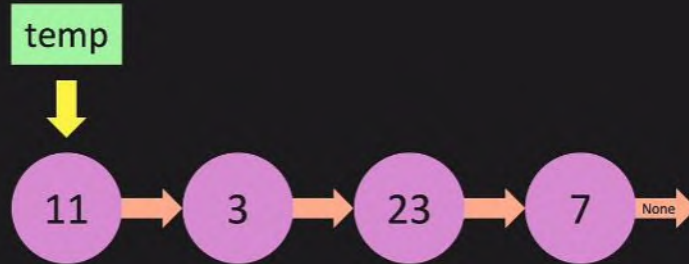


Linked List : Print List

```
def print_list(self):  
    temp = self.head
```

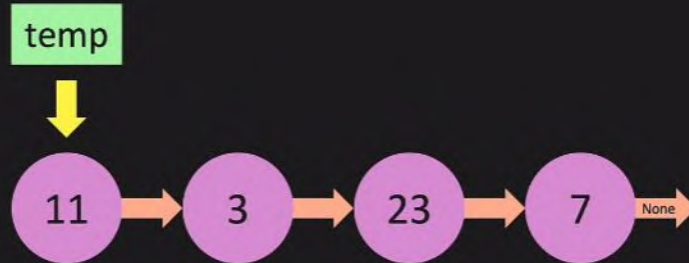
Linked List : Print List

```
def print_list(self):  
    temp = self.head
```



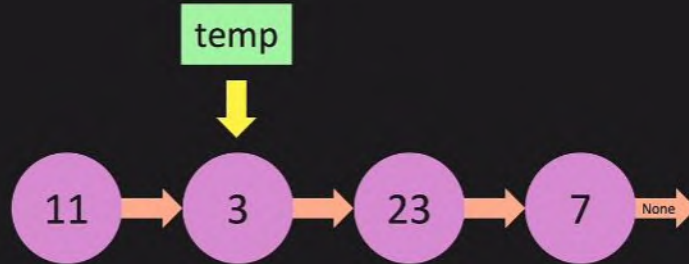
Linked List : Print List

```
def print_list(self):  
    temp = self.head  
    while temp is not None:  
        print(temp.value)  
        temp = temp.next
```



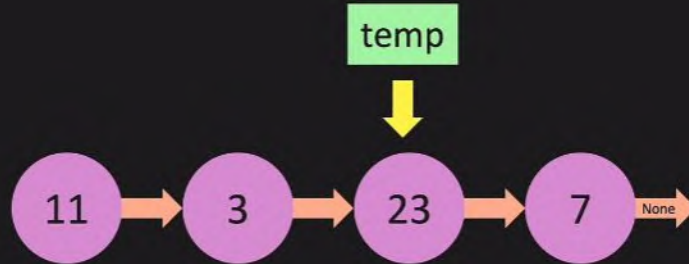
Linked List : Print List

```
def print_list(self):  
    temp = self.head  
    while temp is not None:  
        print(temp.value)  
        temp = temp.next
```



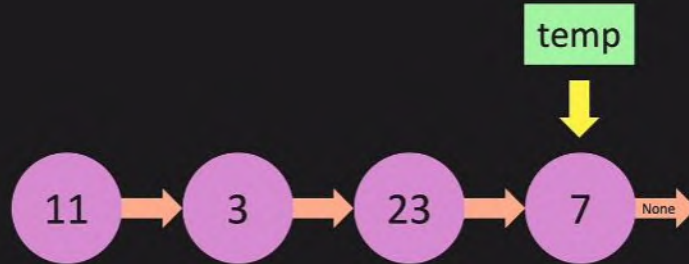
Linked List : Print List

```
def print_list(self):  
    temp = self.head  
    while temp is not None:  
        print(temp.value)  
        temp = temp.next
```



Linked List : Print List

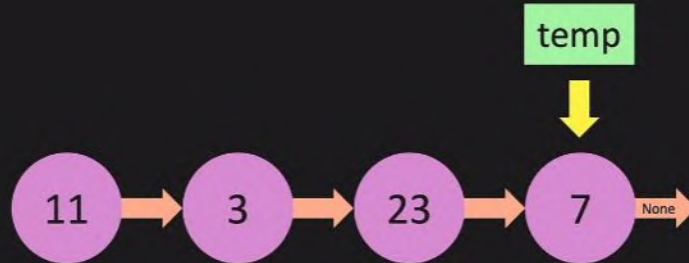
```
def print_list(self):  
    temp = self.head  
    while temp is not None:  
        print(temp.value)  
        temp = temp.next
```



Dimana posisi head pada akhir
dari Print List?

Linked List : Print List

```
def print_list(self):  
    temp = self.head  
    while temp is not None:  
        print(temp.value)  
        temp = temp.next
```





06

Linked List Append

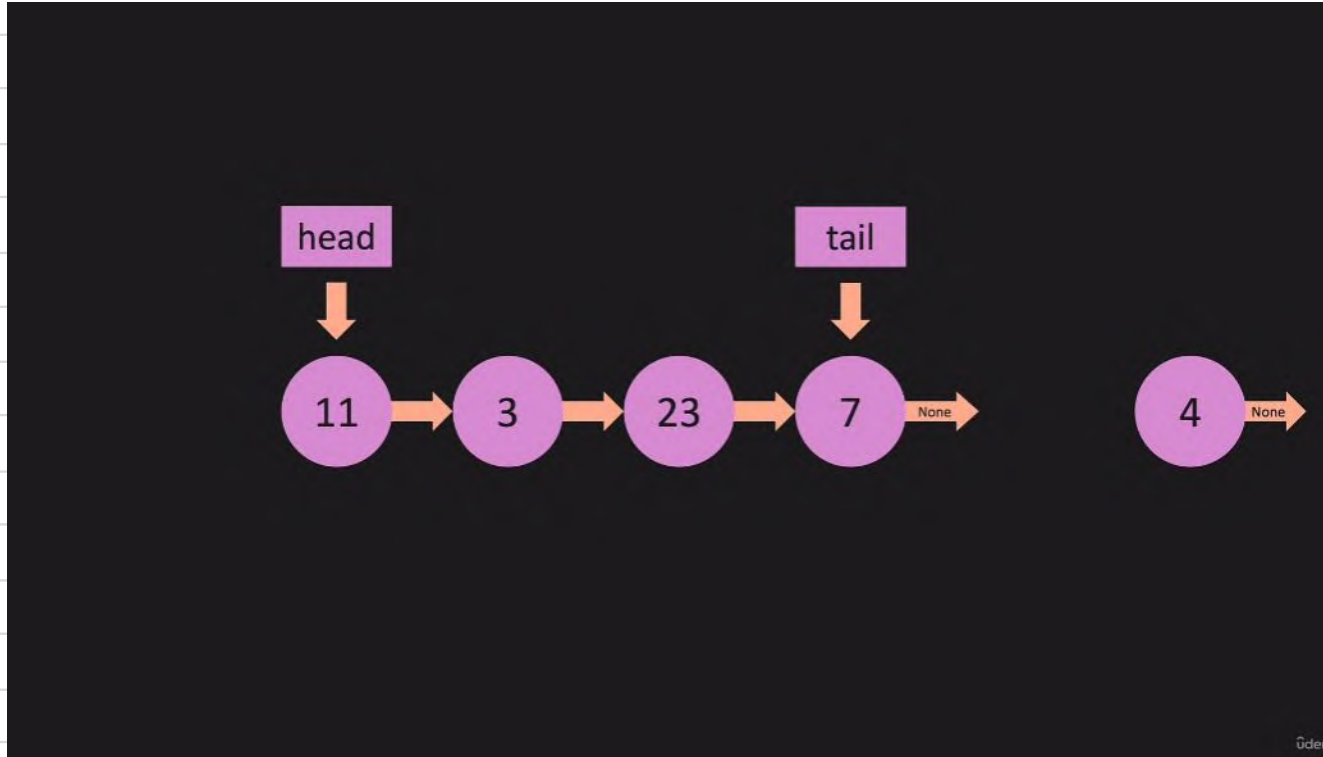


Linked List : Append

- Apa itu Append...?

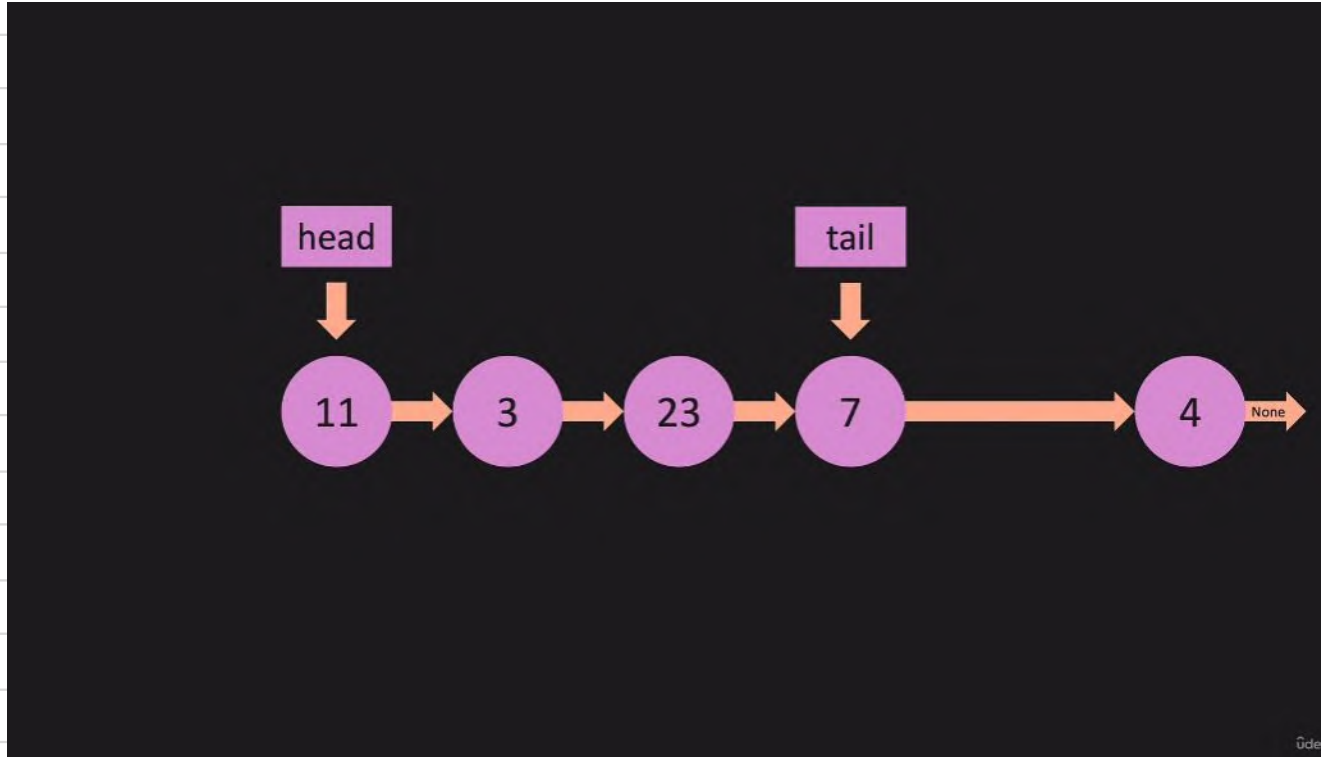
Buat node baru

Linked List : Append



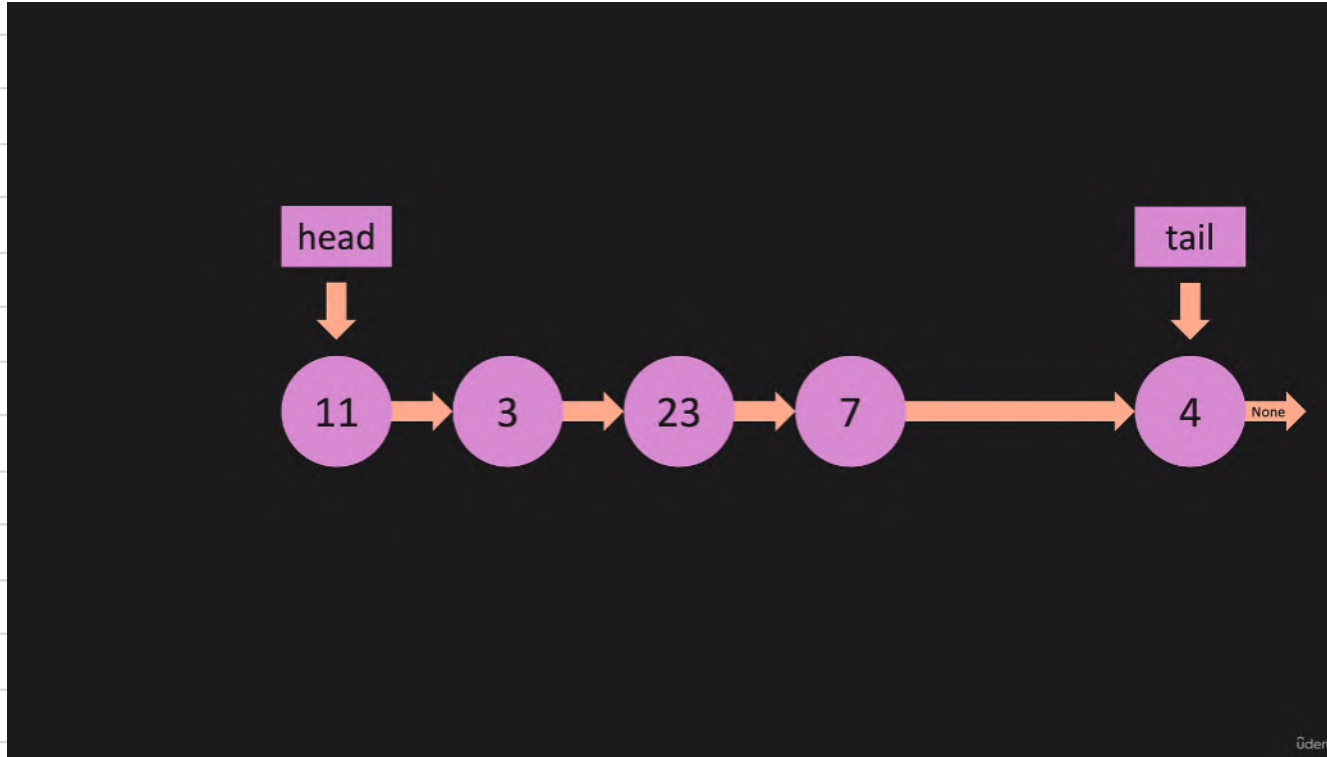
Arahkan tail.next ke node baru

Linked List : Append



Arahkan tail ke node baru

Linked List : Append

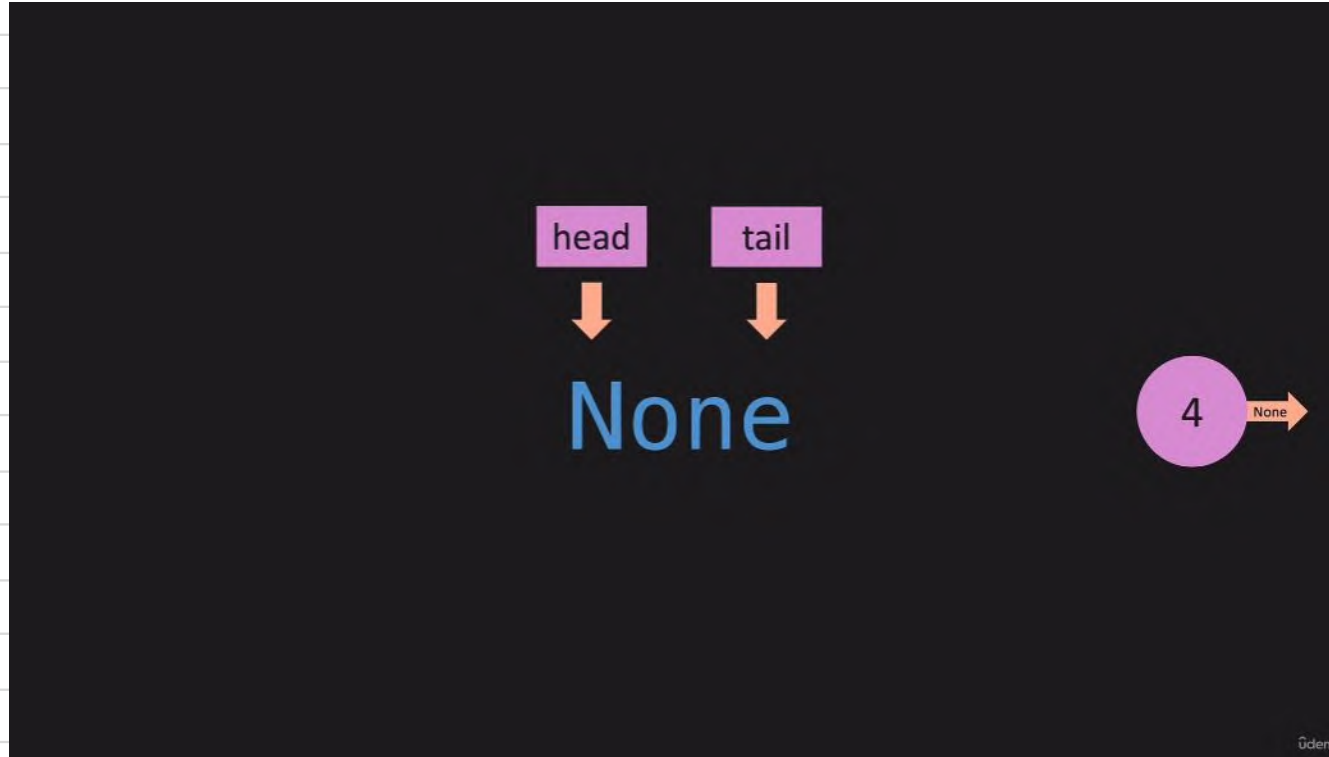


Linked List : Append

- Namun ada beberapa kondisi yang harus mendapatkan perhatian khusus

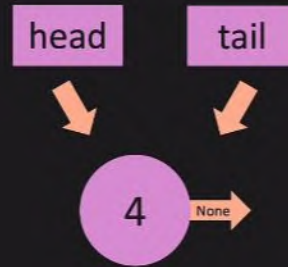
Bagaimana append bekerja?

Linked List : Append

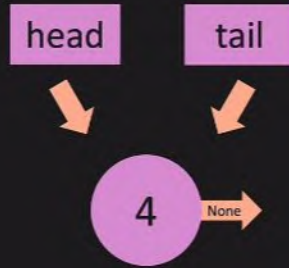


Bagaimana append bekerja?

Linked List : Append



Linked List : Append



```
if self.head is None:  
    self.head = new_node  
    self.tail = new_node
```

Linked List : Append Function

```
def append(self, value):  
    new_node = Node(value)
```



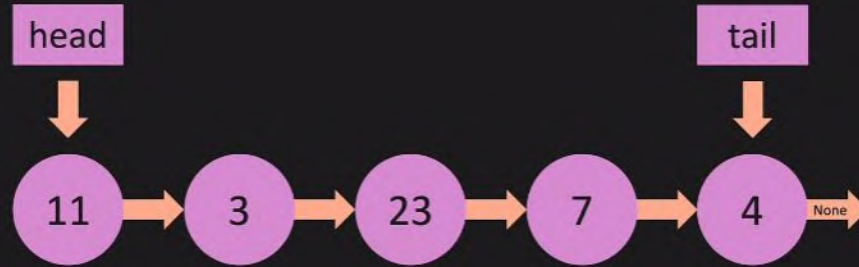
Linked List : Append Function

```
def append(self, value):  
    new_node = Node(value)  
    if self.head is None:  
        self.head = new_node  
        self.tail = new_node
```

Linked List : Append Function

```
def append(self, value):  
    new_node = Node(value)  
    if self.head is None:  
        self.head = new_node  
        self.tail = new_node  
    else:
```

Linked List : Append Function



```
else:  
    self.tail.next = new_node  
    self.tail = new_node
```

Linked List : Append Function

```
def append(self, value):  
    new_node = Node(value)  
    if self.head is None:  
        self.head = new_node  
        self.tail = new_node  
    else:  
        self.tail.next = new_node  
        self.tail = new_node
```

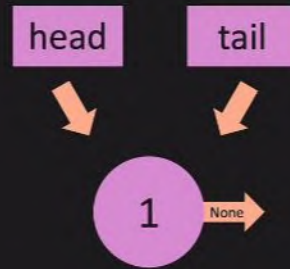

Linked List : Append Function

```
def append(self, value):  
    new_node = Node(value)  
    if self.head is None:  
        self.head = new_node  
        self.tail = new_node  
    else:  
        self.tail.next = new_node  
        self.tail = new_node  
    self.length += 1
```

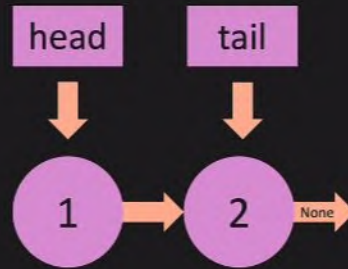
Linked List : Append Function

```
def append(self, value):  
    new_node = Node(value)  
    if self.head is None:  
        self.head = new_node  
        self.tail = new_node  
    else:  
        self.tail.next = new_node  
        self.tail = new_node  
    self.length += 1  
    return True
```

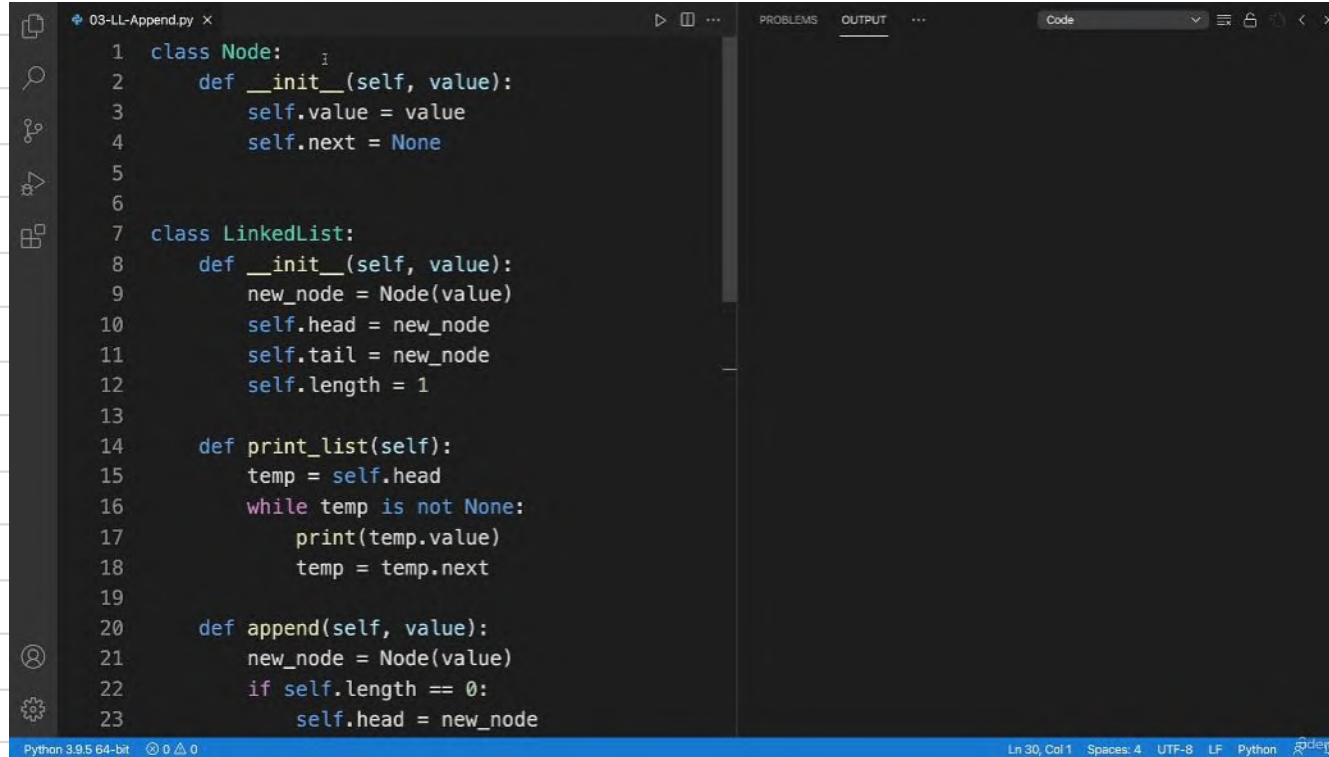
Linked List : Append Skenario Percobaan



Linked List : Append Skenario Percobaan



Linked List : Append Skenario Percobaan



```
03-LL-Append.py X
1 class Node:
2     def __init__(self, value):
3         self.value = value
4         self.next = None
5
6
7 class LinkedList:
8     def __init__(self, value):
9         new_node = Node(value)
10        self.head = new_node
11        self.tail = new_node
12        self.length = 1
13
14    def print_list(self):
15        temp = self.head
16        while temp is not None:
17            print(temp.value)
18            temp = temp.next
19
20    def append(self, value):
21        new_node = Node(value)
22        if self.length == 0:
23            self.head = new_node
```

The image shows a code editor window titled '03-LL-Append.py'. The code defines a 'Node' class with an '__init__' method that takes a 'value' and sets 'self.value' and 'self.next' (to None). It also defines a 'LinkedList' class with an '__init__' method that creates a new node, sets 'self.head' and 'self.tail' to that node, and sets 'self.length' to 1. The 'print_list' method iterates through the list starting from 'self.head' and prints each node's value. The 'append' method creates a new node and, if the list is empty (length == 0), sets 'self.head' to the new node.

Python 3.9.5 64-bit 0 0 0 Ln 30, Col 1 Spaces: 4 UTF-8 LF Python

Linked List : Append Skenario Percobaan

```
def append(self, value):  
    new_node = Node(value)  
    if self.length == 0:  
        self.head = new_node  
        self.tail = new_node  
    else:  
        self.tail.next = new_node  
        self.tail = new_node  
    self.length += 1
```

```
my_linked_list = LinkedList(1)
```

```
my_linked_list.append(2)
```

```
my_linked_list.print_list()
```

Linked List : Append Skenario Percobaan

```
def append(self, value):  
    new_node = Node(value)  
    if self.length == 0:  
        self.head = new_node  
        self.tail = new_node  
    else:  
        self.tail.next = new_node  
        self.tail = new_node  
    self.length += 1
```

```
my_linked_list = LinkedList(1)
```

```
my_linked_list.append(2)
```

```
my_linked_list.print_list()
```

1

2



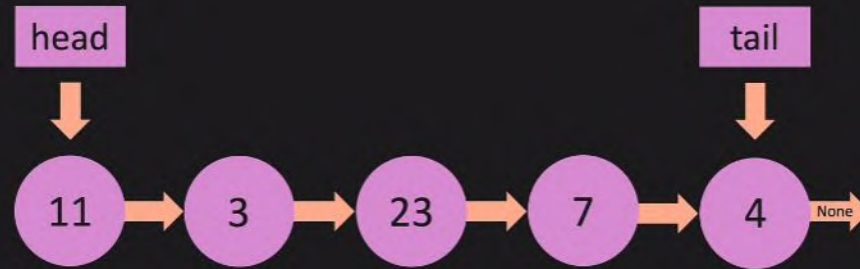
07

Linked List Pop



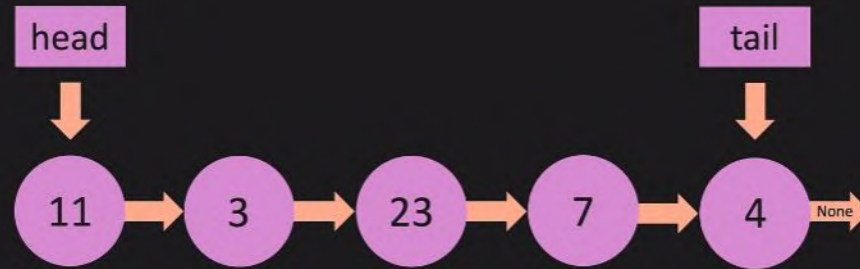
Diketahui suatu list

Linked List : Pop



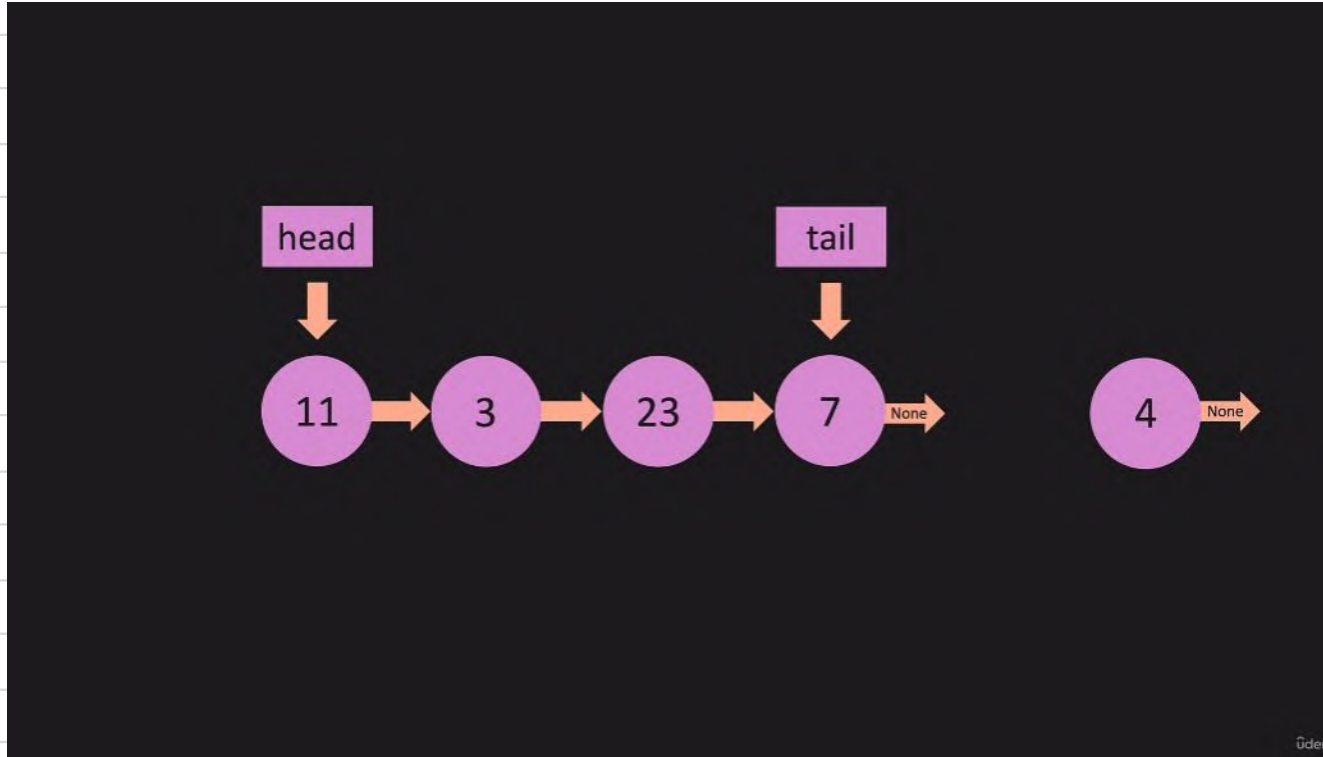
Linked List : Pop

Bagaimana jika ingin
menghapus node paling
belakang? (Pop)



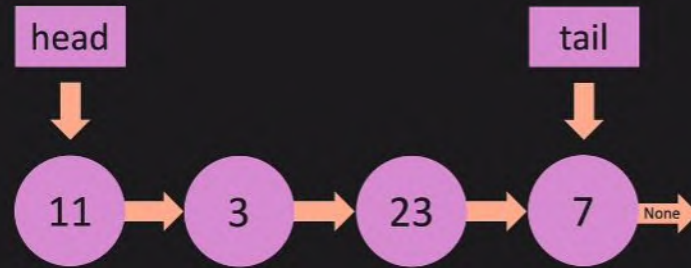
Linked List : Pop

Bagaimana jika ingin menghapus node paling belakang? (Pop)



Linked List : Pop

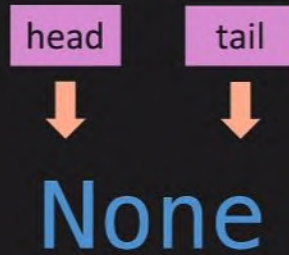
Bagaimana jika ingin menghapus node paling belakang? (Pop)



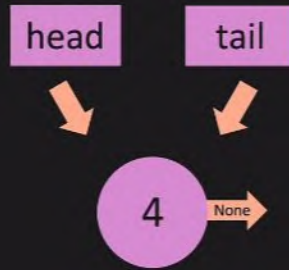
Linked List : Pop

- Namun ada beberapa kondisi yang harus mendapatkan perhatian khusus

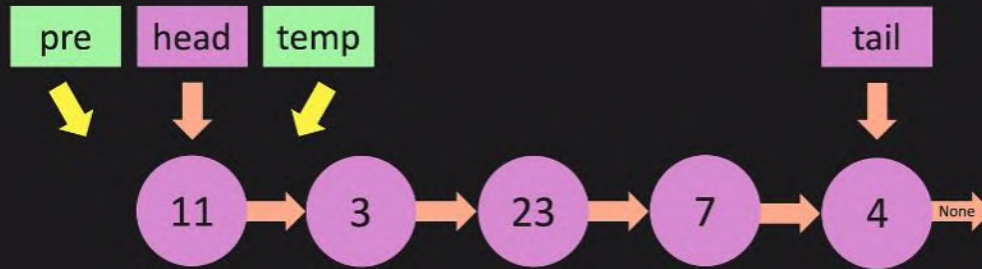
Linked List : Pop



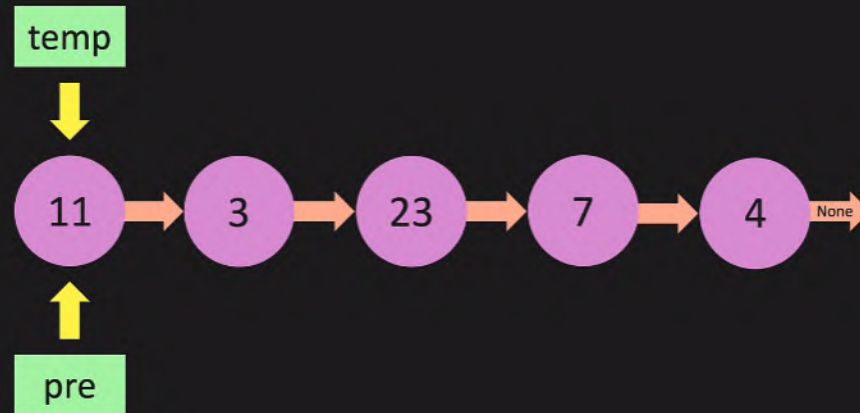
Linked List : Pop



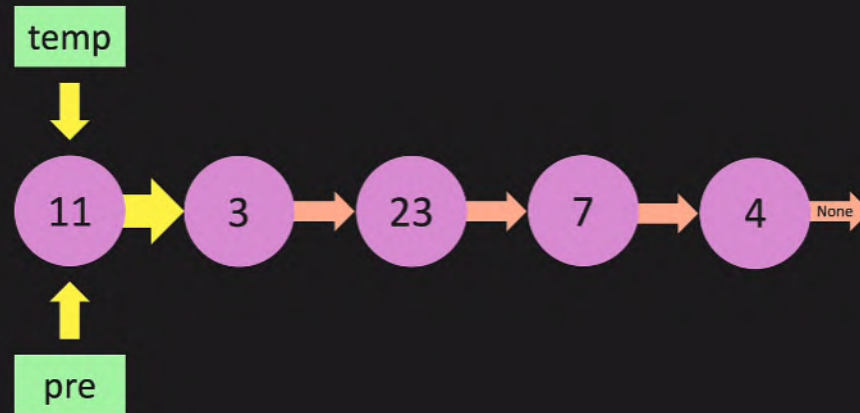
Linked List : Pop – Kasus Umum



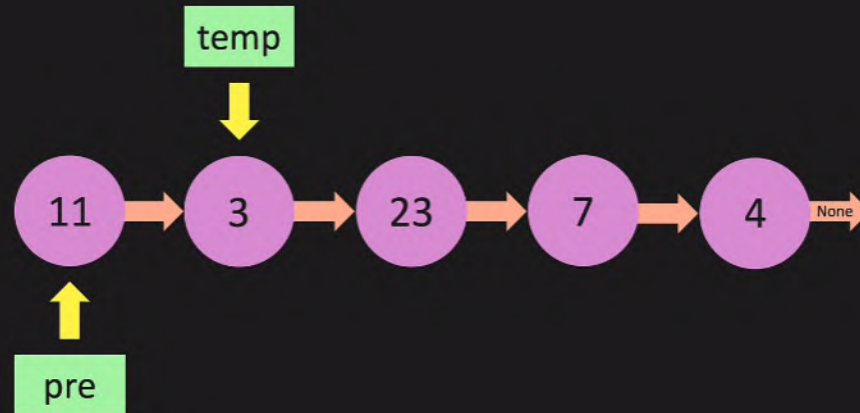
Linked List : Pop – Kasus Umum



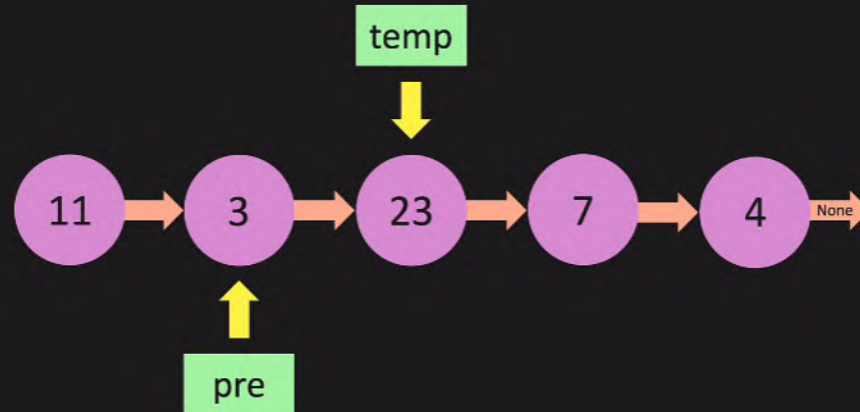
Linked List : Pop – Kasus Umum



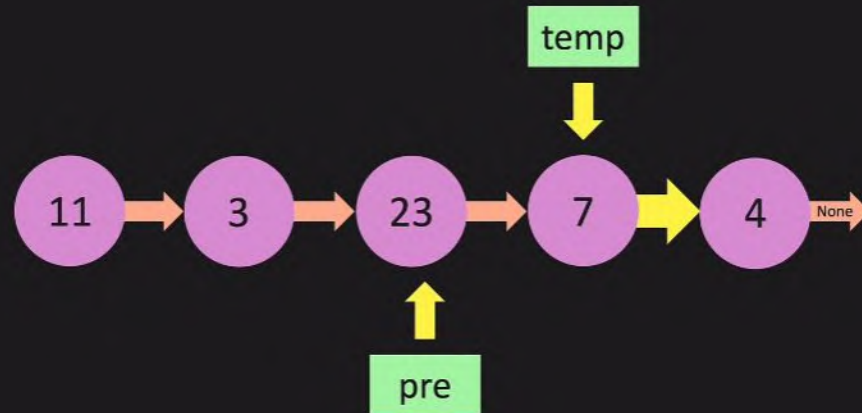
Linked List : Pop – Kasus Umum



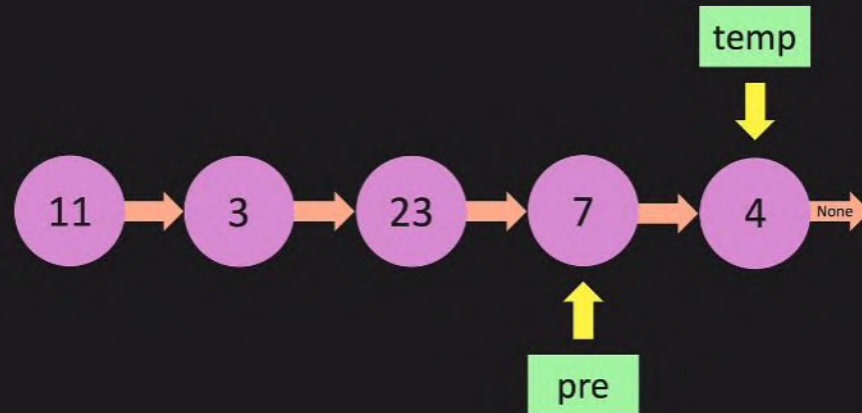
Linked List : Pop – Kasus Umum



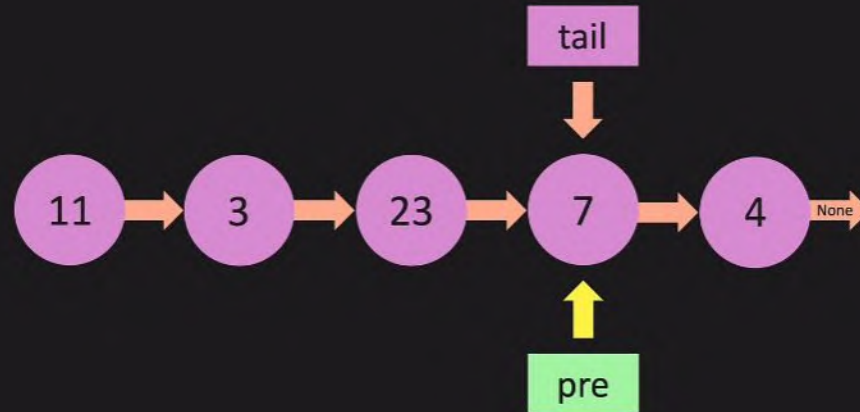
Linked List : Pop – Kasus Umum



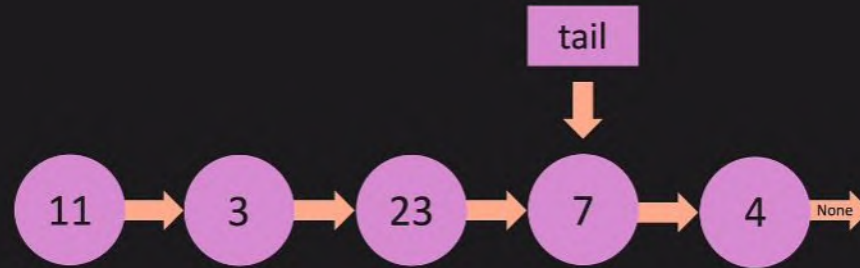
Linked List : Pop – Kasus Umum



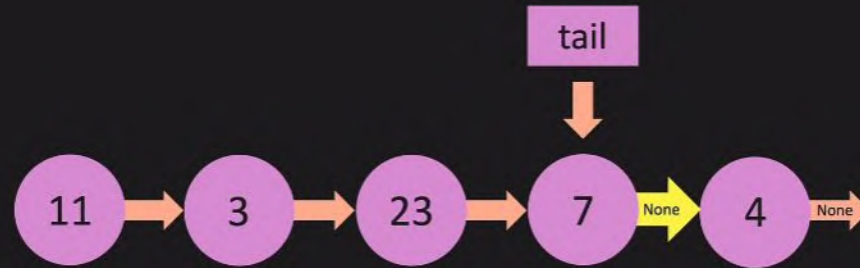
Linked List : Pop – Kasus Umum



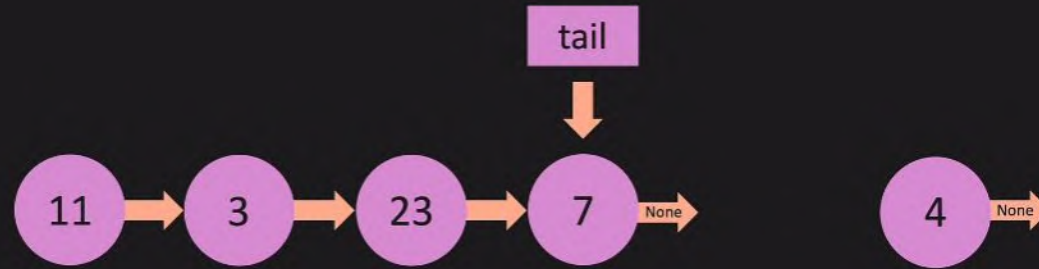
Linked List : Pop – Kasus Umum



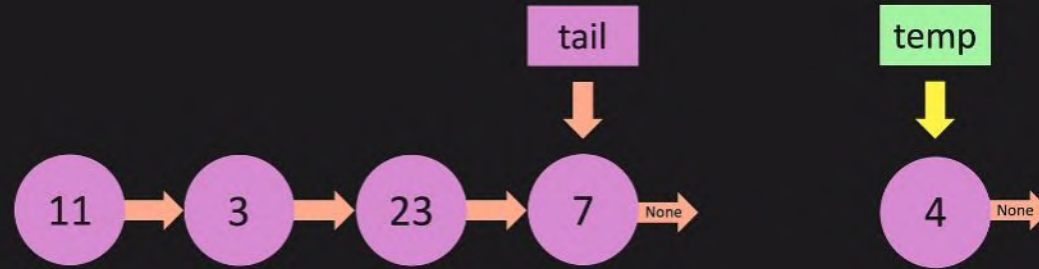
Linked List : Pop – Kasus Umum



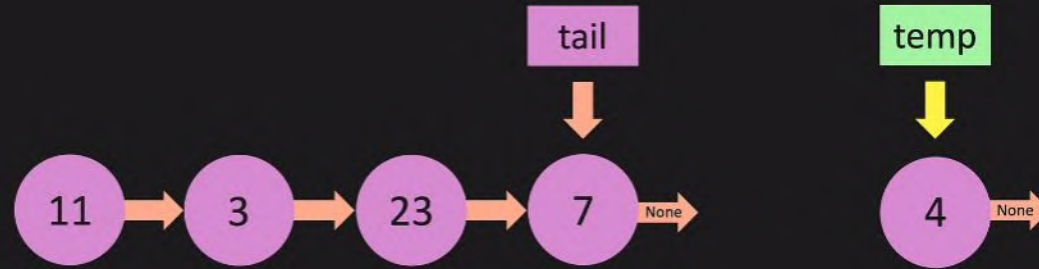
Linked List : Pop – Kasus Umum



Linked List : Pop – Kasus Umum

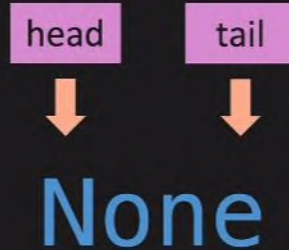


Linked List : Pop – Kasus Umum



Linked List : Pop Code

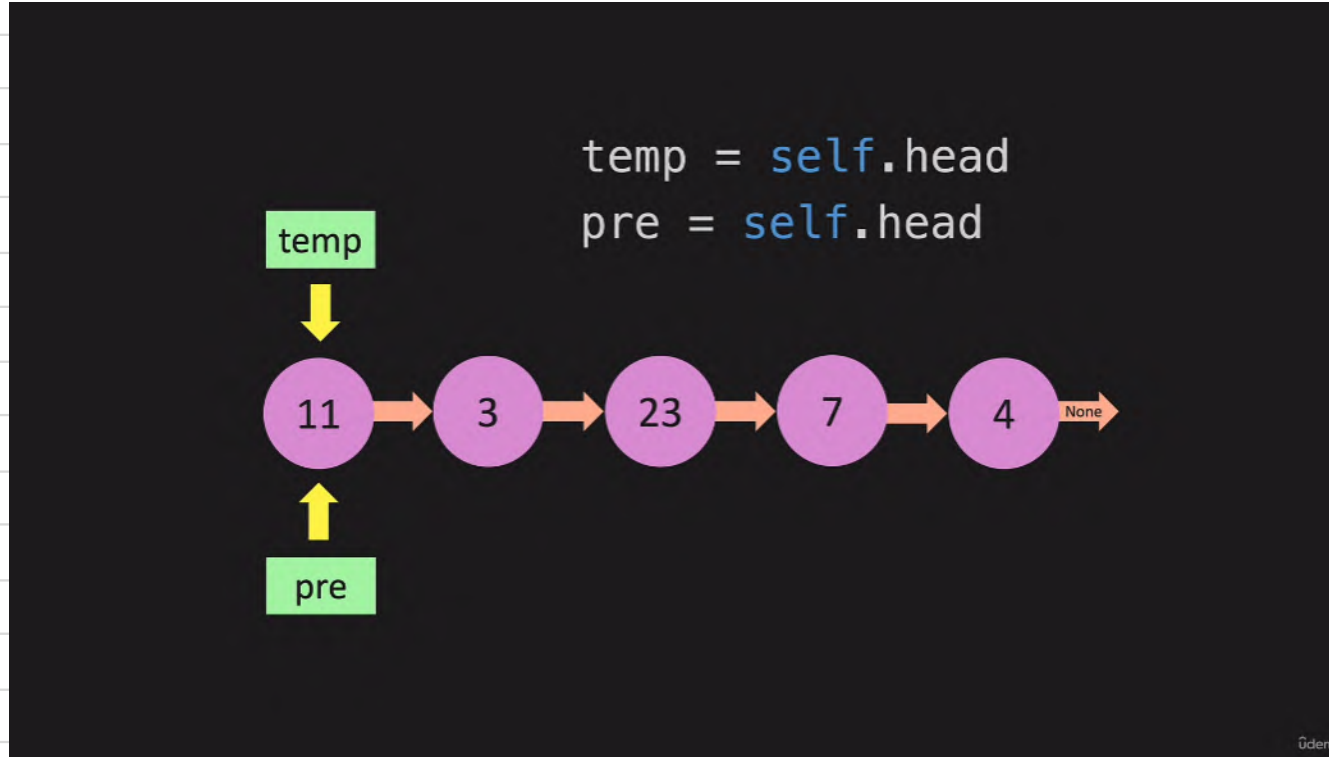
```
def pop(self):  
    if self.length == 0:
```



Linked List : Pop Code

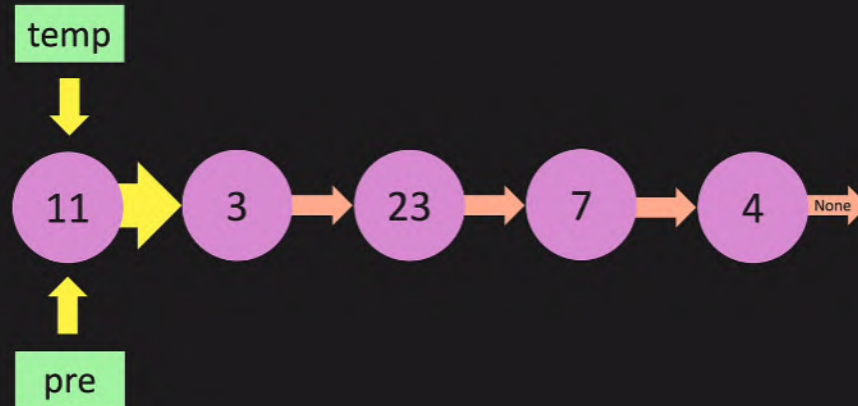
```
def pop(self):  
    if self.length == 0:  
        return None
```

Linked List : Pop Code



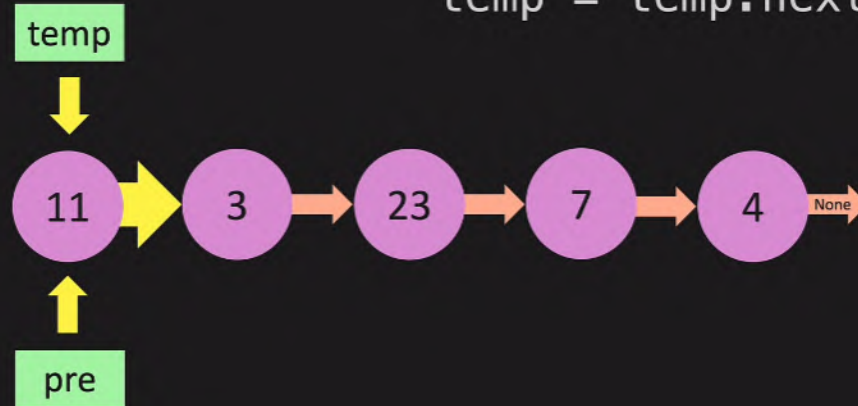
Linked List : Pop Code

```
while(temp.next):
```



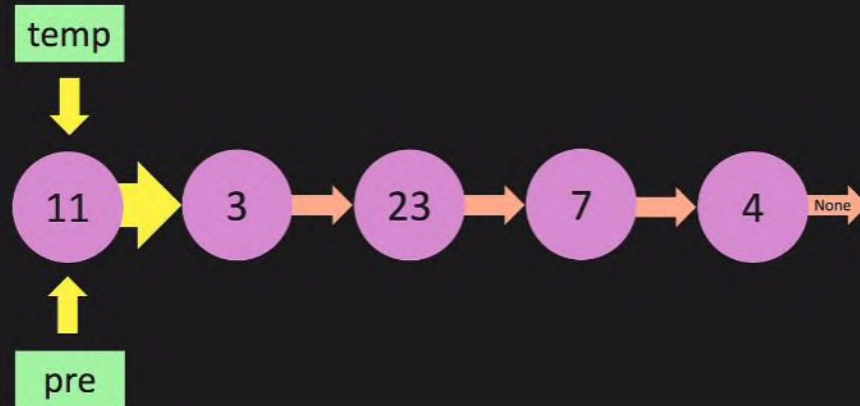
Linked List : Pop Code

```
while(temp.next):  
    pre = temp  
    temp = temp.next
```



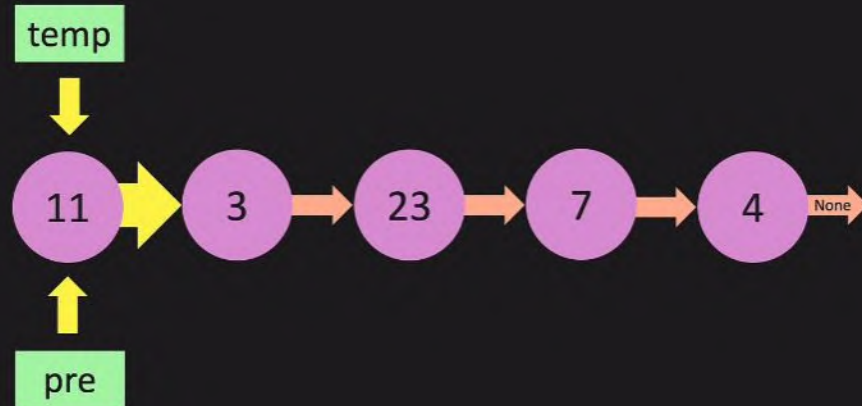
Linked List : Pop Code

```
while(temp.next):  
    pre = temp  
    temp = temp.next
```



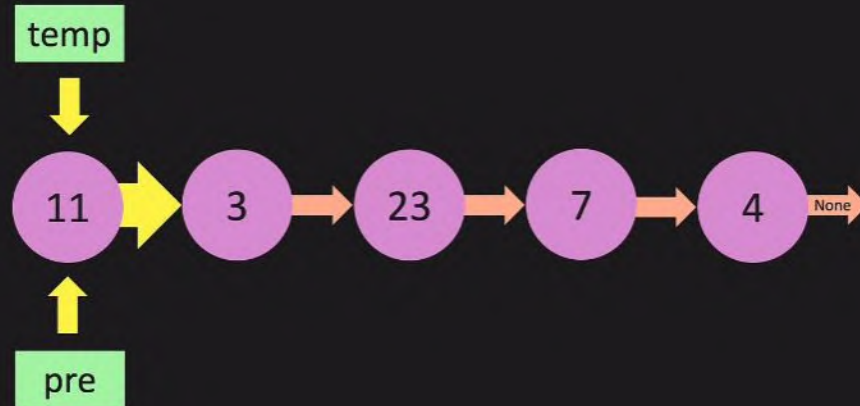
Linked List : Pop Code

```
while(temp.next):  
    pre = temp  
    temp = temp.next
```



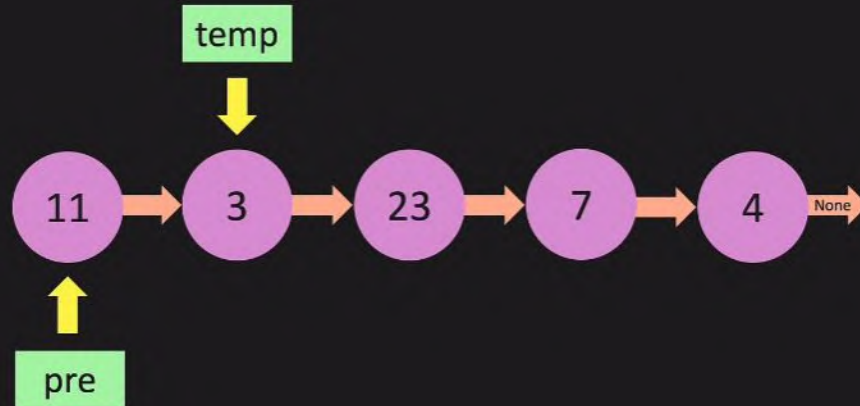
Linked List : Pop Code

```
while(temp.next):  
    pre = temp  
    temp = temp.next
```



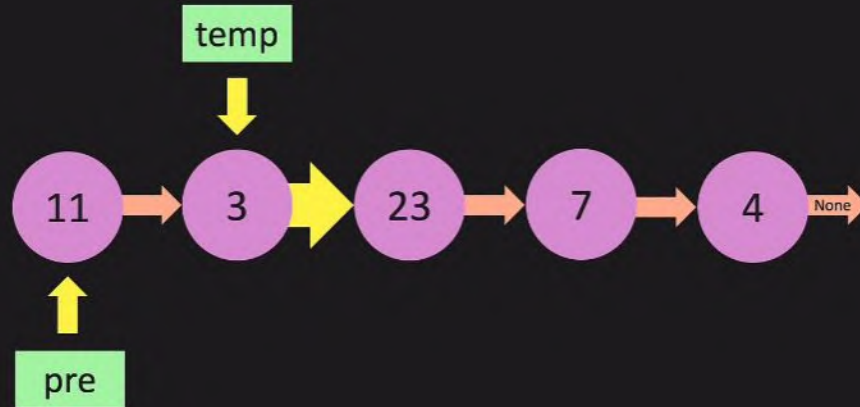
Linked List : Pop Code

```
while(temp.next):  
    pre = temp  
    temp = temp.next
```



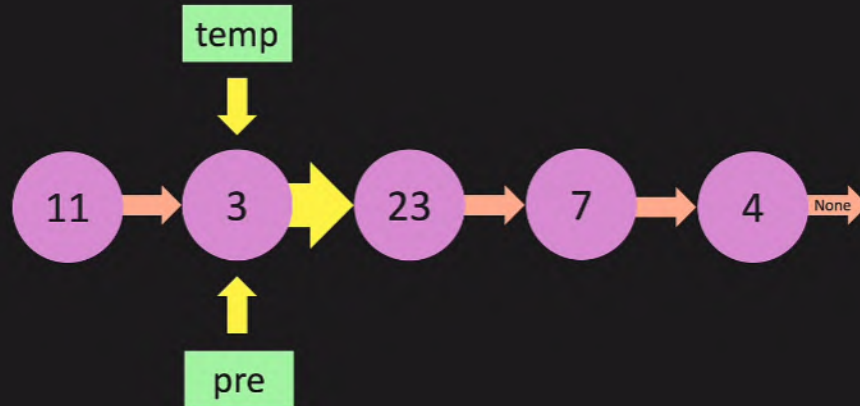
Linked List : Pop Code

```
while(temp.next):  
    pre = temp  
    temp = temp.next
```



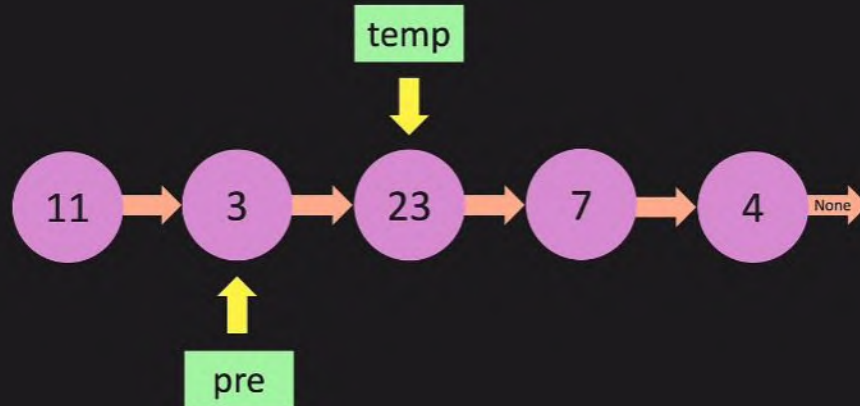
Linked List : Pop Code

```
while(temp.next):  
    pre = temp  
    temp = temp.next
```



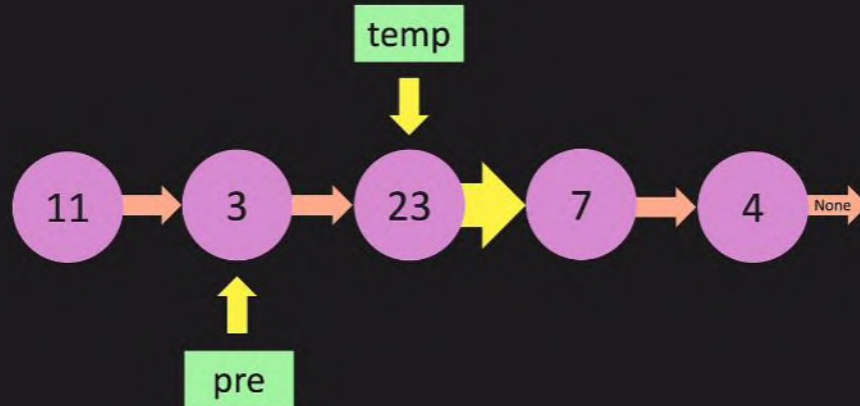
Linked List : Pop Code

```
while(temp.next):  
    pre = temp  
    temp = temp.next
```



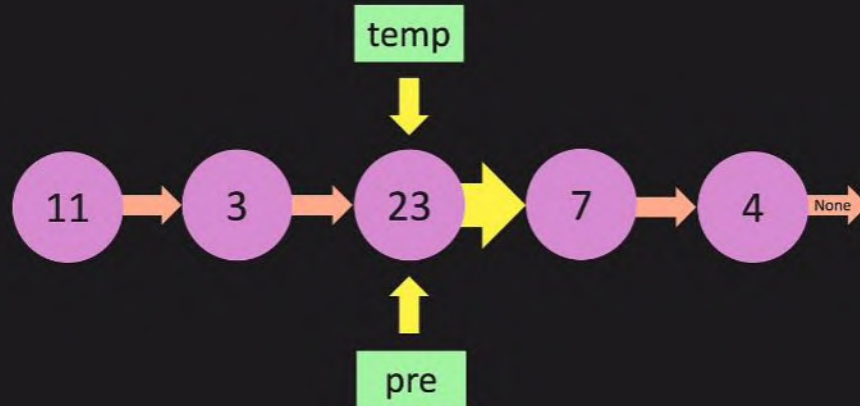
Linked List : Pop Code

```
while(temp.next):  
    pre = temp  
    temp = temp.next
```



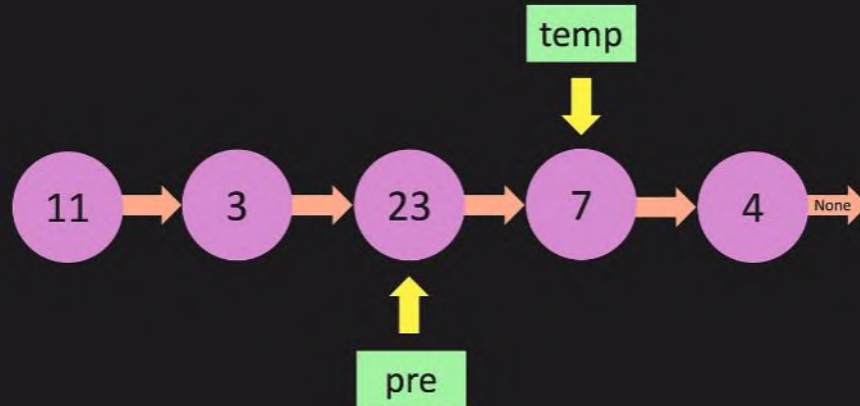
Linked List : Pop Code

```
while(temp.next):  
    pre = temp  
    temp = temp.next
```



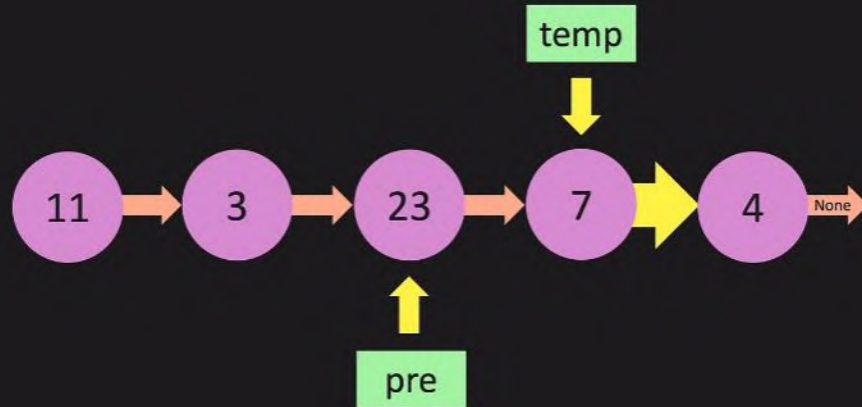
Linked List : Pop Code

```
while(temp.next):  
    pre = temp  
    temp = temp.next
```



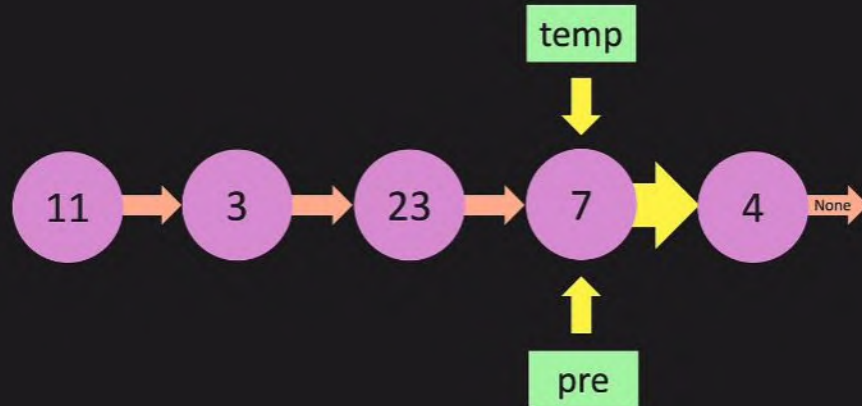
Linked List : Pop Code

```
while(temp.next):  
    pre = temp  
    temp = temp.next
```



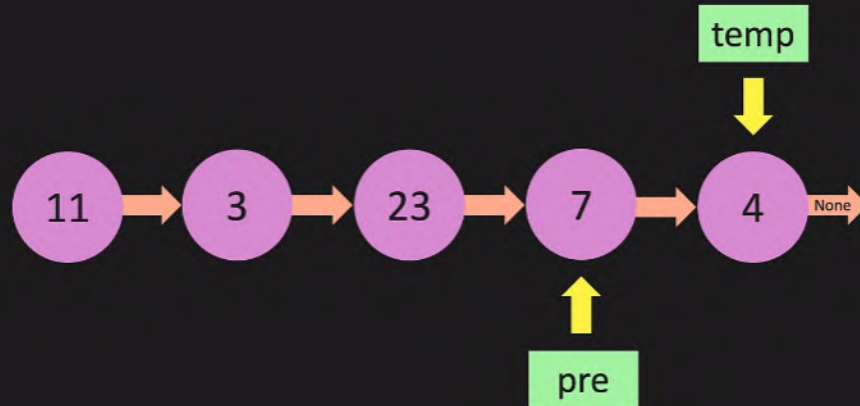
Linked List : Pop Code

```
while(temp.next):  
    pre = temp  
    temp = temp.next
```



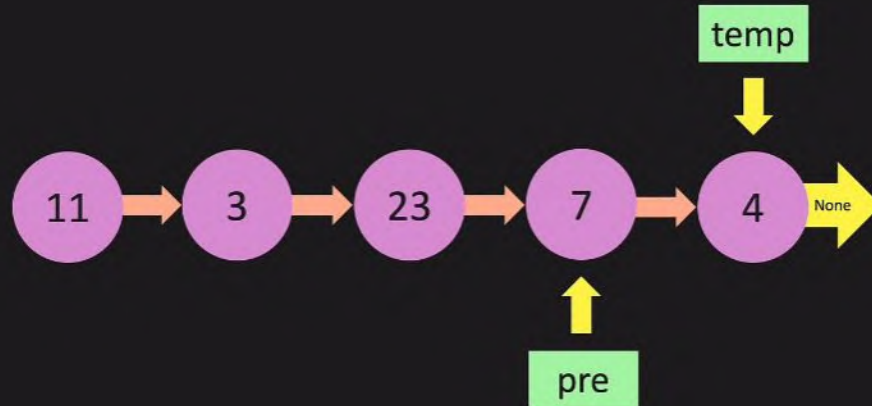
Linked List : Pop Code

```
while(temp.next):  
    pre = temp  
    temp = temp.next
```



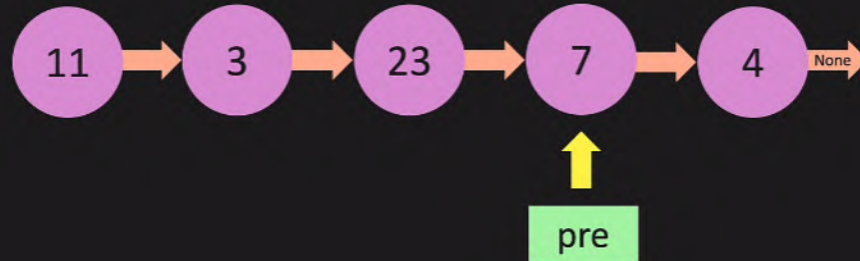
Linked List : Pop Code

```
while(temp.next):  
    pre = temp  
    temp = temp.next
```



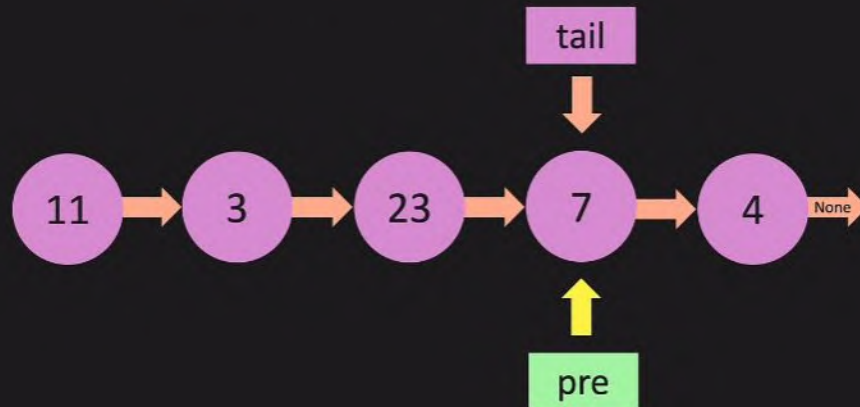
Linked List : Pop Code

```
while(temp.next):  
    pre = temp  
    temp = temp.next  
self.tail = pre
```



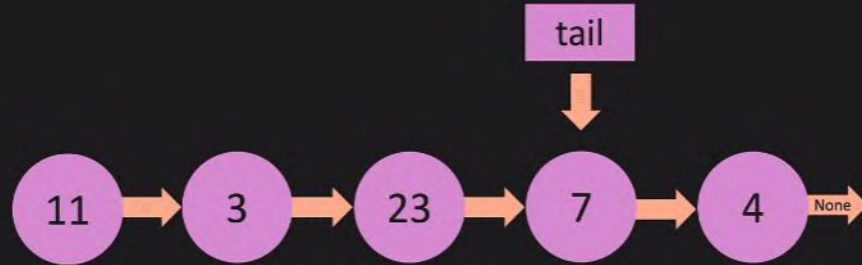
Linked List : Pop Code

```
while(temp.next):  
    pre = temp  
    temp = temp.next  
self.tail = pre
```



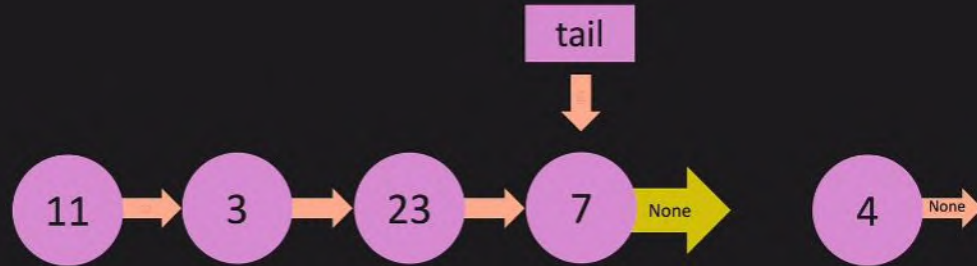
Linked List : Pop Code

```
while(temp.next):  
    pre = temp  
    temp = temp.next  
self.tail = pre  
self.tail.next = None
```



Linked List : Pop Code

```
while(temp.next):  
    pre = temp  
    temp = temp.next  
self.tail = pre  
self.tail.next = None
```



Linked List : Pop Code

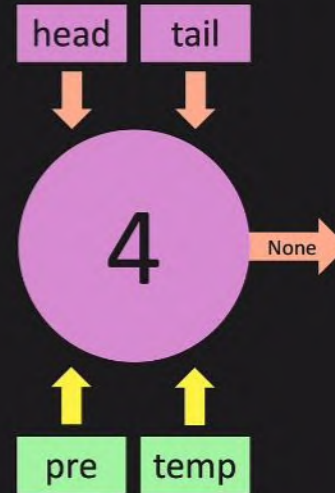
```
def pop(self):  
    if self.length == 0:  
        return None  
    temp = self.head  
    pre = self.head  
    while(temp.next):  
        pre = temp  
        temp = temp.next  
    self.tail = pre  
    self.tail.next = None
```

Linked List : Pop Code

```
def pop(self):  
    if self.length == 0:  
        return None  
    temp = self.head  
    pre = self.head  
    while(temp.next):  
        pre = temp  
        temp = temp.next  
    self.tail = pre  
    self.tail.next = None  
    self.length -= 1
```

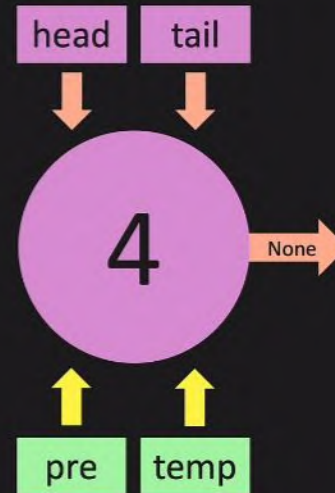
Linked List : Pop Code

```
def pop(self):  
    if self.length == 0:  
        return None  
    temp = self.head  
    pre = self.head  
    while(temp.next):  
        pre = temp  
        temp = temp.next  
    self.tail = pre  
    self.tail.next = None  
    self.length -= 1
```



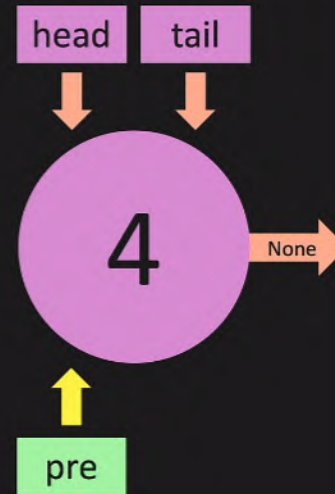
Linked List : Pop Code

```
while(temp.next):  
    pre = temp  
    temp = temp.next  
self.tail = pre  
self.tail.next = None  
self.length -= 1
```



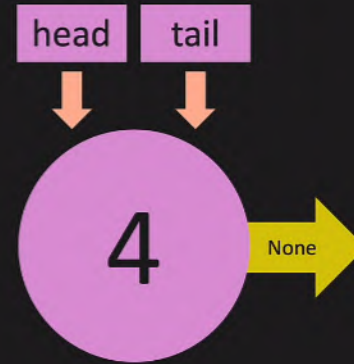
Linked List : Pop Code

```
self.tail = pre  
self.tail.next = None  
self.length -= 1
```



Linked List : Pop Code

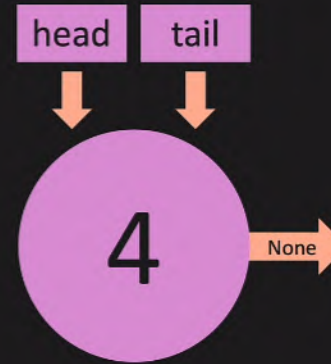
```
self.tail.next = None  
self.length -= 1
```



Linked List : Pop Code

```
self.length -= 1
```

1

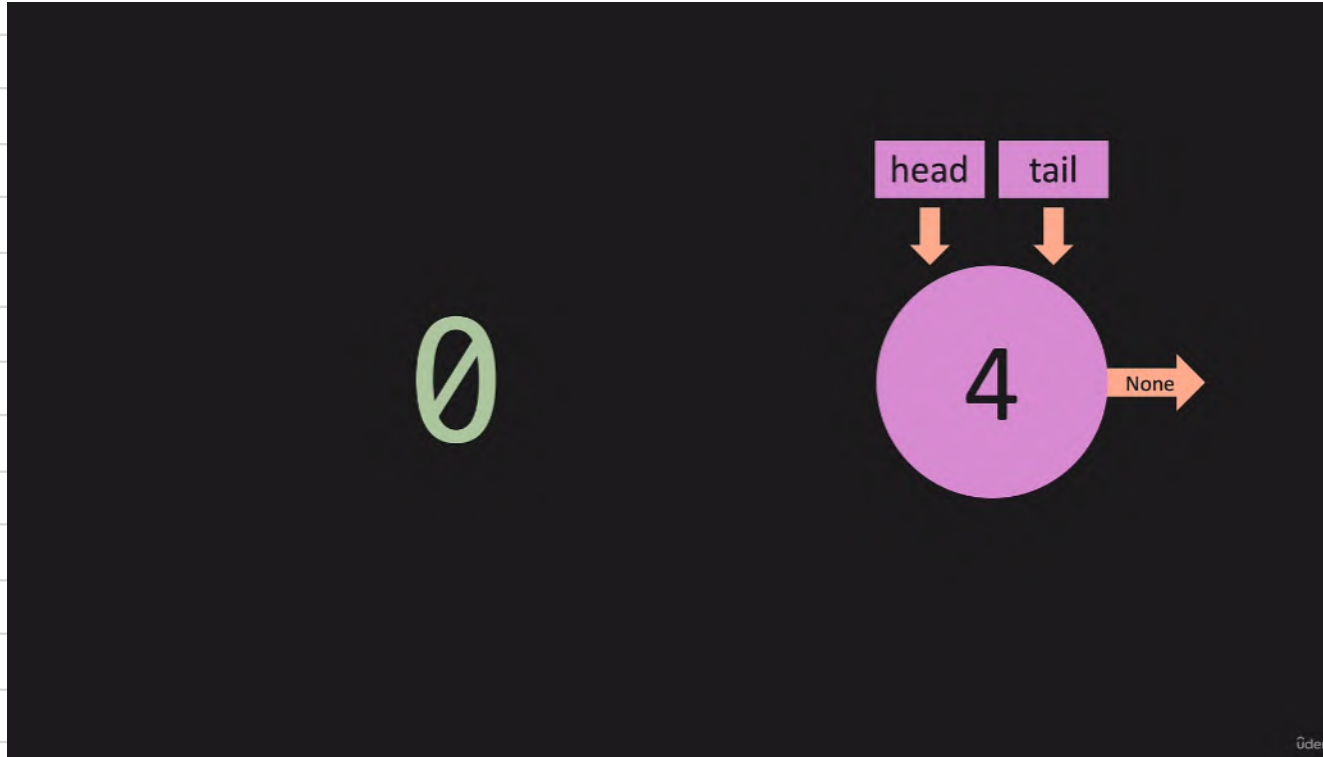


Linked List : Pop Code

Kasus #2

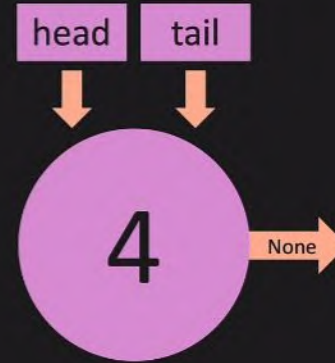
Kondisi menjadi tidak valid

Length = 0 padahal ada 1 node



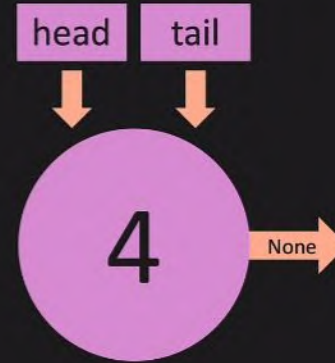
Linked List : Pop Code

```
def pop(self):  
    if self.length == 0:  
        return None  
    temp = self.head  
    pre = self.head  
    while(temp.next):  
        pre = temp  
        temp = temp.next  
    self.tail = pre  
    self.tail.next = None  
    self.length -= 1  
    if self.length == 0:
```



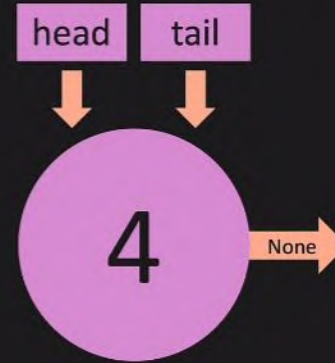
Linked List : Pop Code

```
def pop(self):  
    if self.length == 0:  
        return None  
    temp = self.head  
    pre = self.head  
    while(temp.next):  
        pre = temp  
        temp = temp.next  
    self.tail = pre  
    self.tail.next = None  
    self.length -= 1  
    if self.length == 0:
```



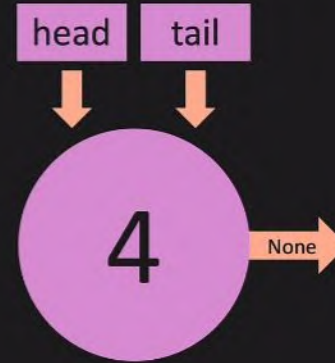
Linked List : Pop Code

```
def pop(self):  
    if self.length == 0:  
        return None  
    temp = self.head  
    pre = self.head  
    while(temp.next):  
        pre = temp  
        temp = temp.next  
    self.tail = pre  
    self.tail.next = None  
    self.length -= 1  
    if self.length == 0:
```



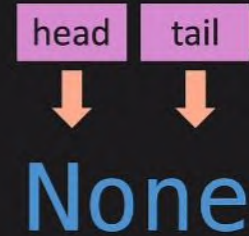
Linked List : Pop Code

```
def pop(self):  
    if self.length == 0:  
        return None  
    temp = self.head  
    pre = self.head  
    while(temp.next):  
        pre = temp  
        temp = temp.next  
    self.tail = pre  
    self.tail.next = None  
    self.length -= 1  
    if self.length == 0:  
        self.head = None  
        self.tail = None
```



Linked List : Pop Code

```
def pop(self):  
    if self.length == 0:  
        return None  
    temp = self.head  
    pre = self.head  
    while(temp.next):  
        pre = temp  
        temp = temp.next  
    self.tail = pre  
    self.tail.next = None  
    self.length -= 1  
    if self.length == 0:  
        self.head = None  
        self.tail = None
```



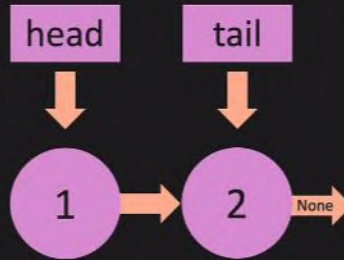
Linked List : Pop Code

```
def pop(self):
    if self.length == 0:
        return None
    temp = self.head
    pre = self.head
    while(temp.next):
        pre = temp
        temp = temp.next
    self.tail = pre
    self.tail.next = None
    self.length -= 1
    if self.length == 0:
        self.head = None
        self.tail = None
    return temp
```

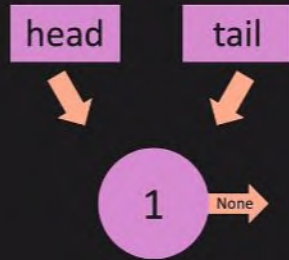
Linked List : Pop Code

```
def pop(self):  
    if self.length == 0:  
        return None  
    temp = self.head  
    pre = self.head  
    while(temp.next):  
        pre = temp  
        temp = temp.next  
    self.tail = pre  
    self.tail.next = None  
    self.length -= 1  
    if self.length == 0:  
        self.head = None  
        self.tail = None  
    return temp
```

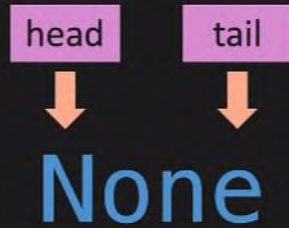
Linked List : Pop Skenario Percobaan



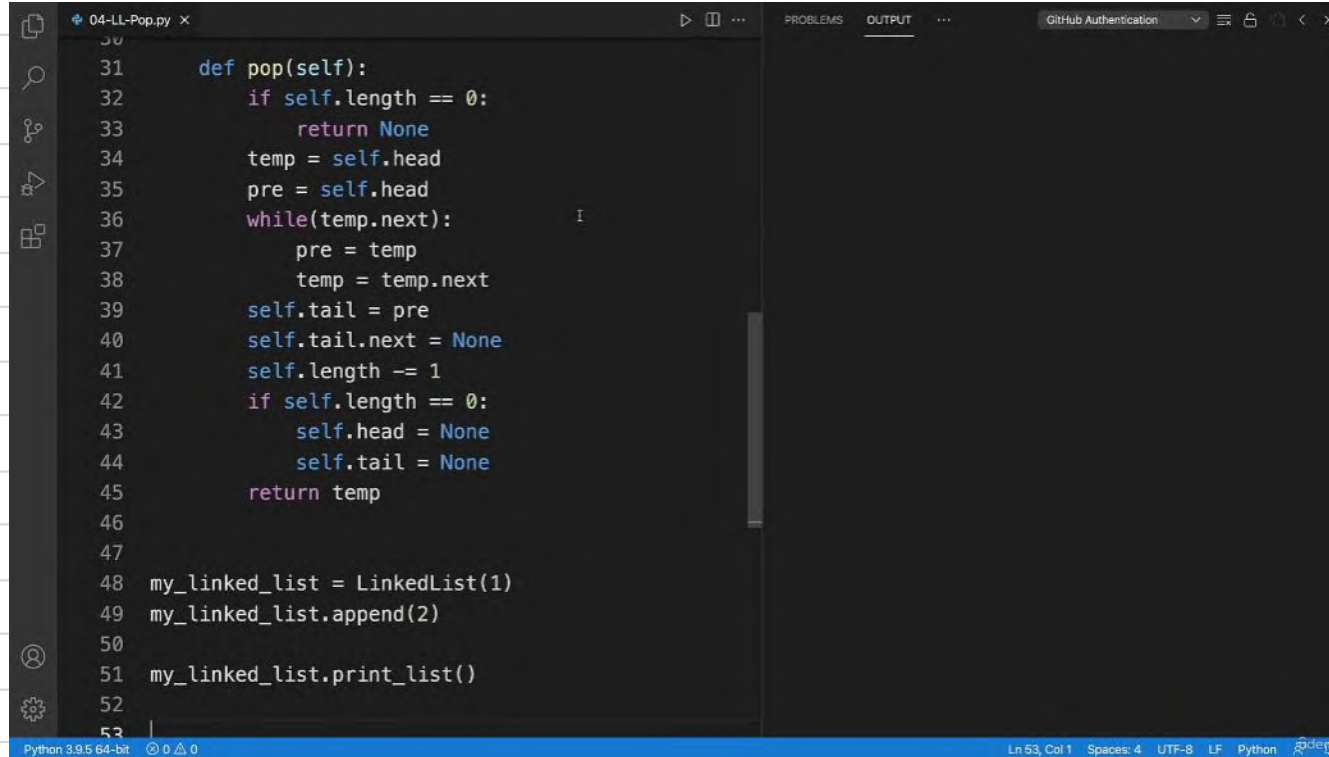
Linked List : Pop Skenario Percobaan



Linked List : Pop Skenario Percobaan



Linked List : Pop Code

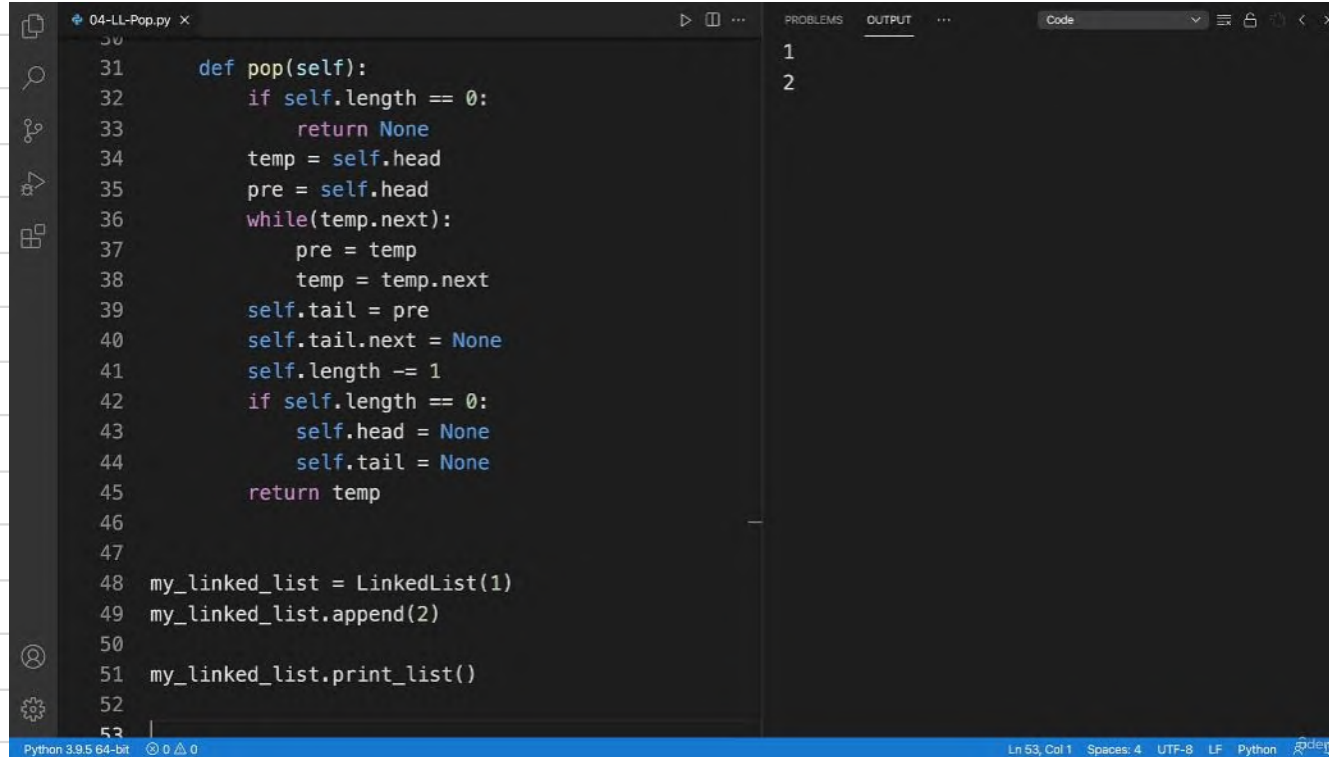


The image shows a code editor window with a dark theme. The file name is '04-LL-Pop.py'. The code is written in Python and implements a 'pop' method for a linked list. The method removes the last node from the list and returns it. The code is as follows:

```
31 def pop(self):
32     if self.length == 0:
33         return None
34     temp = self.head
35     pre = self.head
36     while(temp.next):
37         pre = temp
38         temp = temp.next
39     self.tail = pre
40     self.tail.next = None
41     self.length -= 1
42     if self.length == 0:
43         self.head = None
44         self.tail = None
45     return temp
46
47
48 my_linked_list = LinkedList(1)
49 my_linked_list.append(2)
50
51 my_linked_list.print_list()
52
53
```

The editor interface includes a sidebar on the left with icons for Explorer, Search, Source Control, Run and Debug, and Extensions. The top right shows 'PROBLEMS' and 'OUTPUT' tabs, and a 'GitHub Authentication' button. The bottom status bar indicates 'Python 3.9.5 64-bit', '0 0 0' errors/warnings/hints, and 'Ln 53, Col 1 Spaces: 4 UTF-8 LF Python'.

Linked List : Pop Code



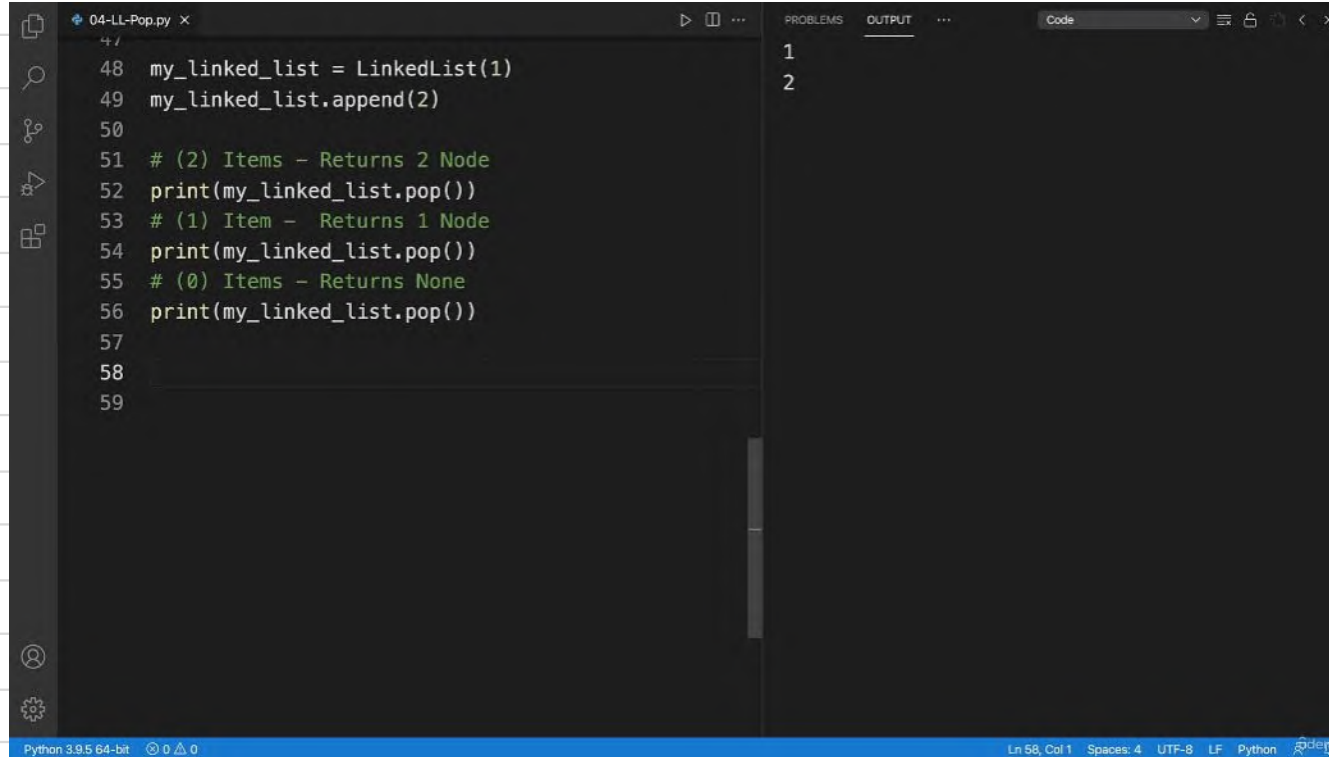
```
04-LL-Pop.py x
30
31 def pop(self):
32     if self.length == 0:
33         return None
34     temp = self.head
35     pre = self.head
36     while(temp.next):
37         pre = temp
38         temp = temp.next
39     self.tail = pre
40     self.tail.next = None
41     self.length -= 1
42     if self.length == 0:
43         self.head = None
44         self.tail = None
45     return temp
46
47
48 my_linked_list = LinkedList(1)
49 my_linked_list.append(2)
50
51 my_linked_list.print_list()
52
53
```

PROBLEMS OUTPUT ... Code

1
2

Python 3.9.5 64-bit 0 0 0 Ln 53, Col 1 Spaces: 4 UTF-8 LF Python

Linked List : Pop Code



The screenshot shows a code editor with a file named '04-LL-Pop.py'. The code defines a linked list and performs pop operations. The output pane on the right shows the results of these operations.

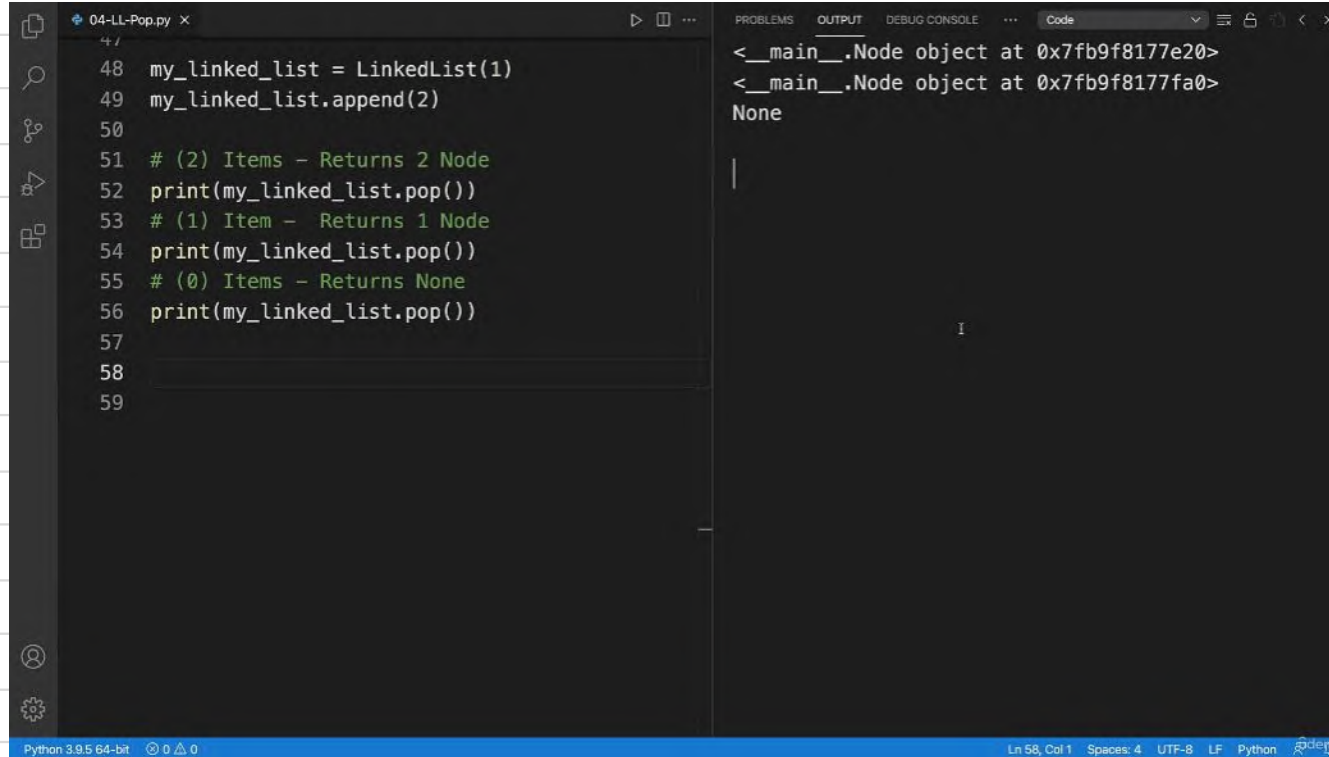
```
48 my_linked_list = LinkedList(1)
49 my_linked_list.append(2)
50
51 # (2) Items - Returns 2 Node
52 print(my_linked_list.pop())
53 # (1) Item - Returns 1 Node
54 print(my_linked_list.pop())
55 # (0) Items - Returns None
56 print(my_linked_list.pop())
57
58
59
```

Output:

```
1
2
```

Python 3.9.5 64-bit | 0 0 0 | Ln 58, Col 1 | Spaces: 4 | UTF-8 | LF | Python | ideone

Linked List : Pop Code



The image shows a screenshot of a Visual Studio Code editor window. The editor is open to a file named '04-LL-Pop.py'. The code in the editor is as follows:

```
48 my_linked_list = LinkedList(1)
49 my_linked_list.append(2)
50
51 # (2) Items - Returns 2 Node
52 print(my_linked_list.pop())
53 # (1) Item - Returns 1 Node
54 print(my_linked_list.pop())
55 # (0) Items - Returns None
56 print(my_linked_list.pop())
57
58
59
```

The output panel on the right shows the results of the execution:

```
<__main__.Node object at 0x7fb9f8177e20>
<__main__.Node object at 0x7fb9f8177fa0>
None
```

The status bar at the bottom indicates the editor is using Python 3.9.5 64-bit, and the current cursor position is Line 58, Column 1.

Linked List : Pop Code

```
self.tail = pre
self.tail.next = None
self.length -= 1
if self.length == 0:
    self.head = None
    self.tail = None
return temp.value
```

```
my_linked_list = LinkedList(1)
my_linked_list.append(2)
```

```
# (2) Items - Returns 2 Node
print(my_linked_list.pop())
# (1) Item - Returns 1 Node
print(my_linked_list.pop())
# (0) Items - Returns None
```

```
<__main__.Node object at 0x0000000000000000>
<__main__.Node object at 0x0000000000000000>
None
```

```
|
```

```
I
```

Linked List : Pop Code

```
self.tail = pre
self.tail.next = None
self.length -= 1
if self.length == 0:
    self.head = None
    self.tail = None
return temp.value
```

```
my_linked_list = LinkedList(1)
my_linked_list.append(2)
```

```
# (2) Items - Returns 2 Node
print(my_linked_list.pop())
# (1) Item - Returns 1 Node
print(my_linked_list.pop())
# (0) Items - Returns None
```

2
1
None

Terima Kasih



Ada Pertanyaan?