



Struktur Data

Meet 05



08

Linked List Prepend

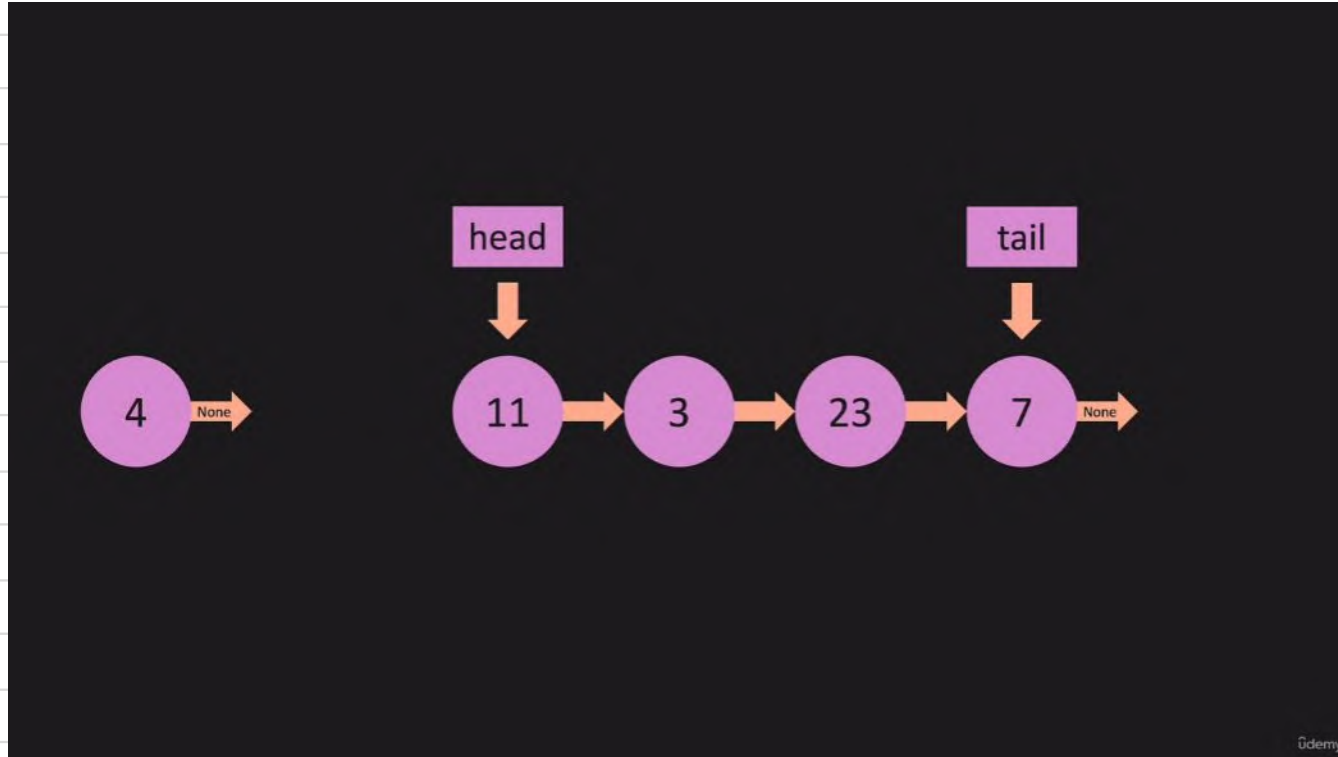


Linked List : Prepend

- Apa itu Prepend...?

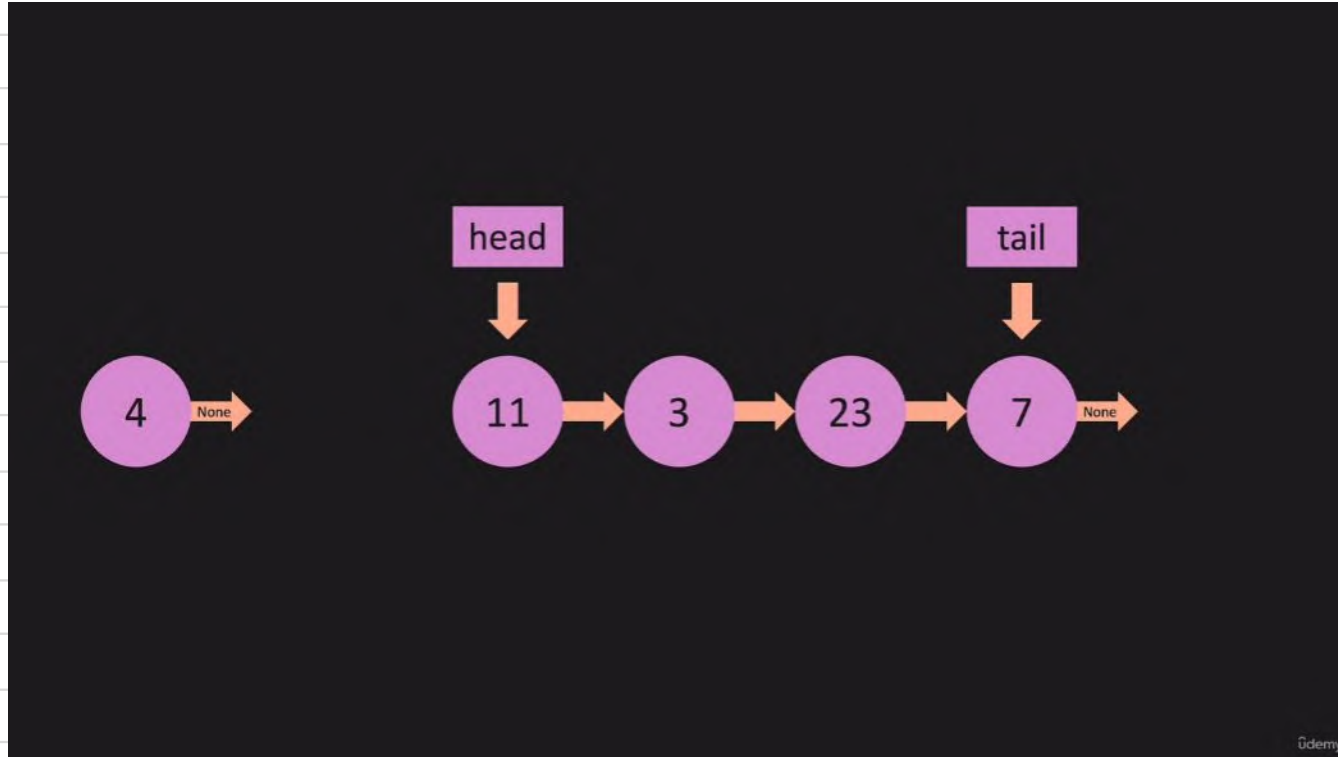
Diketahui suatu list

Linked List : Prepend



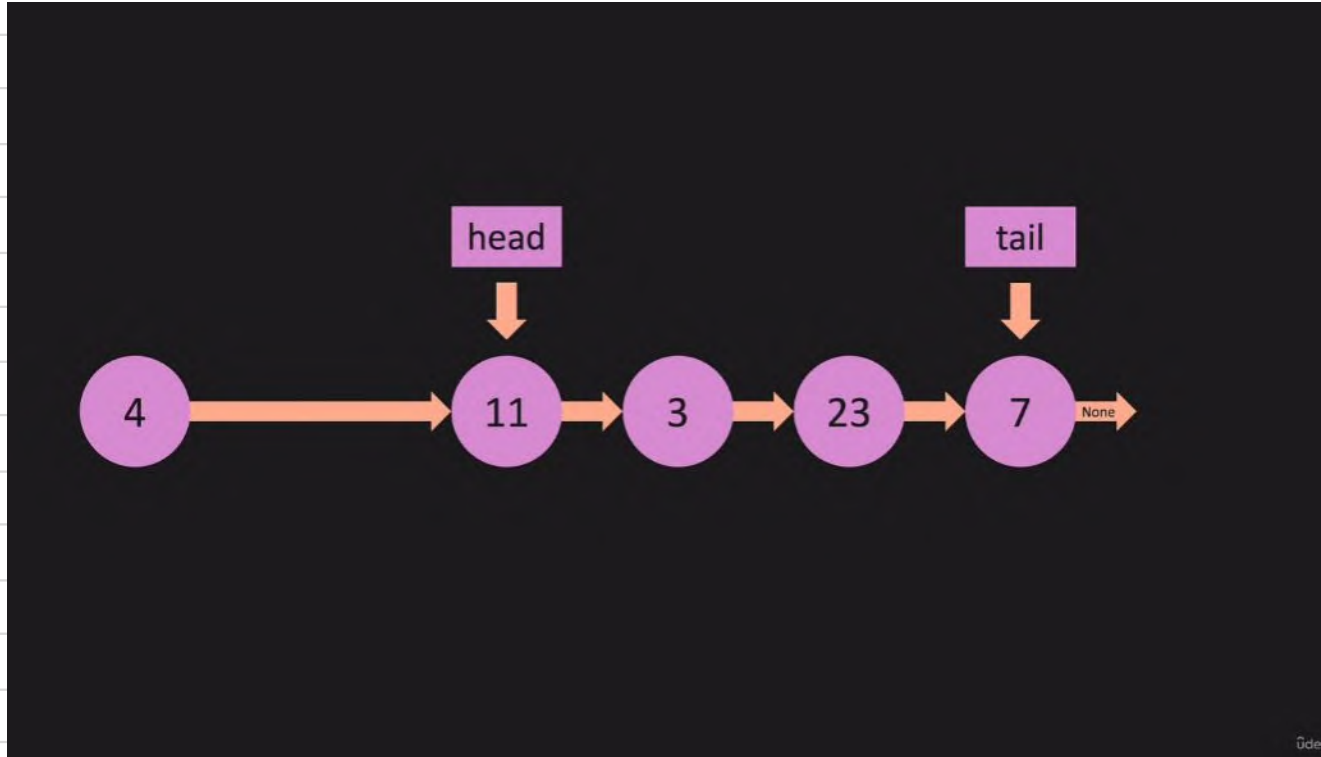
Linked List : Prepend

Bagaimana jika ingin
menambahkan node 4
Di depan linked list?



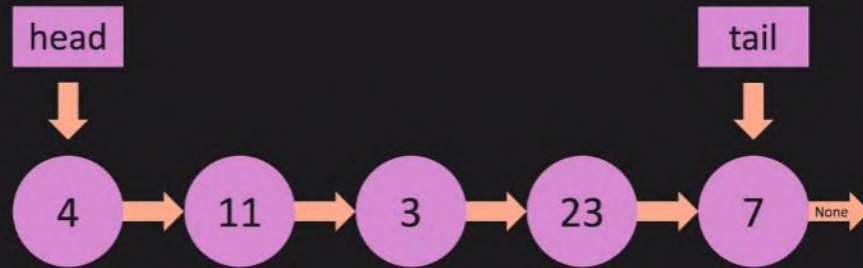
Linked List : Prepend

Bagaimana jika ingin
menambahkan node 4
di depan linked list?



Linked List : Prepend

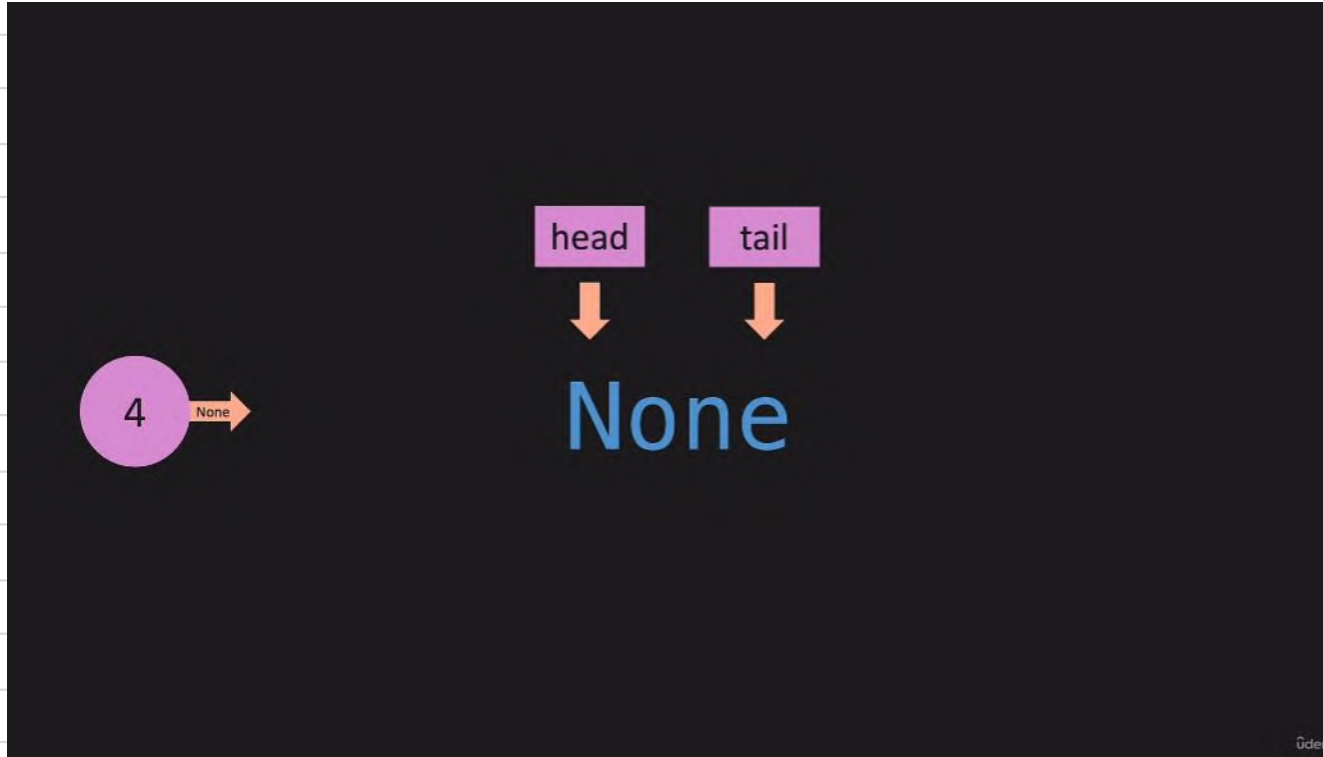
Bagaimana jika ingin
menambahkan node 4
di depan linked list?



Linked List : Prepend

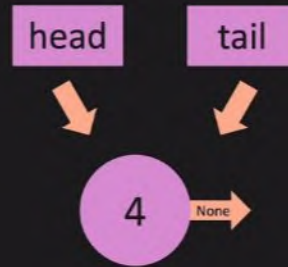
- Namun ada kondisi yang harus mendapatkan perhatian khusus

Linked List : Prepend



Linked List : Prepend

Kasus #1
Seharusnya...

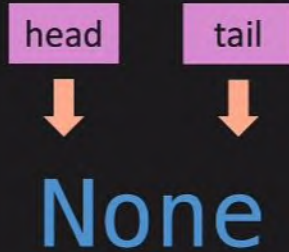


Linked List : Prepend Code

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

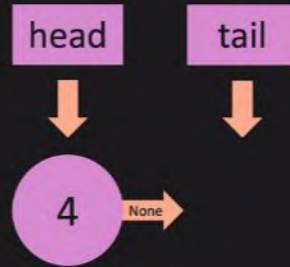


Linked List : Prepend Code



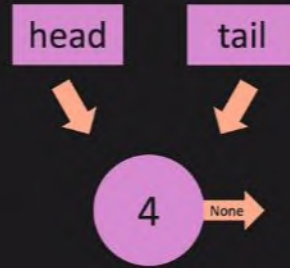
```
if self.length == 0:
```

Linked List : Prepend Code



```
if self.length == 0:  
    self.head = new_node
```

Linked List : Prepend Code



```
if self.length == 0:  
    self.head = new_node  
    self.tail = new_node
```

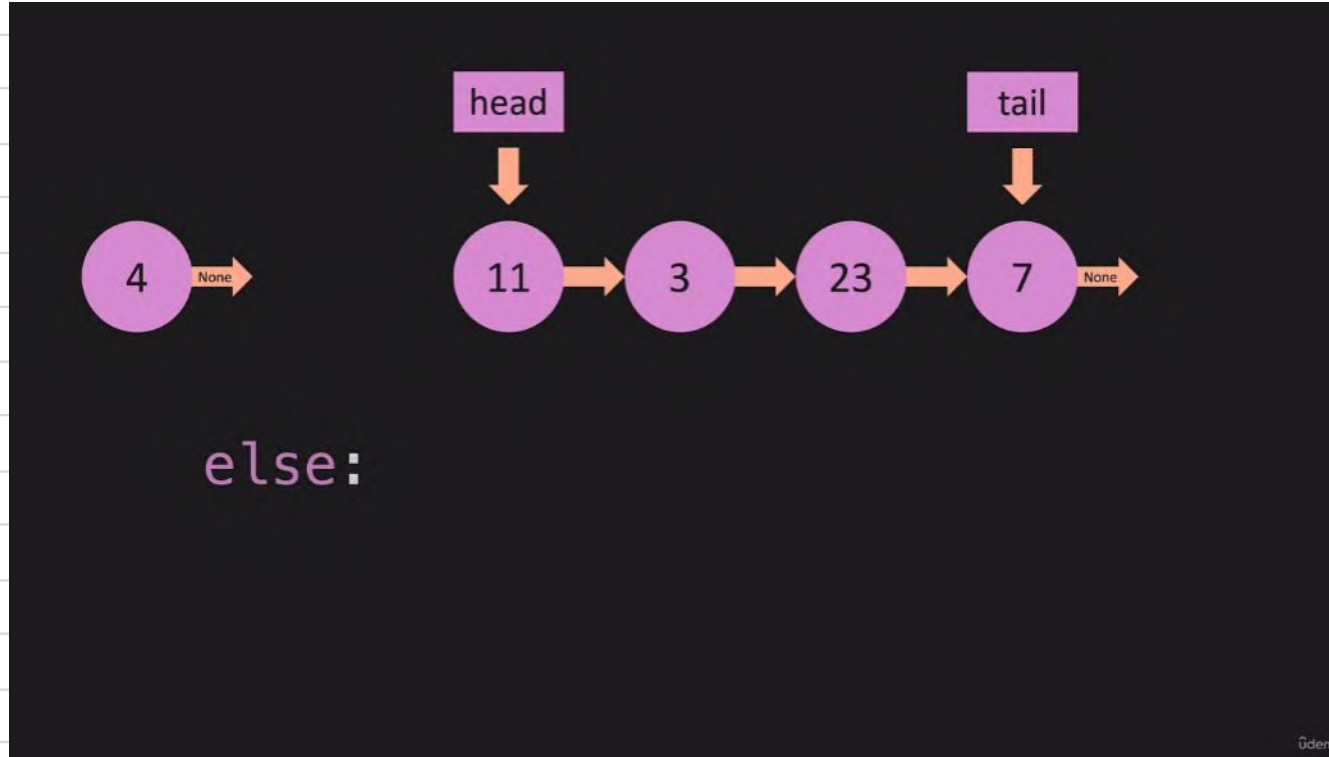
Linked List : Prepend Code

```
def prepend(self, value):  
    new_node = Node(value)  
    if self.length == 0:  
        self.head = new_node  
        self.tail = new_node
```

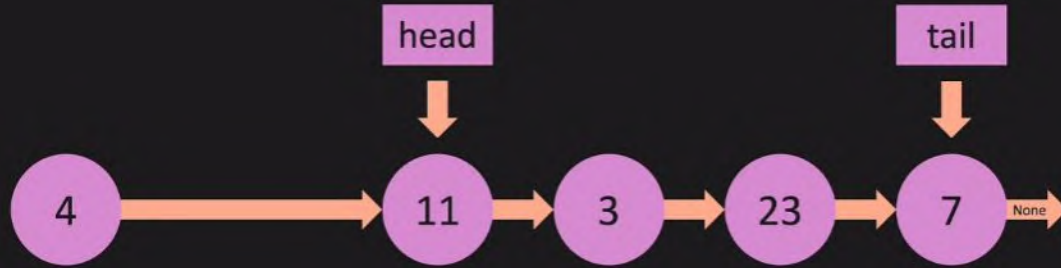
Linked List : Prepend Code

```
def prepend(self, value):  
    new_node = Node(value)  
    if self.length == 0:  
        self.head = new_node  
        self.tail = new_node  
    else:
```


Linked List : Prepend Code



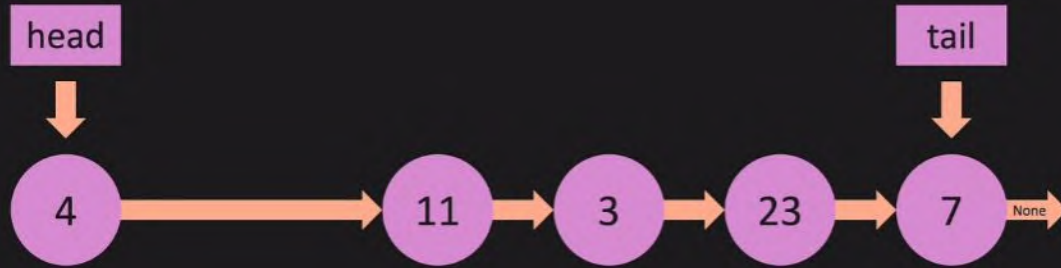
Linked List : Prepend Code



```
else:
```

```
    new_node.next = self.head
```

Linked List : Prepend Code



else:

```
new_node.next = self.head  
self.head = new_node
```

Linked List : Prepend Code

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

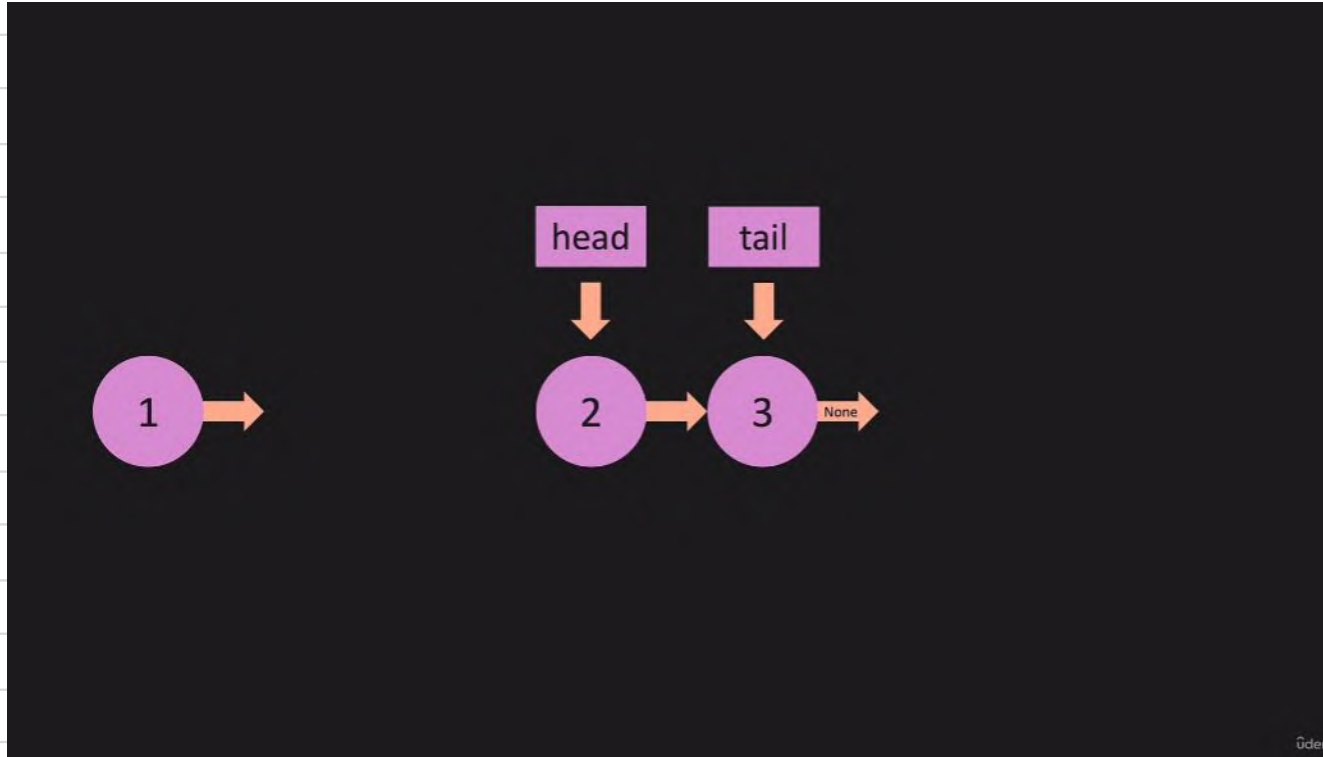
Linked List : Prepend Code

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

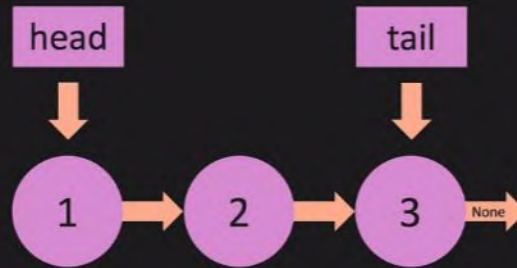
Linked List : Prepend Code

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

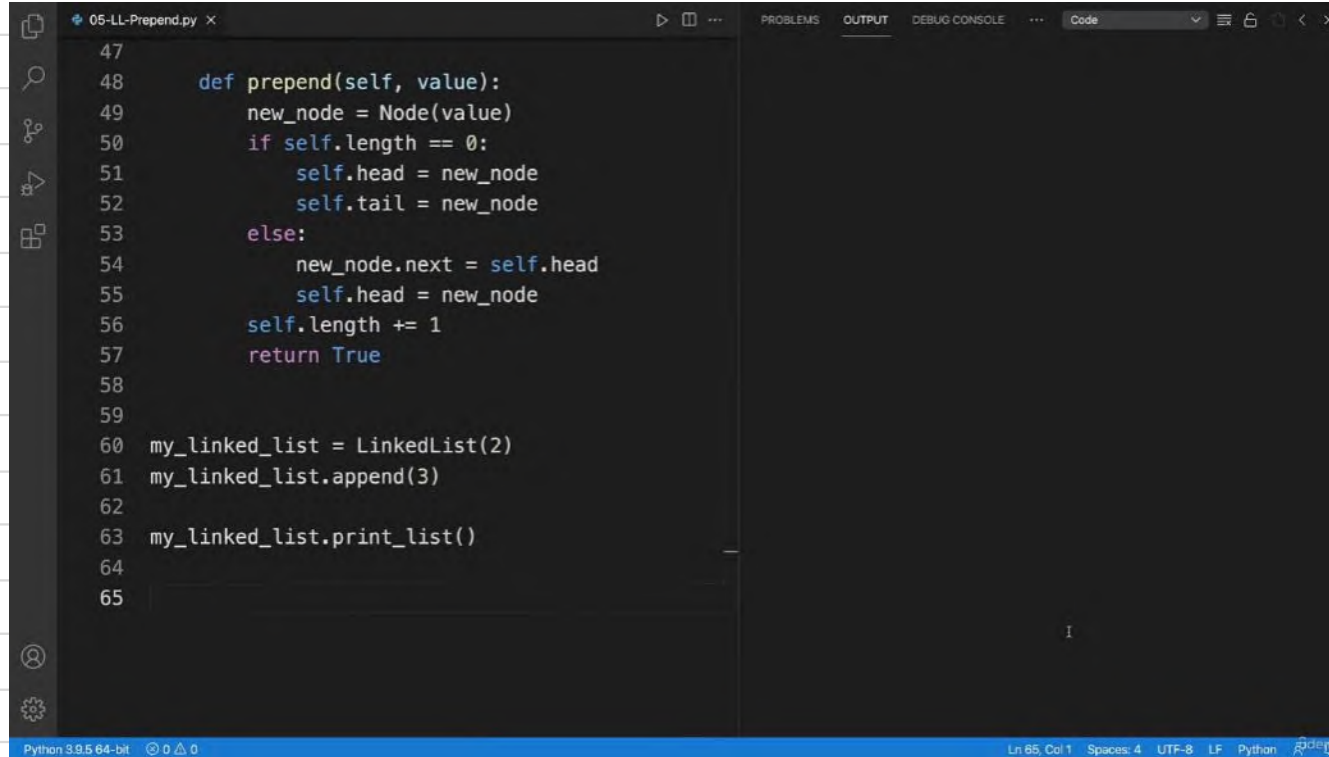
Linked List : Prepend Skenario Percobaan



Linked List : Prepend Skenario Percobaan



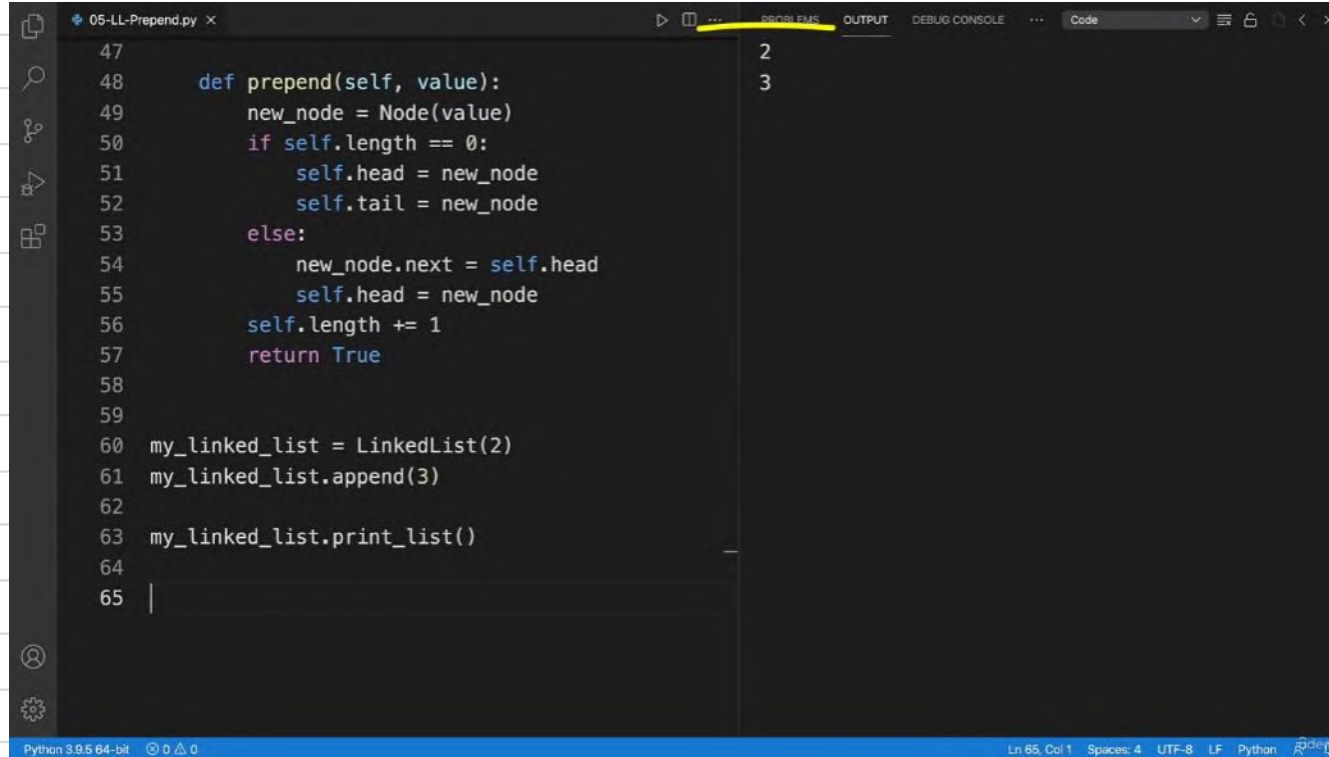
Linked List : Prepend Skenario Percobaan



The image shows a code editor window with a dark theme. The file name is '05-LL-Prepend.py'. The code is written in Python and implements a linked list with a prepend method. The editor has a sidebar on the left with icons for Explorer, Search, Source Control, Run and Debug, and Extensions. The top right has tabs for PROBLEMS, OUTPUT, DEBUG CONSOLE, and Code. The bottom status bar shows 'Python 3.9.5 64-bit', '0 0 0', and 'Ln 65, Col 1 Spaces: 4 UTF-8 LF Python'.

```
47
48     def prepend(self, value):
49         new_node = Node(value)
50         if self.length == 0:
51             self.head = new_node
52             self.tail = new_node
53         else:
54             new_node.next = self.head
55             self.head = new_node
56         self.length += 1
57         return True
58
59
60 my_linked_list = LinkedList(2)
61 my_linked_list.append(3)
62
63 my_linked_list.print_list()
64
65
```

Linked List : Prepend Skenario Percobaan



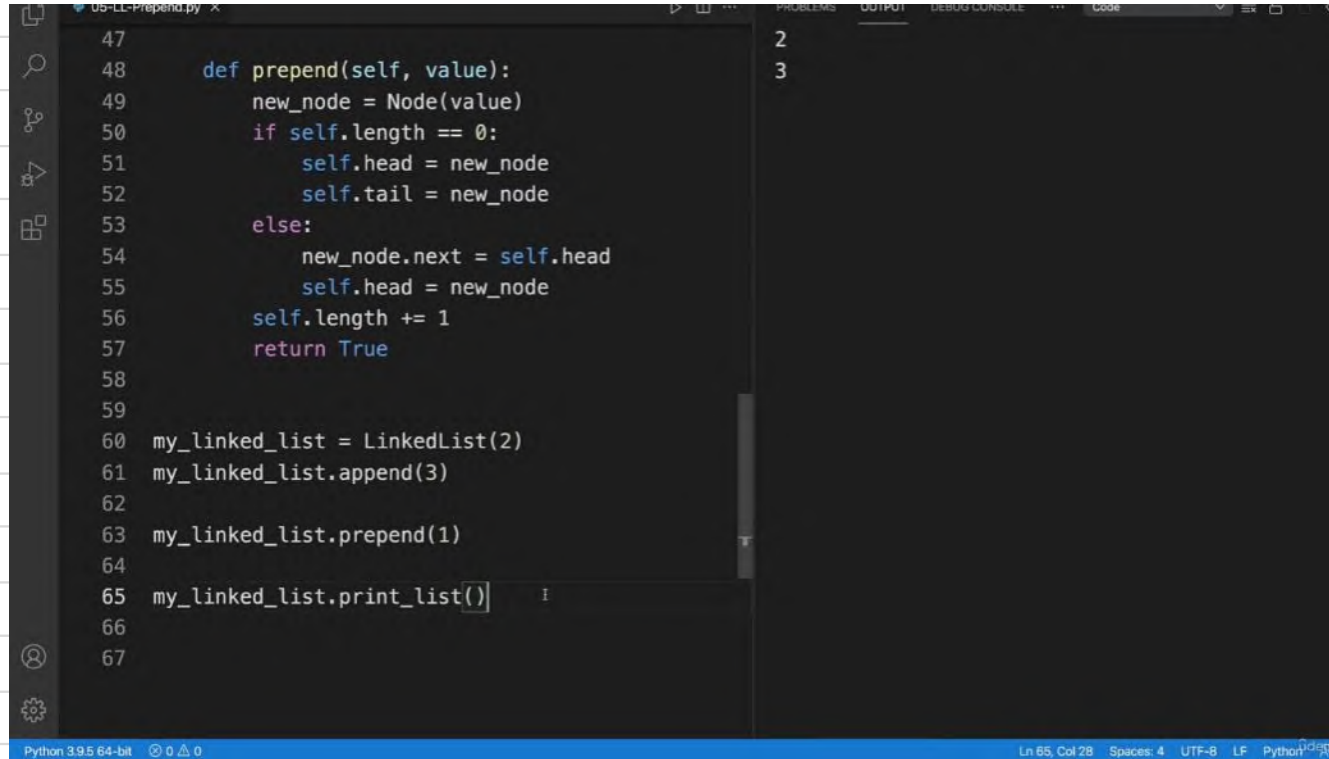
The screenshot shows a Python IDE with a file named '05-LL-Prepend.py'. The code defines a `prepend` method for a linked list. The method creates a new node with the given value. If the list is empty (`self.length == 0`), it sets both `self.head` and `self.tail` to the new node. Otherwise, it sets `new_node.next` to `self.head` and updates `self.head` to the new node. The length is incremented by 1, and the method returns `True`. Below the method definition, the code creates a `my_linked_list` object, appends the value 3, and then prints the list. The output pane on the right shows the values 2 and 3, indicating the list contains these two elements. The status bar at the bottom indicates the Python version is 3.9.5 64-bit and the current line is 65, column 1.

```
47
48     def prepend(self, value):
49         new_node = Node(value)
50         if self.length == 0:
51             self.head = new_node
52             self.tail = new_node
53         else:
54             new_node.next = self.head
55             self.head = new_node
56             self.length += 1
57         return True
58
59
60 my_linked_list = LinkedList(2)
61 my_linked_list.append(3)
62
63 my_linked_list.print_list()
64
65 |
```

2
3

Python 3.9.5 64-bit 0 0 0 Ln 65, Col 1 Spaces: 4 UTF-8 LF Python

Linked List : Prepend Skenario Percobaan



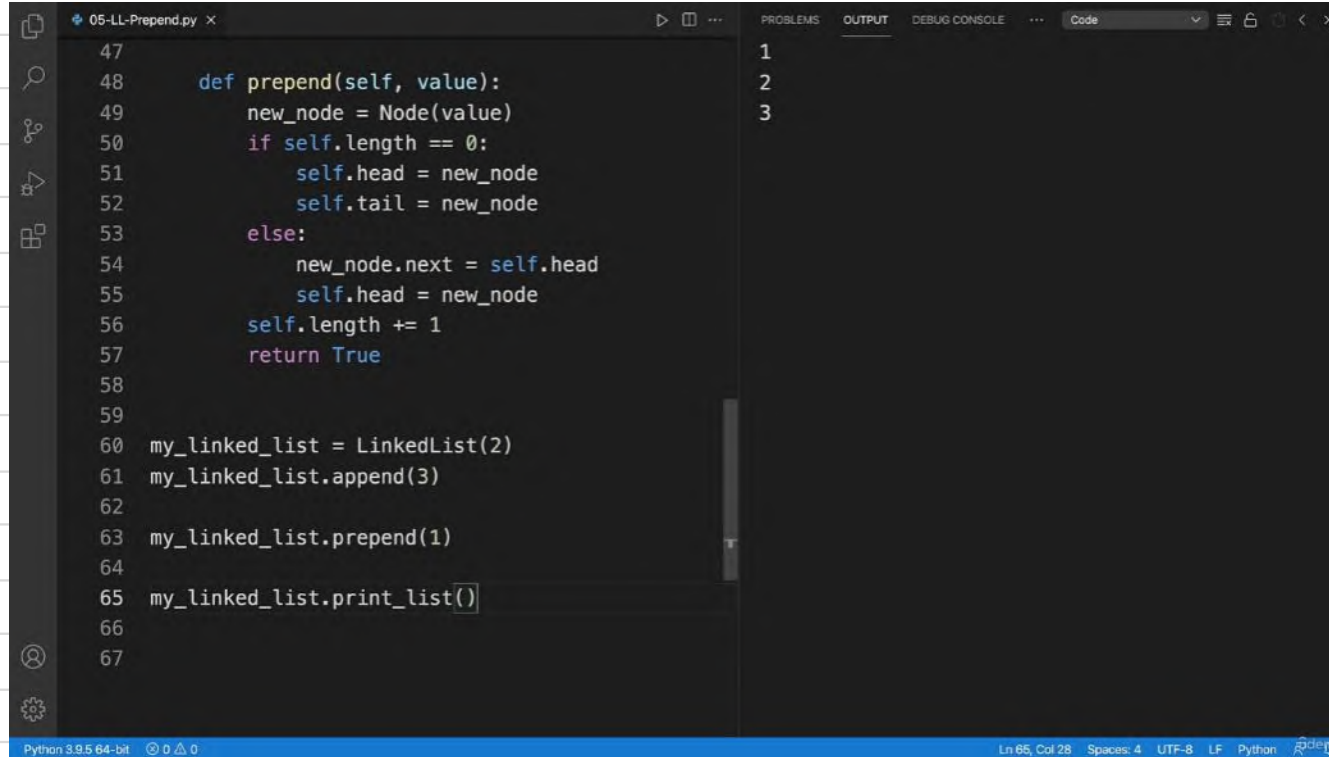
The screenshot shows a Python IDE with a file named '05-LL-Prepend.py'. The code defines a `prepend` method for a linked list, which creates a new node and inserts it at the head. The main code creates a linked list with 2 nodes, appends 3, and then prepends 1. The output pane shows the values 2 and 3, indicating the state of the list before the final prepend operation.

```
47
48     def prepend(self, value):
49         new_node = Node(value)
50         if self.length == 0:
51             self.head = new_node
52             self.tail = new_node
53         else:
54             new_node.next = self.head
55             self.head = new_node
56         self.length += 1
57         return True
58
59
60 my_linked_list = LinkedList(2)
61 my_linked_list.append(3)
62
63 my_linked_list.prepend(1)
64
65 my_linked_list.print_list()
66
67
```

2
3

Python 3.9.5 64-bit 0 0 0 Ln 65, Col 28 Spaces: 4 UTF-8 LF Python405

Linked List : Prepend Skenario Percobaan



```
05-LL-Prepend.py x
47
48     def prepend(self, value):
49         new_node = Node(value)
50         if self.length == 0:
51             self.head = new_node
52             self.tail = new_node
53         else:
54             new_node.next = self.head
55             self.head = new_node
56         self.length += 1
57         return True
58
59
60 my_linked_list = LinkedList(2)
61 my_linked_list.append(3)
62
63 my_linked_list.prepend(1)
64
65 my_linked_list.print_list()
66
67
```

1
2
3

Python 3.9.5 64-bit 0 0 0 Ln 65, Col 28 Spaces: 4 UTF-8 LF Python

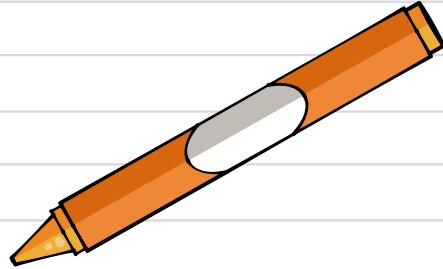


09



Linked List

Pop First

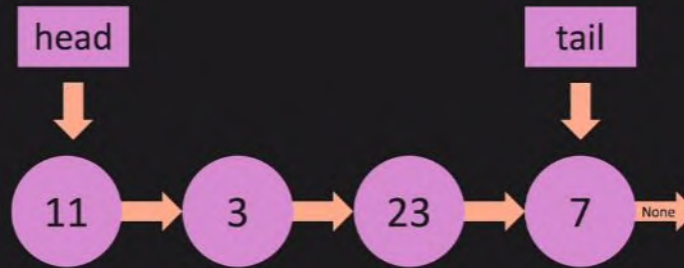


Linked List : Pop First

- Apa itu Pop First...?

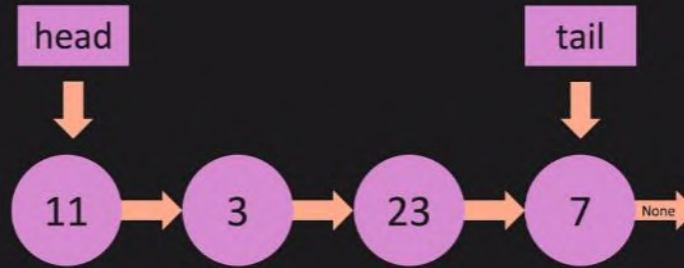
Diketahui suatu list

Linked List : Pop First



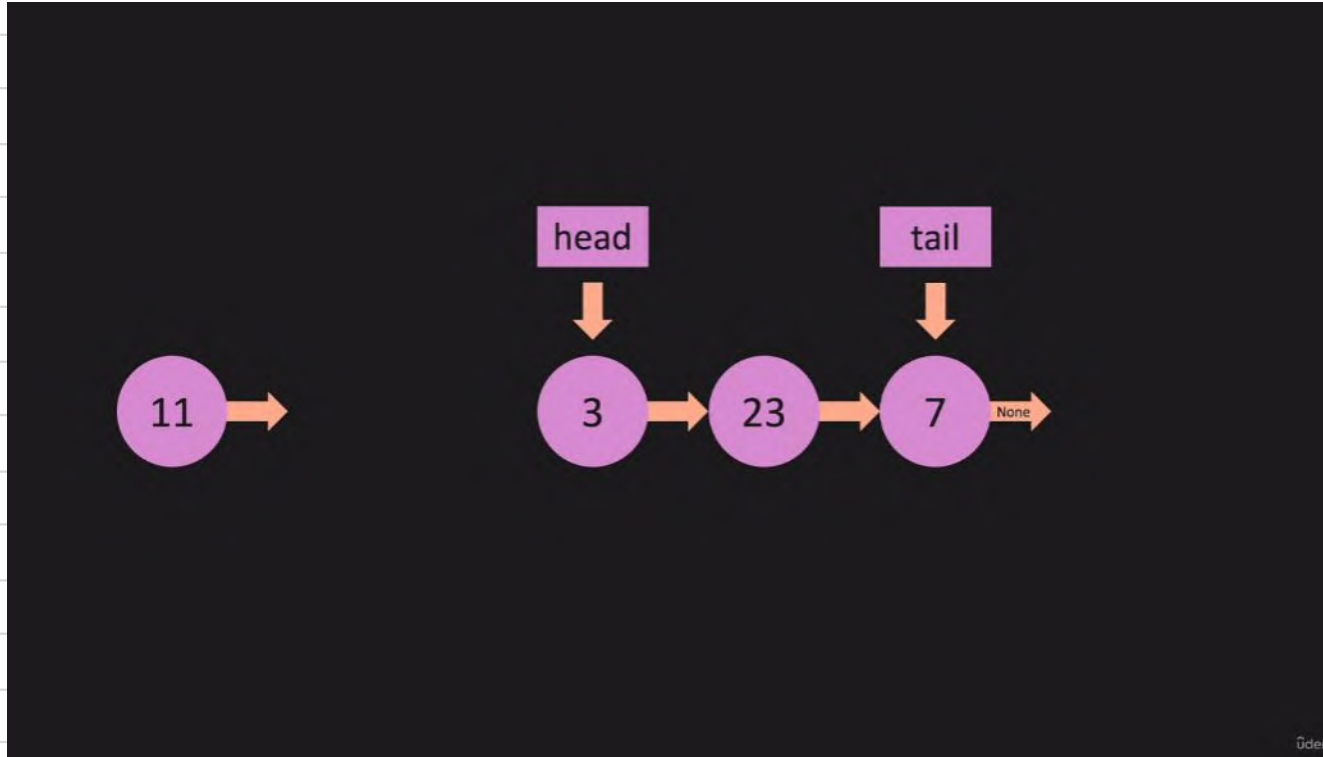
Linked List : Pop First

Bagaimana jika ingin
menghapus node
di depan linked list?



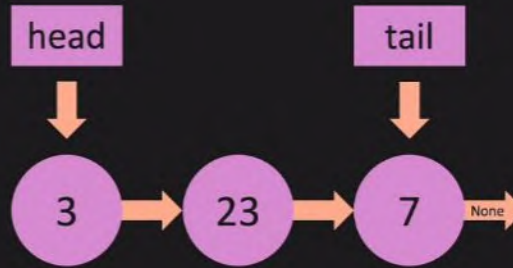
Linked List : Pop First

Bagaimana jika ingin
menghapus node
di depan linked list?



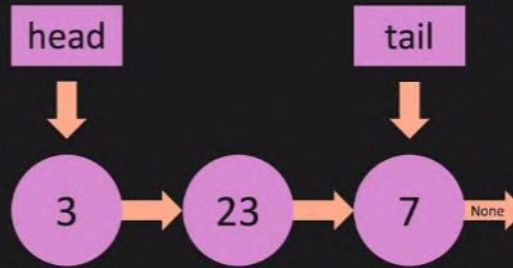
Linked List : Pop First

Bagaimana jika ingin
menghapus node
di depan linked list?



Linked List : Pop First

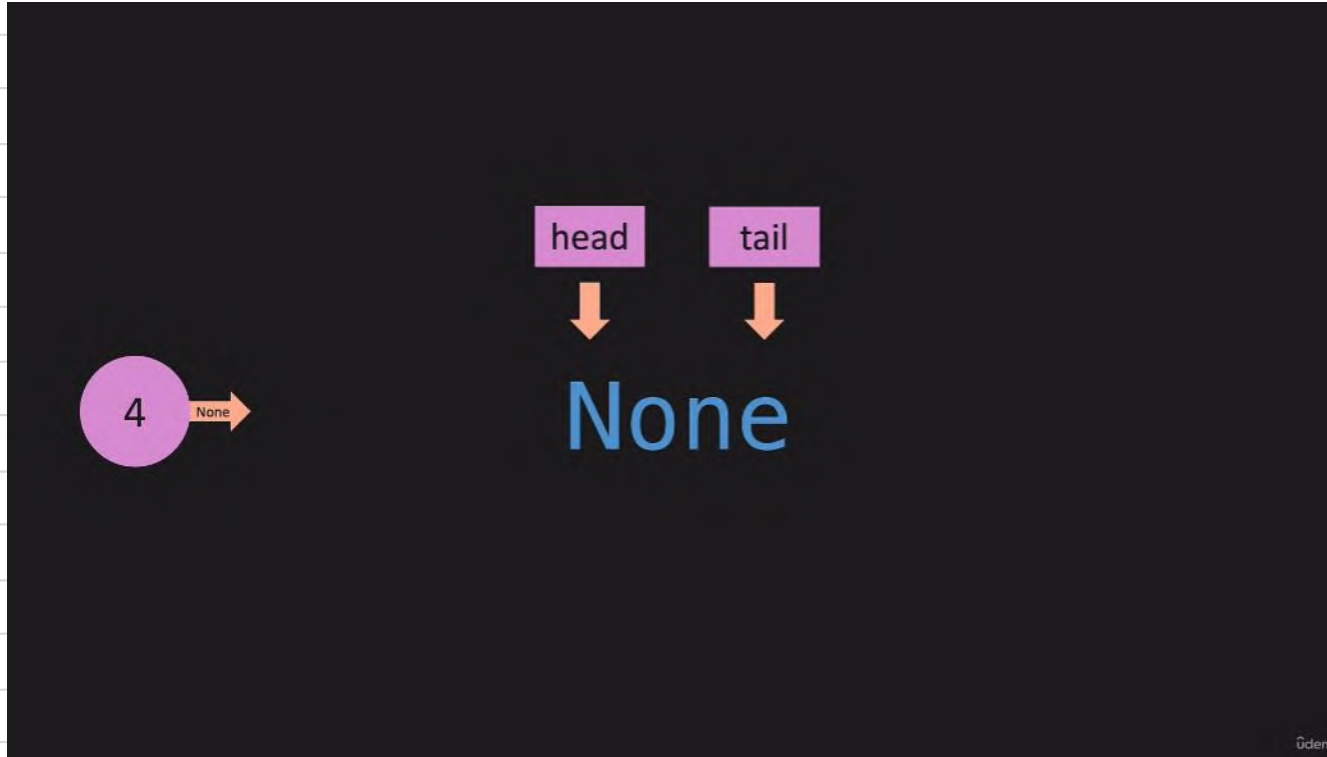
Bagaimana jika ingin
menghapus node
di depan linked list?



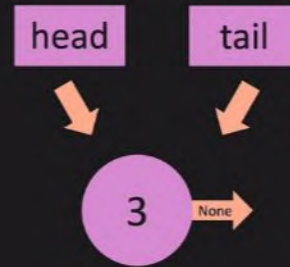
Linked List : Pop First

- Namun ada beberapa kondisi yang harus mendapatkan perhatian khusus

Linked List : Pop First

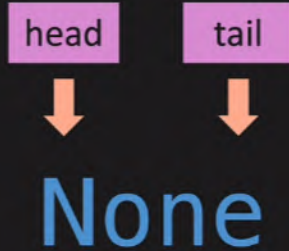


Linked List : Pop First



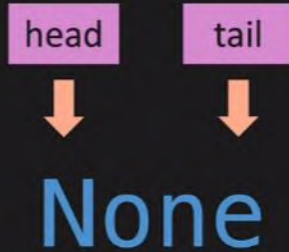
Linked List : Pop First Code

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

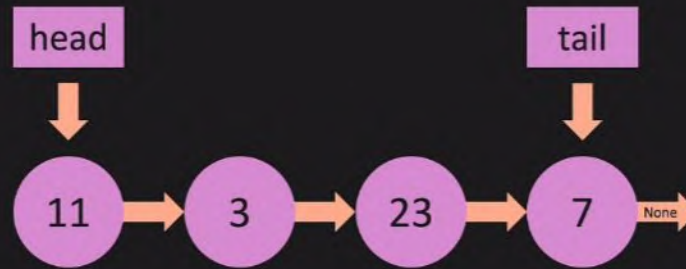


Linked List : Pop First Code

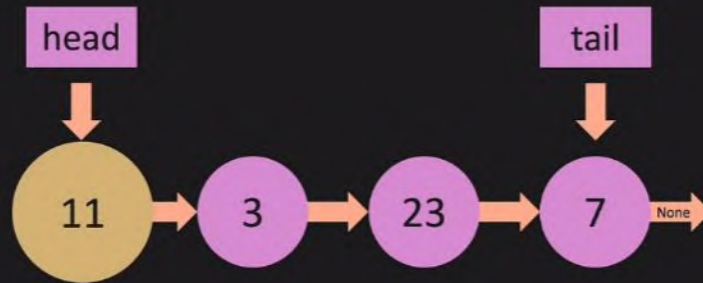
```
def pop_first(self):  
    if self.length == 0:  
        return None
```



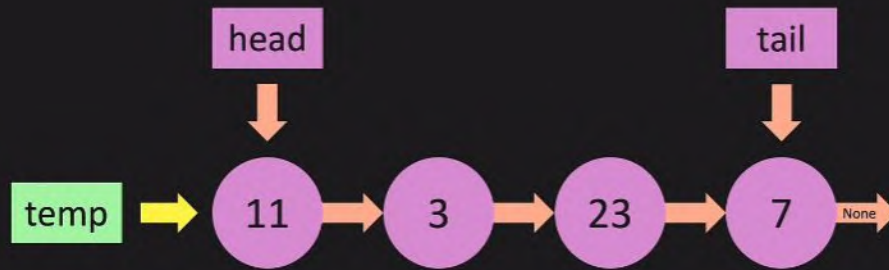
Linked List : Pop First



Linked List : Pop First

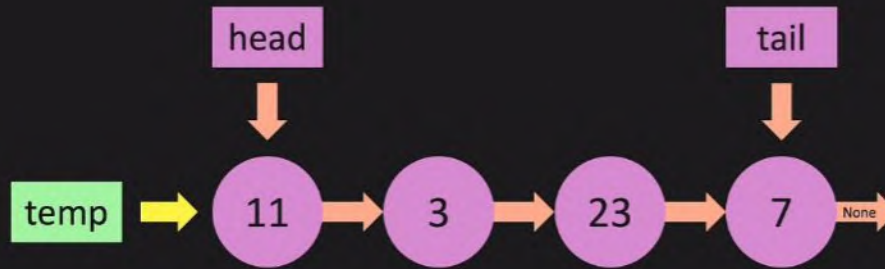


Linked List : Pop First



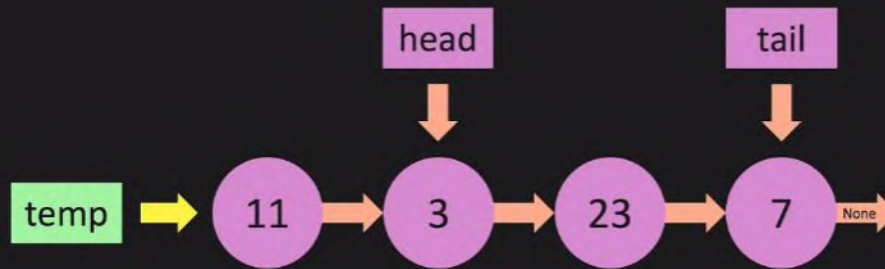
Linked List : Pop First Code

```
temp = self.head
```



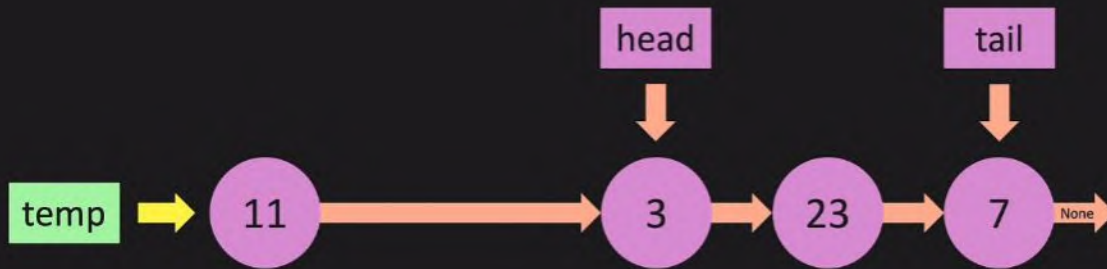
Linked List : Pop First Code

```
temp = self.head  
self.head = self.head.next
```



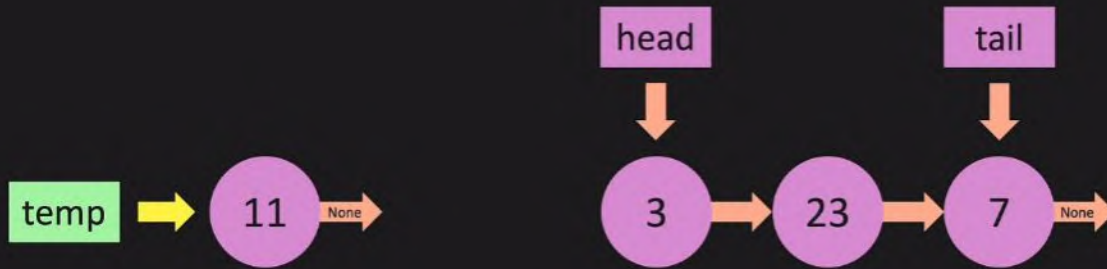
Linked List : Pop First Code

```
temp = self.head  
self.head = self.head.next
```



Linked List : Pop First Code

```
temp = self.head  
self.head = self.head.next  
temp.next = None
```

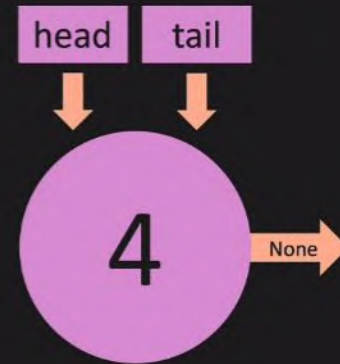


Linked List : Pop First Code

```
temp = self.head  
self.head = self.head.next  
temp.next = None  
self.length -= 1
```

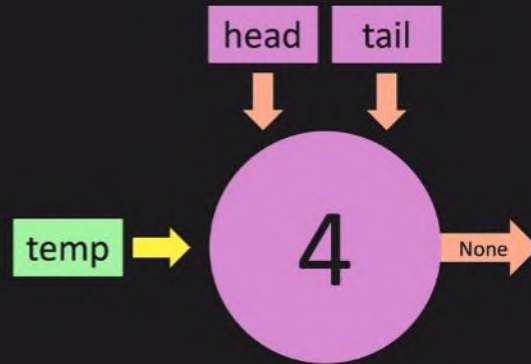

Linked List : Pop First Code

```
temp = self.head  
self.head = self.head.next  
temp.next = None  
self.length -= 1
```



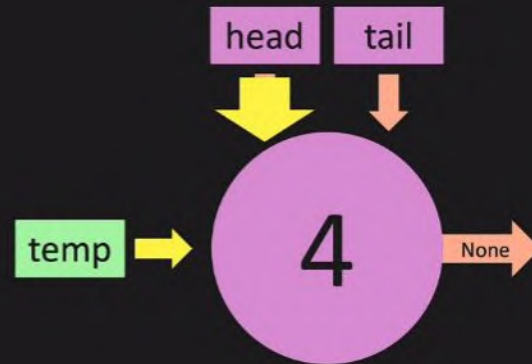
Linked List : Pop First Code

```
temp = self.head  
self.head = self.head.next  
temp.next = None  
self.length -= 1
```



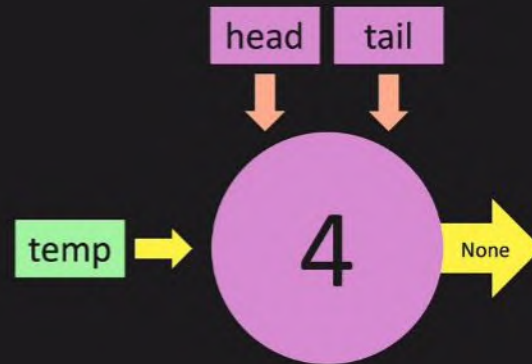
Linked List : Pop First Code

```
self.head = self.head.next  
temp.next = None  
self.length -= 1
```



Linked List : Pop First Code

```
self.head = self.head.next  
temp.next = None  
self.length -= 1
```



Linked List : Pop First Code

```
self.head = self.head.next  
temp.next = None  
self.length -= 1
```



Linked List : Pop First Code

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



Linked List : Pop First Code

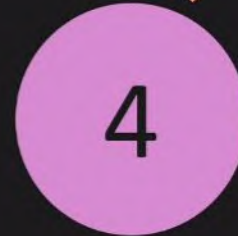
```
self.length -= 1
```

head



None

tail



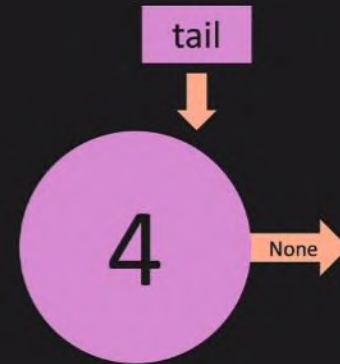
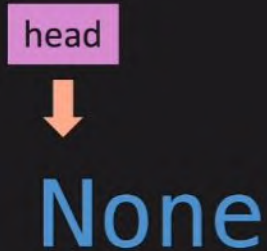
None

Linked List : Pop First Code



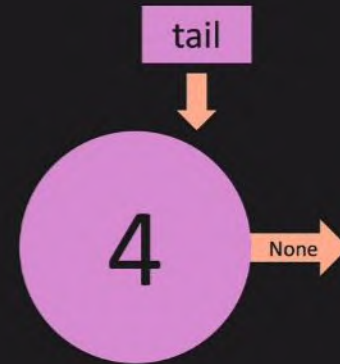
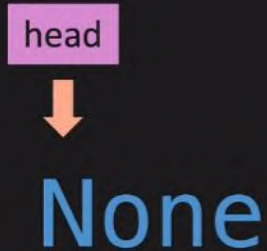
Linked List : Pop First Code

```
if self.length == 0:
```



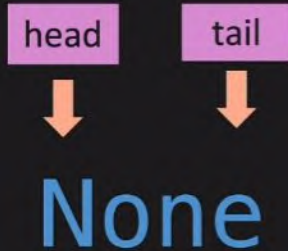
Linked List : Pop First Code

```
if self.length == 0:  
    self.tail = None
```



Linked List : Pop First Code

```
if self.length == 0:  
    self.tail = None
```



Linked List : Pop First Code

```
def pop_first(self):  
    → if self.length == 0:  
        return None  
    temp = self.head  
    self.head = self.head.next  
    temp.next = None  
    self.length -= 1  
    if self.length == 0:  
        self.tail = None
```

Linked List : Pop First Code

```
def pop_first(self):  
    → if self.length == 0:  
        return None  
    temp = self.head  
    self.head = self.head.next  
    temp.next = None  
    self.length -= 1  
    if self.length == 0:  
        self.tail = None
```

Linked List : Pop First Code

```
def pop_first(self):  
    if self.length == 0:  
        return None  
    temp = self.head  
    self.head = self.head.next  
    temp.next = None  
    self.length -= 1  
    → if self.length == 0:  
        self.tail = None
```

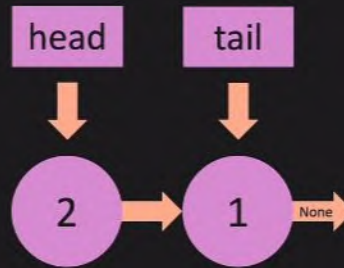
Linked List : Pop First Code

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

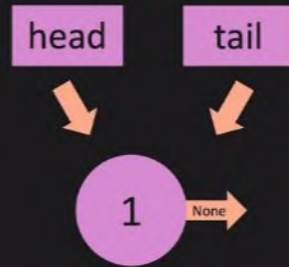
Linked List : Pop First Code

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

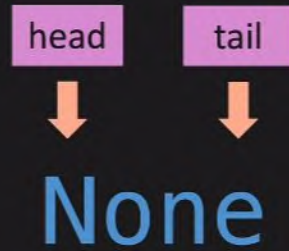

Linked List : Pop First Skenario Percobaan



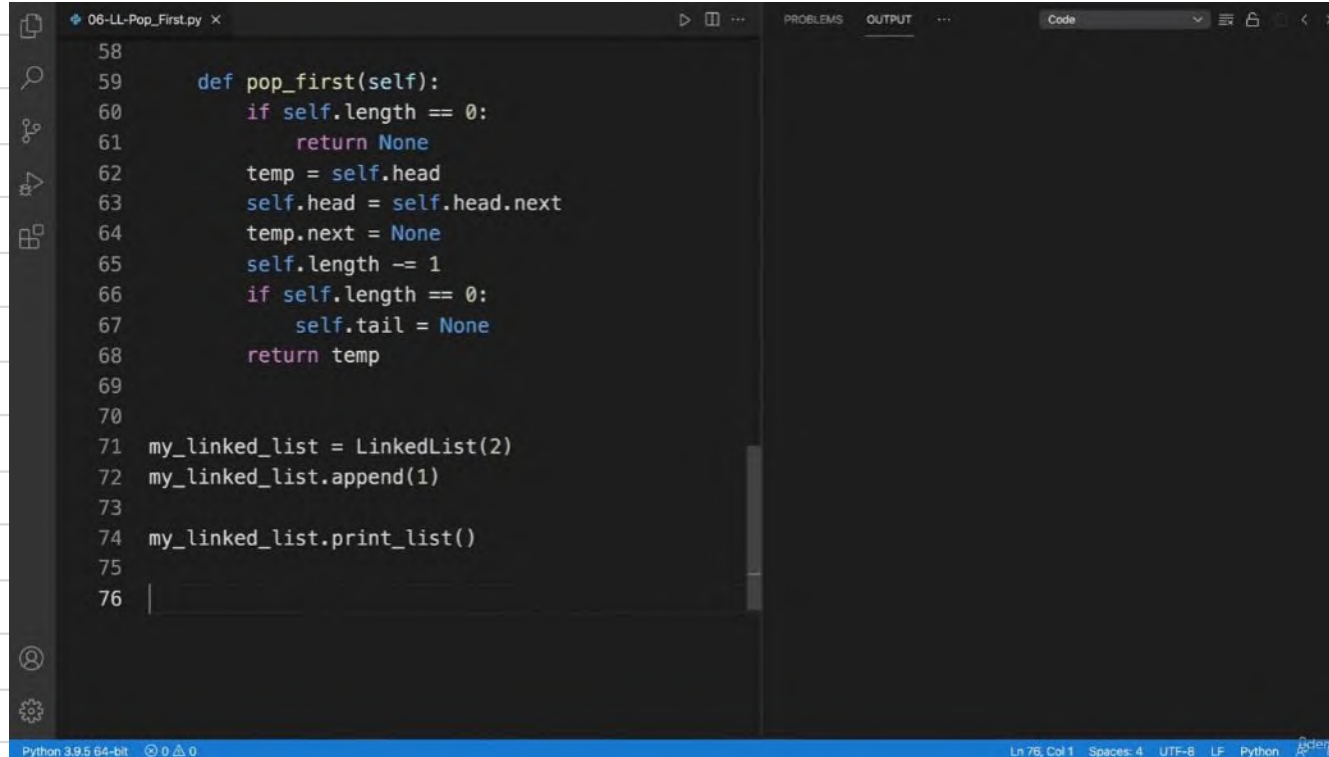
Linked List : Pop First Skenario Percobaan



Linked List : Pop First Skenario Percobaan



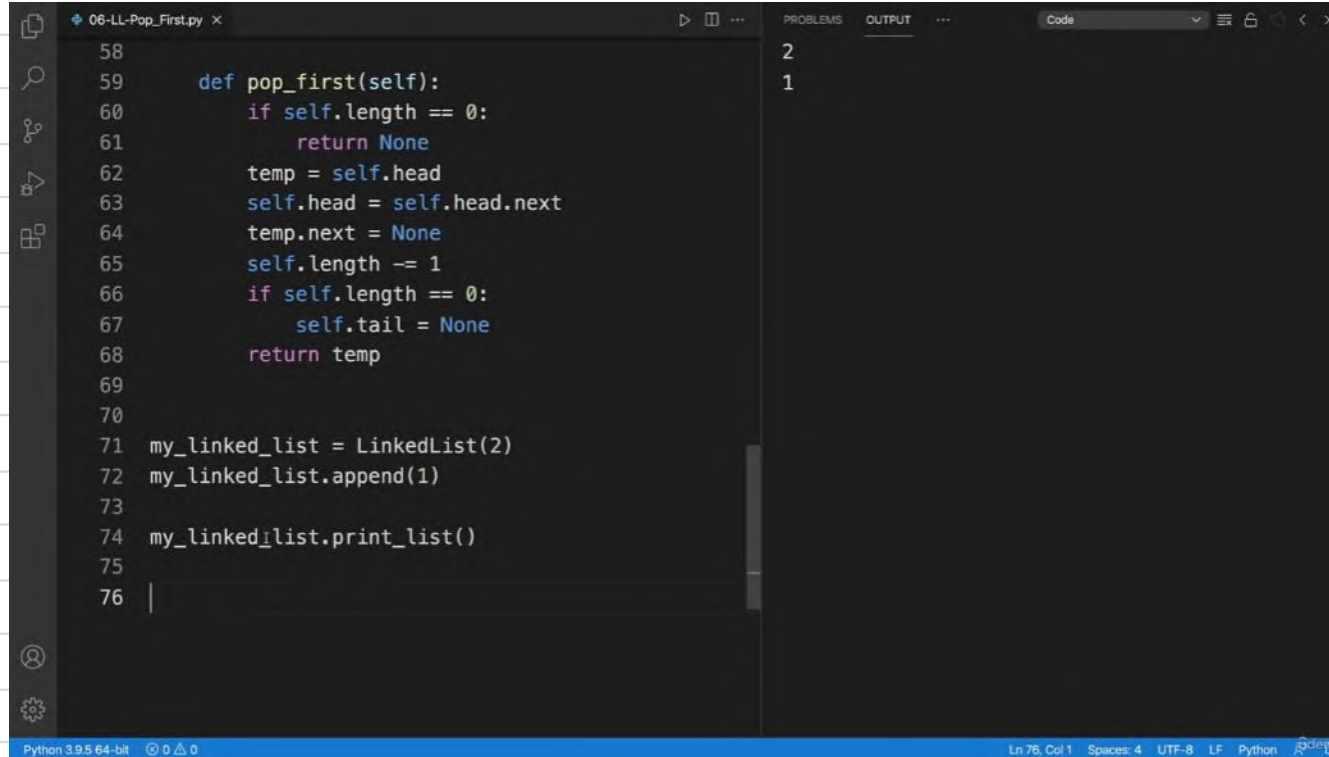
Linked List : Pop First Skenario Percobaan



```
58
59     def pop_first(self):
60         if self.length == 0:
61             return None
62         temp = self.head
63         self.head = self.head.next
64         temp.next = None
65         self.length -= 1
66         if self.length == 0:
67             self.tail = None
68         return temp
69
70
71 my_linked_list = LinkedList(2)
72 my_linked_list.append(1)
73
74 my_linked_list.print_list()
75
76
```

The screenshot shows a code editor with a dark theme. The file name is '06-LL-Pop_First.py'. The code defines a `pop_first` method for a linked list, which removes the first node and returns it. Below the method, there is a test scenario where a `LinkedList` object is created with 2 nodes, the value 1 is appended, and the list is printed. The status bar at the bottom indicates 'Python 3.9.5 64-bit', 'Ln 76, Col 1', 'Spaces: 4', 'UTF-8', 'LF', and 'Python'.

Linked List : Pop First Skenario Percobaan



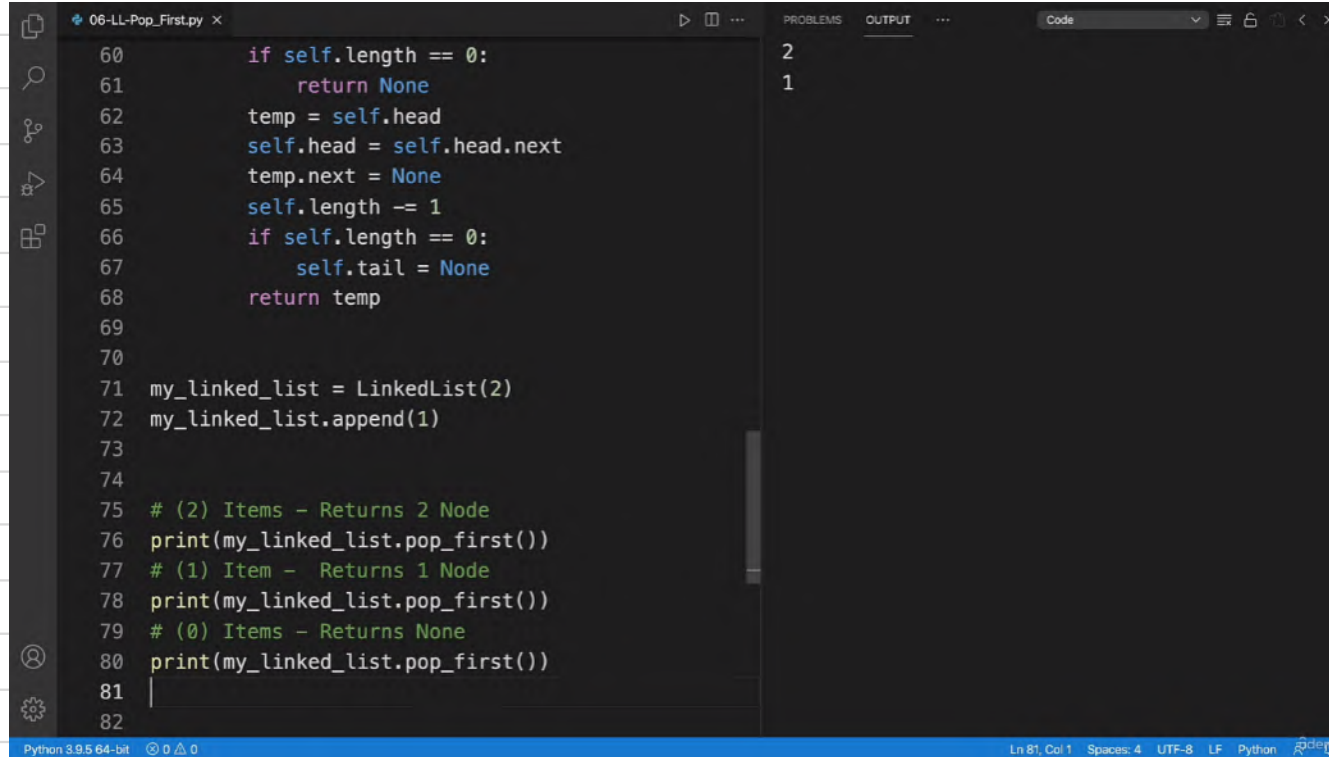
The screenshot shows a code editor with a file named '06-LL-Pop_First.py'. The code defines a `pop_first` method for a linked list, which removes the first node and returns the next node. Below the method, a linked list is created with two nodes (values 2 and 1), and the `pop_first` method is called. The output window on the right shows the result of the pop operation, which is the value 2.

```
58
59     def pop_first(self):
60         if self.length == 0:
61             return None
62         temp = self.head
63         self.head = self.head.next
64         temp.next = None
65         self.length -= 1
66         if self.length == 0:
67             self.tail = None
68         return temp
69
70
71 my_linked_list = LinkedList(2)
72 my_linked_list.append(1)
73
74 my_linked_list.print_list()
75
76 |
```

2
1

Python 3.9.5 64-bit 0 0 0 Ln 76, Col 1 Spaces: 4 UTF-8 LF Python

Linked List : Pop First Skenario Percobaan



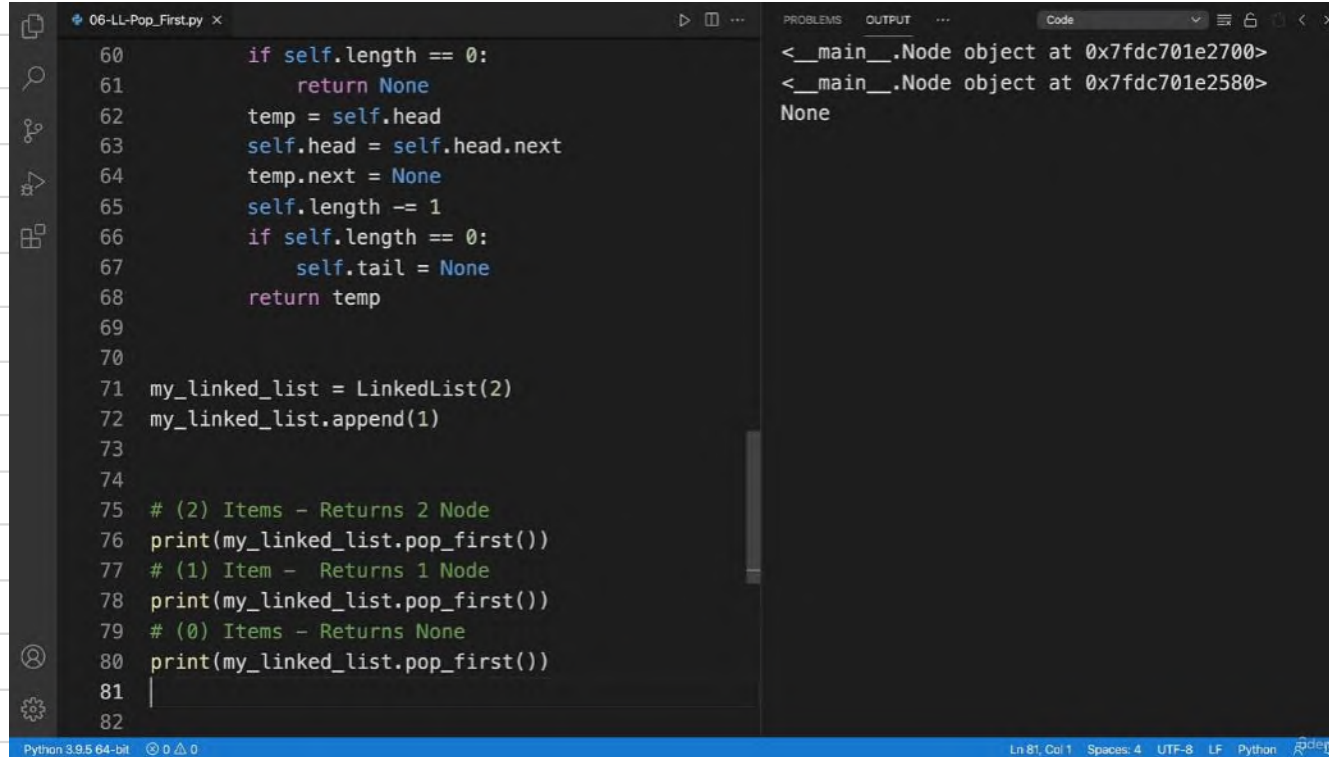
The image shows a code editor window with a dark theme. The file name is '06-LL-Pop_First.py'. The code is written in Python and implements a linked list with a pop_first method. The editor has a sidebar on the left with icons for Explorer, Search, Source Control, Run and Debug, and Extensions. The top right has tabs for PROBLEMS, OUTPUT, and Code. The bottom status bar shows 'Python 3.9.5 64-bit', 'Ln 81, Col 1', 'Spaces: 4', 'UTF-8', 'LF', 'Python', and a logo.

```
60     if self.length == 0:
61         return None
62     temp = self.head
63     self.head = self.head.next
64     temp.next = None
65     self.length -= 1
66     if self.length == 0:
67         self.tail = None
68     return temp
69
70
71 my_linked_list = LinkedList(2)
72 my_linked_list.append(1)
73
74
75 # (2) Items - Returns 2 Node
76 print(my_linked_list.pop_first())
77 # (1) Item - Returns 1 Node
78 print(my_linked_list.pop_first())
79 # (0) Items - Returns None
80 print(my_linked_list.pop_first())
81
82
```

2
1

Python 3.9.5 64-bit 0 0 0 Ln 81, Col 1 Spaces: 4 UTF-8 LF Python

Linked List : Pop First Skenario Percobaan



The screenshot shows a Python IDE with a file named '06-LL-Pop_First.py'. The code defines a linked list and tests its pop operation. The output pane shows the results of the pop operation.

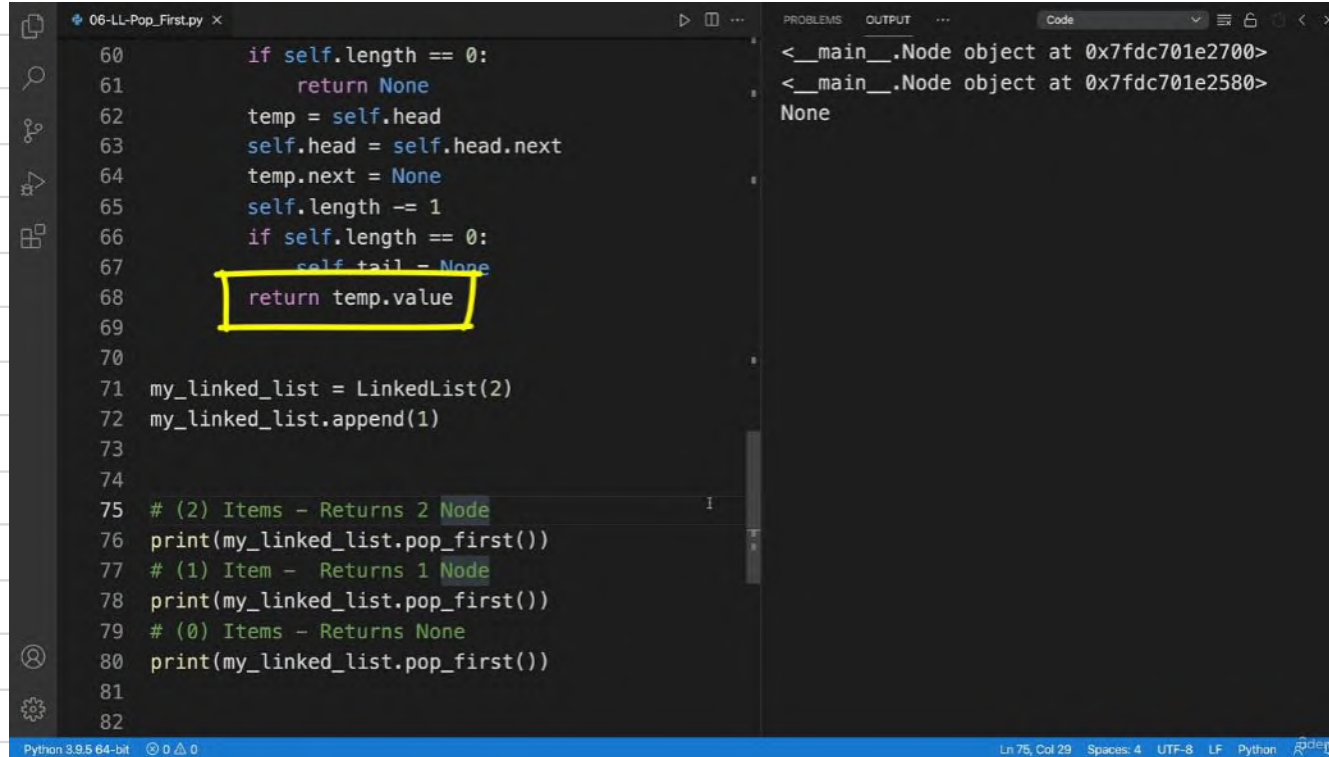
```
06-LL-Pop_First.py x
60     if self.length == 0:
61         return None
62     temp = self.head
63     self.head = self.head.next
64     temp.next = None
65     self.length -= 1
66     if self.length == 0:
67         self.tail = None
68     return temp
69
70
71 my_linked_list = LinkedList(2)
72 my_linked_list.append(1)
73
74
75 # (2) Items - Returns 2 Node
76 print(my_linked_list.pop_first())
77 # (1) Item - Returns 1 Node
78 print(my_linked_list.pop_first())
79 # (0) Items - Returns None
80 print(my_linked_list.pop_first())
81
82
```

PROBLEMS OUTPUT ... Code

```
<__main__.Node object at 0x7fdc701e2700>
<__main__.Node object at 0x7fdc701e2580>
None
```

Python 3.9.5 64-bit 0 0 0 Ln 81, Col 1 Spaces: 4 UTF-8 LF Python

Linked List : Pop First Skenario Percobaan



The screenshot shows a Python IDE with a file named '06-LL-Pop_First.py'. The code defines a linked list and tests its pop operation. The 'pop_first' method is shown with a yellow box around the return statement. The output pane shows the results of the pop operations.

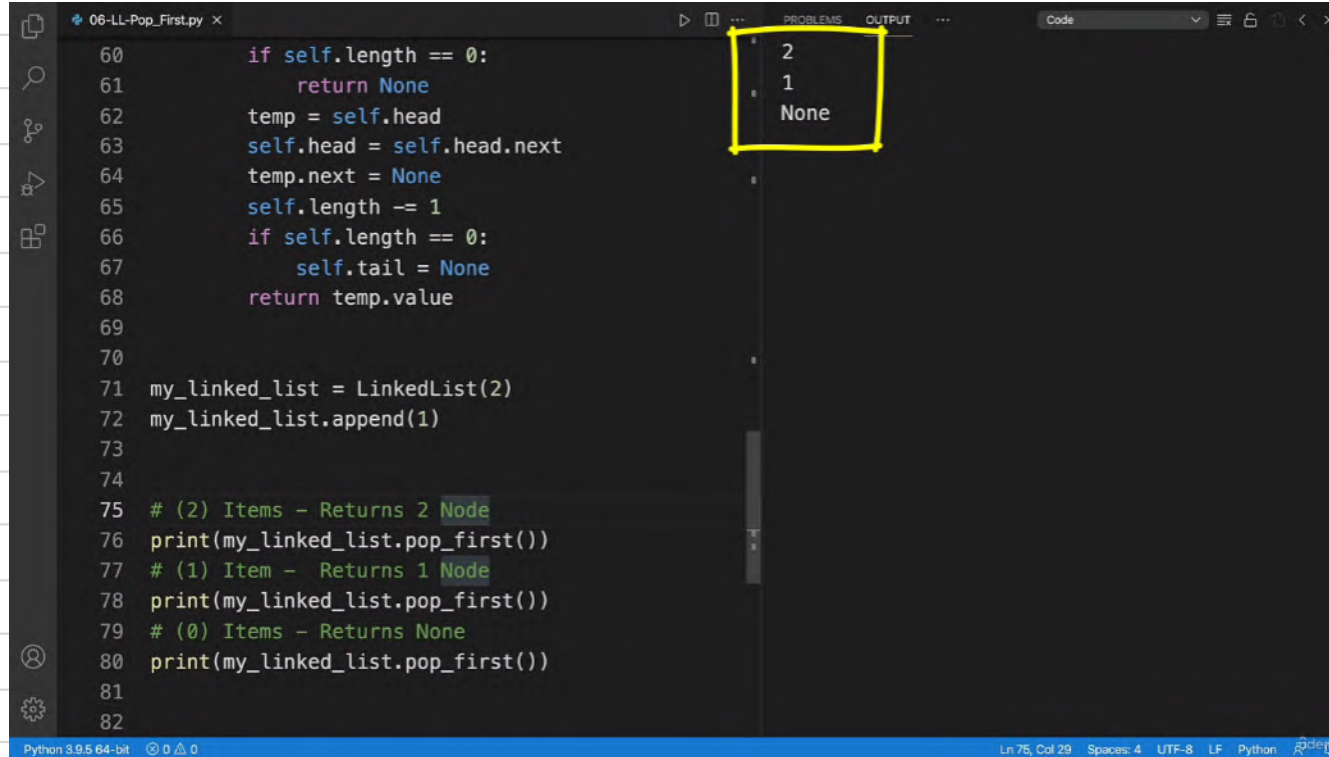
```
06-LL-Pop_First.py
60     if self.length == 0:
61         return None
62     temp = self.head
63     self.head = self.head.next
64     temp.next = None
65     self.length -= 1
66     if self.length == 0:
67         self.tail = None
68     return temp.value
69
70
71 my_linked_list = LinkedList(2)
72 my_linked_list.append(1)
73
74
75 # (2) Items - Returns 2 Node
76 print(my_linked_list.pop_first())
77 # (1) Item - Returns 1 Node
78 print(my_linked_list.pop_first())
79 # (0) Items - Returns None
80 print(my_linked_list.pop_first())
81
82
```

PROBLEMS OUTPUT

```
<__main__.Node object at 0x7fdc701e2700>
<__main__.Node object at 0x7fdc701e2580>
None
```

Python 3.9.5 64-bit 0 0 0 Ln 75, Col 29 Spaces: 4 UTF-8 LF Python

Linked List : Pop First Skenario Percobaan



```
06-LL-Pop_First.py x
60     if self.length == 0:
61         return None
62     temp = self.head
63     self.head = self.head.next
64     temp.next = None
65     self.length -= 1
66     if self.length == 0:
67         self.tail = None
68     return temp.value
69
70
71 my_linked_list = LinkedList(2)
72 my_linked_list.append(1)
73
74
75 # (2) Items - Returns 2 Node
76 print(my_linked_list.pop_first())
77 # (1) Item - Returns 1 Node
78 print(my_linked_list.pop_first())
79 # (0) Items - Returns None
80 print(my_linked_list.pop_first())
81
82
```

2
1
None

Python 3.9.5 64-bit 0 0 0 Ln 75, Col 29 Spaces: 4 UTF-8 LF Python

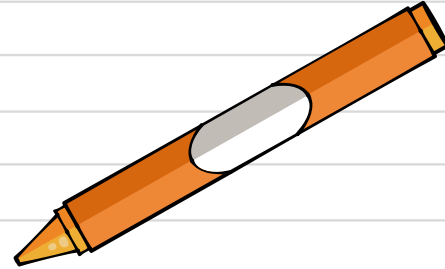


10



Linked List

Get



Linked List : Get

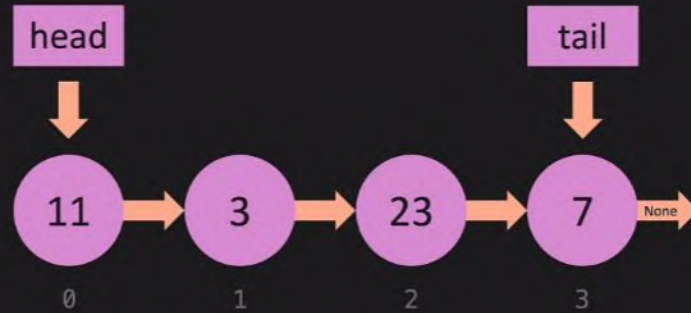
- Apa itu Get...?

Linked List : Get Code

```
def get(self, index):
```

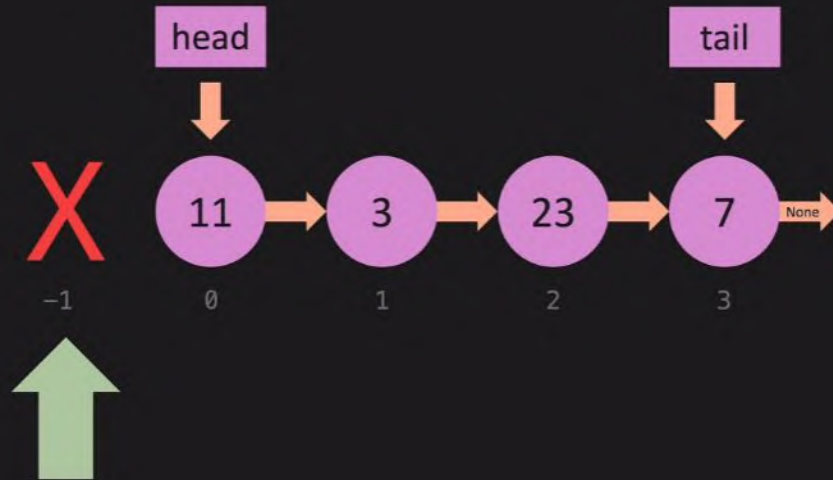
Linked List : Get Code

```
def get(self, index):
```



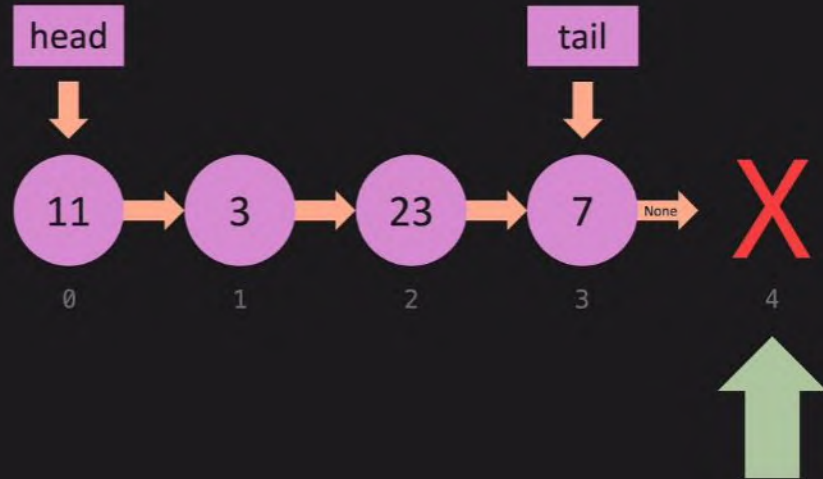
Linked List : Get Code

```
def get(self, index):
```



Linked List : Get Code

```
def get(self, index):
```



Linked List : Get Code

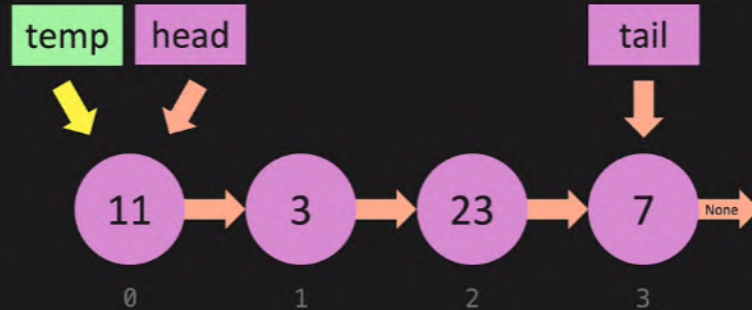
```
def get(self, index):  
    if index < 0 or index >= self.length:
```


Linked List : Get Code

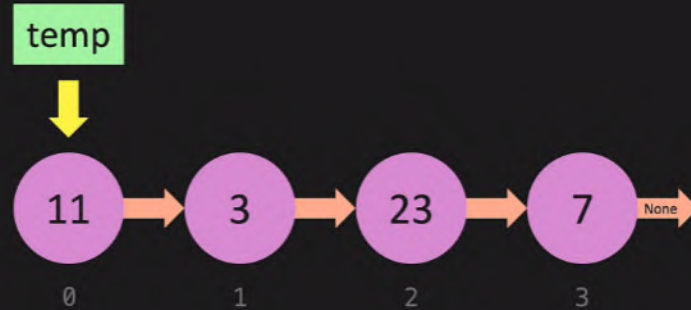
```
def get(self, index):  
    if index < 0 or index >= self.length:  
        return None
```

Linked List : Get Code

```
temp = self.head
```



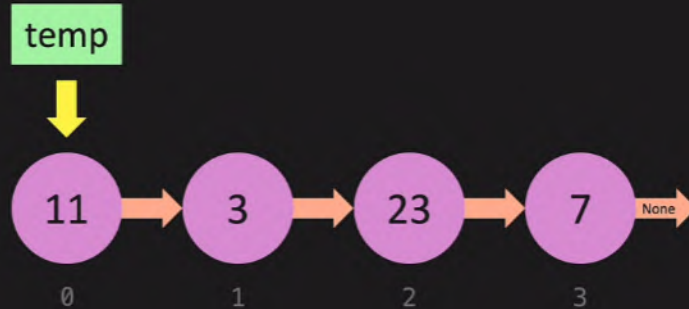
Linked List : Get Code



Misal index = 2

Linked List : Get Code

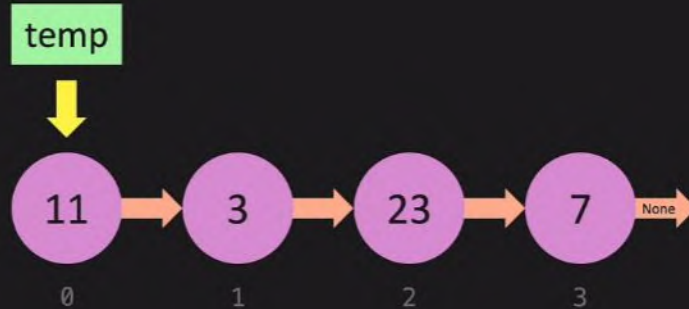
```
for _ in range(index):
```



Misal index = 2

Linked List : Get Code

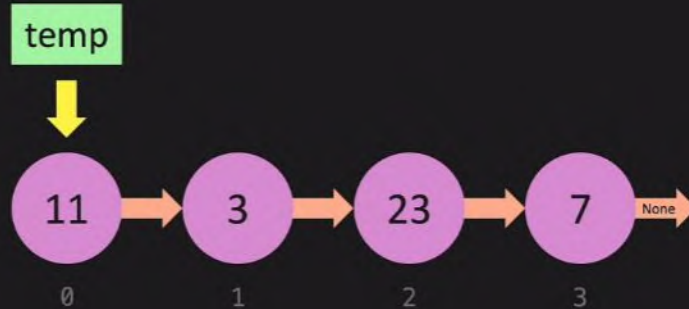
```
for _ in range(index):  
    temp = temp.next
```



Misal index = 2

Linked List : Get Code

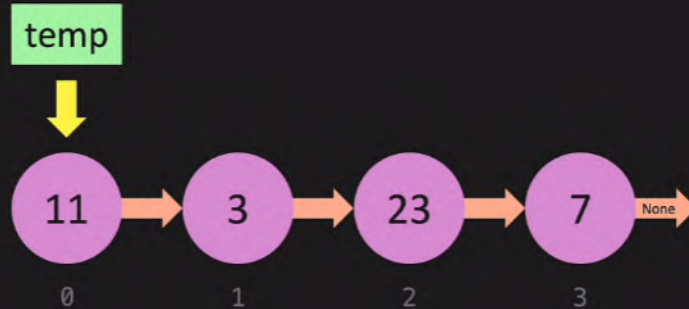
```
for i in range(index):  
    temp = temp.next
```



Misal index = 2

Linked List : Get Code

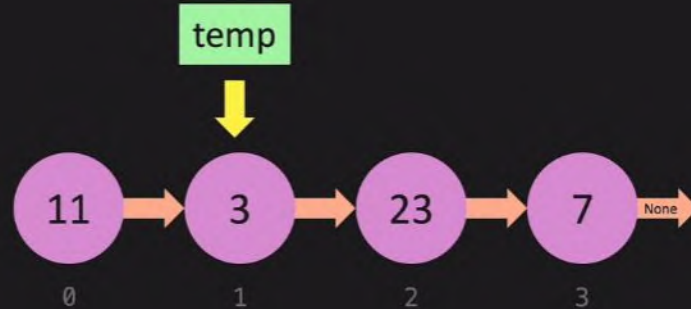
```
for i in range(index):  
    temp = temp.next  
    print(i)
```



Misal index = 2

Linked List : Get Code

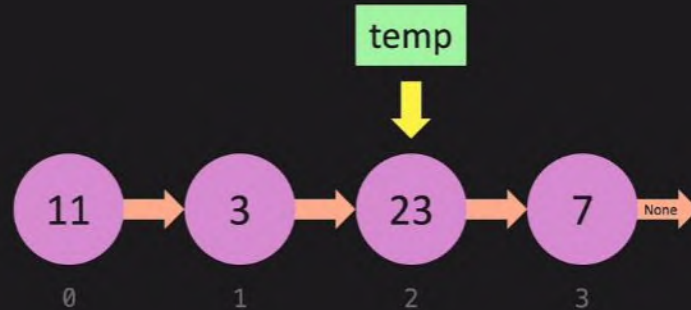
```
for _ in range(index):  
    temp = temp.next
```



Misal index = 2

Linked List : Get Code

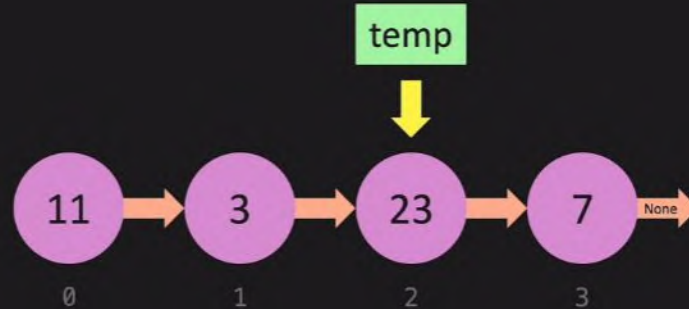
```
for _ in range(index):  
    temp = temp.next
```



Misal index = 2

Linked List : Get Code

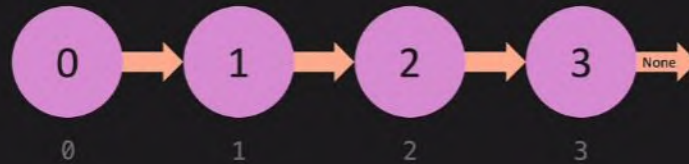
```
for _ in range(index):  
    temp = temp.next  
return temp
```



Linked List : Get Code

```
def get(self, index):  
    if index < 0 or index >= self.length:  
        return None  
    temp = self.head  
    for _ in range(index):  
        temp = temp.next  
    return temp
```

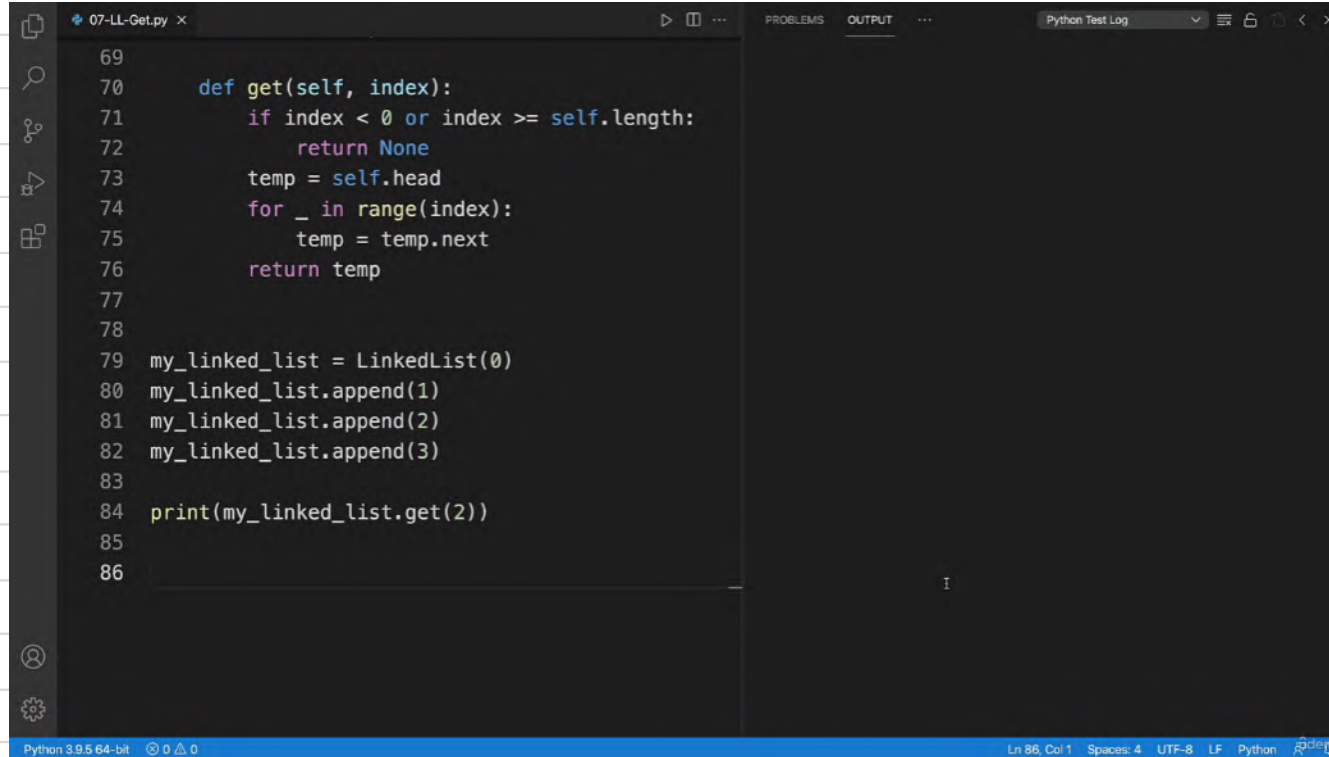
Linked List : Get Skenario Percobaan



Linked List : Get Skenario Percobaan



Linked List : Get Skenario Percobaan

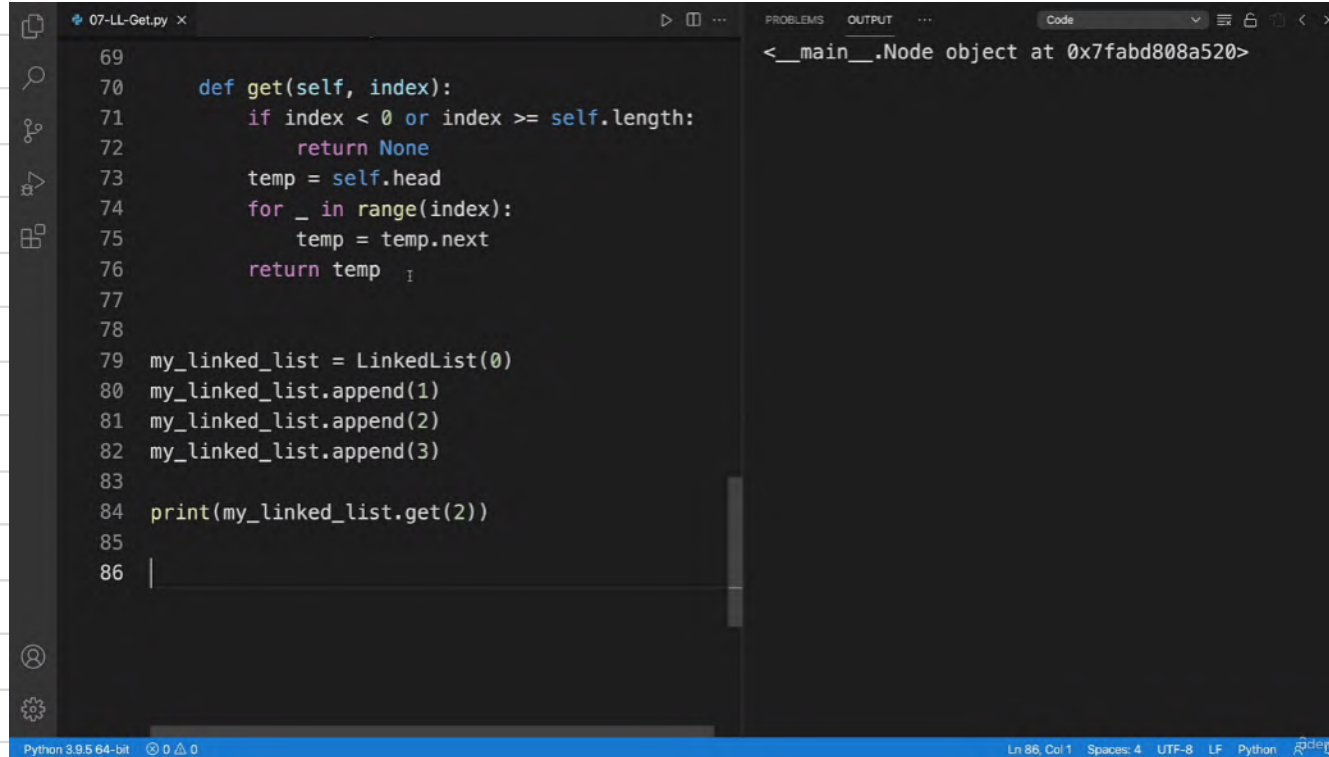


The screenshot shows a Python IDE with a file named '07-LL-Get.py'. The code defines a `get` method for a linked list and tests it. The `get` method checks if the index is within the bounds of the list. If not, it returns `None`. If yes, it traverses the list from the head to the specified index and returns the node. The test code creates a linked list, appends values 1, 2, and 3, and then prints the value at index 2.

```
69
70     def get(self, index):
71         if index < 0 or index >= self.length:
72             return None
73         temp = self.head
74         for _ in range(index):
75             temp = temp.next
76         return temp
77
78
79 my_linked_list = LinkedList(0)
80 my_linked_list.append(1)
81 my_linked_list.append(2)
82 my_linked_list.append(3)
83
84 print(my_linked_list.get(2))
85
86
```

The IDE interface includes a sidebar with icons for Explorer, Search, Source Control, Run and Debug, and Extensions. The top bar shows 'PROBLEMS', 'OUTPUT', and 'Python Test Log'. The bottom status bar indicates 'Python 3.9.5 64-bit', 'Ln 86, Col 1', 'Spaces: 4', 'UTF-8', 'LF', 'Python', and a logo.

Linked List : Get Skenario Percobaan



The screenshot shows a Python IDE with a file named '07-LL-Get.py'. The code defines a `get` method for a linked list and demonstrates its usage. The output pane shows the result of the `get` method call.

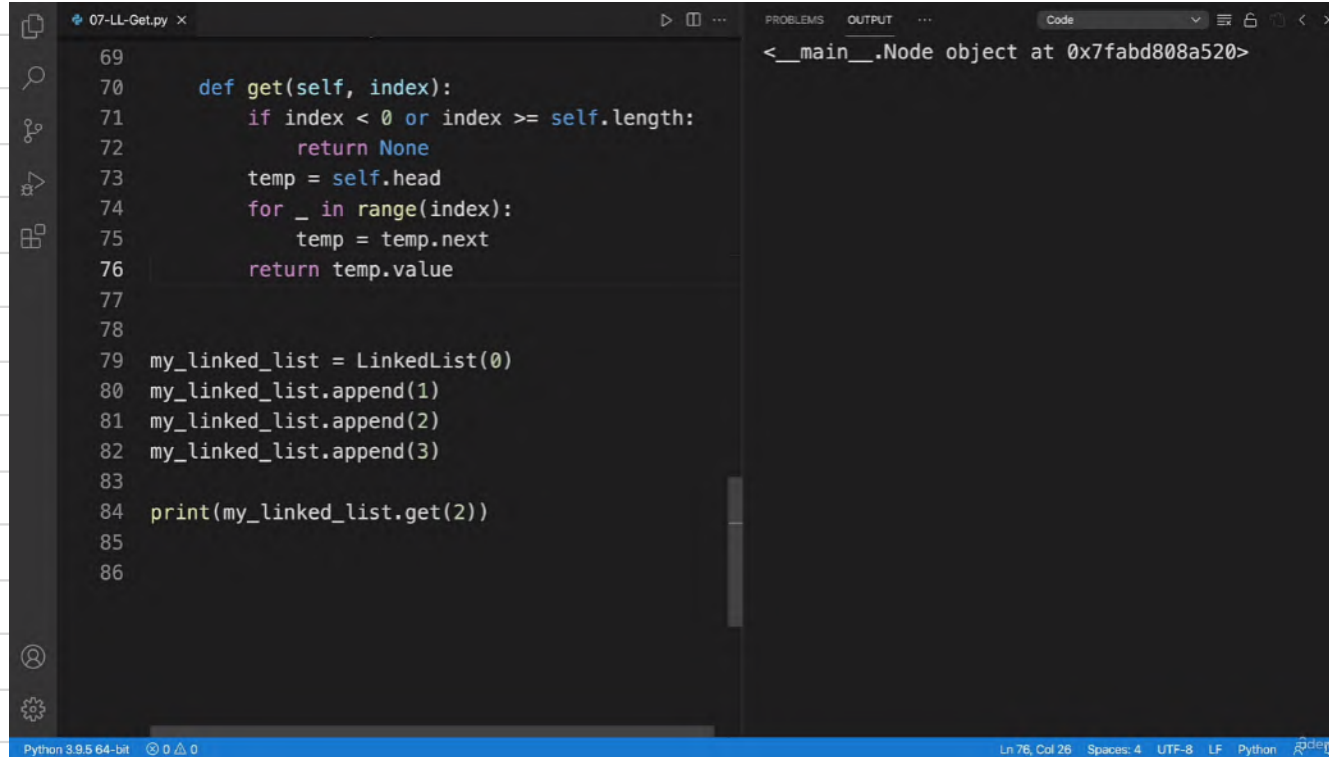
```
69
70     def get(self, index):
71         if index < 0 or index >= self.length:
72             return None
73         temp = self.head
74         for _ in range(index):
75             temp = temp.next
76         return temp
77
78
79 my_linked_list = LinkedList(0)
80 my_linked_list.append(1)
81 my_linked_list.append(2)
82 my_linked_list.append(3)
83
84 print(my_linked_list.get(2))
85
86
```

Output:

```
<__main__.Node object at 0x7fabd808a520>
```

Python 3.9.5 64-bit | 0 0 0 | Ln 86, Col 1 | Spaces: 4 | UTF-8 | LF | Python |

Linked List : Get Skenario Percobaan



The screenshot shows a Python IDE with a file named '07-LL-Get.py'. The code defines a linked list with a 'get' method and a test script. The output window shows the result of the 'get' method call.

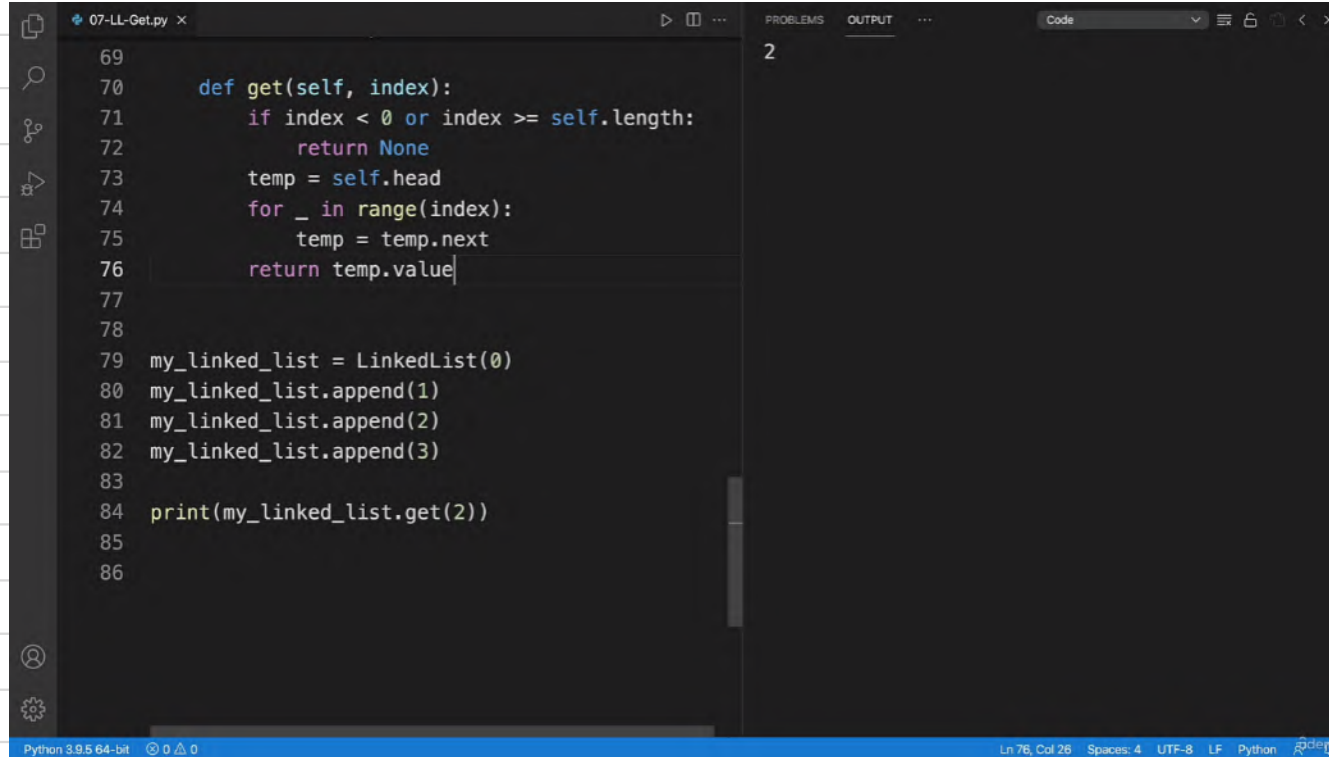
```
69
70     def get(self, index):
71         if index < 0 or index >= self.length:
72             return None
73         temp = self.head
74         for _ in range(index):
75             temp = temp.next
76         return temp.value
77
78
79 my_linked_list = LinkedList(0)
80 my_linked_list.append(1)
81 my_linked_list.append(2)
82 my_linked_list.append(3)
83
84 print(my_linked_list.get(2))
85
86
```

PROBLEMS OUTPUT ... Code

<__main__.Node object at 0x7fabd808a520>

Python 3.9.5 64-bit 0 0 0 Ln 76, Col 26 Spaces: 4 UTF-8 LF Python

Linked List : Get Skenario Percobaan



The screenshot shows a Python IDE with a file named '07-LL-Get.py'. The code defines a linked list with a 'get' method and demonstrates its use. The output pane shows the result of the 'get' method call.

```
69
70     def get(self, index):
71         if index < 0 or index >= self.length:
72             return None
73         temp = self.head
74         for _ in range(index):
75             temp = temp.next
76         return temp.value
77
78
79 my_linked_list = LinkedList(0)
80 my_linked_list.append(1)
81 my_linked_list.append(2)
82 my_linked_list.append(3)
83
84 print(my_linked_list.get(2))
85
86
```

PROBLEMS OUTPUT ... Code

2

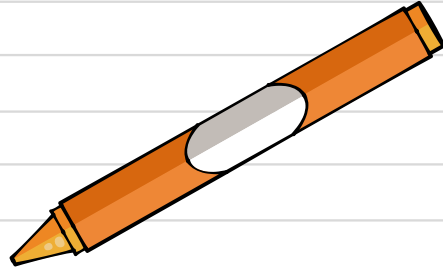
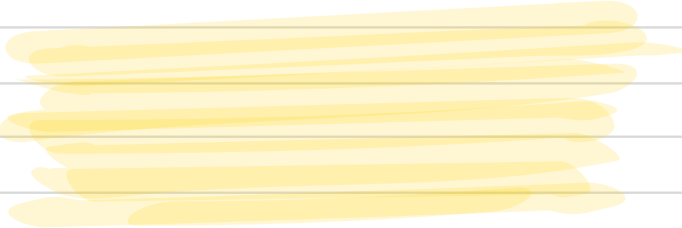
Python 3.9.5 64-bit 0 0 0 Ln 76, Col 26 Spaces: 4 UTF-8 LF Python



11



Linked List Set



Linked List : Set

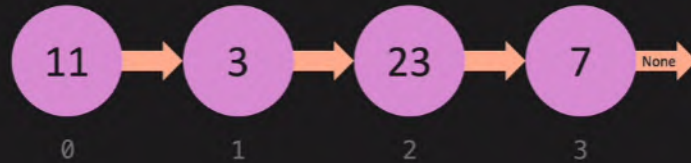
- Apa itu Set...?

Linked List : Set Code

```
def set_value(self, index, value):
```

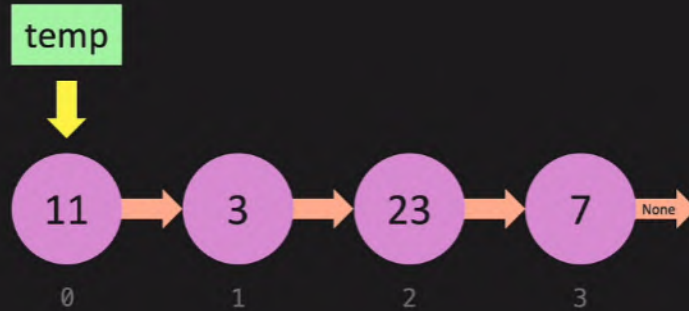
Linked List : Set Code

```
def set_value(self, index, value):
```



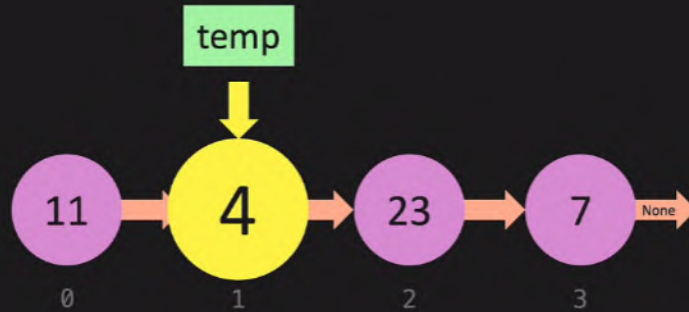
Linked List : Set Code

```
def set_value(self, index, value):
```



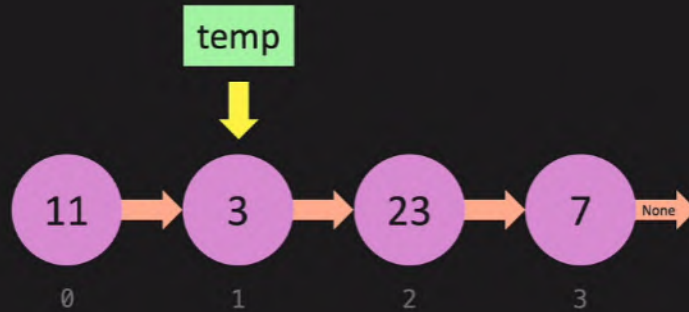
Linked List : Set Code

```
def set_value(self, index, value):
```



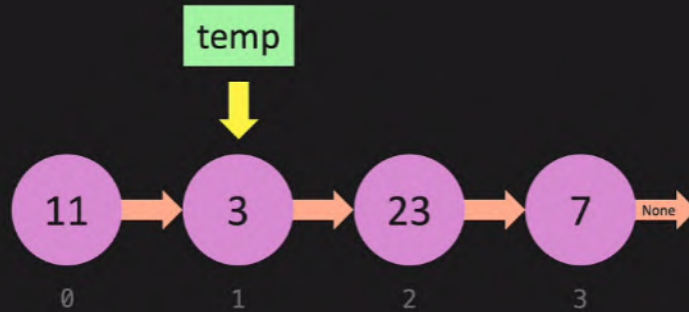
Linked List : Set Code

```
def set_value(self, index, value):
```



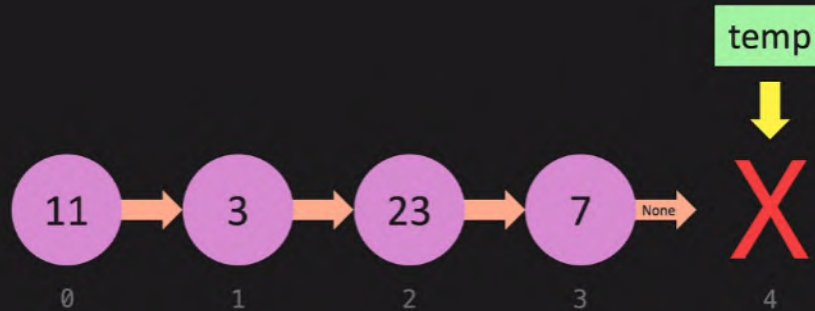
Linked List : Set Code

```
def set_value(self, index, value):  
    temp
```



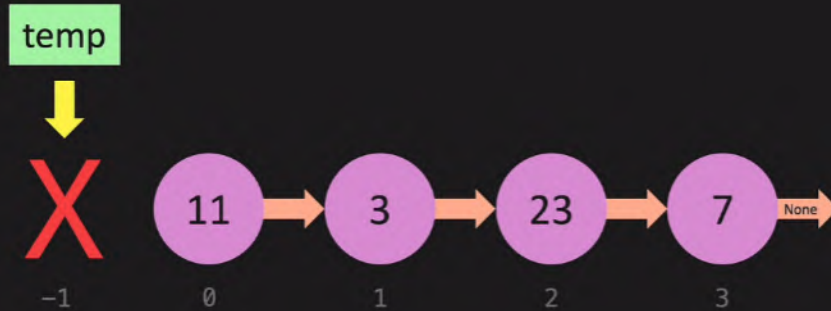
Linked List : Set Code

```
def set_value(self, index, value):  
    temp
```



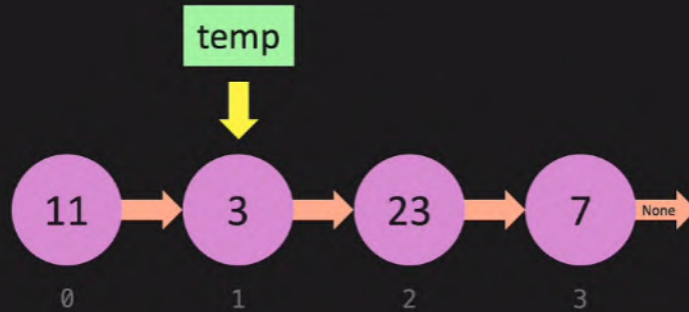
Linked List : Set Code

```
def set_value(self, index, value):  
    temp
```



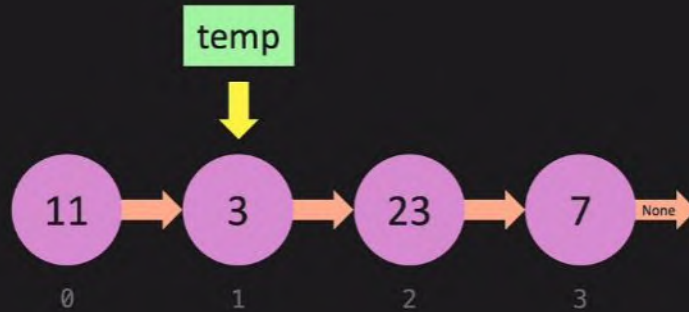
Linked List : Set Code

```
def set_value(self, index, value):  
    temp = self.get(index)
```



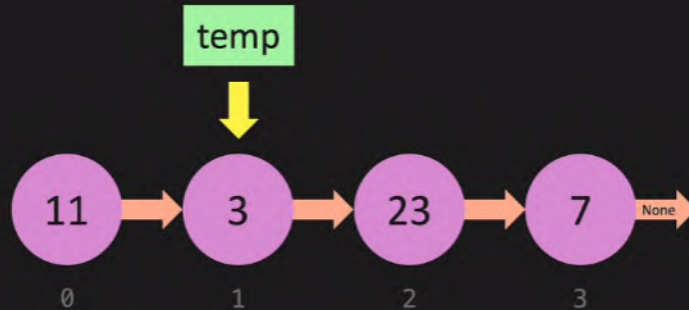
Linked List : Set Code

```
def set_value(self, index, value):  
    temp = self.get(index)  
    if temp:
```



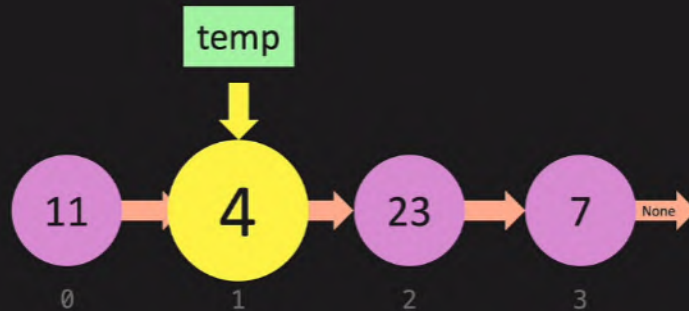
Linked List : Set Code

```
def set_value(self, index, value):  
    temp = self.get(index)  
    if temp:  
        temp.value = value
```



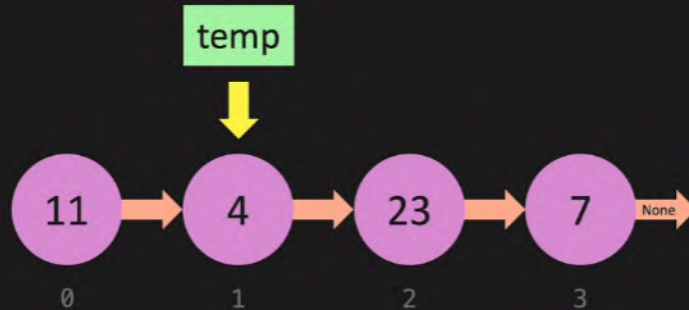
Linked List : Set Code

```
def set_value(self, index, value):  
    temp = self.get(index)  
    if temp:  
        temp.value = value
```



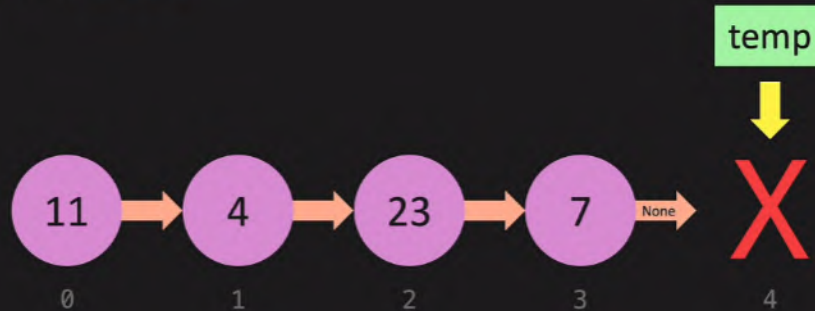
Linked List : Set Code

```
def set_value(self, index, value):  
    temp = self.get(index)  
    if temp:  
        temp.value = value  
    return True
```



Linked List : Set Code

```
def set_value(self, index, value):  
    temp = self.get(index)  
    if temp:  
        temp.value = value  
        return True  
    return False
```



Linked List : Set Skenario Percobaan



Linked List : Set Skenario Percobaan



Linked List : Set Skenario Percobaan

```
def set_value(self, index, value):  
    temp = self.get(index)  
    if temp:  
        temp.value = value  
        return True  
    return False
```

```
my_linked_list = LinkedList(11)  
my_linked_list.append(3)  
my_linked_list.append(23)  
my_linked_list.append(7)  
  
my_linked_list.print_list()
```

```
11  
3  
23  
7
```

Linked List : Set Skenario Percobaan

```
def set_value(self, index, value):  
    temp = self.get(index)  
    if temp:  
        temp.value = value  
        return True  
    return False
```

```
my_linked_list = LinkedList(11)  
my_linked_list.append(3)  
my_linked_list.append(23)  
my_linked_list.append(7)
```

```
my_linked_list.set_value(1,4)
```

```
my_linked_list.print_list()
```

```
11  
3  
23  
7
```

Linked List : Set Skenario Percobaan

```
def set_value(self, index, value):  
    temp = self.get(index)  
    if temp:  
        temp.value = value  
        return True  
    return False
```

```
my_linked_list = LinkedList(11)  
my_linked_list.append(3)  
my_linked_list.append(23)  
my_linked_list.append(7)  
  
my_linked_list.set_value(1,4)  
  
my_linked_list.print_list()
```

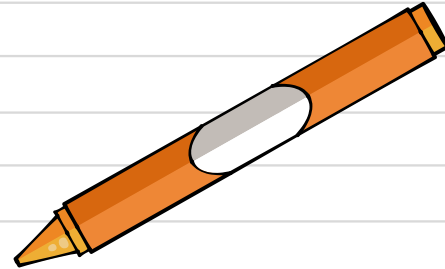
11
4
23
7



12



Linked List Insert



Linked List : Insert

- Apa itu Insert...?

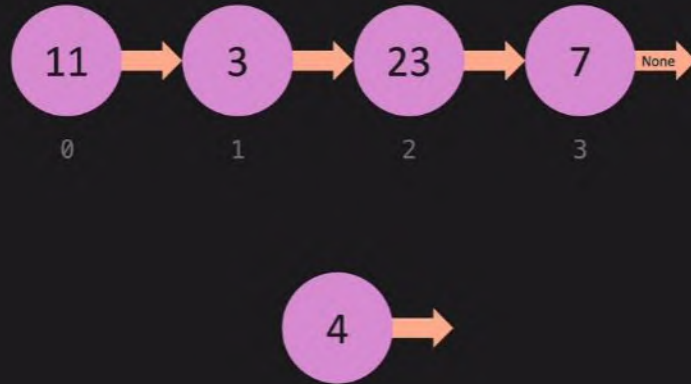
Linked List : Insert

```
def insert(self, index, value):
```

Linked List : Insert



Linked List : Insert

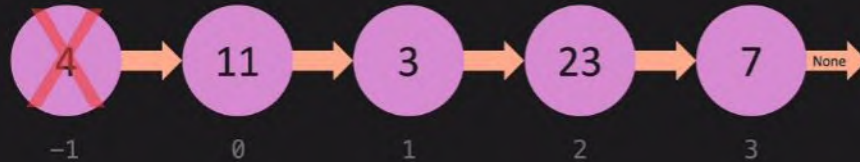


Linked List : Insert



Linked List : Insert

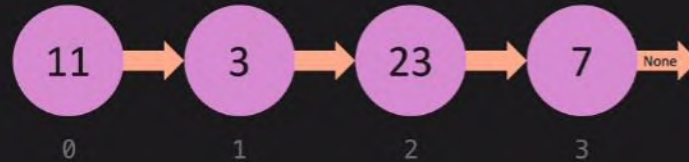
index < 0



Linked List : Insert

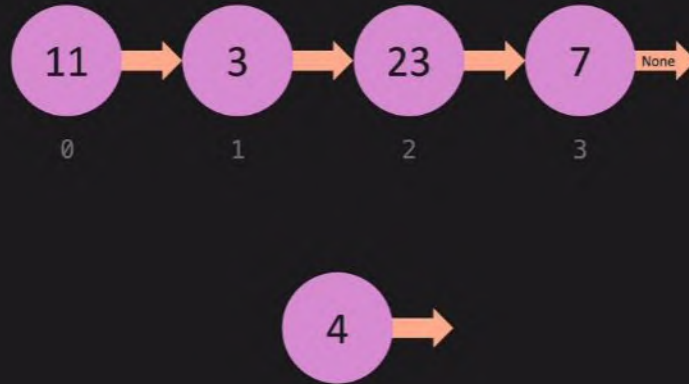
`index < 0`

`index > self.length`

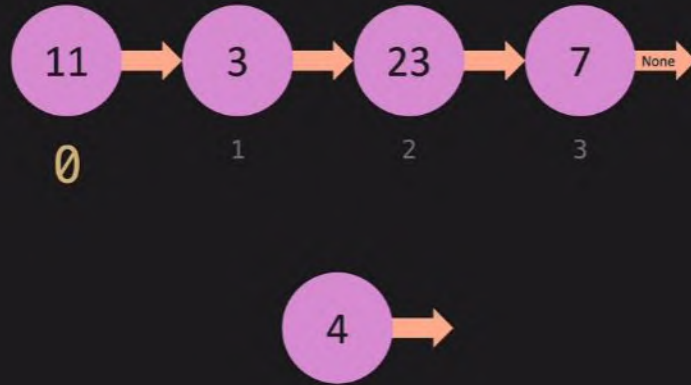


Linked List : Insert

```
if index < 0 or index > self.length:  
    return False
```

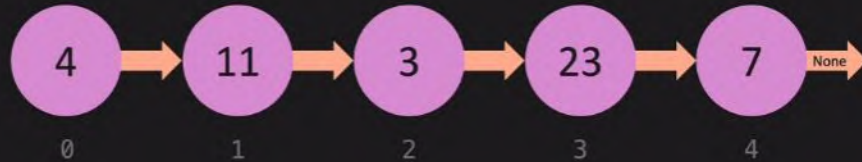


Linked List : Insert

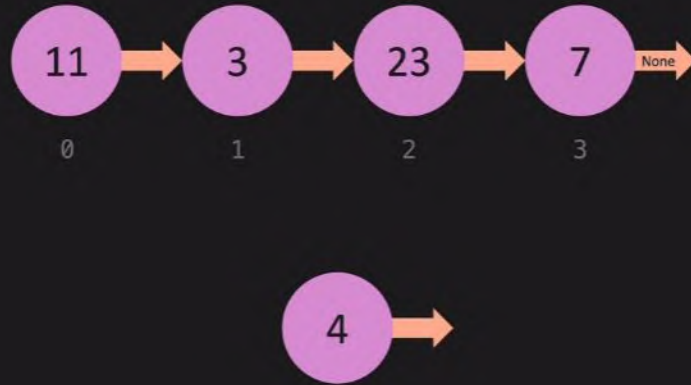


Linked List : Insert

```
if index == 0:  
    return self.prepend(value)
```



Linked List : Insert



Linked List : Insert

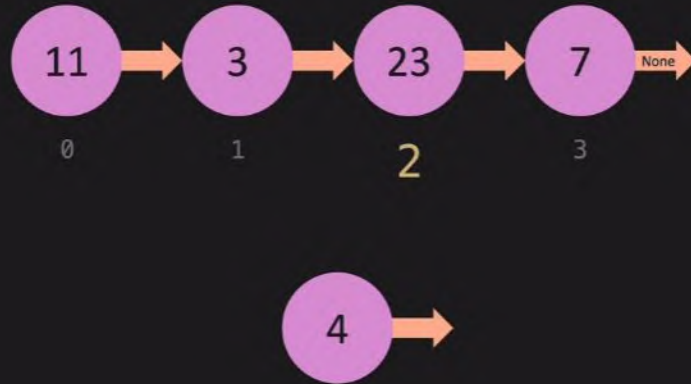
```
if index == self.length:  
    return self.append(value)
```



Linked List : Insert

```
if index < 0 or index > self.length:  
    return False  
if index == 0:  
    return self.prepend(value)  
if index == self.length:  
    return self.append(value)
```

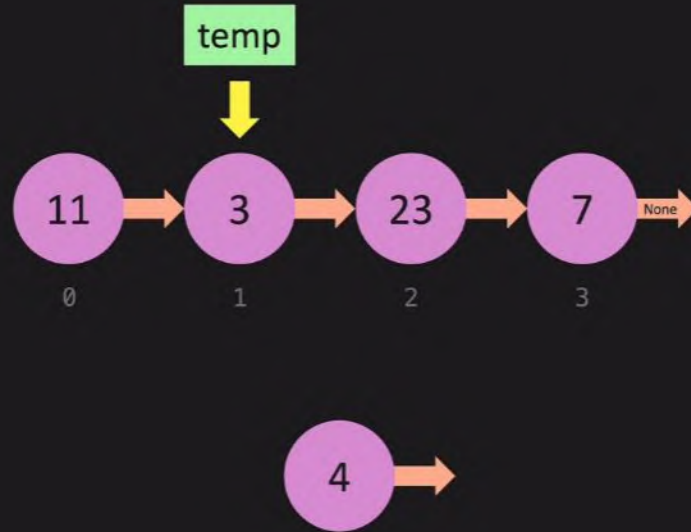
Linked List : Insert



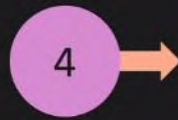
Linked List : Insert



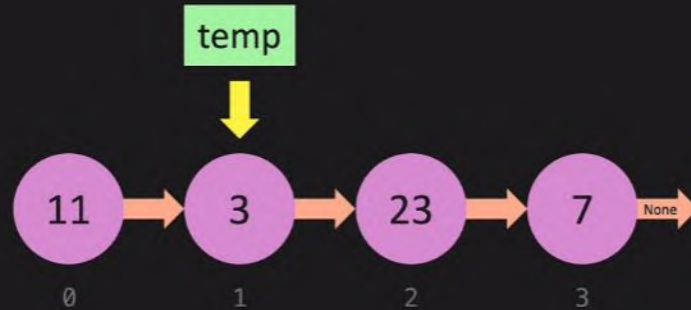
Linked List : Insert



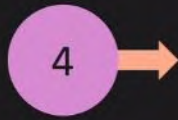
Linked List : Insert



`new_node = Node(value)`



Linked List : Insert

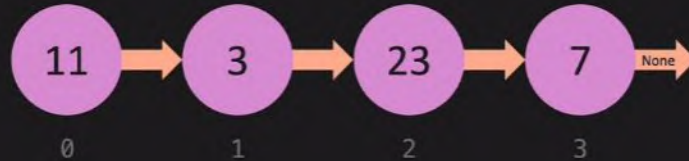


```
new_node = Node(value)
```

temp



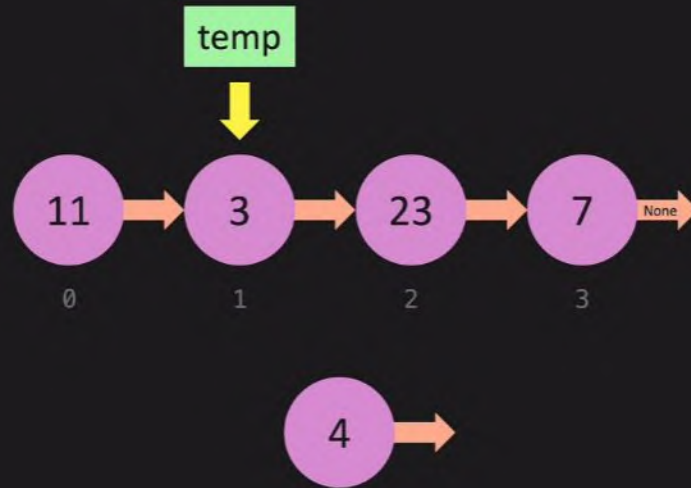
```
temp = self.get(index - 1)
```



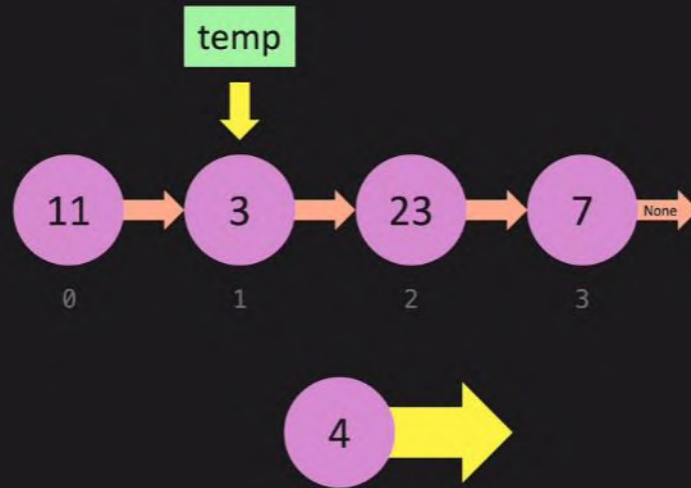
Linked List : Insert

```
def insert(self, index, value):  
    if index < 0 or index > self.length:  
        return None  
    if index == 0:  
        return self.prepend(value)  
    if index == self.length:  
        return self.append(value)  
    new_node = Node(value)  
    temp = self.get(index - 1)
```

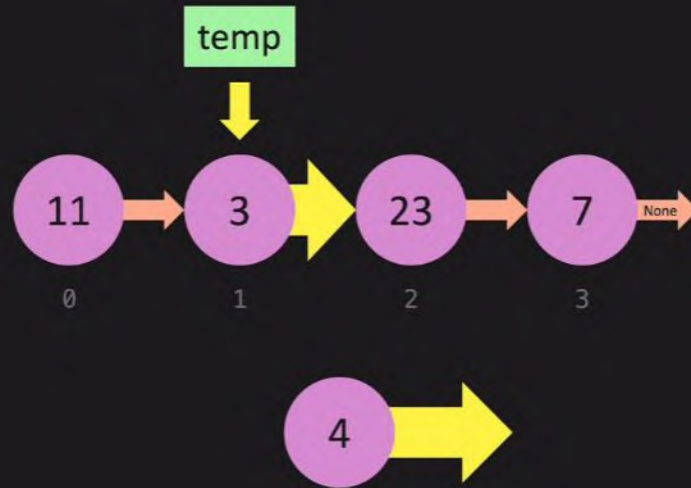
Linked List : Insert



Linked List : Insert

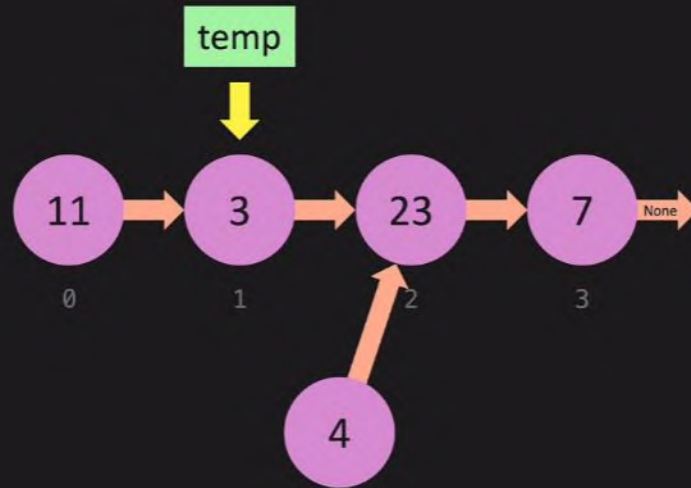


Linked List : Insert



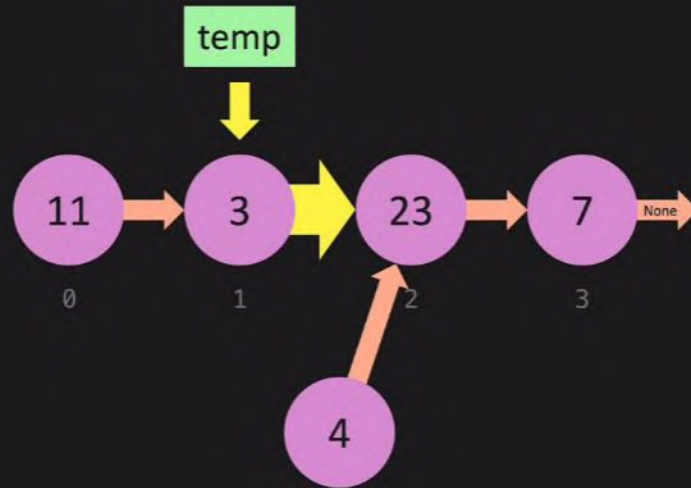
Linked List : Insert

```
new_node.next = temp.next
```



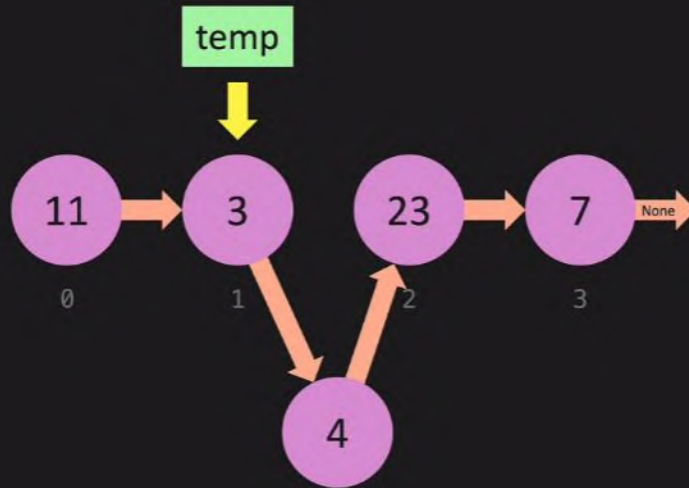
Linked List : Insert

```
new_node.next = temp.next
```



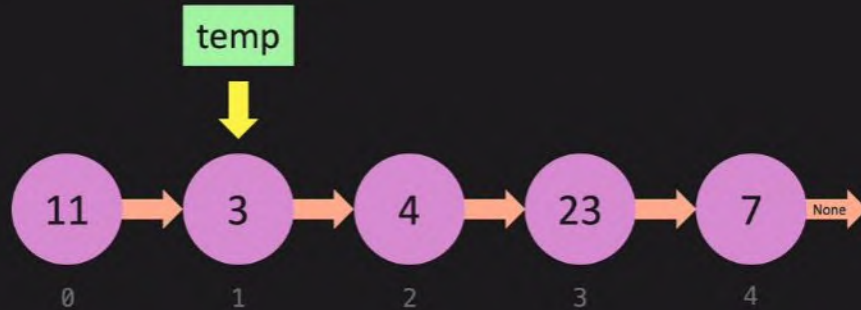
Linked List : Insert

```
new_node.next = temp.next  
temp.next = new_node
```



Linked List : Insert

```
new_node.next = temp.next  
temp.next = new_node
```



Linked List : Insert

```
def insert(self, index, value):  
    if index < 0 or index > self.length:  
        return False  
    if index == 0:  
        return self.prepend(value)  
    if index == self.length:  
        return self.append(value)  
    new_node = Node(value)  
    temp = self.get(index - 1)  
    new_node.next = temp.next  
    temp.next = new_node
```

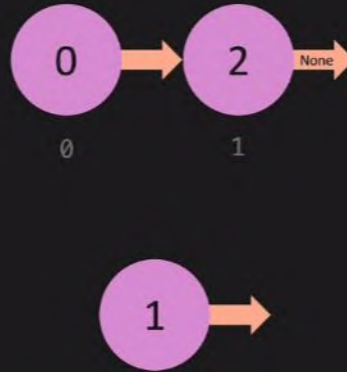
Linked List : Insert

```
def insert(self, index, value):  
    if index < 0 or index > self.length:  
        return False  
    if index == 0:  
        return self.prepend(value)  
    if index == self.length:  
        return self.append(value)  
    new_node = Node(value)  
    temp = self.get(index - 1)  
    new_node.next = temp.next  
    temp.next = new_node  
    self.length += 1
```

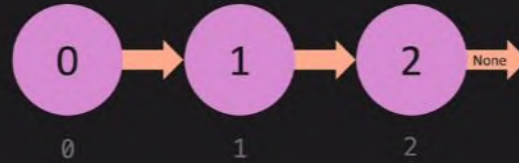
Linked List : Insert

```
def insert(self, index, value):  
    if index < 0 or index > self.length:  
        return False  
    if index == 0:  
        return self.prepend(value)  
    if index == self.length:  
        return self.append(value)  
    new_node = Node(value)  
    temp = self.get(index - 1)  
    new_node.next = temp.next  
    temp.next = new_node  
    self.length += 1  
    return True
```

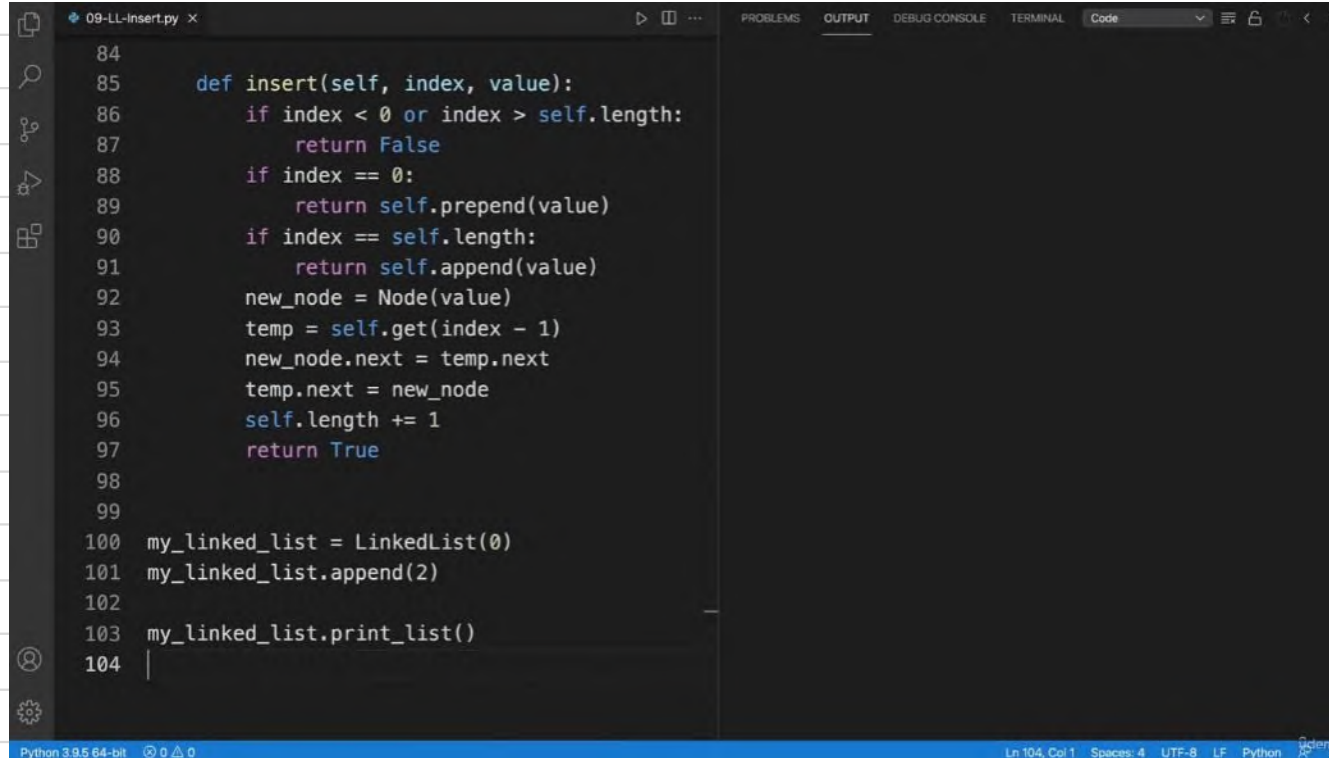
Linked List : Insert Skenario Percobaan



Linked List : Insert Skenario Percobaan



Linked List : Insert Skenario Percobaan

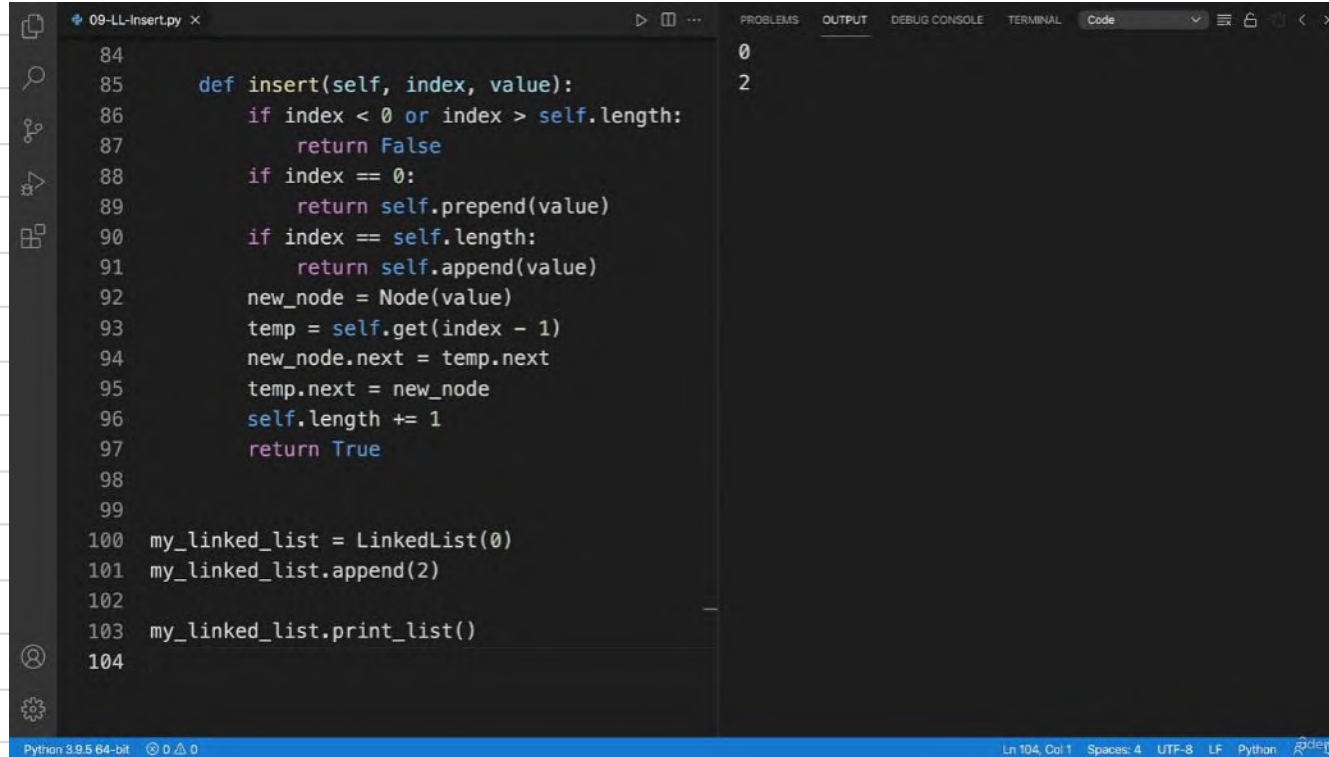


```
09-LL-Insert.py x
84
85     def insert(self, index, value):
86         if index < 0 or index > self.length:
87             return False
88         if index == 0:
89             return self.prepend(value)
90         if index == self.length:
91             return self.append(value)
92         new_node = Node(value)
93         temp = self.get(index - 1)
94         new_node.next = temp.next
95         temp.next = new_node
96         self.length += 1
97         return True
98
99
100 my_linked_list = LinkedList()
101 my_linked_list.append(2)
102
103 my_linked_list.print_list()
104 |
```

Python 3.9.5 64-bit

Ln 104, Col 1 Spaces: 4 UTF-8 LF Python

Linked List : Insert Skenario Percobaan



The image shows a code editor window titled "09-LL-Insert.py". The editor contains Python code for a linked list. The code defines an `insert` method that takes `self`, `index`, and `value` as arguments. It checks if the index is valid (between 0 and the current length of the list). If the index is 0, it prepends the new node. If the index is equal to the current length, it appends the new node. Otherwise, it inserts the new node at the specified index by updating the `next` pointer of the previous node. The code also initializes a `my_linked_list` object, appends the value 2, and prints the list.

```
84
85     def insert(self, index, value):
86         if index < 0 or index > self.length:
87             return False
88         if index == 0:
89             return self.prepend(value)
90         if index == self.length:
91             return self.append(value)
92         new_node = Node(value)
93         temp = self.get(index - 1)
94         new_node.next = temp.next
95         temp.next = new_node
96         self.length += 1
97         return True
98
99
100 my_linked_list = LinkedList(0)
101 my_linked_list.append(2)
102
103 my_linked_list.print_list()
104
```

The right side of the editor shows the `OUTPUT` tab with the following content:

```
0
2
```

The status bar at the bottom indicates "Python 3.9.5 64-bit", "Ln 104, Col 1", "Spaces: 4", "UTF-8", "LF", "Python", and a logo.

Linked List : Insert Skenario Percobaan

```
my_linked_list = LinkedList(0)
my_linked_list.append(2)

my_linked_list.insert(1,1)

my_linked_list.print_list()
|
```

0
2

I

Linked List : Insert Skenario Percobaan

```
my_linked_list = LinkedList(0)
my_linked_list.append(2)

my_linked_list.insert(1,1)

my_linked_list.print_list()
|
```

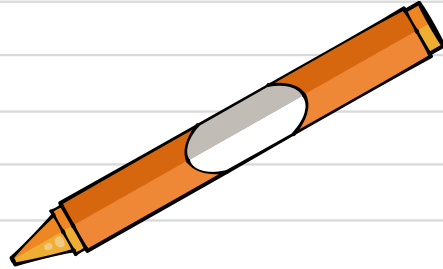
0
1
2



13



Linked List Remove



Linked List : Remove

- Apa itu Remove...?

Linked List : Remove

```
def remove(self, index):
```

Linked List : Remove



Linked List : Remove

`index < 0`



Linked List : Remove

`index < 0`

`index >= self.length`



Linked List : Remove

```
if index < 0 or index >= self.length:  
    return None
```



Linked List : Remove



Linked List : Remove

```
if index == 0:  
    return self.pop_first()
```



Linked List : Remove



Linked List : Remove

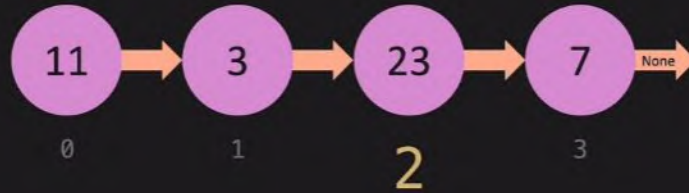
```
if index == self.length - 1:  
    return self.pop()
```



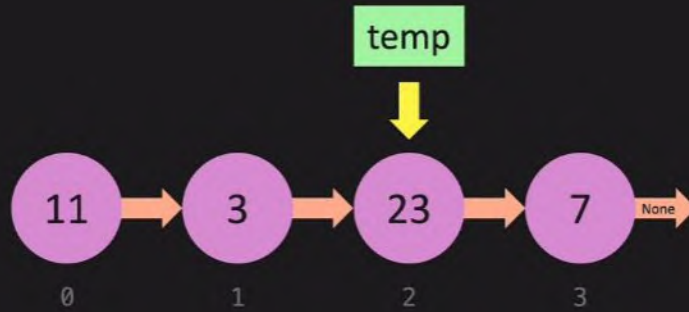
Linked List : Remove



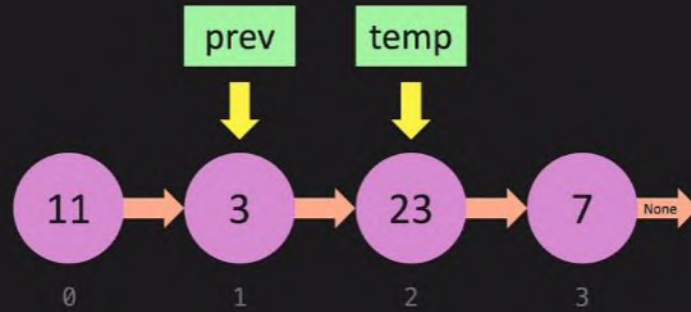
Linked List : Remove



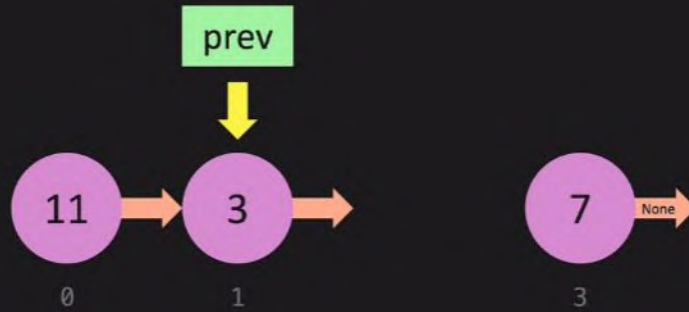
Linked List : Remove



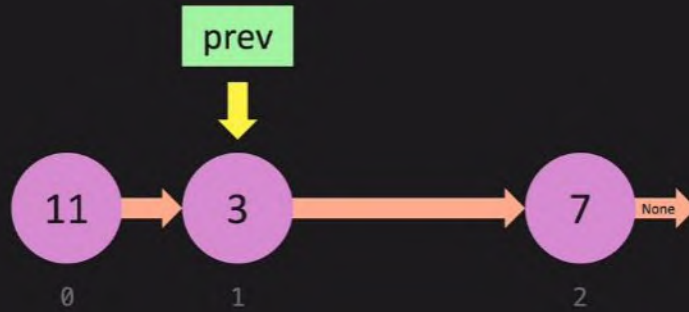
Linked List : Remove



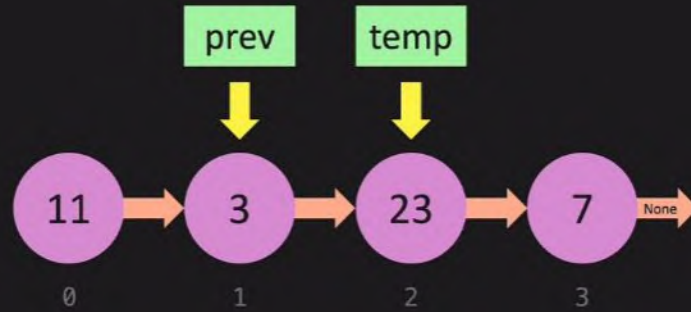
Linked List : Remove



Linked List : Remove



Linked List : Remove



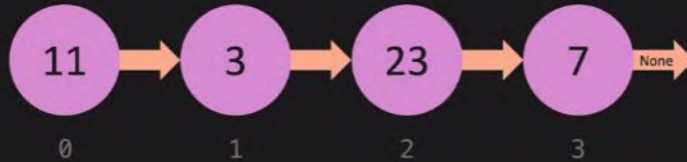
Linked List : Remove

prev

```
prev = self.get(index - 1)
```



temp



Linked List : Remove

prev

```
prev = self.get(index - 1)
```



temp

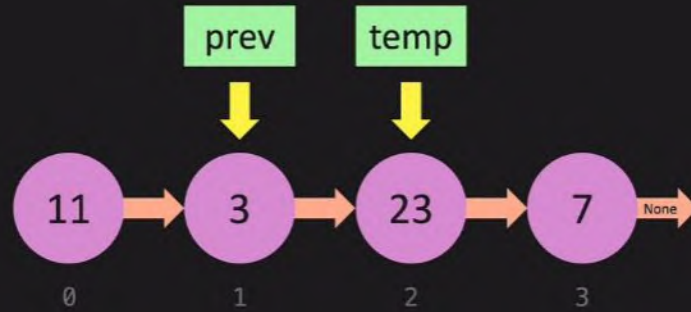
```
temp = prev.next
```



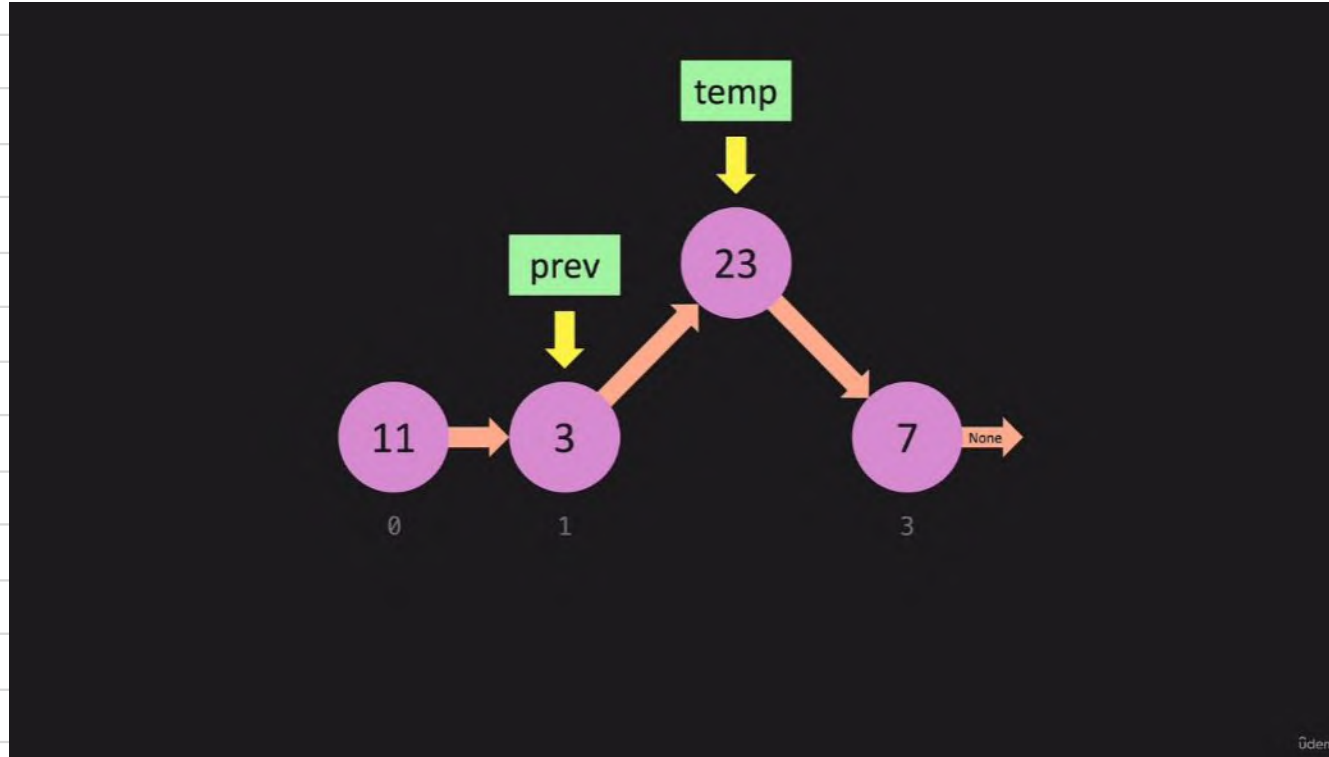
Linked List : Remove

```
def remove(self, index):  
    if index < 0 or index >= self.length:  
        return None  
    if index == 0:  
        return self.pop_first()  
    if index == self.length - 1:  
        return self.pop()  
    prev = self.get(index - 1)  
    temp = prev.next
```

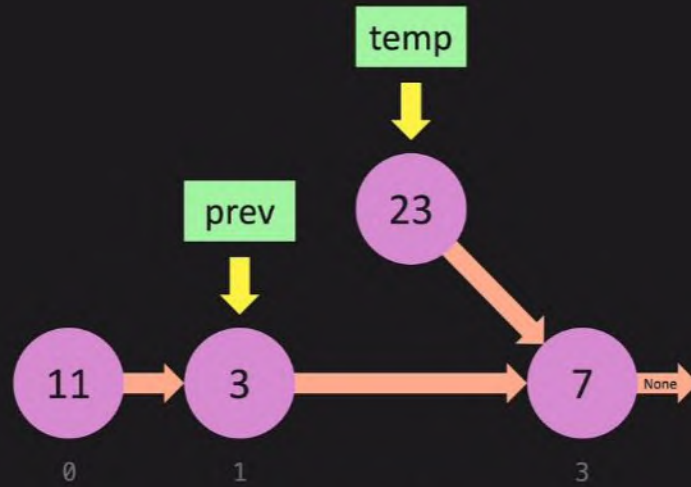
Linked List : Remove



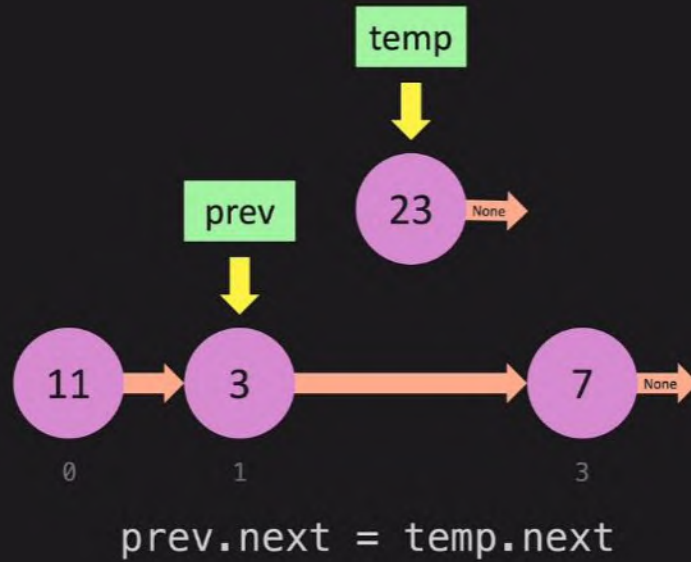
Linked List : Remove



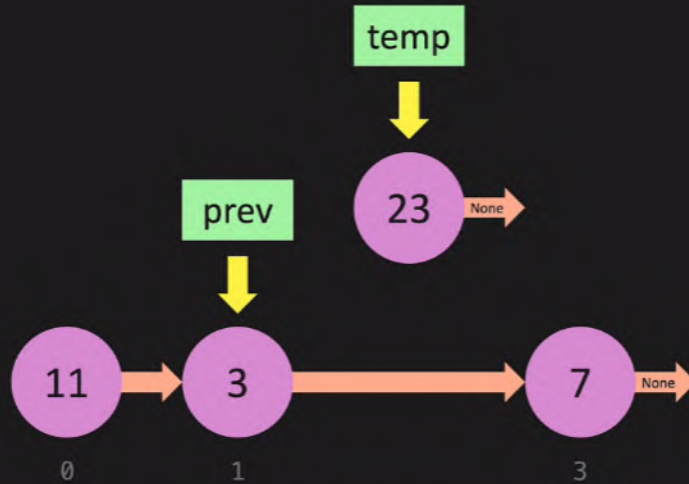
Linked List : Remove



Linked List : Remove

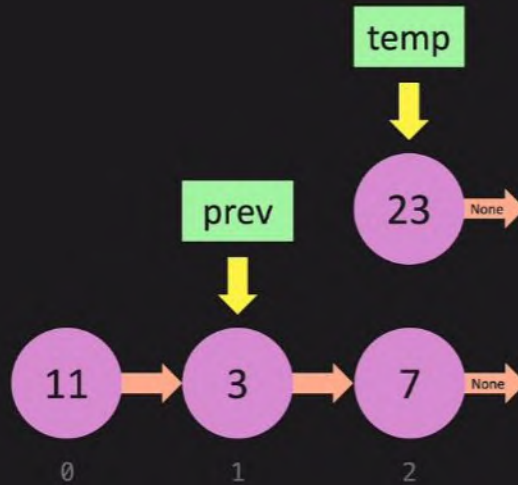


Linked List : Remove



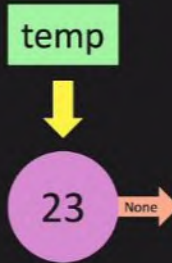
```
prev.next = temp.next  
temp.next = None
```

Linked List : Remove



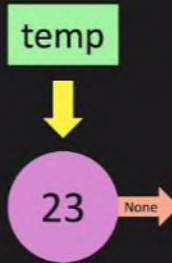
```
prev.next = temp.next  
temp.next = None
```

Linked List : Remove



```
prev.next = temp.next  
temp.next = None  
self.length -= 1
```

Linked List : Remove



```
prev.next = temp.next  
temp.next = None  
self.length -= 1  
return temp
```

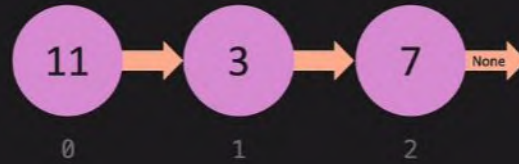
Linked List : Remove

```
def remove(self, index):  
    if index < 0 or index >= self.length:  
        return None  
    if index == 0:  
        return self.pop_first()  
    if index == self.length - 1:  
        return self.pop()  
    prev = self.get(index - 1)  
    temp = prev.next  
    prev.next = temp.next  
    temp.next = None  
    self.length -= 1  
    return temp
```

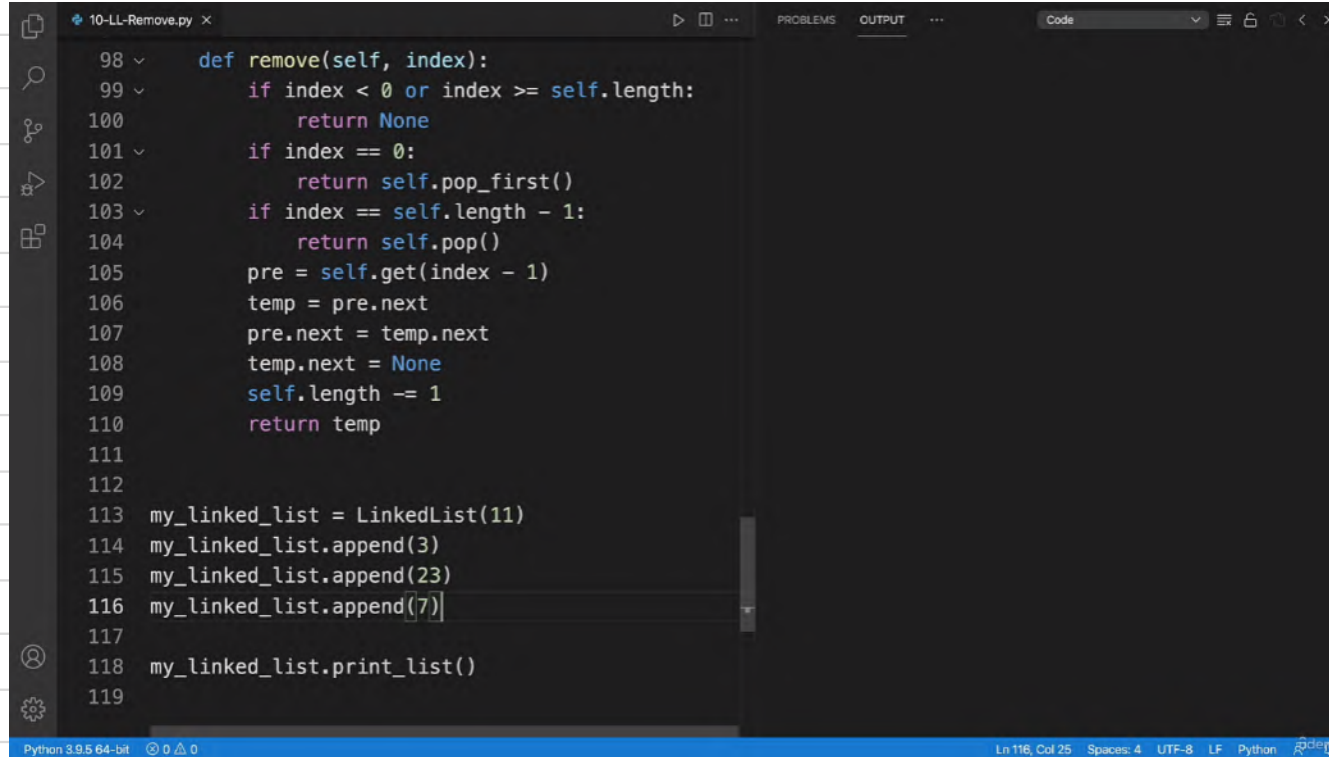

Linked List : Remove



Linked List : Remove



Linked List : Remove



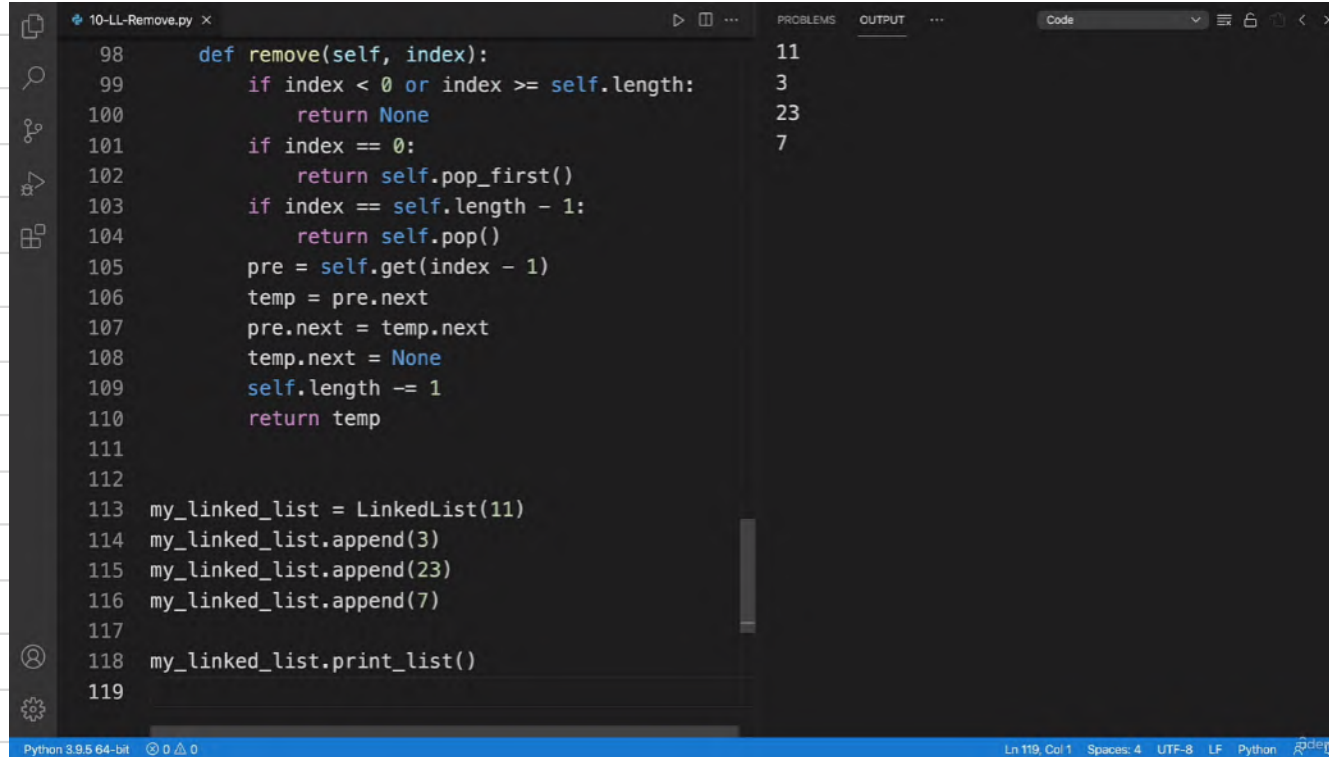
```
10-LL-Remove.py x
def remove(self, index):
    if index < 0 or index >= self.length:
        return None
    if index == 0:
        return self.pop_first()
    if index == self.length - 1:
        return self.pop()
    pre = self.get(index - 1)
    temp = pre.next
    pre.next = temp.next
    temp.next = None
    self.length -= 1
    return temp

my_linked_list = LinkedList(11)
my_linked_list.append(3)
my_linked_list.append(23)
my_linked_list.append(7)

my_linked_list.print_list()
```

Python 3.9.5 64-bit 0 0 0 Ln 116, Col 25 Spaces: 4 UTF-8 LF Python

Linked List : Remove

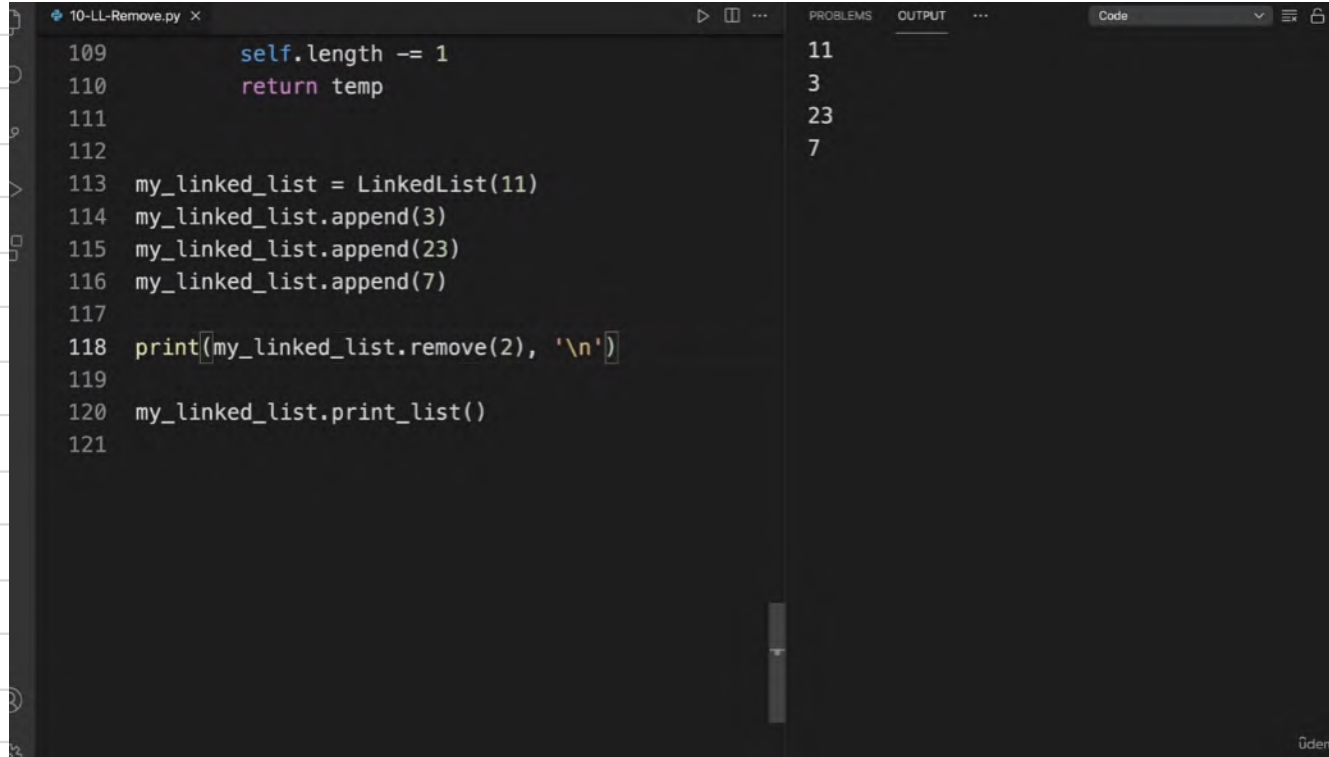


```
10-LL-Remove.py x
def remove(self, index):
    if index < 0 or index >= self.length:
        return None
    if index == 0:
        return self.pop_first()
    if index == self.length - 1:
        return self.pop()
    pre = self.get(index - 1)
    temp = pre.next
    pre.next = temp.next
    temp.next = None
    self.length -= 1
    return temp

my_linked_list = LinkedList(11)
my_linked_list.append(3)
my_linked_list.append(23)
my_linked_list.append(7)
my_linked_list.print_list()
```

Python 3.9.5 64-bit 0 0 0 Ln 119, Col 1 Spaces: 4 UTF-8 LF Python

Linked List : Remove



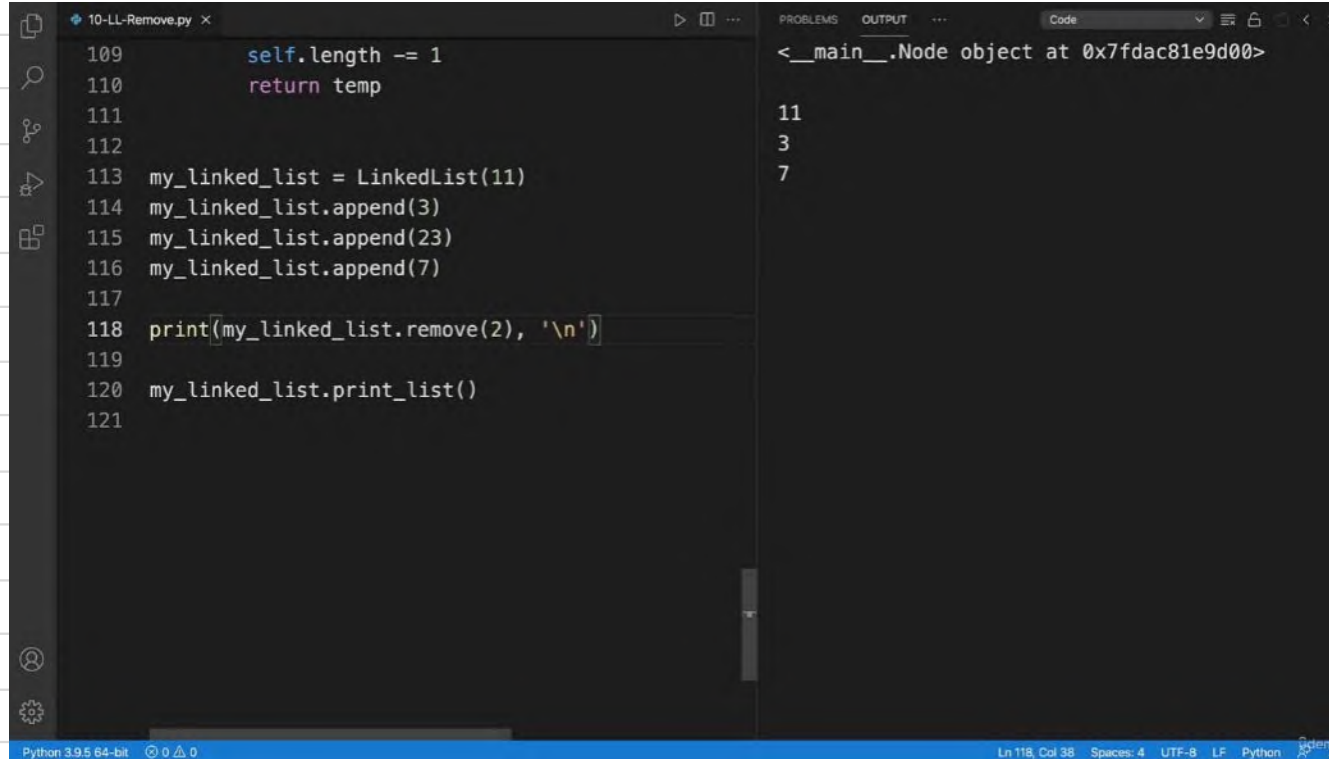
The image shows a code editor window with a file named '10-LL-Remove.py'. The code defines a linked list and performs a removal operation. The output panel on the right shows the state of the list before and after the removal.

```
109         self.length -= 1
110         return temp
111
112
113 my_linked_list = LinkedList(11)
114 my_linked_list.append(3)
115 my_linked_list.append(23)
116 my_linked_list.append(7)
117
118 print(my_linked_list.remove(2), '\n')
119
120 my_linked_list.print_list()
121
```

OUTPUT

```
11
3
23
7
```

Linked List : Remove



The screenshot shows a Python IDE with a file named '10-LL-Remove.py'. The code defines a linked list and performs a removal operation. The output window shows the state of the list after the removal.

```
109     self.length -= 1
110     return temp
111
112
113 my_linked_list = LinkedList(11)
114 my_linked_list.append(3)
115 my_linked_list.append(23)
116 my_linked_list.append(7)
117
118 print(my_linked_list.remove(2), '\n')
119
120 my_linked_list.print_list()
121
```

OUTPUT

```
<__main__.Node object at 0x7fdac81e9d00>
11
3
7
```

Python 3.9.5 64-bit | 0 0 0 | Ln 118, Col 38 | Spaces: 4 | UTF-8 | LF | Python

Linked List : Remove

```
        self.length -= 1  
        return temp.value
```

```
my_linked_list = LinkedList(11)  
my_linked_list.append(3)  
my_linked_list.append(23)  
my_linked_list.append(7)  
  
print(my_linked_list.remove(2), '\n')  
  
my_linked_list.print_list()
```

<__main__.Node object

11
3
7

Linked List : Remove

```
        self.length -= 1
        return temp.value

my_linked_list = LinkedList(11)
my_linked_list.append(3)
my_linked_list.append(23)
my_linked_list.append(7)

print(my_linked_list.remove(2), '\n')

my_linked_list.print_list()
```

23

11

3

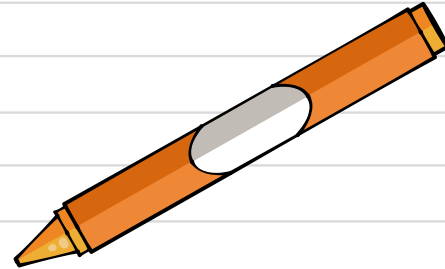
7



14



Linked List Reverse



Linked List : Reverse

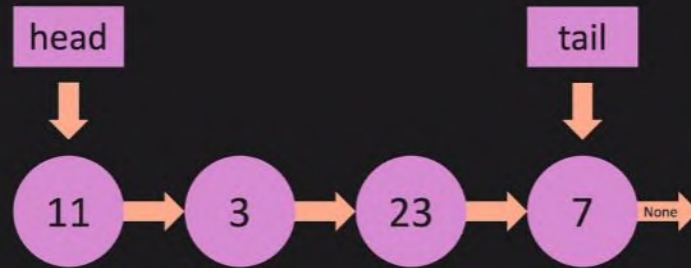
- Apa itu Reverse...?

Linked List : Reverse

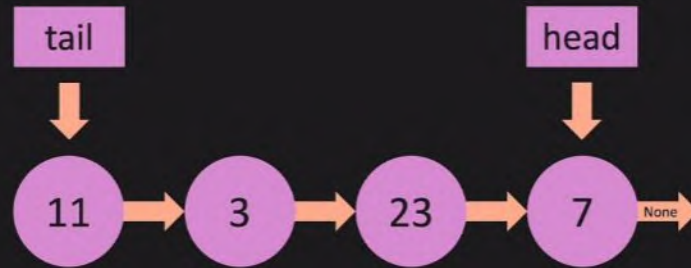
```
def reverse(self):
```

Diketahui suatu list

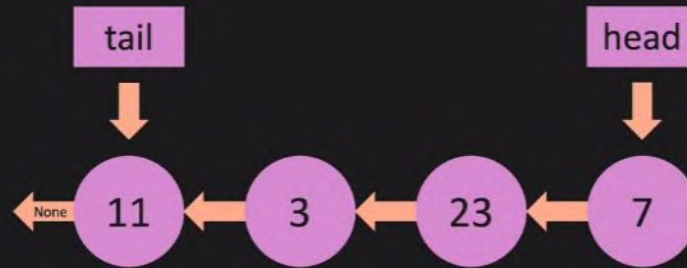
Linked List : Reverse



Linked List : Reverse



Linked List : Reverse



Terima Kasih



Ada Pertanyaan?