Data Strcture
LAB 6

PREPARED BY:
AYOOB ABDULMUNEM
Ahmed Eskander Mezher

Lecture Outline: -

1- Insert a node in the specific index of the linked list

2- Delete a node from the beginning in the liked list

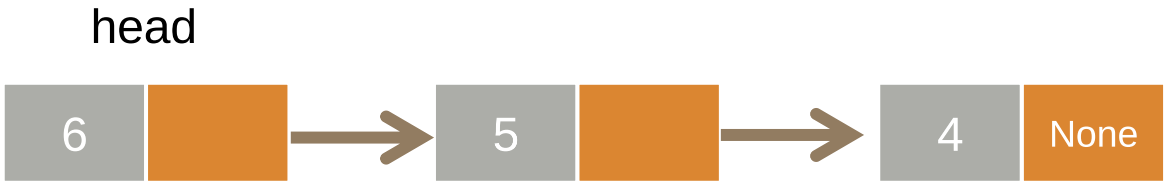
3- Delete a node from the end in the liked list

4- Delete a node from the specific index in the linked list

1- Insert a node in the specific index of the linked list

2- Insert a node in the specific index of the linked list

```
def insert_mid(self,data,index):
    nod = Node(data)
    p=self.head
    for i in range (index-1):
        if p.next != None:
            p=p.next
        else:
            print("Error Out of range")
            return
    nod.next=p.next
    p.next=nod
```

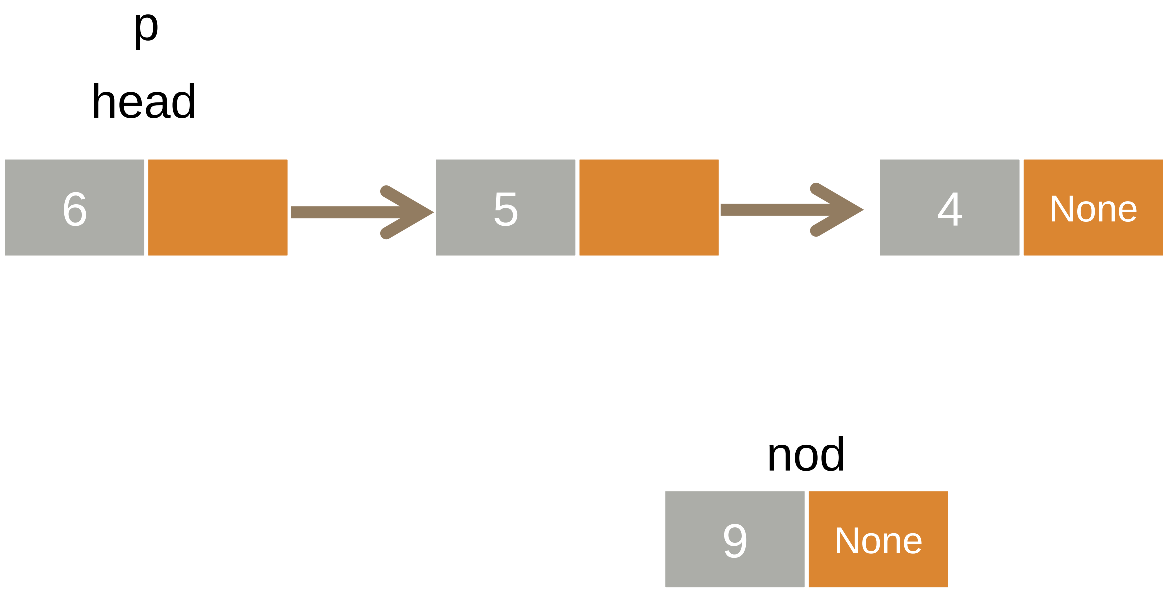


```
nod=linklist()
nod.insert_begin(4)
nod.insert_begin(5)
nod.insert_begin(6)
nod.insert_mid(9,2)
```

2- Insert a node in the specific index of the linked list

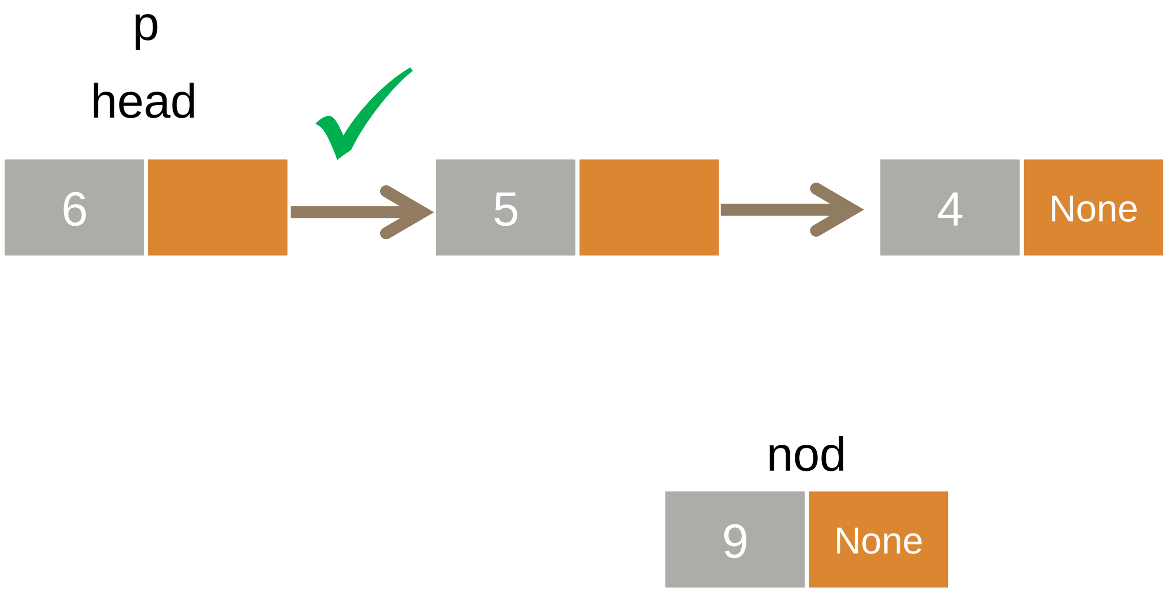
```
def insert_mid(self, data,index):
    nod = Node(data)
    p=self.head
    for i in range (index-1):
        if p.next != None:
            p=p.next
        else:
            print("Error Out of range")
            return
    nod.next=p.next
    p.next=nod
```

```
nod=linklist()
nod.insert_begin(4)
nod.insert_begin(5)
nod.insert_begin(6)
nod.insert_mid(9,2)
```



2- insert a node in the specific index of the linked list

```
def insert_mid(self, data,index):
    nod = Node(data)
    p=self.head
    for i in range (index-1):(0 → 1)
        if p.next != None:
            p=p.next
        else:
            print("Error Out of range")
            return
    nod.next=p.next
    p.next=nod
```



p

head

✓

6 | | 5 | | 4 | None

nod

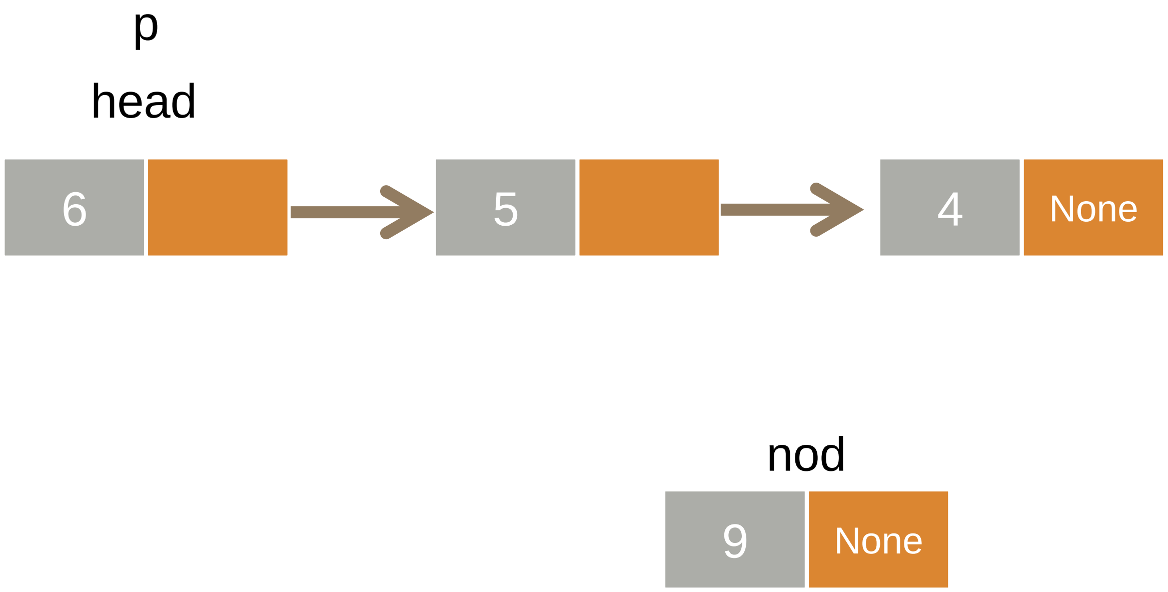
9 | None

```
nod=linklist()
nod.insert_begin(4)
nod.insert_begin(5)
nod.insert_begin(6)
nod.insert_mid(9,2)
```

2- insert a node in the specific index of the linked list

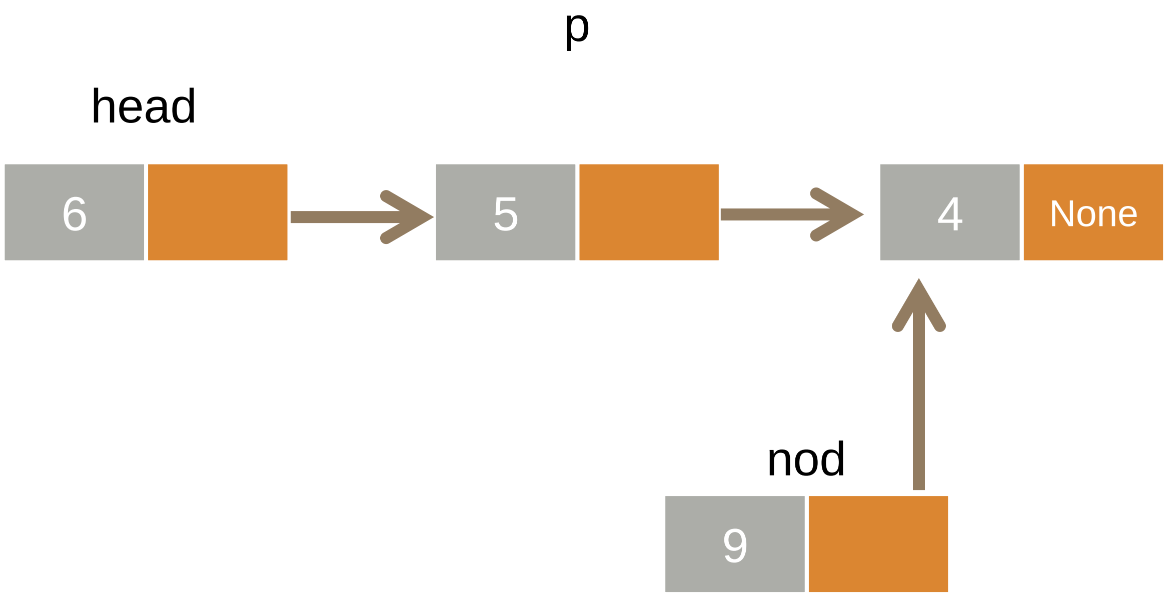
```
def insert_mid(self, data,index):
    nod = Node(data)
    p=self.head
    for i in range (index-1):
        if p.next != None:
            p=p.next
        else:
            print("Error Out of range")
            return
    nod.next=p.next
    p.next=nod
```

```
nod=linklist()
nod.insert_begin(4)
nod.insert_begin(5)
nod.insert_begin(6)
nod.insert_mid(9,2)
```



2- Insert a node in the specific index of the linked list

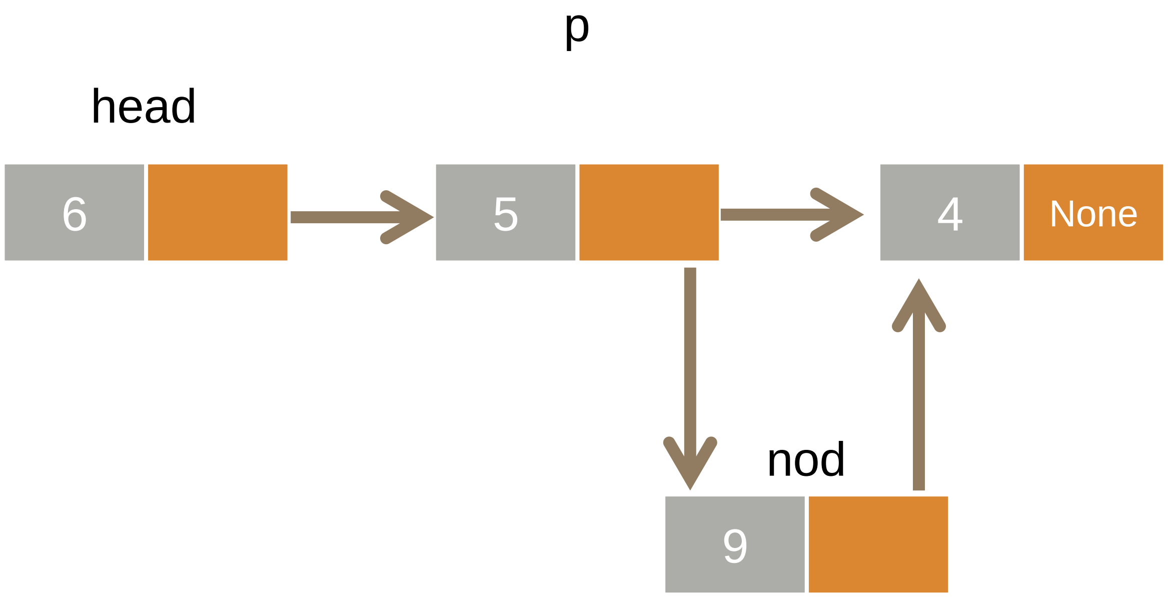
```
def insert_mid(self, data,index):
    nod = Node(data)
    p=self.head
    for i in range (index-1):
        if p.next != None:
            p=p.next
        else:
            print("Error Out of range")
            return
    nod.next=p.next
    p.next=nod

nod=linklist()
nod.insert_begin(4)
nod.insert_begin(5)
nod.insert_begin(6)
nod.insert_mid(9,2)
```



2- insert a node in the specific index of the linked list

```
def insert_mid(self, data,index):
    nod = Node(data)
    p=self.head
    for i in range (index-1):
        if p.next != None:
            p=p.next
        else:
            print("Error Out of range")
            return
    nod.next=p.next
    p.next=nod
```

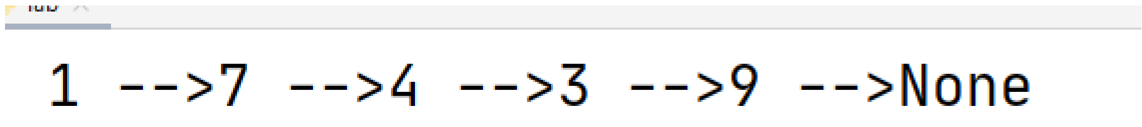
```
nod=linklist()
nod.insert_begin(4)
nod.insert_begin(5)
nod.insert_begin(6)
nod.insert_mid(9,2)
```



```
class Node:
    def __init__(self,data,next=None):
        self.data = data
        self.next = next
class linklist:
    def __init__(self,head=None):
        self.head=head
    def insert_begin(self,data):
        nod=Node(data)
        if self.head==None:
            self.head=nod
            return
        nod.next=self.head
        self.head=nod
    def insert_mid(self,data,index):
        nod = Node(data)
        p=self.head
        for i in range (index-1):
            if p.next != None:
                p=p.next
            else:
                print("Error Out of range")
                return
        nod.next=p.next
        p.next=nod
```

```
def insert_end(self,data):
        nod=Node(data)
        if self.head==None:
            self.head=nod
            return
        p=self.head
        while p.next !=None:
            p=p.next
        p.next=nod
    def insert(self,data,index=None):
        if index==0:
            self.insert_begin(data)
        elif index==None:
            self.insert_end(data)
        else:
            self.insert_mid(data,index)
    def printNod(self):
        p=self.head
        while p!=None:
            print(p.data,"-->",end="")
            p=p.next
        print("None")
nod=linklist()
nod.insert(4)
nod.insert(7,0)
nod.insert(9)
nod.insert(3,2)
nod.insert(1,0)
nod.printNod()
```

Output

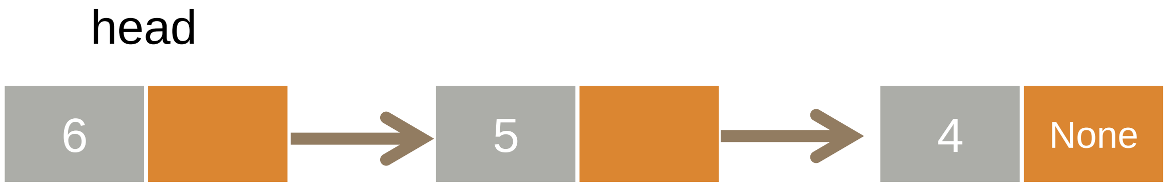


```
1 -->7 -->4 -->3 -->9 -->None
```

2- Delete a node from the beginning in the liked list

Delete a node from the beginning in the liked list

```
def del_begin(self):
    if self.length() <= 1:
        self.head = None
        return
    p=self.head
    self.head=p.next
    del p
```

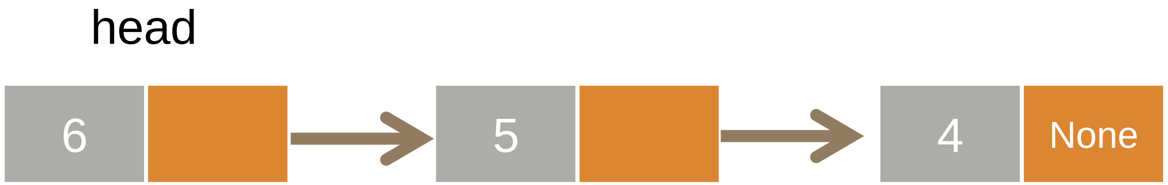


```
nod=linklist()
nod.insert(6)
nod.insert(5)
nod.insert(4)
nod.del_begin()
nod.del_begin()
nod.del_begin()
```

Delete a node from the beginning in the liked list

```
def del_begin(self):
    if self.length() <= 1: ❌
        self.head = None
        return
    p=self.head
    self.head=p.next
    del p
```

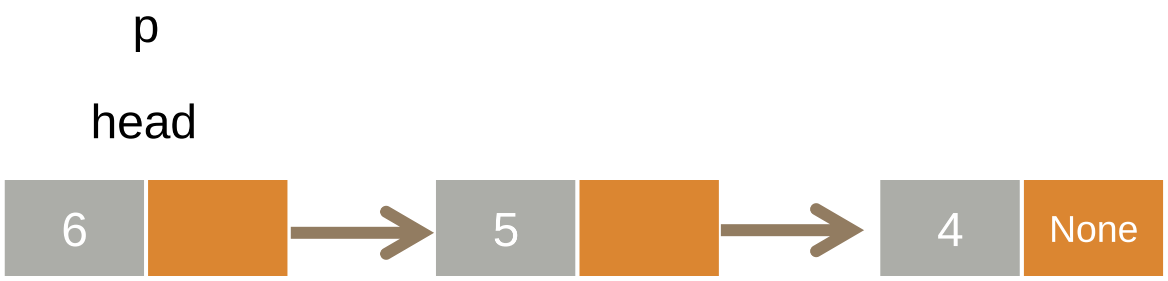
length = 3



```
nod=linklist()
nod.insert(6)
nod.insert(5)
nod.insert(4)
→ nod.del_begin()
nod.del_begin()
nod.del_begin()
```

Delete a node from the beginning in the liked list

```
def del_begin(self):
    if self.length() <= 1:
        self.head = None
        return
    p=self.head
    self.head=p.next
    del p
```

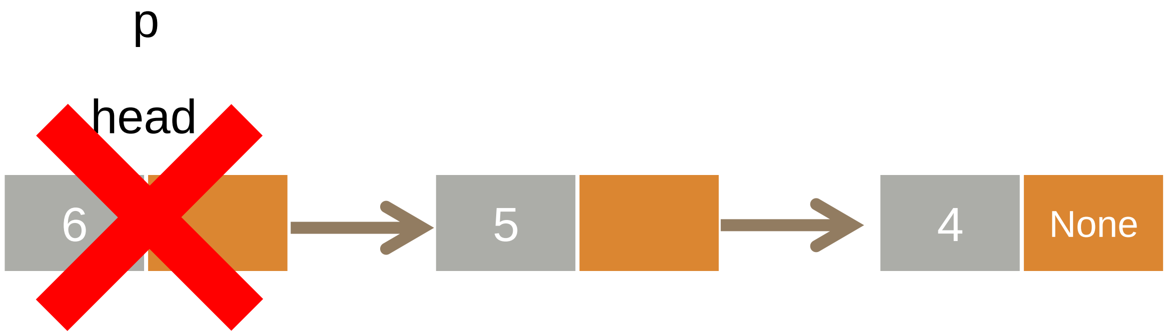


```
nod=linklist()
nod.insert(6)
nod.insert(5)
nod.insert(4)
nod.del_begin()
nod.del_begin()
nod.del_begin()
```

Delete a node from the beginning in the liked list

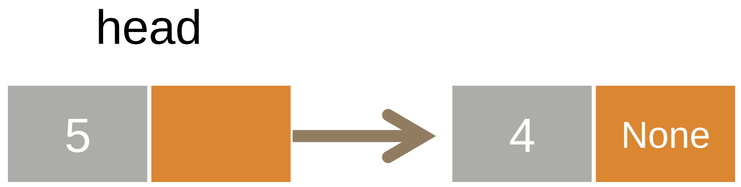
```
def del_begin(self):
    if self.length() <= 1:
        self.head = None
        return
    p=self.head
    self.head=p.next
    del p
```

```
nod=linklist()
nod.insert(6)
nod.insert(5)
nod.insert(4)
nod.del_begin()
nod.del_begin()
nod.del_begin()
```



Delete a node from the beginning in the liked list

```
def del_begin(self):
    if self.length() <= 1:
        self.head = None
        return
    p=self.head
    self.head=p.next
    del p
```



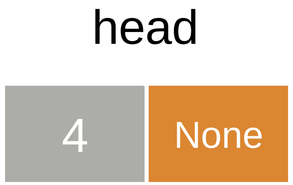
```
nod=linklist()
nod.insert(6)
nod.insert(5)
nod.insert(4)
nod.del_begin()
nod.del_begin()
nod.del_begin()
```

Repeat all the previous steps

Delete a node from the beginning in the liked list

```
def del_begin(self):
    if self.length() <= 1:  ✓
        self.head = None
        return
    p=self.head
    self.head=p.next
    del p
```

length = 1



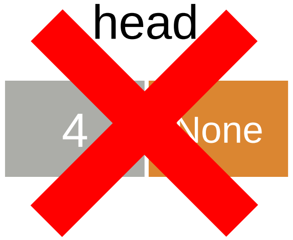
```
nod=linklist()
nod.insert(6)
nod.insert(5)
nod.insert(4)
nod.del_begin()
nod.del_begin()
→ nod.del_begin()
```

Delete a node from the beginning in the liked list

```
def del_begin(self):
    if self.length() <= 1:
        self.head = None
        return
    p=self.head
    self.head=p.next
    del p
```

```
nod=linklist()
nod.insert(6)
nod.insert(5)
nod.insert(4)
nod.del_begin()
nod.del_begin()
nod.del_begin()
```

head= None



Delete a node from the beginning in the liked list

```
def del_begin(self):
    if self.length() <= 1:
        self.head = None
        return
    p=self.head
    self.head=p.next
    del p
```

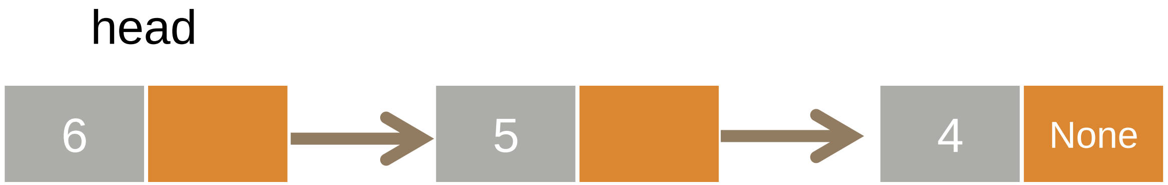
```
nod=linklist()
nod.insert(6)
nod.insert(5)
nod.insert(4)
nod.del_begin()
nod.del_begin()
nod.del_begin()
```

head= None

3- Delete a node from the end in the liked list

Delete a node from the end in the liked list

```
def del_end(self):
    if self.length()<=1:
        self.head=None
        return
    p = self.head
    while p.next.next != None:
        p = p.next
    q = p.next
    p.next = None
    del q
```

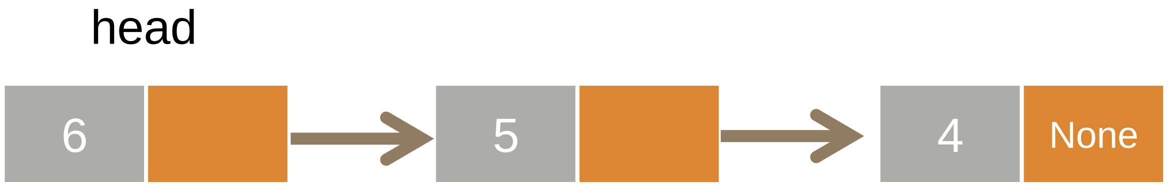


```
nod=linklist()
nod.insert(6)
nod.insert(5)
nod.insert(4)
nod.del_end()
nod.del_end()
nod.del_end()
```

Delete a node from the end in the liked list

```
def del_end(self):
    if self.length()<=1:
        self.head=None
        return
    p = self.head
    while p.next.next != None:
        p = p.next
    q = p.next
    p.next = None
    del q
```

✕ length = 3

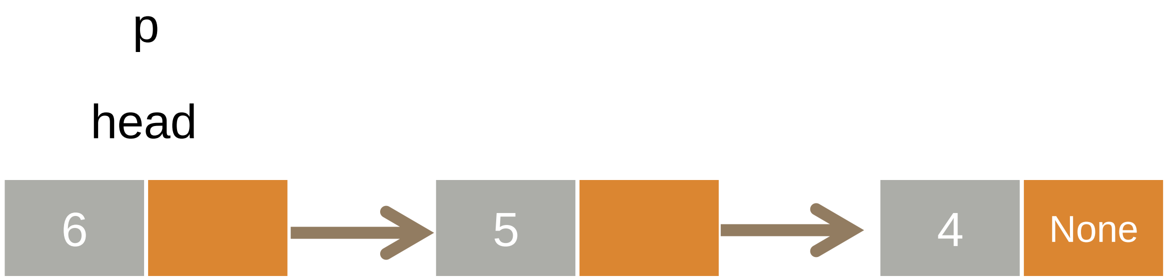


```
  nod=linklist()
  nod.insert(6)
  nod.insert(5)
  nod.insert(4)
→ nod.del_end()
  nod.del_end()
  nod.del_end()
```

Delete a node from the end in the liked list

```
def del_end(self):
    if self.length()<=1:
        self.head=None
        return
    p = self.head
    while p.next.next != None:
        p = p.next
    q = p.next
    p.next = None
    del q
```

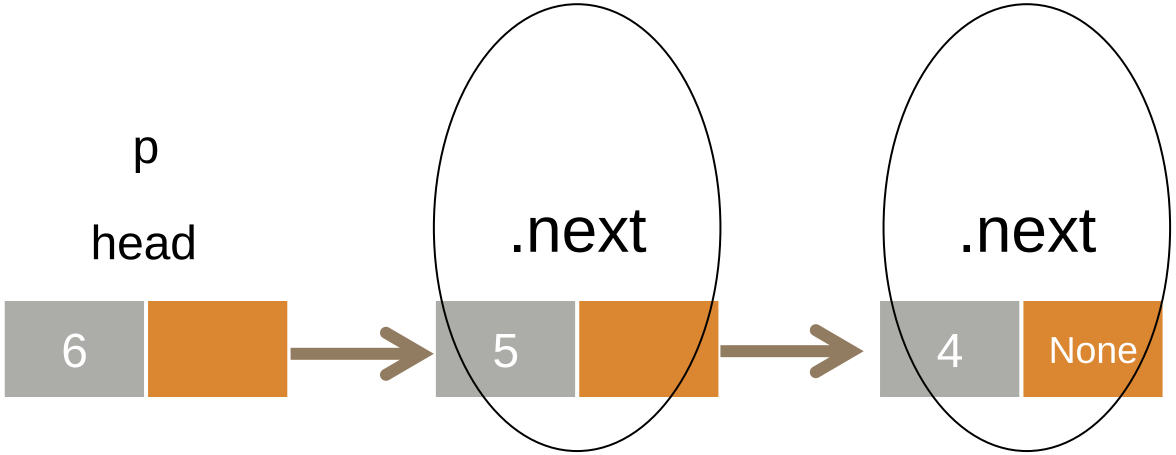
```
nod=linklist()
nod.insert(6)
nod.insert(5)
nod.insert(4)
nod.del_end()
nod.del_end()
nod.del_end()
```



Delete a node from the end in the liked list

```
def del_end(self):
    if self.length()<=1:
        self.head=None
        return
    p = self.head
    while p.next.next != None: ✓
        p = p.next
    q = p.next
    p.next = None
    del q
```

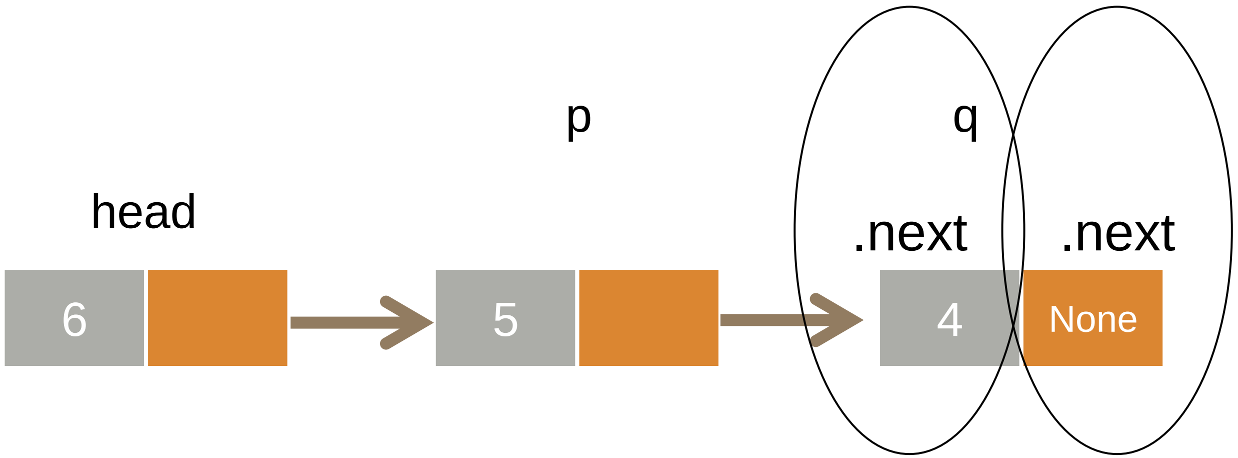
```
nod=linklist()
nod.insert(6)
nod.insert(5)
nod.insert(4)
nod.del_end()
nod.del_end()
nod.del_end()
```



Delete a node from the end in the liked list

```
def del_end(self):
    if self.length()<=1:
        self.head=None
        return
    p = self.head
    while p.next.next != None:
        p = p.next
    q = p.next
    p.next = None
    del q
```

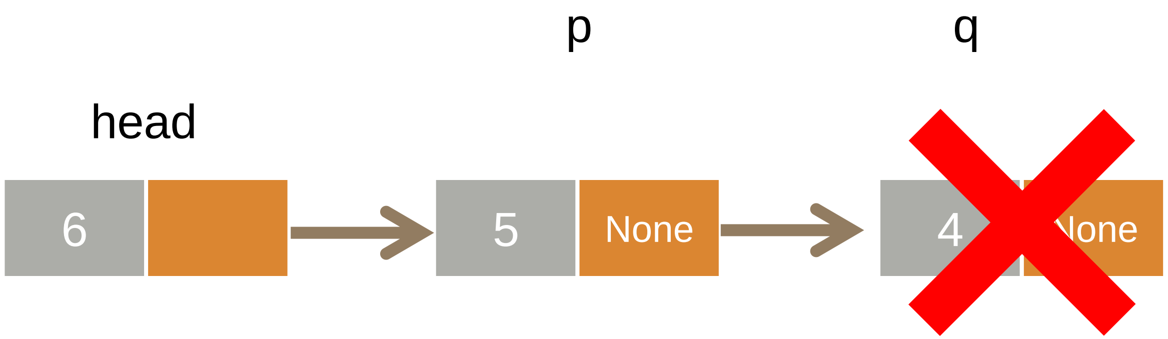
```
nod=linklist()
nod.insert(6)
nod.insert(5)
nod.insert(4)
nod.del_end()
nod.del_end()
nod.del_end()
```



Delete a node from the end in the liked list

```
def del_end(self):
    if self.length()<=1:
        self.head=None
        return
    p = self.head
    while p.next.next != None:
        p = p.next
    q = p.next
    p.next = None
    del q
```

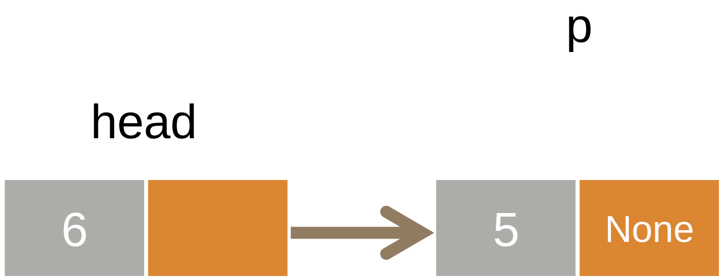
```
nod=linklist()
nod.insert(6)
nod.insert(5)
nod.insert(4)
nod.del_end()
nod.del_end()
nod.del_end()
```



Delete a node from the end in the liked list

```
def del_end(self):
    if self.length()<=1:
        self.head=None
        return
    p = self.head
    while p.next.next != None:
        p = p.next
    q = p.next
    p.next = None
    del q
```

✗ length = 2

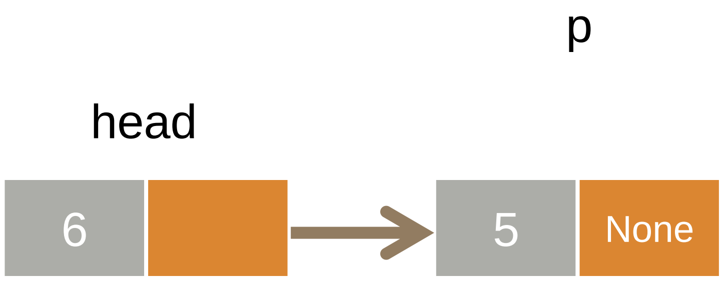


```
nod=linklist()
nod.insert(6)
nod.insert(5)
nod.insert(4)
nod.del_end()
→ nod.del_end()
nod.del_end()
```

Delete a node from the end in the liked list

```
def del_end(self):
    if self.length()<=1:
        self.head=None
        return
    p = self.head
    while p.next.next != None:
        p = p.next
    q = p.next
    p.next = None
    del q
```

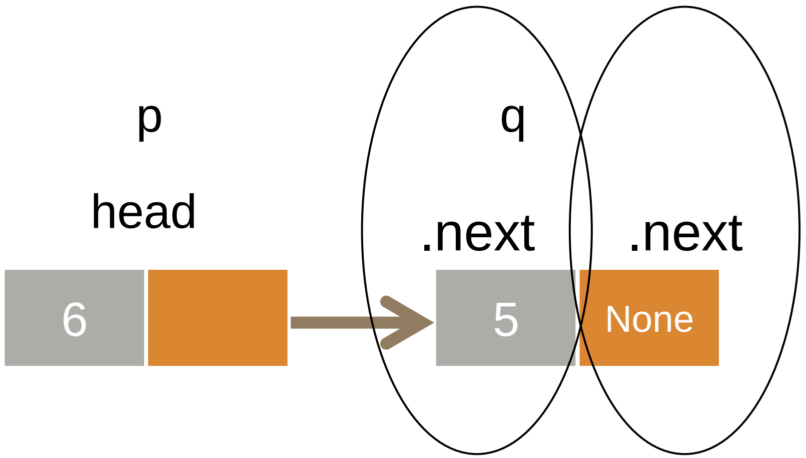
✗ length = 2



```
nod=linklist()
nod.insert(6)
nod.insert(5)
nod.insert(4)
nod.del_end()
nod.del_end()
nod.del_end()
```

Delete a node from the end in the liked list

```
def del_end(self):
    if self.length()<=1:
        self.head=None
        return
    p = self.head
    while p.next.next != None:
        p = p.next
    q = p.next
    p.next = None
    del q
```

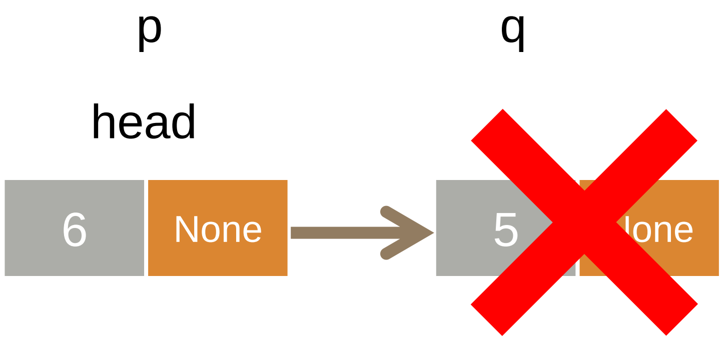


```
nod=linklist()
nod.insert(6)
nod.insert(5)
nod.insert(4)
nod.del_end()
nod.del_end()
nod.del_end()
```

Delete a node from the end in the liked list

```
def del_end(self):
    if self.length()<=1:
        self.head=None
        return
    p = self.head
    while p.next.next != None:
        p = p.next
    q = p.next
    p.next = None
    del q
```

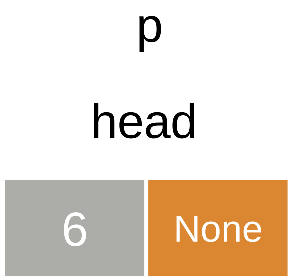
```
nod=linklist()
nod.insert(6)
nod.insert(5)
nod.insert(4)
nod.del_end()
nod.del_end()
nod.del_end()
```



Delete a node from the end in the liked list

```
def del_end(self):
    if self.length()<=1:
        self.head=None
        return
    p = self.head
    while p.next.next != None:
        p = p.next
    q = p.next
    p.next = None
    del q
```

✓ length = 1

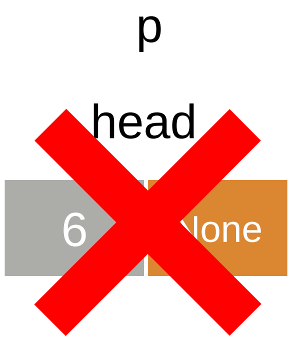


```
    nod=linklist()
    nod.insert(6)
    nod.insert(5)
    nod.insert(4)
    nod.del_end()
    nod.del_end()
→   nod.del_end()
```

Delete a node from the end in the liked list

```
def del_end(self):
    if self.length()<=1:
        self.head=None
        return
    p = self.head
    while p.next.next != None:
        p = p.next
    q = p.next
    p.next = None
    del q
```

```
nod=linklist()
nod.insert(6)
nod.insert(5)
nod.insert(4)
nod.del_end()
nod.del_end()
nod.del_end()
```



head= None

Delete a node from the end in the liked list

```
def del_end(self):
    if self.length()<=1:
        self.head=None
        return
    p = self.head
    while p.next.next != None:
        p = p.next
    q = p.next
    p.next = None
    del q
```

head= None

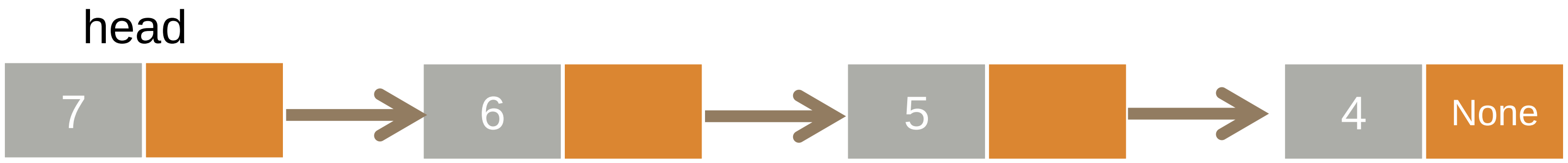
```
nod=linklist()
nod.insert(6)
nod.insert(5)
nod.insert(4)
nod.del_end()
nod.del_end()
nod.del_end()
```

4- Delete a node from the specific index in the linked list

Delete a node from the specific index in the linked list

```
def del_mid(self,index):
    p = self.head
    if index>0 and index<self.length()-1:
        for i in range(index-1):
            p=p.next
    else:
        print("Error Out of range")
        return
    q=p.next
    p.next=q.next
    del q
```

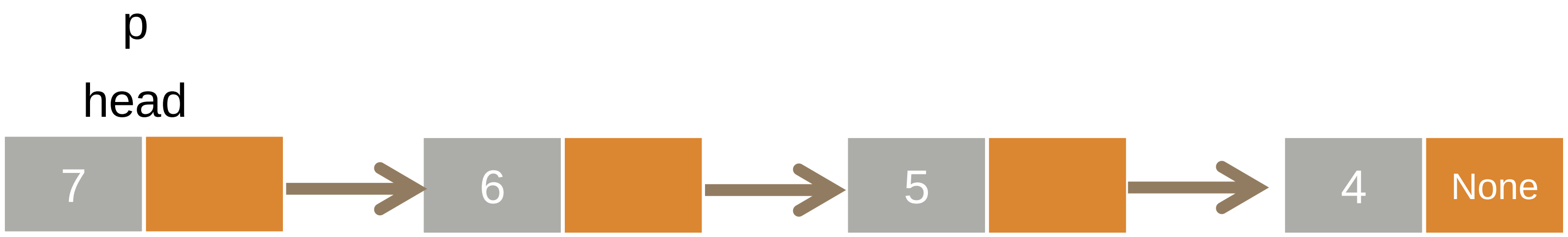
```
nod=linklist()
nod.insert(7)
nod.insert(6)
nod.insert(5)
nod.insert(4)
nod.del_mid(2)
```



Delete a node from the specific index in the linked list

```
def del_mid(self,index):
    p = self.head
    if index>0 and index<self.length()-1:✓   2>0   and   2<3
        for i in range(index-1):
            p=p.next
    else:
        print("Error Out of range")
        return
    q=p.next
    p.next=q.next
    del q
```

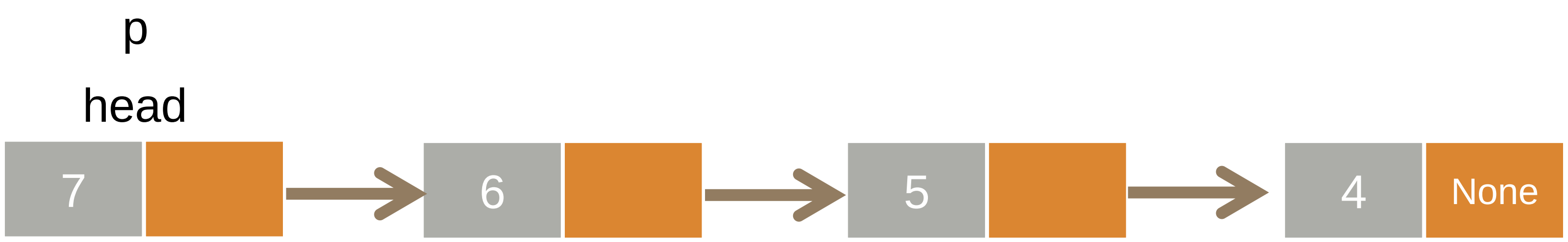
```
nod=linklist()
nod.insert(7)
nod.insert(6)
nod.insert(5)
nod.insert(4)
nod.del_mid(2)
```



Delete a node from the specific index in the linked list

```
def del_mid(self,index):
    p = self.head
    if index>0 and index<self.length()-1:
        for i in range(index-1):          0 to 1
            p=p.next
    else:
        print("Error Out of range")
        return
    q=p.next
    p.next=q.next
    del q
```

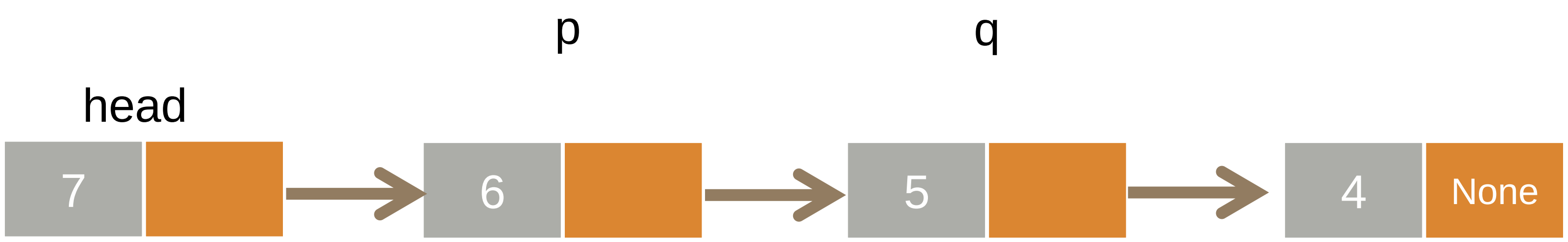
```
nod=linklist()
nod.insert(7)
nod.insert(6)
nod.insert(5)
nod.insert(4)
nod.del_mid(2)
```



Delete a node from the specific index in the linked list

```
def del_mid(self,index):
    p = self.head
    if index>0 and index<self.length()-1:
        for i in range(index-1):
            p=p.next
    else:
        print("Error Out of range")
        return
    q=p.next
    p.next=q.next
    del q
```

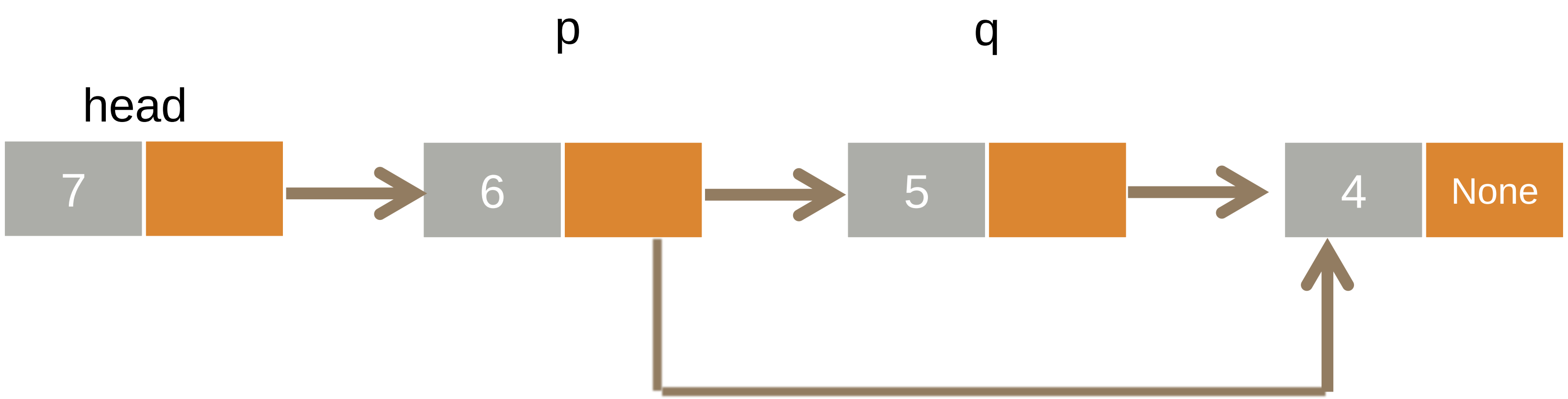
```
nod=linklist()
nod.insert(7)
nod.insert(6)
nod.insert(5)
nod.insert(4)
nod.del_mid(2)
```



Delete a node from the specific index in the linked list

```
def del_mid(self,index):
    p = self.head
    if index>0 and index<self.length()-1:
        for i in range(index-1):
            p=p.next
    else:
        print("Error Out of range")
        return
    q=p.next
    p.next=q.next
    del q
```

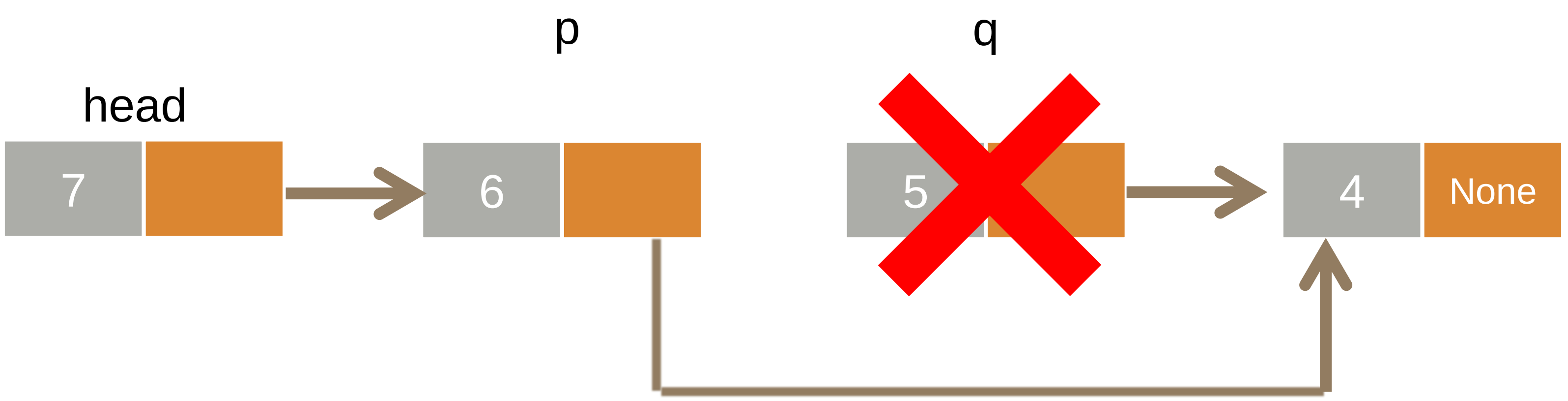
```
nod=linklist()
nod.insert(7)
nod.insert(6)
nod.insert(5)
nod.insert(4)
nod.del_mid(2)
```



Delete a node from the specific index in the linked list

```
def del_mid(self,index):
    p = self.head
    if index>0 and index<self.length()-1:
        for i in range(index-1):
            p=p.next
    else:
        print("Error Out of range")
        return
    q=p.next
    p.next=q.next
    del q
```

```
nod=linklist()
nod.insert(7)
nod.insert(6)
nod.insert(5)
nod.insert(4)
nod.del_mid(2)
```



Thank you