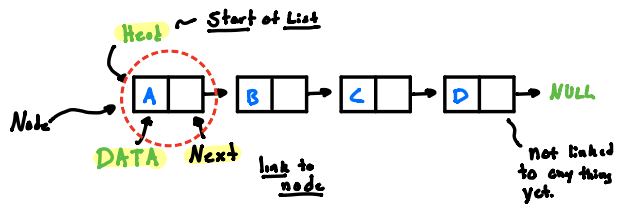


Singly Linked Lists

Insertion



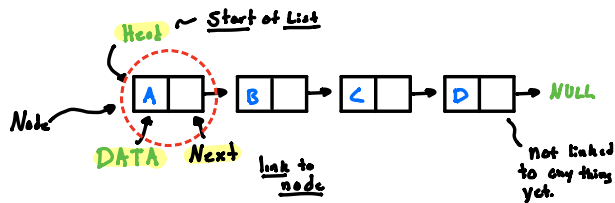
Array vs. Linked List

	Array	Linked List
Insertion/Deletion	$O(n)$	$O(1)$
Access Element	$O(1)$	$O(n)$
Contiguous Memory?	Yes	No

Move entire array for array

have to have pointer to traverse until we find the Value we are looking for

Python Implementation of a Linked List



Insertion

- 3 ways
- Append to end of list
- Prepend to Beginning
- Insert after Node

Append to end of the linked list



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

Node Class

```
class LinkedList:
    def __init__(self):
        self.head = None

    def print_list(self):
        '''Printing the linked list'''
        cur_node = self.head
        # while current node is not null
        while cur_node:
            print(cur_node.data)
            cur_node = cur_node.next
```

prints our linked list

```
def append(self, data):
    '''Adding an element to the back of the linked list'''
    new_node = Node(data)
```

← Turns our element into a node

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

if there isn't a head yet then our element becomes one.

```
last_node = self.head
while last_node.next:
    last_node = last_node.next
last_node.next = new_node
```

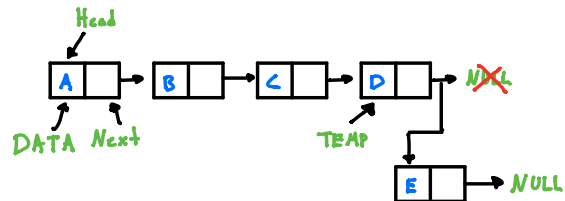
```
l1 = LinkedList()
l1.append("A")
l1.append("B")
l1.append("C")
l1.append("D")
```

```
print("Initial list")
l1.print_list()
```

Initial list
A
B
C
D

Append node to end of the list

Process:

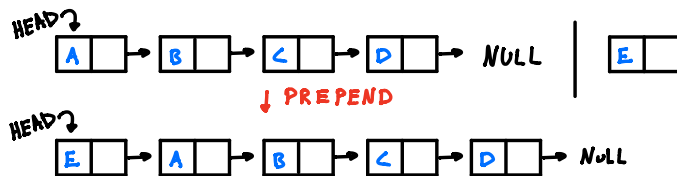


Keeps track of our head node, were at the start of the list (called last_node, because that's what it will eventually point to)

- We want to move through the list
 - Until it's pointing to NULL, then that's where we will input our new node.

- ① • While the next pointer is not NULL, we will continue in this loop
- What we're doing in this loop is moving the head pointer to the right
- After the loop has concluded, the last node will point to the last node

Add element to the beginning of the Linked List (Prepend)



- Create new node
- Set new node should point to A
- Move head from A to E

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

class LinkedList:
    def __init__(self):
        self.head = None

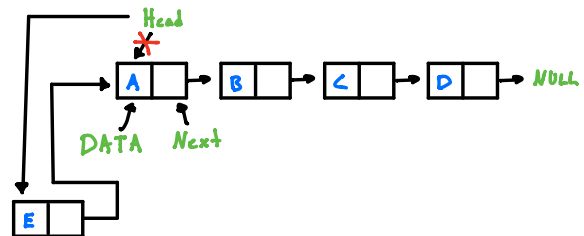
    def print_list(self):
        cur_node = self.head
        while cur_node:
            print(cur_node.data)
            cur_node = cur_node.next

    def append(self, data):
        new_node = Node(data)

        if self.head is None:
            self.head = new_node
            return

        last_node = self.head
        while last_node.next:
            last_node = last_node.next
        last_node.next = new_node
```

Process:



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

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

Create new node

- Set new node should point to the head of the list

Set head to the new node that we just added

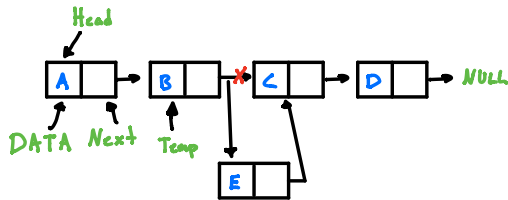
```
l1 = LinkedList()
l1.append("A")
l1.append("B")
l1.append("C")
l1.append("D")
```

```
l1.prepend("E")
```

```
l1.print_list()
```

E
A
B
C
D

Insert and element after a Node



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

class LinkedList:
    def __init__(self):
        self.head = None

    def print_list(self):
        cur_node = self.head
        while cur_node:
            print(cur_node.data)
            cur_node = cur_node.next

    def append(self, data):
        new_node = Node(data)

        if self.head is None:
            self.head = new_node
            return

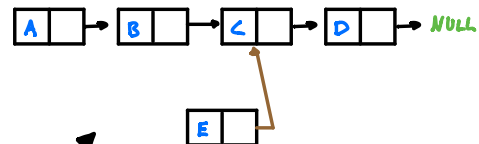
        last_node = self.head
        while last_node.next:
            last_node = last_node.next
        last_node.next = new_node

    def prepend(self, data):
        new_node = Node(data)

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

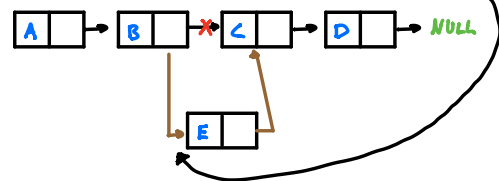
```
def insert_after_node(self, prev_node, data):
    if not prev_node:
        print("Previous node is not in the list")
        return

    new_node = Node(data)  ← Create new node
    new_node.next = prev_node.next
    prev_node.next = new_node
```



Setting pointer of our new node to next node

The Previous node now needs to point to our new node



```
l1 = LinkedList()
l1.append("A")
l1.append("B")
l1.append("C")
l1.append("D")
```

```
l1.insert_after_node(l1.head.next, "E")
```

```
l1.print_list()
```

A
B
E
C
D

This is node B