Data Structures

DLL Insertion

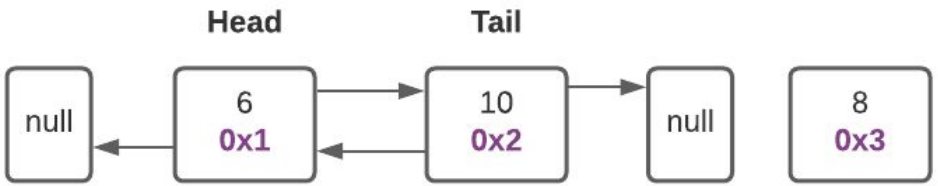
Mostafa S. Ibrahim
Teaching, Training and Coaching for more than a decade!

Artificial Intelligence & Computer Vision Researcher
PhD from Simon Fraser University - Canada
Bachelor / Msc from Cairo University - Egypt
Ex-(Software Engineer / ICPC World Finalist)

Insert end

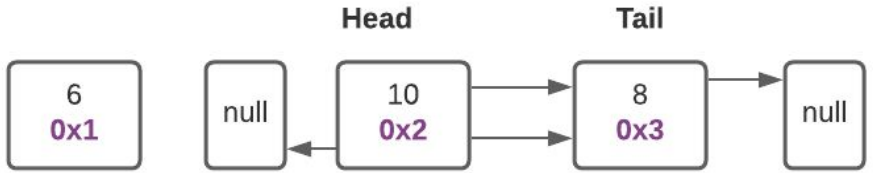
```
def insert_end(self, value):
    node = Node(value)
    self._add_node(node)

    if not self.head:
        self.head = self.tail = node
    else:
        self._link(self.tail, node)
        self.tail = node
```

```
@staticmethod
def _link(first, second):
    if first:
        first.next = second
    if second:
        second.prev = first
```

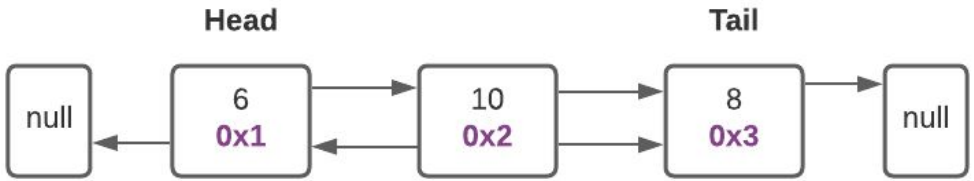


Insert front

- Also, exactly the same
- Just link properly




```
def insert_front(self, value):
    item = Node(value)
    self._add_node(item)

    self._link(item, self.head)
    self.head = item

    if self.length == 1:
        self.tail = self.head
```



Insert Sorted

- This same homework code.
- Can you rewrite the _embed_after()?

```
def insert_sorted(self, value):
    # 3 special cases for simplicity
    if not self.length or value <= self.head.data:
        self.insert_front(value)
    elif self.tail.data <= value:
        self.insert_end(value)
    else:
        prev, cur = None, self.head
        while cur:
            if value <= cur.data:
                self._embed_after(prev, value)
                break
            prev, cur = cur, cur.next
```

Insert Sorted

- Again, the same logic, but instead of next linking, we link the 2 directions
  - We have 3 nodes: A, B, C ⇒ link (B, C), then link (A, B)
  - Tip: the link function makes our life easier (code and logic)

```
def _embed_after(self, node, value):
    # Add a node with value between node and its next
    new_node = Node(value)
    self._add_node(new_node)
    self._link(new_node, node.next)
    self._link(node, new_node)
```

"Acquire knowledge and impart it to the people."

"Seek knowledge from the Cradle to the Grave."