

In [1]:

```
1 class Node():
2     def __init__(self, val):
3         self.value = val
4         self.next = None
5         self.prev = None
6
7 class DoublyLinkedList():
8     def __init__(self):
9         self.head = None
10        self.tail = None
11
12    def add_node(self, val):
13        new_node = Node(val)
14        if self.head == None:
15            self.head = new_node
16            self.tail = new_node
17        else:
18            current_node = self.head
19            while current_node.next != None:
20                current_node = current_node.next
21            current_node.next = new_node
22            new_node.prev = current_node
23            self.tail = new_node
24
25    def traverse_forward(self):
26        if self.head != None:
27            current_node = self.head
28            while current_node.next != None:
29                print ("Traverse forward current", current_node.value)
30                current_node = current_node.next
31            print ("Traverse forward current", current_node.value)
32        else:
33            print ("No nodes")
34            return False
35
36    def traverse_back(self):
37        if self.tail != None:
38            current_node = self.tail
39            while current_node.prev != None:
40                print ("Traverse backwards current", current_node.value)
41                current_node = current_node.prev
42            print ("Traverse backwards current", current_node.value)
43        else:
44            print ("No nodes")
45            return False
46
47    def remove_node_from_end(self):
48        if self.head != None:
49            current_node = self.head
50            while current_node.next.next != None:
51                current_node = current_node.next
52            current_node.next.prev = None
53            current_node.next = None
```

```

54         self.tail = current_node
55
56     def remove_node(self, val):
57         if self.head != None:
58             current_node = self.head
59             if self.head.value == val:
60                 self.head.next.prev = None
61                 self.head = self.head.next
62             elif self.tail.value == val:
63                 self.tail.prev.next = None
64                 self.tail = self.tail.prev
65             else:
66                 while current_node.next.value != val:
67                     current_node = current_node.next
68                     if current_node.next.value != val:
69                         print ("Value not in list")
70                         return False
71                 current_node.next.next.prev = current_node
72                 current_node.next = current_node.next.next
73
74     def insert_node_after(self, val, insert_val):
75         if self.head != None:
76             current_node = self.head
77             new_node = Node(insert_val)
78             if self.head.value == val:
79                 self.head.next.prev = new_node
80                 new_node.prev = self.head
81                 new_node.next = self.head.next
82                 self.head.next = new_node
83             elif self.tail.value == val:
84                 self.tail.next = new_node
85                 new_node.prev = self.tail
86                 self.tail = new_node
87             else:
88                 while current_node.value != val:
89                     current_node = current_node.next
90                     if current_node.value != val:
91                         print ("Value not in list")
92                         return False
93                 new_node.next = current_node.next
94                 current_node.next = new_node
95                 new_node.next.prev = new_node
96                 new_node.prev = current_node
97
98     def print_as_list(self):
99         value_list = []
100         if self.head != None:
101             current_node = self.head
102             while current_node.next != None:
103                 value_list.append(current_node.value)
104                 current_node = current_node.next
105             value_list.append(current_node.value)
106             print (value_list)
107         else:

```

```
108         print ("No nodes")
109         return False
110
111     dll = DoublyLinkedList()
112     dll.add_node(10)
113     dll.add_node(20)
114     dll.add_node(30)
115     dll.add_node(35)
116     dll.print_as_list()
117     # dll.remove_node_from_end()
118     # dll.remove_node(35)
119     # dll.print_as_list()
120     # dll.remove_node(10)
121     dll.print_as_list()
122     dll.insert_node_after(35, 25)
123     dll.print_as_list()
124
125
```

[10, 20, 30, 35]

[10, 20, 30, 35]

[10, 20, 30, 35, 25]

In []:

1