# Algorithms on Singly LinkedList

1. Write algorithms for the following:
    a. **Front**

    Step 1: Create a new element with the given value.
    Step 2: Check whether the list is empty (start == null).
    Step 3: If the list is empty:
        Set newElement -> link = NULL
        Set start = newElement
    Else:
        Set newElement -> link = start
        Set start = newElement

    b. **End**

    Step 1: Create a new node with the given value.
    Step 2: Check whether the linked list is empty (head == NULL).
    Step 3: If it is empty:
        Set new node -> next = NULL
        Set head = new node
    Else:
        Initialize temp = head
        While temp -> next != NULL (i.e., temp is not the last node):
            Move temp to the next node (temp = temp -> next)
        Set temp -> next = new node
        Set new node -> next = NULL

    c. **In between.**

        Step 1: Create a new node containing the given value.
        Step 2: Check if the linked list is empty (head == NULL).
        Step 3: If the list is empty:
                Set newNode -> next = NULL
                Set head = newNode
        Else:
            Initialize a temporary pointer temp = head and a counter position = 0
            While temp is not NULL and position < desired_position - 1:
                Move temp to the next node (temp = temp -> next)
                Increment position by 1
            If temp becomes NULL before reaching the desired position:
                Display the message "Position out of bounds"
                Exit the function
            Set newNode -> next = temp -> next
            Set temp -> next = newNode

2. Create a Python program to insert an element in the singly linked list:

    a. Front

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

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

    def insertEle(self,data):
        newnode = Node(data)
        newnode.next = self.head
        self.head = newnode

    def display(self):
        current = self.head
        while current:
            print(current.data,end=" -> ")
            current = current.next
        print("NULL")

l = LinkedList()
l.insertEle(10)
l.display()
```

Output:

```
10 -> NULL
```

    **b. End**

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

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

    def insertEnd(self,data):
        new_node = Node(data)
        if self.head is None:
                self.head = new_node
                return
        temp = self.head
```

```
        while temp.next:
                temp = temp.next

        temp.next = new_node

    def insertEle(self,data):
        newnode = Node(data)
        newnode.next = self.head
        self.head = newnode

    def display(self):
        current = self.head
        while current:
            print(current.data,end=" -> ")
            current = current.next
        print("NULL")

l = LinkedList()
l.insertEle(30)
l.insertEnd(190)
l.insertEnd(0)
l.insertEnd(90)
l.insertEnd(80)
l.display()
```

Output:

```
30 -> 190 -> 0 -> 90 -> 80 -> NULL
```

**c. In Between**

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

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

    def insertEnd(self,data):
        new_node = Node(data)
        if self.head is None:
                self.head = new_node
                return
        temp = self.head
        while temp.next:
                temp = temp.next

        temp.next = new_node
```

```
    def insertEle(self,data):
        newnode = Node(data)
        newnode.next = self.head
        self.head = newnode

    def insertAtPos(self, data, position):
        new_node = Node(data)

        if position == 1:
            new_node.next = self.head
            self.head = new_node
            return

        temp = self.head
        for _ in range(position - 2):
            if temp is None:
                print("Position out of bounds")
                return
            temp = temp.next

        if temp is None:
            print("Position out of bounds")
            return

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

    def display(self):
        current = self.head
        while current:
            print(current.data,end=" -> ")
            current = current.next
        print("NULL")

l = LinkedList()
l.insertEle(30)
l.insertEnd(80)
l.insertAtPos(100, 3)
l.display()
```

Output

```
30 -> 80 -> 100 -> NULL
```