Day 14

Detect and remove loop in linked list

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

class linkedlist:
    def __init__(self):
        self.head = None
    def push(self,new_data):
        new_node = Node(new_data)
        new_node.next = self.head
        self.head = new_node
def DetectandRemoveloop(self):
    if self.head is None:
        return
    if self.head.next is None:
        return
```

```
slow = self.head.next.next
fast = fast.next.next
while (fast is not None):
    if fast.next is None
        break
    if slow == fast:
        break
    slow = slow.next
    fast = fast.next.next
if slow == fast:
    slow = self.head
    while (slow.next!=fast.next):
        slow = slow.next
        fast = fast.next
    fast.next = None
def printlist(self):
    temp = self.head
    while (temp):
        print(temp.data)
        temp = temp.next
```

```
L = linked list()
L.head = Node(50)
L.head.next = Node(20)
L.head.next.next = Node(15)
L.head.next.next.next = Node(4)
L.head.next.next.next.next = Node(10)

L.head.next.next.next.next = L.head.next

L.detectandremove loop()

L.print list()
```

output:

50 20 15 4 10

Deleting Node:

* find previous node of the node to be deleted.
* Change the next of previous node
* free memory for the node to be deleted.

Programme :-

```
Class Node :

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

class linkedlist :

    def __init__(self):
        self.head = None

    def push(self, new_data):
        new_node = Node(new_data)
        new_node.next = self.head
        self.head = new_node

    def deletenode(self, key):
        temp = self.head
        if (temp.data ==key):
            self.head = temp.next
            temp = None
            return
```

```
while (temp is not None):
    if temp.data == key
        break
    prev = temp
    temp = temp.next
if (temp == None):
    return
    ~~Prev.next~~
    prev.next = temp.next
    temp = None
def printlist(self):
    temp = self.head
    while (temp):
        print("%d" %(temp.data)),
        temp = temp.next

L = linkedlist()
L.push(7)
L.push(1)
L.push(3)
L.push(2)
L.deletenode(1)
L.printlist()
```

output:

2 3 7

Reverse a linked List:

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

Class linkedlist:
    def __init__(self):
        self.head = None

    def reverseutil(self, curr, prev):
        if curr.next is None:
            self.head = curr
            curr.next = prev
            return
        next = curr.next
        curr.next = prev
        self.reverseutill(next, curr)
```

```
def reverse(self):
    if self.head is None:
        return
    self.reverseutil(self.head, None)

def printlist(self):
    temp = self.head
    while(temp):
        print(temp.data)
        temp = temp.next

l = linkedlist()
l.push(1)
l.push(2)
l.push(3)
l.push(4)
l.reverse()
l.printlist()
```

Output :

4 3 2 1