CSE220

ID: 20301138.

Mating a Node chass:

Class Node:

def__init__(self):

Self. data = 0

self. next = None.

Making a linked list class:

Class My list:

def __init__ (self):

Self. head = None.

Coanting Nodes:

def num (~~head~~ Self):

~~headhead = head.next~~

~~count += 1~~

total = 0

while ~~selhead~~ self.head is not none

head = head.next

~~Count += 1~~

total += 1

return total.

Finding mid ~~[illegible]~~ node:

def ~~def~~ mid (head):

if ~~head is not~~ None:

```
def shawlist (Self ~~head~~):
    if Self.head ~~head~~ is None:
        print ("Empty list")

    new_node = ~~head~~ self.head
    while new_node ~~nd~~ is ~~not None~~ not None
        ~~print (nd.d~~
        print (new_node.data, end=">")


def reverse (Self ~~head~~):

    tmp1 = self.head
    tmp2 = None

    while (tmp1 is not None):
        next = tm1.next
        tmp1.next = tmp2
        tmp2 = tmp1
        tmp1 = next
    Self.head = tmp2          #tail value will
                               become null.
```

```
def mide(surf):
    total = 0
    nh = ~~hea~~self.head
    while nh is not None:
        total += 1
        nh = n.next

    if ~~head~~ self.head is not None:
        nh = sterf. head
        for ~~for~~ i in range (1, count):
            ~~pass~~
            ~~if (i == k):~~
```

List1 = mylist ()

List1. num (1)

List1 . num (2)

List1. (num)(3)

List 1. num (4)

List 1 . num (5)

~~List 1. showlist ()~~

List 1. reverse ()

List 1. mid ()

List 1. show (list) .