In [3]:

```
class stack:
    def __init__(self):
        self.arr=[]
    def stack_push(self,value):
        self.arr.append()
    def stack_pop(self):
        self.arr.pop()
    def printstack(self):
        print(self.arr)
```

In [4]:

```
class queue:
    def __init__(self):
        self.arr=[]
    def enequeue(self,value):
        self.arr.append(value)
    def dequeue(self):
        self.arr.pop(0)
    def printQueue(self):
        print(self.arr)
```

In [4]:

```
#balancing parethesis
#input is a string which contains only one type of parenthesis,check if its balanced or not
# "()())()"
```

In [5]:

```
class stack:
    def __init__(self):
        self.arr=[]
    def add(self,value):
        if value not in self.arr:
            self.arr.append(value)
            return True
        else:
            return False
    def peek(self):
        return self.arr[-1]
array=stack()
array.add("Revathi")
array.add("Meher")
array.add("Teja")
array.peek()
print(array.peek())
```

```
Teja
```

In [3]:

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

class LinkedList:
    def add_ele_at_start(self,head,value):
        new_node=Node(value)
        new_node.next=head
        head=new_node
        return head

    def add_element(self,head,value):
        new_node=Node(value) #step1
        temp=head
        while temp.next!=None: #step2
            temp=temp.next
        temp.next=new_node #step3

    def remove_element(self):

    def print_list(self,head):
        temp=head
        while temp!=None:
            print(temp.data)
            temp=temp.next
        print()

    def search_element(self,value):pass

    def insert(self,head,value,pos):
        new_node=Node(value) #s1
        curr=head
        prev=None
        while pos!=0:
            prev=curr
            curr=curr.next
            pos=pos-1
        prev.next=new_node
        new_node.next=curr

obj=LinkedList()
head=Node(10)
obj.add_element(head,20)
obj.add_element(head,30)
obj.add_element(head,40)
head=obj.add_ele_at_start(head,50)
obj.print_list(head)
obj.insert(head,100,2)
obj.print_list(head)
```

```
  File "C:\Users\MURTHY\AppData\Local\Temp/ipykernel_6372/1435953901.py", line 22
    def print_list(self,head):
    ^
IndentationError: expected an indented block
```

In [ ]: