```
class ArrayStack:
    def __init__(self):
        self.stack = []
        self.character = []
        self.index = -1
    def pushInStack(self, element, index):
        self.stack += [element]
        self.character += [index]
        self.index += 1
    def peekFromStack(self):
        if self.index == -1:
            return "Empty Stack"
        else:
            return self.stack[self.index]

    def peekpos(self):
        if self.index == -1:
            return "Empty Stack"
        else:
            return self.character[self.index]

    def popFromStack(self):
        if self.index == -1:
            return "Empty Stack"
        else:
            temp = self.stack[self.index]
            self.stack = self.stack[:-1]
            self.character = self.character[:-1]
            self.index -= 1
            return temp

class Parentheses:
    def __init__(self, string):
        self.string = string

    def checking(self):
        check = True
        count = 0
        obj = ArrayStack()
        prntss = ["[","]","(",")","{","}"]
        for i in self.string:
            count += 1
            if i in prntss:
                if i == "(" or i == "{" or i == "[":
                    obj.pushInStack(i,count)
                elif (obj.index) == -1:
                    check = False
                    return f"This expression is NOT correct.\n" \
                           f"Error at character # {count}. ‘{i}‘- not opened."
                    break
```

```
                elif i == ")":
                    remove =  obj.peekFromStack()
                    if remove != "(":
                        check = False
                        return f"This expression is NOT correct.\n" \
                               f"Error at character # {obj.peekpos()}. ‘{remove}‘- not closed
                        break
                    else:
                        obj.popFromStack()
                elif i == "]":
                    remove =  obj.peekFromStack()
                    if remove != "[":
                        check = False
                        return f"This expression is NOT correct.\n" \
                               f"Error at character # {obj.peekpos()}. ‘{remove}‘- not closed
                        break
                    else:
                        obj.popFromStack()
                elif i == "}":
                    remove =  obj.peekFromStack()
                    if remove != "{":
                        check = False
                        return f"This expression is NOT correct.\n" \
                               f"Error at character # {obj.peekpos()}. ‘{remove}‘- not closed
                        break
                    else:
                        obj.popFromStack()
        if check == True and obj.index == -1:
            return "This expression is correct."
        else:
            return f"This expression is NOT correct.\n" \
                   f"Error at character # {obj.peekpos()}. ‘{remove}‘- not closed."

# Task 2
class Node:
    def __init__(self,element,pos, next):
        self.val = element
        self.pos = pos # pos in string
        self.next = next

class LinkedListStack:
    def __init__(self):
        self.head = None
        self.address = None
    def pushInStack(self, element, index):
        obj = Node(element, index, None)
        if self.head == None:
            self.head = obj
            self.address = obj
        else:
            obj.next = self.address
            self.address = obj
```

```
            self.head = obj
    def peekFromStack(self):
        if self.head == None:
            return "Empty stack"
        else:
            return self.head.val
    def peekpos(self):
        if self.head == None:
            return "Empty stack"
        else:
            return self.head.pos
    def popFromStack(self):
        if self.head == None:
            return "Empty stack"
        else:
            temp = self.head
            self.address = self.address.next
            self.head = self.head.next
            temp.val = None
            temp.pos = None
            temp = None
    def headcheking(self):
        return self.head

class ParenthesesLinkedList:
    def __init__(self, string):
        self.string = string
    def checking(self):
        check = True
        count = 0
        obj = LinkedListStack()
        prntss = ["[","]","(",")","{","}"]
        for i in self.string:
            count += 1
            if i in prntss:
                if i == "(" or i == "{" or i == "[":
                    obj.pushInStack(i,count)
                elif (obj.headcheking()) == None:
                    check = False
                    return f"This expression is NOT correct.\n" \
                           f"Error at character # {count}. ‘{i}‘- not opened."
                    break
                elif i == ")":
                    remove =  obj.peekFromStack()
                    if remove != "(":
                        check = False
                        return f"This expression is NOT correct.\n" \
                               f"Error at character # {obj.peekpos()}. ‘{remove}‘- not closed
                        break
                    else:
                        obj.popFromStack()
```

```
                elif i == "]":
                    remove =  obj.peekFromStack()
                    if remove != "[":
                        check = False
                        return f"This expression is NOT correct.\n" \
                               f"Error at character # {obj.peekpos()}. ‘{remove}‘- not closed
                        break
                    else:
                        obj.popFromStack()
                elif i == "}":
                    remove =  obj.peekFromStack()
                    if remove != "{":
                        check = False
                        return f"This expression is NOT correct.\n" \
                               f"Error at character # {obj.peekpos()}. ‘{remove}‘- not closed
                        break
                    else:
                        obj.popFromStack()
            if check == True and obj.headcheking() == None:
                return "This expression is correct."
            else:
                return f"This expression is NOT correct.\n" \
                       f"Error at character # {obj.peekpos()}. ‘{remove}‘- not closed."


    print("Task 1---Array based stack\n")
    print("**Example 1 - '1+2*(3/4)'**")
    expession1 = Parentheses("1+2*(3/4)")
    print(expession1.checking())

    print("\n")
    print("**Example 2 - '1+2*[3*3+{4-5(6(7/8/9)+10)-11+(12*8)]+14'**")
    expession2 = Parentheses("1+2*[3*3+{4-5(6(7/8/9)+10)-11+(12*8)]+14")
    print(expession2.checking())

    print("\n")
    print("**Example 3 - '1+2*[3*3+{4-5(6(7/8/9)+10)}-11+(12*8)/{13+13}]+14'**")
    expession3 = Parentheses("1+2*[3*3+{4-5(6(7/8/9)+10)}-11+(12*8)/{13+13}]+14")
    print(expession3.checking())

    print("\n")
    print("**Example 4 - '1+2]*[3*3+{4-5(6(7/8/9)+10)-11+(12*8)]+14'**")
    expession4 = Parentheses("1+2]*[3*3+{4-5(6(7/8/9)+10)-11+(12*8)]+14")
    print(expession4.checking())

    print("\n")
    print("-----------------------------------------------------")
    print("\n")

    print("Task 2---Linked List based stack\n")
    print("**Example 1 - '1+2*(3/4)'**")
    expession5 = ParenthesesLinkedList("1+2*(3/4)")
```

```
print(expession5.checking())

print("\n")
print("**Example 2 - '1+2*[3*3+{4-5(6(7/8/9)+10)-11+(12*8)]+14'**")
expession6 = ParenthesesLinkedList("1+2*[3*3+{4-5(6(7/8/9)+10)-11+(12*8)]+14")
print(expession6.checking())

print("\n")
print("**Example 3 - '1+2*[3*3+{4-5(6(7/8/9)+10)}-11+(12*8)/{13+13}]+14'**")
expession7 = ParenthesesLinkedList("1+2*[3*3+{4-5(6(7/8/9)+10)}-11+(12*8)/{13+13}]+14")
print(expession7.checking())

print("\n")
print("**Example 4 - '1+2]*[3*3+{4-5(6(7/8/9)+10)-11+(12*8)]+14'**")
expession8 = ParenthesesLinkedList("1+2]*[3*3+{4-5(6(7/8/9)+10)-11+(12*8)]+14")
print(expession8.checking())
```

```
     Task 1---Array based stack

     **Example 1 - '1+2*(3/4)'**
     This expression is correct.


     **Example 2 - '1+2*[3*3+{4-5(6(7/8/9)+10)-11+(12*8)]+14'**
     This expression is NOT correct.
     Error at character # 10. ‘{‘- not closed.


     **Example 3 - '1+2*[3*3+{4-5(6(7/8/9)+10)}-11+(12*8)/{13+13}]+14'**
     This expression is correct.


     **Example 4 - '1+2]*[3*3+{4-5(6(7/8/9)+10)-11+(12*8)]+14'**
     This expression is NOT correct.
     Error at character # 4. ‘]‘- not opened.


     ------------------------------------------------------


     Task 2---Linked List based stack

     **Example 1 - '1+2*(3/4)'**
     This expression is correct.


     **Example 2 - '1+2*[3*3+{4-5(6(7/8/9)+10)-11+(12*8)]+14'**
     This expression is NOT correct.
     Error at character # 10. ‘{‘- not closed.


     **Example 3 - '1+2*[3*3+{4-5(6(7/8/9)+10)}-11+(12*8)/{13+13}]+14'**
     This expression is correct.
```

```
**Example 4 - '1+2]*[3*3+{4-5(6(7/8/9)+10)-11+(12*8)]+14'**
This expression is NOT correct.
Error at character # 4. ‘]‘- not opened.
```