In [1]:
```python
# Problem 1
def reverse_list(l):
    if (len(l) == 0):
        return []
    else:
        return [l[-1]] + reverse_list(l[:-1])

# Problem 1 Test Cases:
print(reverse_list([]))
print(reverse_list([1,2,3,4,5,6]))
print(reverse_list([0]))
print(reverse_list([4,2,5,6,7,8,1,3,4,5]))
print(reverse_list([1,2,3,4,5,6,7,8,9,0,1,2,3,4,5]))
```

```
[]
[6, 5, 4, 3, 2, 1]
[0]
[5, 4, 3, 1, 8, 7, 6, 5, 2, 4]
[5, 4, 3, 2, 1, 0, 9, 8, 7, 6, 5, 4, 3, 2, 1]
```

In [1]:
```python
# Problem 2
def srev(l):
    if (len(l) == 0):
        return []
    else :
        a = l[-1]
        if type(a) is list:
            return [srev(a)] + srev(l[:-1])
        else:
            return [l[-1]] + srev(l[:-1])

# Problem 2 Test Cases:
print(srev([1, [5,4], 2, [3,4]]))
print(srev([]))
print(srev([1,2,3,4,5,6,7]))
print(srev([1,[[3,4,6,7],4,3,[5,6,7]],6])) # Should work now
```

```
[[4, 3], 2, [4, 5], 1]
[]
[7, 6, 5, 4, 3, 2, 1]
[6, [[7, 6, 5], 3, 4, [7, 6, 4, 3]], 1]
```

Problem 3

Step 1: Define the Problem

Let f(n) be defined as the following

$$f(n) = n^3 + (n+1)^3 + (n+2)^3$$

Let our domain be defined as

$$D = N$$

Where N is defined as the set of all natural numbers {0,1,2,..}

We define P(n) as

$$P(n) = f(n) mod 9$$

We want P(n) to be true when P(n) = 0

Step 2: Checking Base Case and 2 more

Our base case is 0

$$f(0) = 0^3 + (0+1)^3 + (0+2)^3 = 9 mod 9 = 0$$

So P(0) is true

$$f(1) = 1^3 + (1+1)^3 + (1+2)^3 = 36 mod 9 = 0$$

So P(1) is true

$$f(2) = 2^3 + (2+1)^3 + (2+2)^3 = 99 mod 9 = 0$$

So P(2) is true

Step 3: Inheritance

P(n-1) = f(n-1)

$$f(n-1) = (n-1)^3 + (n)^3 + (n+1)^3$$
$$f(n-1) = (n-1)^3 + (n)^3 + (n+1)^3$$
$$f(n-1) = 3n^3 + 6n$$
$$0 = 3n^3 + 6n$$
$$0 = n(3n^2 + 6)$$
$$n = 0$$

Step 4: Conclusion

We can conclude since we have met all conditions of Proof by Induction that for every n in N, f(n)mod9=0, where f(n) is defined in the above steps

Problem 4

Step 1: Define the Problem

Let f(n) be defined as the following

$$f(n) = f(n-1) + n^2/(2n-1)(2n+1)$$

Let g(n) be defined as the following

$$g(n) = n(n+1)/2(2n+1)$$

Let our domain be defined as

$$D = N$$

Where N is defined as the set of all natural numbers {0,1,2,..}

We define P(n) as true IFF

$$f(n) = g(n)$$

Step 2: Checking Base Case and 2 Other Cases

Our base case is 0

$$f(0) = 0^2/(2(0)-1)(2(0)+1) = 0$$
$$g(0) = 0(0+1)/2(2(0)+1) = 0 = f(0)$$

So P(0) is true

$$f(1) = 1^2/(2(1)-1)(2(1)+1) = 1/3$$
$$g(1) = 1(1+1)/2(2(1)+1) = 2/6 = 1/3 = f(1)$$

So P(1) is true

$$f(2) = f(1) + 2^2/(2(2)-1)(2(2)+1) = 1/3 + 4/15 = 6/10$$
$$g(2) = 2(2+1)/2(2(2)+1) = 6/10 = f(2)$$

So P(2) is true

Step 3: Inheritance f(n-1) = g(n-1)

$$f(n-1) = n^2 - 2n + 1/(2n-3)(2n-1)$$
$$f(n-1) = n^2 - 2n + 1/4n^2 - 8n + 3$$
$$g(n-1) = n^2 - n/4n - 2$$

Step 4: Conclusion

We can conclude that for every n in N, f(n) = g(n) where f(n) and g(n) are defined in the above steps.

In [2]:
```python
# Problem 5a - 5d
```

Problem 6a:

For j in range(len(L)):

while j < 2, you will always get 1.

when j > 2, you get j-1 + j-2, if that sum is greater than j then you continue

Problem 6b:

while j < 2, you will always get 1.

when j > 2, you get j-1 + j-2, if that sum is greater than j then you continue

Problem 7:

-Let L = {a,b | a + b = a if b = 0}

-When a = 0, a + b = a , 0 + 0 = 0. L isn't empty cause (0,0) is apart.

-(a+0)' = (a)' = a'

-a' is in L

-L = N due to Peano's Postulate P4

Conclusion from Lecture Note 07

Problem 8:

-Let L = {a,b|a+b = 0 if a,b = 0}

-When a = 0, b = 0, a+b=0, 0+0=0. L isn't empty cause (0,0) is apart.

-(a+0)' = (a)' = a'

-(0+b)' = (b)' = b'

-a' and b' is in L

-L = N due to Peano's Postulate P4

In [ ]: