PROBLEM 01 ----------

In [119]:
```
# IMPORTING REQUIRED PACKAGES AND CONSTANTS
import numpy as np
from scipy.constants import pi
```

In [120]:
```
# DEFINING AREA FUNCTION (There's a reason for using an array input of parameters, it's in exercise 2)
surface_area = lambda parameters : 4*pi*parameters[0]**2

# DEFINING INITIAL PARAMETERS
r_val, r_uncert = 0.310, 0.001
```

Error propagation method

The uncertainty in the measeurement of the surface area S is determined by:

$$\delta S = \left|\frac{\partial S}{\partial r}\delta r\right| = 8\pi r \delta r$$

In [121]:
```
# DETERMININIG THE VALUE OF THE SURFACE AREA WITH UNCERTAINTY
S = surface_area([r_val])
delta_S = 8*pi*r_val*r_uncert
print('S = ', S, ' m2 , ΔS = ', delta_S, ' m2')
```

```
S =  1.2076282160399165  m2 , ΔS =  0.007791149780902688  m2
```

Finally:

$$\boxed{S = (1000 \pm 8) \times 10^{-3} m^2}$$

Variational method

In [122]:
```
delta_up = abs(surface_area([r_val+r_uncert]) - surface_area([r_val]))
delta_down = abs(surface_area([r_val-r_uncert]) - surface_area([r_val]))
delta_S = 0.5*(delta_up + delta_down)
print('S = ', S, ' m2 , ΔS = ', delta_S, ' m2')
```

```
S =  1.2076282160399165  m2 , ΔS =  0.007791149780902673  m2
```

Finally:

$$\boxed{S = (1000 \pm 8) \times 10^{-3} m^2}$$

As we can see, the values for the uncertainty in the measurement of S varies slightly within each method. However, for our reporting purposes it remains intact.

PROBLEM 02 ----------

In [123]:
```
# DEFINING A FUNCTION THAT RETURNS THE UNCERTAINTY USING VARIATIONAL METHOD (Each function must receive a parameters array)
def uncertainty_var(values, deltas, function):
    values, deltas = np.array(values), np.array(deltas)
    delta_matrix = deltas*np.identity(values.shape[0])
    sum = 0
    for delta in delta_matrix:
        #sum += (0.5*(abs(function(val+delta) - function(val)) + abs(function(val-delta) - function(val))))**2
        sum += (0.5*(abs(function(values + delta) - function(values)) + abs(function(values-delta) - function(values))))**2
    return np.sqrt(sum)
```

In [124]:
```
# TESTING THE FUNCTION ABOVE WITH PROBLEM 1 DATA
print(uncertainty_var([r_val],[r_uncert],surface_area))
print('Works!')
```

```
0.007791149780902673
Works!
```

In [125]:
```
# DEFINING q FUNCTION
q = lambda parameters : parameters[0]*parameters[1]**2 + parameters[0]**2

# DEFINING INITIAL PARAMETERS
x, x_uncert = 8.0, 0.2
y, y_uncert = 4.0, 0.1
```

Error propagation method

The uncertainty in the measeurement of q is determined by:

$$\delta q = \sqrt{\left(\frac{\partial q}{\partial x}\delta x\right)^2 + \left(\frac{\partial q}{\partial y}\delta y\right)^2} = \sqrt{[(y^2 + 2x)\delta x]^2 + (2xy\delta y)^2}$$

In [126]:
```
# DETERMININIG THE VALUE OF q WITH UNCERTAINTY
q_val = q([x,y])
delta_q = np.sqrt(((y**2 + 2*x)*x_uncert)**2 + (2*x*y*y_uncert)**2)
print('q = ', q_val, ' , Δq = ', delta_q)
```

```
q =  192.0  , Δq =  9.05096679918781
```

Finally

$$\boxed{q = 200 \pm 9}$$

Variational method

In [127]:
```
delta_q = uncertainty_var([x,y], [x_uncert,y_uncert], q)
print('q = ', q_val, ' , Δq = ', delta_q)
```

```
q =  192.0  , Δq =  9.050966799187806
```

Finally

$$\boxed{q = 200 \pm 9}$$

As we can see, the values for the uncertainty in the measurement of q varies slightly within each method. However, for our reporting purposes it remains intact.

In [ ]: