# Rectangle_0.4

June 24, 2019

Rectangle - Neutral Axis in Middle - Rev 0.4

**Abstract:**

Given: breadth or width b and depth or height d units of each

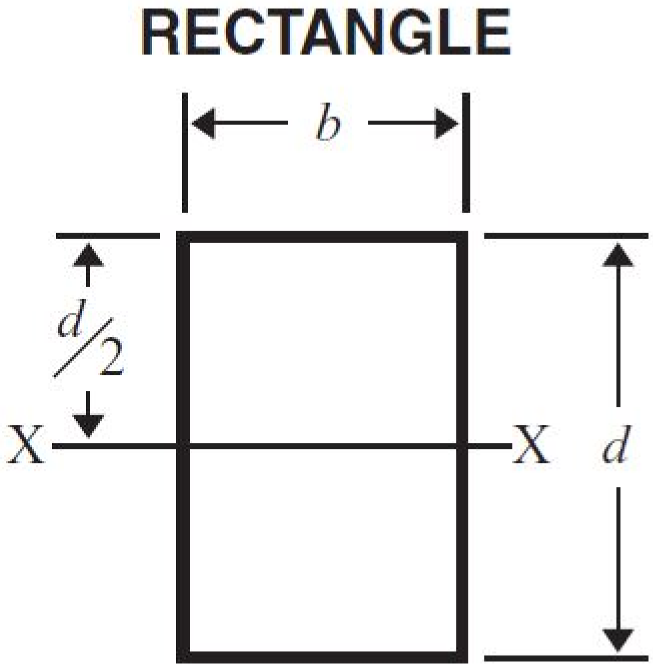
Returns: Area, Section Modulus, Plastic Modulus, Second Moment of Inertia, Radius of Gyration, and distance to centroid

**Instructions:**

Enter breadth, depth and units below the following code cell:

```
In [7]: from IPython.display import Image
        Image(
            filename="./Rectangle_with_Neutral_Axis_in_centre.JPG",
            embed=True
        )
```

Out[7]:



In [8]:

```
# Setup units
from pint import UnitRegistry
unit = UnitRegistry()
quantity = unit.Quantity
unit.default_format = '~' # ~ for unit abreviations, P for pretty print, or both
# Define symbols for common units
m = unit.meter; mm = unit.millimeter; cm = unit.centimeter;
inch = unit.inch; ft = unit.foot;
unit_choices = ['mm', 'cm', 'm', 'inch', 'ft',]
kN = unit.kilonewton; MPa = unit.megapascal; psi = unit.psi

# Setup widgets for interactivity
from ipywidgets import interact, interactive, fixed, interact_manual, HBox, Label
import ipywidgets as widgets
from IPython.display import display

widget_b_label = widgets.Label('Width or breadth, $b$:')
widget_b = widgets.FloatSlider(
    value = 89.0,
    min = 0.0,
    max = 99999,
    step = 0.001,
    disabled = False,
)
widget_b_units = widgets.Dropdown(
    options = unit_choices,
    value = 'mm',
    description = 'Unit',
    disabled = False,
    )

widget_d_label = widgets.Label('Height or depth, $d$:')
widget_d = widgets.FloatSlider(
    value = 89.0,
    min = 0.0,
    max = 99999,
    step = 0.001,
    disabled = False,
)
widget_d_units = widgets.Dropdown(
    options = unit_choices,
    value = 'mm',
    description = 'Unit',
    disabled = False,
    )

widget_output_label = widgets.Label('Desired unit for results:')
widget_output_units = widgets.Dropdown(
    options = unit_choices,
    value = 'mm',
    description = 'Unit',
    disabled = False,
    )

# Display instructions and widgets to user
print('Enter dimensions and units below.')
print('Note: Press ENTER after changing value in floating point text boxes to register
display(
    HBox([widget_b_label, widget_b, widget_b_units, ]),
    HBox([widget_d_label, widget_d, widget_d_units, ]),
    HBox([widget_output_label, widget_output_units]),
    )
```

```
Enter dimensions and units below.
Note: Press ENTER after changing value in floating point text boxes to register changes.
```

```
Out[8]: HBox(children=(Label(value='Width or breadth, $b$:'), FloatSlider(value=89.0, max=99999

Out[8]: HBox(children=(Label(value='Height or depth, $d$:'), FloatSlider(value=89.0, max=99999

Out[8]: HBox(children=(Label(value='Desired unit for results:'), Dropdown(description='Unit', o
```

**Formulas:**

Area, $A = bd$

Elastic Section Modulus, $S_x = bd^2/6$, $S_y = db^2/6$

Plastic Section Modulus, $Z_x = bd^2/4$, $Z_y = db^2/4$

Second Moment of Inertia, $I_x = bd^3/12$, $I_y = db^3/12$

Radius of Gyration, $r_x = d/\sqrt{12}$, $r_y = b/\sqrt{12}$

Distance from bottom left corner to centroid, $c_x = d/2$, $c_y = b/2$

```
In [9]: b = quantity(widget_b.value, widget_b_units.value)
        d = quantity(widget_d.value, widget_d_units.value)

        # Convert b and d to desired output units
        b.ito(widget_output_units.value)
        d.ito(widget_output_units.value)

        # Derive Geometric Properties - See bottom of notebook for pretty formulas
        A = b*d
        Sx = (b*d**2)/6
        Sy = (d*b**2)/6
        Zx = (b*d**2)/4
        Zy = (d*b**2)/4
        Ix = (b*d**3)/12
        Iy = (d*b**3)/12
        rx = d/(12**0.5)
        ry = b/(12**0.5)
        cx = d/2
        cy = b/2

        # Define Output
        print('Given:')
        print(' Width or breadth, b = {0:n} {1} and'.format(b.magnitude, b.units))
        print(' Height or depth, d = {0:n} {1}'.format(d.magnitude, d.units))
        print('')
        print('Geometric Properties:')
        print(' Area, A = {0:n} {1}'.format(A.magnitude, A.units))
        print(' Major Elastic Section Modulus, Sx = {0:n} {1}'.format(Sx.magnitude,
            Sx.units))
        print(' Minor Elastic Section Modulus, Sy = {0:n} {1}'.format(Sy.magnitude,
            Sy.units))
        print(' Major Plastic Section Modulus, Zx = {0:n} {1}'.format(Zx.magnitude,
            Zx.units))
        print(' Minor Plastic Section Modulus, Zy = {0:n} {1}'.format(Zy.magnitude,
            Zy.units))
        print(' Major Second Moment of Inertia, Ix = {0:n} {1}'.format(Ix.magnitude,
            Ix.units))
        print(' Minor Second Moment of Inertia, Iy = {0:n} {1}'.format(Iy.magnitude,
            Iy.units))
        print(' Major Radius of Gyration, rx = {0:n} {1}'.format(rx.magnitude,
            rx.units))
        print(' Minor Radius of Gyration, ry = {0:n} {1}'.format(ry.magnitude,
            ry.units))
        print(' Distance from Major Axis to Extreme Fibre, cx = {0:n} {1}'.format(
            cx.magnitude, cx.units))
        print(' Distance from Minor Axis to Extreme Fibre, cy = {0:n} {1}'.format(
            cy.magnitude, cy.units))
```

```
Given:
 Width or breadth, b = 38.1 mm and
 Height or depth, d = 139.7 mm

Geometric Properties:
 Area, A = 5322.57 mm ** 2
 Major Elastic Section Modulus, Sx = 123927 mm ** 3
 Minor Elastic Section Modulus, Sy = 33798.3 mm ** 3
 Major Plastic Section Modulus, Zx = 185891 mm ** 3
 Minor Plastic Section Modulus, Zy = 50697.5 mm ** 3
 Major Second Moment of Inertia, Ix = 8.65631e+06 mm ** 4
 Minor Second Moment of Inertia, Iy = 643858 mm ** 4
 Major Radius of Gyration, rx = 40.3279 mm
 Minor Radius of Gyration, ry = 10.9985 mm
 Distance from Major Axis to Extreme Fibre, cx = 69.85 mm
 Distance from Minor Axis to Extreme Fibre, cy = 19.05 mm
```

**Revision History:**
- Rev 0.4 23-Jun-2019 E.Durham Added widgets for input of dimensions and units and graphic - Rev 0.3 19-Jun-2019 E.Durham Added units
- Rev 0.2 18-Jun-2019 E.Durham Added Plastic Section Modulus and revised formatting
- Rev 0.0 28-Mar-2019 E.Durham Created notebook using Figure 11.4 from Wood Design Manual 2017

**ToDo / Issues:** - Add interaction to formulas so that values are immediately updated as user updates dimensions and units so that the user does not need to re-run each of the cells to get results - Add demonstration of symPy package - Floating point text boxes do not show more than 2 digits - Add autoscaling or right-sizing to rounding function and then results