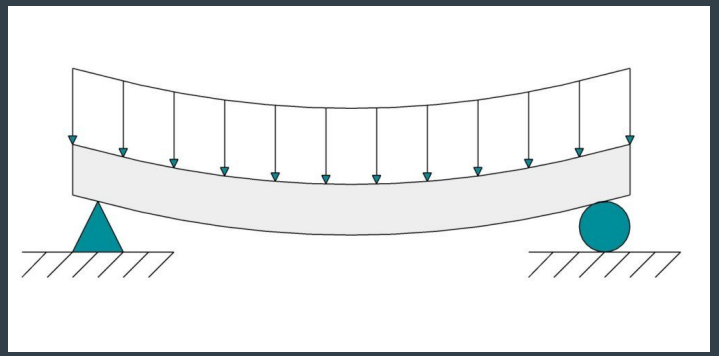
Beam Deflection App

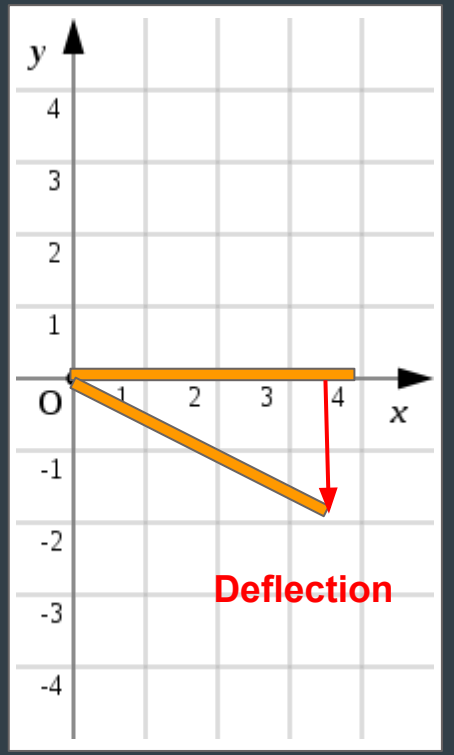
Clueless

Pranshu Adhikari, Joe Dreger, Justin Lam, Rachel Paddock

Beam Deflection

- The displacement in the y direction of any point on the axis of the beam
- Important for assessing structural integrity and material failure





Two Cases

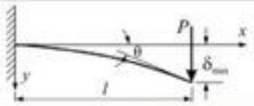
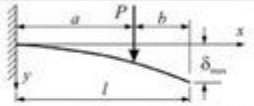
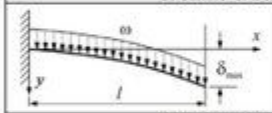
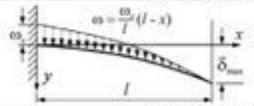
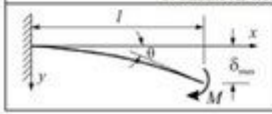
Fixed Support

Pin-Roller Support

Equation for Beam Deflection

Name (Symbol)	Units
Force (P)	N
Dimensions (L,A,B,H,R)	m
Elastic Modulus (E)	Pa
Distributive Force (q)	N/m
Moment (Mo)	Nm

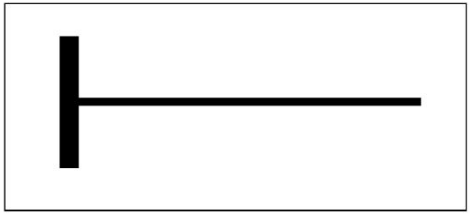
BEAM DEFLECTION FORMULAE

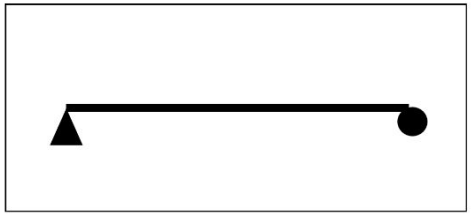
BEAM TYPE	SLOPE AT FREE END	DEFLECTION AT ANY SECTION IN TERMS OF x	MAXIMUM DEFLECTION
1. Cantilever Beam – Concentrated load P at the free end			
	$\theta = \frac{Pl^2}{2EI}$	$y = \frac{Px^2}{6EI}(3l - x)$	$\delta_{max} = \frac{Pl^3}{3EI}$
2. Cantilever Beam – Concentrated load P at any point			
	$\theta = \frac{Pa^2}{2EI}$	$y = \frac{Px^2}{6EI}(3a - x)$ for $0 < x < a$ $y = \frac{Pa^2}{6EI}(3x - a)$ for $a < x < l$	$\delta_{max} = \frac{Pa^2}{6EI}(3l - a)$
3. Cantilever Beam – Uniformly distributed load ω (N/m)			
	$\theta = \frac{\omega l^3}{6EI}$	$y = \frac{\omega x^2}{24EI}(x^2 + 6l^2 - 4lx)$	$\delta_{max} = \frac{\omega l^4}{8EI}$
4. Cantilever Beam – Uniformly varying load: Maximum intensity ω_o (N/m)			
	$\theta = \frac{\omega_o l^3}{24EI}$	$y = \frac{\omega_o x^2}{120lEI}(10l^3 - 10l^2 x + 5lx^2 - x^3)$	$\delta_{max} = \frac{\omega_o l^4}{30EI}$
5. Cantilever Beam – Couple moment M at the free end			
	$\theta = \frac{Ml}{EI}$	$y = \frac{Mx^2}{2EI}$	$\delta_{max} = \frac{Ml^2}{2EI}$

https://www.yumpu.com/en/document/view/12369231/beam-deflection-formulas

How our App Works

Title Slide





Choose Variables

Add diagram for all variables

Q, P, M (button)

Magnitude = (type in)

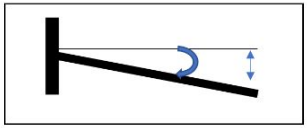
A = (type)

R, B, H, L (type)

Material Type (E) = (button)

Concrete, wood, steel, aluminum, plastic

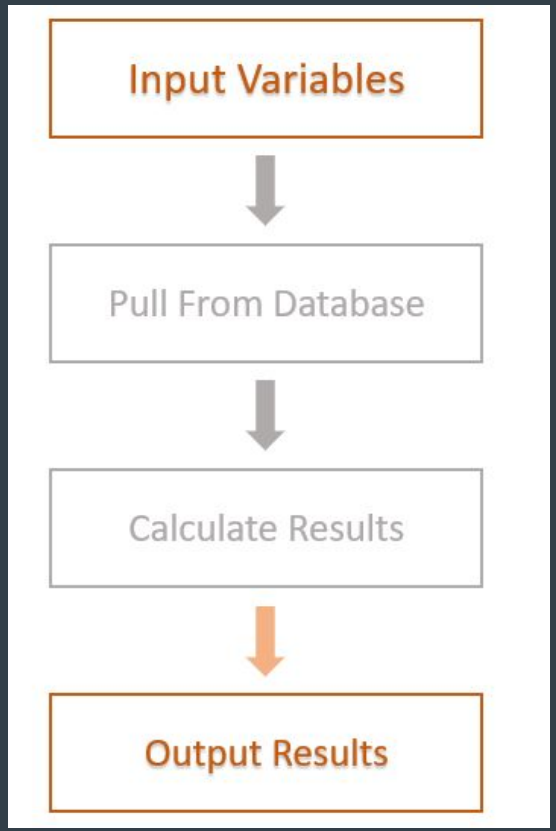
Output Slide



Deflection =

Angle A =

Workflow



kivy

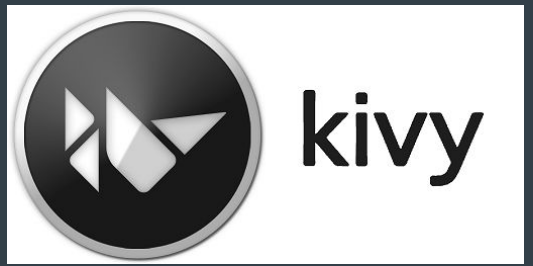
BeautifulSoup4

NumPy pandas matplotlib

kivy

Packages

- Numpy
  - Beam deflection equations
- Beautiful soup
  - Pull constants from websites
- Pandas
  - Store values in a DataFrame
- Kivy
  - GUI



Code: Website Pull and Data Frame

BeautifulSoup Requests gets data from website

```
import requests
import pandas as pd
#Get Data from the website and store in Data Frame
url = 'https://www.engineeringtoolbox.com/young-modulus-d_417.html'
html = requests.get(url).content
df_list = pd.read_html(html)
df = df_list[-1]
df.to_csv('MaterialProperties.csv')
df.columns = ['Material', 'YM', 'UTS', 'YS']
```

Pandas stores data in an easy to access format

```
In [3]: df
Out[3]:
                                              Material          YM  UTS   YS
0                                         ABS plastics  1.4 - 3.1   40  NaN
1    A53 Seamless and Welded Standard Steel Pipe - ...         NaN  331  207
2    A53 Seamless and Welded Standard Steel Pipe - ...         NaN  414  241
3            A106 Seamless Carbon Steel Pipe - Grade A         NaN  330  205
4            A106 Seamless Carbon Steel Pipe - Grade B         NaN  415  240
..                                                 ...         ...  ...  ...
122                                          Uranium         170  NaN  NaN
123                                         Vanadium         131  NaN  NaN
124                                     Wrought Iron  190 - 210  NaN  NaN
```

Code: Find Specific E Value

```
def getE(Material):
    #Search for the getE() input material
    i = Material
    n = 0
    for n in range(len(data)):
        #when material is found store the Young's Modulus
        if i == data[n]['Material']:
            Epull = data[n]['YM']
        else:
            n = n + 1

    #Check if no E value is given in table
    if Epull == 'NaN':
        print("There is no Young's Modulus Available")
    #Check if a range is given and return the lowest E value
    else:
        E = (float(Epull.split('-')[0])*(10**6)) #Convert GPa to Pa
        print("The Young's Modulus is ",E," Pa")
```

```
In [5]: getE('Wrought Iron')
The Young's Modulus is  190000000.0  Pa
```

Calculator: Deflection Equations

Function that takes dimensions, type of support, value of load, and the elastic modulus

- 0 if value does not exist

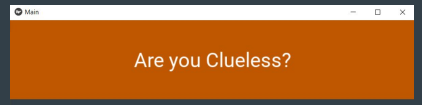
```
def BD(R,B,H,L,A,support,q,P,Mo,E):
    #first calculating the area and moment of inertia
    if R == 0:
        #Shape is a rectangle haha
        Area = H*B
        MoI = (1/12)*B*H**3
        print('The area is:',Area)
        print('MoI is:', MoI)
    elif B == 0:
        #shape is a circle
        Area = np.pi*R**2
        Dia = 2*R
        MoI = (1/64)*np.pi*Dia**4
        print('The area is:',Area)
        print('MoI is:', MoI)
    else:
        print('Error, no shape given')
```

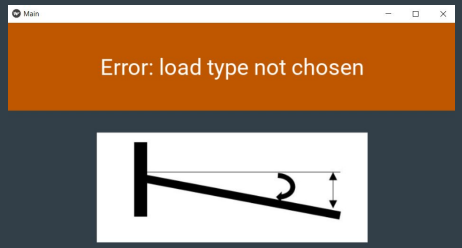
```
elif support == 'tri':
    if q != 0:
        angle_left = ((2*L-A)**2)*(q*(A**2))/(24*L*E*MoI)
        angle_right = -1*(2*(L**2)-(A**2))*(q*(A**2))/(24*L*E*MoI)
        #deflection is in the middle
        deflection = (4*(L**2)-(7*A*L)+(3*(A**2)))*(q*(A**3))/(24*L*E*MoI)
        print('Deflection:',deflection)
        print('Angle (left):',angle_left)
        print('Angle (right):',angle_right)

    if P != 0:
        C = L - A #this is the other length (called this C instead of B since I
        angle_left = (L+C)*(P*A*C)/(6*L*E*MoI)
        angle_right = (L+A)*(P*A*C)/(6*L*E*MoI)
        print('Angle (left):',angle_left)
        print('Angle (right):',angle_right)
        if A >= C:
            deflection = (3*(L**2)-4*(C**2))*(P*C)/(48*E*MoI)
            print('Deflection:',deflection)
        else: #if A<C
            deflection = (3*(L**2)-4*(A**2))*(P*A)/(48*E*MoI)
            print('Deflection:',deflection)

    if Mo != 0:
        C = L - A
        angle_left = (6*A*L-(3*(A**2))-2*(L**2))*Mo/(6*L*E*MoI)
        angle_right = (3*(A**2)-(L**2))*Mo/(6*L*E*MoI)
        print('Angle (left):',angle_left)
        print('Angle (right):',angle_right)
        #I got this formula off of pinterest so...
        if A<=(L/2):
            deflection = -1*(np.sqrt(3))*Mo*(((L**2)-(A**2))**1.5)/(27*E*L*MoI)
            print('Deflection:',deflection)
        else:
            deflection = (np.sqrt(3))*Mo*(((L**2)-(C**2))**1.5)/(27*E*L*MoI)
            print('Deflection:',deflection)
```

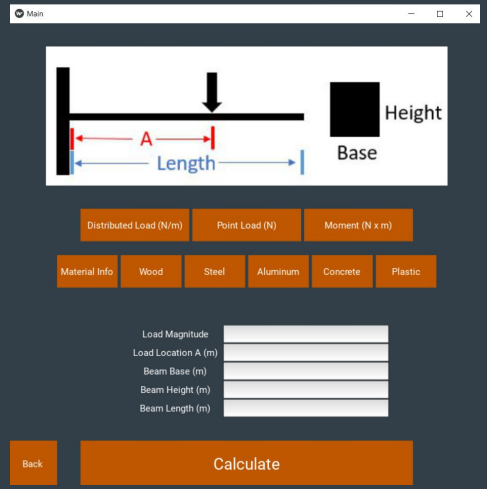
Setup Front-End

```
#Create App Screens
class Start_Window(Screen):
class Variables_Window(Screen):
class Output_Window(Screen):

class Window_Manager(ScreenManager):
```

```
WindowManager:
    Connection_Window:
    Variables_Window:
    Output_Window:
```



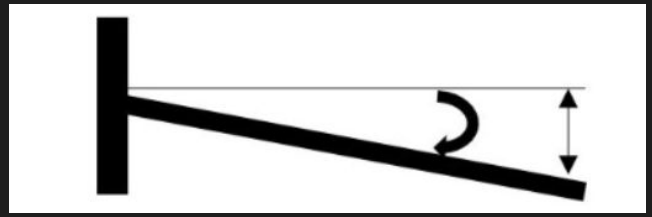




```
<Output_Window>:
    FloatLayout:
        Button:

        TextInput:

        Label:

        Image:
```

Calculate

Load Magnitude	
Load Location A (m)	
Beam Base (m)	
Beam Height (m)	
Beam Length (m)	



Import Pages into App

```
#Compile App
kv = Builder ("kv file name")

class Main_App(App):
    def build(self):
        return kv
```

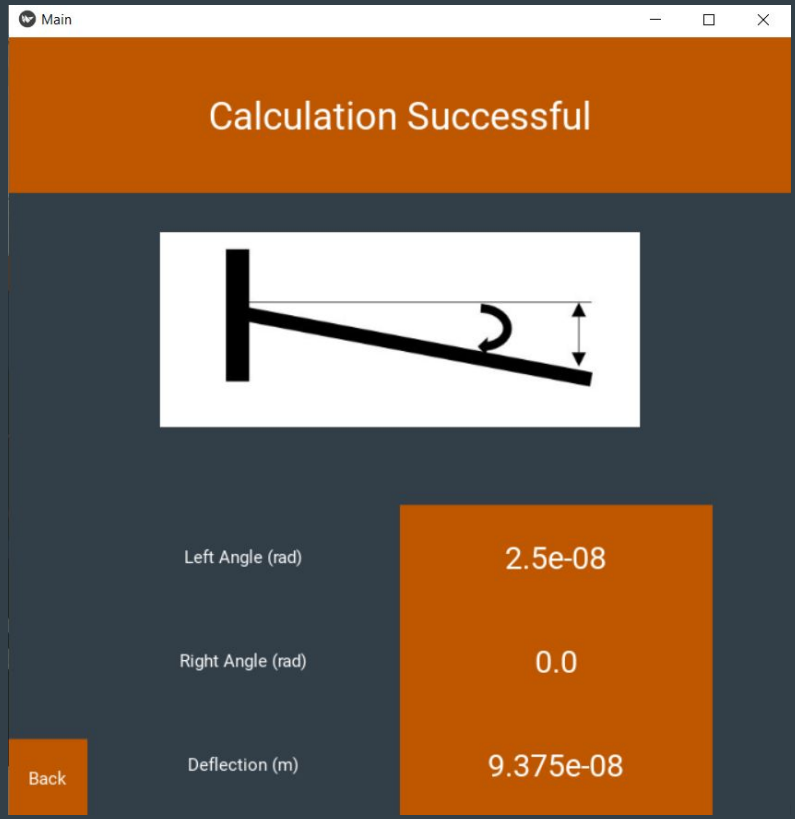
Calculator - GUI

```
"Use global variables"
variable_list = []
output_list = []
```

```
class VariablesWindow(Screen):

    def collect_variables():

        variable_list [] = "variable value"

        #Error Handling
        if variable == "empty":
            error_message = "ERROR"
            return

    def database_pull():

    def calculator():

        "insert calculations"
        output_list [] = "load/angle value"
```

```
"Use global variables"
variable_list = []
output_list = []
```

```
class OutputWindow (Screen):

    def output_pass(self):
        print("angle_left =", round(output_list [0],4))
        print("angle_right =", round(output_list [1],4))
        print("deflection =", round(output_list [2],4))
```



Useful Links

- Young's Modulus, Tensile Strength and Yield Strength Values for some Materials (engineeringtoolbox.com)
- https://numpy.org/
- https://pandas.pydata.org/
- https://pypi.org/project/beautifulsoup4/

https://github.com/jus3003/Python-Project

Questions?

```
<MyPopup>:

    TabbedPanel: (Material tabs)

    Carousel: (Material information sliders)
```

Github

https://github.com/jus3003/Python-Project