6935621

#https://github.com/derrickyebs

import numpy as np L = 12

#length of beam in meters w =

10 #intensity of load in KN/m

#A

#Bending moment(M) and shear force(V) at the first end, x=0 x

= 0

M = (w*(-6*x**2 + 6*L*x-L**2))/12 V =

w*(L/2 - x) m= 'The bending moment at

x=0 is ' n= 'the shear force at x=0 is ' print()

print('(a.1)' + m + str(M) + ' and ', n + str(V))

#Bending moment(M) and shear force(V) at the first end, x=L=10

x = L

M = (w*(-6*x**2 + 6*L*x-L**2))/12

V = w*(L/2 - x) a= 'The bending

moment at x=L is ' b= 'the shear

force at x=L is '

print() print('(a.2)' + m + str(M) +' and ', n +

str(V))

#B

When the bending moment is zero, we get an expression x**2 - Lx + L**2/6 = 0

#from the expression

a = 1

b = -L

c = L**2/6

#Using the Almighty formula the two roots are;

discriminant = b**2 - 4*a*c

root_1b = (-b + np.sqrt(discriminant))/2*a

root_2b = (-b - np.sqrt(discriminant))/2*a

print()

print('(b) The points of contra-flexure are {0} and {1}'.format(root_1b,root_2b))

#C

When the shear force is zero, x = L/2

""" x = L/2

print()

print('(c) The point at which V=0 is {}'.format(x))

#D

p = 0

s = 0.01

```
q = L + s

x = np.arange(p,q,s)

M = (w*(-6*x**2 + 6*L*x-L**2))/12

print()

print('(d) Using the initialized variable,the bending moment at each step in the array is {0}'.format(M))
```

#E

```
V = w*(L/2 - x)

print()

print('(e) The shear force for each step along the span is {}'.format(V))
```

#F

Let the absolute value of the bending moment array be AM

Also let the minimum AM be m_AM

```
AM =  abs(M)

m_AM = min(AM)
```

When the bending moment is m_AM, we get an expression x**2 - Lx + (L**2/6)+(2*m_AM)/w = 0

```
#from the above expression

a = 1

b = -L
```

```
c = (L**2/6)+(2*m_AM)/w

#Using the Almighty formula the two roots are;

discriminant = b**2 - 4*a*c

root_1f = (-b + np.sqrt(discriminant))/2*a

root_2f = (-b - np.sqrt(discriminant))/2*a

print()

print('(f) The points along L at which the absolute values of the bending moment array is minimum are {0} and {1}'.format(root_1f,root_2f))


#G

"""

Let the relative errors be r_e

"""

r_e1 = ((root_1b - root_1f)/root_1b*100)

r_e2 = ((root_2f - root_2b)/root_2f*100)

print()

print('(g) The relative errors between estimated points of contra-flexure are {0}% and {1}%'.format(r_e1,r_e2))


#H

"""

Let the maximum bending moment be max_M and the minimum bending moment be min_M

"""

#for the maximum

max_M = max(M)

"""
```

```
When the bending moment is max_M, we get an expression x**2 - Lx + (L**2/6)+(2*max_M)/w = 0

"""

a= 1

b = -L

c = (L**2/6)+(2*max_M)/w

#Using the Almighty formula the two roots are;

discriminant = b**2 - 4*a*c

root_1 = (-b + np.sqrt(discriminant))/2*a

root_2 = (-b - np.sqrt(discriminant))/2*a print()

print('(h.1) The points at which the maximum bending moment occur are {0} and
{1}'.format(root_1,root_2))


#for the minimum

min_M = min(M)

"""

When the bending moment is min_M, we get an expression x**2 - Lx + (L**2//6)+(2*min_M)/w = 0

"""

 a = 1

b = -L

c = (L**2//6)+(2*min_M)/w

#Using the Almighty formula the two roots are;

discriminant = b**2 - 4*a*c

root_1 = (-b - np.sqrt(discriminant))/2*a

root_2 = (-b + np.sqrt(discriminant))/2*a print()

print('(h.2) The points at which the minimum bending moment occur are {0} and
```

```
{1}'.format(root_1,root_2))
```