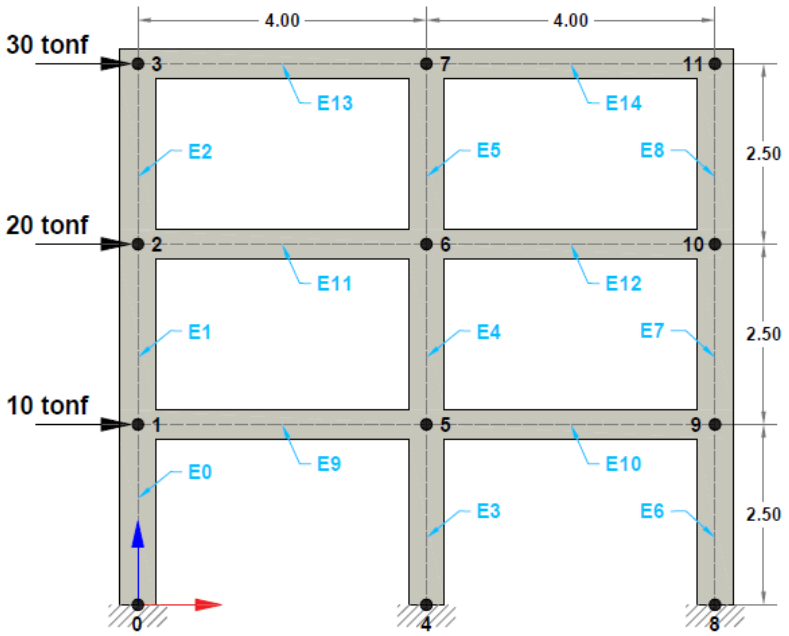
Linear 2D frame under horizontal static loads (NLSTA).

This problem describes the static analysis of a linear two-dimensional frame under point loads. The analysis is performed in a single step.

Input and output files for this problem are available in the examples folder of this REPO (notebooks\Examples).



- Element type for columns and beams: 2
- Columns cross section *0.50 m x 0.50 m*
- Beams cross section *0.40 m x 0.40 m*
- Material profile for all elements is concrete with elastic modulus of *2000000 tonf/m²* and specific weight of *2.4 tonf/m³*

Internal forces, together with a simple vertification of static global equilibrium are available in the file:

notebooks\Examples\Ex_01\Output.xls

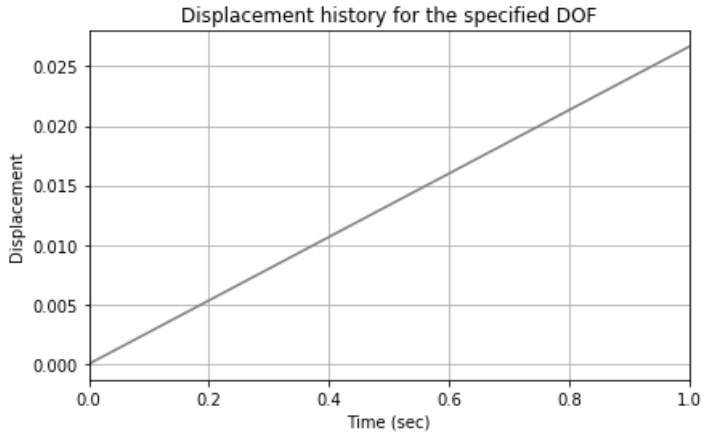
In [1]:
```
%matplotlib inline
import matplotlib.pyplot as plt
import numpy as np
import sympy as sym
from os import sys
sys.path.append("../source/")
from STRUCTURE import Struct_DYN
from postprocesor import *

# Execute analysis
displacement,folder,IBC,nodes,elements,ninc,T,MvarsGen,ILFGen = Struct_DYN("Examples/E
01/01_INPUT/")
```

```
----------------------
Number of nodes: 12
Number of elements: 15
Number of equations: 27
Number of equations after constraints: 27
----------------------
Natural periods of the system :  Not computed,static system solution
----------------------
Time step for solution: 0.002 sec
Number of time increments: 500
----------------------
Duration for system solution: 0:00:00.931524
Duration for the system's solution: 0:00:00.932524
Duration for post processing: 0:00:00
----------------------
Analysis terminated successfully!
----------------------
```

Results

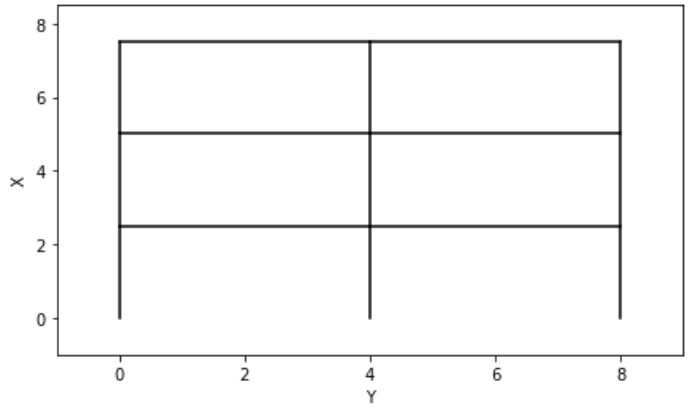
The displacement response along the horizontal direction for node 3 is shown below.

In [2]:
```
fig   = NodalDispPLT(displacement[6,:], T, ninc, ylabel = "Displacement")
```



The code can also display the structure under study for verification purposes.

In [3]:
```
model = GrafModel(elements, nodes)
```



In [4]:
```
from IPython.core.display import HTML
def css_styling():
    styles = open('./nb_style.css', 'r').read()
    return HTML(styles)
css_styling()
```

Out[4]:

In [ ]:
```

```