Exercício 4 - EC tipo placa

Exercício 4 - EC tipo placa

Importando as bibliotecas

```
from sympy import *
import numpy as np
from scipy.optimize import fsolve
import matplotlib.pyplot as plt
init_printing(pretty_print=true)
```

Inserindo os dados do problema

```
Ly = 600/1000
df = 0.76/2000
dcl = 0.38/1000
dch = 2.89/1000
Ny = 5
dy = Ly/Ny
Lz = 63/1000
kf = 120
kcl = 180
h = 8000
M = 0.4
Cp = 4186
Q = 11100/Ny
```

T0 = Tcha = 30

Definição das distâncias

Lfcl = df/2 + dcl/2

Lfi = df/2

Lcli = dcl/2

Lchi = dch/2

Definição das áreas

Ax = dy*Lz

Ayf = df*Lz

Aycl = dcl*Lz

Aych = dch*Lz

Definição das condutâncias

Gfi = kf*Ax/Lfi

Gcli = kcl*Ax/Lcli

Gfcl = (Gfi*Gcli)/(Gfi+Gcli)

Gcls = kcl*Ax/Lfi

Gchs = h*Ax

Gch = M*Cp

Gf = kf*Ayf/dy

Gcl = kcl*Aycl/dy

Cálculo das equações de energia e exibição dos resultados

```
Tcam = []
linha = []
Temp = []
Tlabel = [[],4]
Xlabel = np.array(['T f','T cl','T s','T ch'])

def Tcam(z):
 Tf = z[0]
 Tcl = z[1]
 Ts = z[2]
 Tch = z[3]
 Tcha = T0

 F = np.empty((4))
 F[0] = Q + Gfcl*(Tcl-Tf)
 F[1] = Gfcl*(Tf-Tcl) + Gcls*(Ts-Tcl)
 F[2] = Gcls*(Tcl-Ts) + Gchs*(Tch-Ts)
 F[3] = Gchs*(Ts-Tch) + 2*Gch*(T0-Tch)

 return F

z0 = np.array([1,1,1,1])
z = fsolve(Tcam,z0)
linha.append(z)
print('cam 1  --->','Tf =', round (z[0],2),'[oC]', ' Tcl=', round(z[1],2),'[oC]','
Ts=', round(z[2],2),'[oC]',' Tch=', round(z[3],2),'[oC]')
plt.figure()
plt.subplot(211)
plt.plot(Xlabel,z,'bo',Xlabel,z,'r--')
plt.show()
```

```
 print('------------------------------------------------------------------')

j = 0
for j in range(1,Ny):

  j+=1
  Tfa = z[0]
  Tcla = z[1]
  Tsa = z[2]
  Tcha = z[3]


  def Tcam(z):

   Tf = z[0]
   Tcl = z[1]
   Ts = z[2]
   Tch = z[3]

   F = np.empty((4))
   F[0] = Q + Gfcl*(Tcl-Tf) + Gf*(Tfa-Tf)
   F[1] = Gfcl*(Tf-Tcl) + Gcl*(Tcla-Tcl) + Gcls*(Ts-Tcl)
   F[2] = Gcls*(Tcl-Ts) + Gchs*(Tch-Ts)
   F[3] = Gchs*(Ts-Tch) + Gch*(Tcha-Tch)

   return F

  z0 = np.array([1,1,1,1])
  z = fsolve(Tcam,z0)
```

```
  print('cam', j,' --->','Tf =', round (z[0],2),'[oC]', ' Tcl =', round(z[1],2),'[oC]','
Ts =', round(z[2],2),'[oC]',' Tch =', round(z[3],2),'[oC]')


  linha.append(z)
  Temp.append(linha)
  plt.subplot(211)
  plt.plot(Xlabel,z,'bo',Xlabel,z,'r--')
  plt.show()
  print('----------------------------------------------------------------------')


  Xlabel = np.array(['T f','T cl','T s','T ch'])


 j+=1
```