Contents

```python
import numpy as np
import matplotlib
import matplotlib.pyplot as plt
import json  # optionnel : pas necessaire pour un premier test
import csv   # optionnel : pour améliorer la lecture, plus tard

#Constants
M = 29.0e-3
R = 8.31
P0 = 1.0e5
g0 = 9.8
RT = 6.4e3
pi = np.pi

#Data
zexp = np.array([0.0,
                 5.0,
                 10.0,
                 12.0,
                 20.0,
                 25.0,
                 30.0,
                 35.0,
                 40.0,
                 45.0,
                 48.0,
                 52.0,
                 55.0,
                 60.0,
                 65.0,
                 70.0,
                 75.0,
                 80.0,
                 84.0,
                 92.0,
                 95.0,
                 100.0])

Texp = np.array([15.0,
                 -18.0,
                 -49.0,
                 -56.0,
                 -56.0,
                 -51.0,
                 -46.0,
                 -37.0,
                 -22.0,
                 -8.0,
                 -2.0,
                 -2.0,
                 -7.0,
                 -17.0,
                 -33.0,
                 -54.0,
                 -65.0,
                 -79.0,
                 -86.0,
                 -86.0,
                 -81.0,
                 -72.0])


# Travail fait en classe lundi

def T(z,unite):
```

```
    z_km = z / 1000 #conversion
    alpha = 1 # facteur de conversion

    if unite == 'C':
        alpha = 0

    i = 0
    while z_km > zexp[i+1]: # recherche de l'indice i
        i = i + 1

    rate =  ( Texp[i+1] - Texp[i] ) / ( zexp[i+1] - zexp[i] )
    temperature = alpha*273 + Texp[i] + rate * (z_km - zexp[i])
    return temperature


# Suite


#Gravity
def g(z):
    return g0 * RT**2 / (RT + z)**2

#Temperature
N = 1000
zmax = 100.0e3  #c'est ici que je définis l'altitude max #deplace
dz = zmax / (N-1)
print(N, zmax, dz)
zatm = np.array([ k * dz for k in range(N) ])
Tatm = np.array([ T(zatm[k], 'C') for k in range(N) ])
TatmK = np.array([ T(zatm[k], 'K') for k in range(N) ])
gatm = np.array([ g(zatm[k]) for k in range(N)])

fig, ax = plt.subplots()
ax.plot( TatmK,zatm)
ax.plot( Tatm,zatm)
plt.savefig("mon profil")



#Pressure

#Calcul du champ de pression par une méthode d'Euler (basique)
Patm = [P0]
gatm = [g0]
deltap = 0
gradient = 0
for k in range(N-1):
    gradient = - M * g( zatm[k] ) / (R * TatmK[k] )
    deltap = gradient * dz
    Patm.append( Patm[k] + deltap )
    gatm.append( gatm[k] )
Patm = np.array(Patm)
print(M,R,P0,g0,RT)


#Mass

#calcul de la masse d'air par la méthode des rectangles
#situé entre deux sphères d'altitude z et z+dz

def masse_atm(z):
    masse = 0
    k = 0

    Cte = 4*pi*M/R
    while zatm[k] < z:
        dm = Cte * dz * (RT + z)**2 * Patm[k] / T(zatm[k],'K')
        masse = masse + dm
```

```
134             k = k+1
135         return masse
136 
137 mtot = masse_atm(100e3)
138 print(mtot)
139 
140 Patm
```

1000 100000.0 100.10010010010011 0.029 8.31 100000.0 9.8 6400.0 2.2013954250074244e+16