Please always load python's numerical (numpy) and graphics (matplotlib) libraries
import numpy as np
import matplotlib.pyplot as plt

(1) Simple calculations in Python

Type the follow commands and see what happens. Press Shift+Return after every line.

Please note this section contains two errors.

1+1

#what does Python to with this funny line?
3 -- 3

#there are two different way to divide two integers
1001/10

1001//10

cos(0)

np.cos(0)
np.cos(90)

np.cos(np.radians(90))

4^4

4**4

1.0e6

log(10.0)

np.log(10.0)

np.log10(10.0)

np.pi

(2) Now define your own variables (This section contains four errors ;=)
(3) Compute the density, $\rho$, of the Earth by defining two variables for the radius $R=6371$ km and the mass $M=5.9736 \times 10^{24}$ kg. Use the formulas $\rho=M/V$ and volume $V=\frac{4\pi}{3}R^3$. Compare your answer to the ambient densities of water ($1$ g/cc), rock ($2.7$ g/cc), and iron ($7.8$ g/cc).

(4) Vectors in Python. Two options: Lists and Numpy arrays

```python
# First let us use a 'list'
list = [ 1, 2, 3, 3, 5, 5 ]
print(list)
print(list[1])
print(len(list))
print(type(list))

# Second use a one-dimensional numpy array. This will be used most often throughout this course.
a = np.array([ 1, 2, 3, 3, 5, 5 ])
print(a)
print(len(a))
print(a.shape)
print(a.shape[0])
print(type(a))

b = np.zeros(15)
print(b)
```
# Let us perform some operations on entire vectors - see what happens in every case

```python
import numpy as np

c = np.ones(6)
print(c)
print(a+c)
print(a*(c+1))
print(np.sqrt(a))
```

(5) Let us generate some XY plots

```python
# Compute the x and y coordinates for points on a sine curve
x = np.arange(0, 10.0, 1.0)
y = np.sin(x)
print(x)
print(y)

# Plot the points using matplotlib
plt.plot(x, y)
plt.show()  # You must call plt.show() to make graphics appear.

# For a test, please make this curve look very smooth. You only need to change one number. Then change this back.

y2 = np.cos(x)

# Plot the points using matplotlib
plt.plot(x, y, label='sin(x)')
plt.plot(x, y2,label='cos(x)')
plt.legend()
plt.show()  # You must call plt.show() to make graphics appear.

# let us make this plot pretty (please use only 10 x values)
plt.plot(x,y*0,'k--',linewidth=1,dashes=(10,3));
plt.plot(x,y,'rD-',linewidth=2,markersize=8,mec='r',mew=2, mfc='pink',label="sin(x)")
plt.plot(x,y2,'bo--',linewidth=4,markersize=10,mec='b',mew=2, mfc='white',dashes=(5,1 ,1,1),label="cos(x)")
plt.fill_between(x,y,y*0,color=(1.0,0.94,0.94),lw=0)
plt.xlabel('My instructor made me add this axis.')
plt.ylabel('My dog ate my online homework.')
plt.legend()
plt.show()
(6) Let us generate a 3D plot

```python
n=51
L = 10.0
x = np.arange(-L, L, L/n)
y = np.arange(-L, L, L/n)
X, Y = np.meshgrid(x, y)

R = np.sqrt(X**2+Y**2) + 1e-14
Z = np.sin(R) / R

#this requires some extra libraries
from mpl_toolkits.mplot3d import Axes3D
from matplotlib import cm
from matplotlib.ticker import LinearLocator, FormatStrFormatter

fig = plt.figure(figsize=(10,10))
ax = fig.gca(projection='3d')
surf = ax.plot_surface(X, Y, Z, cmap=cm.coolwarm,linewidth=0, antialiased=False)
plt.show()
```

(7) Let load data file that contains the Earth temperature record in columns 3 (time in years) and 5 (temperature in Kelvin)

```python
data = np.loadtxt('ice_core_temperature_data.txt', usecols=(2,4)) #why 2 and 4, not 3 and 5?
print(data)

plt.plot(data[:,0],data[:,1],'r-',linewidth=1,markersize=0,mec='r',mew=2, mfc='pink', label="sin(x)")
plt.show()
```
```python
# Hey, this plot is too small. I cannot see anything!
plt.rcParams['figure.figsize'] = [15, 5]
plt.plot(data[:,0], data[:,1], 'r-', linewidth=1, markersize=0, mec='r', mew=2, mfc='pink',
label="sin(x)"");
plt.show()

temp = data[:,1]
print(sum(temp))
print(temp.shape)
average = sum(temp) / temp.shape[0]
print(average)

plt.rcParams['figure.figsize'] = [15, 5]
plt.plot(data[:,0], data[:,1], 'r-', linewidth=1, markersize=0, mec='r', mew=2, mfc='pink',
label="sin(x)"");
plt.fill_between(data[:,0], data[:,1], average, color=(1.0, 0.90, 0.90), lw=0)
plt.xlabel('Years before 1950')
plt.ylabel(r'Temperature - reference temperature [K]')
plt.show()
```