Activity 02 : Numerical Integration
Trapezoidal rule & Simpson's rule

1. Manipulate `linspace()` as vectorial object :

a. Within your folder, create a new notebook and name it: ***TP02.ipynb***

b. Write, then execute the following cell codes:

```python
import numpy as np
import matplotlib.pyplot as plt
```
Python

```python
a = 0
b = 10
n = 10
x = np.linspace(a, b, n+1)
```
Python

c. After that, add the next cell code and execute it. What could you say about **`f`** and **`f[:]`**?

```python
print(x, '\n')
print(x[:])
```
Python

d. Modify the previous cell code as follows, and notice what each line provides.

```python
print('x = ', x, '\n')
print('x[:] = ', x[:], '\n')
print('x[1:] = ', x[1:], '\n')
print('x[:n]= ', x[:n], '\n')
print('x[odd] = ', x[1:n:2], '\n')
print('x[even] = ', x[:n:2], '\n')
print('Sum of x-values = ', sum(x[:]), '\n')
print('Sum of odd values of x = ', sum(x[1:n:2]), '\n')
print('Sum of even values of x = ', sum(x[2:n:2]), '\n')
```
Python

Activity 02 : Numerical Integration
Trapezoidal rule & Simpson's rule

2. Numerical integration

a. Now let's try to write a python code able to evaluate the integration of a given function. To do, add new code cells with a new **`linspace()`** definition to evaluate a $\boldsymbol{sin}(\boldsymbol{x})$ on the interval $[\mathbf{0}, \boldsymbol{\pi}]$.

```python
a = 0
b = np.pi
n = 10
h = (b - a) / n
x = np.linspace(a, b, n+1)
f = np.sin(x)
```
Python

```python
print('h = (b-a)/N = ', h, '\n')
print('x = ', x, '\n')
print('f(x) =', f)
plt.plot(x, f);
```
Python

b. Use the following code to use the left rectangle rule to evaluate the definite integral of $\boldsymbol{sin}(\boldsymbol{x})$:

```python
I_rectL = h * sum(f[:n])
err_rectL = 2 - I_rectL

print('Left Rectangle rule gives : ', I_rectL)
print('with error : ', err_rectL)
```
Python

c. Repeat this by using right rectangle rule, Midpoint rule, Trapezoidal rule, and Simpson's rule

```python
I_rectR = h * sum(f[1:])
I_mid = h * sum(np.sin((x[1:] + x[:n])/2))
I_trapz = (h/2)*(f[0] + 2 * sum(f[1:n]) + f[n])
I_simp = (h/3) * (f[0] + + 4*sum(f[1:n:2]) + 2*sum(f[2:n:2]) + f[n])
```
Python

d. Try your code with other functions on their corresponding intervals:

$h(x) = \cos(x) \; on \; [0, {}^{\pi}/_{2}]; \; g(x) = \sqrt{1 + cos^2(x)} \; on \; [0, \pi]; \;\; k(x) = e^x \; on \; [-5, 5]$

$p(x) = e^{-x^2/2} \; on \; [-5, 5]$