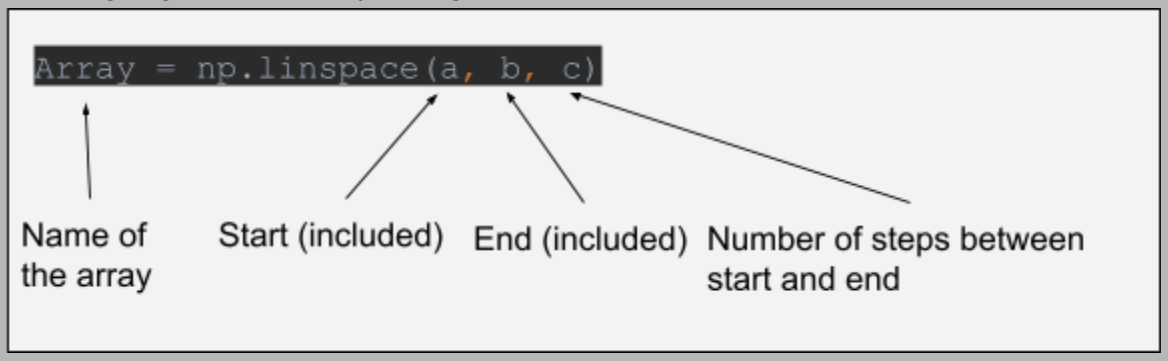
by Falk Geurten

Explanation - ex01_cfd0

Exercise 3

a.
We are going to create arrays using np.linspace:



N should be 11 and x_i is supposed to be between 0 and 1 so:

Array = np.linspace(0, 1, 10+1) or
Array = np.linspace(0, 1, 11)

the array now looks like this: [0. 0.1 0.2 0.3 0.4 0.5 0.6 0.7 0.8 0.9 1.]

The 10+1 instead of 11 exists to show that the +1 is only there because the 0 is included and we want simple and easy numbers at the moment.
The name of the array can be anything we want.
We will always shorten “numpy.” to “np.”

---

b.
We are going to change the difference between the individual numbers in our array to 0.025 instead of 0.1:

We can achieve that, by multiplying the 10 in 10+1 by 4. This works because our difference goes from 0.1 to 0.025 and 0.1 divided by 4 is 0.025.

Array = np.linspace(0, 1, 10*4+1) or
Array = np.linspace(0, 1, 41)

the array now looks like this: [0. 0.025 0.05 0.075 0.1 0.125 0.15 0.175 0.2 0.225 0.25 0.275 0.3 0.325 0.35 0.375 0.4 0.425 0.45 0.475 0.5 0.525 0.55 0.575 0.6 0.625 0.65 0.675 0.7 0.725 0.75 0.775 0.8 0.825 0.85 0.875 0.9 0.925 0.95 0.975 1.]

To see and be sure about ur arrays print them with print().

c.
We are going to define some mathematical functions (y1, y2 and y3):
The first step to doing this is to use the previous array for the x values:

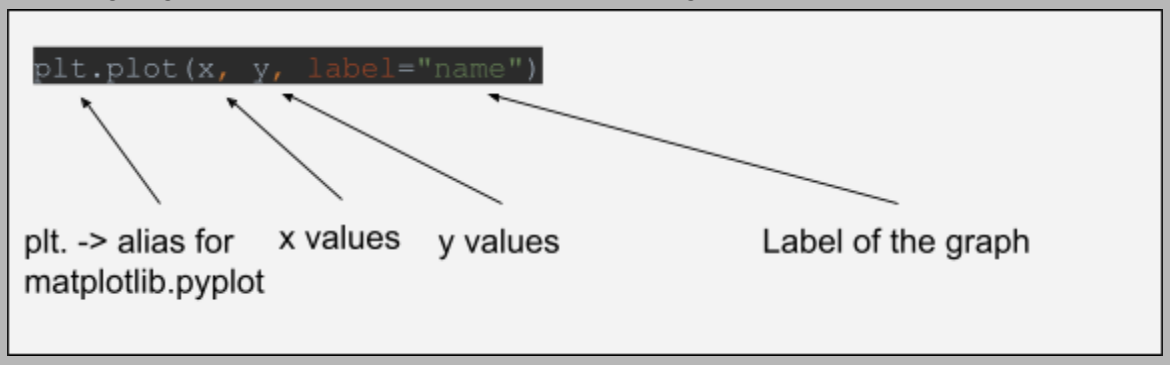
x = np.linspace(0, 1, 41)

The second step is to define the functions using np for sin() and π. (fyi in Python x^y is written as x**y

```
y1 = 1-2*x
y2 = (x-0.4)**2
y3 = np.sin(2*np.pi*x)
```

---

d.
We are going to plot the mathematical functions using plot():




```
plt.plot(x, y1, label="y1 - f1(x)")
plt.plot(x, y2, label="y1 - f2(x)")
plt.plot(x, y3, label="y1 - f3(x)")
```

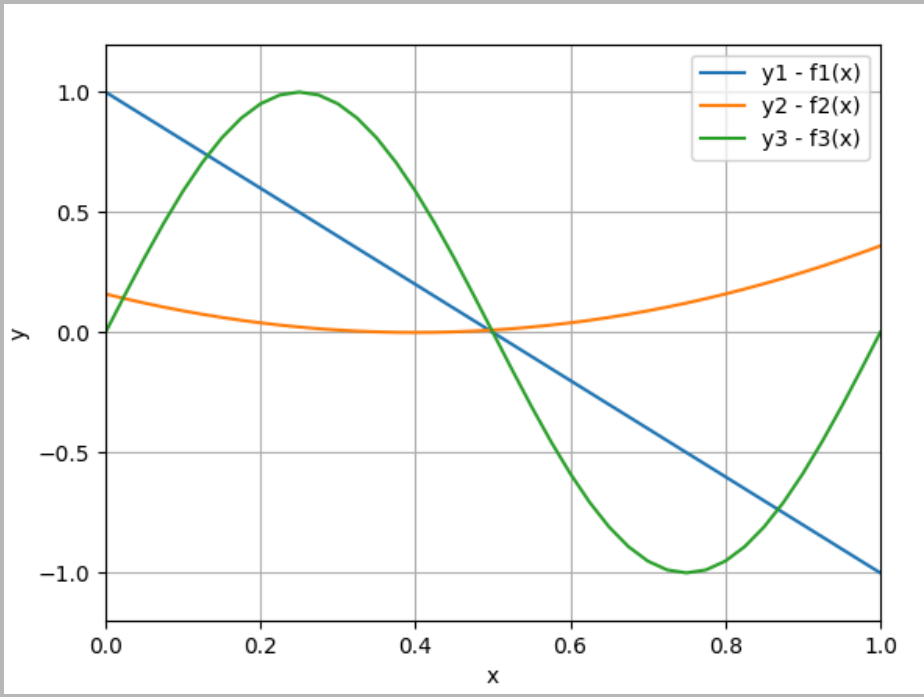
To make the plot more readable, we are going to add axis labels, grid lines and legend:

```
plt.xlabel("x")
plt.ylabel("y")
plt.legend()
plt.grid()
```

The last thing we do is making limits for both axis and plotting it:

```
plt.xlim(0, 1)
plt.ylim(-1.2, 1.2)
plt.show()
```

The graph (output) should now look like this:



---

e.
We now want the third graph to only be plotted from 0 to 0.5:
There are many ways to do this task. The easiest one is to copy the code from exercise c and d.

```
x = np.linspace(0, 1, 41)
y3 = np.sin(2*np.pi*x)
plt.plot(x, y3, label="y3 - Additional Task")
```

And then limit the x axis to 0.5 and plot the graph:

```
plt.xlim(0, 0.5)
plt.ylim(-1.2, 1.2)
plt.show()
```

This is how the graph should look: