in numpy, creating a matrix of 100X100 items is like this:

import numpy as np

x = np.ndarray((100, 100))

x.shape # outputs: (100, 100)

numpy internally stores all these 10000 items in an array of 10000 items regardless of the shape of this object, this allows us to change the shape of this array into any dimensions **as long as the number of items on the array does not change**

for example, reshaping our object to 10X1000 is ok as we keep the 10000 items:

x = x.reshape((10, 1000))

reshaping to 10X2000 wont work as we does not have enough items on the list

x.reshape((10, 2000))

ValueError: total size of new array must be unchanged

so back to the -1 question, what it does is the notation for unknown dimension, meaning: let numpy fill the missing dimension with the correct value so my array remain with the same number of items.

so this:

x = x.reshape((10, 1000))

is equivalent to this:

x = x.reshape((10, -1))

internally what numpy does is just calculating 10000 / 10 to get the missing dimension.

-1 can even be on the start of the array or in the middle.

the above two examples are equivalent to this:

x = x.reshape((-1, 1000))

if we will try to mark two dimensions as unknown, numpy will raise an exception as it cannot know what we are meaning as there are more than one way to reshape the array.

x = x.reshape((-1, -1))

ValueError: can only specify one unknown dimension