Task 1: Implementation of the software artefact Cube

INTRODUCTION:

In this project we have decided to use **Python** because it is a programming language which is really legible and very easy to use as well as it has got a lot of practical libraries in order to make our implementation easier. Moreover, we have chosen this programming language for doing several tests, which is also easy to develop. This implementation can be seen in the GitHub repository: https://github.com/Equecevi/A1-02.git. Besides, it also contains all the information about the members of the group.

IMPLEMENTATION:

About the implementation, we have done several methods at this moment:

- An **initial constructor** in order to create an object Cube importing the .json file.
- We have decided to implement each side of the cube with an attribute because in python the object attributes are elements of an inside dictionary, so is the same.
- We have chosen the numpy library to implement the side matrices for various reasons:
    - This implementation is made in a way is **better in terms of memory efficiency**, because in python lists are so inefficient.
    - Is **easier** for us to work with matrices, we can access the columns in a easier way, and matrix iterators are implemented in an easier and efficient way.
    - We can **adjust better the used memory** for each element of the matrix, in this case 8 bits integers, which is better than the normal python integer (32 bits)
- **Movement methods**: This general method is divided into three specific ones, depending on the axis that we want to move the cube: Back-Front, Up-Down and Left-Right. Those methods first checked if the movement is going to be forward (+90º) or backward (-90º), depending if the letter is in upper case or in lower case.
- Creation of the **md5 configuration** which returns the identifier of the cube state which depends of the colour's combination.
- Moreover, we have used some libraries in order to make easier the representation of the movements. These libraries are: **numpy, math, json** and **hashlib**. These two final libraries are to import and export into .json file and to implement the md5.
- The final method allows us to represent the Cube using the library **matplotlib.** This implementation is divided into two steps. The first one plots the general matrix which will contain all the faces of the Cube. And the second one plots every face of the cube.

<u>TESTING:</u>

We have implemented an extra library which is called ***pytest***. This library allows us to write small tests using *assert* statement.

- The first test checks if the valid movements of a 3x3x3 cube are generated correctly.
- The second one checks what happen if the md5 is equal or different from the initial one, once the cube has been turned.
- The third one checks that if one cube has been turned four times (360º) from the same axis it would be the same.
- And the final one is going to be the same as in the previous test but with all the valid movements.