Homework 1

In this homework assignment you will create new Python classes and upload your code to github so they can be tested on their servers. You will implement classes to represent quaternions and octonions in the file `extended_numbers.py`, in classes named `Q` and `O` respectively.

Quaternions are an extension of the complex numbers. Recall the complex numbers can be written as $a = x + \boldsymbol{i}y$, with the property $\boldsymbol{i}^2 = -1$. Quaternions are written as $a = a_0 + a_1\boldsymbol{i} + a_2\boldsymbol{j} + a_3\boldsymbol{k}$, where a_i are real numbers. The square of $\boldsymbol{i}$, $\boldsymbol{j}$, and $\boldsymbol{k}$ are all equal to -1. We also have $\boldsymbol{i} \times \boldsymbol{j} = \boldsymbol{k}$, $\boldsymbol{j} \times \boldsymbol{i} = -\boldsymbol{k}$, as well as cyclic permutations of these identities. Thus, multiplication of quaternions is *not commutative*. Your quaternion class `Q` will

- Instantiate a quaternion from a list (or tuple or array) of four numbers. You should store the data as a `numpy` array.
- Determine if two quaternions are equal.
- Add two quaternions.
- Subtract two quaternions.
- Multiply a quaternion by a real number. Note that you will need to implement both `__mul__` and `__rmul__` methods for this.
- Multiply two quaternions by each other.

A test script testing these features is on the course canvas page.

Octonions are an extension of the quaternions. We can write an octonion as

$$a = \sum_i a_i \boldsymbol{e}_i,$$

where $\boldsymbol{e}_0$ is the identity, so

$$\boldsymbol{e}_0 \times \boldsymbol{e}_i = \boldsymbol{e}_i \times \boldsymbol{e}_0 = \boldsymbol{e}_i$$

For $i \neq 0$, we have that $\boldsymbol{e}_i^2 = -\boldsymbol{e}_0$. Otherwise, the multiplication rule is that

$$\boldsymbol{e}_i \times \boldsymbol{e}_j = \epsilon_{i,j,k}\boldsymbol{e}_k.$$

The tensor $\epsilon_{i,j,k}$ is totally antisymmetric, i.e., if you interchange any two indices, it flips signs $\epsilon_{i,j,k} = -\epsilon_{j,i,k}$. There are different conventions for octonions, but we will use that $\epsilon_{1,2,3} = \epsilon_{1,4,5} = \epsilon_{1,7,6} = \epsilon_{2,4,6} = \epsilon_{2,5,7} = \epsilon_{3,4,7} = \epsilon_{3,6,5} = 1$. The quaternions are the restriction of the octonions to $i \in [0, 3]$.

Multiplication of octonions is *not associative*. That means that

$$(a \times b) \times c \neq a \times (b \times c).$$

For this reason, we will not use the `*` operator to encode multiplication of octonions. It would be confusing to interpret code like `a*b*c` (does that mean `(a*b)*c` or `a*(b*c)`?). Thus, we will define a `mul` function, so $a * b$ is given by `a.mul(b)`. You will implement an octonion class `O` that will

- Instantiate an octonion from a list (or tuple or array) of eight numbers. You should store the data as a `numpy` array.
- Determine if two octonions are equal.
- Add two octonions
- Subtract two octonions
- Multiply an octonion by a real number using `*`. Note that you will need to implement both `__mul__` and `__rmul__` methods for this.
- Implement a `mul` method that multiplies two octonions together.

When the homework is due, I will upload a new test script to your github repo to test the octonion class. If you miss any tests, you have one week to update your code, after which I will rerun the tests. You get half credit for any tests you pass on the second try that you missed the first time.

For both quaternions and octonions, I also encourage you to write a `__repr__` function, which could be useful for debugging.