Subject : EIE2108 Lab 1 Report

Student ID: 19069748D

Name: Kwok Kevin

Details of Lab 1:

During the lab activity, students' mission is creating a class named Quaternion, which can handle quaternion algebras.

An instance of the class is a quaternion (= a+ b**i** + c**j**+ d**k**) in a form of list [a, b, c, d]. There are 5 methods in the definition of students' class: Add(), Mul(), Conj(), Inv() and Norm(). Those functions through the class will be used to make the 5 aforementioned operations respectively.

While students finish the defining of the class, they are required to provide 3 set of non-zero quaternion objects ((z1, z2 and z3). Base on these 3 set of non-zero quaternion objects, there are 6 computations to compete:

1. $z_3 z_2$
2. $z_1 - z_3$
3. $z_1^* z_3$
4. z_3^{-3}
5. $\|z_2\|$
6. $z_1^3 z_2^* z_3^{-4} (z_1 + z_3)^2 / \|z_2\| (z_1 + z_2)$

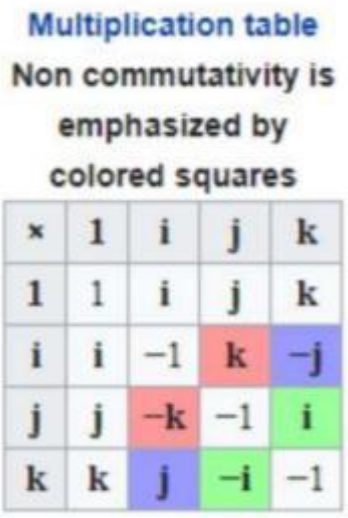
Background:

In mathematics, the quaternions are a number system that extends the complex numbers. Quaternions are generally represented in the form:

$$a + b\mathbf{i} + c\mathbf{j} + d\mathbf{k}$$

where a, b, c, and d are real numbers, and i, j, and k are the fundamental quaternion units. The product of 2 fundamental quaternion units has the following properties:

- $\mathbf{i} \cdot 1 = 1 \cdot \mathbf{i} = \mathbf{i}, \qquad \mathbf{j} \cdot 1 = 1 \cdot \mathbf{j} = \mathbf{j}, \qquad \mathbf{k} \cdot 1 = 1 \cdot \mathbf{k} = \mathbf{k}.$
- $\mathbf{i}^2 = \mathbf{j}^2 = \mathbf{k}^2 = \mathbf{ijk} = -1 .$
- $\mathbf{ij} = \mathbf{k}, \qquad \mathbf{ji} = -\mathbf{k},$
  $\mathbf{jk} = \mathbf{i}, \qquad \mathbf{kj} = -\mathbf{i},$
  $\mathbf{ki} = \mathbf{j}, \qquad \mathbf{ik} = -\mathbf{j}.$



Multiplication table
Non commutativity is emphasized by colored squares

×	1	i	j	k
1	1	i	j	k
i	i	−1	k	−j
j	j	−k	−1	i
k	k	j	−i	−1

Table 1

Table 1 summaries the products of different fundamental quaternion units. Some basic algebra operations of quaternions are listed as follows:

Background:(Con't)

Some basic algebra operations of quaternions are listed as follows:

1. Addition

   For two elements $z_1 = a_1 + b_1\mathbf{i} + c_1\mathbf{j} + d_1\mathbf{k}$ and $z_2 = a_2 + b_2\mathbf{i} + c_2\mathbf{j} + d_2\mathbf{k}$, we have

$$z_1 + z_2 = (a_1 + a_2) + (b_1 + b_2)\,\mathbf{i} + (c_1 + c_2)\,\mathbf{j} + (d_1 + d_2)\,\mathbf{k}\,;$$

2. Multiplication

   For two elements $z_1 = a_1 + b_1\mathbf{i} + c_1\mathbf{j} + d_1\mathbf{k}$ and $z_2 = a_2 + b_2\mathbf{i} + c_2\mathbf{j} + d_2\mathbf{k}$, their product, called the **Hamilton product** $(a_1 + b_1\mathbf{i} + c_1\mathbf{j} + d_1\mathbf{k})$ $(a_2 + b_2\mathbf{i} + c_2\mathbf{j} + d_2\mathbf{k})$, is determined by the products of the basis elements and the distributive law. This gives the following expression:

$$\begin{aligned} z_1 z_2 = {} & a_1a_2 + a_1b_2\mathbf{i} + a_1c_2\mathbf{j} + a_1d_2\mathbf{k} + b_1a_2\mathbf{i} + b_1b_2\mathbf{i}^2 + b_1c_2\mathbf{ij} + b_1d_2\mathbf{ik} \\ & + c_1a_2\mathbf{j} + c_1b_2\mathbf{ji} + c_1c_2\mathbf{j}^2 + c_1d_2\mathbf{jk} + d_1a_2\mathbf{k} + d_1b_2\mathbf{ki} + d_1c_2\mathbf{kj} + d_1d_2\mathbf{k}^2 \end{aligned}$$

   Based on the properties shown in Table1, it can be reformulated as

$$\begin{aligned} z_1 z_2 = {} & a_1a_2 - b_1b_2 - c_1c_2 - d_1d_2 + (a_1b_2 + b_1a_2 + c_1d_2 - d_1c_2)\mathbf{i} \\ & + (a_1c_2 - b_1d_2 + c_1a_2 + d_1b_2)\mathbf{j} + (a_1d_2 + b_1c_2 - c_1b_2 + d_1a_2)\mathbf{k} \end{aligned}$$

3. Conjugate

   The conjugate of $q = a_1 + b_1\mathbf{i} + c_1\mathbf{j} + d_1\mathbf{k}$ is $q^* = a_1 - b_1\mathbf{i} - c_1\mathbf{j} - d_1\mathbf{k}$.

4. Inverse

   Every nonzero quaternion has an inverse with respect to the Hamilton product

$$(a + b\,\mathbf{i} + c\,\mathbf{j} + d\,\mathbf{k})^{-1} = \frac{1}{a^2 + b^2 + c^2 + d^2}\,(a - b\,\mathbf{i} - c\,\mathbf{j} - d\,\mathbf{k})$$

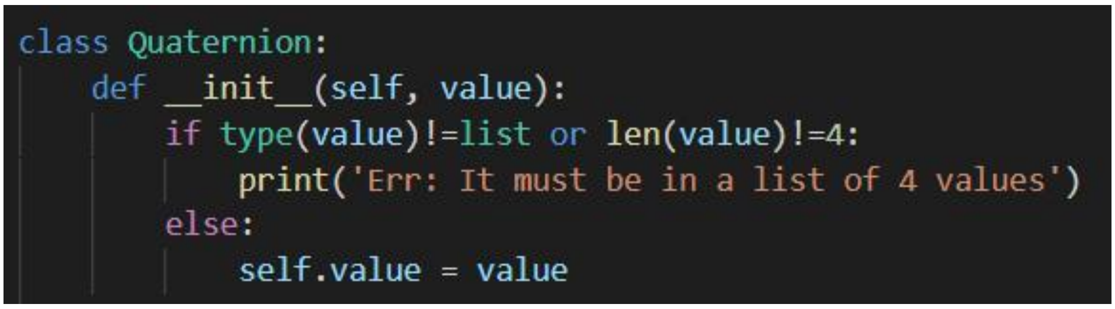
5. Norm

   The square root of the product of a quaternion with its conjugate is called its *norm* and is denoted as $\|q\|$. In formulation, it is expressed as follows:

$$\|q\| = \sqrt{qq^*} = \sqrt{q^*q} = \sqrt{a^2 + b^2 + c^2 + d^2}$$

Analysis and implementation of Codes:

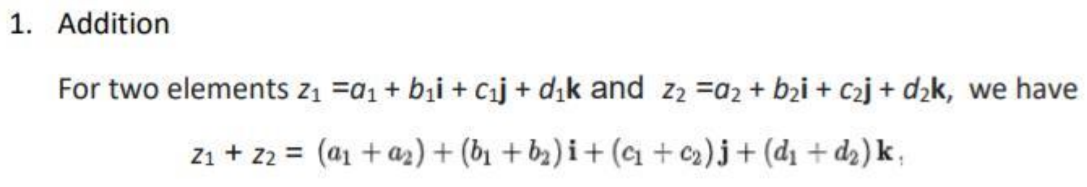
First of all, I have to create a class and the following function to make people enter the correct format of quaternion objects



```python
class Quaternion:
    def __init__(self, value):
        if type(value)!=list or len(value)!=4:
            print('Err: It must be in a list of 4 values')
        else:
            self.value = value
```

Second, there are 5 function to be created to realize the 5 aforementioned operations respectively, which are Add(), Mul(), Conj(), Inv() and Norm()

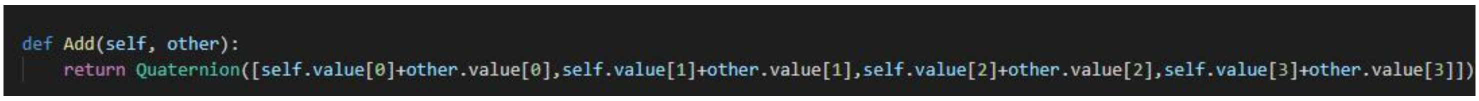
For definition of the Add():



1. Addition

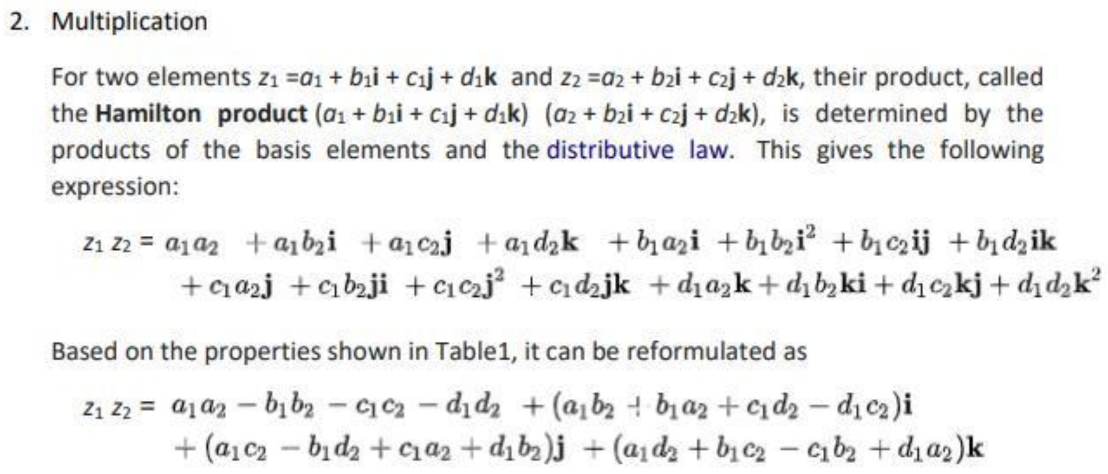
For two elements $z_1 = a_1 + b_1\mathbf{i} + c_1\mathbf{j} + d_1\mathbf{k}$ and $z_2 = a_2 + b_2\mathbf{i} + c_2\mathbf{j} + d_2\mathbf{k}$, we have

$$z_1 + z_2 = (a_1 + a_2) + (b_1 + b_2)\,\mathbf{i} + (c_1 + c_2)\,\mathbf{j} + (d_1 + d_2)\,\mathbf{k},$$



```python
def Add(self, other):
    return Quaternion([self.value[0]+other.value[0],self.value[1]+other.value[1],self.value[2]+other.value[2],self.value[3]+other.value[3]])
```

For definition of the Mul():



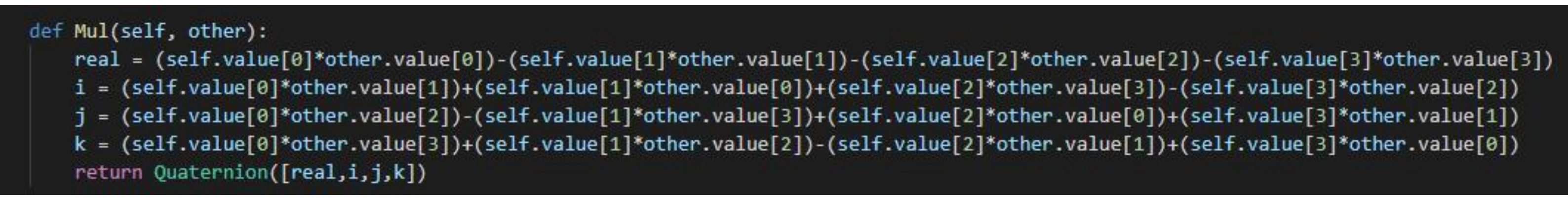
2. Multiplication

For two elements $z_1 = a_1 + b_1\mathbf{i} + c_1\mathbf{j} + d_1\mathbf{k}$ and $z_2 = a_2 + b_2\mathbf{i} + c_2\mathbf{j} + d_2\mathbf{k}$, their product, called the **Hamilton product** $(a_1 + b_1\mathbf{i} + c_1\mathbf{j} + d_1\mathbf{k})\ (a_2 + b_2\mathbf{i} + c_2\mathbf{j} + d_2\mathbf{k})$, is determined by the products of the basis elements and the distributive law. This gives the following expression:

$$\begin{aligned} z_1 z_2 = {} & a_1a_2 + a_1b_2\mathbf{i} + a_1c_2\mathbf{j} + a_1d_2\mathbf{k} + b_1a_2\mathbf{i} + b_1b_2\mathbf{i}^2 + b_1c_2\mathbf{ij} + b_1d_2\mathbf{ik} \\ & + c_1a_2\mathbf{j} + c_1b_2\mathbf{ji} + c_1c_2\mathbf{j}^2 + c_1d_2\mathbf{jk} + d_1a_2\mathbf{k} + d_1b_2\mathbf{ki} + d_1c_2\mathbf{kj} + d_1d_2\mathbf{k}^2 \end{aligned}$$

Based on the properties shown in Table1, it can be reformulated as

$$\begin{aligned} z_1 z_2 = {} & a_1a_2 - b_1b_2 - c_1c_2 - d_1d_2 + (a_1b_2 + b_1a_2 + c_1d_2 - d_1c_2)\mathbf{i} \\ & + (a_1c_2 - b_1d_2 + c_1a_2 + d_1b_2)\mathbf{j} + (a_1d_2 + b_1c_2 - c_1b_2 + d_1a_2)\mathbf{k} \end{aligned}$$

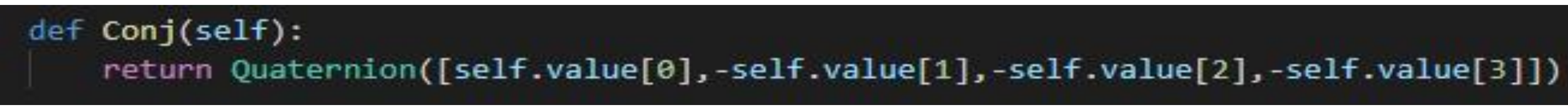


```python
def Mul(self, other):
    real = (self.value[0]*other.value[0])-(self.value[1]*other.value[1])-(self.value[2]*other.value[2])-(self.value[3]*other.value[3])
    i = (self.value[0]*other.value[1])+(self.value[1]*other.value[0])+(self.value[2]*other.value[3])-(self.value[3]*other.value[2])
    j = (self.value[0]*other.value[2])-(self.value[1]*other.value[3])+(self.value[2]*other.value[0])+(self.value[3]*other.value[1])
    k = (self.value[0]*other.value[3])+(self.value[1]*other.value[2])-(self.value[2]*other.value[1])+(self.value[3]*other.value[0])
    return Quaternion([real,i,j,k])
```

For definition of the Conj():

3. Conjugate

The conjugate of $q = a_1 + b_1\mathbf{i} + c_1\mathbf{j} + d_1\mathbf{k}$ is $q^* = a_1 - b_1\mathbf{i} - c_1\mathbf{j} - d_1\mathbf{k}$.



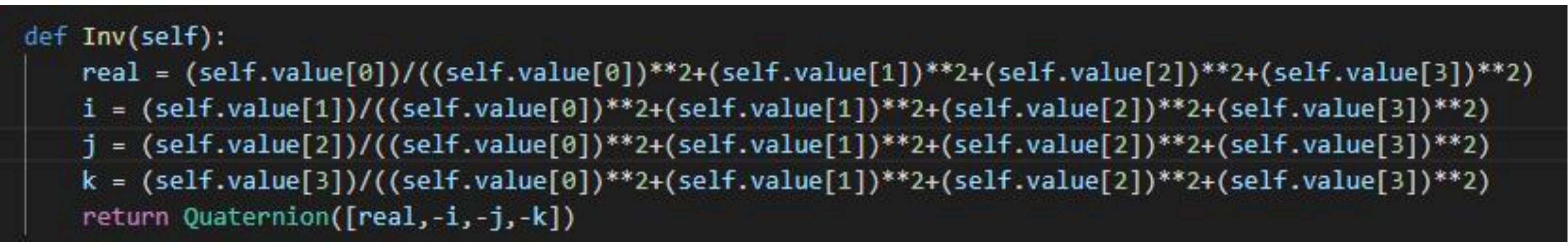
```
def Conj(self):
    return Quaternion([self.value[0],-self.value[1],-self.value[2],-self.value[3]])
```

For definition of the Inv():

4. Inverse

Every nonzero quaternion has an inverse with respect to the Hamilton product

$$(a + b\mathbf{i} + c\mathbf{j} + d\mathbf{k})^{-1} = \frac{1}{a^2 + b^2 + c^2 + d^2}(a - b\mathbf{i} - c\mathbf{j} - d\mathbf{k})$$



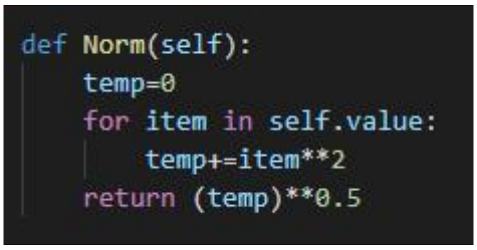
```
def Inv(self):
    real = (self.value[0])/((self.value[0])**2+(self.value[1])**2+(self.value[2])**2+(self.value[3])**2)
    i = (self.value[1])/((self.value[0])**2+(self.value[1])**2+(self.value[2])**2+(self.value[3])**2)
    j = (self.value[2])/((self.value[0])**2+(self.value[1])**2+(self.value[2])**2+(self.value[3])**2)
    k = (self.value[3])/((self.value[0])**2+(self.value[1])**2+(self.value[2])**2+(self.value[3])**2)
    return Quaternion([real,-i,-j,-k])
```

For definition of the Norm():

5. Norm

The square root of the product of a quaternion with its conjugate is called its *norm* and is denoted as $\|q\|$. In formulation, it is expressed as follows:

$$\|q\| = \sqrt{qq^*} = \sqrt{q^*q} = \sqrt{a^2 + b^2 + c^2 + d^2}$$



```
def Norm(self):
    temp=0
    for item in self.value:
        temp+=item**2
    return (temp)**0.5
```

After setting up the functions mentioned above:

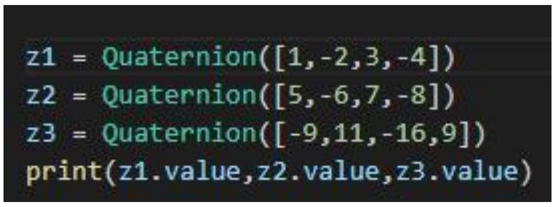
I would set 3 non-zero quaternion objects

Z1 = 1-2i+3j-4k

Z2 = 5-6i+7j-8k

Z3 = -9+11i-16j+9k

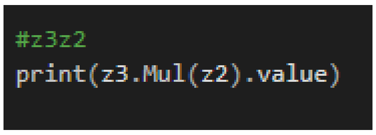
And then print 3 of them out



```python
z1 = Quaternion([1,-2,3,-4])
z2 = Quaternion([5,-6,7,-8])
z3 = Quaternion([-9,11,-16,9])
print(z1.value,z2.value,z3.value)
```

The leaving mission do the 6 tasks with these 3 zx number sets

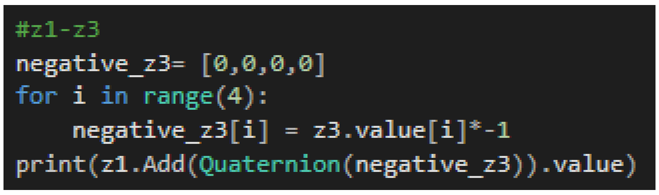
For 1. $z_3 z_2$



```python
#z3z2
print(z3.Mul(z2).value)
```

Since we have defined the multiplication function. With completing task 1, just simply take the value with command that we built up in the class and print it out like above.

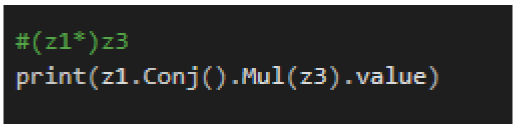
For 2. $z_1 - z_3$



```python
#z1-z3
negative_z3= [0,0,0,0]
for i in range(4):
    negative_z3[i] = z3.value[i]*-1
print(z1.Add(Quaternion(negative_z3)).value)
```

Since we only build the quaternion with it but not subtraction. We have make the z3 number set become negative with the for loop to make all z3 number become negative side. Then add them z1 and negative z3 to fulfill task 2.

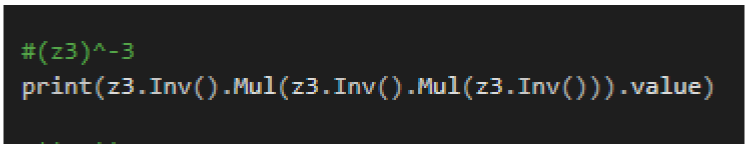
For 3. $z_1^* z_3$



```python
#(z1*)z3
print(z1.Conj().Mul(z3).value)
```

Similar to task 1, just z1 have to work with conjugate before the multiplication with z3.

For task 4. z_3^{-3}



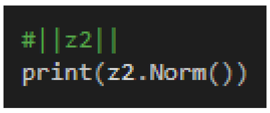
```
#(z3)^-3
print(z3.Inv().Mul(z3.Inv().Mul(z3.Inv())).value)
```

We have to do z3 with the index -3 and -3 = -1 * 3
So we can understand the task 4 as inverse of z3 takes 3 times multiplication of itself. Therefore, there are :

z3 inverse * z3 inverse * z3 inverse

shown above

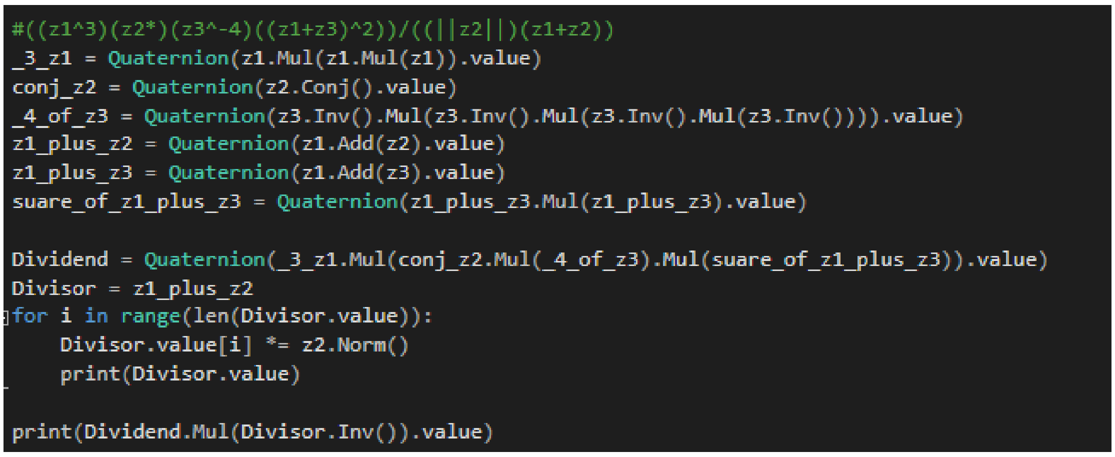
For task 5. $\|z_2\|$



```
#||z2||
print(z2.Norm())
```

Since we have defined the norm function to take the root of square of object. It directly complete what task 5 works, just simply take the z2 number set with command that we built up in the class and print it out like above.

For task 6. $z_1^3 z_2^* z_3^{-4}\,(z_1+z_3)^2/\,\|z_2\|(z_1+z_2)$



```python
#((z1^3)(z2*)(z3^-4)((z1+z3)^2))/((||z2||)(z1+z2))
_3_z1 = Quaternion(z1.Mul(z1.Mul(z1)).value)
conj_z2 = Quaternion(z2.Conj().value)
_4_of_z3 = Quaternion(z3.Inv().Mul(z3.Inv().Mul(z3.Inv().Mul(z3.Inv()))).value)
z1_plus_z2 = Quaternion(z1.Add(z2).value)
z1_plus_z3 = Quaternion(z1.Add(z3).value)
suare_of_z1_plus_z3 = Quaternion(z1_plus_z3.Mul(z1_plus_z3).value)

Dividend = Quaternion(_3_z1.Mul(conj_z2.Mul(_4_of_z3).Mul(suare_of_z1_plus_z3)).value)
Divisor = z1_plus_z2
for i in range(len(Divisor.value)):
    Divisor.value[i] *= z2.Norm()
    print(Divisor.value)

print(Dividend.Mul(Divisor.Inv()).value)
```

This task 6 is a final boss that fusion from task 1 to task 5.
Since there are all elements required though the functions we have built before.

The first step to complete this task 6.
We have to define the value of the element in this task
Like z1^3,conjugate of z2, z3^-4 , (z1+z3)^2, ||z2|| and (z1+z2)

For z1^3, we have to do the multiplication 2 time of itself.
For conjugate of z2, take the class function command -conj with z2.
For z3^-4, like task 4, multiplicate with inversed it 4 times.
For (z1+z3), we take the addition with z1 and z3, the square will take it later.
For ||z2||, similar to task 5,dricetly do the command that the function we build up before in the class.
For (z1+z2), like z1+ z3, just do the addition with function add in class.

Then we take the dividend and divisor with the correct elements.

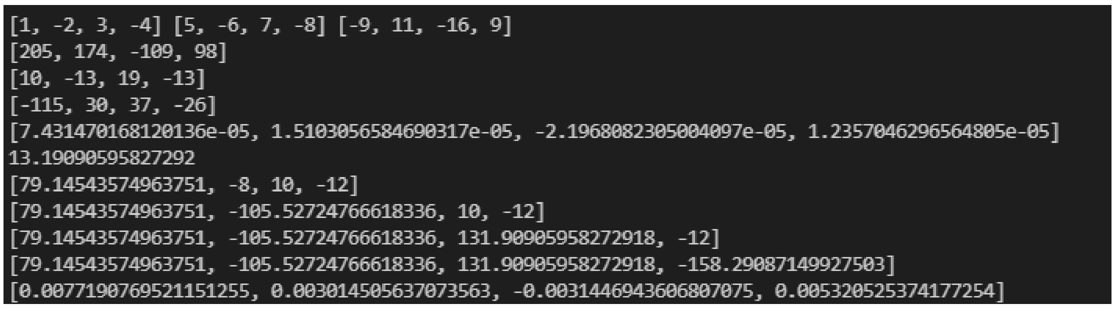
Dividend will be $z_1^3 z_2^* z_3^{-4}\,(z_1+z_3)^2$

Divisor will be $\|z_2\|(z_1+z_2)$

The for loop is to complete the divisor multiplicate between norm.z2 and (z1+z2).

And the make the task works.

This is the outcome of the program.



```
[1, -2, 3, -4] [5, -6, 7, -8] [-9, 11, -16, 9]
[205, 174, -109, 98]
[10, -13, 19, -13]
[-115, 30, 37, -26]
[7.431470168120136e-05, 1.5103056584690317e-05, -2.1968082305004097e-05, 1.2357046296564805e-05]
13.19090595827292
[79.14543574963751, -8, 10, -12]
[79.14543574963751, -105.52724766618336, 10, -12]
[79.14543574963751, -105.52724766618336, 131.90905958272918, -12]
[79.14543574963751, -105.52724766618336, 131.90905958272918, -158.29087149927503]
[0.0077190769521151255, 0.003014505637073563, -0.0031446943606807075, 0.005320525374177254]
```