P3: Into the imaginary realm
https://www.w3schools.com/cpp/cpp_classes.asp

Complex Number Class - 13/02/21

1) Design a class to represent the complex number with appropriate constructors. Your constructor should be overloaded appropriately with the correct overloading method. Implement your operators as member functions in your class.

To make my constructor I declared a function, with no type, that would then read in type of two floats. This was overloaded so that it would set the real and imaginary variables of the class to the values it read in.

```
class complex
{
public:
    complex(float re, float im)
    {
        real = re;
        imaginary = im;
    }
    float Re(void)
    {
        return real;
    }
    float Im(void)
    {
        return imaginary;
    }
    float mod(void)
    {
        return sqrt(pow(real, 2) + pow(imaginary, 2));
    }

private:
    float real;
    float imaginary;
};
```

2) Design appropriate overloading schemes for out-of-class overloading support for basic build-in operators.

I decided which operators I wanted to create overloading schemes for, namely the arithmetic and the comparative operators, and then overloaded them so that they would perform the operation on the class and return the appropriate value.

```
    complex operator+(complex &a) const
    {
        return complex(real + a.real, imaginary + a.imaginary);
    }
    complex operator-(complex &a) const
    complex operator*(complex &a) const
    complex operator/(complex &a) const

    void operator+=(complex &a)
    {
        real += a.real;
        imaginary += a.imaginary;
    }
    void operator-=(complex &a)
    void operator*=(complex &a)
    void operator/=(complex &a)

    bool operator==(complex &a) const
    {
        if ((real == a.real) && (imaginary == a.imaginary))
        {
            return 1;
        }
        else
        {
            return 0;
        }
    }
bool operator!=(complex &a) const
```

Impedance of circuits - 16/02/21

3) How do you find the real part of impedance in an RLC circuit?

$$Z = R + jX$$
$$\Re Z = R$$

4) How do you calculate the imaginary part of the impedance of an RLC circuit?

$$Z = R + jX$$
$$\Im Z = X$$
$$\Im Z = X_L - X_C$$
$$\Im Z = \omega L - \frac{1}{\omega C}$$

5) Write a simple function that takes in the total series resistance, capacitance, inductance and frequency of a series RLC circuit and convert it into impedance.

```
float RLC(float resistance, float inductance, float capacitance, float freq)
{
    float omega = 2 * M_PI * freq;

    float XL = omega * inductance;
    float XC = 1 / (omega * capacitance);

    return sqrt(pow(resistance, 2) + pow(XL - XC, 2));
}
```

6) Why is impedance important in a circuit?

Impedance is important in the calculator of the response of the circuit as ohm's law describes the voltage as:

$$V = IZ$$

And so to calculate the current of the RLC circuit it will be :

$$I = \frac{V}{Z}$$

Implement complex class - 17/02/21

To test the functions I created a main function that would initialise a complex number then print the result of the function to the terminal.

```
int main()
{
    complex A = complex(3, 4);

    std::cout << "Your value is: ";
    print_cartesian(A);

    std::cout << "The real part is: " << A.Re() << std::endl;
    std::cout << "The imaginary part is: " << A.Im() << std::endl;
    std::cout << std::endl;

    std::cout << "The modulus is: " << A.mod() << std::endl;
    std::cout << "The argument is: " << A.arg() << std::endl;
    std::cout << std::endl;

    std::cout << "The conjugate of your value is: ";
    print_cartesian(A.conj());
}
```

When the function returned a complex class I created a function that would correctly print the values to terminal. When more complex variables are needed to be printed this is useful as it reduces the amount of times the line has to be used.

```
void print_cartesian(complex a)
{
    if (a.Im() >= 0)
    {
        std::cout << a.Re() << " + j" << a.Im() << std::endl;
    }
    else
    {
        std::cout << a.Re() << " - j" << a.conj().Im() << std::endl;
    }
}
```

The resulting output to terminal for my main function was.

```
Your value is: 3 + j4
The real part is: 3
The imaginary part is: 4

The modulus is: 5
The argument is: 0.927295

The conjugate of your value is: 3 - j4
```

Convert RLC to impedance - 17/02/21

Due to misunderstanding the prep I had to modify my RLC function to return a complex class. To do this I changed the function type and modified the return function. This ultimately simplified my circuit.

For my main function I substituted the values from:
http://physicstasks.eu/1540/series-rlc-circuit

Into my RLC function to calculate the return value.

```
int main()
{
    float res = 50;
    float ind = 0.3;
    float cap = 0.000015;
    float freq = 50;

    complex imp = RLC(res, ind, cap, freq);

    std::cout << "Your complex impedance is: ";
    print_cartesian(imp);

    std::cout << "Your absolute impedance is: " << imp.mod();
}

complex RLC(float resistance, float inductance, float capacitance, float freq)
{
    float omega = 2 * M_PI * freq;

    float XL = omega * inductance;
    float XC = 1 / (omega * capacitance);

    return complex(resistance, XL - XC);
}
```

The output of the impedance calculated by my code was:

```
Your complex impedance is: 50 - j117.959
Your absolute impedance is: 128.118
```

RLC circuit simulator - 17/02/21

To calculate the current of the circuit I created a new RLC class. The classes constructor was based on my previous RLC function. I then had methods for returning values relating to the total impedance of the circuit and the complex current of the circuit.

```
class RLC
{
public:
    RLC(float resistance, float inductance, float capacitance, float volt,
float freq)
    {
        voltage = volt;
        omega = 2 * M_PI * freq;

        float XL = omega * inductance;
        float XC = 1 / (omega * capacitance);

        impedance = complex(resistance, XL - XC);
        current = complex(voltage, 0) / impedance;
    }

    complex imp(void)
    {
        return impedance;
    }
    complex cur()
    {
        return current;
    }

private:
    float voltage;
    float omega;
    complex impedance;
    complex current;
};
```

To create a user interface, I modified my main function to read in the values the values from the command line. It would then create the RLC class from this, hence calculating current.

```
int main()
{
    float res, ind, cap;
    float volt, freq;

    std::cout << "Your resistance is? ";
    std::cin >> res;
    std::cout << "Your inductance is? ";
    std::cin >> ind;
    std::cout << "Your capacitance is? ";
    std::cin >> cap;
    std::cout << "Your voltage amplitude is? ";
    std::cin >> volt;
    std::cout << "Your voltage frequency is? ";
    std::cin >> freq;

    RLC circuit = RLC(res, ind, cap, volt, freq);
    std::cout << std::endl << "Your current in complex form is: ";
    print_cartesian(circuit.cur());
    std::cout << "The magnitude of your current is: ";
    std::cout << circuit.cur().mod();
    std::cout << std::endl << "The phase difference between the voltage and current is: ";
    std::cout << 0 - circuit.cur().arg();
}
```

The values I calculated are consistent with those on the linked site.

```
Your resistance is? 50
Your inductance is? 0.3
Your capacitance is? 0.000015
Your voltage amplitude is? 25
Your voltage frequency is? 50

Your current in complex form is: 0.0761532 + j0.179659
The magnitude of your current is: 0.195132
The phase difference between the voltage and current is: -1.16988
```

In the series RLC cir

$I_m = 0.2$ A.

The phase difference

$\varphi = -67°$.