To implement the design of the class a few design decisions were made. The key decision are explained and justified below:

1. The class implemented operator overloads to perform:
    a. Addition, subtraction, multiplication and division of two ranges to produce a resulting range
    b. Addition, subtraction, multiplication and division of ranges and a native number of same type where the range is the left operand in the equation.
    c. Addition, subtraction, and multiplication of ranges and a native number of same type where the range is the right operand in the equation.
    d. For multiplication and division, sorting networks were used to optimize the determination of the upper and lower bounds.
    e. Other trivial numerical operators like >,<, =, += were also overloaded to perform corresponding operations.
    f. Beyond the above, other complexities like square root of range, division by 0, and division of a native numerical type by a range were deemed unnecessarily complex and were hence not implemented.

2. In terms of constructors the decision was made to implement initialization of a range from two number (upper and lower), no number/default to 0, copying another range into a new one and finally creating a range that represents a singular number (5-5).

3. In the context to which this class was applied, it was also deemed necessary to create member function to get the upper and lower bounds as well as the difference between the two and a midpoint which may be useful in situations where a range lies across a boundary.

4. Again, for the whole class was templated on one variable in the sense that you could not initialise a range where the lower bound was a float and the upper an int for example. In the same way you could not perform any arithmetic operations between and integer based range and a float based range to get an integer based result. (No casting between types was considered).

A few tests were run to validate the code and some of the results are shown below. A full test is available when the attached program is run.

R1 = 1:2, R2 = 3:4

Test	**Result**
R1 +R2	4 : 6
R1-R2	-3 : -1
R1/R2	0.25 : 0.666667
R1*R2	3 : 8
R1/-2.6	-0.769231 : -0.384615
2.6*R1	2.6 : 5.2
5-R1	3 : 4