Laboratory 11 — Classes and Objects

In this laboratory, we are going to build a class for rational numbers.

1. **Euclid's algorithm for the greatest common divisor:** Euclid's algorithm is a famous algorithm that calculates the greatest common divisor (gcd) of two numbers. A first version is based on the fact that if a number divides a and b, it also divides their sum and their difference. This implies that the set of divisors of two numbers a and b does not change if we replace the larger of the two with the difference of the larger minus the smaller number. This quite complex argument implies that the gcd also does not change if we subtract the smaller number from the larger one. For example,
   $\text{gcd}(252,122) = \text{gcd}(130,122) = \text{gcd}(122,8) = \text{gcd}(114,8) = \text{gcd}(96,8) =$
   $= \text{gcd}(88,8) = \ldots = \text{gcd}(16,8) = \text{gcd}(8,8) = \text{gcd}(0,8) = 8.$
   We can speed up the many subtractions in this calculation indicated by the ellipsis by using the modulo operator. In the final step of this calculation, all numbers divide zero, therefore the gcd of any non-zero number and 0 is that number. Implement Euclid's algorithm given below:

```
function gcd(a,b):
    if b is 0:
      return a
    else:
      return gcd(b, a%b)
```

2. Define the class Rational. The class has two integer fields, enumerator and denominator. Only the enumerator can be negative. The init-function takes two integers as arguments. If the prospective denominator is zero, a ZeroDivision exception should be raised. Both enumerator and denominator should be divided by their gcd. For example, if the prospective enumerator is -24 and the prospective denominator is -36, then the stored enumerator is 2 and the stored denominator is 3.

3. Define a string function (`__str__`) that returns for example "-10/9" if the stored enumerator is -10 and the stored denominator is 9.

4. Overwrite addition, subtraction, multiplication, and division. Remember that enumerator and denominator need to be free of common divisors. This however should be taken care of by the initializer. Also, raise a Zero Division Exception for a division by zero.

5. Create a method `tofloat` that returns the floating point value of a rational number.

6. Create comparison operators for equality, inequality, smaller, greater, smaller or equal, and greater or equal.