# Tartalomjegyzék

# 4/1. Modulo ring

Define a class called **Moduloz** representing modulo n numbers (integers).

For example in modulo 5:

```
4 + 3 = 2 (because 7 % 5 = 2)
2 - 3 = 4 (because -1 % 5 = 4)
4 * 3 = 2 (because 12 % 5 = 2)
```

You don't have to implement the operations yet, just define the **__init__** and the **__str__** methods.

In the constructor you will have two parameters. The first one is the base of the modulo, the second one is the actual number.

The base will be a positive integer, the value will be an integer.

The **__str__** should return a string, containing the value.

For example: a = Moduloz(5, 7) print a

Should print: 2

# 4/2. Modulo ring operations

Implement the **__add__**, **__sub__**, **__mul__** methods for the previous **Moduloz** class!

For example in modulo 5:

```
4 + 3 = 2 (because 7 % 5 = 2)
2 - 3 = 4 (because -1 % 5 = 4)
4 * 3 = 2 (because 12 % 5 = 2)
```

Mind that the operations should return an object of class Moduloz, not an integer (int)!

For example:

```
a = Moduloz(7, 9)
b = Moduloz(7, 12)

print a + b
print a - b
print a * b
```

should print:

```
0
4
3
```

In the test outputs you can see the sum, difference and the product of the two input numbers. Hint: Use the previous exercise as a starting point. wiki article about rings

## 4/3. Matrix class

Define a class called Matrix for representing matrices.

You have to implement the **__init__** and **__str__** methods. The constructor has one parameter, a list of list of numbers which is the elements of the matrix. The **__str__** should return a multi-line string, containing the matrix in a tabular-like format.

For example:

```
m = Matrix([[1, 2], [13, 4], [5, 6]])
print m
```

should print this:

```
   1   2
  13   4
   5   6
```

The numbers are padded to the right in 4 characters width. There are 3 spaces before each element, except the 13 because there are 2 spaces there.

## 4/4. Matrix operations

Implement the **__add__**, **__sub__**, **__mul__** methods for the previous **Matrix** class.

The matrices will be square shaped, so every operation is compatible.

For example:

```
m1 = Matrix([[1, 2], [3, 4]])
m2 = Matrix([[1, 0], [0, 2]])

print a + b
print a - b
print a * b
```

should print this:

```
   2   2
   3   6

   0   2
   3   2

   1   4
   3   8
```

In the test you can see the sum, difference and the dot product of the two input matrices.

Hint: Use the previous exercise as a starting point. wiki article about rings