Operator overloading

rules

- each operator has a complementary function
- implement the function as a member or extension function
- operator precedence cannot be changed

❗ do not abuse operator overloading - only use it when the behaviour can be deduced intuitively

Operator overloading

overloadable operators

- unary operations

```
+a, -a, !a, a++, a—
```

- binary operations

```
a + b, a - b, a * b
a / b, a % b
a..b, a in b, a !in b
a += b, a -= b, a *= b
a /= b, a %= b
a < b, a > b
a >= b, a <= b
a == b, a != b
```

- invoke operator: `()`

```
a(), a(i), a(i, j),
a(i_1, …, i_n)
```

- indexed access operator: `[]`

```
a[i], a[i_1, …, i_n]
a[i] = b, a[i_1, i_n] = b
```

- property delegation operators *

* will be covered on the ***Delegation*** section of the course