Operator overloading

plus

$$\begin{tikzcd}\boxed{\texttt{a + b}} \arrow[r] & \boxed{\texttt{a.plus(b)}}\end{tikzcd}$$

```
operator fun Point.plus(other: Point): Point {
    return Point(x + other.x, y + other.y)
}
```

```
Point(1, 2) + Point(2, 3)
```

Arithmetic operations

expression	**function name**
`a + b`	`plus`
`a - b`	`minus`
`a * b`	`times`
`a / b`	`div`
`a % b`	`mod`

No restrictions on parameter type

```
operator fun Point.times(scale: Int): Point {
    return Point(x * scale, y * scale)
}
```

```
Point(1, 2) * 3
```

Unary operations

$$\begin{tikzcd} \boxed{\texttt{+a}} \arrow[r] & \boxed{\texttt{a.unaryPlus()}} \end{tikzcd}$$

Unary arithmetic operations

expression	**function name**
`+a`	`unaryPlus`
`-a`	`unaryMinus`
`!a`	`not`
`++a, a++`	`inc`
`--a, a--`	`dec`

Assignment operations

$$\begin{tikzcd} & \boxed{\texttt{a = a.plus(b)}} \\ \boxed{\texttt{a += b}} \arrow[ur] \arrow[r] & \boxed{\texttt{a.plusAssign(b)}} \end{tikzcd}$$

Conventions for lists

```
val list = listOf(1, 2, 3)
val newList = list + 2

val mutableList = mutableListOf(1, 2, 3)
mutableList += 4
```

What will be printed?

```kotlin
val list1 = listOf(1, 2, 3)
var list2 = list1
list2 += 4
println(list1)
println(list2)
```

1.
```
[1, 2, 3]
[1, 2, 3, 4]
```

2.
```
[1, 2, 3, 4]
[1, 2, 3, 4]
```

?

What will be printed?

```kotlin
val list1 = listOf(1, 2, 3)
var list2 = list1
list2 += 4
println(list1)
println(list2)
```

[✓ 1.]
```
[1, 2, 3]
[1, 2, 3, 4]
```

2.
```
[1, 2, 3, 4]
[1, 2, 3, 4]
```

What will be printed?

```
val list1 = listOf(1, 2, 3)
var list2 = list1
list2 += 4
println(list1)
println(list2)
```

1.
```
[1, 2, 3]
[1, 2, 3, 4]
```

2.
```
[1, 2, 3, 4]
[1, 2, 3, 4]
```

Prefer `val` to `var`

```
var list = listOf(1, 2, 3)
list += 4
```

new list is created:

```
list = list + 4
```

Operations on Lists

```kotlin
val list1 = listOf(1, 2, 3)
var list2 = list1
list2 += 4
println(list1)
println(list2)
```

Operations on Lists

```
val list1 = mutableListOf(1, 2, 3)
val list2 = list1
list2 += 4
println(list1)
println(list2)
```

```
[1, 2, 3, 4]
[1, 2, 3, 4]
```