Algoritmisch Denken

Pre- vs postincrement

Onderwerp

`i++` vs `++i`

Wat is het verschil?

Expressies

```
1 + 2
5 * 3 + 2
true || false
[1, 2, 3].slice(1)
```

zijn *expressies*

Expressie Evaluatie

Expressies evalueren tot een waarde

Expressie		Waarde
`1 + 2`	$\longrightarrow$	`3`
`5 * 3 + 2`	$\longrightarrow$	`17`
`true || false`	$\longrightarrow$	`true`

We kunnen deze waarde bv. toekennen aan een variabele

Side Effects

Expressies kunnen ook *side effects* hebben:
ze kunnen een waarde *wijzigen*

Expressie	**Side Effect**
`i = 5`	`i` bevat voortaan `5`
`i += 2`	`i` werd met 2 verhoogd
`[1, 2, 3].pop()`	laatste element verwijderd

Assignment

Expressies met side effects evalueren ook tot een waarde

`x = 5` heeft als side effect dat `x` wordt gelijkgesteld aan `5` maar wat is de waarde van `x = 5`?

```
y = (x = 5)
```

Waar is `y` nu gelijk aan?

Side Effects

$$\texttt{x} \;=\; 5$$

heeft als waarde de waarde van het rechterlid, dus 5

Dit laat toe om te schrijven

$$\texttt{y} \;=\; \texttt{x} \;=\; 5$$

Zowel $\texttt{x}$ als $\texttt{y}$ zijn nu gelijk aan 5

Pre-Increment

`++i`

Side effect: `i` wordt met één verhoogd

Waarde: de *nieuwe* waarde van `i`

Post-Increment

`i++`

Side effect: `i` wordt met één verhoogd

Waarde: de *oude* waarde van `i`

Voorbeeld

```
let a = 0;
let b = ++a;
```

Variabele	Waarde
a	1
b	1

Voorbeeld

```
let a = 0;
let b = a++;
```

Variabele	Waarde
a	1
b	0

Aangeraden Gebruik

Gebruik `i++` of `++i` niet als deel van een andere expressie. Zo gebruik je de waarde niet en is er geen verschil meer tussen beide

Afgeraden

```
x = i++ + ++j;
```

Aangeraden

```
j++; // of ++j
x = i + j;
i++; // of ++i
```