Cheatsheets / **Learn Kotlin**

Conditional Expressions

If Expressions

An `if` expression is a conditional that runs a block of code when its condition has a `true` value.

```
var morning = true

if (morning) {
  println("Rise and shine!")
}
// Prints: Rise and shine!
```

Else Expressions

An `else` expression is a conditional that runs a block of code only when the conditions contained in the previous expressions have `false` values.

```
var rained = false

if (rained) {
  println("No need to water the plants tod
ay.")
} else {
  println("Plants need to be watered!")
}
// Prints: Plants need to be watered!
```

Else-If Expressions

An `else` - `if` expression allows for more conditions to be evaluated within an `if` / `else` expression. You can use multiple `else` - `if` expressions as long as they appear after the `if` expression and before the `else` expression.

```
var age = 65

if (age < 18 ) {
  println("You are considered a minor.")
} else if (age < 60) {
  println("You are considered an adult.")
} else {
  println("You are considered a senior.")
}

// Prints: You are considered a senior.
```

Comparison Operators

Comparison operators are symbols that are used to compare two values in order to return a result of `true` or `false`. Comparison operators include `>`, `<`, `>=`, `<=`.

```
var myAge = 19
var sisterAge = 11
var cousinAge = 11

myAge > sisterAge  // true
myAge < cousinAge  // false
myAge >= cousinAge // true
myAge <= sisterAge // false
```

Logical Operators

Logical operators are symbols used to evaluate the relationship between two or more Boolean expressions in order to return a `true` or `false` value.

Logical operators include `!`, `&&`, and `||`.

```
var humid = true
var raining = true
var jacket = false

println(!humid)
// Prints: false

println(jacket && raining)
// Prints: true

println(humid || raining)
// Prints: true
```

The AND Operator: &&

The logical AND operator (`&&`) is used to compare the relationship between two Boolean expressions and will only return a `true` value if both expressions are `true`.

```
var humid = true
var raining = true
var shorts = false
var sunny = false

// true AND true
println(humid && raining) //  true

// true AND false
println(humid && shorts) //  false

// false AND true
println(sunny && raining) //  false

// false AND false
println(shorts && sunny) // false
```

The OR Operator : ||

The logical OR operator (`||`) is used to compare the relationship between two Boolean expressions and will return `true` when at least one of the expressions are `true` .

```
var late = true
var skipBreakfast = true
var underslept = false
var checkEmails = false


// true OR true
println(skipBreakfast || late) //  true

// true OR false
println(late || checkEmails) //  true

// false OR true
println(underslept || late) //  true

// false OR false
println(checkEmails || underslept) // fals
e
```

The NOT Operator: !

The logical NOT operator (`!`) evaluates the value of a Boolean expression and then returns its negated value.

```
var hungry = true
var full = false

println(!hungry) //  false
println(!full) //  true
```

Order of Evaluation

The order of evaluation when using multiple logical operators in a single Boolean expression is:

1. Expressions placed in parentheses.
2. NOT(`!`) operator.
3. AND(`&&`) operator.
4. OR(`||`) operator.

```
!true && (false || true) // false
/*
(false || true) is evaluated first returni
ng true. Then,
!true && true is evaluated returning the f
inal result, false.
*/

!false && true || false // true
/*
!false is evaluated first returning true.
Then true && true are evaluated, returning
 true. Then, true || false is evaluated wh
ich ends up returning true.
*/
```

Nested Conditionals

A nested conditional is a conditional that exists within another conditional.

```
var studied = true
var wellRested = true

if (wellRested) {
    println("Best of luck today!")
  if (studied) {
    println("You should be prepared for your exam!")
  } else {
    println("Take a few hours to study before your exam!")
  }
}

// Prints: Best of luck today!
// Prints: You should be prepared for your exam!
```

When Expressions

A `when` expression controls the flow of code by evaluating the value of a variable in order to determine what code gets executed.

```
var grade = "A"

when(grade) {
  "A" -> println("Excellent job!")
  "B" -> println("Very well done!")
  "C" -> println("You passed!")
  else -> println("Close! Make sure to perpare more next time!")
}
// Prints: Excellent job!
```

The Range Operator

The range operator (`..`) is used to create a succession of number or character values.

```
var height = 46 // inches

if (height in 1..53) {
  println("Sorry, you must be at least 54 inches to ride the rollercoaster.")
}
// Prints: Sorry, you must be at least 54 inches to ride the rollercoaster.
```

Equality Operators

Equality operators are symbols that are used to compare the equivalence of two values in order to return `true` or `false`. Equality operators include `==` and `!=`.

```
var myAge = 22
var sisterAge = 21

myAge == sisterAge // false
myAge !== sisterAge // true
```