

Computer Science & Engineering 120 Learning to Code

Making Decisions

Christopher M. Bourke
cbourke@cse.unl.edu

Part I: Comparison & Logical Operators

Topic Overview

- ▶ Numeric comparison operators
- ▶ String comparison & lexicographic ordering
- ▶ Logical Operators

Numeric Comparisons I

- ▶ We need a way to compare the value stored in variables
- ▶ Compare the relative value of two variables
- ▶ Compare the value stored in one variable with a fixed value (literal)
- ▶ Comparisons:
 - ▶ Are two values equal or not equal?
 - ▶ Is one value greater than or equal to/lesser than or equal to another?
 - ▶ Is one value strictly greater/lesser than another?

Numeric Comparisons II

- ▶ Standard mathematical expressions:

$= \neq \geq \leq > <$

- ▶ Mathematical symbols are not part of the standard keyboard
- ▶ Instead, we use a combination of regular characters:

`===` `!==` `>=` `<=` `>` `<`

Numeric Comparisons

Equality & Inequality

- ▶ Use *three* equals signs, `===` to compare for equality
- ▶ Use `!==` to compare for inequality

```
1 var x = 10, y = 20, z = 10, r;  
2  
3 r = (x === y); //false  
4 r = (x === z); //true  
5 r = (x !== y); //true  
6 r = (x !== z); //false  
7  
8 r = (x === 10); //true  
9 r = (y === 10); //false  
10 r = (x !== 30); //true  
11  
12 r = (x+y === 30); //true  
13 r = (x+z === y); //true
```

Numeric Comparisons

Inequality Operators

Inequality operators: strict or non-strict; order of operands matters

```
1 var x = 10, y = 20, z = 10, r;  
2  
3 r = (x < y); //true  
4 r = (x < z); //false  
5 r = (x <= y); //true  
6 r = (x <= z); //true  
7  
8 r = (x > y); //false  
9 r = (x > z); //false  
10 r = (x >= y); //false  
11 r = (x >= z); //true  
12  
13 r = (x < 10); //false  
14 r = (x <= 10); //true  
15 r = (y < 10); //false  
16 r = (x+z < 20); //false  
17 r = (x+z <= 20); //true
```

String Comparisons

Lexicographic Ordering

- ▶ The same comparison operators can be used with strings
- ▶ The relative ordering of strings is *lexicographic ordering*
- ▶ Follows the ASCII text table
<https://en.wikipedia.org/wiki/ASCII>
- ▶ Numbers and letters in *alphanumeric* order
- ▶ Upper case letters before lower case
- ▶ Numbers before letters

String Comparisons

Examples

```
1 var a = "Apple", b = "apple", c = "zebra", r;  
2  
3 r = (a === b); //false  
4 r = (a === "Apple"); //true  
5 r = (a < b); //true  
6 r = (a <= "Apple"); //true  
7 r = (c < b); //false  
8 r = (b < "apples"); //true  
9 r = ("52 Apples" < a); //true
```

Logical Operators

- ▶ Need a way to combine comparisons into more complex statements
- ▶ A way to check a *range* of values
- ▶ A way to check if *either* condition A *or* condition B holds
- ▶ A way to check if condition A *and* condition B hold

Logical Operators

Negation Operator

- ▶ We can negate any statement by applying the negation operator
- ▶ Operator is an exclamation point (similar to `!==`)

```
1 var a = 10, b = true, r;  
2 r = !(a < 20); //equivalent to (a >= 20)  
3 r = !b; //false
```

Logical Operators

And Operator

- ▶ Syntax: use two ampersands, `&&`, applies to two operands
- ▶ Evaluates to true if and only if *both* operands evaluate to true

```
1 var a = 10, b = true, r;  
2 r = (a >= 0 && a <= 10); //true  
3 r = (a === 10 && b); //true  
4 r = (a !== 10 && b); //false  
5 r = (a === 10 && !b); //false
```

Logical Operators

Or Operator

- ▶ Syntax: use two vertical bars, `||`, applies to two operands
- ▶ Evaluates to true if *either* operand is true (or if *both* are)

```
1 var a = 10, b = true, r;  
2  
3 r = (a === 10 || b); //true  
4 r = (a !== 10 || b); //true  
5 r = (a < 0 || a > 10); //false  
6 r = (a === 10 || !b); //true  
7 r = (a !== 10 || !b); //false
```

Logical Operators

Precedence

- ▶ In arithmetic, multiplication/division is done before addition/subtraction
- ▶ In logic, similar order:
 1. Negation `!`
 2. And `&&`
 3. Or `||`
- ▶ The following are *not* equivalent:
`a && (b || c)`
`a && b || c`

Part II: Conditional Statements

Topic Overview

- ▶ Conditional `if`-statement
- ▶ Conditional `if-else`-statement
- ▶ Conditional `if-else-if`-statement
- ▶ Nesting

If Statement

- ▶ Code can execute (or not) based on some *condition*
- ▶ If the condition holds, the code executes
- ▶ If the condition does not hold, the code does not execute
- ▶ Syntax: keyword `if`, condition is placed inside parentheses
- ▶ Conditioned *code block* is encapsulated in brackets

If Statement

```
1 if(a > 0) {  
2   console.log("a is positive!");  
3 }
```

If-Else Statement

- ▶ Based on some *condition*, we could execute one piece of code or another piece of code
- ▶ If the condition holds, Code Block A executes
- ▶ If the condition does not hold, Code Block B executes
- ▶ *Exactly* one and only one of these executes
- ▶ Syntax: keywords `if` and `else`

If-Else Statement

```
1 if(huskersScore < miamiScore) {
2   console.log("Huskers lose :(");
3 } else {
4   console.log("Huskers win!");
5 }
```

If-Else-If Statement

- ▶ Can generalize to more than one condition
- ▶ Syntax: keyword `else if`
- ▶ The first condition that evaluates to true is the *only* one that executes
- ▶ The final `else` block is optional (just as it was with an `if` statement)

If-Else-If Statement

```
1 if(huskerScore < miamiScore) {
2   console.log("hurricanes win!");
3 } else if(miamiScore < huskerScore) {
4   console.log("huskers win!");
5 } else {
6   console.log("lets go to overtime");
7 }
```

Nesting Statements

- ▶ Can *nest* statements within other statements

```
1 if(a > 0) {
2   if(a % 2 === 0) {
3     console.log("a is a positive even number");
4   } else {
5     console.log("a is a positive odd number");
6   }
7 } else {
8   console.log("a is not positive");
9 }
```

Exercise I

Sound loudness is measured in decibels. Write code to output a characterization of a sound based on the levels in the following table.

Decibel	Characterization
$d \leq 50$	quiet
$50 < d \leq 70$	intrusive
$70 < d \leq 90$	annoying
$90 < d \leq 110$	very annoying
$110 < d \leq 130$	medical threat
$d > 130$	uncomfortable

Exercise II

```
1 if(decibel <= 50) {
2   console.log("quiet");
3 } else if(decibel > 50 && decibel <= 70) {
4   console.log("intrusive");
5 } else if(decibel <=90) {
6   console.log("annoying");
7 } else if(decibel <= 110) {
8   console.log("very annoying");
9 } else if(decibel <= 130) {
10  console.log("medical threat");
11 } else {
12  console.log("uncomfortable");
13 }
```