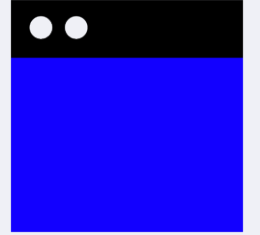




**Code
Academy**



Gytis Juozenas

JavaScript duomenų tipai, operatoriai, sąlyginiai sakiniai

2022

JavaScript programavimo kalba




„JavaScript“ operatoriai

Paskyrimo operatoriai (Assignment operators)

Sudėtiniai paskyrimų operatoriai (Compound assignment operators)

Daugiau apie operatorius [čia](#)

Name	Shorthand operator	Meaning
Assignment	<code>x = y</code>	<code>x = y</code>
Addition assignment	<code>x += y</code>	<code>x = x + y</code>
Subtraction assignment	<code>x -= y</code>	<code>x = x - y</code>
Multiplication assignment	<code>x *= y</code>	<code>x = x * y</code>
Division assignment	<code>x /= y</code>	<code>x = x / y</code>
Remainder assignment	<code>x %= y</code>	<code>x = x % y</code>
Exponentiation assignment 	<code>x **= y</code>	<code>x = x ** y</code>
Left shift assignment	<code>x <<= y</code>	<code>x = x << y</code>
Right shift assignment	<code>x >>= y</code>	<code>x = x >> y</code>
Unsigned right shift assignment	<code>x >>>= y</code>	<code>x = x >>> y</code>
Bitwise AND assignment	<code>x &= y</code>	<code>x = x & y</code>
Bitwise XOR assignment	<code>x ^= y</code>	<code>x = x ^ y</code>
Bitwise OR assignment	<code>x = y</code>	<code>x = x y</code>



“JavaScript” operatoriai


Palyginimo operatoriai (Comparison operators)

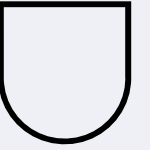
Comparison operators		
Operator	Description	Examples returning true
Equal (==)	Returns <code>true</code> if the operands are equal.	<code>3 == var1</code> <code>"3" == var1</code> <code>3 == '3'</code>
Not equal (!=)	Returns <code>true</code> if the operands are not equal.	<code>var1 != 4</code> <code>var2 != "3"</code>
Strict equal (===)	Returns <code>true</code> if the operands are equal and of the same type. See also Object.is and sameness in JS .	<code>3 === var1</code>
Strict not equal (!==)	Returns <code>true</code> if the operands are of the same type but not equal, or are of different type.	<code>var1 !== "3"</code> <code>3 !== '3'</code>
Greater than (>)	Returns <code>true</code> if the left operand is greater than the right operand.	<code>var2 > var1</code> <code>"12" > 2</code>
Greater than or equal (>=)	Returns <code>true</code> if the left operand is greater than or equal to the right operand.	<code>var2 >= var1</code> <code>var1 >= 3</code>
Less than (<)	Returns <code>true</code> if the left operand is less than the right operand.	<code>var1 < var2</code> <code>"2" < 12</code>
Less than or equal (<=)	Returns <code>true</code> if the left operand is less than or equal to the right operand.	<code>var1 <= var2</code> <code>var2 <= 5</code>



“JavaScript” operatoriai

Aritmetiniai operatoriai (Arithmetic operators)

Operator	Description	Example
Remainder (%)	Binary operator. Returns the integer remainder of dividing the two operands.	12 % 5 returns 2.
Increment (++)	Unary operator. Adds one to its operand. If used as a prefix operator (++x), returns the value of its operand after adding one; if used as a postfix operator (x++), returns the value of its operand before adding one.	If x is 3, then ++x sets x to 4 and returns 4, whereas x++ returns 3 and, only then, sets x to 4.
Decrement (-)	Unary operator. Subtracts one from its operand. The return value is analogous to that for the increment operator.	If x is 3, then --x sets x to 2 and returns 2, whereas x-- returns 3 and, only then, sets x to 2.
Unary negation (-)	Unary operator. Returns the negation of its operand.	If x is 3, then -x returns -3.
Unary plus (+)	Unary operator. Attempts to convert the operand to a number, if it is not already.	+ "3" returns 3. + true returns 1.
Exponentiation operator (**) 	Calculates the base to the exponent power, that is, base ^{exponent}	2 ** 3 returns 8. 10 ** -1 returns 0.1.



Primityvūs JavaScript duomenų tipai (teorija)

Boolean - reiškia loginį subjektą ir gali turėti dvi reikšmes: `true` arba `false`.

Šis tipas dažniausiai naudojamas norint išsaugoti „taip“ / „ne“ reikšmes.

Pvz.:

```
let nameFieldChecked = true;
```

```
let ageFieldChecked = false;
```

Boolean vertės taip pat gaunamos palyginus:

```
let isGreater = 4 > 1;
```

```
console.log( isGreater ); // true.
```



Primityvūs JavaScript duomenų tipai (teorija)

Boolean - reiškia loginį subjektą ir gali turėti dvi reikšmes: **true** arba **false**.

“Falsy” reikšmė yra vertė, kuri laikoma *false*, Boolean kontekste.

Yra 8 “falsy” vertės:

false	The keyword false
0	The number zero
-0	The number negative zero
0n	BigInt , when used as a boolean, follows the same rule as a Number. 0n is <i>falsy</i> .
""	Empty string value
null	null - the absence of any value
undefined	undefined - the primitive value
NaN	NaN - not a number

”Truthy” values yra viskas nėra aprašyti falsy dalyje.



“JavaScript” operatoriai

Loginiai operatoriai (Logical operators)

Loginiai operatoriai paprastai naudojami su loginėmis (Boolean) reikšmėmis t.y. Jie grąžina loginę reikšmę.

Loginiai operatoriai aprašyti šioje lentelėje:

Operator	Usage	Description
Logical AND (<code>&&</code>)	<code>expr1 && expr2</code>	Returns <code>expr1</code> if it can be converted to <code>false</code> ; otherwise, returns <code>expr2</code> . Thus, when used with Boolean values, <code>&&</code> returns <code>true</code> if both operands are true; otherwise, returns <code>false</code> .
Logical OR (<code> </code>)	<code>expr1 expr2</code>	Returns <code>expr1</code> if it can be converted to <code>true</code> ; otherwise, returns <code>expr2</code> . Thus, when used with Boolean values, <code> </code> returns <code>true</code> if either operand is true; if both are false, returns <code>false</code> .
Logical NOT (<code>!</code>)	<code>!expr</code>	Returns <code>false</code> if its single operand that can be converted to <code>true</code> ; otherwise, returns <code>true</code> .



“JavaScript” operatoriai

Loginiai operatoriai (Logical operators)

&& (loginio IR) operatoriaus pavyzdžiai:

```
var a1 = true && true;    // t && t bus true  
var a2 = true && false;   // t && f bus false  
var a3 = false && true;    // f && t bus false  
var a4 = false && (3 == 4); // f && f bus false
```




“JavaScript” operatoriai

Loginiai operatoriai (Logical operators)

II (loginio ARBA) operatoriaus pavyzdžiai:

```
var o1 = true || true; // t || t bus true  
var o2 = false || true; // f || t bus true  
var o3 = true || false; // t || f bus true  
var o4 = false || (3 == 4); // f || f bus false
```



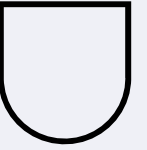
“JavaScript” operatoriai

Loginiai operatoriai (Logical operators)

! (loginio NE) operatoriaus pavyzdžiai:

```
var n1 = !true; // !t bus false
```

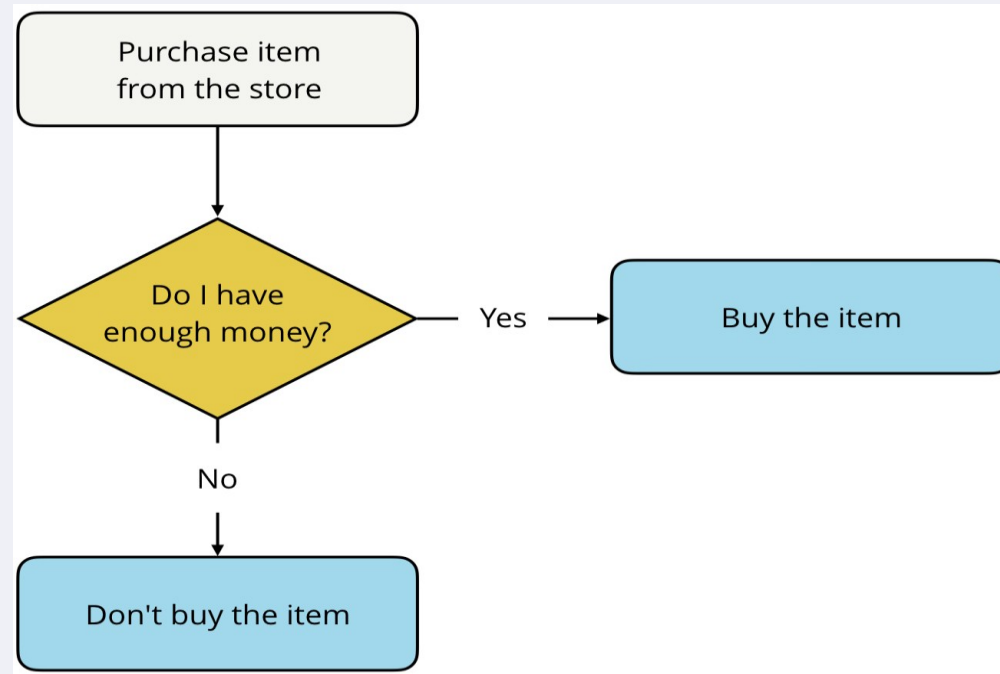
```
var n2 = !false; // !f bus true
```



JavaScript sąlygos sakiniai (conditionals)

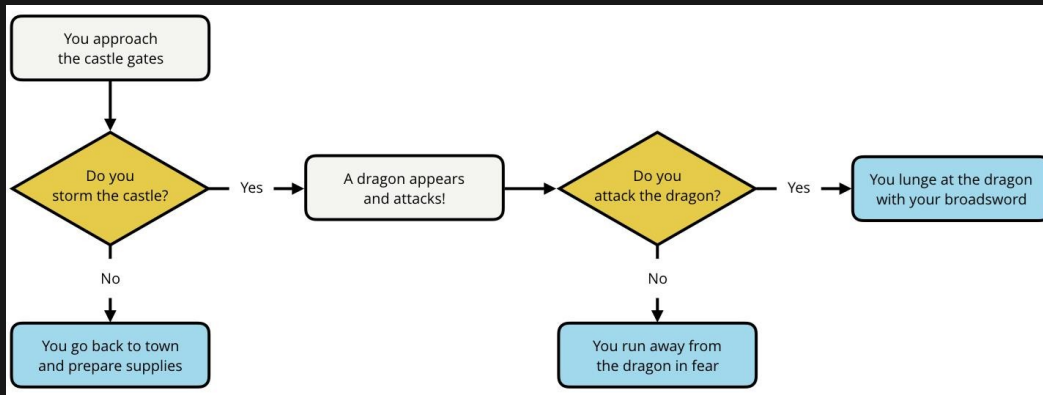
Pažvelkite į dešinėje pateiktą schemą. Koks duomenų tipas geriausiai atspindėtų (Yes / No), jei turite pakankamai pinigų daiktui įsigyti?

- A – Number
- B – String
- C – Boolean
- D – Undefined





JavaScript sąlygos sakiniai (conditionals)



Čia yra kita schema, kaip privažiuoti prie pilies vartų. Jei nuspręsite šturmuoti pilį, koks yra tiesioginis rezultatas?

A – “You go back to town and prepare supplies” B

– “A dragon appears and attacks!”

C – “You run away from the dragon in fear”



JavaScript sąlygos sakiniai (conditionals)

if...else statements

If...else leidžia vykdyti tam tikrus kodo fragmentus, atsižvelgiant į įvykdytą sąlygą ar sąlygų rinkinį.

Sintaksė:

```
if (/^ this expression is true ^/) {  
    // run this code  
} else {  
    // run this code  
}
```



JavaScript sąlygos sakiniai (conditionals)

if...else statements

If...else nepaprastai naudinga, nes leidžia jums pasirinkti, kurį kodo fragmentą norite vykdyti pagal išraiškos rezultatą.

Pvz.,

```
let a = 1;
```

```
let b = 2;
```

```
if (a >= b) {
```

```
  console.log("a yra daugiau negu b");
```

```
} else {
```

```
  console.log("a yra mažiau arba lygu b");
```

```
}
```



JavaScript sąlygos sakiniai (conditionals)

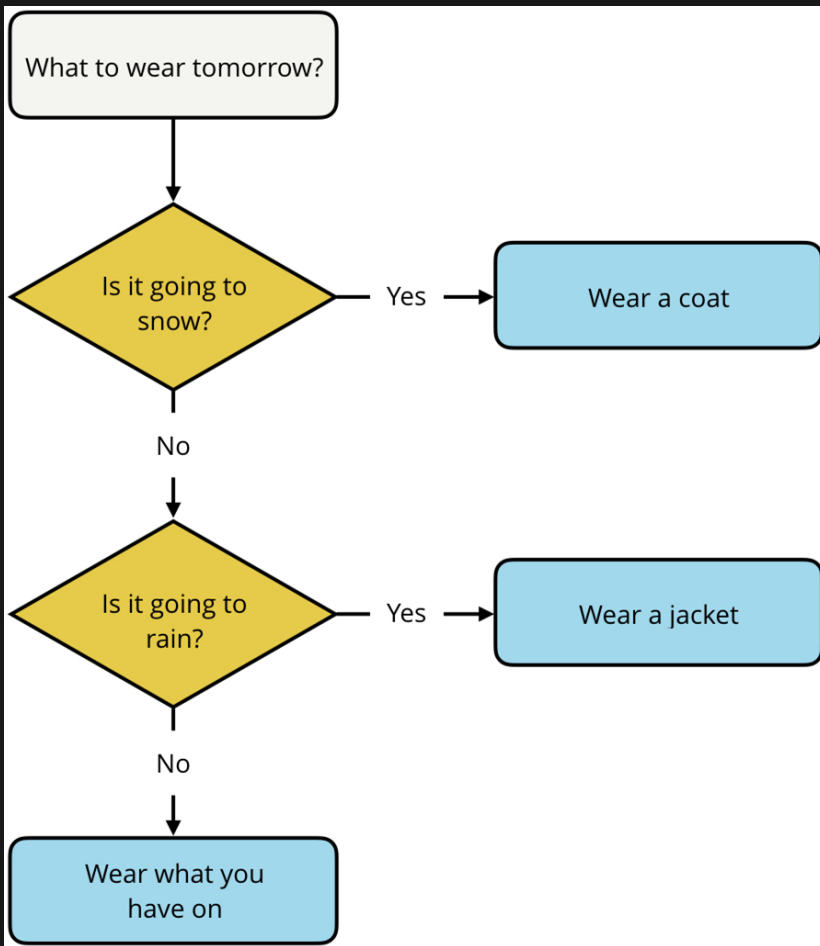
if...else statements

Keletas svarbių dalykų apie if...else

“if” sakinyje esanti vertė visada paverčiama *true* arba *false*. Priklausomai nuo vertės, bus vykdomas kodas, esantis “if” sakinio viduje, arba kodas, esantis “else” sakinio viduje, bet ne abu.

Kodas, esantis teiginių “if” ir “else” viduje, yra apsuptas {...}, kad būtų galima atskirti sąlygas ir nurodyti, kurį kodą reikia naudoti.

“If” gali būti be “else”, tačiau “else” be “if” ne.



JavaScript sąlygos sakiniai (conditionals)

Kai kuriose situacijose, dviejų sąlygų nepakanka.
Pažiūrėkime į kairėje esančią situaciją.



JavaScript sąlygos sakiniai (conditionals)

else if statement

JavaScript kalboje galite pavaizduoti antrinę patikrinimą naudodami papildomą teiginį, vadinamą **else if statement**.

```
let weather = "sunny";

if (weather === "snow")
  { console.log("Bring a
coat.");
} else if (weather === "rain")
  { console.log("Bring a rain
jacket.");
} else {
  console.log("Wear what you have on.");
}
```



Užduotis nr. 1

Kas bus atvaizduojama consolė'je?

```
let money = 100.50;
```

```
let price = 105.50;
```

```
if (money > price) {
```

```
  console.log("You paid extra, here's your change."); // A
```

```
  } else if (money === price) {
```

```
    console.log("You paid the exact amount, have a nice day!"); // B
```

```
  } else {
```

```
    console.log("That's not enough, you still owe me money."); // C
```

```
  }
```



Užduotis nr. 2

Sukurkite kodą kuris naudodamasis if else kondicija pagal iš anksto nustatytą bėgiko (naudojant let) poziciją konsolėje išmestų bėgiko vardą ir kokį medalį jis gavo (pvz.: Jonas gavo sidarbinį medalį). Jums reikės sukurti 3 kintamuosius ir du iš jų nustatyti iš anksto:

1. Bėgiko vardas (aprašyti iš anksto reiktų);
2. Pozicija kurią užėmė bėgikas (aprašyti iš anksto reiktų);
3. Ir neaprašytą kintamąjį kuris bus tiesiog medaliui nustatyti, pagal kurį tikrinsim kokį gaus, pagal užimtą poziciją.



Užduotis nr. 2

SPRENDIMAS:

```
let runner = "Kendyll";
let position = "4";
let medal;
if (position === 1) {
    medal = "gold";
} else if (position === 2) {
    medal = "silver";
} else if (position === 3) {
    medal = "bronze";
} else if (position === 4) {
    medal = "wood";
} else {
    medal = "no";
}
console.log(runner + " received " + medal + " medal.");
```