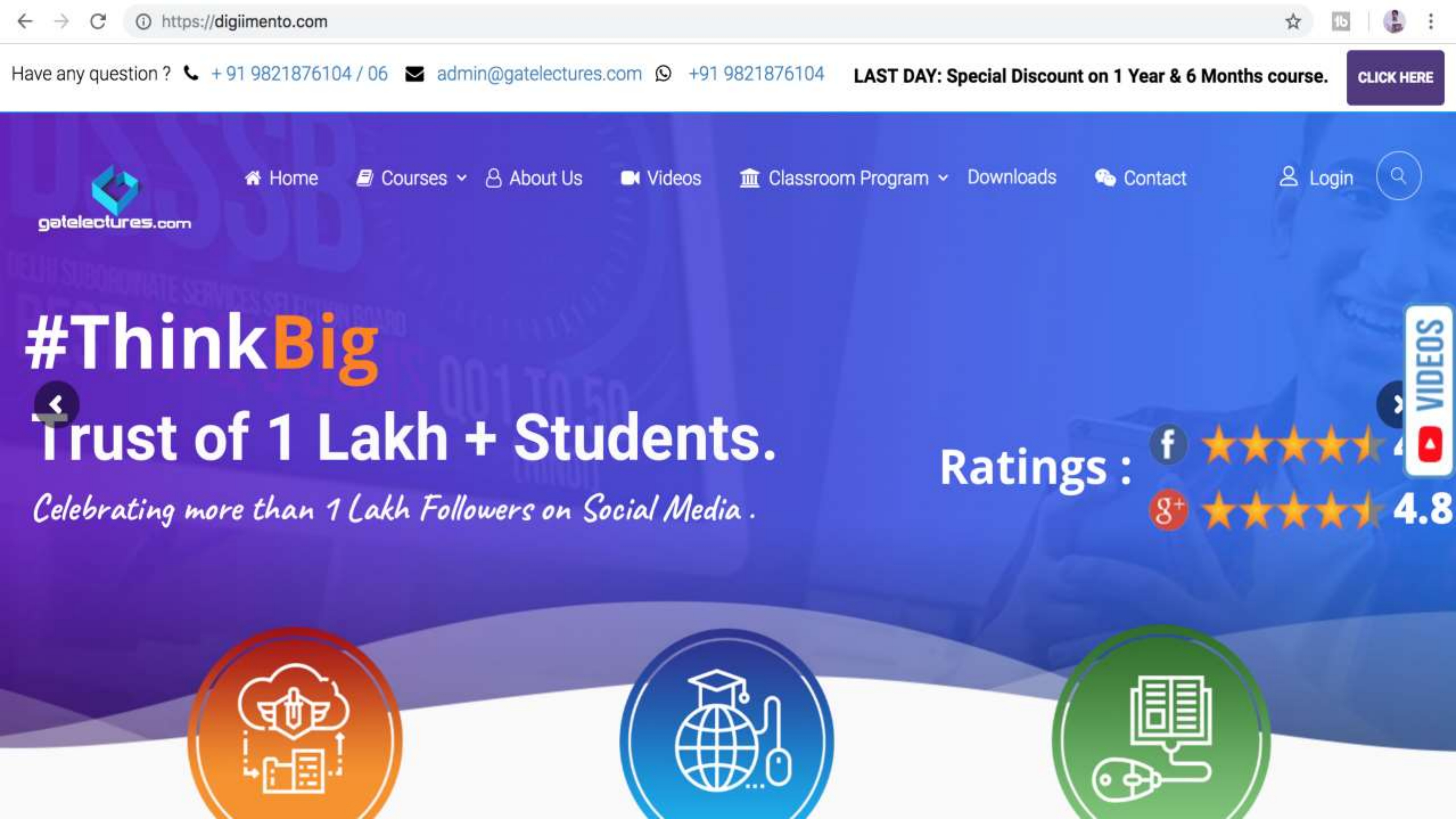
digimento

EDUCATING ON GO...

JAVA SCRIPT

The unauthorized copying, sharing or distribution of copyrighted material is strictly prohibited.



https://digiimento.com

Have any question ? + 91 9821876104 / 06 admin@gatelectures.com +91 9821876104 **LAST DAY: Special Discount on 1 Year & 6 Months course.** CLICK HERE

#ThinkBig

Trust of 1 Lakh + Students.

Celebrating more than 1 Lakh Followers on Social Media .

Ratings : 4 4.8

Comparisons

- Many comparison operators we know from maths:
- Greater/less than: a > b, a < b.
- Greater/less than or equals: a >= b, a <= b.
- Equality check is written as a == b (please note the double equation sign =. A single symbol a = b would mean an assignment).
- Not equals. In maths the notation is ≠, in JavaScript it's written as an assignment with an exclamation sign before it: a != b.

Get Access to Complete Video Lectures › Call : WhatsApp 9821876104/02 › email us at: admin@gatelectures.com

Boolean is the result

- Just as all other operators, a comparison returns a value. The value is of the boolean type.
- true – means “yes”, “correct” or “the truth”.
- false – means “no”, “wrong” or “a lie”.

Get Access to Complete Video Lectures Call : WhatsApp 9821876104/02 email us at: admin@gatelectures.com

For example:

```
alert( 2 > 1 );  // true (correct)
alert( 2 == 1 ); // false (wrong)
alert( 2 != 1 ); // true (correct)
```

A comparison result can be assigned to a variable, just like any value:

```
let result = 5 > 4; // assign the result of the comparison
alert( result ); // true
```

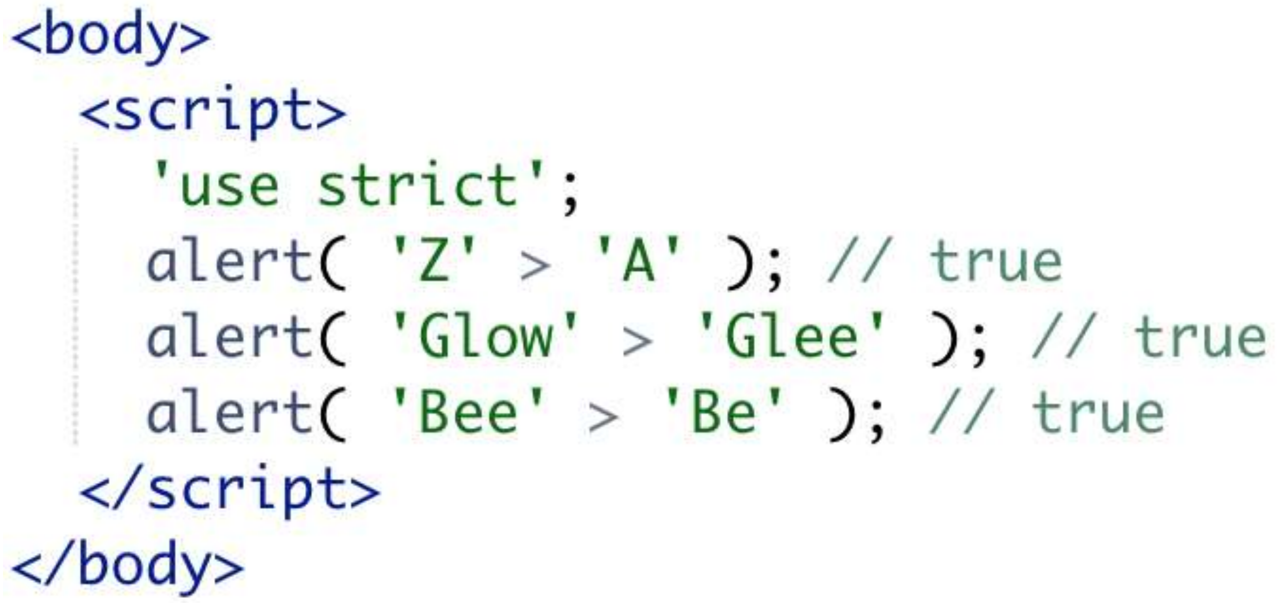
String comparison

- **To see which string is greater than the other, the so-called “dictionary” or “lexicographical” order is used.**
- **In other words, strings are compared letter-by-letter.**

The algorithm to compare two strings is simple:

1. Compare first characters of both strings.
2. If the first one is greater(or less), then the first string is greater(or less) than the second. We’re done.
3. Otherwise if first characters are equal, compare the second characters the same way.
4. Repeat until the end of any string.
5. If both strings ended simultaneously, then they are equal. Otherwise the longer string is greater.

- In the example, the comparison 'Z' > 'A' gets the result at the first step.
- Strings "Glow" and "Glee" are compared character-by-character:
- G is the same as G.
- l is the same as l.
- o is greater than e. Stop here. The first string is greater.



```
<body>
  <script>
    'use strict';
    alert( 'Z' > 'A' ); // true
    alert( 'Glow' > 'Glee' ); // true
    alert( 'Bee' > 'Be' ); // true
  </script>
</body>
```

- The comparison algorithm given is roughly equivalent to the one used in book dictionaries or phone books. But it’s not exactly the same.
- For instance, case matters. A capital letter "A" is not equal to the lowercase "a". Which one is greater? Actually, the lowercase "a" is. Why? Because the lowercase character has a greater index in the internal encoding table (Unicode).

Get Access to Complete Video Lectures Call : WhatsApp 9821876104/02 email us at: admin@gatelectures.com

When compared values belong to different types, they are converted to numbers.

For example:

```
alert( '2' > 1 ); // true, string '2' becomes a number 2
alert( '01' == 1 ); // true, string '01' becomes a number 1
```

For boolean values, `true` becomes `1` and `false` becomes `0`, that's why:

```
alert( true == 1 ); // true
alert( false == 0 ); // true
```

consequence

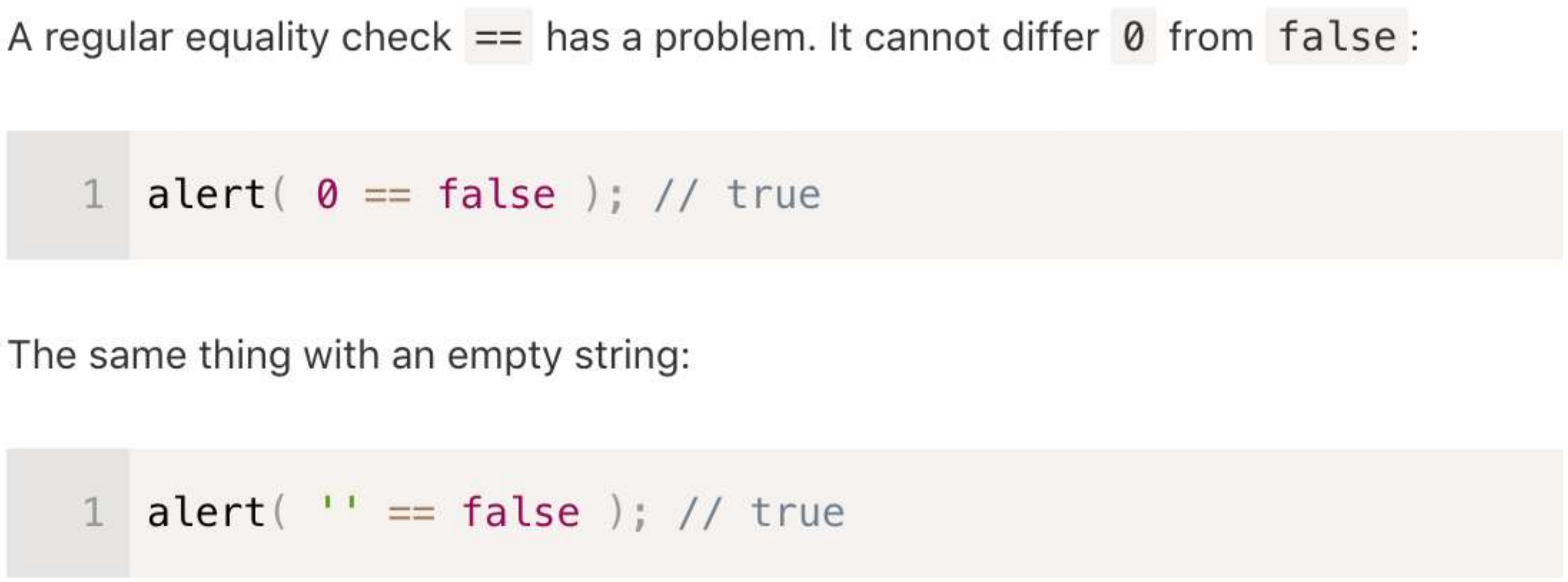
It is possible that at the same time:

- Two values are equal.
- One of them is `true` as a boolean and the other one is `false` as a boolean.

For example:

```
let a = 0;
alert( Boolean(a) ); // false

let b = "0";
alert( Boolean(b) ); // true

alert(a == b); // true!
```

Strict equality



A regular equality check `==` has a problem. It cannot differ `0` from `false`:

```
alert( 0 == false ); // true
```

The same thing with an empty string:

```
alert( '' == false ); // true
```

- That’s because operands of different types are converted to a number by the equality operator ==. An empty string, just like false, becomes a zero.

Get Access to Complete Video Lectures Call : WhatsApp 9821876104/02 email us at: admin@gatelectures.com

What to do if we'd like to differentiate 0 from false?

- In other words, if a and b are of different types, then a === b immediately returns false without an attempt to convert them.
- There also exists a "strict non-equality" operator !==, as an analogy for !=.
- The strict equality check operator is a bit longer to write, but makes it obvious what's going on and leaves less space for errors.

```
<html>

<body>
  <script>
    'use strict';
    alert( 0 === false );
    // false, because the types are different
  </script>
</body>

</html>
```

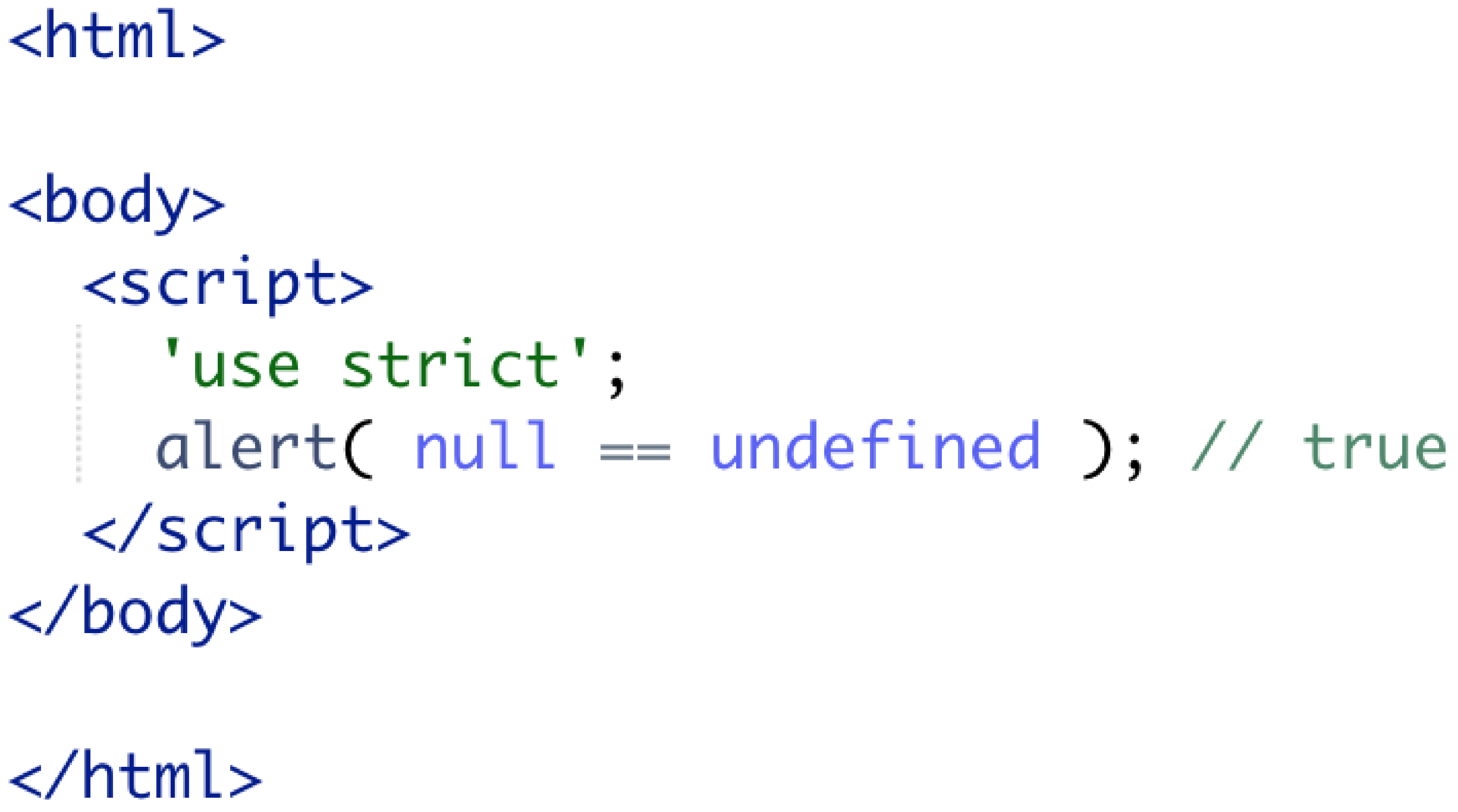
Comparison with null and undefined

- For a strict equality check ===These values are different, because each of them belongs to a separate type of its own.

```
<html>


<body>
  <script>
    'use strict';
    alert( null === undefined ); // false
  </script>
</body>


</html>
```

digimento EDUCATING ON GO... JAVASCRIPT gatelectures.com

- For a non-strict check ==There’s a special rule. These two are a “sweet couple”: they equal each other (in the sense of ==), but not any other value.

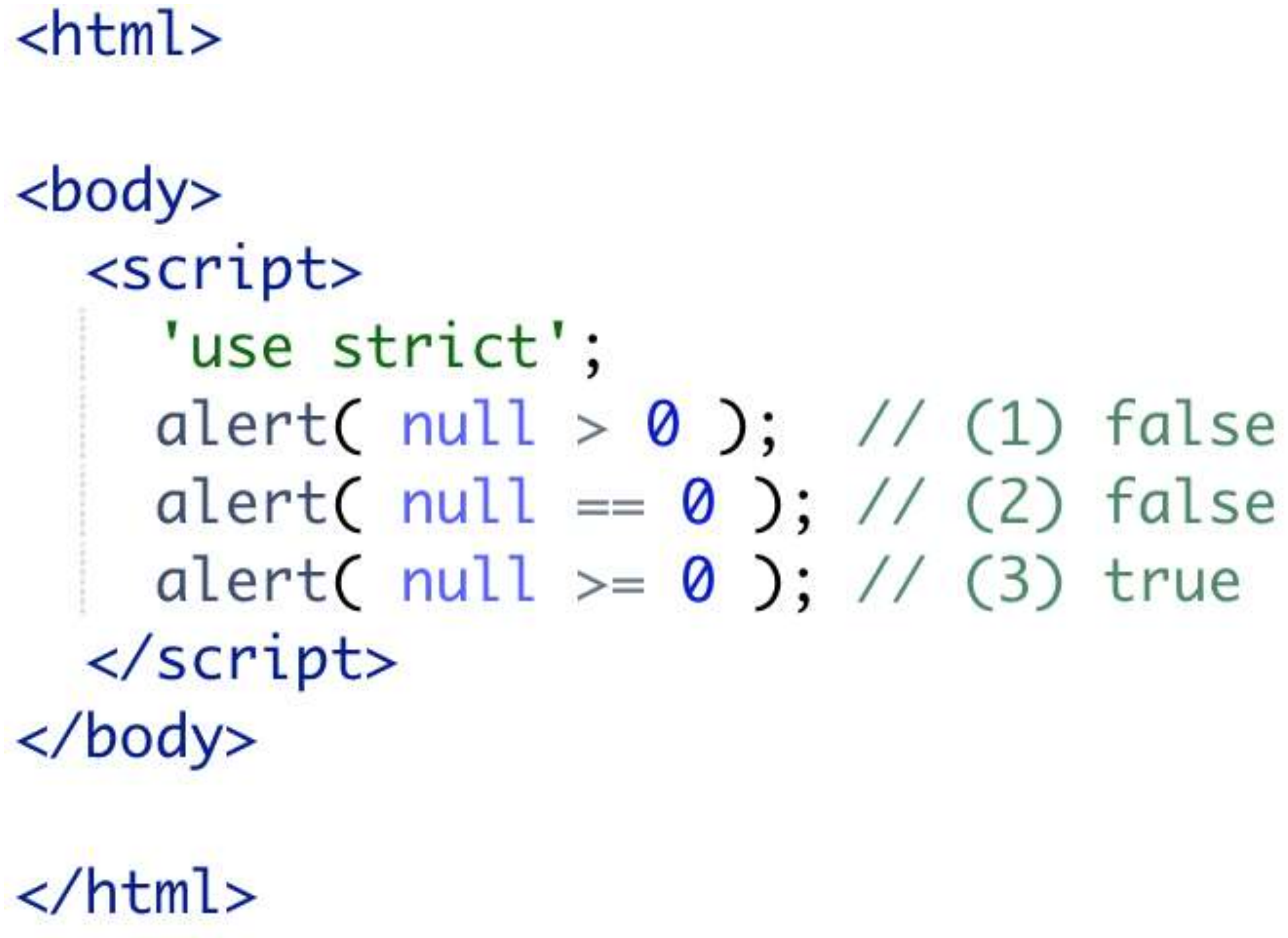


```
<html>


<body>
  <script>
    'use strict';
    alert( null == undefined ); // true
  </script>
</body>


</html>
```

Get Access to Complete Video Lectures Call : WhatsApp 9821876104/02 email us at: admin@gatelectures.com

digimento EDUCATING ON GO...
gatelectures.com

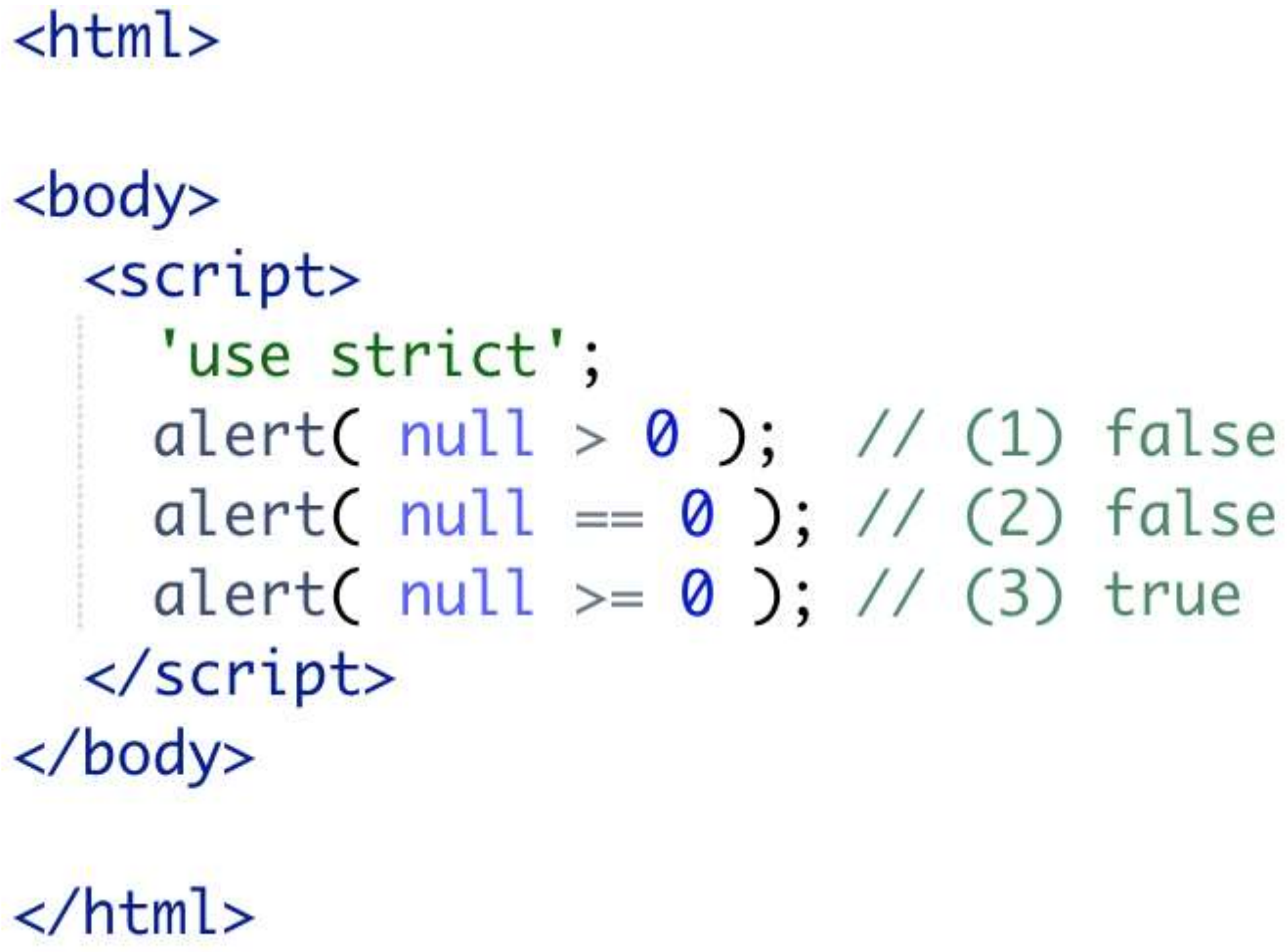
- For maths and other comparisons < > <= >=Values null/undefined are converted to a number: null becomes 0, while undefined becomes NaN.



```
<html>


<body>
  <script>
    'use strict';
    alert( null > 0 );  // (1) false
    alert( null == 0 ); // (2) false
    alert( null >= 0 ); // (3) true
  </script>
</body>


</html>
```

Get Access to Complete Video Lectures › Call : WhatsApp 9821876104/02 › email us at: admin@gatelectures.com

- The reason is that an equality check == and comparisons > < >= <= work differently. Comparisons convert null to a number, hence treat it as 0. That’s why (3) null >= 0 is true and (1) null > 0 is false.
- On the other hand, the equality check == for undefined and null i s defined such that, without any conversions, they equal each other and don’t equal anything else. That’s why (2) null == 0 is false.

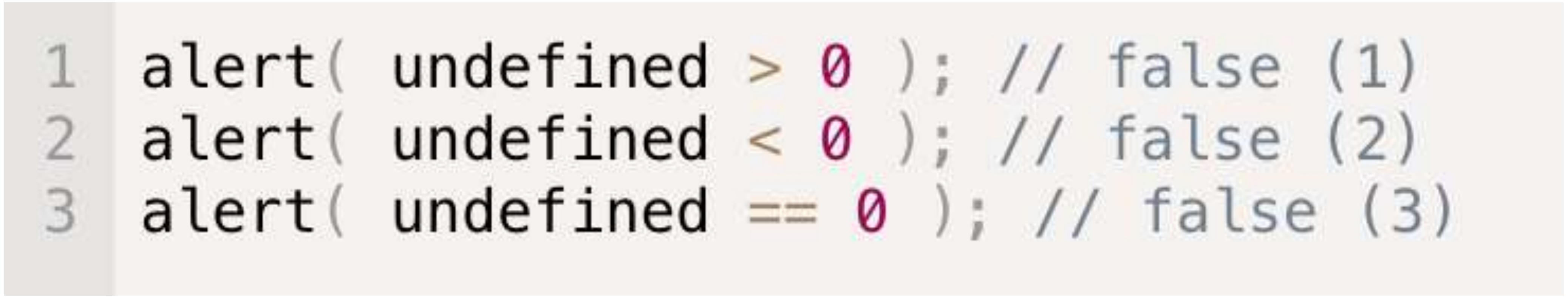


```
<html>

<body>
  <script>
    'use strict';
    alert( null > 0 );  // (1) false
    alert( null == 0 ); // (2) false
    alert( null >= 0 ); // (3) true
  </script>
</body>

</html>
```

Get Access to Complete Video Lectures > Call : WhatsApp 9821876104/02 > email us at: admin@gatelectures.com

An incomparable undefined

- Comparisons (1) and (2) return false because undefined gets converted to NaN. And NaN is a special numeric value which returns false for all comparisons.
- The equality check (3) returns false, because undefined only equals null and no other value.



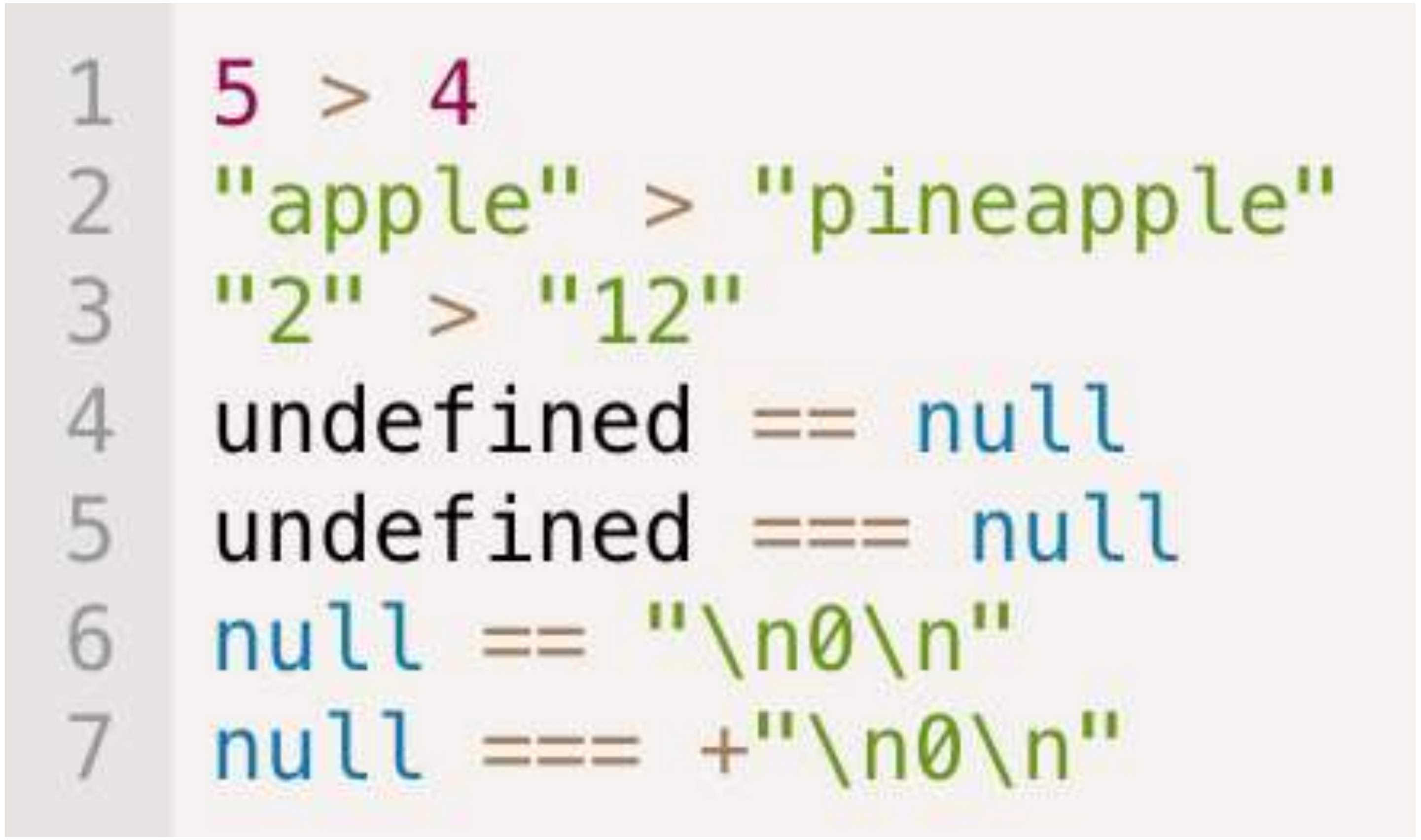
```
alert( undefined > 0 ); // false (1)
alert( undefined < 0 ); // false (2)
alert( undefined == 0 ); // false (3)
```

Get Access to Complete Video Lectures Call : WhatsApp 9821876104/02 email us at: admin@gatelectures.com

Summary

- Comparison operators return a logical value.
- Strings are compared letter-by-letter in the “dictionary” order.
- When values of different types are compared, they get converted to numbers (with the exclusion of a strict equality check).
- Values null and undefined equal == each other and do not equal any other value.
- Be careful when using comparisons like > or < with variables that can occasionally be null/undefined. Making a separate check for null/undefined is a good idea.

Get Access to Complete Video Lectures Call : WhatsApp 9821876104/02 email us at: admin@gatelectures.com



```
5 > 4
"apple" > "pineapple"
"2" > "12"
undefined == null
undefined === null
null == "\n0\n"
null === +"\n0\n"
```

Get Access to Complete Video Lectures › Call : WhatsApp 9821876104/02 › email us at: admin@gatelectures.com

```
1 5 > 4 → true
2 "apple" > "pineapple" → false
3 "2" > "12" → true
4 undefined == null → true
5 undefined === null → false
6 null == "\n0\n" → false
7 null === +"\n0\n" → false
```

Some of the reasons:

1. Obviously, true.
2. Dictionary comparison, hence false.
3. Again, dictionary comparison, first char of `"2"` is greater than the first char of `"1"`.
4. Values `null` and `undefined` equal each other only.
5. Strict equality is strict. Different types from both sides lead to false.
6. See (4).
7. Strict equality of different types.

Get Access to Complete Video Lectures Call : WhatsApp **9821876104/02** email us at: **admin@gatelectures.com**

- ✓ console.log("20" > 10); //true, String "20" is converted to Number 20
- ✓ console.log('01' == 1); // true, String '01' is converted to Number 01
- ✓ console.log(null < 10); //true, because null is converted to 0 in Comparison Operators, Hence 0 < 10 is true
- ✓ console.log(null > 0); //false, here null is converted to 0 in Comparison Operators, Hence 0 > 0 is false
- ✓ console.log(undefined <= 0); //fasle, undefined is converted to NaN in Comparison operators hence Nan <= 0 is false
- ✓ console.log(undefined == 0); //false, undefined is not converted to NaN in double Equality Comparison operators, it remains the same. Hence Undefined == 0 is false
- ✓ console.log(undefined == 1); //false, undefined is not converted to NaN in double Equality Comparison operators. Hence undefined == 1 is false
- ✓ console.log(null == 0); //false, null is not converted to 0 when double equality operator is used hence null == 0 is false
- ✓ console.log(null == undefined); //true Sweet Couple
- ✓ console.log(true == 1); //true, true is converted to a number 1,
- ✓ console.log(false == 0); //true, false is converted to zero when using comparison operators
- ✓ console.log(0 == false); //true, false is converted to zero
- ✓ console.log('' == false); //true, false is converted to zero and empty string is converted to zero always in JS
- ✓ console.log(0 === false); //false, types are different

Get Access to Complete Video Lectures Call : WhatsApp 9821876104/02 email us at: admin@gatelectures.com