EJERCICIOS DE EXPRESIONES 3 *(COMPARACIONES DE IGUALDAD)*

Igualdad estricta ===

1. Dadas las siguiente expresiónes en JavaScript, para cada una aplicando las reglas de orden de agrupación (precedencia y asociatividad) y orden de evaluación:
   - pon paréntesis de forma que, sin modificar su funcionalidad, haga claro en qué orden se evalúa.
   - indica el valor resultante de la expresión y de las variables, si las hay

```
a) true === true

b) true === Boolean(true)

c) true === new Boolean(true)

d) Boolean(true) ===
     new Boolean(true)

e) 0n === -0n

f) 0 === 0n
```

```
g) '0' === 0;

h) undefined === true

i) NaN === NaN;

j) NaN !== NaN

k) -0 === 0

l) undefined === undefined

m) -Infinity === -3/0
```

```
n) let a = {};
   let b = {};
   if (a === b)
    console.log("hola");

o) let a = {};
   let b = a;
   if (a === b)
      console.log("hola");
```

Igualdad no estricta ==

2. Dadas las siguiente expresiónes en JavaScript, para cada una aplicando las reglas de orden de agrupación (precedencia y asociatividad) y orden de evaluación:
   - pon paréntesis de forma que, sin modificar su funcionalidad, haga claro en qué orden se evalúa.
   - indica el valor resultante de la expresión y de las variables, si las hay

```
a) true == Boolean(true)

b) false == Boolean(false)

c) true == new Boolean(true)

d) new Boolean(true) ==
    new Boolean(true)

e) "" == false

f) "hola" == true

g) -0.0 == false

h) 34 * 0/-1 == false

i) undefined == undefined

j) null == null

k) +0 == -0
```

```
l) '23' == 23

m) -'0'/10 == -0

n) [ 1, 2] == '1,2'

o) null == undefined

p) [ 1, 2] == '1, 2'

q) !(Nan == NaN)

r) -Infinity == -3/0

s) undefined == false

t) undefined == true

u) NaN == NaN
```

```
v) NaN != NaN

w) new String('ola') == 'ola'

x) let a = {};
   let b = {};
   if (a == b)
    console.log("hola");

y) let a = {};
   let b = a;
   if (a == b)
      console.log("hola");
```

Valores falsy y valores truthy

3. Dadas las siguiente expresiónes en JavaScript, para cada una aplicando las reglas de orden de agrupación (precedencia y asociatividad) y orden de evaluación:
   - pon paréntesis de forma que, sin modificar su funcionalidad, haga claro en qué orden se evalúa.
   - indica el valor resultante de la expresión y de las variables, si las hay

```
a) let a = 1;
   if (false) a++;

b) let a = 1;
   if (false) a++;;

c) let a = 0;
```

```
f) let a = 1;
   if (!'') a++;

g) let a = 1;
   if (!``) --a;

h) let a = 1;
```

```
l) let a = 1;
   if ('foo')
    a-- ? 1 : 0;

m) let a = 0;
   if ([] || a--)
```

```
  let resultado =
   eval('if( a++) a--; else --a;');
```

d)
```
let a = 0;
  let resultado =
   eval('if( a++) a--; else a--;');
```

e)
```
let a = 1;
  eval('if (!document.all) a++;');
  let a =
    eval('if (!document.all) a++;');
```

```
  if (0/0) a++;
```

i)
```
let a = 1;
  if (!undefined) a++;
```

j)
```
let a = 1;
  if (undefined == false) a++;
```

k)
```
let a = 1;
  if (null) a++; else a--;
```

```
    ++a;
```

n)
```
let a = 0;
  let b = 1;
  if ('.' && a--)
    --b;
```

o)
```
let a = 1;
  if (--a) a++;
```

p)
```
let a = 0;
  if (a++) a++;
```