UNIVERSIDAD AUTONÓMA DE NUEVO LEÓN

FACULTAD DE CIENCIAS FISICO MATEMATICAS

UNIDAD DE APRENDIZAJE: DISEÑO ORIENTADO A OBJETOS

TAREA #2

PROFESOR. MIGUEL ÁNGEL SALAZAR

ALUMNA. VALERIA MARTÍNEZ DE LA ROSA

MATRICULA. 1678575

San Nicolás de los garza, nuevo león a 26 de enero de 2017

1. IGUAL NO SIEMPRE ES IGUAL

```
1 == 1; //true
'foo' == 'foo'; //true
[1,2,3] == [1,2,3]; //false
```

   Esto es algo absolutamente habitual - uno es igual a uno y es foo foo. La matriz [1,2,3] no es igual a [1,2,3] porque es simplemente un tipo de referencia. Esto es común para la mayoría de los lenguajes. JavaScript convierte implícitamente los valores en los dos lados del operador == para cuerdas y los compara. Esto es una cosa más que debemos tener cuidado para. En lugar de == es mejor utilizar ===.
2. USTED PUEDE FINGIR AMBITO

   El ámbito en el que se ejecuta algo define qué variables son accesibles. Free-standing JavaScript (es decir JavaScript que no se ejecuta dentro de una función) opera dentro del ámbito global del objeto, a la que todo tiene acceso; mientras que las variables locales declaradas en funciones sólo son accesibles dentro de esa función, no afuera.
3. FIREFOX READS AND RETURNS COLORS IN RGB, NOT HEX.

   Mientras que la mayoría de los navegadores le avise ff9900, Firefox devuelve rgb(255, 153, 0), el equivalente RGB. Un montón de funciones de JavaScript están ahí fuera para la conversión de RGB a hexagonal.
4. INDEFINIDO PUEDE DEFINIRSE

   Undefined no es en realidad una palabra reservada en JavaScript, a pesar de que tiene un significado especial y es la única manera de determinar si una variable no está definida.
5. ARRAY ASSINITY

   Sumar dos arreglos vacíos da una cadena vacía, pero si sumamos un arreglo vacío más un objeto vacío y vi se versa es igual a cero, pero un objeto vacío y un objeto vacío no es igual a un número.
6. ARITMETICA BOOLEANA

```
js> true+true===2
true
js> true-true===0
true
```

```
js>true===1
false
```

Pareciera que la verdad es igual a 1, sin embargo en la 2da función lo desmiente.

7. COMILLAS

```
var persona = {
nombre: 'David',
'& propiedad raro': 'YYCJS'
}

var prop = 'nombre';

person.name // -> David
 persona [ 'nombre'] // -> David
 persona [ "nombre"] // -> David
 persona [prop] // -> David

 persona [ "propiedad extraños & '] // -> YYCJS

// ERROR
 persona. y rara propiedad
```

En JavaScript no hay diferencia entre usar comillas dobles o simples.

8. NO USA COMPILADOR
   JavaScript no utiliza compilador por lo que sólo podemos encontrar “errores de ejecución”
9. Las funciones son tratadas como cualquier otra variable, en JS podríamos decir que todo es igual de importante.
10. NULO ES UN OBJETO
    Nulo significa total ausencia de valor significativo, sin embargo en JavaScript null es un objeto

```
alert(typeof null); //alerts 'object'
```