GUIA No 1 DE JAVA SCRIPT

Los programas son instrucciones que se deben seguir

Javascript es un lenguaje con muchas posibilidades, utilizado para crear pequeños programas que luego son insertados en una página web y en programas más grandes, orientados a objetos mucho más complejos. Con Javascript podemos crear diferentes efectos e interactuar con nuestros usuarios.

¿Dónde puedo ver funcionando Javascript?

Entre los diferentes servicios que se encuentran realizados con Javascript en Internet se encuentran:
- ***Correo***
- ***Chat***
- ***Buscadores de Información***

También podemos encontrar o crear códigos para insertarlos en las páginas como:
- ***Reloj***
- ***Contadores de visitas***
- ***Fechas***
- ***Calculadoras***
- ***Validadores de formularios***
- ***Detectores de navegadores e idiomas***

En este ejemplo nos muestra en pantalla un aviso de Buenvenidad al curso

Ejemplo 1

```
<html>
<head>
</head>
<body>
<script type="text/javascript">
document.write("BIENVENIDOS Al CURSO JAVA SCRIPT");
</script>
</body>
</html>
```

Los Comentarios

Los comentarios en el código de programación sirven para aclarar cada sentencia, procedimiento, declaración, etc.

Puede usar:

// Este tipo de comentario sirve en una línea.

/* */ Este tipo de comentario es multilinea

Las Declaraciones son los pasos que se deber seguir, lo que se debe hacer

Ejemplo 1.1:

```
document.write("BIENVENIDOS AL SENA");
```

Ejemplo 2

```
<html>
<head>
</head>
<body>
<script type="text/javascript">
// este es el contenedor que sale en pantalla
document.write("BIENVENIDOS AL SENA");
</script>
</body>
</html>
```

= operando de asignación
X contenedor
Var variable

Los tipos de variables son:

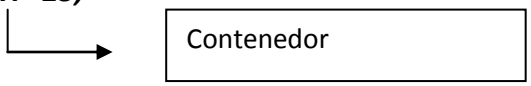
1) Variables de numero
2) Variables de cadena

3) Variable valida
4) Variable false
5) Variable null

VARIABLES

Es espacios reservados que les podemos dar un valor , para definir una variable se debe inicia con Var

Var X =23;



Las variables pueden ser

Numericas , cadena están lleva texto y van entre " "

True, false , Null que indica cero, nada n nulo

Ejemplo 3

```
<html>
<head>
<title>sena</title>
</head>
<body>
<script type="text/javascript">
var x = 25;
document.write(x);
</script>
</body>
</html></html>
```

Variables global y locales

En el presente codigo de trabaja con una variable llamada a , se crea una función

Ejemplo 3.1

```
<html>
<head>
<title>sena</title>
</head>
<body>
<script type="text/javascript">
	var a = "SENA";
	function local(){
		document.write("A mi me gusta el " + a +"<br/>");
	};
	local();
	document.write("SI SE PUEDE " + a +"<br/>");

</script>
</body>
</html>
```

Operadores

=	asignación
+	Suma
-	Resta
*	Multiplicación.
/	División

Ejemplo 4

```
<html>
<head>
<title>sena</title>
</head>
<body>
```

```
<script type="text/javascript">
var edad = 25;
var nombre="PEDRO";
var ciudad="Bogota";
var tiempo = 4;
var x =edad + tiempo;
document.write(edad +tiempo+1+"</br>");
document.write(" Hola" +"</br>"+  nombre + " y tengo" + edad);
</script>
</body>
</html>
```

FUNCIONES

Las funciones son pequeños programas dentro del programa grande.

Ejemplo 5:

```
<html>
<head>
<title>sena</title>
</head>
<body>
<script type="text/javascript">
function pedro(){
alert("BIENVENIDOS A JAVASCRIPT");
};
// hacer el llamado de la función, este va fuera de la función
pedro();
</script>
</body>
</html>
```

Ejemplo 5.1

```
<html>
<head>
<title>sena</title>
</head>
<body>
<script type="text/javascript">
function pedro(){
alert("BIENVENIDOS A JAVASCRIPT");
```

```
};
</script>
<form>
<input type="button"value="SENA"onclick="pedro()"/>
</form>
</body>
</html>
```

Ejemplo 5.2

```
<html>
<head>
<title>sena</title>
</head>
<body>
<script type="text/javascript">
 function pedro(nombre){
    document.write("HOLA " + nombre +"</br>");
 };
          pedro("Manuel Garavito");
          pedro("Orlando Supelano");
          pedro("consuelo  Rondorn");
          pedro("Pedro Castañeda");
</script>

</body>
</html>
```

Usted puede agrupar las funciones para que el código le quede mas ordenado

Ejemplo 5.3

```
<html>
<head>
<title>sena</title>
</head>
<body>
<script type="text/javascript">
          function uno(){
                    document.write("Soy Sena");
          };
          function dos(){
```

```
            document.write(" APRENDIZ");

        };
        function tres(){
        uno();
        dos();
        };
 tres();

</script>

</body>
</html>
```

DECLARACION IF

La declaración del If , es un si condicional o una pregunta lógica, que si cumple la condición realiza algo de lo contrario realizara la segunda opción

Ejemplo 6

```
<html>
<head>
<title>sena</title>
</head>
<body>
<script type="text/javascript">
        var pan = 10;
        var queso = 10;
        if(pan==queso)
        {
                document.write("Quiero desayuno!");
        };

</script>

</body>
</html>
```

Ejemplo 6.1

Uso del if con el else

```
<html>
<head>
<title>sena</title>
</head>
<body>
<script type="text/javascript">

	if(2==2)
	 {

		document.write("son igual!");
	}else{
		document.write("No son igual!");
	}

</script>

</body>
</html>
```

Nesting

Las dos comparaciones son validad esto sería nesting

Ejemplo 7

```
head>
<title>sena</title>
</head>
<body>
<script type="text/javascript">
          var nombre="pedro";
          var edad=30;
          if(nombre=="pedro"){
          if(edad==30){
          alert("SI FUNCIONA!!");
           /* del if de adentro

          }
          /* es del priemr if
  }


</script>
</body>
</html>
```

Ejemplo 7.1

```
<html>
<head>
<title>sena</title>
</head>
<body>
<script type="text/javascript">
          var nombre="pedro";
          var edad=30;
          if(nombre=="pedro"){
          if(edad==31){
          alert("SI FUNCIONA!!");
           /* del if de adentro */

          } else
                    {
                    document.write("La edad no es igual");
                    }

          /* es del priemr if*/

  }else
     {
          document.write("else 2");
      }

</script>

</body>
</html>
```

Ejemplo 7.2 funciona el primer if

```
<html>
<head>
<title>sena</title>
</head>
<body>
<script type="text/javascript">
          var nombre="pedro";
```

```
        var edad=30;
        if(nombre=="diego"){
        if(edad==30){
        alert("SI FUNCIONA!!");
         /* del if de adentro */

        } else
                {
                document.write("La edad no es igual");
                }

        /* es del priemr if*/

  }else
     {
        document.write("Aqui funciona el primer if");
     }

</script>

</body>
</html>
```

Ejemplo 7.3

El operador && (i) sirve para encadenar
El operador || (o) este operador lo puede hacer Alt +124
Con la condición (o) se ejecuta cuando alguna de las dos condiciones es verdadera.

```
<html>
<head>
<title>sena</title>
</head>
<body>
<script type="text/javascript">
        var mes="julio";
        var dia=21;
        if((mes=="julio") && (dia==21)){
          document.write("FELIZ CUMPLIAÑOS MI PC");
        }
```

```
</script>

</body>
</html>
```

Ejemplo 7.3

```
<html>
<head>
<title>sena</title>
</head>
<body>
<script type="text/javascript">
          var mes="julio";
          var dia=21;
         if((mes=="Agosto") || (dia==21)){
           document.write("FELIZ CUMPLIAÑOS MI PC");
         }

</script>

</body>
</html>
```

Declaración switch

Preguntas y respuestas

Ejemplo 8

```
<html>
<head>
<title>sena</title>
</head>
<body>
<script type="text/javascript">
          var comida = "queso";
          switch(comida){
          case "pan":
           document.write("Desayuno");
          break;
          case "leche":
           document.write("Liquido");
          break;
          case "queso":
           document.write("plato fuerte");
          break;
          default:
     document.write("No hay plata para el Desayuno");
 }
</script>

</body>
</html>
```

Si no cumple la condición sale el mensaje por defecto

Ejemplo 8.1

```
<html>
<head>
<title>sena</title>
</head>
<body>
<script type="text/javascript">
          var comida = "miel";
          switch(comida){
          case "pan":
           document.write("Desayuno");
          break;
          case "leche":
           document.write("Liquido");
          break;
          case "queso":
           document.write("plato fuerte");
          break;
          default:
     document.write("No hay plata para el Desayuno");
 }
</script>

</body>
</html>
```

Simplificación de operaciones matemáticas

Ejemplo 9

```
<html>
<head>
<title>sena</title>
</head>
<body>
<script type="text/javascript">
          var x = 2;
```

```
        document.write(x);
          document.write("<br/>");
          document.write(x = x +2);
          document.write("<br/>");
          document.write(x = x +2);

</script>

</body>
</html>
```

Ejemplo 9.1

```
<html>
<head>
<title>sena</title>
</head>
<body>
<script type="text/javascript">
          var x = 2;

        document.write(x+=2);
          document.write("<br/>");
          document.write(x *=2);
          document.write("<br/>");
          document.write(x *= +3);

</script>

</body>
</html>
```

Formas del uso los operadores:

+=
-=
*=
/=
%= sirve para el resto de las divisiones
X = x+4
X +=4
++ incremento
--

Incremente por 1:

Ejemplo 9.2

```
<html>
<head>
<title>sena</title>
</head>
<body>
<script type="text/javascript">
        var x = 2;

      document.write(x+=4);
        document.write("<br/>");
        document.write(++x);
        document.write("<br/>");
        document.write(++x);

</script>

</body>
</html>
```

Bucles
sirve para repetir un números de veces un código
(for loop) y while   hace los mismo solo cambia es la sintaxis

Ejemplo 10

```
<html>
<head>
<title>sena</title>
</head>
<body>
<script type="text/javascript">
        for(x=0;x<10;x+=4){
        document.write(x + " ASI MIPC <br/>");

        }
```

```
</script>

</body>
</html>
```

Ejemplo 10.1

```
<html>
<head>
<title>sena</title>
</head>
<body>
<script type="text/javascript">
        for(x=0;x<10;x++){
        document.write(x + " ASI MIPC <br/>");

        }
</script>

</body>
</html>
```

Ejemplo 10.2

```
<html>
<head>
<title>sena</title>
</head>
<body>
<script type="text/javascript">
        var x = 1;
         while(x<=10){
        document.write(" Hola Pedro Castañeda<br/>");
        x++;

        }
</script>

</body>
</html>
```

Bucle do while

Ejemplo 10.3

```
<html>
<head>
<title>sena</title>
</head>
<body>
<script type="text/javascript">
        var x = 5;
        do{
      document.write("este es mi ejercicio <br/>");
        x++;
        }
        while(x<=10);

</script>

</body>
</html>
```

Eventos en java Script

Sirve para ejecutar un cierto código en el momento lo desee

Los eventos se capturan mediante los manejadores de eventos. El proceso a realizar se programa mediante funciones JavaScript llamadas por los manejadores de eventos

Manejador	Versión	Se produce cuando...
onAbort	1.1	El usuario interrumpe la carga de una imagen
onBlur	1.0	Un elemento de formulario, una ventana o un marco pierden el foco
onChange	1.0 (1.1 para*FileUpload*)	El valor de un campo de formulario cambia
onClick	1.0	Se hace *click* en un objeto o formulario
onDblClick	1.2 (no en Mac)	Se hace *click* doble en un objeto o formulario
onDragDrop	1.2	El usuario arrastra y suelta un objeto en la ventana

onError	1.1	La carga de un documento o imagen produce un error
onFocus	1.1 (1.2 para *Layer*)	Una ventana, marco o elemento de formulario recibe el foco
onKeyDown	1.2	El usuario pulsa una tecla
onKeyPress	1.2	El usuario mantiene pulsada una tecla
onKeyUp	1.2	El usuario libera una tecla
onLoad	1.0 (1.1 para *image*)	El navegador termina la carga de una ventana
onMouseDown	1.2	El usuario pulsa un botón del ratón
onMouseMove	1.2	El usuario mueve el puntero
onMouseOut	1.1	El puntero abando una área o enlace
onMouseOver	1.0 (1.1 para *area*)	El puntero entra en una área o imagen
onMouseUp	1.2	El usuario libera un botón del ratón
onMove	1.2	Se mueve una ventana o un marco
onReset	1.1	El usuario limpia un formulario
onResize	1.2	Se cambia el tamaño de una ventana o marco
onSelect	1.0	Se selecciona el texto del campo texto o área de texto de un formulario
onSubmit	1.0	El usuario envía un formulario
onUnload	1.0	El usuario abandona una página

Ejemplo de evento:

```
<INPUT TYPE="text" onChange="CompruebaCampo(this)">
```

Ejemplo 11

```
<html>
<head>
<title>sena</title>
</head>
<body>
  <form>

          <input type="button"onClick="alert('HOla Pedro!!');" value=" Entrar"/>

</form>

</body>
</html>
```

Ejemplo 11.1

```
<title>sena</title>
 <script type="text/javascript">
         function alerta(){
         alert(" Me gusta aprender!");
     }
  </script>
</head>
<body>


          <a href="#" onMouseOver="alerta();"> COLOMBIA</a>

</body>
</html>
```

Programación orientada a objetos

Un Objeto es unidad de datos con sus propios métodos y propiedades

OBJETO = JOSE

PROPIETADES = ALTO

METODOS: lo que puede hacer Ejemplo Dormir, Trabajar.

Las propiedades se guardan en variables.

LAS VARIABLES TAMBIEN SON OBJETOS

Ejemplo 12

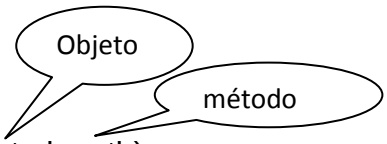
```
<html>
<head>
<title>pedrocastaneda</title>
 <script type="text/javascript">
        var texto = "mi casa es grande";
        /* esta propiedad sirve  contar los caracteres */
        document.write(texto.length);

   </script>
</head>

<body>

</body>
</html>
```

Los metodo

Es la acción lo que queremos hacer

Objeto.metodo

document es en java la parte visible en la página web , como es en html la etiqueta body.



document.write(texto.length)

Ejemplo 12.1

```
<html>
<head>
<title>pedrocastaneda</title>
 <script type="text/javascript">
          function persona(nombre,edad)
          {
        this.nombre = nombre;
        this.edad = edad;
          this.añodenacimiento = nacimiento;
          }
          /* este es  un metodo  en forma de funcion */
        function nacimiento(){
           var años = 2013 - this.edad;
         return años;
         }

          /* este es un objeto */
    var diego = new persona ("diego", 12)

   </script>
</head>


<body>

   <script type="text/javascript">
          /* edad */
          document.write(diego.añodenacimiento());

 </script>

</body>
</html>
```

Ejemplo 12.2

```
<html>
<head>
<title>pedrocastaneda</title>
 <script type="text/javascript">
          function persona(nombre,edad)
          {
        this.nombre = nombre;
        this.edad = edad;
```

```
          this.añodenacimiento = nacimiento;
          }
          /* este es  un metodo  en forma de funcion */
        function nacimiento(){
           var años = 2013 - this.edad;
         return años;
         }

          /* este es un objeto */
    var diego = new persona ("diego", 12)

   </script>
</head>

<body>

   <script type="text/javascript">
          /* edad */
          document.write(diego.nombre + " tiene "+ diego.edad + " años"
        + " Nacio en " + diego.añodenacimiento());

 </script>

</body>
</html>
```

Ejemplo 12.3

```
<html>
<head>
<title>pedrocastaneda</title>
 <script type="text/javascript">
          function persona(nombre,edad)
          {
        this.nombre = nombre;
        this.edad = edad;
          this.añodenacimiento = nacimiento;
          }
          /* este es  un metodo  en forma de funcion */
        function nacimiento(){
           var años = 2013 - this.edad;
         return años;
         }

          /* este es un objeto */
```

```
   var diego = new persona ("diego", 12);
   var pedro= new persona ( "pedro" , 46);
  </script>
</head>


<body>

  <script type="text/javascript">
        /* edad */
        document.write(pedro.nombre + " tiene "+ pedro.edad + " años"
      + " Nacio en " + pedro.añodenacimiento());

 </script>

</body>
</html>
```

Ejemplo 12.4

```
<html>
<head>
<title>pedrocastaneda</title>
 <script type="text/javascript">
        consuelo = {nombre:"Consuelo", edad:38, altura: 1.68}
        alexander = {nombre: "Alexander" , edad:30, altura: 1.85}

  </script>
</head>


<body>

  <script type="text/javascript">

        document.write(consuelo.nombre + " y " +
      alexander.nombre + " son amigos. ");

 </script>

</body>
</html>
```

Método rápido

objeto

propiedades

consuelo = {nombre:"Consuelo", edad:38, altura: 1.68}

Ejemplo 12.5

```
<html>
<head>
<title>pedrocastaneda</title>
 <script type="text/javascript">
        consuelo = {nombre:"Consuelo", edad:38, altura: 1.68}
        alexander = {nombre: "Alexander" , edad:30, altura: 1.85}

   </script>
</head>


<body>

   <script type="text/javascript">

        document.write( consuelo.edad);

 </script>

</body>
</html>
```

Métodos

Método	Descripción	Objeto JavaScript
anchor (Método)	Pone un delimitador HTML que tiene un atributo NAME alrededor del texto.	**String**
apply (Método)	Aplica un método de un objeto y sustituye otro objeto por el objeto actual.	**Function**

atEnd (Método)	Devuelve un valor de tipo Boolean que indica si el enumerador está al final de la colección.	**Enumerator**
big (Método)	Pone etiquetas HTML <BIG> alrededor del texto.	**String**
bind (Método)	Crea una función que está asociada a un objeto especificado y que puede tener parámetros iniciales concretos.	**Function**
blink (Método)	Pone etiquetas HTML <BLINK> alrededor del texto.	**String**
bold (Método)	Pone etiquetas HTML <B> alrededor del texto.	**String**
call (Método)	Llama a un método de un objeto y sustituye el objeto actual por otro.	**Function**
charAt (Método)	Devuelve el carácter que se encuentra en el índice especificado.	**String**
charCodeAt (Método)	Devuelve la codificación Unicode del carácter que se especifique.	**String**
compile (Método)	Compila una expresión regular y la convierte en un formato interno.	**Regular Expression**
concat (Método, Array)	Devuelve una matriz nueva que se compone de una combinación de dos matrices.	**Array**
concat (Método, String)	Devuelve un objeto **String** que contiene la concatenación de las dos cadenas proporcionadas.	**String**
dimensions (Método)	Devuelve el número de dimensiones de un objeto VBArray.	**VBArray**
every (Método)	Comprueba si una función de devolución de llamada definida devuelve **true**para todos los elementos de una matriz.	**Array**
exec (Método)	Ejecuta una búsqueda en una cadena especificada.	**Regular Expression**
filter (Método)	Llama a una función de devolución de llamada definida	**Array**

	para cada elemento de una matriz y devuelve una matriz de aquellos valores para los que esa función devuelve **true**.	
fixed (Método)	Pone etiquetas HTML <TT> alrededor del texto.	**String**
fontcolor (Método)	Pone etiquetas HTML <FONT> con un atributo COLOR alrededor del texto.	**String**
fontsize (Método)	Pone etiquetas HTML <FONT> con un atributo SIZE alrededor del texto.	**String**
forEach (Método)	Llama a una función de devolución de llamada definida para cada elemento de una matriz.	**Array**
getDate (Método)	Devuelve el valor de día del mes usando la hora local.	**Date**
getDay (Método)	Devuelve el valor de día de la semana usando la hora local.	**Date**
getFullYear (Método)	Devuelve el valor de año usando la hora local.	**Date**
getHours (Método)	Devuelve el valor de horas usando la hora local.	**Date**
getItem (Método)	Devuelve el elemento que se encuentra en la ubicación especificada.	**VBArray**
getMilliseconds (Método)	Devuelve el valor de milisegundos usando la hora local.	**Date**
getMinutes (Método)	Devuelve el valor de minutos usando la hora local.	**Date**
getMonth (Método)	Devuelve el valor de mes usando la hora local.	**Date**
getSeconds (Método)	Devuelve el valor de segundos usando la hora local.	**Date**
getTime (Método)	Devuelve el valor de tiempo en un objeto **Date** en milisegundos desde la medianoche del 1 de enero de 1970.	**Date**
getTimezoneOffset	Devuelve la diferencia en minutos entre la hora del	**Date**

(Método)	equipo host y la hora universal coordinada (UTC).	
getUTCDate (Método)	Devuelve el valor de día del mes usando la hora UTC.	**Date**
getUTCDay (Método)	Devuelve el valor de día de la semana usando la hora UTC.	**Date**
getUTCFullYear (Método)	Devuelve el valor de año usando la hora UTC.	**Date**
getUTCHours (Método)	Devuelve el valor de horas usando la hora UTC.	**Date**
getUTCMilliseconds (Método)	Devuelve el valor de milisegundos usando la hora UTC.	**Date**
getUTCMinutes (Método)	Devuelve el valor de minutos usando la hora UTC.	**Date**
getUTCMonth (Método)	Devuelve el valor de mes usando la hora UTC.	**Date**
getUTCSeconds (Método)	Devuelve el valor de segundos usando la hora UTC.	**Date**
getVarDate (Método)	Devuelve el valor de VT_DATE de un objeto **Date**.	**Date**
getYear (Método)	Devuelve el valor de año.	**Date**
hasOwnProperty (Método)	Devuelve un valor Boolean que indica si un objeto tiene una propiedad con el nombre especificado.	Multiple
indexOf (Método, Array)	Devuelve el índice de la primera aparición de un valor de una matriz.	**Array**
indexOf (Método, String)	Devuelve la posición del carácter donde tiene lugar la primera repetición de una subcadena dentro de un objeto **String**.	**String**
isPrototypeOf	Devuelve un valor Boolean que indica si un objeto	Multiple

(Método)	existe en la cadena de prototipos de otro objeto.	
italics (Método)	Pone etiquetas HTML <I> alrededor del texto.	**String**
item (Método)	Devuelve el elemento actual de la colección.	**Enumerator**
join (Método)	Devuelve un objeto **String** formado por todos los elementos de una matriz concatenados entre sí.	**Array**
lastIndexOf (Método, Array)	Devuelve el índice de la última aparición de un valor especificado de una matriz.	**Array**
lastIndexOf (Método, String)	Devuelve la última repetición de una subcadena dentro de un objeto**String**.	**String**
lbound (Método)	Devuelve el menor valor de índice que se utiliza en la dimensión especificada de un objeto VBArray.	**VBArray**
link (Método)	Pone un delimitador HTML que tiene un atributo HREF alrededor del texto.	**String**
localeCompare (Método)	Devuelve un valor que indica si dos cadenas son equivalentes en la configuración regional actual.	**String**
map (Método)	Llama a una función de devolución de llamada definida para cada elemento de una matriz y devuelve una matriz que contiene los resultados.	**Array**
match (Método)	Devuelve, como matriz, los resultados de una búsqueda en una cadena utilizando el objeto **Regular Expression** proporcionado.	**String**
moveFirst (Método)	Restablece el elemento actual de la colección en el primer elemento.	**Enumerator**
moveNext (Método)	Desplaza el elemento actual al siguiente elemento de la colección.	**Enumerator**
pop (Método)	Quita el último elemento de una matriz y lo devuelve.	**Array**
propertyIsEnumerable	Devuelve un valor Boolean que indica si una propiedad	Multiple

(Método)	especificada forma parte de un objeto y si se puede enumerar.	
push (Método)	Anexa nuevos elementos a una matriz y devuelve la nueva longitud de la matriz.	**Array**
reduce (Método)	Acumula un solo resultado llamando a una función de devolución de llamada definida para todos los elementos de una matriz. El valor devuelto de la función de devolución de llamada es el resultado acumulado, y se proporciona como argumento en la siguiente llamada a dicha función.	**Array**
reduceRight (Método)	Acumula un solo resultado llamando a una función de devolución de llamada definida para todos los elementos de una matriz, en orden descendente. El valor devuelto de la función de devolución de llamada es el resultado acumulado, y se proporciona como argumento en la siguiente llamada a dicha función.	**Array**
replace (Método)	Devuelve una copia de una cadena con texto reemplazado utilizando una expresión regular.	**String**
reverse (Método)	Devuelve un objeto **Array** con los elementos invertidos.	**Array**
search (Método)	Devuelve la posición de la primera coincidencia de subcadena en una búsqueda de expresión regular.	**String**
setDate (Método)	Establece el día del mes numérico usando la hora local.	**Date**
setFullYear (Método)	Establece el valor de año usando la hora local.	**Date**
setHours (Método)	Establece el valor de horas usando la hora local.	**Date**
setMilliseconds (Método)	Establece el valor de milisegundos usando la hora local.	**Date**
setMinutes (Método)	Establece el valor de minutos usando la hora local.	**Date**
setMonth (Método)	Establece el valor de mes usando la hora local.	**Date**

setSeconds (Método)	Establece el valor de segundos usando la hora local.	**Date**
setTime (Método)	Establece el valor de fecha y hora en el objeto **Date**.	**Date**
setUTCDate (Método)	Establece el día numérico del mes usando la hora UTC.	**Date**
setUTCFullYear (Método)	Establece el valor de año usando la hora UTC.	**Date**
setUTCHours (Método)	Establece el valor de horas usando la hora UTC.	**Date**
setUTCMilliseconds (Método)	Establece el valor de milisegundos usando la hora UTC.	**Date**
setUTCMinutes (Método)	Establece el valor de minutos usando la hora UTC.	**Date**
setUTCMonth (Método)	Establece el valor de mes usando la hora UTC.	**Date**
setUTCSeconds (Método)	Establece el valor de segundos usando la hora UTC.	**Date**
setYear (Método)	Establece el valor de año usando la hora local.	**Date**
shift (Método)	Quita el primer elemento de una matriz y lo devuelve.	**Array**
slice (Método, Array)	Devuelve una sección de una matriz.	**Array**
slice (Método, String)	Devuelve una sección de una cadena.	**String**
small (Método)	Pone etiquetas HTML <SMALL> alrededor del texto.	**String**
some (Método)	Comprueba si una función de devolución de llamada definida devuelve **true**para cualquier elemento de una matriz.	**Array**
sort (Método)	Devuelve un objeto **Array** con los elementos ordenados.	**Array**

splice (Método)	Quita elementos de una matriz, inserta nuevos elementos en su lugar si procede y devuelve los elementos eliminados.	**Array**
split (Método)	Devuelve la matriz de cadenas resultante de la separación de una cadena en subcadenas.	**String**
strike (Método)	Pone etiquetas HTML <STRIKE> alrededor del texto.	**String**
sub (Método)	Pone etiquetas HTML <SUB> alrededor del texto.	**String**
substr (Método)	Devuelve una subcadena que comienza en una posición especificada y tiene una longitud especificada.	**String**
substring (Método)	Devuelve la subcadena en la ubicación especificada dentro de un objeto**String**.	**String**
sup (Método)	Pone etiquetas HTML <SUP> alrededor del texto.	**String**
test (Método)	Devuelve un valor Boolean que indica si existe o no un patrón en una cadena de búsqueda.	**Regular Expression**
toArray (Método)	Devuelve una matriz JavaScript estándar convertida a partir de un objeto VBArray.	**VBArray**
toDateString (Método)	Devuelve una fecha como un valor alfanumérico.	**Date**
toExponential (Método)	Devuelve una cadena que contiene un número representado en notación exponencial.	**Number**
toFixed (Método)	Devuelve una cadena que representa un número en notación de punto fijo.	**Number**
toGMTString (Método)	Devuelve una fecha convertida en cadena utilizando la hora media de Greenwich (GMT).	**Date**
toISOString (Método)	Devuelve una fecha como un valor alfanumérico en formato ISO.	**Date**
toJSON (Método)	Se utiliza para transformar datos de un tipo de objeto antes de la serialización JSON.	**Date**

toLocaleDateString (Método)	Devuelve una fecha como un valor alfanumérico apropiado para la configuración regional actual del entorno host.	**Date**
toLocaleLowerCase (Método)	Devuelve una cadena donde todos los caracteres alfabéticos se han convertido a minúsculas, según la configuración regional actual del entorno host.	**String**
toLocaleString (Método)	Devuelve un objeto convertido en cadena usando la configuración regional actual.	Multiple
toLocaleTimeString (Método)	Devuelve una hora como un valor alfanumérico apropiado para la configuración regional actual del entorno host.	**Date**
toLocaleUpperCase (Método)	Devuelve una cadena donde todos los caracteres alfabéticos se han convertido a mayúsculas, según la configuración regional actual del entorno host.	**String**
toLowerCase (Método)	Devuelve una cadena donde todos los caracteres alfabéticos se han convertido a minúsculas.	**String**
toPrecision (Método)	Devuelve una cadena que contiene un número representado en notación exponencial o de punto fijo con un número especificado de dígitos.	**Number**
toString (Método)	Devuelve una representación alfanumérica de un objeto.	Multiple
toTimeString (Método)	Devuelve una hora como un valor alfanumérico.	**Date**
toUpperCase (Método)	Devuelve una cadena donde todos los caracteres alfabéticos se han convertido a mayúsculas.	**String**
toUTCString (Método)	Devuelve una fecha convertida en cadena usando la hora UTC.	**Date**
trim (Método)	Devuelve una cadena donde se han quitado los caracteres de espacio en blanco y de terminador de línea iniciales y finales.	**String**
ubound (Método)	Devuelve el mayor valor de índice utilizado en la	**VBArray**

	dimensión especificada del objeto VBArray.	
unshift (Método)	Inserta nuevos elementos al principio de una matriz.	**Array**
valueOf (Método)	Devuelve el valor primitivo del objeto especificado.	Multiple

Ejemplo 13

```
<html>
<head>
<title>pedrocastaneda</title>
 <script type="text/javascript">
        var fecha = new Date();
        var año = fecha.getFullYear();
  </script>
</head>


<body>

  <script type="text/javascript">

        document.write(año);

 </script>

</body>
</html>
```

Ejemplo 13.1

```
<html>
<head>
<title>pedrocastaneda</title>
 <script type="text/javascript">
        var fecha = new Date();
        var año = fecha.getFullYear();
        var hora = fecha.getHours();
  </script>
```

```
</head>


<body>

   <script type="text/javascript">

          document.write("Año: "+ año + "<br/>");

          document.write("Hora:"+hora + "<br/>");
 </script>

</body>
</html>
```

Ejemplo 13.2

```
<html>
<head>
<title>pedrocastaneda</title>
 <script type="text/javascript">
          var fecha = new Date();
          fecha.setFullYear(2000);

          fecha.setDate(25);


   </script>
</head>


<body>

   <script type="text/javascript">

          document.write(fecha);


 </script>

</body>
</html>
```

Ejemplo 13.3 reloj manual

```
<html>
<head>
<title>pedrocastaneda</title>
 <script type="text/javascript">
        var fecha= new Date();
        var hora= fecha.getHours();
        var min =fecha.getMinutes();
        var seg = fecha.getSeconds();


   </script>
</head>

<body>

   <script type="text/javascript">

        document.write(hora+ ":" + min+ ":" +seg);


 </script>

</body>
</html>
```

El siguiente código me hace el reloj que cambie automáticamente:

Ejemplo 13.4

```
<html>
<head>
<title>pedrocastaneda</title>
 <script type="text/javascript">
   function reloj(){
        var fecha= new Date();
        var hora= fecha.getHours();
```

```
        var min =fecha.getMinutes();
        var seg = fecha.getSeconds();
        var recarga = setTimeout('reloj()',500);
        document.getElementById('print').innerHTML = hora + ":" + min + ":" + seg;
        }

  </script>
</head>


<body onload="reloj()">

  <script type="text/javascript">


 </script>

<div id="print"></div>

</body>
</html>
```

Arrays
Los arrays lo conforma indces estos empiezan con cero i van incrementado
En esto caso son corsa, aveo,sprint,bmw
Donde corsa es= 0, aveo = 1, sprint= 2, bmw=3

```
var carro= new Array("corsa","aveo","sprint","bmw")
```

Ejemplo 14

```
<html>
<head>
<title>pedrocastaneda</title>


</head>

<body >

  <script type="text/javascript">

        var carro= new Array("corsa","aveo","sprint","bmw");
        document.write(carro[3]);
 </script>
```

```
<div id="print"></div>

</body>
</html>
```

Ejemplo 14.1

En este emplo vemos una forma diferente de presentar los arreglos
En esta línea se define líneas que lleva el arreglo:

var estudiantes= new Array(5);
usted puede digitar el siguiente código para ver la nueva forma de ver los arreglos:

```
<html>
<head>
<title>pedrocastaneda</title>
</head>
<body >

  <script type="text/javascript">

        var estudiantes= new Array(5);
        estudiantes[0]="Diego";
        estudiantes[1]="Stella";
        estudiantes[2]="Jhon";
        estudiantes[3]="Maria";
        estudiantes[4]="Juan";


        document.write(estudiantes[4]);

 </script>

<div id="print"></div>

</body>
</html>
```

Ejemplo 14.2

Esn tes código vemos la propiedad length, que sirve para contar los caracteres cono son en la siguiente frase: “me gusta trabajar con amor"; tenemos un total de 28 caracteres.

```
<html>
```

```
<head>
<title>pedrocastaneda</title>
</head>
<body >

  <script type="text/javascript">

        var texto= " me gusta trabajar  con amor";

        document.write(texto.length);

 </script>


</body>
</html>
```

Ejemplo 14.3

Imprimir Array con bucle for

```
<html>
<title>pedrocastaneda</title>
</head>
<body >

  <script type="text/javascript">

        var musica = new Array("guitarra","piano","bateria","violin");

        for(a = 0;a < musica.length; a++){

        document.write(musica[a] + "<br/>");
        }

 </script>


</body>
</html>
```

Ejemplo 14.4
Une dos arreglos en uno usando la propiedad concat

```
</head>
<body >

  <script type="text/javascript">

        var progra = new Array("html","js","Css");
        var progra2 = new Array("php","mysql","ruby");

        var total = progra2.concat(progra);

        document.write(total[5]);


 </script>


</body>
</html>
```

Metodos join, pop y push

Uso del join:

Ejemplo 15

```
<html>
<head>
<title>pedrocastaneda</title>
</head>
<body >

  <script type="text/javascript">

        var progra = new Array("html","js","Css");
        var cadena = progra.join(": ");

        document.write(cadena);


 </script>


</body>
```

```
</html>
```

Metodo pop este borra el ultimo valor o índice

Ejemplo 15.1

```
<html>
<head>
<title>pedrocastaneda</title>
</head>
<body >

   <script type="text/javascript">

          var progra = new Array("html","js","Css");

          progra.pop();

          var cadena = progra.join(": ");

          document.write(cadena);


 </script>


</body>
</html>
```

Con el método push agregamos   dotos
Ejemplo 15.2

```
<html>
<head>
<title>pedrocastaneda</title>
</head>
<body >
```

```
  <script type="text/javascript">

        var progra = new Array("html","js","Css");

        progra.pop();
        progra.push("casa","apartamento");

        var cadena = progra.join(",  ");

        document.write(cadena);


 </script>


</body>
</html>
```

Metodos reverse y sort

Ejemplo 16

```
html>
<head>
<title>pedrocastaneda</title>
</head>
<body >

  <script type="text/javascript">

        var progra = new Array("html","js","Css");
        var num = new Array(21,7,4,17,12);


        document.write(progra +"<br/>" +num);
```

```
 </script>

</body>
</html>
```

Uso del sort , este método sirve para ordenar
Reverse, este método hace darle la vuelta

Ejemplo 16.1

```
<html>
<head>
<title>pedrocastaneda</title>
</head>
<body >

  <script type="text/javascript">

      var progra = new Array("html","js","Css");
      var num = new Array(6,2,5,4,1,3);

         progra.reverse();
         num.sort();
          document.write(progra +"<br/>" + num);
```

```
 </script>
</body>
</html>
```

USTED SI PUEDE