2020

ARRAYS_1 Y ARRAYS_2

JOHN ESTEBAN PERDOMO SOTO
JHON ESTEBAN RESTREPO ESCOBAR

UTP | Pereira

1 CONTENIDO

2 PRESENTACIÓN

La presente monografía es creada para ampliar los conceptos de los dos códigos bases de arrays expuestos en la clase de introducción a la informática. Donde vamos a encontrar funciones básicas que se utilizan al usar los arrays en un código.

AUTOR: Jhon Esteban Restrepo Escobar>

1004735572

Jhonesteban.restrepo@utp.edu.co

<GITHUB Estudiante>

AUTOR: <John Esteban Perdomo Soto>

<1004776460>

<john.perdomo@utp.edu.co>

<GITHUB Estudiante>

3 ARRAYS-01

El código del primer programa crea la función texto la cual se utiliza para al ser ejecutada mostrar la lista que separe los programas y para poner el titulo de cada programa llamando la función texto y seguida de esta el texto que se quiera apreciar.

Después se crea un array, justo después de eso se crea un ciclo donde se asigna la suma de 1 en 1 y finalmente se expone la función donde se representa la secuencia de números que va ser vista al ejecutar el código.

ARRAY QUE MUESTRA LOS NUMEROS DEL 1 HASTA EL 149.

```
function texto( cadena, lineas_post = 0 ) {
  if (typeof(cadena) == "string") {
    if (cadena.indexOf("(::)") == 0) {
      document.write("============================================================<br>");
    }
  }
  document.write( cadena );
  for (var i = 0; i < lineas_post; i++) {
    document.write("<br>");
  }
}
texto("(::) TALLER SOBRE MANEJO INTEGRAL DE ARRAYS EN JAVASCRIPT", 2);
var numeros = [];

for (var i = 0; i < 149; ++i) {
  numeros[i] = i+1;
}

texto("(::) Array con los numeros desde el 1 hasta el 149", 2)
for (var i = 0; i < 149; i++) {
  texto(numeros[i] + "-");
}
```

A continuación el programa ejecutado.

```
============================================================
(::) TALLER SOBRE MANEJO INTEGRAL DE ARRAYS EN JAVASCRIPT

============================================================
(::) Array con los numeros desde el 1 hasta el 149

1-2-3-4-5-6-7-8-9-10-11-12-13-14-15-16-17-18-19-20-21-22-23-24-25-26-27-28-29-30-31-32-33-34-35-36-37-38-39-40-41-42-43-44-45-46-47-48-49-50-51-52-53-54-55-56-57-58-59-60-61-62-63-64-65-66-67-68-69-70-71-72-73-74-75-76-77-78-79-80-81-82-83-84-85-86-87-88-89-90-91-92-93-94-95-96-97-98-99-100-101-102-103-104-105-106-107-108-109-110-111-112-113-114-115-116-117-118-119-120-121-122-123-124-125-126-127-128-129-130-131-132-133-134-135-136-137-138-139-140-141-142-143-144-145-146-147-148-149-
```

Este programa lo que hace es identificar los elementos de un array y hacer una suma de todos sus elementos en este caso se suman los números del 1 al 5 y da como resultado el numero 15.

```
var numeros = [1,2,3,4,5];

var suma = numeros[0] + numeros[1] + numeros[2] + numeros[3] + numeros[4];

texto("", 2);
texto("(::) Suma manual de los elementos del array: " + numeros, 2)
texto("Suma = " + suma, 2);
```

A continuación, el código ejecutado.

```
============================================================
(::) Suma manual de los elementos del array: 1,2,3,4,5

Suma = 15
```

A comparación del anterior programa este no toma los elementos del array individualmente, sino que los hace parte de un ciclo y los suma.

```
var numeros = [1,2,3,5,8,13,21];

var suma = 0;

for (var i = 0; i < numeros.length; ++i)
{
  suma += numeros[i];
}

texto("(::) Suma de elementos del array: " + numeros + " utilizando ciclos", 2);
texto("Suma = " + suma, 2);
```

A continuación, el programa ejecutado.

```
============================================================
(::) Suma de elementos del array: 1,2,3,5,8,13,21 utilizando ciclos

Suma = 53
```

El siguiente programa cumple con la función de dividir una frase. El programa tiene una variable llamada frase donde se escribe la frase que se quiere separar y en la siguiente línea de código esta la función “frase.split” donde se utiliza para separar una cadena y se define “ “ donde por cada espacio que se detecte se divida la palabra.

```
texto("(::) Se divide una frase en palabras. La frase es = '" + frase + "'", 2);
for (var i = 0; i < palabras.length; ++i)
{
  texto("Palabra: " + i + ": " + palabras[i], 1);
}

texto("", 1);
```

A continuación el código en ejecución.

```
============================================================
(::) Se divide una frase en palabras. La frase es = 'la programación orientada a objetos en javascript'

Palabra: 0: la
Palabra: 1: programación
Palabra: 2: orientada
Palabra: 3: a
Palabra: 4: objetos
Palabra: 5: en
Palabra: 6: javascript
```

Este código cumple con la función de buscar un nombre dentro de un array y decir en qué posición del array está, Esto se hace con el condicional que examina el array y con las otras funciones que se ejecutan.

```
var nombres = ["David", "Sofía", "Ramón", "Carlos", "María"];
texto("(::) Lista de nombres = ", 1);
texto("(-) " + nombres, 2);

nombre = "Ramón";

texto("(-) Se está buscando el nombre: Ramón", 1);
var posicion = nombres.indexOf(nombre);

if (posicion >= 0) {
  texto("(-) Encontrado " + nombre + " en la posición " + posicion, 2);
}
else {
  texto("(-) " + nombre + " no encontrado en el array.", 2);
}
```

A continuación el código ejecutado.

(::) Lista de nombres =
(-) David,Sofía,Ramón,Carlos,María

(-) Se está buscando el nombre: Ramón
(-) Encontrado Ramón en la posición 2

El siguiente programa puede encontrar el mismo nombre requerido en diferentes posiciones del array, En este caso en la primera y la ultima posición.

```
var nombres = ["David", "Miguel", "Sofía", "Miguel", "Teresa", "Miguel", "Ramón", "Carlos", "Miguel", "María"];

texto("(::) Se busca un nombre en la lista = ", 1);
texto("(-) " + nombres, 2);

var nombre = "Miguel";

texto("(-) El nombre a buscar es: " + nombre, 1);

var primeraPosicion = nombres.indexOf(nombre);

texto("(-) Encontrado " + nombre + " en la primera posición = " + primeraPosicion, 1);

var ultimaPosicion = nombres.lastIndexOf(nombre);

texto("(-) Encontrado " + nombre + " en la última posición = " + ultimaPosicion, 2);
```

A continuación, el código en ejecución.

(::) Se busca un nombre en la lista =
(-) David,Miguel,Sofía,Miguel,Teresa,Miguel,Ramón,Carlos,Miguel,María

(-) El nombre a buscar es: Miguel
(-) Encontrado Miguel en la primera posición = 1
(-) Encontrado Miguel en la última posición = 8

Los siguientes códigos cumplen con la función de unir arrays y obtener el string equivalente a un array respectivamente utilizando los mismo elementos dentro del array esto gracias a los comandos Join y toString.

```
var nombres_union = nombres.join();

texto("(::) Unión de un array utilizando join. Todos sus elementos se unen en una sola cadena: ", 2)
texto(nombres_union, 2)

nombres_union = nombres.toString();
texto("(::) Utilización del método toString para obtener el string equivalente de un array", 2);
texto(nombres_union, 2);
```

A continuación, los dos códigos en ejecución.

```
------------------------------------------------------------
(::) Unión de un array utilizando join. Todos sus elementos se unen en una sola cadena:

David,Sofía,Ramón,Carlos,Miguel,María

============================================================
(::) Utilización del método toString para obtener el string equivalente de un array

David,Sofía,Ramón,Carlos,Miguel,María
```

En este código se concatenan dos arrays, es decir se unen los elementos de dos arrays.

```
var cadena_1 = ["Miguel", "Carlos", "Antonio", "Daniel", "María"];
var cadena_2 = ["Ramón", "Sofía", "Bernardo"];

texto("(::) Concatenación de arrays utilizando concat",  2);

texto("(-) Primer array  = " + cadena_1, 1);
texto("(-) Segundo array = " + cadena_2, 2);

texto("(-) Primera forma de concatenacion. cadena_2 se concatena a cadena_1", 1);
var concatenacion = cadena_1.concat(cadena_2);
texto("(-) " + concatenacion, 2);

concatenacion = cadena_2.concat(cadena_1);

texto("(-) Segunda forma de concatenacion. cadena_1 se concatena a cadena_2", 1);
texto("(-) " + concatenacion, 2);
```

A continuación el programa en ejecucion.

```
============================================================
(::) Concatenación de arrays utilizando concat

(-) Primer array = Miguel,Carlos,Antonio,Daniel,María
(-) Segundo array = Ramón,Sofía,Bernardo

(-) Primera forma de concatenacion. cadena_2 se concatena a cadena_1
(-) Miguel,Carlos,Antonio,Daniel,María,Ramón,Sofía,Bernardo

(-) Segunda forma de concatenacion. cadena_1 se concatena a cadena_2
(-) Ramón,Sofía,Bernardo,Miguel,Carlos,Antonio,Daniel,María
```

Este programa divide un array mediante el comando Splice, lo que hace es que los elementos del array los extrae y los pone en otro array en distinto orden y luego hace lo mismo con los elementos restantes.

```
var concatenacion = ["Miguel","Carlos","Antonio","Ramón","Sofía","Daniel","María"];

texto("(::) Ilustración del método de división por partes utilizando splice", 2)
texto("(-) El array original = " + concatenacion, 1);

var unaParte = concatenacion.splice(3,3);

var otraParte = concatenacion;

texto("(-) Una parte splice(3,3) = " + unaParte, 1);
texto("(-) Otra parte (lo que sobra) = " + otraParte, 2);
```

A continuación el código en ejecución.

```
================================================================
(::) Ilustración del método de división por partes utilizando splice

(-) El array original = Miguel,Carlos,Antonio,Ramón,Sofía,Daniel,María
(-) Una parte splice(3,3) = Ramón,Sofía,Daniel
(-) Otra parte (lo que sobra) = Miguel,Carlos,Antonio,María
```

Los tres siguientes programas cumplen con funciones similares de formas mas eficientes que otras, con códigos mas extensos que otros, pero en si cumplen funciones similares, a continuación, se presentan los 3 códigos para que sean comparados.

```
var numeros = [1,2,3,4,5];
texto("(::) Agregando elementos a un array (al final)", 2)
texto("(-) Array antes de agregar un elemento al final: " + numeros, 1); // 1,2,3,4,5
numeros.push(6);
texto("(-) Array después de agregar el elemento: " + numeros, 2); // 1,2,3,4,5,6
var numeros = [2,3,4,5];
texto("(::) Agregando nuevo número al comienzo del array (método ineficiente)", 2);
texto("(-) Array inicial: " + numeros, 1);
var nuevo_numero = 1;
var N = numeros.length;
for (var i = N; i >= 0; --i) {
numeros[i] = numeros[i-1];
}
numeros[0] = nuevo_numero;
texto("(-) Adición del valor 1 al comienzo del array. Números = " + numeros, 2); // 1,2,3,4,5
texto("(::) Agregando elementos al comienzo de un array. Utlización de unshift", 2);
var numeros = [2,3,4,5];
texto("(-) Números inicial = " + numeros, 1); // 2,3,4,5
var nuevo_numero = 1;
numeros.unshift(nuevo_numero);
texto("(-) Agregado el 1: " + numeros, 2); // 1,2,3,4,5
numeros = [3,4,5];
texto("(-) Números inicial = " + numeros, 1); // 3,4,5
numeros.unshift(1,2);
texto("(-) Agregados dos valores, el 1 y el 2. Nuevo array = " + numeros, 2); // 1,2,3,4,5
```

A continuación, los tres programas en ejecución.

```
============================================================
(::) Agregando elementos a un array (al final)

(-) Array antes de agregar un elemento al final: 1,2,3,4,5
(-) Array después de agregar el elemento: 1,2,3,4,5,6

============================================================
(::) Agregando nuevo número al comienzo del array (método ineficiente)

(-) Array inicial: 2,3,4,5
(-) Adición del valor 1 al comienzo del array. Números = 1,2,3,4,5

============================================================
(::) Agregando elementos al comienzo de un array. Utlización de unshift

(-) Números inicial = 2,3,4,5
(-) Agregado el 1: 1,2,3,4,5

(-) Números inicial = 3,4,5
(-) Agregados dos valores, el 1 y el 2. Nuevo array = 1,2,3,4,5
```

Estos codigos corresponden a programas con funciones similares, igual que los anteriores donde se modifican arrayas a traves de distintos comandos y se comparan por su eficiencia o el poco código que se utiliza y muestran como funciona en la imagen siguiente.

```
texto("(::) Removiendo un dato al final del array", 2);
texto("(-) Array original = " + numeros, 1);
numeros.pop();
texto("(-) Array sin el último elemento: " + numeros, 2); // 1,2,3,4,5
var numeros = [9,1,2,3,4,5];
texto("(::) Removiendo elementos del comienzo de un array. Se utiliza 'unshift'", 2);

texto("(-) Valor original de Números = " + numeros, 1);
numeros.shift();
texto("(-) Se elimina el primer elemento del array: " + numeros, 2); // 1,2,3,4,5

var numeros = [1,2,3,7,8,9];
texto("(::) Agregando elementos en la mitad de un array. ", 2);
texto("(-) Array Números (antes) = " + numeros, 1); // 1,2,3,7,8,9

var nuevos_elementos = [4,5,6];
numeros.splice(3,0,nuevos_elementos);

texto("(-) Números (después). Se agrega [4,5,6] a partir de la tercera posición = " + numeros, 2); // 1,2,3,4,5,6,7,8,9

var numeros = [1,2,3,7,8,9];
texto("(::) Agregando elementos en la mitad de un array. Método directo, agregando los valores en la instrucción: ", 2);
texto("(-) Números (antes) = " + numeros, 1); // 1,2,3,7,8,9

numeros.splice(3,0,4,5,6);
texto("(-) Números (después) = " + numeros, 2); // 1,2,3,4,5,6,7,8,9

var numeros = [1,2,3,100,200,300,400,4,5];

texto("(::) Eliminando elementos en la mitad de un array. Se utiliza splice, según se aprecia en la instrucción", 2);
texto("(-) Números (antes) = " + numeros, 1); // 1,2,3,100,200,300,400,4,5
numeros.splice(3,4);
texto("(-) Números (después) = " + numeros, 2); // 1,2,3,4,5
```

```
==========================================================
:) Removiendo un dato al final del array

) Array original = 1,2,3,4,5,9
) Array sin el último elemento: 1,2,3,4,5

==========================================================
:) Removiendo elementos del comienzo de un array. Se utiliza 'unshift'

) Valor original de Números = 9,1,2,3,4,5
) Se elimina el primer elemento del array: 1,2,3,4,5

==========================================================
:) Agregando elementos en la mitad de un array.

) Array Números (antes) = 1,2,3,7,8,9
) Números (después). Se agrega [4,5,6] a partir de la tercera posición = 1,2,3,4,5,6,7,8,9

==========================================================
:) Agregando elementos en la mitad de un array. Método directo, agregando los valores en la instrucción:

) Números (antes) = 1,2,3,7,8,9
) Números (después) = 1,2,3,4,5,6,7,8,9

==========================================================
:) Eliminando elementos en la mitad de un array. Se utiliza splice, según se aprecia en la instrucción

) Números (antes) = 1,2,3,100,200,300,400,4,5
) Números (después) = 1,2,3,4,5
```

El siguiente código se invierte un array, donde se reciben los elementos del array y luego se invierten.

```
var numeros = [1,2,3,4,5];
texto("(::) Invirtiendo arrays: Número = " + numeros, 2);

numeros.reverse();

texto("(-) Array invertido: " + numeros, 2); // 5,4,3,2,1
```

A continuacion el código en ejecucion.

```
==========================================================
(::) Invirtiendo arrays: Número = 1,2,3,4,5

(-) Array invertido: 5,4,3,2,1
```

El siguiente código cumple con la función de ordenar de manera alfabéticamente los elementos de un array.

```
var nombres = ["David","Miguel","Carlos","María","Bernardo","Ramón"];
texto("(::) Ordenando el array = " + nombres, 2);

nombres.sort();
texto("(-) Array ordenado = " + nombres, 2);
```

Codigo ejecutado.

```
============================================================
(::) Ordenando el array = David,Miguel,Carlos,María,Bernardo,Ramón

(-) Array ordenado = Bernardo,Carlos,David,María,Miguel,Ramón
```

Los dos codigos siguientes son la referencia de como se puede ordenar un array en este caso de números de manera ascendente el primer programa tiene un fallo y en el segundo ya se muestran los números en su respectivo orden gracias a las mejoras que tiene donde se expresan una función que compara los números entre si y va eligiendo secuencialmente el menor.

```
texto("(::) Ordenamiento de números. En su forma básica, los números se ordenan mal", 2);

var numeros = [3,1,2,100,4,200];
texto("(-) Array original = " + numeros, 1);

numeros.sort();
texto("(-) Array ordenado = " + numeros, 2); // 1,100,2,200,3,4

////////////////////////////////////////////
// Ordenando un array (números) - Corregido //
////////////////////////////////////////////

// Se debe utilizar una función de comparación
function comparar(num1, num2) {
return num1 - num2;
}

texto("(::) Ordenamiento numérico corregido", 2);

var numeros = [3,1,2,100,4,200];

texto("(-) Array original = " + numeros, 1);

numeros.sort(comparar);

texto("(-) Array ordenado = " + numeros, 2);   // 1,2,3,4,100,200
```

```
==========================================================
(::) Ordenamiento de números. En su forma básica, los números se ordenan mal

(-) Array original = 3,1,2,100,4,200
(-) Array ordenado = 1,100,2,200,3,4

==========================================================
(::) Ordenamiento numérico corregido

(-) Array original = 3,1,2,100,4,200
(-) Array ordenado = 1,2,3,4,100,200

----------------------------------------------------------
```

El siguiente código representa un programa elaborado con iteración utilizando el comando Foreach donde va recibir los elementos de un array y va calcular el cuadrado de estos números como se aprecia a continuación.

```
function cuadrado(numero) {
    texto(numero.toString() + " --- " + (numero * numero), 1);
}
var numeros = [1,2,3,4,5,6,7,8,9,10];

texto("(::) Cuadrados, utilizando forEach", 2);

numeros.forEach(cuadrado);
```

```
====================================
(::) Cuadrados, utilizando forEach

1 --- 1
2 --- 4
3 --- 9
4 --- 16
5 --- 25
6 --- 36
7 --- 49
8 --- 64
9 --- 81
10 --- 100
```

```
function esPar(numero) {
    return numero % 2 == 0;
}

var numeros = [1,2,3,4,5,6,7,8,9,10];
var algunoPar = numeros.some(esPar);

texto("(::) Verificación de si algún número es par", 2)

texto("(-) Array de números = " + numeros, 1);
if (algunoPar) {
    texto("(-) Algunos números son par", 2);
}
else {
    texto("(-) Ningún número es par", 2);
}
numeros = [1,3,5,7,9];
algunoPar = numeros.some(esPar);

texto("(-) Array de números = " + numeros, 1);
if (algunoPar) {
    texto("(-) Algunos números son par", 2);
}
else {
    texto("(-) Ningún número es par", 2);
}
```

Este código cumple con la función de determinar si elementos de un array es par o no, como se muestra a continuación.

==========

(::) Verificación de si algún número es par

(-) Array de números = 1,2,3,4,5,6,7,8,9,10
(-) Algunos números son par

(-) Array de números = 1,3,5,7,9
(-) Ningún número es par

4 ARRAYS-02

```
// Función para imprimir texto en la pantalla
function texto( cadena, lineas_post = 0 ) {
  if (typeof(cadena) == "string") {
    if (cadena.indexOf("(::)") == 0) {
      document.write("============================================================<br>");
    }
  }
  document.write( cadena );
  for (var i = 0; i < lineas_post; i++) {
    document.write("<br>");
  }
}

function tabla( arr ) {
    for (var i = 0; i < arr.length; i++) {
        for (var j = 0; j < arr[i].length; j++) {
            texto( arr[i][j] + " -- ");
        }
        texto("", 1);
    }
}

texto("(::) TALLER SOBRE MANEJO INTEGRAL DE ARRAYS EN JAVASCRIPT - PARTE 2", 2);

texto("(::) DEFINICIÓN Y USO DE ARREGLOS EN DOS DIMENSIONES", 2);

var arreglo1 = ["Juan", 24, 18000];     // Fila 0
```

```
// Juan : Columna 0, 24 : Columna 1, 18000 : Columna 2

// Juan está en: [Fila 0][Columna 0]

// Correctamente escrito: Juan está en: [0][0]

// 24 está en: [0][1]
// 18000 está en: [0][2]

var arreglo2 = ["Ana", 30, 30000];       // Fila 1
var arreglo3 = ["Teresa", 28, 41000];    // Fila 2
var arreglo4 = ["Carlos", 31, 28000];    // Fila 3
var arreglo5 = ["Antonio", 29, 35000];   // Fila 4

var salario = [arreglo1, arreglo2, arreglo3, arreglo4, arreglo5];

var notas = [

    ['Juan', 4],
    ['Sofía', 5],
    ['Carlos', 3],
    ['Alberto', 5],
    ['María', 5]
```

```
texto("(-) EJEMPLO: Acceder al arreglo3 en la posición 1 = " + salario[2][1], 2);
texto("(-) A) Cada variable definida con el nombre antecedente: 'arreglo'", 1);
texto("(-) es un array. Y dicho array se coloca dentro de otro array (salario)", 2);

texto("(-) B) No olvidar que todos los valores en los array", 1);
texto("(-) se cuentan empezando en 0. Es decir, 0, 1, 2, etcétera", 2);

texto("(-) C) El arreglo 3 fue colocado en la fila 2. Este valor se", 1);
texto("(-) obtiene al ver que arreglo1 se colocó en la fila 0, ", 1);
texto("(-) arreglo2 se colocó en la fila 1, y arreglo 3", 1);
texto("(-) se colocó en la fila 2. Por eso se tiene: salario[2]", 2);

texto("(-) D) Al decir que se accede al arreglo 3 en la posición 1,", 1);
texto("(-) estamos hablando de acceder a una columna. Y como las columnas ", 1);
texto("(-) están numeradas empezando en 0, se tiene: posición 0 sería", 1);
texto("(-) la columna cero, y posición 1 sería la columna 1. Entonces:", 2);

texto("(-) arreglo3 en la posición 1 equivale a: salario[2][1]", 2);
```

```
texto("(::) IMPRIMIENDO UN ARREGLO EN DOS DIMENSIONES", 2);

// Este ciclo es para las filas
for (var fila = 0; fila < salario.length; fila++) {

    // Este ciclo es para las columnas
    for (var columna = 0; columna < salario[fila].length; columna++) {

        // Aquí se accede a cada elemento del array
        texto( salario[fila][columna] + " --- " );

    }
    texto("", 1);
}

texto("(::) AGREGANDO ELEMENTOS", 2);

texto("(-) Agregando un elemento al final:", 1);
texto("(-) salario.push(['Julio', 32, 35000]); ", 2);

salario.push(['Julio', 32, 35000]);

texto("(::) ELIMINANDO ELEMENTOS", 2);
texto("(-) Eliminando el elemento final:", 1);
texto("(-) salario.pop();", 2);

salario.pop();

texto("(::) FUNCIONES GENERALES", 2);
texto("(-) Javascript no posee manejo de arreglos bidimensionales", 1);
texto("(-) de manera similar al manejo unidimensional", 1);
texto("(-) Por este motivo, se deben crear funciones según se requiera", 2);
```

```
texto("(::) O B J E T O S", 2);

// Un objeto se define en Javascript utilizando una función

// El siguiente objeto es un punto, con sus coordenadas (x, y)
// en el plano cartesiano

// Los nombres de los objetos tienen la primera letra en mayúscula
// (esto es una convención, no es obligatorio)

// Todos los objetos poseen en su interior datos y otras funciones
// denominados métodos

// Para el caso inicial que nos ocupa, únicamente utilizaremos dos datos:
// La coordenada x
// La coordenada y

function Punto(x,y) {
    // Explicación: this.x es el nombre de las variables dentro del objeto
    //
    // Los nombres de todas las variables internas de un objeto deben
    // poseer la frase: this.

    this.x = x;
    this.y = y;
}

// Se crean cuatro puntos
var p1 = new Punto(1,2);
var p2 = new Punto(3,5);
var p3 = new Punto(2,8);
var p4 = new Punto(4,4);
```

```
// La función: dibujarPuntos, presenta los puntos contenidos
// en un array (el cual está conformado por puntos)
function dibujarPuntos(arr) {
    // Se recorren todos los puntos contenidos en el arreglo recibido
    for (var i = 0; i < arr.length; ++i) {
        // Se muestra el punto iésimo en la pantalla
        texto("Punto " + parseInt(i+1) + "= " + arr[i].x + ", " + arr[i].y, 1);
    }
}

// Se agrega un nuevo punto, al final del arreglo 'puntos'

// Se define el nuevo punto. El punto es p5 con la coordenada (12, -3)
var p5 = new Punto(12,-3);

// Agrega el punto p5 al final del arreglo
puntos.push(p5);

texto("Después de agregar el punto (12, -3) ", 2);

// Se muestran todos los puntos en la pantalla de la computadora
// En el array ya se incluye el último punto adicionado
dibujarPuntos(puntos);

texto("", 1);

// Con esta instrucción se elimina el primer elemento del arreglo 'puntos'
puntos.shift();

texto("Después de extraer el primero ", 2);

// Se muestra el array con el punto eliminado
dibujarPuntos(puntos);
```

Este programa representa lo relacionado con la definición y uso de los arreglos en dos dimensiones, donde se crean funciones ya que Javascript no cuenta con el manejo de arreglos bidimensionales.