RESUMEN POR LEANDRO AMAYA

SE PUEDE AGREGAR COSAS , PERO NO ELIMINAR

# JAVASCRIPT

Java y JavasCript son lo mismo?

NO, las unicas similitudes son el nombre y partes de su sintaxys(“”, = , if , etc)

Pero sirven , funcionan y son de tipos de lenguaje diferente (Pregunta de Entrevista)

JavaScript Caracteristicas.

-JavaScript es un lenguaje interpretado, a diferencia de java que hace un proceso de

Convertir el codigo en codigo maquina y luego compilarlo, JavaScript puede compilarse sin tener

Que convertirse en lenguaje maquina

-JavaScript pose tipos de datos variables, es decir que el programa interpreta el tipo de dato de la variable y no hace falta especificar el tipo de dato tiene preferencia a elejir String cuando puede

(number,String,bigint,etc)

-en JavaScritp un arreglo puede constar de varios valores, y a su vez esos valores ser de diferentes tipos de datos

-Javascript al igual que java pose estructuras de control (lenguaje de paradigmas)

If,else // switch // for, for in , for of // while , do while

-JavaScript al igual que java pose TRY-CATCH lo que permite manejar errores

-JavaScritp simula el POO(lenguaje orientado a objetos) por lo que pose objetos y funciones (function)

function (parametro a) { return a + b; }

Las funciones pueden guardar su retorno en una variable

var\_almacenar = function (parametro a) { return a + b; }

N: no se tiene que especificar tipo de dato de parametros , ni retorno ni variable que almacena

-JavasScript pose un objeto de tipo DATE

# Libro java:

# Teoria

-Expresion: Un fragmento de código que produce un valor se llama una expresión

...Cualquier cosa que produzca un valor es una expresión en JavaScript

-Funcion:  Una función es un fragmento de programa envuelto en un valor

Ejecutar una función se llama invocar, llamar, o aplicar la función.

-Entorno: La colección de enlaces y sus valores que existen en un momento dado se llama entorno

-Argumentos: Los valores dados a las funciones se llaman argumentos

-Valor de retorno: Cuando una función produce un valor, se dice que retorna ese valor

-Ejecucion condicional: carretera ramificada donde el programa tome la rama adecuada basada en la situación en cuestión

-Control de flujo bucle: manera de ejecutar un fragmento de código múltiples veces.

-La instrucción break tiene el efecto de salir inmediatamente del bucle que la contiene

-palabra clave continue Cuando se encuentra continue en el cuerpo de un bucle, el control salta fuera del cuerpo y continúa con la siguiente iteración del bucle.

-En el caso de preferir un switch a un if/else hay que considerar el case de un switch Continuará ejecutándose, ... hasta que alcance una declaración break

-Un comentario es un fragmento de texto que forma parte de un programa pero que es completamente ignorado por la computadora (// o /\*\*/)

-puedes encontrar la longitud de una cadena escribiendo .length (variable.length)

-Una instrucción return determina el valor que devuelve la función

-Cada enlace tiene un ámbito, que es la parte del programa en la que el enlace es visible (enlace = variable) (existen enlaces globales y enlaces locales)

-múltiples grados de localidad (funciones dentro de otras funciones,etc)

- Funciones crean su ambito y pueden ver ambitos que los rodean (otras funciones)

## - Funciones de flecha const variable = (parametros)=>{}se lee “esta entrada (los parámetros) produce este resultado (el cuerpo)”.

 -Una función que hace referencia a enlaces de ámbitos locales a su alrededor se llama una clausura.

-Una función que se llama a sí misma se llama recursiva.

La recursividad es 3 veces mas lenta que los bucles for

# Caracteisticas

var&&var (es true cuando ambas son true)

Var // var (es true cuando una sea true)

Var ?? var (toma la izquierda como valor a menos que sea nulo o indefinido)

Cuando solo se le el primer var y se decide el resultado sin el segundo por reglas de los operadores Esa omision se llama evaluacion de cortocircuito

Las funciones siempre retornan undifined a menos que se les especifique otro return

Cuando se pasa como argumento diez(a) {a = 10} y a se le pone a= 5 , function diez(a)

No se modifica. Solo se modifican tipos primitivos y objetos que son seteados espeficicamente sus atributos en el metodo

Se pueden pasar por parametro mayores argumentos de los que se nesesitan dentro del bloque del metodo (a,b,c,d,e,f,g,etc)

Tambien se usan para valores por defecto

El type de una funcion siempre sera una funcion

Consol.log(arguments) = retorna los argumentos que se les dio a la funcion

Los arreglos son objetos

Los arreglos tienen metodos para agregar, sumar , eliminar , etc items del arreglo

innerHTML no es igual a text cont (inner crea html y text content solo pone texto)

Las funciones pueden ser llamadas antes o despues de declararse(no importa el orden )

Matriz variable = {

“arreglo”: [ {“string uno”}{”string dos”}]

}

# DOM

El **DOM** es la representación en memoria del documento HTML como un **árbol de nodos**. Permite que JavaScript lea y modifique la estructura, el contenido y el estilo de la página en tiempo real.

**- Nodo / Elemento**: cada etiqueta HTML es un nodo

Se utiliza una interfaz llamada document ( funciona como referencia al mismo documento html )

para acceder a elementos del html

Syntaxis:

Seleccionar por ID

const elTitulo = document.getElementById('titulo');

Seleccionar por clase (devuelve una colección HTML)

const misConts = document.getElementsByClassName('cont');

Seleccionar por etiqueta (devuelve una colección HTML)

const misParrafos = document.getElementsByTagName('p');

Seleccionar por selector CSS (devuelve el primero que encuentre)

const primerParrafo = document.querySelector('.parrafo');

Seleccionar TODOS los que coincidan (devuelve una lista estática de nodos)

const todosParrafos = document.querySelectorAll('.parrafo');

N: para selecionar ej java cuando uses selector all se seleciona como un arreglo por posiciones

variable[posicionNumerica]

.classList

Se usa para obtener las clases de un elemento html

List.add(“estilo que quieres añadir”)

List.remove(“estilo que quieres eliminar”)

.ClassName

Cambia todas las clases que tiene el elemento por las que escribas (no se puede una por una)

elemento.className = ‘clase / clases’

# InnerHTML

Se usa para modificar contenido dentro de un html

Ejemplo :

<p id=”salu”> Hola </p>

Let parrafo = document.getElementbyId(“salu”)

parrafo.innerHTML = “Chau”

# Attribute (attribute= elejir atributo)

Se usa para modificar un valor dentro de un atributo

Ejemplo :

<img id=”img” (src = “img1.png”)>

Let imagen= document.getElementbyId(“img”)

imagen.src= “ img2.png”

# style.property (property= elejir propiedad)

Se usa para modificar el estilo de un elemento html

Ejemplo :

<h1 <style> = colur: red <style> id=”col”> property </h1>

Let color= document.getElementbyId(“col”)

imagen.style.color = “red”