Actividad:

Resolución a problemas algorítmicos aplicando estructuras de almacenamiento

GA3-220501093-AA3-EV01

Aprendiz:

Wilmer Jair Espinosa Silva

CC: 1.095.910.391

Instructor:

ISRAEL ARBONA GUERRERO

Servicio Nacional de aprendizaje-SENA

Curso: TECNOLOGÍA EN ANÁLISIS Y DESARROLLO DE SOFTWARE

Ficha: 2455285

Evidencia de conocimiento: GA3-220501093-AA3-EV01 bases teóricas de estructuras de almacenamiento en memoria

Esta evidencia se centra en consolidar los conceptos básicos relacionados con los lenguajes de programación, entornos de codificación e instalación y la sintaxis del lenguaje de JavaScript.

Para su desarrollo es importante la lectura del componente formativo. Elaborar un documento en el cual se registren los siguientes elementos:

Principales diferencias entre los lenguajes compilados e interpretados.

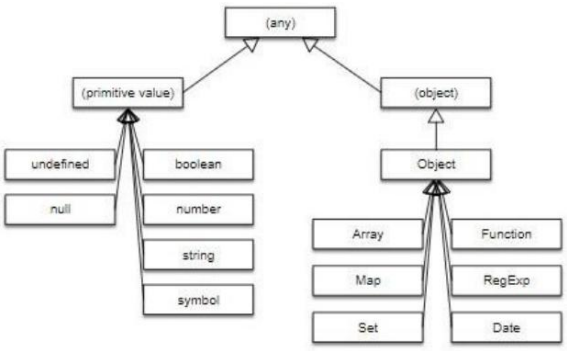
LENGUAJES COMPILADOS	LENGUAJES INTERPRETADOS
Un lenguaje compilado generauna fila binario no modificable.	Un lenguaje interpretado es escrito enun lenguaje de programación definidoy modificable en cada momento.
Las instrucciones vienen enviadas directamente al procesador.	Las instrucciones deben traducirseantes de llegar al procesador.
Se requieren dos pasos separados para ejecutar el programa desde el código fuente.	El código fuente se ejecuta a travésde un solo comando.
Dado que el programa ya se ha traducido, la ejecución es más rápida.	El programa debe traducirse cada vez aumentando el tiempo de ejecución.
El programa solo se puede ejecutar en ciertas máquinas y sistemas operativos.	El programa funciona en todas las máquinas y sistemas.

Los errores de compilación impiden que se compile el código	Los errores de compilación son visibles solo si se inicia el programa.
Ejemplos de lenguajes compilados son C, C++, Delphi	Ejemplos de lenguajes interpretadosson Python, JavaScript, Perl, PHP

Características principales de JavaScript

Su sintaxis es similar a la de Java, débilmente tipado, es case sensitivo, no existen las constantes, basado no orientado a objetos. <Universalidad= de usos, seguridad de ejecución, páginas más ligeras de cargar.

Tipos de datos primitivos y uso en JavaScript




```
typeof "hola!" // "string"
typeof 42 // "number"
typeof true // "boolean"
typeof null // "object" ???
typeof undefined // "undefined"
typeof Symbol // "symbol"
typeof   n // "bigint"
```

Operadores en JavaScript

Los operadores permiten manipular las variables, realizar operaciones matemáticas, comparaciones lógicas o asignaciones.

Existen varios tipos de operadores

Operador de Asignación: Asigna un valor en nuestras variables.

```
const variableA = 5;
let variableB = 'Hola';
variableB = 10;
```

Operador de Incremento y Decremento: Permite incrementar o decrementar en una unidad el valor de la variable.

```
//incremento
let contador = 0;
contador++;
console.log(contador)                    // 2


//decremento
let contador = 5;
contador--;
console.log(contador)                    // 4
```

Operadores Lógicos: Nos permite tomar decisiones sobre las instrucciones incluso nos permite negar una instrucción.

NEGACIÓN(!)

```
const isVisible = true;
console.log(!isVisible)                  // false
```

AND(&&)

```
const valor1 = true;
const valor2 = false;
console.log(valor1 && valor2)            // false
```

OR(II)

```
const valor1 = true;
const valor2 = false;
console.log(valor1 || valor2)            // true
```

Operadores Relacionales: Las usamos para evaluar expresiones.

```
const numero1 = 3;
const numero2 = 5;
let resultado;
resultado = numero1 > numero2;
resultado = numero1 < numero2;
resultado = numero1 >= numero2;
resultado = numero1 <= numero2;
```

Operadores de Igualdad: Existe estricta se evalúa la expresión sea igual o diferente, el tipo y el valor y no estricta se evalúa solo el valor.

```
console.log( 0 == 0);                          // true
console.log( "" == 0 );                        // true
console.log( false == 0);                      // true
console.log( undefined != 0);                  // true
console.log( null == 0);                       // false


console.log('1' === 1);                        // false
console.log(2 === 2);                          // true
console.log('abc' !== 123);                    // true
```

Operadores Aritméticos: Evalúan la expresión y devuelve un único resultado.

```
const numero1 = 10;
const numero2 = 5;
let resultado = numero1 / numero2;                  // resultado es 2
resultado = 3 + numero1;                            // resultado es 13
resultado = numero2 - 4;                            // resultado es 1
resultado = numero1 * numero2;                      // resultado es 50
```