Condicionales compuestas

CARLOS MANUEL NUÑEZ LOPEZ

INSTRUCTOR

¿Qué es una condición compuesta?

- Una condición compuesta combina dos o más condiciones simples usando operadores lógicos. Se utilizan comúnmente en estructuras de control como if, while, etc

```
Si (edad >= 18) Y (tieneLicencia == VERDADERO) Entonces
    Mostrar "Puedes conducir"
FinSi
```

Operadores lógicos

- Estos operadores se utilizan para enlazar dos o mas condiciones y se representan de la siguiente forma:
- Y (AND): Devuelve verdadero solo si ambas condiciones son verdaderas. En pseudocódigo, se puede representar con && o Y.-> Ejemplo
  - 2==2 Y 4==4
- O (OR):Devuelve verdadero si al menos una de las condiciones es verdadera. En pseudocódigo, se puede representar con || o O:
  - 2==2 || 8==1
- NO (NOT):Invierte el valor booleano de una condición. Si es verdadero, lo convierte en falso, y viceversa. En pseudocódigo, se puede representar con ! o NO!(2==2)

Ejemplo

- Si un algoritmo necesita verificar si un usuario es mayor de edad (más de 18 años) y tiene un documento de identidad válido, podría usar el operador lógico Y (AND) para combinar estas dos condiciones. Por ejemplo:

```
Si (edad > 18 Y documento_valido) Entonces
    // Acciones a realizar si el usuario cumple con ambas condiciones
FinSi
```

Tabla de la verdad

- Las tablas de la verdad muestran todas las combinaciones posibles de valores lógicos (VERDADERO o FALSO) para los operadores.

Operador `AND` (Y lógico)

A	B	A AND B
VERDADERO	VERDADERO	VERDADERO
VERDADERO	FALSO	FALSO
FALSO	VERDADERO	FALSO
FALSO	FALSO	FALSO

📌 Solo es VERDADERO si **ambas condiciones** lo son.

Operador `OR` (O lógico)

A	B	A OR B
VERDADERO	VERDADERO	VERDADERO
VERDADERO	FALSO	VERDADERO
FALSO	VERDADERO	VERDADERO
FALSO	FALSO	FALSO

📌 Es VERDADERO si **al menos una condición** lo es.

Operador `NOT` (Negación)

A	NOT A
VERDADERO	FALSO
FALSO	VERDADERO

📌 Invierte el valor de la condición.

Ejemplos prácticos

1. Comprobar si un número está entre dos valores

```pseudocode
Si (numero >= 10) Y (numero <= 20) Entonces
    Mostrar "Número válido"
Sino
    Mostrar "Número fuera de rango"
FinSi
```

2. Validar acceso con múltiples condiciones

```pseudocode
Si (usuario == "admin") O (tienePermiso == VERDADERO) Entonces
    Mostrar "Acceso concedido"
Sino
    Mostrar "Acceso denegado"
FinSi
```

Ejercicio para practicar

- Escribe un algoritmo que reciba dos valores: edad y nacionalidad. Si la edad es mayor o igual a 18 y la nacionalidad es “Colombiana", debe imprimir "Puede votar". En otro caso, "No puede votar".

Condicionales simples

CARLOS MANUEL NUÑEZ LOPEZ

INSTRUCTOR

- En programación y algoritmos, una condición simple es una instrucción que evalúa si una expresión lógica es verdadera (true) o falsa (false) para decidir qué camino tomar en el flujo del programa.

¿Qué es una condición?

- Una condición es una comparación entre valores, por ejemplo:
  - a > b
  - x == 10
  - edad >= 18

- Estas comparaciones devuelven un resultado booleano:
  - true si la condición se cumple.
  - false si no se cumple.

Operadores relacionales.

- Los operadores relacionales permiten definir la relación entre 2 o más expresiones.

Operador	Descripción
<	Menor Que
>	Mayor Que
==	Igual Que
<=	Menor o Igual Que
>=	Mayor o Igual Que
!=	Diferente De

- **Expresión1** Operador relacional **Expresión2**
- Donde el resultado de la operación anterior se evalúa en términos de verdadero o falso.

Condición	Resultado	Descripción
A < 4	FALSO	A vale 4 por lo tanto no es Menor Que 4
A > 4	FALSO	A vale 4 por lo tanto no es Mayor Que 4
A == 4	VERDADERO	A vale 4 por lo tanto si es Igual Que 4
A <= 4	VERDADERO	A vale 4 por lo tanto aunque no es menor que 4, si es Igual
A >= 4	VERDADERO	A vale 4 por lo tanto aunque no es Mayor que 4 si es Igual
A != 4	FALSO	A vale 4 por lo no es Diferente De 4

Ejemplo: evalúe las siguientes condiciones para A=5

Condición	Resultado	Descripción
A < 4	FALSO	A vale 5 por lo tanto no es Menor Que 4
A > 4	VERDADERO	A vale 5 por lo tanto si es Mayor Que 4
A == 4	FALSO	A vale 5 por lo tanto no es Igual Que 4
A <= 4	FALSO	A vale 5 por lo tanto no es menor que ni es Igual a 4
A >= 4	VERDADERO	A vale 5 por lo tanto si es Mayor que 4 aunque no es Igual
A != 4	VERDADERO	A vale 5 por lo sí es Diferente De 4

Estructura básica: if

- La estructura condicional más básica es el if (si):

```
si (condición) entonces
    ejecutar instrucciones
fin si
```

Ejemplo práctico: Verificar si un número es positivo

```
Inicio
    Escribir "Ingresa un número:"
    Leer numero

    Si (numero > 0) entonces
        Escribir "El número es positivo."
    Fin Si
Fin
```

Operadores lógicos.

- Los operadores lógicos permiten la combinación de condiciones para formar una sola expresión lógica, para estos permiten obtener un resultado lógico al complementarse con los operadores relacionales.

Operador	Nombre	Descripción
!	Negación (NO)	Al usar este operador si la condición arroja falso, el operador la convierte a verdadero y viceversa
&&	Conjunción (Y)	Es verdadero solo si los 2 (o todos) los operando son verdadero, caso contrario arroja falso
||	Disyunción (O)	Es verdadero cuando al menos un operando es verdadero

operando1 Operador_lógico **operando2**

A	B	!A	!B	A && B	A || B
F	F	V	V	F	F
V	F	F	V	F	V
F	V	V	F	F	V
V	V	F	F	V	V

TABLA DE LA VERDAD

Condicionales Anidados.

- Cuando se trabaja con condicionales, es muy común encontrar casos en los que después de tomar una decisión, se requiera seguir un posible camino donde se tengan que implementar nuevas condiciones, para esto se aplican las estructuras condicionales, donde en cada bloque de SI o SINO, pueden existir nuevas condiciones y dentro de estas nuevos procesos y así sucesivamente.

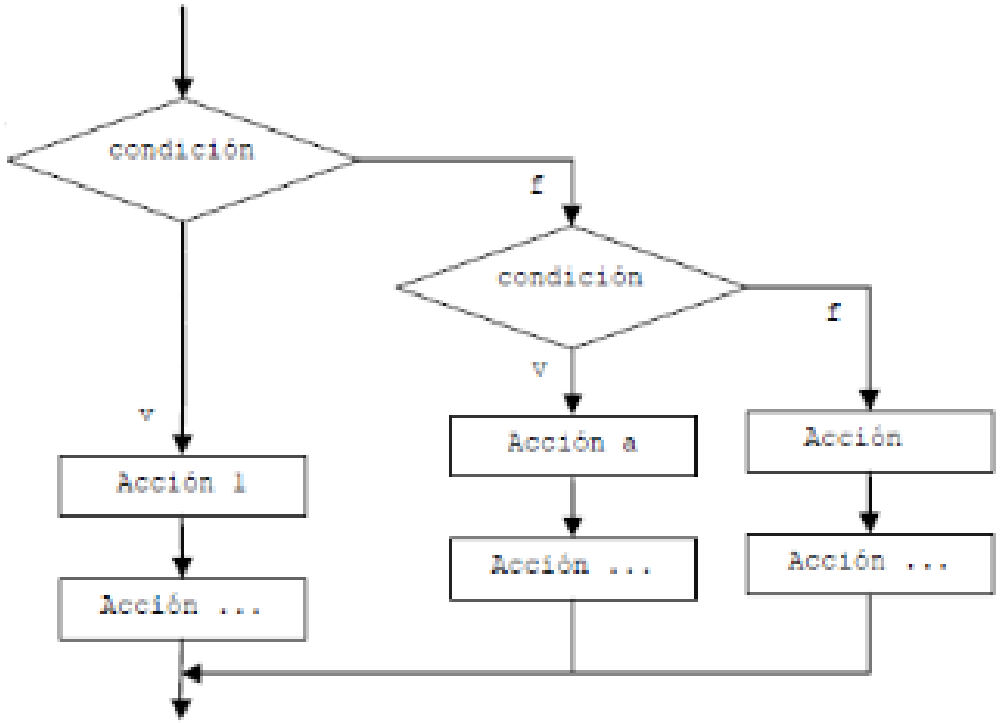
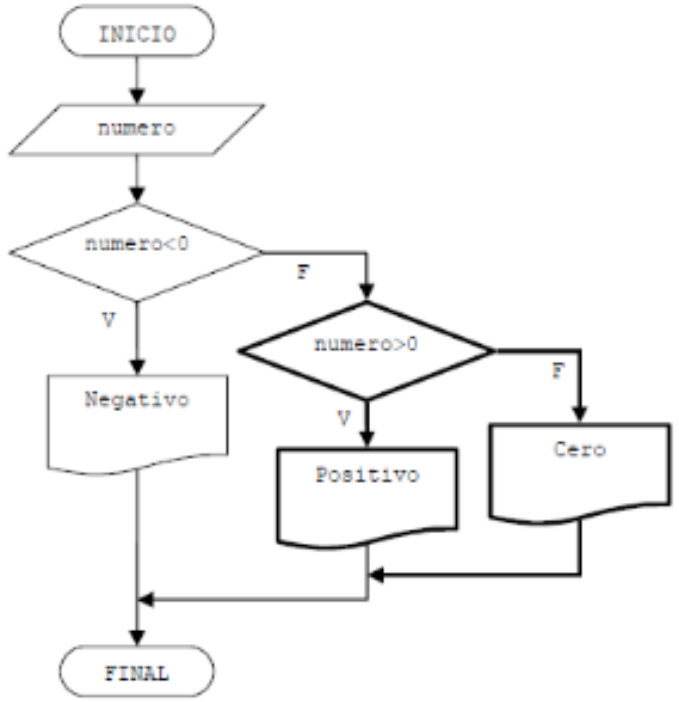
Diagrama de Flujo	Pseudocódigo
	... SI (condición) { Acción 1 Acción ... } SINO { SI (condición) { Acción a Acción ... } SINO { Acción Acción ... } } ...

Diagrama de Flujo	Pseudocódigo

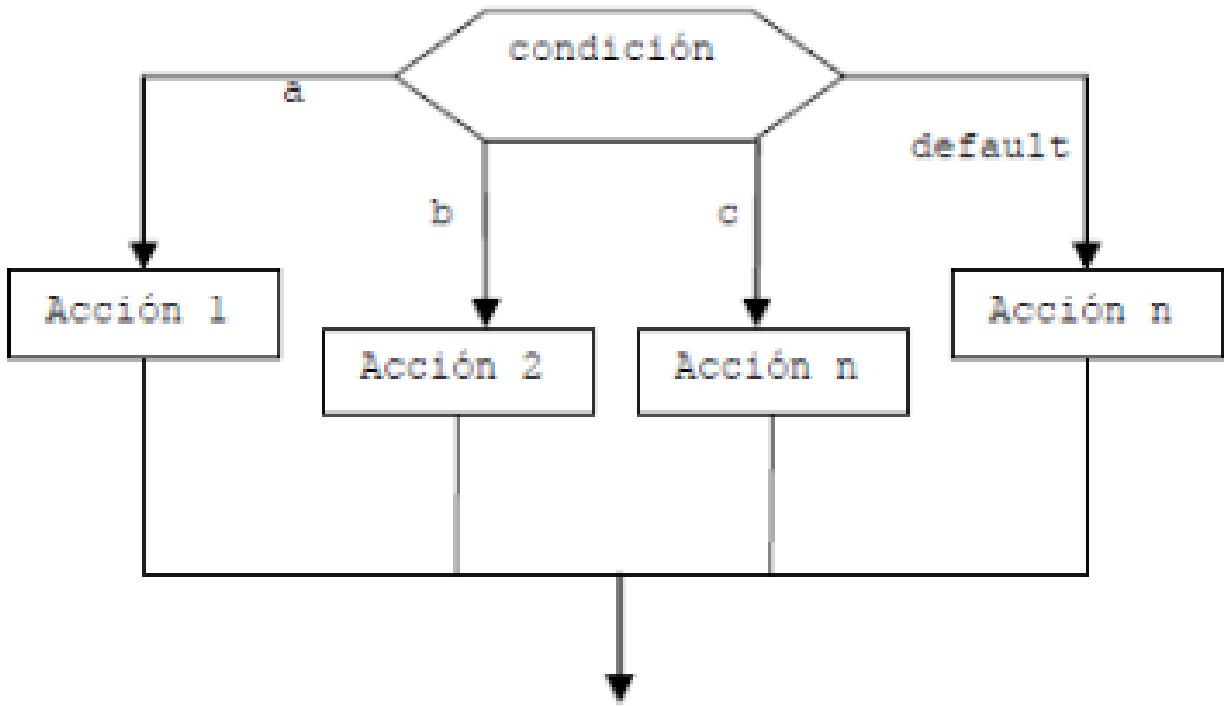
	inicio
	entero numero;
	lea numero;
	si(numero<0)
	{
	imprima "Negativo";
	}
	sino
	{
	si(numero>0)
	{
	imprima "Positivo";
	}
	sino
	{
	imprima "Cero";
	}
	}
	fin

Ejemplos: Algoritmo que determine si un número es positivo, negativo o cero.

Condicionales Múltiples (SEGÚN SEA - CASO) .

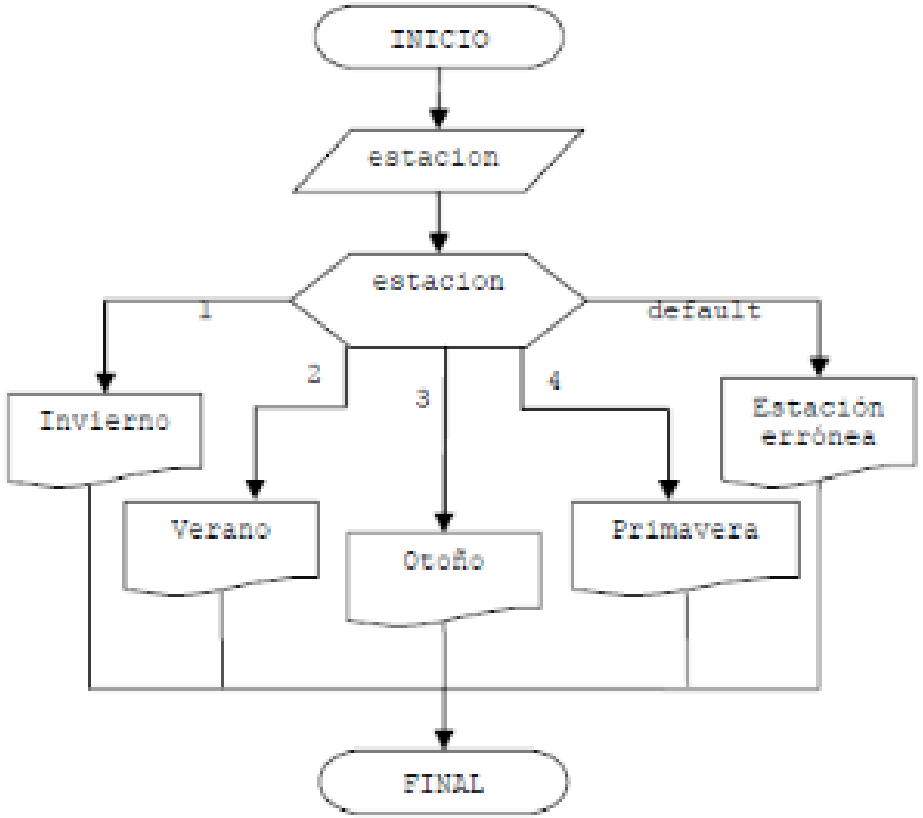
- Este tipo de estructuras permite escoger un camino a seguir dependiendo de una condición (que en este caso se comporta diferente a las condiciones ya vistas) que recibe la estructura Según Sea, la condición se evalúa y dependiendo de su contenido se escoge el flujo con las acciones correspondientes entre la cantidad de caminos posibles.

Diagrama de Flujo	Pseudocódigo

	SEGÚN SEA(condicion)
	{
	CASO a:
	Acción 1
	CASO b:
	Acción 2
	CASO c:
	Acción 3
	DEFAULT:
	Acción n
	
	}

Ejemplo: Cree un algoritmo que, basado en la siguiente tabla, lea un número entero e imprima el nombre de la estación correspondiente.

Código	Estación
1	Invierno
2	Verano
3	Otoño
4	Primavera

Diagrama de Flujo	Pseudocódigo
	


```
inicio
 entero estacion;
 lea estacion;
 segun sea(estacion)
 {
   caso 1: imprima "Invierno";
           break;
   caso 2: imprima "Verano";
           break;
   caso 3: imprima "Otoño";
           break;
   caso 4: imprima "Primavera";
           break;
   default: imprima "Estación errónea";
           break;
 }
fin
```



Introduccion Java Script

CARLOS MANUEL NUÑEZ LOPEZ

INSTRUCTOR

¿Qué es una variable?

Una variable es un espacio en memoria donde se puede guardar información (un valor) que se puede usar y/o cambiar durante la ejecución de un programa.

Analogía: Piensa en una variable como una caja con una etiqueta (el nombre de la variable) donde puedes guardar algo.

Formas modernas de declarar variables

```
let edad = 25;
edad = 30; // ✅ se puede cambiar
```

let: para variables que pueden cambiar

Uso típico: cuando necesitas cambiar el valor más adelante (por ejemplo, en un bucle o cálculo).

Formas modernas de declarar variables

```
const PI = 3.1416;
// PI = 3.15; ❌ Error: no se puede reasignar
```

Uso típico: para valores fijos como números constantes o referencias que no cambiarán.

Tip: Si puedes usar const, úsalo. Es más seguro.

Ejemplo

```
let edad = 20;
console.log("Edad:", edad);
```

```
const GRAVEDAD = 9.8;
console.log("Gravedad:", GRAVEDAD);
```

Tipos de datos

Tipo	Ejemplo
Números	`let x = 10;`
Cadenas	`let nombre = "Ana";`
Booleanos	`let activo = true;`
Arrays	`let lista = [1, 2, 3];`
Objetos	`let persona = { nombre: "Luis", edad: 30 };`
Null	`let x = null;`
Undefined	`let y;` (no tiene valor aún)

¿Qué es un Input y un Output?

- Input (entrada): Es la información que el usuario proporciona al programa.
- Output (salida): Es la información que el programa muestra al usuario.
- Analogía: Tú (el usuario) haces una pregunta (input) al programa, y el programa responde con una salida (output).

INPUT: cómo recibir datos del usuario

- Usando prompt() (modo simple, en el navegador)
- Esto abre una ventana emergente en el navegador donde el usuario puede escribir.
- El dato que escribe se guarda en la variable.
- 🔸 Todos los datos que vienen de prompt son strings.

Ejemplo

```
let edadTexto = prompt("¿Cuál es tu edad?");
let edad = Number(edadTexto); // convierte a número
console.log("Tu edad es:", edad);
```

OUTPUT: cómo mostrar datos

- console.log() – salida a la consola del navegador
- alert() – muestra una ventana emergente

```
console.log("Hola mundo");   alert("Bienvenido al sistema");
```

Ejemplo

```
let nombre = prompt("¿Cómo te llamas?");
let edad = Number(prompt("¿Cuántos años tienes?"));

console.log("Usuario:", nombre);
console.log("Edad:", edad);

alert("Hola " + nombre + ", tienes " + edad + " años.");
```

Buenas practicas

- ✅ Convertir a número si vas a hacer cálculos: `Number(prompt(...))`
- ✅ Usar `console.log()` para depurar mientras programas
- ✅ Usar `alert()` con moderación (puede ser molesto)
- ✅ Evitar confiar 100% en los datos del usuario (validar si es necesario)

¿Qué son las buenas prácticas de programación?

- Son reglas, consejos y hábitos recomendados para escribir código que sea:
- Claro (fácil de entender)
- Consistente (sigue un estilo)
- Mantenible (otros pueden continuar el trabajo)
- Libre de errores comunes.

Nombres de variables claros y significativos

MAL:

```javascript
let x = 5;
```

BIEN:

```javascript
let edadDelUsuario = 5;
```

¿Qué es Camel Case?

- Camel Case es una convención de escritura en la que:
- La primera palabra empieza en minúscula
- Las siguientes empiezan con mayúscula
- Sin espacios ni guiones.
- *Se llama así porque las mayúsculas parecen las jorobas de un camello.*

Incorrecto	Correcto (Camel Case)
`Nombreusuario`	`nombreUsuario`
`Edad_usuario`	`edadUsuario`
`suma_total`	`sumaTotal`

Usa **Camel Case** en:

- Variables → `let numeroIngresado`
- Funciones → `calcularPromedio()`
- Parámetros → `function saludarUsuario(nombreUsuario)`

Indentar correctamente

- El código dentro de bloques { } debe ir sangrado (espaciado hacia adentro).

```
if (edad >= 18) {
  console.log("Eres mayor de edad"); // bien indentado
}
```

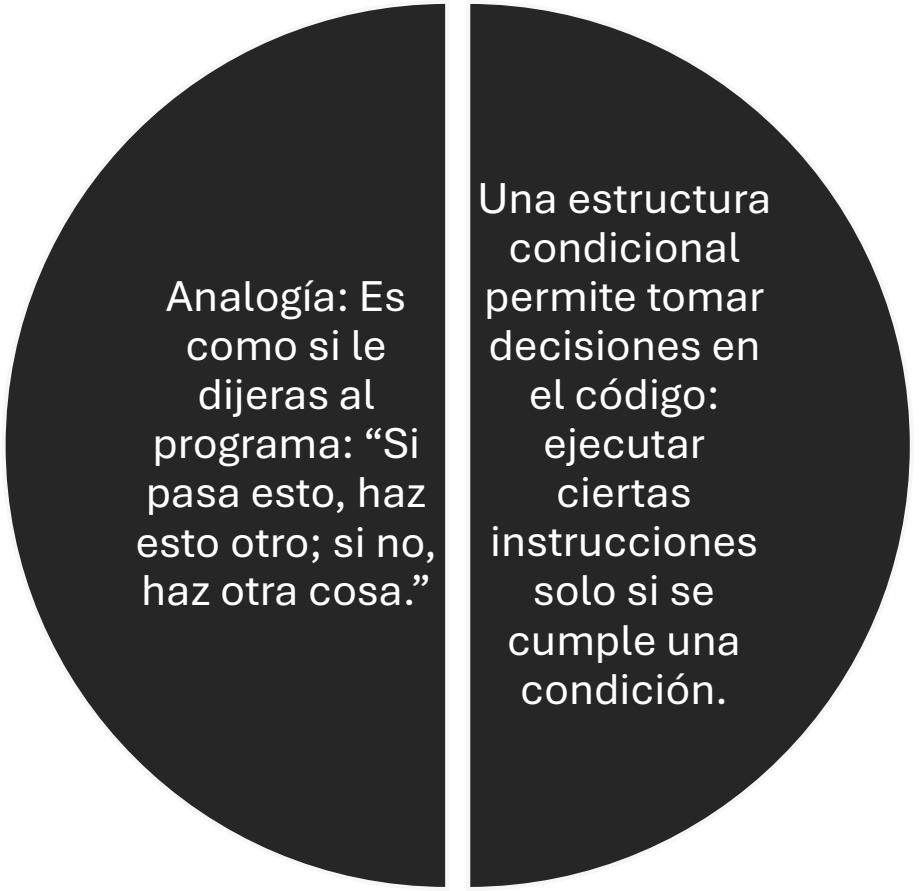
Practica

Adivina mi edad: Declarar una variable `edad`, cambiar su valor, mostrar el resultado.

Calculadora de área: Usar `const PI` y `let radio` para calcular el área de un círculo.

Simulación de registro de usuario: Crear variables con `let` para `nombre`, `email`, `activo`.

¿Qué es una estructura condicional?



Sintaxis básica: if (si)

```
if (condición) {
  // código que se ejecuta si la condición es verdadera
}
```

Ejemplo

```
let edad = 18;
if (edad >= 18) {
  console.log("Eres mayor de edad");
}
```

If - else: alternativa si no se cumple la condición

```
if (condición) {
  // se ejecuta si es verdadera
} else {
  // se ejecuta si es falsa
}
```

Ejemplo

```
let hora = 14;
if (hora < 12) {
  console.log("Buenos días");
} else {
  console.log("Buenas tardes");
}
```

if...else if...else: múltiples condiciones

```
if (condición1) {
  // si se cumple la condición1
} else if (condición2) {
  // si se cumple la condición2
} else {
  // si no se cumple ninguna
}
```

Ejemplo

```
let nota = 8;


if (nota >= 9) {
  console.log("Excelente");
} else if (nota >= 6) {
  console.log("Aprobado");
} else {
  console.log("Desaprobado");
}
```

Comparador	Significado	Ejemplo (`a = 5`, `b = 3`)
`==`	Igualdad (sin tipo)	`a == "5"` → `true`
`===`	Igualdad estricta	`a === "5"` → `false`
`!=`	Distinto	`a != b` → `true`
`!==`	Distinto estricto	`a !== "5"` → `true`
`>`	Mayor que	`a > b` → `true`
`<`	Menor que	`a < b` → `false`
`>=`	Mayor o igual	`a >= 5` → `true`
`<=`	Menor o igual	`a <= 4` → `false`

Operadores logicos

Operator	Description	Example
&&	and	(x < 10 && y > 1) is true
||	or	(x == 5 || y == 5) is false
!	not	!(x == y) is true

Practica

1. Verificar si un número es positivo o negativo.
2. Decidir si un estudiante aprueba con nota ≥ 6.
3. Mostrar un saludo según la hora (mañana, tarde o noche).
4. Usar el operador ternario para mostrar si un número es par o impar.

CARLOS MANUEL NUÑEZ LOPEZ
Instructor

TALLER 1

OBJETIVO

- Desarrollar la lógica de programacion

FORMULACIÓN DE LAS ACTIVIDADES DE APRENDIZAJE

Para reforzar sus conocimientos lógicos, se presentan los siguientes algoritmos de nivel principiante.

- Escribir un Pseudocódigo - la edad y peso de una persona y posteriormente imprimirla.
- Escribir un Pseudocódigo que calcule el área de un triángulo recibiendo como entrada el valor de base y altura.|
- Escribir Pseudocódigo que calcule el área de un círculo.
- Escribir un Pseudocódigo que calcule la conversión de metros a centimetrimetos.
- Elaborar un algoritmo que solicite el numero de respuestas correctas, incorrectas y en blanco, correspondientes a postulantes y muestre su puntaje final considerando, que por cada respuesta correcta tendrá 4 puntos, respuestas incorrectas tendrá -1 punto y respuestas en blanco 0 puntos.
- Se requiere un algoritmo para elaborar una planilla de empleados, para ello se dispone de sus horas laboradas en el mes, asi como la tarifa por hora.
- Elaborar un algoritmo que permita sacar el promedio de una materia, la cual se compone de 4 notas de 1 a 5.

SOLUCION DE EJERCICIO.

- Número **positivo o negativo**
  - Leer un número y mostrar si es positivo, negativo o cero.
- Par **o impar**
  - Leer un número y mostrar si es par o impar.
- Mayor **de edad**
  - Leer la edad de una persona y decir si es mayor de edad (>=18) o menor.
- Comparar **dos números**
  - Leer dos números y mostrar cuál es mayor o si son iguales.
- Múltiplo **de 5**
  - Verificar si un número ingresado es múltiplo de 5.

Calificación aprobatoria

- Leer una nota del 0 al 10 y decir si aprueba (>=6) o no.

Rango de edad

- Determinar si una persona es adulto (18+).

Número dentro de un rango

- Leer un número y verificar si está entre 1 y 100.

Taller 2 Condiciones simples

Carlos Manuel Nuñez
Instructor

Ejercicios de practica para condicionales compuestas.

1. Pide un número. Mostrar "Número válido" si está entre 50 y 100 (inclusive).
2. Pide dos notas. Si ambas son mayores o iguales a 6, mostrar "Aprobado". Si una de ellas es menor, mostrar "Debe recuperar".
3. Pide si es de noche (booleano) y si es fin de semana (booleano). Si al menos una es verdadera, mostrar "Puedes descansar".
4. Pide un número. Si el número es menor que 0 o mayor que 100, mostrar "Fuera de rango"