BOOTCAMP

Generación T

streambe

Objetivos de la clase

1 - Introducción Al Uso De Condicionales Y Estructuras De Control.
2 - Operadores De Comparación.
3 - Estructura condicional Switch().
4 – Intro a Loops
5 – Intro a Funciones
6 - Funciones Con Parámetros

Programación Estructurada: Condicionales y Control de Flujo

streambe GENERACIÓN T

Empezamos…

streambe GENERACIÓN T

Condicionales: La Base de la Decisión

¿Qué son?

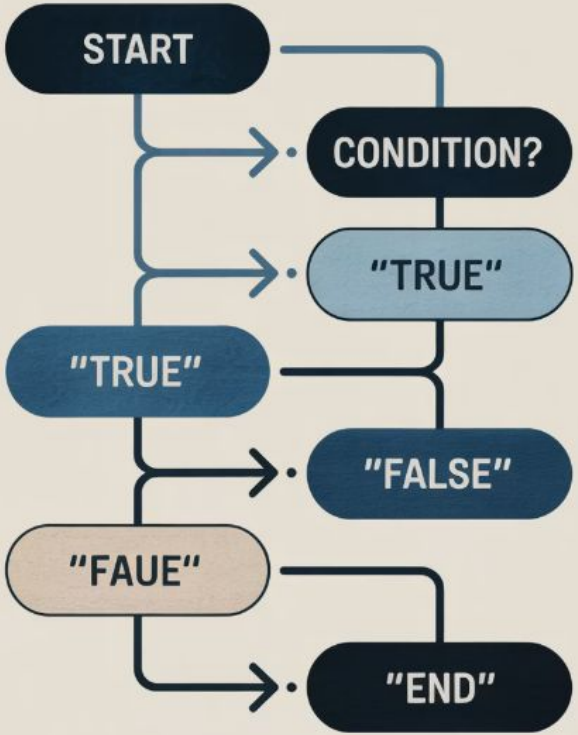
Herramientas para tomar decisiones dentro del código.

Tipos principales

if, if-else, if-else if-else.

Sintaxis básica

if (condición) { código }



Condicionales: La Base de la Toma de Decisiones

Evaluación de condición

El programa evalúa si una expresión es verdadera o falsa.

Bifurcación

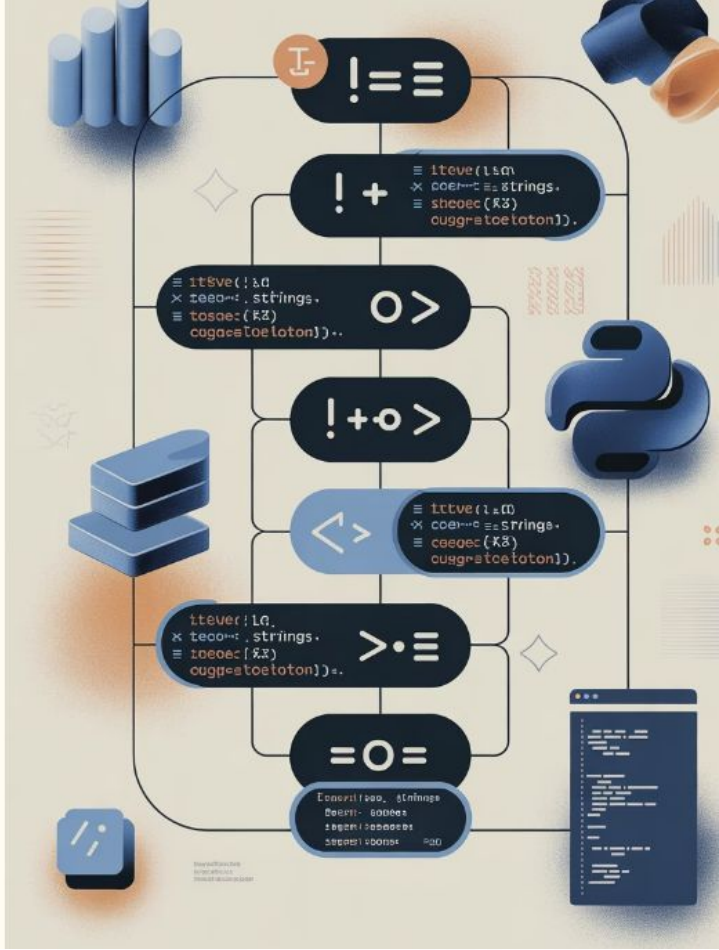
El flujo se divide según el resultado de la evaluación.

Ejecución condicional

Solo se ejecuta el bloque de código correspondiente.

Operadores de Comparación

Operador	Significado	Ejemplo
==	Igual a	x == 5
!=	Diferente de	x != 10
>	Mayor que	edad > 18
<	Menor que	temp < 0
>=	Mayor o igual	nota >= 5
<=	Menor o igual	precio <= 100



¿Qué Diferencia Hay Entre = Y ==?

Operador de Asignación (=)

Asigna un valor a una variable.

- No evalúa condiciones
- Modifica el valor almacenado
- Es una instrucción, no una expresión

Ejemplo: **edad = 18** guarda el valor 18 en la variable edad.

Operador de Igualdad (==)

Compara si dos valores son iguales.

- Devuelve verdadero o falso
- No modifica variables
- Se usa en condicionales

Ejemplo: **if (edad == 18)** evalúa si edad tiene el valor 18.

Errores Comunes

Confundir estos operadores provoca bugs difíciles de detectar.

- Usar = en condiciones asigna valores
- Usar == para asignar no modifica variables
- Algunos lenguajes usan === para igualdad estricta

Comentarios en el Código: Tu Mejor Aliado




Documentación

Explican el propósito y funcionamiento de cada parte del programa.

Colaboración

Facilitan que otros programadores entiendan tu lógica rápidamente.

Depuración

Ayudan a identificar errores y mantener el código.

Futuro

Tu yo del futuro agradecerá entender el código meses después.

Los comentarios son texto que el compilador ignora. Nos permiten explicar nuestro código sin afectar su funcionamiento.

Operador Ternario: Condicionales en Una Línea

Una forma elegante y compacta de escribir decisiones simples en tu código.

Condición

Una expresión que evalúa a verdadero o falso.

Valor si Verdadero

El resultado que se devuelve cuando la condición es verdadera.

Valor si Falso

El resultado que se devuelve cuando la condición es falsa.

La sintaxis es: **condición ? valorSiVerdadero : valorSiFalso**

Ejemplo:

```
let mensaje = edad >= 18 ? "Adulto" : "Menor";
```

🛎️ Recordá:

Valores Falsy

- Cadena vacía ("")
- Valor 0
- null
- undefined
- NaN

Valores Truthy

- Cadenas con contenido
- Números (excepto 0)
- Arrays (incluso vacíos)
- Objetos
- Funciones

Aplicación Práctica

Los valores truthy/falsy permiten simplificar condicionales. Útil para validar entradas de usuario y establecer valores por defecto.

Switch: Múltiples Caminos, Una Decisión

Expresión de control

Un valor que determina qué caso ejecutar

Casos posibles

Valores específicos a comparar con la expresión

Bloques de código

Instrucciones a ejecutar para cada caso

Caso por defecto

Se ejecuta cuando ningún caso coincide

Loops: La Magia de la Repetición

Inicialización

Preparamos las variables para comenzar el ciclo

Condición

Evaluamos si debemos continuar iterando

Actualización

Modificamos variables para la siguiente iteración

Ejecución

Realizamos las operaciones del bucle

Tipos de Bucles: For, While y Do-While

For

Ideal cuando conocemos el número exacto de iteraciones.

```
for (i=0; i<10; i++) {
  // código
}
```

While

Útil cuando la condición de parada no es predecible.

```
while (condición) {
  // código
}
```

Do-While

Garantiza al menos una ejecución del código.

```
do {
  // código
} while (condición);
```

Funciones: Bloques de Código Reutilizables

Declaración

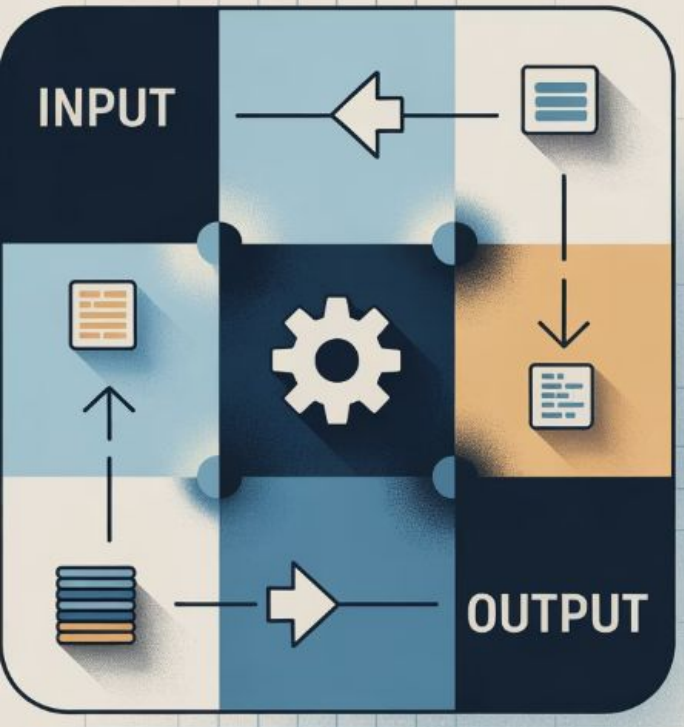
Definimos el nombre y estructura de la función

Reutilización

Llamamos a la función desde diferentes partes del código

Encapsulamiento

Agrupamos código relacionado en unidades lógicas



Funciones con Parámetros y Retorno

Parámetros

Son los datos que la función recibe para procesar.

- Pueden ser opcionales u obligatorios
- Permiten personalizar el comportamiento

Procesamiento

Las operaciones realizadas con los parámetros.

- Cálculos matemáticos
- Manipulación de texto

Retorno

El resultado final que devuelve la función.

- Puede ser un valor o ninguno (void)
- Define qué obtenemos al llamar la función

Hands on!

streambe GENERACIÓN T

Muchas gracias.

streambe GENERACIÓN T

Nuestras Redes

www.generaciont.org
www.streambe.com
www.instagram.com/generaciont_ar
www.tiktok.com/@generaciont
generaciont@generaciont.org
Cel: 11 61331747

GENERACIÓN T

streambe