COLOMBIA POTENCIA DE LA VIDA

TIC

TALENTO TECH

DESARROLLO *WEB*

FULL STACK

NIVEL BÁSICO

Lógica de Programación

cymetria | tecnalia colombia

Objetivo la Lógica de Programación

El objetivo de este bootcamp es proporcionar a los participantes una comprensión sólida de la **lógica de programación**, enseñando los principios fundamentales para resolver problemas mediante el uso de **algoritmos y estructuras de control**. Los estudiantes aprenderán a **analizar problemas, diseñar soluciones eficientes y escribir código estructurado**, sin importar el lenguaje de programación que utilicen en el futuro.

Al finalizar, los participantes podrán:

- ✅ Comprender y aplicar los conceptos básicos de la lógica de programación.
- ✅ Diseñar y desarrollar algoritmos para resolver problemas computacionales
.
- ✅ Implementar estructuras de control como condicionales y bucles.
- ✅ Escribir código eficiente y bien estructurado en cualquier lenguaje de programación.

¿Por qué la Lógica de Programación es la Base de la Programación?

La **lógica de programación** es el pilar fundamental del desarrollo de software, ya que permite estructurar pensamientos de manera clara para escribir código funcional. Sin una buena lógica, el código sería desordenado, ineficiente y difícil de mantener.

🔷 **Independencia del lenguaje:** Los principios de la lógica se aplican en cualquier lenguaje de programación.

🔷 **Optimización del código:** Mejora el rendimiento y evita errores.

🔷 **Facilita la depuración:** Un código bien estructurado es más fácil de corregir y mejorar.

🔷 **Es la base de cualquier paradigma de programación:** Desde la programación estructurada hasta la orientada a objetos y funcional.

Actividad: "El Algoritmo Humano"

Objetivo:

Los campistas experimentarán cómo funcionan los algoritmos al dar instrucciones detalladas para realizar una tarea cotidiana. Esto refuerza el pensamiento lógico, la secuenciación de pasos y el manejo de errores (depuración).

- Materiales:

pizarras para escribir instrucciones.

Papel y lápiz para anotar errores y mejoras.

- Desarrollo de la actividad:

División en Grupos: Se 3 equipos campistas.

- Elección de una Tarea Sencilla:

Cada equipo elige una tarea cotidiana

1. Hacer un Sándwich
2. Bañarse
3. Colocarse los Zapatos

- Escriben un conjunto de pasos detallados cumplir las tareas
- Un miembro del equipo actúa como la "máquina" y sigue exactamente las instrucciones dadas.

¿Qué es un Algoritmo?

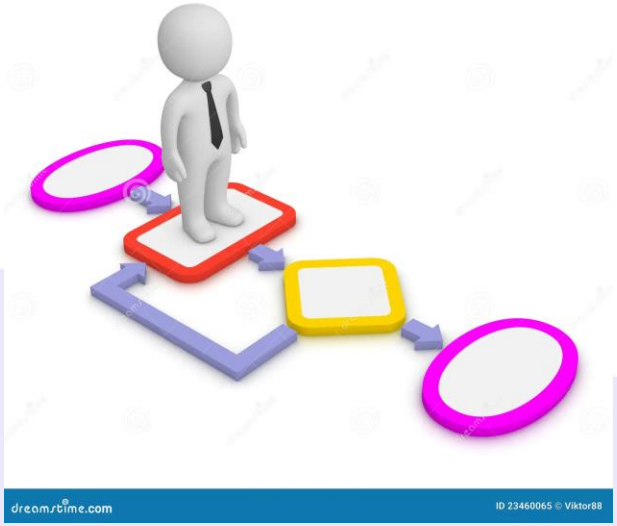
Un **algoritmo** es un conjunto ordenado de pasos o instrucciones diseñadas para resolver un problema o realizar una tarea específica. Es la base de cualquier programa informático.

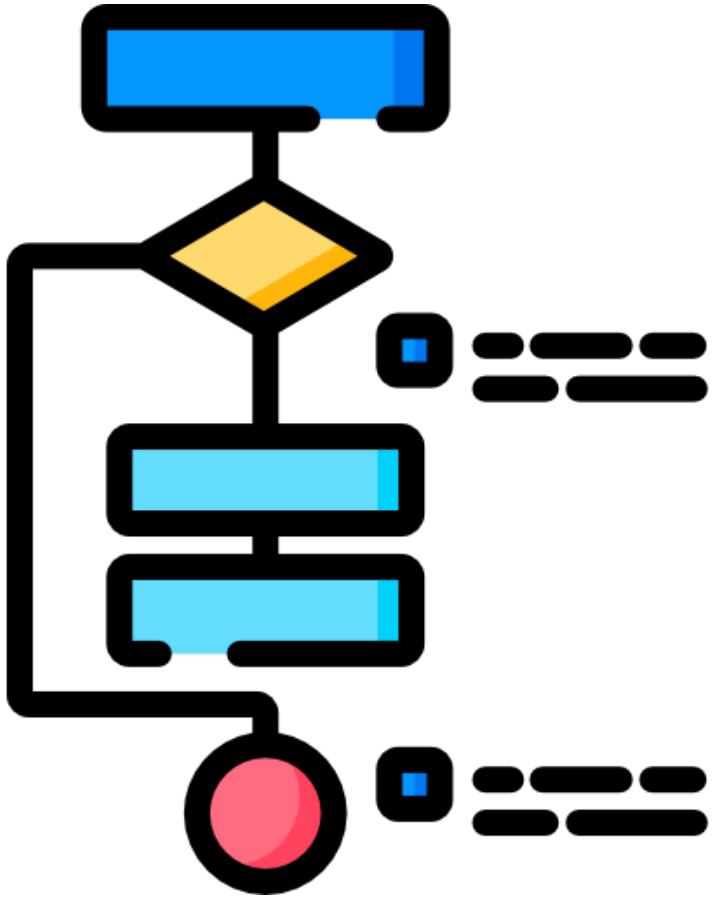
Ejemplo de Algoritmo

📌 **Problema:** Hacer un café.

Pasos del algoritmo:

1.Hervir agua.
2.Poner café en una taza.
3.Agregar azúcar o leche si se desea.
4.Verter el agua caliente en la taza.
5.Revolver y servir.





https://www.flaticon.es/icono-gratis/algoritmo_1197408

Las 3 Secciones de Todo Algoritmo

Todo algoritmo consta de tres partes fundamentales, estas secciones son esenciales para organizar la lógica de cualquier programa y garantizar que funcione correctamente. Veamos cada una con detalle:

1. Entrada

Es la sección donde se reciben los datos necesarios para que el algoritmo pueda ejecutarse. Estos datos pueden ser ingresados por un usuario, leídos desde un archivo o generados de alguna otra forma.

- **Ejemplo:** Si queremos hacer un algoritmo para sumar dos números, la **entrada** serán esos dos números.

2. Proceso

Aquí se realizan las operaciones o cálculos necesarios utilizando los datos de entrada. Es el "corazón" del algoritmo donde se define la lógica del programa.

- **Ejemplo:** En nuestro caso de la suma, el **proceso** es la operación matemática: número1 + número2.

3. Salida

Es la sección donde se muestra el resultado del algoritmo, ya sea en pantalla, guardado en un archivo o enviado a otro sistema.

- **Ejemplo:** Mostrar en pantalla el resultado de la suma.

Características de un Algoritmo

Un buen algoritmo debe cumplir con las siguientes características:

1. **Preciso:** Cada paso debe estar bien definido y sin ambigüedades.
2. **Definido:** Si se sigue correctamente, siempre debe dar el mismo resultado.
3. **Finito:** Debe terminar en algún momento, no puede ejecutarse indefinidamente.
4. **Eficiente:** Debe usar los mínimos recursos posibles (memoria y tiempo).
5. **Ordenado:** Sus pasos deben estar estructurados de manera lógica y secuencial.

Estructuras de Control en un Algoritmo

1 Estructura Secuencial

Ejecuta las instrucciones en orden, de arriba hacia abajo.

📌 **Ejemplo en Pseudocódigo:**

nginx Copy Edit

```
Inicio
   Leer nombre
   Escribir "Hola ", nombre
Fin
```

Características de un Algoritmo

Estructura Condicional (Selección - if)

Permite tomar decisiones basadas en una condición.

📌 Ejemplo en Pseudocódigo:

```nginx
Inicio
   Leer edad
   Si edad >= 18 Entonces
      Escribir "Eres mayor de edad"
   Sino
      Escribir "Eres menor de edad"
Fin
```

Estructura Repetitiva (Bucles - for, while)

Permite ejecutar un bloque de código varias veces.

📌 Ejemplo en Pseudocódigo (Contar del 1 al 5 con `for`)

```css
Inicio
   Para i desde 1 hasta 5 hacer
      Escribir i
Fin
```

¿Qué es un pseudocodigo?

El pseudocódigo es una forma de representar algoritmos utilizando una mezcla de lenguaje natural y estructuras propias de la programación, sin depender de una sintaxis específica de un lenguaje de programación. Se usa para planificar la lógica de un programa antes de escribir el código real.

Características del pseudocódigo:

- **Claridad**: Debe ser fácil de entender para cualquier persona con conocimientos de lógica de programación.
- **Estructurado**: Utiliza estructuras de control como secuencias, condiciones y bucles.
- **Independiente del lenguaje**: No se escribe en ningún lenguaje de programación en particular.
- **Ordenado**: Sigue una secuencia lógica de pasos para resolver un problema.

Ejemplo Básico para Explicar la Diferencia entre Realidad, Algoritmo y Pseudocódigo

Vamos a trabajar con una actividad cotidiana: hacer un sándwich.

1 Descripción en la Vida Real (Ejemplo Real):

◆ Instrucciones en lenguaje común:

1.Ir a la cocina.
2.Sacar dos rebanadas de pan.
3.Tomar los ingredientes (jamón, queso, mantequilla, etc.).
4.Untar mantequilla en una rebanada de pan.
5.Colocar el jamón y el queso sobre el pan.
6.Poner la otra rebanada encima.
7.Servir el sándwich en un plato y comer.

2 Conversión a Algoritmo (Forma Estructurada):

◆ División en Entrada, Proceso y Salida:

Entrada:
•Ingredientes: Pan, jamón, queso, mantequilla.
•Utensilios: Cuchillo, plato.
Proceso:
1.Tomar dos rebanadas de pan.
2.Untar mantequilla en una rebanada.
3.Colocar jamón y queso sobre la rebanada con mantequilla.
4.Tapar con la otra rebanada de pan.
5.Colocar el sándwich en un plato.
Salida:
•Sándwich listo para comer.

3 Conversión a Pseudocódigo (Forma más Técnica):

Inicio
Entrada:
Definir pan, jamón, queso, mantequilla como **ingredientes**
Definir cuchillo, plato como **utensilios**
Proceso
Tomar dos rebanadas de pan Untar mantequilla en una rebanada Colocar jamón y queso sobre la rebanada Poner la otra rebanada encima Colocar el sándwich en un plato
Salida
Escribir "Sándwich listo para comer" **Fin**

Diferencia entre Realidad, Algoritmo y Pseudocódigo

- ✅ La vida real: Explicaciones informales.
- ✅ El algoritmo: Proceso estructurado en Entrada, Proceso y Salida.
- ✅ El pseudocódigo: Representación técnica de la lógica del algoritmo.

Ejercicio en Clase: Determinar si un Número es Positivo, Negativo o Cero

Este ejercicio ayudará a los campistas a comprender cómo se transforma un problema de la vida real en un algoritmo y luego en pseudocódigo.

1 Descripción en la Vida Real (Lenguaje Cotidiano)

Imagina que alguien te pregunta si un número es positivo, negativo o cero. ¿Cómo lo responderías?