

# Introduccion a la programacion

---

## 1. ¿Qué es un algoritmo? Definición y Ejemplos

---

### Definición formal

Un **algoritmo** es una secuencia finita, ordenada y precisa de pasos que resuelven un problema.

Característica	Descripción
Finitud	Tiene un número limitado de pasos
Precisión	No hay ambigüedad
Entrada	Datos necesarios para iniciar
Salida	Resultado del proceso
Eficiencia	Uso óptimo de recursos

---

### Definición intuitiva

Un algoritmo es como una **receta de cocina**:  
tiene ingredientes (entradas), pasos (procesos), y  
un plato final (salida).

---

## Ejemplos de algoritmos cotidianos

- 🔍 Hacer una tortilla
  - 🩺 Tomar la presión arterial
  - 🐕 Pasear al perro
- 

## 2. Diagramas de flujo: símbolos, construcción y ejemplos





---

¿Qué es un diagrama de flujo?

Un diagrama de flujo es una forma gráfica de representar un algoritmo usando **símbolos estándar**.

---

### Símbolos básicos

Símbolo	Significado
 Óvalo	Inicio o fin
 Rectángulo	Proceso o acción
 Rombo	Decisión / condición
 Flecha	Dirección del flujo

 Herramientas útiles:

- <https://draw.io>

- <https://lucidchart.com>

---

### 3. Pseudocódigo: sintaxis, lógica y traducción desde diagramas

---

¿Qué es el pseudocódigo?

Es una forma textual de escribir algoritmos con estructura clara, sin seguir la sintaxis de un lenguaje real.

---

### 4. Instrucciones básicas: asignación, condicionales, bucles

---

 Tipos

- Asignación:  $x \leftarrow 5$
- Condicional: Si  $x > 0$  entonces
- Bucle: Mientras  $x < 10$  hacer

---

 Ejemplo (Python)

```
x = 5
```

```
if x > 0:
```

```
print("Positivo")  
while x < 10:  
    x += 1
```

---

### Actividades

1. Implementa un bucle que imprima los números del 1 al 5.
  2. Escribe un condicional que clasifique la edad de una persona.
- 

## 5. Expresiones por categoría: aritméticas, lógicas y de texto

---

### Tipos

Tipo	Ejemplo
Aritmética	$(3 + 5) * 2$
Lógica	$x > 3 \text{ Y } z < 7$
Texto	"Hola" + " Mundo"

---

### Actividad

1. Escribe tres expresiones de cada tipo.
2. Evalúa el resultado y explica el por qué.

---

## 6. Condicionales: if, else, switch/case

---

### Tipos

- `if` y `else`
- `elif` (en Python)
- `switch/case` (en otros lenguajes como C o Java)

---

### Ejemplo

`nota = 6`

`if nota >= 5:`

`print("Aprobado")`

`else:`

`print("Suspendido")`

---

### Actividad

1. Crea un switch que devuelva el día de la semana según un número.
  2. Escribe un algoritmo con múltiples condiciones encadenadas.
- 

## 7. Tipos de bucles: while, do-while, for, do-until

---

### Comparación

Bucle	Se evalúa antes	Usa contador	Se ejecuta siempre 1 vez
while	Sí	No	No
do-while	No	No	Sí
for	Sí	Sí	No
do-until	No	A veces	Sí

---

### Actividad

1. Crea un bucle que imprima los números pares del 1 al 20.
  2. Modifícalo para que se detenga si encuentra el 12.
-

## 8. Tipos de datos básicos

---

### Tipos comunes

Tipo	Ejemplo
Entero	1, -3
Real	3.14, -2.5
Booleano	True, False
Carácter	'a', '%'
Cadena	"Texto"

---

### Actividad

1. Declara una variable de cada tipo.
  2. Realiza una operación válida con cada una.
- 

## 9. Funciones, procedimientos y rutinas

---

### ¿Qué son?

- Función: bloque de código que devuelve un valor.
- Procedimiento: ejecuta acciones, sin retornar valor.

- Rutina: término general para ambos.

---

#### Ejemplo (Pseudocódigo)

Funcion Sumar(a, b)

Retornar  $a + b$

FinFuncion

---

#### Actividad

1. Escribe una función que reciba dos números y retorne el mayor.
  2. Crea un procedimiento que muestre tu nombre tres veces.
- 

## 10. Parámetros de entrada, salida y entrada/salida

---

#### Tipos de parámetros

Tipo	¿Qué hace?
Entrada	Se usa como dato de entrada, no se modifica



Salida	Devuelve un valor desde la función
--------	------------------------------------

Entrada/Salida	Se modifica dentro y fuera de la función
----------------	--

---

### Ejemplo

Funcion Sumar(PorValor a, PorValor b, PorReferencia resultado)

resultado  $\leftarrow$  a + b



FinFuncion

---

## 11. Paso de parámetros: por valor vs por referencia

---

### Comparativa

Tipo	¿Se modifica la variable original?	Uso común
Por valor	 No	Números simples
Por referencia	 Sí	Arreglos, objetos

---

### Actividad

1. Prueba pasar una lista por valor y por referencia.
2. Observa si se modifica fuera de la función.

---

## 12. Estructuras de datos: listas, pilas, colas, matrices, listas enlazadas

---

### Resumen

Estructura	Tipo de acceso	Ejemplo de uso
Lista	Índices	Lista de nombres
Pila (LIFO)	Último entra	Deshacer acciones
Cola (FIFO)	Primero entra	Impresión en cola
Matriz	Fila y columna	Tablas, imágenes
Lista enlazada	Nodo→nodo	Navegación de páginas

---

### Actividad

1. Simula una pila con las páginas visitadas en un navegador.

2. Representa una cola de espera en un hospital.

---

## 13. Árboles y grafos

---

### Árboles

- Jerárquicos: padre, hijos
  - Tipos: binarios, AVL, B, binarios de búsqueda
- 

### Grafos

- Nodos y conexiones (aristas)
  - Tipos: dirigidos, no dirigidos, ponderados
  - Aplicaciones: mapas, redes sociales, sistemas de transporte
- 

### Actividad

1. Representa un árbol genealógico.
  2. Crea un grafo que modele una red de amigos.
- 

## 14. Notaciones algorítmicas: $O$ , $\Omega$ , $\Theta$ , etc.

---

### Tabla de complejidades

Notación	Tipo de eficiencia	Ejemplo
$O(1)$	Constante	Acceso directo a un array
$O(n)$	Lineal	Búsqueda simple
$O(\log n)$	Logarítmica	Búsqueda binaria
$O(n \log n)$	Log-lineal	QuickSort, MergeSort
$O(n^2)$	Cuadrática	Burbuja, selección

---

### Actividad

1. Clasifica los siguientes algoritmos: búsqueda binaria, ordenamiento burbuja, acceso a arreglo.
  2. Explica cuándo conviene usar cada uno.
- 

## 15. Gráficas y tablas de rendimiento algorítmico

---

### Curvas de crecimiento (representación mental)

- Constante: plano horizontal
  - Lineal: recta ascendente
  - Logarítmica: curva que se aplanan
  - Cuadrática: curva creciente rápida
- 

### Actividad

1. Dibuja estas curvas y ordénalas de más a menos eficiente.
  2. Relaciona algoritmos con su curva.
- 

## 16. Paradigmas de lenguajes de programación

---

### Tipos de paradigmas

Paradigma	Características	Lenguajes comunes
Imperativo	Secuencia de instrucciones	C, Python
Funcional	Funciones puras, sin estado	Haskell, F#
Lógico	Reglas y hechos	Prolog

Orientado a Clases y objetos Java, C++  
objetos

---

#### Actividad

1. Clasifica varios lenguajes por paradigma.
  2. Escribe una misma operación en estilo imperativo y funcional.
- 

## 17. Introducción a punteros y lenguaje máquina

---

#### Conceptos

- Puntero: almacena dirección de memoria
  - Stack: memoria para llamadas de funciones
  - Heap: memoria dinámica
  - Registros: variables internas del procesador (ej. AX, BX)
- 

#### Actividad

1. Simula cómo una variable ocupa memoria.
2. Dibuja el stack y heap durante una ejecución.

---

## 18. Compilación, linkado y librerías

---

### Proceso

1. Compilar: fuente → objeto
2. Linkado: une objetos y librerías
3. Librerías:
  - Estáticas (.lib): se incluyen en el ejecutable
  - Dinámicas (.dll, .so): se cargan en ejecución

---

### Actividad

1. Compila un programa con una librería externa.
2. Investiga si tu sistema usa DLL o SO.

---

## 19. Transpilación, uglify, minimize y ofuscación

---

### Definiciones

- Transpilación: convertir entre lenguajes similares (TypeScript → JS)

- Minimizar: eliminar espacios, reducir tamaño
  - Uglify: sin formato, ilegible
  - Ofuscar: esconder lógica para proteger código
- 




### Actividad

1. Usa <https://javascript-minifier.com> para minificar código.
  2. Prueba una herramienta de ofuscación.
- 

## Recursos Recomendados por Tema

---





### 1. Algoritmos y Estructuras de Datos

-  [Khan Academy – Algoritmos \(español\)](#)  
Curso interactivo con visualizaciones y ejercicios.
-  [Visualgo.net](#)  
Animaciones visuales de estructuras de datos y algoritmos clásicos.
-  [GeeksforGeeks – Fundamentos de Algoritmos](#)  
Artículos extensos con código y visualizaciones.
-  YouTube:  
 [Curso completo de estructuras de datos en español \(Píldoras Informáticas\)](#)






---

## 2. Diagramas de Flujo y Pseudocódigo

-  [Programiz – Diagrams and Flowcharts](#)  
Explicación clara sobre símbolos, usos y ejemplos.
-  YouTube:
  -  [Diagrama de flujo explicado desde cero \(Profe Alex\)](#)
  -  [Pseudocódigo explicado fácil \(Código Facilito\)](#)




---

## 3. Tipos de Datos, Expresiones y Variables




-  [W3Schools – Variables y tipos](#)  
(Puedes elegir el lenguaje: Python, C, JavaScript...)
-  YouTube:
  -  [Variables y tipos de datos en programación \(HolaMundo\)](#)

---




## 4. Bucles, Condicionales e Instrucciones Básicas

-  [Programiz – Control Flow](#)  
Control de flujo para múltiples lenguajes.
  -  YouTube:
    -  [Bucles y condicionales – Curso de programación \(EDteam\)](#)
-




## ■ 5. Compilación, Linkado y Librerías

-  [GeeksforGeeks – Compilation Process](#)  
Excelente explicación con esquemas y fases.
  -  YouTube:  
 [¿Qué pasa al compilar un programa? \(Píldoras Informáticas\)](#)
- 

## ■ 6. Punteros y Lenguaje de Bajo Nivel

-  [Programiz – Punteros en C](#)  
Tutorial con código visualizado.
  -  YouTube:  
 [Punteros para principiantes \(Fazt Code\)](#)
- 

## ■ 7. Notación Big-O y Complejidad

-  [Big-O Cheat Sheet](#)  
Tabla visual de algoritmos con complejidades.
  -  YouTube:  
 [Notación Big O explicada fácil \(Academind\)](#)
- 

## ■ 8. Paradigmas de Programación

-  [GeeksforGeeks – Paradigms](#)  
Comparativa detallada.

-  YouTube:  
 [Paradigmas de Programación explicados \(Código Facilito\)](#)