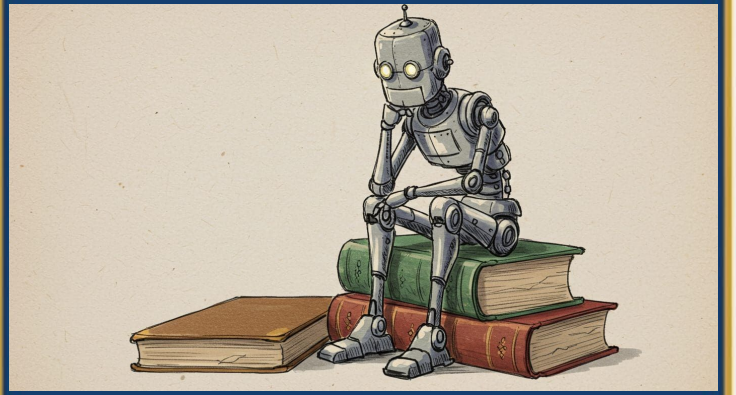
Unidad 1: Introducción a la Programación y Pensamiento Computacional



De la intuición humana a las instrucciones científicas

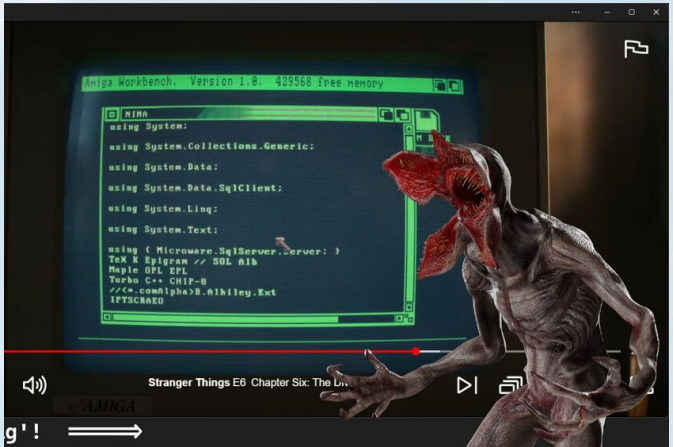
Temas Selectos para CT
2026-2

¿Cómo empezamos a programar realmente?

- No es 'aprender a programar', es aprender a 'dar instrucciones sin ambigüedad'.
- Un algoritmo = Una receta científica verificable.
- Un lenguaje de programación = idioma estricto
- Entradas claras -> Pasos simples -> Salidas comprobables.
- *Si no puedes describirlo, no lo puedes programar.*

El Cerebro y los Datos

- **“Memorias” (contenido)** : palabras, lugares, caras, olores.
- **“Operaciones” (procedimientos)** : cómo haces algo (multiplicar, manejar, escribir).
- **“Procesos automáticos”** : cosas que el cerebro hace “en segundo plano” (consolidación durante el sueño, hábitos, predicciones).
- ¿Para qué y cómo programamos?



¿Qué es un código?
¿Qué es un algoritmo?

R en clase

R extendida:

Un **algoritmo** es una receta: una secuencia de pasos claros y sin ambigüedad que, dadas unas entradas (lo que tienes), produce una salida (lo que quieres) para resolver un problema o cumplir un objetivo.

El algoritmo existe independientemente de la computadora: puedes describirlo en español, con un diagrama o con ejemplos. Lo esencial es que los pasos sean lo bastante precisos como para que otra persona (o tú mismo mañana) pueda seguirlos y llegar al mismo resultado.

El **código** es la forma de escribir ese algoritmo como instrucciones en un lenguaje. Cuando lo escribes en un **lenguaje formal** (Python, R, etc.), esas instrucciones pueden ser ejecutadas por una computadora.

¿Qué es el pseudocódigo?

El pseudocódigo es una forma de representar código, como algoritmos, funciones y otros procesos, utilizando una combinación de lenguaje natural y elementos similares al lenguaje de programación.

Se llama «pseudocódigo» porque no es realmente ejecutable. En cambio, es una forma de que los humanos comprendan y planifiquen la lógica de la programación, describir los pasos de un programa de forma que sea fácil de entender para nosotros, sin dejar de ser lo suficientemente detallado como para convertirse rápidamente en un lenguaje de programación específico.

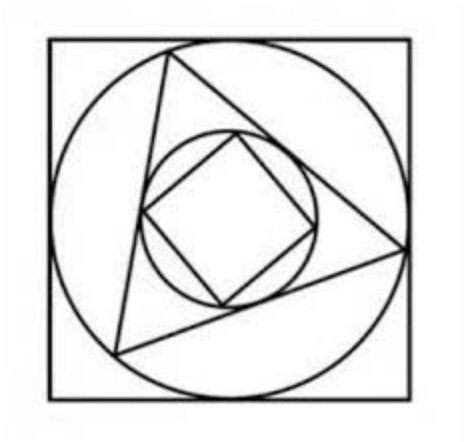
Experimento de memoria

- Voy a leer una serie de números, uno por segundo.
- Cuando termine, escriban la serie exactamente en el mismo orden.
- No se vale grabar audio, ni repetir en voz alta. Solo escuchen y al final escriban.

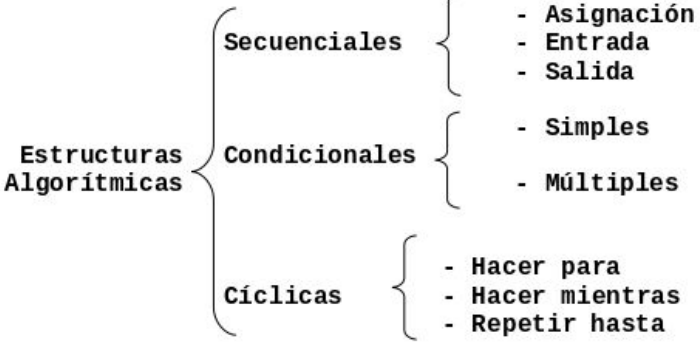
¿Cómo podríamos hacer un algoritmo para ordenarle a un robot que nos prepare un sandwich?

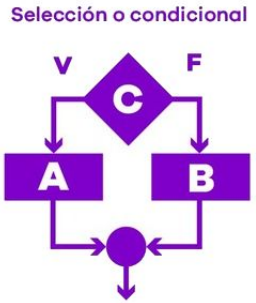
Prepare una lista de instrucciones, claras y secuenciales, que le permitan a un robot (intérprete) hacer su sandwich favorito.

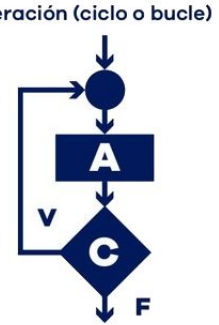
Ejercicio de copiar una imagen



Un algoritmo es ...



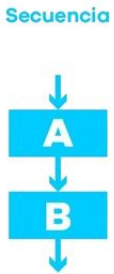


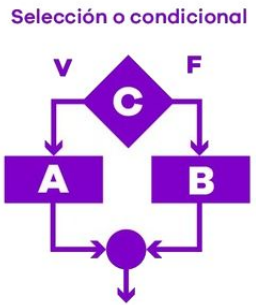


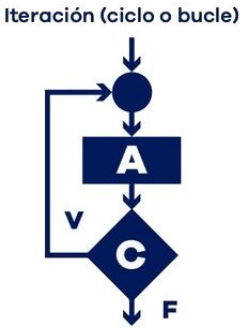
Algo importante: **las variables y ... las sentencias...**

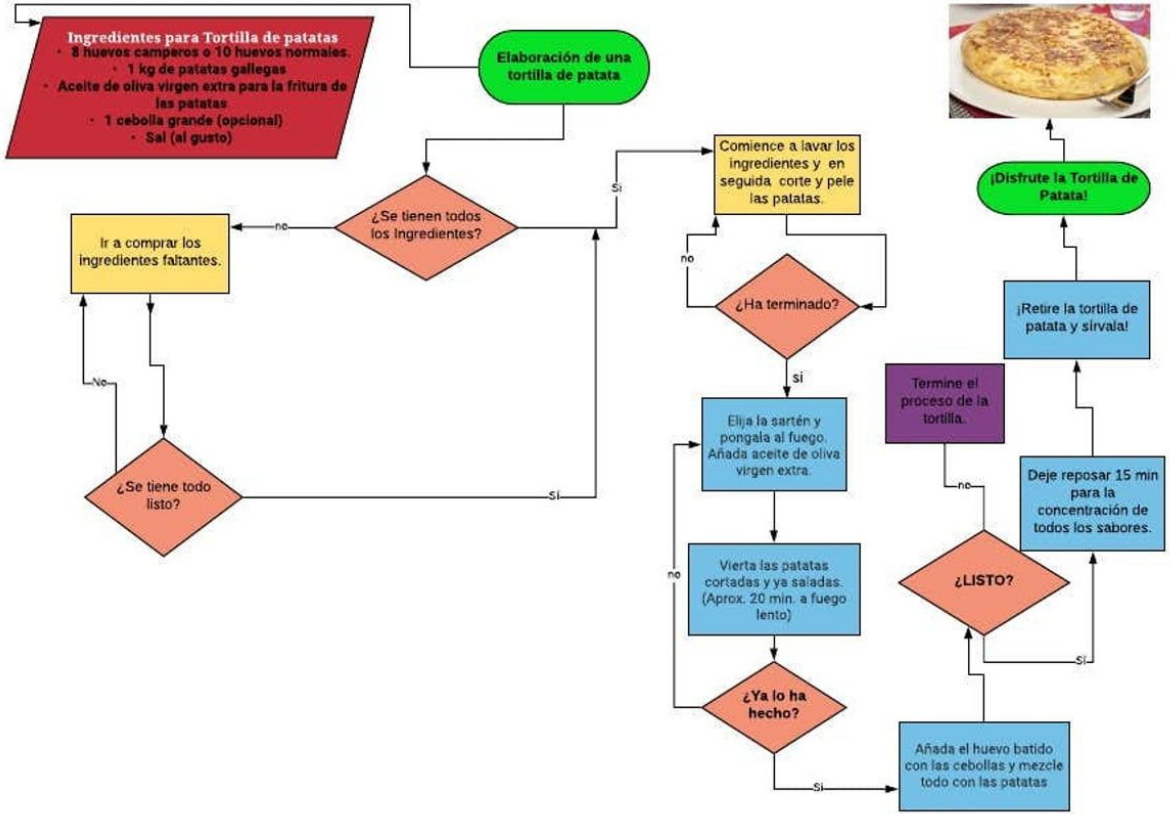
Para el pseudocódigo hay varias formas de organizarse ...

```
Inicio
    Mostrar  'mensaje'
    leer  variable
    Repetir
            instrucción
    Hasta
    si condición
        acción
    sino  alternativa
Fin
```








Ingredientes para Tortilla de patatas
· 8 huevos camperos o 10 huevos normales.
· 1 kg de patatas gallegas
· Aceite de oliva virgen extra para la fritura de las patatas
· 1 cebolla grande (opcional)
· Sal (al gusto)
Elaboración de una tortilla de patata
¿Se tienen todos los ingredientes?
no
Ir a comprar los ingredientes faltantes.
Si
Comience a lavar los ingredientes y en seguida corte y pele las patatas.
¿Ha terminado?
no
Si
Elija la sartén y pongala al fuego. Añada aceite de oliva virgen extra.
Vierta las patatas cortadas y ya saladas. (Aprox. 20 min. a fuego lento)
no
¿Ya lo ha hecho?
Si
Añada el huevo batido con las cebollas y mezcle todo con las patatas
¿LISTO?
no
Si
Termine el proceso de la tortilla.
Deje reposar 15 min para la concentración de todos los sabores.
¡Retire la tortilla de patata y sirvala!
¡Disfrute la Tortilla de Patata!
¿Se tiene todo listo?
No
Si

Un poco de lógica: (and, or, not)

Operator Name	Operator Symbol	Functionality	Example
Logical AND	and	If both the operands are true, then the condition becomes true.	X = True Y = False X and Y = False
Logical OR	or	If any of the two operands are true, then the condition becomes true.	X = True Y = False X or Y = True
Logical NOT	not	Used to reverse the logical state of its operand.	X = True Y = False not(X and Y) = True

1 = Verdadero , 0 = Falso

Negación

q	¬
1	0
0	1

$\neg p$

Conjunción

p	q	∧
1	1	1
1	0	0
0	1	0
0	0	0

$p \wedge q$

La **conjunción**, similar al castellano «...y...»; es verdadera solo cuando ambas proposiciones son verdaderas. Es equivalente al producto en el Álgebra de Boole.

«Muchos vivos merecerían la muerte y algunos que mueren merecen la vida»

Disyunción

p	q	∨
1	1	1
1	0	1
0	1	1
0	0	0

$p \vee q$

La **disyunción** es verdadera siempre y cuando sean verdaderas alguna de las variables o ambas. No se corresponde exactamente tampoco con la disyunción gramatical «...o...», pues no expresa simplemente la alternancia entre las dos opciones. Es igualmente verdadera cuando ambas proposiciones son verdaderas. Su equivalente en el Álgebra de Boole es la suma.

«¿Me deseas un buen día o quieres decir que hoy es un buen día lo quiera o no?»

Las condiciones

Condicional

p	q	→
1	1	1
1	0	0
0	1	1
0	0	1

$p \rightarrow q$

El **condicional**, también llamado implicación, niega la posibilidad de que la primera variable sea cierta sin que lo sea la segunda. Corresponde a lo que vulgarmente sería «si... entonces...». Debe apuntarse que la condicionalidad no es bidireccional: p no puede concluirse a partir de q. Tampoco expresa ninguna implicación causal.

«En caso de duda, Meriadoc, sigue siempre a tu olfato»

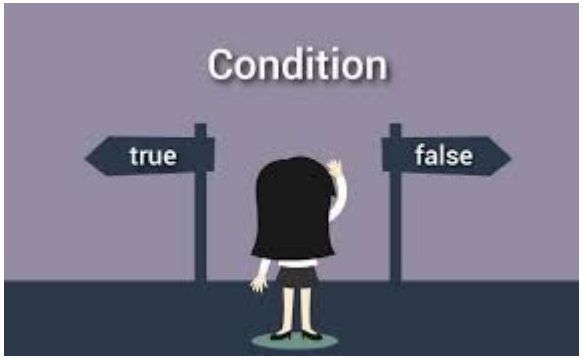
Bicondicional

p	q	↔
1	1	1
1	0	0
0	1	0
0	0	1

$p \leftrightarrow q$

El **bicondicional** o condicional recíproco restringe su valor de verdad o bien cuando ambas variables son ciertas o cuando ambas son falsas. En lenguaje ordinario sería «...si y solo si...». En este caso sí es bidireccional de forma que $(p \rightarrow q) \wedge (q \rightarrow p)$. Tanto el bicondicional como el condicional cumplen el principio de que, dadas unas premisas verdaderas, la conclusión nunca puede ser falsa, un principio que será trascendental cuando veamos reglas de inferencia.

«Solo tu puedes decidir qué hacer con el tiempo que se te ha dado»



La programación saca nuestro lado tóxico ...
debemos condicionar y limitar muchas cosas.

Si ocurre x **entonces** pasa y

o ocurre a **entonces** pasa b

Mientras ocurre z **se** hace k

hasta que pase i descansas

Y no olviden que hay que ser MUY EXPLÍCITOS

Las 3 estructuras que dominan todo

1. Secuencia: Paso A -> Paso B -> Paso C.
2. Selección: SI ocurre X, haz Y. SI NO, haz Z.
3. Repetición (Loops): PARA cada píxel, PARA cada estación, PARA cada año.

En Geociencias, casi siempre iteramos en TIEMPO o ESPACIO.

Tu 'Receta' para cualquier problema

1. Objetivo: ¿Qué quiero resolver?
2. Entradas: ¿Qué datos tengo? (Variables, unidades, dimensiones).
3. Salida: ¿Qué debo obtener? (resultado, número, imagen).
4. Supuestos: ¿Qué asumo del problema? (Ej. algo que deba considerar).
5. Casos Límite: ¿Qué pasa si algo raro sucede, lo debería considerar?

Para crear un código es necesario seguir unos pasos

1. Saber qué queremos que haga el código

Para crear un código es necesario seguir unos pasos

1. Saber qué queremos que haga el código
2. Escribir el pseudocódigo:
    a. Establecer las instrucciones por medio de lógica

Para crear un código es necesario seguir unos pasos

1. Saber qué queremos que haga el código
2. Escribir el pseudocódigo:
    a. Establecer las instrucciones por medio de lógica
3. Abrir la terminal, colab, notebook...
4. ...
5. ...
6. ...
7. ...
8. Llorar...

Para crear un código es necesario seguir unos pasos

1. Saber qué queremos que haga el código
2. Escribir el pseudocódigo:
    a. Establecer las instrucciones por medio de lógica
3. Abrir la terminal, colab, notebook...
4. Comenzar con el algoritmo

Hagamos un algoritmo, su función es que los alumnos se titulen de la licenciatura.

Algoritmo: El proceso de titulación

- Entradas (variables): % de créditos, Servicio Social (S/N), Método elegido.
- Condiciones lógicas: requerimientos administrativos y académicos?
- Estados: Inicio -> Estudiante curricular -> Pasante -> Titulado.

Para el pseudocódigo...

1. **Abre tu editor de texto:** La mayoría de las veces, el pseudocódigo se escribe en un editor de texto. Puedes elegir tu favorito y abrir un nuevo archivo.
2. **Define tu objetivo:** Determina la finalidad de tu programa o función. ¿Qué quieres que haga?
3. **Sepáralo en partes:** Divide el problema en trozos más pequeños y manejables. Esto puede ayudarte a pensar en el problema con más claridad y facilitar la organización de las piezas para que funcionen donde y cuando deban.
4. **Organízalo en pasos:** Escribe los pasos de tu programa en orden lógico. Utiliza un lenguaje natural y evita utilizar construcciones o métodos de programación específicos, como estructuras de control o conversión de tipos.
5. **Sangría en las líneas:** Utiliza la sangría para mostrar la estructura de tu código. Por ejemplo, puedes sangrar las líneas de código que pertenecen a un bucle.
6. **Pruébalo:** Prueba tu pseudocódigo para asegurarte de que es claro y lógico. Puedes hacerlo recorriendolo verbalmente o pidiendo a otra persona que lo lea y te informe de lo que cree que debe hacer el pseudocódigo.

Por ejemplo:

Quiero un programa que intente adivinar el número que estoy pensando.

Por ejemplo:

Quiero un programa que intente adivinar el número que estoy pensando. Me preguntará si estoy pensando un número hasta que le atine.

Por ejemplo:

Quiero un programa que intente adivinar el número que estoy pensando, hasta que le atine y luego me pregunte si quiero jugar de nuevo.

Por ejemplo:

Quiero un programa que intente adivinar el número que estoy pensando entre 1 y 100, hasta que le atine y luego me pregunte si quiero jugar de nuevo.

Tarea 1 (En parejas):

Piensa en un programa simple que te gustaría crear y crea el pseudocódigo como hicimos en clase.