1A – Bienvenida

# ¿Por qué Programación Imperativa?

La programación no es solo ver números verdes sobre un fondo negro en la oscuridad de un cuarto: programar es crear soluciones creativas a un problema, es resolver una problemática, es descubrir vías de resolución de conflictos. En esta materia vamos a entender cómo funcionan los problemas lógicos y cómo descomponerlos para poder resolverlos de manera eficiente.

Tenemos el honor de ser la puerta de entrada de muchos a la programación, a un paradigma distinto de pensar el mundo, vamos a cimentar las bases que luego servirán para que las demás materias construyan, por esto mismo esta materia es tan importante y troncal en tu cursada, ya que te va a dar las herramientas para poder desarrollarte como programador.

Esta materia será fundamental para comenzar a conocer el mundo de la programación.

# Pensamiento Computacional

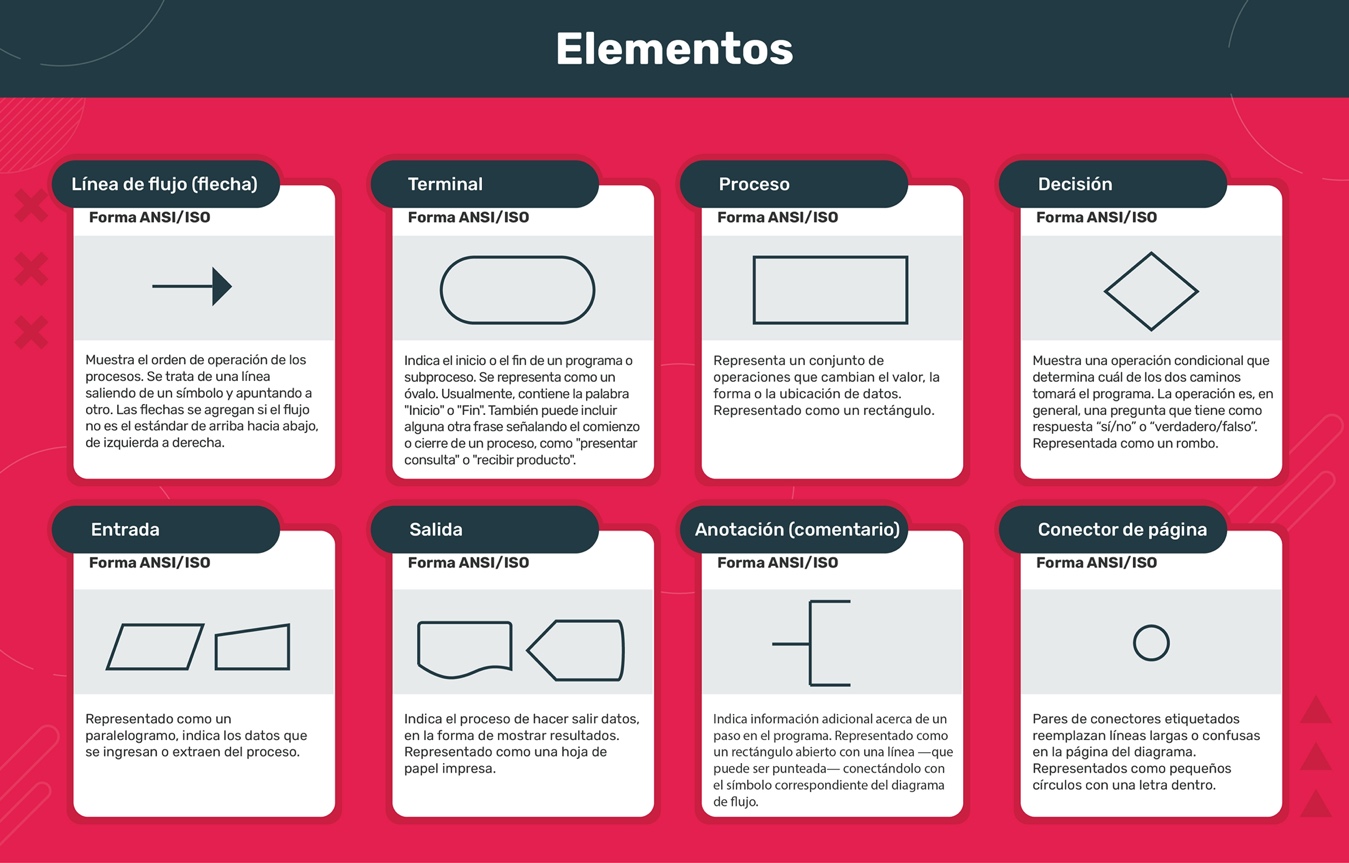
Cuando vamos a cocinar algo, empezamos buscando la receta para ver qué ingredientes y utensilios vamos a necesitar. De la misma forma, en una analogía rápida, el pensamiento computacional aparece como el paso previo a programar y consiste en adaptar nuestra forma de razonar para poder resolver un problema lógico. Es importante entender cómo se tiene que pensar al programar para poder resolver un problema considerando que nuestro receptor no tiene “sentido común”, sino que solo recibe instrucciones precisas para ejecutarlas.

Por eso, antes de empezar a programar, tenemos que aprender a resolver problemas y entender qué debemos tener en cuenta para poder transmitirle a una computadora las instrucciones que queremos que ejecute.

# Diagramas de flujo

El **diagrama de flujo, flujograma o diagrama de actividades** es la **representación gráfica de un algoritmo o proceso**. Se utiliza en disciplinas como programación, economía, procesos industriales y psicología cognitiva.

Se trata de una herramienta que nos permite determinar los pasos para poder llevar adelante un proceso o una tarea. Así, el diagrama de flujo resulta útil para pensar la resolución de un problema, sin importar qué **lenguaje que vayamos a usar**.



<https://1drv.ms/b/s!AvKEz5VBGr4QmCvwfTWoZGjusyqH?e=4WP6pn>

# Hacer un origami

Como vimos en la anterior clase, el pensamiento computacional es un elemento clave en el desarrollo de nuestras capacidades de programación.

En la siguiente actividad vamos a identificar diferentes algoritmos para realizar una tarea.



Como vimos en la actividad anterior, algunos aspectos del pensamiento computacional se pueden trasladar a nuestra vida cotidiana. Ya desde pequeños nos enseñan rutinas y pasos para realizar diversas tareas. Si bien en humanos es distinto que en las computadoras —ya que estas no pueden “asumir” situaciones— la idea es similar: **un algoritmo es una serie de pasos para resolver un problema.**

¿Cuál es la diferencia con una computadora? Que, por ejemplo, si nos estamos lavando los dientes y no hay agua o nos quedamos sin pasta de dientes, las personas pueden buscar soluciones basadas en experiencias anteriores o improvisar una solución, en el caso de las computadoras, si un paso programado no se puede cumplir, ese proceso no avanzará y quedará inconcluso.

# Paradigmas de programación - del símbolo al texto

1. Paradigmas de programación  
   ¿Qué son los paradigmas de programación?  
   Diferentes estilos de usar la programación para resolver un problema.

Programación Imperativa

Los programas consisten en una sucesión de instrucciones o conjunto de sentencias, paso a paso. Dentro del paradigma imperativo encontramos estos enfoques:

* **Programación estructurada**La programación estructurada es un tipo de programación imperativa donde el flujo de control se define mediante bucles anidados, condicionales y subrutinas, en lugar de a través de GO TO.
* **Programación procedimental  
  Este paradigma de programación consiste en basarse en un número muy bajo de expresiones repetidas, englobadas todas en un procedimiento o función y llamarlo cada vez que tenga que ejecutarse.**
* **Programación modular  
  Consiste en dividir un programa en módulos o subprogramas con el fin de hacerlo más manejable y legible. Se trata de una evolución de la programación estructurada para resolver problemas de programación más complejos.**

Programación Declarativa

Este paradigma no necesita definir algoritmos puesto que describe el problema en lugar de encontrar una solución al mismo. Así, utiliza el principio del razonamiento lógico para responder a las preguntas o cuestiones consultadas.

Se divide en dos:

* **Programación lógica  
  Un programa puede ser descripto definiendo ciertas relaciones entre conjuntos de objetos, a partir de las cuales otras pueden ser calculadas empleando deducción.**
* **Programación funcional  
  Se basa en la evaluación de expresiones, no en la ejecución de instrucciones.**

Programación Orientada a Objetos

Se construyen modelos de objetos que representan elementos (objetos) del problema a resolver, que tienen características y funciones. Permite separar los diferentes componentes de un programa. Se acerca de alguna manera a cómo expresaríamos las cosas en la vida real.

Programación Reactiva

Se basa en escuchar lo que emite un evento o cambios en el flujo de datos, en donde los objetos reaccionan a los valores que reciben de dicho cambio.

Los sistemas reactivos deben ser:

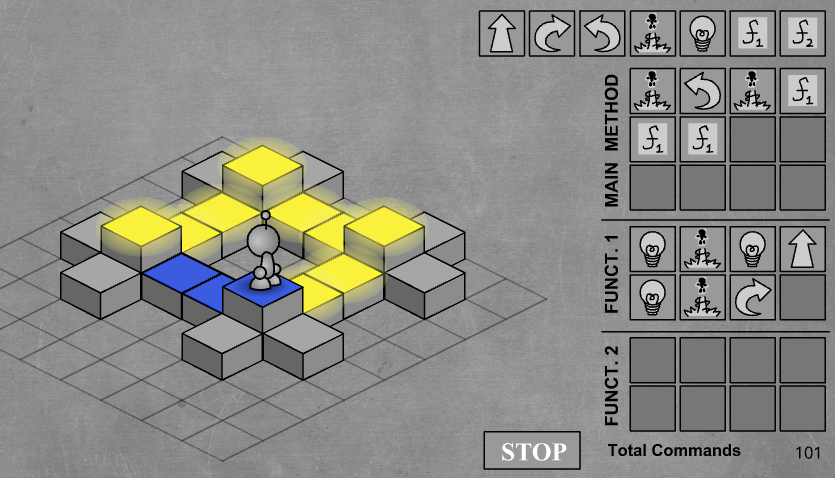
* **Responsivos:** aseguran la calidad del servicio cumpliendo unos tiempos de respuesta establecidos.
* **Resilientes:** se mantienen responsivos incluso cuando se enfrentan a situaciones de error.
* **Elásticos:** se mantienen responsivos incluso ante aumentos en la carga de trabajos.
* **Orientados a mensajes**: minimizan el acoplamiento entre componentes al establecer interacciones basadas en el intercambio de mensajes de manera asíncrona.

1. Comunicación y lenguaje

**Comunicación:** Transmisión de señales de un emisor y a un receptor mediante un código común  (el lenguaje).

1. Órdenes mediante símbolos (lightbot)

Un símbolo carga un significado. Una computadora es una máquina que los interpreta, gracias a los significados puestos por el humano.



* **Simbolos primitivos**Contienen un significado diseñado por la persona que programó el juego.
* F1 y F2 (Funciones 1 y 2)

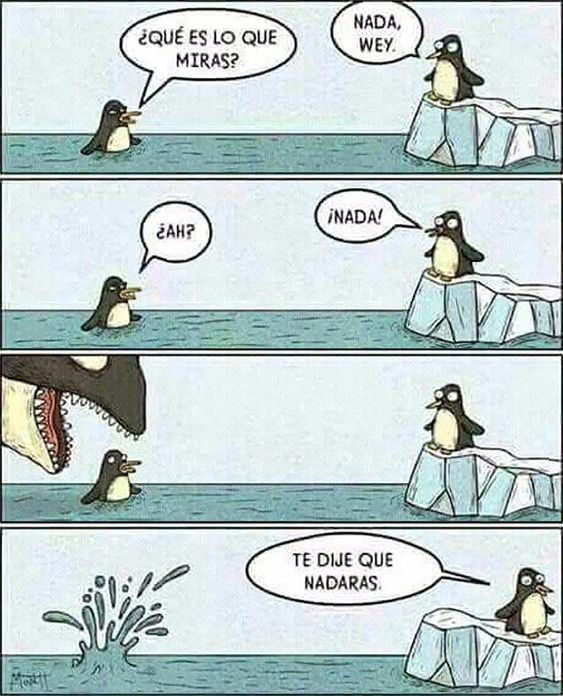
Símbolos primitivos que tendrán el significado diseñado por nosotros, la persona que programa el comportamiento del personaje

* Programa principal  
  Acá determinamos cuál será el comportamiento del personaje
* Funciones o procedimientos  
  Acá determinamos cuál será el comportamiento de nuestras funciones y, por lo tanto del personaje.

Sintaxis: modo estructurado de combinar los símbolos y de asignarles significados.

1. Lenguaje y ambigüedad

El texto es la evolución del símbolo. Una combinación de estos símbolos amplía las posibilidades. Podemos ahora codificar un mensaje en este lenguaje textual.

Lo textual en lo cotidiano

Cuando alguien dice “¡a comer!, siéntense en la mesa” no quiere que se sienten “en” la mesa, sino en las sillas que hay alrededor.   
  
“Tomate un taxi”, “el nene no me come”, entre otras frases, invitan a una doble interpretación.

El cerebro humano puede interpretar estas ambigüedades, resignificarlas hasta que tengan sentido dentro del contexto —cena, estar en la calle, charlando con el dr.—.

Las computadoras —por el momento— no pueden hacer eso. Son máquinas muy complejas que necesitan reglas simples. No debe haber ambigüedad.