

FULL STACK DEVELOPER BOOTCAMP

Módulo 1

PENSAMIENTO COMPUTACIONAL



TEMARIO

- Lógica
- Premisa
- Sesgo
- Silogismo
- Falacias
- Algoritmos
- Tipos de algoritmos

Objetivos del módulo

- Aprender las dimensiones del pensamiento lógico.
- Adquirir capacidades para el pensamiento secuencial y proposicional.

Lógica

La lógica comprende la aplicación de una serie de técnicas para la construcción y el desarrollo del pensamiento.

Tiene como objeto de estudio de los **principios formales** que permiten establecer un razonamiento correcto.

Existe un tipo de lógica, llamada binaria, que trabaja con variables que sólo toman dos valores discretos.

SI/NO

OFF/ON



Con la lógica tratamos de establecer la validez de que una conclusión se siga de un conjunto de premisas a partir del razonamiento que utilizamos en la vida cotidiana.

Pero en nuestro desafío de pensar computacionalmente y en mayores grados de abstracción, la lógica binaria de representación es la base de las memorias de las computadoras y de sus sistemas de operaciones.

La lógica nos permite también construir un lenguaje discreto, algorítmico que establecerá nuestro modo de decir qué hacer y los pasos para hacerlo a las computadoras

Premisa

***Todo enunciado de lo que
es posible señalar que es
verdadero o falso***

Elementos básicos del razonamiento lógico

- a. El pensamiento lógico es **deductivo**.
- b. Es **analítico** porque segmenta toda la información que se posee
- c. El pensamiento lógico es **racional** y no fantasioso o imaginativo.
- d. Es **preciso y exacto**.
- e. Es un pensamiento que se desarrolla de forma **lineal y secuencial**, es decir, paso a paso hasta alcanzar una conclusión.

Principios del pensamiento lógico

- f. **El principio de no contradicción.** Según el cual algo no puede ser y no ser a la vez (A y $\neg A$ no pueden ser ciertos a la vez).
- g. **El principio de identidad.** Según el cual algo siempre es idéntico a sí mismo (A siempre es igual a A).
- h. **El principio del tercero excluido.** Según el cual algo es o no es verdadero, sin que existan gradaciones posibles (A o entonces $\neg A$)
<https://concepto.de/logica/#ixzz7NSbunQ9d>

El pensamiento lógico está compuesto por premisas que se recopilan, organizan y luego se genera una conclusión.

- Para salir de viaje con mis amigos necesito tener suficiente dinero, si ahorro todos los meses parte de mi sueldo, entonces podré viajar con ellos.
- En el noticiero del clima dijeron que hay un 50% de probabilidades de que llueva. Cuando salga de casa para el trabajo me llevaré el paraguas.
- Todos los seres vivos necesitan alimentarse para vivir. Yo soy un ser vivo, y debo alimentarme todos los días para tener energía y llevar a cabo mis actividades diarias.

Silogismo

El **silogismo** tiene **dos premisas (mayor y menor)**, a partir de las cuales se llega a **una conclusión**.

Es la base del razonamiento deductivo, ya que partiendo de dos juicios se infiere uno nuevo.

La “premisa mayor”, es aquella que sirve de punto de partida, y es la más general; por su parte, la “premisa menor” sirve de intermediario y es menos general, y de ellas dos se deduce la conclusión del razonamiento.

En base a la anterior información, el ejemplo más clásico del silogismo es:

- Todos los hombres son mortales. (Premisa mayor)
- Pedro es hombre. (Premisa menor)
- Luego, Pedro es mortal. (Conclusión)

Si quieren seguir profundizando:

<https://www.significados.com/silogismo>

Silogismo

Reglas para su validez:

- Un silogismo contiene tres proposiciones.
- En dos premisas negativas no se puede concluir nada.
- En dos premisas positivas no puede obtenerse una conclusión negativa.
- Si una premisa es negativa, la conclusión es negativa, y viceversa.
- De dos premisas particulares no se saca conclusión.
- El término medio no puede entrar en la conclusión.

Ejercicio

En una de las tres cajas hay un tesoro, la única ayuda que dispones es saber que uno y solo uno de los letreros está mal. ¿Dónde está el tesoro?



I
El tesoro está aquí



II
El tesoro no está en este cofre



III
El tesoro no está en el
cofre del centro

Repaso de introducción a falacias



Falacias formales

Afirmación del consecuente



Si A entonces B, B por lo tanto A

Falacias formales

Generalización apresurada



No hay suficiente pruebas como para llegar a esa conclusión

Falacias No formales

Ad Hominem



*Se ataca a la persona
que presenta el
argumento y no al
argumento en sí.*

Falacias No formales

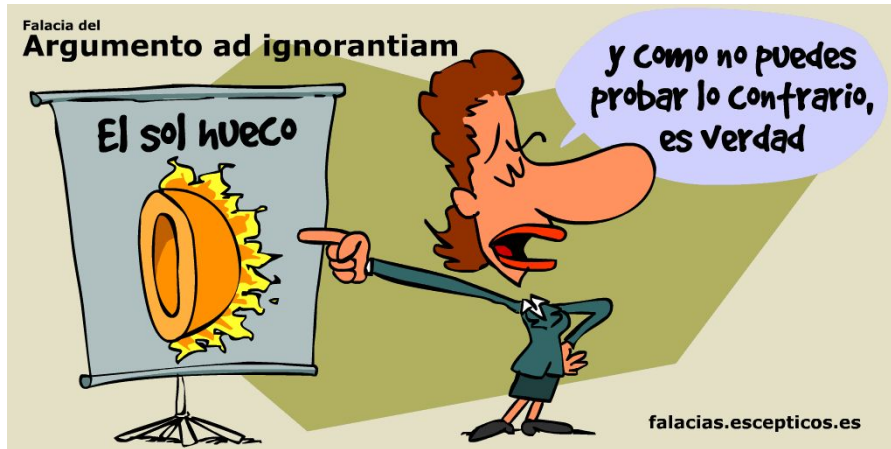
Falacia post hoc ergo propter hoc



Un autor comete esta falacia cuando asume que, dado que un acontecimiento sucede a otro, aquél fue causado por éste.

Falacias No formales

Falacia ad ignorantiam



Como no hay pruebas que nieguen el argumento, el argumento es verdadero.

Falacias No formales

Falacia ad baculum



Se plantean consecuencias negativas si no se comparte el argumento.

Algoritmos

PENSAMIENTO COMPUTACIONAL



A modo de introducción



De dónde viene la palabra algoritmo

La palabra algoritmo proviene del nombre del matemático Abu Jafar Muhamed ibn Musä al Khwàrizm quien nació en 780 A.C. en Bagdad y murió en 850 A.C. Fue tal el trabajo de este matemático, que se bautizó al método estructurado para encontrar soluciones en honor a su nombre.

Pero entonces ¿Qué es?

Es un **conjunto ordenado y finito de operaciones** que deben seguirse para resolver un problema.

Es un conjunto ordenado de operaciones, lo que quiere decir qué es una cadena de instrucciones precisas que deben seguirse por orden. (Receta de Cocina)











Su objetivo es resolver un problema, lo que significa que tiene un objetivo delimitado.

Algoritmo

Cuando un desarrollador crea un programa, en esencia lo que está creando es un conjunto de algoritmos. Un programa de ordenador es un conjunto de órdenes que se dan a la máquina, escritas en un lenguaje concreto, para que realice una serie de operaciones determinadas con el fin de obtener un resultado.

Diagrama de flujo

Se utilizan símbolos gráficos que representan cada paso del algoritmo. Es la notación más usada para escribir algoritmos. Son fáciles de leer y son excelentes auxiliares para describir procesos.

Símbolo	Significado
	Inicio / Fin del Algoritmo
	Entrada de Datos
	Proceso
	Salida
	Selección Simple/Doble
	Conector en la misma hoja
	Conector en otra hoja
	Ciclo "para"
	Subrutina
	Selección múltiple

Algoritmo, paso previo a escribir

En programación, definir el algoritmo supone el paso previo a ponerse a escribir el código. Primero debemos encontrar la forma de obtener la solución al problema (definir el algoritmo informático), para luego, a través del código, poder indicarle a la máquina qué acciones queremos que lleve a cabo. De este modo, un programa informático no sería más que un conjunto de algoritmos ordenados y codificados en un lenguaje de programación para poder ser ejecutados en un ordenador.

Partes de un algoritmo

Las tres partes de un algoritmo son:

1. **Input (entrada).** Información que damos al algoritmo con la que va a trabajar para ofrecer la solución esperada.
2. **Proceso.** Conjunto de pasos para que, a partir de los datos de entrada, llegue a la solución de la situación.
3. **Output (salida).** Resultados, a partir de la transformación de los valores de entrada durante el proceso.

Características de los algoritmos

- **Precisos.** Objetivos, sin ambigüedad.
- **Ordenados.** Presentan una secuencia clara y precisa para poder llegar a la solución.
- **Finitos.** Contienen un número determinado de pasos.
- **Concretos.** Ofrecen una solución determinada para la situación o problema planteados.
- **Definidos.** El mismo algoritmo debe dar el mismo resultado al recibir la misma entrada.

Tipos de algoritmos

Existen diversas clasificaciones de algoritmos, en función de diferentes criterios. Según su sistema de signos (cómo describen los pasos a seguir), se distingue entre algoritmos cuantitativos y cualitativos, si lo hacen a través de cálculos matemáticos o secuencias lógicas. Asimismo, si requieren o no el empleo de un ordenador para su resolución, se clasifican en computacionales y no computacionales.



<https://profile.es/blog/que-es-un-algoritmo-informatico/>

Principios de programación

Hay una diferencia entre saber programar y saber escribir código.

- Saber escribir código es, saber la sintaxis del lenguaje, saber las posibilidades del lenguaje, etc.
- Mientras que saber programar es saber pensar como programador, pensar en algoritmos.

¿Cómo podemos aprender a pensar como programador o por lo menos practicarlo?

Tratar de pensar todo como algoritmos, ejemplo, si quisiéramos describir la acción de salir de la cama y llegar al trabajo (En una época sin pandemia).

El algoritmo sería el siguiente:

1. Salir de la cama
2. Quitarle el pijama
3. Darse una ducha
4. Vestirse
5. Desayunar
6. Salir de casa y viajar al trabajo.

Pseudocódigo

Escribir pseudocódigo es escribir instrucciones en lenguaje natural de forma de poder luego traspasarlas cómodamente al lenguaje de programación que corresponda.

Si bien en la diaria no se utiliza este lenguaje de programación, es bueno conocerlo a esta altura para poder tratar de pensar todo de forma lógica y poder luego pasarlo de forma eficaz al código.

Ejemplo

Consideremos el siguiente enunciado:

Una clase de diez alumnos hizo un examen. Las calificaciones (enteros en el rango de 0 a 100) correspondientes a este examen están a su disposición. Determine el promedio de la clase en este examen.

Sabemos que el promedio de algo es la suma de todos sus casos (en este caso suma de calificaciones), dividido la cantidad de calificaciones sumadas.

Por lo que sabemos que vamos a sumar valores a un total, que vamos a iterar sobre un listado de alumnos y luego vamos a imprimir en pantalla el resultado. Por lo tanto en pseudocódigo tenemos lo siguiente:

- Inicializar total en 0
- Inicializar nro_alumno a 1
- Mientras nro_alumno menor o igual a 10
 - Leer siguiente calificación
 - Sumar calificación a total
 - sumar 1 a nro_alumno
- Calcular promedio = total / nro_alumno
- Imprimir promedio

```
const qualifications = [5,2,3,7,8,9,10,11,1,10]
let total = 0;
let counter = 1;
while (counter <= 10){
  total += qualifications[counter -1] // Los arreglos arrancan en 0
  counter ++;
}
let average = total/counter;
console.log(average)
```

Diagramas de flujo

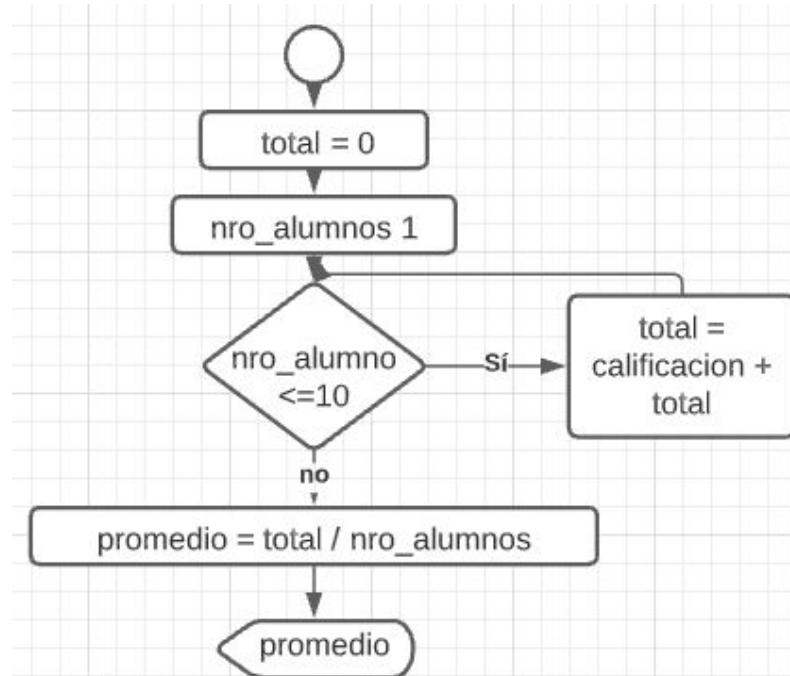
Otra forma de visualizar código antes de escribirlo es por intermedio de diagramas de flujo:

Ejemplo

Consideremos el siguiente enunciado:

Una clase de diez alumnos hizo un examen. Las calificaciones (enteros en el rango de 0 a 100) correspondientes a este examen están a su disposición. Determine el promedio de la clase en este examen.

- Inicializar total en 0
- Inicializar nro_alumno a 1
- Mientras nro_alumno menor o igual a 10
 - Leer siguiente calificación
 - Sumar calificación a total
 - sumar 1 a nro_alumno
- Calcular promedio = $\text{total} / \text{nro_alumno}$
- Imprimir promedio





¡GRACIAS!