

Módulo 1. Lógica y algoritmos



A continuación, desarrollaremos los contenidos que se vinculan con los algoritmos y la lógica. En la primera parte, veremos características de los algoritmos, qué partes los componen, cuáles son los tipos de algoritmos y los pasos de solución de problemas. En relación con la lógica, estudiaremos la lógica proposicional, el álgebra de Boole, las leyes de Morgan y las funciones lógicas.

≡ Video de inmersión

≡ Unidad 1. Lógica

≡ Unidad 2. Algoritmos

≡ Video de habilidades

≡ Cierre

≡ Referencias


Video de inmersión

Verify to continue

We detected a high number of errors from your connection. To continue, please confirm that you are human (not a spambot).

vimeo

I'm not a robot

 reCAPTCHA
Privacy - Terms

CONTINUAR

Unidad 1. Lógica

La lógica puede definirse como la disciplina que formaliza el estudio de los métodos de razonamiento. Es un conjunto de técnicas y teoremas que nos permiten modelar y definir formas de razonar. Las estructuras del pensamiento y las reglas que las gobiernan se estudian en la lógica.

Los objetivos fundamentales de la lógica son los siguientes:

- eliminar la ambigüedad del lenguaje natural.
- Establecer reglas que determinen la validez de un razonamiento.

En informática, la lógica es de mucha importancia, ya que es imprescindible que seamos capaces de crear proposiciones para que el computador las pueda ejecutar. Recuerda que el computador no es capaz de interpretar, por lo que, mientras menor sea la ambigüedad, mejor serán nuestros códigos.

Tema 1. Lógica

Existen dos conceptos fundamentales en lógica: el juicio y la proposición. El juicio es el acto mental por medio del cual pensamos un determinado enunciado, mientras que la proposición es lo que se piensa en dicho acto. Una proposición puede definirse como el significado de un pensamiento.


Para representar una proposición, se utilizan sentencias, una serie de signos con un significado determinado.

Una característica fundamental de una proposición es que se debe poder definir si es verdadera o falsa.

Por ejemplo, veamos las siguientes oraciones:

- La hoja es blanca.
- ¿Qué es el océano?
- Juan sabe leer.
- ¡Pórtese bien!

Analizando cada una de las oraciones, solo podemos decir que son verdaderas o falsas la primera y la tercera, ya que la hoja puede ser blanca o no, y Juan puede saber leer o no. La segunda oración, al ser una pregunta, no afirma nada sobre ninguna cosa; por último, la cuarta oración es una orden que puede ser cumplida o no, pero, al igual que la pregunta, no afirma nada sobre ninguna cosa.

 Una proposición lógica es toda oración declarativa que pueda decirse si es verdadera o falsa.

Es muy importante indicar que una proposición nunca puede ser verdadera y falsa. Esto se corresponde con los principios fundamentales de la lógica proposicional:

Principio de no contradicción —

Dadas dos proposiciones contradictorias entre sí, no pueden ser ambas verdaderas. Por ejemplo, si tenemos una proposición que indica que -1 es un valor negativo y otra que indica que -1 es un valor

positivo, solo una de estas oraciones es verdadera, no pueden ser verdaderas ambas, ya que son contradictorias.

Principio de tercero excluido —

Dadas dos proposiciones contradictorias entre sí, no pueden ser ambas falsas. Siguiendo con el ejemplo anterior, como ambas proposiciones son contradictorias, no puede ser que ambas sean falsas a la vez; por lo tanto, una es verdadera y la otra es falsa.

Principio de identidad —

Toda proposición es idéntica en sí misma. Este es, sin duda, el principio más complejo, sin embargo, nos dice algo sumamente simple, porque indica que cada proposición es igual a sí misma. Por ejemplo, el círculo es un círculo y no es un rectángulo, o un auto es un auto y no una bicicleta. Toda proposición tiene una identidad propia.

El objetivo de la lógica proposicional no es determinar si algo es verdadero o falso, sino más bien las estructuras en la que las proposiciones intervienen y las reglas que regulan las combinaciones. Por ejemplo:

- el elefante es un mamífero.
- 5 es un entero.
- La luna gira en torno a la tierra.

Las proposiciones anteriores son de tipo simple, ya que pueden ser ciertas o no, pero estudiar si son realmente ciertas no es responsabilidad de la lógica, sino más bien de la ciencia correspondiente.

i Los valores de verdad indican dos posibles estados: verdadero o falso. Cuando hablamos de valores de verdad de una proposición, nos referimos a si esta es verdadera o falsa.

Ejemplos

Revisemos algunos ejemplos para identificar el valor de verdad de una proposición.

Tabla 4: Ejemplos de proposiciones

Proposición	Valor de Verdad	Observación
Todas las personas hablan español.	Falso	No todas las personas en el mundo hablan español, ya que existen muchos otros idiomas.
Algunas personas hablan inglés.	Verdadero	Existen algunas personas que saben hablar en inglés.
¿Hablas inglés?	N/A	Esta frase no es una proposición, ya que no es posible identificar si es verdadero o falso.

Fuente: elaboración propia.


Las proposiciones simples no siempre son suficientes para expresar lo que necesitamos; por este motivo, también existen las proposiciones complejas.

Las proposiciones simples pueden combinarse de diferentes maneras para generar proposiciones más complejas, utilizando, entre ellas, determinados operadores llamados conectores lógicos. Por ejemplo:

Si termino los trabajos prácticos a tiempo, **entonces** puedo ir al cine.

De lo anterior, podemos identificar dos proposiciones:

- termino los trabajos prácticos a tiempo.
- Puedo ir al cine.

 Una proposición compuesta es la combinación de proposiciones simples mediante conectivos lógicos.

¿Cómo determinamos si una proposición compuesta es verdadera o falsa? Depende de los valores de verdad asignados a las proposiciones simples que la componen y de la forma en la cual se conectan, es decir, de los conectores lógicos.

Para poder realizar operaciones sobre las proposiciones mediante conectores lógicos, se utilizan variables proposicionales. Estas variables son simplemente formas simples de representar una proposición, como seudónimos. Por ejemplo:

p: Juan sabe hablar inglés.

La variable p representa la proposición **Juan sabe hablar inglés**. Ahora, cada vez que nos referimos a esta oración, podemos utilizar simplemente p y no tener que escribirla por completo.

La variable p puede ser verdadera (representado por una letra **V**) o falsa (representado por una letra **F**). Podemos utilizar una tabla denominada tabla de verdad, para ver todos los posibles valores que puede tener una proposición:

Tabla 5: Tabla de verdad de la variable proposicional

p
Verdadero.
Falso

Fuente: elaboración propia.

Como lo vimos antes, una proposición solo puede tener dos valores verdaderos (V) o falso (F); por esta razón, la tabla de verdad de una proposición simple solo tiene estas dos alternativas.

Negación

El conector lógico de negación invierte el valor de verdad de una proposición. Cuando aplicamos este conector a una proposición lógica verdadera, el resultado es falso. Cuando lo aplicamos a una proposición lógica falsa, el resultado es verdadero.

i Dada una proposición lógica p , se define como negación de p a la proposición $\neg p$ (que se lee como no p), que es verdadera si p es falsa, y falsa si p es verdadera.

Por ejemplo, dada la proposición p definida:

p : Juan sabe hablar inglés.

¿Cuál es $\neg p$ (no p)?

La respuesta es que $\neg p$ va a depender del valor de p . Los posibles valores se encuentran en la tabla de verdad:

Tabla 6: Tabla de verdad de p y $\neg p$

p	$\neg p$
V	¿?
F	¿?

Fuente: elaboración propia.

La negación de p es falsa si p es verdadera y verdadera si p es falsa, porque siempre es el valor contrario al que posee la variable, por lo que la

tabla de verdad se completaría de la siguiente forma:

Tabla 7: Tabla de verdad de p y $\neg p$ resuelta

p	$\neg p$
V	F
F	V

Fuente: elaboración propia.

Finalmente, $\neg p$ puede ser representada de la siguiente forma:

$\neg p$: Juan no sabe hablar inglés.

Cuando indicamos la expresión «hoy no es un día de verano», en realidad, la proposición es «hoy es un día de verano», que se encuentra negada por el operador «no»; por lo tanto, quedaría de la siguiente forma:

- p : hoy es un día de verano.
- $\neg p$: hoy no es un día de verano.

Conjunción

Este operador conecta dos proposiciones simples mediante el término y. Esta es una de las formas más comunes de combinar oraciones en nuestro lenguaje. Por ejemplo:


A Lucas le gusta caminar y a José le gusta correr.

Podemos encontrar dos proposiciones simples en la oración anterior:

- p: A Lucas le gusta caminar.
- q: A José le gusta correr.

La conjunción utiliza el símbolo \wedge ; de este modo, las proposiciones anteriores podrían ser representadas de la siguiente forma:

$$p \wedge q.$$

 Dadas dos proposiciones p y q, llamaremos conjunción de p y q a la proposición $p \wedge q$ (que se lee p y q), que es verdadera si ambas, p y q, lo son, o es falsa en todos los otros casos.

Siguiendo con el ejemplo anterior, podemos elaborar la siguiente tabla de verdad:

Tabla 8: Tabla de verdad de $p \wedge q$

p	q	$p \wedge q$
V	V	¿?

F	F	¿?
V	F	¿?
F	V	¿?

Fuente: elaboración propia.

Como ahora tenemos dos proposiciones, las posibles combinaciones son mayores. Mientras más proposiciones tengamos, mayores serán las posibles combinaciones. La cantidad exacta es 2 elevado al número de proposiciones a combinar. En este caso 2^2 , lo que nos da un total de 4 filas.

Para resolver la tabla anterior, debemos volver a la definición de conjunción. El resultado es verdadero solamente cuando las dos proposiciones son verdaderas, para todos los demás casos el resultado es falso.

Tabla 9: Tabla de verdad de $p \wedge q$ resuelta

p	q	$p \wedge q$
V	V	V
F	F	F
V	F	F
F	V	F

Fuente: elaboración propia.

Disyunción


La disyunción une dos proposiciones simples mediante el término o. Por ejemplo:

Andrés viaja a Londres o viaja a París.

La proposición anterior está compuesta por dos proposiciones:

- p: Andrés viaja a Londres.
- q: Andrés viaja a París.

El valor de verdad de la proposición compuesta depende de los valores de verdad de las dos proposiciones. El resultado de la disyunción $p \vee q$ será falso solamente cuando ambas sean falsas, y verdadero en todos los otros casos. Se comporta de forma contraria a la conjunción, en esta oportunidad basta con que una de las dos proposiciones simples sea verdadera para que la proposición compleja sea verdadera.

 Dadas dos proposiciones p y q, llamaremos disyunción de p y q a la proposición $p \vee q$ (que se lee p o q), que es falsa si ambas, p y q, lo son, o es verdadera en todos los otros casos.

La tabla de verdad para este ejemplo sería de la siguiente forma:

Tabla 10: Tabla de verdad de $p \vee q$

p	q	$p \vee q$
V	V	¿?
F	F	¿?
V	F	¿?
F	V	¿?

Fuente: elaboración propia.

Para encontrar los resultados, debemos remitirnos a la definición de disyunción. El resultado es falso únicamente cuando ambas proposiciones son falsas, para lo demás el resultado es verdadero.

Tabla 11: Tabla de verdad de $p \vee q$ resuelta

p	q	$p \vee q$
V	V	V
F	F	F
V	F	V
F	V	V

Fuente: elaboración propia.

Resumen

Veamos a continuación un resumen de los conectores vistos:

Tabla 12: Resumen de conectores proposicionales

Conectivo	Símbolo	Operación	Significado
No	\neg	Negación	No es el caso de p
Y	\wedge	Conjunción	p y q
O	\vee	Disyunción	p o q

Fuente: elaboración propia.

Ejemplos

Podemos realizar combinaciones entre los diferentes conectores. Tomemos, por ejemplo, la siguiente frase para la aprobación de una asignatura: «tener promedio sobre 6, o tener promedio sobre cuatro, y no tener notas bajo 4».

Existen 3 proposiciones incluidas dentro de la expresión:

- tener promedio sobre 6.

- Tener promedio sobre 4.
- No tener notas bajo 4.

La última de estas proposiciones se encuentra negada:

- **no** tener notas bajo 4.

Asignaremos una variable a cada proposición:

- p = Tener promedio sobre 6.
- q = Tener promedio sobre 4.
- $\neg z$ = **No** tener notas bajo 4.

En este caso, queda de la siguiente forma:

- $p \vee q \wedge \neg z$.

Por último, debemos identificar que la segunda y la tercera proposición se encuentran unidas, ambas deben cumplirse para juntas para poder aprobar; en este caso, utilizamos paréntesis para poder agruparlas. La expresión queda de la siguiente manera:

- $p \vee (q \wedge \neg z)$.

Tema 2. Álgebra

El álgebra de Boole nació a mediados del siglo XIX, cuando George Boole presentó dos libros que explicaban la idea de tratar las proposiciones lógicas utilizando las matemáticas.

Se puede definir como un sistema matemático basado en los valores **1 (verdadero) y 0 (falso)**, en conjunto con las operaciones AND, OR y NOT. Las variables en este sistema no pueden tomar valores diferentes a 1 y 0, y se denominan variables booleanas.

Lo más importante para tener en cuenta cuando estudiamos álgebra es que nos ayuda a desarrollar una capacidad de razonamiento lógica; esto es fundamental a la hora de programar.

Operación AND

Esta operación se corresponde con la conjunción (y) de las proposiciones lógicas. La operación lógica AND se representa con el símbolo \cdot . Lo que corresponde a una multiplicación Lógica.

Por ejemplo, dados dos valores booleanos $A = 1$ y $B = 0$, ¿cuál es el resultado de la operación $A \cdot B$?

A la hora de resolver operaciones booleanas, podemos utilizar el siguiente procedimiento:

1

transcribir la operación que queremos realizar:

$$A \cdot B = \text{¿?}$$

2

Reemplazar las variables por sus valores; en este caso, A es 1 y B es 0:

$$1 \cdot 0 = \text{¿?}$$

3

Ejecutar la operación. En este ejemplo, la operación es AND, que corresponde a la conjunción en proposiciones lógicas. Recordemos que 1 significa verdadero y 0 significa falso. Recordemos también que el resultado de una conjunción es verdadero solo si ambas proposiciones son verdaderas, y falso en los otros casos. En este caso, sería lo siguiente:

$$1 \text{ (verdadero)} \cdot 0 \text{ (falso)} = 0 \text{ (falso)}.$$

Veamos otro ejemplo; en este caso $A = 1$ y $B = 1$. Sigamos los pasos definidos anteriormente:

1

transcribir la operación que queremos realizar:

$$A \cdot B = \text{¿?}$$

2

Reemplazar las variables por sus valores, en este caso A es 1 y B es 1:

$$1 \cdot 1 = \text{¿?}$$

3

Ejecutar la operación. En este ejemplo, la operación es AND, que corresponde a la conjunción en proposiciones lógicas. Recordemos que 1 significa verdadero y 0 significa falso. Recordemos también que el resultado de una conjunción es verdadero solo si ambas proposiciones son verdaderas, y falso en los otros casos. En este caso, sería lo siguiente:

$$1 \text{ (verdadero)} \cdot 1 \text{ (verdadero)} = 1 \text{ (verdadero)}.$$



La conjunción u operación AND corresponde a una multiplicación lógica, es decir, que cada vez que multipliquemos por 0 (falso) el resultado será 0. Por lo tanto, la única oportunidad de que el resultado sea 1 (verdadero) es que ambas variables sean 1.

Operación OR

Esta operación corresponde con la disyunción de las proposiciones lógicas. La operación lógica OR se representa con el símbolo +. Lo que corresponde a una suma lógica.

Por ejemplo, dados dos valores booleanos $A = 1$ y $B = 0$, ¿cuál es el resultado de la operación $A + B$?

Utilizamos el procedimiento definido en la sección anterior:

1

transcribir la operación que queremos realizar:

$$A + B = \text{¿?}$$

2

Reemplazar las variables por sus valores, en este caso A es 1 y B es 0:

$$1 + 0 = \text{¿?}$$

3

Ejecutar la operación. En este ejemplo, la operación es OR, que corresponde a la disyunción en proposiciones lógicas. Recordemos que 1 significa verdadero y 0 significa falso. Recordemos también que el resultado de una disyunción es falso solo si ambas proposiciones son falsas, y verdadero en los otros casos. En este caso, sería lo siguiente:

$$1 \text{ (verdadero)} + 0 \text{ (falso)} = 1 \text{ (verdadero)}.$$

En el caso que $A = 0$ y $B = 0$, ¿cuál sería el resultado?

1

Transcribir la operación que queremos realizar:

$$A + B = \text{¿?}$$

2

Reemplazar las variables por sus valores, en este caso A es 0 y B es 0:

$$0 + 0 = \text{¿?}$$

3

Ejecutar la operación. En este caso, la operación es OR, que corresponde a la disyunción en proposiciones lógicas. Sería lo siguiente:

$$0 \text{ (falso)} + 0 \text{ (falso)} = 0 \text{ (falso)}.$$



La disyunción u operación OR corresponde a una suma lógica, es decir, que cada vez que sumemos 1 (verdadero) el resultado será 1. Por lo tanto, la única oportunidad de que el resultado sea 0 (falso) es que ambas variables sean 0.

Operación NOT


Esta operación se corresponde con la negación de las proposiciones lógicas. La operación lógica NOT se representa con el símbolo \neg . Por ejemplo, dado el valor booleano $A = 1$, ¿cuál es el resultado de la operación $A \neg$?

Utilizamos el procedimiento definido en la sección anterior:

- 1 transcribir la operación que queremos realizar:
 $A' = ?$
- 2 Reemplazar las variables por sus valores, en este caso A es 1:
 $1' = ?$
- 3 Ejecutar la operación. En este ejemplo, la operación es NOT, que corresponde a la negación en proposiciones lógicas. Recordemos que 1 significa verdadero y 0 significa falso. Recordemos también que el resultado de una negación es falso si la proposición es verdadera, y verdadera si la proposición es falsa. En este caso, sería lo siguiente:
 $1 \text{ (verdadero)}' = 0 \text{ (falso)}$

En el caso de que $A = 0$, ¿cuál sería el resultado?

- 1 Transcribir la operación que queremos realizar:
 $A' = ?$
- 2 Reemplazar las variables por sus valores, en este caso A es 0:
 $0' = ?$
- 3 Ejecutar la operación. En este ejemplo, la operación es NOT, que corresponde a la negación en proposiciones lógicas. Sería lo siguiente:
 $0 \text{ (falso)}' = 1 \text{ (verdadero)}$

 La operación NOT establece que, dada una variable booleana cuyo valor es igual a 1, el resultado de aplicar esta operación será 0. Si la variable es igual a 0, el resultado de aplicar esta operación será igual a 1.

Tema 3. Leyes

En el siglo XIX, August De Morgan, un matemático y lógico británico, formuló dos leyes utilizando el álgebra de Boole:

1

la negación de la conjunción es la disyunción de las negaciones.

2

La negación de la disyunción es la conjunción de las negaciones.

Esto se interpreta de la siguiente manera, dadas dos variables booleanas, A y B:

1

$(A \text{ AND } B)'$ es igual a $A' \text{ OR } B'$.

2

$(A \text{ OR } B)'$ es igual a $A' \text{ AND } B'$.

Veamos un ejemplo para la primera regla. Dado $A = 1$ y $B = 0$, resolvemos la primera de la ecuación:

- escribimos la ecuación:
 $(A \text{ AND } B)'$ es igual a $A' \text{ OR } B'$.
- Reemplazamos valores:
 $(1 \text{ AND } 0)'$ es igual a $1' \text{ OR } 0'$.
- Realizamos operaciones:
 $0'$ es igual a 1 OR 1.1 es igual a 1.

Con lo que queda demostrada la primera regla. Veamos la segunda para los mismos valores:

- escribimos la ecuación:
 $(A \text{ OR } B)'$ es igual a $A' \text{ AND } B'$.
- Reemplazamos valores:
 $(1 \text{ OR } 0)'$ es igual a $1' \text{ AND } 0'$.
- Realizamos operaciones:
 $1'$ es igual a 0 AND 1.0 es igual a 0.

Con lo anterior, demostramos la validez de la segunda ley de Morgan.

Leyes de Morgan

$$(A \text{ AND } B)' = A' \text{ OR } B'$$

$$(A \text{ OR } B)' = A' \text{ AND } B'$$

Tema 4. Funciones

Hasta ahora, hemos trabajado solamente con dos variables, a modo de simplificar lo más posible los teoremas y las propiedades descritas. Pero existe la posibilidad de tener más de dos variables y, también, tener más de dos operadores.

Una función lógica es la expresión que define las operaciones para realizar sobre un conjunto de variables, para obtener un resultado.

Por ejemplo, dada la siguiente función:

- $s = (A + B) \cdot [(C \cdot A) + (D \cdot B')]$.

En el ejemplo anterior, tenemos la función s definida por un conjunto de operaciones sobre las variables A , B , C y D .

Las funciones lógicas se resuelven de la misma manera que hemos estado operando con variables y operadores booleanos hasta ahora. Para resolver la función, primero debemos obtener los valores de las variables; en este caso, serán los siguientes:

- $A = 0$.

- $B = 0$.

- $C = 1$.

- $D = 0$.

Ahora que tenemos los valores de las variables, procedemos a resolver la función:

- escribimos la función:
 $s = (A + B) \cdot [(C \cdot A) + (D \cdot B')]$.

- Reemplazamos valores:
 $s = (0 + 0) \cdot [(1 \cdot 0) + (0 \cdot 0')]$.

- Realizamos operaciones:
 $s = (0 + 0) \cdot [(1 \cdot 0) + (0 \cdot 0')]$.

$$s = 0 \cdot [0 + (0 \cdot 1')].$$

$$s = 0 \cdot [0 + (0 \cdot 0)].$$

$$s = 0 \cdot [0 + 0].$$

$$s = 0 \cdot 0.$$

$$s = 0.$$

Hemos resuelto la ecuación, y hemos llegado a la conclusión de que su resultado es cero, de acuerdo con los valores definidos de las variables.

Existe cierta jerarquía a la hora de resolver una función, la cual detallamos a continuación:

1

primero resolvemos los paréntesis. En caso de existir subexpresiones, resolvemos de adentro hacia afuera. Por ejemplo $[(1 \cdot 0) + (0 \cdot 0')]$. En este caso, resolvemos $(1 \cdot 0)$ y $(0 \cdot 0')$.

2

Segundo, resolvemos el operador NOT, por ejemplo $(0 \cdot 1')$ nos quedaría $(0 \cdot 0)$.

3

Tercero, resolvemos los operadores OR y AND.

4


En caso de tener operadores del mismo nivel de jerarquía, debemos considerar la resolución de izquierda a derecha. Por ejemplo, $0 \cdot 0 + 0$, primero resolvemos el AND y luego el OR.

CONTINUAR

Unidad 2. Algoritmos

Los problemas son algo cotidiano en nuestra vida, tenemos que resolver situaciones tanto en nuestro trabajo (llegar a tiempo con la entrega de un determinado proyecto) como en lo personal (entender cuánto podemos gastar de acuerdo con nuestro salario). Estamos resolviendo problemas casi todo el tiempo y, aunque no nos demos cuenta, siempre seguimos un conjunto de pasos para resolverlos. Para entregar un proyecto a tiempo, primero vemos cuánto trabajo nos queda por hacer para verificar si realmente podemos entregarlo en la fecha establecida o, cuando calculamos cuánto podemos gastar, siempre miramos nuestros ingresos primero. Ese conjunto de pasos que utilizamos para resolver problemas se denominan algoritmos. Cuando los problemas son relativamente simples, podemos resolverlos mentalmente en pocos segundos; cuando esos problemas son más complejos (como crear un sistema de **software**), existen técnicas y procedimientos que nos ayudan a diseñar los pasos que deberíamos realizar para solucionarlos.

Podemos definir un algoritmo formalmente como es un conjunto definido de instrucciones, con un orden y una cantidad determinada de pasos (no existen algoritmos con infinitos pasos). Los algoritmos permiten llevar a cabo una actividad mediante pasos sucesivos que no generen dudas a quien deba realizar dicha actividad.

 Finito: cuando se menciona que algo es finito significa que tiene un final. Los algoritmos tienen que tener un paso final, de lo contrario, no es un algoritmo.

Tema 1. Características de los algoritmos

Si bien los algoritmos pueden ser muy diferentes de acuerdo con la complejidad y la estrategia de resolución del problema, comparten ciertas características. Estos deben cumplir con los siguientes aspectos:

- **Ordenados.** Los algoritmos no son solo el conjunto de pasos, se debe definir también la secuencia en la que dichos pasos se deben realizar. Por ejemplo, no es lo mismo poner en el horno la cena por media hora y luego encender el horno. Existe cierto orden que se debe cumplir para lograr el objetivo; en este caso, encender el horno y luego poner la cena por media hora.
- **Finitos:** los pasos de un algoritmo tienen que tener una cantidad determinada de pasos y, eventualmente, tener al menos un paso final. No existen algoritmos con un conjunto infinito de pasos. Esto sucede porque los algoritmos nos permiten obtener la solución a un problema y necesitamos llegar a su final para conseguirlo. Si no tuvieran un fin, nunca obtendríamos el resultado.
- **Definidos:** dado un problema determinado, si se sigue el mismo algoritmo, se debe llegar a los mismos resultados. En el caso de seguir el algoritmo para armar un mueble, por ejemplo, una mesa, por mucho que yo repita indefinidamente el algoritmo, siempre me entregará una mesa, nunca una silla. Los algoritmos tienen un objetivo a cumplir y esto se refleja en el resultado.

Por ejemplo, supongamos que queremos preparar un helado de frutilla, los pasos de la receta serían los siguientes:

1. cortar las frutillas en trozos pequeños.
2. Mezclar azúcar con frutillas.

3. Dejar reposar una hora.
4. Añadir leche, limón y azúcar.
5. Triturar con batidora.
6. Mezclar con crema de leche batida.
7. Colocar la mezcla en el congelador.
8. Esperar una hora y servir.

Veamos si el algoritmo anterior cumple con las características:

- **Posee un orden:** efectivamente, el algoritmo está ordenado por una secuencia de números que indican el orden que se debe seguir. Antes de añadir leche, limón y azúcar, debemos cortar las frutillas, mezclarlas con azúcar y dejarlas reposar.
- **Es finito:** el total de pasos es un número determinado, son 8 pasos en total y, después del último, obtenemos el resultado.
- **Es definido:** si realizamos estos pasos exactamente de la misma manera, obtendremos el mismo tipo de helado (frutilla).

Ahora sabemos que, cada vez que leemos una receta de cocina, estamos viendo un conjunto de pasos finitos y definidos, con un orden determinado: estamos viendo un algoritmo.

En el caso de la programación, los algoritmos son un poco más complejos, pero siguen las mismas bases que una receta. Tienen pasos, orden y un mismo resultado, si se ejecutan dos o más veces.

Veamos un algoritmo simple de programación. Tenemos que determinar los pasos para sumar dos números y, al resultado, multiplicarlo por dos (2). ¿Cuáles serían los pasos que se deben seguir?



- 1 Obtener el valor del primer número de la suma.
- 2 Obtener el valor del segundo número de la suma.
- 3 Realizar la operación suma entre dichos números.
- 4 Realizar la operación multiplicación entre el resultado y el número dos (2).

Por ejemplo, si el primer valor es tres (3) y el segundo uno (1), el algoritmo se comportaría de la siguiente manera:

- 1 obtener el valor del primer número de la suma: este número es igual a 3.
- 2 Obtener el valor del segundo número de la suma: este número es igual a 1.
- 3 Realizar la operación suma entre dichos números: $3 + 1 = 4$.
- 4 Realizar la operación multiplicación entre el resultado y el número dos (2): $4 \times 2 = 8$.

El resultado de aplicar el algoritmo con los números tres (3) y uno (1) es igual a ocho (8).

Programar la suma y multiplicación de tres números es algo muy distinto a elaborar helado de frutilla, pero podemos ver que ambos algoritmos tienen las mismas características: ambos tienen un número finito de pasos, dichos pasos están ordenados, y cada paso está definido de forma tal que cada vez que ejecutamos los algoritmos obtenemos los mismos resultados.


 Los algoritmos pueden estar tan definidos cuanto queramos; en este caso, utilizamos no más de dos renglones, pero cada paso puede ser explicado en el detalle necesario.

Figura 1: Características de un algoritmo

Ordenado

Finito

Definido

Tema 2. Partes de un algoritmo

Así como cada algoritmo posee las mismas características, también tiene las mismas partes. Para cada algoritmo, se puede definir que tendrá los siguientes elementos:

1

Entrada: los algoritmos siempre poseen una entrada; al ser nada más que una secuencia de pasos, se necesita una entrada a la cual aplicar los pasos. Son los insumos que le entregarás al algoritmo para que pueda comenzar a trabajar.

2

Proceso: el proceso refiere a los pasos definidos, en los que manipulamos la entrada para llegar a un resultado.

3

Salida: una vez que terminamos de aplicar todos los pasos, siempre obtenemos un resultado. El resultado de las operaciones o pasos de un algoritmo sobre una entrada específica se denomina salida.

Siguiendo con el ejemplo de la receta de helado, ¿cuáles serían las entradas de ese algoritmo? Los ingredientes son la entrada, sería muy difícil hacer helado de frutilla sin las frutillas o sin azúcar.

Todos los algoritmos deben tener una entrada, porque los pasos en sí no pueden hacer nada si no tienen dónde aplicarse.

Ahora bien, ¿cuál sería la salida? El helado resultante de aplicar el último paso es la salida. Sin una salida, los algoritmos no servirían para nada, ¿para qué vamos a hacer una secuencia de pasos que no da un resultado? Siempre se crea un resultado después de ejecutar un algoritmo.

En el segundo ejemplo, el algoritmo de suma y multiplicación, tenemos como entrada dos números (el número utilizado para multiplicar no es una entrada, porque se especifica en los pasos que debe ser igual

a dos). Una vez que tenemos las dos entradas, realizamos los pasos y obtenemos el resultado de la operación. Ese resultado es la salida.

Figura 2: Partes de un algoritmo



Fuente: elaboración propia.

Ejemplo

Identificar correctamente las partes de un algoritmo es importante, ya que nos permite tener más claro su funcionamiento, los insumos que requiere y lo que podemos esperar de él. De los siguientes algoritmos, identifiquemos cuáles son entradas, procesos y salidas.

Tabla 1. Ejemplo

Identificar las partes de un algoritmo que calcule los m ² de una pared	
Pasos	Clasificación
1. Ingresar alto de la habitación.	Entrada
2. Ingresar ancho de la habitación.	Entrada

3. Multiplicar alto por ancho.	Proceso
4. Mostrar resultado.	Salida

Fuente: elaboración propia.

Tabla 2. Ejemplo

Identificar las partes de un algoritmo que calcule el promedio de 3 notas.	
Pasos	Clasificación
1. Ingresar nota 1 de la asignatura.	Entrada
2. Ingresar nota 2 de la asignatura.	Entrada
3. Ingresar nota 3 de la asignatura.	Entrada
4. Sumar nota 1, nota 2 y nota 3.	Proceso
5. Dividir la sumatoria en 3.	Proceso
6. Mostrar el resultado.	Salida

Fuente: elaboración propia.

Tabla 3. Ejemplo

Identificar las partes de un algoritmo que realice la revisión técnica de un vehículo	
Pasos	Clasificación
1. Ingresar el auto a la revisión.	Entrada
2. Revisar alineación.	Proceso
3. Revisar suspensión.	Proceso
4. Revisar frenos.	Proceso
5. Revisar luces.	Proceso
6. Revisar emisiones.	Proceso
7. Generar resultado de la revisión.	Proceso
8. Entregar resultado de la revisión.	Salida
9. Entregar vehículo.	Salida

Fuente: elaboración propia.

Tema 3. Tipos de algoritmo

Además de tener características y partes, los algoritmos se pueden clasificar de diferentes maneras de acuerdo con el lenguaje con el que se describen sus pasos, respecto al objetivo o función que buscan realizar, y en relación con la estrategia que utilizan para llegar al resultado.

Lenguaje

Podemos clasificar los algoritmos en virtud de si se escriben con un lenguaje natural, es decir, con palabras, o a través de cálculos numéricos, esto es, con números y operadores como la suma y la resta.

Formalmente, se los clasifica de la siguiente forma:

Cualitativos —

Se utilizan palabras para definir los pasos.

Por ejemplo, un algoritmo cualitativo sería el conjunto de pasos para limpiar correctamente un piso:

1. mojar el trapo en agua limpia.
2. Ubicar el trapo en el palo de piso.
3. Pasar el trapo sobre el piso.
4. Si el piso no está completamente limpio, volver al paso 1.
5. Si está limpio, terminar con la limpieza.

Cuantitativos —

Se utilizan números y operadores.

Para un algoritmo cuantitativo, vamos a utilizar un ejemplo en el que queremos resolver la siguiente ecuación: $(2 + 3) \times 2 - 4$.

- $(2 + 3) \times 2 - 4$.
- $5 \times 2 - 4$.
- $10 - 4$.
- 6.

Obtenemos que el resultado es igual a 6.

Figura 3: Tipos de algoritmo de acuerdo con el lenguaje



Fuente: elaboración propia.

Función

Podemos tener algoritmos que cumplan diferentes funciones que, en general, se pueden clasificar en las siguientes:

- **Búsqueda:** estos algoritmos se encargan de encontrar un determinado ítem dentro de un conjunto. Por ejemplo, buscar un número dentro de una lista o una palabra en un libro. Los más conocidos son los algoritmos que utilizan los buscadores de internet, como, por ejemplo, Google. Si quieres conocer más, puedes ir a la página del sitio para ver los detalles: Google, (2018). Cómo funciona la Búsqueda de Google | Algoritmos de búsqueda. Recuperado de https://www.google.com/intl/es_es/search/howsearchworks/algorithms/
- **Ordenamiento:** su objetivo es ordenar un conjunto de valores de acuerdo con un criterio específico. El criterio puede ser ordenar alfabéticamente, en orden numérico, entre otros. Por ejemplo, dada una lista de palabras, ordenarlas de manera alfabética. Algunos ejemplos de esta clase de algoritmos en nuestra vida cotidiana son las recomendaciones que te hacen aplicaciones como Netflix, YouTube o Spotify, que ordenan su contenido de acuerdo con tus preferencias.
- **Encaminamiento:** algoritmos que buscan cómo debe ser la trayectoria de un objeto desde un lugar a otro. Por ejemplo, si tenemos que viajar a tres ciudades diferentes, este tipo de algoritmo nos ayudará a decidir cuál es la forma más corta de hacer el viaje. Google Maps utiliza este tipo de algoritmo, el que nos permite obtener rutas con menor cantidad de tiempo en el trayecto o menos cantidad de kilómetros recorridos. Si quieres conocer más, puedes ir a la página del sitio para ver más detalles: **Google Developers**, (2022). Encoded Polyline Algorithm Format | Google Maps Platform | Google Developers. Recuperado de <https://developers.google.com/maps/documentation/utilities/polylinealgorithm>

Figura 4: Tipos de algoritmo de acuerdo con la función



Fuente: elaboración propia.

Estrategia

Por último, los algoritmos se pueden clasificar en relación con la estrategia que utilizan para llegar al resultado.

DETERMINISTA

PROBABILÍSTICO

VORAZ

Cada paso dentro del algoritmo es lineal. Esto significa que cada paso (a excepción del primero) tiene un solo predecesor y un solo sucesor (a excepción del último). No existen caminos alternativos de acuerdo con sus entradas. Dentro de estos algoritmos, encontramos las funciones matemáticas.

DETERMINISTA

PROBABILÍSTICO

VORAZ

Este tipo de algoritmo utiliza valores pseudoaleatorios como entrada para poder llegar a una solución. Estos algoritmos se ejecutan la cantidad suficiente como para determinar un nivel de confianza en su resultado. ¿Has jugado alguna vez a la batalla naval? ¿Dónde debes hundir los barcos de tu enemigo? La selección de la casilla es aleatoria, el algoritmo es siempre el mismo y, en algunas ocasiones, el resultado es correcto, es decir, que seleccionas una casilla en la que se encuentra un barco.

DETERMINISTA

PROBABILÍSTICO

VORAZ

Los algoritmos voraces evalúan cada una de las posibilidades, y eligen la mejor de ellas. Esto se puede utilizar al buscar la ruta óptima para llegar de un lugar a otro, o en la red, la ruta que debe seguir un paquete para tomar el menor tiempo.

Si quieres conocer más al respecto, puedes mirar el siguiente libro:

[Fuente: Tirado Domínguez, G. \(2010\). El problema del viajante con múltiples pilas. Universidad Complutense de Madrid. Recuperado de https://elibro.net/es/ereader/ipp/89117?page=67.](https://elibro.net/es/ereader/ipp/89117?page=67)

Figura 5: Tipos de algoritmo de acuerdo con la estrategia



Tema 4. Pasos de solución de problemas

Para poder encontrar la solución de un problema, existe un conjunto de pasos que podemos seguir, los cuales nos ayudarán a estructurar nuestro trabajo.

Interpretación del problema. Análisis

El análisis del problema es el proceso en el que se interpretan el problema y las posibles soluciones.

Por ejemplo, supongamos que debemos viajar de una ciudad a otra; seguramente, tenemos varios caminos, cada uno con sus ventajas y desventajas (uno es más corto, otro es pavimentado, etcétera). Existen varios métodos, también llamados paradigmas, que nos permiten analizar un problema, es decir, encontrar posibles pasos, resultados y soluciones a un problema.

Cómo indica Trejos Buriticá, en su libro *Lógica de programación*:

¿Cuál es el primer paso que debemos dar cuando nos enfrentamos a un problema? Lo primero que debemos tener muy claro es ¿cuál es el problema a resolver? Es evidente que no podemos avanzar hacia la casa de un amigo si no sabemos en dónde vive, porque las posibilidades de que lleguemos son casi nulas. De manera que lo primero a conocer muy bien es el problema cuya solución la vamos a denotar con el nombre **objetivo**. (2017, p. 17).

Algo que nos puede ayudar cuando tenemos problemas complejos es separar el problema en problemas más pequeños que podamos manipular. De esta forma, el análisis de problemas complejos se hace más simple.

Abstracción

Abstraerse de un problema es concentrarse en lo que es importante para solucionarlo.

Siguiendo con el ejemplo anterior, no es importante quién es el gobernador de la ciudad, no nos aporta información útil para lo que queremos solucionar. Lo que nos importa es usar el mejor camino; de lo demás, debemos abstraernos y, por lo tanto, no tomarlo en cuenta.

Formular la estrategia de resolución

Una vez que hemos analizado el problema y nos hemos abstraído de lo que no es importante, se necesita elaborar un plan: el conjunto de pasos que se deben seguir para llegar al resultado deseado.

Volviendo al ejemplo del viaje, el plan podría representarse como las rutas que se van a tomar, cuándo se necesita doblar, por cuáles pueblos pasar, etcétera.

Nuevamente, citaremos a Trejos Buriticá en su libro *Lógica de programación*:

Tener claro el objetivo nos permite algo adicional. Aquí voy a utilizar una frase que, aunque un poco romántica, nos va a ilustrar claramente: el **objetivo** es el faro que, solo cuando está bien claro, nos ilumina el camino para lograrlo. Cuando el objetivo está suficientemente claro, podemos vislumbrar un camino lógico para llegar hasta él. Ese camino lógico va a tener un nombre, dada la orientación de este libro, y ese nombre es **algoritmo**. (2017, p. 18).

Aquí, estableceremos la estrategia que sea más efectiva para cumplir con el objetivo, que está directamente relacionado con el problema a resolver. Con un detalle suficiente como para que no quede duda de lo que se debe hacer.

Verificar la estrategia de resolución

Es muy importante verificar que los pasos descritos en el paso anterior funcionen de la manera deseada. Para ello, debemos probar el plan, ejecutar esos pasos y validar que el resultado sea el deseado.

En nuestro ejemplo, tendríamos que hacer un viaje para probar que llegamos en el tiempo planificado. Durante el viaje, podríamos encontrar cortes de ruta o algún otro impedimento que no tuvimos en cuenta que hagan demorar el viaje.

De esta prueba, podríamos concluir que el algoritmo cumple con el objetivo planteado o que se requieren hacer cambios para poder ajustar el camino de nuestra estrategia.

Ajustar la estrategia

En el caso de que la prueba no haya dado los resultados deseados, debemos cambiar el plan, teniendo en cuenta la nueva información.

Si en el camino encontramos muchos cortes en una ruta particular, podemos tomar otra ruta que sea más rápida en el próximo viaje.

Es importante realizar pruebas y ajustar la estrategia hasta que tengamos un resultado deseado.

Figura 6: Pasos para la resolución de problemas

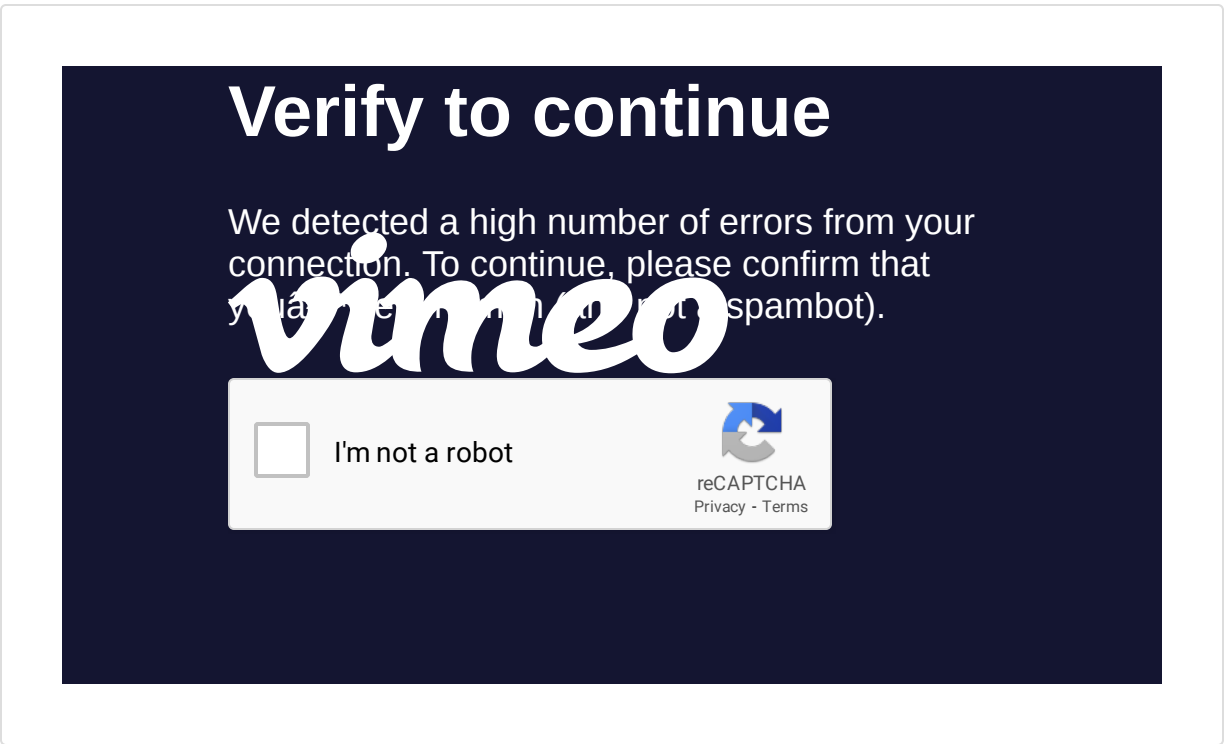




Fuente: elaboración propia.

CONTINUAR

Video de habilidades



¿Qué realizan los operadores lógicos?

- Comparación entre valores.
- Operaciones aritméticas.

- Cálculos.
- Ejecución de funciones.
- Definición de datos.

SUBMIT

¿Cuándo es verdadero el operador AND?

- Cuando un elemento es verdadero y el otro falso.
- Cuando los dos son falsos.
- Nunca, siempre es falso.
- Sólo si los dos elementos son verdaderos.

SUBMIT

¿Cuándo el operador OR es verdadero?

- Cuando los dos son falsos.
- Cuando cualquiera de los elementos es verdadero.
- Siempre es verdadero.
- Siempre es falso.

SUBMIT

¿Qué sucede cuando aplicamos un operador NOT?

- El resultado siempre es falso.
- El resultado siempre es verdadero.

- Cambia el valor de falso a verdadero y viceversa.
- No existe el operador.

SUBMIT

Los operadores lógicos AND y OR nunca pueden producir un mismo resultado ya que son operaciones totalmente opuestas.

- Verdadero.
- Falso.

SUBMIT

CONTINUAR

Cierre

Algoritmos —

Es un conjunto definido de instrucciones, con un orden y una cantidad determinados (no existen algoritmos con infinitos pasos). Los algoritmos permiten llevar a cabo una actividad mediante pasos sucesivos que no generen dudas a quien deba realizar dicha actividad.

Características de los algoritmos —

Estos deben ser de la siguiente manera:

- ordenados.
- Finitos.
- Definidos.

Partes de un algoritmo —

Para cada algoritmo, se puede definir que tendrá:

- entrada.
- Proceso.
- Salida.

Lógica —

La lógica puede definirse como la disciplina que formaliza el estudio de los métodos de razonamiento.

Lógica proposicional —

Una proposición lógica es toda oración declarativa que pueda decirse si es verdadera o falsa.

Álgebra de Boole —

Se puede definir como un sistema matemático basado en los valores 1 (verdadero) y 0 (falso), en conjunto con las operaciones AND, OR y NOT. Las variables en este sistema no pueden tomar valores diferentes a 1 y 0, y se denominan variables booleanas.

Funciones lógicas —

Una función lógica es la expresión que define las operaciones para realizar sobre un conjunto de variables, para obtener un resultado.

CONTINUAR

Referencias

Google (2018) *Cómo funciona la Búsqueda de Google | Algoritmos de búsqueda*. Recuperado de https://www.google.com/intl/es_es/search/howsearchworks/algorithms/

Google Developers (2022). *Encoded Polyline Algorithm Format | Google Maps Platform | Google Developers*. Recuperado de <https://developers.google.com/maps/documentation/utilities/polylinealgorithm>

Tirado Domínguez, G. (2010). *El problema del viajante con múltiples pilas*. Universidad Complutense de Madrid.

Trejos Buriticá, O. I. (2017). *Lógica de programación*. Bogotá, Colombia: Ediciones de la U.

CONTINUAR