

Operaciones lógicas

$\neg p \rightarrow$ Negación ("No p ")

$p \wedge q \rightarrow$ conjunción (" p Y q ").

$p \vee q \rightarrow$ Disyunción (" p O q ").

$p \rightarrow q \rightarrow$ implicación ("Si p , Entonces q ").

$p \leftrightarrow q \rightarrow$ Bicondicional (" p Si Y Solo si q ")

Tipos

Según tabla de la verdad

Tautología \rightarrow Siempre da V (verdadero)

Contradicción \rightarrow Siempre da F (falso)

Contingencia \rightarrow A veces V a veces F.

$(p \vee \neg p) \rightarrow$ Siempre verdadero \rightarrow tautología

$(p \wedge \neg p) \rightarrow$ Siempre falso \rightarrow contradicción

Tablas de la verdad

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

conclusión $p \wedge q$ solo es verdadero cuando las p y q son verdaderas

Operaciones de la Verdad

$\neg p \rightarrow$ Negación (Complemento de valor)

$p \wedge q \rightarrow$ Verdadero (Si los 2 son verdaderos)

$p \vee q \rightarrow$ Falso (Si los 2 son falsos)

$p \rightarrow q \rightarrow$ Falso (Solo si p es verdadero y q es falso)

$p \leftrightarrow q \rightarrow$ Verdadero (Si p y q son iguales)

Qué es un algoritmo

Un algoritmo es un conjunto de pasos para resolver un problema
se puede escribir:

Pseudocódigo (es un lenguaje intermedio)

Diagrama de flujo (dígitos con flechas y figuras)

Código real (Python, Java, C++, PHP)

Algoritmo lógico

Número Primo

Un número primo si:

es mayor o igual que 2

solo se puede dividir entre 1 y el mismo

Entrada = número n

Si $n < 2 \rightarrow$ no es primo

Sí: NO

Probar dividir entre todos los números desde 2 hasta la raíz de n

Si algún número divide exacto \rightarrow no es primo

Si ninguno divide \rightarrow es primo

ARRAYS

Primitivo	→ estructura para guardar varios valores
Unidimensional	→ lista o vector
Bidimensional	→ tabla o matriz
Multidimensional	→ varios dimensiones (cubo)
Asociativo	→ usa claves en lugar de índices

Ejemplo de ARRAYS ; Asociativo y unidimensional → JSON

CICLOS

for (para) → cuando se cuentas veces repetir
while (mientras) → mientras se cumpla una condición
Do while / Repeat until (Repetir Hasta)

Condicionales

if → Ejecuta un bloque de código solo si la condición es verdadera
if - else → Si la condición ejecuta un bloque, de lo contrario ejecuta otro
if - else if - else → Permite evaluar varias condiciones
switch → Se usa cuando una variable puede tomar varios valores posibles

Funciones

Son bloques de código que permiten reutilizar instrucciones en un programa, es a la vez de repetir código (característica)

- tiene la condice (para if's etc)
- pueden recibir参參etros (entradas)
- pueden retornar valores (salidas)
- hace el código con mas ordenado, reutilizable y facil de mantener

include → reutiliza funciones → si o si hace el uso de un
exced → incorpora funciones → ofciates o varaciones

Onboarding ⇒ es un proceso integral de adaptación e integración de un nuevo empleado

fullfilmente → proceso lógico que ocurre cuando el usuario realiza una compra hasta que recibe el producto

Checkout → verificación

funciones de diagrama)

Casos de uso → Que hace el usuario con el sistema

Clases → Cómo se organiza el sistema internamente

Actividades → Cómo fluyen los procesos en el sistema

Flujo → Lógica paso a paso

Poo

Clase → Molde o Plantilla

Objeto → Instancia

Encapsulamiento → • Oculta detalles internos y expone solo lo necesario

→ • Se logra con modificadores de acceso (public, private, protected)

Herencia → • Una clase puede heredar atributos y métodos de otra
• Se puede reutilizar código

Polimorfismo → • Un mismo método puede tener diferentes formas según el contexto

• Sobre carga → (overloading) mismo nombre
• Sobrescritura → (overriding) redirigir

Absctraccion

- Crea clases abstractas o interfaces para definir que hace un objeto sin implementar como lo hace (clase abstracta \Rightarrow una clase que tiene que ser implementada pero no modificar)

abierto y cerrado \rightarrow abierto para implementar cerrado para modificar