Recibe una cálida:

# Bienvenida!

Te estábamos esperando 😁







# Algoritmos

Plan formativo: Desarrollo de Aplicaciones Full Stack Java Trainee V2.0





## HOJA DE RUTA

¿Cuáles skill conforman el programa?









### REPASO CLASE ANTERIOR



En la clase anterior trabajamos 📚:

- Expresiones aritméticas
- Operadores aritméticos
- Operadores relacionales







## LEARNING PATHWAY



Fundamentos de la programación en Java

Teoría de conjuntos y lógica proposicional. Expresiones lógicas. Tablas de la verdad. Lógica Proposicional: Tablas de la verdad Creando una Tabla de la Verdad





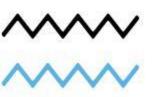


### OBJETIVOS DE APRENDIZAJE

¿Qué aprenderemos?



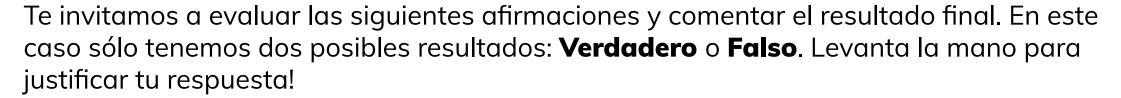
- Comprender la Teoría de Conjuntos y la Lógica Proposicional
- Reconocer las expresiones lógicas y su implementación
- Interpretar y aplicar las Tablas de la Verdad







#### ¿Verdadero o Falso?: 🙌



#### Afirmaciones: <u></u>

- Todos los mamíferos respiran aire Las ballenas son mamíferos
   Conclusión: Las ballenas respiran aire.
- Los planetas giran alrededor de una estrella La Tierra es un planeta Sirio es una estrella
   Conclusión: La Tierra gira alrededor de Sirio.

#### Afirmaciones: <u></u>

- Todos los pájaros vuelan
   Los pingüinos son un tipo de pájaro
   Conclusión: Los pingüinos vuelan.
- Los reptiles son animales de sangre fría Las serpientes son reptiles
   Conclusión: Las serpientes son animales de sangre fría.







Teoría de conjuntos y lógica proposicional





# Teoría de Conjuntos

### ¿Qué es y para qué sirve?

La teoría de conjuntos es una rama de las matemáticas que estudia las propiedades de los conjuntos y sus operaciones. En programación, la teoría de conjuntos se utiliza para:

- **Modelar datos**: Los conjuntos permiten representar colecciones de valores o datos de manera abstracta. Por ejemplo, el conjunto de todos los usuarios registrados en un sistema.
- Trabajar con relaciones entre datos: sirven para relacionar datos de forma lógica y extraer información útil.
- Resolver problemas algorítmicos: Muchos problemas computacionales se modelan utilizando conjuntos y sus operaciones, como por ejemplo algoritmos de búsqueda y ordenamiento.





---

# Lógica Proposicional

### ¿Qué es y para qué sirve?

La lógica proposicional se refiere al estudio de las proposiciones y cómo se combinan para formar declaraciones más complejas.

---

En programación, utilizamos la lógica proposicional para **tomar decisiones basadas en condiciones**. Podemos combinar proposiciones utilizando **operadores lógicos** como **AND, OR y NOT**, para evaluar si una afirmación es **verdadera** o **falsa**, y así controlar el flujo de nuestro programa.







# Expresiones lógicas

**Son declaraciones que evalúan si una afirmación es verdadera o falsa.** Utilizamos operadores lógicos como AND, OR y NOT para combinar y comparar valores.

- Conjunción: Si queremos verificar si dos condiciones se cumplen simultáneamente, usamos el operador AND. El simbolismo de referencia es &&.
- -¦-
- **Disyunción**: Si queremos verificar si al menos una de las condiciones es verdadera, usamos el operador **OR**. El simbolismo de referencia es **||.**
- Negación: El operador NOT invierte el valor de una expresión. El simbolismo de referencia es !.







#### ¿Qué son y para qué se utilizan?:

Consisten en una tabla que relaciona todas las combinaciones posibles de valores de verdad de las variables o proposiciones de una expresión, con el resultado de evaluar la expresión completa. En programación se utilizan para diseñar algoritmos y probar que ciertas condiciones se cumplen según lo esperado.

Las tablas de verdad sirven para:

- Comprobar si un argumento es válido.
- Demostrar equivalencias entre expresiones.
- Verificar circuitos lógicos y digitales.
- Simplificar expresiones booleanas.
- Modelar problemas de lógica.





+

#### Conjunción &&, AND:

La conjunción de dos proposiciones es **verdadera únicamente** en el caso en que **ambas proposiciones lo sean.** 

Columnas representadas:

• **p** : Premisa 1

• **q**: Premisa 2

• **p && q:** Conclusión o resultado



р	q	p && c
V	٧	V
V	F	F
F	٧	F
F	F	F





### Disyunción ||, OR:

La disyunción entre dos proposiciones es **verdadera** cuando **al menos una de dichas proposiciones lo es**.

### Columnas representadas:

• **p** : Premisa 1

• **q**: Premisa 2

• p | q: Conclusión o resultado



p	q	p    q
V	V	v
V	F	V
F	V	V
F	F	F





#### Negación!, NOT:

El valor de verdad de la **negación** de una proposición **verdadera es falso** y el valor de verdad de la negación de una proposición **falsa es verdadero**.

Columnas representadas:

• **p** : Premisa 1

• !p : Negación premisa 1









### LIVE CODING

Ejemplo en vivo

### Implementando Tablas de la Verdad:

Supongamos que estamos desarrollando un sistema que recomienda películas según las preferencias del usuario.

### Las reglas son:

- Si al usuario le gustan las películas de acción **O** las películas de ciencia ficción, recomendar Alien.
- Si al usuario le gustan las películas románticas Y las películas de drama, recomendar Titanic.

Tiempo: 25 minutos





### LIVE CODING

Ejemplo en vivo

# Construimos tablas de verdad para evaluar en qué casos se cumplen R1 y R2:

Podemos representar estas reglas como:

- A: Al usuario le gustan películas de acción
- B: Al usuario le gustan películas de ciencia ficción
- C: Al usuario le gustan películas románticas
- D: Al usuario le gustan películas de drama
- R1: Recomendar Alien = A OR B
- R2: Recomendar Titanic = C AND D





### LIVE CODING

Ejemplo en vivo

### Aquí podemos ver los posibles valores para R1 y R2:







# Ejercicio N° 1 Creando una Tabla de la Verdad





### Creando una Tabla de la Verdad

### Contexto: 🙌

Las Tablas de la Verdad se utilizan para evaluar diferentes tipos de condiciones, y son aplicables a todos los ámbitos.

En las breakout rooms, deberán completar una tabla de la verdad en equipos según la referencia.

### Consigna: 🚣

Crear una tabla de la verdad para cada expresión:

- Expresión: A **AND** (B **OR** C)

Expresión: NOT(A OR B) AND C

Tiempo : 20 minutos

### Paso a paso: 🔅

- Primero debes crear una columnas para cada premisa (A, B, C).
- Luego debes completar, comenzando la primer fila con todas las expresiones como Verdaderas.
- ¡Recuerda las reglas de prioridad!





---

# ¿Alguna consulta?



### RESUMEN

¿Qué logramos en esta clase?



- Comprender el concepto de la Teoría de Conjuntos
- Entender y aplicar los conceptos de la lógica proposicional.
- Reconocer e implementar los elementos y operadores de las Tablas de la Verdad.







### **#WorkingTime**

Continuemos ejercitando

#### ¡Antes de cerrar la clase! Te invitamos a: 👇 👇 🔷





- Repasar nuevamente la grabación de esta clase
- Revisar el material compartido en la plataforma de Moodle (lo que se vio en clase y algún ejercicio adicional)
  - a. Lectura Módulo 4, Lección 2: páginas 11 13
- Traer al próximo encuentro, todas tus dudas y consultas para verlas antes de iniciar nuevo tema.







-1-

# Muchas Gracias!

Nos vemos en la próxima clase 🤎



*M* alkemy