

# ALGORITMO

Descripción de un proceso **paso a paso** que permite llegar a la solución de un problema

## Suma de dos números

1. Ingresar el valor de los dos números (a y b)
2. Realizar la operación:  $c = a + b$
3. Visualizar el resultado

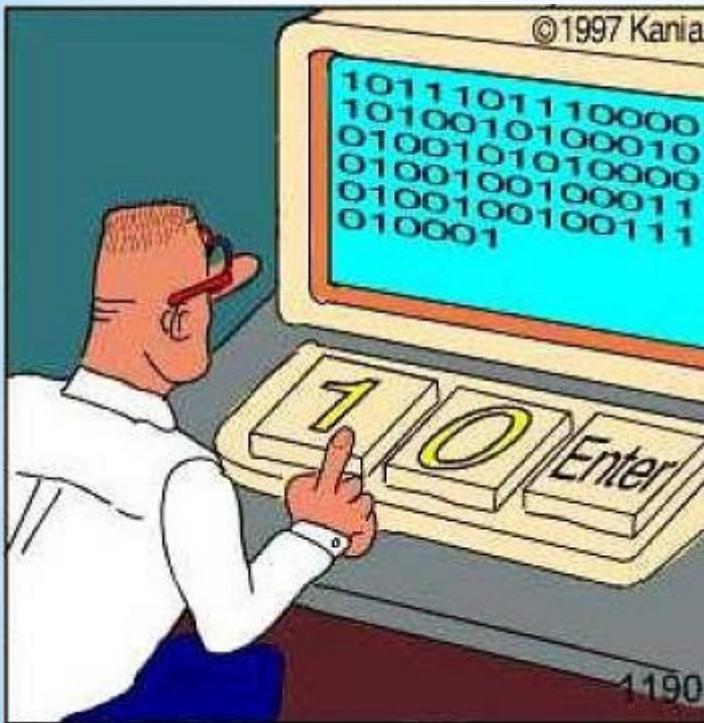


# PROGRAMA

Descripción para la computadora de un proceso que obtiene cierto resultado útil para alguien



# COMUNICACIÓN



```
PROGRAM
VAR
    TIMER1 : TON;
    TIMER2 : TON;
    TAG-1 : BOOL;
    TAG-2 : INT;
    TIN : TIME;
    TOUT : TIME;
END_VAR
-----
LD TAG-1
ST TIMER1.IN
GOTO MARK1
CAL TIMER1(
    PT:TIN1,
    ET:>TOUT1)
LD
    TIMER1.Q
ST
    TIMER2.IN
-----
MARK1:
LD TAG-2
AND 130
OR 3
```

LEARN ROBOTICS

## INSTRUCTION LIST

- ✓ Best for compact code
- ✓ Great for Time Critical code
- ✓ One Command per Line
- ✗ More difficult to resolve errors

# LENGUAJE DE PROGRAMACIÓN

Permite expresar una serie de instrucciones que podrán ser realizadas por una computadora



# ELEMENTOS DE UN LENGUAJE

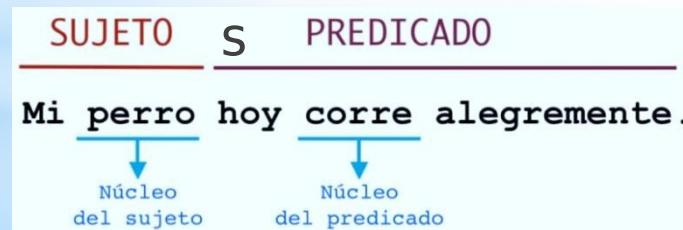
Alfabeto



Léxico



Sintaxi

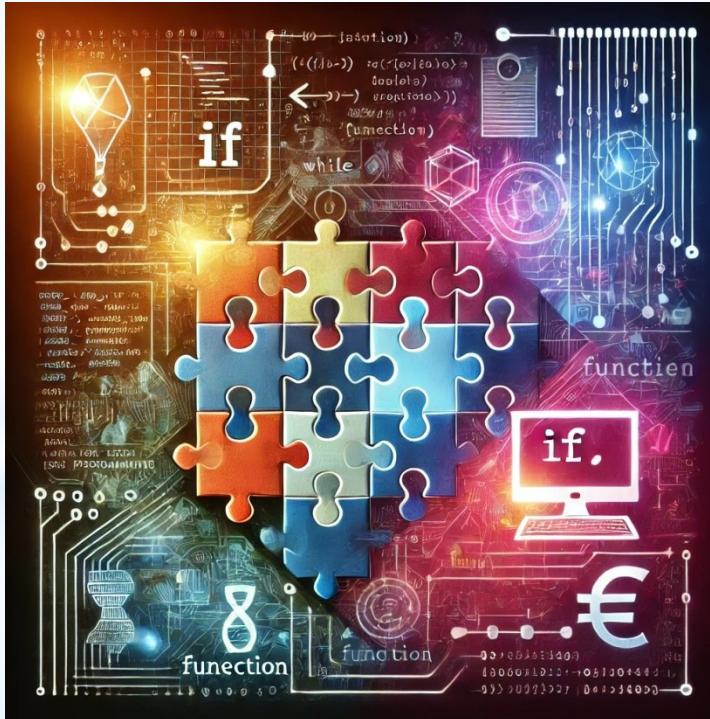


Semántica



# PROGRAMACIÓN

Componer los elementos de un lenguaje de programación en el orden que provocará un efecto deseado



# NIVELES LENGUAJES DE PROGRAMACIÓN

## Alto

```
/**  
 * Suma de dos números enteros  
 * @version 1.0 03/12/2024  
 * @author Ing. Angel Brito Segura  
 */  
fun add(a: Int, b: Int): Int {  
    return a + b  
}  
  
fun main() {  
    print("Ingresa el primer número entero: ")  
    val numero1 = readLine()?.toIntOrNull() ?: 0  
    print("Ingresa el segundo número entero: ")  
    val numero2 = readLine()?.toIntOrNull() ?: 0  
    println("La suma de los números es: ${add(numero1, numero2)}")  
}
```

## Mediano Nivel

```
#include <stdio.h>  
int add( int x, int y );  
int main()  
{  
    int x;  
    int y;  
    printf( "Escribe los dos números que se sumarán: " );  
    scanf( "%d", &x );  
    scanf( "%d", &y );  
    printf( "La suma de los números es %d\n" add( x, y ) );  
    getchar();  
}  
int add( int x, int y )  
{  
    return x + y;  
}
```

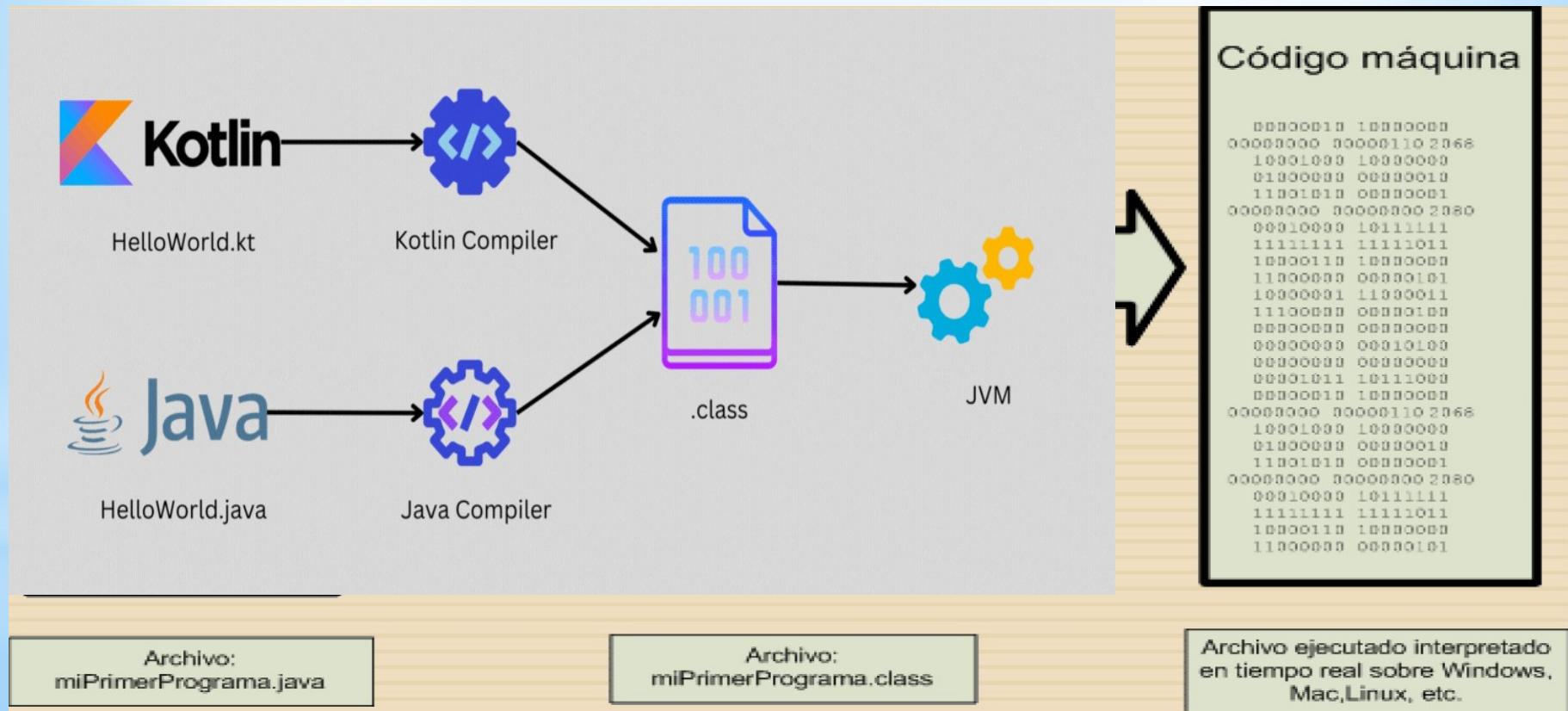
## Bajo

```
PROCESSOR 16f877 ;Indica versión del procesador (PIC16F877)  
INCLUDE <p16f877.inc> ;Librería de la versión del procesador PIC16F877  
  
K equ H'26' ;Variable K en la dirección de memoria 26 hexadecimal  
L equ H'27' ;Variable L en la dirección de memoria 27 hexadecimal  
M equ H'28' ;Variable M en la dirección de memoria 28 hexadecimal  
  
ORG 0 ;Especifica el vector de reset inicie en 0  
GOTC INICIO ;Ir a la dirección de memoria que se encuentra INICIO  
ORG 5 ;Indica origen para inicio de programa  
  
INICIO: MOVE K,W ;Mover lo que tenga K al registro W  
        ADDWF L,0 ;Sumar L con lo que tenga W  
        MOVWF M ;Mover lo que tenga W a dirección de memoria de L  
        GOTC INICIO ;Ir a la dirección de memoria que se encuentra INICIO  
END ;Termina el programa
```

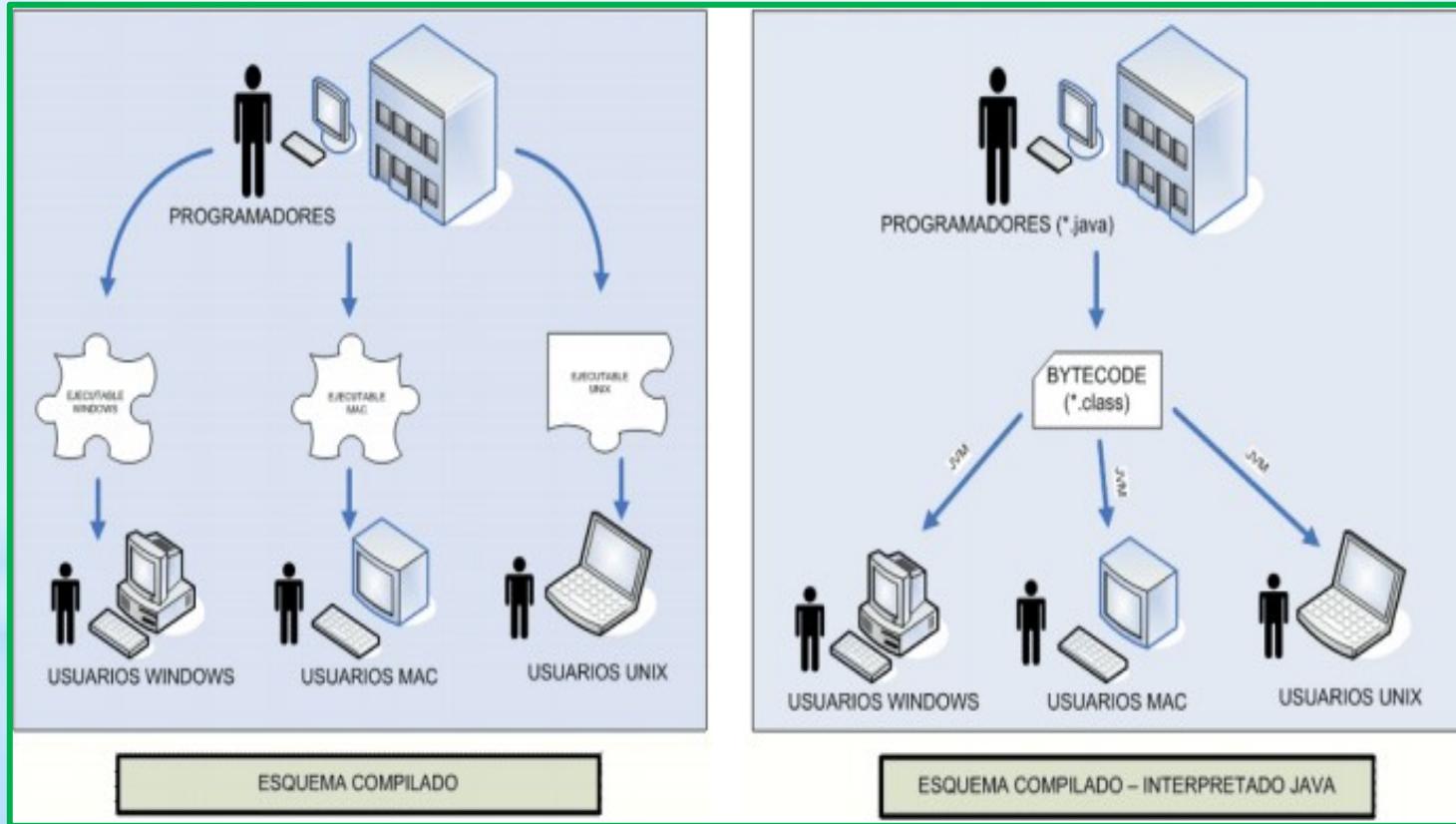
# TIPOS LENGUAJES DE PROGRAMACIÓN



# LENGUAJE INTERMEDIO



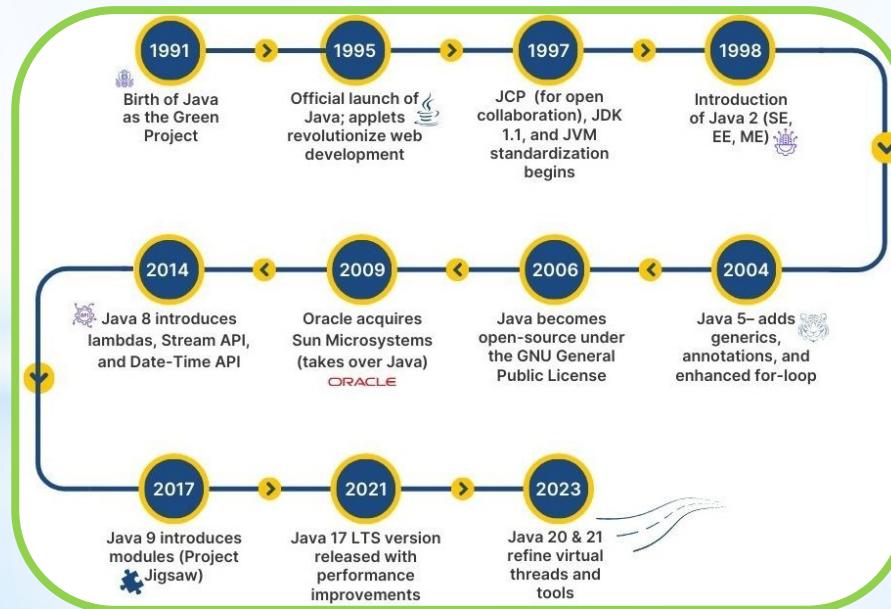
# LENGUAJE INTERMEDIO



# JAVA

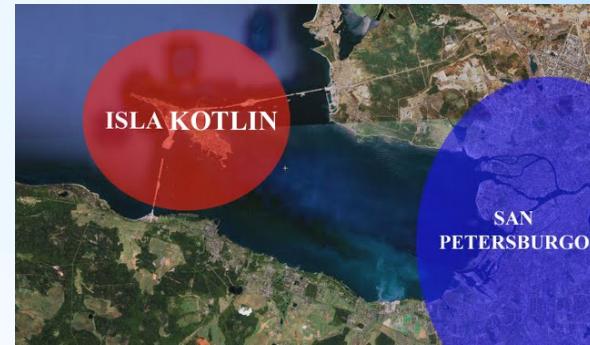


- \* Lenguaje de programación orientado a objetos
- \* Comercializado por primera vez en 1995 por Sun Microsystems



# Kotlin

- \* Lenguaje de programación de alto nivel moderno
- \* Desarrollado por JetBrains
- \* Lenguaje pragmático
- \* Multiparadigma
- \* Sintaxis muy clara y concisa, similar a la de Java
- \* Inferencia de tipos



This infographic compares Kotlin and Java across various dimensions:

- Creation:** Kotlin was created by JetBrains in 2012 and sponsored by Google since 2017. Java was created by Sun in 1995 and bought by Oracle in 2009.
- Compatibility:** Both are 100% compatible with Java and its libraries.
- OOP vs Functional:** Kotlin is not strictly object-oriented, while Java is.
- Syntax Complexity:** Kotlin's syntax is short and direct, while Java's is more complex.
- Null Safety:** Kotlin does not have null values, unlike Java.
- Performance:** Nulls can cause runtime problems in Java.
- Type Inference:** Kotlin supports type inference.
- Typing:** Kotlin is strongly typed, while Java is statically typed.

# APLICACIONES

 ALGUNAS APPS MÓVILES CREADAS CON KOTLIN  
[www.nubecollectiva.com](http://www.nubecollectiva.com)

<b>Uber</b> Uber (Android) Aplicación que permite a los usuarios pedir taxi de manera segura y confiable.	<b>NETFLIX</b> Netflix (Android) Permite a los usuarios ver películas, series y otras categorías de entretenimiento.	<b>slack</b> Slack (Android) Aplicación de mensajería para la comunicación entre miembros de un equipo de trabajo.
<b>Trello</b> Trello (Android) Aplicación que permite gestionar nuestras tareas mediante tableros organizados.	<b>duolingo</b> Duolingo (Android) Aplicación que permite aprender diferentes idiomas y obtener certificaciones.	<b>YouTube Studio</b> YouTube Studio (Android) Aplicación que permite ver sus métricas a los creadores de contenido de YouTube.
<b>RUNTASTIC</b> Adidas Runtastic (Android) Aplicación que permite calcular distancias recorridas al correr, calorías quemadas, etc.	<b>Gmail</b> Gmail (Android) Aplicación que permite enviar y recibir correos electrónicos. Cuenta con funciones avanzadas.	<b>Pinterest</b> Pinterest (Android) Red social que permite compartir imágenes y videos mediante pinches, tableros, etc.

**NOTA:** Las empresas y proyectos mencionados usan Kotlin junto con otras tecnologías. Existen otras empresas y proyectos que también usan Kotlin en sus proyectos.



# HERRAMIENTAS DE TRABAJO

\* En línea:

\* Kotlin Playground: <https://play.kotlinlang.org/>

\* JDoodle:

<https://www.jdoodle.com/compile-kotlin-online>

\* Entornos de Desarrollo (IDE)



Android Studio



IJ

I

\* Kotlin Programming Compiler (app)

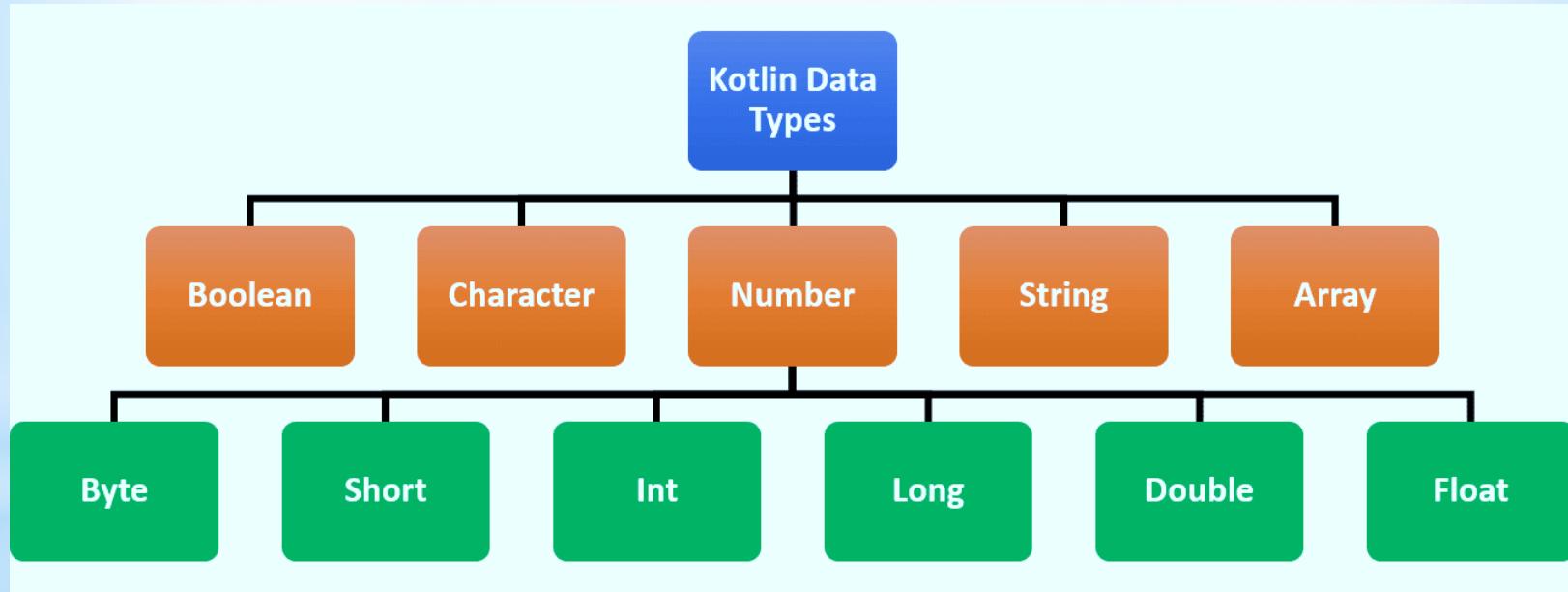


Integrado

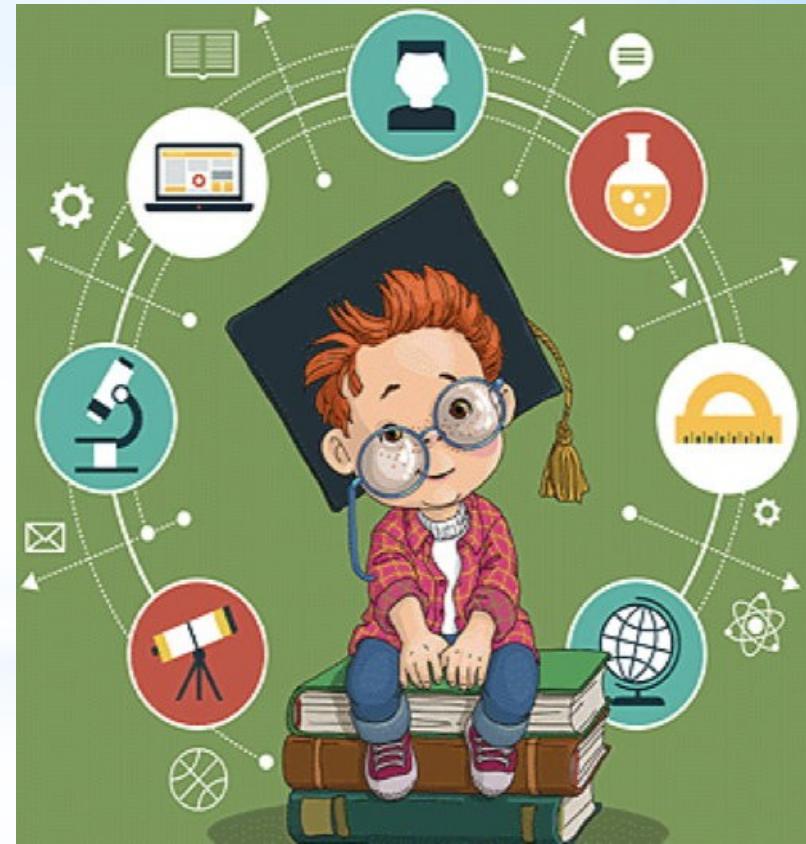


# LITERAL

Notación para representar valores fijos



# DATOS



# TIPO DE DATO - NUMÉRICO



Decimal
0
1
2
3
4
5
6
7
8
9
10
11
12
13
14
15

Binario
0000
0001
0010
0011
0100
0101
0110
0111
1000
1001
1010
1011
1100
1101
1110
1111

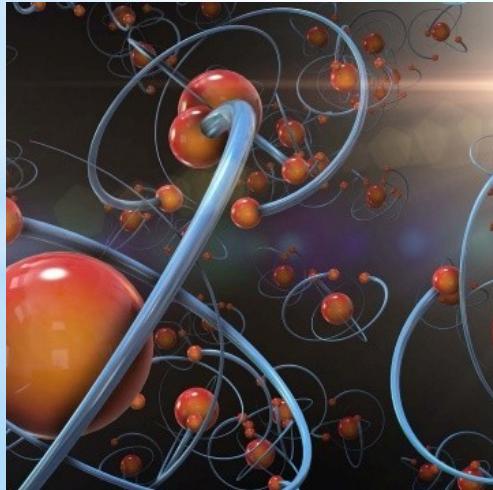
Hexadecimal
0
1
2
3
4
5
6
7
8
9
A
B
C
D
E
F

# NOTACIÓN CIENTÍFICA

Representación sencilla de números muy pequeños/grandes

$$a \times 10^n$$

Coeficiente      Base      Exponente



# OPERADOR

Símbolo capaz de realizar operaciones con valores

## expression

500

3.14

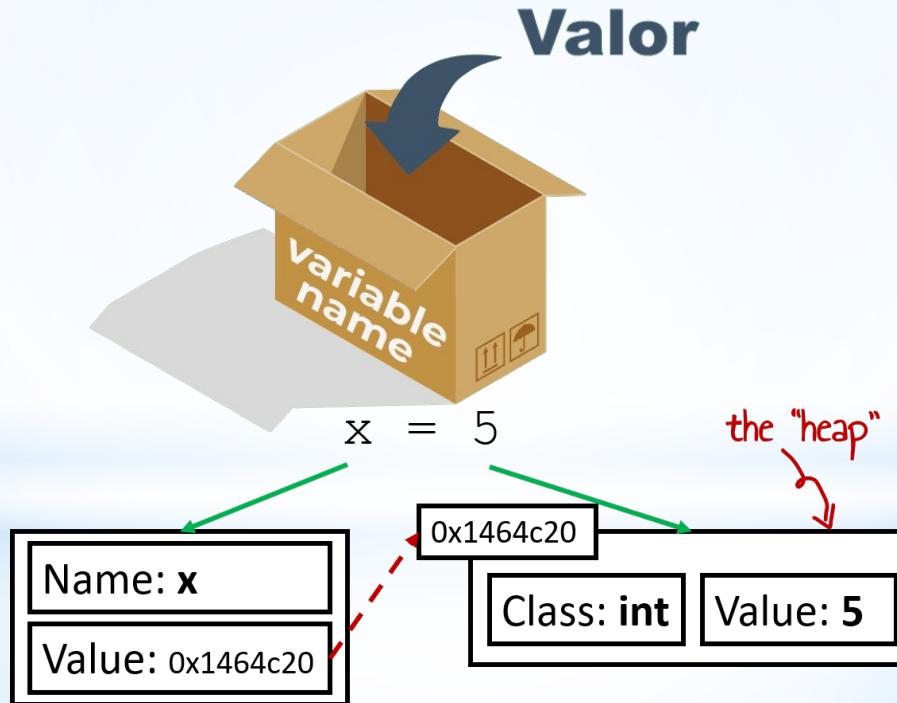
200 + 300

10.0 + 5.0

Prioridad	Operador					
1	+ (unario)			- (unario)		
2	*	/			%	
3	+ (binario)		- (binario)			
4	<	<=		>		>=
5	==				!=	
6	=	+=	-=	*=	/=	%=

# VARIABLE

Ubicación nombrada reservada para almacenar valores



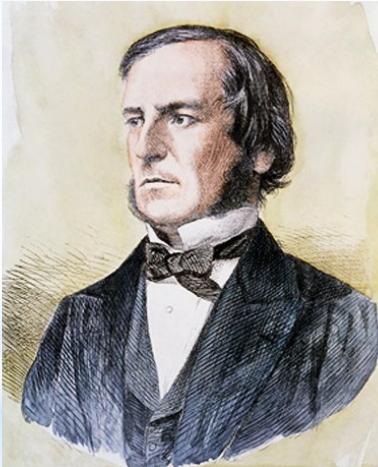
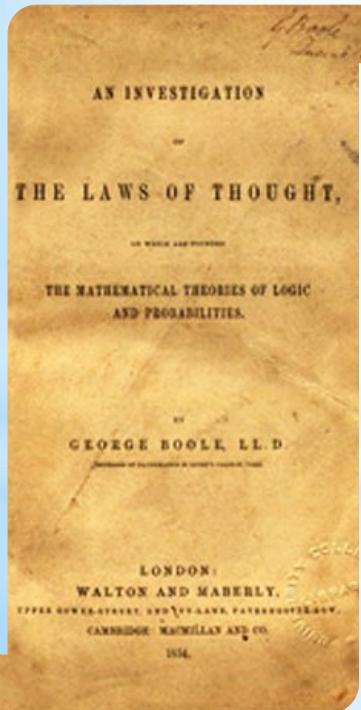
# NOMBRADO VARIABLES

- \*Sensitivo a mayúsculas y minúsculas.
- \*Puede incluir solo letras, dígitos y guiones bajos.
- \*No puede comenzar con un dígito ni keyword (palabras clave del lenguaje).
- \*Espacios solo permitidos con comillas simples invertidas: `



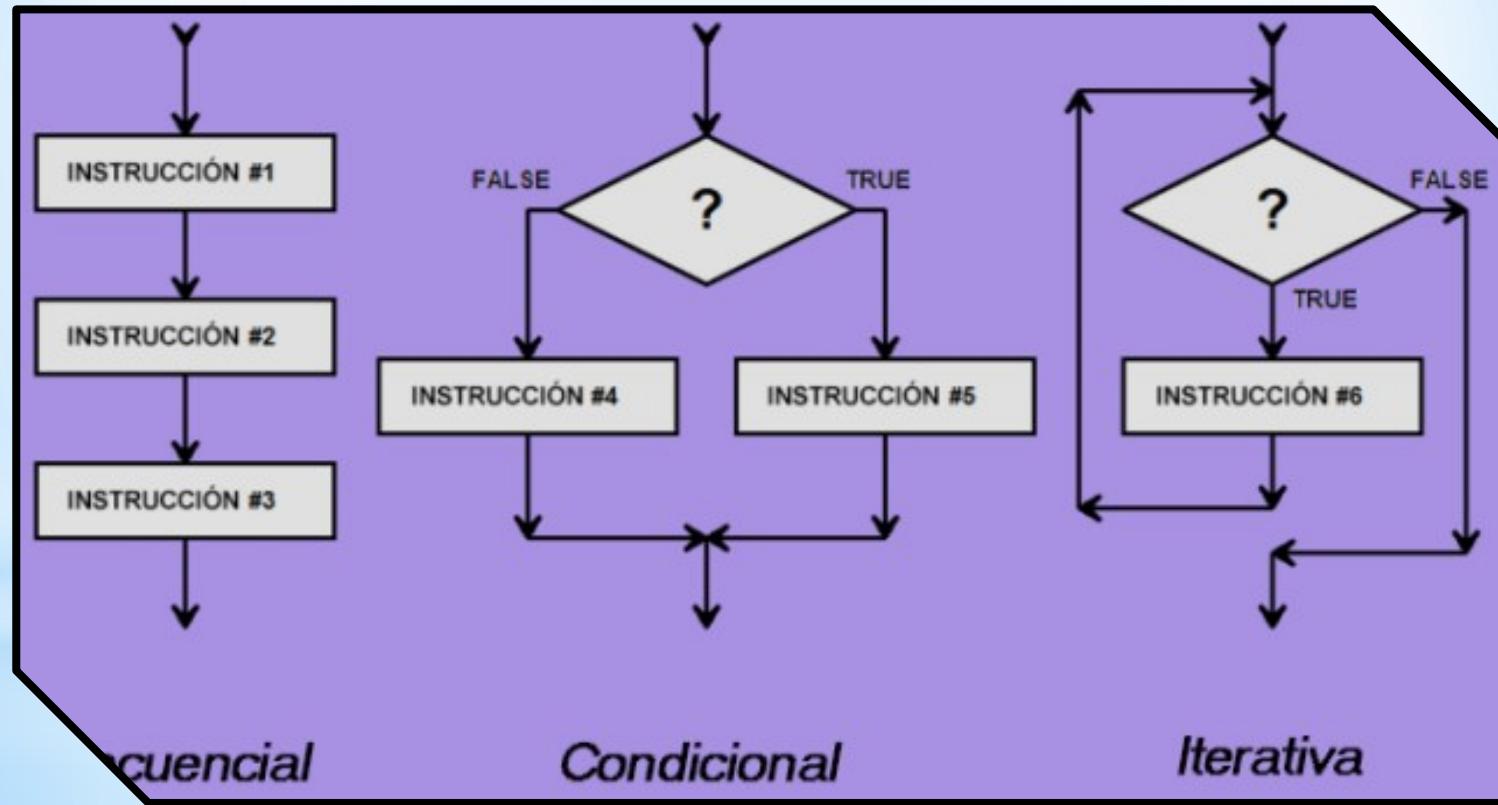
# TIPO DE DATO - BOOLEANOS

Se emplean para representar la veracidad



A	B	A&B (AND)	A  B (OR)	!A (NOT)
true	true	true	true	false
true	false	false	true	false
false	true	false	true	true
false	false	false	false	true

# ESTRUCTURAS DE CONTROL



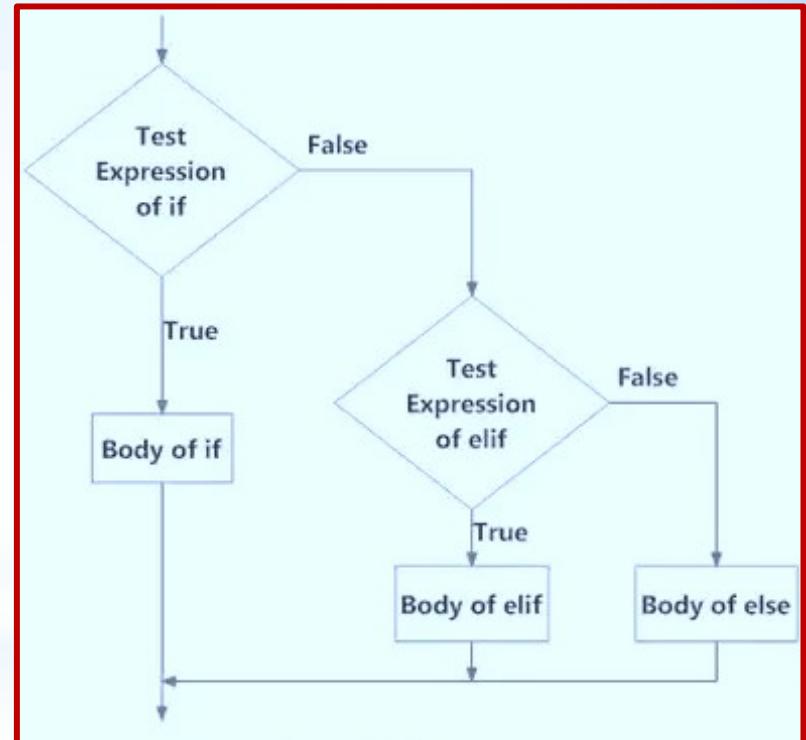
# SENTENCIA CONDICIONAL

```
if( condition 1 ) {  
    body 1  
} else if( condition 2 ) {  
    body 2  
} else {  
    body 3  
}
```

Diagram illustrating the structure of a conditional statement:

- ① "when" keyword
- ② conditions
- ③ values
- ④ "else" condition

```
val reaction = when {  
    temperature > 55 -> "It's too hot!"  
    temperature < 40 -> "It's too cold!"  
    else -> "It's just right!"  
}
```



# SENTENCIAS ITERATIVAS



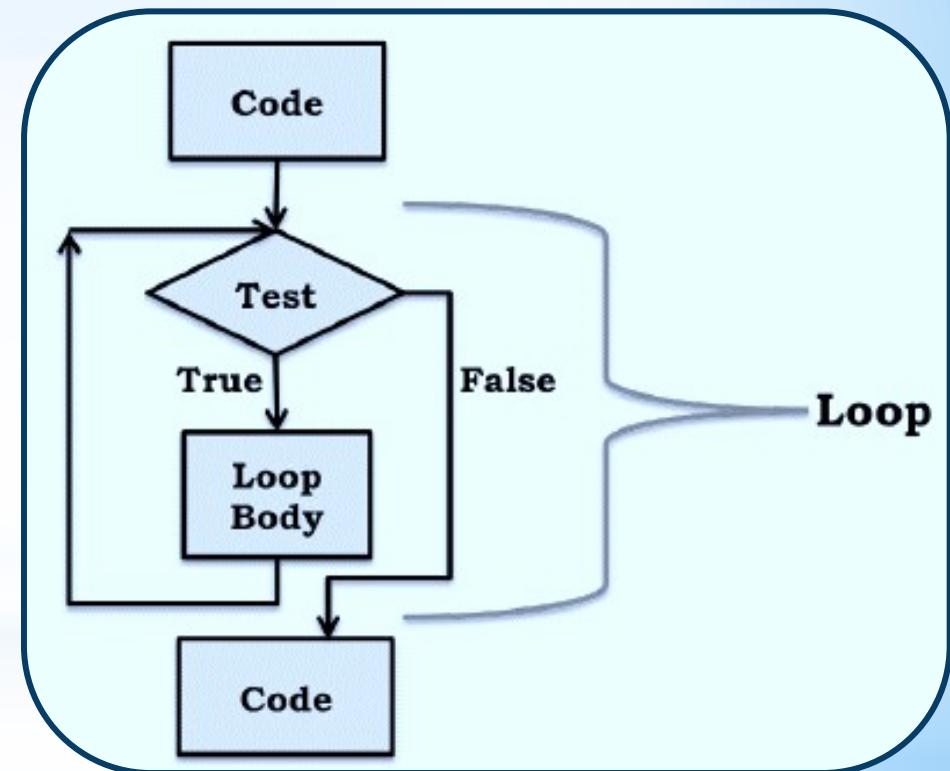
## Loops in Kotlin

```
- □ X  
while (condition) {  
}
```

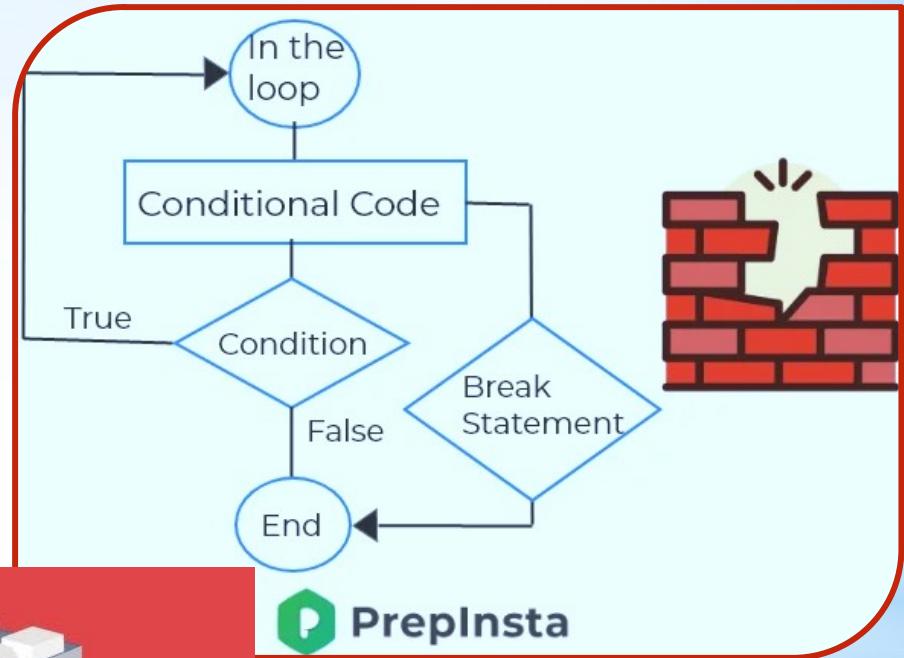
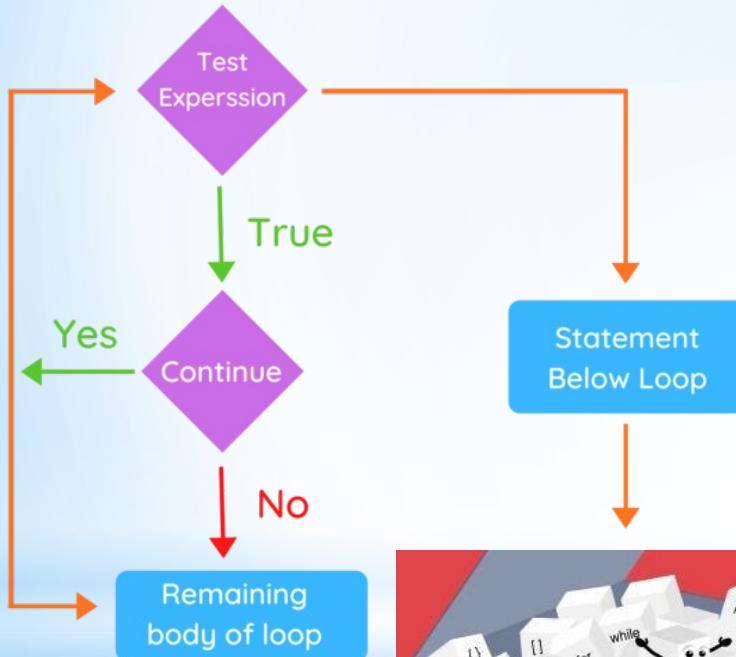
```
- □ X  
do {  
}  
} while (condition);
```

```
- □ X  
for (i in start..end) {  
}
```

```
repeat( times ) {  
    // code  
}
```



# BREAK & CONTINUE



Preplinsta

# OPERADORES LÓGICOS

**AND**

	True	False
--	------	-------

True	True	False
False	False	False

**OR**

	True	False
--	------	-------

True	True	True
False	True	False

<b>Q</b>	<b>Q'</b>
----------	-----------

0	1
---	---

1	0
---	---



$$Q = \bar{Q}$$

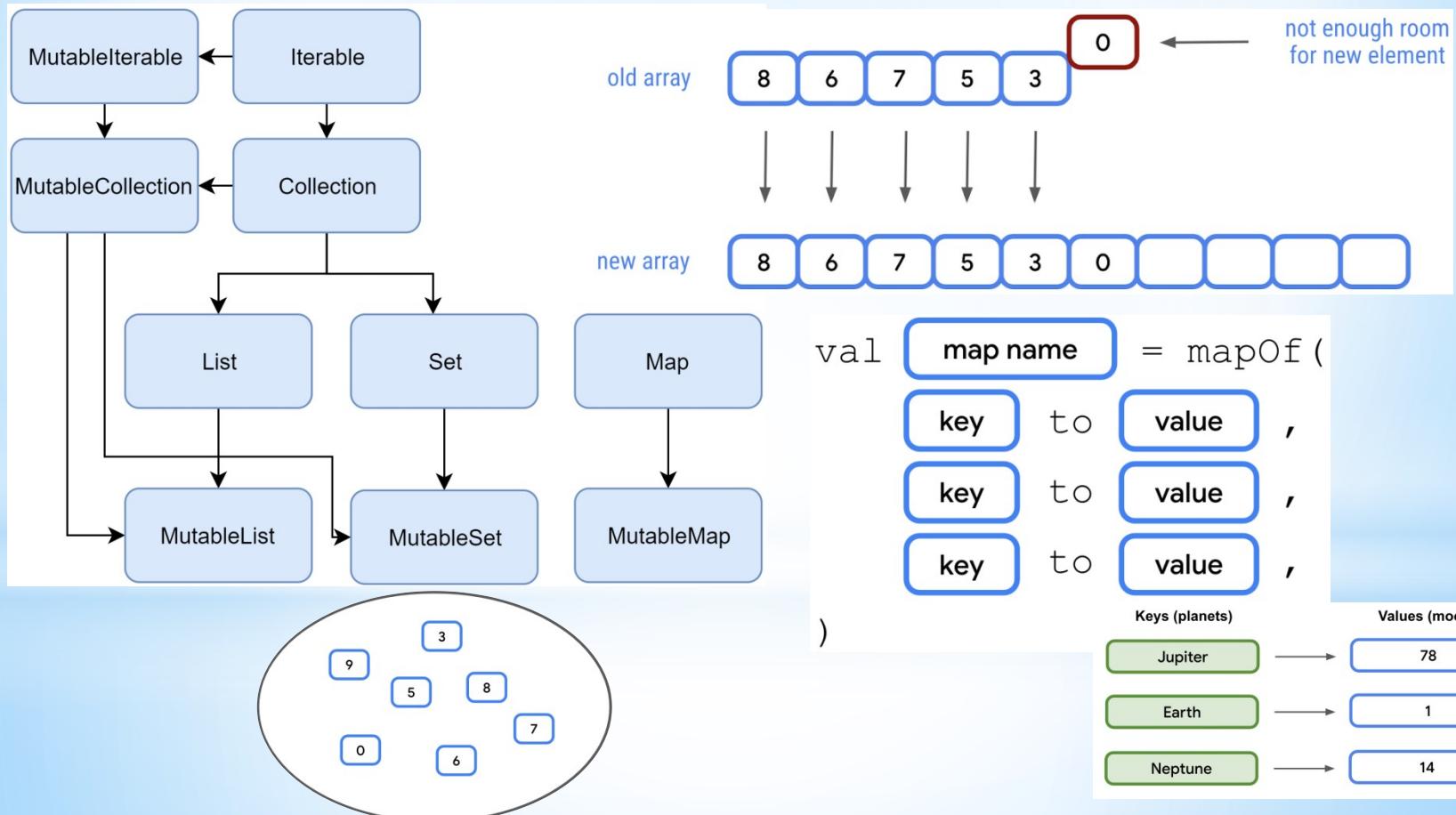
# ARREGLOS

- \* Estructura con valores de datos, que están almacenados de forma contigua en memoria
- \* Todos los elementos son referenciados por un mismo nombre y tienen el mismo tipo de dato
- \* Son mutables y de tamaño fijo

```
val variable name = arrayOf<data type>(element1, element2, ...)
```

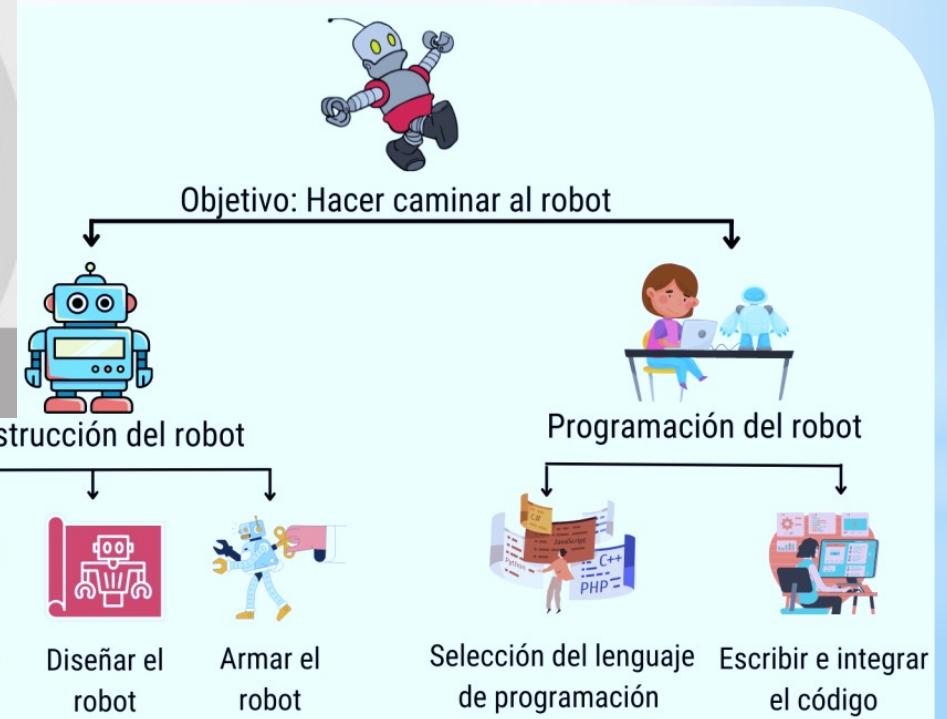


# COLECCIONES

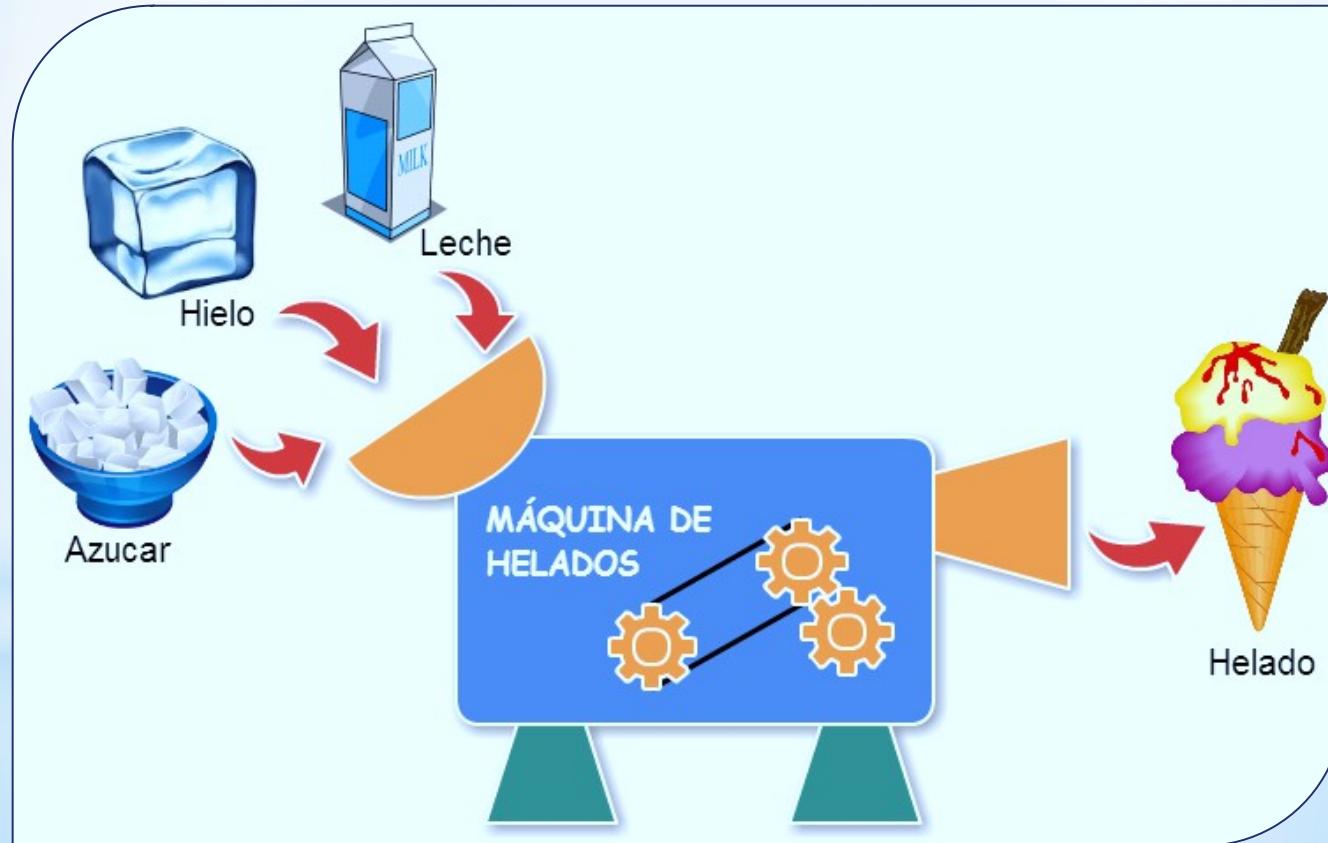


# DESCOMPOSICIÓN

Dividir un problema en piezas aisladas



# FUNCIONES Y MÉTODOS



# ERRORES

Erros

De compilación

De sintaxis

Procesos no válidos

De ejecución

Procesos no válidos

Lógicos tipo bucle infinito

Lógicos tipo resultado incorrecto

Errores gestionados



```
/media/kishan/SSD-Data-0/Kishan/Android_WorkSpace/2821/MedicalApp/app/build/generated/source/kapt/debug/com/medical/medicalapp/ui/activity/Hilt_LoginActivity.java:73: error: method getActivityFactory in class DefaultViewModelFactories cannot be applied to given types;  
    return DefaultViewModelFactories.getActivityFactory(this);  
           ^  
      required: ComponentActivity.Factory  
      found: Hilt_LoginActivity  
      reason: actual and formal argument lists differ in length  
/media/kishan/SSD-Data-0/Kishan/Android_WorkSpace/2821/MedicalApp/app/build/generated/source/kapt/debug/com/medical/medicalapp/ui/activity/Hilt_RegisterActivity.java:73: error: method getActivityFactory in class DefaultViewModelFactories cannot be applied to given types;  
    return DefaultViewModelFactories.getActivityFactory(this);  
           ^  
      required: ComponentActivity.Factory  
      found: Hilt_RegisterActivity  
      reason: actual and formal argument lists differ in length  
/media/kishan/SSD-Data-0/Kishan/Android_WorkSpace/2821/MedicalApp/app/build/generated/source/kapt/debug/com/medical/medicalapp/ui/fragment/Hilt_DoctorMyPatientFragment.java:188: error: method getFragmentManager in class DefaultViewModelFactories cannot be applied to given types;  
    return DefaultViewModelFactories.getFragmentManager(this);  
           ^  
      required: Fragment.Factory  
      found: Hilt_DoctorMyPatientFragment  
      reason: actual and formal argument lists differ in length  
Note: Some input files use or override a deprecated API.  
Note: Recompile with -Xlint:deprecation for details.  
5 errors  
  
FAILURE: Build failed with an exception.
```

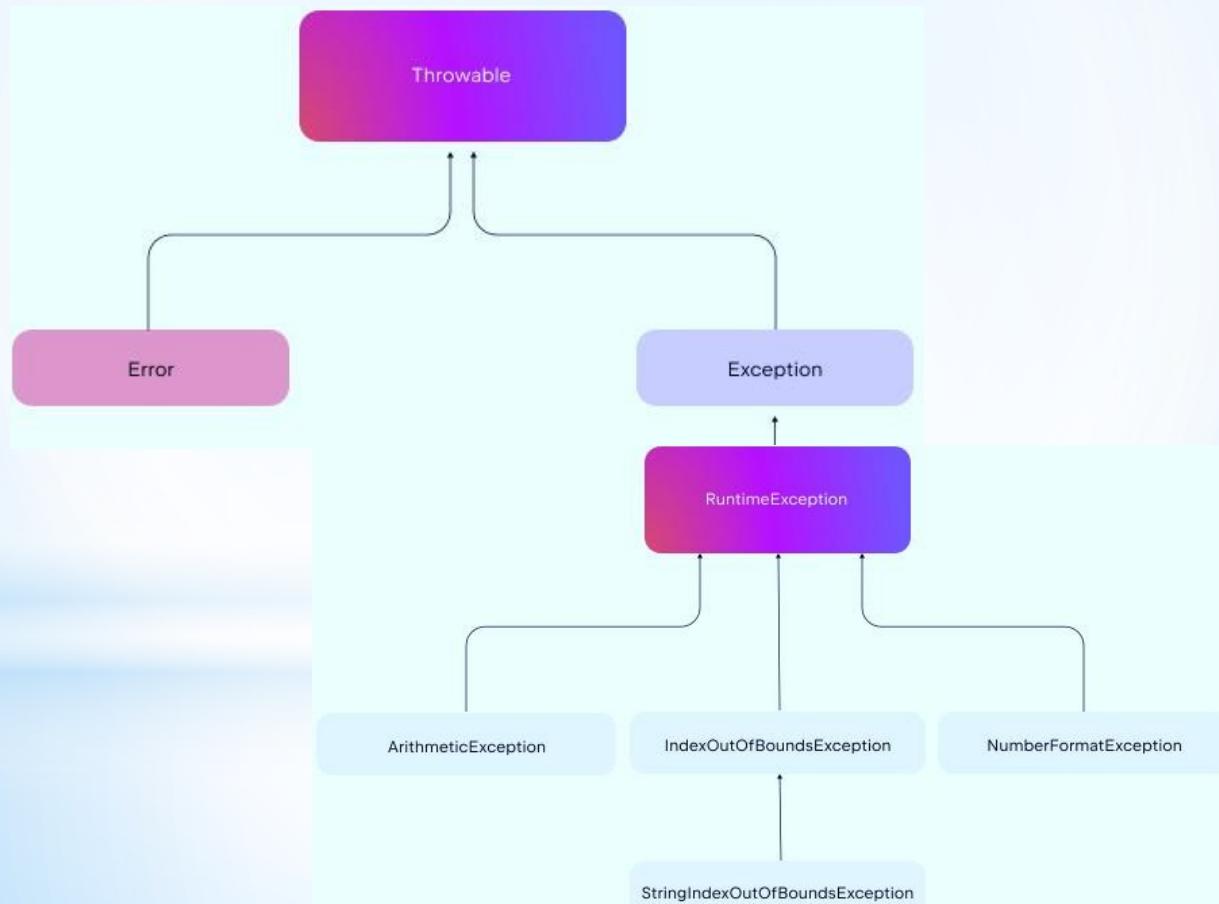
Runtime Error

Failed resolution of: Lcom/google/android/gms/tasks/Tasks;

END APPLICATION



# JERARQUÍA DE EXCEPCIONES



# REFERENCIAS

- \* Eckel, B. & Isakova, S. (2020). **Atomic Kotlin**. Mindview LLC: Leanpub, pp. 35-40.
- \* Griffiths D. & Griffiths D. (2016). **Head First Kotlin**. Canada: O'Reilly Media, pp. 44-132.
- \* Jemerov, D. & Isakova, S. (2017). **Kotlin in Action**. USA: Manning Publications Co, pp. 20-22.
- \* JetBrains Academy (2024). **Introduction to Kotlin**. Hyperskill Platform
- \* Macías, J. (2022). **CCNAv7: Introducción a Redes - Sistemas de numeración**. Semestre 2022-1 (agosto-noviembre). Facultad de Ingeniería, UNAM.
- \* Skeen, J. & Greenhalgh, D. (2018). **Kotlin Programming: The Big Nerd Ranch Guide**. Atlanta: Pearson, pp. 42-61.
- \* Varios (2024). **Kotlin Tutorial**. Enero 4, 2025, de W3Schools. Sitio web: [https://www.w3schools.com/KOTLIN/kotlin\\_intro.php](https://www.w3schools.com/KOTLIN/kotlin_intro.php)