

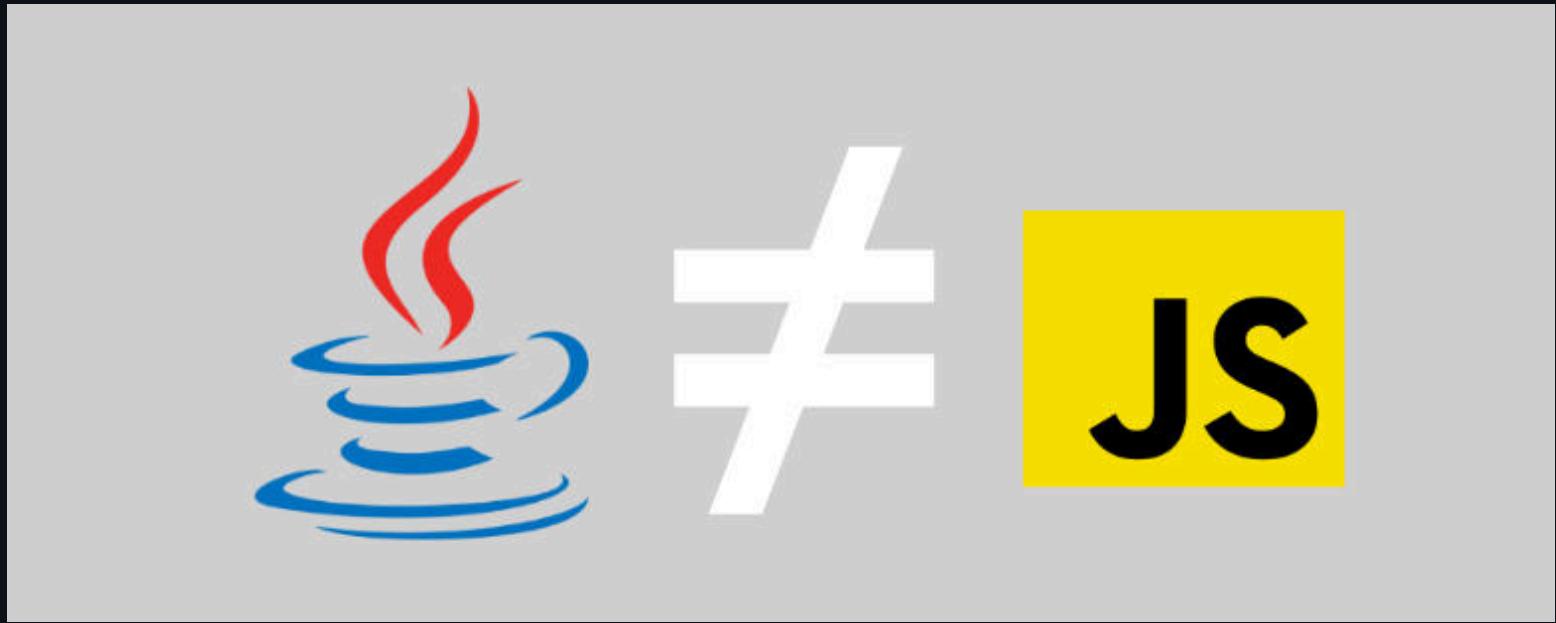
# Estructura de datos y algoritmos

Rodrigo Alvarez

[rodrigo.alvarez2@mail\\_udp.cl](mailto:rodrigo.alvarez2@mail_udp.cl)

# Que es java?

# Que es java?



# Que es java?

- Desarrollado por Sun Microsystems
- Lanzado en 1995
- Actualmente propiedad de Oracle
- Open Source



# Que es java?

- Lenguaje compilado a bytecode  
(corre encima de la jvm)
- Fuertemente orientado a objetos
- Sintaxis similar a C++
- Multiplataforma



Domina todos lenguajes de programación estudiando en EDteam

[ed.team/cursos](http://ed.team/cursos)



## MAJOR COMPANIES THAT USE



# Hello world

Java Hello World      JAVA ▾      RUN ►

```
1
```

STDIN      Input for the program ( Optional )

Output:

Click on RUN button to see the output

Tipo dato	Tamaño	Descripción
byte	1 byte	números del -128 al 127
short	2 bytes	-32,768 al 32,767
int	4 bytes	-2,147,483,648 al 2,147,483,647
long	8 bytes	-9,223,372,036,854,775,808 al 9,223,372,036,854,775,807
float	4 bytes	números racionales, hasta 7 dígitos decimales
double	8 bytes	hasta 15 dígitos decimales
boolean	1 bit (?)	true o false
char	2 bytes	un único carácter

## especificación tamaños

# Operadores aritméticos

Operator	Name	Description	Example
+	Addition	Adds together two values	x + y
-	Subtraction	Subtracts one value from another	x - y
*	Multiplication	Multiplies two values	x * y
/	Division	Divides one value by another	x / y
%	Modulus	Returns the division remainder	x % y
++	Increment	Increases the value of a variable by 1	++x
--	Decrement	Decreases the value of a variable by 1	--x

# Operadores de asignación

Operator	Example	Same As
=	x = 5	x = 5
+=	x += 3	x = x + 3
-=	x -= 3	x = x - 3
*=	x *= 3	x = x * 3
/=	x /= 3	x = x / 3
%=	x %= 3	x = x % 3

# Operadores de comparación

Operator	Name	Example
<code>==</code>	Equal to	<code>x == y</code>
<code>!=</code>	Not equal	<code>x != y</code>
<code>&gt;</code>	Greater than	<code>x &gt; y</code>
<code>&lt;</code>	Less than	<code>x &lt; y</code>
<code>&gt;=</code>	Greater than or equal to	<code>x &gt;= y</code>
<code>&lt;=</code>	Less than or equal to	<code>x &lt;= y</code>

# Operadores lógicos

Operator	Name	Description	Example
&&	Logical and	true if both statements are true	<code>x &lt; 5 &amp;&amp; x &lt; 10</code>
	Logical or	true if one of the statements is true	<code>x &lt; 5    x &lt; 4</code>
!	Logical not	Reverse the result	<code>!(x &lt; 5 &amp;&amp; x &lt; 10)</code>

# I/O: Input/Output en java

427p45t7h

JAVA ▾ RUN ►

```
1
```

STDIN

```
2 3
```

---

Output:

```
Ingresá dos ints:  
el primer numero es: 2  
el segundo numero es: 3  
La suma de los numeros es: 5
```

# Scanner class

La clase `Scanner` se utiliza para obtener la entrada del usuario, y es parte del paquete `java.util`.

Method	Description
<code>nextBoolean()</code>	Reads a <code>boolean</code> value from the user
<code>nextByte()</code>	Reads a <code>byte</code> value from the user
<code>nextDouble()</code>	Reads a <code>double</code> value from the user
<code>nextFloat()</code>	Reads a <code>float</code> value from the user
<code>nextInt()</code>	Reads a <code>int</code> value from the user
<code>nextLine()</code>	Reads a <code>String</code> value from the user
<code>nextLong()</code>	Reads a <code>long</code> value from the user

# Scanner

The screenshot shows a Java code editor interface. At the top right, there are two buttons: "JAVA ▾" and a pink "RUN ▶" button. The code area contains the following Java code:

```
427nv54rs
1
public class Main {
    public static void main(String[] args) {
        System.out.print("Ingrese una palabra");
        String palabra = System.console().readLine();
        System.out.println("La palabra ingresada es: " + palabra);

        System.out.print("Ingrese un numero");
        int numero = Integer.parseInt(System.console().readLine());
        System.out.println("El numero ingresado es: " + numero);

        System.out.print("Ingrese un numero flotante");
        double numeroFlotante = Double.parseDouble(System.console().readLine());
        System.out.println("El numero flotante ingresado es: " + numeroFlotante);
    }
}
```

Below the code, the input section shows "STDIN" with the input "hola 3". A horizontal line separates the input from the output. The output section is labeled "Output:" and displays the program's response:

Ingrese una palabra  
La palabra ingresada es: hola  
Ingrese un numero  
El numero ingresado es: 3  
Ingrese un numero flotante  
El numero flotante ingresado es: 3.14

# if, else, while, for

427nvkqmd      JAVA ▾      RUN ►

1      STDIN  
17

---

Output:

```
the number is odd
numeros pares:
2
4
6
8
10
12
14
16
divisibles por 3:
3
6
9
```

# Strings

The screenshot shows a Java code editor interface. At the top right are buttons for "JAVA ▾" and "RUN ▶". The code area contains the following Java code:

```
427nu2xbm
1
23
public class Main {
    public static void main(String[] args) {
        System.out.println("Hello");
        System.out.println("World");
        System.out.println("Hello World. Goodbye");
        System.out.println(true);
        System.out.println(false);
    }
}
```

Below the code, under "STDIN", is the input "23". Under "Output:" is the resulting console output:

```
H
e
l
l
o

W
o
r
l
d
Hello World. Goodbye
true
false
```

## String methods

# Arrays

427p4c9xn

JAVA ▾ RUN ►

1

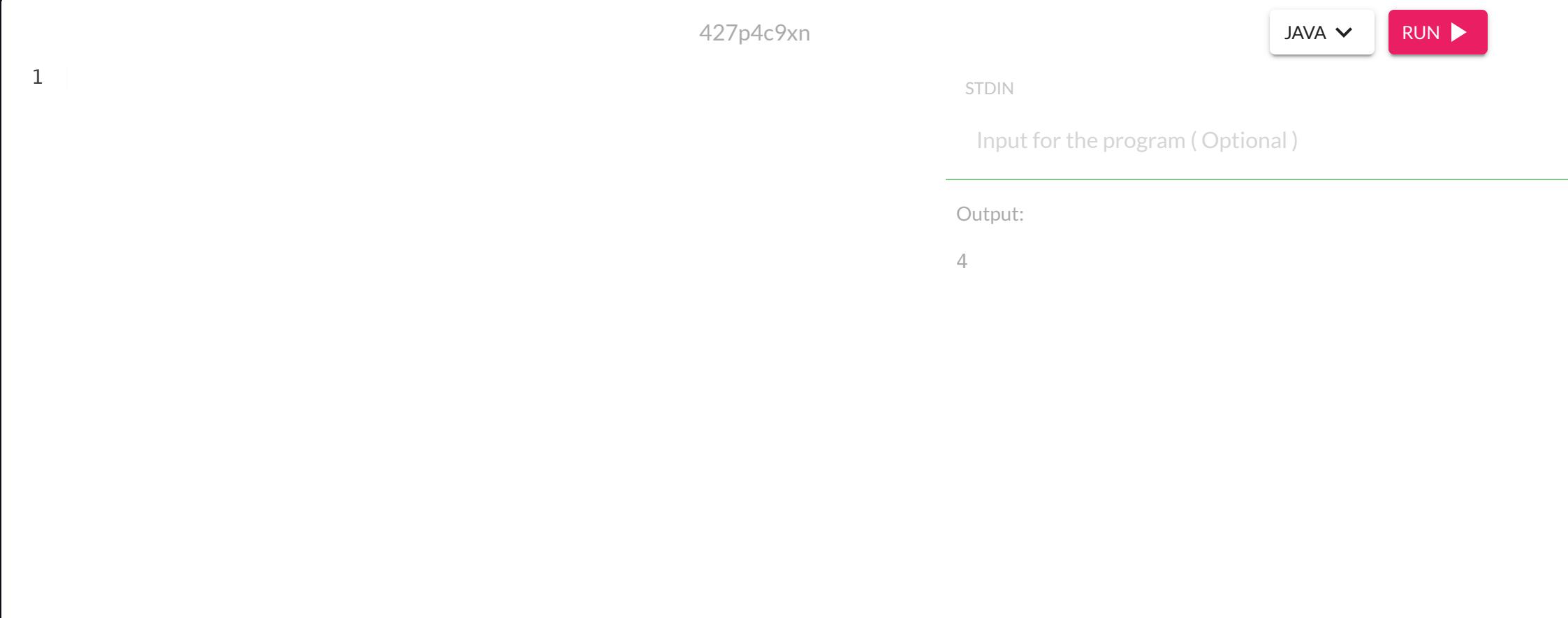
STDIN

Input for the program ( Optional )

---

Output:

4



# Métodos

```
[modificador de acceso] [tipo_retorno] nombre_funcion(tipo_1 parametro_1, tipo_2 parametro_2, ...){  
    // instrucciones  
    return valor;  
}
```

# Métodos

A screenshot of a Java code editor interface. At the top right, there are two buttons: "JAVA ▾" and a pink "RUN ▶" button. Below these buttons, the text "427p5fbvj" is displayed. In the main editor area, the number "1" is visible on the left. To the right of the editor, the word "STDIN" is followed by a light gray placeholder text "Input for the program ( Optional )". Below this, the word "Output:" is followed by the text "hi!". The entire interface is set against a white background.

```
1
427p5fbvj
STDIN
Input for the program ( Optional )
Output:
hi!
```

# Clases

A screenshot of a Java code editor interface. At the top right, there are two buttons: "JAVA ▾" and a red "RUN ▶" button. Below these buttons, the text "427pmavjh" is displayed. On the left side of the editor, the number "1" is visible, indicating the current line of code. The main code area contains the following Java code:

```
1 public class Clases {  
2     public static void main(String[] args) {  
3         System.out.println("miauuuu");  
4     }  
5 }
```

Below the code area, there are two sections: "STDIN" and "Input for the program (Optional)". The "Input for the program (Optional)" section is currently empty. To the right of the code area, the word "Output:" is followed by the text "miauuuu", which is the expected output of the program.

# Herencia

A screenshot of a Java code editor interface. At the top right, there are two buttons: "JAVA ▾" and a pink "RUN ▶" button. Below these buttons, the text "427pqryv5m" is displayed. In the main area, the number "1" is shown above a blank white space where code would typically be written. To the right of this white space, the word "STDIN" is followed by a light gray placeholder text "Input for the program ( Optional )". A horizontal line separates this from the "Output:" section below. The output text "Tuut, tuut!" and "Ford 2023 4" is displayed in black font.

```
1
STDIN
Input for the program ( Optional )

Output:
Tuut, tuut!
Ford 2023 4
```

## Interfaces

Las interfaces en Java son una colección de métodos abstractos y constantes que pueden ser implementados por cualquier clase que las implemente.

Una interfaz define un conjunto de métodos y su firma, pero no proporciona implementaciones de los métodos.

# Interfaces

A screenshot of an online Java code editor interface. At the top right, there are two buttons: "JAVA ▾" and a pink "RUN ▶" button. Below these buttons, the text "427pruj44" is displayed. On the left side of the editor area, the number "1" is shown above a single-line of code. The code consists of a class definition with a constructor and a method. The method uses the `System.out.println()` statement to output the string "The pig says: wee wee Zzz". To the right of the code, there are sections for "STDIN" and "Input for the program (Optional)". A horizontal line separates the code from the output section. The output section is labeled "Output:" and shows the printed text "The pig says: wee wee Zzz".

```
1
public class Pig {
    public Pig() {
        System.out.println("The pig says: wee wee");
        System.out.println("Zzz");
    }
}
```

# Generics

A screenshot of a Java code editor interface. At the top right, there are two buttons: "JAVA ▾" and a pink "RUN ▶" button. Below these buttons, the text "427psdcfw" is displayed. In the main editor area, the number "1" is visible on the left margin. To the right of the editor, the word "STDIN" is followed by a light gray placeholder text "Input for the program ( Optional )". Below the editor, the word "Output:" is followed by the number "3".

```
427psdcfw
1
STDIN
Input for the program ( Optional )
Output:
3
```

- `ArrayList`: Implementa un **TDA List** en array manipulando el tamaño de forma dinámica.
- `LinkedList`: Implementa un **TDA List** sobre una lista de nodos.
- `Queue`: Implementa un **TDA FIFO** en una lista de nodos.
- `Stack`: Implementa un **TDA LIFO** en una lista de nodos.
- `Map`: Implementa una estructura **TDA key-value** en un árbol o una tabla hash.
- `Set`: Implementa una estructura **TDA unique value** en un árbol o una tabla hash.



