



Clase N°6

# Repaso while (y otros)

Taller de Programación  
Computines



# Tipos de Datos



Atributos asociados a cada dato que indican al computador cómo administrarlos. Se destacan:

- 1) Número entero (**int**)
- 2) Números reales (**double**)
- 3) Caracteres (**char**)
- 4) Texto (**String**)
- 5) Valores de verdad (**boolean**)

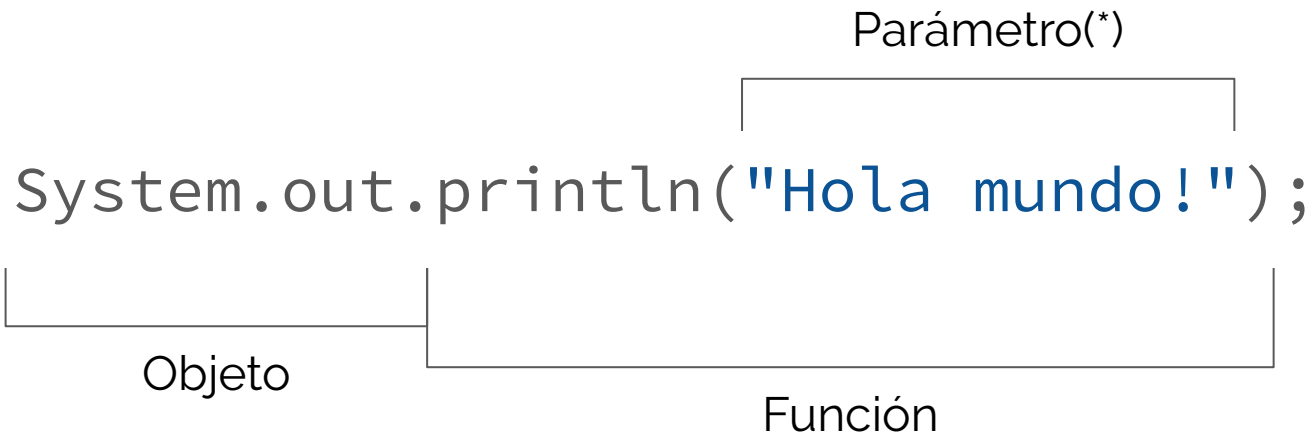
# Output

Parámetro(\*)

```
System.out.println("Hola mundo!");
```

Objeto

Función



(\*) En este caso está recibiendo "Hola Mundo", un String.

# Scanner

Para utilizarlo hay que importarlo

```
import java.util.Scanner;
```

Lo tenemos que crear y asignar a una variable

```
Scanner scanner = new Scanner(System.in);
```

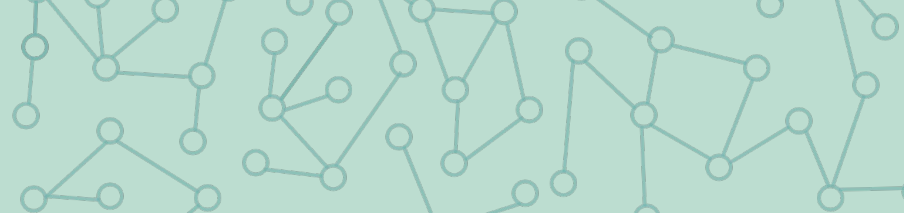
Después, le podemos pedir que nos diga lo que se ingresa

```
String linea = scanner.nextLine();
```

```
int numero = scanner.nextInt();
```

Notemos que hay que importar el Scanner, y que el resultado de nextLine se guarda como un String. Lo siguiente que le pedimos es un entero.

# Expresiones



Expresiones básicas para hacer operaciones entre variables

- 1) Suma, resta, multiplicación y división (+, -, \*, /)
- 2) Módulo, que es el resto de la división entera entre dos números (%)
- 3) Comparaciones (==, <=, >=, >, <)

# Operaciones

```
public class Main {  
    public static void main(String[] args) {  
        int a = 7;  
        int b = 4;  
  
        int result = a + b;  
        // El valor de result es 11  
  
        result = a - b;  
        // Ahora el valor de result es 3  
  
        result = 2 * a;  
        // Ahora el valor de result es 14  
  
        result = a / b;  
        // Ahora el valor de result es 1  
  
        result = a % b;  
        // ahora el valor de result es 3  
  
    }  
}
```

## ¿Cuáles patrones pueden observar en éste código?

1. ¿Cómo terminan algunas líneas? ¿Por qué es así?
2. Identifica los siguientes elementos:
  - a. Creaciones de una variable
  - b. Asignaciones de una variable
  - c. Comentarios
  - d. Expresiones
3. ¿Cuál es el valor de a y b al final del código?
4. ¿Cuál es la diferencia entre las primeras tres líneas? ¿Por qué?

# Comparaciones

```
public class Main {  
  
    public static void main(String[] args) {  
  
        int a = 3;  
        boolean comparison = (a == 3);  
  
        int b = 4;  
        comparison = (a == b);  
  
        comparison = (a >= 10);  
  
    }  
}
```

# Comparaciones

```
public class Main {  
  
    public static void main(String[] args) {  
  
        int a = 3;  
        boolean comparison = (a == 3);  
        // comparison es true  
  
        int b = 4;  
        comparison = (a == b);  
        // comparison ahora es false  
  
        comparison = (a >= 10);  
        // comparison es false, porque a no es mayor o igual a 10  
  
    }  
  
}
```



# Comparación entre palabras

```
String nombre1 = "Rodrigo";  
String nombre2 = "Vicente";  
boolean comparacion1 = nombre1.equals(nombre2);  
System.out.println(comparacion1); // false
```

`.equals()`

```
int largo1 = nombre1.length();  
int largo2 = nombre2.length();  
boolean comparacion2 = largo1 == largo2;  
System.out.println(comparacion2); // true
```

`.length()`

# Uso de varios operadores booleanos

```
boolean a = true;
```

```
boolean b = false;
```

```
boolean c = false;
```

```
a && b && c;
```

```
a || b || c;
```

```
a && (!b) && (!c);
```

```
(!a) || b || c;
```

```
(a && b) || (!c);
```

# Uso de varios operadores booleanos

```
boolean a = true;
```

```
boolean b = false;
```

```
boolean c = false;
```

```
a && b && c; // false
```

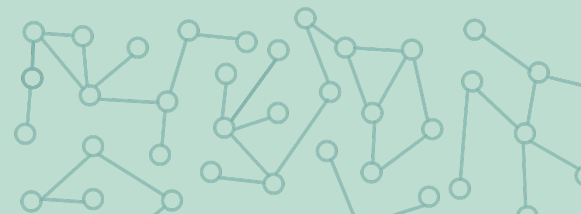
```
a || b || c; // true
```

```
a && (!b) && (!c); // true
```

```
(!a) || b || c; // false
```

```
(a && b) || (!c); // true
```

# If - else

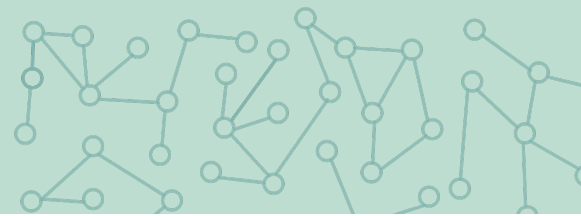


```
if (condicion) {  
    // bloque de código que sólo se ejecuta  
    // en caso de que sí se cumpla la condición  
}  
  
else if (condicion2) {  
    // bloque de código que sólo se ejecuta  
    // en caso de que sí se cumpla la condición 2  
}  
  
else {  
    // bloque de código que sólo se ejecuta  
    // en caso de que no se cumpla ninguna condición  
}
```

# Volvemos

Ciclos while

# Dato Freak



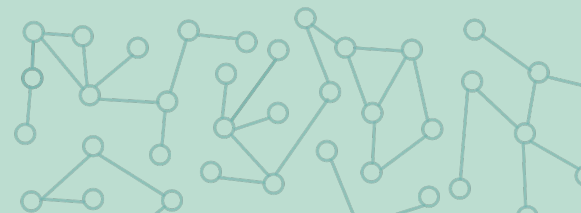
¿Cuánto creen que puede costar un error de código?

# Dato Freak



La Mariner 1 de la NASA fue la primera misión del Programa Mariner en intentar sobrevolar Venus. La misión tuvo que ser abortado por un error de software que causó un desvío en su trayectoria.

¿El error? Un guión mal transcrito.



¿Cuánto creen que puede costar un error de código?

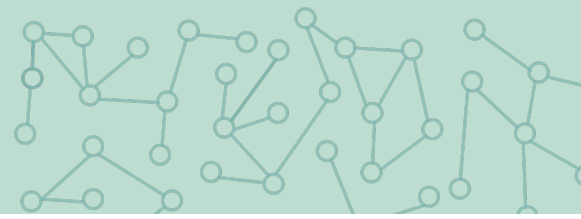
**150 Millones de dólares**



# Volvemos

Ciclos while

# Ejercicio 1



Imprimir los primeros 5 números naturales.

# Ejercicio 1



Imprimir los primeros 5 números naturales.

```
System.out.println(1);
```

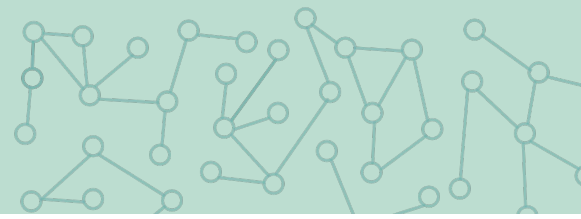
```
System.out.println(2);
```

```
System.out.println(3);
```

```
System.out.println(4);
```

```
System.out.println(5);
```

## Ejercicio 2



Imprimir los primeros 1.000.000 números naturales.

## Ejercicio 2

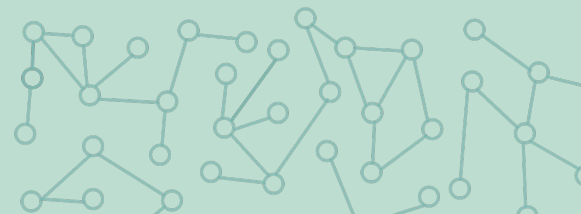
Imprimir los primeros 1.000.000 números naturales.



# Ciclos al rescate!! 2.0

(n❤️❤️)⊃—☆°.\*

# Usos



- Permiten ahorrar la repetición innecesaria de código.
- Ayudan cuando se debe hacer algo un número indefinido de veces.
- Validar input del usuario.

# Ciclos while



Hacer algo antes

Mientras se cumpla una **condición**:

Hacer algo

Hacer algo después



# Cómo hacerlo en código

A decorative graphic in the top right corner consisting of a network of small circles (nodes) connected by thin lines (edges), forming a complex, interconnected web-like structure.

Imprimir los números enteros entre el 0 y N.

# Cómo hacerlo en código

Imprimir los números enteros entre el 0 y N.

```
int numero = 0;
int N = 5;
System.out.println("Entraré al ciclo!");

while (numero < N){
    System.out.println("El número es " + numero);
    numero += 1;
}
System.out.println("Salí del ciclo!");
```

# Cómo hacerlo en código

A decorative graphic in the top right corner of the slide. It consists of a network of small, light teal circles (nodes) connected by thin, light teal lines (edges). The nodes are arranged in a somewhat irregular, branching pattern, resembling a simplified molecular structure or a network graph. The background of the slide is a solid light teal color.

Ahora, como imprimir desde el N al 0?

# Cómo hacerlo en código

Ahora, como imprimir desde el N al 0?

- \\_ (ツ) \\_ / -

# Cómo hacerlo en código

Ahora, como imprimir desde el N al 0?

```
int N = 5;
int numero = N;
System.out.println("Entraré al ciclo!");

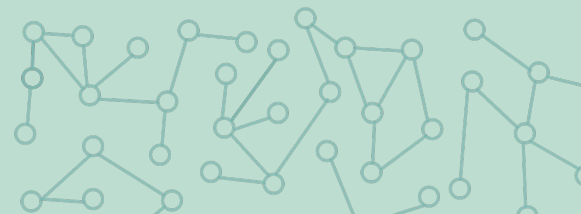
while (numero >= 0){
    System.out.println("El número es " + numero);
    numero -= 1;
}
System.out.println("Salí del ciclo!");
```

# Cómo hacerlo en código

Ahora, como imprimir desde el N al 0?

()

# Loop infinito



Este programa no se detendrá nunca, ¿porqué?

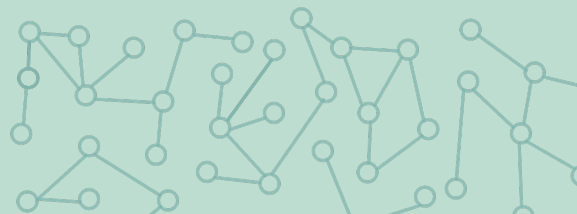
```
boolean booleano = true;
```

```
System.out.println("Entraré al ciclo!");
```

```
while (booleano){  
    System.out.println("Sigo en el ciclo!");  
}
```

```
System.out.println("Salí del ciclo!");
```

# Sentencia break



Este programa hará una sola iteración en el loop.

```
boolean booleano = true;
```

```
System.out.println("Entraré al ciclo!");
```

```
while (booleano){  
    System.out.println("Sigo en el ciclo!");  
    break;  
}
```

```
System.out.println("Salí del ciclo!");
```



# Cómo hacerlo en código

A decorative pattern in the top right corner consisting of a network of interconnected nodes and lines, resembling a graph or a molecular structure, in a light teal color.

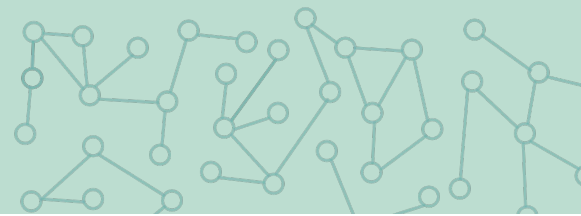
Imprimir los números enteros entre el 0 y N con break.

# Cómo hacerlo en código

Imprimir los números enteros entre el 0 y N con break.

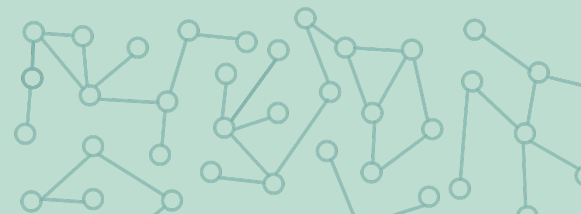
```
int numero = 0;
int N = 5;
System.out.println("Entraré al ciclo!");
while (true){
    if (numero >= N){
        break;
    }
    System.out.println("El número es " + numero);
    numero += 1;
}
System.out.println("Salí del ciclo!");
```

# Ejercicios



Imprimir los números enteros entre el 0 y un número entero  $N$  entregado por el usuario.

# Ejercicios



Imprimir los números enteros entre el 0 y un número entero N entregado por el usuario.

```
int numero = 0;
Scanner sc = new Scanner(System.in);
int N = sc.nextInt();
System.out.println("Entraré al ciclo!");
while (numero < N){
    System.out.println("El número es " + numero);
    numero += 1;
}
System.out.println("Salí del ciclo!");
```

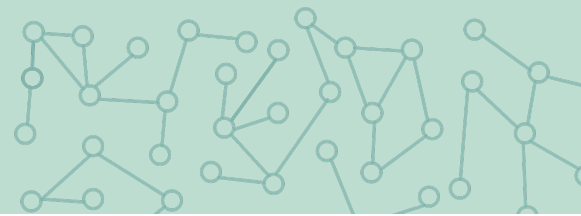
# Ejercicio (Hackerrank)

A decorative pattern in the top right corner consisting of a network of small circles (nodes) connected by thin lines (edges), resembling a graph or molecular structure.

Escribe un programa que primero reciba un número entero, llamémoslo N. Seguido de esto deberá leer las siguientes N líneas, las cuales contendrán un número entero cada una. La salida del programa deberá ser la suma de todas esas líneas.

# Ejercicio (Hackerrank)

```
Scanner sc = new Scanner(System.in);
int cantidad = 0;
int suma = 0;
int N = sc.nextInt();
while (cantidad < N){
    int numero = sc.nextInt();
    suma += numero;
    cantidad += 1;
}
System.out.println(suma);
```



Testcase 0 ✓

Testcase 1 ✗

**Your code did not pass this test case.**

**Input (stdin)**

0

**Your Output (stdout)**

0

**Expected Output**

~ no response on stdout ~

**Compiler Message**

Wrong Answer

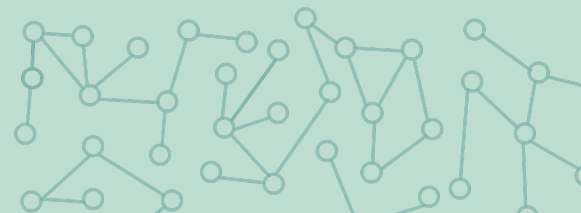
# Ejercicio (Hackerrank)

$(n \overline{\cup}) \supset - \star^{\circ} *$

```
Scanner sc = new Scanner(System.in);
int cantidad = 0;
int suma = 0;
int n = sc.nextInt();
while (cantidad < n){
    int numero = sc.nextInt();
    suma += numero;
    cantidad += 1;
}
if (n > 0) {
    System.out.println(suma);
}
```



# Actividad



La Cultura Programística pide:

Palabras que puedes encontrar en un código hecho en Java (No cuentan "Strings" ni números) como : `public-static-void-main-string-args`

# Ejercicio (Hackerrank)

```
Scanner sc = new Scanner(System.in);
int cantidad = 0;
boolean gano = true;
while (cantidad < 20) {
    String palabra = sc.nextLine();
    if (estaRepetida(palabra)){
        gano = false;
        break;
    }
    cantidad += 1;
}
if (gano) {
    System.out.println("Ganaron");
}
else {
    System.out.println("Perdieron");
}
```

Clase N°6

# Mentimeter

Taller de Programación  
Computines



# Ejercicio 1 (Mentimeter)

```
boolean uno = true;

boolean dos = false;

int num = 3;

int num2 = 7;

while(num < num2){

    if(uno != dos){

        System.out.println("Hola");

    }

    num++;

}
```

# Ejercicio 2 (Mentimeter)

```
boolean uno = true;

boolean dos = false;

int num = 3;

int num2 = 7;

while((num < num2) && (uno != dos)){

    if(num > 5){

        System.out.println("Hola");

    }

    num++;

}
```

# Ejercicio 3 (Mentimeter)

```
boolean uno = true;
boolean dos = false;
int numero = 80;
int num = 3;
int num2 = 7;
while((num < num2) && (uno != dos)){
    if(num > 5){
        uno = false;
    }
    num++;
}
System.out.println(num);
```