

# Tecnologías de la Información II

## Clase 02: Expresiones

Daniela Opitz, Diego Caro  
[dopitz@udd.cl](mailto:dopitz@udd.cl)



Basada en presentaciones oficiales de libro Introduction to Programming in Python (Sedgewick, Wayne, Dondero).

Disponible en <https://introcs.cs.princeton.edu/python>

# Clase de Hoy

- Tips para imprimir strings
- Operadores aritméticos
- Operadores de comparación (==, !=, <, <=, >=, >)
- Operadores de lógica (or, and, not)
- Tablas de verdad
- Condicionales: if, else, elif

# Tips para imprimir strings

01\_strings\_utilities.py

```
#Imprimir elementos separados por coma
print ('manzana', 'naranja', 'pera')

#Imprimir elementos separados por un tab (\t)
print('manzana\tnaranja')

#Imprimir elementos separados por una linea (\n)
print ('manzana\nnaranja')

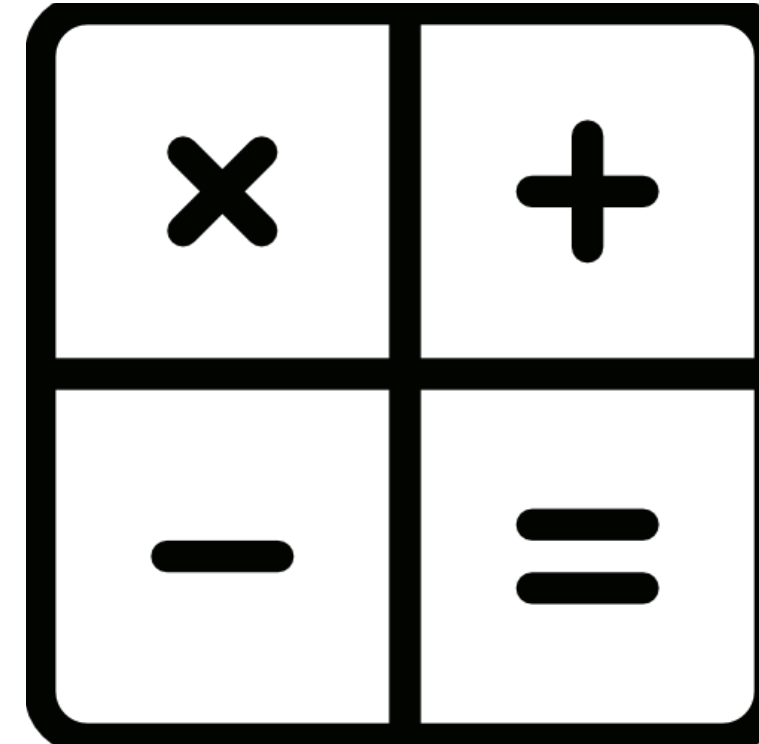
#Imprimir string que utilizan apostrofes
print ('It\'s raining')

#Imprimir string con comillas
print("Hola", "\"Hola\"")

#Imprimir backslash (\)
print ('"\" es backslash')
```

# Operador

- Tiene diversos significados dependiendo del contexto (matemáticas, física, lógica, etc).
- Para nosotros será un símbolo (o palabra) que indica que debe llevarse cabo una operación específica.



**Algunos operadores**

# Operadores Aritméticos

- Sea  $a = 10$  y  $b = 20$

Operador	Descripción	Ejemplo
+	Suma	$a + b = 30$
-	Resta	$a - b = -10$
*	Multiplicación	$a * b = 200$
/	División	$b / a = 2$
%	Módulo – Devuelve el resto de la división	$b \% a = 0$
**	Exponente – Realiza exponencial	$a ** b = 10 \text{ a la } 20$
//	División baja - Devuelve el entero de la división	$9 // 2 = 4$ y $9,0 // 2,0 = 4,0$

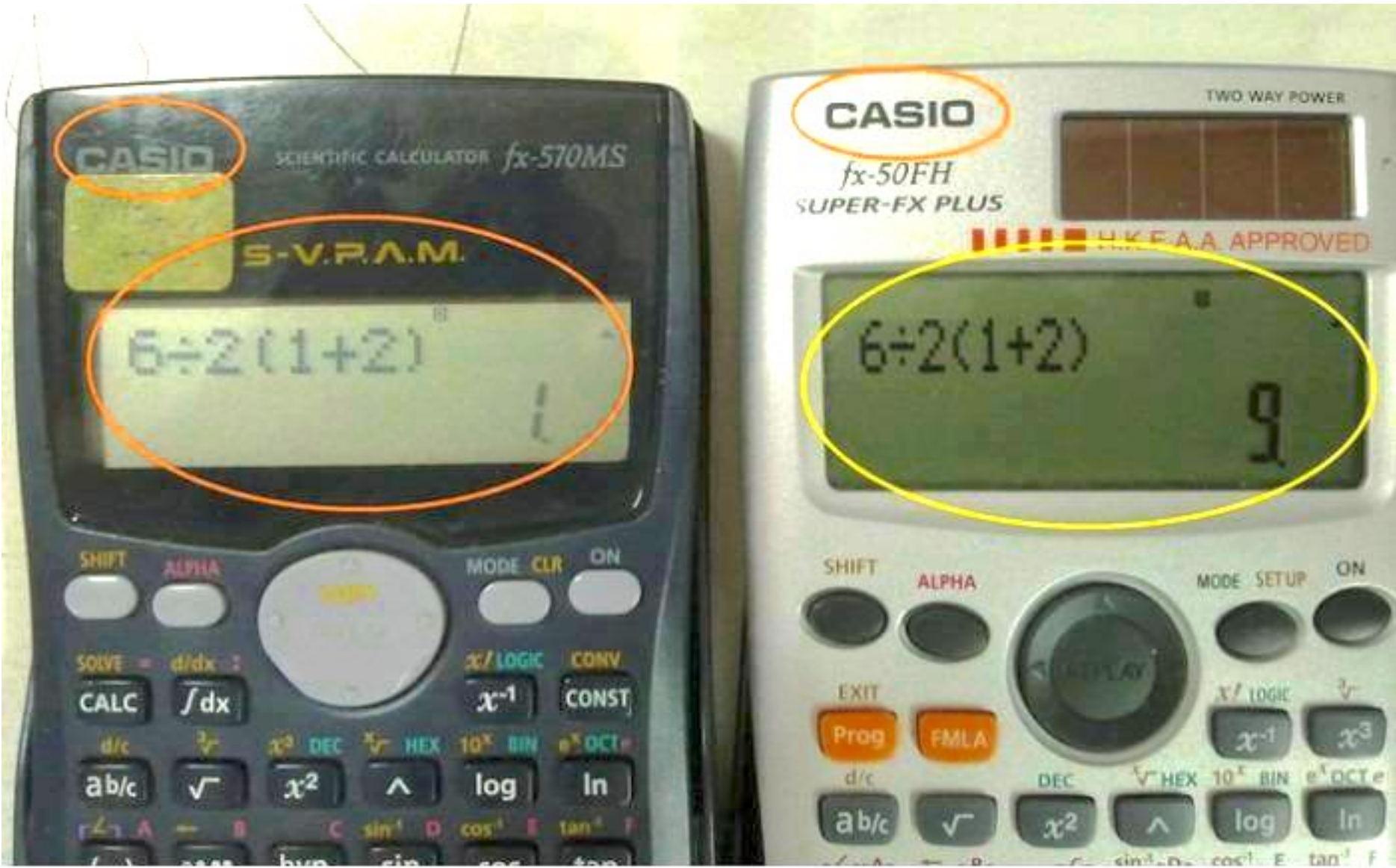
**Operadores Aritméticos en Python**

# Orden

- Orden en el que se evalúa una expresión.

Prioridad	1	2	3	4	5	6
Operador	( )	*	/	%	+	-

Expresión	Resultado
2+3*7	
6-2*4	
(6-2)*4	
4*5%3	
(12*(-1))	
(10+1)%9	
6/2(1+2)	





# Operadores de Comparación

- Permiten verificar si variables cumplen algunas reglas básicas.
- Devuelven un valor de tipo bool (**True**, **False**)

Sintáxis	Operador	Ejemplo	Resultado
a < b	menor que	2 < 5	
a <= b	menor o igual	2 <= 2	
a >= b	mayor o igual	2 >= 32	
a > b	mayor	0 > -1	
a == b	igual	1 == -1	
a != b	distinto	1 != -1	

```
a = 2
b = -1
print('a < b:', a < b)
print('a > b:', a > b)
print('a <= b:', a <= b)
print('a >= b:', a >= b)
print('a == b:', a == b)
print('a != b:', a != b)
```



```
a < b: False
a > b: True
a <= b: False
a >= b: True
a == b: False
a != b: True
```

# Operadores Lógicos

- Los operadores lógicos proporcionan un resultado de acuerdo al cumplimiento o no de una cierta condición.
- Operadores lógicos en Python (**or**, **and**, **not**)
- El resultado de **or** es verdadero cuando cualquiera de los operadores lo es.
- El resultado de **and** es verdadero solo cuando ambos operadores lo son.
- El operador **not** invierte el valor del operando.

Sintáxis	Operador
not	Negación
and	Conjunción
or	Disyunción



# Operadores Lógicos

Tabla de Verdad

a	b	not a	a or b	a and b
False	False	True	False	False
False	True	True	True	False
True	False	False	True	False
True	True	False	True	True

Prioridad	1	2	3	4	5
Operador	==	!=	not	and	or

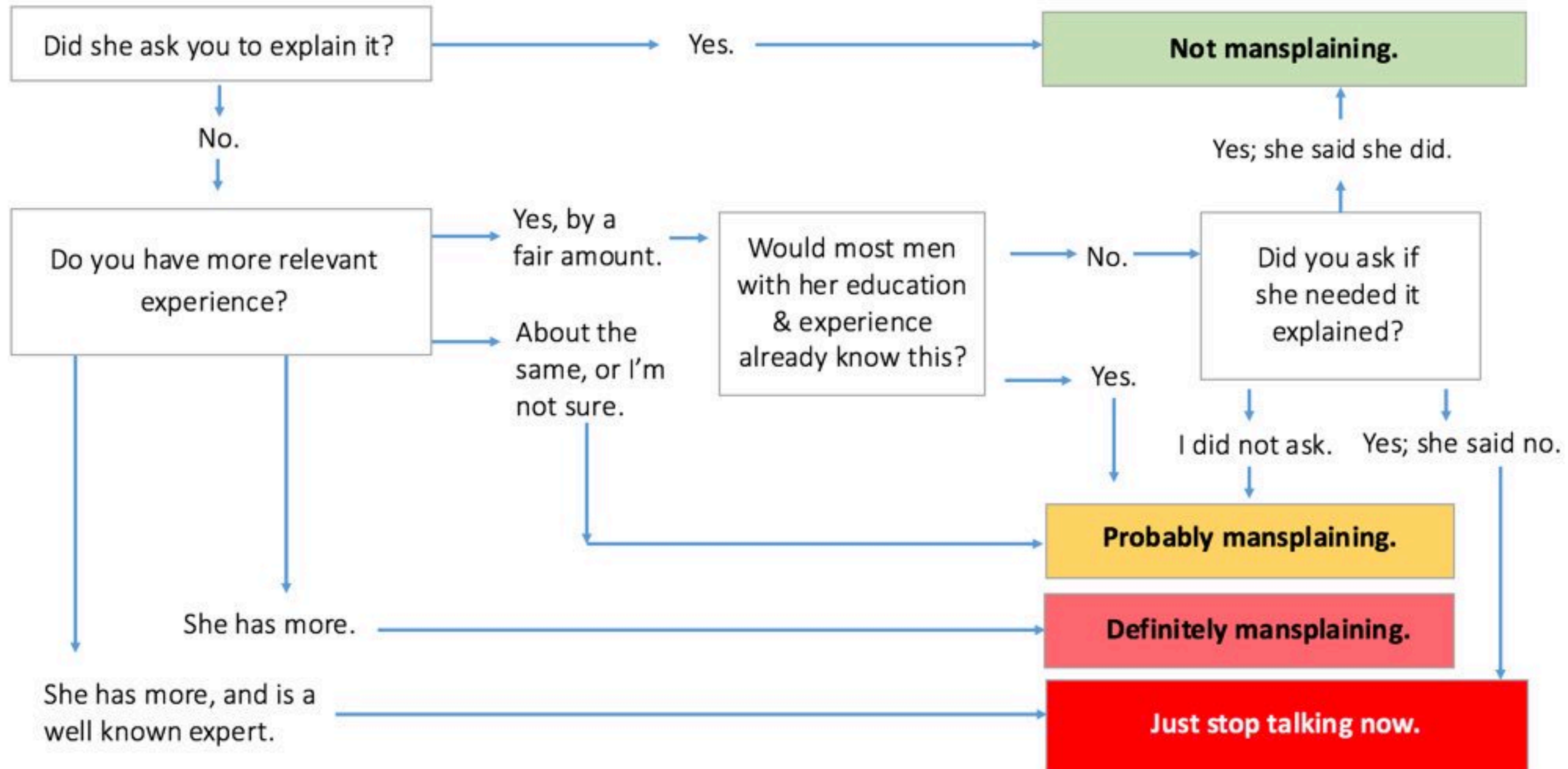
```
a = True
b = False
c = a or b
print('not a =', not a)
print('a or b =', a or b)
print('a and b =', a and b)
print('c =', c)
```



```
not a = False
a or b = True
a and b = False
c = True
```

# Mansplaining

## Am I mansplaining?



<https://twitter.com/kimgoodwin/status/1020029572266438657>

# Condicionales

- Palabras claves: **if**, **elif**, **else**
- Permiten condicionar la ejecución de uno o varios bloques de sentencias al cumplirse una o varias condiciones.
- Sintaxis:
  - **if**: *condición*: aquí van las instrucciones que se ejecutan si la condición es cierta (**True**)
  - **else**: conjunto de instrucciones que se ejecutan cuando la condición es falsa (**False**)
  - **elif**: combinación de if y else que extiende la secuencia de **if** para ejecutar una sentencia diferente

```
1 a = 50
2 b = 10
3 if a > b:
4     print('a es mayor que b')
```

# Condicionales

- ¿Cómo resuelto un problema que involucra condicionales?
1. Descomponer el problema en etapas
  2. Para cada etapa reconocer qué reglas se deben cumplir
  3. Ejecutar acciones por cada regla

```
1 a = 5
2 b = 10
3 if a > b:
4     # cuando la condición se cumple
5     print('a es mayor que b')
6 else:
7     # cuando la condición no se cumple
8     print('a es menor o igual que b')
```

# Condicionales

- Probar varias condiciones: elif <condicion>

```
1 a = int(input('ingrese a: '))
2 b = int(input('ingrese b: '))
3 if a > b:
4     # Cuando la condición se cumple
5     print('a es mayor que b')
6 elif a < b:
7     # Cuando la condición se cumple
8     print('a es menor que b')
9 else:
10    # Cuando ni la primera, ni la segunda
11    # condición se cumplen
12    print('a es igual a b')
```

```
1 a = int(input('ingrese a: '))
2 b = int(input('ingrese b: '))
3 if a > b:
4     # Cuando la condición se cumple
5     print('a es mayor que b')
6 else:
7     if a < b:
8         # Cuando la condición se cumple
9         print('a es menor o igual que b')
10    else:
11        # Cuando ni la primera, ni la segunda
12        # condición se cumplen
13        print('a es igual a b')
```

**Ambos códigos son  
equivalentes!**

# Primer Algoritmo!

- Pregunta:

¿Cuándo un año es bisiesto?

- Respuesta:

Es divisible por 400, o bien es divisible por 4 pero no por 100