

1- Operadores y Expresiones

Operadores

Son cálculos llevados a cabo sobre dos argumentos conocidos como operandos

Operando [operador] Operando

+ - / *

Expresiones

Los operadores aritméticos (+, -, /, *) dan lugar a expresiones de distintos tipos:

- Aritméticas si ambos operandos son valores literales:

$$2 + 5 \quad -1.4 * 54 \quad 1/2.5$$

- Algebraicas si al menos un operando es una variable:

$$radio * 3.14 \quad (nota_1 + nota_2) / 2$$

Es una incógnita y si el álgebra sirve para algo es precisamente para trabajar con incógnitas.

2- Tipos de datos lógicos



Constantes booleanas

True = 1

False = 0

Puedo crear variables de tipo lógico o booleano

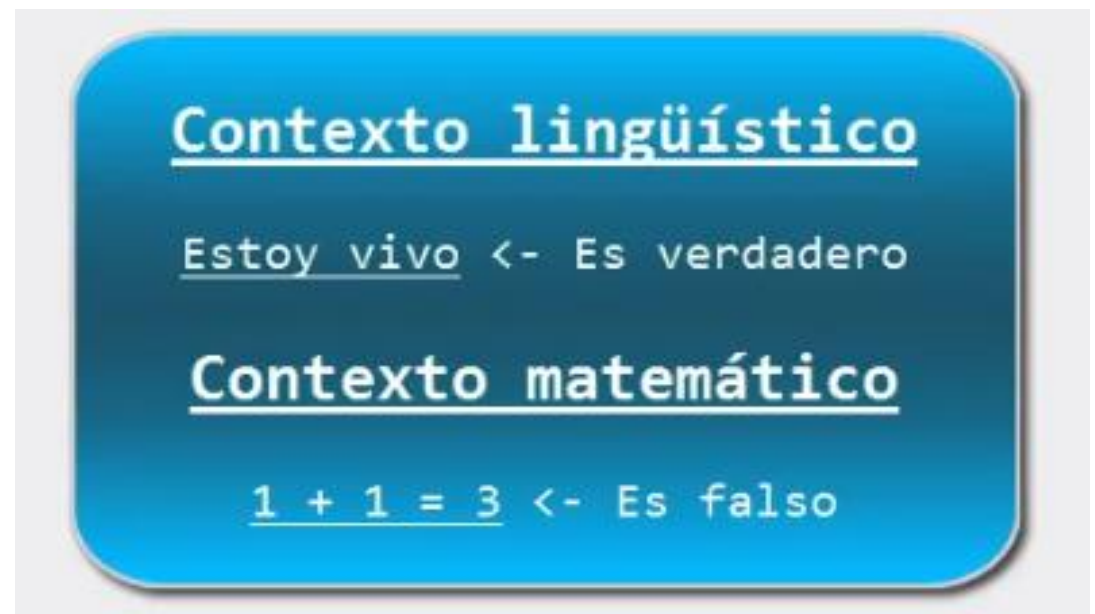
quiere_salir = True

Expresiones lógicas

1 + 1 es igual que 3 ?

tengo a= 2, b= 3

a es mayor que b ?



3- Operadores relacionales

Se utilizan para evaluar condicionales; como respuesta obtendremos booleanos.

Símbolo	Operación	Ejemplo	Resultado
==	Igual que	5==5	True
!=	Diferente que	4!=5	True
>	Mayor que	98>100	False
<	Menor que	12>=39	False
>=	Mayor o Igual	6>=6	True
<=	Menor o Igual	3<=8	True

File Edit View Insert Cell Kernel Help



```
In [1]: 1 + 1 == 3
```

```
Out[1]: False
```

```
In [2]: 1 + 1 == 2
```

```
Out[2]: True
```

Comparan dos valores y devuelven Verdadero o Falso

```
In [1]: # Igual que  
3 == 2
```

Out[1]: False

```
In [2]: # Distinto de  
3 != 2
```

Out[2]: True

```
In [3]: # Mayor que  
3 > 2
```

Out[3]: True

```
In [4]: # Menor que  
3 < 2
```

Out[4]: False

```
In [11]: # Mayor o igual que  
3 >= 4
```

Out[11]: False

```
In [14]: # Menor o igual que  
3 <= 4
```

Out[14]: True

```
In [15]: a = 10  
b = 5  
a > b
```

```
Out[15]: True
```

```
In [16]: b != a
```

```
Out[16]: True
```

```
In [17]: a == b*2
```

```
Out[17]: True
```


Comparación de cadenas

```
In [18]: "Hola" == "Hola"
```

```
Out[18]: True
```

```
In [19]: "Hola" != "Hola"
```

```
Out[19]: False
```

```
In [20]: c = "Hola"  
c[0] == "H"
```

```
Out[20]: True
```

```
In [21]: c[-1] == "a"
```

```
Out[21]: True
```

Comparación de listas

```
In [22]: l1 = [0,1,2]  
         l2 = [2,3,4]  
         l1 == l2
```

Out[22]: False

```
In [23]: len(l1) == len(l2)
```

Out[23]: True

```
In [24]: l1[-1] == l2[0]
```

Out[24]: True

Comparación de booleanos

```
In [25]: True == True
```

```
Out[25]: True
```

```
In [26]: False == True
```

```
Out[26]: False
```

```
In [27]: False != True
```

```
Out[27]: True
```

```
In [28]: True > False
```

```
Out[28]: True
```

```
In [29]: False > True
```

```
Out[29]: False
```

Operadores lógicos

- Not

```
In [1]: not True
```

```
Out[1]: False
```

```
In [2]: not True == False
```

```
Out[2]: True
```

Símbolo	Operación	Ejemplo	Resultado
and	Conjunción	12>2 and 5<10	True
or	Disyunción	9!=6 or 8<=5	True
not	Negación	not True	False

- And

```
In [3]: True and True
```

```
Out[3]: True
```

```
In [4]: True and False
```

```
Out[4]: False
```

```
In [5]: False and True
```

```
Out[5]: False
```

```
In [6]: False and False
```

```
Out[6]: False
```

- Or

```
In [12]: True or True
```

```
Out[12]: True
```

```
In [13]: True or False
```

```
Out[13]: True
```

```
In [14]: False or True
```

```
Out[14]: True
```

```
In [15]: False or False
```

```
Out[15]: False
```

Conjunción

- Viene de conjunto
- Sinónimo de unido, contiguo o cercano
- Agrupa uniendo

Disyunción

- Viene de disyunto
- Sinónimo de separado, apartado o distante
- Agrupa separando

a = 0 o Falso

b = 1 o Verdadero

c = 0 o Falso

entonces

$$a \text{ or } b \text{ or } c = 0 + 1 + 0 = 1$$

$$a \text{ and } b \text{ and } c = 0 * 1 * 0 = 0$$

```
a = 13  
a > 10 and a < 20
```

```
a = 45  
a > 10 and a < 20
```

```
c = "SALIR"  
c == "EXIT" or c == "FIN" or c == "SALIR"
```

```
In [11]: c = "Hola Mundo"  
len(c) >= 20 and c[0] == "H"
```

```
In [17]: c = "OTRA COSA"  
c == "EXIT" or c == "FIN" or c == "SALIR"
```

Out[17]: False

```
In [20]: c = "Lector" ]  
c[0] == "H" or c[0] == "h"
```

Out[20]: False

Expresiones anidadas



The screenshot shows the Jupyter Notebook interface. The title bar reads "jupyter Lección 4 - Expresiones anidadas (unsaved changes)". The menu bar includes "File", "Edit", "View", "Insert", "Cell", "Kernel", and "Help". The toolbar contains icons for saving, adding cells, undo, redo, copy, paste, and navigation. The code cell contains the following Python code:

```
In [1]: a = 10  
        b = 5  
        a * b - 2**b >= 20 and not (a % b) != 0
```

The output of the cell is:

```
Out[1]: False
```

Establecer las normas de precedència

Primero los paréntesis porque tienen prioridad

```
10 * 5 - 2**5 >= 20 and not (10 % 5) != 0  
10 * 5 - 2**5 >= 20 and not 0 != 0 # paréntesis
```



```
In [1]: a = 10  
b = 5  
a * b - 2**b >= 20 and not (a % b) != 0
```

Segundo, expresiones aritméticas por sus propias reglas

$10 * 5 - \underline{2**5}$	≥ 20	and	not 0	$\neq 0$	
$\underline{10 * 5} - 32$	≥ 20	and	not 0	$\neq 0$	# exponente
$\underline{50 - 32}$	≥ 20	and	not 0	$\neq 0$	# multiplicación
18	≥ 20	and	not 0	$\neq 0$	# resta

Tercero, relacionales

$\underline{18 \geq 20}$	and	not	$\underline{0 \neq 0}$
False	and	not	False

Finalmente, lógicos

False	and	<u>not False</u>
<u>False</u>	and	True
False		

Práctica P01

Calcula mentalmente, y luego programa y comprueba las siguientes sentencias:

$a = 7 + 12 * 5 - 8 + 1$

$b = 21 / 3 - 8 * 0.5$

$c = 69 \% 3 - 6 ** 2$

1. $a == b$

2. $a > b$

3. $c > b$

4. $c > \text{abs}(c)$

5. $a/10 == \text{math.sqrt}(-1 * c)$

6. $a/b * -1 != 56 - b$

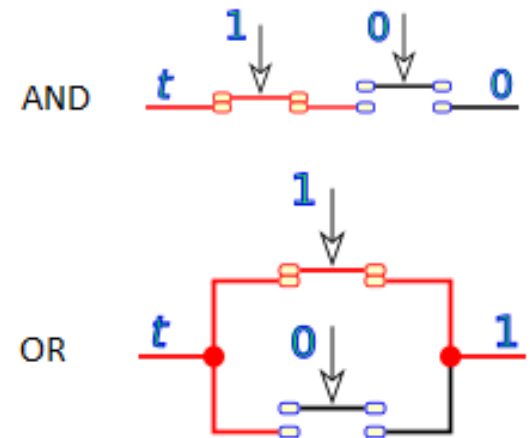
7. $a != b \text{ and } c != b$

8. $a > b \text{ or } b > c$

9. $c > a \text{ or } c > b$

Algebra de Boole

El álgebra de Boole, también llamada álgebra booleana, en electrónica digital, informática y matemática es una estructura algebraica que esquematiza las operaciones lógicas.



$a + b = b + a$	$a * b = b * a$
$a + (b + c) = (a + b) + c$	$a * (b * c) = (a * b) * c$
$a + (b * c) = (a + b) * (a + c)$	$a * (b + c) = (a * b) + (a * c)$
$a + a * b = a$	$a * (a + b) = a$

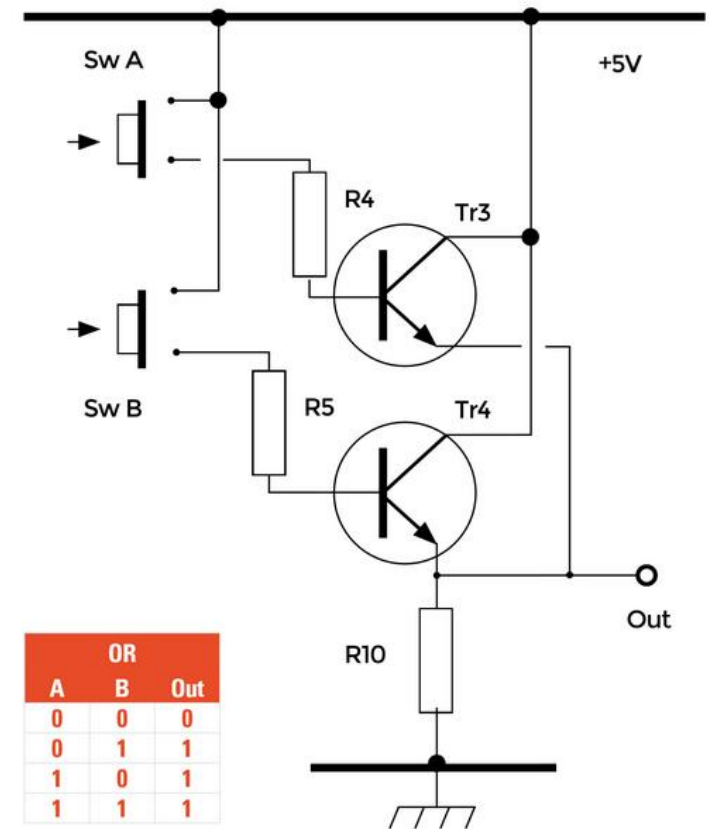
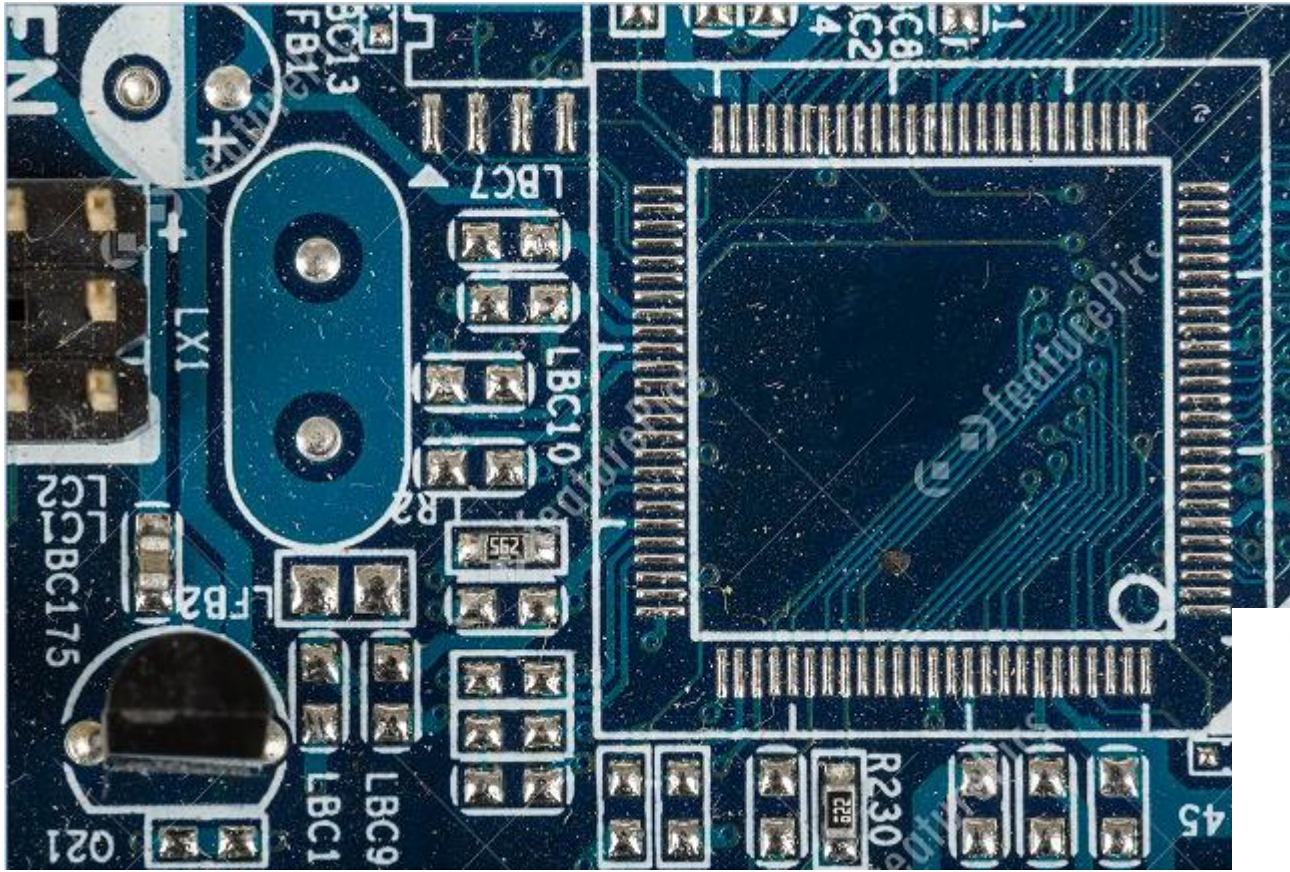
$$\overline{a + b} = \bar{a} \cdot \bar{b}$$

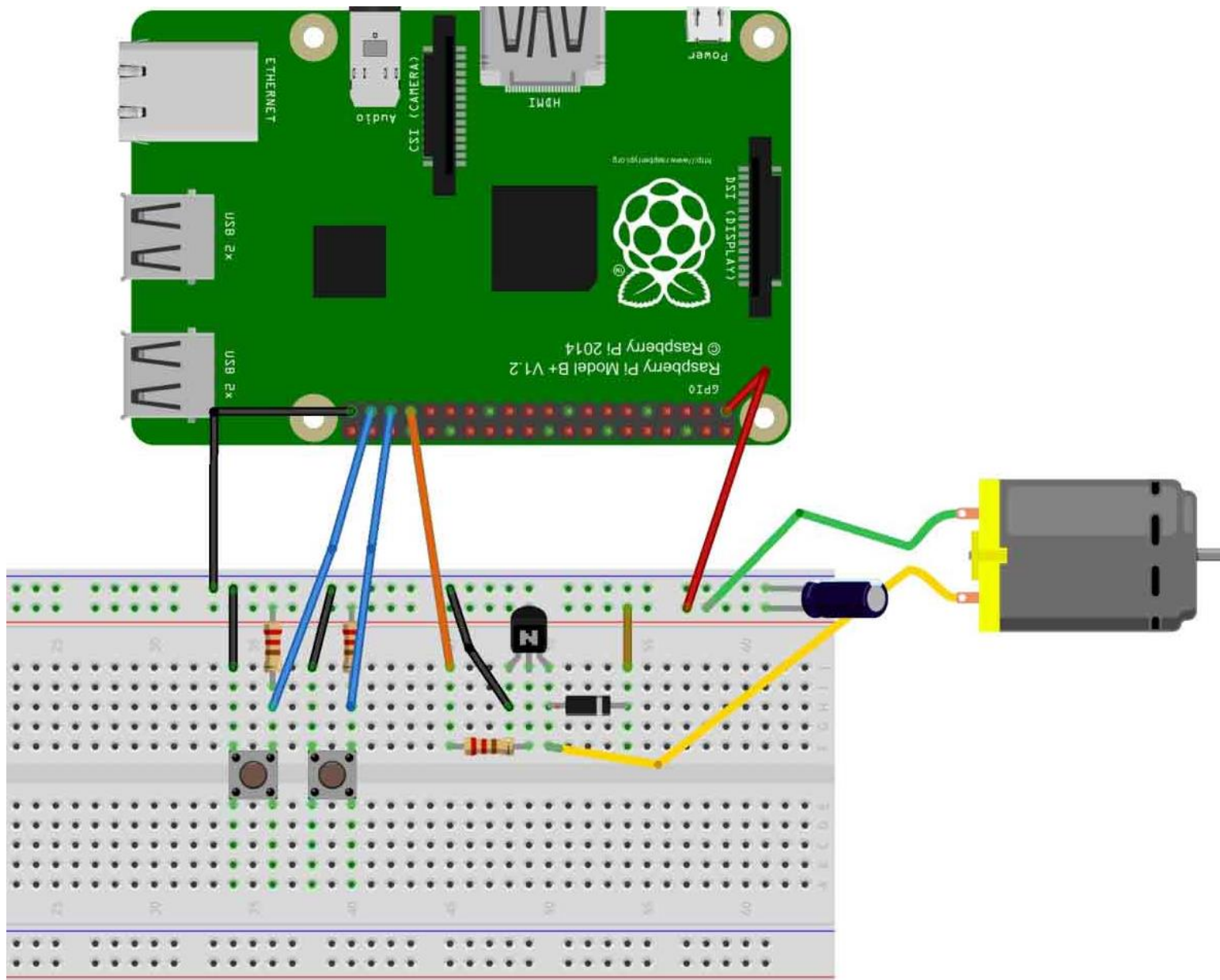
$$\overline{a \cdot b} = \bar{a} + \bar{b}$$

Empleando esta notación las leyes de De Morgan se representan:

$$\text{NOT } (a \text{ OR } b) = \text{NOT } a \text{ AND NOT } b$$

$$\text{NOT } (a \text{ AND } b) = \text{NOT } a \text{ OR NOT } b$$





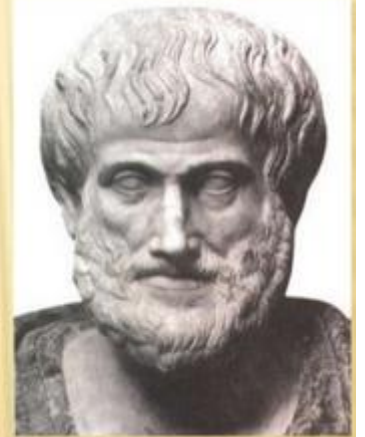
<https://projects.raspberrypi.org/en/projects/halfadder/3>

Lógica Clásica



✧ Aristóteles (384 a. C. – 322 a. C.)

Fundador de la lógica como herramienta básica de todas las ciencias. Sostuvo que la verdad se manifiesta en el juicio verdadero y el argumento válido en el silogismo.



Lógica de proposiciones

Una proposición lógica es cualquier expresión que puede ser verdadera o falsa, pero no las dos al mismo tiempo. Algunos ejemplos de proposiciones son:

- El año empieza con el mes de enero.
- Enero tiene 30 días
- $1 + 1 = 2$



p	q	$p \text{ or } q$	$\text{not } (p \text{ or } q)$	$p \text{ and } q$
p	q	$(p \vee q)$	$\sim (p \vee q)$	$(p \wedge q)$

p = El año empieza en enero.

q = Enero tiene 30 días

NOMBRE DE LA LEY	LOGICA DE PROPOSICIONES
1. Idempotencia	<ul style="list-style-type: none"> $p \wedge p \Leftrightarrow p$ $p \vee p \Leftrightarrow p$
2. Identidad	<ul style="list-style-type: none"> $p \wedge (V) \Leftrightarrow p$ $p \vee (F) \Leftrightarrow p$
3. Dominación	<ul style="list-style-type: none"> $p \wedge (F) \Leftrightarrow (F)$ $p \vee (V) \Leftrightarrow (V)$
4. Conmutativa	<ul style="list-style-type: none"> $p \wedge q \Leftrightarrow q \wedge p$ $p \vee q \Leftrightarrow q \vee p$
5. Asociativa	<ul style="list-style-type: none"> $p \wedge q \wedge r \Leftrightarrow (p \wedge q) \wedge r \Leftrightarrow p \wedge (q \wedge r)$ $p \vee q \vee r \Leftrightarrow (p \vee q) \vee r \Leftrightarrow p \vee (q \vee r)$
6. Distributiva	<ul style="list-style-type: none"> $p \wedge (q \vee r) \Leftrightarrow (p \wedge q) \vee (p \wedge r)$ $p \vee (q \wedge r) \Leftrightarrow (p \vee q) \wedge (p \vee r)$
7. Complementación: <ul style="list-style-type: none"> Contradicción Tercero excluido 	<ul style="list-style-type: none"> $p \wedge \neg p \Leftrightarrow (F)$ $p \vee \neg p \Leftrightarrow (V)$
8. Involución	<ul style="list-style-type: none"> $\neg(\neg p) \Leftrightarrow p$ doble negación $\neg(\neg(\neg p)) \Leftrightarrow \neg p$ triple negación
9. D' Morgan	<ul style="list-style-type: none"> $\neg(p \wedge q) \Leftrightarrow \neg p \vee \neg q$

Tablas de Verdad

Control de calefacción

a = temperatura bajo umbral

b = fin de semana

p	q	$p \text{ or } q$	$\text{not } (p \text{ or } q)$	$p \text{ and } q$
p	q	$(p \vee q)$	$\sim (p \vee q)$	$(p \wedge q)$
V	V	V	F	V
V	F	V	F	F
F	V	V	F	F
F	F	F	V	F

Si quiero que la calefacción se encienda cuando esté por debajo del umbral y solo en fin de semana, Cual es la condición ?

a	b	$a \text{ or } b$	$\overline{a \text{ or } b}$	\bar{a}	\bar{b}	$\bar{a} \text{ and } \bar{b}$
V	V	V	F	F	F	F
V	F	V	F	F	V	F
F	V	V	F	V	F	F
F	F	F	V	V	V	V

Cuando escribimos tablas binarias partimos de valores binarios mínimos a máximos.

p	q	p or q	not(p or q)	p and q
0	0	0	1	0
0	1	1	0	0
1	0	1	0	0
1	1	1	0	1

2^2 2^n

En Python True = 1 y False = 0

Boolean es una extensión de un integer

p	q	r	p or q	p and q
0	0	0	1	0
0	0	1	0	0
0	1	0	0	0
0	1	1	0	0
1	0	0	1	0
1	0	1	1	0
1	1	0	1	0
1	1	1	1	1

2^3

Práctica P02

1. Escribe en tu cuaderno la tabla de verdad de las siguientes condiciones

$$a + b$$

$$(\text{no } a) * b$$

2. Escribe en tu cuaderno la tabla de verdad de p, q, r y las operaciones

$$(p * q) + r$$

$$(p * r) + (q * r)$$

Operadores de asignación

```
In [1]: a = 0
```

```
In [2]: a += 1 # suma en asignación  
a
```

equivale a `a = a + 1`

```
Out[2]: 1
```

`a = 10`

```
In [8]: a -= 10 # resta en asignación  
a
```

equivale a `a = a - 10`

```
Out[8]: 0
```

```
In [9]: a = 5  
a *= 2 # producto en asignación  
a
```

equivale a `a = a * 2`

```
Out[9]: 10
```

```
In [10]: a /= 2 # división en asignación  
a
```

```
Out[10]: 5.0
```

```
In [11]: a %= 2 # módulo en asignación
```

```
In [12]: a
```

```
Out[12]: 1.0
```

```
In [13]: a **= 10
```

```
In [14]: a
```

```
Out[14]: 1.0
```

```
In [ ]: a = 5
```

Práctica : Calcula mentalmente y comprueba los resultados parciales y el resultado final.

$a = 7$

$a += 3$

$a /= 5$

$a **= 4$

$a -= 20$

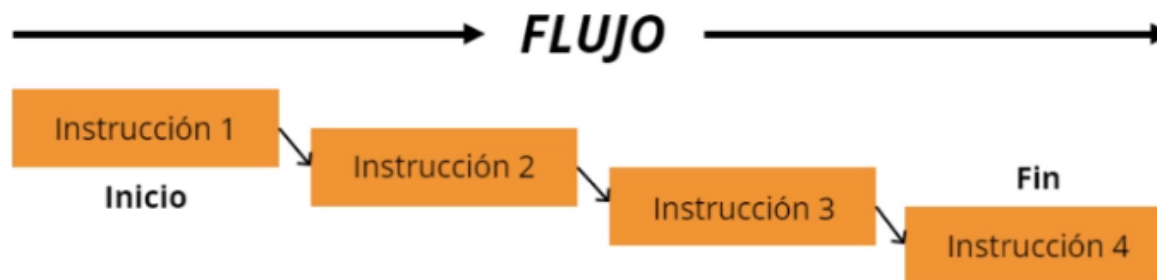
...empezando

Control de Flujo

Estructuras condicionales IF
ELSE

El flujo

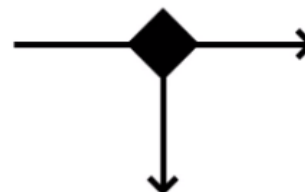
Se entiende así a la sucesión de instrucciones de un programa informático



Sin embargo para manipular datos no es suficiente con realizar cálculos o resolver expresiones necesitamos

Sentencias de control

Condicionales: Para elegir entre distintas posibilidades



Iterativas: Para repetir un bloque de instrucciones



Sentencias condicionales

Sentencia If (Si)

Permite dividir el flujo de un programa en diferentes caminos

Posiblemente sea la más famosa y utilizada en toda la programación porque nos permite condicionar el

```
if True :  
    print ("Se cumple la condición")
```

Se cumple la condición


```
: if not False:  
    print("Se cumple la condición")  
    print("También se muestre este print")
```

Se cumple la condición
También se muestre este print

```
In [5]: a = 5  
        if a == 2:  
            print("a vale 2")  
        if a == 5:  
            print("a vale 5")
```

```
In [6]: a = 5  
        b = 10  
        if a == 5:  
            print("a vale",a)  
            if b == 10:  
                print("y b vale",b)
```

Qué resultado se produce ?

```
if a==5 and b == 10:  
    print("a vale 5 y b vale 10")
```

Qué resultado se produce ?

Sentencia Else (Sino)

Se encadena a un If para indicar el caso contrario
cuando no se cumple la condición

```
n = 10  
if n % 2 == 0:  
    print(n,"es un número par")  
else:  
    print(n,"es un número impar")
```

Qué resultado se produce ?

Sentencia Elif (Sino Si)

Se encadena a un If para comprobar otra posible condición,
siempre que la anterior no se cumpla

```
comando = "SALIR"
if comando == "ENTRAR":
    print("Bienvenido al sistema")
elif comando == "SALUDAR":
    print("Hola, espero que te lo estés pasando bien aprendiendo Python")
elif comando == "SALIR":
    print("Saliendo del sistema...")
else:
    print("Este comando no se reconoce")
```

Práctica P03

```
nota = float(input("Introduce una nota: "))
if nota >= 9:
    print("Sobresaliente")
elif nota >= 7:
    print("Notable")
elif nota >= 6:
    print("Bien")
elif nota >= 5:
    print("Suficiente")
else:
    print("Insuficiente")
```

Introduce una nota:

1. Introduce este programa y comprueba que funciona.
2. Añade : Si la nota es 10 mostrar Matrícula de Honor

```
nota = float(input("Introduce una nota: "))
if nota >= 9:
    print("Sobresaliente")
if nota >= 7 and nota < 9:
    print("Notable")
if nota >= 6 and nota < 7:
    print("Bien")
```

```
if nota >= 5 and nota < 6:
    print("Suficiente")
if nota < 5:
    print("Insuficiente")
```

Introduce una nota: 8
Notable

3. AVANZADO -> Tipo de Calificación:

Antes de pedir la nota pide el **Tipo de Calificación** (1=detallada, 2=breve):

- Si el **Tipo de Calificación** es “detallada”, se aplican las reglas anteriores.
- Si el **Tipo de Calificación** es “breve”, se aplican las siguientes
 - Si la nota es mayor o igual a 9 mostrar “**Excelente**”
 - Si la nota es mayor o igual que 4.8 y menor que 9 mostrar “**Apto**”
 - Si la nota es menor que 4.8 mostrar “**No Apto**”.

4. AVANZADO -> Multi-Idioma:

Partiendo de lo anterior, antes de pedir el Tipo de Calificación, pide el **Idioma** (a elegir entre 3).

Muestra el resultado en el idioma requerido, no vale duplicar el código de condiciones.

Ayuda

En Python existe un tipo de datos especial para resolver este ejercicio, se llama **diccionario**, pero todavía no lo hemos estudiado.

Vamos a resolverlo usando **listas de cadenas**, un recurso que tenemos en casi todos los lenguajes.

Tenemos los mensajes o frases almacenados en los distintos idiomas, respetando las mismas posiciones en todos los idiomas.

Cuando sepamos el idioma copiaremos las frases del idioma elegido sobre una lista general y en adelante usaremos los mensajes de la lista general.

```
#4. AVANZADO i Multidioma
```

```
frases_cat = ["Matrícula d'Honor", "Excel.lent", "Notable", "Bé", "Aprobat", "Suspés", "Apte", "No Apte"]  
frases_cas = ["Matrícula de Honor", "Excelente", "Notable", "Bien", "Aprobado", "Suspenso", "Apto", "No Apto"]  
frases_ang = ["Honor", "Excellent", "Notorious", "Good", "Passed", "Failed", "Has aptitude", "Don't has aptitude"]
```

Solución ejercicio multi-idioma

#4. AVANZADO i Multitidioma

```
frases_cat = ["Matrícula d'Honor", "Excel.lent", "Notable", "Bé", "Aprobat", "Suspés", "Apte", "No Apte"]  
frases_cas = ["Matrícula de Honor", "Excelente", "Notable", "Bien", "Aprobado", "Suspenso", "Apto", "No Apto"]  
frases_ang = ["Honor", "Excellent", "Notorious", "Good", "Passed", "Failed", "Has aptitude", "Don't has aptitude"]
```

Luego pedimos el idioma y copiamos la lista del idioma deseado, sobre una lista general.

```
idioma = int(input("Idioma (0=català, 1=castellà, 2=anglès)"))  
tipus = int(input("Tipus qualificació (1=detallada, 2=breu)"))  
nota = float(input("Entri la qualificació : "))
```

pedimos datos



```
if idioma == 2 :  
    frases = frases_ang  
elif idioma == 1 :  
    frases = frases_cas  
else :  
    frases = frases_cat
```

**copiamos la lista de frases del
idioma a la lista general**



Cuando deseamos usar una frase, la buscamos en la posición de la lista en que se había declarado.

```
if tipus == 1 :
    if nota == 10 :
        print (frases[0])  #"Matrícula d'Honor"
    elif nota >= 9 :
        print (frases[1])  # "Excel.lent"
    elif nota >= 7 :
        print (frases[2])  #"Notable"
    elif nota >= 6 :
        print (frases[3])  #"Bé"
    elif nota >= 5 :
        print (frases[4])  #"Aprobat"
    else :
        print (frases[5])  # "Suspés"
else:
    if nota >= 9 :
        print (frases[1])  # "Excel.lent"
    elif nota >= 4.8 and nota < 9:
        print (frases[6])  #"Apte"
    else :
        print (frases[7])  #"No Apte"
```

Los comentarios nos ayudan a entender el programa, ya que el número de la posición no nos da información sobre el contenido de la frase.

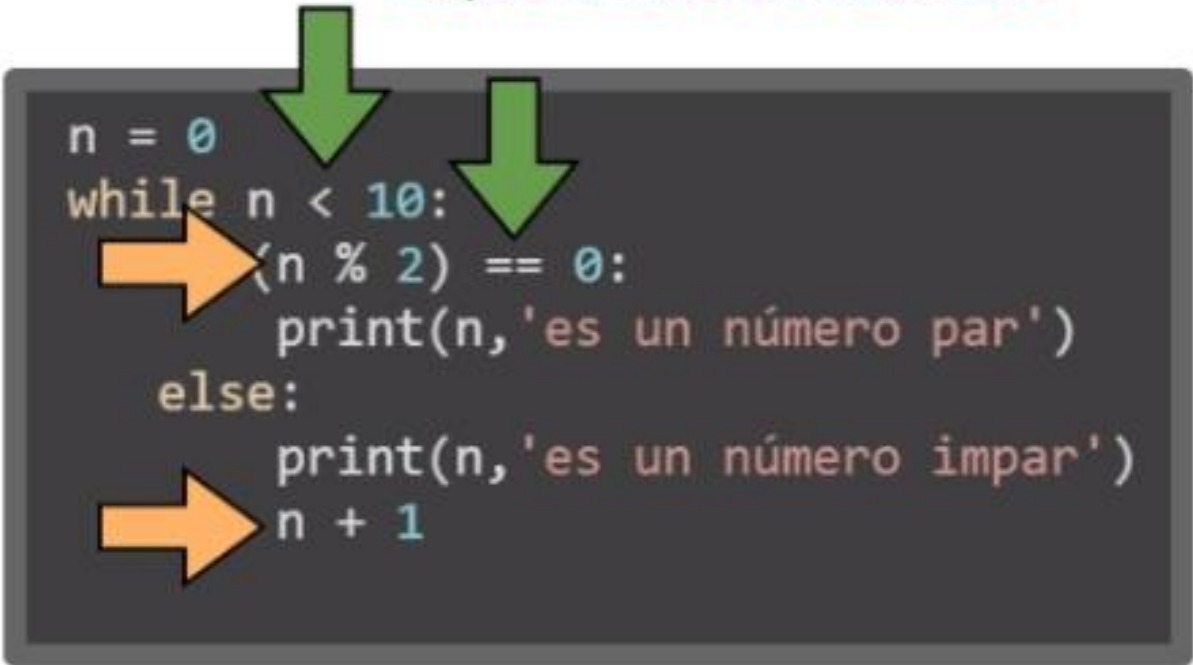
Revisando código

Cuántas expresiones algebraicas hay ?

```
n = 0
while n < 10:
    if (n % 2) == 0:
        print(n, 'es un número par')
    else:
        print(n, 'es un número impar')
    n = n + 1
```

Expresiones relacionales

*Expresiones
Algebraicas*



```
n = 0
while n < 10:
    (n % 2) == 0:
        print(n, 'es un número par')
    else:
        print(n, 'es un número impar')
    n + 1
```

The diagram illustrates the mapping of Python code to algebraic and relational expressions. A dark gray rectangular box contains the Python code. Two green arrows point from the text 'Expresiones relacionales' to the relational expressions ' $n < 10$ ' and ' $(n \% 2) == 0$ ' in the code. Two orange arrows point from the text 'Expresiones Algebraicas' to the algebraic expressions ' $n + 1$ ' and ' $(n \% 2) == 0$ ' in the code.

Práctica P04

1) Realiza un programa que lea 2 números por teclado y determine los siguientes aspectos (es suficiente con mostrar True o False):

- Si los dos números son iguales
- Si los dos números son diferentes
- Si el primero es mayor que el segundo
- Si el segundo es mayor o igual que el primero

2) Pide por pantalla una "clave" de 4 a 7 caracteres de longitud y comprueba :

- Si la clave introducida es menor que 4 muestra :
"clave muy corta, tiene n caracteres. El mínimo es 4"
- Si la clave es mayor que 7 muestra :
"clave demasiado larga excede en n caracteres"
- Si la clave es igual a espacios o clave = "admin" or "gato" muestra -> "clave inválida, elige otra"

3) Realiza un programa que cumpla el siguiente algoritmo utilizando siempre que sea posible operadores de asignación:

- Guarda en una variable `numeroMagico` el valor 12345679 (sin el 8)
- Lee por pantalla otro `numeroUsuario`, especifica que sea entre 1 y 9
- Multiplica el `numeroUsuario` por 9 en sí mismo
- Multiplica el `numeroMagico` por el `numeroUsuario` en sí mismo
- Finalmente muestra el "valor final" del `numeroMagico` por pantalla

P04.1 Comparar dos nombres (números)

```
num1 = int(input ("entri un nombre del 1 a 100: "))
num2 = int(input ("un altre nombre del 1 a 100: "))

print (" els nombres són iguals ", num1 == num2)
print (" els nombres són dierents ", num1 != num2)
print (" el primer és > que el segón ", num1 > num2)
print (" el segón és >= que el primer ", num2 >= num1)
```

P04.2 Solució Entri una clau

```
clave = input("Introdueix la clau : ")
if len(clave) < 4 :
    print ("Error : clau curta.el mínim és 4,")
elif len(clave) > 7 :
    print ("Error : clau massa larga, el màxim es 7")
elif clave in ['admin', 'gato'] :
    print ("Clau invàlida ")
else:
    print ("Clau acceptada")
```