# Introducción a la programación

Profesor: Lic. Sergio Eduardo Torres

Licenciatura en Gestión de Tecnología de la Información

UNPAZ
Universidad Nacional de José C. Paz

# Clase V

## Temario

Estructuras Iterativas
- while
- while anidadas

Contadores
Acumuladores
Banderas

Licenciatura en Gestión de Tecnología de la Información

UNPAZ
Universidad Nacional de José C. Paz

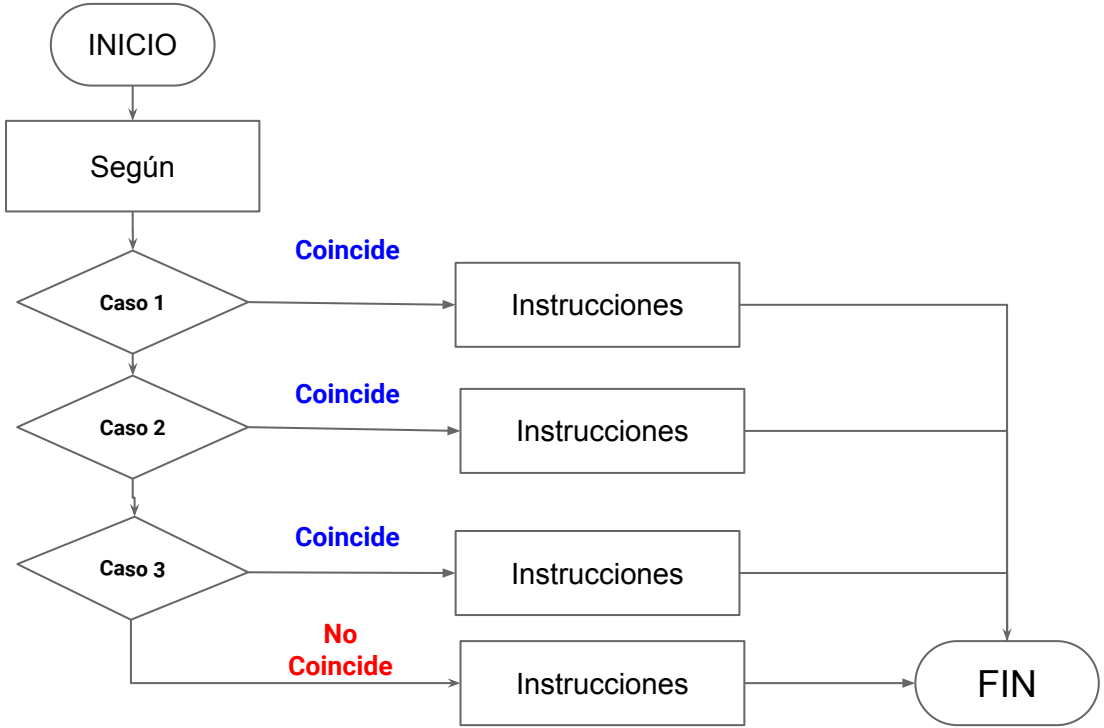
**Estructuras iterativas**

- While
  - True
  - Continue
  - Else
  - Infinito
  - Ctrl + C
- while anidadas

**Contadores**
**Acumuladores**
**Banderas**

# Estructuras de decisión. Condicionales Patrones match case

La estructura match case, permite comparar un valor entre distintas opciones(casos) y si no coincide ninguno se opta por un caso por default.

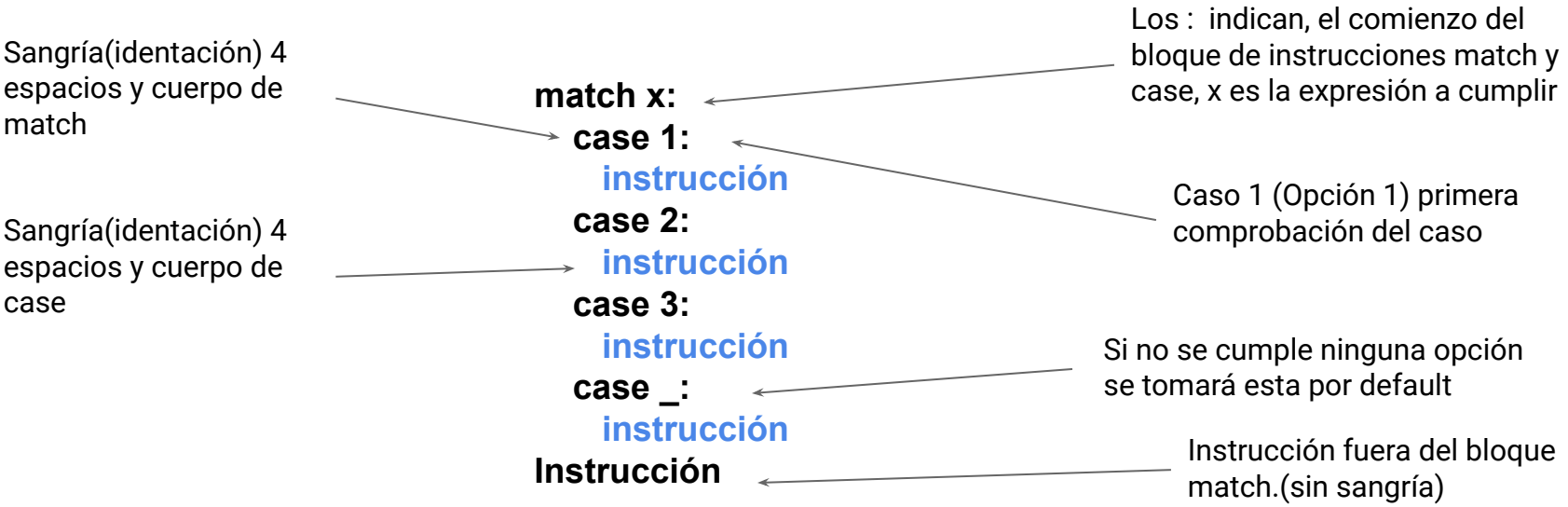



PSEUDOCÓDIGO:

```
INICIO
Según expresión hacer
        Opción (caso 1)
                INSTRUCCIONES
        Opción (caso 2)
                INSTRUCCIONES
        Opción (caso 3)
                INSTRUCCIONES
        SINO
                INSTRUCCIONES
    FIN
```

# Estructuras de decisión. Condicionales Patrones match case

Una declaración de (**match case**) coincidencia compara el valor de una variable dada con diferentes casos(opciones), también conocido como patrón




**Ejercitación**. Escribir un programa que simule un semáforo, el usuario debe seleccionar un color entre Rojo, Amarillo y Verde, y el programa mostrara un mensaje para cada color, si el color no es valido dará aviso.

# Estructuras de decisión. Condicionales Patrones match case

**Ejercitación**. Escribir un programa que simule un semáforo, el usuario debe seleccionar un color entre Rojo, Amarillo y Verde, y el programa mostrara un mensaje para cada color, si el color no es valido dará aviso .

Entrada : Color del semáforo
Proceso : Verificar si es rojo
Verificar si es amarillo
Verificar si es verde
Salida : Mostrar mensaje dependiendo del color
**pseudocódigo**:

**INICIO**
ingreso por teclado una *opción*
**Según opción hacer**
opción R: mostrar “No puede pasar está en rojo”
opción A: mostrar “Atención esta en Amarillo”
opción V: mostrar “Puede pasar está en verde”
opcion _: mostrar “Opción no es válida”
**FIN**

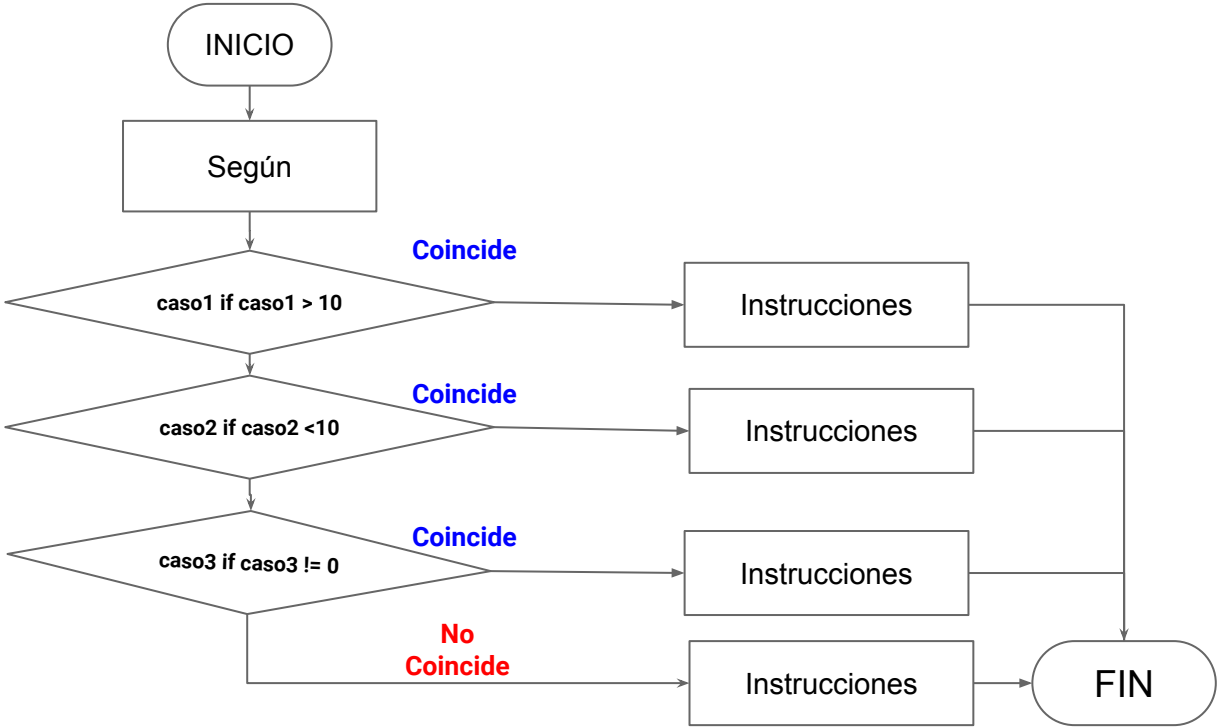
```
semaforo =input('''Ingrese un color del semaforo:
(R)ojo
(A)marillo
(V)erde
Opción: ''').upper()

match semaforo:
    case "R":
        print("No puede pasar esta en ROJO")
    case "A":
        print("Precaución esta en AMARILLO")
    case "V":
        print("Puede pasar esta en Verde")
    case _ :
        print("La opción no es válida")
```

```
Ingrese un color del semaforo:
(R)ojo
(A)marillo
(V)erde
Opción: R
No puede pasar esta en ROJO
```

# Estructuras de decisión. Condicionales Patrones match case

La estructura match case,también puede contener condiciones para eso incorpora un if dentro del caso




PSEUDOCÓDIGO:

INICIO
**Según** expresión **hacer**
    **Opción (caso1 condición)**
        INSTRUCCIONES
    **Opción (caso2 condición)**
        INSTRUCCIONES
    **Opción (caso3 condición)**
        INSTRUCCIONES
    **SINO**
        INSTRUCCIONES
**FIN**

Ejercitación.

Escribir un programa que muestre en qué etapa de la vida se encuentra según la edad ingresada
Fases:

| | |
|---|---|
| pre natal | (embarazo) |
| Infancia | (0 a 6 años de edad) |
| Niñez | (6 a 12 años de edad) |
| Adolescencia | (12 a 20 años de edad) |
| Juventud | (20 a 25 años de edad) |
| Adultez | (25 a 60 años de edad) |
| Ancianidad | (60 años en adelante) |

Ejercitación.

Escribir un programa que muestre en qué etapa de la vida se encuentra según la edad ingresada. Fases: pre natal (embarazo), Infancia (0 a 6 años de edad), Niñez (6 a 12 años de edad), Adolescencia (12 a 20 años de edad), Juventud (20 a 25 años de edad), Adultez (25 a 60 años de edad) Ancianidad (60 años en adelante)

```
print("Etapas de la vida segun la edad")

edad = int(input("Ingrese la edad de la persona: "))

match edad:
    case edad if edad < 0:
        print("Prenatal")
    case edad if edad >=0 and edad <=12:
        print("Niñez")
    case edad if edad >12 and edad <=20:
        print("Adolescencia")
    case edad if edad >20 and edad <=25:
        print("Juventud")
    case edad if edad >25 and edad <=60:
        print("Adultez")
    case edad if edad >60:
        print("Ancianidad ")
```

```
Etapas de la vida segun la edad
Ingrese la edad de la persona: 12
Niñez
```

# Estructuras iterativas.

Las estructuras iterativas, también conocidos como **ciclos, bucles o estructuras de control repetitivas o iterativas.**

Un ciclo o bucle te permite repetir una o varias instrucciones cuantas veces lo necesitemos.

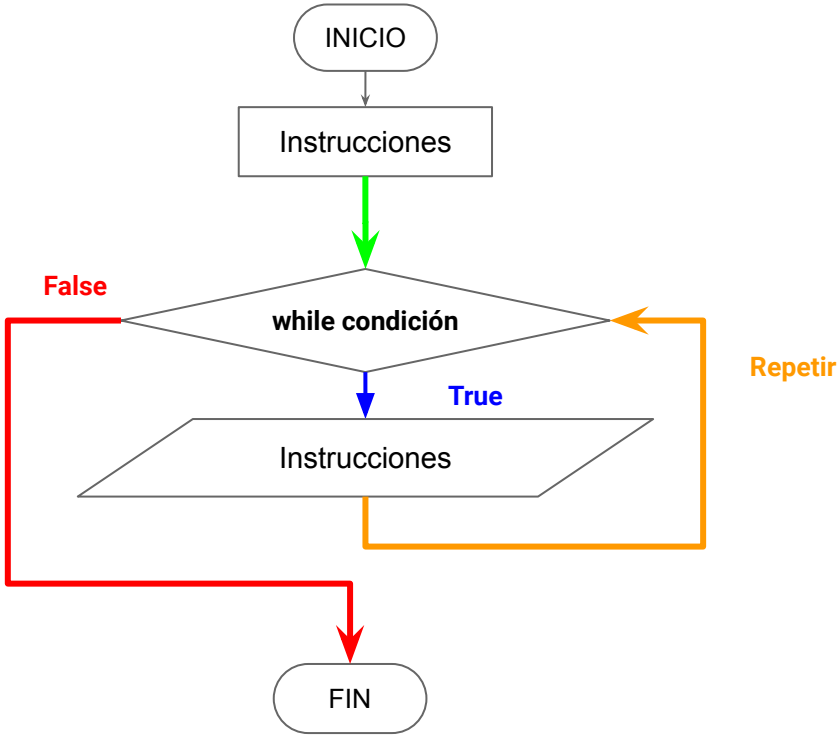
## Tipos de ciclos

**while (Mientras)** **for (Para)**

En Python no existe el ciclo do-while

# Estructuras iterativas. while (mientras)

El ciclo while (mientras) se usa para iterar(repetir) un bloque de código siempre que la condición sea verdadera.
Usamos este ciclo cuando no sabemos la cantidad de veces que necesitamos iterar de antemano.

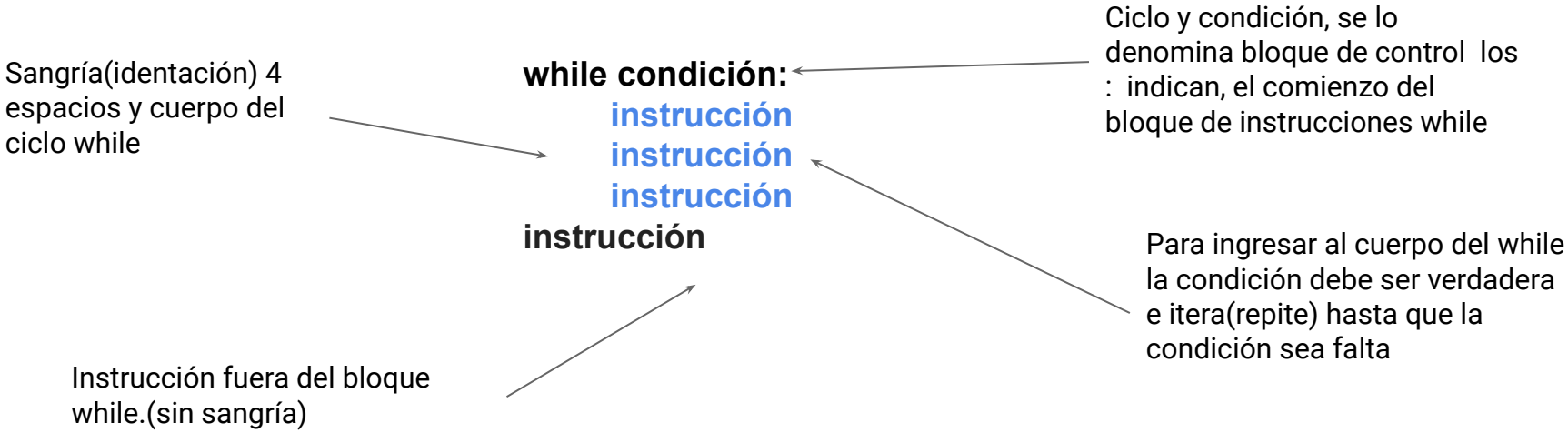



PSEUDOCÓDIGO:

**INICIO**
    INSTRUCCIONES
    **mientras** condición **hacer**
        INSTRUCCIONES
    **Repetir**
**FIN**

# Estructuras iterativas. while (mientras)

En el bucle while, primero se comprueba la condición(bloque de control). Solo ingresa al cuerpo del ciclo si la condición es True (verdadera). Entonces ejecuta las instrucciones dentro del cuerpo del ciclo y vuelve a verificar la condición. Se repite el ciclo hasta que la condición sea false(falsa)




**Ejercitación**. Escribir un programa que solicite ingresar números por teclado y los muestre por pantalla, finalizará cuando presione cero, al finalizar mostrará el mensaje Fin del programa

# Estructuras iterativas. while (mientras)

**Ejercitación**. Escribir un programa que solicite ingresar números por teclado y los muestre por pantalla, finalizará cuando presione cero, al finalizar mostrará el mensaje Fin del programa

```
print("Programa iterativo, ingrese numeros, para terminar ingrese 0 (cero)\n")

numero = int(input("Ingrese un numero: "))
while numero != 0:
    print(f"Numero: {numero}")
    numero = int(input("Ingrese otro numero: "))


print("Fin del programa")
```

```
Ingrese un numero: 2
Numero: 2
Ingrese otro numero: 5
Numero: 5
Ingrese otro numero: 0
Fin del programa
```

# Estructuras iterativas. while (mientras) else, continue.

La estructura iterativa while soporta las siguientes instrucciones:

- **else,** While también posee un bloque cuando la condición no se cumple es opcional , **pero este bloque no es repetitivo.**
- **while true**, El ciclo siempre se ejecutara porque  es una condición infinita siempre será verdadera.
- **break**, La declaración **break,** rompe los ciclos.
- **Continue**:  Detiene el ciclo donde aparece esta instrucción, salta todas las demás que están por debajo y vuelve a iterar.
- Otra forma de romper un ciclo es presionando las teclas **control + c,** aunque  esta última es una salida de emergencia y directamente sale del programa.

.

```
    def decode(self, input, final=False):
KeyboardInterrupt: Execution interrupted
```

# Estructuras iterativas. while (mientras) else

Veamos el primer ejercicio  pero resuelto de otra forma utilizando else

```
print("Programa iterativo, ingrese numeros, para terminar ingrese 0 (cero)\n")

numero = int(input("Ingrese un numero: "))
while numero != 0:
    print(f"Numero: {numero}")
    numero = int(input("Ingrese otro numero: "))

else:
    print(f"Numero: {numero}")

print("Fin del programa")
```

```
Programa iterativo, ingrese numeros, para terminar ingrese 0 (cero)

Ingrese un numero: 1
Numero: 1
Ingrese otro numero: 2
Numero: 2
Ingrese otro numero: 0
Numero: 0
Fin del programa
```

# Estructuras iterativas. while (mientras) True, break.

Veamos el mismo ejercicio pero resuelto de otra forma utilizando True y break

```
print("Programa iterativo, ingrese numeros, para terminar ingrese 0 (cero)\n")


while True:
    numero = int(input("Ingrese un numero: "))
    print(f"Numero: {numero}")
    if (numero == 0):
        break

print("Fin del programa")
```

```
Ingrese un numero: 5
Numero: 5
Ingrese un numero: 8
Numero: 8
Ingrese un numero: 0
Numero: 0
Fin del programa
```

**while true**, es una condición infinita siempre será verdadera.

Se rompe el ciclo solo si en el código existe la declaración **break**

o si presionamos las teclas **control + c,** aunque esta última es una salida de emergencia.

# Estructuras iterativas. while (mientras) continue

Veamos un ejercicio parecido al anterior pero este imprimirá todos los números ingresados que sean positivos pero no los negativos, utilizando continue

```
print("Programa iterativo, ingrese numeros, para terminar ingrese 0 (cero)\n")

while True:
    numero = int(input("Ingrese un numero: "))
    if (numero < 0):
        continue
    print(f"Numero: {numero}")

print("Fin del programa")
```

```
Programa iterativo, ingrese numeros, para terminar ingrese 0 (cero)

Ingrese un numero: 10
Numero: 10
Ingrese un numero: -20
Ingrese un numero: 30
Numero: 30
Ingrese un numero: 0
Numero: 0
```

# Estructuras iterativas. Ejercitación

1. Escribir un programa que almacene en una CONSTANTE una contraseña, luego solicite al usuario ingresar la contraseña, si es correcta mostrará el mensaje la contraseña es correcta y terminará el programa, de lo contrario mostrará el mensaje contraseña incorrecta intente nuevamente, y se solicitara nuevamente la contraseña hasta que la esta sea la correcta.

# Estructuras iterativas. Ejercitación

1. Escribir un programa que almacene en una CONSTANTE una contraseña, luego solicite al usuario ingresar la contraseña, si es correcta mostrará el mensaje la contraseña es correcta y terminará el programa, de lo contrario mostrará el mensaje contraseña incorrecta intente nuevamente, y se solicitara nuevamente la contraseña hasta que la esta sea la correcta.

```
PASS = "M!c1Av3"

while True:
    usu_pass= input("Ingrese la contraseña: ")
    if (PASS == usu_pass):
        print("Contraseña correcta")
        break
    else:
        print("Contraseña incorrecta intente nuevamente")
```

```
Ingrese la contraseña: 123
Contraseña incorrecta intente nuevamente
Ingrese la contraseña: M!c1Av3
Contraseña correcta
```

# Estructuras iterativas. Contador

Un contador es una variable entera que la utilizamos para contar cuando ocurre un suceso.

- Se inicializa a un valor inicial.
  **cont = 0;**

- Se incrementa, cuando ocurre el suceso que estamos contando y se le suma 1.
  **cont = cont + 1;**

  Otra forma de incrementar el contador:
  **cont += 1**

**Ejercicio**: Escriba un programa que permita el ingreso de números por teclado, cada número que ingrese debe tener una posición empezando por el 1, el siguiente el 2.... Cada vez que se ingrese un número mostrará la posición y el valor del número ingresado, pero si ingresa el cero 0 el programa debe terminar.

Ejemplo 1- nro: 325
2- nro: 4
0- Programa terminado

```
cont = 0
while True:
    numeros=int(input("Ingrese un numero: "))
    if numeros == 0:
        print("Programa terminado")
        break

    cont = cont + 1
    print(f"{cont} - numero: {numeros}")
```

```
Ingrese un numero: 6
1 - numero: 6
Ingrese un numero: 8
2 - numero: 8
Ingrese un numero: 0
Programa terminado
```

# Estructuras iterativas. Acumulador

Un acumulador es una variable numérica que permite ir acumulando operaciones. Permite ir haciendo operaciones parciales

Se **inicializa** un valor inicial según la operación que se va a acumular: **a 0 si es una suma, y a 1 si es un producto**.

Se **acumula** un valor intermedio.

acum_venta_total = 0
acum_venta_total = venta + acum_venta_total

**Ejercicio**: Escriba un programa que permita el ingreso de números por teclado, Cada vez que se ingrese un número se sumará al número , y debe mostrar por pantalla el número ingresado y el acumulado, pero si ingresa el cero 0 el programa debe terminar.

Ejemplo 1- ingreso:2 acumulado 2
2- ingreso:5 acumulado 7
0- Programa terminado

```
acumulador = 0
while True:
    numeros=float(input("Ingrese un numero: "))
    if numeros == 0:
        print("Programa terminado")
        break

    acumulador = acumulador + numeros
    print(f"Ingresado:{numeros:.2f} - El acumulado es {acumulador:.2f}")
```

```
Ingrese un numero: 10
Ingresado:10.00 - El acumulado es 10.00
Ingrese un numero: 20
Ingresado:20.00 - El acumulado es 30.00
Ingrese un numero: 5
Ingresado:5.00 - El acumulado es 35.00
Ingrese un numero: 0
Programa terminado
```

.

# Estructuras iterativas. While (mientras)

Ejercitación.

1. Escribir un programa que muestre automáticamente los números del 1 al 10.
2. Escribir un programa que muestre la tabla del 3, sin multiplicar (utilice un acumulador)

# Estructuras iterativas. While (mientras)

Ejercitación.

1. Escribir un programa que muestre automáticamente los números del 1 al 10.
2. Escribir un programa que muestre la tabla del 3, sin multiplicar (utilice un acumulador)

```
cont=1

while cont <=10:
    print(f"Numero:{cont}")
    cont= cont+1
```

```
Numero:1
Numero:2
Numero:3
Numero:4
Numero:5
Numero:6
Numero:7
Numero:8
Numero:9
Numero:10
Finalizado
```

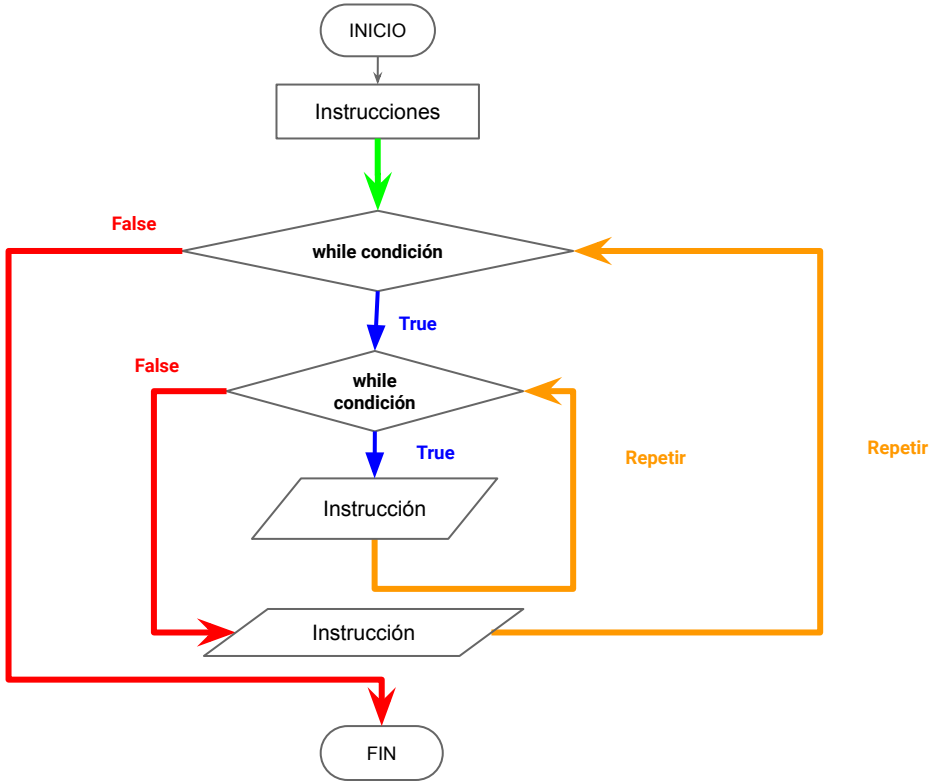
```
cont=1
tabla=3

while cont <=10:
    print(f"3 x {cont} = {tabla}")
    cont = cont + 1
    tabla = tabla + 3
```

```
3 x 1 = 3
3 x 2 = 6
3 x 3 = 9
3 x 4 = 12
3 x 5 = 15
3 x 6 = 18
3 x 7 = 21
3 x 8 = 24
3 x 9 = 27
3 x 10 = 30
```

# Estructuras iterativas. while (mientras) anidadas

El ciclo while (mientras) se puede anidar uno dentro de otro.

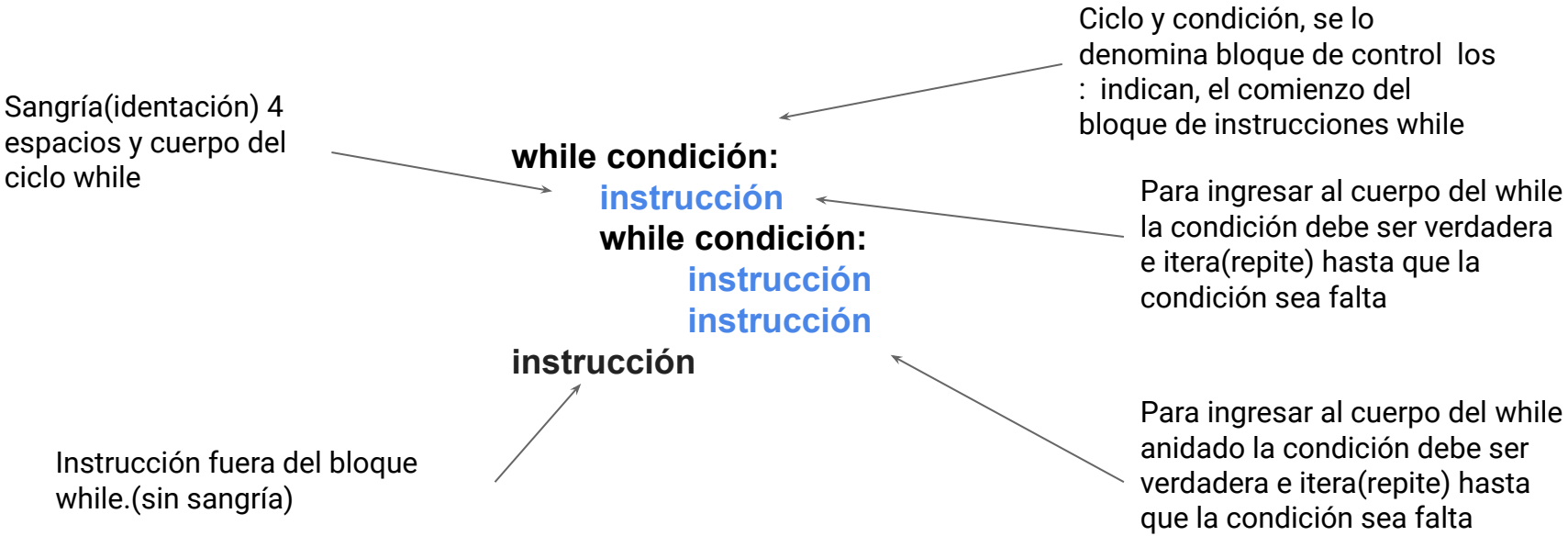



PSEUDOCÓDIGO:

**INICIO**
    Instrucciones
    **mientras** condición True **hacer**
        Instrucciones
        **mientras** condición True **hacer**
            Instrucciones
        **Repetir**
        Instrucciones
    **Repetir**
**FIN**

# Estructuras iterativas. while (mientras) anidadas

En el bucle while anidado encontramos un bucle dentro de otro.




**Ejercitación**. Escribir un programa que permita generar y mostrar todas las combinaciones desde el 0 hasta el 2

# Estructuras iterativas. while (mientras)

**Ejercitación.** Escribir un programa que permita generar y mostrar todas las combinaciones desde el 0  hasta el 2

# Estructuras iterativas. while (mientras)

**Ejercitación.** Escribir un programa que permita generar y mostrar todas las combinaciones desde el 0 hasta el 2

```
i = 0
j = 0

while i < 3:
    while j < 3:
        print(i,j)
        j = j + 1
    i = i + 1
    j = 0
    print("\n")
```

```
0 0
0 1
0 2


1 0
1 1
1 2


2 0
2 1
2 2
```

# Estructuras iterativas. while (mientras) anidadas

**Ejercitación**. Escribir un programa que muestre las tablas del 1 al 3.

# Estructuras iterativas. while (mientras) anidadas

**Ejercitación**. Escribir un programa que muestre las tablas del 1 al 3.

```
i = 1
j = 1
resul = 0

while i <= 3:
    print(f"Tabla de {i}")
    while j < 10:
        resul = i * j
        print(f"{i} x {j} = {resul}")
        j = j + 1

    i = i + 1
    j = 1
    resul = 0
    print("\n")
```

```
Tabla de 1
1 x 1 = 1
1 x 2 = 2
1 x 3 = 3
1 x 4 = 4
1 x 5 = 5
1 x 6 = 6
1 x 7 = 7
1 x 8 = 8
1 x 9 = 9
1 x 10 = 10


Tabla de 2
2 x 1 = 2
2 x 2 = 4
2 x 3 = 6
2 x 4 = 8
```


# Introducción a la programación

## Licenciatura en Gestión de Tecnología de la Información