Universidad Nacional de San Agustín de Arequipa

Escuela Profesional de Ingeniería de Telecomunicaciones

Ingeniero Renzo Bolivar - Docente DAIE

Curso : Computación 1

PRÁCTICA 03

PYTHON - Estructuras Condicionales

Descripción:

En el presente notebook se define **Introducción a Listas** asi como las **Estructuras Condicionales** en el lenguaje de programación Python.

Objetivos:

1. Introducción al uso de **listas**.

1. Estructura Condicional con los comandos `if` , `elif` , `else`

1. Entorno de trabajo en **Jupyter Notebook** y **Spyder**.

1. Practica **Colaborativa** con sincronización en **GitLab**.

Nota: ***NO OLVIDAR INICIAR EL TRABAJO COLABORATIVO:*** en directorio: `Practica3` (con carpetas de Apellidos)

> Primero:
>
> > **JEFE DE PROYECTO:** Inicia la practica **Sincronizando** repositorio local con proyecto GitLab.
> >
> > > (Ver Video: "Trabajo Colaborativo: Inicio Jefe de Proyecto").

> Segundo:
>
> > **COLABORADORES:** Los colaboradores **Clonan** el proyecto de GitLab creado por el **Jefe de Proyecto** a su repositorio local.
> >
> > > (Ver Video: "Trabajo Colaborativo: Inicio Colaboradores").

DESARROLLO

Cada alumnos en su carpeta Personal (APELLIDO) deberán entregar **02 archivos :** 01 archivo **Practica03.ipynb** archivo informe de **Jupyter Notebook** 01 archivo nuevo en el entorno de desarrollo de

Spyder llamado **Practica03.py**

- Ingresamos al entorno de desarrollo **Jupyter Notebook**, abrimos un terminal en Linux:
  - `conda activate Computacion1`
  - `jupyter notebook`

- Ingresamos al entorno de desarrollo **Spyder**, abrimos un terminal en Linux:
  - `conda activate Computacion1`
  - `spyder`

En los archivos deben definir las partes de **Desarrollo** y **Ejemplos** utilizando Markdown con HTML en informe de **Jupyter Notebook** y en el archivo codigo fuente de **Spyder** utilizar Varias Líneas COMENTARIO.

1. Introducción a Listas

Lista: es una colección ordenada de **valores**. Los valores que componen una lista se llaman sus elementos, o sus artículos. Usaremos el término elemento o ítem para que signifique lo mismo. Las listas son ***similares a las cadenas de texto***, que son colecciones ordenadas de caracteres, excepto que los elementos de una lista pueden ser de cualquier tipo.

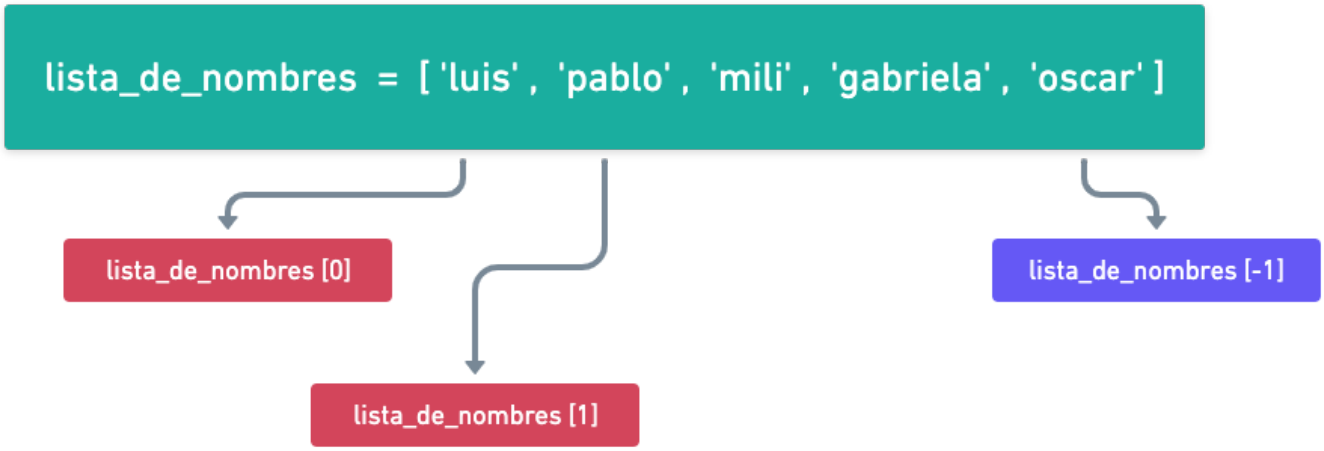
In [ ]:
```
#Cadena de texto o String

#Accediendo al primer elemento de la Cadena de texto
```

Divida una cadena en `una lista` donde cada palabra sea un elemento de la lista:

In [ ]:
```
#método split convierte una cadena a lista
```

Los elemento de la lista pueden ser de `cualquier tipo` :

In [ ]:
```
#lista se crea con "[]" y se separa con","
```



In [ ]:

Modificando elementos en una lista, porque la lista es `mutable` :

In [ ]:

Adicionando elementos a una lista :

In [ ]:

Borrando elementos de una lista :

In [ ]:

Ordenamos una lista :

In [ ]:

In [ ]:

Longitud de una lista :

In [ ]:

2. Estructuras Condicionales

Condiciones Simples :

In [ ]:

In [ ]:

Operador	Significado
==	Igual a
!=	Diferente
<	Menor que
>	Mayor que
<=	Menor igual que
>=	Mayor igual que

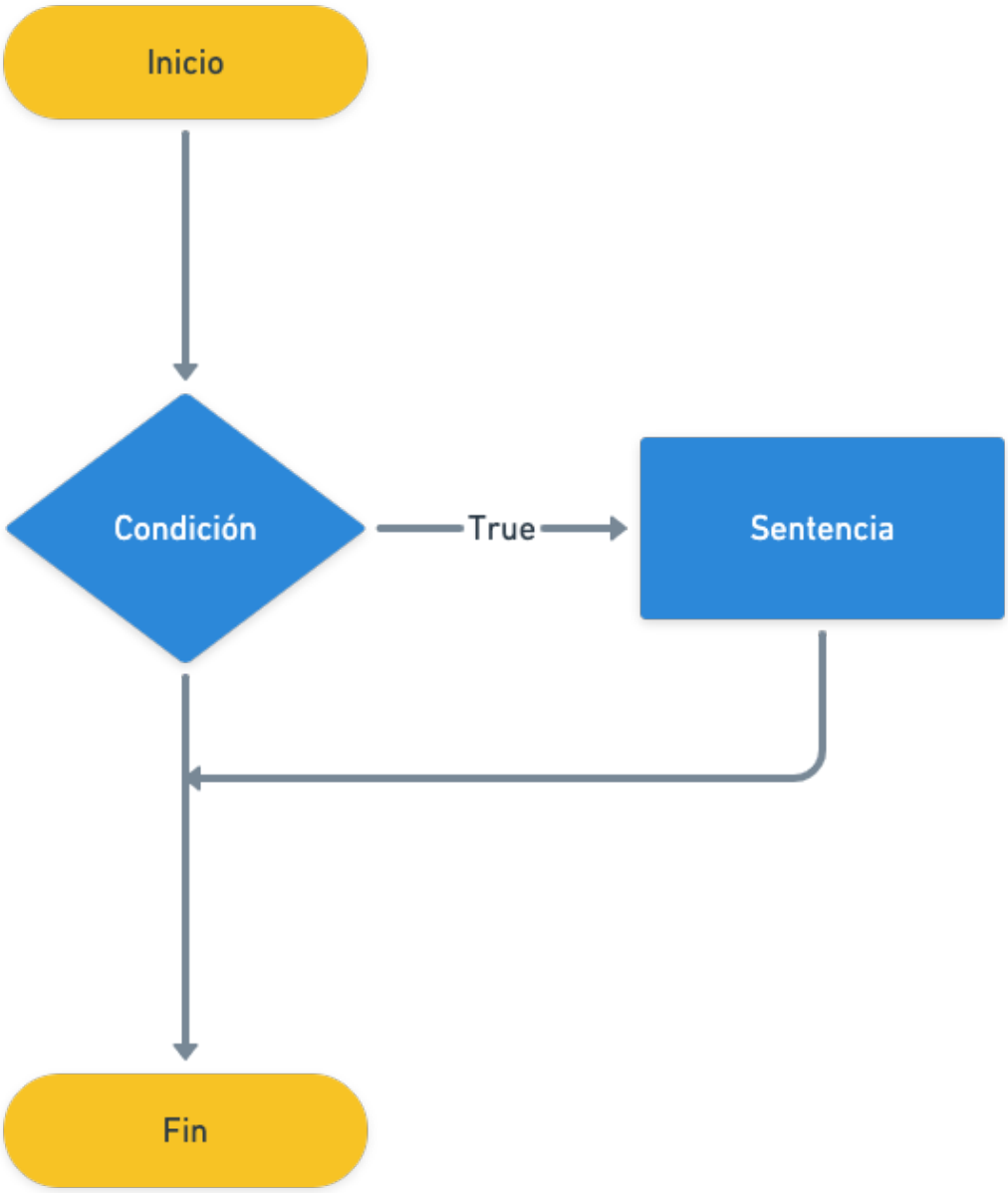
In [ ]:

In [ ]:

Comando `if` - Condicional Simple

Ejecuta *la(s) sentencia(s)* solo `"Si"` la **condición** es verdad.

$$\begin{aligned} &if\ condicion: \\ &\quad Sentencia \\ &\quad ..... \end{aligned}$$

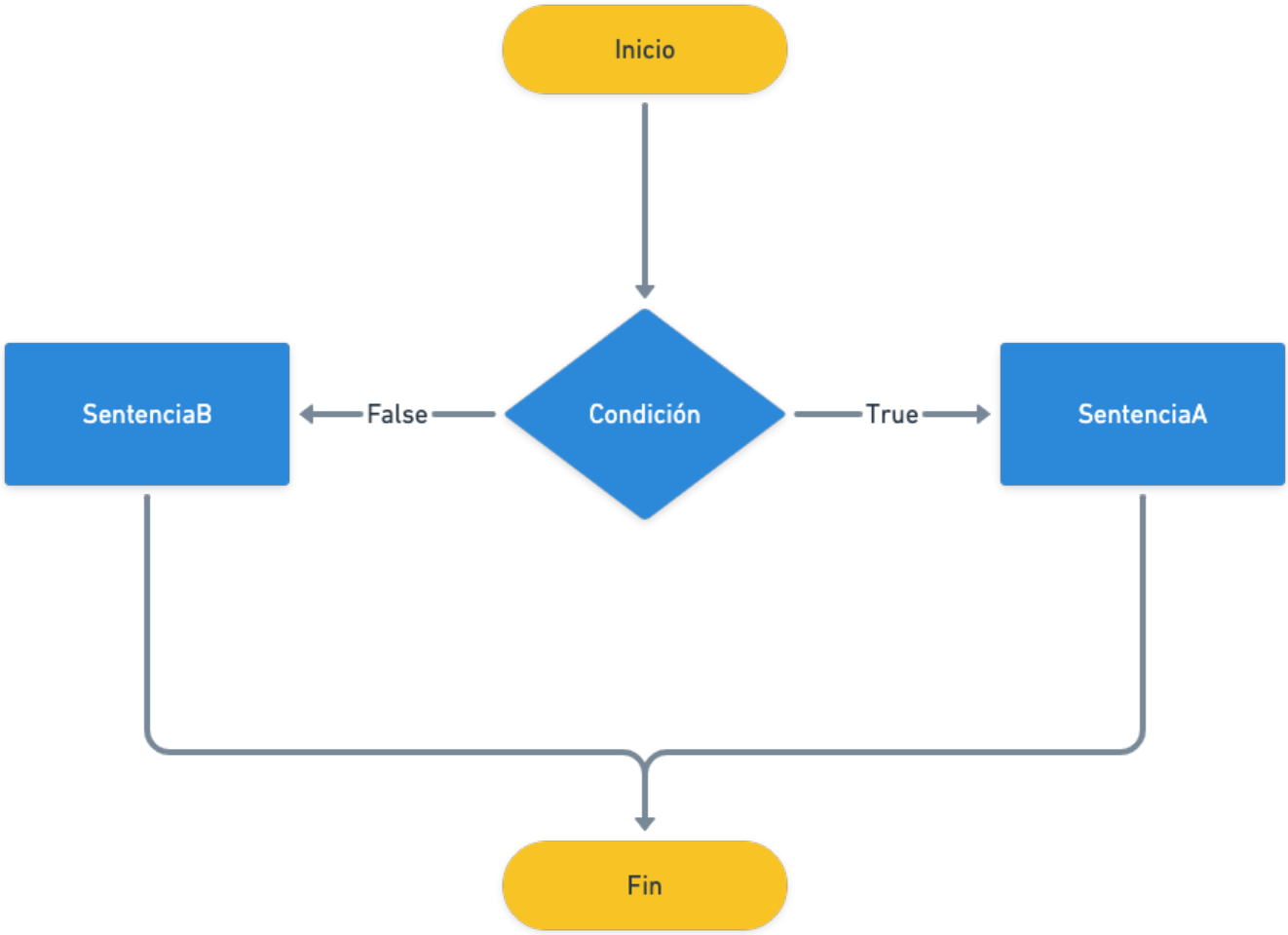


In [ ]:

Comando `if - else` - Condicional

Ejecuta **la(s) sentencia(s)** solo `"Si"` la condición es verdad; `"SiNo"` ejecuta sentencias despues de ***"Else"***.

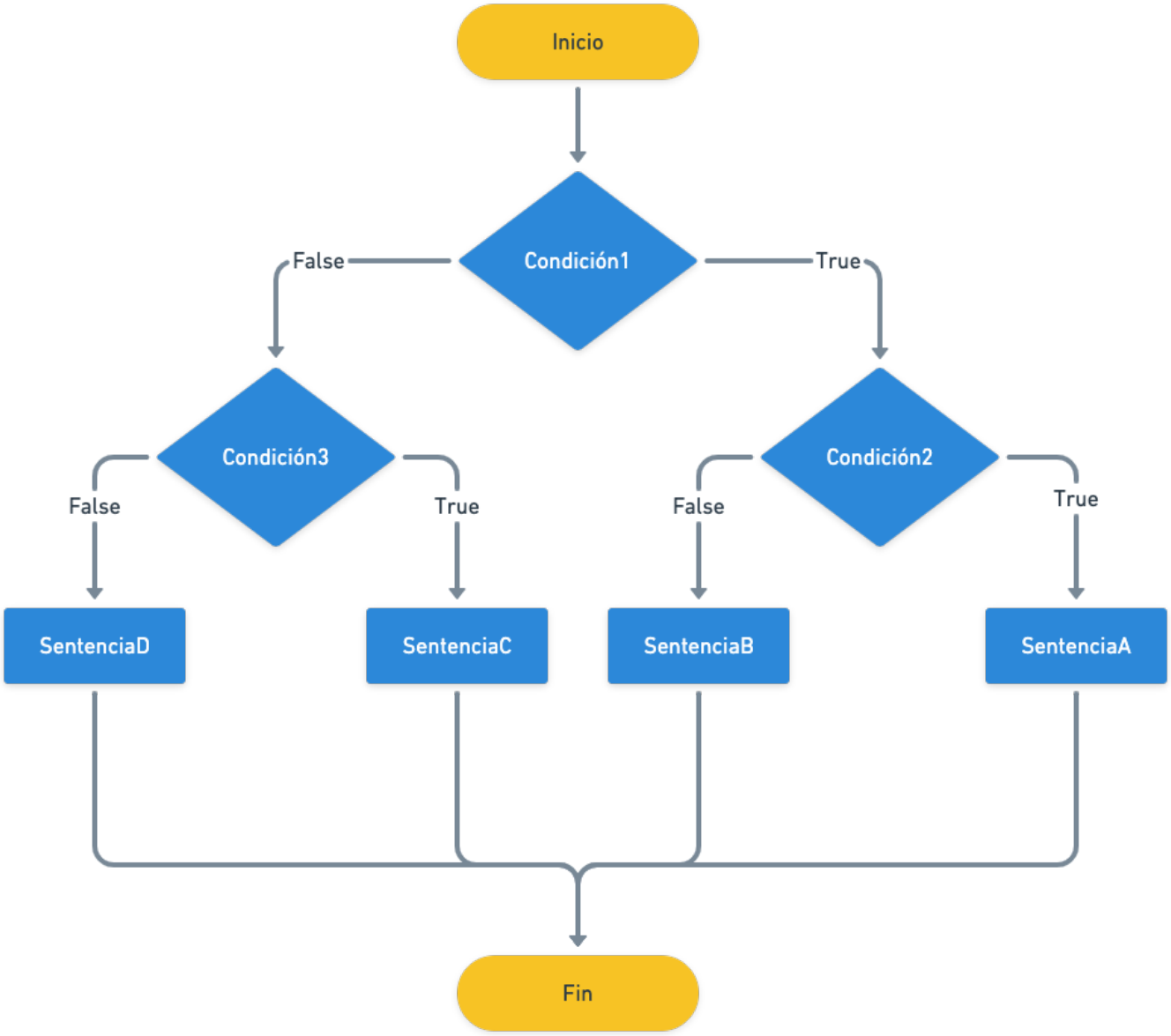
$$
\begin{aligned}
&if\ condicion: \\
&\qquad SentenciaA \\
&\qquad \ldots\ldots\ldots \\
&else: \\
&\qquad SentenciaB \\
&\qquad \ldots\ldots\ldots
\end{aligned}
$$



In [ ]:

Comando `if - else` - Condicional anidado

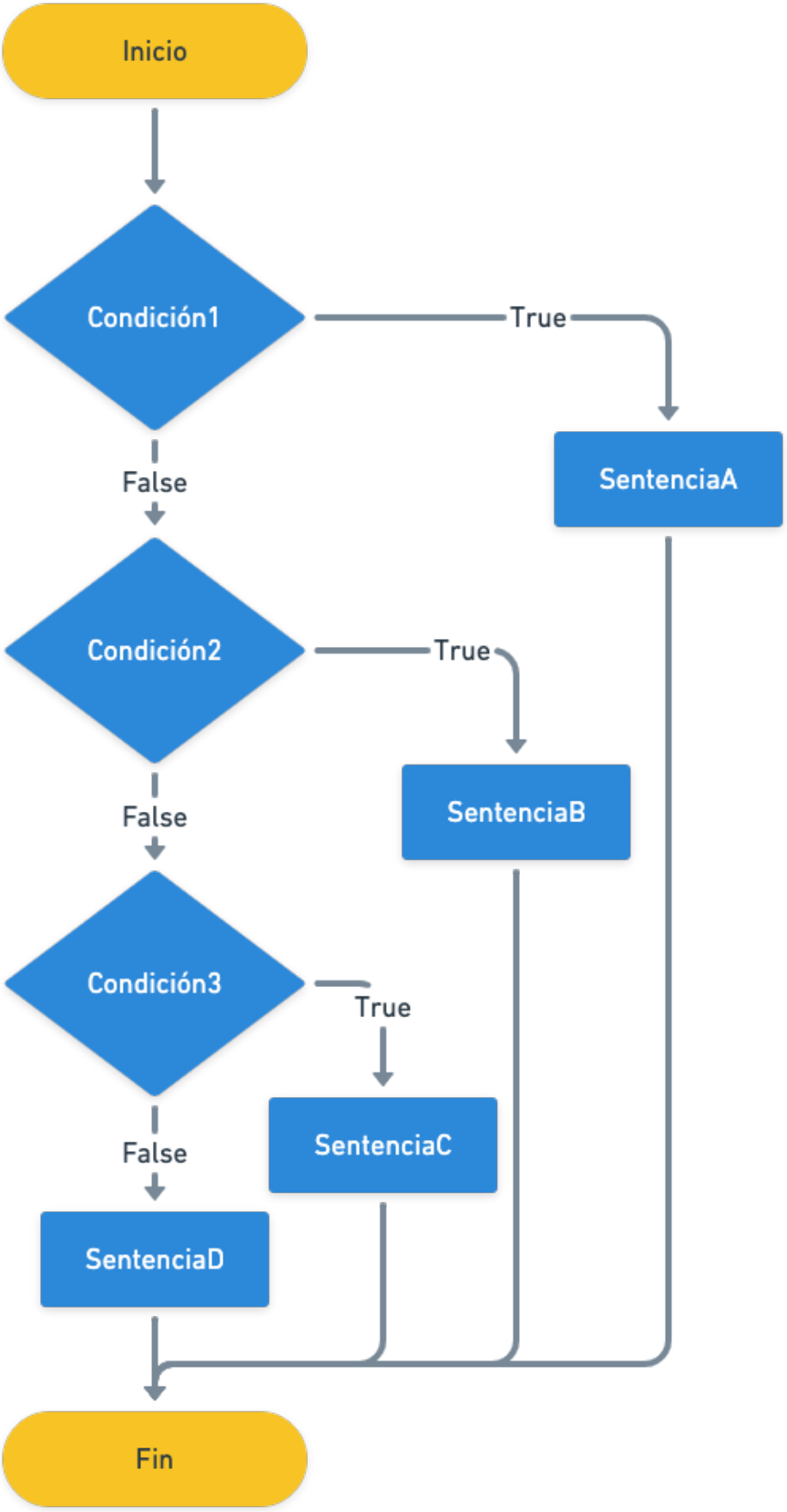
$$
\begin{aligned}
&if\ condicion1: \\
&\quad if condicion2: \\
&\quad\quad SentenciaA \\
&\quad else: \\
&\quad\quad SentenciaB \\
&else: \\
&\quad if condicion3: \\
&\quad\quad SentenciaC \\
&\quad else: \\
&\quad\quad SentenciaD
\end{aligned}
$$



In [ ]:

Comando `if - elif - else` - Condicional encadenado

$$
\begin{aligned}
&if\ condicion1: \\
&\quad SentenciaA \\
&\quad \ldots\ldots\ldots \\
&elif\ condicion2: \\
&\quad SentenciaB \\
&\quad \ldots\ldots\ldots \\
&elif\ condicion3: \\
&\quad SentenciaC \\
&\quad \ldots\ldots\ldots \\
&else: \\
&\quad SentenciaD \\
&\quad \ldots\ldots\ldots
\end{aligned}
$$


Inicio
Condición1
True
SentenciaA
False
Condición2
True
SentenciaB
False
Condición3
True
SentenciaC
False
SentenciaD
Fin

In [ ]:

EJEMPLOS

Deberan entregar **02 archivos :** 01 archivo digitado en **Practica03.ipynb** (Actual) y 01 archivo nuevo digitado en el entorno de desarrollo de **Spyder** llamado **Practica03.py**

Listas

Operadores con varias listas :

In [1]:
```
lista1 = [10,20,30]
lista2 = [40,50,60]

print(lista1 + lista2)
```

```
[10, 20, 30, 40, 50, 60]
```

Mejorando la impresión de varias listas :

In [2]:
```
listaA = ['audi','toyota','mazda']
puerta = [2,4]

print("Mi auto es: " + listaA[-1].title())
print(f"Mi auto {listaA[-2].title()} tiene {puerta[-2]} puertas")    # elementos de lista
```

```
Mi auto es: Mazda
Mi auto Toyota tiene 2 puertas
```

Opreadores matemáticos de una lista :

In [3]:
```
lista3 = [100,500,200,7000,-200,]

print(sum(lista3))                  #Sumando cada elemento de una lista con la función "su
print(max(lista3))                  #Máximos de los elemento de una lista con la función
print(min(lista3))                  #Mínimo de los elemento de una lista con la función "m
```

```
7600
7000
-200
```

In [4]:
```
lista1 = list(range(7))                     #lista de 0 a 6 NO CONSIDERA 7
lista2 = list(range(1,10))                  #lista de de 1 a 9 NO CONSIDERA 10
lista3 = list(range(-10,2))                 #lista de -10 a 1 NO CONSIDERA 2
lista4 = list(range(200,100,-20))            #lista de 200 a 100 restando 20
lista5 = list(range(200,100,20))            #lista vacia rango inicia en 200 avanza sumando

print(lista1)
print(lista2)
print(lista3)
print(lista4)
print(lista5)
```

```
[0, 1, 2, 3, 4, 5, 6]
[1, 2, 3, 4, 5, 6, 7, 8, 9]
[-10, -9, -8, -7, -6, -5, -4, -3, -2, -1, 0, 1]
[200, 180, 160, 140, 120]
[]
```

Error comun en una lista :

In [5]:
```
autos = ['mazda', 'audi', 'toyota', 'subaru', 'bmw', 'lamborghini']
```

```
print(autos[6].upper())
```

```
---------------------------------------------------------------------------
IndexError                                Traceback (most recent call last)
/var/folders/lx/t0_sy3b51q5c3890h91zsr8r0000gn/T/ipykernel_11357/2685536732.py in <module>
      1 autos = ['mazda', 'audi', 'toyota', 'subaru', 'bmw', 'lamborghini']
      2
----> 3 print(autos[6].upper())

IndexError: list index out of range
```

list index out of range==>> El indice esta fuera del rango Vamos a CORREGIR indicando un

$\in dicec \text{ or } rec \rightarrow$

In [6]:

```
autos = ['mazda', 'audi', 'toyota', 'subaru', 'bmw', 'lamborghini']

print(autos[4].upper())
```

```
BMW
```

Condicionales

Hacer un programa que determine si un numero ingresado por pantalla es positivo o negativo

In [7]:

```
#METODO 1 : Utilizando if Anidado
x = int(input("Ingrese un numero entero, positivo o negativo : "))
if x > 0:
    y = "positivo"
else:
    if x < 0:
        y = "negativo"
    else :
        if x == 0:
            y = "cero"
print("\nEl numero : {}".format(x))
print("Es un numero : {}".format(y))
```

```
Ingrese un numero entero, positivo o negativo : -8

El numero : -8
Es un numero : negativo
```

In [8]:

```
#METODO 2 : Utilizando  elif

#****************** FUNCIÓN detecta POSITIVOS NEGATIVOS ***************
def det(xx):
    if xx < 0:
        p = 'negativo'
    elif xx > 0:
        p = 'positivo'
    else:
        p = 'cero'
        print(p)
    return p

###Ingresamos NUMERO ************************************************
num = int(input("Ingrese un numero entero, positivo o negativo : "))    #Probar varias vece
###LAMAMOS A LA FUNCION *********************************************
```

```
#FUNCION devuelve un resultado capturado en variable "resp"
resp = det(num)

#Imprimo RESULTADO
print("\nEl numero : {}".format(num))
print("Es un numero : {}".format(resp))
```

```
Ingrese un numero entero, positivo o negativo : 76

El numero : 76
Es un numero : positivo
```

Hacer un programa que pueda ingresar radio del Circulo y calcule: Perímetro, Área y Diámetro

In [9]:

```
#METODO 1 : Utilizando if Anidado
from math import pi

radio = float(input("Ingrese el radio de un círculo: "))

#Menú
print("Seleccione una opción: ")
print("a) Calcular el diámetro.")
print("b) Calcular el perímetro.")
print("c) Calcular el área.")
opcion = input("Digita a, b, o c y presiona enter:  ")

if opcion == "a":
    diametro = 2 * radio
    print("\nEl diámetro del Círculo es {0}.".format(diametro))
else:
    if opcion == "b":
        perimetro = 2 * pi * radio
        print("\nEl perímetro del Círculo es {0}.".format(perimetro))
    else:
        if opcion == "c":
            area = pi * radio ** 2
            print("\nEl área del Círculo es {0}.".format(area))
        else:
            print("\nSolo hay tres opciones: a, b, c.")
            print('Tu has tecleado "{0}".'.format(opcion))
```

```
Ingrese el radio de un círculo: 23
Seleccione una opción:
a) Calcular el diámetro.
b) Calcular el perímetro.
c) Calcular el área.
Digita a, b, o c y presiona enter:  c

El área del Círculo es 1661.9025137490005.
```

In [10]:

```
#METODO 2 : Utilizando  elif
from math import pi

radio = float(input("Ingrese el radio de un círculo: "))

#Menú
print("Seleccione una opción: ")
print("a) Calcular el diámetro.")
print("b) Calcular el perímetro.")
print("c) Calcular el área.")
opcion = input("Digita a, b, o c y presiona enter:  ")
```

```
if opcion == "a":
    diametro = 2 * radio
    print("El diámetro del Círculo es {0}.".format(diametro))
elif opcion == "b":
    perimetro = 2 * pi * radio
    print("El perímetro del Círculo es {0}.".format(perimetro))
elif opcion == "c":
    area = pi * radio ** 2
    print("El área del Círculo es {0}.".format(area))
else:
    print("Solo hay tres opciones: a, b, c.")
    print('Tu has tecleado "{0}".'.format(opcion))
```

```
Ingrese el radio de un círculo: 34
Seleccione una opción:
a) Calcular el diámetro.
b) Calcular el perímetro.
c) Calcular el área.
Digita a, b, o c y presiona enter:  a
El diámetro del Círculo es 68.0.
```

Hacer un programa que verifique si se ha comprado `carne` en la lista del mercado para hacer Parrillada

In [11]:

```
# Lista de Alimentos
alimento = ['arroz', 'papas', 'camotes', 'leche', 'carne', 'cebolla', 'tomate', 'lechuga']

if 'carne' in alimento:                    #Condicional Si "carne" esta en la lista de ali
    print('Hoy comemos rica Parrilla!!!!')
else:
    print('Hoy comemos Verduras')
```

```
Hoy comemos rica Parrilla!!!!
```

Hacer un programa que verifique en una jugeria si tienen la fruta que deseas y te prepare un jugo

In [12]:

```
# ******* FUNCION JUGERÍA ***************************************************
def preparar(fruit):
    if fruit in fruta:
        print(f'\nSi tenemos {antojo.upper()} !!!, te prepararemos un Jugo')
    else:
        print(f'\nHoy no tenemos {antojo.upper()}, pero regresa Mañana')


# ******* Lista de Frutas ***************************************************
fruta = ['platano', 'manzana', 'pera', 'naranja', 'mandarina', 'uva']
antojo = input('Ingresa la fruta que deseas comer hoy : ')

preparar(antojo)          #Probar varias veces con futas que esten y no esten en la lista

print('\nGracias por tu Preferencia !!!, Vuelve Pronto')
```

```
Ingresa la fruta que deseas comer hoy : pera

Si tenemos PERA !!!, te prepararemos un Jugo

Gracias por tu Preferencia !!!, Vuelve Pronto
```

Hacer un programa que verifique mayoria de edad

In [13]:
```
#************** FUNCIÓN que detecte la mayoria de edad ************
def mayoria(nom,xx):
    if xx >= 18:
        print(f'\n{nom.upper()} es Mayor de Edad con {xx} años de edad')
    else:
        print(f'\n{nom.upper()} es Menor de Edad con {xx} años de edad')

#************** INGRESA DATOS *********************************
nombre = input('Ingresa tu Nombre : ')
edad = int(input('Ingrese su edad : '))

mayoria(nombre,edad)          #Probar varias veces con mayores y menores de edad, hombres y
```

```
Ingresa tu Nombre : Juanito
Ingrese su edad : 12

JUANITO es Menor de Edad con 12 años de edad
```

EJERCICIOS

El Grupo deberá ejecutar **Los 07 Ejercicios** en la carpeta **EJERCICIOS**

Deberan entregar **02 archivos :** 01 archivo digitado en **Ejercicios03.ipynb** archivo informe de **Jupyter Notebook** 01 archivo nuevo digitado en el entorno de desarrollo de **Spyder** llamado **Ejercicios03.py**

1.- Hacer un programa que verifique la lista existente de compras de una Jugería, si se compro:

- "manzanas" o "peras" y si es Lunes, debera indicar que se puede desayunar jugo de dichas frutas.
- "platano" o "fresa" y "leche" y siendo Martes, debera indicar que se puede desayunar jugo de leche de dichas frutas.
- "yogurt" y "cereales" y siendo Miercoles, debera indicar que se tomara dicho desayuno.
- Si es del Jueves al Domingo verificar si hay papaya, debera indicar que se puede desayuinar dicha fruta. ### Utilizar `listas y funciones`

2.- Hacer un programa que ingrese por teclado 20 numeros y luego visualizar un `menu` de opciones para hallar los valores estadisticos: media, mediana y frecuencia. Utilizar `listas y funciones` .

3.- Hacer un programa que ingrese por teclado 10 objetos de casa, luego visualizar un `menu` de opciones para ordenar la lista e imprimir de forma: 1) ordenar de forma ascendente 2) ordenar de forma descendente. Utilizar `listas y funciones` .

4.- Hacer un programa que ingrese por teclado una contraseña con un nombre, si el nombre y contraseña ingresada se encuentra en la lista de contraseñas y en la lista de nombres, Les da la bienvenida el sistema y te reconoce como `Trabajador de la empresa MicroSecret` . Si la contraseña o nombre es incorrecta te rechaza el acceso. Utilizar `listas y funciones` .

5.- Hacer un programa que verifique una lista de almacen de una Carpintería:

- Si tiene clavos y madera se puede construir una ventana.
- Si tiene plancha metal y soldadura se puede construir puerta metalica.
- Si tiene plancha de madera y madera se puede construir una puerta.
- Si no tiene ninguna de las anteriores, solicitar compra de materiales. ### Utilizar `listas y funciones` .

6.- Hacer un Programa de selección para jovenes postulantes a la Marina de Guerra del Perú, verificando el ingreso de los siguientes datos:

- Deben ser Mujeres mayores de edad, con altura mínima de 1,60 metros, con peso entre 55 y 60 kilos y tenga en la prueba mayor a 65 puntos.
- Deben ser Varones mayores de edad, con altura mínima de 1,65 metros, con peso entre 55 y 60 kilos y tenga en la prueba mayor a 65 puntos.
- Si no cumplen debe salir un mensaje de "NO SELECCIONADO".

Utilizar `listas y funciones` .

7.- Hacer un Programa de verificación de datos para obtención Tarjeta Única de Circulación (TUC) electrónica:

- Deben ser Mujeres/Hombres mayores de edad, aprobar examen médico, prueba de conocimiento mayor a 80 puntos, prueba de manejo mayor a 95 puntos, verificar pago de 10 soles, debe imprimir un mensaje de "APTO TUC".
- Si no cumplen cualquiera de las opciones debe imprimir un mensaje de "NO APTO TUC".

Utilizar `listas y funciones` .

SINCRONIZAR CON GITLAB

- Ingresamos a la linea de comando.
- En nuestro directorio de `Practica3` .
- Añadimos los ULTIMOS cambios en nuestro repositorio local **git**:

```
(Programacion1) renzo@misti:~$ git add -A
```

```
(Programacion1) renzo@misti:~$ git commit -m "Finalizando Ejercicios de la Practica 03"
```

- Actualizamos nuestros repositorio local con los cambios del respositorio de **GitLab**:

```
(Programacion1) renzo@misti:~$ git pull origin master
```

- Enviamos nuestros cambios al repositorio remoto de **GitLab**:

```
(Programacion1) renzo@misti:~$ git push origin master
```

- Para tener ***evidencia del trabajo realizado*** envía un solo integrante al Aula Virtual, la carpeta con todo el proyecto en **zip**.

- Finaliza la `Práctica 03`

BIBLIOGRAFÍA

[1] Tutorial de Python https://docs.python.org/es/3/tutorial/index.html

[2] Curso Python en Kaggle https://www.kaggle.com/colinmorris/hello-python

[3] How to Think Like a Computer Scientist http://openbookproject.net/thinkcs/python/english3e/

[4] Automate the Boring Stuff with Python https://automatetheboringstuff.com/