Manual python clase por clase

## Clase 1: Introducción a Python

### Objetivos de la clase:

- Entender los conceptos básicos de Python.

- Aprender a instalar y configurar Python.

- Familiarizarse con el entorno de desarrollo.

- Escribir y ejecutar tu primer programa en Python.

### 1. Instalación y Configuración

#### Instalación de Python:

1. \*\*Descargar Python:\*\*

- Ve a la [página oficial de descargas de Python](https://www.python.org/downloads/).

- Descarga la última versión de Python para tu sistema operativo.

2. \*\*Instalar Python:\*\*

- Abre el archivo descargado y sigue las instrucciones de instalación.

- Asegúrate de marcar la opción "Add Python to PATH" durante la instalación.

#### Instalación de un Entorno de Desarrollo:

- \*\*Visual Studio Code (VS Code):\*\*

- Descarga e instala VS Code desde [su página oficial](https://code.visualstudio.com/).

- Instala la extensión de Python en VS Code.

### 2. Escribiendo tu Primer Programa

#### Hola Mundo:

1. Abre VS Code y crea un nuevo archivo con extensión `.py` (por ejemplo, `hola\_mundo.py`).

2. Escribe el siguiente código en el archivo:

```python

print("Hola, Mundo!")

```

3. Guarda el archivo y ejecútalo:

- Abre la terminal en VS Code (puedes usar el atajo `Ctrl + ~`).

- Navega hasta el directorio donde guardaste el archivo.

- Ejecuta el archivo escribiendo `python hola\_mundo.py` en la terminal.

### 3. Conceptos Básicos de Python

#### Variables y Tipos de Datos:

```python

# Variables

nombre = "Juan"

edad = 25

altura = 1.75

es\_estudiante = True

# Imprimir valores

print(nombre)

print(edad)

print(altura)

print(es\_estudiante)

```

#### Operadores:

```python

# Operadores aritméticos

suma = 10 + 5

resta = 10 - 5

multiplicacion = 10 \* 5

division = 10 / 5

# Operadores de comparación

es\_igual = 10 == 5

es\_diferente = 10 != 5

es\_mayor = 10 > 5

es\_menor = 10 < 5

```

#### Estructuras de Control:

```python

# Condicionales

if edad > 18:

print("Eres mayor de edad.")

else:

print("Eres menor de edad.")

# Bucles

# Bucle for

for i in range(5):

print(i)

# Bucle while

contador = 0

while contador < 5:

print(contador)

contador += 1

```

#### Funciones:

```python

# Definir una función

def saludar(nombre):

return f"Hola, {nombre}!"

# Llamar a la función

mensaje = saludar("Juan")

print(mensaje)

```

### Tarea

#### Parte 1:

1. Instala Python y VS Code en tu computadora.

2. Crea un archivo llamado `primer\_programa.py`.

3. Escribe un programa que:

- Solicite al usuario su nombre, edad y altura.

- Imprima un mensaje de saludo que incluya su nombre.

- Calcule cuántos años tendrá el usuario en 10 años y lo imprima.

- Calcule la altura del usuario en centímetros y la imprima.

#### Parte 2:

1. Crea un archivo llamado `calculadora.py`.

2. Escribe un programa que:

- Solicite al usuario dos números.

- Imprima la suma, resta, multiplicación y división de esos números.

### Ejemplo de solución para `primer\_programa.py`:

```python

# Solicitar información al usuario

nombre = input("¿Cuál es tu nombre? ")

edad = int(input("¿Cuántos años tienes? "))

altura = float(input("¿Cuál es tu altura en metros? "))

# Saludo

print(f"Hola, {nombre}!")

# Calcular edad en 10 años

edad\_futura = edad + 10

print(f"En 10 años tendrás {edad\_futura} años.")

# Calcular altura en centímetros

altura\_cm = altura \* 100

print(f"Tu altura en centímetros es {altura\_cm} cm.")

```

### Ejemplo de solución para `calculadora.py`:

```python

# Solicitar números al usuario

num1 = float(input("Ingresa el primer número: "))

num2 = float(input("Ingresa el segundo número: "))

# Realizar operaciones

suma = num1 + num2

resta = num1 - num2

multiplicacion = num1 \* num2

division = num1 / num2

# Imprimir resultados

print(f"Suma: {suma}")

print(f"Resta: {resta}")

print(f"Multiplicación: {multiplicacion}")

print(f"División: {division}")

Clase 2: Estructuras de Datos en Python

### Objetivos de la clase:

- Aprender sobre las principales estructuras de datos en Python.

- Entender cómo manipular listas, tuplas, conjuntos y diccionarios.

- Escribir programas prácticos utilizando estas estructuras de datos.

### 1. Listas

Las listas son colecciones ordenadas y mutables que permiten almacenar múltiples elementos.

#### Crear y Acceder a Listas:

```python

# Crear una lista

frutas = ["manzana", "banana", "cereza"]

# Acceder a elementos

print(frutas[0]) # manzana

print(frutas[1]) # banana

# Modificar elementos

frutas[0] = "kiwi"

print(frutas) # ['kiwi', 'banana', 'cereza']

```

#### Métodos de Listas:

```python

# Añadir elementos

frutas.append("naranja")

print(frutas) # ['kiwi', 'banana', 'cereza', 'naranja']

# Eliminar elementos

frutas.remove("banana")

print(frutas) # ['kiwi', 'cereza', 'naranja']

# Longitud de la lista

print(len(frutas)) # 3

# Recorrer una lista

for fruta in frutas:

print(fruta)

```

### 2. Tuplas

Las tuplas son colecciones ordenadas e inmutables.

#### Crear y Acceder a Tuplas:

```python

# Crear una tupla

punto = (10, 20)

# Acceder a elementos

print(punto[0]) # 10

print(punto[1]) # 20

# No se pueden modificar los elementos de una tupla

# punto[0] = 30 # Esto causaría un error

```

### 3. Conjuntos

Los conjuntos son colecciones desordenadas de elementos únicos.

#### Crear y Operar con Conjuntos:

```python

# Crear un conjunto

numeros = {1, 2, 3, 4, 5}

# Añadir y eliminar elementos

numeros.add(6)

numeros.remove(2)

print(numeros) # {1, 3, 4, 5, 6}

# Operaciones de conjuntos

pares = {2, 4, 6, 8}

impares = {1, 3, 5, 7}

union = pares.union(impares)

interseccion = pares.intersection(impares)

diferencia = pares.difference(impares)

print(union) # {1, 2, 3, 4, 5, 6, 7, 8}

print(interseccion) # set()

print(diferencia) # {8, 2, 4, 6}

```

### 4. Diccionarios

Los diccionarios son colecciones desordenadas de pares clave-valor.

#### Crear y Acceder a Diccionarios:

```python

# Crear un diccionario

alumno = {

"nombre": "Juan",

"edad": 21,

"carrera": "Ingeniería"

}

# Acceder a valores

print(alumno["nombre"]) # Juan

print(alumno["edad"]) # 21

# Modificar valores

alumno["edad"] = 22

print(alumno) # {'nombre': 'Juan', 'edad': 22, 'carrera': 'Ingeniería'}

# Añadir nuevos pares clave-valor

alumno["universidad"] = "UNAM"

print(alumno) # {'nombre': 'Juan', 'edad': 22, 'carrera': 'Ingeniería', 'universidad': 'UNAM'}

```

#### Recorrer un Diccionario:

```python

# Recorrer claves

for clave in alumno:

print(clave)

# Recorrer valores

for valor in alumno.values():

print(valor)

# Recorrer claves y valores

for clave, valor in alumno.items():

print(f"{clave}: {valor}")

```

### Tarea

#### Parte 1: Manipulación de Listas y Tuplas

1. Crea un archivo llamado `lista\_tupla.py`.

2. Escribe un programa que:

- Cree una lista con los nombres de tus cinco películas favoritas.

- Imprima la lista completa.

- Reemplace la segunda película con una nueva y vuelva a imprimir la lista.

- Cree una tupla con los nombres de tres ciudades que te gustaría visitar.

- Imprima cada ciudad en la tupla usando un bucle `for`.

#### Parte 2: Uso de Conjuntos y Diccionarios

1. Crea un archivo llamado `conjunto\_diccionario.py`.

2. Escribe un programa que:

- Cree un conjunto con los números del 1 al 10.

- Añada los números 11 y 12 al conjunto.

- Imprima el conjunto resultante.

- Cree un diccionario que almacene información sobre un libro (título, autor, año de publicación).

- Imprima cada clave y valor del diccionario en un formato legible.

### Ejemplo de solución para `lista\_tupla.py`:

```python

# Lista de películas favoritas

peliculas = ["Inception", "Interstellar", "The Dark Knight", "Memento", "Dunkirk"]

print("Lista de películas:", peliculas)

# Reemplazar la segunda película

peliculas[1] = "Tenet"

print("Lista modificada:", peliculas)

# Tupla de ciudades a visitar

ciudades = ("Tokyo", "Paris", "New York")

print("Ciudades a visitar:")

for ciudad in ciudades:

print(ciudad)

```

### Ejemplo de solución para `conjunto\_diccionario.py`:

```python

# Conjunto de números

numeros = {1, 2, 3, 4, 5, 6, 7, 8, 9, 10}

numeros.add(11)

numeros.add(12)

print("Conjunto de números:", numeros)

# Diccionario de un libro

libro = {

"titulo": "Cien Años de Soledad",

"autor": "Gabriel García Márquez",

"año": 1967

}

print("Información del libro:")

for clave, valor in libro.items():

print(f"{clave}: {valor}")