**Informe Teórico: Conceptos Aplicados**

**Santino Barone**

**Maximo Quiroga**

**1. Listas**

Las listas son estructuras de datos **ordenadas y mutables** que permiten almacenar múltiples elementos en un solo contenedor.

* Se crean usando corchetes: mi\_lista = [1, 2, 3].
* Se puede acceder a sus elementos por índice: mi\_lista[0] devuelve 1.
* Permiten operaciones como agregar (append), eliminar (remove), recorrer (for) y modificar elementos.

**Ejemplo:**

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

frutas.append("kiwi")

**2. Diccionarios**

Los diccionarios son estructuras de datos **no ordenadas** que almacenan información en **pares clave-valor**.

* Se crean con llaves {}: persona = {"nombre": "Ana", "edad": 25}.
* Se accede a los valores usando la clave: persona["nombre"] devuelve "Ana".
* Permiten agregar, modificar y eliminar pares clave-valor.

**Ejemplo:**

notas = {"Juan": [7, 8, 9], "Ana": [10, 9, 8]}

**3. Funciones**

Las funciones son bloques de código reutilizables que realizan una tarea específica.

* Se definen con la palabra clave def:

def saludar(nombre):

print(f"Hola, {nombre}")

* Pueden recibir parámetros y devolver resultados (return).
* Ayudan a **organizar el código**, reducir repetición y mejorar la legibilidad.

**4. Condicionales**

Permiten tomar decisiones según condiciones lógicas:

* Se usan con if, elif y else.
* Evalúan expresiones que devuelven **True** o **False**.

**Ejemplo:**

edad = 18

if edad >= 18:

print("Mayor de edad")

else:

print("Menor de edad")

**5. Ordenamientos**

Python permite **ordenar listas** de manera ascendente o descendente.

* Método sort() para listas: numeros.sort().
* Función sorted() devuelve una lista ordenada sin modificar la original: sorted(numeros, reverse=True)

**Ejemplo:**

numeros = [5, 2, 9]

numeros.sort() # [2, 5, 9]

**6. Estadísticas básicas**

Se pueden calcular valores estadísticos simples con listas de números:

* **Promedio:** sum(lista)/len(lista)
* **Máximo y mínimo:** max(lista) y min(lista)
* **Suma:** sum(lista)

**Ejemplo:**

notas = [7, 8, 9]

promedio = sum(notas)/len(notas)

**7. Archivos CSV**

CSV (Comma-Separated Values) es un formato para almacenar datos tabulares.

* Se puede leer y escribir usando la librería csv de Python.
* Permite organizar datos como si fueran tablas (filas y columnas).

**Ejemplo de lectura:**

import csv

with open("datos.csv", "r") as archivo:

lector = csv.reader(archivo)

for fila in lector:

print(fila)

**Ejemplo de escritura:**

with open("datos.csv", "w", newline="") as archivo:

escritor = csv.writer(archivo)

escritor.writerow(["Nombre", "Edad"])

escritor.writerow(["Ana", 25])

**8. Flujo de operaciones principales**

El flujo típico en un programa que maneja datos en Python puede representarse así:

Inicio

│

▼

Leer datos (CSV o entrada del usuario)

│

▼

Almacenar datos en estructuras (listas o diccionarios)

│

▼

Procesar datos

├─ Aplicar condicionales

├─ Calcular estadísticas básicas

└─ Ordenar información

│

▼

Definir funciones para tareas repetitivas

│

▼

Generar resultados / guardar en CSV

│

▼

Fin