Ejercicios adaptados al **ámbito biosanitario** para practicar el uso de **listas**, **tuplas**, **sets** y **diccionarios** en Python

---

1️⃣ Listas: Gestión de Pacientes en una Clínica

Ejercicio:
Tienes una lista de pacientes que se han registrado en una clínica. Realiza las siguientes operaciones:

1. Agrega un nuevo paciente a la lista.
2. Elimina a un paciente que ya no se presenta.
3. Encuentra el índice del paciente "María López".
4. Ordena la lista de pacientes alfabéticamente.

Pistas:

- Usa los métodos `append()`, `remove()`, `index()`, y `sort()`.

```
pacientes = ["Juan Pérez", "María López", "Carlos Gómez", "Ana Torres"]
```

2️⃣ Tuplas: Parámetros Vitales de un Paciente

Ejercicio:
Tienes una tupla con los parámetros vitales de un paciente, que no se deben modificar:

1. Accede y muestra el nombre del paciente.
2. Muestra la presión arterial (sistólica y diastólica) del paciente.
3. Si el valor de saturación de oxígeno es menor a 90, muestra un mensaje de alerta.

Pistas:

- Usa índices para acceder a los valores de la tupla.

```
parametros_vitales = ("Juan Pérez", 36.5, 120, 80, 98)  # (nombre, temperatura,
```

```
sistólica, diastólica, saturación O2)
```

3️⃣ Sets: Diagnósticos Únicos de Enfermedades

Ejercicio:
Tienes un conjunto de diagnósticos de enfermedades de varios pacientes. Realiza las siguientes operaciones:

1. Agrega un nuevo diagnóstico al conjunto.
2. Elimina un diagnóstico que ya no se está utilizando.
3. Encuentra la intersección entre los diagnósticos de dos pacientes.
4. Verifica si un conjunto de diagnósticos está completamente incluido en otro conjunto.

Pistas:

- Usa los métodos `add()`, `remove()`, `intersection()`, y `issubset()`.

```
diagnosticos = {"diabetes", "hipertensión", "asma", "influenza", "COVID-19"}
```

4️⃣ Diccionarios: Historia Clínica de un Paciente

Ejercicio:
Tienes un diccionario con la información de un paciente. Realiza las siguientes operaciones:

1. Agrega un nuevo tratamiento para el paciente.
2. Elimina el contacto de emergencia.
3. Muestra el nombre del paciente, su edad y los diagnósticos.
4. Verifica si el paciente tiene un tratamiento para "hipertensión" y, en caso afirmativo, muestra el medicamento.

Pistas:

- Usa los métodos `update()`, `pop()`, `get()`, y `keys()`.

```
historia_clinica = {
    "nombre": "María López",
    "edad": 45,
```

```
    "diagnosticos": ["diabetes", "hipertensión"],
    "tratamientos": {"metformina": "500 mg/día", "enalapril": "10 mg/día"},
    "contacto_emergencia": {"nombre": "Carlos López", "teléfono": "555-1234"}
}
```

5️⃣ Listas: Seguimiento de Medicamentos Administrados

Ejercicio:
Tienes una lista de medicamentos administrados a pacientes durante la semana. Realiza las siguientes operaciones:

1. Agrega un nuevo medicamento al final de la lista.
2. Elimina el medicamento que se administró el martes.
3. Cuenta cuántas veces se administró el medicamento "paracetamol".
4. Ordena la lista de medicamentos administrados en orden alfabético.

Pistas:

- Usa los métodos `append()`, `remove()`, `count()`, y `sort()`.

```
medicamentos_admin = ["paracetamol", "ibuprofeno", "metformina", "insulina", "paracetamol"]
```

6️⃣ Tuplas: Medicamentos y Dosis Prescritos

Ejercicio:
Tienes una tupla con los nombres de los medicamentos y las dosis prescritas de un paciente:

1. Accede y muestra el medicamento "metformina" junto con su dosis.
2. Muestra la cantidad total de medicamentos prescritos.

Pistas:

- Usa índices para acceder a los elementos de la tupla.

```
medicamentos_prescritos = ("metformina", "500 mg", "enalapril", "10 mg", "insulina", "20 UI")
```

7️⃣ Sets: Registro de Enfermedades Preexistentes

Ejercicio:
Tienes un conjunto de enfermedades preexistentes de un paciente:

1. Agrega una nueva enfermedad al registro.
2. Elimina una enfermedad del conjunto.
3. Verifica si el paciente tiene "diabetes" en su registro.

Pistas:

- Usa los métodos `add()`, `remove()`, y la operación `in`.

```
enfermedades_preexistentes = {"hipertensión", "asma", "diabetes"}
```

8️⃣ Diccionarios: Datos de Paciente con Consultas

Ejercicio:
Tienes un diccionario con datos de pacientes y sus consultas médicas:

1. Agrega una nueva consulta para un paciente específico.
2. Elimina un paciente que ya no pertenece al registro.
3. Muestra todas las consultas de un paciente.

Pistas:

- Usa los métodos `update()`, `pop()`, y `values()`.

```
pacientes_registro = {
    "Juan Pérez": ["consulta 1", "consulta 2"],
    "María López": ["consulta 1", "consulta 2"]
}
```

9️⃣ Listas: Pacientes en Espera para Consulta

Ejercicio:
Se tiene una lista de pacientes que esperan para la consulta. Realiza las siguientes operaciones:

1. Agrega 3 nuevos pacientes a la lista.
2. Elimina al paciente que tiene más tiempo esperando.
3. Muestra la lista de pacientes en orden de llegada (del primero al último).

Pistas:

- Usa los métodos `append()`, `remove()`, y `sort()`.

```
pacientes_espera = ["Carlos Gómez", "Ana Torres", "Juan Pérez"]
```

🔟 Tuplas: Medicamentos Recetados por Especialidad

Ejercicio:
Se tienen tuplas de medicamentos prescritos para distintas especialidades médicas. Realiza las siguientes operaciones:

1. Encuentra los medicamentos recetados en la especialidad de "cardiología".
2. Obtén el primer medicamento recetado en "neurología".

Pistas:

- Usa índices y un ciclo para acceder a las tuplas de cada especialidad.

```
medicamentos_cardiologia = ("atenolol", "amlodipino", "lisinopril")
medicamentos_neurologia = ("gabapentina", "lamotrigina")
```

1️⃣1️⃣ Sets: Diagnósticos Comunes entre Pacientes

Ejercicio:
Tienes un conjunto de diagnósticos de varios pacientes. Encuentra los

diagnósticos comunes entre dos pacientes y aquellos que son exclusivos de cada uno.

Pistas:

- Usa los métodos `intersection()` y `difference()`.

```
diagnosticos_paciente_1 = {"diabetes", "hipertensión", "insuficiencia renal"}
diagnosticos_paciente_2 = {"hipertensión", "asma", "artritis"}
```

1️⃣2️⃣ Diccionarios: Historial de Medicamentos de Pacientes

Ejercicio:
Tienes un diccionario con los medicamentos recetados a diferentes pacientes. Realiza las siguientes operaciones:

1. Agrega un medicamento a un paciente específico.
2. Muestra los medicamentos recetados a un paciente dado.
3. Si un paciente tiene más de 3 medicamentos recetados, muestra un mensaje indicando "Paciente con tratamiento complejo".

Pistas:

- Usa los métodos `update()`, `get()`, y `len()`.

```
medicamentos_pacientes = {
    "Juan Pérez": ["metformina", "enalapril"],
    "María López": ["insulina", "paracetamol", "ibuprofeno"]
}
```

1️⃣3️⃣ Listas: Seguimiento de Resultados de Laboratorio

Ejercicio:
Tienes una lista con los resultados de los análisis de sangre de varios pacientes. Realiza las siguientes operaciones:

1. Añade los resultados de 3 nuevos pacientes.
2. Ordena los resultados por valores numéricos (por ejemplo, niveles de glucosa).
3. Filtra los resultados donde el nivel de glucosa es superior a 200.

Pistas:

- Usa los métodos `append()`, `sort()`, y un ciclo para filtrar.

```
resultados_laboratorio = [(105, "Juan Pérez"), (245, "Carlos Gómez"), (190, "Ana Torres")]
```

1️⃣4️⃣ Tuplas: Consulta Médica con Fechas

Ejercicio:
Tienes tuplas con la fecha de las consultas médicas de los pacientes. Realiza las siguientes operaciones:

1. Accede a la fecha de la consulta de un paciente específico.
2. Verifica si el paciente tiene una consulta programada dentro de los próximos 7 días.

Pistas:

- Usa índices y operaciones con fechas.

```
consultas_paciente = ("2025-02-01", "Juan Pérez")
```

1️⃣5️⃣ Sets: Enfermedades Crónicas y Agudas

Ejercicio:
Tienes un conjunto de enfermedades crónicas y otro de enfermedades agudas. Encuentra las enfermedades comunes y las que son exclusivas de cada conjunto.

Pistas:

- Usa los métodos `intersection()` y `difference()`.

```
enfermedades_cronicas = {"hipertensión", "diabetes", "asma"}
enfermedades_agudas = {"influenza", "neumonía", "diabetes"}
```

1️⃣6️⃣ Diccionarios: Asignación de Medicamentos por Paciente

Ejercicio:
Tienes un diccionario con los pacientes y sus respectivos tratamientos. Realiza las siguientes operaciones:

1. Añade un nuevo medicamento a un paciente específico.
2. Muestra los tratamientos de todos los pacientes con más de 2 medicamentos.
3. Verifica si un paciente tiene un tratamiento para "hipertensión".

Pistas:

- Usa los métodos `update()`, `len()`, y `get()`.

```
tratamientos_pacientes = {
    "Juan Pérez": ["metformina", "enalapril"],
    "María López": ["insulina", "paracetamol", "ibuprofeno"]
}
```

1️⃣7️⃣ Listas: Control de Pacientes en Sala de Espera

Ejercicio:
Tienes una lista con los pacientes que han llegado a la sala de espera. Realiza las siguientes operaciones:

1. Añade un nuevo paciente que acaba de llegar.
2. Elimina al paciente que ya ha sido atendido.
3. Muestra el primer y el último paciente de la lista.

Pistas:

- Usa los métodos `append()`, `remove()`, y acceso por índices.

```
sala_espera = ["Juan Pérez", "María López", "Carlos Gómez"]
```

1️⃣8️⃣ Tuplas: Medicamentos Prescritos y Dosis

Ejercicio:
Tienes una tupla con medicamentos y sus respectivas dosis para un paciente. Realiza las siguientes operaciones:

1. Accede a los medicamentos prescritos y su dosis.
2. Verifica si la dosis de "metformina" es superior a 500 mg.

Pistas:

- Usa índices y un ciclo.

```
medicamentos_prescritos = ("metformina", "500 mg", "enalapril", "10 mg")
```

1️⃣9️⃣ Sets: Medicamentos en Stock

Ejercicio:
Tienes un conjunto con los medicamentos disponibles en el stock del hospital. Realiza las siguientes operaciones:

1. Añade un nuevo medicamento al stock.
2. Elimina un medicamento que está fuera de stock.
3. Verifica si un medicamento específico está en stock.

Pistas:

- Usa los métodos `add()`, `remove()`, y la operación `in`.

```
stock_medicamentos = {"paracetamol", "ibuprofeno", "insulina"}
```

2️⃣0️⃣ Diccionarios: Seguimiento de Pacientes por Médico

Ejercicio:
Tienes un diccionario que asocia a cada médico con una lista de pacientes que ha atendido. Realiza las siguientes operaciones:

1. Añade un nuevo paciente a la lista de un médico.
2. Elimina un paciente de la lista de un médico.
3. Muestra todos los pacientes atendidos por un médico específico.

Pistas:

- Usa los métodos `update()`, `pop()`, y acceso por claves.

```
pacientes_medico = {
    "Dr. López": ["Juan Pérez", "Carlos Gómez"],
    "Dr. Fernández": ["Ana Torres", "María López"]
}
```

Estos ejercicios te permitirán reforzar el manejo de estructuras de datos en Python mientras trabajas en situaciones reales del sector salud.