Libro de Ejercicios Resueltos: Colecciones de Datos en Python

Ejercicio 1: Manejo de Listas

Enunciado:

Crea una lista con los nombres de cinco productos. Posteriormente, realiza las siguientes operaciones:

1. Añade un nuevo producto al final de la lista.
2. Ordena la lista alfabéticamente.
3. Elimina el segundo producto de la lista.
4. Inserta un producto en la segunda posición.
5. Utiliza una comprensión de listas para crear una nueva lista con los nombres de los productos que comienzan con la letra 'M'.

Solución:

```
python

# 1. Crear la lista de productos
productos = ["Laptop", "Teclado", "Ratón", "Monitor", "Impresora"]

# 2. Añadir un nuevo producto al final
productos.append("Auriculares")

# 3. Ordenar la lista alfabéticamente
productos.sort()
print("Lista ordenada:", productos)

# 4. Eliminar el segundo producto
productos.pop(1)
print("Después de eliminar el segundo producto:", productos)

# 5. Insertar un producto en la segunda posición
productos.insert(1, "Cámara")
print("Después de insertar un producto:", productos)

# 6. Crear una nueva lista con productos que comiencen con 'M'
productos_m = [producto for producto in productos if
producto.startswith('M')]
print("Productos que comienzan con 'M':", productos_m)
```

Resultado:

```
python

Lista ordenada: ['Auriculares', 'Impresora', 'Laptop', 'Monitor',
'Ratón', 'Teclado']
Después de eliminar el segundo producto: ['Auriculares', 'Laptop',
'Monitor', 'Ratón', 'Teclado']
Después de insertar un producto: ['Auriculares', 'Cámara', 'Laptop',
'Monitor', 'Ratón', 'Teclado']
```

```
Productos que comienzan con 'M': ['Monitor']
```

Ejercicio 2: Uso de Tuplas

Enunciado:

Crea una tupla con los días de la semana. Luego, realiza lo siguiente:

1. Muestra el tercer día de la semana.
2. Intenta modificar uno de los elementos de la tupla (esto debería generar un error).
3. Convierte la tupla en una lista, modifica uno de los días, y vuelve a convertirla en una tupla.

Solución:

```
python

# 1. Crear la tupla con los días de la semana
dias_semana = ("Lunes", "Martes", "Miércoles", "Jueves", "Viernes",
"Sábado", "Domingo")

# 2. Mostrar el tercer día
print("Tercer día:", dias_semana[2])

# 3. Intentar modificar un elemento (esto generará un error)
# dias_semana[1] = "Lunes"  # Esto produce un error porque las tuplas
son inmutables

# 4. Convertir la tupla a una lista, modificarla, y convertirla de
nuevo a tupla
dias_semana_lista = list(dias_semana)
dias_semana_lista[1] = "Lunes"
dias_semana_modificada = tuple(dias_semana_lista)
print("Tupla modificada:", dias_semana_modificada)
```

Resultado:

```
python

Tercer día: Miércoles
Tupla modificada: ('Lunes', 'Lunes', 'Miércoles', 'Jueves', 'Viernes',
'Sábado', 'Domingo')
```

Ejercicio 3: Operaciones con Conjuntos

Enunciado:

Crea dos conjuntos: uno con los nombres de los empleados del departamento de ventas y otro con los empleados del departamento de marketing. Realiza las siguientes operaciones:

1. Muestra los empleados que trabajan en ambos departamentos.
2. Muestra los empleados que solo trabajan en el departamento de ventas.
3. Muestra todos los empleados sin duplicados (unión de ambos departamentos).

Solución:

```
python

# Crear los conjuntos
ventas = {"Ana", "Luis", "Carlos", "Marta"}
marketing = {"Marta", "Carlos", "Pedro", "Lucía"}

# 1. Empleados en ambos departamentos (intersección)
en_ambos = ventas.intersection(marketing)
print("Empleados en ambos departamentos:", en_ambos)

# 2. Empleados solo en ventas (diferencia)
solo_ventas = ventas.difference(marketing)
print("Empleados solo en ventas:", solo_ventas)

# 3. Todos los empleados sin duplicados (unión)
todos = ventas.union(marketing)
print("Todos los empleados:", todos)
```

Resultado:

```
python

Empleados en ambos departamentos: {'Carlos', 'Marta'}
Empleados solo en ventas: {'Ana', 'Luis'}
Todos los empleados: {'Ana', 'Luis', 'Carlos', 'Pedro', 'Marta',
'Lucía'}
```

Ejercicio 4: Gestión de Inventarios con Diccionarios

Enunciado:

Crea un diccionario para gestionar el inventario de una tienda, donde las claves sean los nombres de los productos y los valores sean subdiccionarios con el precio y la cantidad en stock. Realiza las siguientes operaciones:

1. Añade un nuevo producto al inventario.
2. Modifica el precio de un producto.
3. Reduce el stock de un producto después de una venta.
4. Muestra todos los productos disponibles.

Solución:

```
python

# Crear el diccionario de inventario
inventario = {
    "Laptop": {"precio": 1200, "stock": 5},
    "Ratón": {"precio": 25, "stock": 50},
```

```
    "Teclado": {"precio": 45, "stock": 30}
}

# 1. Añadir un nuevo producto al inventario
inventario["Monitor"] = {"precio": 300, "stock": 15}

# 2. Modificar el precio de un producto
inventario["Laptop"]["precio"] = 1100

# 3. Reducir el stock de un producto después de una venta
inventario["Ratón"]["stock"] -= 1

# 4. Mostrar todos los productos disponibles
for producto, detalles in inventario.items():
    print(f"Producto: {producto}, Precio: {detalles['precio']}, Stock:
{detalles['stock']}")
```

Resultado:

```
python

Producto: Laptop, Precio: 1100, Stock: 5
Producto: Ratón, Precio: 25, Stock: 49
Producto: Teclado, Precio: 45, Stock: 30
Producto: Monitor, Precio: 300, Stock: 15
```