Evaluación Final - Proyecto Integral

Proyecto: Sistema de Gestión de Biblioteca

Crea un **programa completo de gestión de biblioteca** que integre **listas, diccionarios y tuplas**. El sistema debe permanecer activo en un menú principal hasta que el usuario elija salir.

Requisitos del Sistema:

1. Gestión de Libros

- **Agregar libro**: Título, autor, año, género, disponible (True/False)
- **Ver todos los libros**: Mostrar lista completa con información
- **Buscar libro por título**: Encontrar y mostrar detalles
- **Eliminar libro**: Por título

2. Gestión de Usuarios

- **Registrar usuario**: Nombre, email, carrera (guardar como tupla: `(nombre, email, carrera)`)
- **Ver usuarios registrados**: Mostrar lista completa
- **Eliminar usuario**: Por nombre

3. Préstamos

- **Realizar préstamo**: Seleccionar usuario, seleccionar libro (solo si está disponible)
  - Guardar: `{usuario: [libro1, libro2, ...], ...}`
  - Cambiar disponibilidad del libro a `False`
- **Devoluciones**: Usuario devuelve libro, vuelve a estar disponible
- **Ver préstamos de un usuario**: Mostrar todos sus libros prestados
- **Libros actualmente no disponibles**: Mostrar cuáles están prestados

4. Reportes

- **Género más prestado**: Contar prestamos por género
- **Usuario con más préstamos**: Encontrar el que tiene más libros
- **Libros disponibles**: Contar y mostrar

5. Estructura de Datos Recomendada:

```
# Libros: lista de diccionarios
libros = [
```

```
    {"titulo": "Clean Code", "autor": "Robert Martin", "año": 2008,
     "genero": "Técnico", "disponible": True},
    {"titulo": "1984", "autor": "George Orwell", "año": 1949,
     "genero": "Ficción", "disponible": False}
]

# Usuarios: lista de tuplas
usuarios = [("Ana García", "ana@example.com", "Sistemas"),
            ("Carlos López", "carlos@example.com", "Ingeniería")]

# Préstamos: diccionario
prestamos = {
    "Ana García": ["Clean Code", "1984"],
    "Carlos López": ["Python Basics"]
}
```

Menú Principal:

```
========== SISTEMA DE BIBLIOTECA ==========

1. Gestión de Libros
   1.1. Agregar libro
   1.2. Ver todos los libros
   1.3. Buscar libro
   1.4. Eliminar libro

2. Gestión de Usuarios
   2.1. Registrar usuario
   2.2. Ver usuarios
   2.3. Eliminar usuario

3. Préstamos y Devoluciones
   3.1. Realizar préstamo
   3.2. Devolver libro
   3.3. Ver préstamos de usuario
   3.4. Ver libros no disponibles

4. Reportes
   4.1. Género más prestado
   4.2. Usuario con más préstamos
   4.3. Libros disponibles

5. Salir
```

Requisitos de Implementación:

✅ DEBE usar:

- Listas de diccionarios para libros
- Tuplas para usuarios
- Diccionarios para préstamos
- Bucles y condicionales
- Validación de entradas
- Manejo de errores (try-except)

✅ **Funcionalidades extra (opcional, suma puntos):**

- Mostrar fecha y hora de préstamo
- Calcular multa por atraso
- Sistema de ratings (calificar libros)
- Buscar por autor o género

Ejemplo de Interacción:

```
========== SISTEMA DE BIBLIOTECA ==========
1. Gestión de Libros
2. Gestión de Usuarios
3. Préstamos y Devoluciones
4. Reportes
5. Salir

Opción: 1
  1.1. Agregar libro
  1.2. Ver todos los libros
  1.3. Buscar libro
  1.4. Eliminar libro

Opción: 1.1
Título: Clean Code
Autor: Robert Martin
Año: 2008
Género: Técnico
✅ Libro agregado

Opción: 2
  2.1. Registrar usuario
  2.2. Ver usuarios
  2.3. Eliminar usuario

Opción: 2.1
Nombre: Ana García
Email: ana@example.com
Carrera: Sistemas
✅ Usuario registrado
```

```
Opción: 3
  3.1. Realizar préstamo
  3.2. Devolver libro
  3.3. Ver préstamos
  3.4. Ver no disponibles

Opción: 3.1
Usuario: Ana García
Libro: Clean Code
✅ Préstamo realizado

Opción: 5
¡Hasta luego!
```

SOLUCIÓN

```python
# SISTEMA INTEGRAL DE GESTIÓN DE BIBLIOTECA
# Integra: Listas, Diccionarios y Tuplas

import os
from datetime import datetime

print("=" * 60)
print("SISTEMA DE GESTIÓN DE BIBLIOTECA".center(60))
print("=" * 60)
print()

# ESTRUCTURA DE DATOS
libros = []
usuarios = []
prestamos = {}

def limpiar_pantalla():
    """Limpiar la pantalla (opcional)"""
    os.system('cls' if os.name == 'nt' else 'clear')

def mostrar_menu_principal():
    """Mostrar menú principal"""
    print("-" * 60)
    print("MENÚ PRINCIPAL")
    print("-" * 60)
    print("1. Gestión de Libros")
    print("2. Gestión de Usuarios")
    print("3. Préstamos y Devoluciones")
    print("4. Reportes")
    print("5. Salir")
    print("-" * 60)
```

```
def mostrar_menu_libros():
    """Menú de libros"""
    print("-" * 60)
    print("GESTIÓN DE LIBROS")
    print("-" * 60)
    print("1.1. Agregar libro")
    print("1.2. Ver todos los libros")
    print("1.3. Buscar libro")
    print("1.4. Eliminar libro")
    print("1.5. Volver")
    print("-" * 60)

def mostrar_menu_usuarios():
    """Menú de usuarios"""
    print("-" * 60)
    print("GESTIÓN DE USUARIOS")
    print("-" * 60)
    print("2.1. Registrar usuario")
    print("2.2. Ver usuarios")
    print("2.3. Eliminar usuario")
    print("2.4. Volver")
    print("-" * 60)

def mostrar_menu_prestamos():
    """Menú de préstamos"""
    print("-" * 60)
    print("PRÉSTAMOS Y DEVOLUCIONES")
    print("-" * 60)
    print("3.1. Realizar préstamo")
    print("3.2. Devolver libro")
    print("3.3. Ver préstamos de usuario")
    print("3.4. Ver libros no disponibles")
    print("3.5. Volver")
    print("-" * 60)

def mostrar_menu_reportes():
    """Menú de reportes"""
    print("-" * 60)
    print("REPORTES")
    print("-" * 60)
    print("4.1. Género más prestado")
    print("4.2. Usuario con más préstamos")
    print("4.3. Libros disponibles")
    print("4.4. Volver")
    print("-" * 60)

# ==================== FUNCIONES DE LIBROS ====================

def agregar_libro():
    """Agregar un nuevo libro"""
    print()
    titulo = input("Título del libro: ").strip()
```

```
    # Verificar si ya existe
    if any(libro["titulo"].lower() == titulo.lower() for libro in libros):
        print("❌ El libro ya existe en la biblioteca")
        return

    autor = input("Autor: ").strip()
    try:
        año = int(input("Año de publicación: "))
    except ValueError:
        print("❌ Ingresa un año válido")
        return

    genero = input("Género: ").strip()

    libros.append({
        "titulo": titulo,
        "autor": autor,
        "año": año,
        "genero": genero,
        "disponible": True
    })
    print(f"✅ Libro '{titulo}' agregado a la biblioteca")
    print()

def ver_todos_libros():
    """Ver todos los libros"""
    print()
    if len(libros) == 0:
        print("📭 No hay libros en la biblioteca")
    else:
        print("📚 LIBROS EN LA BIBLIOTECA")
        for i, libro in enumerate(libros, 1):
            estado = "✅ Disponible" if libro["disponible"] else "❌ Prestado"
            print(f"\n{i}. {libro['titulo']}")
            print(f"   Autor: {libro['autor']}")
            print(f"   Año: {libro['año']}")
            print(f"   Género: {libro['genero']}")
            print(f"   Estado: {estado}")
    print()

def buscar_libro():
    """Buscar un libro por título"""
    print()
    titulo = input("Título a buscar: ").strip()

    encontrado = None
    for libro in libros:
        if libro["titulo"].lower() == titulo.lower():
            encontrado = libro
            break

    if encontrado:
```

```
            print(f"\n✅ Libro encontrado:")
            print(f"   Título: {encontrado['titulo']}")
            print(f"   Autor: {encontrado['autor']}")
            print(f"   Año: {encontrado['año']}")
            print(f"   Género: {encontrado['genero']}")
            print(f"   Estado: {'Disponible' if encontrado['disponible'] else 'Prestado'}")
        else:
            print(f"❌ Libro '{titulo}' no encontrado")
        print()

    def eliminar_libro():
        """Eliminar un libro"""
        print()
        titulo = input("Título del libro a eliminar: ").strip()

        libro_encontrado = None
        for libro in libros:
            if libro["titulo"].lower() == titulo.lower():
                libro_encontrado = libro
                break

        if libro_encontrado:
            libros.remove(libro_encontrado)
            print(f"✅ Libro '{titulo}' eliminado")
        else:
            print(f"❌ Libro '{titulo}' no encontrado")
        print()

    # ==================== FUNCIONES DE USUARIOS ====================

    def registrar_usuario():
        """Registrar un nuevo usuario"""
        print()
        nombre = input("Nombre del usuario: ").strip()

        # Verificar si ya existe
        if any(usuario[0].lower() == nombre.lower() for usuario in usuarios):
            print("❌ El usuario ya está registrado")
            return

        email = input("Email: ").strip()
        carrera = input("Carrera: ").strip()

        # Agregar como tupla
        usuarios.append((nombre, email, carrera))
        prestamos[nombre] = []
        print(f"✅ Usuario '{nombre}' registrado")
        print()

    def ver_usuarios():
        """Ver todos los usuarios"""
        print()
        if len(usuarios) == 0:
```

```
        print("👥 No hay usuarios registrados")
    else:
        print("👥 USUARIOS REGISTRADOS")
        for i, usuario in enumerate(usuarios, 1):
            print(f"\n{i}. {usuario[0]}")
            print(f"   Email: {usuario[1]}")
            print(f"   Carrera: {usuario[2]}")
    print()

def eliminar_usuario():
    """Eliminar un usuario"""
    print()
    nombre = input("Nombre del usuario a eliminar: ").strip()

    usuario_encontrado = None
    for usuario in usuarios:
        if usuario[0].lower() == nombre.lower():
            usuario_encontrado = usuario
            break

    if usuario_encontrado:
        usuarios.remove(usuario_encontrado)
        if nombre in prestamos:
            del prestamos[nombre]
        print(f"✅ Usuario '{nombre}' eliminado")
    else:
        print(f"❌ Usuario '{nombre}' no encontrado")
    print()

# ==================== FUNCIONES DE PRÉSTAMOS ====================

def realizar_prestamo():
    """Realizar un préstamo"""
    print()

    if len(usuarios) == 0:
        print("❌ No hay usuarios registrados")
        print()
        return

    if len(libros) == 0:
        print("❌ No hay libros disponibles")
        print()
        return

    # Mostrar usuarios
    print("Usuarios registrados:")
    for i, usuario in enumerate(usuarios, 1):
        print(f"{i}. {usuario[0]}")

    try:
        num_usuario = int(input("Selecciona número de usuario: ")) - 1
        if num_usuario < 0 or num_usuario >= len(usuarios):
```

```
            print("❌ Opción no válida")
            print()
            return
    except ValueError:
        print("❌ Ingresa un número válido")
        print()
        return

    nombre_usuario = usuarios[num_usuario][0]

    # Mostrar libros disponibles
    disponibles = [libro for libro in libros if libro["disponible"]]

    if len(disponibles) == 0:
        print("❌ No hay libros disponibles para prestar")
        print()
        return

    print("\nLibros disponibles:")
    for i, libro in enumerate(disponibles, 1):
        print(f"{i}. {libro['titulo']} - {libro['autor']}")

    try:
        num_libro = int(input("Selecciona número de libro: ")) - 1
        if num_libro < 0 or num_libro >= len(disponibles):
            print("❌ Opción no válida")
            print()
            return
    except ValueError:
        print("❌ Ingresa un número válido")
        print()
        return

    libro_prestado = disponibles[num_libro]

    # Realizar préstamo
    libro_prestado["disponible"] = False
    prestamos[nombre_usuario].append(libro_prestado["titulo"])
    print(f"✅ Préstamo realizado a {nombre_usuario}: {libro_prestado['titulo']}")
    print()

def devolver_libro():
    """Devolver un libro"""
    print()

    nombre_usuario = input("Nombre del usuario: ").strip()

    if nombre_usuario not in prestamos or len(prestamos[nombre_usuario]) == 0:
        print("❌ Este usuario no tiene préstamos")
        print()
        return

    print(f"Libros prestados a {nombre_usuario}:")
```

```
        for i, titulo in enumerate(prestamos[nombre_usuario], 1):
            print(f"{i}. {titulo}")

        try:
            num_libro = int(input("Número del libro a devolver: ")) - 1
            if num_libro < 0 or num_libro >= len(prestamos[nombre_usuario]):
                print("❌ Opción no válida")
                print()
                return
        except ValueError:
            print("❌ Ingresa un número válido")
            print()
            return

        titulo_devuelto = prestamos[nombre_usuario].pop(num_libro)

        # Marcar como disponible
        for libro in libros:
            if libro["titulo"] == titulo_devuelto:
                libro["disponible"] = True
                break

        print(f"✅ '{titulo_devuelto}' devuelto por {nombre_usuario}")
        print()

    def ver_prestamos_usuario():
        """Ver préstamos de un usuario"""
        print()
        nombre = input("Nombre del usuario: ").strip()

        if nombre not in prestamos:
            print(f"❌ Usuario '{nombre}' no encontrado")
        elif len(prestamos[nombre]) == 0:
            print(f"📭 {nombre} no tiene préstamos activos")
        else:
            print(f"📚 Libros prestados a {nombre}:")
            for i, titulo in enumerate(prestamos[nombre], 1):
                print(f"{i}. {titulo}")
        print()

    def ver_no_disponibles():
        """Ver libros que no están disponibles"""
        print()
        no_disponibles = [libro for libro in libros if not libro["disponible"]]

        if len(no_disponibles) == 0:
            print("✅ Todos los libros están disponibles")
        else:
            print("❌ LIBROS NO DISPONIBLES")
            for i, libro in enumerate(no_disponibles, 1):
                print(f"{i}. {libro['titulo']} - {libro['autor']}")
        print()
```

```
# ==================== FUNCIONES DE REPORTES ====================

def genero_mas_prestado():
    """Encontrar el género más prestado"""
    print()

    conteo_generos = {}
    for usuario_libros in prestamos.values():
        for titulo_lib in usuario_libros:
            for libro in libros:
                if libro["titulo"] == titulo_lib:
                    genero = libro["genero"]
                    conteo_generos[genero] = conteo_generos.get(genero, 0) + 1

    if len(conteo_generos) == 0:
        print("📭 No hay préstamos registrados")
    else:
        genero_max = max(conteo_generos, key=conteo_generos.get)
        print(f"🏆 Género más prestado: {genero_max} ({conteo_generos[genero_max]} préstamos)'
    print()

def usuario_mas_prestamos():
    """Encontrar usuario con más préstamos"""
    print()

    if len(prestamos) == 0:
        print("📭 No hay préstamos")
        print()
        return

    usuario_max = max(prestamos, key=lambda u: len(prestamos[u]))
    cantidad = len(prestamos[usuario_max])

    if cantidad == 0:
        print("📭 No hay préstamos activos")
    else:
        print(f"👑 Usuario con más préstamos: {usuario_max} ({cantidad} libros)")
    print()

def libros_disponibles():
    """Contar libros disponibles"""
    print()
    disponibles = [libro for libro in libros if libro["disponible"]]

    print(f"📚 Libros disponibles: {len(disponibles)} de {len(libros)}")
    if len(disponibles) > 0:
        print("\nLista:")
        for i, libro in enumerate(disponibles, 1):
            print(f"{i}. {libro['titulo']} - {libro['autor']}")
    print()

# ==================== BUCLE PRINCIPAL ====================
```

```
    while True:
        mostrar_menu_principal()
        opcion = input("Elige una opción: ").strip()

        # GESTIÓN DE LIBROS
        if opcion == "1":
            while True:
                mostrar_menu_libros()
                sub_opcion = input("Elige una opción: ").strip()

                if sub_opcion == "1.1":
                    agregar_libro()
                elif sub_opcion == "1.2":
                    ver_todos_libros()
                elif sub_opcion == "1.3":
                    buscar_libro()
                elif sub_opcion == "1.4":
                    eliminar_libro()
                elif sub_opcion == "1.5":
                    break
                else:
                    print("❌ Opción no válida\n")

        # GESTIÓN DE USUARIOS
        elif opcion == "2":
            while True:
                mostrar_menu_usuarios()
                sub_opcion = input("Elige una opción: ").strip()

                if sub_opcion == "2.1":
                    registrar_usuario()
                elif sub_opcion == "2.2":
                    ver_usuarios()
                elif sub_opcion == "2.3":
                    eliminar_usuario()
                elif sub_opcion == "2.4":
                    break
                else:
                    print("❌ Opción no válida\n")

        # PRÉSTAMOS Y DEVOLUCIONES
        elif opcion == "3":
            while True:
                mostrar_menu_prestamos()
                sub_opcion = input("Elige una opción: ").strip()

                if sub_opcion == "3.1":
                    realizar_prestamo()
                elif sub_opcion == "3.2":
                    devolver_libro()
                elif sub_opcion == "3.3":
                    ver_prestamos_usuario()
                elif sub_opcion == "3.4":
```

```
                ver_no_disponibles()
            elif sub_opcion == "3.5":
                break
            else:
                print("❌ Opción no válida\n")

    # REPORTES
    elif opcion == "4":
        while True:
            mostrar_menu_reportes()
            sub_opcion = input("Elige una opción: ").strip()

            if sub_opcion == "4.1":
                genero_mas_prestado()
            elif sub_opcion == "4.2":
                usuario_mas_prestamos()
            elif sub_opcion == "4.3":
                libros_disponibles()
            elif sub_opcion == "4.4":
                break
            else:
                print("❌ Opción no válida\n")

    # SALIR
    elif opcion == "5":
        print("\n✅ ¡Gracias por usar el Sistema de Biblioteca!")
        print("=" * 60)
        break

    else:
        print("❌ Opción no válida\n")
```

¡PROYECTO COMPLETADO! 🎉

Este código integra todos los conceptos de los PDF 1-4:

- ✅ Condicionales (if, elif, else)
- ✅ Bucles (while, for)
- ✅ Listas y operaciones
- ✅ Diccionarios y anidamiento
- ✅ Tuplas inmutables
- ✅ Funciones y modularidad
- ✅ Manejo de datos complejos
- ✅ Validación de entradas
- ✅ Menús interactivos