Python EJ 19

CRUD Centro deportivo

En este ejercicio hay que crear una base de datos para un centro deportivo en la que se puedan registrar clientes, ver a los clientes registrados, actualizar o borrar clientes, registrar, visualizar, actualizar o borrar actividades, registrar, visualizar, actualizar o borrar entrenadores y registrar, visualizar y borrar inscripciones.

· Crear una base de datos

· Conectarse a la base de datos

· Definir funciones para las opciones del menú

· Definir funciones para crear los menús

· Crear el menú principal

bdd.py

import mysql.connector

def conectar(): #Conecta con la base de datos

    conexion = mysql.connector.connect(

        host="localhost",    # Cambia según tu configuración

        user="root",      # Cambia según tu configuración

        password="curso",  # Cambia según tu configuración

        database="centro\_deportivo"

    )

    return conexion

clientes.py

from bdd import conectar

def crear\_cliente(nombre, edad, tipo\_membresia):

    conexion = conectar()

    cursor = conexion.cursor()

    consulta = "INSERT INTO clientes (nombre, edad, tipo\_membresia) VALUES (%s, %s, %s)"

    cursor.execute(consulta, (nombre, edad, tipo\_membresia))

    conexion.commit()

    cursor.close()

    conexion.close()

def leer\_clientes():

    conexion = conectar()

    cursor = conexion.cursor()

    cursor.execute("select \* from clientes")

    clientes = cursor.fetchall()

    cursor.close

    conexion.close

    return clientes

def actualizar\_cliente(id\_cliente, nuevo\_nombre, nueva\_edad, nuevo\_tipo\_membresia):

    conexion = conectar()

    cursor = conexion.cursor()

    consulta = "UPDATE clientes SET nombre = %s, edad = %s, tipo\_membresia = %s where id\_cliente = %s"

    cursor.execute(consulta, (nuevo\_nombre, nueva\_edad, nuevo\_tipo\_membresia, id\_cliente))

    conexion.commit()

    cursor.close()

    conexion.close()

def borrar\_cliente(id\_cliente):

    conexion = conectar()

    cursor = conexion.cursor()

    consulta = "delete from clientes where id\_cliente = %s"

    cursor.execute(consulta, (id\_cliente,))

    conexion.commit()

    consulta = "delete from inscripciones where id\_cliente = %s"

    cursor.execute(consulta, (id\_cliente,))

    conexion.commit()

    cursor.close()

    conexion.close()

actividades.py

from bdd import conectar

def crear\_actividad(nombre\_actividad, horario, duracion, id\_entrenador):

    conexion = conectar()

    cursor = conexion.cursor()

    consulta = "INSERT INTO actividades (nombre\_actividad, horario, duracion, id\_entrenador) VALUES (%s, %s, %s, %s)"

    cursor.execute(consulta, (nombre\_actividad, horario, duracion, id\_entrenador))

    conexion.commit()

    cursor.close()

    conexion.close()

def leer\_actividades():

    conexion = conectar()

    cursor = conexion.cursor()

    cursor.execute("select\*from actividades")

    actividades = cursor.fetchall()

    cursor.close

    conexion.close

    return actividades

def actualizar\_actividad(id\_actividad, nuevo\_nombre\_actividad, nuevo\_horario, nueva\_duracion, nuevo\_id\_entrenador):

    conexion = conectar()

    cursor = conexion.cursor()

    consulta = "UPDATE actividades SET nombre\_actividad = %s, horario = %s,  duracion = %s, id\_entrenador = %s WHERE id\_actividad = %s"

    cursor.execute(consulta, (nuevo\_nombre\_actividad, nuevo\_horario, nueva\_duracion, nuevo\_id\_entrenador, id\_actividad))

    conexion.commit()

    cursor.close()

    conexion.close()

def borrar\_actividad(id\_actividad):

    conexion = conectar()

    cursor = conexion.cursor()

    consulta = "delete from actividades where id\_actividad = %s"

    cursor.execute(consulta, (id\_actividad,))

    conexion.commit()

    consulta = "delete from inscripciones where id\_actividad = %s"

    cursor.execute(consulta, (id\_actividad,))

    conexion.commit()

    cursor.close()

    conexion.close()

entrenadores.py

from bdd import conectar

def crear\_entrenador(nombre\_entrenador, especialidad):

    conexion = conectar()

    cursor = conexion.cursor()

    consulta = "INSERT INTO entrenadores (nombre\_entrenador, especialidad) VALUES (%s, %s)"

    cursor.execute(consulta, (nombre\_entrenador, especialidad))

    conexion.commit()

    cursor.close()

    conexion.close()

def leer\_entrenadores():

    conexion = conectar()

    cursor = conexion.cursor()

    cursor.execute("select\*from entrenadores")

    entrenadores = cursor.fetchall()

    cursor.close

    conexion.close

    return entrenadores

def actualizar\_entrenador(id\_entrenador, nuevo\_nombre\_entrenador, nueva\_especialidad):

    conexion = conectar()

    cursor = conexion.cursor()

    consulta = "UPDATE entrenadores SET nombre\_entrenador = %s, especialidad = %s WHERE id\_entrenador = %s"

    cursor.execute(consulta, (nuevo\_nombre\_entrenador, nueva\_especialidad, id\_entrenador))

    conexion.commit()

    cursor.close()

    conexion.close()

def borrar\_entrenador(id\_entrenador):

    conexion = conectar()

    cursor = conexion.cursor()

    consulta = "delete from entrenadores where id\_entrenador = %s"

    cursor.execute(consulta, (id\_entrenador,))

    conexion.commit()

    consulta = "delete from actividades where id\_entrenador = %s"

    cursor.execute(consulta, (id\_entrenador,))

    conexion.commit()

    cursor.close()

    conexion.close()

inscripciones.py

from bdd import conectar

def crear\_inscripcion(id\_cliente, id\_actividad):

    conexion = conectar()

    cursor = conexion.cursor()

    consulta = "INSERT INTO inscripciones (id\_cliente, id\_actividad) VALUES (%s, %s)"

    cursor.execute(consulta, (id\_cliente, id\_actividad))

    conexion.commit()

    cursor.close()

    conexion.close()

def leer\_inscripciones():

    conexion = conectar()

    cursor = conexion.cursor()

    cursor.execute("select inscripciones.id\_inscripcion, clientes.\*, actividades.\* from (inscripciones inner join clientes) inner join actividades where inscripciones.id\_cliente = clientes.id\_cliente and inscripciones.id\_actividad = actividades.id\_actividad")

    clientes = cursor.fetchall()

    cursor.close

    conexion.close

    return clientes

def borrar\_inscripcion(id\_inscripcion):

    conexion = conectar()

    cursor = conexion.cursor()

    consulta = "delete from inscripciones where id\_inscripcion = %s"

    cursor.execute(consulta, (id\_inscripcion,))

    conexion.commit()

    cursor.close()

    conexion.close()

funciones.py

from clientes import crear\_cliente, leer\_clientes, actualizar\_cliente, borrar\_cliente

from actividades import crear\_actividad, leer\_actividades, actualizar\_actividad, borrar\_actividad

from entrenadores import crear\_entrenador, leer\_entrenadores, actualizar\_entrenador, borrar\_entrenador

from inscripciones import crear\_inscripcion, leer\_inscripciones, borrar\_inscripcion

def mostrar\_menu():

    print("\n=== Gestión del Centro Deportivo ===")

    print("1. Gestión de Clientes")

    print("2. Gestión de Actividades")

    print("3. Gestión de Entrenadores")

    print("4. Gestión de Inscripciones")

    print("5. Salir")

def menu\_clientes():

    print("\n=== Gestión de Clientes ===")

    print("1. Registrar nuevo cliente")

    print("2. Leer clientes existentes")

    print("3. Actualizar un cliente")

    print("4. Eliminar un cliente")

    print("5. Volver")

def menu\_actividades():

    print("\n=== Gestión de Actividades ===")

    print("1. Registrar nueva actividad")

    print("2. Leer actividades existentes")

    print("3. Actualizar una actividad")

    print("4. Eliminar una actividad")

    print("5. Volver")

def menu\_entrenadores():

    print("\n=== Gestión de Entrenadores ===")

    print("1. Registrar nuevo entrenador")

    print("2. Leer entrenadores existentes")

    print("3. Actualizar un entrenador")

    print("4. Eliminar un entrenador")

    print("5. Volver")

def menu\_inscripciones():

    print("\n=== Gestión de Inscripciones ===")

    print("1. Registrar nueva inscripción")

    print("2. Leer inscripciones existentes")

    print("3. Eliminar una inscripción")

    print("4. Volver")

def ejecutar\_opcion\_clientes(opcion):

    if opcion == 1:

        nombre = input("Ingrese el nombre del nuevo cliente: ")

        edad = int(input("Ingrese la edad del nuevo cliente: "))

        tipo\_membresia = input("Ingrese el tipo de membresía del nuevo cliente: ")

        crear\_cliente(nombre, edad, tipo\_membresia)

        print("Cliente registrado exitosamente.")

    elif opcion == 2:

        clientes = leer\_clientes()

        print("\nListado de Clientes:")

        for id\_cliente, nombre, edad, tipo\_membresia in clientes:

            print(f"ID: {id\_cliente}, Nombre: {nombre}, Edad: {edad} Membresía: {tipo\_membresia}")

    elif opcion == 3:

        id\_cliente = int(input("Ingrese el ID del cliente a actualizar: "))

        nuevo\_nombre = input("Ingrese el nuevo nombre del cliente: ")

        nueva\_edad = input("Ingrese la nueva edad del cliente: ")

        nuevo\_tipo\_membresia = input("Ingrese el nuevo tipo de membresía del cliente: ")

        actualizar\_cliente(id\_cliente, nuevo\_nombre, nueva\_edad, nuevo\_tipo\_membresia)

        print("Cliente actualizado exitosamente.")

    elif opcion == 4:

        id\_cliente = int(input("Ingrese el ID del cliente a eliminar: "))

        borrar\_cliente(id\_cliente)

        print("Cliente eliminado exitosamente.")

    elif opcion == 5:

        return

    else:

        print("Opción no válida. Intente de nuevo.")

def ejecutar\_opcion\_actividades(opcion):

    if opcion == 1:

        nombre\_actividad = input("Ingrese el nombre de la nueva actividad: ")

        horario = input("Ingrese el horario de la nueva actividad: ")

        duracion = int(input("Ingrese la duración de la nueva actividad: "))

        id\_entrenador = int(input("Ingrese el ID del entrenador de la nueva actividad: "))

        crear\_actividad(nombre\_actividad, horario, duracion, id\_entrenador)

        print("Actividad registrada exitosamente.")

    elif opcion == 2:

        actividades = leer\_actividades()

        print("\nListado de Actividades:")

        for id\_actividad, nombre\_actividad, horario, duracion, id\_entrenador in actividades:

            print(f"ID: {id\_actividad}, Nombre: {nombre\_actividad}, Horario: {horario} Duración: {duracion}, Entrenador: {id\_entrenador}")

    elif opcion == 3:

        id\_actividad = int(input("Ingrese el ID de la actividad a actualizar: "))

        nuevo\_nombre = input("Ingrese el nuevo nombre de la  actividad: ")

        nuevo\_horario = input("Ingrese el nuevo horario de la actividad: ")

        nueva\_duracion = input("Ingrese la nueva duración de la actividad: ")

        nuevo\_entrenador= input("Ingrese el nuevo entrenador de la actividad: ")

        actualizar\_actividad(id\_actividad, nuevo\_nombre, nuevo\_horario, nueva\_duracion, nuevo\_entrenador)

        print("Actividad actualizada exitosamente.")

    elif opcion == 4:

        id\_actividad = int(input("Ingrese el ID de la actividad a eliminar: "))

        borrar\_actividad(id\_actividad)

        print("Actividad eliminada exitosamente.")

    elif opcion == 5:

        return

    else:

        print("Opción no válida. Intente de nuevo.")

def ejecutar\_opcion\_entrenadores(opcion):

    if opcion == 1:

        nombre = input("Ingrese el nombre del nuevo entrenador: ")

        especialidad = input("Ingrese la especialidad del nuevo entrenador: ")

        crear\_entrenador(nombre, especialidad)

        print("Entrenador registrado exitosamente.")

    elif opcion == 2:

        entrenadores = leer\_entrenadores()

        print("\nListado de Entrenadores:")

        for id\_entrenador, nombre, especialidad in entrenadores:

            print(f"ID: {id\_entrenador}, Nombre: {nombre}, Especialidad: {especialidad}")

    elif opcion == 3:

        id\_entrenador = int(input("Ingrese el ID del entrenador a actualizar: "))

        nuevo\_nombre = input("Ingrese el nuevo nombre del entrenador: ")

        nueva\_especialidad = input("Ingrese la nueva especialidad del entrenador: ")

        actualizar\_entrenador(id\_entrenador, nuevo\_nombre, nueva\_especialidad)

        print("Entrenador actualizado exitosamente.")

    elif opcion == 4:

        id\_entrenador = int(input("Ingrese el ID del entrenador a eliminar: "))

        borrar\_entrenador(id\_entrenador)

        print("Entrenador eliminado exitosamente.")

    elif opcion == 5:

        return

    else:

        print("Opción no válida. Intente de nuevo.")

def ejecutar\_opcion\_inscripciones(opcion):

    if opcion == 1:

        id\_cliente = int(input("Ingrese el ID del cliente de la inscripción: "))

        id\_actividad = int(input("Ingrese el ID de la actividad de la inscripción: "))

        crear\_inscripcion(id\_cliente, id\_actividad)

        print("Inscripción registrada exitosamente.")

    elif opcion == 2:

        entrenadores = leer\_inscripciones()

        print("\nListado de Inscripciones:")

        for id\_inscripcion, id\_cliente, nombre, edad, tipo\_membresia, id\_actividad, nombre\_actividad, horario, duracion, id\_entrenador in entrenadores:

            print(f"ID: {id\_inscripcion}, ID cliente: {id\_cliente}, Cliente: {nombre}, Edad: {edad}, Membresía: {tipo\_membresia}, ID actividad: {id\_actividad}, Actividad: {nombre\_actividad}, Horario: {horario} Duración: {duracion}, Entrenador: {id\_entrenador}")

    elif opcion == 3:

        id\_inscripcion = int(input("Ingrese el ID de la inscripción a eliminar: "))

        borrar\_inscripcion(id\_inscripcion)

        print("Inscripción eliminada exitosamente.")

    elif opcion == 4:

        return

    else:

        print("Opción no válida. Intente de nuevo.")

main.py

import funciones

def main():

    continuar = True

    while continuar:

        funciones.mostrar\_menu()  # Mostrar el menú de opciones

        try:

            menu = int(input("Seleccione un menú: "))

            if menu == 1:

                mismo\_menu = True

                while mismo\_menu:

                    funciones.menu\_clientes()

                    opcion = int(input("Seleccione una opción: "))

                    funciones.ejecutar\_opcion\_clientes(opcion)

                    if opcion == 5:

                        mismo\_menu = False

            if menu == 2:

                mismo\_menu = True

                while mismo\_menu:

                    funciones.menu\_actividades()

                    opcion = int(input("Seleccione una opción: "))

                    funciones.ejecutar\_opcion\_actividades(opcion)

                    if opcion == 5:

                        mismo\_menu = False

            if menu == 3:

                mismo\_menu = True

                while mismo\_menu:

                    funciones.menu\_entrenadores()

                    opcion = int(input("Seleccione una opción: "))

                    funciones.ejecutar\_opcion\_entrenadores(opcion)

                    if opcion == 5:

                        mismo\_menu = False

            if menu == 4:

                mismo\_menu = True

                while mismo\_menu:

                    funciones.menu\_inscripciones()

                    opcion = int(input("Seleccione una opción: "))

                    funciones.ejecutar\_opcion\_inscripciones(opcion)

                    if opcion == 5:

                        mismo\_menu = False

            if menu == 5:

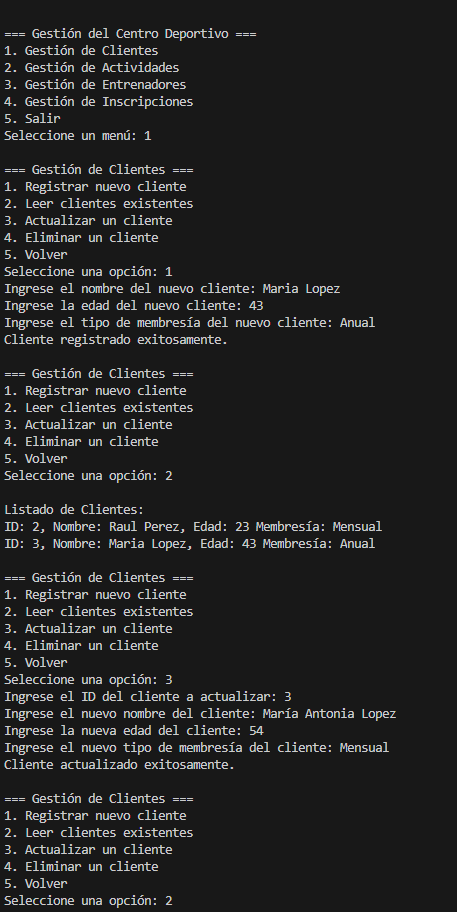
                break

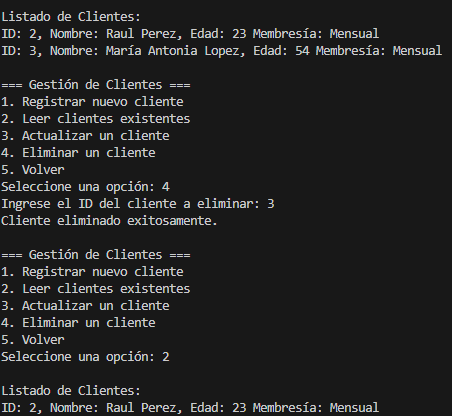
        except ValueError:

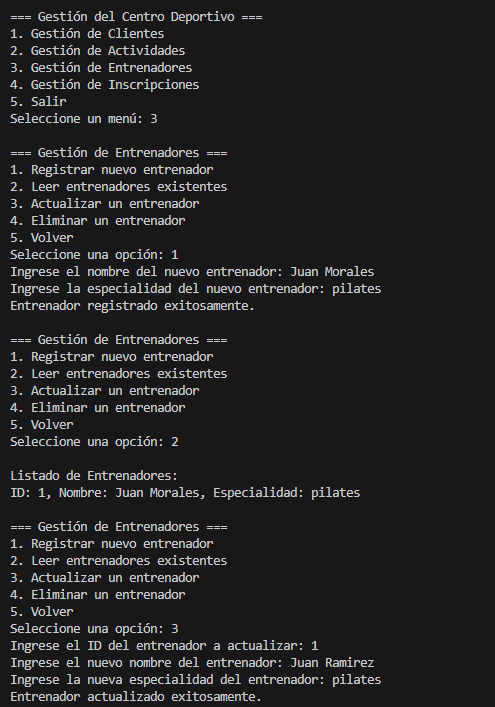
            print("Error: Debe ingresar un número válido.")

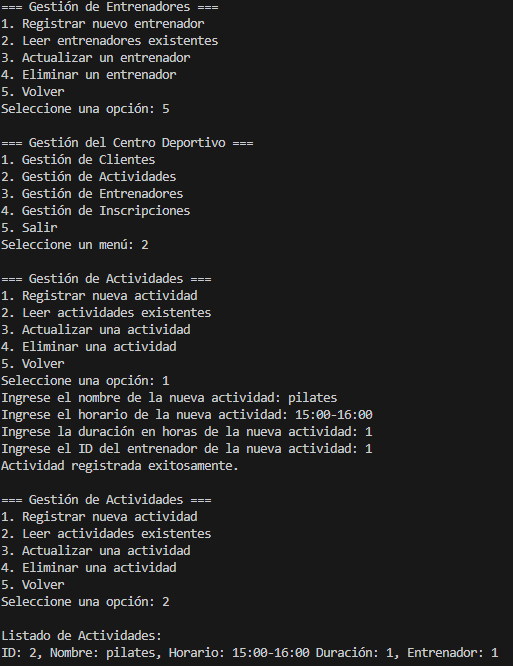
if \_\_name\_\_ == "\_\_main\_\_":

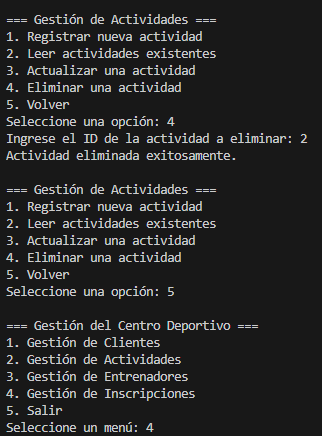
    main()

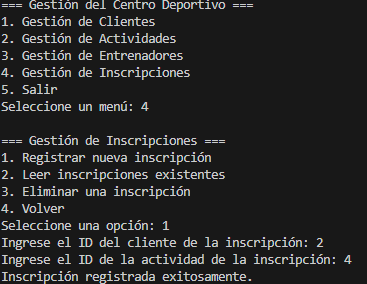


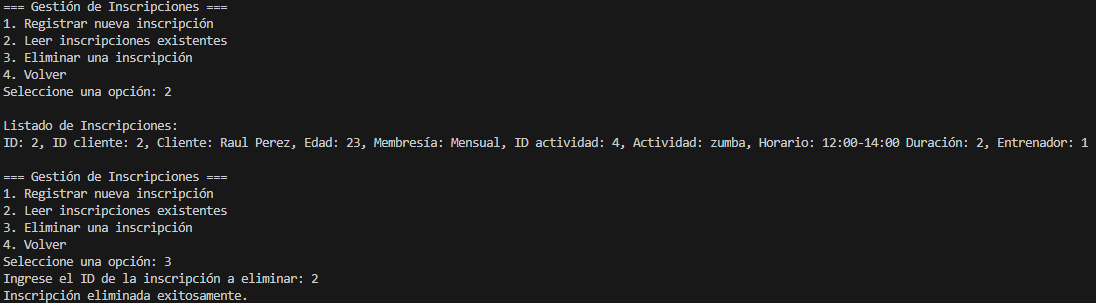


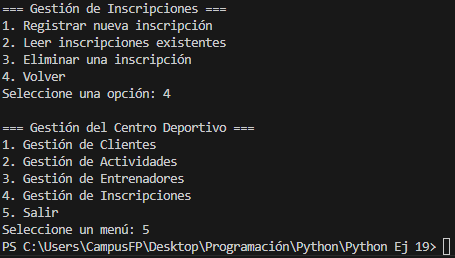












<https://github.com/Rodrigo-Garcia-Ortiz/Programacion.git>