

Laboratorio 2 Programación.

Integrantes:

David Gaviria Ruano

Julian David Ramirez Mayag



Profesores:

Alexander Parrado

Julian Dario Barrero

Universidad del Quindío

Facultad de ingeniería

Ingeniería Electrónica

Programación.

2024-02

Resumen:

El proyecto se centra en el desarrollo de un programa que simula trivia, este mismo gestiona usuarios y permite la participación en un juego de preguntas y respuestas. Este programa utiliza archivos para almacenar la información de los usuarios, incluyendo sus nombres, contraseñas y puntajes, y también importamos las preguntas y respuestas desde otros archivos creados manualmente.

Las principales funcionalidades incluyen el registro de nuevos usuarios, la gestión de sesiones (iniciar y cerrar sesión) y la actualización de puntajes según las respuestas del usuario en el juego. El programa verifica el registro de los usuarios para asegurar que solo aquellos registrados puedan acceder a sus datos y jugar.

Además, el sistema importa preguntas y respuestas desde archivos externos, organizadas por categorías, lo que permite un menor consumo de recursos y personalización en las preguntas presentadas a los usuarios.

Se realizó en 3 prácticas de laboratorio para el buen uso de archivos para la realización del proyecto final, en estos laboratorios se pone a prueba los conocimientos adquiridos en teoría.

También a mejorar el desempeño manejando los servidores, para importar información de entrada o salida.

Palabras claves: *Código, preguntas, importar, servidores, laboratorios, tuplas, archivos.*

Introducción:

Esta práctica de laboratorio se desarrolló con el propósito de implementar una versión del juego “Trivia”, permitiendo a los usuarios interactuar directamente con un servidor que pueda ser capaz de registrar usuarios, gestionar sesiones y puntajes, y proporcionar preguntas por cada categoría. El programa ofrece una interacción, donde las personas pueden responder preguntas, y se evalúa su desempeño para actualizar su puntaje al final de cada ronda.

Cada usuario podrá salir o entrar del juego con facilidad, también cuando acaba de responder preguntas puede seguir jugando eligiendo otra categoría.

Objetivo general:

Desarrollar un programa que simule un juego de trivia, capaz de gestionar usuarios y sesiones, almacenar y actualizar datos de manera eficiente, e importar preguntas y respuestas desde archivos externos.

Además de aplicar el uso de archivos, funciones y el correcto funcionamiento de un servidor.

Objetivos específicos:

- Implementar una versión funcional del juego de trivia, asegurando que cumpla con las especificaciones dadas, incluyendo la interacción con el usuario y la presentación de preguntas.
- Diseñar un sistema que permita almacenar y gestionar un conjunto de preguntas por categoría, asegurando que se mantenga un registro de las preguntas respondidas correctamente para evitar repeticiones.
- Asegurar que las preguntas de cada categoría sean variadas y cubran diferentes aspectos del tema, evitando que el juego se vuelva repetitivo.
- Que informe si un usuario ya está registrado, entrar y salir de la sesión.

Materiales

- Computador
- Laboratorios de robótica
- Software visual studio
- Asesorías dadas por los profesores
- Material de internet “videos, guías”
- GitHub

Procedimientos:

Este programa implementa una parte del juego “trivia” que gestiona usuarios y carga preguntas desde archivos externos. Se utilizó la librería “os” para manejar rutas de archivos y definir una estructura en la que las preguntas y respuestas se almacenan en un diccionario, organizado por categorías. La función “loadTrivia()” carga las preguntas y respuestas desde archivos de texto “lo que tiene como nombre archivos”, mientras que “manageUser()” permite registrar o iniciar sesión “manejar el uso de usuarios”, verificando si un usuario ya está registrado. Los datos de usuarios, que incluyen nombre, contraseña y respuestas correctas registrados, se gestionan con las funciones “loadUsers()” y “saveUsers()”, almacenándolos en un archivo “usuarios_trivia”. El juego permite a los usuarios seleccionar una categoría y responder preguntas de opción múltiple, las cuales se muestran con las funciones “askQuestion()” y “checkAnswer()”, que verifican si la respuesta es correcta. En cada ronda, la función “playRound()” actualiza los puntajes y guarda el progreso del usuario. Finalmente, “triviaGame()” gestiona el flujo general del juego, permitiendo múltiples rondas y categorías. El programa finaliza preguntando al usuario si desea seguir jugando o no, y actualiza los puntajes al terminar cada sesión, dando a conocer el desempeño de cada usuario.

Código funcional:

```

import os

# Ruta a la carpeta donde tienes los archivos

ruta_archivos = "C:\\Users\\JULIAN\\Desktop\\proyecto_trivia"

archivo_usuarios = os.path.join(ruta_archivos, "usuarios_trivia.txt")


# Lista para mantener los usuarios conectados

usuarios_conectados = set()


# Función para cargar preguntas y respuestas desde los archivos

def loadTrivia():

    trivia = {}

    with open(os.path.join(ruta_archivos, "preguntas_trivia.txt"), "r", encoding='utf-8') as
file_preguntas, \

        open(os.path.join(ruta_archivos, "respuestas_trivia.txt"), "r", encoding='utf-8') as
file_respuestas:

        categoria = None

        for pregunta_linea, respuesta_linea in zip(file_preguntas, file_respuestas):

            pregunta_linea = pregunta_linea.strip()

```

```

    respuesta_linea = respuesta_linea.strip()

    if pregunta_linea.startswith("#"): # Detectar nueva categoría

        categoria = pregunta_linea[1:].strip()

        trivia[categoria] = []

    elif categoria:

        pregunta, *opciones = pregunta_linea.split("|")

        trivia[categoria].append((pregunta, opciones, respuesta_linea.lower()))

    return trivia


# Función para registrar un nuevo usuario

def registerUser(usuarios):

    username = input("Crea tu nombre de usuario: ").strip()

    if username in usuarios:

        print("El usuario ya está registrado. Intenta iniciar sesión.")

        return None, usuarios

    password = input("Crea una contraseña: ").strip()

    usuarios[username] = {'password': password, 'correctas': 0, 'conectado': "conectado"}

    saveUsers(usuarios)

    print("Usuario registrado exitosamente.")

```



```
usuarios_conectados.add(username)
```

```
return username, usuarios
```

```
# Función para iniciar sesión
```

```
def loginUser(usuarios):
```

```
    username = input("Introduce tu nombre de usuario: ").strip()
```

```
    if username not in usuarios:
```

```
        print("El usuario no está registrado.")
```

```
        return None, usuarios
```

```
    password = input("Introduce tu contraseña: ").strip()
```

```
    if usuarios[username]['password'] == password:
```

```
        print(";Inicio de sesión exitoso! Bienvenido, " + username)
```

```
        usuarios[username]['conectado'] = "conectado" # Marcar como conectado
```

```
        saveUsers(usuarios)
```

```
        usuarios_conectados.add(username) # Añadir a usuarios conectados
```

```
        return username, usuarios
```

```
    else:
```

```
        print("Contraseña incorrecta. Inténtalo de nuevo.")
```

```
        return None, usuarios
```

Función para cargar usuarios desde el archivo

```
def loadUsers():
```

```
    usuarios = { }
```

```
    if os.path.exists(archivo_usuarios):
```

```
        with open(archivo_usuarios, "r", encoding='utf-8') as file:
```

```
            for linea in file:
```

```
                datos = linea.strip().split(",")
```

```
                if len(datos) == 3: # Si faltan datos de estado de conexión
```

```
                    username, password, correctas = datos
```

```
                    conectado = "desconectado" # Valor por defecto si no está presente
```

```
                elif len(datos) == 4: # Si todos los datos están presentes
```

```
                    username, password, correctas, conectado = datos
```

```
                else:
```

```
                    print(f"Error: formato incorrecto en la línea: {linea}")
```

```
                    continue
```

```
            usuarios[username] = {
```

```
                'password': password,
```

```
                'correctas': int(correctas),
```

```

        'conectado': conectado # Guardamos el estado de conexión directamente

    }

    return usuarios

# Función para guardar usuarios y sus datos

def saveUsers(usuarios):

    with open(archivo_usuarios, "w", encoding='utf-8') as file:

        for username, info in usuarios.items():

            file.write(f'{username},{info['password']},{info['correctas']},{info['conectado']}\n')

# Función para mostrar la pregunta y sus opciones

def askQuestion(pregunta_info):

    pregunta, opciones, _ = pregunta_info

    print(f'Pregunta: {pregunta}')

    letras = ['a', 'b', 'c', 'd']

    for i, opcion in enumerate(opciones):

        print(f'{letras[i]}. {opcion}')

# Función que verifica si la respuesta es correcta

```

```

def checkAnswer(pregunta_info, user_choice):

    _, opciones, respuesta_correcta = pregunta_info

    letras = ['a', 'b', 'c', 'd']

    try:

        opcion_seleccionada = opciones[letras.index(user_choice)].strip().lower()

        return respuesta_correcta == opcion_seleccionada

    except (IndexError, ValueError):

        return False


# Función para manejar una ronda de preguntas

def playRound(trivia, categoria, num_preguntas=10, username="", usuarios={}):

    preguntas = trivia[categoria]

    correctas = 0

    # Mezclamos las preguntas para que el orden varíe en cada ronda.

    preguntas = preguntas[:num_preguntas]

    for pregunta_info in preguntas:

        askQuestion(pregunta_info)

        user_choice = input("Elige una opción (a-d): ").lower()

        if checkAnswer(pregunta_info, user_choice):

```

```

        print("¡Correcto!")

        correctas += 1

    else:

        print("Respuesta incorrecta.")

# Actualizar las respuestas correctas del usuario

usuarios[username]['correctas'] += correctas

saveUsers(usuarios)

print(f"\nHas respondido correctamente {correctas} preguntas en esta ronda.")

print(f"Total de preguntas correctas de {username}:
{usuarios[username]['correctas']}")

# Función para ver usuarios conectados

def verUsuariosConectados():

    print("\n---Usuarios Conectados---")

    if usuarios_conectados:

        for user in usuarios_conectados:

            print(user)

    else:

        print("No hay usuarios conectados.")

```

```
# Función para cerrar sesión
```

```
def logout(username, usuarios):
```

```
    if username in usuarios_conectados:
```

```
        usuarios_conectados.remove(username)
```

```
        usuarios[username]['conectado'] = "desconectado" # Marcar como desconectado
```

```
        saveUsers(usuarios)
```

```
        print(f"{username} ha cerrado sesión.")
```

```
    else:
```

```
        print(f"{username} no está conectado.")
```

```
# Función principal del juego trivia
```

```
def triviaGame():
```

```
    trivia = loadTrivia()
```

```
    usuarios = loadUsers()
```

```
    username = None # Ningún usuario está conectado al principio
```

```
    while True:
```

```
        print("\nOpciones:")
```

```
        print("1. Iniciar sesión")
```

```
print("2. Registrar nuevo usuario")

print("3. Ver usuarios conectados")

print("4. Jugar trivia")

print("5. Cerrar sesión")

print("6. Salir")


opcion = input("Selecciona una opción: ").strip()


if opcion == '1':

    # Iniciar sesión

    username, usuarios = loginUser(usuarios)

    if username:

        # Si el usuario inicia sesión, pasa directamente a jugar

        categorias = list(trivia.keys())

        print("\nElige una categoría:")

        for i, cat in enumerate(categorias):

            print(f"{i+1}. {cat}")


    try:

        categoria_index = int(input("Introduce el número de la categoría: ")) - 1
```

```

    if 0 <= categoria_index < len(categorias):

        categoria = categorias[categoria_index]

        playRound(trivia, categoria, username=username, usuarios=usuarios)

    else:

        print("Número de categoría inválido")

except ValueError:

    print("Entrada no válida")


elif opcion == '2':

    # Registrar nuevo usuario

    username, usuarios = registerUser(usuarios)

    if username:

        # Si el usuario se registra, pasa directamente a jugar

        categorias = list(trivia.keys())

        print("\nElige una categoría:")

        for i, cat in enumerate(categorias):

            print(f"{i+1}. {cat}")

        try:

            categoria_index = int(input("Introduce el número de la categoría: ")) - 1

```



```

        if 0 <= categoria_index < len(categorias):

            categoria = categorias[categoria_index]

            playRound(trivia, categoria, username=username, usuarios=usuarios)

        else:

            print("Número de categoría inválido")

    except ValueError:

        print("Entrada no válida")

elif opcion == '3':

    # Ver usuarios conectados

    verUsuariosConectados()

elif opcion == '4':

    if username:

        categorias = list(trivia.keys())

        print("\nElige una categoría:")

        for i, cat in enumerate(categorias):

            print(f'{i+1}. {cat}')

    try:

```

```
categoria_index = int(input("Introduce el número de la categoría: ")) - 1

if 0 <= categoria_index < len(categorias):

    categoria = categorias[categoria_index]

    playRound(trivia, categoria, username=username, usuarios=usuarios)

else:

    print("Número de categoría inválido")

except ValueError:

    print("Entrada no válida")

else:

    print("Primero debes iniciar sesión o registrarte.")

elif opcion == '5':

    # Cerrar sesión

    if username:

        logout(username, usuarios)

        username = None

    else:

        print("No has iniciado sesión.")

elif opcion == '6':
```

```
print("¡Gracias por jugar!")
```

```
break
```

```
else:
```

```
print("Opción no válida, por favor selecciona de nuevo.")
```

```
# Ejecutar el trivia
```

```
triviaGame()
```

Pruebas unitarias y resultados:

Código con ingreso de usuarios

```
server.py

Opciones:
1. Iniciar sesión
2. Registrar nuevo usuario
3. Ver usuarios conectados
4. Jugar trivia
5. Cerrar sesión
6. Salir
Selecciona una opción: 1
Introduce tu nombre de usuario: ever
Introduce tu contraseña: 1234
¡Inicio de sesión exitoso! Bienvenido, ever
```

Código en funcionamiento: en este código se puede mirar el correcto funcionamiento y los resultados dados.

```
--Usuarios--
Introduce tu nombre de usuario: julian
El usuario 'julian' ya está registrado.
¿Deseas iniciar sesión con esta cuenta? (s/n): n
Por favor, introduce un nombre de usuario no registrado.
Introduce tu nombre de usuario: fernando
Crea una contraseña: 3178
Usuario registrado exitosamente.

Elige una categoría:
1. Deportes
2. Historia
Introduce el número de la categoría: 1
Pregunta: ¿Cuándo anotó su primer gol Messi?
a. 1 de mayo del 2005
b. 10 de junio de 2004
c. 15 de enero de 2007
d. 10 de junio de 2008
Elige una opción (a-d): a
¡Correcto!
Pregunta: ¿Cuántas copas mundiales tiene Pelé?
a. 5 mundiales
b. 2 mundiales
c. 1 mundial
d. 3 mundiales
Elige una opción (a-d): c
Respuesta incorrecta.
Pregunta: ¿Cuántos goles anotó Cristiano Ronaldo en toda su carrera?
a. 900
b. 850
c. 1000
d. 770
Elige una opción (a-d): █
```

Fig 1: código en funcionamiento.

En esta parte del código se puede observar una lista de usuarios con su respectiva contraseña y su su puntaje

```

≡ usuarios_trivia.txt
1  ever,1234,0
2  david,12345,0
3  julian,1089,4
4  fernando,3178,0
5

```

Fig 2: Usuarios almacenados en archivos.

En la siguiente imagen se presenta el archivo en el cual se encuentran almacenadas las preguntas en su respectiva categoría.

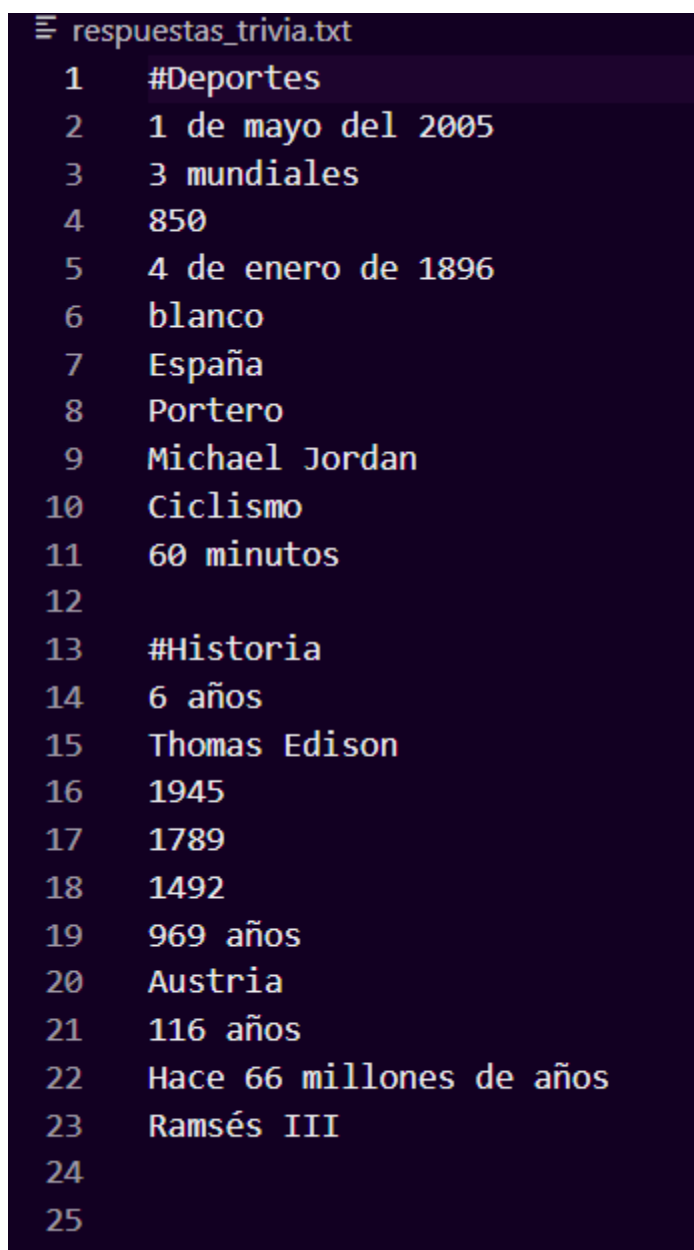
```

1  #Deportes
2  ¿Cuándo anotó su primer gol Messi?|1 de mayo del 2005|10 de junio de 2004|15 de enero de 2007|10 de junio de 2008
3  ¿Cuántas copas mundiales tiene Pelé?|5 mundiales|2 mundiales|1 mundial|3 mundiales
4  ¿Cuántos goles anotó Cristiano Ronaldo en toda su carrera?|900|850|1000|770
5  ¿La invención del voleibol fue en?|9 de febrero de 1885|4 de enero de 1896|30 de febrero de 1987|15 de abril de 1882
6  Un cubo de Rubik tiene caras en rojo, naranja, verde, amarillo, azul y... ¿Qué otro color?|morado|negro|blanco|gris
7  ¿Quién ganó el mundial de fútbol de 2010?|Francia|España|Brasil|Colombia
8  ¿En qué posición juega el cancerbero de un equipo de fútbol?|Lateral|Extremo|Portero|Delantero
9  ¿Quién se considera el mejor jugador de baloncesto de todos los tiempos?|Michael Ballack|Julian Smith|Kobe Bryant|Michael Jordan
10 ¿Qué tipo de competición es el Giro de Italia?|Golf|Ciclismo|Fútbol|Baloncesto
11 ¿Cuánto dura un partido de balonmano?|70 minutos|60 minutos|40 minutos|30 minutos
12
13 #Historia
14 ¿Cuántos años duró la Segunda Guerra Mundial?|6 años|3 años|4 años|10 años
15 ¿Quién inventó la bombilla?|Thomas Edison|Nikola Tesla|Alexander Graham Bell|Ever Kirchhoff
16 ¿Cuándo acabó la II Guerra Mundial?|1956|1930|1945|1940
17 ¿En qué año se produjo la Revolución Francesa?|1740|1789|1840|1850
18 ¿En qué año llegó Cristóbal Colón a América?|1500|1420|1492|1450
19 Según la Biblia, ¿cuántos años vivió Matusalén?|800 años|80 años|100 años|969 años
20 ¿En qué país nació Adolf Hitler?|colombia|Austria|Rusia|Inglaterra
21 ¿Cuánto duró 'La Guerra de los Cien Años'?|100 años|116 años|50 años|120 años
22 ¿Hace cuánto se extinguieron los dinosaurios?|Hace 66 millones de años|Hace 100 millones de años|Hace 40 años|Todas las anteriores
23 ¿Quién fue el último faraón de Egipto?|Gaviria II|Amosis I|Akenatón|Ramsés II
24
25

```

Fig 3: Preguntas almacenadas en Archivos.

En la siguiente imagen se presenta el archivo en el cual se encuentran almacenadas las respuestas en su respectiva categoría.



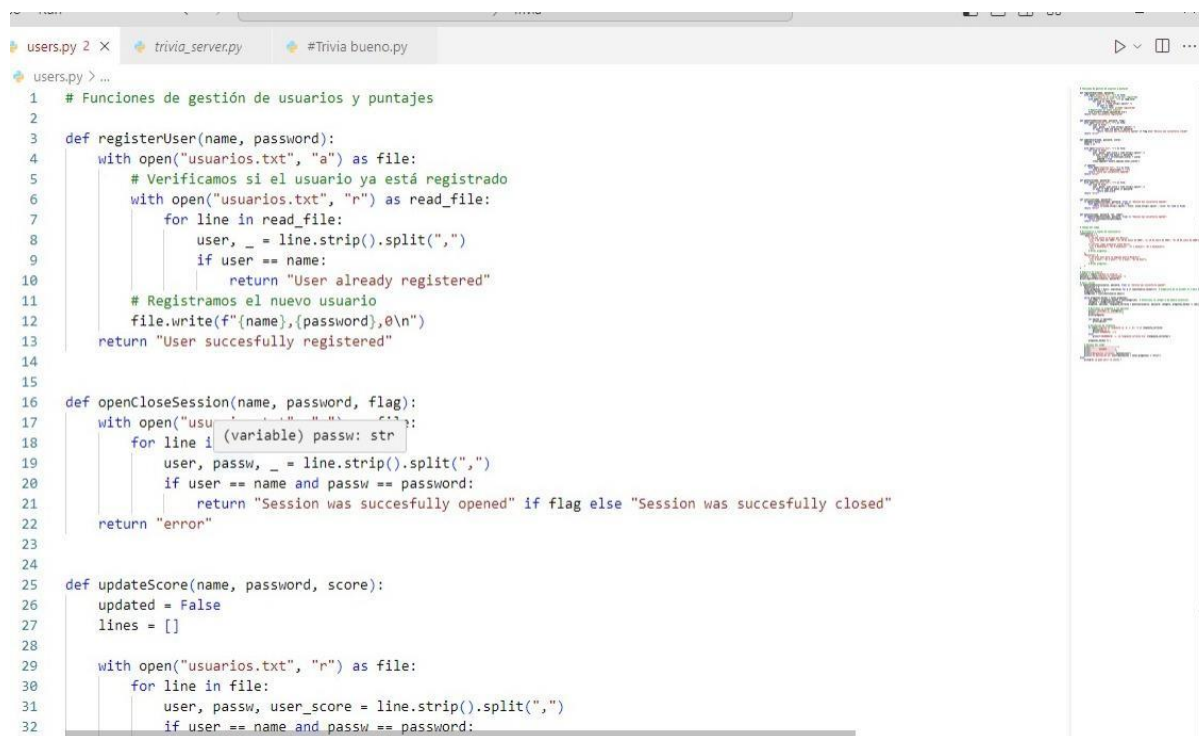
```
≡ respuestas_trivia.txt
1  #Deportes
2  1 de mayo del 2005
3  3 mundiales
4  850
5  4 de enero de 1896
6  blanco
7  España
8  Portero
9  Michael Jordan
10 Ciclismo
11 60 minutos
12
13 #Historia
14 6 años
15 Thomas Edison
16 1945
17 1789
18 1492
19 969 años
20 Austria
21 116 años
22 Hace 66 millones de años
23 Ramsés III
24
25
```

Fig 4: Respuestas almacenadas en archivos.

Pruebas de escritorio:

1. En la primera versión del código se realizó un código muy largo y deficiente donde tenía muchos errores y el consumo era excesivo de memoria.

Además, se agregaron los códigos presentados en los laboratorios anteriores los cuales estaban hechos con diccionarios y no ayudaron a mejorar su eficiencia.



```

users.py 2 X  trivia_server.py  #Trivia bueno.py
users.py > ...
1  # Funciones de gestión de usuarios y puntajes
2
3  def registerUser(name, password):
4      with open("usuarios.txt", "a") as file:
5          # Verificamos si el usuario ya está registrado
6          with open("usuarios.txt", "r") as read_file:
7              for line in read_file:
8                  user, _ = line.strip().split(",")
9                  if user == name:
10                     return "User already registered"
11             # Registramos el nuevo usuario
12             file.write(f"{name},{password}\n")
13             return "User succesfully registered"
14
15
16 def openCloseSession(name, password, flag):
17     with open("usuarios.txt", "r") as file:
18         for line i (variable) passw: str:
19             user, passw, _ = line.strip().split(",")
20             if user == name and passw == password:
21                 return "Session was succesfully opened" if flag else "Session was succesfully closed"
22     return "error"
23
24
25 def updateScore(name, password, score):
26     updated = False
27     lines = []
28
29     with open("usuarios.txt", "r") as file:
30         for line in file:
31             user, passw, user_score = line.strip().split(",")
32             if user == name and passw == password:

```

Fig 5: Primera versión del código.

2. Código modificado con ayuda del docente donde se resolvieron dudas del código anterior, como se puede observar aún no compilaba el código.

The screenshot shows a Python IDE with two panels. The top panel displays the code for `users.py` and `question`. The bottom panel shows a traceback error.

```

users.py > question
71 def usersList(name, password):
72     if openCloseSession(name, password, True) == "Session was successfully opened":
73         with open("usuarios.txt", "r") as file:
74             return [f"{line.strip().split(',')[0]}: {line.strip().split(',')[2]}" for line in file if line.strip()]
75     return "error"
76
77 # Las preguntas deben estar en archivos, un archivo por categoria
78 def question(name, password, cat, index):
79     #if openCloseSession(name, password, True) == "Session was succesfully opened":
80     #    return cuestionario[cat][index]
81     #return "error"
82     pass

```

The traceback error in the bottom panel is as follows:

```

Traceback (most recent call last):
  File "C:\Python\Lib\site-packages\requests\sessions.py", line 589, in request
    resp = self.send(prepared_request, **send_kwargs)
  File "C:\Python\Lib\site-packages\requests\sessions.py", line 703, in send
    r = adapter.send(request, **kwargs)
  File "C:\Python\Lib\site-packages\requests\adapters.py", line 700, in send
    raise ConnectionError(e, request=request)
requests.exceptions.ConnectionError: HTTPConnectionPool(host='localhost', port=80): Max retries exceeded with url: /register (Caused by NewConnectionError('<urllib3.connection.HTTPConnection object at 0x000001A04A257320>: Failed to establish a new connection: [WinError 10061] No se puede establecer una conexión ya que el equipo de destino denegó expresamente dicha conexión'))

```

[Done] exited with code=1 in 5.256 seconds

Fig 6: Segunda versión del código.

3. En la última prueba de escritorio se contaba con un código que ya compilaba, pero su uso de memoria era ineficiente, por ejemplo: Import random es una función que consume muchos recursos, fue removida para optimizarlo en su versión final.



```
users.py > ...
1 import os
2 import random
3
4 # Ruta a la carpeta donde tienes los archivos
5 ruta_archivos = "C:\\Users\\JULIAN\\Desktop\\proyecto_trivia"
6
7 # Función para cargar preguntas y respuestas desde los archivos
8 def loadTrivia():
9     trivia = {}
10     with open(os.path.join(ruta_archivos, "preguntas_trivia.txt"), "r", encoding='utf-8') as file_preguntas, \
11         open(os.path.join(ruta_archivos, "respuestas_trivia.txt"), "r", encoding='utf-8') as file_respuestas:
12
13         categoria = None
14         for pregunta_linea, respuesta_linea in zip(file_preguntas, file_respuestas):
15             pregunta_linea = pregunta_linea.strip()
16             respuesta_linea = respuesta_linea.strip()
17             if pregunta_linea.startswith("#"): # Detectar nueva categoría
18                 categoria = pregunta_linea[1:].strip()
19                 trivia[categoria] = []
20             elif categoria:
21                 pregunta, *opciones = pregunta_linea.split("|")
22                 trivia[categoria].append({
23                     "pregunta": pregunta,
24                     "opciones": opciones,
25                     "respuesta": respuesta_linea.lower()
26                 })
27     return trivia
28
29 # Función que muestra la pregunta y sus opciones
30 def askQuestion(pregunta_info):
31     print(f"Pregunta: {pregunta_info['pregunta']}")
32     letras = ['a', 'b', 'c', 'd']
```

Fig 7: Tercera versión del código.

Conclusiones:

- El uso de archivos externos para almacenar preguntas y respuestas permite una mayor personalización del contenido y reduce la necesidad de modificar el código para agregar nuevas preguntas.
- Actualización automática de los puntajes, esto ofrece mayor conformidad para que el usuario tenga control de su puntuación.
- El proyecto enseña las buenas prácticas en el manejo de archivos, lo cual es crucial para desarrollar programas grandes que puedan gestionar datos de forma eficiente.
- La capacidad del programa de importar datos de manera eficiente, importando datos es una manera muy eficiente para poder reducir el consumo de recursos y el manejo de códigos llega a ser menos tedioso

Referencias bibliográficas

- C8d, 6afo 53fo 7d9d. (s/f). *Download Python*. Python.org. Recuperado el 24 de octubre de 2024, de <https://www.python.org/downloads/>
- López-Parrado, A. (s/f). *Lab2*.