Laboratorio No 1. Lógica de programación

Integrantes

Catalina Arcila Grisales

Angela María Bolaños Moncayo

Juan Camilo Poveda Delgado

Ingeniería electrónica

Programación

Profesor

Alexander López - Parrado

24 de septiembre del 2024

Laboratorio 1, Lógica de programación

Resumen- En el siguiente documento se presenta la implementación de un juego estilo "Preguntados" en Python, utilizando listas y diccionarios para almacenar preguntas y respuestas. El juego selecciona preguntas aleatoriamente y evalúa las respuestas del usuario, actualizando la puntuación y eliminando preguntas respondidas para optimizar el uso de memoria. El sistema incluye validaciones para asegurar que el usuario seleccione respuestas válidas y permite finalizar el juego en cualquier momento. Finalmente, se analizan los conceptos aprendidos y aplicados, como el uso de estructuras de datos, manejo de memoria, y la importancia de un flujo lógico en el diseño de programas interactivos.

Palabras clave - Diccionarios, sentencias condicionales, listas, memoria, Python.

I. INTRODUCCIÓN

En este documento se demuestra el uso de los conocimientos y habilidades adquiridas durante el proceso de aprendizaje en programación con el lenguaje Python, esto, para aplicarlo en un nuevo reto, usando arreglos de datos, sentencias condicionales e iteraciones, donde el docente, además introduce nuevas funciones de Python como lo son los diccionarios. Esto, con la finalidad de comprender y aprender a utilizar correctamente el lenguaje Python.

A continuación, se presenta el reto que pide crear un juego al estilo de preguntados, donde se dará una explicación más profunda sobre su funcionamiento, además de la solución que se ofrece para este juego, reflejando el uso y aprendizaje de lo enseñado en clases.

II. OBJETIVOS

Objetivo general:

Analizar el problema estipulado por el docente para hallar la solución más eficaz mediante conocimientos anteriormente adquiridos, además de aprender a manejarlos en un lenguaje introducido anteriormente llamado Python.

Objetivos específicos:

- Fortalecer el pensamiento lógico.
- Comprender el funcionamiento de diferentes funciones en Python.
- Comprender el funcionamiento de diccionarios, listas y arreglos en Python.
- Trabajar en equipo para hallar soluciones.

III. MÉTODOS E INSTRUMENTOS

A continuación, la solución al ejercicio propuesto por el docente en Python con su respectivo diagrama de flujo:

El reto de esta práctica de laboratorio consiste en implementar una **versión reducida para terminal del juego Preguntados (*Trivia Crack*)** (https://en.wikipedia.org/wiki/Trivia_Crack). En ese sentido, se espera que el juego soporte las siguientes características:

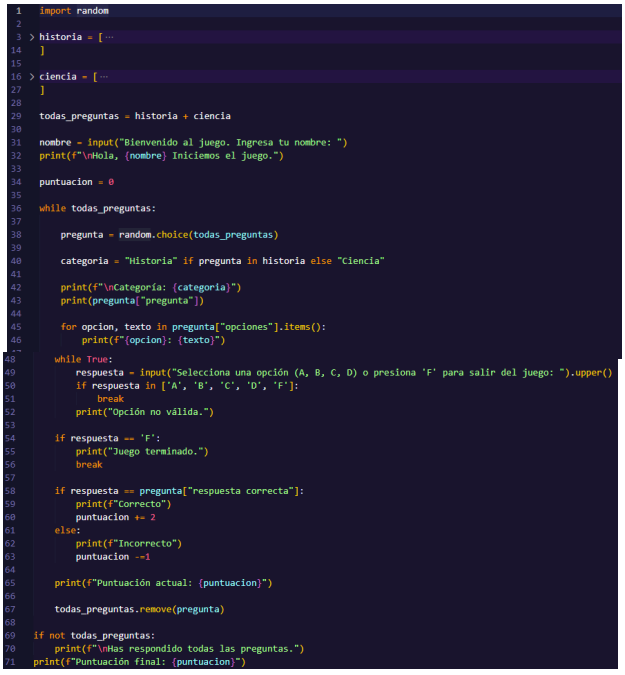
- Un único jugador.
- Dos categorías de preguntas.
- 10 preguntas por categoría.

Una vez se inicie el programa, se le debe solicitar al usuario su nombre, el cual deberá ser usado durante el progreso en el juego. Posteriormente, el programa generará una categoría de forma aleatoria y a continuación desplegará la pregunta aleatoria, junto con las opciones de respuesta. A continuación, el usuario deberá seleccionar su opción de respuesta.

El mecanismo de puntuación y penalización del usuario es de libre elección por parte del equipo de trabajo, en todo caso el usuario debe poder finalizar el juego en cualquier momento. Si el usuario ha respondido correctamente todas las preguntas de las dos categorías, el programa debe finalizar con un mensaje de felicitación para el usuario.

Fig. 1. Ejercicio, juego preguntados.

Para este ejercicio se pide crear un juego al estilo de preguntados. Inicialmente, se propone el siguiente código en Python:



```python
import random

historia = [...]

ciencia = [...]

todas_preguntas = historia + ciencia

nombre = input("Bienvenido al juego. Ingresa tu nombre: ")
print(f"\nHola, {nombre} Iniciemos el juego.")

puntuacion = 0

while todas_preguntas:

    pregunta = random.choice(todas_preguntas)

    categoria = "Historia" if pregunta in historia else "Ciencia"

    print(f"\nCategoría: {categoria}")
    print(pregunta["pregunta"])

    for opcion, texto in pregunta["opciones"].items():
        print(f"{opcion}: {texto}")

    while True:
        respuesta = input("Selecciona una opción (A, B, C, D) o presiona 'F' para salir del juego: ").upper()
        if respuesta in ['A', 'B', 'C', 'D', 'F']:
            break
        print("Opción no válida.")

    if respuesta == 'F':
        print("Juego terminado.")
        break

    if respuesta == pregunta["respuesta correcta"]:
        print(f"Correcto")
        puntuacion += 2
    else:
        print(f"Incorrecto")
        puntuacion -=1

    print(f"Puntuación actual: {puntuacion}")

    todas_preguntas.remove(pregunta)

if not todas_preguntas:
    print(f"\nHas respondido todas las preguntas.")
print(f"Puntuación final: {puntuacion}")
```

Fig. 1.1. Código del ejercicio 1 en Python.

Para empezar, se arman dos listas de diccionarios donde se almacenarán: “pregunta”, “opciones” y “respuesta correcta”. A continuación, el programa solicita al jugador ingresar su nombre y luego lo saluda (línea 31), esto, para que se sienta en un ambiente más familiar al de un juego, consiguiente a esto, el programa se ejecutara teniendo en cuenta las dos listas anteriormente mencionadas (historia y ciencia). Estas listas se combinan en “todas_preguntas” (línea 29), que contiene todas las preguntas disponibles.

Relacionado a lo anterior, el programa utiliza un bucle while (línea 36), que se ejecuta mientras haya preguntas disponibles en todas_preguntas. Dentro de este mismo se selecciona una pregunta al azar con random.choice(). Se muestra la pregunta junto con sus opciones de respuesta y se solicita al usuario que seleccione una opción (A, B, C, D) o que presione 'F' para salir del juego. Si el usuario responde con 'F', el juego termina, de lo contario, se verifica si la respuesta ingresada coincide con la respuesta correcta. Si es correcta, se incrementa 2 puntos a la puntuación; si no, se muestra que es incorrecta y resta 1 punto a la puntuación. A medida que el usuario va respondiendo más preguntas, se actualiza la puntuación y se elimina la pregunta utilizada de la lista todas_preguntas para que no vuelva a aparecer.

Finalmente, el juego termina cuando el usuario ha respondido todas las preguntas mostrando en pantalla el puntaje final. Por último, el diagrama de flujo:

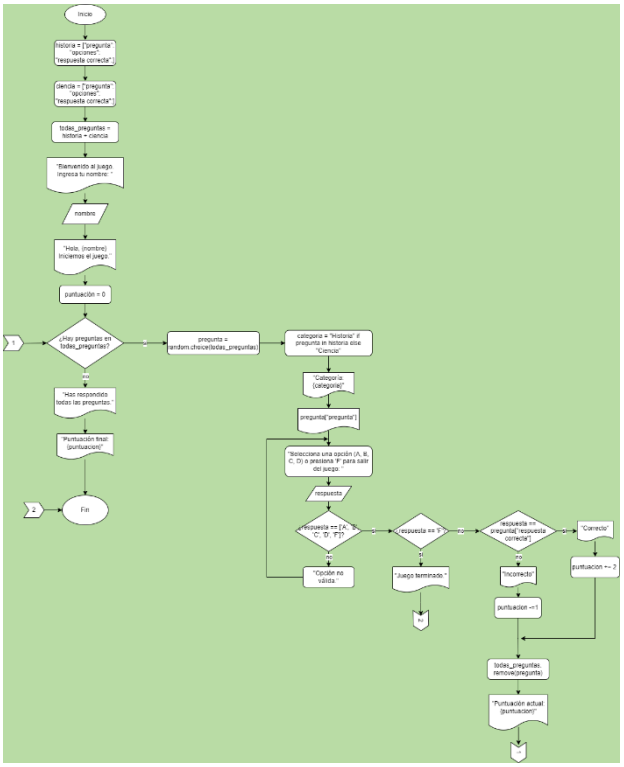


Fig 1.2. Diagrama de flujo del ejercicio 1.

Uso del espacio en memoria:

Cada pregunta se almacena en un diccionario, y las preguntas están organizadas en dos listas: historia y ciencia. Luego, estas listas se combinan en una sola (todas_preguntas). Aunque se podría optimizar el uso de memoria reduciendo el número de listas, se decidió hacerlo de esta forma para que el código sea más legible.

Por otro lado, después de que el jugador responde una pregunta, esta se elimina de la lista todas_preguntas usando todas_preguntas.remove(pregunta). Esto libera espacio de memoria al eliminar elementos que ya no son necesarios.

CONCLUSIONES

1- El uso de random.choice() para seleccionar una pregunta al azar y la eliminación posterior de la misma con remove() garantiza que las preguntas no se repitan, lo cual es ideal para un juego de preguntas.

2- Los diccionarios permiten organizar cada pregunta junto con sus opciones y la respuesta correcta de manera clara y estructurada. Esto facilita el acceso a los datos durante el juego, ya que el programa puede extraer tanto la pregunta como las opciones y verificar la respuesta de manera eficiente

3- El juego sigue un flujo lógico simple basado en un ciclo de preguntas aleatorias, lo que permite una interacción dinámica y eficiente con el usuario. El bucle garantiza que el jugador responda todas las preguntas disponibles o tenga la opción de salir en cualquier momento. Además de implementar validaciones básicas para evitar entradas no válidas, garantizando que el usuario solo pueda seleccionar las opciones permitidas o salir del juego, lo que mejora la robustez del programa y previene errores de ejecución.