¿Quién quiere ser millonario?

1. **INTRODUCCIÓN**
   El objetivo de este proyecto es crear un juego por consola inspirado en **"Quién quiere ser millonario"**. El jugador tendrá que responder hasta 15 preguntas, organizadas en 3 niveles de dificultad.
   Para ello, utilizarás diccionarios, funciones, excepciones personalizadas, y listas para manejar bancos de preguntas.
2. **OBJETIVOS GENERALES**

   Se debe desarrollar un juego completo y funcional que:

   - Muestre un menú inicial que permita ver las instrucciones, jugar o salir.
   - Haga preguntas al usuario según diferentes niveles de dificultad
   - Permita plantarse antes de responder
   - Maneje errores mediante excepciones personalizadas durante la ejecución
   - Utilice diccionarios para almacenar preguntas
   - Utilice funciones para dividir el programa en partes organizadas.
3. **REQUISITOS GENERALES**
   - El código debe estar escrito en Python 3.
   - El código debe estar organizado en varias funciones. **No se permite escribir todo en el bloque principal.**
   - No se permitirá usar Programación Orientada a Objetos (creación de clases), salvo la definición de excepciones personalizadas.
   - Se deben usar diccionarios para almacenar las preguntas.
4. **ESTRUCTURA DEL PROGRAMA**

   Debes utilizar las siguientes **excepciones**:

   OpcionMenuInvalida(Exception): se lanza cuando se elige una opción inválida en el menú.

   RespuestaInvalida(Exception): se lanza cuando se pone una respuesta inválida al responder una pregunta.

   Debes utilizar las siguientes **funciones**:
   mostrar_menu()
   Muestra el menú principal con las opciones: 1. Jugar, 2. Instrucciones y 3. Salir.
   leer_opcion_menu()
   Pide al usuario una opción. Si no es 1, 2 o 3, se lanza la excepción personalizada *OpcionMenuInvalida*.

mostrar_instrucciones()

Explica al jugador cómo funciona el juego:

- Hay 15 preguntas
- Se dividen por dificultad (1-5: fácil, 6-10: media, 11-15: difícil)
- Se pueden plantar escribiendo P
- Se debe responder con A, B, C o D
- Si aciertas subes el premio
- Si fallas pierdes y te quedas con el premio del último nivel seguro superado
- Hay niveles seguros (pregunta 5 y 10)

crear_preguntas()

Genera tres listas: preguntas_facil, preguntas_media y preguntas_dificil. Cada lista contiene 5 preguntas almacenadas en diccionarios con enunciado, opciones (su valor es otro diccionario con las claves A, B, C y D) y respuesta correcta.

obtener_pregunta_por_numero()

Dado un número de pregunta del 1 al 15, devuelve la pregunta adecuada eliminándola de la lista correspondiente.

mostrar_pregunta()

Muestra el número de pregunta, el premio actual, el premio en juego, el enunciado y las opciones A/B/C/D.

leer_respuesta()

Lee la respuesta del usuario (A, B, C, D o P). Si es incorrecta, lanza la excepción *RespuestaInvalida*.

jugar_partida()

Contiene la lógica del juego: lectura de preguntas, cálculo de premios, niveles seguros, fallos y plantas.

main()

Gestiona el programa principal, llama al menú y controla errores con excepciones.

5. PRODUCTO FINAL ESPERADO

El alumno debe entregar un fichero .py funcional con todas las funciones implementadas, las 15 preguntas inventadas por él y el manejo correcto de listas, diccionarios y excepciones.

El programa debe permitir jugar completamente, llegar hasta la pregunta 15 y manejar los posibles errores cometidos por el usuario.