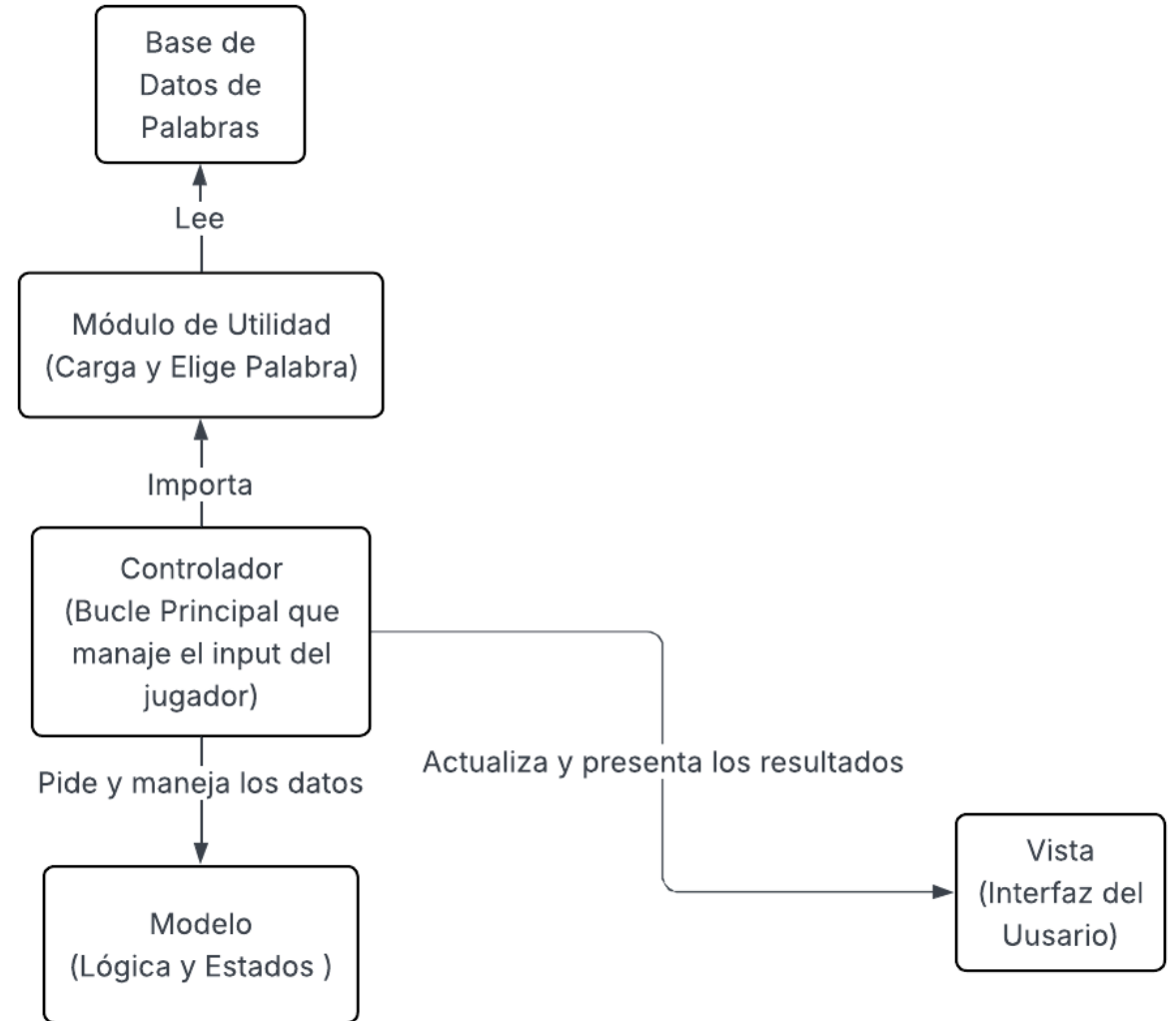
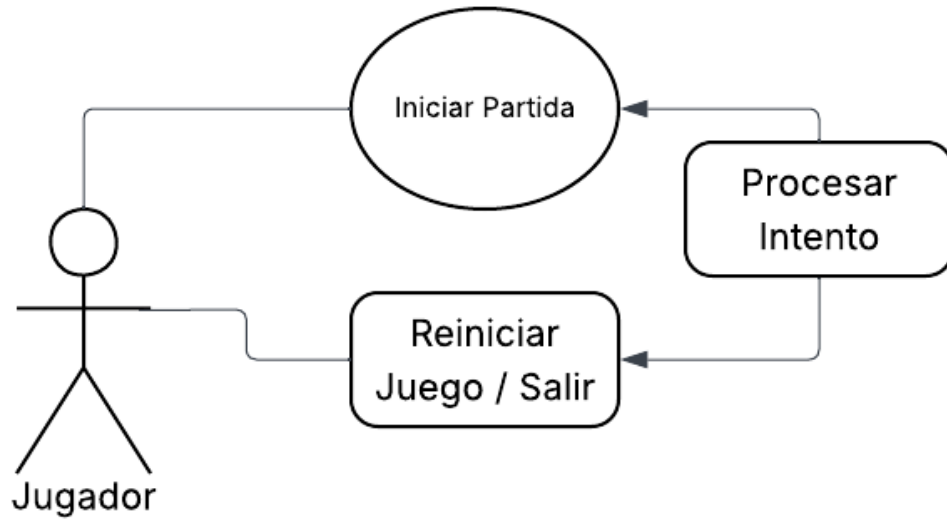


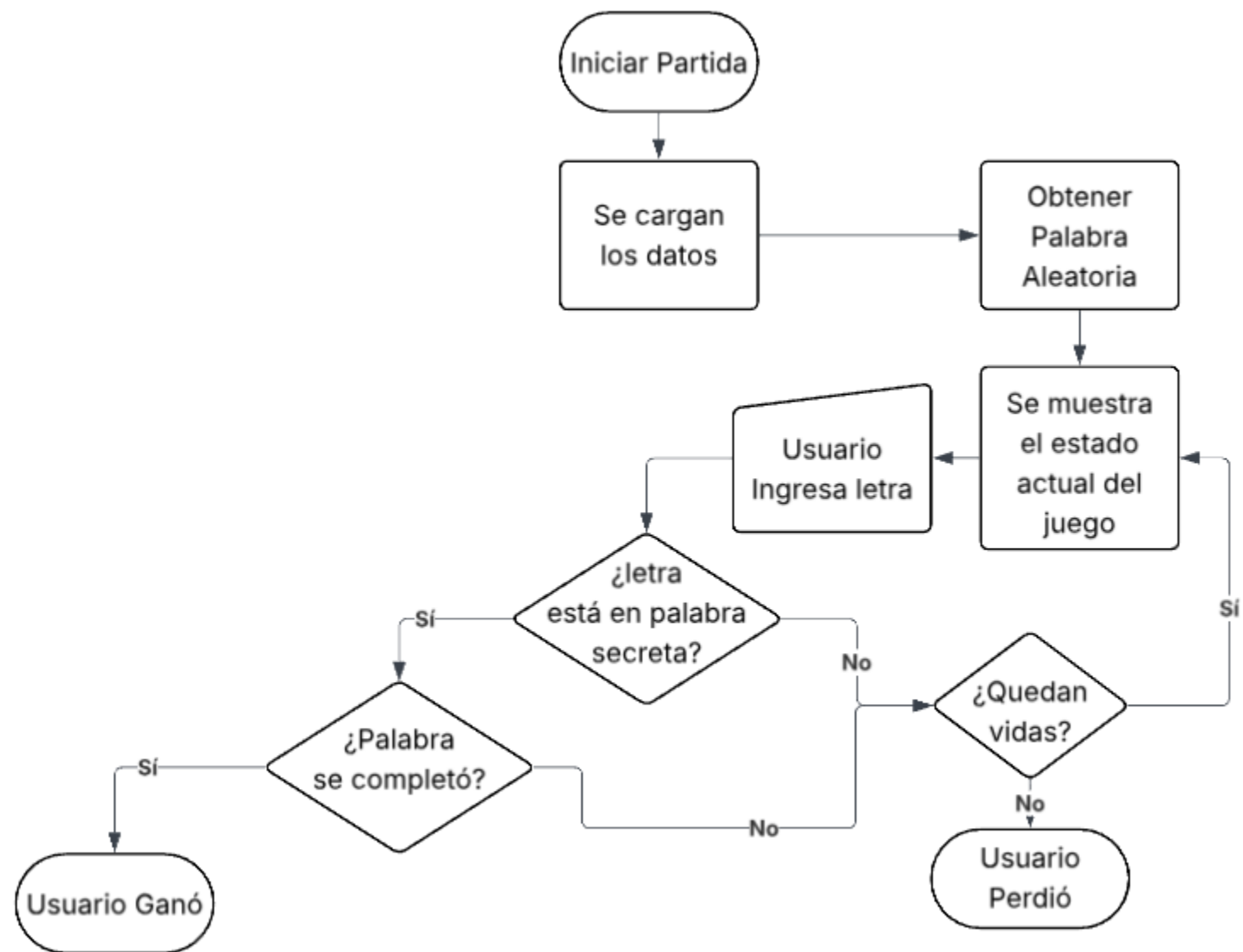


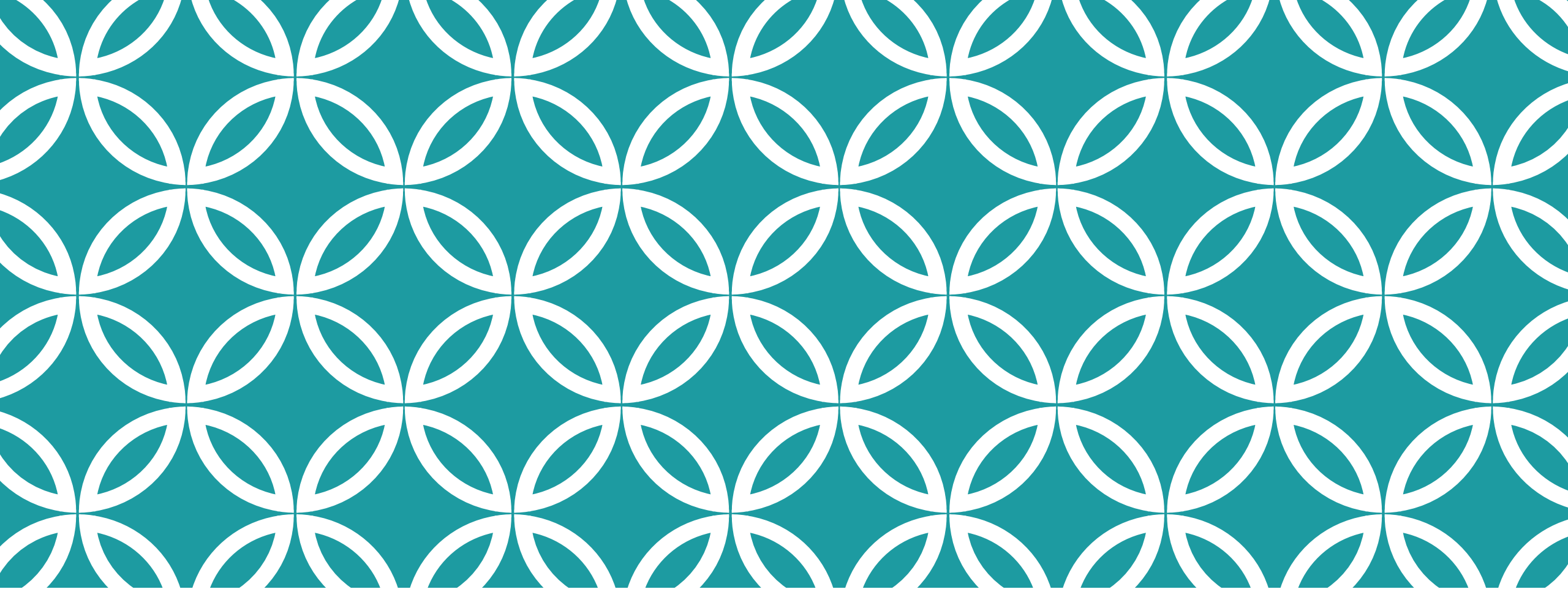
JUEGO DEL AHORCADO

by Ronald Vera

DIAGRAMAS








IMPLEMENTACIÓN

datos.py
logica.py
main.py



```
1  # Lista de palabras para el juego.
2  PALABRAS = ['PYTHON', 'PROGRAMACION', 'AHORCADO', 'JUEGO', 'INMUTABLE']
3
4  # Diccionario que mapea el número de vidas restantes al arte ASCII.
5  # Clave (int): Vidas | Valor (str): Dibujo
6  AHORCADO_DIBUJOS = {
7      6: "",
8      5: " 0",
9      4: " 0\n|",
10     3: " 0\n/|",
11     2: " 0\n/|\\",
12     1: " 0\n/|\\\\n/",
13     0: " 0\n/|\\\\n/  \\"
14 }
```



```
1  import random
2
3  def obtener_palabra_random(lista_palabras):
4      """
5      Elige una palabra al azar.
6      """
7      return random.choice(lista_palabras).upper()
8
9  def formatear_palabra_secreta(palabra_secreta, letras_adivinadas):
10     """
11     Mapea la palabra secreta a una cadena visible.
12     Si la letra está en 'letras_adivinadas', la muestra. Si no, pone '_'.
13     """
14     letras_visibles = [letra if letra in letras_adivinadas else '_' for letra in palabra_secreta]
15     return ' '.join(letras_visibles)
16
17  def obtener_estado_juego(palabra_secreta, letras_adivinadas, vidas):
18     """
19     Analiza si el juego ha terminado.
20     Retorna: 'GANO', 'PERDIO', o 'JUGANDO'
21     """
22     # Usamos conjuntos (sets) para verificar matemáticamente si
23     # todas las letras de la palabra están dentro de las adivinadas.
24     if set(palabra_secreta).issubset(letras_adivinadas):
25         return 'GANO'
26
27     if vidas <= 0:
28         return 'PERDIO'
29
30     return 'JUGANDO'
```

```
1 from datos import PALABRAS, AHORCADO_DIBUJOS
2 from logica import obtener_palabra_random, formatear_palabra_secreta, obtener_estado_juego
3
4 def iniciar_juego():
5     # --- CONFIGURACIÓN INICIAL DEL ESTADO ---
6     palabra_secreta = obtener_palabra_random(PALABRAS)
7     vidas = 6
8     letras_adivinadas = set() # Conjunto vacío para guardar letras
9
10    print("--- BIENVENIDO AL JUEGO DEL AHORCADO ---")
11
12    # --- BUCLE PRINCIPAL DEL JUEGO ---
13    while True:
14
15        # 1. MOSTRAR LA "VISTA" AL USUARIO
16        print("\n" + "="*30)
17        print(f"Vidas restantes: {vidas}")
18        print(AHORCADO_DIBUJOS.get(vidas, "X"))
19        print("Palabra: " + formatear_palabra_secreta(palabra_secreta, letras_adivinadas))
20
21        # 2. VERIFICAR EL ESTADO DEL JUEGO
22        estado = obtener_estado_juego(palabra_secreta, letras_adivinadas, vidas)
23
24        if estado == 'GANO':
25            print(f"\n¡Felicidades! Adivinaste la palabra: {palabra_secreta}")
26            break # Rompemos el ciclo while (se adivino la palabra)
27
28        if estado == 'PERDIO':
29            print(f"\n¡Game Over! La palabra era: {palabra_secreta}")
30            break # Rompemos el ciclo while (ya no hay vidas)
31
32        # 3. ENTRADA DE DATOS (INPUT)
33        intento = input("Adivina una letra: ").upper().strip()
34
35        # 4. VALIDACIÓN
36        if not intento or len(intento) > 1 or not intento.isalpha():
37            print("Entrada inválida. Debe ser una sola letra.")
38            continue # Saltamos al inicio del while sin cambiar vidas ni letras
39
40        if intento in letras_adivinadas:
41            print(f"Ya habías intentado con '{intento}'.")
42            continue # Vuelve al inicio (la letra ingresada ya se usó)
43
44        # 5. ACTUALIZACIÓN DEL ESTADO
45        # Agregamos la letra al conjunto
46        letras_adivinadas.add(intento)
47
48        if intento not in palabra_secreta:
49            print(f"¡Mala suerte! '{intento}' no está en la palabra.")
50            vidas -= 1 # Reducimos la vida
51        else:
52            print(f"¡Bien hecho! '{intento}' es correcta.")
53
54    if __name__ == "__main__":
55        iniciar_juego()
```