

3. **Ahorcado (Hangman):** Implementación de la clásica adivinanza de palabras.

¡Claro! Aquí tienes una implementación simple del juego del **Ahorcado** en **Python**.

Este juego selecciona una palabra al azar de una lista y te permite adivinar letras para descubrirla antes de que se agoten tus intentos.

Python

```
import random
```

```
## ? Implementación del Juego del Ahorcado
```

```
def jugar_ahorcado():
```

```
    """
```

```
        Función principal para ejecutar el juego del Ahorcado.
```

```
    """
```

```
# Lista de palabras para el juego
```

```
palabras = [
```

```
    "python", "programacion", "computadora", "algoritmo",
    "desarrollo", "interfaz", "variable", "funcion", "clase"
```

```
]
```

```
# 1. Seleccionar una palabra al azar
```

```
palabra_secreta = random.choice(palabras).upper()
```

```
# 2. Inicializar el estado del juego
```

```
letras_adivinadas = set() # Letras que el jugador ya ha adivinado
```

```
intentos_restantes = 6 # Número de fallos permitidos
```

```
juego_terminado = False
```

```
print("=====")
```

```
print("    ¡Bienvenido al Ahorcado!    ")
```

```
print("=====")
```

```
print(f"La palabra tiene {len(palabra_secreta)} letras.")
```

```
# Bucle principal del juego
```

```
while not juego_terminado:
```

```
    # 3. Mostrar el estado actual de la palabra (p. ej., P _ T H O N)
```

```
    palabra_mostrada = ""
```

```
    for letra in palabra_secreta:
```

```
        if letra in letras_adivinadas:
```

```
            palabra_mostrada += letra + " "
```

```
        else:
```

```
            palabra_mostrada += " _ "
```

```
    print("\n--")
```

```
    print(f"Palabra: {palabra_mostrada}")
```

```
    print(f"Intentos restantes: {intentos_restantes}")
```

```
    print(f"Letras usadas: {', '.join(sorted(list(letras_adivinadas)))}")
```

```
# Condición de victoria
```

```
if "_" not in palabra_mostrada:
```

```
    print("\n? ¡Felicitaciones! Has adivinado la palabra.")
```

```

juego_terminado = True
break

# Condición de derrota
if intentos_restantes == 0:
    print("\n? ¡Oh no! Te has quedado sin intentos.")
    print(f"La palabra secreta era: **{palabra_secreta}**")
    juego_terminado = True
    break

# 4. Pedir la entrada del jugador
intento = input("\nIntroduce una letra o la palabra completa: ").upper().strip()

# Validar la entrada
if not intento.isalpha():
    print("! Entrada no válida. Por favor, introduce una letra o una palabra.")
    continue

# Opción 1: Adivinar la palabra completa
if len(intento) > 1:
    if intento == palabra_secreta:
        print(f"\n? ¡INCREÍBLE! Has adivinado la palabra: **{palabra_secreta}**")
        juego_terminado = True
    else:
        intentos_restantes -= 1
        print(f"X Palabra incorrecta. Pierdes un intento.")
    continue

# Opción 2: Adivinar una sola letra
letra_adivinada = intento[0]

if letra_adivinada in letras_adivinadas:
    print(f" Ya intentaste la letra '{letra_adivinada}'. Intenta con otra.")
    continue

letras_adivinadas.add(letra_adivinada)

if letra_adivinada in palabra_secreta:
    print(f"✓ ¡Acierto! La letra '{letra_adivinada}' está en la palabra.")
else:
    intentos_restantes -= 1
    print(f"X Fallo. La letra '{letra_adivinada}' no está.")

print("====")
print("      Fin del juego.      ")
print("====")

# Iniciar el juego
if __name__ == "__main__":
    jugar_ahorcado()

```

Cómo Jugar

1. **Guarda** el código anterior en un archivo llamado, por ejemplo, ahorcado.py.

2. **Ejecuta** el archivo desde tu terminal: `python ahorcado.py`.
3. El juego te indicará la longitud de la palabra.
4. Podrás **introducir una letra** cada vez.
 - Si la letra está en la palabra, se revelará su posición.
 - Si la letra no está, perderás un intento.
5. Alternativamente, puedes intentar **adivinar la palabra completa** en cualquier momento. Si fallas, perderás un intento.
6. Ganas si descubres la palabra antes de agotar los **6 intentos** permitidos.
¿Te gustaría que añadiera una representación gráfica básica del ahorcado (usando caracteres ASCII) o que te mostrara cómo usar un conjunto de palabras más grande?

¡Excelente! Añadir la representación gráfica del ahorcado (usando caracteres ASCII) hace que el juego sea mucho más visual y entretenido.

Aquí tienes la versión actualizada del código de **Ahorcado** que incluye el dibujo que se va completando con cada fallo:

Python

```
import random
```

```
## ? Dibujos ASCII para el Ahorcado
```

```
def obtener_dibujo(intentos_fallidos):
```

```
    """
```

Retorna el dibujo ASCII del ahorcado según el número de fallos.

```
    """
```

```
    etapas = [
```

Etapa 0: Sin fallos

```
        """
```

```
        -----
```

```
        | |
```

```
        |
```

```
        |
```

```
        |
```

```
        -----
```

```
        """
```

Etapa 1: Cabeza

```
        """
```

```
        -----
```

```
        | |
```

```
        O |
```

```
        |
```

```
        |
```

```
        -----
```

```
        """
```

Etapa 2: Cuerpo

```
        """
```

```
        -----
```

```
        | |
```

```
        O |
```

```
        | |
```

```
        |
```

```

    |
-----
""",  

# Etapa 3: Brazo izquierdo  

"""  

  

    -----  

    | |  

    O |  

    /| |  

    |  

-----  

""",  

# Etapa 4: Ambos brazos  

"""  

  

    -----  

    | |  

    O |  

    /|\ |  

    |  

-----  

""",  

# Etapa 5: Pierna izquierda  

"""  

  

    -----  

    | |  

    O |  

    /|\ |  

    / |  

    |  

-----  

""",  

# Etapa 6: Ambas piernas (Juego terminado - Derrota)  

"""  

  

    -----  

    | |  

    O |  

    /|\ |  

    /\ |  

    |  

-----  

"""
]  

# Aseguramos que el índice no exceda el número de etapas  

return etapas[min(intentos_fallidos, len(etapas) - 1)]  

  

## ? Implementación del Juego del Ahorcado (con ASCII)  

  

def jugar_ahorcado():  

"""
Función principal para ejecutar el juego del Ahorcado con dibujos ASCII.  

"""
# Lista de palabras

```

```

palabras = [
    "python", "programacion", "computadora", "algoritmo",
    "desarrollo", "interfaz", "variable", "funcion", "clase"
]

palabra_secreta = random.choice(palabras).upper()
letras_adivinadas = set()
max_intentos = 6
intentos_fallidos = 0 # Usamos esta variable para indexar el dibujo
juego_terminado = False

print("=====")
print("  ¡Bienvenido al Ahorcado!  ")
print("=====")

# Bucle principal del juego
while not juego_terminado:

    # 1. Mostrar el dibujo actual del ahorcado
    print(obtener_dibujo(intentos_fallidos))

    # 2. Mostrar el estado actual de la palabra
    palabra_mostrada = ""
    for letra in palabra_secreta:
        if letra in letras_adivinadas:
            palabra_mostrada += letra + " "
        else:
            palabra_mostrada += " _ "

    print("\n---")
    print(f"Palabra: {palabra_mostrada}")
    print(f"Intentos restantes: {max_intentos - intentos_fallidos}")
    print(f"Letras usadas: {', '.join(sorted(list(letras_adivinadas)))}")

    # 3. Condición de victoria
    if "_" not in palabra_mostrada:
        print("\n? ¡Felicitaciones! Has adivinado la palabra.")
        juego_terminado = True
        break

    # 4. Condición de derrota
    if intentos_fallidos >= max_intentos:
        print("\n? ¡Oh no! Te has quedado sin intentos.")
        print(f"La palabra secreta era: **{palabra_secreta}**")
        juego_terminado = True
        break

    # 5. Pedir la entrada del jugador
    intento = input("\nIntroduce una letra o la palabra completa: ").upper().strip()

    if not intento.isalpha():
        print("! Entrada no válida. Por favor, introduce una letra o una palabra.")
        continue

    # Opción A: Adivinar la palabra completa

```

```

if len(intento) > 1:
    if intento == palabra_secreta:
        print(f"\n? ¡INCREÍBLE! Has adivinado la palabra: **{palabra_secreta}**")
        juego_terminado = True
    else:
        intentos_fallidos += 1
        print(f"X Palabra incorrecta. Pierdes un intento.")
    continue

# Opción B: Adivinar una sola letra
letra_adivinada = intento[0]

if letra_adivinada in letras_adivinadas:
    print(f" Ya intentaste la letra '{letra_adivinada}'. Intenta con otra.")
    continue

letras_adivinadas.add(letra_adivinada)

if letra_adivinada in palabra_secreta:
    print(f"✓ ¡Acierto! La letra '{letra_adivinada}' está en la palabra.")
else:
    intentos_fallidos += 1
    print(f"X Fallo. La letra '{letra_adivinada}' no está.")

print("====")
print("      Fin del juego.      ")
print("====")

```

```

# Iniciar el juego
if __name__ == "__main__":
    jugar_ahorcado()

```

Novedades Clave

- **Función obtener_dibujo(intentos_fallidos):** Esta función contiene una lista (etapas) con las representaciones del ahorcado en diferentes fases. El índice de la lista corresponde al número de fallos.
- **Actualización del Bucle:** En cada turno, se llama a obtener_dibujo(intentos_fallidos) para mostrar el estado visual del muñeco, haciendo el juego más dinámico.
¡Ahora el juego es más completo! ¿Te gustaría que implementara alguna otra de las aplicaciones que mencionaste, como un **generador de contraseñas** o un **calculadora de IMC**?