Práctica Evaluable - Juegos en Python

Francisco José Barrera Román
Sebastian Aramayo Ayarde
Aarón Fernandez Gilgado
Zhiyang Ye

1º DAW

Índice

Juego 1 - Adivina el número

1º Definimos las variables:

```python
#Importa para poder generar un número aleatorio
import random

intentos = 0 # Creamos una variable para acumular intentos

vidas = 3 # Creamos una variable con el numero de vidas

numeros_dichos = [] # Creamos una lista para guardar todos los numeros dichos

numero_secreto = random.randint( a: 1,  b: 10) # Generamos un numero aleatorio

nombre = input("Introduce tu nombre:") # Preguntamos el nombre a nuestro jugador
```

2º Primeros “prints” para dar la bienvenida y guiar al jugador:

```python
print("Bienvenido ",nombre,)
print("En estos momentos me encuentro pensando en un número.")
print("¿Eres capaz de adivinarlo?. Es un número del 1 al 10 y solo vas a tener 3 intentos.")
```

3º A continuación creamos la función principal del juego, un bucle que se repita mientras que el jugador no haya adivinado el número o tengamos vidas restantes:

```python
while numero_secreto not in numeros_dichos and vidas >0: #Creamos un bucle que se ejecute siempre y cuando

    numero_jugador = int(input("Escribeme un número: ")) #Aseguramos que sea un número entero

    if numero_jugador not in numeros_dichos: #Creamos una condición para saber si el número ha salido ante

        if numero_jugador > numero_secreto: #Subcondición donde el numero del jugador se compara con el nu
            print("El número es mayor que el número secreto")
            vidas -= 1 #Resta una vida en la variable
            numeros_dichos.append(numero_jugador) #Lo agregamos a la lista de dichos
            print("Te quedan ", vidas, " vidas")
            print("############################")  # Separador

        if numero_jugador < numero_secreto: #Subcondición donde el numero del jugador se compara con el nu
            print("El número es menor que el número secreto")
            vidas -= 1
            numeros_dichos.append(numero_jugador) #Lo agregamos a la lista de dichos
            print("Te quedan ",vidas," vidas")
            print("############################")  #Separador

        if numero_jugador == numero_secreto:
            print("Enhorabuena, has ganado el juego")
            print("Te han quedado ",vidas," vidas")
            numeros_dichos.append(numero_jugador)

    else:#
        print("Este número ya está dicho")
        print("Esto son los numeros dichos: ",numeros_dichos)
        print("############################")  # Separador
```

En este bucle observamos varias interacciones donde:

- Condiciones:
  - Si el numero ingresado no es igual a los anteriores, como condición inicial, añade sub condiciones:
    - Si el número del jugador es mayor al número secreto:
      - Sale un primer mensaje comunicando que el número dicho es mayor, a continuación se resta una de las vidas del jugador, luego añadimos el número dicho a una lista para que no se repita y por último un separador.

```
if numero_jugador > numero_secreto: #Subcondición donde el num
    print("El número es mayor que el número secreto")
    vidas -= 1 #Resta una vida en la variable
    numeros_dichos.append(numero_jugador) #Lo agregamos a la l
    print("Te quedan ", vidas, " vidas")
    print("############################")  # Separador
```

    - Si el número del jugador es menor al número secreto:
      - Sale un primer mensaje comunicando que el número dicho es mayor, a continuación se resta una de las vidas del jugador, luego añadimos el número dicho a una lista para que no se repita y por último un separador.

```
if numero_jugador < numero_secreto: #Subcondición dond
    print("El número es menor que el número secreto")
    vidas -= 1
    numeros_dichos.append(numero_jugador) #Lo agregam
    print("Te quedan ",vidas," vidas")
    print("############################")  #Separador
```

    - Si el número del jugador es igual al número secreto:
      - Sale un primer mensaje comunicando que ha ganado, le acompaña otro informando de las vidas restantes y se vuelven a añadir a la lista de números dichos.

```
if numero_jugador == numero_secreto:
    print("Enhorabuena, has ganado el juego")
    print("Te han quedado ",vidas," vidas")
    numeros_dichos.append(numero_jugador)
```

  - En caso de que el número sea igual a los anteriores, aparece un mensaje comunicando la situación pero no le quita ninguna de las vidas.

```
else:#En caso de que no coincida los numeros anteriores se informa
    print("Este número ya está dicho")
    print("Esto son los numeros dichos: ",numeros_dichos)
    print("############################")  # Separador
```

Y para finalizar el bucle en caso de que el valor de la variable “vida” llegue a 0, el bucle finaliza y se le informa al jugador de su derrota.

```
if vidas == 0: #Una vez que la variable vida llega a 0 se termina el bucle y se termina el juego.
    print("¡Has perdido!")
    print("El número secreto era ",numero_secreto)
```

Juego 2- Tres en rayas

Este código es un juego de tres en raya (tic-tac-toe) para dos jugadores que se ejecuta en la consola. A continuación te explico cada parte del código en detalle:

1. Definición del tablero

raya.py

```
tablero = [[0,1,2],[3,4,5],[6,7,8]]  # Creamos una lista bidimensional con las posiciones del tablero, inicializadas de 0 a 8
```

Aquí se crea una lista bidimensional de 3x3 para representar el tablero del juego. Inicialmente, cada celda contiene un número (del 0 al 8) que representa su posición. Esto será reemplazado por una "X" o una "O" cuando los jugadores coloquen sus piezas.

2. Ingreso de los nombres de los jugadores

```
jugadorX = input("Ingrese tu nombre jugadorX: ")  # Creamos una variable para almacenar el nombre del jugador X
jugador0 = input("Ingrese tu nombre jugador0: ")  # Creamos una variable para almacenar el nombre del jugador 0
```

Aquí se solicita al jugador que ingresen sus nombres. Las variables `jugadorX` y `jugador0` almacenan los nombres de los jugadores que usarán las piezas "X" y "O" respectivamente.

3. Función `pinta_tablero`

```
# Función que pinta el tablero en la consola
def pinta_tablero():  3 usages
    for fila in tablero:  # Recorre cada fila del tablero
        print(fila)  # Imprime la fila actual
```

Esta función imprime el tablero en la consola. El tablero es una lista bidimensional, por lo que la función recorre cada fila de la lista y la imprime. Después de cada turno, el tablero se actualizará y se mostrará al jugador el estado actual.

4. Función `ganador`

```python
# Función que verifica si el jugador ha ganado
def ganador(jugador):  4 usages
    # Comprobación de filas: Si todas las posiciones en una fila son iguales al símbolo del jugador
    for fila in tablero:
        if fila[0] == fila[1] == fila[2] == jugador:  # Verifica las horizontales
            return True

    # Comprobación de columnas: Verifica si todas las posiciones de una columna son iguales al símbolo del jugador
    for i in range(3):  # Repite 3 veces (una por columna)
        if tablero[0][i] == tablero[1][i] == tablero[2][i] == jugador:  # Verifica las verticales
            return True

    # Comprobación de diagonales:
    if tablero[0][0] == tablero[1][1] == tablero[2][2] == jugador:  # Diagonal de izquierda a derecha
        return True

    if tablero[0][2] == tablero[1][1] == tablero[2][0] == jugador:  # Diagonal de derecha a izquierda
        return True

    return False  # Si no se encontró un ganador, retorna False
```

Esta función verifica si el jugador actual ha ganado. Lo hace de las siguientes maneras:

- **Comprobación de filas**: Verifica si todas las posiciones en una fila son iguales al símbolo del jugador (`X` u `0`).
- **Comprobación de columnas**: Verifica si todas las posiciones en una columna son iguales al símbolo del jugador.
- **Comprobación de diagonales**: Verifica si alguna de las diagonales tiene todas sus posiciones iguales al símbolo del jugador.

Si cualquiera de estas condiciones se cumple, la función devuelve `True`, lo que significa que el jugador ha ganado.

5. Función `colocar_pieza`

```python
# Función para que el jugador coloque su pieza
def colocar_pieza(jugador):  3 usages
    posicion = int(input("Ingrese la posicion donde quieres colocar tu pieza: "))  # El jugador ingresa la posición donde colocar su pieza
    posicion_valida = 0  # Variable para indicar si la posición es válida

    # Recorremos las filas y columnas del tablero
    for i in range(len(tablero)):  # Recorre las filas del tablero
        for j in range(len(tablero[i])):  # Recorre las columnas de cada fila
            if tablero[i][j] == posicion:  # Si el valor de la celda coincide con la posición ingresada por el jugador
                tablero[i][j] = jugador  # Coloca la pieza (X u O) en el tablero
                posicion_valida = 1  # Marca la posición como válida

    # Si la posición ingresada no era válida (la casilla ya estaba ocupada)
    if posicion_valida == 0:
        print("Tu pieza no puede colocar")  # Imprime un mensaje de error
        colocar_pieza(jugador)  # Llama de nuevo a la función para que el jugador vuelva a intentar
```

Esta función permite que el jugador coloque su pieza en el tablero. Se le pide al jugador que ingrese la posición (un número del 0 al 8) donde quiere colocar su pieza:

- **Ingreso de la posición**: El jugador ingresa la posición donde quiere colocar su "X" o "O".
- **Comprobación de posición válida**: El código verifica si la posición está libre (si el número en esa posición aún no ha sido reemplazado por "X" u "O").
    - Si la posición es válida, la función reemplaza el número en esa posición por el símbolo del jugador.
    - Si la posición ya está ocupada, se le muestra un mensaje de error y se vuelve a pedir que elija otra posición.

6. Inicio del juego

```
49  # Inicia el juego
50  pinta_tablero()  # Pinta el tablero inicial
51
52  turnos = 1  # Variable que indica el turno actual
```

Aquí se imprime el tablero inicial y se establece una variable `turnos` que llevará la cuenta del número de turnos jugados.

7. Bucle principal del juego

```
# Bucle para permitir hasta 9 turnos (ya que el tablero tiene 9 posiciones)
for i in range(9):
    if turnos % 2 == 0:  # Si el número de turno es par, es el turno del jugador X
        print("Es el turno de ", jugadorX)
        colocar_pieza("X")  # El jugador X coloca su pieza
        pinta_tablero()  # Pinta el tablero actualizado
    else:  # Si el número de turno es impar, es el turno del jugador O
        print("Es el turno de", jugadorO)
        colocar_pieza("O")  # El jugador O coloca su pieza
        pinta_tablero()  # Pinta el tablero actualizado

    # Verifica si el jugador X ha ganado
    if ganador("X"):
        print(jugadorX, "ha hecho 3 en raya")  # Informa que el jugador X ha ganado
        break  # Termina el juego si hay un ganador

    # Verifica si el jugador O ha ganado
    if ganador("O"):
        print(jugadorO, "ha hecho 3 en raya")  # Informa que el jugador O ha ganado
        break  # Termina el juego si hay un ganador

    turnos += 1  # Incrementa el número de turnos
```

Este es el bucle que maneja el flujo del juego, permitiendo hasta 9 turnos (el número máximo de movimientos posibles en un tablero de 3x3). Aquí está el flujo:

1. **Determinación del turno**: El turno se determina en base a si `turnos` es par o impar:
   - Si `turnos % 2 == 0`, es el turno del jugador X.
   - Si `turnos % 2 != 0`, es el turno del jugador O.
2. **Colocar la pieza**: Llama a la función `colocar_pieza` para que el jugador actual elija una posición y coloque su pieza.
3. **Pintar el tablero**: Después de cada movimiento, el tablero actualizado se muestra.
4. **Verificar si hay un ganador**: Después de cada movimiento, se verifica si el jugador actual ha ganado:
   - Si `ganador("X")` es `True`, imprime que el jugador X ha ganado y sale del bucle.
   - Si `ganador("O")` es `True`, imprime que el jugador O ha ganado y sale del bucle.
5. **Incrementar el turno**: Después de cada movimiento, el contador de turnos aumenta en 1.

8. Verificar empate

```
# Si después de los 9 turnos no hay un ganador, se declara empate
if not ganador("X") and not ganador("O"):  # Si ninguno ha ganado
    print("El juego terminó en empate.")  # Informa que el juego ha terminado en empate
```

Si se han completado los 9 turnos sin que haya un ganador, se imprime un mensaje indicando que el juego terminó en empate.

Resumen del flujo:

1. El tablero se inicializa y los jugadores ingresan sus nombres.
2. Se ejecuta el bucle principal, alternando entre los jugadores X y O.
3. Cada jugador elige dónde colocar su pieza.
4. Después de cada turno, se verifica si alguien ha ganado o si hay un empate.
5. El juego termina cuando hay un ganador o se completan los 9 turnos sin ganador (empate).

Este código implementa un juego simple de tres en raya completamente funcional que se juega en la consola.

Juego 3 - Palabras Encadenadas

Este código implementa un juego de palabras para múltiples jugadores.El objetivo es que los participantes mencionen palabras que comiencen con las dos últimas letras de la palabra anterior.El juego se desarrolla en rondas y se puede reiniciar si los jugadores así lo desean.

Se importa el módulo `random`, que se utiliza para seleccionar una palabra inicial de manera aleatoria.

```
1    import random
```

Función `juego()`:

Propósito:

- Gestiona todo el flujo del juego, desde la configuración inicial hasta la finalización de la partida.

Funcionamiento:

- Solicita al usuario el número de jugadores (mínimo 2).
- Recoge los nombres de los jugadores y los almacena en una lista.
- Selecciona una palabra inicial al azar de una lista predefinida para iniciar la partida.

```
def juego():  # Función principal que inicia y controla el juego  2 usages
    print("##################################") #Separador
    while True:
        # Solicitar el número de jugadores (mínimo 2)
        num_jugadores = input("¿Cuántos jugadores participarán? (2 o más): ")
        if num_jugadores.isdigit() and int(num_jugadores) >= 2:
            num_jugadores = int(num_jugadores)
            break  # Salir del bucle cuando el número de jugadores es válido
        else:
            print("Entrada inválida. Debe ser un número entero de 2 o más jugadores.")

    # Registrar los nombres de los jugadores
    jugadores = [input(f"Ingrese el nombre del jugador {i + 1}: ") for i in range(num_jugadores)]

    # Mostrar una palabra inicial aleatoria
    palabras_iniciales = ["Colorear", "Onda", "Rural", "Casa", "Puñetazo", "Bota", "Cálido", "Polvo", "Falsificación",
                          "Ilustre"]
    print(random.choice(palabras_iniciales))
```

Rondas del Juego:

- Los jugadores deben ingresar palabras que cumplan con la regla de comenzar con las dos últimas letras de la palabra anterior.
- Si la palabra ingresada es válida y no ha sido usada antes, se añade a la lista de palabras utilizadas y el juego continúa.
- Si la palabra no es válida o ya ha sido usada, el jugador pierde, y se ofrece la opción de reiniciar el juego.

```
39
40          if es_palabra_valida(palabra, ultima_palabra):  # Verifica si la palabra es válida
41              if palabra.lower() not in palabra_dicha:  # Verifica si la palabra no ha sido usada antes
42                  palabra_dicha.append(palabra.lower())  # Añade la palabra a la lista de usadas
43                  ultima_palabra = palabra.lower()  # Actualiza la última palabra válida
44                  print(f"La palabra '{palabra}' es válida. Continua el siguiente jugador.")
45                  print("###################################") #Separador
46              else:
47                  # La palabra ya fue usada, el jugador pierde
48                  print(f"La palabra '{palabra}' ya ha sido utilizada. Has perdido.")
49                  if input("¿Quieres jugar de nuevo? (Escriba *si* o *no*): ").lower() == "si":
50                      juego()  # Reiniciar el juego si el jugador quiere volver a jugar
51                  else:
52                      print("Gracias por jugar. ¡Hasta la próxima!")
53                      break
54          else:
55              # La palabra no cumple la regla de empezar con las dos últimas letras de la palabra anterior
56              print(
57                  f"La palabra '{palabra}' no es válida. Debe comenzar con las dos últimas letras de '{ultima_palabra}'.")
58
59
```

Finalización:

- El juego termina si un jugador ingresa la palabra "fin"
- También finaliza si los jugadores deciden no volver a jugar después de una ronda completa.

```
32      while True:
33          # Solicitar una palabra del jugador o terminar el juego con "fin"
34          palabra = input("Ingrese la palabra que desea enviar (Si no encuentras, escriba *fin*): ")
35
36          if palabra.lower() == "fin":  # Terminar el juego si el jugador escribe "fin"
37              print("Juego ha terminado")
38              break
```

Volver a jugar:

Propósito:

- Permitir que el juego se repita después de cada ronda completa.

Funcionamiento:

- Después de cada ronda, se pregunta a los jugadores si desean volver a jugar.
- Si la respuesta es afirmativa, se reinicia la función `juego()`.
- Si no, el programa finaliza con un mensaje de despedida.

```
while True:
    juego()  # Inicia el juego
    # Pregunta si se quiere jugar de nuevo al finalizar una ronda completa
    if input("¿Quieres jugar de nuevo? (Escriba *si* o *no*): ").lower() != "si":
        print("Gracias por jugar. ¡Hasta la próxima!")
        break  # Termina el bucle si no se quiere volver a jugar
```

Juego 4 - Hundir barcos

Lo primero que tendremos que hacer será el “import random” para que nos genere un número aleatorio que después usaremos. Lo siguiente sería crear dos variables para pedir el nombre de los jugadores que participarán, luego de esto será crear otras dos variables para crear lo que sería el tablero de cada jugador y por último será crear otras dos variables para poner ocultas el tablero de cada jugador.

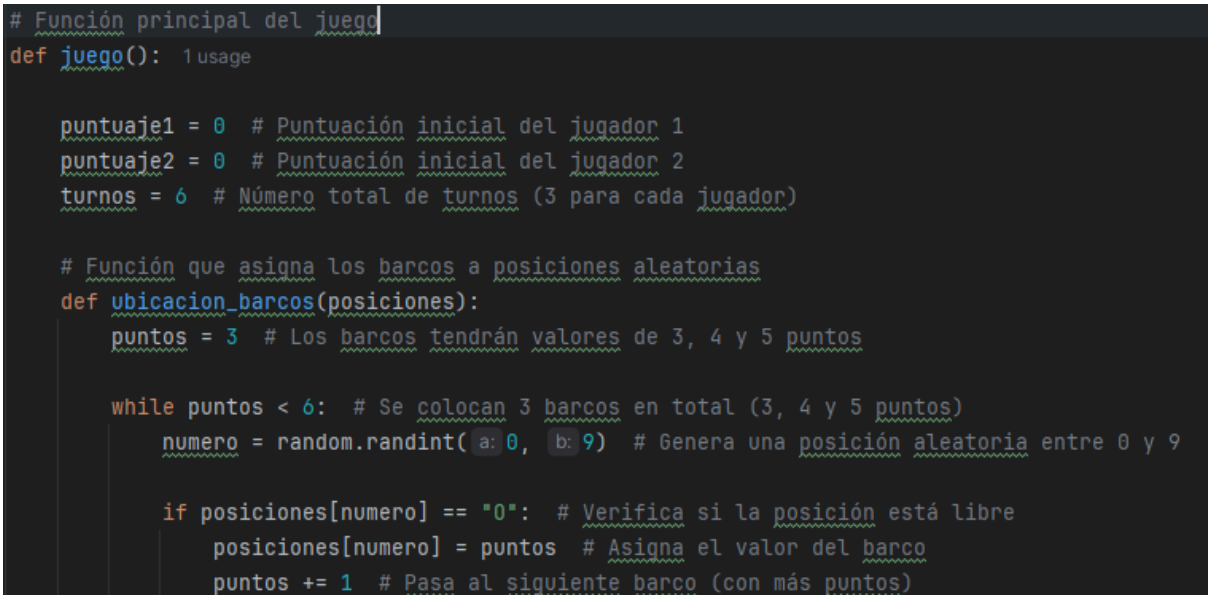
```
import random  # Importa el módulo random para generar números aleatorios

# Se solicitan los nombres de los dos jugadores
jugador1 = input("Ingrese el nombre del primer jugador: ")
jugador2 = input("Ingrese el nombre del segundo jugador: ")

# Estas listas representan las posiciones de los barcos para cada jugador
posiciones1 = ["0","0","0","0","0","0","0","0","0","0"]  # Posiciones del jugador 1
posiciones2 = ["0","0","0","0","0","0","0","0","0","0"]  # Posiciones del jugador 2

# Estas listas muestran las posiciones ocultas para que el otro jugador no vea dónde están los barcos
posiciones1ocultas = ["0","0","0","0","0","0","0","0","0","0"]  # Posiciones ocultas del jugador 1
posiciones2ocultas = ["0","0","0","0","0","0","0","0","0","0"]  # Posiciones ocultas del jugador 2
```

En la primera función lo que hacemos es asignar el puntaje de cada jugador y los turnos totales que tendrán entre los dos. Dentro de la segunda función tendremos una variable en la que sabremos el valor de puntos de los barcos que serán 3 los cuales valdrán 3, 4 y 5 puntos. Luego tendremos un bucle while en el que la condición será que mientras los puntos sean menores a 6 nos genere un número aleatorio entre 0 y 9 y luego dentro del bucle tendremos una condición en la que la condición sería que verifique si la posición en la que el jugador quiere disparar está libre luego dentro lo que hará será asignar el valor del barco al que le ha dado y por último sumará los puntos correspondientes.



```
# Función principal del juego
def juego():  1 usage

    puntuaje1 = 0  # Puntuación inicial del jugador 1
    puntuaje2 = 0  # Puntuación inicial del jugador 2
    turnos = 6  # Número total de turnos (3 para cada jugador)

    # Función que asigna los barcos a posiciones aleatorias
    def ubicacion_barcos(posiciones):
        puntos = 3  # Los barcos tendrán valores de 3, 4 y 5 puntos

        while puntos < 6:  # Se colocan 3 barcos en total (3, 4 y 5 puntos)
            numero = random.randint( a: 0,  b: 9)  # Genera una posición aleatoria entre 0 y 9

            if posiciones[numero] == "0":  # Verifica si la posición está libre
                posiciones[numero] = puntos  # Asigna el valor del barco
                puntos += 1  # Pasa al siguiente barco (con más puntos)
```

Luego tendremos otra función en la que se ejecutará el turno de disparo del jugador. Lo primero que tenemos es una lista de opciones en la que el jugador pueda disparar, lo siguiente será pintarla por pantalla, luego tendremos una variable en la que se le preguntará al jugador en qué posición quiera disparar y por último lo que se hará será restar 1 para ajustar el índice. Luego tendremos una condición que cumplir que será si la posición en la que ha disparado el jugador sea en blanco, es decir, si no hay barco situado en él, si esa condición se cumple pintaremos por consola “Agua” para que el jugador sepa que no ha tocado ningún barco, lo siguiente cambia la marca principal del sitio por una “A” para que el jugador sepa donde ha disparado, por último se pinta por consola el tablero oculto con la actualización y el puntaje actual del jugador. Luego tendremos un else en la que será que el jugador ha tocado un barco por tanto pintará por consola “¡Tocado y hundido!” para que el jugador sepa que ha tocado y hundido un barco por tanto lo siguiente que hará será sumar el valor del barco a su puntaje y cambiará el valor principal donde se encontraba el barco por una “X”, por último pintará el tablero oculto con la actualización y mostrar la nueva puntuación del jugador. Y por último retornará la puntuación actualizada del jugador.

```python
# Función que ejecuta el turno de disparo de un jugador
def buscar_barcos(jugador, posiciones, puntuaje, posiciones_ocultas):
    lista = [1, 2, 3, 4, 5, 6, 7, 8, 9, 10]  # Lista de opciones para que el jugador dispare
    print(lista)  # Muestra la lista de posiciones disponibles
    posicion = int(input("Introduce que numero quieres disparar: "))  # El jugador elige una posición
    posicion -= 1  # Restamos 1 para ajustar el indice (listas en Python empiezan en 0)

    if posiciones[posicion] == "0":  # Si en esa posición no hay barco
        print("¡Agua!")  # Se informa que no se ha golpeado un barco
        posiciones_ocultas[posicion] = "A"  # Marca "A" de "agua" en la lista oculta
        print(posiciones_ocultas)  # Muestra el tablero oculto con la actualización
        print(jugador," tiene ", puntuaje," puntos.")  # Muestra la puntuación actual del jugador

    else:  # Si en la posición había un barco
        print("¡Tocado y hundido!")  # Informa que se ha golpeado un barco
        puntuaje += posiciones[posicion]  # Suma el valor del barco a la puntuación
        posiciones_ocultas[posicion] = "X"  # Marca "X" de golpe acertado en la lista oculta
        print(posiciones_ocultas)  # Muestra el tablero oculto con la actualización
        print(jugador," tiene ", puntuaje," puntos.")  # Muestra la nueva puntuación del jugador

    return puntuaje  # Retorna la puntuación actualizada del jugador
```

Aquí tendremos un bucle while el cual se cumplirá mientras los turnos sean mayor que cero, y lo que hará será alternar los turnos para cada jugador, en él tendremos un condicional en el que si el turno es par será pintará una línea de hashtags a modo de separación de cada turno y también pintará “es el turno de” y el nombre del jugador 1 por último asignamos una variable con el puntaje del jugador 1 que sea igual a la función de buscar barcos para que pueda disparar. Luego tendremos un else que hará lo mismo que el primer condicional solo que para el jugador 2. Por último se restará 1 al número de turnos.

```
# Bucle principal del juego: se alternan los turnos
while turnos > 0:

    if turnos % 2 == 0:  # Si el turno es par, es el turno del jugador 1
        print("#####################################################")
        print("Es el turno de", jugador1)
        puntuaje1 = buscar_barcos(jugador1, posiciones2, puntuaje1, posiciones2ocultas)  # El jugador 1 dispara

    else:  # Si el turno es impar, es el turno del jugador 2
        print("#####################################################")
        print("Es el turno de", jugador2)
        puntuaje2 = buscar_barcos(jugador2, posiciones1, puntuaje2, posiciones1ocultas)  # El jugador 2 dispara

    turnos -= 1  # Se reduce el número de turnos en 1
```

Este condicional lo que hace es comparar el puntaje del jugador 1 con la del jugador 2, si esta condición se cumple pintará el nombre del jugador 1 y que ha ganado con su número de puntos y el nombre del jugador 2 con su número de puntos. Luego tendremos otra condición la cual para que se cumpla tiene que no haberse cumplido la primera condición y lo siguiente es que se tiene que cumplir la condición que tendrá éste la cual es que si el puntaje del jugador 2 es mayor que la del jugador 1 pintará el nombre del jugador 2 y que ha ganado con su número de puntos y el nombre del jugador 1 junto con su número de puntos. Por último tendremos un else que se cumplirá solo si no se ha cumplido ninguna de las dos anteriores condiciones y pintará “han quedado empate”.

```
# Compara las puntuaciones al final del juego para determinar el ganador
if puntuaje1 > puntuaje2:
    print("Ha ganado", jugador1, "con", puntuaje1, "puntos", "y", jugador2, "ha conseguido", puntuaje2, "puntos")

elif puntuaje2 > puntuaje1:
    print("Ha ganado", jugador2, "con", puntuaje2, "puntos", "y", jugador1, "ha conseguido", puntuaje1, "puntos")

else:
    print("Han quedado empate")  # Si las puntuaciones son iguales, es empate
```

Por último tendremos que llamar a la primera función que contiene todo el juego para que éste se ejecute.

```
# Llamada para iniciar el juego
juego()
```