🚢 Clase 3: Guía de Estudio: Proyecto Final - Guerra Naval

1. Introducción al Juego 🎮

El proyecto "Guerra Naval" es una implementación del clásico juego de estrategia donde dos jugadores colocan sus barcos en un tablero y se turnan para atacar. El objetivo es hundir todos los barcos del oponente. Este proyecto pone a prueba tus habilidades en programación con Python, incluyendo:

- Variables
- Programación orientada a objetos
- Uso de librerías

---

2. Fundamentos del Juego 🧩

Estructura del Juego:

- **Tablero**: Una matriz donde se colocan los barcos.
- **Barcos**: Diferentes tipos de barcos con distintos tamaños y orientaciones (horizontal o vertical).

- **Turnos**: Los jugadores se turnan para atacar las posiciones del tablero del oponente.

Tipos de Barcos:

1. **Destructor**: Tamaño 2.
2. **Submarino**: Tamaño 3.
3. **Acorazado**: Tamaño 4.

Colocación de Barcos:

- Los jugadores ingresan las coordenadas (fila, columna) y la orientación del barco.
- El código imprime el estado del tablero después de cada colocación.

---

3. Desarrollo del Proyecto 💻

Paso a Paso:

1. **Bienvenida al Juego**:
   - Mensaje inicial al usuario.
2. **Colocación de Barcos**:
   - Solicitar al jugador 1 que ingrese las coordenadas y orientación de sus barcos.
   - Imprimir la matriz del tablero con los barcos colocados.
   - Repetir para el jugador 2.
3. **Turnos de Ataque**:
   - Permitir que el jugador 1 ataque.
   - Indicar si el ataque fue exitoso (impacto) o fallido (agua).

Uso de la Librería `random`:

- Implementar un modo de juego donde un solo jugador se enfrenta a la máquina.
- La máquina selecciona posiciones aleatorias para atacar.

---

4. Ejemplo de Código 📜

Aquí tienes un fragmento del código que podrías usar para implementar la lógica básica del juego:

```
class Barco:
    def __init__(self, nombre, tamaño, fila, columna, orienta
cion):
        self.nombre = nombre
        self.tamaño = tamaño
        self.fila = fila
        self.columna = columna
        self.orientacion = orientacion

class Tablero:
    def __init__(self):
        self.matriz = [["~" for _ in range(10)] for _ in rang
e(10)]

    def colocar_barco(self, barco):
        # Lógica para colocar el barco en la matriz
        pass

    def atacar(self, fila, columna):
        # Lógica para realizar un ataque
        pass

# Ejemplo de uso
tablero_jugador1 = Tablero()
```

```
destructor = Barco("Destructor", 2, 0, 0, "horizontal")
tablero_jugador1.colocar_barco(destructor)
```

---

5. Ejercicios Prácticos 🚀

Ejercicio 1: Crear Clases

Crea una clase `Jugador` que contenga:

- Nombre del jugador.
- Método para colocar barcos.
- Método para atacar.

Solución Paso a Paso:

```
class Jugador:
    def __init__(self, nombre):
        self.nombre = nombre
        self.tablero = Tablero()

    def colocar_barco(self, barco):
        self.tablero.colocar_barco(barco)

    def atacar(self, fila, columna):
        return self.tablero.atacar(fila, columna)
```

Ejercicio 2: Lógica de Ataque

Implementa la lógica de ataque en la clase `Tablero`. Si un ataque es exitoso, marca la posición como "X", de lo contrario, como "O".

Solución:

```
def atacar(self, fila, columna):
    if self.matriz[fila][columna] == "~":
```

```
            self.matriz[fila][columna] = "O"  # Agua
            return False
        elif self.matriz[fila][columna] != "~":
            self.matriz[fila][columna] = "X"  # Impacto
            return True
```

Ejercicio 3: Modo de Juego con Máquina

Implementa un modo de juego donde un jugador se enfrenta a una máquina. La máquina debe seleccionar posiciones aleatorias para atacar.

Solución:

```
def ataque_maquina(self):
    fila = random.randint(0, 9)
    columna = random.randint(0, 9)
    return self.atacar(fila, columna)
```

6. Conclusión 🎉

El proyecto "Guerra Naval" es una excelente manera de aplicar tus conocimientos en programación orientada a objetos y lógica de programación. ¡Diviértete programando y jugando! 🎈

Reflexión:

Explora la documentación y mejora tu juego con nuevas características. ¡No olvides compartir tus avances! 🚀