Fabiola de Arizón de Callataÿ

PyRisk

Descripción del proyecto inspirado de Risk:

Este proyecto implementa un juego de estrategia basado en un tablero de territorios, donde el objetivo es maximizar las conquistas utilizando combinaciones de tropas. El juego incluye reglas específicas sobre las tropas disponibles, los recursos máximos (puntos de ejército) y los territorios enemigos a conquistar. El sistema permite la personalización de las reglas, el diseño del tablero y las estrategias de ataque. A continuación, se explica en detalle cómo funciona cada parte del proyecto acompañado de ejemplos de uso para las diferentes posibilidades de juego.

Estructura del Proyecto

1. **reglas.py**: Este módulo contiene las reglas del juego, con los datos iniciales, que se pueden modificar: incluyen los tipos de tropas disponibles, sus costos, la cantidad de puntos máximos que un jugador puede usar para asignar tropas, y la cantidad de territorios enemigos a conquistar.
   También tiene una opción para priorizar atacar territorios "débiles" con baja defensa, que el jugador puede activar o desactivar.

```python
# Valores por defecto del juego
reglas = {
    "puntos_maximos": 20,
    "tropas": {
        "Infantería": {"Fuerza": 1, "Costo": 1},
        "Caballería": {"Fuerza": 3, "Costo": 3},
        "Artillería": {"Fuerza": 5, "Costo": 5},
    },
    "territorios": 3,
    "priorizar_debiles": True  #BONUS : regla añadida para priorizar territorios débiles
}
```

2. **tablero.py**: Aquí se crea el tablero del juego. El tablero está compuesto por territorios con diferentes niveles de defensa y tipos de terreno. El jugador debe ingresar los datos manualmente a través de la interfaz de cada territorio, incluyendo su defensa y el tipo de terreno (montaña, llanura o valle). El tablero se genera de forma dinámica y se puede visualizar en cualquier momento durante el juego.

   ejemplo de tablero:

```
--- Tablero actual ---
Madrid: Defensa = 10, Terreno = Llanura
Pirineos: Defensa = 8, Terreno = Montaña
Málaga: Defensa = 5, Terreno = Valle
```

   El tablero se representa mediante un **diccionario** de Python, donde cada clave es el nombre de un territorio y el valor es otro diccionario que contiene las propiedades de ese territorio. Este enfoque es eficiente porque permite almacenar y acceder fácilmente a la información relacionada con cada territorio.

3. **juego.py**: Contiene la lógica principal del juego. Aquí se generan combinaciones de tropas basadas en los puntos disponibles, se calculan las mejores órdenes de ataque y se evalúan las victorias posibles según la configuración de tropas y el orden de ataque. Además, el sistema tiene una funcionalidad de optimización que elige la mejor combinación de tropas y el mejor orden de ataque para maximizar las conquistas.

   Dentro de este archivo, las funciones principales son:
   - **generar_combinaciones**: En el juego, el jugador asigna puntos para formar combinaciones de tropas (Infantería, Caballería, Artillería). Esta función evalúa todas las combinaciones posibles sin superar el límite de puntos. Aquí se configura que debe haber al menos una unidad de infantería, una de caballería y una de artillería en el ejército.
   - **mejor_combinación: La función** evalúa todas las combinaciones posibles de tropas generadas por la función generar_combinaciones y calcula cuántas victorias tendría cada configuración en el tablero. Devuelve la combinación de tropas que maximiza las conquistas con los recursos disponibles.
   - **generar_órdenes**: Según si el criterio de "priorizar territorios débiles", está activado, los territorios enemigos se ordenan en ese orden o se generan todas las permutaciones posibles.
   - **calcular_victorias**: La función evalúa cómo cada combinación de tropas puede conquistar territorios en función del orden de ataque. Compara la fuerza total de las tropas con la defensa de cada territorio. Si la fuerza es mayor o igual a la defensa de un territorio, se considera una victoria y el territorio es conquistado.
   - **optimizar_ataque**: La función evalúa todas las combinaciones de tropas y todas las órdenes de ataque posibles, eligiendo la que maximiza el número de victorias. Esto se logra comparando las victorias obtenidas con cada configuración y seleccionando la mejor.

   Es aquí donde también está configurada la opción de jugar con estrategia, es decir ordenando tropas en función del tipo de territorio, lo que permite una mejor planificación del ataque. Al clasificar las tropas según su efectividad frente a diferentes tipos de terreno, la estrategia optimiza los recursos para conquistar territorios más fácilmente. Esta clasificación ayuda a elegir qué tropas enviar a qué territorio según su defensa y tipo de terreno, maximizando así las probabilidades de victoria. El orden de las tropas influye directamente en la ejecución eficiente de las estrategias de ataque.

```python
def asignar_tropas_por_terreno(tablero, combinacion):
    """
    Asigna tropas a los territorios en función del tipo de terreno y la estrategia definida.
    """
    estrategia_terreno = {
        "Montaña": {"prioridad": ["Artillería", "Infantería"], "peso": {"Artillería": 2, "Infantería": 1}},
        "Llanura": {"prioridad": ["Caballería", "Infantería"], "peso": {"Caballería": 3, "Infantería": 1}},
        "Valle": {"prioridad": ["Infantería", "Caballería"], "peso": {"Infantería": 2, "Caballería": 1}},
    }
```

4. **main.py**: Este es el archivo principal que arranca la ejecución del juego. Contiene el flujo principal donde el usuario puede interactuar con el juego. Desde aquí, se puede:
   - actualizar las reglas, como el número de territorios, el costo de las tropas y la prioridad de atacar territorios débiles.
   - crear un tablero dinámico donde se ingresan los territorios, su defensa y el tipo de terreno.
   - visualizar el estado actual del tablero y las reglas en cualquier momento.
   - elegir qué tipo de partida jugar (con o sin estrategia).