

Explanation of Code Functionality

This program is a turn-based game implemented using Python's pygame library.

Library list: pygame, random.

Player and enemy characters with health, attack, defense, and special abilities.

The background image and enemy image were generated by AI, and all the rest of the image material was original by me. Background music and audio material to group Internet free open source material.

The whole program is divided into four parts:

Classes

Initialization and Assets

Define Functions

Main Loop

Classes

This section organizes characters, games, and sound materials into their respective classes for easy subsequent calls. As for why the image material was not written as class, it was because the image material was first added to the program, I forgot to write class at that time, and at that time I was studying the animation effect (deleted later, image move too fast and can not slow down), there were a lot of code needed to capture the position of the image, the impact was too great, so I gave up the modification. Adding music material later was the realization that writing a class was a good way to do it

Character shows the character's name, hp, and attack power, as well as attack, defense, special powers, and AI logic

Set the role to participate in the Game, switch the round and judge the end of the game in the game, by writing these two classes, if I want to add multiple roles to let the player choose one to play, it will be very convenient. As for why I didn't do this, it's because now I draw the player's image myself, and that's what I want the character to look like, so I didn't add character selection

The last class is the one that manages all the sounds

Initialization and Assets

This is very similar to my original version of the code, mainly some of the most basic Settings of pygame, screen, fonts, colors, and very important variables that will be needed later.

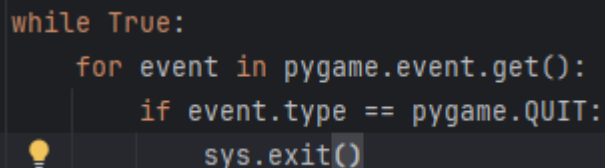
Define Functions

These are all functions needed in the main loop of the game, most of which are UI-related, such as drawing a health Bar, drawing a button, drawing a start menu, etc. There are also some functional functions, such as initializing the game, special CD, etc

Main Loop

The main point I want to explain in this part is the logic of mouse click button. It can be said that buttons are not real buttons or fake buttons, they are just some images drawn on the screen, but their.rect properties are clear, so when capturing mouse action, as long as the coordinates of the mouse click area are the same as the.rect of the button image, You can think of it as clicking a button. All button functions are implemented according to this logic.

My original version of the main loop was:



```
while True:
    for event in pygame.event.get():
        if event.type == pygame.QUIT:
            sys.exit()
```

Is to close the window by referring to sys, sys.exit(), later found not to use, set a variable running = True, when needed to end it equal to False.

The main loop's structure is: first judge the game state, menu, playing, or end. Then the next level is draw UI for each state. In playing state, it is divided into two parts by attribution of turn, the player Part and the enemy part, depending on whose turn it is, the player part performs different actions by collecting the position of the player's mouse click, while the enemy part performs actions by accepting random results generated by the AI (AI logic in the Class Part)

Finally refresh the screen

UI and graph

The graphical user interface could enhances the game's usability and aesthetics. Health bars and numeric indicators provide clear visual feedback on the status of the player and enemy. Buttons for actions are intuitive, changing color on hover and disabling when unavailable. Temporary action flags appear near characters to indicate their actions, while the action log provides a detailed record of recent moves, distinguishing player and enemy actions with color coding. When the game ends, a dedicated game-over screen displays the winner and offers options to replay or quit, seamlessly transitioning between gameplay and conclusion.