Project 5: Image Transformation and Interaction in a Pixel Environment

Objective: Implement a Python program to control the movements and transformations of a knight sprite within a 256x256 pixel background image. This project will involve direct pixel manipulation without the use of libraries such as PIL or NumPy.

Provided Materials:

- File:
    - A 256x256 background image.
    - A 32x32 face sprite image.
    - Skeleton code to start your implementation.

Key Requirements:

Sprite Class:

- Implement a Sprite class to encapsulate the sprite's image data and position.
- Include methods for moving the sprite (up, down, left, right by 8 pixels) and resetting to a default spawn position.

Direct Image Transformation:

- Implement direct transformation to overlay the sprite onto the background image, without using libraries like PIL or NumPy.
- To overlay the sprite onto the background, you would iterate over each pixel of the sprite and replace or blend the corresponding pixel in the background image with it.
- This involves adjusting the x and y coordinates of the sprite relative to the background image's coordinates.

Movement and Reset Actions:

- Allow the sprite to move 8 pixels in the specified directions and reset to its spawn point when commanded.

- Ensure the sprite does not move beyond the boundaries of the background image.

Interactive Elements:

- Implement interactions with two special objects in the environment: a star and a mushroom.
- If the sprite touches the star, invert its image colors.
- If the sprite touches the mushroom, resize the sprite to a 32x32 image. Each pixel of the sprite should be transformed into a 4x4 block of the same color.
- These transformations should be reversible if the reset action is invoked.

Boundary Box Calculation:

- You will need to calculate the boundary boxes for the star and the mushroom yourself. This involves determining the coordinates where these objects are located on the background image.

Visual Output:

- Use matplotlib to display the background image with the sprite overlaid on it after each action.
- Ensure that the transformations (inversion and resizing) are visible in the output.

User Interaction:

- Provide a text-based interface for the user to input commands for moving the sprite, resetting its position, and ending the program.
- Display a list of possible actions and their effects to the user.

YOUR TASKS:

touches_star Method:

- Estimate the boundaries of the star.
- This task is located in the touches_star method of the Sprite class.

touches_mushroom Method:

- Estimate the boundaries of the mushroom.
- This task is located in the touches_mushroom method of the Sprite class.

move Method:

- Write a method that will alter the position of the sprite by 16 pixels either left, right, up, or down.
- This task is located in the move method of the Sprite class.

overlay_on_background Method:

- Calculate the x pixel position relative to (0,0) of the background image.
- Calculate the y pixel position relative to (0,0) of the background image.
- These tasks are located in the overlay_on_background method of the Sprite class.

display_image Function:

- Write code using plt from Matplotlib to display the new image.
- This task is located in the display_image function outside of the Sprite class.

User Interaction Loop:

- Write a user interface for up, down, left, right, reset, and quit commands. Use the class move and reset methods.
- This task is located within the user interaction while loop at the end of the script.

For each task, you'll need to fill in the appropriate logic or code to achieve the desired functionality.