This code is an implementation of the classic Snake game using the Pygame library in Python. Let's go through the main parts of the code and explain their functionality:

1. Initializing the Game:
    - Pygame is initialized using `pygame.init()`.
    - Various color variables are defined using RGB values for later use in the game.
    - Display dimensions (`dis_width` and `dis_height`) are set.
    - The game window is created using `pygame.display.set_mode()` and the window caption is set using `pygame.display.set_caption()`.
    - A clock object is created to control the frame rate of the game.
2. Defining Game Variables:
    - `snake_block` represents the size of each snake block.
    - `snake_speed` represents the speed at which the snake moves.
    - `font_style` and `score_font` are defined for rendering text on the game screen.
3. Defining Game Functions:
    - `Your_score(score)` function renders and displays the current score on the game screen.
    - `our_snake(snake_block, snake_list)` function draws the snake on the game screen based on the current positions of the snake blocks.
    - `message(msg, color)` function displays a message on the screen with the specified color.
4. Main Game Loop:
    - The `gameLoop()` function is defined to contain the main game logic.
    - The game loop runs until `game_over` is set to True.
    - The game loop contains a nested loop that handles the game over state (`game_close == True`).
    - When the game is over, the screen is filled with red color, a game over message is displayed, and the player's score is shown.
    - The nested loop handles key events when the game is over. If the player presses 'Q', the game ends. If 'C' is pressed, a new game loop is started to play again.
5. Event Handling:
    - The game loop handles pygame events. If the player closes the game window, `pygame.QUIT` event is triggered and `game_over` is set to True.
    - If arrow keys are pressed, the direction of the snake's movement is updated accordingly.
6. Snake Movement and Collision:
    - The snake's head position (`x1` and `y1`) is updated based on the direction of movement (`x1_change` and `y1_change`).

- If the snake hits the boundary of the game screen, the game over state is triggered (`game_close = True`).
- The snake's head position is added to `snake_List`, and if the length of the list exceeds the length of the snake, the tail block is removed (`del snake_List[0]`).
- If the snake's head collides with any other block in `snake_List`, indicating it has bitten itself, the game over state is triggered.
- The snake and food are drawn on the game screen using `our_snake()` and `pygame.draw.rect()` functions, respectively.

7. Eating the Food:
    - If the snake's head position matches the position of the food, a new food position is generated randomly within the game screen, and the snake's length is increased by 1.
8. Controlling Game Speed:
    - The `clock.tick(snake_speed)` limits the frame rate of the game to ensure smooth animation.
9. Exiting the Game:
    - When the game loop ends, `pygame.quit()` is called to uninitialize Pygame, and `quit()` is called to exit the program.

This code allows the player to control a snake using arrow keys, eat food, and grow longer. The game ends when the snake hits the boundary of the game screen or bites itself. The player can choose to play again or quit after losing the game.