```
import pygame.font
from pygame.sprite import Group

from ship import Ship


class Scoreboard:
    """
    Klassen til at rapportere og vise scoreinformation.
    """

    def __init__(self, ai_game):
        """
        Initialiserer attributter til scorekeeping.
        """
        self.ai_game = ai_game
        self.screen = ai_game.screen
        self.screen_rect = self.screen.get_rect()
        self.settings = ai_game.settings
        # Henter spillets statistikker (f.eks. score, highscore,
niveau).
        self.stats = ai_game.stats

        # Indstillinger for skrifttype og farve til visning af
score.
        # Sætter tekstfarven til mørkegrå.
        self.text_color = (30, 30, 30)
        # Bruger en standard skrifttype i størrelsen 48.
        self.font = pygame.font.SysFont(None, 48)

        # Definer de første instanser af score, highscore, niveau og
antal liv.
        # Opretter det billede, der viser den aktuelle score.
        self.prep_score()
        # Opretter billede til highscore.
        self.prep_high_score()
        # Opretter billede til niveau.
        self.prep_level()
        # Opretter billeder, der viser hvor mange liv spilleren har
tilbage.
        self.prep_ships()

    def prep_score(self):
        """
        Omdanner den aktuelle score til et grafisk billede
        """
        # Runder scoren til nærmeste 10
        rounded_score = round(self.stats.score, -1)
        # Formaterer scoren med kommaer som tusindseparator (f.eks.
1,000).
        score_str = f"{rounded_score:,}"
        # Omdanner den formaterede score til et billede (render)
        # ved hjælp af den valgte skrifttype.
        self.score_image = self.font.render(score_str, True,
                self.text_color, self.settings.bg_color)

        # Viser scoren øverst til højre på skærmen.
        # Får rektanglen (position og dimension) af score-billedet.
        self.score_rect = self.score_image.get_rect()
        # Placerer det 20 pixels fra højre side af skærmen.
        self.score_rect.right = self.screen_rect.right - 20
        # Placerer det 20 pixels fra toppen af skærmen.
        self.score_rect.top = 20

    def prep_high_score(self):
        """
        Omdanner highscoren til et grafisk billede
        """
        # Runder highscoren til nærmeste 10.
        high_score = round(self.stats.high_score, -1)
        # Formaterer highscoren med tusindseparator.
        high_score_str = f"{high_score:,}"
        # Omdanner highscoren til et billede (render) ved hjælp
        # af den valgte skrifttype.
        self.high_score_image = self.font.render(high_score_str,
True,
                self.text_color, self.settings.bg_color)

        # Centrér highscoren øverst på skærmen.
        # Får rektanglen af highscore-billedet.
        self.high_score_rect = self.high_score_image.get_rect()
        # Centrér det horisontalt på skærmen.
        self.high_score_rect.centerx = self.screen_rect.centerx
        # Placer det på samme højde som den aktuelle score.
        self.high_score_rect.top = self.score_rect.top

    def prep_level(self):
        """
        Omdanner niveauet til et grafisk billede
        """
        # Omdanner niveauet til en streng.
        level_str = str(self.stats.level)
        # Omdanner niveauet til et billede (render)
        # ved hjælp af den valgte skrifttype.
        self.level_image = self.font.render(level_str, True,
                self.text_color, self.settings.bg_color)

        # Placerer niveauet lige under scoren.
        # Får rektanglen af niveau-billedet.
        self.level_rect = self.level_image.get_rect()
        # Justerer det horisontalt i forhold til scoren.
        self.level_rect.right = self.score_rect.right
        # Placerer det lige under scoren med en afstand på 10
pixels.
        self.level_rect.top = self.score_rect.bottom + 10

    def prep_ships(self):
        """
```

```
        Viser hvor mange rumskibe (liv) spilleren har tilbage.
        """
        # Opretter en ny gruppe til at holde alle de små rumskibe.
        self.ships = Group()
        for ship_number in range(self.stats.ships_left):
            # Opretter et lille rumskib.
            ship = Ship(self.ai_game)
            # Placerer hvert rumskib med lidt afstand horisontalt.
            ship.rect.x = 10 + ship_number * ship.rect.width
            # Placerer rumskibene øverst på skærmen.
            ship.rect.y = 10
            # Tilføjer rumskibet til gruppen.
            self.ships.add(ship)

    def check_high_score(self):
        """
        Tjekker, om spilleren har opnået en ny highscore
        """
        # Tjekker om den aktuelle score er højere end highscoren.
        if self.stats.score > self.stats.high_score:
            # Hvis ja, opdateres highscoren.
            self.stats.high_score = self.stats.score
            # Omdanner den nye highscore til et billede.
            self.prep_high_score()

    def show_score(self):
        """
        Tegner score, highscore, niveau og resterende liv (rumskibe)
på skærmen
        """
        # Tegner den aktuelle score på skærmen.
        self.screen.blit(self.score_image, self.score_rect)
        # Tegner highscoren på skærmen.
        self.screen.blit(self.high_score_image,
self.high_score_rect)
        # Tegner niveauet på skærmen.
        self.screen.blit(self.level_image, self.level_rect)
        # Tegner de små rumskibe (tilbageværende liv) på skærmen.
        self.ships.draw(self.screen)
```