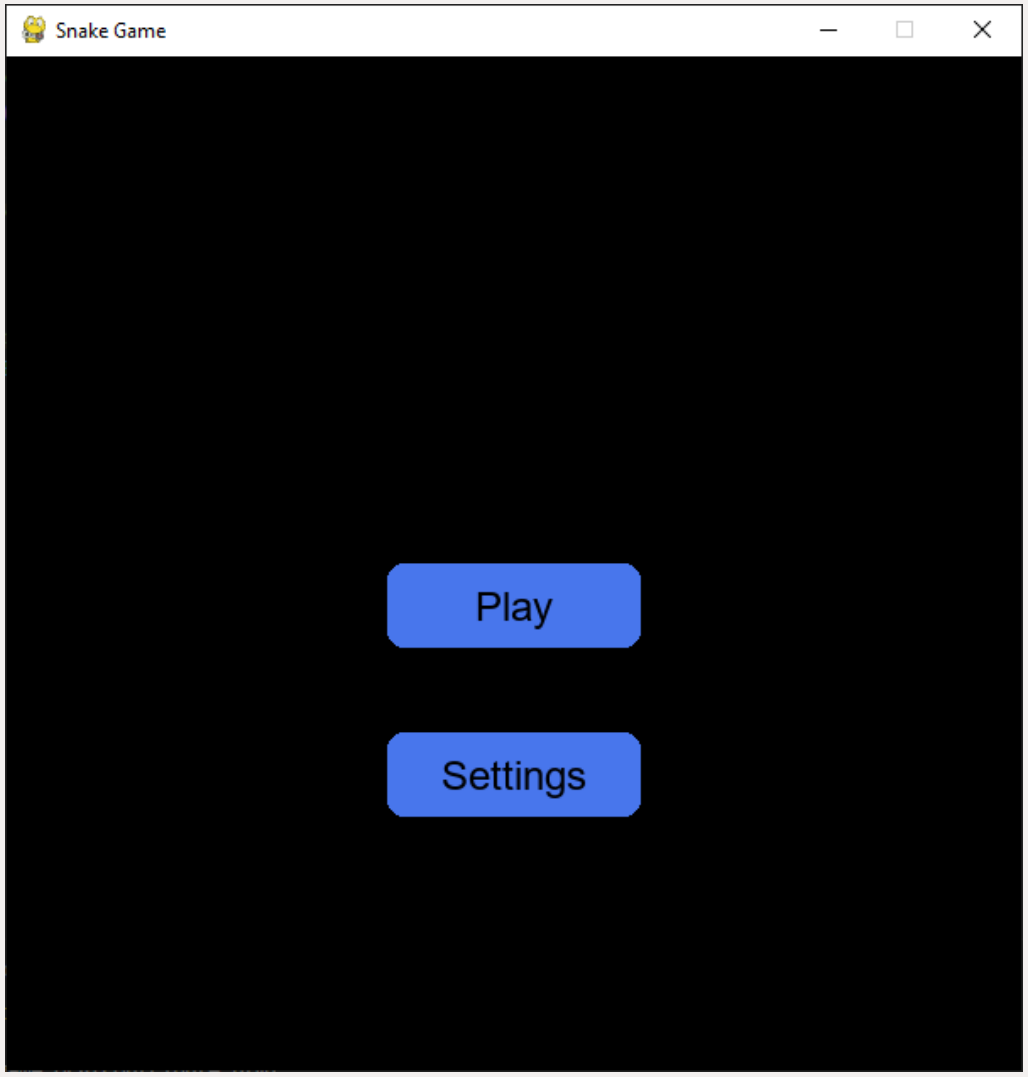
Название проекта:
"Игра Змейка с расширенной функциональностью"

АВТОР ПРОЕКТА: ЖЕВЛАКОВ ДМИТРИЙ

Описание проекта

- Данный проект представляет собой классическую игру **Змейка**, но с улучшенной графикой и настройками.
- Игрок управляет змейкой, собирая еду и увеличивая свой счёт.
- Особенность проекта – возможность выбора цвета змейки, типа еды и отображение лица змейки.



Цели проекта

- "Создать классическую игру 'Змейка' на Python с использованием Pygame."
- "Добавить новые функции: настройку графики, выбор еды, анимацию змейки."
- "Разработать удобное меню с настройками и отображением счета."

Используемые технологии

- **Язык программирования:** Python
- **Графическая библиотека:** Pygame
- **Среда разработки:** VS Code / PyCharm

Основные классы проекта

- `Snake` – отвечает за движение змейки, рост при поедании еды и столкновения.
- `Food` – отвечает за генерацию еды и возможность выбора её изображения.
- `SnakeGame` – управляет игровым процессом и логикой.
- `MainMenu` – отображает стартовое меню с настройками.

```python
class Snake:
    def __init__(self, color, grid_size):
        self.body = [[grid_size * 5, grid_size * 5]]
        self.direction = (grid_size, 0)
        self.color = color
        self.grid_size = grid_size

    def move(self):
        new_head = [self.body[0][0] + self.direction[0], self.body[0][1] + self.direction[1]]
        self.body.insert(0, new_head)
        self.body.pop()
```

Отрисовка змейки и еды

- Голова змейки содержит глаза, нарисованные с помощью Pygame.
- Игрок может выбрать изображение еды (яблоко, апельсин, арбуз).

```python
def draw(self, screen):
    # Рисуем голову
    head_x, head_y = self.body[0]
    pygame.draw.rect(
        screen,
        self.color,
        pygame.Rect(head_x, head_y, self.grid_size, self.grid_size),
    )

    # Рисуем глаза на голове
    eye_radius = self.grid_size // 5
    eye_offset_x = self.grid_size // 4
    eye_offset_y = self.grid_size // 4

    # Левый глаз
    pygame.draw.circle(
        screen,
        (255, 255, 255),  # Белый цвет
        (head_x + eye_offset_x, head_y + eye_offset_y),
        eye_radius,
    )
    pygame.draw.circle(
        screen,
        (0, 0, 0),  # Чёрный цвет
        (head_x + eye_offset_x, head_y + eye_offset_y),
        eye_radius // 2,
    )

    # Правый глаз
    pygame.draw.circle(
        screen,
        (255, 255, 255),  # Белый цвет
        (head_x + self.grid_size - eye_offset_x, head_y + eye_offset_y),
        eye_radius,
    )
    pygame.draw.circle(
        screen,
        (0, 0, 0),  # Чёрный цвет
        (head_x + self.grid_size - eye_offset_x, head_y + eye_offset_y),
        eye_radius // 2,
    )

    # Рисуем остальные сегменты тела
    for segment in self.body[1:]:
        pygame.draw.rect(
            screen,
            self.color,
            pygame.Rect(segment[0], segment[1], self.grid_size, self.grid_size),
        )
```

Реализация меню

- При запуске отображается главное меню с настройками.
- Игрок может выбрать цвет змейки, тип еды и начать игру.

```python
class MainMenu:
    def __init__(self, screen):
        self.screen = screen
        self.best_score = 0
        self.running = True
        self.font = pygame.font.SysFont("Arial", 36)
        self.small_font = pygame.font.SysFont("Arial", 24)

        # Настройки по умолчанию
        self.selected_snake_color = SNAKE_COLOR
        self.selected_food_image = "image1.png"  # По умолчанию яблоко

    def draw_text(self, text, y, color=FACE_COLOR):
        text_surface = self.font.render(text, True, color)
        text_rect = text_surface.get_rect(center=(WIDTH // 2, y))
        self.screen.blit(text_surface, text_rect)

    def draw_button(self, text, x, y, button_color, text_color):
        button_rect = pygame.Rect(x, y, 150, 50)
        pygame.draw.rect(self.screen, button_color, button_rect, border_radius=10)
        text_surface = self.small_font.render(text, True, text_color)
        text_rect = text_surface.get_rect(center=button_rect.center)
        self.screen.blit(text_surface, text_rect)
        return button_rect
```

Подсчет очков

- Счет увеличивается при поедании еды.
- После завершения игры отображается лучший результат.

```
def draw_score(self):
    score_text = self.font.render(f"Score: {self.score}", True, FACE_COLOR)
    self.screen.blit(score_text, (10, 10))  # Отображаем счёт в левом верхнем углу
```

Выводы и итоги

- Игра успешно реализована с возможностью настройки параметров.
- Использованы принципы объектно-ориентированного программирования.
- Можно добавить дополнительные улучшения: уровни сложности, новые механики.