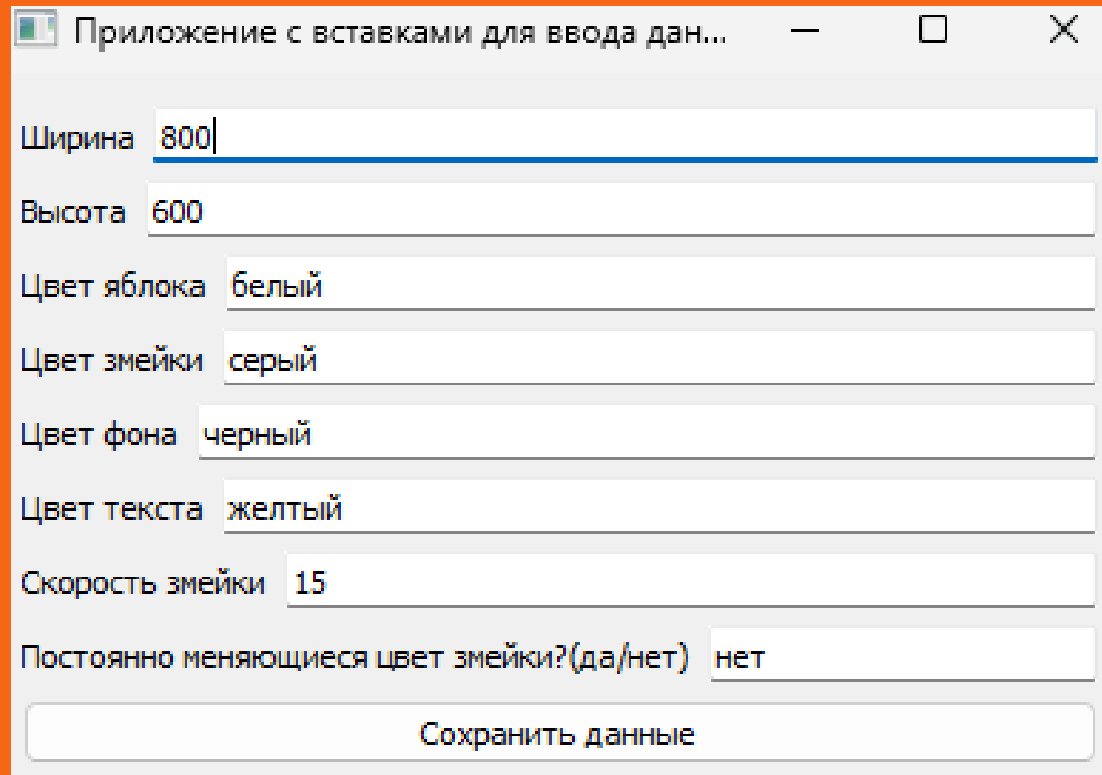


# ИГРА "ЗМЕЙКА"

1. ПРОЕКТ: "ЗМЕЙКА".
2. АВТОР: СВИРИДОВ АЛЕКСАНДР
3. ОПИСАНИЕ ИДЕИ: ДАННЫЙ ПРОЕКТ СОЗДАН МНОЙ ДЛЯ ЗНАКОМСТВА С БИБЛИОТЕКОЙ PYGAME.



ЗАДУМКА НЕЗАМЫСЛОВАТАЯ - СОЗДАНИЕ ИГРЫ "ЗМЕЙКА". ПЕРЕД ИГРОЙ МОЖНО ЗАДАТЬ КОМФОРТНЫЕ ПАРАМЕТРЫ. СНАЧАЛА ВЫБИРАЕМ ЦВЕТА ФОНА, САМОЙ ЗМЕЙКИ, ЯБЛОКА, КОТОРОЕ СОБИРАЕТ ЗМЕЙКА, ТЕКСТА. ИХ НАХОДИМ В ПРИЛОЖЕНИИ, СОЗДАННОМ С ПОМОЩЬЮ БИБЛИОТЕКИ PYQT5. ДАЛЕЕ ВЫБИРАЕМ СКОРОСТЬ ЗМЕЙКИ. ЗАТЕМ ПРОИСХОДИТ ОПРЕДЕЛЕНИЕ ПАРАМЕТРОВ ОКНА - УСТАНОВЛИВАЕТСЯ РАЗМЕР ЯЧЕЙКИ И СОЗДАЕТСЯ ИГРОВОЕ ОКНО, В КОТОРОМ БУДЕТ ПРОИСХОДИТЬ ВСЯ ИГРОВАЯ ДЕЯТЕЛЬНОСТЬ.



Приложение с вставками для ввода дан...

Ширина 800

Высота 600

Цвет яблока белый

Цвет змейки серый

Цвет фона черный

Цвет текста желтый

Скорость змейки 15

Постоянно меняющиеся цвет змейки?(да/нет) нет

Сохранить данные

ОПРЕДЕЛЕНИЕ КЛАССА SNAKE: ЭТОТ КЛАСС СОДЕРЖИТ МЕТОДЫ, КОТОРЫЕ УПРАВЛЯЮТ ДВИЖЕНИЕМ ЗМЕЙКИ, ЕЁ РОСТОМ И ОТРИСОВКОЙ НА ЭКРАНЕ. ЗМЕЙКА ОБЫЧНО СОСТОИТ ИЗ СЕГМЕНТОВ, КАЖДЫЙ ИЗ КОТОРЫХ ИМЕЕТ СВОИ КООРДИНАТЫ И НАПРАВЛЕНИЕ ДВИЖЕНИЯ.

```
# Класс змейки
class Snake:
    def __init__(self):
        self.length = 1
        self.positions = [((WINDOW_WIDTH // 2), (WINDOW_HEIGHT // 2))]
        self.direction = random.choice([UP, DOWN, LEFT, RIGHT])
        self.color = color_of_zmiy

    def get_head_position(self):
        return self.positions[0]

    def move(self):
        cur = self.get_head_position()
        x, y = self.direction
        new = (((cur[0] + (x * CELL_SIZE)) % WINDOW_WIDTH), (cur[1] + (y * CELL_SIZE)))
        if len(self.positions) > 2 and new in self.positions[2:]:
            self.reset()
        else:
            self.positions.insert(0, new)
            if len(self.positions) > self.length:
                self.positions.pop()

    def reset(self):
        global score, maxx
        if int(score) > int(maxx):
            with open('max_score.txt', 'w', encoding='utf8') as file:
                print(str(score).strip(), file=file)
            font = pygame.font.Font(name=None, size=36)
            max_score_text = font.render(text=f"Max Score: {maxx}", antialias=True, color_of_text)
            window.blit(max_score_text, dest=(10, 30))
        score = 0
        self.length = 1
        self.positions = [((WINDOW_WIDTH // 2), (WINDOW_HEIGHT // 2))]
        self.direction = random.choice([UP, DOWN, LEFT, RIGHT])
```

ОПРЕДЕЛЕНИЕ КЛАССА APPLE: В ЭТОМ КЛАССЕ РЕАЛИЗОВАНЫ МЕТОДЫ ДЛЯ ГЕНЕРАЦИИ СЛУЧАЙНОЙ ПОЗИЦИИ ЯБЛОКА НА ЭКРАНЕ И ЕГО ОТРИСОВКИ. ЯБЛОКО ЯВЛЯЕТСЯ ОБЪЕКТОМ, КОТОРЫЙ ЗМЕЙКА ДОЛЖНА СОБИРАТЬ ДЛЯ УВЕЛИЧЕНИЯ СВОЕЙ ДЛИНЫ.

ОПРЕДЕЛЕНИЕ ГЛОБАЛЬНЫХ ПЕРЕМЕННЫХ ДЛЯ НАПРАВЛЕНИЙ ДВИЖЕНИЯ: ЗДЕСЬ ОПРЕДЕЛЯЮТСЯ ГЛОБАЛЬНЫЕ ПЕРЕМЕННЫЕ, КОТОРЫЕ ХРАНЯТ ИНФОРМАЦИЮ О НАПРАВЛЕНИЯХ ДВИЖЕНИЯ ЗМЕЙКИ (ВВЕРХ, ВНИЗ, ВЛЕВО, ВПРАВО). ЭТИ ПЕРЕМЕННЫЕ ИСПОЛЬЗУЮТСЯ ДЛЯ УПРАВЛЕНИЯ ДВИЖЕНИЕМ ЗМЕЙКИ В ИГРЕ.

```
# Класс яблока
class Apple:
    def __init__(self):
        self.position = (0, 0)
        self.color = color_of_apple
        self.randomize_position()

    def randomize_position(self):
        self.position = (random.randint(a: 0, (WINDOW_WIDTH // CELL_SIZE - 1)) * CELL_SIZE,
                        random.randint(a: 0, (WINDOW_HEIGHT // CELL_SIZE - 1)) * CELL_SIZE)

    def draw(self, surface):
        pygame.draw.rect(surface, self.color, rect(self.position[0], self.position[1], CELL_SIZE, CELL_SIZE))

# Глобальные переменные
UP = (0, -1)
DOWN = (0, 1)
LEFT = (-1, 0)
RIGHT = (1, 0)

# Основной игровой цикл
def main():
    global score_text, max_score_text, score, maxx
    with open('max_score.txt', 'r', encoding='utf8') as file:
        maxx = file.read().strip('\n')

    running = True
    clock = pygame.time.Clock()
    snake = Snake()
    apple = Apple()
    score = 0

    while running:
        for event in pygame.event.get():
            if event.type == pygame.QUIT:
                running = False
            elif event.type == pygame.KEYDOWN:
                if event.key == pygame.K_UP:
                    snake.move(UP)
                elif event.key == pygame.K_DOWN:
                    snake.move(DOWN)
                elif event.key == pygame.K_LEFT:
                    snake.move(LEFT)
                elif event.key == pygame.K_RIGHT:
                    snake.move(RIGHT)
            else:
                snake.move()
```

ОСНОВНОЙ ИГРОВОЙ ЦИКЛ (MAIN): ЭТОТ ЦИКЛ СОДЕРЖИТ ОСНОВНУЮ ЛОГИКУ ИГРЫ, ВКЛЮЧАЯ ОБРАБОТКУ СОБЫТИЙ (НАПРИМЕР, НАЖАТИЯ КЛАВИШ), УПРАВЛЕНИЕ ЗМЕЙКОЙ, ВЗАИМОДЕЙСТВИЕ С ЯБЛОКОМ, ОТРИСОВКУ ЭЛЕМЕНТОВ ИГРЫ, ПОДСЧЕТ ОЧКОВ И ОБНОВЛЕНИЕ ЭКРАНА. ЭТОТ ЦИКЛ ПРОДОЛЖАЕТСЯ ДО ТЕХ ПОР, ПОКА ИГРА НЕ ЗАВЕРШИТСЯ



ВЫВОД : В РЕЗУЛЬТАТЕ ПОЛУЧИЛОСЬ ПОНЯТНОЕ И УДОБНОЕ РАЗВЛЕКАТЕЛЬНОЕ  
ПРИЛОЖЕНИЕ.

СПАСИБО ЗА ВНИМАНИЕ!

