Assignment: Snake DIY

20181205 Gina Roe

Github link: https://github.com/Kakucho26/Snake-DIY/

1. Sound & Music

Pygame.mixer provides playing sound and music. Unlike pygame.mixer.Sound, pygame.mixer.music.load is overloaded once you assign it again. You cannot play multiple music throughout the game. Pygame.mixer.Sound is played everytime the corresponding action happens. Pygame.mixer.music is played when instance Game is created, with its argument -1 which means it will be played over and over.

```
# 사운드 전역변수
SOUND_BGM = pygame.mixer.music.load("assets/snake.ogg")
SOUND_MOVE = pygame.mixer.Sound("assets/move.ogg")
SOUND_EATING = pygame.mixer.Sound("assets/eating.ogg")
```

```
    # 뱀 방향 조정
    def control(self, xy):
        if (xy[0] * -1, xy[1] * -1) == self.direction:
            return
        else:
            self.direction = xy
        SOUND_MOVE.play()
```

```
class Game(object):
    def __init__(self):
        self.snake = Snake()
        self.feed = FeedGray()
        self.feed2 = FeedRainbow()
        self.speed = 20
        pygame.mixer.music.play(-1)
```

2. Use image instead of draw.rect() or draw.circle()

The images of two kinds of feed and the snake are loaded the same way.

```
    def draw(self, screen):
        skin = pygame.image.load("assets/snakeskin.png")
```

```
        for i,j in self.positions:
            screen.blit(skin, (i, j))
```

3. Multiple feed

Class FeedRainbow is a chance for those who feel their snake is too long and fast to control. It is instanced in Class Game with the name feed2.

```
# 먹이 객체. 먹을 수록 느려짐
class FeedRainbow(object):
    def __init__(self):
        self.position = (0, 0)
        self.create()

    # 먹이 생성
    def create(self):
        x = random.randint(0, GRID_WIDTH - 1)
        y = random.randint(0, GRID_HEIGHT - 1)
        self.position = x * GRID_SIZE, y * GRID_SIZE

    # 먹이 그리기
    def draw(self, screen):
        feed = pygame.image.load("assets/feed_rainbow.png")
        screen.blit(feed, (self.position[0], self.position[1]))
        # rect = pygame.Rect((self.position[0], self.position[1]), (GRID_SIZE,
GRID_SIZE))
        # pygame.draw.rect(screen, self.color, rect)
```

Connection in class Snake:

```
    def eat2(self):
        if self.length > 2:
            self.length -= 1
            self.positions.pop()
        SOUND_EATING.play()
```

Connection in class Game:

```
    def run_logic(self):
        self.snake.move()
        self.check_eat(self.snake, self.feed, self.feed2)
        self.speed = (20 + self.snake.length) / 4

    # 뱀이 먹이를 먹었는지 체크
    def check_eat(self, snake, feed, feed2):
        if snake.positions[0] == feed.position:
```

```
            snake.eat()
            feed.create()
        if snake.positions[0] == feed2.position:
            snake.eat2()
            feed2.create()
```