코드쌤: 지혜야, 슬기야! 우리가 지금까지 배운 마법들을 총동원해서 드디어 하나의 완성된 게임을 만날 시간이 왔어! 바로 **'동전 수집 게임'**! 이 게임에는 점수도 있고, 목숨도 있고, 게임 오버 화면도 나오고, 심지어 다시 시작하는 기능까지 들어있단다! 정말 기대되지 않니?

지혜: 와! 진짜 게임 같아요! 점수랑 목숨이라니! 제가 좋아하는 게임들도 다 그런 게 있거든요! 🤩

슬기: 게임 오버도 있어요? 😱 그럼 다시 할 수도 있는 거예요? 대박! 빨리 어떻게 만드는 건지 알고 싶어요!

코드쌤: 좋아! 이 게임은 우리가 지금까지 배운 내용들이 어떻게 어우러져 하나의 멋진 작품이 되는지 보여주는 아주 좋은 예시야. 자, 그럼 이 게임의 설계도를 함께 살펴보자!

(코드 조각 1: 게임의 기본 설정 - 무대와 규칙 정하기!)

```
import pygame, random

pygame.init()

WINDOW_WIDTH = 1000 # 화면이 더 넓어졌네!
WINDOW_HEIGHT = 600 # 화면도 더 높아졌고!

window_surface = pygame.display.set_mode((WINDOW_WIDTH, WINDOW_HEIGHT))
pygame.display.set_caption("몬스터의 동전 수집 대작전!") # 캡션 변경!

FPS = 60
clock = pygame.time.Clock()

# --- 게임의 중요한 규칙들을 미리 정해두자! ---
STARTING_COIN_VELOCITY = 5  # 동전이 처음 움직이는 속도
STARTING_PLAYER_LIVES = 5   # 플레이어가 처음 가지는 목숨 수

player_velocity = 5         # 플레이어(몬스터)가 움직이는 속도
coin_velocity = STARTING_COIN_VELOCITY # 현재 동전 속도 (점점 빨라질 거야!)
player_lives = STARTING_PLAYER_LIVES   # 현재 플레이어 목숨
score = 0                   # 현재 점수 (처음엔 0점!)
BUFFER_DISTANCE = 100       # 동전이 화면 밖에서 나타날 때의 여유 거리

# --- 게임에 사용할 색깔들도 미리 이름표를 붙여두자! ---
GREEN = (0, 255, 0)
DARKGREEN = (10, 50, 10)
WHITE = (255, 255, 255)
BLACK = (0, 0, 0)
```

```
# --- 글꼴도 미리 준비! 영어 폰트, 한글 폰트 둘 다! ---
# (NightPumpkind-1GpGv.ttf, MaruBuri-Regular.ttf 파일이 코드와 같은 폴더 또는 지정된 경로에 있어야 해요!)
font = pygame.font.Font("NightPumpkind-1GpGv.ttf", 32)
korean_font = pygame.font.Font("MaruBuri-Regular.ttf", 50)

# --- 게임 화면에 보여줄 글자들을 미리 만들고 위치도 정해두자! (준비물 챙기기!) ---
# 점수 텍스트
score_text = font.render("Score: " + str(score), True, GREEN, DARKGREEN)
score_rect = score_text.get_rect()
score_rect.topleft = (10, 10) # 화면 왼쪽 위에!

# 게임 제목 텍스트 (한글!)
title_text = korean_font.render("동전 수집 게임", True, WHITE, DARKGREEN)
title_rect = title_text.get_rect()
title_rect.top = 10                # 위에서 조금 떨어지고
title_rect.centerx = WINDOW_WIDTH/2 # 가로 중앙에!

# 목숨 텍스트
lives_text = font.render("Lives: " + str(player_lives), True, GREEN, DARKGREEN)
lives_rect = lives_text.get_rect() # score_rect가 아니라 lives_text.get_rect() 여야 할 것 같아요! (설명은
lives_rect.topright = (WINDOW_WIDTH - 10, 10) # 화면 오른쪽 위에!

# 게임 오버 텍스트 (처음엔 안 보이지만 미리 준비!)
game_over_text = korean_font.render("Gave Over", True, GREEN, DARKGREEN) # "Game Over" 오타! (설명은 원본
game_over_rect = game_over_text.get_rect()
game_over_rect.center = (WINDOW_WIDTH/2, WINDOW_HEIGHT/2) # 화면 정중앙에!

# 다시 시작 안내 텍스트 (게임 오버 시에만 보임)
continuous_text = font.render("Press any key to play again", True, WHITE) # continue가 아니라 continuous!
continuous_rect = continuous_text.get_rect()
continuous_rect.center = (WINDOW_WIDTH/2, WINDOW_HEIGHT/2 + 60) # 게임 오버 아래에!

# --- 효과음과 배경음악도 미리 불러오자! ---
# (success.mp3, negative_beeps-6008.mp3, hofman-138068.mp3 파일이 코드와 같은 폴더 또는 지정된 경로에!)
success_sound = pygame.mixer.Sound("success.mp3")
miss_sound = pygame.mixer.Sound("negative_beeps-6008.mp3")
pygame.mixer.music.load("hofman-138068.mp3")
pygame.mixer.music.set_volume(0.5) # 배경음악 볼륨은 50%로!
pygame.mixer.music.play(-1, 0.0)   # 게임 시작과 함께 무한 반복 재생!
```

```
# --- 플레이어(몬스터)와 동전 이미지, 초기 위치 설정! ---
player_image = pygame.image.load("monster.png")
player_rect = player_image.get_rect()
player_rect.midleft = (10, WINDOW_HEIGHT/2) # 플레이어는 왼쪽 중간에서 시작!

coin_image = pygame.image.load("coin.png")
coin_rect = coin_image.get_rect()
# 동전은 처음엔 화면 오른쪽 바깥에서 대기! (BUFFER_DISTANCE 만큼 떨어진 곳)
coin_rect.x = WINDOW_WIDTH + BUFFER_DISTANCE
coin_rect.y = random.randint(64, WINDOW_HEIGHT - 64) # y위치는 위아래 약간의 여유를 두고 무작위!
```

지혜: 와! 게임 시작 전에 준비하는 게 정말 많네요! STARTING_COIN_VELOCITY나 STARTING_PLAYER_LIVES처럼 게임 규칙을 나타내는 변수도 있고, 색깔, 글꼴, 글자 내용이랑 위치까지 다 미리 정해두는군요!

코드쌤: 맞아, 지혜야! 마치 요리하기 전에 레시피를 보고 필요한 재료와 도구를 다 꺼내놓는 것과 같아. 이렇게 게임에 필요한 요소들을 미리 변수로 만들어두고, 초기값을 설정해두면 나중에 게임 규칙을 바꾸거나 코드를 이해하기가 훨씬 쉬워진단다. BUFFER_DISTANCE 같은 건 동전이 갑자기 화면에 뿅! 나타나면 어색하니까, 화면 오른쪽 바깥에서 스르륵~ 등장하도록 여유 공간을 주는 역할을 해.

슬기: 글꼴도 영어 폰트랑 한글 폰트 두 개나 쓰고, 게임 제목은 한글로 “동전 수집 게임”이라고 나오네요! 멋있다! 그리고 `lives_rect = score_text.get_rect()` 이 부분은 `lives_text.get_rect()`가 맞는 것 같아요! game_over_text에 “Gave Over”는 “Game Over” 오타 같기도 하고요! 😆

코드쌤: 오! 슬기 정말 꼼꼼한데! 맞아. 실제 코드를 짤 때는 그런 작은 실수들이 종종 일어나기도 해. 중요한 건 우리가 그걸 발견하고 고칠 수 있다는 거지! (지적해줘서 고마워! 설명에서는 일단 코드에 있는 그대로 따라가되, 그런 부분을 짚어주는 건 아주 좋아!) 그리고 소리랑 음악도 게임 시작하자마자 바로 설정하고 배경음악은 바로 재생시키네! 분위기가 확 살겠는걸?

코드쌤: 자, 이제 게임의 기본 무대와 배우, 소품들이 다 준비됐으니, 본격적으로 게임이 어떻게 흘러가는지 게임 루프 안을 들여다보자!

(코드 조각 2: 게임의 핵심 로직 - 움직이고, 부딪히고, 점수내고!)

```
# ... (이전 코드는 그대로) ...

running = True
while running:
    for event in pygame.event.get():
        if event.type == pygame.QUIT:
            running = False

    keys = pygame.key.get_pressed()
    # 플레이어는 위아래로만 움직이네! 화면 위쪽 경계(64)도 신경쓰고!
```

```
if keys[pygame.K_UP] and player_rect.top > 64: # 위쪽 64픽셀은 못 가게 막았네 (아마 점수판 자리?)
    player_rect.y -= player_velocity
elif keys[pygame.K_DOWN] and player_rect.bottom < WINDOW_HEIGHT:
    player_rect.y += player_velocity

# --- 동전의 움직임과 플레이어와의 상호작용! ---

# 1. 동전이 화면 왼쪽 밖으로 나갔다면? (놓쳤다면!)
if coin_rect.x < 0:
    player_lives -= 1     # 목숨 하나 깎이고!
    miss_sound.play()     # "실패!" 효과음 재생!
    coin_rect.x = WINDOW_WIDTH + BUFFER_DISTANCE # 동전은 다시 오른쪽 바깥에서 대기!
    coin_rect.y = random.randint(64, WINDOW_HEIGHT - 64) # y위치도 새로 무작위로!
# 2. 아직 화면 안에 있다면?
else:
    coin_rect.x -= coin_velocity # 동전은 계속 왼쪽으로 스르륵~ (점점 빨라짐!)

# 3. 플레이어(몬스터)가 동전을 꿀꺽! 했다면? (충돌!)
if player_rect.colliderect(coin_rect):
    score += 1            # 점수 1점 올리고!
    success_sound.play()  # "성공!" 효과음 재생!
    coin_velocity += 1    # 다음 동전은 더 빠르게! (난이도 상승!)
    coin_rect.x = WINDOW_WIDTH + BUFFER_DISTANCE # 동전은 다시 오른쪽 바깥에서 대기!
    coin_rect.y = random.randint(64, WINDOW_HEIGHT - 64) # y위치도 새로!

# --- 목숨을 다 썼다면? 게임 오버! ---
if player_lives == 0:
    pygame.mixer.music.stop() # 슬픈 배경음악은 끄고...
    window_surface.blit(game_over_text, game_over_rect)     # "Game Over" 텍스트 보여주고!
    window_surface.blit(continuous_text, continuous_rect) # "다시 하려면 아무 키나 누르세요" 보여주고
    # 마지막 목숨 상태도 업데이트해서 보여줘야지!
    lives_text = font.render("Lives: " + str(player_lives), True, GREEN, DARKGREEN)
    window_surface.blit(lives_text, lives_rect)
    pygame.display.update() # 게임 오버 화면 업데이트!

    # --- 게임 오버 상태에서는 잠시 멈춰서 키 입력을 기다리자! (새로운 루프!) ---
    is_pause = True
    while is_pause:
        for event in pygame.event.get():
```

```
                if event.type == pygame.QUIT: # 여기서도 창 닫으면 완전히 종료!
                    is_pause = False
                    running = False # 바깥쪽 메인 루프도 꺼야지!

                elif event.type == pygame.KEYDOWN: # 아무 키나 눌리면?
                    # 모든 걸 처음 상태로 되돌리자! (초기화!)
                    score = 0
                    player_lives = STARTING_PLAYER_LIVES
                    coin_velocity = STARTING_COIN_VELOCITY
                    player_rect.midleft = (10, WINDOW_HEIGHT/2) # 플레이어 위치도 처음으로! (원래 코드엔
                    pygame.mixer.music.play(-1, 0.0) # 신나는 배경음악 다시 시작!
                    is_pause = False # "잠시 멈춤" 루프를 빠져나감! (메인 게임 루프로 돌아감)


    # --- 항상 화면에 그려줘야 하는 것들! (점수, 제목, 목숨 등) ---
    # 목숨과 점수는 계속 바뀌니까, 매번 새로 render해서 이미지를 만들어줘야 해!
    lives_text = font.render("Lives: " + str(player_lives), True, GREEN, DARKGREEN)
    score_text = font.render("Score: " + str(score), True, GREEN, DARKGREEN)

    window_surface.fill(BLACK) # 매번 배경을 새로 칠하고!
    window_surface.blit(score_text, score_rect)
    window_surface.blit(title_text, title_rect)
    window_surface.blit(lives_text, lives_rect)

    window_surface.blit(player_image, player_rect) # 플레이어(몬스터) 그리고!
    window_surface.blit(coin_image, coin_rect)   # 동전 그리고!

    pygame.display.update() # 최종 화면을 보여줘!
    clock.tick(FPS)         # 시간아, 흘러라! (일정하게)

pygame.quit() # 메인 루프가 끝나면 게임 종료!
```

지혜: 와! 게임 루프 안에서 정말 많은 일이 일어나네요! 플레이어는 위아래로만 움직이는데, 화면 위쪽 64픽셀은 못 가게 막혀있어요! 아마 거기가 점수랑 제목이 나오는 자리라서 그런가 봐요.

코드쌤: 정확해! 게임 화면을 디자인할 때 이렇게 UI(사용자 인터페이스) 영역과 실제 게임 플레이 영역을 구분하기도 한단다. 그리고 동전의 움직임을 잘 봐봐. 화면 왼쪽으로 사라지면 목숨이 줄고 “실패” 소리가 나지? 이건 플레이어가 동전을 ’놓쳤다’는 뜻이야.

슬기: 그리고 몬스터가 동전이랑 부딪히면(colliderect) 점수가 오르고 “성공” 소리가 나요! 게다가 coin_velocity += 1 이 부분! 동전을 먹을수록 다음 동전이 더 빨리 움직이는 거잖아요! 점점 어려워지겠는데요?

😱

코드쌤: 맞아! 그게 바로 게임의 **난이도 조절** 중 하나야. 플레이어가 게임에 익숙해질수록 조금씩 더 도전적인 상황을 만들어주는 거지. 그리고 동전을 먹거나 놓치면, 동전은 항상 화면 오른쪽 바깥(`WINDOW_WIDTH + BUFFER_DISTANCE`)에서 새로운 y위치로 다시 나타나네.

지혜: `player_lives == 0` 이 되면 `pygame.mixer.music.stop()`으로 배경음악도 멈추고, "Game Over"랑 "다시 하려면 아무 키나 누르세요" 라는 글자가 떠요! 그리고 그 안에서 `while is_pause:` 라는 또 다른 루프가 돌아요! 이건 뭐예요?

코드쌤: 아주 중요한 부분을 발견했어, 지혜! `player_lives == 0`이 되면 게임은 사실상 끝나지만, 바로 종료하지 않고 "게임 오버" 상태에서 잠시 멈추는 거야. 이 `while is_pause:` 루프는 **"아무 키나 누를 때까지 기다리는"** 역할을 해. 만약 창 닫기 버튼을 누르면 게임이 완전히 끝나고, 다른 아무 키나 누르면?

슬기: `score = 0`, `player_lives = STARTING_PLAYER_LIVES`… 아! 모든 점수랑 목숨, 동전 속도, 몬스터 위치까지 다~ 처음 상태로 돌아가고 배경음악도 다시 시작해요! 그리고 `is_pause = False`가 되니까 저 "잠시 멈춤" 루프를 빠져나와서 다시 원래 게임 루프가 돌아가는 거군요! 이게 바로 **'다시 시작'** 기능이네요!

코드쌤: 빙고! 바로 그거야! 게임 오버 후 "다시 도전!" 할 수 있게 만드는 아주 멋진 로직이지. 그리고 게임 루프의 거의 마지막 부분을 보면 `lives_text`와 `score_text`를 매번 새로 `render`하고 있어. 왜 그럴까?

지혜: 아! 목숨이랑 점수는 게임 중에 계속 바뀌는 값이니까, 바뀔 때마다 글자 이미지를 새로 만들어줘야 화면에도 업데이트된 내용이 보이겠네요!

코드쌤: 완벽해! 오늘 우리는 하나의 게임이 어떻게 설계되고, 다양한 요소들이 어떻게 상호작용하는지 정말 깊이 있게 살펴봤어. 게임의 상태(시작, 플레이 중, 게임 오버)에 따라 다른 처리를 하고, 변수들을 활용해서 게임의 규칙과 흐름을 만들고, 사용자에게 정보를 보여주고, 다시 시작하는 기회까지 주는 것! 이게 바로 게임 프로그래밍의 큰 그림이란다!

슬기: 선생님, 오늘 배운 거 전부 다 사용해서 저만의 게임을 만들어보고 싶어요! 몬스터 대신 제가 좋아하는 캐릭터 넣고, 동전 대신 보석 먹는 게임으로요! ✨

코드쌤: 아주 멋진 생각이야, 슬기! 오늘 배운 이 '동전 수집 게임'의 코드를 바탕으로 너희들만의 아이디어를 더해서 새로운 게임을 만들어보는 것, 그게 바로 최고의 코딩 공부란다! 너희들의 멋진 게임을 기대할게!