1. 코드

```
import pygame
import math
import random

# Pygame 초기화
pygame.init()

# 화면 크기 설정
WIDTH, HEIGHT = 800, 600
screen = pygame.display.set_mode((WIDTH, HEIGHT))
pygame.display.set_caption("탄성 충돌 게임")

# 색상
WHITE = (255, 255, 255)
BLACK = (0, 0, 0)
RED = (255, 0, 0)
BLUE = (0, 0, 255)
GREEN = (0, 255, 0)

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

# 물리 엔진 설정
class Ball:
    def __init__(self, x, y, radius, color, mass=1):
        self.x = x
        self.y = y
        self.radius = radius
        self.color = color
        self.mass = mass
        self.vx = random.uniform(-5, 5)
        self.vy = random.uniform(-5, 5)
```

```
    def draw(self):
        pygame.draw.circle(screen, self.color, (int(self.x), int(self.y)), self.radius)

    def move(self):
        self.x += self.vx
        self.y += self.vy

        # 화면 경계 충돌
        if self.x - self.radius < 0 or self.x + self.radius > WIDTH:
            self.vx *= -1
        if self.y - self.radius < 0 or self.y + self.radius > HEIGHT:
            self.vy *= -1

def detect_collision(ball1, ball2):
    dx = ball1.x - ball2.x
    dy = ball1.y - ball2.y
    distance = math.sqrt(dx**2 + dy**2)

    if distance < ball1.radius + ball2.radius:
        return True
    return False

def resolve_collision(ball1, ball2):
    dx = ball1.x - ball2.x
    dy = ball1.y - ball2.y
    distance = math.sqrt(dx**2 + dy**2)

    if distance == 0:
        return

    # 충돌 방향 단위 벡터
    nx = dx / distance
    ny = dy / distance

    # 상대 속도
```

```
    dvx = ball1.vx - ball2.vx
    dvy = ball1.vy - ball2.vy

    # 속도 차원의 투영
    velocity_along_normal = dvx * nx + dvy * ny

    # 물체가 서로 멀어지고 있으면 무시
    if velocity_along_normal > 0:
        return

    # 충돌 반발 계수 (탄성 충돌)
    e = 1.0

    # 충돌량 계산
    j = -(1 + e) * velocity_along_normal
    j /= (1 / ball1.mass + 1 / ball2.mass)

    # 충격량을 각 물체에 적용
    impulse_x = j * nx
    impulse_y = j * ny

    ball1.vx += impulse_x / ball1.mass
    ball1.vy += impulse_y / ball1.mass
    ball2.vx -= impulse_x / ball2.mass
    ball2.vy -= impulse_y / ball2.mass

# 게임 초기화
player = Ball(100, 100, 20, BLUE, mass=2)
goal = pygame.Rect(WIDTH - 100, HEIGHT // 2 - 50, 100, 100)

balls = [Ball(random.randint(50, WIDTH - 50), random.randint(50, HEIGHT - 50),
20, RED) for _ in range(5)]

running = True
while running:
```

```
    screen.fill(WHITE)

    # 종료 이벤트
    for event in pygame.event.get():
        if event.type == pygame.QUIT:
            running = False

    # 플레이어 조작
    keys = pygame.key.get_pressed()
    if keys[pygame.K_LEFT]:
        player.vx -= 0.2
    if keys[pygame.K_RIGHT]:
        player.vx += 0.2
    if keys[pygame.K_UP]:
        player.vy -= 0.2
    if keys[pygame.K_DOWN]:
        player.vy += 0.2

    # 공 움직임
    player.move()
    for ball in balls:
        ball.move()

    # 충돌 처리
    for ball in balls:
        if detect_collision(player, ball):
            resolve_collision(player, ball)
        for other_ball in balls:
            if ball != other_ball and detect_collision(ball, other_ball):
                resolve_collision(ball, other_ball)

    # 목표 지점
    pygame.draw.rect(screen, GREEN, goal)
    if goal.collidepoint(player.x, player.y):
        print("목표 도달!")
```

```
            running = False

        # 공 그리기
        player.draw()
        for ball in balls:
            ball.draw()

        pygame.display.flip()
        clock.tick(FPS)

pygame.quit()
```

2. 코드 설명

1) 초기 설정

- **라이브러리 임포트:**
  - **pygame은 게임 개발에 필요한 그래픽 및 입력 처리를 담당.**
  - **math와 random은 물리 계산과 무작위 속도 초기화를 위해 사용.**
- **화면 설정:**
  - **해상도: WIDTH = 800, HEIGHT = 600.**
  - **게임 화면 생성: pygame.display.set_mode와 pygame.display.set_caption.**

---

2) 공(Ball) 클래스

- **Ball 클래스는 게임 내 공 객체를 정의한다.**
  - **__init__: 공의 초기 위치, 반지름, 색상, 질량, 속도를 설정한다.**
  - **draw: 공을 화면에 그린다.**
  - **move: 속도에 따라 공을 이동시키고, 화면 가장자리에 부딪히면 속도를 반전한다.**

---

3) 충돌 감지와 처리

- **detect_collision(ball1, ball2):**
    - **두 공 사이의 거리 계산한다.**
    - **두 공의 반지름 합보다 거리가 짧으면 충돌로 간주한다.**
- **resolve_collision(ball1, ball2):**
    - **충돌의 물리적 계산을 처리한다.**
    - ****충격량(impulse)**을 계산하여 각 공의 속도를 수정한다.**
    - **단위 벡터 방향과 상대 속도를 이용해 탄성 충돌을 시뮬레이션한다.**

---

4) 게임 초기화

- **플레이어와 목표:**
    - **player: 파란색 공. 초기 위치는 (100, 100)이다.**
    - **goal: 초록색 사각형. 목표 지점은 화면 오른쪽 끝에 있는 정사각형이다..**
- **공:**
    - **빨간 공 5개가 랜덤한 위치에 생성되며, 각각 랜덤한 속도를 가진다.**

---

5) 게임 루프

- **화면 갱신:**
    - **화면을 하얗게 채운다. -> screen.fill(WHITE).**
- **종료 이벤트:**
    - **게임 창을 닫는 이벤트를 처리한다.**
- **플레이어 조작:**
    - **방향키를 이용하여 속도를 수정한다.**
- **공 움직임:**

    - 모든 공(플레이어 및 적)의 위치 업데이트한다.
- 충돌 처리:
    - 플레이어와 적 공 간 충돌을 처리한다.
    - 적 공끼리의 충돌을 처리한다.
- 목표 도달:
    - 플레이어가 목표 지점에 도달하면 메시지를 출력하고 게임을 종료한다.
- 화면 출력:
    - 모든 공과 목표를 그린 후 화면을 갱신한다.

---

6) 종료

- 루프가 끝나면 pygame.quit()으로 Pygame을 종료한다.