```python
import pygame

pygame.init() # 초기화 (반드시 필요)

# 화면 크기 설정
screen_width = 480 # 가로크기
screen_height = 640 # 세로 크기
screen = pygame.display.set_mode((screen_width, screen_height))

# 화면 타이틀 설정
pygame.display.set_caption("Nado Game") # 게임 이름

# 초당 프레임 수 FPS
# 프레임 수가 많으면 화면이 부드러워짐 (움직임이)
clock = pygame.time.Clock()

# 배경 이미지 불러오기
background = pygame.image.load("/Users/yoona/Documents/Python/pygame_bas/background.png")

# 캐릭터(스프라이트) 불러오기
character = pygame.image.load("/Users/yoona/Documents/Python/pygame_bas/zzio.png")
character_size = character.get_rect().size # 이미지의 크기를 구해옴
character_width = character_size[0]  # 캐릭터의 가로 크기
character_height = character_size[1] # 캐릭터의 세로 크기
character_x_pos = (screen_width / 2) - 35   # 화면 가로의 절반 크기에 해당되는 곳에 위치
character_y_pos = screen_height - character_height  # 화면 세로 크기 가장 아래에 해당되는 곳에 위치

# 이동할 좌표
to_x = 0
to_y = 0

# 이동 속도
character_speed = 0.5

# 적 enemy 캐릭터
enemy = pygame.image.load("/Users/yoona/Documents/Python/pygame_bas/enemy.png")
enemy_size = enemy.get_rect().size # 이미지의 크기를 구해옴
enemy_width = enemy_size[0]  # 캐릭터의 가로 크기
enemy_height = enemy_size[1] # 캐릭터의 세로 크기
enemy_x_pos = (screen_width / 2) - (enemy_width / 2)  # 화면 가로의 절반 크기에 해당되는 곳에 위치
enemy_y_pos = (screen_height / 2) - (enemy_height / 2) # 화면 세로 크기 가장 아래에 해당되는 곳에 위치


# 폰트 정의
game_font = pygame.font.Font(None, 40) # 폰트 객체 생성 (폰트, 크기)

# 총 시간
total_time = 10

# 시작 시간
start_ticks = pygame.time.get_ticks() # 시작 tick 을 받아옴


# 이벤트 루프
running = True # 게임이 진행중인가를 확인
```

```
while running :
    dt = clock.tick(60) # 게임 화면의 초당 프레임 수를 결정

    for event in pygame.event.get():  # 어떤 이벤트가 발생하였는가?
        if event.type == pygame.QUIT: # 창을 닫는 버튼을 누르면
            running = False  # 게임이 진행중이 아님

        if event.type == pygame.KEYDOWN:      # 키가 눌러졌는지 확인
            if event.key == pygame.K_LEFT:    # 캐릭터를 왼쪽으로
                to_x -= character_speed  # to_x = to_x -5
            elif event.key == pygame.K_RIGHT: # 캐릭터를 오른쪽으로
                to_x += character_speed
            elif event.key == pygame.K_UP:    # 캐릭터를 위로
                to_y -= character_speed
            elif event.key == pygame.K_DOWN:  # 캐릭터를 아래로
                to_y += character_speed

        if event.type == pygame.KEYUP: # 방향키를 떼면 멈춤
            if event.key == pygame.K_LEFT or event.key == pygame.K_RIGHT:
                to_x = 0
            elif event.key == pygame.K_UP or event.key == pygame.K_DOWN:
                to_y = 0

    character_x_pos += to_x * dt  # fps 변화에 상관없이 게임 속도 일정하게 유지시킬 수 있음
    character_y_pos += to_y * dt

    # 캐릭터 화면 밖으로 못나가게
    # 가로 경계값 처리
    if character_x_pos < 0:
        character_x_pos = 0
    elif character_x_pos > screen_width - character_width:
        character_x_pos = screen_width - character_width

    # 세로 경계값 처리
    if character_y_pos < 0:
        character_y_pos = 0
    elif character_y_pos > screen_height - character_height:
        character_y_pos = screen_height - character_height

    # 충돌 처리를 위한 rect 정보 업데이트
    character_rect = character.get_rect()
    character_rect.left = character_x_pos
    character_rect.top = character_y_pos

    enemy_rect = enemy.get_rect()
    enemy_rect.left = enemy_x_pos
    enemy_rect.top = enemy_y_pos

    # 충돌 체크
    if character_rect.colliderect(enemy_rect): # 충돌 했는지를 확인
        print("충돌했어요")
        running = False


    screen.blit(background, (0, 0)) # 배경 그리기
    screen.blit(character, (character_x_pos, character_y_pos)) # 캐릭터 그리기
    screen.blit(enemy, (enemy_x_pos, enemy_y_pos))  # 적 그리기
```

```
    # 타이머 집어 넣기
    # 경과 시간 계산
    elapsed_time = (pygame.time.get_ticks() - start_ticks) / 1000
    # 경과 시간(ms)을 1000 으로 나누어서 초(s) 단위로 표시

    timer = game_font.render(str(int(total_time - elapsed_time)), \
        True, (255, 255, 255))  # 총 시간에서 흐른 시간을 뺀 것 10,9,8..
        # 출력할 글자, True, 글자 색상(흰색)
    screen.blit(timer, (10, 10))

    # 시간이 0 이하이면 게임 종료
    if total_time - elapsed_time <= 0:
        print("Time Out")
        running = False


    pygame.display.update() # while 문을 돌면서 계속 게임화면을 다시 그리는 것

# 잠시 대기
pygame.time.delay(2000) # 2 초 정도 대기 (ms)

# pygame 종료
pygame.quit()
```