15_collisions.py

```
# Most of this code is copied from programs 11 and 13

WIDTH = 500

alien = Actor("alien")
alien.pos = (0, 50)
box = Rect((20, 20), (100, 100))

def draw():
    screen.clear()
    screen.draw.filled_rect(box, "red")
    alien.draw()

def update():
    if keyboard.right:
        alien.x = alien.x + 2
    elif keyboard.left:
        alien.x = alien.x - 2
    box.x = box.x + 2
    if box.x > WIDTH:
        box.x = 0
    if alien.colliderect(box):
        print("hit")


# TODO
# joystick input (again), vertical movement (again)
# make the box chase the alien
# print number of times hit (the score)
```

16_collisions2_sound_animation.py

```
# Most of this code is copied from program 15
WIDTH = 500
alien = Actor("alien")
alien.pos = (0, 50)
box = Rect((20, 20), (100, 100))

def draw():
    screen.clear()
    screen.draw.filled_rect(box, "red")
    alien.draw()
def update():
    if keyboard.right:
        alien.x = alien.x + 2
    elif keyboard.left:
        alien.x = alien.x - 2
    box.x = box.x + 2
    if box.x > WIDTH:
        box.x = 0
# PLAY SOUND AND SHOW IMAGE WHEN HIT
    if alien.colliderect(box):
        alien.image = 'alien_hurt'
        sounds.eep.play()
    else:
        alien.image = 'alien'
# TODO
# Record your own sound effect
# Add more boxes or sprites that move in different ways to avoid
# Add a second alien controlled by different keys or gamepad
```

17_mouse_input.py

```
# similiar to program 16 but
#   * box has been removed
#   * mouse function for clicking on alien
#   * score display

alien = Actor("alien")
alien.pos = (0, 50)
score = 0

def draw():
    screen.clear()
    alien.draw()
    screen.draw.text("Score "+str(score), (0,0))

def update():
    if keyboard.right:
        alien.x = alien.x + 2
    elif keyboard.left:
        alien.x = alien.x - 2
    alien.image = 'alien'

def on_mouse_down(pos, button):
    global score
    if button == mouse.LEFT and alien.collidepoint(pos):
        alien.image = 'alien_hurt'
        sounds.eep.play()
        score = score + 1
```

18_mouse_movement.py

```
# wiggle your mouse around the screen!

alien = Actor("alien")

def draw():
    screen.clear()
    alien.draw()

def on_mouse_move(pos):
    alien.pos = pos

#TODO:
# what happens if you delete line 8 and replace it with this:
#
#   animate(alien, pos=pos, duration=1, tween='bounce_end')
#
# what happens if you change on_mouse_move to on_mouse_down?
# can you make a game with one alien controlled by mouse
#    and another controlled by keyboard?
```

19_joystick_tester.py

```
# Example of controller input and example of for loops but
# mostly here so I can test your controllers.
# YOU DONT NEED TO TYPE THIS ONE (unless you really want to)
import pygame

def update():
    screen.clear()
    joystick_count = pygame.joystick.get_count()
    y = 0
    for i in range(joystick_count):
        joystick = pygame.joystick.Joystick(i)
        joystick.init()
        name = joystick.get_name()
        axes = joystick.get_numaxes()
        buttons = joystick.get_numbuttons()
        hats = joystick.get_numhats()
        screen.draw.text(
            "Joystick {} name: {} axes:{} buttons:{} hats:{}".format(
                i, name, axes, buttons, hats), (0, y))
        y += 14
        for i in range(axes):
            axis = joystick.get_axis(i)
            screen.draw.text("Axis {} value: {:>6.3f}".format(i, axis), (20, y))
            y += 14
        for i in range(buttons):
            button = joystick.get_button(i)
            screen.draw.text("Button {:>2} value: {}".format(i, button), (20, y))
            y += 14
        for i in range(hats):
            hat = joystick.get_hat(i)
            screen.draw.text("Hat {} value: {}".format(i, str(hat)), (20, y))
            y += 14
```

20_loops.py

```
# draw red cirlces using a loop
# draw green circles using a loop within another loop

WIDTH = 500
HEIGHT = 500

def draw():
    screen.clear()
    for x in range(0, WIDTH, 40):
        screen.draw.filled_circle((x, 20), 20, "red")

    for x in range(0, WIDTH, 40):
        for y in range(60, HEIGHT, 40):
            screen.draw.filled_circle((x, y), 10, "green")

#TODO:
# import random and make the positions random, e.g.
#     random.randint(0, 100)
# learn about RGB colour and make random colours (difficult)
# create a timer variable and change colours based on time (difficult)
```

21_arrays.py

```
# Create an empty array, use a loop to fill it with aliens
# Draw the aliens, move the aliens
# Add a new alien when the mouse is clicked

WIDTH = 500

aliens = []
for i in range(0,10):
    aliens.append(Actor('alien', (i*30, i*30)))

def draw():
    screen.clear()
    for alien in aliens:
        alien.draw()

def update():
    for alien in aliens:
        alien.x += 2
        if alien.x > WIDTH:
            alien.x = 0

def on_mouse_down(pos, button):
    aliens.append(Actor('alien', pos))

#TODO:
# Go back to your previous game (e.g. program 16)
# make an array of bullets that shoot when you
# press the space bar
```

22_animation.py

```
# Most of this code is copied from program 15

alien = Actor("alien")
alien.pos = (200, 200)

def draw():
    screen.clear()
    alien.draw()

def update():
    if keyboard.right:
        alien.x = alien.x + 2
    elif keyboard.left:
        alien.x = alien.x - 2

images = ["alien_hurt", "alien"]
image_counter = 0

def animateAlien():
    global image_counter
    alien.image = images[image_counter % len(images)]
    image_counter += 1

clock.schedule_interval(animateAlien, 0.2)

# TODO
# make the alien animate faster
# add another image to list of images
# draw your own animation, e.g. a man walking left
#   and make it play when the left key is pressed
```

23_simple_physics.py

```
# Make a ball that bounces using simple velocity vector (vx and vy)

WIDTH = 500
HEIGHT = 500

ball = Rect((200, 400), (20, 20))
vx = 1
vy = 1

def draw():
    screen.clear()
    screen.draw.filled_rect(ball, "red")

def update():
    global vx, vy
    ball.x += vx
    ball.y += vy
    if ball.right > WIDTH or ball.left < 0:
        vx = -vx
    if ball.bottom > HEIGHT or ball.top < 0:
        vy = -vy

#TODO
# Make the ball get faster each time it hits the sides
```

24_pong.py

```
# Classic bat and ball game

WIDTH = 500
HEIGHT = 500

ball = Rect((150, 400), (20, 20))
bat = Rect((200, 480), (60, 20))
vx = 4
vy = 4

def draw():
    screen.clear()
    screen.draw.filled_rect(ball, "red")
    screen.draw.filled_rect(bat, "white")

def update():
    global vx, vy
    ball.x += vx
    ball.y += vy
    if ball.right > WIDTH or ball.left < 0:
        vx = -vx
    if ball.colliderect(bat) or ball.top < 0:
        vy = -vy
    if ball.bottom > HEIGHT:
        exit()
    if(keyboard.right):
        bat.x += 2
    elif(keyboard.left):
        bat.x -= 2

#TODO
# Add another bat at the top of the screen for player 2
# Add bricks (Rects) that disappear when the ball hits them
```