```
    |0  |1  |2  |3  |4  |5  |6  |7  |8
1   #!/usr/bin/python
2   # Snapper
3   # Code Angel
4
5   import sys
6   import os
7   import pygame
8   from pygame.locals import *
9   import random
10
11  # Define the colours
12  DARK_GREEN = (0, 98, 7)
13  DARK_GREY = (70, 70, 70)
14  WHITE = (255, 255, 255)
15
16  # Define constants
17  SCREEN_WIDTH = 640
18  SCREEN_HEIGHT = 480
19  SCOREBOARD_HEIGHT = 24
20  SCOREBOARD_MARGIN = 4
21
22  # Camera viewfinder constants
23  CAM_LEFT_BORDER = 9
24  VIEWFINDER_WIDTH = 44
25  CAM_TOP_BORDER = 21
26  VIEWFINDER_HEIGHT = 30
27
28  GAME_LIVES = 3
29
30  # Setup
31  os.environ['SDL_VIDEO_CENTERED'] = '1'
32  pygame.mixer.pre_init(44100, -16, 2, 512)
33  pygame.mixer.init()
34  pygame.init()
35  game_screen = pygame.display.set_mode((SCREEN_WIDTH, SCREEN_HEIGHT))
36  pygame.display.set_caption('Snapper')
37  clock = pygame.time.Clock()
38  font = pygame.font.SysFont('Arial Narrow Bold', 24)
39
40
41  # Load images
    |0  |1  |2  |3  |4  |5  |6  |7  |8
```

```
    |0  |1  |2  |3  |4  |5  |6  |7  |8
background_image = pygame.image.load('background.png').convert()
foreground_image = pygame.image.load('foreground.png').convert_alpha()
camera_image = pygame.image.load('camera.png').convert_alpha()
camera_flash_image = pygame.image.load('camera_flash.png').convert_alpha()
lives_image = pygame.image.load('camera_lives.png').convert_alpha()
snap_image = pygame.image.load('snap.png').convert_alpha()
miss_image = pygame.image.load('miss.png').convert_alpha()
rabbit_image = pygame.image.load('rabbit.png').convert_alpha()
owl_image = pygame.image.load('owl.png').convert_alpha()
deer_image = pygame.image.load('deer.png').convert_alpha()
squirrel_image = pygame.image.load('squirrel.png').convert_alpha()

# Load sounds
camera_sound = pygame.mixer.Sound('click.ogg')
miss_sound = pygame.mixer.Sound('miss.ogg')


def main():

    # Initialise variables
    mouse_button_pressed = False

    snap_visible = False
    miss_visible = False

    pygame.mouse.set_visible(False)

    animal_rect = pygame.Rect(0, 0, 0, 0)

    # Dictionary to store the animals
    animals = {
                'animal_1': {'type': 'rabbit', 'x_loc': 290, 'y_loc': 120, 'time': 60, 'points': 10},
                'animal_2': {'type': 'rabbit', 'x_loc': 382, 'y_loc': 318, 'time': 60, 'points': 10},
                'animal_3': {'type': 'rabbit', 'x_loc': 96, 'y_loc': 304, 'time': 60, 'points': 10},
                'animal_4': {'type': 'rabbit', 'x_loc': 358, 'y_loc': 159, 'time': 60, 'points': 10},
                'animal_5': {'type': 'rabbit', 'x_loc': 466, 'y_loc': 155, 'time': 60, 'points': 10},
                'animal_6': {'type': 'rabbit', 'x_loc': 202, 'y_loc': 297, 'time': 60, 'points': 10},
                'animal_7': {'type': 'rabbit', 'x_loc': 265, 'y_loc': 318, 'time': 60, 'points': 10},
                'animal_8': {'type': 'rabbit', 'x_loc': 367, 'y_loc': 344, 'time': 60, 'points': 10},
                'animal_9': {'type': 'owl', 'x_loc': 387, 'y_loc': 46, 'time': 75, 'points': 5},
                'animal_10': {'type': 'owl', 'x_loc': 295, 'y_loc': 47, 'time': 75, 'points': 5},
    |0  |1  |2  |3  |4  |5  |6  |7  |8
```



```
    |0  |1  |2  |3  |4  |5  |6  |7  |8
                'animal_11': {'type': 'owl', 'x_loc': 474, 'y_loc': 235, 'time': 75, 'points': 5},
                'animal_12': {'type': 'owl', 'x_loc': 574, 'y_loc': 132, 'time': 75, 'points': 5},
                'animal_13': {'type': 'owl', 'x_loc': 23, 'y_loc': 126, 'time': 75, 'points': 5},
                'animal_14': {'type': 'squirrel', 'x_loc': 17, 'y_loc': 113, 'time': 40, 'points': 20},
                'animal_15': {'type': 'squirrel', 'x_loc': 567, 'y_loc': 124, 'time': 40, 'points': 20},
                'animal_16': {'type': 'squirrel', 'x_loc': 448, 'y_loc': 278, 'time': 40, 'points': 20},
                'animal_17': {'type': 'squirrel', 'x_loc': 452, 'y_loc': 359, 'time': 40, 'points': 20},
                'animal_18': {'type': 'squirrel', 'x_loc': 304, 'y_loc': 301, 'time': 40, 'points': 20},
                'animal_19': {'type': 'squirrel', 'x_loc': 62, 'y_loc': 279, 'time': 40, 'points': 20},
                'animal_20': {'type': 'deer', 'x_loc': 106, 'y_loc': 87, 'time': 90, 'points': 1},
                'animal_21': {'type': 'deer', 'x_loc': 268, 'y_loc': 84, 'time': 90, 'points': 1},
                'animal_22': {'type': 'deer', 'x_loc': 302, 'y_loc': 90, 'time': 90, 'points': 1},
                'animal_23': {'type': 'deer', 'x_loc': 392, 'y_loc': 127, 'time': 90, 'points': 1}
    }

    animal = get_random_animal(animals)
    animal_timer = int(animal.get('time'))
    no_animal_timer = 0

    animal_visible = True

    score = 0
    lives = GAME_LIVES
    hi_score = 0

    # Main game loop
    while True:

        # Check for mouse and key presses
        for event in pygame.event.get():

            # Mouse button clicked
            mouse_button_pressed = False
            if event.type == pygame.MOUSEBUTTONDOWN:
                mouse_button_pressed = True

            # Return key pressed when game over
            key_pressed = pygame.key.get_pressed()
            if key_pressed[pygame.K_RETURN] and lives == 0:
                if score > hi_score:
                    hi_score = score
    |0  |1  |2  |3  |4  |5  |6  |7  |8
```

```
    |0  |1  |2  |3  |4  |5  |6  |7  |8
124
125                 lives = GAME_LIVES
126                 score = 0
127
128                 animal = get_random_animal(animals)
129                 animal_timer = int(animal.get('time'))
130                 no_animal_timer = 0
131
132                 animal_visible = True
133
134                 snap_visible = False
135                 miss_visible = False
136
137             # User quits
138             if event.type == QUIT:
139                 pygame.quit()
140                 sys.exit()
141
142         # Set the camera centre to the location of the mouse pointer
143         mouse_pos = pygame.mouse.get_pos()
144
145         camera rect = camera image.get rect()
146         camera_rect.centerx = mouse_pos[0]
147         camera_rect.centery = mouse_pos[1]
148
149         camera_width = camera_image.get_width()
150         camera_height = camera_image.get_height()
151
152         # Prevent the camera going off the screen
153         if camera_rect.centerx < camera_width / 2:
154             camera_rect.centerx = camera_width / 2
155         if camera_rect.centerx > SCREEN_WIDTH - camera_width / 2:
156             camera_rect.centerx = SCREEN_WIDTH - camera_width / 2
157
158         if camera_rect.centery < camera_height / 2 + SCOREBOARD_HEIGHT:
159             camera_rect.centery = camera_height / 2 + SCOREBOARD_HEIGHT
160         if camera_rect.centery > SCREEN_HEIGHT - camera_height / 2:
161             camera_rect.centery = SCREEN_HEIGHT - camera_height / 2
162
163         # Calculate the camera's viewfinder rectangle
164         viewfinder_left = camera_rect.left + CAM_LEFT_BORDER
    |0  |1  |2  |3  |4  |5  |6  |7  |8
```

```
    |0  |1  |2  |3  |4  |5  |6  |7  |8
165         viewfinder_top = camera_rect.top + CAM_TOP_BORDER
166         viewfinder_rect = pygame.Rect(viewfinder_left, viewfinder_top, VIEWFINDER_WIDTH, VIEWFINDER_HEIGHT)
167
168         # If there is an animal visible, decrease the animal_timer and work out the animal rect
169         if animal_visible is True:
170             animal_timer -= 1
171
172             # If the animal timer reaches zero, hide the animal and pause
173             if animal_timer == 0:
174                 no_animal_timer = random.randint(30, 120)
175                 animal_visible = False
176
177             animal_x = int(animal.get('x_loc'))
178             animal_y = int(animal.get('y_loc'))
179
180             if animal.get('type') == 'rabbit':
181                 animal_rect = pygame.Rect(animal_x, animal_y, rabbit_image.get_width(), rabbit_image.get_height())
182             elif animal.get('type') == 'owl':
183                 animal_rect = pygame.Rect(animal_x, animal_y, owl_image.get_width(), owl_image.get_height())
184             elif animal.get('type') == 'deer':
185                 animal_rect = pygame.Rect(animal_x, animal_y, deer_image.get_width(), deer_image.get_height())
186             else:
187                 animal_rect = pygame.Rect(animal_x, animal_y, squirrel_image.get_width(),
188 squirrel_image.get_height())
189
190         # Countdown the no animal timer, and when it hits zero get a new animal
191         if animal_visible is False:
192             no_animal_timer -= 1
193
194             if no_animal_timer == 0:
195                 if lives > 0:
196                     animal = get_random_animal(animals)
197                     animal_timer = int(animal.get('time'))
198                     animal_visible = True
199
200                 snap_visible = False
201                 miss_visible = False
202
203         # The player has clicked the mouse to take a photograph
204         if mouse_button_pressed is True:
205             if snap_visible is False and miss_visible is False and lives > 0:
    |0  |1  |2  |3  |4  |5  |6  |7  |8
```

```
    |0  |1  |2  |3  |4  |5  |6  |7  |8
206
207                 # Check to see whether they got the animal in the viewfinder, and that the animal is visible
208                 if viewfinder_rect.colliderect(animal_rect):
209                     if animal_visible is True:
210                         score += animal_timer * int(animal.get('points'))
211                         snap_visible = True
212                         camera_sound.play()
213
214                     else:
215                         miss_visible = True
216                         lives -= 1
217                         miss_sound.play()
218
219                 else:
220                     miss_visible = True
221                     lives -= 1
222                     miss_sound.play()
223
224                 # Hide the animal and pause
225                 animal_visible = False
226                 animal_timer = 0
227                 no animal timer = 120
228
229         # Draw background
230         game_screen.blit(background_image, [0, 0])
231
232         # If there is an animal visible, draw animal
233         if animal_visible is True:
234             animal_x = int(animal.get('x_loc'))
235             animal_y = int(animal.get('y_loc'))
236
237             # Blit the correct animal onto the screen on top of background but below foreground
238             if animal.get('type') == 'rabbit':
239                 game_screen.blit(rabbit_image, [animal_x, animal_y])
240             elif animal.get('type') == 'owl':
241                 game_screen.blit(owl_image, [animal_x, animal_y])
242             elif animal.get('type') == 'deer':
243                 game_screen.blit(deer_image, [animal_x, animal_y])
244             else:
245                 game_screen.blit(squirrel_image, [animal_x, animal_y])
246
    |0  |1  |2  |3  |4  |5  |6  |7  |8
```



```
    |0  |1  |2  |3  |4  |5  |6  |7  |8
247         # Draw the foreground overlay image
248         game_screen.blit(foreground_image, [0, 0])
249
250         # Draw Camera
251         if snap_visible is True or miss_visible is True:
252             game_screen.blit(camera_flash_image, camera_rect)
253         else:
254             game_screen.blit(camera_image, camera_rect)
255
256         # Draw the snap or miss image
257         snap_or_miss_border = (VIEWFINDER_WIDTH - snap_image.get_width()) / 2
258         snap_or_miss_rect = pygame.Rect(viewfinder_left + snap_or_miss_border, viewfinder_top,
259                                         snap_image.get_width(), snap_image.get_height())
260
261         if snap_visible is True:
262             game_screen.blit(snap_image, snap_or_miss_rect)
263         elif miss_visible is True:
264             game_screen.blit(miss_image, snap_or_miss_rect)
265
266         # Display score board
267         score_text = 'Score: ' + str(score)
268         display_scoreboard_data(score_text, 'Left')
269
270         hi_score_text = 'Hi: ' + str(hi_score)
271         display_scoreboard_data(hi_score_text, 'Centre')
272
273         # Display the lives remeaining
274         for life in range(1, lives + 1):
275             life_xloc = SCREEN_WIDTH - life * (lives_image.get_width() + 2 * SCOREBOARD_MARGIN)
276             life_y_loc = SCREEN_HEIGHT - SCOREBOARD_HEIGHT
277             game_screen.blit(lives_image, [life_xloc, life_y_loc])
278
279         if lives == 0:
280             display_game_over()
281
282         pygame.display.update()
283         clock.tick(60)
284
285
286 # Get a random animal from the dictionary
287 def get_random_animal(animals):
    |0  |1  |2  |3  |4  |5  |6  |7  |8
```

```
    |0  |1  |2  |3  |4  |5  |6  |7  |8
    random_animal = random.choice(list(animals.keys()))
    return animals.get(random_animal)


# Handle the text display
def display_scoreboard_data(scoreboard_text, alignment):
    display_text = font.render(scoreboard_text, True, WHITE)
    text_rect = display_text.get_rect()

    text_loc = [0, 0]

    if alignment == 'Left':
        text_loc = [SCOREBOARD_MARGIN, SCREEN_HEIGHT - SCOREBOARD_HEIGHT]

    elif alignment == 'Centre':
        text_loc = [(SCREEN_WIDTH - text_rect.width) / 2, SCREEN_HEIGHT - SCOREBOARD_HEIGHT]

    game_screen.blit(display_text, text_loc)


# Display end of game message
def display_game_over():

    game_over_rect = (3 * SCOREBOARD_HEIGHT, 8 * SCOREBOARD_HEIGHT,
                      SCREEN_WIDTH - SCOREBOARD_HEIGHT * 6, SCOREBOARD_HEIGHT * 5)

    pygame.draw.rect(game_screen, DARK_GREEN, game_over_rect)

    text_line_1 = font.render('GAME OVER', True, WHITE)
    text_rect_1 = text_line_1.get_rect()
    text_line_1_loc = [(SCREEN_WIDTH - text_rect_1.width) / 2, (SCREEN_HEIGHT / 2) - 16]

    text_line_2 = font.render('Hit RETURN for a new game', True, WHITE)
    text_rect_2 = text_line_2.get_rect()
    text_line_2_loc = [(SCREEN_WIDTH - text_rect_2.width) / 2, (SCREEN_HEIGHT / 2) + 16]

    game_screen.blit(text_line_1, text_line_1_loc)
    game_screen.blit(text_line_2, text_line_2_loc)


    |0  |1  |2  |3  |4  |5  |6  |7  |8
```

```
    |0  |1  |2  |3  |4  |5  |6  |7  |8
328 if __name__ == '__main__':
329     main()
330
    |0  |1  |2  |3  |4  |5  |6  |7  |8
```