```
    |0  |1  |2  |3  |4  |5  |6  |7  |8
 1  # Snake Heart
 2  # Code Angel
 3
 4  # Classes: Player
 5
 6  import pygame
 7
 8  import screen
 9  import utils
10
11  # Define constants
12  PLAYER_WIDTH = 32
13  PLAYER_HEIGHT = 32
14  PLAYER_MOVE = 4
15
16
17  # Player class
18  class Player:
19
20      # Class variables
21      max_lives = 5
22      max_spades = 3
23
24      def __init__(self):
25
26          # Load images
27          self.player_still_image = utils.load_media('image', 'player_still')
28
29          player_down_image = utils.load_media('image', 'player_down')
30          player_down_alt_image = utils.load_media('image', 'player_down_alt')
31          self.player_down_images = [player_down_image, player_down_alt_image]
32
33          player_right_image = utils.load_media('image', 'player_right')
34          player_right_alt_image = utils.load_media('image', 'player_right_alt')
35          self.player_right_images = [player_right_image, player_right_alt_image]
36
37          player_left_image = utils.load_media('image', 'player_left')
38          player_left_alt_image = utils.load_media('image', 'player_left_alt')
39          self.player_left_images = [player_left_image, player_left_alt_image]
40
41          player_up_image = utils.load_media('image', 'player_up')
    |0  |1  |2  |3  |4  |5  |6  |7  |8
```

```
    |0  |1  |2  |3  |4  |5  |6  |7  |8
42          player_up_alt_image = utils.load_media('image', 'player_up_alt')
43          self.player_up_images = [player_up_image, player_up_alt_image]
44
45          self.player_dig_image = utils.load_media('image', 'player_dig')
46          self.skeleton_image = utils.load_media('image', 'skeleton')
47
48          # Load audio
49          self.dig_sound = utils.load_media('audio', 'dig')
50          self.win_game_sound = utils.load_media('audio', 'win_game')
51          self.lose_life_sound = utils.load_media('audio', 'lose_life')
52
53          self.image = self.player_still_image
54          self.rect = pygame.Rect(screen.SCREENWIDTH / 2, screen.SCREENHEIGHT / 2, PLAYER_WIDTH, PLAYER_HEIGHT)
55          self.direction = 'none'
56
57          self.gold = 0
58          self.lives = 5
59          self.spades = 0
60
61          self.sword = [False, False, False, False]
62          self.alive = True
63          self.game_over_win = False
64          self.game_over_lose = False
65
66          self.image_num = 0
67          self.dig_timer = 0
68          self.time_to_dig = 30
69          self.skeleton_time = 0
70
71      # Draw the player
72      def draw(self, display):
73
74          if self.direction == 'right':
75              self.image = self.player_right_images[self.image_num]
76              self.image_num += 1
77
78          elif self.direction == 'left':
79              self.image = self.player_left_images[self.image_num]
80              self.image_num += 1
81
82          elif self.direction == 'up':
    |0  |1  |2  |3  |4  |5  |6  |7  |8
```



```
    |0  |1  |2  |3  |4  |5  |6  |7  |8
            self.image = self.player_up_images[self.image_num]
            self.image_num += 1

        elif self.direction == 'down':
            self.image = self.player_down_images[self.image_num]
            self.image_num += 1

        elif self.dig_timer > 0:
            self.image = self.player_dig_image

        elif self.alive is False:
            self.image = self.skeleton_image

        else:
            self.image = self.player_still_image

        if self.image_num >= len(self.player_up_images):
            self.image num = 0

        display.show_image(self.image, self.rect.x, self.rect.y)

    # Direction updated when the player presses a key (left, right, up, down)
    def set_direction(self, direction):
        self.direction = direction

    # Start digging if the player has pressed the space bar
    # Only if Lucy is not already digging, is still alive, and has at least 1 spade
    def start_digging(self):
        if self.dig_timer == 0 and self.alive is True and self.spades > 0:
            self.dig_sound.play()
            self.dig_timer = self.time_to_dig
            self.spades -= 1

    # Update dig timer
    def dig(self, game_map):
        self.dig_timer -= 1

        # Once the timer reaches 0, draw the trap
        if self.dig_timer == 0:
            game_map.add_trap()

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

```
    |0  |1  |2  |3  |4  |5  |6  |7  |8
124     # The Map object has identified Lucy has walked into water
125     def map_water(self):
126         if self.alive is True:
127             self.die()
128
129     # The Map object has identified Lucy has collected a coin
130     def map_gold(self):
131         self.gold += 1
132
133     # The Map object has identified Lucy has collected a heart
134     def map_heart(self):
135         self.lives += 1
136
137     # The Map object has identified Lucy has collected a spade
138     def map_spade(self):
139         self.spades += 1
140
141     # The Map object has identified Lucy has collected a sword part
142     def map_sword(self, sword_number):
143         self.sword[sword_number - 1] = True
144
145     # The Map object has identified Lucy has walked up to the castle
146     def map_castle(self):
147
148         # The game will be won if:
149         # All parts of the Snake Heart sword have been collected and
150         # the game is not already over
151         if all(sword_part is True for sword_part in self.sword) and self.game_over_win is False:
152             self.game_over_win = True
153             self.win_game_sound.play()
154
155     # Test if Lucy has collided with a monster
156     def check_collision(self, monster):
157         if self.rect.colliderect(monster.rect) and self.alive is True:
158             self.die()
159
160     def die(self):
161         self.lives -= 1
162         self.alive = False
163         self.lose_life_sound.play()
164
    |0  |1  |2  |3  |4  |5  |6  |7  |8
```



```
    |0  |1  |2  |3  |4  |5  |6  |7  |8
165             if self.lives > 0:
166                 self.skeleton_time = 30
167             else:
168                 self.game_over_lose = True
169
170         def skeleton(self, game_map):
171             self.skeleton_time -= 1
172
173             if self.skeleton_time == 0 and self.lives > 0:
174                 self.alive = True
175
176                 # After a life has been lost, teleport to a new location
177                 game_map.portal_move()
178
```

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

