```
    |0  |1  |2  |3  |4  |5  |6  |7  |8
1   # Snake Heart
2   # Code Angel
3
4   # Classes: Map
5
6   import pygame
7   import screen
8   import csv
9   import random
10  import os
11
12  import player_class
13  import utils
14
15
16  # Map class
17  class Map:
18
19      def __init__(self):
20
21          # Load all map images
22          self.water_image = utils.load_media('image', 'water')
23          self.land_image = utils.load_media('image', 'land')
24          self.portal_image = utils.load_media('image', 'portal')
25          self.re_port_image = utils.load_media('image', 're-port')
26          self.beach_image = utils.load_media('image', 'beach')
27          self.gold_image = utils.load_media('image', 'gold')
28          self.trap_image = utils.load_media('image', 'hole')
29          self.heart_image = utils.load_media('image', 'heart')
30          self.spade_image = utils.load_media('image', 'spade')
31
32          self.sword_1_image = utils.load_media('image', 'sword_1')
33          self.sword_2_image = utils.load_media('image', 'sword_2')
34          self.sword_3_image = utils.load_media('image', 'sword_3')
35          self.sword_4_image = utils.load_media('image', 'sword_4')
36
37          self.castle_1_image = utils.load_media('image', 'castle_1')
38          self.castle_2_image = utils.load_media('image', 'castle_2')
39          self.castle_3_image = utils.load_media('image', 'castle_3')
40          self.castle_4_image = utils.load_media('image', 'castle_4')
41          self.castle_5_image = utils.load_media('image', 'castle_5')
    |0  |1  |2  |3  |4  |5  |6  |7  |8
```

```
    |0  |1  |2  |3  |4  |5  |6  |7  |8
42          self.castle_6_image = utils.load_media('image', 'castle_6')
43
44          # Load all map audio
45          self.gold_sound = utils.load_media('audio', 'gold')
46          self.extra_life_sound = utils.load_media('audio', 'extra_life')
47          self.spade_sound = utils.load_media('audio', 'spade')
48          self.sword_sound = utils.load_media('audio', 'sword')
49          self.portal_sound = utils.load_media('audio', 'portal')
50          self.water_sound = utils.load_media('audio', 'water')
51
52          self.map_key = {
53              'w': 'water',
54              'l': 'land',
55              'p': 'portal',
56              'r': 're-port',
57              'b': 'beach',
58              'g': 'gold',
59              't': 'trap',
60              'h': 'heart',
61              'd': 'spade',
62              '1': 'sword 1',
63              '2': 'sword 2',
64              '3': 'sword 3',
65              '4': 'sword 4',
66              'c1': 'castle 1',
67              'c2': 'castle 2',
68              'c3': 'castle 3',
69              'c4': 'castle 4',
70              'c5': 'castle 5',
71              'c6': 'castle 6'
72          }
73
74          # Total number of rows and columns in the map
75          self.screen_cols = int(screen.SCREENWIDTH / screen.TILE_SIZE)
76          self.screen_rows = int(screen.SCREENHEIGHT / screen.TILE_SIZE)
77
78          self.map_cols = None
79          self.map_rows = None
80
81          self.re_port_points = []
82
    |0  |1  |2  |3  |4  |5  |6  |7  |8
```

```
    |0  |1  |2  |3  |4  |5  |6  |7  |8
83          self.map_tile_x = 0
84          self.map_tile_y = 0
85          self.map_tile_step_x = 0
86          self.map_tile_step_y = 0
87
88          self.player_row = int(screen.SCREENHEIGHT / screen.TILE_SIZE / 2 + 1)
89          self.player_col_left = int(screen.SCREENWIDTH / screen.TILE_SIZE / 2)
90          self.player_col_right = int(screen.SCREENWIDTH / screen.TILE_SIZE / 2 + 1)
91
92          self.dx = 0
93          self.dy = 0
94
95          self.portal = False
96          self.d_portal_x = 0
97          self.d_portal_y = 0
98
99          self.level = 0
100         self.tile_list = None
101
102     # Load the CSV map
103     def load_map(self):
104
105         map_file = None
106
107         if self.level == 1:
108             map_file = 'snakeheart map level 1'
109         elif self.level == 2:
110             map_file = 'snakeheart map level 2'
111
112         full_path = os.path.dirname(os.path.realpath(__file__))
113         maps_path = os.path.join(full_path, 'maps')
114         full_filename = os.path.join(maps_path, map_file + '.csv')
115
116         # Open and read the map CSV file and store in 2D list tile list
117         with open(full_filename, 'r') as csvfile:
118             read_map = csv.reader(csvfile)
119             self.tile_list = list(read_map)
120
121         # Calculate total map rows and columns
122         self.map_cols = len(self.tile_list[0])
123         self.map_rows = len(self.tile_list)
    |0  |1  |2  |3  |4  |5  |6  |7  |8
```



```
    |0  |1  |2  |3  |4  |5  |6  |7  |8
124
125         # Find all map re-port points
126         self.find_re_port_points()
127
128     # Scroll map as player moves
129     def scroll(self, direction):
130
131         self.dx = 0
132         self.dy = 0
133
134         if direction == 'right':
135             if self.map_tile_x > 0:
136                 self.dx = player_class.PLAYER_MOVE
137                 self.dy = 0
138
139         elif direction == 'left':
140             if self.map_tile_x < self.map_cols - self.screen_cols - 1:
141                 self.dx = -player class.PLAYER MOVE
142                 self.dy = 0
143
144         elif direction == 'down':
145             if self.map_tile_y > 0:
146                 self.dx = 0
147                 self.dy = player_class.PLAYER_MOVE
148
149         elif direction == 'up':
150             if self.map_tile_y < self.map_rows - self.screen_rows - 1:
151                 self.dx = 0
152                 self.dy = -player_class.PLAYER_MOVE
153
154         # Work out if a complete tile has been scrolled, and if so update map tile x / map tile y
155         self.map_tile_step_x += self.dx
156         self.map_tile_step_y += self.dy
157
158         if self.map_tile_step_x >= screen.TILE_SIZE:
159             self.map_tile_step_x -= screen.TILE_SIZE
160             self.map_tile_x -= 1
161
162         if self.map_tile_step_x < 0:
163             self.map_tile_step_x += screen.TILE_SIZE
164             self.map_tile_x += 1
    |0  |1  |2  |3  |4  |5  |6  |7  |8
```



```
    |0  |1  |2  |3  |4  |5  |6  |7  |8
165
166         if self.map_tile_step_y >= screen.TILE_SIZE:
167             self.map_tile_step_y -= screen.TILE_SIZE
168             self.map_tile_y -= 1
169
170         if self.map_tile_step_y < 0:
171             self.map_tile_step_y += screen.TILE_SIZE
172             self.map_tile_y += 1
173
174
175     # Draw the map items
176     def draw(self, display):
177         for row in range(self.screen_rows + 1):
178             for col in range(self.screen_cols + 1):
179
180                 lucy_row = row + self.map_tile_y
181                 lucy_col = col + self.map_tile_x
182                 tile key = self.tile list[lucy row][lucy col]
183
184                 tile = self.map_key.get(tile_key)
185
186                 display_image = None
187
188                 if tile == 'water':
189                     display_image = self.water_image
190                 elif tile == 'land':
191                     display_image = self.land_image
192                 elif tile == 'portal':
193                     display_image = self.portal_image
194                 elif tile == 're-port':
195                     display_image = self.re_port_image
196                 elif tile == 'gold':
197                     display_image = self.gold_image
198                 elif tile == 'beach':
199                     display_image = self.beach_image
200                 elif tile == 'trap':
201                     display_image = self.trap_image
202                 elif tile == 'heart':
203                     display_image = self.heart_image
204                 elif tile == 'spade':
205                     display_image = self.spade_image
    |0  |1  |2  |3  |4  |5  |6  |7  |8
```

```
    |0  |1  |2  |3  |4  |5  |6  |7  |8
206
207                 elif tile == 'sword 1':
208                     display_image = self.sword_1_image
209                 elif tile == 'sword 2':
210                     display_image = self.sword_2_image
211                 elif tile == 'sword 3':
212                     display_image = self.sword_3_image
213                 elif tile == 'sword 4':
214                     display_image = self.sword_4_image
215
216                 elif tile == 'castle 1':
217                     display_image = self.castle_1_image
218                 elif tile == 'castle 2':
219                     display_image = self.castle_2_image
220                 elif tile == 'castle 3':
221                     display_image = self.castle_3_image
222                 elif tile == 'castle 4':
223                     display image = self.castle 4 image
224                 elif tile == 'castle 5':
225                     display_image = self.castle_5_image
226                 elif tile == 'castle 6':
227                     display_image = self.castle_6_image
228
229                 x_pos = (col - 1) * screen.TILE_SIZE + self.map_tile_step_x
230                 y_pos = (row - 1) * screen.TILE_SIZE + self.map_tile_step_y
231                 display.show_image(display_image, x_pos, y_pos)
232
233     def reset_change(self):
234         self.dx = 0
235         self.dy = 0
236
237     # Test if Lucy has come into contact with any map items
238     def check_player_loc(self, player):
239         player_x = self.player_row + self.map_tile_y + 1
240         player_y_left = self.player_col_left + self.map_tile_x
241         player_y_right = self.player_col_right + self.map_tile_x
242
243         tile_left = self.map_key.get(self.tile_list[player_x][player_y_left])
244         tile_right = self.map_key.get(self.tile_list[player_x][player_y_right])
245
246         touching_water = self.player_touching('water', tile_left, tile_right)
    |0  |1  |2  |3  |4  |5  |6  |7  |8
```

```
    |0  |1  |2  |3  |4  |5  |6  |7  |8
            touching_gold = self.player_touching('gold', tile_left, tile_right)
            touching_portal = self.player_touching('portal', tile_left, tile_right)
            touching_heart = self.player_touching('heart', tile_left, tile_right)
            touching_spade = self.player_touching('spade', tile_left, tile_right)

            touching_sword_1 = self.player_touching('sword 1', tile_left, tile_right)
            touching_sword_2 = self.player_touching('sword 2', tile_left, tile_right)
            touching_sword_3 = self.player_touching('sword 3', tile_left, tile_right)
            touching_sword_4 = self.player_touching('sword 4', tile_left, tile_right)

            touching_castle_1 = self.player_touching('castle 1', tile_left, tile_right)
            touching_castle_2 = self.player_touching('castle 2', tile_left, tile_right)
            touching_castle_3 = self.player_touching('castle 3', tile_left, tile_right)
            touching_castle_4 = self.player_touching('castle 4', tile_left, tile_right)
            touching_castle_5 = self.player_touching('castle 5', tile_left, tile_right)
            touching_castle_6 = self.player_touching('castle 6', tile_left, tile_right)

            if touching_water is True and player.alive is True:
                player.map_water()
                self.water_sound.play()

            elif touching_gold is True:
                player.map_gold()
                self.remove_item('gold', tile_left, tile_right)
                self.gold_sound.play()

            elif touching_heart is True:
                if player.lives < player.max_lives:
                    player.map_heart()
                    self.remove_item('heart', tile_left, tile_right)
                    self.extra_life_sound.play()

            elif touching_spade is True:
                if player.spades < player.max_spades:
                    player.map_spade()
                    self.remove_item('spade', tile_left, tile_right)
                    self.spade_sound.play()

            elif touching_portal is True:
                self.portal_move()
                self.portal_sound.play()
    |0  |1  |2  |3  |4  |5  |6  |7  |8
```

```
    |0  |1  |2  |3  |4  |5  |6  |7  |8
288
289         elif touching_sword_1 is True:
290             player.map_sword(1)
291             self.remove_item('sword 1', tile_left, tile_right)
292             self.sword_sound.play()
293
294         elif touching_sword_2 is True:
295             player.map_sword(2)
296             self.remove_item('sword 2', tile_left, tile_right)
297             self.sword_sound.play()
298
299         elif touching_sword_3 is True:
300             player.map_sword(3)
301             self.remove_item('sword 3', tile_left, tile_right)
302             self.sword_sound.play()
303
304         elif touching_sword_4 is True:
305             player.map_sword(4)
306             self.remove_item('sword 4', tile_left, tile_right)
307             self.sword_sound.play()
308
309         elif touching_castle_1 is True:
310             player.map_castle()
311         elif touching_castle_2 is True:
312             player.map_castle()
313         elif touching_castle_3 is True:
314             player.map_castle()
315         elif touching_castle_4 is True:
316             player.map_castle()
317         elif touching_castle_5 is True:
318             player.map_castle()
319         elif touching_castle_6 is True:
320             player.map_castle()
321
322     # Lucy has dug a hole / trap so add this 2 the map (6 pieces)
323     def add_trap(self):
324         self.add_trap_piece(self.player_row + self.map_tile_y, self.player_col_left + self.map_tile_x)
325         self.add_trap_piece(self.player_row + self.map_tile_y, self.player_col_left + self.map_tile_x + 1)
326         self.add_trap_piece(self.player_row + self.map_tile_y, self.player_col_left + self.map_tile_x + 2)
327         self.add_trap_piece(self.player_row + self.map_tile_y + 1, self.player_col_left + self.map_tile_x)
328         self.add_trap_piece(self.player_row + self.map_tile_y + 1, self.player_col_left + self.map_tile_x + 1)
    |0  |1  |2  |3  |4  |5  |6  |7  |8
```

```
    |0  |1  |2  |3  |4  |5  |6  |7  |8
329         self.add_trap_piece(self.player_row + self.map_tile_y + 1, self.player_col_left + self.map_tile_x + 2)
330
331     # Add a trap piece ('t') only if the space is land ('l')
332     def add_trap_piece(self, row_tile, col_tile):
333         if self.tile_list[row_tile][col_tile] == 'l':
334             self.tile_list[row_tile][col_tile] = 't'
335
336     # Port to a random re-port point
337     def portal_move(self):
338         random_re_port = random.choice(self.re_port_points)
339         new_map_x = random_re_port[1] - 20
340         new_map_y = random_re_port[0] - 15 - 2
341
342         self.d_portal_x = self.map_tile_x - new_map_x
343         self.d_portal_y = self.map_tile_y - new_map_y
344         self.map_tile_x = new_map_x
345         self.map_tile_y = new_map_y
346         self.map_tile_step_x = 0
347         self.map_tile_step_y = 0
348
349         self.portal = True
350
351     # Find all the re-port points in the map and add to the list re_port_points
352     def find_re_port_points(self):
353         for row in range(self.map_rows):
354             for col in range(self.map_cols):
355                 tile = self.map_key.get(self.tile_list[row][col])
356
357                 if tile == 're-port':
358                     self.re_port_points.append([row, col])
359
360     # Is the player touching an item
361     def player_touching(self, item, tile_left, tile_right):
362         player_touching_item = False
363         if (tile_left == item and self.map_tile_step_x > player_class.PLAYER_MOVE) or tile_right == item:
364             player_touching_item = True
365
366         return player_touching_item
367
368     # Remove an item by changing its map value to land ('l')
369     def remove_item(self, item, tile_left, tile_right):
    |0  |1  |2  |3  |4  |5  |6  |7  |8
```



```
    |0  |1  |2  |3  |4  |5  |6  |7  |8
370             if tile_left == item and self.map_tile_step_x > player_class.PLAYER_MOVE:
371                 self.tile_list[self.player_row + self.map_tile_y + 1][self.player_col_left + self.map_tile_x] = 'l'
372             elif tile_right == item:
373                 self.tile_list[self.player_row + self.map_tile_y + 1][self.player_col_right + self.map_tile_x] = 'l'
374
```

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