

The starter code

Your printed copy of `rle_starter.py`. Type it or open the file.

Name: _____

Date: _____

Set up: keep `rle_starter.py` and `pictures.py` in the *same folder*. Open `rle_starter.py` in Spyder and press **F5**. Pictures appear in the Plots pane.

Part 1 Given: do not change

```
5 import matplotlib.pyplot as plt
6 from pictures import MUG, DINOSAUR, MARIO, APPLE, LOCK, GHOST, UK_IRELAND
7 # ^ that line fetches the clues from the file pictures.py (keep it in the same folder)
8
9
10 # ----- GIVEN: you do not need to change this function -----
11 def show_picture(picture):
12     # picture = a list of rows. Each row is a list of 0s (blank) and 1s (coloured).
13     plt.imshow(picture, cmap="Greys", interpolation="nearest")
14     plt.axis("off")
15     plt.show()
16
17
18 # ----- TEST: a tiny picture made by hand -----
19 heart = [
20     [0, 1, 0, 1, 0],
21     [1, 1, 1, 1, 1],
22     [0, 1, 1, 1, 0],
23     [0, 0, 1, 0, 0],
24 ]
25 show_picture(heart)
```

Part 2 Your turn: fill in the TODOs

```
28 # ----- TODO 1: decode ONE row -----
29 def decode_row(clue):
30     row = []          # the row we are building. It starts empty.
31     colour = 0        # we always start with blanks (0)
32
33     # TODO: go through each number in the clue (a for loop over the list).
34     # 1. add that many copies of colour to the row
35     # hint: a loop inside a loop: for i in range(number), then row.append(colour)
36     # 2. then swap the colour: 0 becomes 1, and 1 becomes 0
37     # hint: an if / else, or the shortcut colour = 1 - colour
38
39     return row
40
41
42 # Test it. The answer should be [0, 0, 1, 1, 1, 1, 1, 1, 1, 0, 0, 0]
43 print(decode_row([2, 7, 3]))
```

```

46 # ----- TODO 2: decode a WHOLE picture -----
47 def decode_picture(clues):
48     picture = []    # the picture we are building. It starts empty.
49
50     # TODO: go through each clue in clues.
51     #     1. decode that clue into a row (you already wrote a function for this!)
52     #     2. add the row to the picture    hint: picture.append(...)
53
54     return picture
55
56
57 # Test it on the mug, then look at the Plots pane
58 # show_picture(decode_picture(MUG))
59
60
61 # ----- TODO 3: try the bigger ones -----
62 # show_picture(decode_picture(DINOSAUR))
63 # show_picture(decode_picture(MARIO))
64 # show_picture(decode_picture(UK_IRELAND))    # the one that took ages by hand!

```

Part 3 Stretch: go backwards

```

67 # ----- STRETCH: go backwards (ENCODE) -----
68 def encode_row(row):
69     clue = []
70     # TODO: count how many pixels in a row are the same, and add each count to clue.
71     # hints: start by counting blanks (0). Keep a count variable.
72     #     if the pixel matches the colour we are counting, count += 1
73     #     otherwise, add count to clue, swap the colour, and start counting again.
74     #     Don't forget to add the last count when the loop ends!
75     return clue
76
77 # print(encode_row([0, 0, 1, 1, 1, 1, 1, 1, 1, 0, 0, 0]))    # should print [2, 7, 3]

```

Tick when done: ☐ heart shows ☐ decode_row works ☐ mug shows ☐ UK & Ireland shows
☐ encode_row