

Answer key: coding sheets

INSTRUCTOR COPY

Sheet 05 Trace tables

`decode_row([2, 7, 3])`

step	no.	col.	added	row so far
1	2	0	0, 0	[0, 0]
2	7	1	1 × 7	[0, 0, 1, 1, 1, 1, 1, 1, 1]
3	3	0	0 × 3	[0, 0, 1, 1, 1, 1, 1, 1, 1, 1, 0, 0, 0]

`decode_row([0, 4, 2])`

step	no.	col.	added	row so far
1	0	0	nothing	[]
2	4	1	1 × 4	[1, 1, 1, 1]
3	2	0	0 × 2	[1, 1, 1, 1, 0, 0]

Different about the first step: `range(0)` runs zero times, so nothing is added, but the colour still swaps. The row starts with 1.

`colour = 1 - colour` : 0 becomes 1 ($1 - 0$) and 1 becomes 0 ($1 - 1$). It flips the colour.

Sheet 06 Model answers

`decode_row`

1. Start with an empty row and colour 0.
2. For each number in the clue:
3. add that many copies of colour to the row.
4. Swap colour (0 to 1, 1 to 0).
5. Return the row.

`decode_picture`

1. Start with an empty picture.
2. For each clue: decode it into a row.
3. Add the row to the picture.
4. Return the picture.

Boxes and arrows: [for each clue] → box 1 *take the next clue from clues* → [decode it] → box 2 *row = decode_row(clue)* → [add to picture] → box 3 *picture.append(row)*, then back to the next clue.

Sheet 07 Completed code (same as rle_solution.py)

```
def decode_row(clue):
    row = []
    colour = 0
    for number in clue:
        for i in range(number):
            row.append(colour)
        if colour == 0:
            colour = 1
        else:
            colour = 0
    return row
```

```
def decode_picture(clues):
    picture = []
    for clue in clues:
        row = decode_row(clue)
        picture.append(row)
    return picture
```

```
def encode_row(row):
    clue = []
    colour = 0
    count = 0
    for pixel in row:
        if pixel == colour:
            count += 1
        else:
            clue.append(count)
            colour = pixel
            count = 1
    clue.append(count)
    return clue
```

Test output: `[0, 0, 1, 1, 1, 1, 1, 1, 1, 0, 0, 0]` . Stretch: `encode_row` of that row gives `[2, 7, 3]` . The swap can also be written `colour = 1 - colour` .

Sheet 07 Debugging checklist answers

- **return:** it goes at the very end, indented once (inside the function, outside the loops). Without it the function gives back `None` and the picture is empty.
- **Indents:** `row.append(colour)` is inside the inner loop (2 levels). The swap is inside the outer loop but level with the inner `for` .
- **Swap:** INSIDE the outer loop, AFTER the inner loop.
- **Run:** press the green triangle (or F5) again after every change.

Notes Common mistakes

- **Swap inside the inner loop:** the colour flips every pixel, giving stripes like `[0, 1, 0, 1, ...]` . Fix: dedent the swap.
- **Missing return:** `print` shows `None` , and `show_picture` errors.
- **Indentation:** `append` outside the inner loop adds only one pixel per number.
- **Colour not reset per row:** not a problem here. `colour = 0` is set inside `decode_row` , so every row starts blank. It would be a bug if `colour` were set outside the function.
- **Other:** `row.append` without brackets, or `range(clue)` instead of `range(number)` .
- **Results table:** times vary. Expect the computer to take under a second for every picture, and UK & Ireland to take the class many minutes by hand.