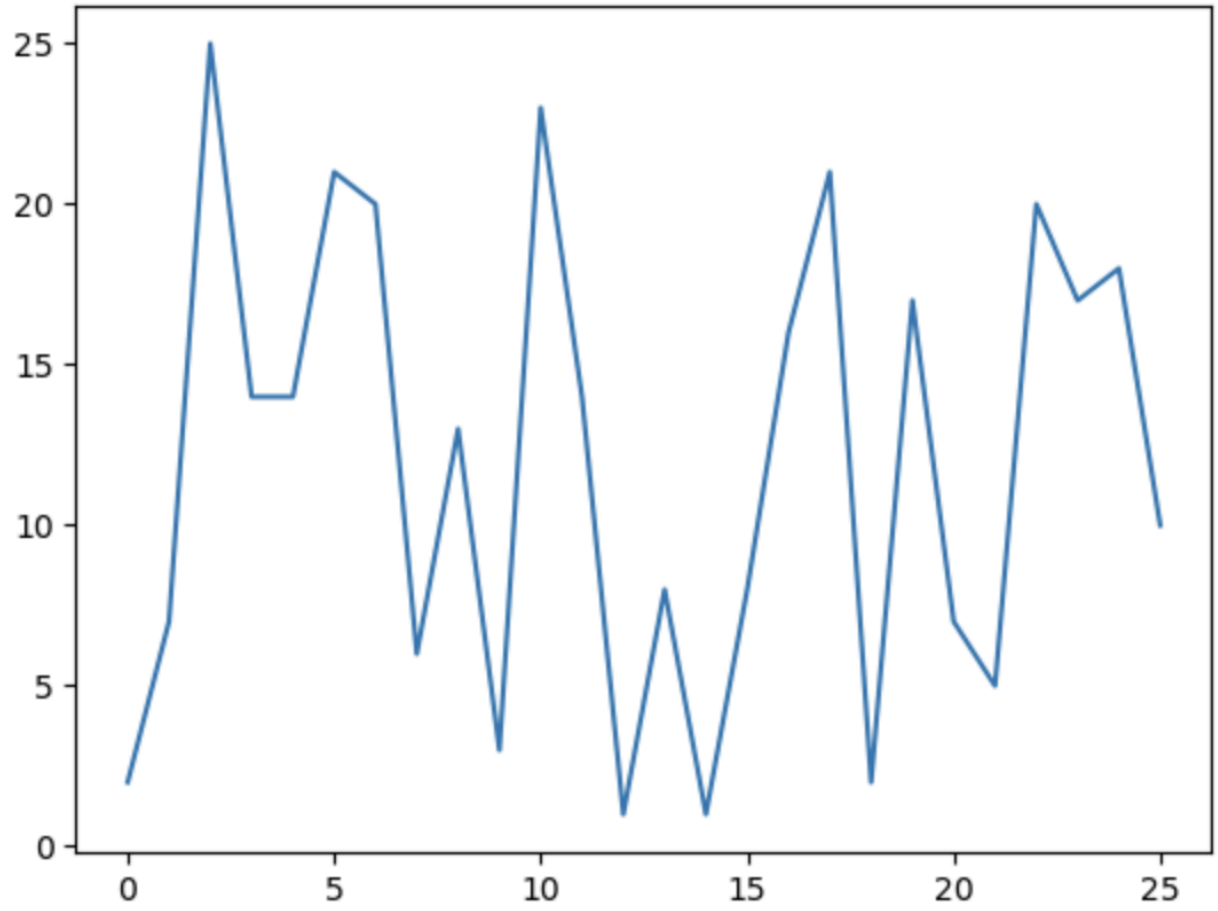
Data Science Camp 2026
Test Tasks

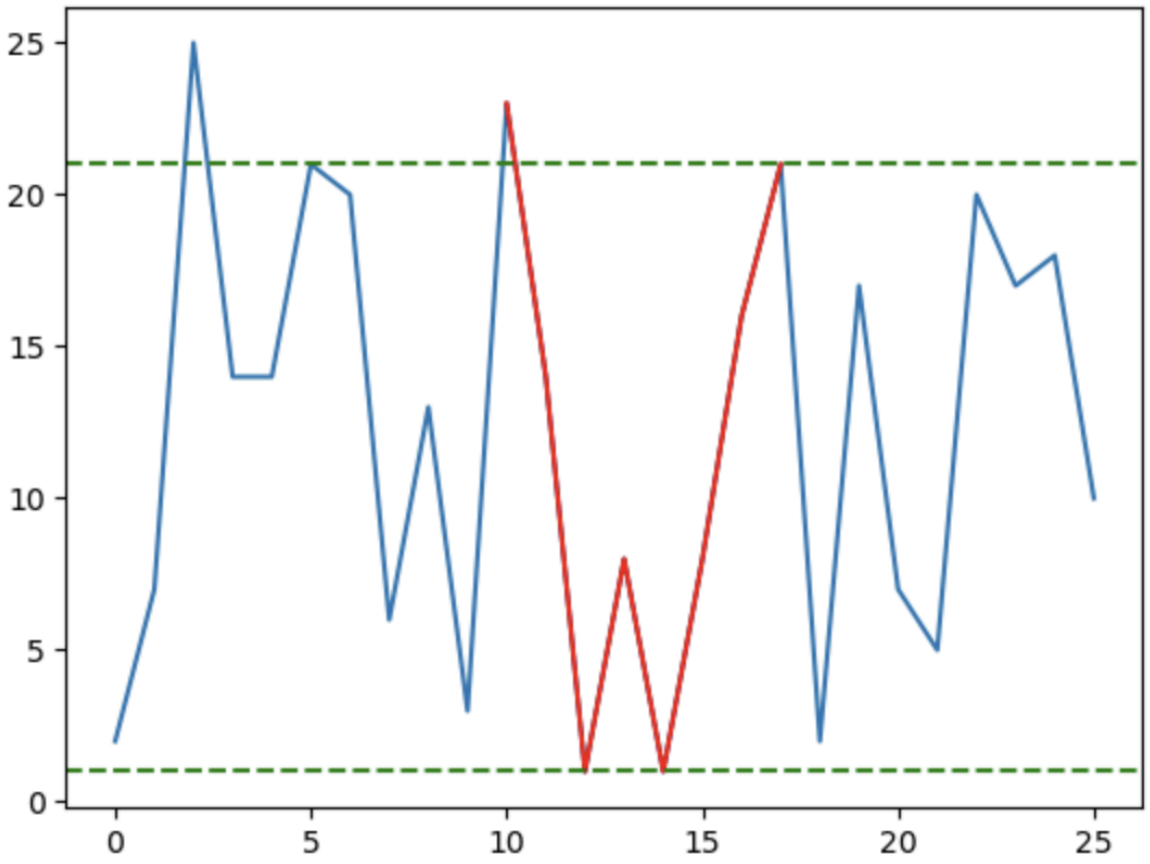
Task 1

Generate the sequence of 26 random integer values from 0 to 26.
Build the plot that joins each neighbour's values as shown on the image for sample ***[2, 7, 25, 14, 14, 21, 20, 6, 13, 3, 23, 14, 1, 8, 1, 8, 16, 21, 2, 17, 7, 5, 20, 17, 18, 10]***



Consider the values as the heights of 2d mountains on which the rain falls from above. Those subsequences that have larger values on boundaries form the lakes.

1) Develop the function that accepts the list of provided 26 integer values and calculates the depth of the deepest lake. Considering the above example, the response should be ***20*** which is the depth of the lake formed by subsequence ***[23, 14, 1, 8, 1, 8, 16, 21]***
2) Visualize the values as heights and highlight the deepest lake.



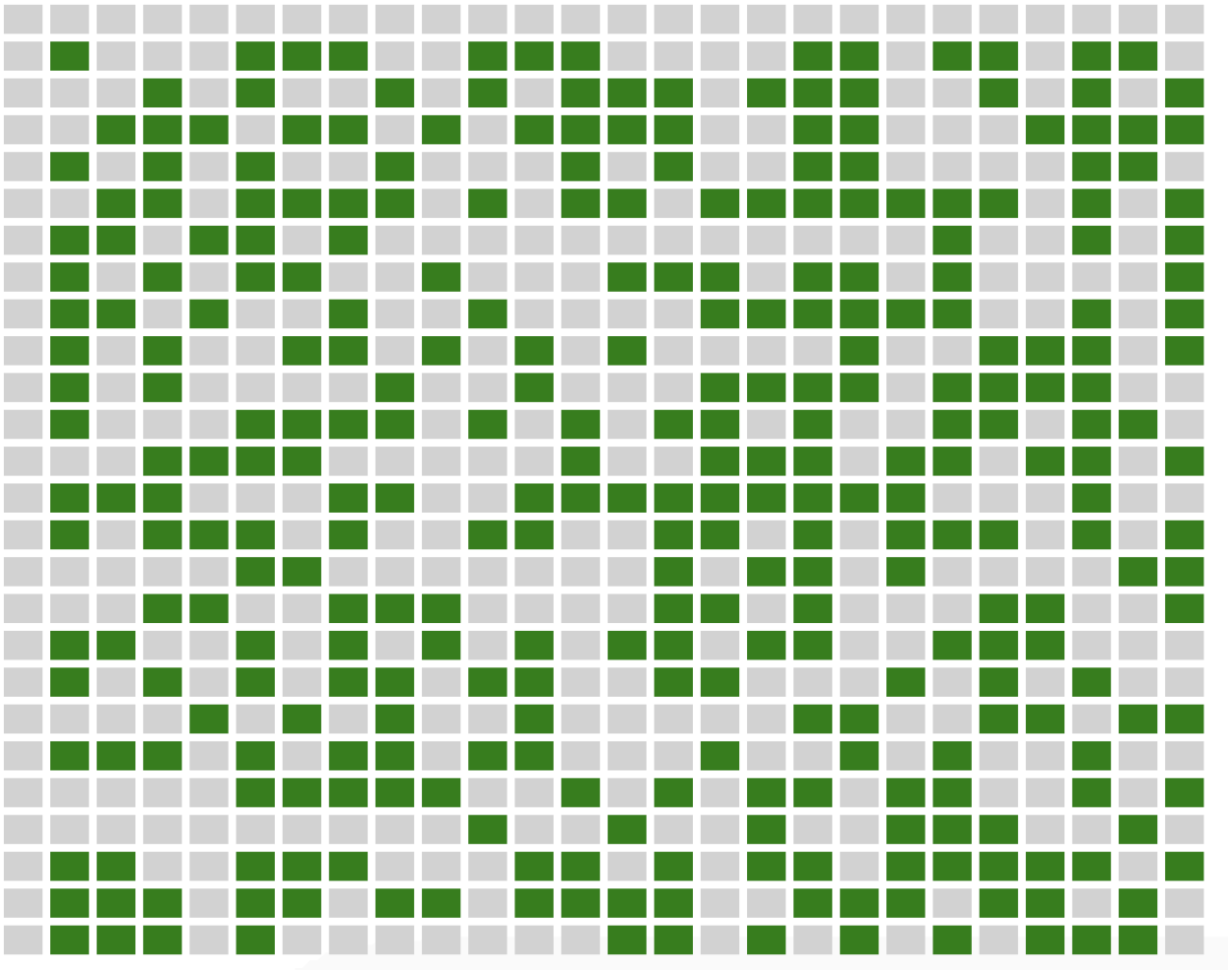
Response format:

Programming code in jupyter notebook/colab/etc. containing the following:

1) Generating list of 26 random values,
2) Function that calculates the depth of deepest lake
3) Result of calculation for provided sample of 26 integers
4) Visualization

Task 2

Consider the matrix of size 26 x 26 filled randomly with binary values ***0*** (cell is “dead”) or ***1*** (cell is “alive”)***.***
The following image represents the sample where green cells are 1 and grey ones are 0:

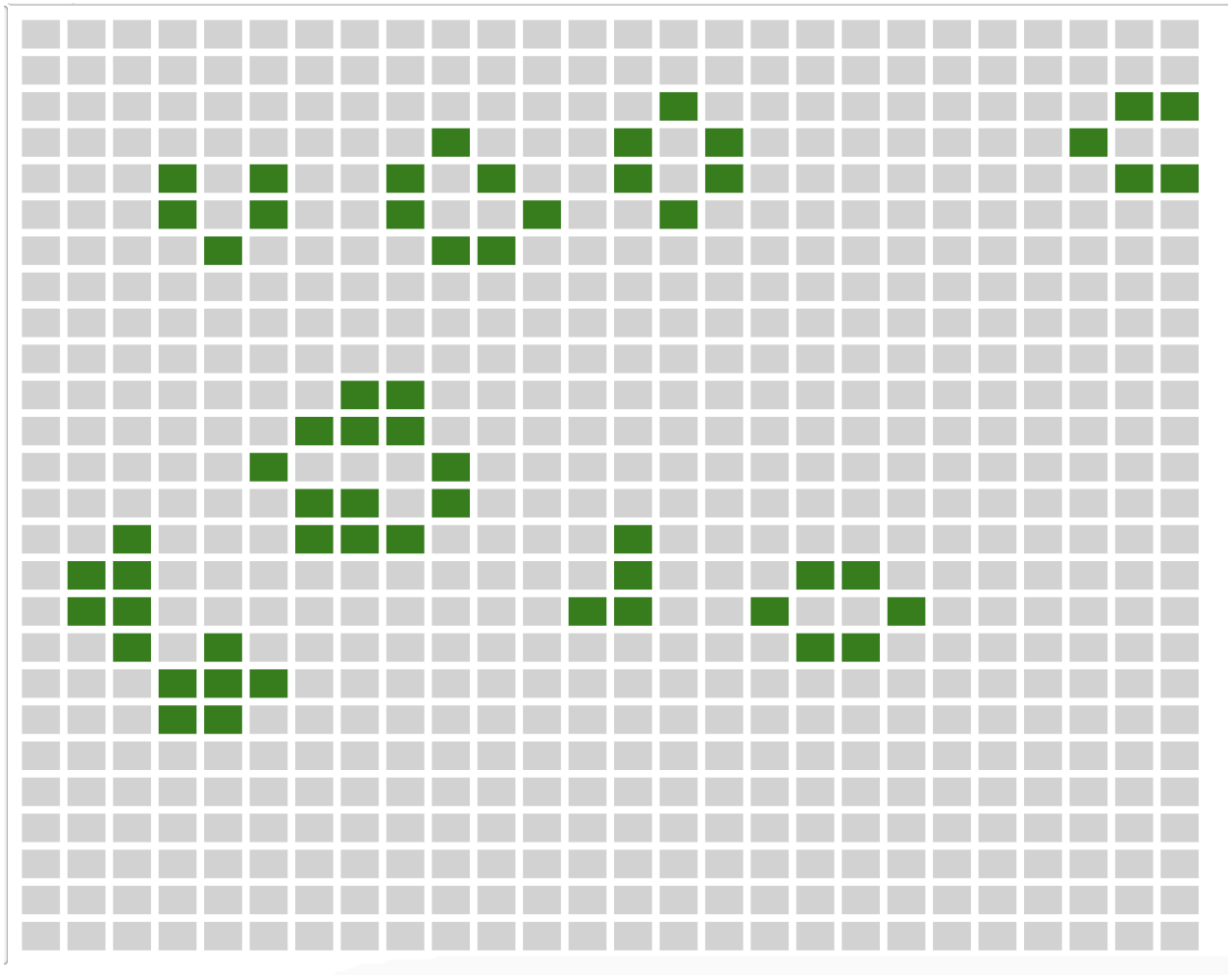


1) Develop the algorithm that accepts the specified binary array of size 26x26 and iterates the steps where each step executes the following rules:
    - If a living cell has two or three living neighbors, it remains alive;
    - if a living cell has one or no living neighbors, it dies of "loneliness";

- if a living cell has four or more living neighbors, it dies from "overpopulation";
- if a dead cell has exactly three living neighbors, it comes to life.

2) (Optionally) visualize the process of execution steps.

The following image represents the result after 20 steps for the mentioned above initialization.



Response format:

Programming code in jupyter notebook/colab/etc. containing the following:

1) Generating the matrix of size 26x26 with binary random values
2) Algorithm that executes iterations over rules
3) Matrix of values after 20 executed steps

5) (Optionally) visualization of results over iteration process

Task 3

The probability of outcome "***H***" (‘Head’) at flipping each of the ***4*** coins (lets call them c1, c2, c3, c4) with a changed center of gravity equal to ***[0,12, 0,27, 0,21, 0,96]*** respectively. One of the coins was chosen at random and the tests began. All further tests are executed for the same coin.

Determine the probability of "***H***" in the next flip after ***9*** of the actual completed tests: ***[H H H T H T H H H]*** (here "***T***" (‘Tail’) is opposite side of the coin).

For example, before the first test, the probability of "***H***" is ***0.39*** (according to the formula of full probability, taking into account the equivalence of the choice of one of the available coins).

Having evidence of "***H***" in the first test, the probability of the hypothesis that the selected coin is c1 / c2 / c3 decreased, and probabilities of hypothesis that c4 increased, and, therefore, the probability to flip "***H***" in the next test now equals to ***0.67***. Similarly, after the release of "***H***" in another test, you need to re-compute the probability of flipping "***H***" in the third flip, and so on.

Recall Bayes rule

$$p(h_1/b) = \frac{p(b/h_1)p(h_1)}{p(b)} = \frac{p(b/h1)p(h_1)}{p(b/h_1)p(h_1) + p(b/h_2)p(h_2)}$$

Response format:

- list of probabilities to the nearest hundredth [0.67,?,? ,?,? ,?,? ,?] (replace ‘?’ with the appropriate values)
- Program code, or description

Submit your work

If you have completed at least one task, or have questions, send the solutions (or a link to git or other storage) or questions to **contact@amazinum.com**

Note: make sure the code is executable, well commented, and the approach is clear to understand.