First, we define the is_unfilled() function to check whether a given pixel in an image is unfilled (with label 0) and returns a True/False statement.

Then we define the fill() function. We first copy the original image to ensure that the image remains unchanged in the test cases. Then, we check the cases where the seed point is invalid (non-integer coordinate(s), negative coordinate(s), outside the image, is a boundary pixel) and return the original image.

For a valid unfilled seed point, we fill it and append it to an empty list. Then we create a while loop to explore four neighbouring points (row +- 1, col +- 1) of the seed point, fill any unfilled points detected by the is.unfill() function and append them to the list. We delete the first element of the list after each loop round to ensure that new neighbouring points are explored. The while loop stops when there are no unfilled neighbouring points, i.e. all points are either filled or boundary points or outside the image. This iterative algorithm ensures that we fill all consecutive unfilled points around the seed point.