Algorithm for Midpoint Circle Drawing and Scaling

Step 1:

Input the radius and the center coordinates of the circle:
- Radius (r)
- Center (xc, yc)

Step 2:

Initialize variables:
- x = 0
- y = r
- p = 1 - r (decision parameter)
- xes = [] (list to store x-coordinates of points)
- yes = [] (list to store y-coordinates of points)

Step 3:

Calculate the initial symmetric points of the circle using the function points_plot:
- This function adds all symmetric points of the circle to xes and yes.

Step 4:

Iteratively calculate points for the circle using the midpoint algorithm:
- Increment x in each iteration.
- If p < 0, update p = p + 2 * x + 1.
- Otherwise, decrement y and update p = p + 2 * (x - y) + 1.
- Calculate the symmetric points for each (x, y).

Step 5:

Define a scaling transformation matrix to scale the circle:
- Example: Scale x by 2 and y by 2.
- Scaling matrix:

  [2, 0, 0]
  [0, 2, 0]
  [0, 0, 1]

Step 6:

Define translation matrices to move the circle for center-preserving scaling:
- Translate to the origin using translation_to_origin matrix.

- Translate back to the original center using translation_back matrix.

Step 7:

Compute the composite transformation matrix:
- Composite = Translation Back * Scaling * Translation to Origin.

Apply this transformation to the circle points.

Step 8:

Plot the original and transformed circles on the same graph:
- Use matplotlib to display the circles with different styles and colors.