EXPERIMENT 2

Scan Conversion of Polygon and Area Filling using Flood-Fill Algorithm

import matplotlib.pyplot as plt import numpy as np from collections import deque

---------- Scan Conversion (Scan-Line Fill) ----------

def scanline_fill(vertices): """ Scan-line polygon filling vertices: list of (x,y) polygon coordinates """ points = [] n = len(vertices) ymin = min(y for _, y in vertices) ymax = max(y for _, y in vertices)

```
# Go line by line
for y in range(ymin, ymax + 1):
    intersections = []
    for i in range(n):
        x1, y1 = vertices[i]
        x2, y2 = vertices[(i + 1) % n]

        if y1 == y2:  # Ignore horizontal edges
            continue
        if y < min(y1, y2) or y > max(y1, y2):
            continue

        # Find intersection
        x = int(x1 + (y - y1) * (x2 - x1) / (y2 - y1))
        intersections.append(x)

    intersections.sort()
```

```
    # Fill between pairs of intersections
    for i in range(0, len(intersections), 2):
        if i+1 < len(intersections):
            for x in range(intersections[i],
intersections[i+1] + 1):
                points.append((x, y))
return points
```

---------- **Flood Fill Algorithm** ----------

def flood_fill(image, start, target_color, replacement_color): """ Flood fill using BFS (efficient, avoids recursion stack overflow) image: 2D numpy array start: (x,y) seed point target_color: color to replace replacement_color: new fill color """ max_y, max_x = image.shape x0, y0 = start if image[y0, x0] != target_color: return

```
q = deque([(x0, y0)])
while q:
    x, y = q.popleft()
    if x < 0 or y < 0 or x >= max_x or y >= max_y:
        continue
    if image[y, x] != target_color:
        continue
    image[y, x] = replacement_color
    q.extend([(x+1,y), (x-1,y), (x,y+1), (x,y-1)])
```

---------- **Visualization** ----------

if **name** == "**main**": # Polygon vertices (simple quadrilateral) polygon = [(20, 10), (60, 10), (70, 40), (30, 50)]

```
# 1. Scan-line polygon filling
filled_points = scanline_fill(polygon)
fx, fy = zip(*filled_points)

plt.figure(figsize=(10,5))

# Show polygon + scanline fill
plt.subplot(1,2,1)
vx, vy = zip(*polygon)
plt.fill(vx + (vx[0],), vy + (vy[0],),
edgecolor="black", fill=False)
plt.scatter(fx, fy, s=5, color="blue")
plt.title("Scan Conversion (Scanline Fill)")
plt.gca().set_aspect("equal", adjustable="box")

# 2. Flood Fill Example
grid = np.zeros((60, 80), dtype=int)

# Draw polygon boundary
for x, y in polygon:
    grid[y, x] = 1
for (x, y) in filled_points:
    grid[y, x] = 1

# Apply flood fill from inside point
flood_fill(grid, (40, 20), 0, 2)

plt.subplot(1,2,2)
plt.imshow(grid, cmap="gray_r", origin="lower")
plt.title("Flood-Fill Algorithm")
plt.gca().set_aspect("equal", adjustable="box")
```

```
plt.show()
```