Experiment 4

Polygon Clipping using Sutherland–hodgman Algorithm

**Aim:**

To write a program that clips a polygon to a specified rectangular window using the Sutherland–Hodgman Polygon Clipping Algorithm and displays the clipped polygon.

**Code:**

import matplotlib.pyplot as plt

def intersect(p1, p2, edge, edge\_val):

x1, y1 = p1

x2, y2 = p2

dx = x2 - x1

dy = y2 - y1

if edge in ('left', 'right'):

if dx == 0:

return (edge\_val, y1)

t = (edge\_val - x1) / dx

y = y1 + t \* dy

return (edge\_val, y)

else:

if dy == 0:

return (x1, edge\_val)

t = (edge\_val - y1) / dy

x = x1 + t \* dx

return (x, edge\_val)

def clip\_with\_edge(polygon, edge, edge\_val):

if not polygon:

return []

def inside(p):

x, y = p

if edge == 'left':

return x >= edge\_val

if edge == 'right':

return x <= edge\_val

if edge == 'bottom':

return y >= edge\_val

if edge == 'top':

return y <= edge\_val

raise ValueError("Unknown edge")

output = []

prev = polygon[-1]

for curr in polygon:

prev\_in = inside(prev)

curr\_in = inside(curr)

if prev\_in and curr\_in:

output.append(curr)

elif prev\_in and not curr\_in:

output.append(intersect(prev, curr, edge, edge\_val))

elif not prev\_in and curr\_in:

output.append(intersect(prev, curr, edge, edge\_val))

output.append(curr)

else:

pass

prev = curr

return output

def sutherland\_hodgman(subject\_polygon, clip\_rect):

"""Clip subject\_polygon to clip\_rect using Sutherland–Hodgman.

subject\_polygon: list of (x,y)

clip\_rect: (xmin, ymin, xmax, ymax)

returns clipped polygon (list of (x,y))

"""

xmin, ymin, xmax, ymax = clip\_rect

clipped = subject\_polygon

clipped = clip\_with\_edge(clipped, 'left', xmin)

clipped = clip\_with\_edge(clipped, 'right', xmax)

clipped = clip\_with\_edge(clipped, 'bottom', ymin)

clipped = clip\_with\_edge(clipped, 'top', ymax)

return clipped

def plot\_polygons(subject\_polygon, clipped\_polygon, clip\_rect, ax=None):

if ax is None:

fig, ax = plt.subplots(figsize=(7,7))

# Original polygon

xs = [p[0] for p in subject\_polygon] + [subject\_polygon[0][0]]

ys = [p[1] for p in subject\_polygon] + [subject\_polygon[0][1]]

ax.plot(xs, ys, '-o', label='Original polygon', linewidth=1.5)

#Clipped polygon

if clipped\_polygon:

cx = [p[0] for p in clipped\_polygon] + [clipped\_polygon[0][0]]

cy = [p[1] for p in clipped\_polygon] + [clipped\_polygon[0][1]]

ax.plot(cx, cy, '-o', label='Clipped polygon', linewidth=2.0)

else:

ax.text(0.02, 0.98, "Result: empty (fully clipped)", transform=ax.transAxes,

va='top', color='red')

# Clipping rectangle

xmin, ymin, xmax, ymax = clip\_rect

rect\_x = [xmin, xmax, xmax, xmin, xmin]

rect\_y = [ymin, ymin, ymax, ymax, ymin]

ax.plot(rect\_x, rect\_y, '--', label='Clipping window', linewidth=1.2)

ax.set\_aspect('equal', 'box')

ax.set\_xlabel('X')

ax.set\_ylabel('Y')

ax.set\_title('Sutherland–Hodgman Polygon Clipping')

ax.legend()

ax.grid(True)

plt.show()

if \_\_name\_\_ == "\_\_main\_\_":

subject = [

(1, 2), (4, 6), (8, 5), (7, 2), (5, 1), (3, 0.5)

]

clip\_window = (2.0, 1.0, 6.0, 4.5)

clipped = sutherland\_hodgman(subject, clip\_window)

print("Original polygon vertices:")

for p in subject:

print(f" {p}")

print("\nClipped polygon vertices:")

if clipped:

for p in clipped:

print(f" ({p[0]:.4f}, {p[1]:.4f})")

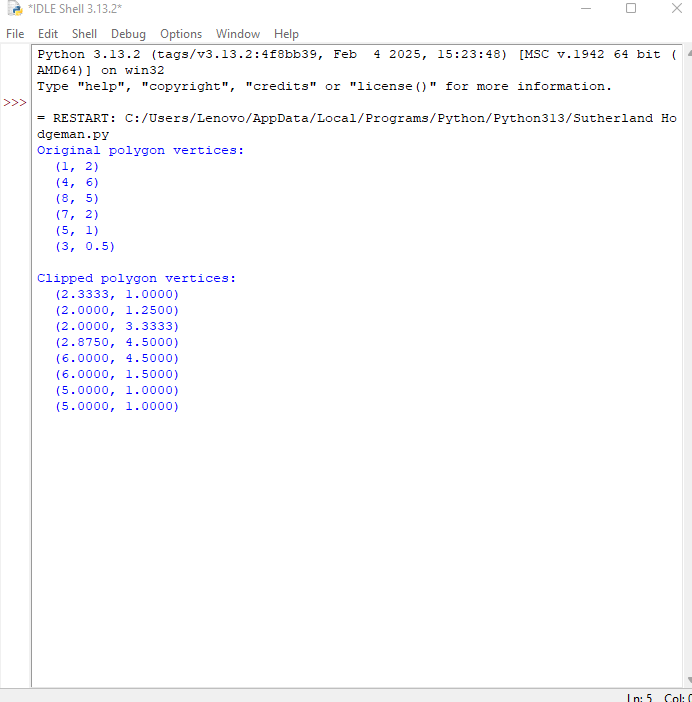
else:

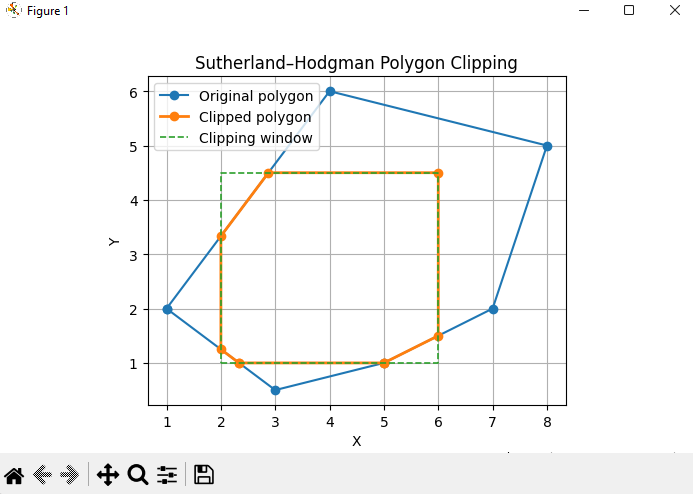
print(" <empty>")

# Plot

plot\_polygons(subject, clipped, clip\_window)

Output:





**Conclusion:**

The polygon was successfully clipped using the Sutherland–Hodgman algorithm against a rectangular clipping window.