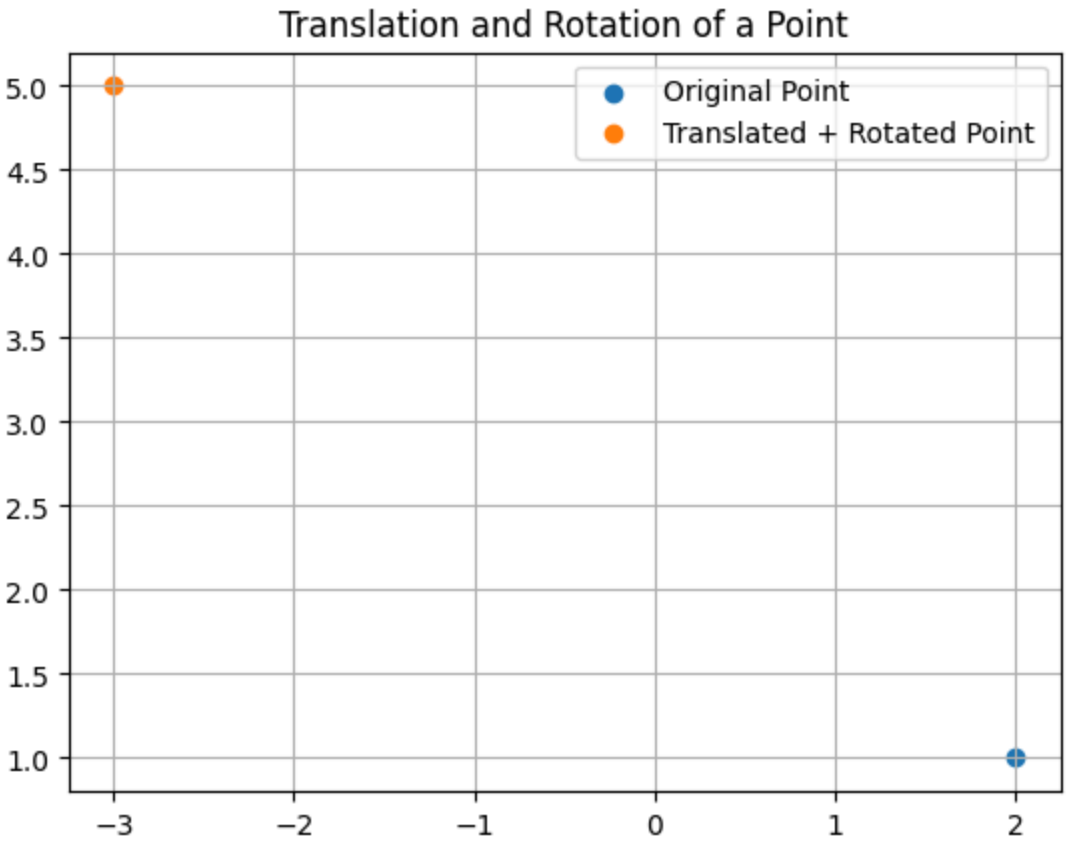
Amit Divekar | Practical 8

Combined Transformations

In [1]:
```
import matplotlib.pyplot as plt
from sympy import Point, Line, Polygon, RegularPolygon, pi, Matrix
```

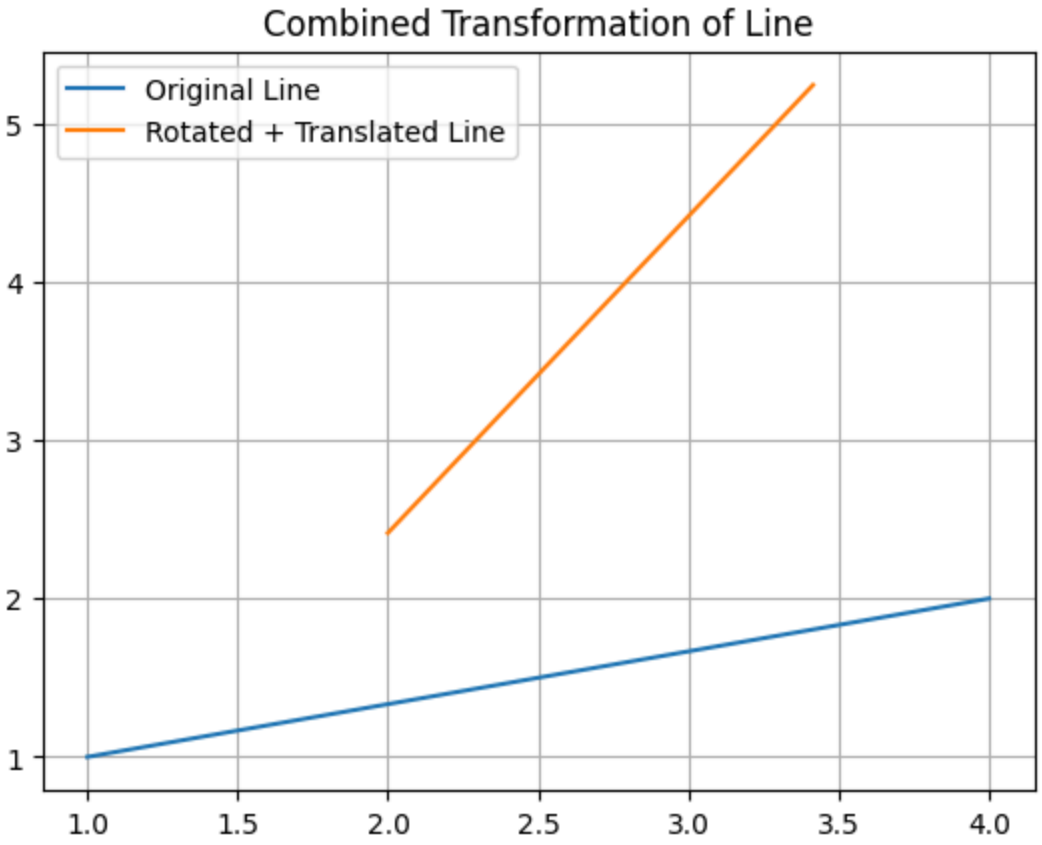
Q1. Apply translation followed by rotation on a point and plot both positions.

In [2]:
```
import matplotlib.pyplot as plt
from sympy import Point, pi
P = Point(2, 1)
P_tr = P.translate(3, 2).rotate(pi/2)
plt.scatter([P.x], [P.y], label="Original Point")
plt.scatter([P_tr.x], [P_tr.y], label="Translated + Rotated Point")
plt.legend()
plt.grid()
plt.title("Translation and Rotation of a Point")
plt.show()
```



Q2. Rotate a line by 45 degrees and then translate it. Plot both the original and transformed lines.

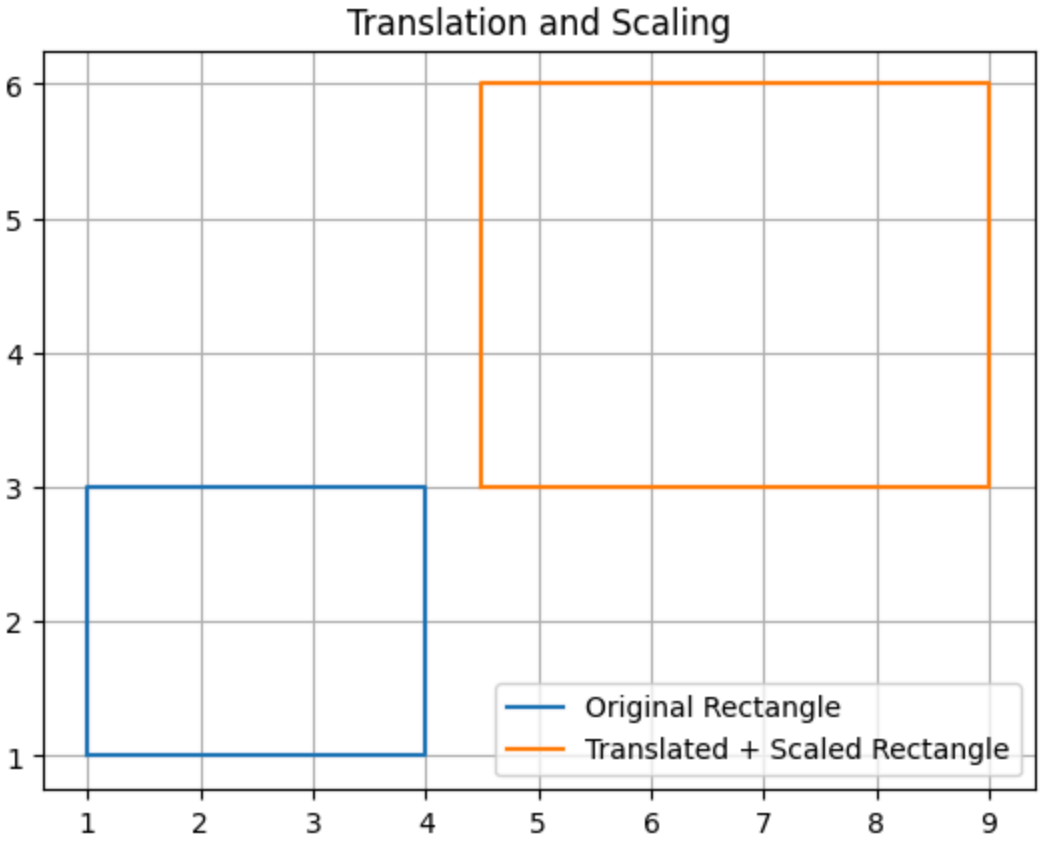
In [3]:
```
import matplotlib.pyplot as plt
from sympy import Line, Point, pi
L = Line(Point(1,1), Point(4,2))
L_rt = L.rotate(pi/4).translate(2, 1)
x1, y1 = [L.p1.x, L.p2.x], [L.p1.y, L.p2.y]
x2, y2 = [L_rt.p1.x, L_rt.p2.x], [L_rt.p1.y, L_rt.p2.y]
plt.plot(x1, y1, label="Original Line")
plt.plot(x2, y2, label="Rotated + Translated Line")
plt.legend()
plt.grid()
plt.title("Combined Transformation of Line")
plt.show()
```



Q3. Translate a rectangle and then apply scaling. Plot the original and transformed rectangles.

In [4]:
```
import matplotlib.pyplot as plt
from sympy import Polygon, Point
R = Polygon(Point(1,1), Point(4,1), Point(4,3), Point(1,3))
R_ts = R.translate(2, 1).scale(1.5, 1.5)
def draw(poly, label):
    x = [p.x for p in poly.vertices] + [poly.vertices[0].x]
```

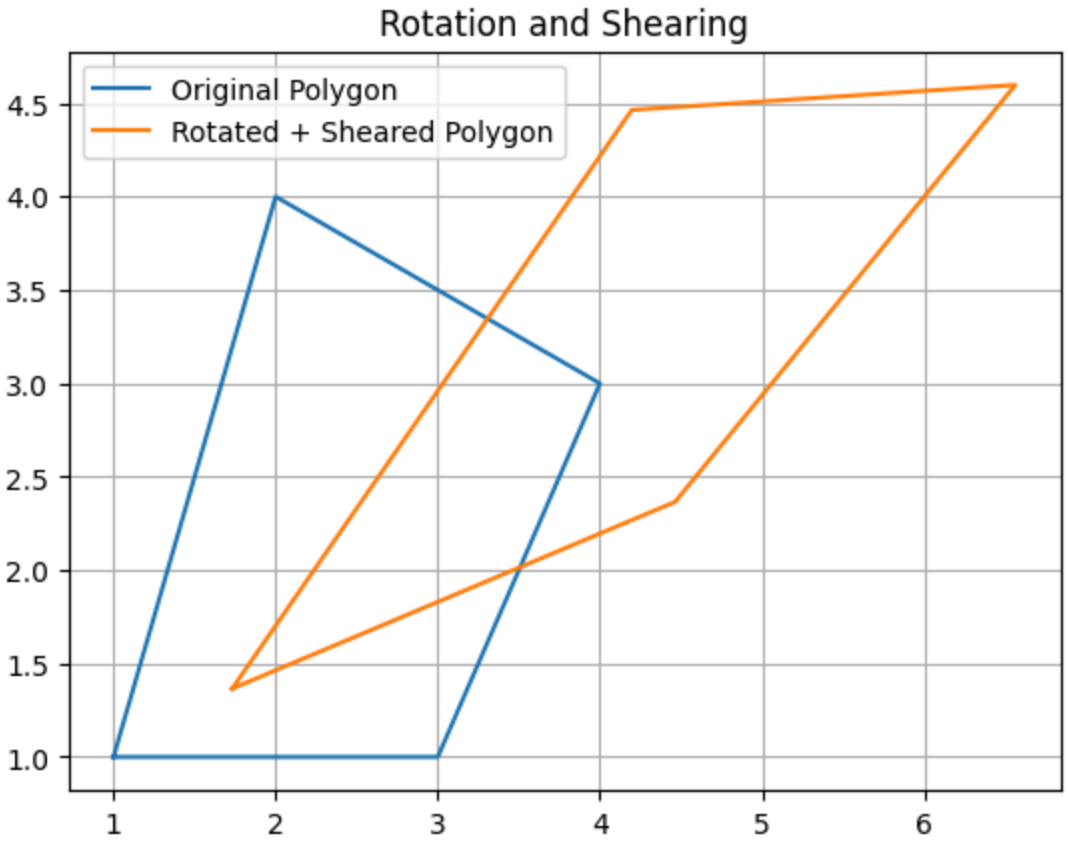
```
    y = [p.y for p in poly.vertices] + [poly.vertices[0].y]
    plt.plot(x, y, label=label)
draw(R, "Original Rectangle")
draw(R_ts, "Translated + Scaled Rectangle")
plt.legend()
plt.grid()
plt.title("Translation and Scaling")
plt.show()
```



Q4. Rotate a polygon and then apply shearing. Plot both figures.

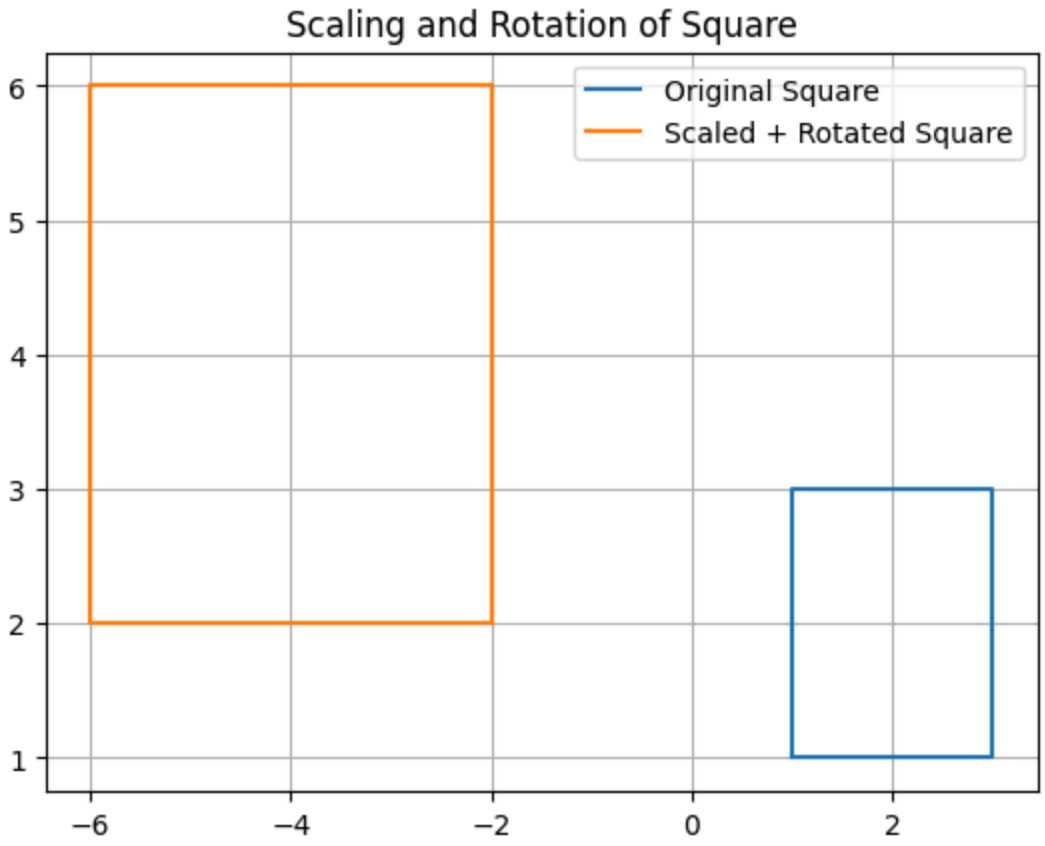
In [5]:
```
import matplotlib.pyplot as plt
from sympy import Polygon, Point, Matrix, pi
P = Polygon(Point(1,1), Point(3,1), Point(4,3), Point(2,4))
P_r = P.rotate(pi/6)
shear = Matrix([[1, 1], [0, 1]])
P_rs = [Point(shear * Matrix(v)) for v in P_r.vertices]
def plot_pts(pts, label):
    x = [p.x for p in pts] + [pts[0].x]
    y = [p.y for p in pts] + [pts[0].y]
    plt.plot(x, y, label=label)
plot_pts(P.vertices, "Original Polygon")
plot_pts(P_rs, "Rotated + Sheared Polygon")
plt.legend()
plt.grid()
```

```
plt.title("Rotation and Shearing")
plt.show()
```



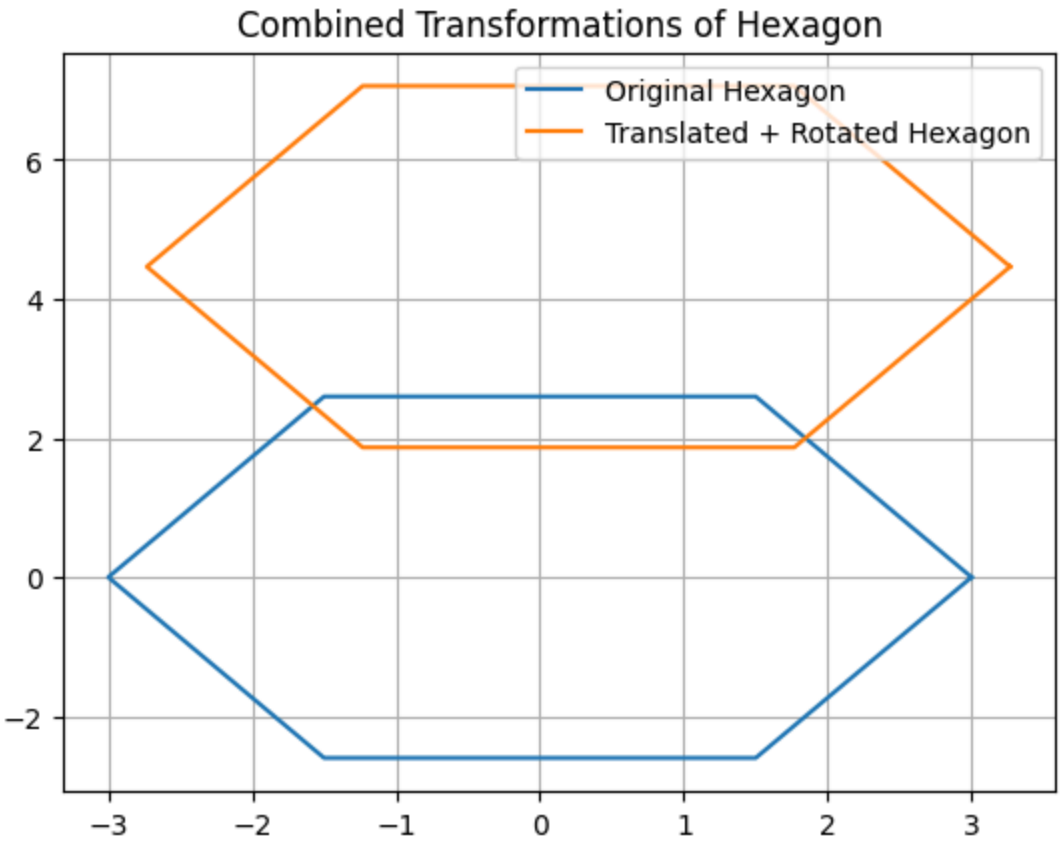
Q5. Scale a square and then rotate it by 90 degrees. Plot both figures.

In [6]:
```
import matplotlib.pyplot as plt
from sympy import Polygon, Point, pi
S = Polygon(Point(1,1), Point(3,1), Point(3,3), Point(1,3))
S_sr = S.scale(2, 2).rotate(pi/2)
def draw_sq(P, label):
    x = [p.x for p in P.vertices] + [P.vertices[0].x]
    y = [p.y for p in P.vertices] + [P.vertices[0].y]
    plt.plot(x, y, label=label)
draw_sq(S, "Original Square")
draw_sq(S_sr, "Scaled + Rotated Square")
plt.legend()
plt.grid()
plt.title("Scaling and Rotation of Square")
plt.show()
```



Q6. Translate a regular hexagon and then rotate it. Plot both figures.

In [7]:
```
import matplotlib.pyplot as plt
from sympy import RegularPolygon, Point, pi
H = RegularPolygon(Point(0,0), 3, 6)
H_tr = H.translate(4, 2).rotate(pi/3)
def plot_hex(hexagon, label):
    x = [v.x for v in hexagon.vertices] + [hexagon.vertices[0].x]
    y = [v.y for v in hexagon.vertices] + [hexagon.vertices[0].y]
    plt.plot(x, y, label=label)
plot_hex(H, "Original Hexagon")
plot_hex(H_tr, "Translated + Rotated Hexagon")
plt.legend()
plt.grid()
plt.title("Combined Transformations of Hexagon")
plt.show()
```


Combined Transformations of Hexagon
Original Hexagon
Translated + Rotated Hexagon
6
4
2
0
−2
−3
−2
−1
0
1
2
3