79.Convex-Hull Problems

Program:

```
import math

def orientation(p, q, r):
    val = (q[1] - p[1]) * (r[0] - q[0]) - (q[0] - p[0]) * (r[1] - q[1])
    if val == 0:
        return 0
    elif val > 0:
        return 1
    else:
        return -1

def distance(p1, p2):
    return (p1[0] - p2[0]) ** 2 + (p1[1] - p2[1]) ** 2

def graham_scan(points):
    # Find the point with the lowest y-coordinate, break ties by x-coordinate
    start = min(points, key=lambda p: (p[1], p[0]))

    # Sort points by polar angle with the start point
    points = sorted(points, key=lambda p: (math.atan2(p[1] - start[1], p[0] - start[0]), distance(start, p)))

    # Initialize the hull with the first two points
    hull = [points[0], points[1]]

    # Process the remaining points
    for p in points[2:]:
        while len(hull) > 1 and orientation(hull[-2], hull[-1], p) != -1:
            hull.pop()
        hull.append(p)

    return hull

# Example usage:
points = [(0, 3), (2, 2), (1, 1), (2, 1), (3, 0), (0, 0), (3, 3)]
hull = graham_scan(points)
print("Convex Hull:", hull)
```

Output:

```
"C:\Program Files\Python312\python.exe" "C:\Work Space\DAA\DAA COADS.PYTHON\program 79.py"
Convex Hull: [(0, 0), (3, 0), (3, 3), (0, 3)]

Process finished with exit code 0
```

Time complexity:

O(n log n)