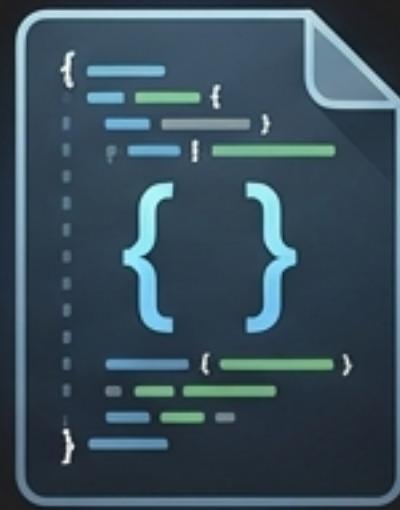


# Algorithmic Art with Python Turtle: จอดรถหัสดรร堪ะสู่ภาพกราฟิก

การสร้างกระบวนการ Computational Thinking  
ผ่านการเขียนโปรแกรมเชิงโครงสร้าง

case\_study.py

# วัตถุประสงค์: มากว่าแค่การเขียนโค้ด

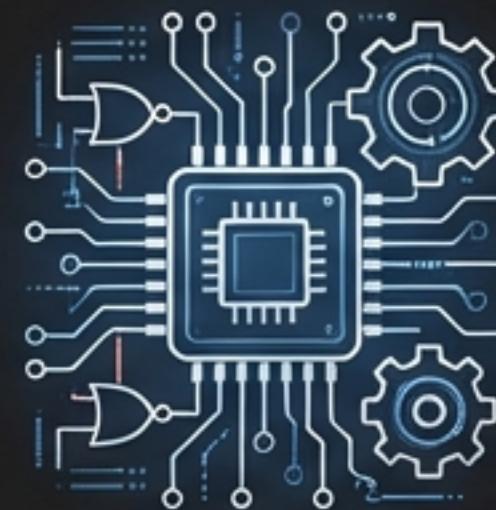


Input (Code)

Computational Thinking:  
การแปลงกระบวนการคิดให้เป็น  
ชุดคำสั่งที่คอมพิวเตอร์เข้าใจได้

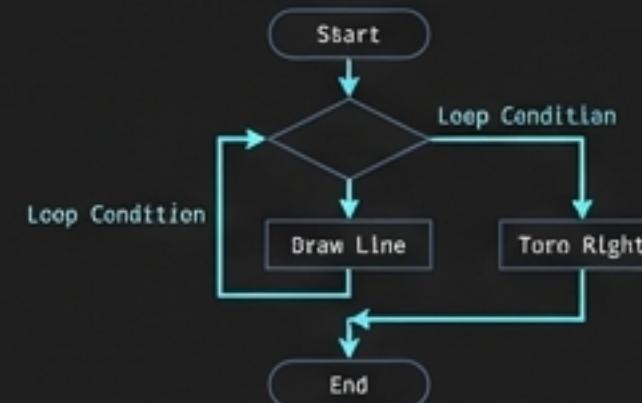
```
for i in range(4):  
    forward(100)  
    right(90)  
}
```

Computational  
Thinking

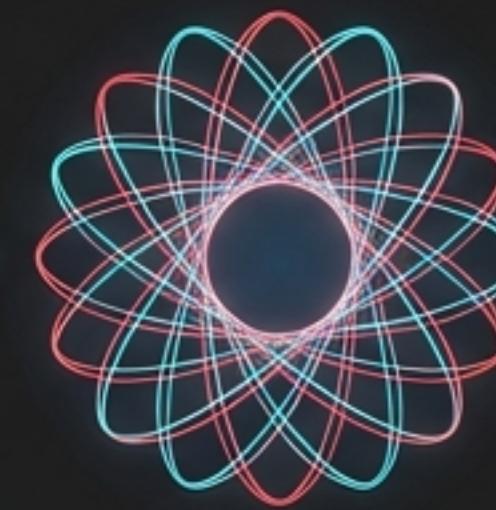


Process (Algorithm)

Algorithmic Visualization:  
การเปลี่ยนล้อจิกที่เป็นนามธรรม  
ให้เป็นผลลัพธ์เชิงประจักษ์

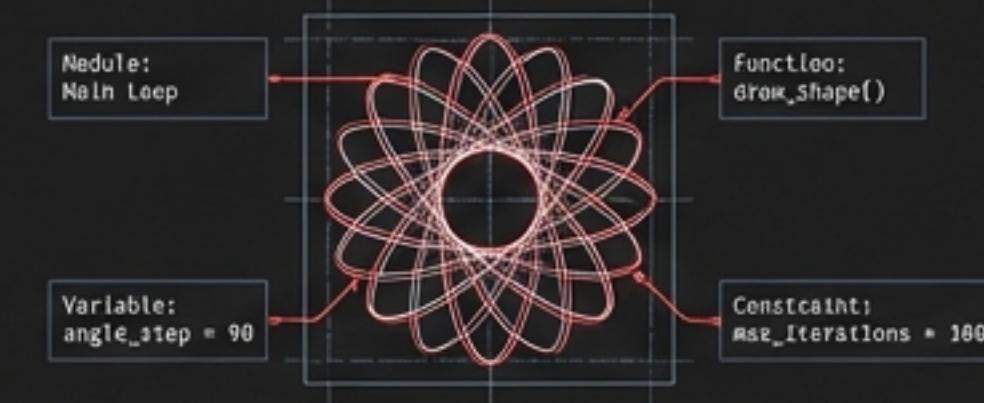


Visualization



Output (Geometric Art)

Architecture Analysis:  
วิเคราะห์โครงสร้างโค้ดเสมือน  
พิมพ์เขียว (Blueprint)

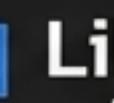


# Phase 1: Environment Setup (การเตรียมสภาพแวดล้อม)

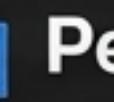
```
import turtle
```

```
turtle = turtle.Turtle()
```

```
turtle.speed(0)
```

 **Library Import:** เรียกใช้โมดูล turtle  
เพื่อเข้าถึงเครื่องมือกราฟิก

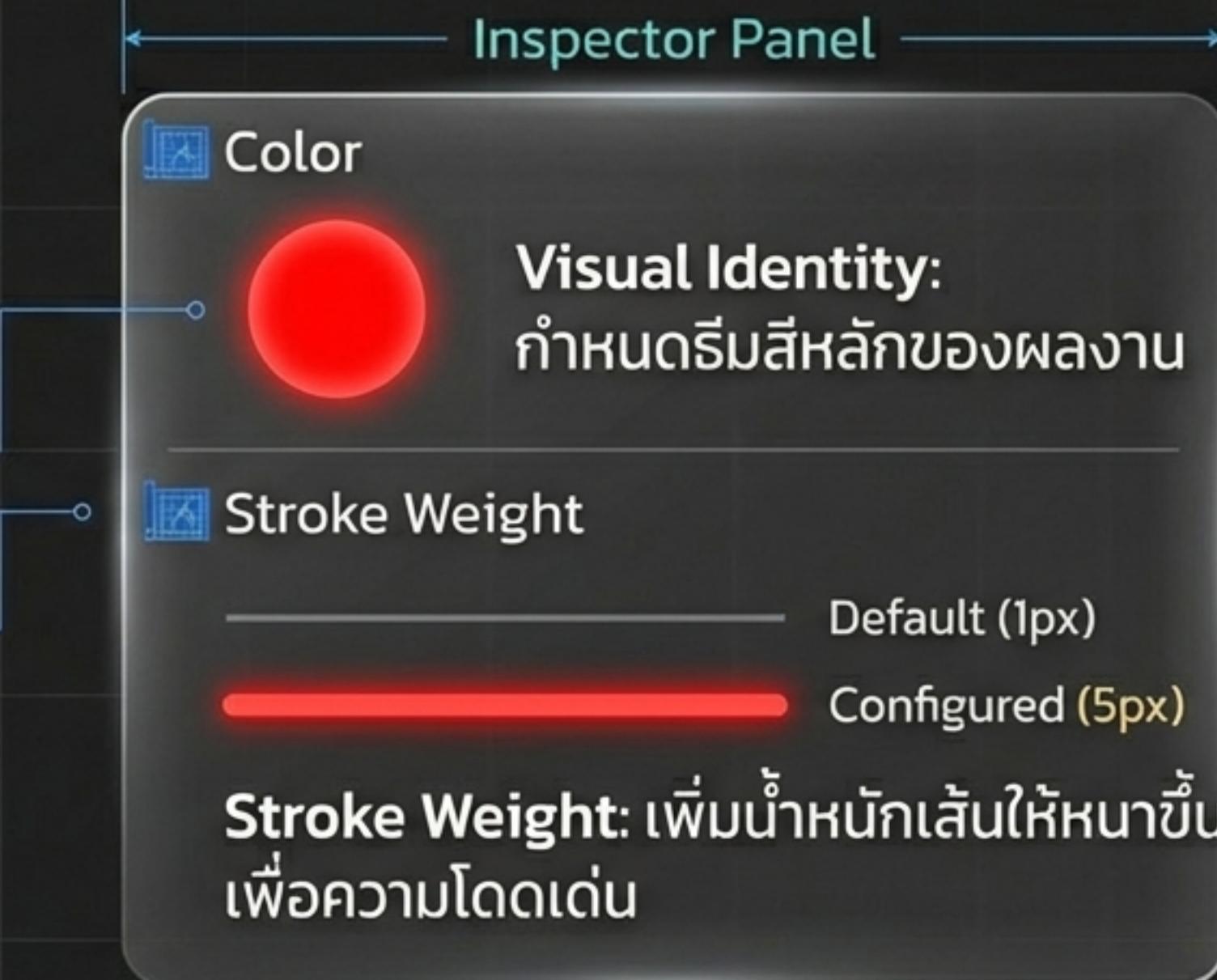
 **Instantiation:** สร้าง Object 'turtle'  
เพื่อเป็นตัวแทนของ 'ปากกา' ในการวาด

 **Performance Optimization:** กำหนด  
ความเร็วสูงสุด (No animation delay)  
เพื่อลดเวลาในการประมวลผล

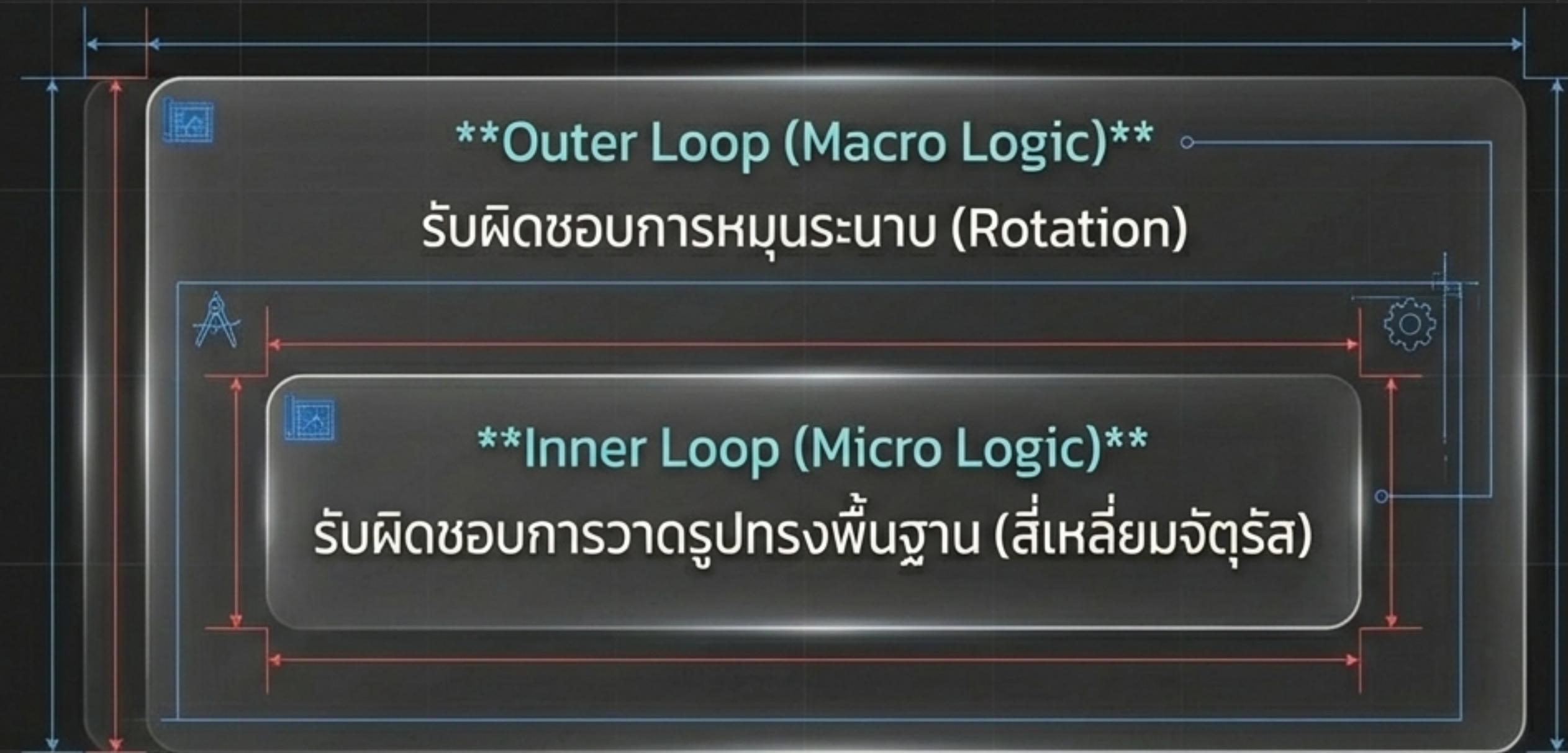
# Aesthetic Configuration (การกำหนดค่าความงาม)

```
4 turtle.color('red')
```

```
5 turtle.pensize(5)
```



# The Core Algorithm: Nested Loops (อัลกอริทึมลูปซ้อนลูป)

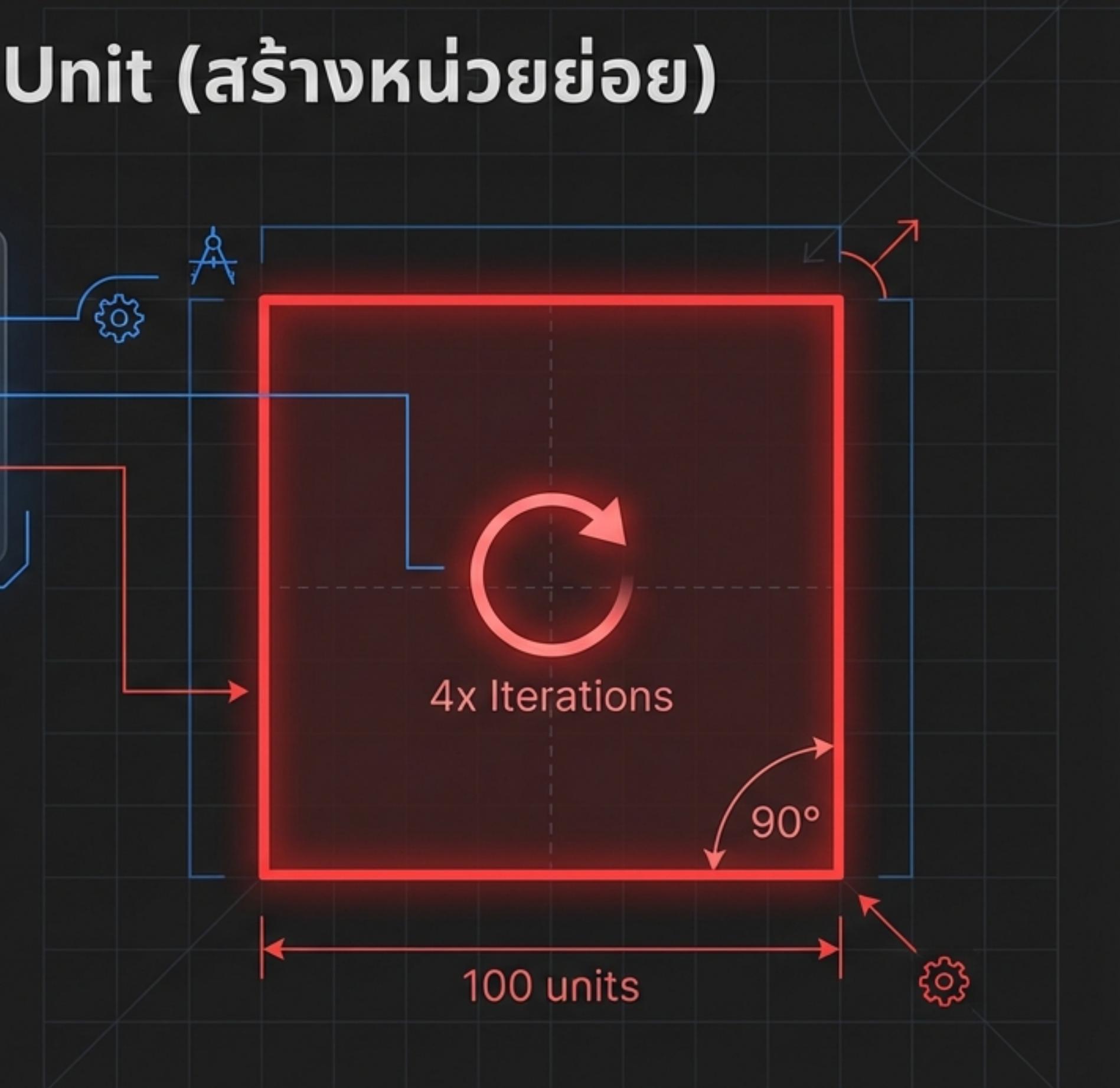


The Power of Repetition: ໂຄດສັນໆ ສາມາຄສຮ້າງໂຄຮງສຮ້າງທີ່ຜົບຜ້ອນໄດ້

# Inner Logic: Creating the Unit (สร้างหน่วยย่อย)

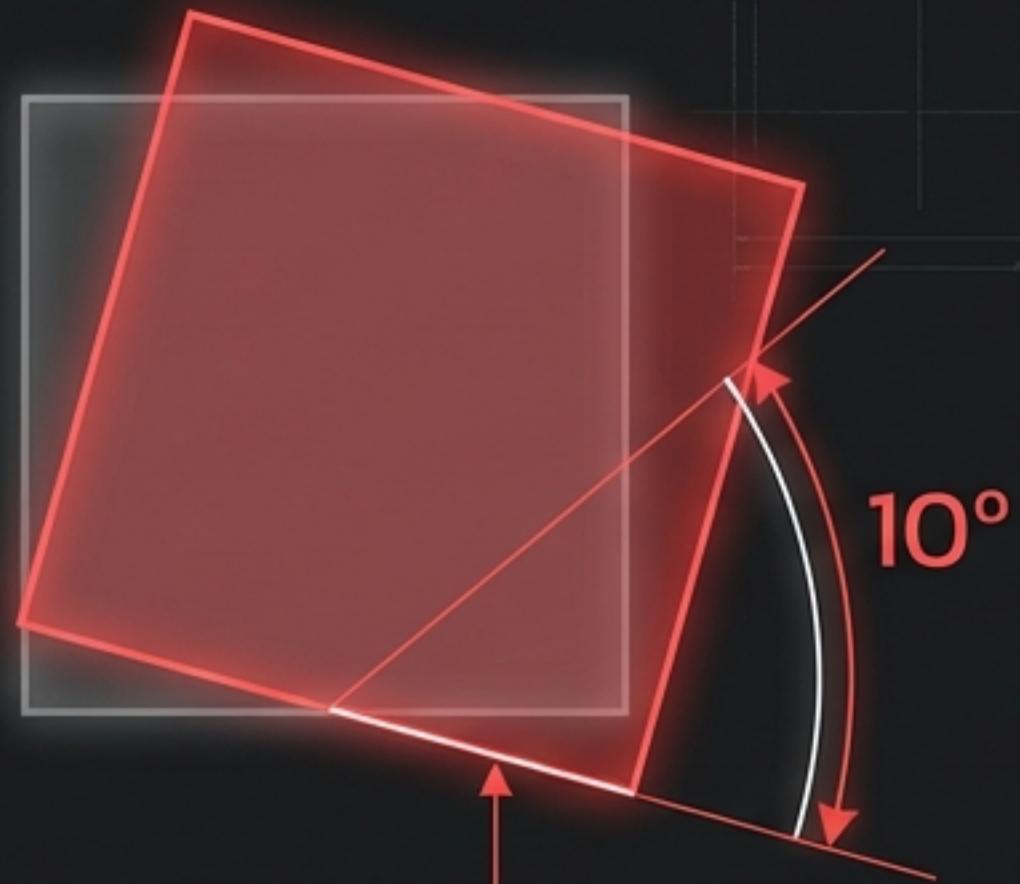
```
9 for j in range(4):  
10     turtle.forward(100)  
11     turtle.right(90)
```

- **Vector Movement:** ลากเส้นตรงระยะ 100 หน่วย / Anuphan/Sukhumvit Set
- **Directional Change:** หักมุมขวา 90 องศาเพื่อสร้างมุมจาก // Anuphan/Sukhumvit Set



# Outer Logic: The Rotation (ការអរគុណសេបាប)

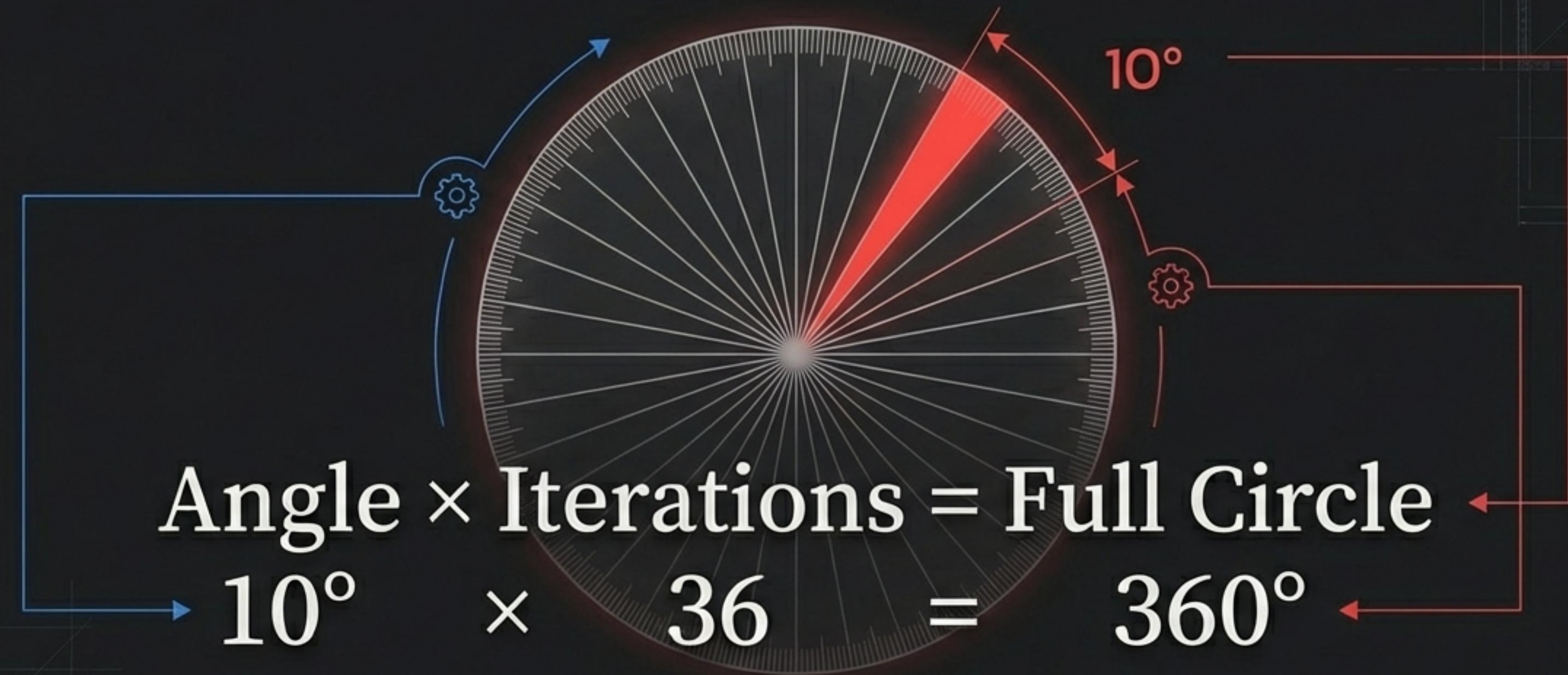
```
for i in range(36):
    # [Inner Loop draws Square]
    turtle.right(10)
```



⚙️ **Incremental Rotation:** អរគុណដើរការ 10 ឯកតា  
ក្នុងការរាយសៀវភៅមួយ 1 រូប

⚙️ **Iteration Count:** range(36) ការបណ្តឹងការ 36 រៀប

# The Mathematics of Symmetry (คณิตศาสตร์แห่งความสมมาตร)



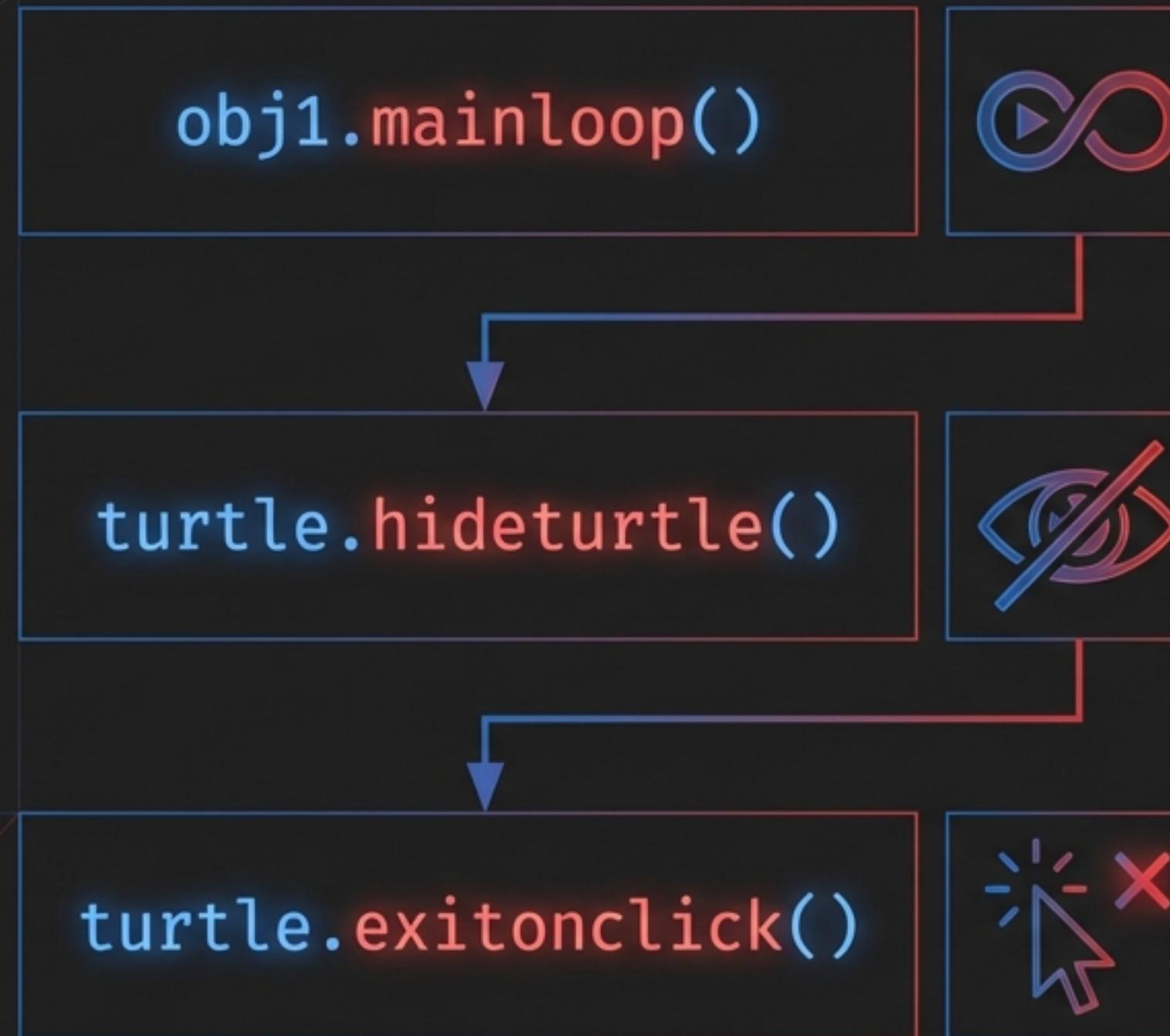
การเลือกใช้ Loop 36 รอบ คูณกับมุม 10 องศา ทำให้เกิดการหมุนครบ 1 รอบวงกลม (360 องศา) พอดี ส่งผลให้ลวดลายบรรจบกันอย่างสมบูรณ์ (Seamless Pattern)

# Execution & Output (ผลลัพธ์การประมวลผล)



Simple Rules, Complex Behavior: กฎพื้นฐานง่ายๆ สู่พฤติกรรมที่ซับซ้อน

# Lifecycle Management (การจัดการวงจรการทำงาน)



**Event Loop:** รอรับ event  
จนกว่าจะจบการทำงาน

**State Cleanup:** ซ่อนตัว cursor  
เพื่อให้แสดงเฉพาะงานศิลปะ

**Clean Exit:** ป้องกันโปรแกรมค้าง  
(Not Responding)

# Protocol: System Termination (ໂປຣໂຕຄອລກາຣັດທີ່ນທັພຍາກ)

## Resource Release:

bye() ກຳລາຍໜ້າຕ່າງກຮາ  
ຟິກແລະ ຄືບໜ່ວຍຄວາມຈຳ

## Canvas Reset:

clear() ແລະ reset() ລ້າງ  
ຄ່າຕ່າງໆ ໃຫ້ກລັບສູ່ສຄານະ  
ເຮັມຕົ້ນ

## Professional Standard:

ກາຣຍືນຍັນວ່າໂປຣແກຣມຈະຈບ  
ກາຣກຳເງານອຍ່າງສມບູຮນ  
(Graceful Shutdown)

```
>_
> turtle.bye()
> turtle.clear()
> turtle.reset()
> System Halted.
```

# Key Takeaways (บทสรุป)



## Modular Logic

การแยกหน้าที่ของ Inner Loop (วิธี) และ Outer Loop (หมุน) ทำให้โค้ดอ่านง่าย



## Precision

คอมพิวเตอร์ช่วยให้วัดซ้ำแม่นยำ 100% ในทุกองศา



## Efficiency

โค้ดเพียงไม่กี่บรรทัดสามารถสร้างงานศิลปะที่ซับซ้อนเกินกว่าจะวาดด้วยมือ



## Controlled Lifecycle

การจัดการ State (Setup -> Loop -> Teardown) คือหัวใจของการเขียนโปรแกรมแบบมืออาชีพ

# The Full Blueprint

```
1 import turtle  
2  
3 turtle = turtle.Turtle()  
4 turtle.speed(0)  
5 turtle.color("red")  
6 turtle.pensize(5)  
7  
8 # Draw a pattern using turtle graphics  
9  
10 for i in range(36):  
11     for j in range(4):  
12         turtle.forward(100)  
13         turtle.right(90)  
14         turtle.right(10)  
15  
16     # End of the pattern  
17  
18     # Hide the turtle  
19  
20     obj1 = turtle.getscreen()  
21     obj1.mainloop()  
22     turtle.hideturtle()  
23     turtle.done()  
24     turtle.exitonclick()  
25     turtle.bye()  
26     turtle.clear()  
27     turtle.reset()
```