

Programming Fundamentals

# Unit 6

# Condition in Python

อ.กীরติบุตร กาญจนเสถียร

# เนื้อหา

- ความหมายของ condition
- If-else
  - ไวยากรณ์ (syntax)
  - else if (elif)
- Nested if
- การใช้งาน operators ใน if-else
- short-hand if

# จุดประสงค์การเรียนรู้

- ทราบความหมายของการเขียนโปรแกรมแบบ condition
- ใช้งาน If-else ได้อย่างถูกต้อง
  - ไวยากรณ์ (syntax)
  - else if (elif)
- ใช้ Nested if ได้อย่างถูกต้อง
- เลือกใช้ operators ใน if-else สำหรับภาษา Python ได้อย่างถูกต้อง
- ใช้งาน if-else ในรูปแบบ short-hand ได้

# Condition คือ ?

Condition แปลว่าเงื่อนไข คือ การตัดสินใจของโปรแกรมว่าจะประมวลผล process ใด โดยดูจากสิ่งที่ระบุไว้ในเงื่อนไขนั้นๆ เช่น หากค่าในตัวแปร “money” น้อยกว่า 500 จะไม่ให้ถอนเงิน เป็นต้น ซึ่งการเขียนเงื่อนไขนั้นเป็นสิ่งที่ถือว่าเป็นพื้นฐานที่โปรแกรมเมอร์ทุกคนควรที่จะต้องทราบและมีความมั่นใจในการใช้งาน

# คำสั่งเงื่อนไข

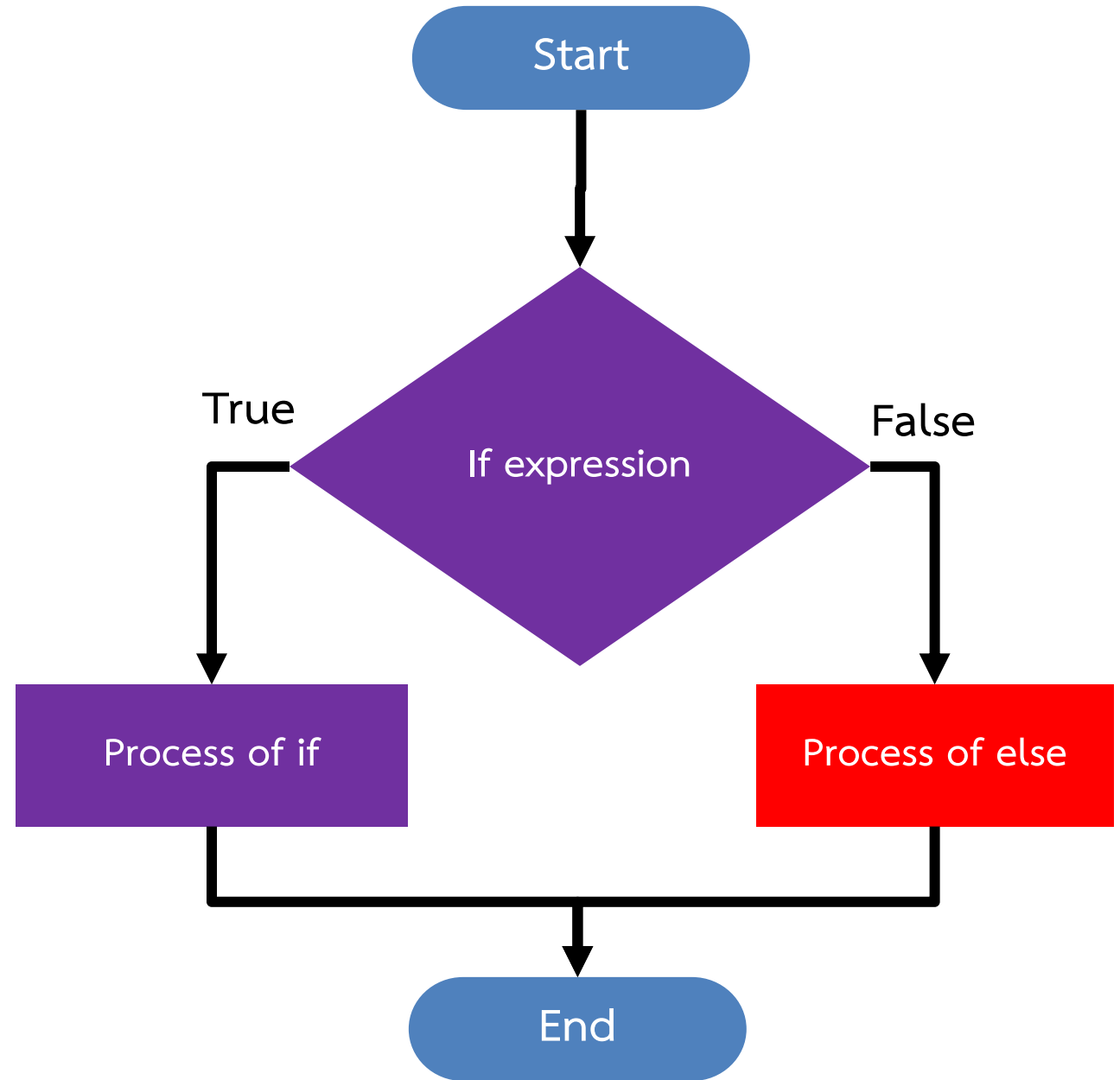
คำสั่งเงื่อนไขใน Python คือ if-else ซึ่งเป็นคำสั่งพื้นฐานที่มีในแทบจะทุกโปรแกรมคอมพิวเตอร์

คำสั่ง	แปลว่า	ในการใช้งานหมายถึง	ลำดับการใช้งาน
if	ถ้า	ถ้าเงื่อนไขเป็นจริง	เงื่อนไขแรกสุด
else	อันอื่นๆ	ถ้าเงื่อนไขใน if เป็นเท็จ แสดงว่าเงื่อนไขอื่นๆเป็นจริง	เงื่อนไขสุดท้ายเท่านั้น (บางครั้งอาจไม่ระบุ เรียกว่า short if)
elif	แล้วถ้า	ถ้าเงื่อนไข if เป็นเท็จ	<ul style="list-style-type: none"><li>- ใช้งานในกรณีที่มีมากกว่า 2 เงื่อนไข</li><li>- เป็นเงื่อนไขตรงกลางระหว่าง if และ else</li><li>- อยู่ในลำดับที่ 2 ถัดจาก if</li></ul>

# โครงสร้างของ if-else

จาก flowchart จะเห็นว่า if-else จะเป็น การตรวจสอบเงื่อนไขว่า expression ใน if เป็นจริงหรือเท็จ หากเป็นจริงก็จะประมวลผล โค้ดใน if และหากเป็นเท็จก็จะประมวลผลโค้ด ใน else

Expression แปลว่า นิพจน์ ในทางคณิตศาสตร์หมายถึงการนำเอา ค่าต่างๆ เช่น ตัวแปร ตัวเลข และสัญลักษณ์มาเขียนร่วมกันอย่างมีความหมาย เช่น  $1+2$ ,  $x < y$ ,  $a \neq b$  เป็นต้น





# ตัวอย่างการใช้งาน if-else

จากภาพ เป็นการกำหนดเงื่อนไขเพื่อตรวจสอบว่าค่าใน `x` เท่ากับ 5 หรือไม่ โดยหากเป็นจริง โปรแกรมจะแสดงข้อความว่า “`x` เท่ากับ 5” ตามคำสั่ง `print` บรรทัดแรก

```
x = 2
y = 6

if x == 5:
    print("x เท่ากับ 5")
else:
    print("x ไม่เท่ากับ 5")

print("จบการทำงาน")
```



และหากเป็นเท็จ จะแสดงข้อความว่า “`x` ไม่เท่ากับ 5” ตามคำสั่ง `print` บรรทัดที่สอง

x ไม่เท่ากับ 5  
จบการทำงาน

# ขอบเขตของ if-else

ในการใช้งาน if-else นั้นจะทำการเขียนคำสั่งลงในไฟล์เหมือนคำสั่งอื่นๆ ซึ่งโปรแกรมเมอร์จะต้องทราบขอบเขตของคำสั่งที่เป็นส่วนหนึ่งของ if-else เพื่อไม่ให้เกิดข้อผิดพลาดในกรณีที่เขียนโค้ดหลายบรรทัดซึ่งอาจทำให้เกิดความสับสนจนเกิด error ได้

ขอบเขตของ if นั้นสังเกตได้อย่างง่ายโดยคำสั่งทุกคำสั่งภายใน if จะถูกเยื้องเข้าไปอย่างน้อย 1 ครั้ง ไม่ว่าจะด้วยการกดปุ่ม “Tab” หรือ “Space Bar” บนคีย์บอร์ดก็ตาม เช่นเดียวกับขอบเขตของ else คำสั่งใน else จะถูกเยื้องเข้าไปเช่นกันดังภาพ

คำสั่ง `print(“จบการทำงาน”)` จะไม่อยู่ใน if-else เนื่องจากไม่ได้ถูกเยื้องเข้าไป โดยจะอยู่ในระนาบเดียวกับ if-else ดังนั้นถือว่าไม่ใช่คำสั่งใน if-else

```
x = 2
y = 6

if x == 5:
    print("x เท่ากับ 5")
else:
    print("x ไม่เท่ากับ 5")

print("จบการทำงาน")
```

x ไม่เท่ากับ 5  
จบการทำงาน



# ขอบเขตของ if-else

โค้ดภายในรอบสีม่วงจะอยู่ใน if-else และเป็นส่วนหนึ่งของ if-else โค้ดทั้งหมดที่เป็นส่วนหนึ่งของ if-else จะต้องเยื้องเข้าไปทั้งหมด ดังภาพ

โค้ดในกรอบสีเขียวจะเป็นส่วนหนึ่งของ if จะทำงานก็ต่อเมื่อเงื่อนไขใน if เป็นจริง สังเกตได้จากการที่โค้ดตั้งแต่บรรทัดที่ 19-24 นั้นอยู่ต่อจาก if ในบรรทัดที่ 18 และทำการเยื้องเข้ามา ซึ่งการเยื้องนี้ถือว่าบรรทัดที่ 19-24 นั้นอยู่ใน if

โค้ดในกรอบสีแดงเป็นส่วนหนึ่งของ else โดยจะทำงานก็ต่อเมื่อ if เป็นเท็จ ซึ่งโค้ดในบรรทัดที่ 26-29 นั้นอยู่ต่อจาก else และเยื้องเข้ามา จึงสรุปได้ว่าโค้ดในส่วนนี้จะทำงานก็ต่อเมื่อ if เป็นเท็จ

โค้ดในบรรทัดที่ 31 นั้นไม่ได้อยู่ใน if-else และไม่เป็นส่วนหนึ่งของ else แม้โค้ดจะอยู่ต่อจาก else ก็ตาม แต่ไม่ได้ถูกเยื้องเข้าไป ดังนั้นไม่ถือว่าอยู่ใน else โค้ดนี้จะแสดงผลทุกครั้งที่มีการประมวลผล if-else เสร็จสิ้น ไม่ว่าโปรแกรมจะทำงานใน if หรือ else ก็ตาม เมื่อประมวลผลใน if-else เสร็จสิ้น ก็จะออกมาประมวลผลโค้ดบรรทัดต่อไปที่ต่อจาก if-else ทันที

```
15 x = 5
16 y = 6
17
18 if x == 5:
19     print("x เท่ากับ 5")
20     z = x+y
21     print("z เท่ากับ x+y")
22     print("z =", z)
23     x = int(input("ระบุค่า x ใหม่: "))
24     print("ค่า x ใหม่ = ", x)
25 else:
26     print("x ไม่เท่ากับ 5")
27     print("y = ", y)
28     y = int(input("ระบุค่า y ใหม่: "))
29     print("ค่า y ใหม่ = ", y)
30
31 print("จบการทำงาน")
```

Shell x

```
x เท่ากับ 5
z เท่ากับ x+y
z = 11
ระบุค่า x ใหม่: 14
ค่า x ใหม่ = 14
จบการทำงาน
```

# Syntax of if-else

การเขียน if-else นั้นจะมี syntax หรือ ไวยากรณ์การเขียนดังนี้

1. เริ่มต้นด้วยคำสั่ง “if” ตามด้วยเงื่อนไข ปิดท้ายด้วย “:”
2. บรรทัดใหม่ ให้เยื้อง (indent) 1 ครั้ง
3. ใส่โค้ดที่จะให้ทำเมื่อเงื่อนไข if เป็นจริง
4. ระบุคำสั่ง “else” ปิดท้ายด้วย “:” (ไม่ระบุเงื่อนไข)
5. บรรทัดใหม่ ให้เยื้อง (indent) 1 ครั้ง
6. ใส่โค้ดที่จะให้ทำเมื่อเงื่อนไข if เป็นเท็จ

if **เงื่อนไข:**

หาก if เป็นจริง ให้ทำตรงนี้

**else:**

หาก if เป็นเท็จ ให้ทำตรงนี้ โดยข้าม if มาเลย

# คำสั่ง else ไม่ต้องระบุเงื่อนไข

สาเหตุที่ else ไม่ต้องระบุเงื่อนไข เนื่องจากเราไม่สนใจเงื่อนไขอื่นๆนอกจากเงื่อนไขที่ระบุใน if เท่านั้น หมายความว่า หากเงื่อนไขที่ระบุใน if เป็นเท็จ เงื่อนไขอื่นๆจะเป็นจริงหรือไม่ก็ตามโปรแกรมก็จะประมวลผลโค้ดใน else อยู่ดี

```
1 if a:
2     # โค้ดส่วนนี้ทำงานหาก a เป็นจริง
3 else:
4     # โค้ดส่วนนี้ทำงานหาก a เป็นเท็จ
```

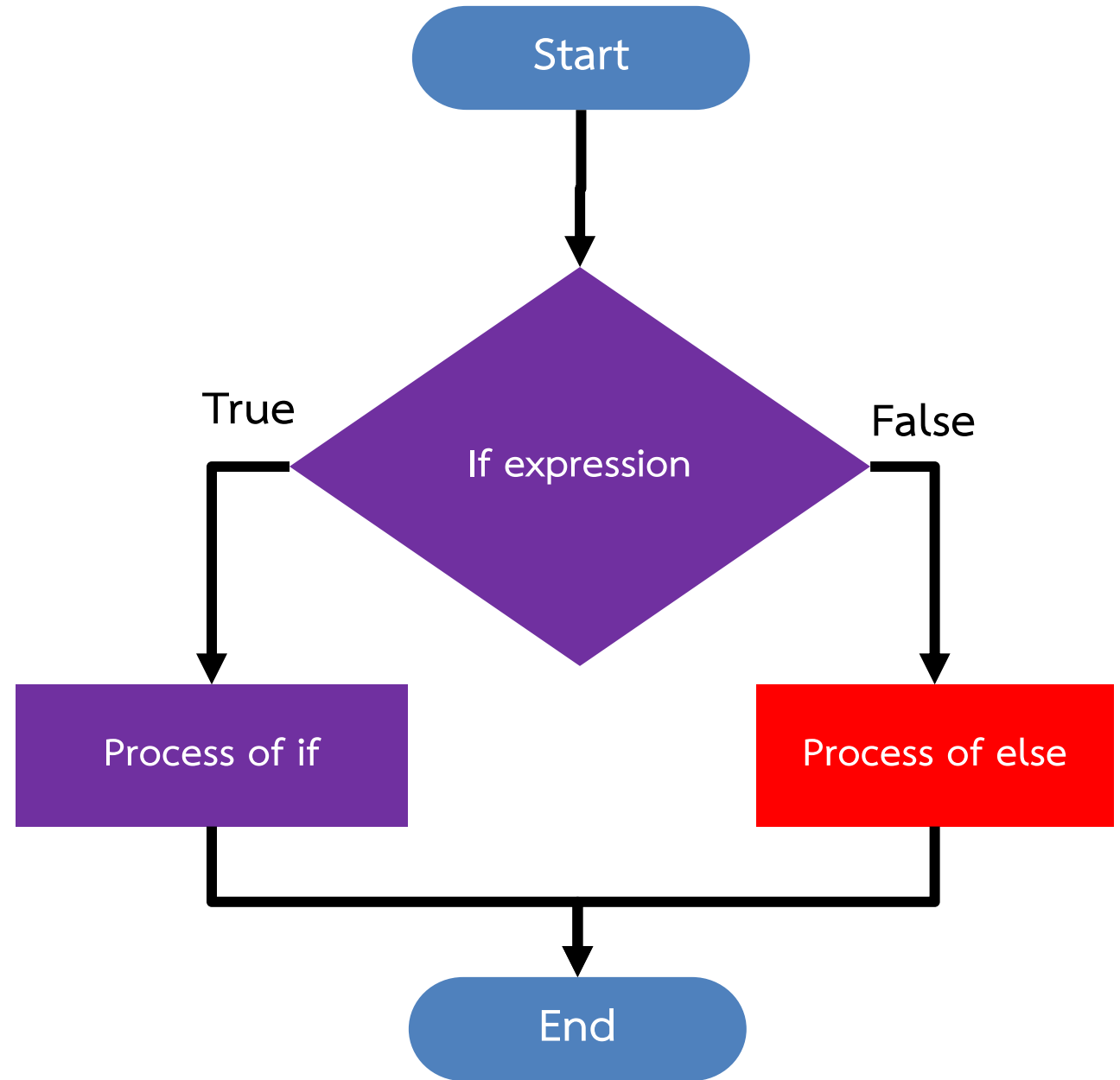
เช่น มีเงื่อนไขอยู่จำนวน 5 เงื่อนไข ได้แก่ a, b, c, d, และ e

เราสนใจเพียงเงื่อนไขเดียวคือ a ดังนั้นเรากำหนดเงื่อนไข a ที่ if และจะประมวลผลเมื่อ a เป็นจริงเท่านั้น

เงื่อนไขอื่นๆจะเป็นเงื่อนไขของ else เนื่องจากเราไม่สนใจเงื่อนไขอื่นๆเลย ไม่ว่า b, c, d, e จะเป็นจริงหรือเท็จ ตราบใดที่ a เป็นเท็จ โปรแกรมจะประมวลผลโค้ดใน else เสมอ

# คำสั่ง else ไม่ต้องระบุเงื่อนไข

จาก flowchart ด้านขวาเป็นโครงสร้างของ if-else จะเป็นว่าเรามีการกำหนดแค่ if เท่านั้น ดังนั้นหาก if เป็นเท็จ ก็จะไปประมวลผลโค้ดใน else แทนที่โดยไม่ต้องระบุเงื่อนไขใดๆอีก



# ตัวอย่าง syntax of if-else

จาก syntax ในสไลด์ก่อนหน้านี้ เขียนเป็นตัวอย่างตามรูป การใช้งาน else จะมีเครื่องหมายและไวยากรณ์เหมือน if

1. เริ่มต้นด้วยคำสั่ง “if” ตามด้วยเงื่อนไข ปิดท้ายด้วย “:”

2. บรรทัดใหม่ ให้เยื้อง (indent) 1 ครั้ง

3. ใส่โค้ดที่จะให้ทำเมื่อเงื่อนไข if เป็นจริง

4. ระบุคำสั่ง “else” ปิดท้ายด้วย “:”

5. บรรทัดใหม่ ให้เยื้อง (indent) 1 ครั้ง

6. ใส่โค้ดที่จะให้ทำเมื่อเงื่อนไข if เป็นเท็จ

```
1 a = 5
2 b = 4
3
4 if a < b:
5     print("a is less than b")
6 else:
7     print("a is more than b")
8
```

```
a is more than b
> |
```

# แบบฝึกหัด if-else (ทำในคาบเรียน)

เขียนโปรแกรมดังต่อไปนี้

1. ตรวจสอบค่า input จำนวน 2 ค่าว่ามากกว่า น้อยกว่า
2. ตรวจสอบค่า input ว่ามีข้อความ “Thailand” อยู่ใน input หรือไม่
3. ตรวจสอบค่า input ว่าเป็นจำนวนเลขคู่หรือคี่
4. ตรวจสอบค่า input ว่าหารด้วยเลข 5 ลงตัวโดยไม่เหลือเศษหรือไม่

# elif

`elif` ย่อมาจากคำว่า “else if” คือ เงื่อนไขเพิ่มเติม โดยปกติแล้วการใช้งาน if-else จะมีเพียงแค่ 2 เงื่อนไขเท่านั้นคือ if และ else โดยหากเงื่อนไขใน if เป็นจริงก็จะประมวลผลโค้ดที่อยู่ใน if และหากเป็นเท็จก็จะประมวลผลโค้ดที่อยู่ใน else แต่หากมีมากกว่า 2 เงื่อนไขก็สามารถใช้คำสั่งเพิ่มเติมคือ `elif` ซึ่งจะเป็นการกำหนดเงื่อนไขเพิ่มในกรณีที่มีมากกว่า 2 เงื่อนไขสำหรับภาษา Python

ขอบเขตของ if นั้นสังเกตได้อย่างง่ายโดยคำสั่งทุกคำสั่งภายใน if จะถูกเยื้องเข้าไป 1 ครั้ง ไม่ว่าจะด้วยการกดปุ่ม “Tab” หรือ “Space Bar” บนคีย์บอร์ดก็ตาม เช่นเดียวกับขอบเขตของ else คำสั่งใน else จะถูกเยื้องเข้าไปเช่นกันดังภาพ

คำสั่ง `print(“จบการทำงาน”)` จะไม่อยู่ใน if-else เนื่องจากไม่ได้ถูกเยื้องเข้าไป โดยจะอยู่ในระนาบเดียวกับ if-else ดังนั้นถือว่าไม่ใช่คำสั่งใน if-else

```
x = 2
y = 6

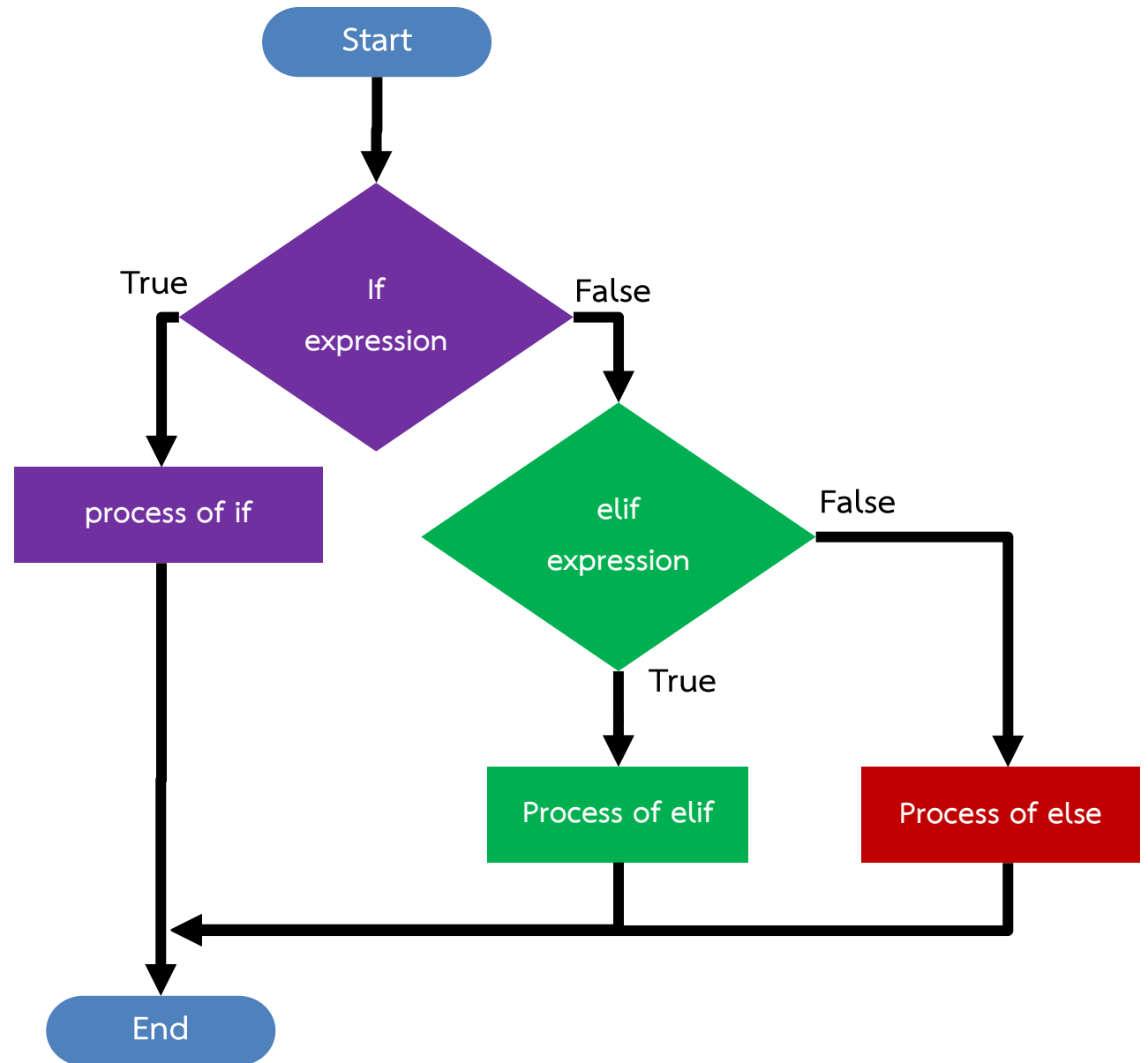
if x == 5:
    print("x เท่ากับ 5")
else:
    print("x ไม่เท่ากับ 5")

print("จบการทำงาน")
```

x ไม่เท่ากับ 5  
จบการทำงาน

# โครงสร้างของ elif

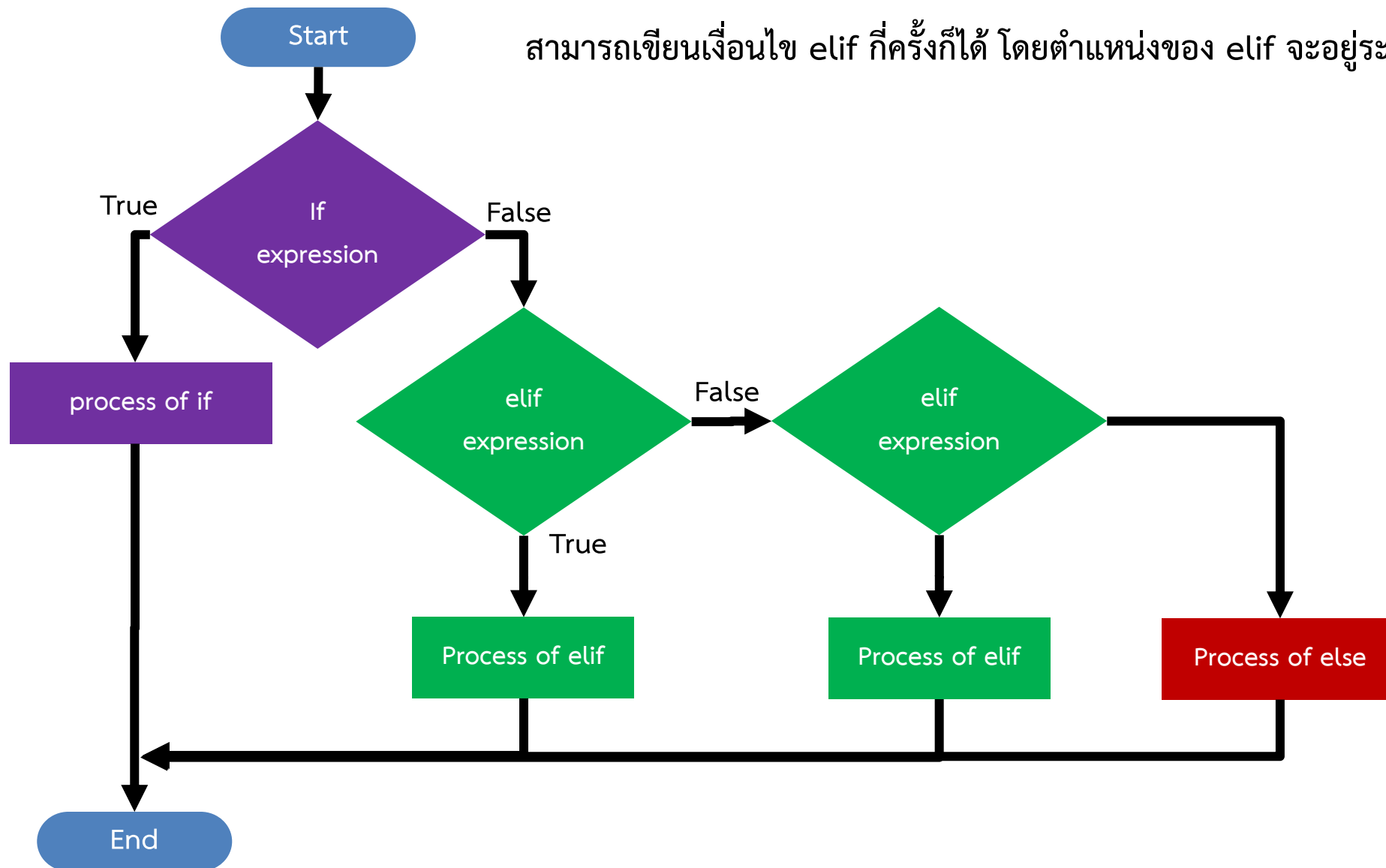
elif นั้นจะเป็นเงื่อนไขที่จะถูกประมวลผล หาก if เป็นเท็จ และจะสามารถมี elif จำนวนเท่าใดก็ได้ เมื่อเขียน elif จะต้องระบุเงื่อนไขด้วยทุกครั้งและมีไวยากรณ์การเขียนโค้ดเหมือนกับ if ทุกประการ หาก elif เป็นเท็จระบบจะประมวลผลโค้ดใน else





# โครงสร้างของ elif (กรณีมี elif มากกว่า 1 เงื่อนไข)

สามารถเขียนเงื่อนไข elif ที่กี่ครั้งก็ได้ โดยตำแหน่งของ elif จะอยู่ระหว่าง if และ else เสมอ



# คำสั่ง elif ต้องระบุ

เราจะใช้งาน elif ก็ต่อเมื่อมีเงื่อนไขที่เราสนใจมากกว่า 1 เงื่อนไข

```
1 if a:
2     # โค้ดส่วนนี้ทำงานหาก a เป็นจริง
3 elif c:
4     # โค้ดส่วนนี้ทำงานหาก a เป็นเท็จและ c เป็นจริง
5 elif d:
6     # โค้ดส่วนนี้ทำงานหาก a, c เป็นเท็จ และ d เป็นจริง
7 else:
8     # โค้ดส่วนนี้ทำงานหาก a, c, d เป็นเท็จ
9
```

เช่น มีเงื่อนไขอยู่จำนวน 5 เงื่อนไข ได้แก่ a, b, c, d, และ e

เราสนใจจำนวน 3 เงื่อนไข ได้แก่ a, c, และ d ดังนั้นเราระบุ a ที่ if และระบุเงื่อนไข c, d ที่ elif (เงื่อนไขละ 1 elif)

และเมื่อ a, c, d เป็นเท็จ ก็จะประมวลผลโค้ดใน else เนื่องจากเราไม่สนใจเงื่อนไขที่เหลือซึ่งก็คือ b, และ e ว่าจะเป็นอย่างจริงหรือเท็จ

หมายเหตุคอมพิวเตอร์จะประมวลผลตรวจสอบทีละเงื่อนไขจากบนลงล่าง

# elif

จากตัวอย่างจะเป็นการกรอกตัวเลขและโปรแกรมจะแสดงผลค่าที่ผู้ใช้กรอกว่า มากกว่า น้อยกว่า หรือเท่ากับ 5 ซึ่งจะมีด้วยกัน 3 เงื่อนไข ได้แก่

คำสั่ง	เงื่อนไข	การใช้งาน
if	$x > 5$	เป็นเงื่อนไขแรก ต้องระบุเงื่อนไข
elif	$x < 5$	เป็นเงื่อนไขที่สอง ต้องระบุเงื่อนไข
else	เงื่อนไขใดๆที่ไม่ใช่ if และ elif	เป็นเงื่อนไขสุดท้าย ไม่ต้องระบุเงื่อนไข

จะเห็นว่า elif นั้นจะเป็นเงื่อนไขเพิ่มเติมและจะปรากฏอยู่ตรงกลางระหว่าง if และ else โดยที่ elif นั้นจะต้องระบุเงื่อนไขและมีไวยากรณ์ทุกอย่างเหมือนกับ if

```
33
34 while True:
35     x = int(input("กรอกตัวเลข: "))
36
37     if x > 5:
38         print("X มากกว่า 5")
39     elif x < 5:
40         print("X น้อยกว่า 5")
41     else:
42         print("X เท่ากับ 5")
43     print("จบ")
44
```

Shell ×

>>> %Run operators.py

```
กรอกตัวเลข: 5
x เท่ากับ 5
จบ
กรอกตัวเลข: -2
x น้อยกว่า 5
จบ
กรอกตัวเลข: 10
x มากกว่า 5
จบ
กรอกตัวเลข:
```

# Syntax of elif

การเขียนคำสั่ง elif นั้นจะมีไวยากรณ์การเขียนเหมือน if ทุกอย่าง จากตัวอย่างจะเห็นว่าคำสั่ง elif ในบรรทัดที่ 39 นั้นจะระบุเงื่อนไขและปิดท้ายด้วย “:” หลังจากนั้นโค้ดในบรรทัดต่อไปจะต้องเยื้องเข้าไป 1 ครั้งและจะเป็นโค้ดที่จะถูกประมวลผลเมื่อเงื่อนไขของ elif เป็นจริง

จากผลลัพธ์ในกรอบสีเหลี่ยมสีแดง จะเห็นว่าผู้ใช้กรอกค่า “-2” ซึ่งมีค่าน้อยกว่า 5 ตามเงื่อนไขในบรรทัดที่ 39 ดังนั้นจึงถือว่าเงื่อนไขใน elif เป็นจริง จากนั้นโปรแกรมจึงประมวลผลโค้ดในบรรทัดที่ 40 ซึ่งเป็นส่วนหนึ่งของ elif เราจึงเห็นผลลัพธ์เป็นข้อความว่า “x น้อยกว่า 5”

```
33
34 while True:
35     x = int(input("กรอกตัวเลข: "))
36
37     if x > 5:
38         print("x มากกว่า 5")
39     elif x < 5:
40         print("x น้อยกว่า 5")
41     else:
42         print("x เท่ากับ 5")
43     print("จบ")
44
```

Shell x

```
>>> %Run operators.py
```

```
กรอกตัวเลข: 5
x เท่ากับ 5
จบ
```

```
กรอกตัวเลข: -2
x น้อยกว่า 5
จบ
```

```
กรอกตัวเลข: 10
x มากกว่า 5
จบ
```

```
กรอกตัวเลข:
```

# จำนวนครั้งการใช้งาน elif

ในกรณีที่มีเงื่อนไขจำนวนมาก จะสามารถเขียนเงื่อนไขเพิ่มเติมได้ในส่วนของ elif และจะเขียน elif จำนวนเท่าใดก็ได้ ดังภาพ จะเห็นว่าเงื่อนไขที่เพิ่มขึ้นมา จะปรากฏอยู่ในคำสั่ง elif เท่านั้น

มีเพียงข้อจำกัดเดียวสำหรับจำนวนการเขียน if-else คือ ในแต่ละครั้งที่มีการใช้งาน if-else นั้นจะสามารถระบุ if และ else ได้เพียงครั้งเดียวเท่านั้น

หมายเหตุ เราสามารถใช้ if-else ก็ครั้งก็ได้ในโปรแกรมเดียวกัน แต่ในแต่ละ if-else จะมี if และ else ได้เพียงครั้งเดียว แต่ elif ได้มากเท่าที่ต้องการ

```
33
34 carName = ["Toyota", "Honda", "BMW", "Isuzu", "Suzuki"]
35 choice = input("กรอกยี่ห้อรถที่ต้องการซื้อ: ")
36
37 if choice == "Toyota":
38     print("เรามียี่ห้อ Toyota จำหน่าย")
39 elif choice == "Honda":
40     print("เรามียี่ห้อ Honda จำหน่าย")
41 elif choice == "BMW":
42     print("เรามียี่ห้อ BMW จำหน่าย")
43 elif choice == "Isuzu":
44     print("เรามียี่ห้อ Isuzu จำหน่าย")
45 elif choice == "Suzuki":
46     print("เรามียี่ห้อ Suzuki จำหน่าย")
47 else:
48     print("ขออภัย เราไม่มียี่ห้อที่คุณต้องการ")
49
50
```

Shell ×

```
>>> %Run operators.py
```

```
กรอกยี่ห้อรถที่ต้องการซื้อ: Mercedes-Benz
```

```
ขออภัย เราไม่มียี่ห้อที่คุณต้องการ
```

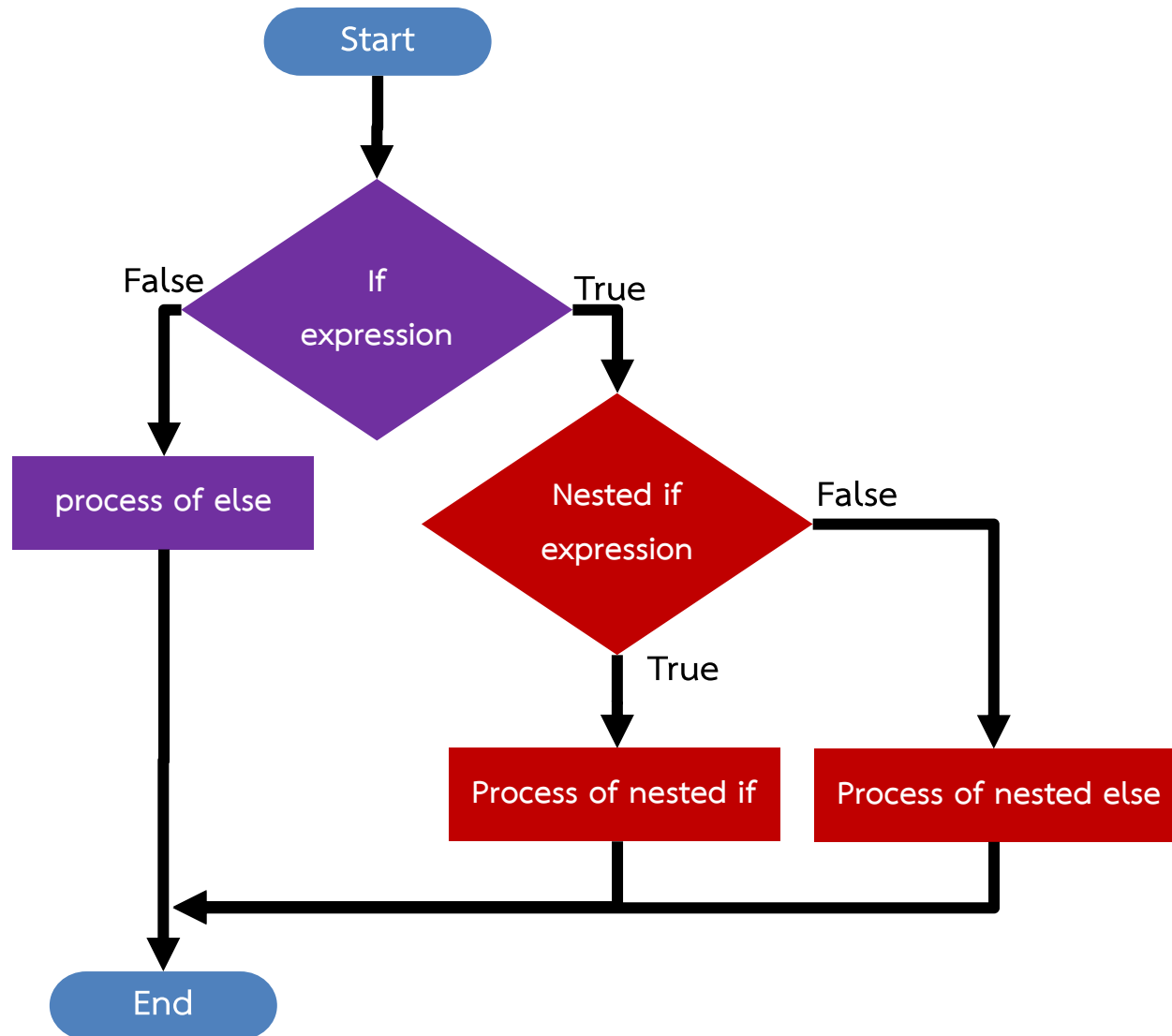
```
>>> |
```

# Nested if

ในการใช้งาน if-else นั้นเราสามารถที่จะเขียนคำสั่ง if-else ซ้อนอยู่ข้างใน if-else อีกอันหนึ่งได้ เรียกว่า “nested if” ซึ่ง if-else ที่อยู่ภายในก็จะถือว่าเป็นส่วนหนึ่งของ if-else หลักอีกที และจะสามารถระบุ if-else ซ้อนกันกี่ชั้นก็ได้

สามารถซ้อน if-else เข้าไปในส่วนของ if, elif, หรือ else ก็ได้ทั้งสิ้นโดยไม่มีข้อจำกัดใดๆ

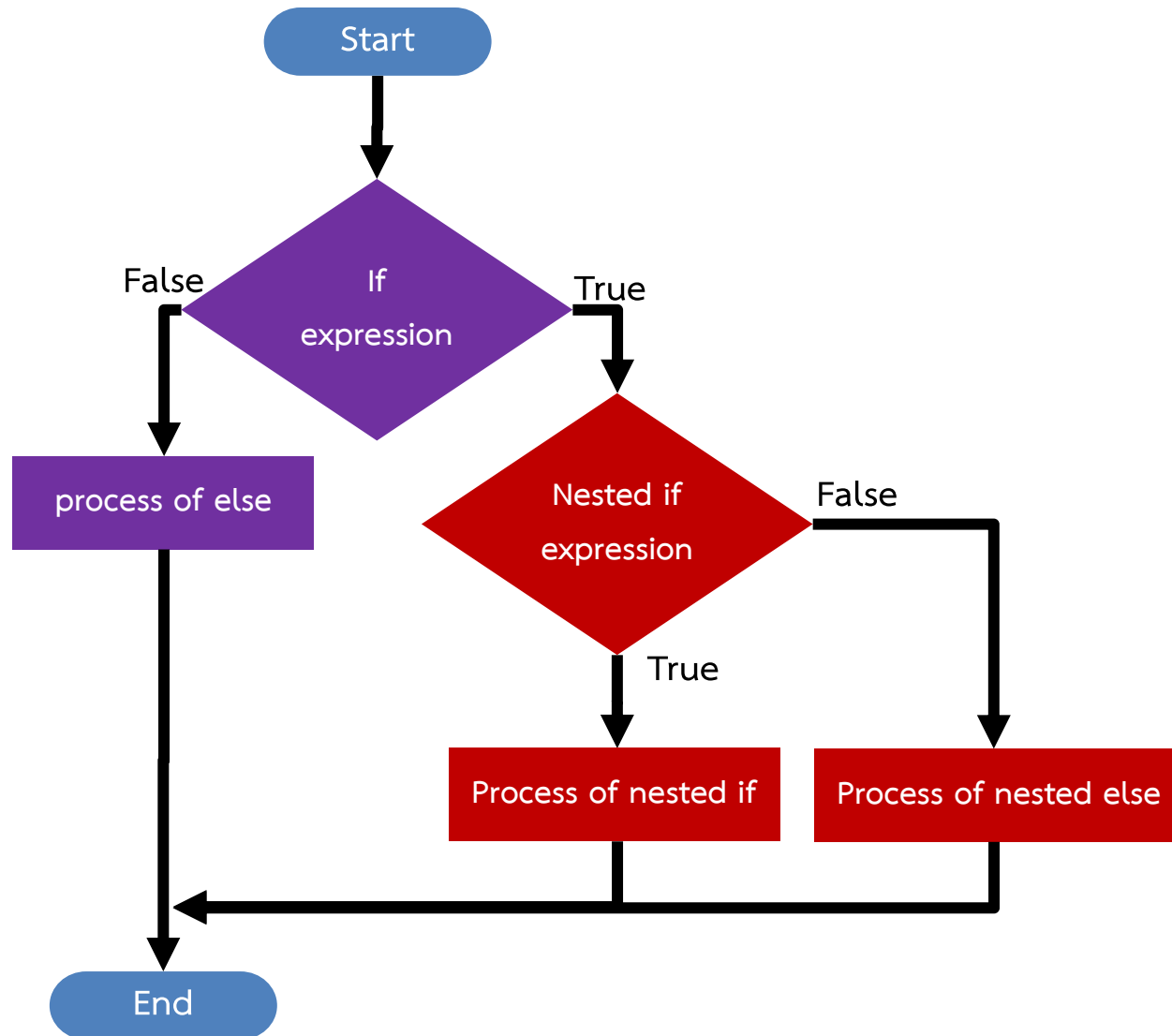
# โครงสร้างของ nested if



จาก flowchart จะเห็นว่ามี if อีกอันหนึ่ง (สีแดง) ซ่อนอยู่ข้างใน if หลัก (สีม่วง) เรียกว่า nested if ซึ่งจะถูกประมวลผลก็ต่อเมื่อ if หลักเป็นจริง

หาก if หลักเป็นเท็จ โปรแกรมก็จะไม่เข้าไปที่ nested if และจะประมวลผลใน “process of else” แทน

# โครงสร้างของ nested if

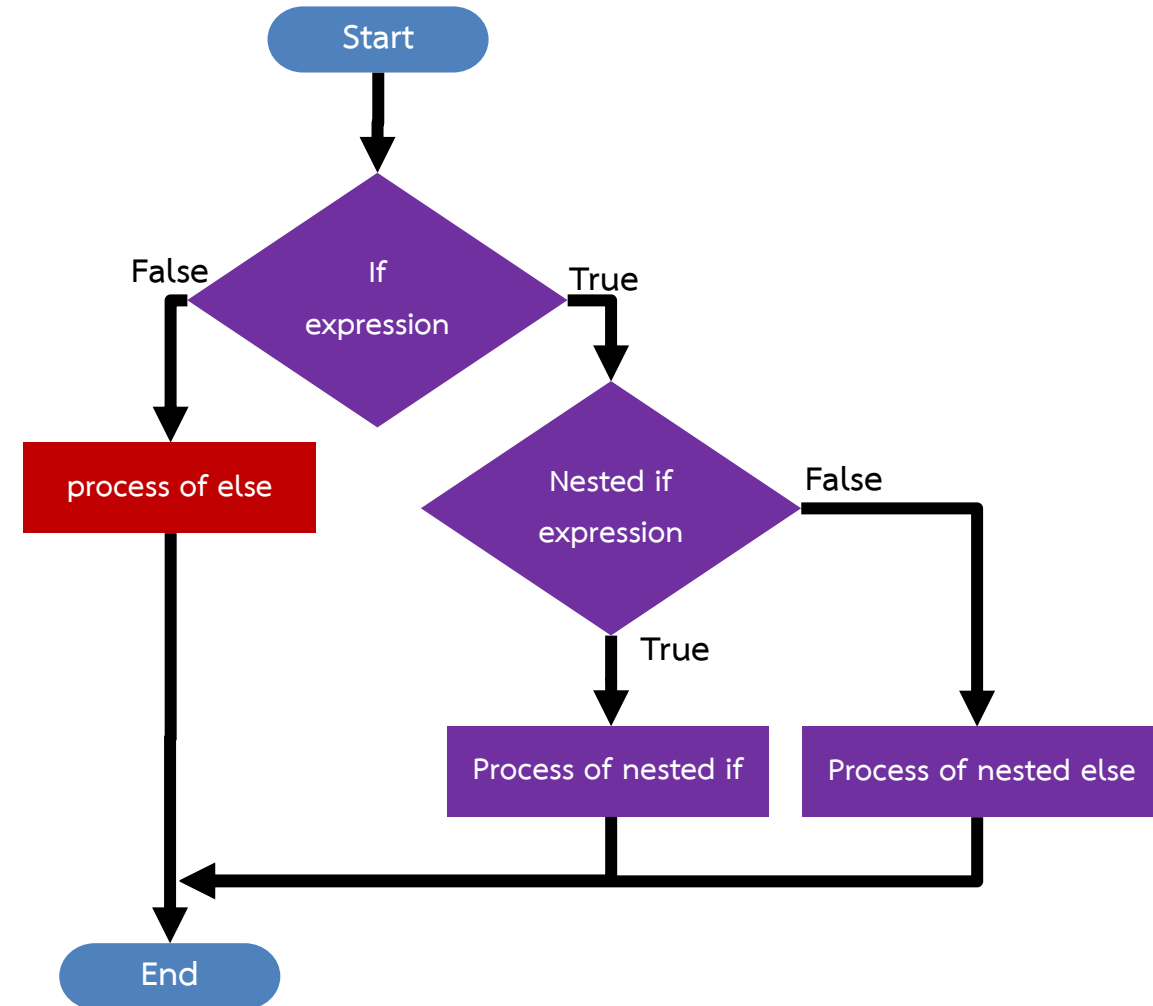
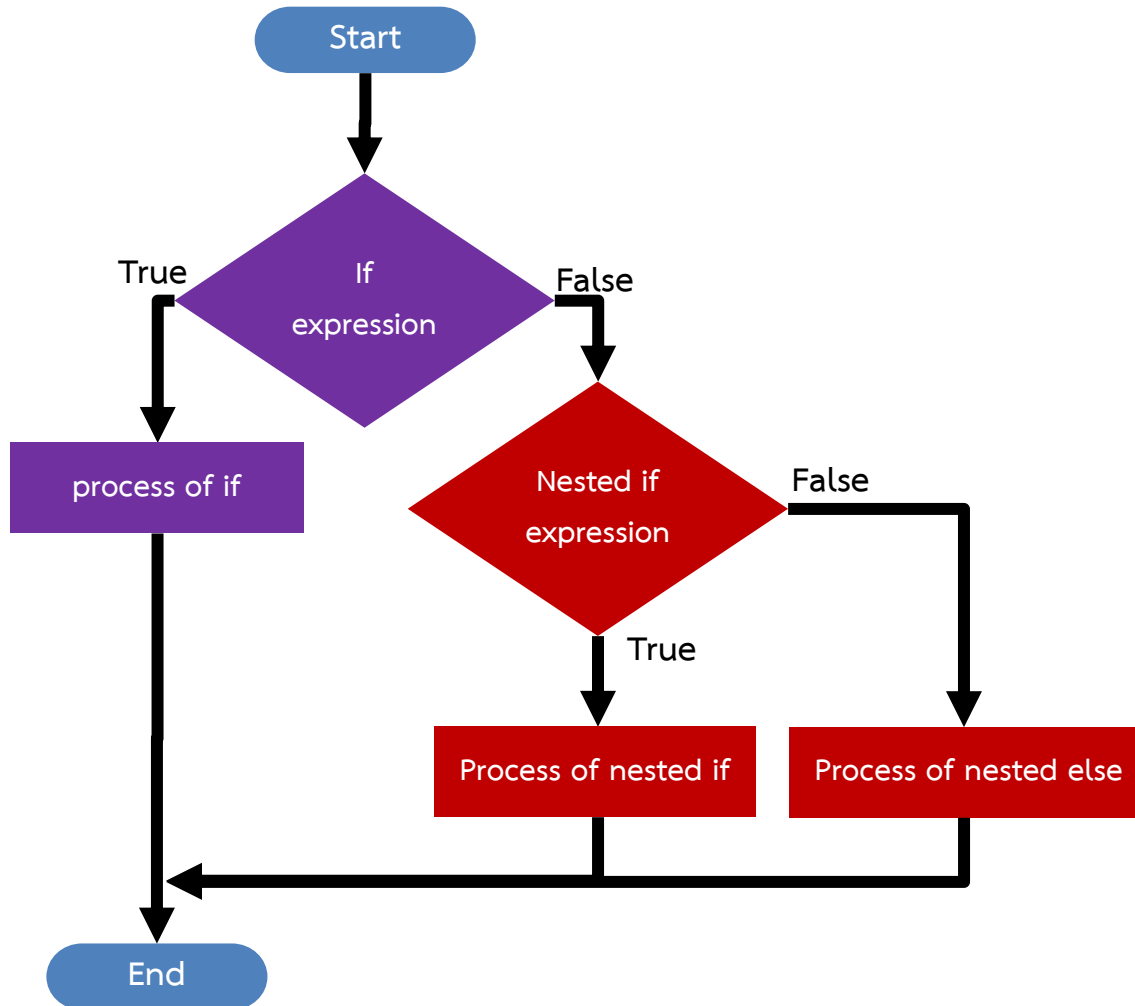


จาก flowchart จะเห็นว่ามี if อีกอันหนึ่ง (สีแดง) ซ่อนอยู่ข้างใน if หลัก (สีม่วง) เรียกว่า nested if ซึ่งจะถูกประมวลผลก็ต่อเมื่อ if หลักเป็นจริง

หาก if หลักเป็นเท็จ โปรแกรมก็จะไม่เข้าไปที่ nested if และจะประมวลผลใน “process of else” แทน



# nested-if อยู่ได้ทั้ง if, elif และ else



# Nested if

จากตัวอย่างจะเห็นว่า if-else ในบรรทัดที่ 7-11 จะเป็น if-else อีกอันที่เป็นส่วนหนึ่งของ if ในบรรทัดที่ 4

โค้ดในบรรทัดที่ 8-9 จะถือว่าเป็นส่วนหนึ่งของ if ในบรรทัดที่ 7 ดังนั้นโค้ดในบรรทัดที่ 5-11 จะถูกประมวลผลก็ต่อเมื่อ if ในบรรทัดที่ 4 เป็นจริง

โค้ดในบรรทัดที่ 8-9 จะถูกประมวลผลก็ต่อเมื่อ if ในบรรทัดที่ 4 และ 7 เป็นจริง

โค้ดในบรรทัดที่ 11 จะประมวลผลก็ต่อเมื่อ if ในบรรทัดที่ 4 เป็นจริงและ if ในบรรทัดที่ 7 เป็นเท็จ

โค้ดในบรรทัดที่ 13 จะถูกประมวลผลก็ต่อเมื่อ if ในบรรทัดที่ 4 เป็นเท็จ

```
1 pinN = 1234
2 pinI = int(input("กรอกรหัสผ่าน: "))
3 balance = 10000
4 if pinI==pinN:
5     print("รหัสถูกต้อง")
6     amount = int(input("ระบุจำนวนที่ต้องการถอน: "))
7     if amount < balance:
8         balance -= amount
9         print("ถอนเงินสำเร็จ\nยอดคงเหลือเท่ากับ:", balance)
10    else:
11        print("จำนวนเงินไม่เพียงพอ")
12 else:
13     print("รหัสไม่ถูกต้อง")
14
```

Shell ×

```
>>> %Run operators.py
```

กรอกรหัสผ่าน: 1234

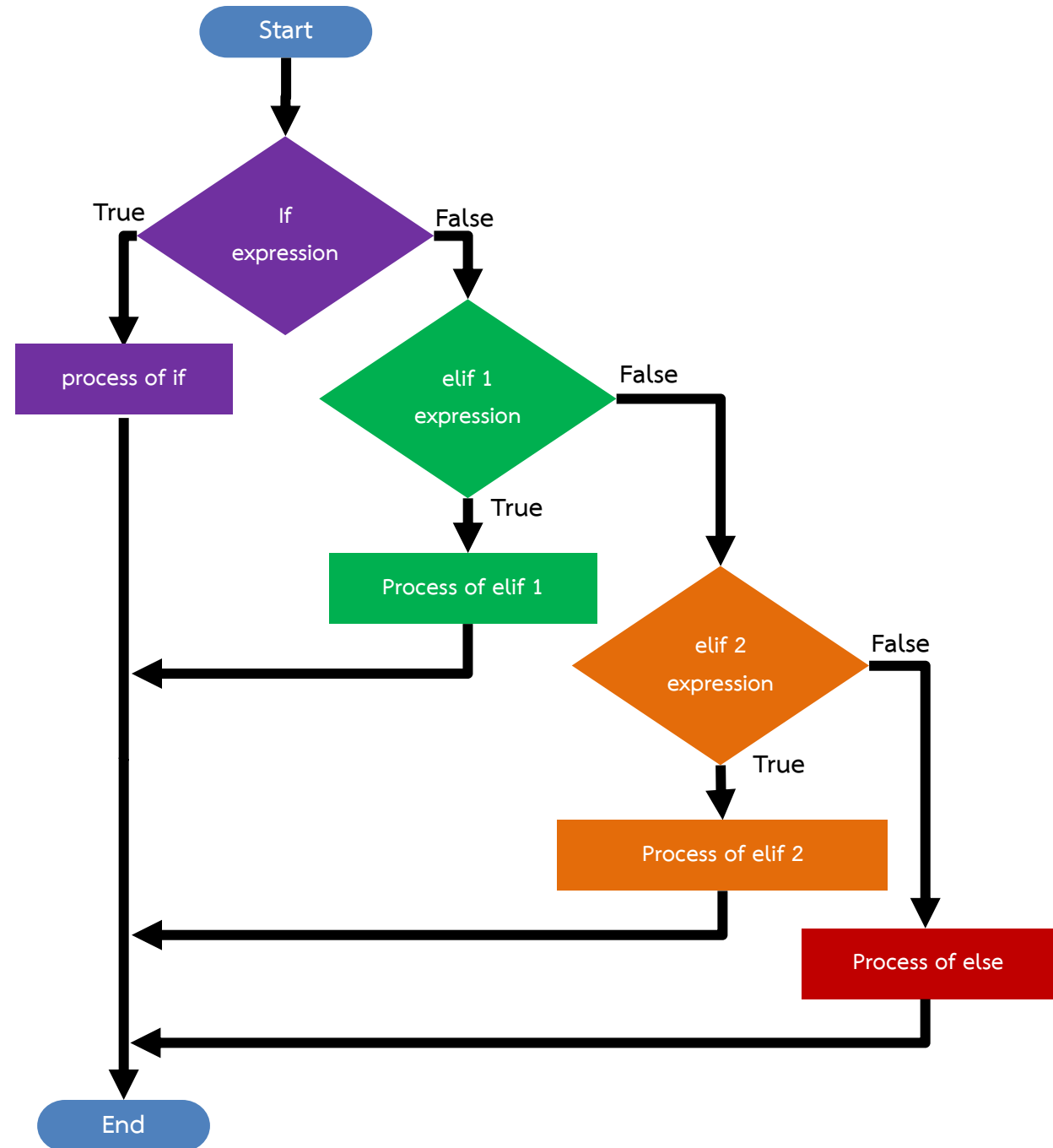
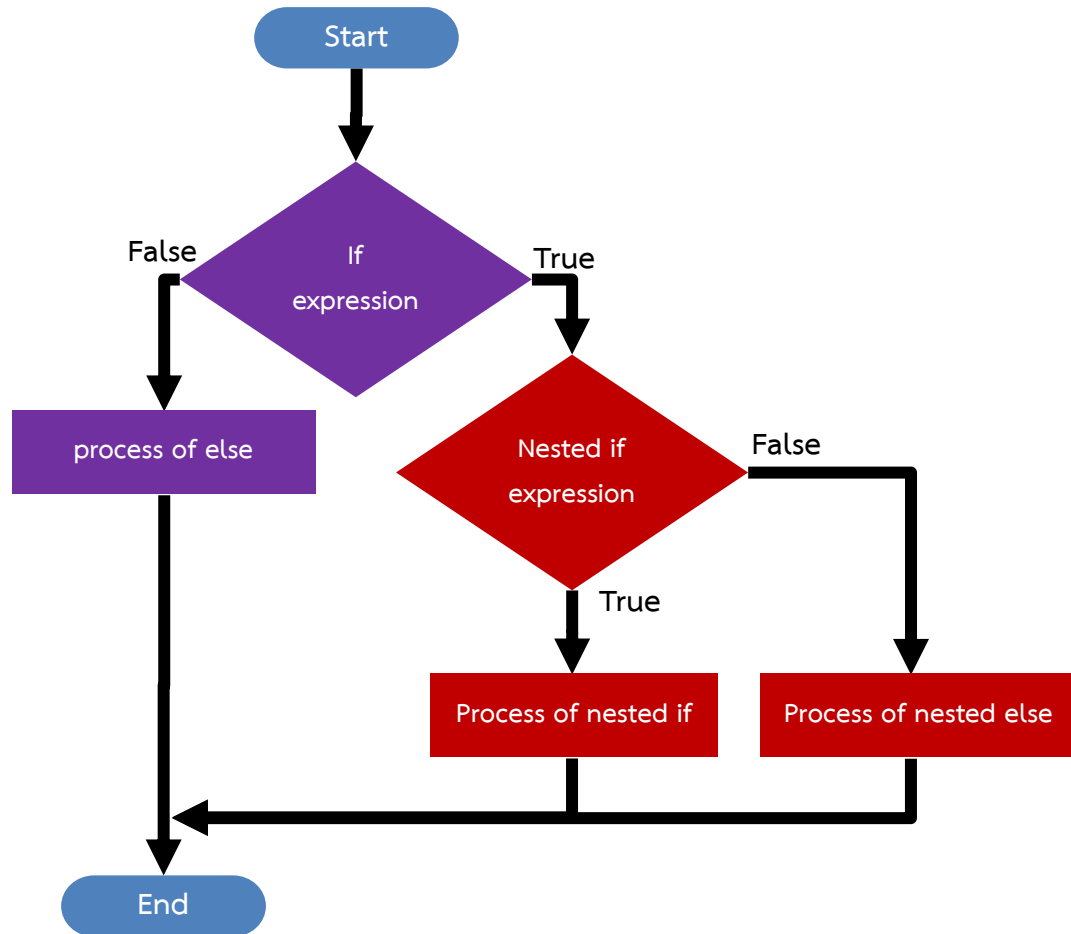
รหัสถูกต้อง

ระบุจำนวนที่ต้องการถอน: 3000

ถอนเงินสำเร็จ

ยอดคงเหลือเท่ากับ: 7000

# โครงสร้างของ nested if และ elif



# การโค้ด nested if และ elif

จาก flowchart ในสไลด์ก่อนหน้านี้ จะเห็นว่าการใช้งาน elif และ nested if นั้นจะมีโครงสร้างที่คล้ายกัน โดย elif นั้นจะดูเหมือนกับว่าเป็น nested if ของ if หลัก แต่ในการเขียนโค้ดนั้นจะสามารถเห็นความแตกต่างได้ค่อนข้างชัดเจน

ภาพบน จะเป็นการใช้งาน nested if ซึ่ง nested if จะต้องเยื้องเข้าไปจาก if หลัก และจะถือว่าเป็นส่วนหนึ่งของ if โดยจะถูกประมวลผลก็ต่อเมื่อ if ในบรรทัดที่ 17 เป็นจริง สังเกตว่าใน nested if จะมี elif ด้วยซึ่งเป็นส่วนหนึ่งของ if-else ระหว่างบรรทัดที่ 19-24

ภาพล่างจะเป็น elif ซึ่งจะเขียนอยู่ในระดับเดียวกับ if หลักในบรรทัดที่ 17 โดยโปรแกรมจะประมวลผลบรรทัดที่ 19 ก็ต่อเมื่อ if ในบรรทัดที่ 17 เป็นเท็จ

```
16
17 if x == y:
18     print("x = y")
19     if y < z:
20         print("y < z")
21     elif y > z:
22         print("y > z")
23     else:
24         print("y = z")
25 else:
26     print("x != y")
27
```

```
16
17 if x < y:
18     print("x < y")
19 elif x > y:
20     print("y > z")
21 else:
22     print("x = y")
23
```

# การใช้งาน operators ใน if-else

เนื่องจากนิพจน์ (expression) ของ if-else นั้นจะถูกประมวลผลว่าเป็นจริงหรือเท็จ ดังนั้นนิพจน์จะต้องพิสูจน์ได้ว่าเป็นจริงหรือเป็นเท็จ เช่น “A < B” หรือ “5 > 6” แม้กระทั่ง string ก็สามารถนำมาใช้ใน if expression ได้ เช่น ““THAILAND” in x” ดังภาพด้านล่าง

```
17
18 y = "AUSTRALIA"
19
20 if "THAILAND" in y:
21     print("TRUE")
22 else:
23     print("FALSE")
24
```

Shell ×

```
>>> %Run operators.py
FALSE
>>>
```

```
17
18 x = "THAILAND"
19
20 if "THAILAND" in x:
21     print("TRUE")
22 else:
23     print("FALSE")
24
```

Shell ×

```
>>> %Run operators.py
TRUE
>>>
```

# ตัวอย่างการใช้งาน operators ใน if-else

ในการเขียน if expression นั้นโดยปกติแล้วจะระบุค่าจำนวน 2 ค่าโดยจะตรวจสอบเงื่อนไขด้วย operator ดังตัวอย่างที่ 1-2 ในตาราง

ในส่วนของตัวอย่างที่ 3-5 จะเห็นว่าเราสามารถเขียนนิพจน์ซ้อนได้ และนำนิพจน์ทางซ้ายและทางขวามาเปรียบเทียบกับในทางตรรกศาสตร์ก็ได้

ค่าแรก	operator	ค่าที่สอง	ความหมาย
A	>	B	ค่าในตัวแปร A <b>มากกว่า</b> ค่าในตัวแปร B
“THAILAND”	in	x	<b>มี</b> คำว่า “THAILAND” <b>ใน</b> ตัวแปร x
a + b	<	20	เมื่อนำ a+b จะมีค่าน้อยกว่า 20
x==y	and	p==r	ค่าในตัวแปร x เท่ากับ y <b>และ</b> ค่าในตัวแปร p เท่ากับ r
x<y	and	p!=r	ค่าในตัวแปร x น้อยกว่า y <b>และ</b> ค่าในตัวแปร p ไม่เท่ากับ r
x%5	!=	0	ค่าใน x เมื่อนำมาหารด้วย 5 แล้วจะไม่มีเศษ (เศษเป็น 0)

# ตัวอย่างการใช้งาน operators ใน if-else

จากภาพ เป็นตัวอย่างการเขียน if expression จากตารางในสไลด์  
ก่อนหน้านี้ โดยเราสามารถระบุ expression ได้ทั้งใน if และ elif แต่ไม่  
ต้องระบุใน else

โดยโปรแกรมจะทำการประมวลผลว่า expression ที่ระบุนั้นเป็นจริง  
หรือเท็จ ซึ่งหากเป็นจริงก็จะประมวลผลใน if และหากเป็นเท็จก็จะไป  
ประมวลผลใน elif หรือ else ตามที่ได้ศึกษาในเนื้อหาก่อนหน้านี้

ในบางตำรา อาจใช้คำว่า condition แทนคำว่า expression

```
if A>B:
```

```
if "THAILAND" in x:
```

```
if a + b < 20:
```

```
if x==y and p==r:
```

```
if x<y and p!=r:
```

```
if x%5 != 0:
```

# operators ที่ไม่สามารถใช้ใน if expression ได้

มีบาง operators ที่ไม่สามารถนำมาใช้งานใน expression ของ if-else ได้ เช่น “=” เนื่องจากเครื่องหมาย “=” นั้นจะเป็น assignment operators ซึ่งก็คือตัวดำเนินการในการกำหนดค่า โดยเราไม่สามารถกำหนดค่าให้กับตัวแปรในขณะที่เปรียบเทียบเงื่อนไขใน if-else ได้ ดังนั้นตัวดำเนินการในประเภนี้จะไม่สามารถนำมาใช้ใน if expression ได้

จากตัวอย่างจะเห็นว่าการนำตัวดำเนินการ “+=” ซึ่งเป็นหนึ่งในตัวดำเนินการในการกำหนดค่า (assignment operators) มาใช้งานผลลัพธ์ที่ได้จะปรากฏว่ามีข้อผิดพลาด (error) ขึ้นและโปรแกรมแสดงข้อความว่า “invalid syntax” คือไวยากรณ์การเขียนนั้นไม่ถูกต้อง

```
17
18 x = 0
19 y = 5
20
21 if x+=y:
22     print("TRUE")
23 else:
24     print("FALSE")
25
```

Shell ×

```
>>> %Run operators.py
Traceback (most recent call last):
  File "C:\Users\PC\Dropbox\RMUTT\slide
tors.py", line 23
    if x+=y:
        ^^
SyntaxError: invalid syntax
>>>
```



# แบบฝึกหัด elif (ทำในคาบเรียน)

เขียนโปรแกรมดังต่อไปนี้

1. ตรวจสอบค่า input จำนวน 2 ค่าว่ามากกว่า น้อยกว่า หรือเท่ากัน
2. ตรวจสอบเพศว่าเป็นผู้ชาย ผู้หญิง หรือไม่ระบุ
3. ตรวจสอบสถานะของผู้ใช้ว่าเป็นเยาวชน วัยรุ่น วัยกลางคน หรือวัยชรา

# short-hand if

หากมีเพียง 1 statement ให้ประมวลผล เราสามารถเขียน statement ในบรรทัดเดียวกับเงื่อนไขของ if ได้ ดังตัวอย่าง

```
1 a = 3
2 b = 4
3
4 if a<b:print("a น้อยกว่า b")
5
```

Shell ×

```
>>> %Run 'for loop.py'
```

```
a น้อยกว่า b
```

```
>>> |
```

# short-hand if-else

หากมีเพียง 1 statement สำหรับ if และ else สามารถเขียนให้อยู่ในบรรทัดเดียวกันได้  
ดังตัวอย่างในรูปทั้ง 2 ด้านขวา

```
1 a = 10
2 b = 8
3
4 print("a มากกว่า b") if a>b else print("a น้อยกว่า b")
5
```

Shell ×

```
>>> %Run 'for loop.py'
a มากกว่า b
>>>
```

```
1 a = 1
2 b = 5
3
4 print("a มากกว่า b") if a>b else print("a น้อยกว่า b")
5
```

Shell ×

```
>>> %Run 'for loop.py'
a น้อยกว่า b
>>>
```

# short-hand if-else

หากมีเงื่อนไขมากกว่า 2 เรายังสามารถเขียนให้อยู่ในรูป short-hand ในบรรทัด

เดียวกันได้ ดังตัวอย่าง จะมี 3 เงื่อนไข คือ a มากกว่า น้อยกว่า หรือเท่ากับ b

```
1 a = 5
2 b = 5
3
4 print("a มากกว่า b") if a>b else print("a น้อยกว่า b") if a<b else print("a เท่ากับ b")
5
```

```
<
Shell x
>>> %Run 'for loop.py'
a เท่ากับ b
```

```
1 a = 5
2 b = 5
3
4 print("a มากกว่า b") if a>b else print("a น้อยกว่า b") if a<b else print("a เท่ากับ b")
5
```

```
<
Shell x
>>> %Run 'for loop.py'
a เท่ากับ b
```

# short-hand if-else

แม้ว่าการเขียนในรูปแบบ short-hand จะทำให้โค้ดอยู่ในบรรทัดเดียวกัน แต่ลำดับการเรียงกันระหว่าง if, elif, และ else ยังคงแบบเดิม ดังรูปด้านล่างจะเห็นว่าเงื่อนไข elif จะอยู่ตรงกลางเหมือนการเขียนแบบปกติ

```
1 a = 5
2 b = 5
3
4 print("a มากกว่า b") if a>b else print("a น้อยกว่า b") if a<b else print("a เท่ากับ b")
```

Shell x

```
>>> %Run 'for loop.py'
a เท่ากับ b
```

Programming Fundamentals

# End of Unit 6

## Condition in Python

# ข้อมูลอ้างอิง

1. ณัฐวัตร คำภักดี. (2561). คู่มือเขียนโปรแกรมด้วยภาษาไพธอน python. กรุงเทพฯ : โปรวิชัน , 2561 โดย, ISBN 9786162046971
2. บัญชา ปะสีละเตสัง. (2019). หนังสือ การเขียนโปรแกรมด้วย Python สำหรับผู้เริ่มต้น. ซีเอ็ดดูเคชั่น, บมจ. ISBN: 9786160833979
3. w3schools. Python Tutorial. <https://www.w3schools.com/python/>
4. programiz. Python if...else Statement. <https://www.programiz.com/python-programming/if-elif-else>
5. geeksforgeeks. Control Flow in Python. <https://www.geeksforgeeks.org/python-if-else/>
6. tutorialspoint. Python IF...ELIF...ELSE Statements. [https://www.tutorialspoint.com/python/python\\_if\\_else.htm](https://www.tutorialspoint.com/python/python_if_else.htm)

## แบบฝึกหัดท้ายบทที่ 6

1. อธิบายความหมายของ condition หรือเงื่อนไขในการเขียนโปรแกรม
2. เขียนโปรแกรมสร้างเครื่องคิดเลขโดยใช้ if-else กำหนดให้มีการบวก ลบ คูณ และหาร
3. สร้างระบบปลดล็อคโทรศัพท์มือถือ โดยอนุญาตให้ใส่รหัสผ่านผิดได้ไม่เกิน 3 ครั้ง จะต้องใช้  
nested-if