

คำสั่งเงื่อนไข
และคำสั่งวนซ้ำ

นิพจน์ทางตรรกศาสตร์

- นิพจน์ทางตรรกศาสตร์ คือ นิพจน์ที่ให้ผลลัพธ์เป็นจริงหรือเท็จเท่านั้น
- ตัวอย่าง ของคำถามที่ได้คำตอบเป็นจริงหรือเท็จ
 - ต้องการรับเครื่องดื่มหรือชาลาเปาหรือไม่?
 - x มากกว่า 10 หรือไม่?
 - หากคำตอบเจอหรือยัง?
 - 5 หหาร 153 ลงตัวหรือไม่?

ค่าคงที่แบบบูล (Boolean constants)

ใช่ → True

ไม่ใช่ → False

ข้อควรระวัง

- ในไพธอน ตัวพิมพ์ใหญ่พิมพ์เล็กมีผล (case sensitive) นั่นคือ
 - **False** กับ **false** มีความหมายไม่เหมือนกัน
- สำหรับค่าคงที่แบบบูล อย่าลืมขึ้นต้นด้วยอักษรพิมพ์ใหญ่:
 - **True**, และ
 - **False**

การสร้างนิพจน์ตรรกศาสตร์

- ตัวดำเนินการเปรียบเทียบ:

- สร้างนิพจน์ตรรกศาสตร์โดยการเปรียบเทียบค่าสองค่า

เท่ากับ	==
ไม่เท่ากับ	!=
มากกว่า	>
มากกว่าหรือเท่ากับ	>=
น้อยกว่า	<
น้อยกว่าหรือเท่ากับ	<=

การสร้างนิพจน์ตรรกศาสตร์

- ตัวดำเนินการแบบบูล (ตัวดำเนินการเชิงตรรกะ)
 - สร้างนิพจน์ตรรกศาสตร์โดยการเชื่อมนิพจน์ตรรกศาสตร์เข้าด้วยกัน

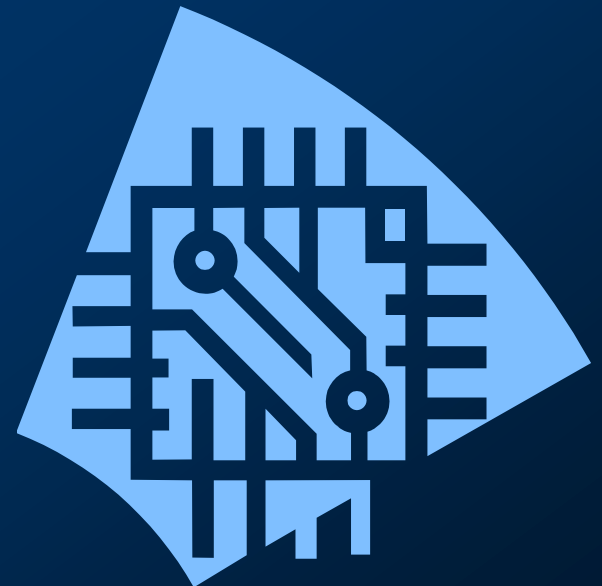
Boolean operators	operators
และ	and
หรือ	or
นิเสธ	not

ทบทวนความจำ

True and True	True
True and False	False
False and True	False
False and False	False

not True	False
not False	True

True or True	True
True or False	True
False or True	True
False or False	False



ตัวอย่างนิพจน์ทางตรรกศาสตร์

สมมติให้ x มีค่าเท่ากับ 4

Expression	Value
$x < 5$	<i>True</i>
$x > 5$	<i>False</i>
$x \leq 5$	<i>True</i>
$5 == x$	<i>False</i>
$x \neq 5$	<i>True</i>
$(3 \neq 4) \text{ and } (7 < 5)$	<i>False</i>
$(4 > 4) \text{ or } (5 \leq 10)$	<i>True</i>

ตัวอย่างการใช้นิพจน์ทางตรรกศาสตร์

- ใช้ตรวจสอบคำตอบของสมการ $x^2 + 9x + 10 = 0$

$$x * x + 9 * x + 10 == 0$$

ผลลัพธ์จะเป็นจริงหาก x เป็นคำตอบของสมการ

- ใช้ตรวจสอบว่าค่าในตัวแปร y เป็นเลขคู่หรือไม่

$$y \% 2 == 0$$

หรือ

$$y \% 2 != 1$$

ผลลัพธ์จะเป็นจริงหากค่าในตัวแปร y เป็นเลขคู่

คำสั่งเงื่อนไข (if statement)



ที่มาของภาพ <http://www.ryt9.com/s/prg/774090>

คำสั่งเงื่อนไข (if statement)

- คำสั่ง if จะคำนวณค่าของนิพจน์ตรรกศาสตร์ที่เป็น **เงื่อนไข** (condition) เพื่อตัดสินใจว่าจะโปรแกรมทำงานในคำสั่งที่คำสั่ง if ควบคุมอยู่หรือไม่
- คำสั่ง if ควบคุมจะอยู่ใน**บล็อก** (block) และคำสั่งนี้จะถูกทำงานเมื่อเงื่อนไขมีค่าเป็น True

ตัวอย่างคำสั่งเงื่อนไข (if statement)



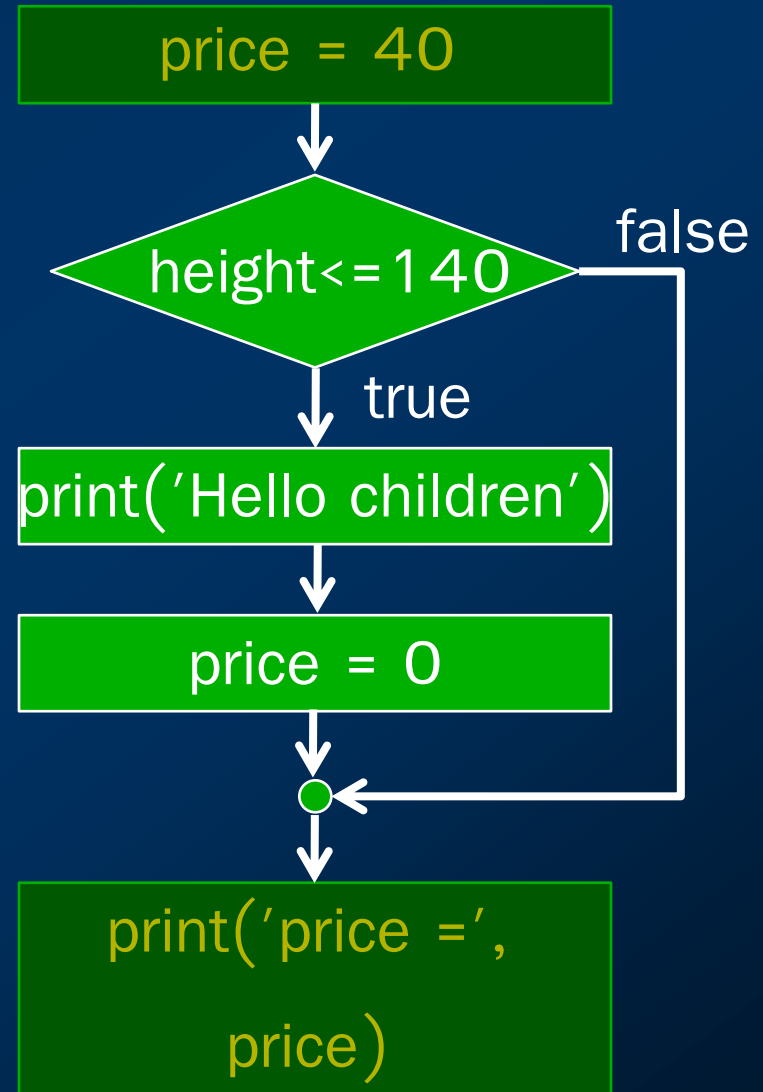
```
price = 40
```

```
if เงื่อนไขควรเป็นอะไร? :
```

```
    print('Hello Children!')
```

```
    price = 0
```

```
print('price =', price)
```



คำสั่งเงื่อนไข (if statement)

- รูปแบบการใช้งาน

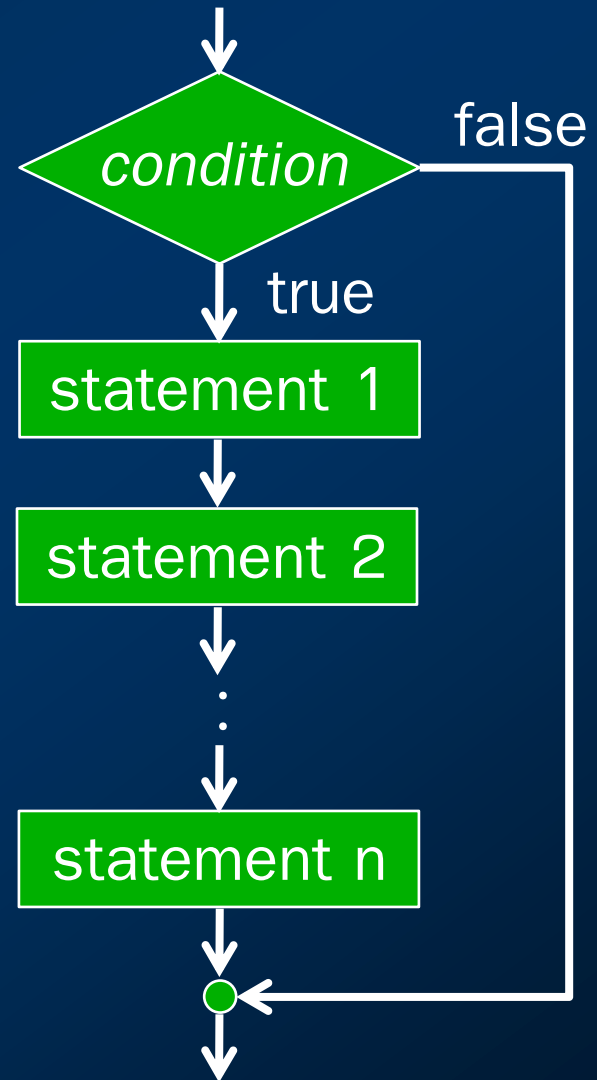
if *condition*:

statement 1

statement 2

:

statement n



การควบคุมการไหลของโปรแกรม

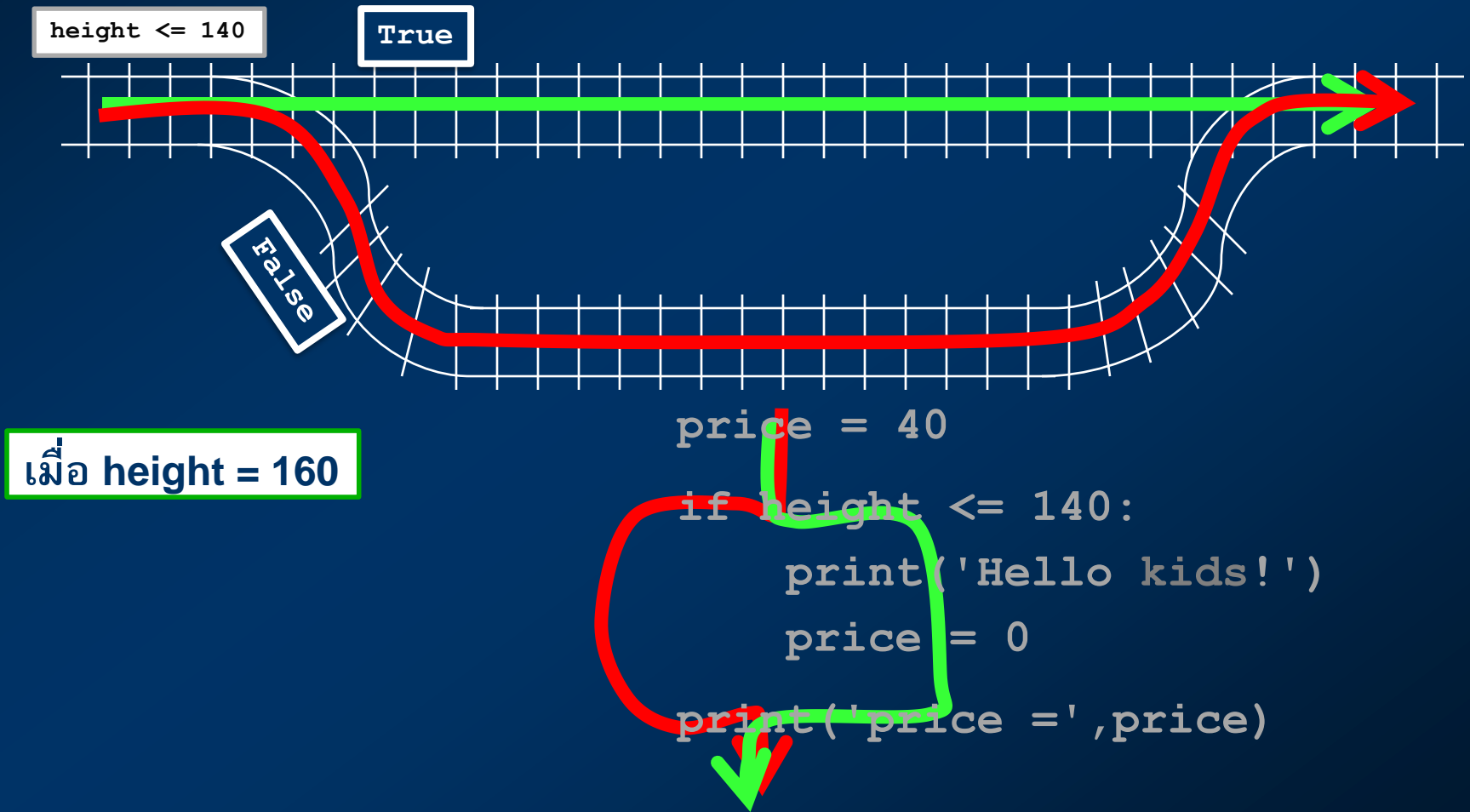
- โปรแกรมทั่วไปจะทำงานเป็นเส้นตรง



```
x = int(input())  
y = int(input())  
print(x+y)  
print("Hello",x)  
z = x * y + 10  
print(z)
```

การควบคุมการไหลของโปรแกรม

- โปรแกรมที่ใช้คำสั่ง if



บล็อก

```
price = 40
if height <= 140:
    print('Hello kids!')
    price = 0
print('price =', price)
```

- คำสั่งที่ย่อหน้าเข้าไปเท่า ๆ กัน จะถูกรวมกันเป็นกลุ่มเรียกว่า**บล็อก (block)**
- เงื่อนไขที่ระบุในคำสั่ง if จะตัดสินใจว่าคำสั่งในบล็อกดังกล่าวจะถูกทำงานหรือไม่

ข้อควรระวัง

- แนวคิดเกี่ยวกับบล็อกเป็นเรื่องสำคัญมากในภาษาไพธอน
- ดังนั้นในการเขียนโปรแกรมควรต้องระวังในการย่อหน้า
 - แต่เมื่อทำไปบ่อย ๆ ก็จะไม่ค่อยได้ไม่ยาก

```
Fdskfjsdlkfslkdjfdsff
fdskfsdflksdlkfdsf:
    fddslfldskf

    fdsfkdsfdsfd

    fdkfddfdfd
    fdkfdlf

    fdslkdskslkdjsld
```

Good

```
Fdskfjsdlkfslkdjfdsff
fdskfsdflksdlkfdsf:
    fddslfldskf

    fdsfkdsfdsfd

    fdkfddfdfd
    fdkfdlf

    fdslkdskslkdjsld
```

Bad

คำสั่ง `pass` สำหรับบล็อกว่าง

- ในไพธอน เราไม่สามารถสร้างบล็อกที่ไม่มีคำสั่งได้

```
if height <= 140:  
    print("I'm here")
```



- ให้ใช้คำสั่ง **`pass`** ในบล็อกดังกล่าวเพื่อระบุว่าไม่ทำอะไรแทน

```
if height <= 140:  
    pass  
    print("I'm here")
```

บล็อกสามารถซ้อนกันได้

```
x = int(input())
if x > 5:
    print("hello")

    if x < 10:
        print("foofoo")
        print("barbar")

    print("well")

print("cheers")
```

- ให้ทายผลลัพธ์ของโปรแกรมดังกล่าวเมื่อค่าของตัวแปร x เป็นดังต่อไปนี้
 - 3
 - 5
 - 7
 - 9
 - 10

สรุปย่อ

- บล็อกที่สร้างโดยการย่อหน้าเป็นลักษณะพิเศษในภาษาไพธอน
 - เป็นสาเหตุหนึ่งที่ทำให้หลายคนชื่นชอบไพธอน
- อย่าลืมระวังการย่อหน้าและการทำงานบล็อก

มูมนักคิด ตอนที่ 0

- ให้ตัวแปร **x** และ **y** เขียนโปรแกรมให้สลับค่าตัวแปรทั้งสองได้อย่างไร?
 - ตัวอย่างเช่น ถ้า $x = 10$ และ $y = 25$ หลังจากโปรแกรมทำงานแล้ว ให้ $x = 25$ และ $y = 10$

x = y

y = x

y = x

x = y

วิธีนี้ใช้ไม่ได้เช่นกัน เพราะอะไร?

วิธีนี้ใช้ไม่ได้ เพราะว่าอะไร?

t = x

x = y

y = t

มูมนักคิด ตอนที่ 1

- ร้านขายลูกอมร้านหนึ่ง มีเงื่อนไขในการขายว่าในการซื้อลูกอมครั้งหนึ่ง ๆ จะต้องซื้อลูกอมอย่างน้อย s ลูก
- ให้ตัวแปร x แทนจำนวนลูกอมที่ต้องการซื้อ เขียนโปรแกรมที่ปรับค่าตัวแปร x ให้เท่ากับจำนวนลูกอมที่คุณต้องซื้อ ตัวอย่างเช่น
 - ถ้า $s = 5$ และ $x = 7$ เมื่อโปรแกรมทำงานแล้ว จะได้ $x = 7$
 - ถ้า $s = 10$ และ $x = 5$ เมื่อโปรแกรมทำงานแล้ว จะได้ $x = 10$

มุมมองนักคิด ตอนที่ 1 (เฉลย)

```
if x < s:  
    x = s
```

มุมนักคิด ตอนที่ 1 (ฟังก์ชัน)

```
def num_candies(want, limit):  
    if want < limit:  
        return limit  
    return want
```

```
limit = 20  
x = int(input("How many?"))  
print("You should buy",  
      num_candies(x, limit),  
      "candies.")
```

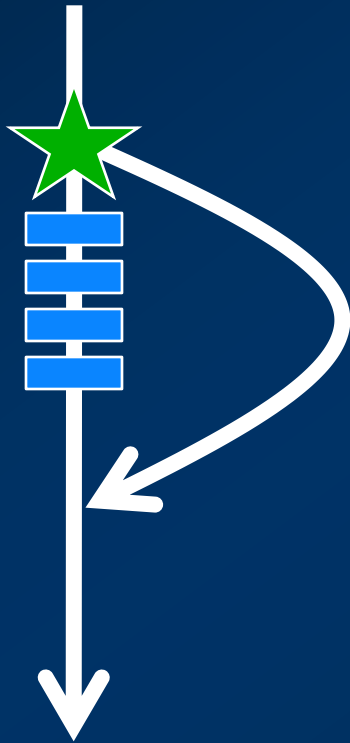

if – else statements



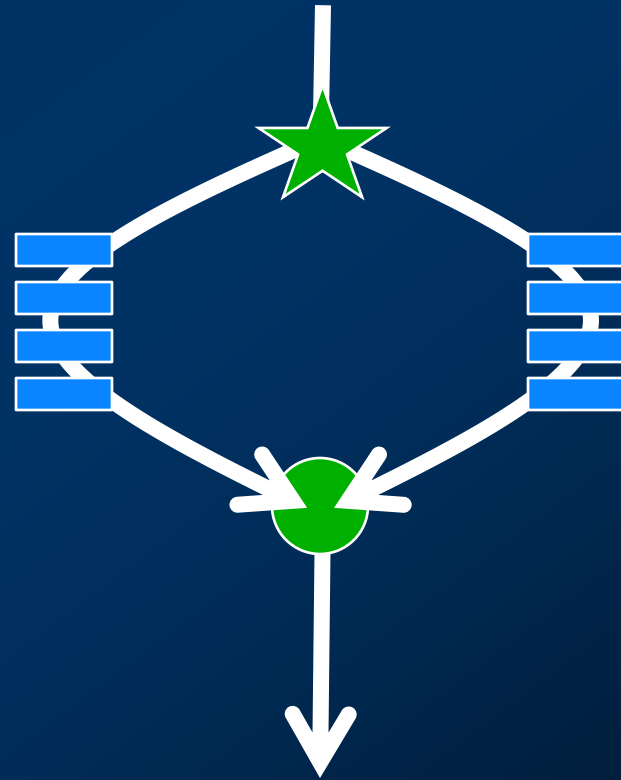
ที่มาของภาพ <http://splinedoctors.com/2009/02/hurry-up-and-choose/>

คำสั่ง if-else

- คำสั่ง if



- คำสั่ง if-else



if – else statements

- เมื่อเงื่อนไข (condition) เป็นจริงจะทำคำสั่ง (statement) ในกลุ่ม T
- เมื่อเงื่อนไข (condition) เป็นเท็จจะทำคำสั่ง (statement) ในกลุ่ม F

- รูปแบบการใช้งาน

if *condition*:

statement T1

statement T2

:

statement Tn

else:

statement F1

statement F2

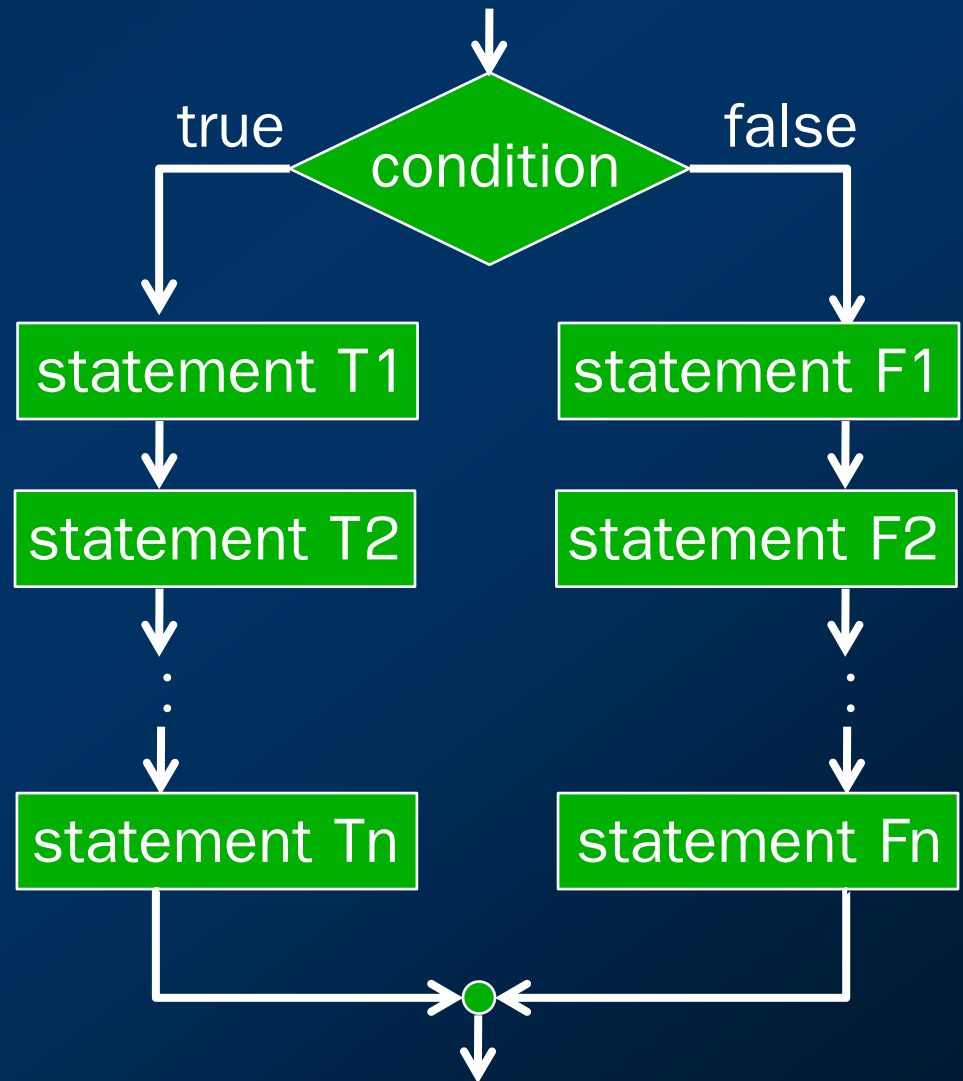
:

statement Fn

- รูปแบบการใช้งาน

```
if (condition) :  
    statement T1  
    statement T2  
    :  
    statement Tn  
else :  
    statement F1  
    statement F2  
    :  
    statement Fn
```

if – else statements



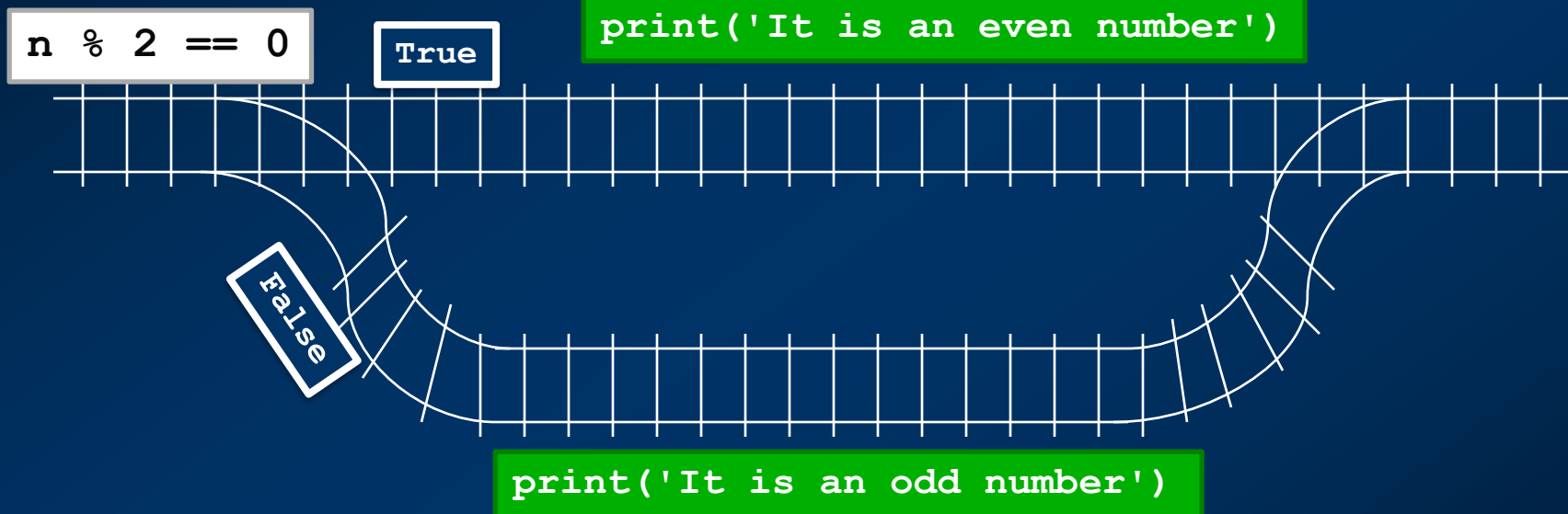
ตัวอย่าง if – else statements

- เขียนส่วนของโปรแกรมที่ตรวจสอบเลขจำนวนเต็มที่เก็บในตัวแปร n ว่าเป็นจำนวนเต็มคู่หรือจำนวนเต็มคี่

Value in N	Output
Even Number	It is an even number.
Odd Number	It is an odd number.

```
if n%2 == 0:  
    print('It is an even number')  
else:  
    print('It is an odd number')
```

การควบคุมการไหลของโปรแกรม



มุมมองนักคิด ตอนที่ 2

- จงเขียนส่วนของโปรแกรมที่ใช้ในตรวจสอบคะแนนของนักเรียนว่าผ่านการทดสอบหรือไม่

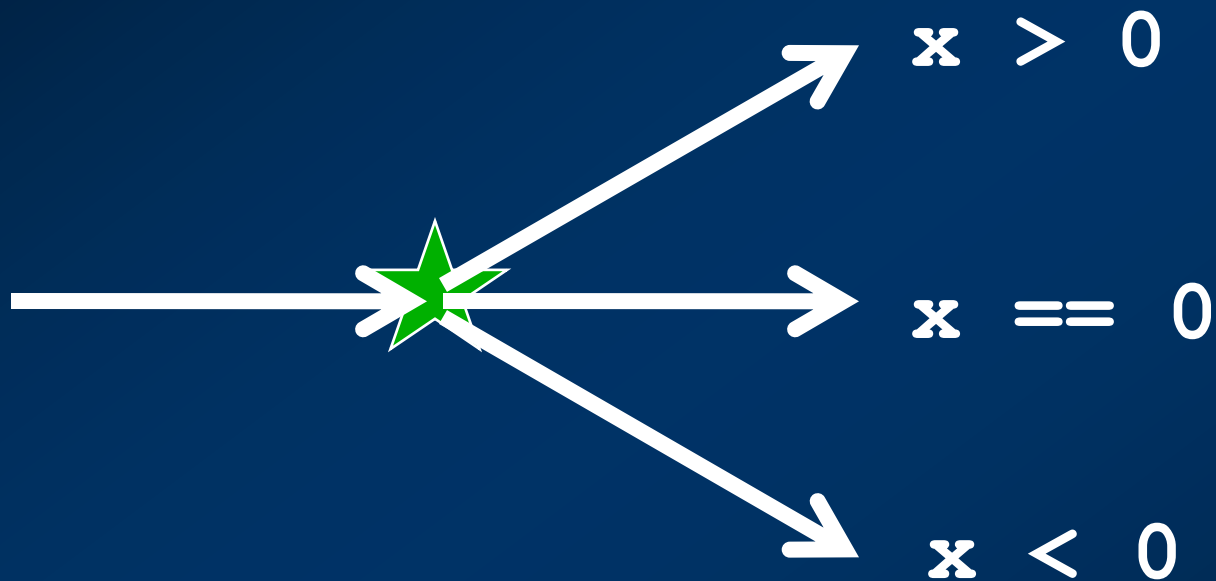
score	Output
Less than 50	You failed.
Other	You passed.



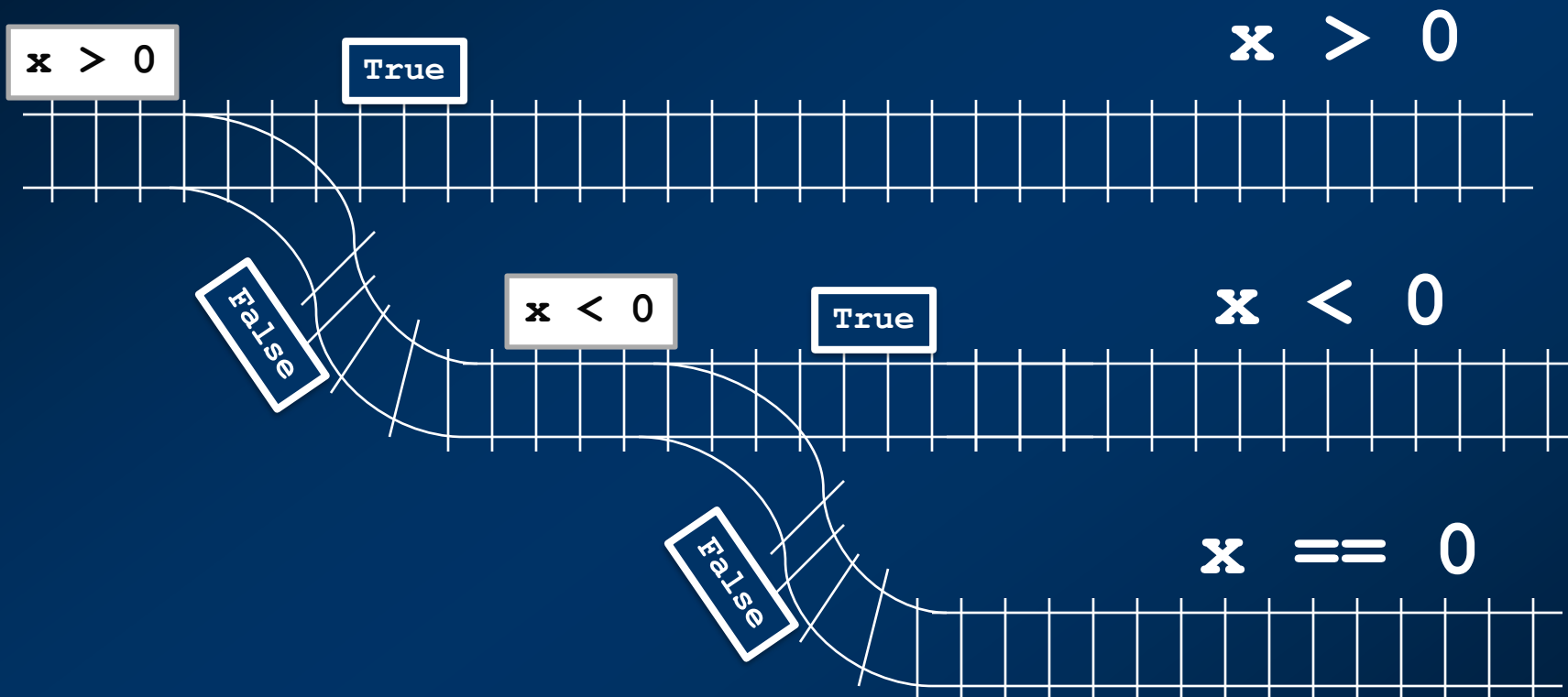
มูมนักคิด ตอนที่ 3

- เขียนโปรแกรมที่อ่านจำนวนเต็มหนึ่งจำนวนแล้วระบุว่าจำนวนเต็มดังกล่าวเป็นจำนวนเต็มบวก จำนวนเต็มลบ หรือศูนย์

มูมนักคิด ตอนที่ 3



มุมมองคิด ตอนที่ 3



มุมมองนักคิด ตอนที่ 3

```
x = int(input("Enter an integer:"))
if x > 0:
    print("A positive integer")
else:
    if x < 0:
        print("A negative integer")
    else:
        print("A zero")
```

- สังเกตว่าเราใช้คำสั่ง if ภายในบล็อกของคำสั่ง if อีกคำสั่งหนึ่ง

คำสั่ง if ที่ซ้อนกัน

- คำสั่ง if ก็เป็นรูปแบบหนึ่งของคำสั่งเช่นกัน ดังนั้นบล็อกที่สามารถใส่คำสั่งอะไรก็ได้ ก็ย่อมจะมีคำสั่ง if อยู่ในนั้นได้ด้วยเช่นกัน
- เรียกว่าคำสั่งเงื่อนไขซ้อนกัน (nested if)

ตัวอย่าง: ผลการสอบ (if)

- คุณได้รับคะแนนการสอบ
 - ถ้าคะแนนมากกว่า 8 จะถือว่าได้คะแนนดี
 - ถ้าคะแนนมากกว่า 4 แต่น้อยกว่าเท่ากับ 8 คุณจะสอบผ่าน
 - ถ้าคะแนนน้อยกว่าหรือเท่ากับ 4, ถือว่าสอบตก

```
if score > 8:  
    print("Good")  
if score > 4 and score <= 8:  
    print("Passed")  
If score <= 4:  
    print("Failed")
```

ตัวอย่าง: ผลการสอบ (nested-if)

- คุณได้รับคะแนนการสอบ
 - ถ้าคะแนนมากกว่า 8 จะถือว่าได้คะแนนดี
 - ถ้าคะแนนมากกว่า 4 แต่น้อยกว่าเท่ากับ 8 คุณจะสอบผ่าน
 - ถ้าคะแนนน้อยกว่าหรือเท่ากับ 4, ถือว่าสอบตก

```
if score > 8:  
    print("Good")  
else:  
    if score > 4:  
        print("Passed")  
    else:  
        print("Failed")
```

ตัวอย่าง: ผลการสอบ (nested-if)

```
if score > 8:  
    print("Good")  
else:  
    if score > 4:  
        print("Passed")  
    else:  
        print("Failed")
```

คุณจะทำหน้าที่บรรทัดนี้เมื่อ?

score > 8


score <= 8 and score > 4

score <= 8 and score <= 4

score <= 4

ตัวอย่าง: ผลการสอบ (คำสั่ง elif)

```
if score > 8:  
    print("Good")  
else:  
    if score > 4:  
        print("Passed")  
    else:  
        print("Failed")
```



```
if score > 8:  
    print("Good")  
elif score > 4:  
    print("Passed")  
else:  
    print("Failed")
```

- โปรแกรมสามารถทำให้อ่านง่ายขึ้นได้ ถ้าใช้คำสั่ง **elif**

if – elif – else statements



ที่มาของภาพ <http://www.flickr.com/photos/29104098@N00/285609610/>

if – elif – else statements

- if – elif – else ใช้ในกรณีที่ต้องการแบ่งเงื่อนไขในการเลือกทำคำสั่งมากกว่าหนึ่งเงื่อนไข

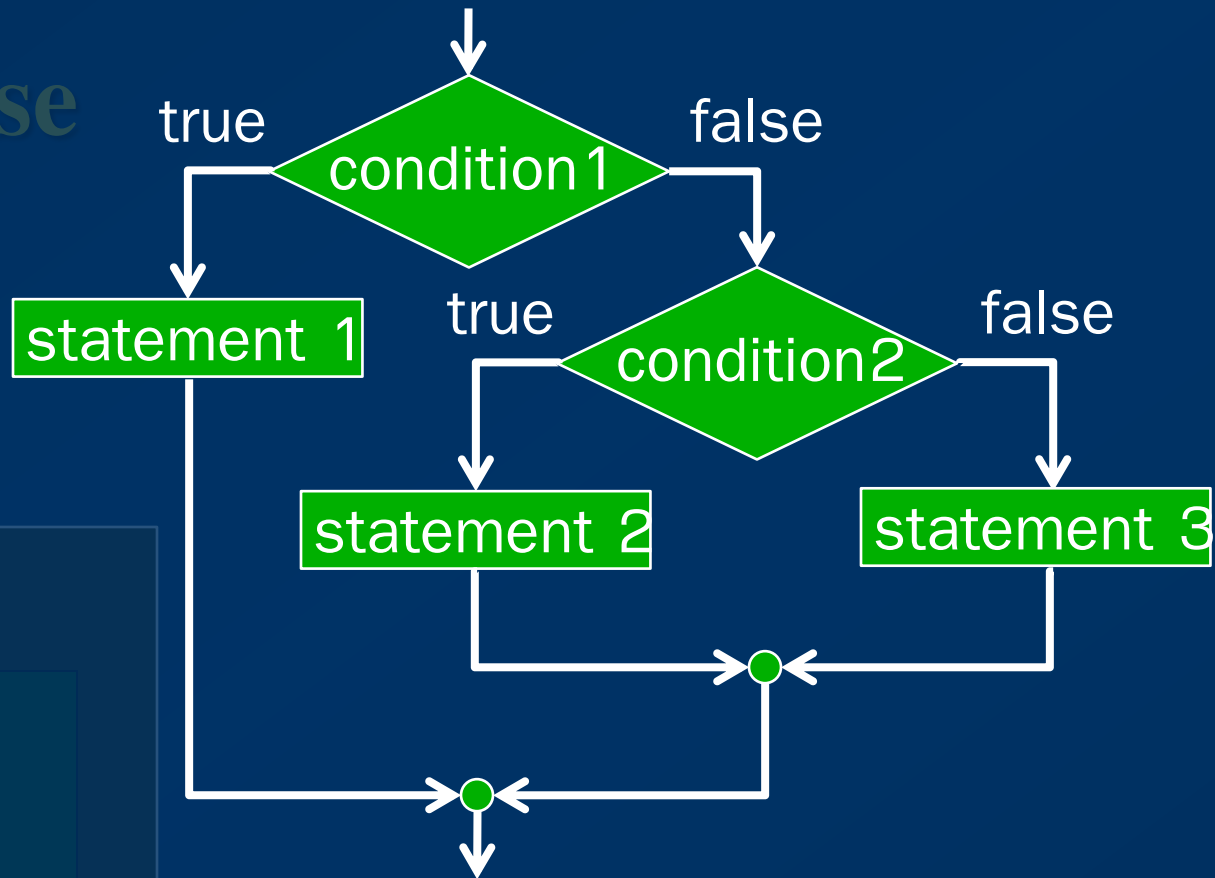
- รูปแบบการใช้งาน

```
if condition1:  
    statement 1  
elif condition2:  
    statement 2  
elif condition3:  
    statement 3  
    :      :      :  
else:  
    statement n
```

if – elif – else statements

- รูปแบบการใช้งาน

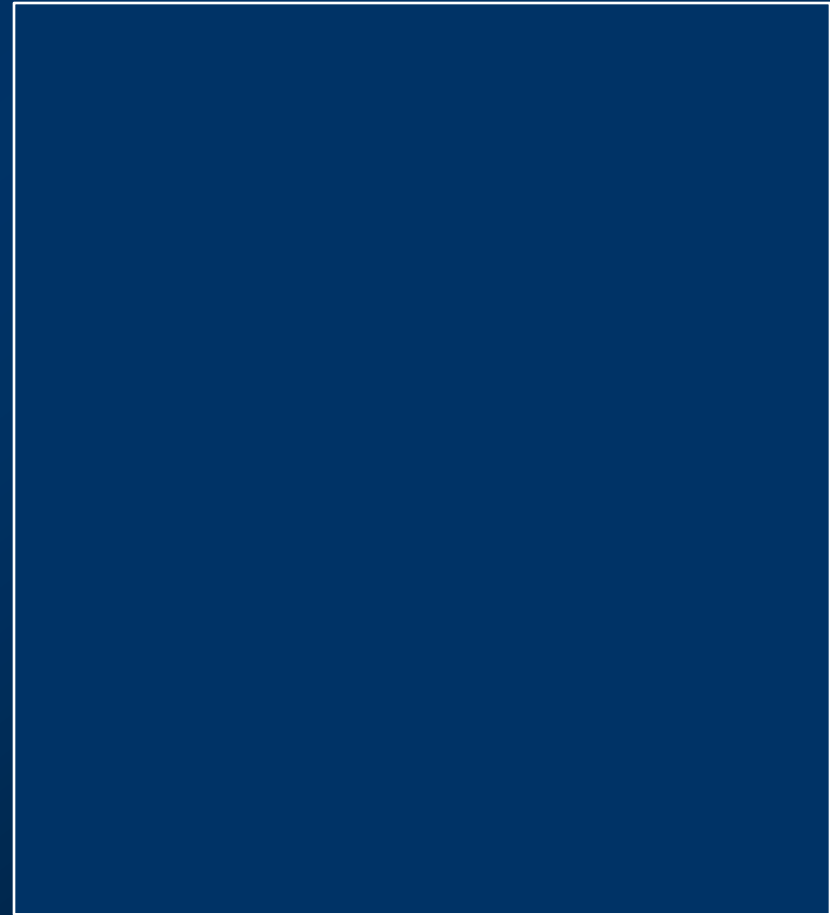
```
if (condition1) :  
    statement 1  
elif (condition2) :  
    statement 2  
else :  
    statement 3
```



มูมนักคิด ตอนที่ 4

- จงเขียนส่วนของโปรแกรมที่หาค่าของฟังก์ชันที่กำหนดให้ต่อไปนี้

$$f(x) = \begin{cases} 2x+10, & x \leq 5 \\ x^2+10, & 5 < x \leq 20 \\ x-10, & 20 < x < 30 \\ 3x, & x \geq 30 \end{cases}$$



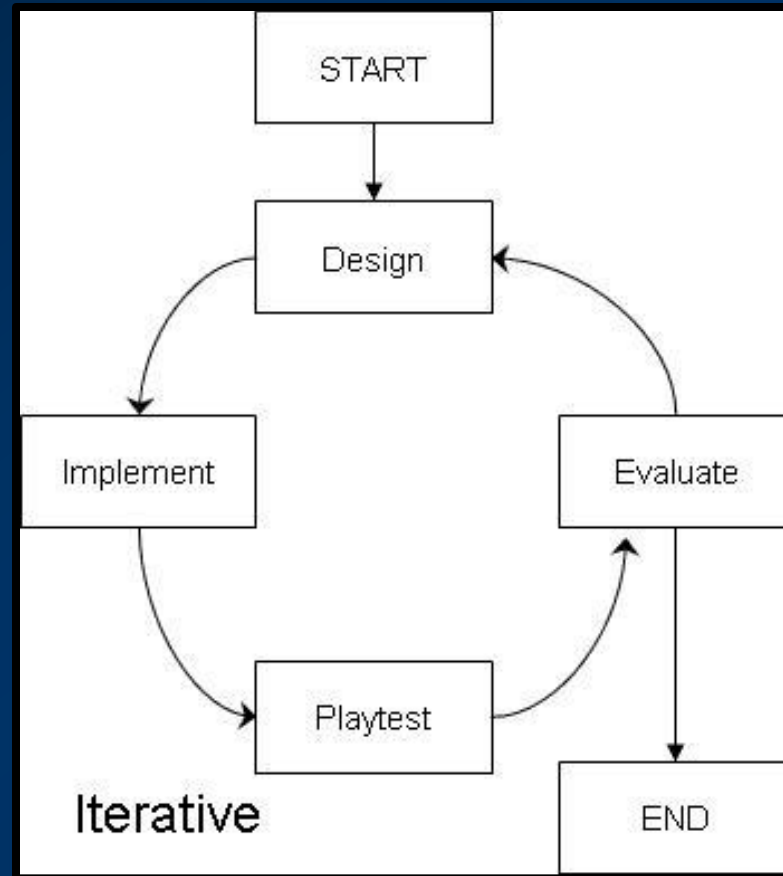
คำสั่งเงื่อนไขแบบซ้อน (nested if)

- เราสามารถเขียนคำสั่ง if ซ้อนอยู่ในคำสั่ง if อีกก็ได้

ตัวอย่างโปรแกรมตรวจสอบเลขบวก เลขลบ หรือศูนย์
แบบใช้ if – elif – else

```
if (n > 0) :  
    print 'It is positive number'  
elif (n < 0):  
    print 'It is negative number'  
else :  
    print 'It is zero number'
```

คำสั่งวนซ้ำ



ที่มาของภาพ <http://gamedesignconcepts.wordpress.com/2009/07/02/level-2-game-design-iteration-and-rapid-prototyping/>

งานซ้ำซ้ำ

- คอมพิวเตอร์สามารถทำงานซ้ำ ๆ กันได้อย่างมีประสิทธิภาพ
- แต่เราจะเขียนโปรแกรมให้คอมพิวเตอร์ทำงานเช่นนั้นได้อย่างไร?

คำสั่งวนซ้ำ

- โปรแกรมนี้ให้ผลลัพธ์เป็นเช่นไร

```
print('I like Bossanova')  
print('I like Bossanova')  
print('I like Bossanova')  
print('I like Bossanova')  
print('I like Bossanova')  
print('I like Bossanova')  
print('I like Bossanova')  
print('I like Bossanova')
```

จะทำเช่นไรหาก
ต้องการให้
โปรแกรมพิมพ์
ข้อความ I like
Bossanova ทั้งหมด
2553 ครั้ง ?

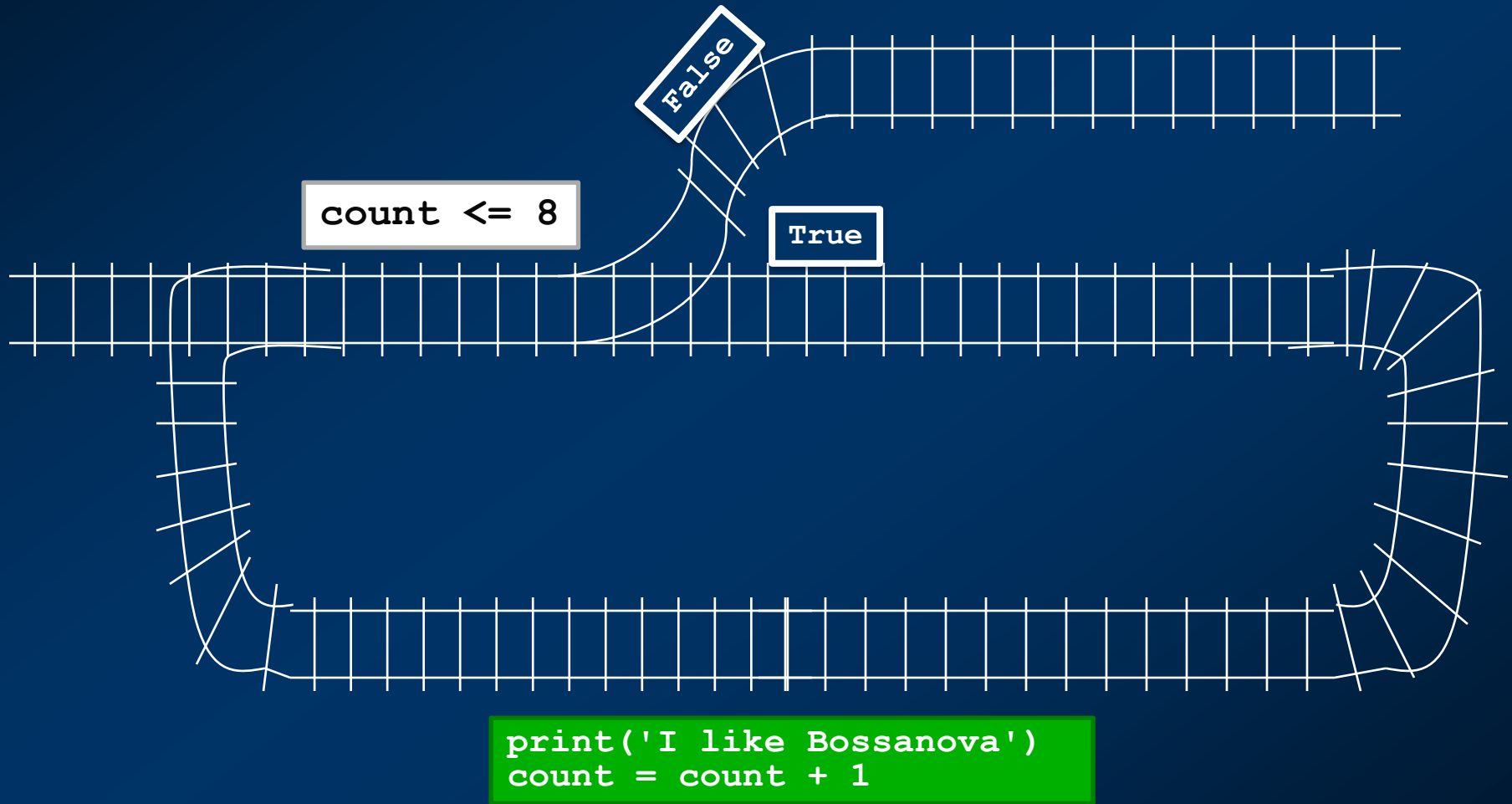
คำสั่งวนซ้ำ

- โปรแกรมเมื่อเปลี่ยนมาใช้คำสั่งวนซ้ำ

```
count = 1
while count <= 8:
    print('I like Bossanova')
    count = count + 1
```

จะทำเช่นไรหาก
ต้องการให้
โปรแกรมพิมพ์
ข้อความ I like
Bossanova ทั้งหมด
2553 ครั้ง ?

โปรแกรมทำงานได้อย่างไร?



คำสั่ง while

- โปรแกรมจะพิจารณาเงื่อนไข condition และจะทำคำสั่งในบล็อก วนซ้ำเท่าที่เงื่อนไขยังเป็นจริง

- รูปแบบการใช้งาน

while condition:

statement 1

statement 2

:

statement n

คำสั่ง while

- รูปแบบการใช้งาน

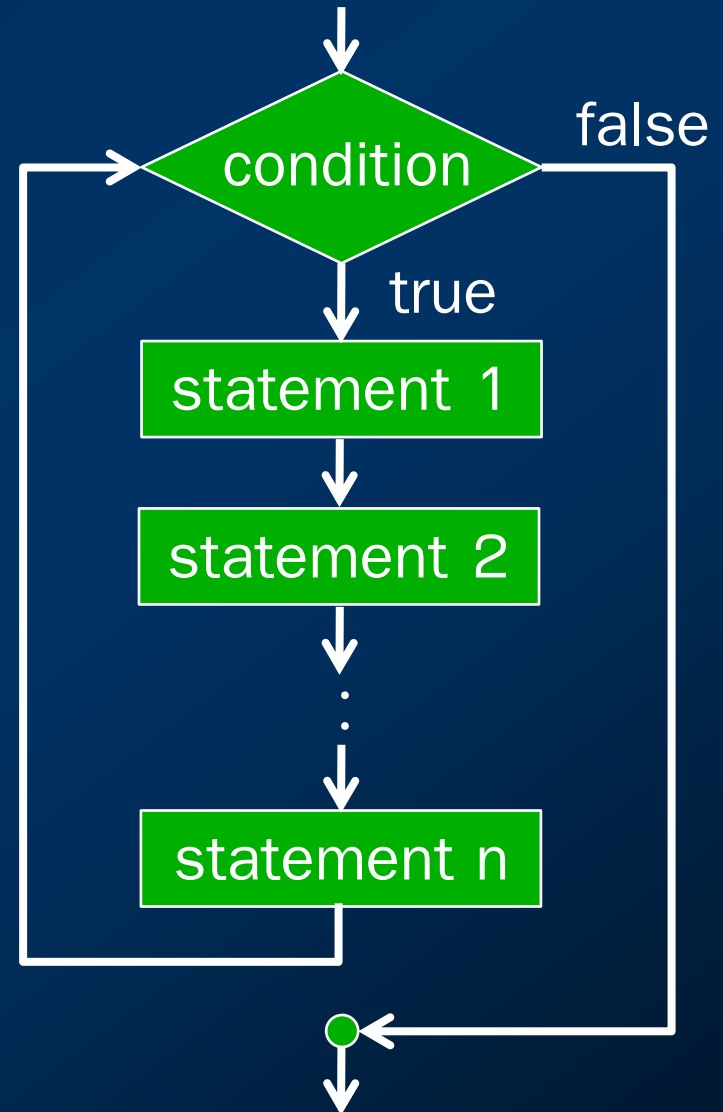
while condition:

statement 1

statement 2

:

statement n



ตัวอย่างคำสั่งวนซ้ำ

- โปรแกรมที่พิมพ์เลขจำนวนเต็มทีละบรรทัดตั้งแต่ 1 ถึง 100

```
n = 1  
while n <= 100:  
    print(n)  
    n = n + 1
```

มุมมองนักคิด ตอนที่ 5

- จงเขียนส่วนของโปรแกรมที่พิมพ์เลขคู่ที่ละบรรทัดตั้งแต่ 100 ถึง 2

มูมนักคิด ตอนที่ 6

- เขียนฟังก์ชัน `sum_to(n)` ที่คำนวณและคืนค่า $1 + 2 + \dots + n$.
- ห้ามใช้สูตร $n*(n+1)/2$.
- เราสามารถเริ่มต้นจากเขียนโปรแกรมที่คำนวณค่าดังกล่าว แล้วค่อยเปลี่ยนให้เป็นฟังก์ชัน

มุมมองนักคิด ตอนที่ 6 (คำใบ้)

- คุณต้องการบวก

- 1,

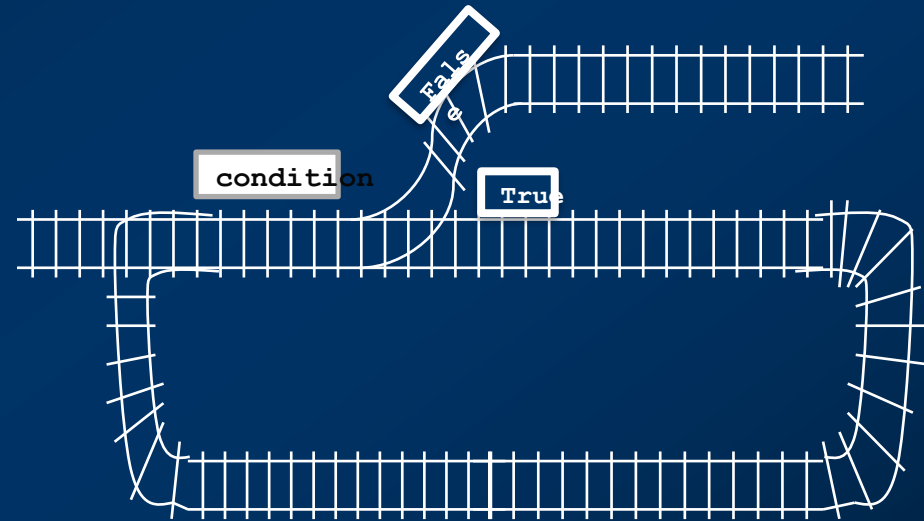
- 2,

- จนถึง n

- ในแต่ละรอบ:

- คุณต้องการเก็บค่าอะไร?

- อะไรคือสิ่งที่เปลี่ยนไปในแต่ละรอบ?



มุมมองนักคิด ตอนที่ 6 (คำใบ้ข้อ)

- พิจารณาค่า $1, 2, \dots, n$
- เราจะใช้ตัวแปร i เพื่อเก็บค่าเหล่านี้
- เขียนโปรแกรมที่ทำงานแค่เปลี่ยนค่าตัวแปร i ให้มีค่าเป็นไปตามรายการดังกล่าว

```
i = 1  
  
while i <= n :  
    i += 1
```

มุมมองนักคิด 6 (คำใบ้อีกอีกอีก)

- คุณต้องการทำอะไรกับตัวแปร i
 - หาผลรวม
- ดังนั้นเราจำเป็นต้องหาตัวแปรเพื่อเก็บค่านั้น

มุมมองนักคิด 6: โปรแกรม

```
n = int(input())
total = 0
i = 1

while i <= n:
    total = total + i
    i = i + 1

print(total)
```

มุมมองนักคิด 6: ฟังก์ชัน

```
def sum_to(n):  
    total = 0  
    i = 1  
  
    while i <= n:  
        total = total + i  
        i = i + 1  
  
    return total
```

มูมนักคิด 7: รหัสผ่าน

- เขียนโปรแกรมที่ให้ผู้ใช้อ้อนรหัสผ่าน จนกระทั่งผู้ใช้อ้อนรหัสผ่านที่ถูกต้อง
 - ให้ใช้รหัสผ่านเป็น 'happy204111'.

```
Enter password: sad111
Sorry.
Enter password: happy111
Sorry.
Enter password: happy204111
Correct.
```

มุมมองนักคิด 7: คำใบ้

- ตอบคำถามต่อไปนี้:
 - เจื่อนไขใดที่ทำให้โปรแกรมยังทำงานช้า ๆ อยู่?
 - ในแต่ละรอบคุณต้องการทำอะไร?
 - คุณต้องทำอะไร ก่อนที่จะสามารถตรวจสอบเจื่อนไขในครั้งแรกได้?

มูมนักคิด 7: คำใบ้

```
pwd = input("Enter password: ")  
while _____:  
    print("Sorry.")  
    _____  
print("Correct.")
```

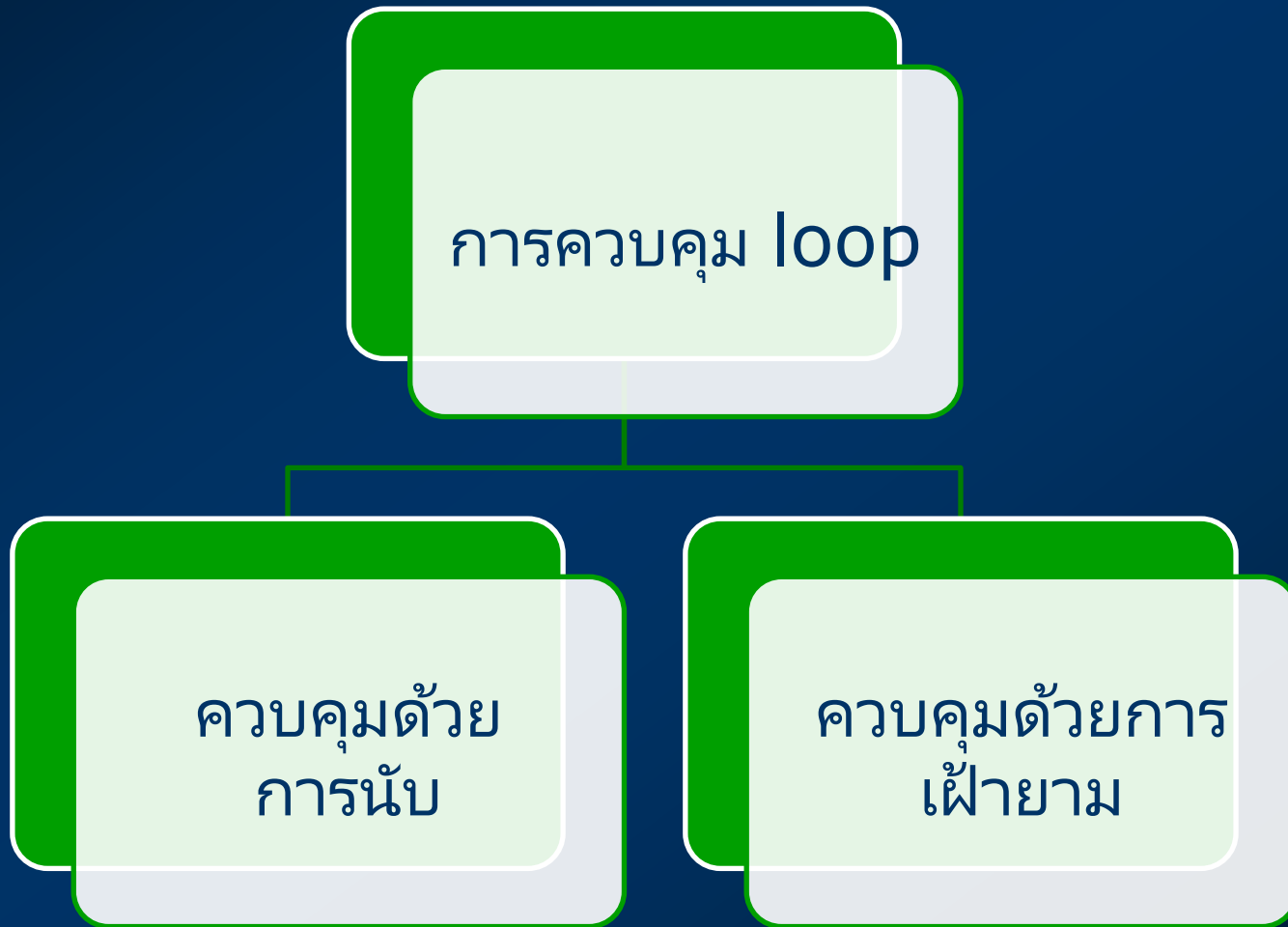
มูมนักคิด 7: เฉลย

```
pwd = input("Enter password: ")  
while pwd != 'happy204111' :  
    print("Sorry.")  
    pwd = input("Enter password: ")  
print("Correct.")
```


การควบคุมการทำซ้ำ

- โดยทั่วไป วิธีการควบคุมการทำซ้ำที่ใช้บ่อย ๆ จะสามารถแบ่งได้เป็นสองประเภท
 - การควบคุมด้วยการนับ
 - การควบคุมโดยการเฝ้ายาม
- **หมายเหตุ** ยังมีวิธีการอื่น ๆ ในการควบคุมการทำซ้ำอีกที่ไม่ได้กล่าวถึงในที่นี้

การควบคุม loop

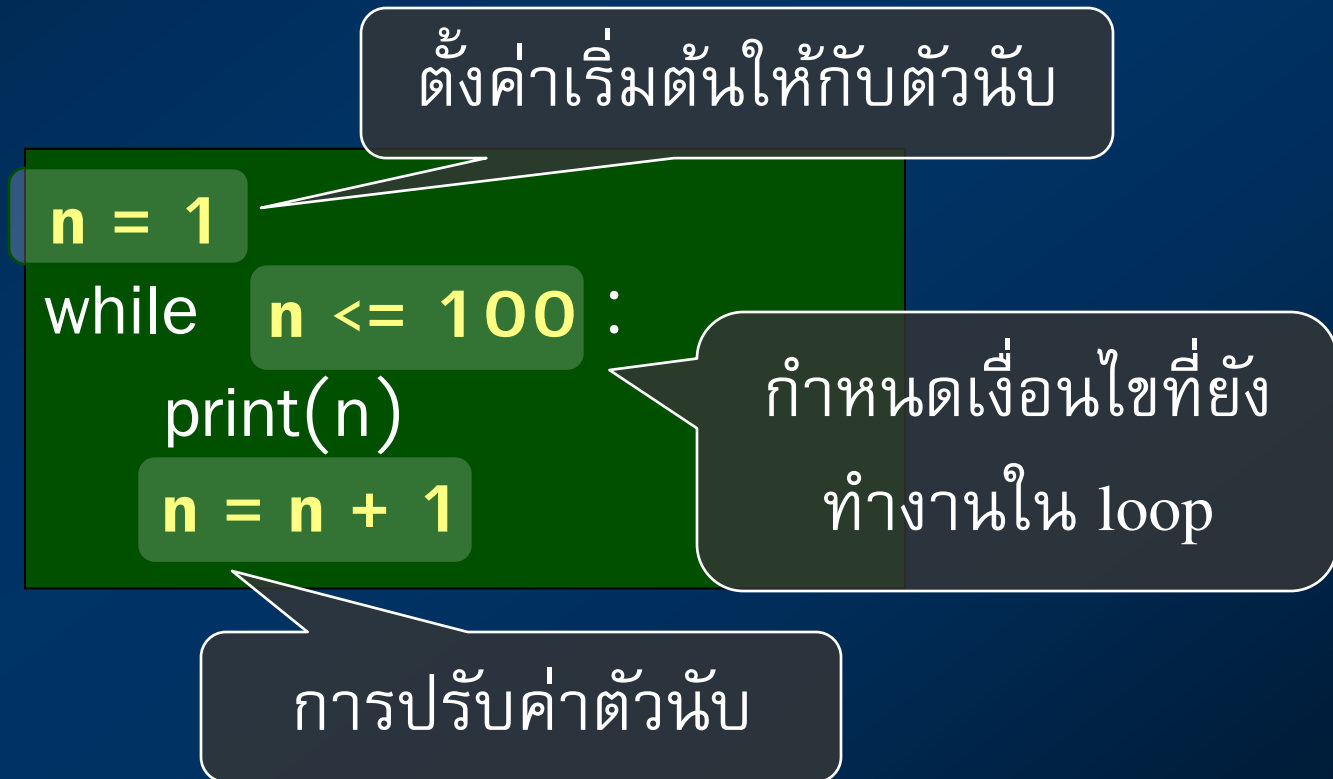


การควบคุม loop ด้วยการนับ

- ใช้วิธีการนับไปเรื่อย ๆ ตราบเท่าที่เงื่อนไขเป็นจริง
- ควรประกอบด้วยสามสิ่งคือ
 - ตั้งค่าเริ่มต้นให้กับตัวนับ
 - กำหนดเงื่อนไขการออกจาก loop
 - การปรับค่าตัวนับ

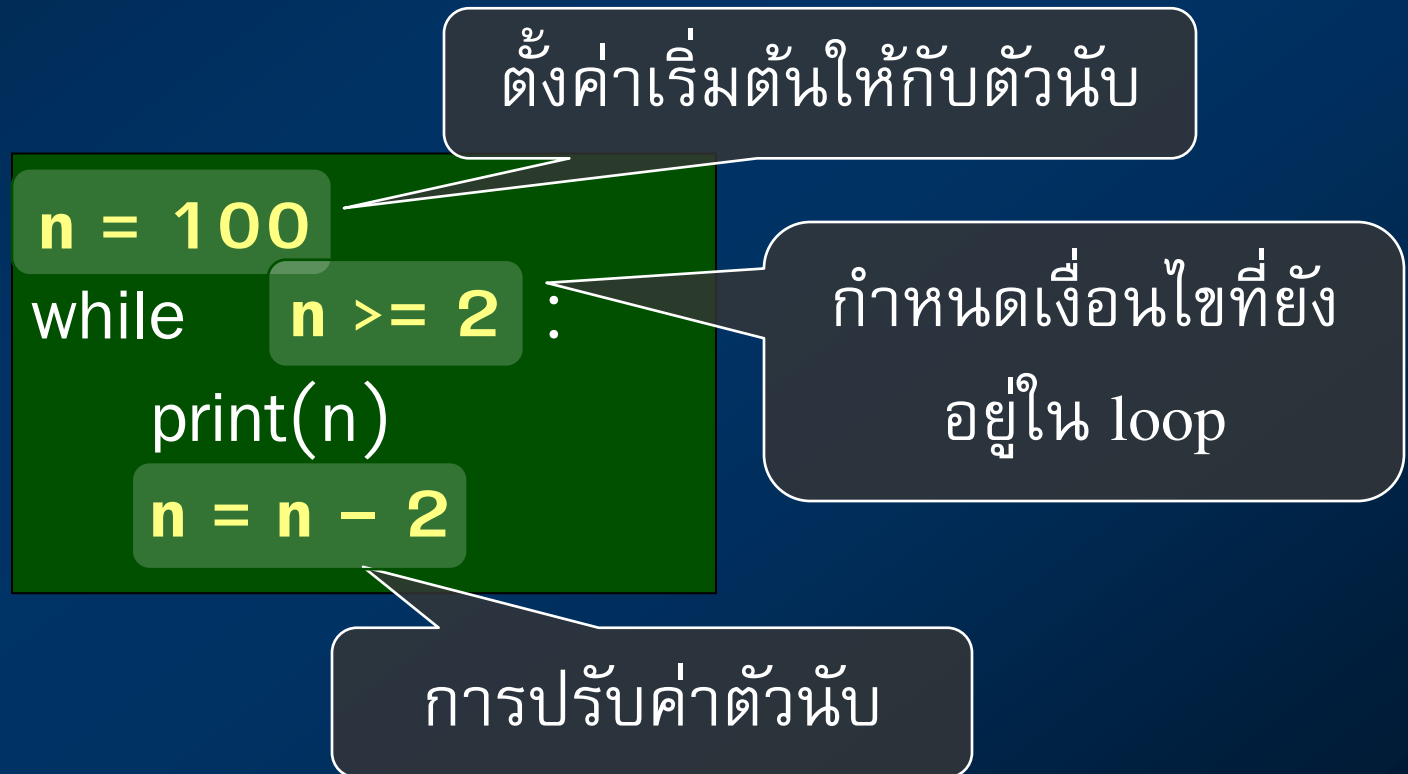
ตัวอย่างคำสั่งวนซ้ำ

- โปรแกรมที่พิมพ์เลขจำนวนเต็มทีละบรรทัดตั้งแต่ 1 ถึง 100



โปรแกรม จากมุมมองนักคิด ตอนที่ 4

- จงเขียนส่วนของโปรแกรมที่พิมพ์เลขคู่ที่ละบรรทัดตั้งแต่ 100 ถึง 2



โปรแกรม จากมุมมองนักคิด ตอนที่ 4

- จงเขียนส่วนของโปรแกรมที่พิมพ์เลขคู่ที่ละบรรทัดตั้งแต่ 100 ถึง 2

```
n = 100  
while ( n >= 2 ) :  
    print(n)  
    n = n - 2
```

จะเกิดอะไรขึ้นหาก
เปลี่ยนเครื่องหมาย
จาก \geq เป็น \leq

จะเกิดอะไรขึ้นหาก
เปลี่ยนจาก $n = n - 2$
เป็น $n = n + 2$

การควบคุมการทำซ้ำโดยการ ฝ้ายาม

```
pwd = input("Enter password: ")  
while pwd != 'happy204111':  
    print("Sorry.")  
    pwd = input("Enter password: ")  
print("Correct.")
```

พิจารณาโครงสร้างของ
การทำซ้ำนี้ละเอียด

การทำซ้ำนี้รออะไร?

โปรแกรมจะไม่หยุดจนกว่า `pwd == 'happy204111'`

มูมนักคิด 9

```
total = 0
n = 0
while n >= 0:
    n = input('Input n : ')
    if n >= 0:
        total = total + n
print('total =", total)
```

โปรแกรมนี้รออะไร?

โปรแกรมจะไม่หยุดจนกว่า $n < 0$

มูมนักคิด ตอนที่ 10

- แฟกตอเรียลของ n (เขียนแทนด้วย $n!$) มีนิยามคือ

$$1 \times 2 \times \dots \times n,$$

หรือ 1 เมื่อ $n = 0$.

- เขียนฟังก์ชัน **fact(n)** ที่คำนวณ $n!$.
 - ในการเริ่มต้น สามารถละเว้นไม่พิจารณากรณีที่ $n = 0$ ได้