Lecture 12 Nested and Variable Loop การวนลูปแบบซ้อนกันและการสร้างตัวแปรระว่างการวน

13 ตุลาคม 2557 11:16

เราได้เห็นโค้ดโปรแกรมเกมเดาตัวเลขในบทที่ 2 ภายใต้ loop while เราใส่บางสิ่งที่เรียกว่า กลุ่มก้อนของโค้ด บล็อค (block) ตามตัวอย่างโค้ดที่เราเคยเห็นดังนี้

```
while guess != secret and tries < 6:
    guess = input("What's yer guess? ")
    if guess < secret:
        print "Too low, ye scurvy dog!"
    elif guess > secret:
        print "Too high, landlubber!"
    tries = tries + 1
```

while loop block

if block

elif block

ในกรอบสี่แดงแสดงบล็อคของกลุ่มโค้ดที่อยู่ภายใต้ while loop และในกรอบสีเทาแสดงบล็อคที่อยู่ภายใต้ if และ elif

Nested Loops ลูปที่อยู่ภายใต้ loop

จากโค้ดการเขียนตารางสูตรคูณแม่ 5

```
multiplier = 5
for i in range(1, 11):
    print(i, "x", multiplier, '=', i*multiplier)
```

```
>>>
1 x 5 = 5
2 x 5 = 10
3 x 5 = 15
4 x 5 = 20
5 x 5 = 25
6 x 5 = 30
7 x 5 = 35
8 x 5 = 40
9 x 5 = 45
10 x 5 = 50
>>>
```

แต่ถ้าเราต้องการแสดงสามแม่ 5, 6, 7 เราจะต้องอาศัยการวนลูปซ้อนลูป (nested loop) เพื่อที่จะเข้าถึงแต่ละรอบ 5, 6, 7 ในแต่ละรอบเราเรียกว่า ไอเทอร์เรชั่น (iteration)

```
for multiplier in range(5, 8):
    for i in range(1, 11):
        print(i, "x", multiplier, '=', i*multiplier)
    print()
```

ผลลัพท์ที่ได้จะพิมพ์ตารางวนสามตารางของแม่สูตรคูณ 5, 6 , 7

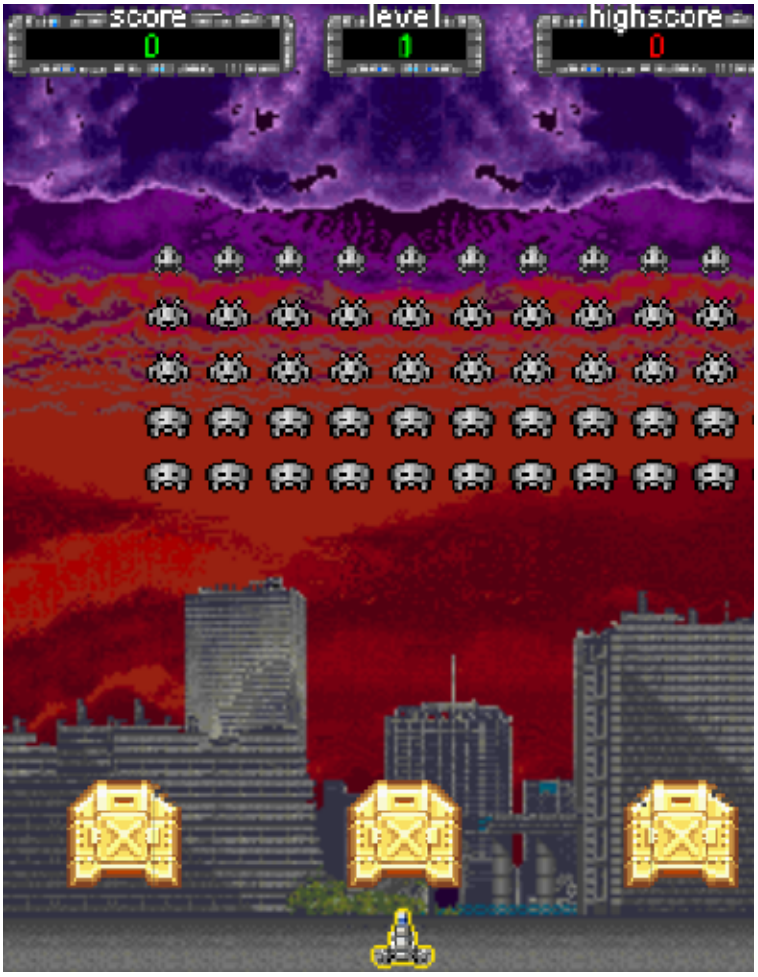
```
>>>
1 x 5 = 5
2 x 5 = 10
3 x 5 = 15
4 x 5 = 20
5 x 5 = 25
6 x 5 = 30
7 x 5 = 35
8 x 5 = 40
9 x 5 = 45
10 x 5 = 50

1 x 6 = 6
2 x 6 = 12
3 x 6 = 18
4 x 6 = 24
5 x 6 = 30
6 x 6 = 36
7 x 6 = 42
8 x 6 = 48
9 x 6 = 54
10 x 6 = 60

1 x 7 = 7
2 x 7 = 14
3 x 7 = 21
4 x 7 = 28
5 x 7 = 35
6 x 7 = 42
7 x 7 = 49
8 x 7 = 56
9 x 7 = 63
10 x 7 = 70
```

Variable loops ตัวแปรเกี่ยวกับการวน

จากตัวอย่างการสร้างตารางสูตรคูณ อาจจะทำให้เราไม่ตื่นเต้นมากนัก เนื่องจากการรันโปรแกรมแต่ละครั้งจะให้ผลลัพท์ที่เหมือนเดิม เนื่องจากเราใส่ค่าตัวเลขไว้คงที่ (constants) เราเรียกกว่าเขียนแบบนี้ว่า ฮาร์ดโค้ด Hard-coded หรือการเขียนแบบไม่มีการเปลี่ยนแปลง บางครั้งเราต้องการให้ผู้ใช้เป็นคนระบุจำนวนเอง เราจะต้องสร้างตัวแปร (variable) เพื่อเก็บค่า
ตัวอย่างเช่น การเขียนเกมส์ Space invader (http://www.agame.com/game/space-invaders-1) ซึ่งเป็นเกมส์ยิงเอเลี่ยนที่มาบุกรุกเป็นแถว เมื่อยิงเอเลี่ยนได้ จะค่อยๆ หายไปทีล่ะตัว



โดยเราจะทดลองเขียนโปรแกรมสร้างแถวของเอเลี่ยน โดยให้ หนึ่งดอกจันทร์ "*" แทนหนึ่งตัว โดยให้ผู้ใช้ระบุจำนวนเอเลี่ยนได้ โดยผลลัพท์ที่คาดหวังคือ

```
>>>
How many stars do you want? 5
* * * * *
```

เราสามารถเขียนโค้ดได้ดังต่อไปนี้

```
numStars = int(input("How many stars do you want? "))
for i in range(1, numStars + 1):
    print("*", end=" ")
```

โดยทั่วไปแล้วการเขียนโปรแกรมจะนิยมเริ่มด้วย 0 ดังนี้

```
numStars = int(input("How many stars do you want? "))
for i in range(0, numStars):
    print("*", end=" ")
```

ซึ่งจะให้ผลลัพท์เหมือนกัน

Variable nested loops ตัวแปรเกี่ยวกับการวนซ้อนการวน

จากโค้ดตัวอย่างที่แล้วเราพิมพ์ * เพียงแค่บรรทัดเดียว ถ้าเราต้องการหลายแถว เราจำเป็นจะต้องสร้าง loop ซ้อนกัน และกำหนดตัวแปรเพิ่ม ดังโค้ดตัวอย่างต่อไปนี้

```
numLines = int(input("How many of stars do you want? "))
numStars = int(input("How many stars do you want? "))
for line in range(0, numLines):
    for star in range(0, numStars):
        print("*", end=" ")
    print()
```

ผลลัพท์ที่ได้

```
How many of stars do you want? 3
```

```
How many of stars do you want? 3
How many stars do you want? 5
* * * * *
* * * * *
* * * * *
```

สองบรรทัดแรกของโค้ด คือการรับค่าอินพุต เก็บค่าจำนวนบรรทัด numLines และ จำนวนสตาร์แต่ละบรรทัด numStars แล้วใช้สองตัวแปรนี้วนด้วย for โดยการเพิ่มในหนึ่งแถวเราจะมีค่าว่า end=" " เพื่อไม่ให้ขึ้นบรรทัดใหม่ แต่พอจบแถวจะเรียกคำสั่ง pirnt() เปล่าๆ เพื่อให้ขึ้นบรรทัดใหม่หลังจากครบจำนวนสตาร์

เราสามารถซ้อนการวนได้อีก ตัวอย่างเช่นต้องการสามชุดของเดิม

```
numBlocks = int(input("Hom many block of stars do you want? "))
numLines = int(input("How many line of each block? "))
numStars = int(input("How many stars per line? "))
for block in range(0, numBlocks):
    for line in range(0, numLines):
        for star in range(0, numStars):
            print("*", end=" ")
        print()
    print()
```

แสดงผลลัพท์ดังนี้

```
>>>
Hom many block of stars do you want? 3
How many line of each block? 4
How many stars per line? 8
* * * * * * * *
* * * * * * * *
* * * * * * * *
* * * * * * * *

* * * * * * * *
* * * * * * * *
* * * * * * * *
* * * * * * * *

* * * * * * * *
* * * * * * * *
* * * * * * * *
* * * * * * * *
```

ตัวอย่างเพิ่มเติมเกี่ยวกับการใช้ตัวแปรภายใต้ลูปซ้อนลูป

```
numBlocks = int(input("Hom many block of stars do you want? "))
for block in range(1, numBlocks + 1):
    print("block = ", block, ", block*2 = ", block*2)
    for line in range(1, block*2):
        for star in range(1, (block + line)*2):
            print("*", end=" ")
        print()
    print()
```

```
>>>
Hom many block of stars do you want? 3
block =  1 , block*2 =  2
* * *

block =  2 , block*2 =  4
* * * * *
* * * * * * *
* * * * * * * * *

block =  3 , block*2 =  6
* * * * * * *
```

```
>>>
Hom many block of stars do you want? 3
block =  1 , block*2 =  2
* * *

block =  2 , block*2 =  4
* * * * *
* * * * * * *
* * * * * * * * *

block =  3 , block*2 =  6
* * * * * * *
* * * * * * * * *
* * * * * * * * * * *
* * * * * * * * * * * * *
* * * * * * * * * * * * * * *
```

ในแต่ละรอบของ block จำนวนบรรทัดจะไม่เท่ากัน
และแต่ละบรรทัดจะไม่มีจำนวน * ไม่เท่ากัน

แสดงตัวเลขจำนวนควบคู่

```
numBlocks = int(input("Hom many block of stars do you want? "))
for block in range(1, numBlocks + 1):
    print("block = ", block, ", block*2 = ", block*2)
    for line in range(1, block*2):
        for star in range(1, (block + line)*2):
            print("*", end=" ")
        print("line =",line, ",star =",star)
    print()
```

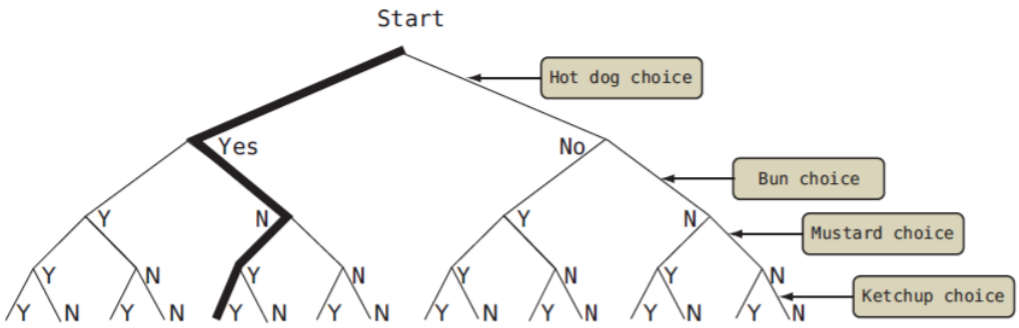
ได้ผลดังนี้

```
Hom many block of stars do you want? 3
block =  1 , block*2 =  2
* * * line = 1 ,star = 3

block =  2 , block*2 =  4
* * * * * line = 1 ,star = 5
* * * * * * * line = 2 ,star = 7
* * * * * * * * * line = 3 ,star = 9

block =  3 , block*2 =  6
* * * * * * * line = 1 ,star = 7
* * * * * * * * * line = 2 ,star = 9
* * * * * * * * * * * line = 3 ,star = 11
* * * * * * * * * * * * * line = 4 ,star = 13
* * * * * * * * * * * * * * * line = 5 ,star = 15
```

ตัวอย่างการใช้ Nested Loop เพิ่มเติม
การหากรณีความเป็นไปได้ทั้งหมดคอมบิเนชั่น (Combinations) ตัวอย่างการกินขนมปังฮอดดอกที่เป็นไปได้ทั้งหมด ถ้ามีส่วนประกอบของขนมปังได้ดังนี้ ฮอดดอก (hot dog) ขนมปัง(bun) ซอสมะเขือเทศ(ketchup) มาสตาร์ด(mustard) หอมหัวใหญ่ (onion) ทั้งหมด 5 อย่าง เราสามารถเขียนเป็นแผนภาพได้ภาพด้านล่างนี้ ซึ่งเราสามารถตัดสินใจว่าจะใส่ (yes) ไม่ใส่ (no) จำนวนกรณีได้ดังนี้



ตัวอย่างแสดงส่วนผสมสี่อย่าง 16 กรณี มีได้ทั้งหมด 4 ชั้น ถ้าครบทั้ง 5 อย่างจะมีทั้งหมด 32 กรณี พิจารณาการรวมกันได้ดังนี้

- ถ้ามีเฉพาะ Hot dog จะได้เพียงแค่สองกรณีคือ ไม่ กับ ใช่
- เมื่อมีขนมปังเพิ่มเข้ามา เราจะได้เพิ่มอีก 2 เท่ากรณีเนื่องจากเป็น 4 จะต้องเพิ่มรอบไปอีกสองครั้ง = 2x2 = 4 กรณี
- และถ้ามีซอสมะเขือเทศจะเพิ่มอีก 2 เท่าเป็น 2x2x2 = 8 กรณี
- ดังนั้นเมื่อพิจารณาทั้ง 5 ส่วนประกอบจะเท่ากับ 2x2x2x2x2 = 32 กรณี

เราสามารถเขียนโปรแกรมได้ดังนี้

```
print("\tDog \tBun \tKetchup\tMustard\tOnions")
count = 1
for dog in [0, 1]:
    for bun in [0, 1]:
        for ketchup in [0, 1]:
            for mustard in [0, 1]:
                for onion in [0, 1]:
                    print("#",count, "\t", dog, "\t", bun,
                          "\t", ketchup, "\t", mustard, "\t",onion)
                    count += 1
```

แสดงผลได้ดังนี้

```
>>>
        Dog     Bun     Ketchup Mustard Onions
# 1      0       0       0       0       0
# 2      0       0       0       0       1
# 3      0       0       0       1       0
# 4      0       0       0       1       1
# 5      0       0       1       0       0
# 6      0       0       1       0       1
# 7      0       0       1       1       0
# 8      0       0       1       1       1
# 9      0       1       0       0       0
# 10     0       1       0       0       1
# 11     0       1       0       1       0
# 12     0       1       0       1       1
# 13     0       1       1       0       0
# 14     0       1       1       0       1
# 15     0       1       1       1       0
# 16     0       1       1       1       1
# 17     1       0       0       0       0
# 18     1       0       0       0       1
# 19     1       0       0       1       0
# 20     1       0       0       1       1
# 21     1       0       1       0       0
# 22     1       0       1       0       1
# 23     1       0       1       1       0
# 24     1       0       1       1       1
# 25     1       1       0       0       0
# 26     1       1       0       0       1
# 27     1       1       0       1       0
# 28     1       1       0       1       1
# 29     1       1       1       0       0
# 30     1       1       1       0       1
# 31     1       1       1       1       0
# 32     1       1       1       1       1
```

โดยค่า 0 แสดงถึงการไม่ใส่ส่วนประกอบนี้และค่า 1 คือมีส่วนประกอบนี้
แต่บางกรณีอาจจะขัดกับความรู้สึกเช่น กรณีที่ 27 มีแค่ซอสมะเขือเทศและมาสตาร์ด แต่ถ้ามีคนสั่งเราจำต้องทำให้เพราะ ลูกค้าถูกเสมอ

คำนวณปริมาณแคลลารี่

จากตัวอย่างการคำนวณส่วนประกอบที่เป็นไปได้ เราสามารถหาปริมาณแคลลารี่ของการกินได้ โดยระบุประมาณแคลลารี่แต่ละองค์ประกอบ เช่น

```
dog_cal = 140
bun_cal = 120
mus_cal = 20
ket_cal = 80
onion_cal = 40
```

และสมการคำนวณได้ดังนี้

```
tot_cal = (dog * dog_cal) + (bun * bun_cal) + \
          (mustard * mus_cal) + (ketchup * ket_cal) + \
          (onion * onion_cal)
```

ถ้ามีไม่ส่วนประกอบจะมีค่าเป็น 0 ประมาณแคลลารี่จะไม่โดยรวมเข้าไปด้วย
ถ้าผลลัพท์ที่ต้องการจะได้เป็นดังรูปนี้

```
>>>
        Dog     Bun     Ketchup Mustard Onions  Calorites
# 1      0       0       0       0       0       0
# 2      0       0       0       0       1       40
# 3      0       0       0       1       0       20
# 4      0       0       0       1       1       60
# 5      0       0       1       0       0       80
# 6      0       0       1       0       1       120
# 7      0       0       1       1       0       100
# 8      0       0       1       1       1       140
# 9      0       1       0       0       0       120
# 10     0       1       0       0       1       160
# 11     0       1       0       1       0       140
# 12     0       1       0       1       1       180
# 13     0       1       1       0       0       200
# 14     0       1       1       0       1       240
# 15     0       1       1       1       0       220
# 16     0       1       1       1       1       260
# 17     1       0       0       0       0       140
# 18     1       0       0       0       1       180
# 19     1       0       0       1       0       160
# 20     1       0       0       1       1       200
# 21     1       0       1       0       0       220
# 22     1       0       1       0       1       260
# 23     1       0       1       1       0       240
# 24     1       0       1       1       1       280
# 25     1       1       0       0       0       260
# 26     1       1       0       0       1       300
# 27     1       1       0       1       0       280
# 28     1       1       0       1       1       320
# 29     1       1       1       0       0       340
# 30     1       1       1       0       1       380
# 31     1       1       1       1       0       360
# 32     1       1       1       1       1       400
```

เราสามารถเขียนโค้ดเพิ่มเติมได้ดังนี้

```
print("\tDog \tBun \tKetchup\tMustard\tOnions\tCalorites")
dog_cal = 140
bun_cal = 120
mus_cal = 20
ket_cal = 80
onion_cal = 40
count = 1
for dog in [0, 1]:
    for bun in [0, 1]:
        for ketchup in [0, 1]:
            for mustard in [0, 1]:
                for onion in [0, 1]:
                    total_cal = (dog * dog_cal) + (bun*bun_cal) +\
                                (mustard* mus_cal) + (ketchup*ket_cal) +\
                                (onion * onion_cal)
                    print("#",count, "\t", dog, "\t", bun,
                          "\t", ketchup, "\t", mustard,
                          "\t",onion, "\t", total_cal)
                    count += 1
```

ทบทวน

1. Nested Loop คือการเขียนโปรแกรมการวนซ้อนการวน
2. Variable loop คือ ตัวแปรที่ใช้ในการวนค่า ซึ่งอาจเปลี่ยนแปลงได้ในแต่ละรอบ
3. Combinations คือ ความเป็นได้ทั้งหมดของทุกกรณี
4. Decision Trees คือ แผนภาพการตัดสินใจแบบต้นไม้

ทดสอบความรู้

1. จงยกตัวอย่างการใช้ variable loop ใน python?
2. จงยกตัวอย่างการใช้ nested loop ใน python?
3. จงบอกจำนวน * ทั้งหมดกี่อัน

```
for i in range(5):
    for j in range(3):
        print("*", end="")
    print()
```

4. ผลลัพธ์ที่ได้จากข้อ 3 คือ
5. ถ้า Decision Tree มีทั้งหมด 4 ระดับ (level) แต่ละระดับแบ่งออกได้เป็นสองค่า จงหาจำนวนกรณีทั้งหมด

จงเขียนโปรแกรมต่อไปนี้

1. จากโค้ดการปล่อยจรวด นับถอยหลังครั้งละหนึ่งวินาที

```
import time
for i in range(10, 0, -1):
    print(i)
    time.sleep(1)
print("Go Go!")
```

จงแก้ไขโค้ดเพื่อให้ได้ผลลัพท์ดังนี้

```
Countdown time: How many seconds? 4
4
3
2
1
Go Go!
```

โปรแกรมสามารถป้อนจำนวนวินาทีได้ แล้ววนตามจำนวนวินาที

2. จากข้อ#1 ให้แก้ไขโปรแกรม ให้พิมพ์จำนวน * ตามจำนวนตัวเลขที่ออกมา ผลลัพท์ของโปรแกรมดังนี้

```
Countdown time: How many seconds? 4
4 * * * *
3 * * *
2 * *
1 *
Go Go!
```