**Lists ในภาษา Python**

April 9, 2017 · 1,943 views

ในบทนี้ คุณจะได้เรียนรู้เกี่ยวกับโครงสร้างข้อมูลแบบ List ในภาษา Python เราจะพูดถึงการสร้างและใช้งาน List ในเบื้องต้น การใช้งานเมธอดและฟังก์ชันเพื่อจัดการข้อมูลภายใน List และการใช้งานคำสั่ง For loop กับ List รวมถึงการ slicing



List (ลิสต์) คือโครงสร้างข้อมูลชนิดหนึ่งในภาษา Python ที่ใช้เก็บข้อมูลแบบลำดับ (Sequence) โดยมี Index เป็นตัวระบุตำแหน่งในการเข้าถึงข้อมูล เราสามารถใช้ List เพื่อเก็บข้อมูลจำนวนมากและหลากหลายประเภทในเวลาเดียวกัน List เป็นประเภทข้อมูลที่ใช้อย่างหลากหลายในการเขียนโปรแกรม นอกจากนี้ ในภาษา Python ยังมี built-in function ที่สามารถทำงานกับ List และใน List ออบเจ็คเองก็มีเมธอดต่างๆ เป็นจำนวนมากที่ช่วยอำนวยความสะดวกในการเขียนโปรแกรม

**การประกาศและใช้งาน List**

List นั้นเป็นตัวแปรประเภทหนึ่ง การใช้งานของมันจะเหมือนกันอาเรย์ในภาษาอื่นๆ ในการประกาศ List นั้นข้อมูลของมันจะอยู่ภายในเครื่องหมาย [] และคั่นสมาชิกแต่ละตัวด้วยเครื่องหมายคอมมา , มาดูตัวอย่างการประกาศ List ในภาษา Python

numbers = [-1, 2, 5, 8, 10, 13]  
names = ['Mateo', 'Danny', 'James', 'Thomas', 'Luke']  
mixed\_type = [-2, 5, 84.2, "Mountain", "Python"]

ในตัวอย่าง เราได้สร้างตัวแปร List สามตัวแปร numbers เป็นตัวแปร List ที่มีสมาชิกเป็นตัวเลขจำนวนเต็ม 6 ตัว names เป็น List ของ String ที่สำหรับเก็บชื่อและมี 5 รายชื่อ และสุดท้ายตัวแปร mixed\_type เป็น List ที่เก็บข้อมูลประเภทต่างๆ แบบรวมกันในตัวแปรเดียวซึ่งมีสมาชิกทั้งหมด 5 ตัว ซึ่งทั้งหมดนี้เป็นการกำหนดสมาชิกให้กับ List พร้อมกับการประกาศตัวแปร ในภาษา Python เราสามารถกำหนดค่าให้กับ List หลังจากประกาศตัวแปรแล้วได้ มาดูตัวอย่าง

numbers = []  
numbers.append(-1)  
numbers.append(2)  
numbers.append(5)  
numbers.append(8)  
numbers.append(10)  
numbers.append(13)  
  
names = ['Mateo', 'Danny']  
names.append('James')  
names.append('Thomas')  
names.append('Luke')  
  
print(numbers)  
print(names)  
  
print('numbers count = ', len(numbers))  
print('names count = ', len(names))

ในตัวอย่าง เป็นการกำหนดค่าให้กับ List หลังจากที่มันถูกสร้างแล้ว เราใช้เมธอด append()เพื่อเพิ่มข้อมูลใหม่เข้าไปใน List ซึ่งข้อมูลที่เพิ่มเข้าจะอยู่ท้ายสุดและเรียง Index เพิ่มขึ้นไปเรื่อยๆ โดยเริ่มจาก 0 ในตัวแปร numbers เราได้เพิ่ม 5 จำนวนเต็มเข้าไปใน List และในตัวแปร names ในตอนแรกได้ประกาศและกำหนดสองชื่อให้กับตัวแปร และเพิ่มเข้าไปภายหลังอีก 3 ชื่อ และฟังก์ชัน len() ใช้เพื่อนับจำนวนสมาชิกภายใน List

[-1, 2, 5, 8, 10, 13]  
['Mateo', 'Danny', 'James', 'Thomas', 'Luke']  
numbers count = 6  
names count = 5

นี่เป็นผลลัพธ์การทำงานของโปรแกรม

**การเข้าถึงข้อมูลภายใน List**

List นั้นใช้ Index สำหรับการเข้าถึงข้อมูล โดย Index ของ List จะเป็นจำนวนเต็มที่เริ่มจาก 0 และเพิ่มขึ้นทีละ 1 ไปเรื่อยๆ ดังนั้น เราจึงสามารถเข้าถึงข้อมูลภายใน List เพื่ออ่านหรืออัพเดทค่าได้โดยตรงผ่าน Index ของมัน นี่เป็นโค้ดการเข้าถึงข้อมูลภายใน List ในภาษา Python

names = ['Mateo', 'Danny', 'James', 'Thomas', 'Luke']  
print('names[0] = ', names[0])  
print('names[3] = ', names[3])  
print('names[-1] = ', names[-1])  
  
# update value  
names[0] = 'Bob'  
print('names[0] = ', names[0])

ในตัวอย่าง เรามีตัวแปร List ที่ชื่อว่า names ดังนั้น เพือเข้าถึงสมาชิกตัวแรกภายใน List ซึ่งก็คือ *"Mateo"* นั้นจะใช้คำสั่ง names[0] และสมาชิกที่มีค่าเป็น *"Thomas"* ซึ่งอยู่ตำแหน่งที่ 4 จะใช้คำสั่ง names[3] สังเกตว่า Index จะลดลงหนึ่งเพราะ Index ของ List นั้นเริ่มต้นจาก 0

print('names[-1] = ', names[-1])

นอกจากนี้ เราสามารถเข้าถึงข้อมูลภายใน List โดยการใช้ Index เป็นจำนวนลบได้ โดยเริ่มจาก -1 ซึ่งเป็นสมาชิกตัวสุดท้ายของ List และ -2 สมาชิกตัวถัดมาและลดลงไปทีละ 1

names[0] = 'Bob'  
print('names[0] = ', names[0])

นี่เป็นการอัพเดทค่าของสมาชิกภายใน List ในตัวอย่างเราได้เปลี่ยนค่าของสมาชิกในตำแหน่งแรกของ List จากเดิมที่เป็น *"Mateo"* ให้เป็น *"Bob"*

names[0] = Mateo  
names[3] = Thomas  
names[-1] = Luke  
names[0] = Bob

นี่เป็นผลลัพธ์การทำงานของโปรแกรม

**การอ่านค่าใน List ด้วยคำสัง For loop**

เนื่องจาก List นั้นเก็บข้อมูลเป็นแบบลำดับและใช้ Index ในการเข้าถึงข้อมูล ดังนั้น เราจึงมักจะใช้คำสั่งวนซ้ำสำหรับการเขียนโปรแกรมที่ทำงานกับ List เพราะทำให้การทำงานรวดเร็วและง่ายขึ้น เช่น การใช้คำสั่งวนซ้ำวนอ่านค่าใน List ที่มีข้อมูลเป็นจำนวนมาก เป็นต้น ต่อไปมาดูตัวอย่างการใช้งานคำสั่ง For loop กับ List ในภาษา Python

numbers = [10, 20, 30, 40, 50, 60, 70]  
sum = 0  
for n in numbers:  
 print(n, end =' ')  
 sum += n  
  
print('sum = ', sum)  
  
names = ['Mateo', 'Danny', 'James', 'Thomas', 'Luke']  
for i in range(0, len(names)):  
 print(names[i].upper(), end =' ')

ในตัวอย่าง เป็นการวนอ่านค่าภายใน List ด้วยการใช้คำสั่ง For loop โดยเราได้แยกการทำงานออกเป็นสองลูป ในลูปแรกเป็นการใช้งานคำสั่ง For loop เพื่อวนอ่านค่าภายใน List numbersโดยตรง โปรแกรมจะวนอ่านค่าไปทีละค่าและนำค่าในแต่ละรอบที่ได้มาใส่ในตัวแปร n เราได้ทำการแสดงผลตัวเลขภายใน List และหาผลรวมของตัวเลขภายใน List โดยเก็บไว้ในตัวแปร sum

ในลูปที่สอง เป็นการใช้คำสั่ง For loop เช่นกัน แต่ในตอนนี้เราจะสร้าง Index ขึ้นมาโดยการใช้ตัวแปร i เป็นตัวรัน Index จาก 0 ถึงขนาดตัวสุดท้ายของ List ที่สร้างจากฟังก์ชัน range()และเราแสดงผลชื่อในตัวพิมพ์ใหญ่ด้วยเมธอด upper() ของ String

10 20 30 40 50 60 70 sum = 280  
MATEO DANNY JAMES THOMAS LUKE

นี่เป็นผลลัพธ์การทำงานของโปรแกรมของการใช้งาน List กับคำสั่ง For loop

**List slicing**

ในภาษา Python เราสามารถตัดข้อมูลจาก List หนึ่งแล้วนำไปสร้างเป็น List ใหม่ได้ โดยวิธีดังกล่าวนั้นเรียกกว่า *slicing* ในการตัดข้อมูลใน List นั้นจะทำในรูปแบบ [a:b] เมื่อ a เป็น Index เริ่มต้นและ b เป็น Index ก่อนสมาชิกตัวสุดท้ายที่ต้องการตัด มาดูตัวอย่างของ list slicing

ch = ['a', 'b', 'c', 'd', 'e', 'f', 'g', 'h']  
  
a = ch[0:4] # a - d  
b = ch[4:9] # e - h  
c = ch[:3] # a - c  
d = ch[3:] # c - h  
e = ch[:] # copy all list, or equivalent to e = ch  
f = ch[0:2] + ch[6:8] # a - b and g - h  
  
print(a)  
print(b)  
print(c)  
print(d)  
print(e)  
print(f)

ในตัวอย่าง เรามีตัวแปร ch ซึ่งมีสมาชิกเป็นตัวอักษรในภาษาอังกฤษจาก a ถึง h หลังจากนั้นเราได้ทำการ slice ข้อมูลจาก List ดังกล่าว ในตัวแปร a นั้นเป็นการตัดเอาข้อมูลจากตำแหน่งที่ 0 ถึง 3 มา ในตัวแปร b นั้นตัดเอาตำแหน่งที่ 4 ถึง 8 ถัดมาเป็นตัวแปร c d และ e เป็นการเว้นว่างตำแหน่งข้างหน้าและข้างหลัง ซึ่งถ้าตำแหน่งข้างหน้าถูกเว้นว่างไว้ เป็นการตัดเอาสมาชิกตั้งแต่ตำแหน่งเริ่มต้นของ List และถ้าตำแหน่งสิ้นสุดถูกเว้นว่างไว้ เป็นการตัดเอาจนถึงสมาชิกตัวสุดท้ายของ List และในตัวแปร f เป็นการตัดเอาสองส่วนของ List มาต่อกัน ในการนำสอง List มาต่อกันนั้นเราจะใช้ตัวดำเนินการ +

['a', 'b', 'c', 'd']  
['e', 'f', 'g', 'h']  
['a', 'b', 'c']  
['d', 'e', 'f', 'g', 'h']  
['a', 'b', 'c', 'd', 'e', 'f', 'g', 'h']  
['a', 'b', 'g', 'h']

นี่เป็นผลลัพธ์การทำงานของโปรแกรม

**การใช้งานคำสั่ง del เพื่อลบข้อมูลใน List**

คำสั่ง del เป็นคำสั่งที่ใช้สำหรับลบตัวแปรใดๆ ออกไปจากหน่วยความจำหรือใช้ยกเลิกตัวแปรที่เคยประกาศไปแล้ว เราสามารถใช้คำสั่ง del เพื่อลบสมาชิกภายใน List ได้เช่นเดียวกัน มาดูตัวอย่างการใช้งานคำสั่ง del ในภาษา Python

ch = ['a', 'b', 'c', 'd', 'e', 'f', 'g', 'h']  
  
del ch[0] # delete a  
 print(ch)  
  
del ch[0] # delete b  
 print(ch)  
  
del ch[2:4] # delete e, f  
 print(ch)  
  
del ch[:] # delete all  
print(ch)

ในตัวอย่าง เป็นการใช้งานคำสั่ง del สำหรับลบสมาชิกภายใน List ในตอนแรก เราได้ลบอักษร a ออกไปจาก List ซึ่งสมาชิกตัวแรกนั้นจะมี Index เป็น 0 ดังนั้น เมื่อการลบเสร็จสิ้น List จะทำการเลื่อนตัวอักษร b มายัง Index 0 แทน ซึ่งสิ่งนี้เป็นสภาวะการสูญเสียตำแหน่งของ List หรือ Index lose หลังจากนั้น เราได้ลบโดยการใช้วิธีการกำหนด Index แบบ slicing เราได้ทำการลบค่าจาก Index 0 ถึง 3 ซึ่งเป็นการลบตัวอักษรจาก e ถึง f และในคำสั่งสุดท้ายเป็นการลบข้อมูลภายใน List ทั้งหมด

['b', 'c', 'd', 'e', 'f', 'g', 'h']  
['c', 'd', 'e', 'f', 'g', 'h']  
['c', 'd', 'g', 'h']  
[]

นี่เป็นผลลัพธ์การทำงานของโปรแกรม

Warning: จากตัวอย่างข้างต้นนั้น เมื่อคุณต้องการลบข้อมูลทั้งหมดภายใน List เราจะใช้ del ch[:] เพราะว่าเป็นการลบแบบ slicing จากตำแหน่งแรกถึงตำแหน่งสุดท้ายและจะทำให้มันกลายเป็น List ว่างปล่าว เมื่อคุณใช้คำสั่ง del ch จะหมายความว่าลบตัวแปรออกไปจากหน่วยความจำแทน

ในบทนี้ คุณได้เรียนรู้เกี่ยวกับการประกาศและใช้งาน List ในภาษา Python นอกจากนี้เรายังใช้คำสัง For loop เพื่อให้ง่ายในการอ่านข้อมูลภายใน List รวมทั้งการตัดข้อมูลใน List ด้วยการ slicing และการใช้คำสั่ง del เพื่อลบข้อมูลภายใน List ออกไป ในบทต่อไป เราจะเป็นการใช้งานฟังก์ชันและเมธอดของ List