แผนบริหารการสอนประจำบทที่ 5

ลีสต์

หัวข้อประจำบท

1. ความหมายของลีสต์
2. การดำเนินการภายในลีสต์
3. การจัดการข้อมูลภายในลีสต์
4. ฟังก์ชันที่เกี่ยวข้องกับลีสต์
5. ลีสต์แบบหลายมิติ

วัตถุประสงค์เชิงพฤติกรรม

1. เพื่อให้ผู้เรียนสามารถอธิบายเกี่ยวกับลีสต์ได้
2. เพื่อให้ผู้เรียนสามารถอธิบายเกี่ยวกับการดำเนินการภายในลีสต์ได้
3. เพื่อให้ผู้เรียนสามารถอธิบายเกี่ยวกับการจัดการข้อมูลภายในลีสต์ได้
4. เพื่อให้ผู้เรียนสามารถอธิบายเกี่ยวกับฟังก์ชันที่เกี่ยวข้องกับลีสต์ได้
5. เพื่อให้ผู้เรียนสามารถอธิบายเกี่ยวกับลีสต์แบบหลายมิติได้
6. เพื่อให้ผู้เรียนสามารถนำความรู้เรื่อง ลีสต์ ประยุกต์ใช้งานในการแก้โจทย์ปัญหาได้อย่างถูกต้อง

วิธีการสอนและกิจกรรม

1. ผู้สอนบรรยายในชั้นเรียน และโปรแกรมนำเสนอ ตามหัวข้อเนื้อหาในเอกสารประกอบการสอน
2. ผู้เรียนทำแบบฝึกหัดท้ายบท และฝึกเขียนโปรแกรมด้วยคอมพิวเตอร์

สื่อการเรียนการสอน

1. เอกสารประกอบการสอน วิชา หลักการเขียนโปรแกรมคอมพิวเตอร์
2. โปรแกรม Python รุ่น 3.8.10
3. โปรแกรม PowerPoint

การวัดผลและการประเมินผล

1. สังเกตจากการอภิปราย การตอบคำถาม และซักถามระหว่างเรียน
2. การปฏิบัติบนเครื่องคอมพิวเตอร์
3. การทำแบบฝึกหัดท้ายบท

บทที่ 5
ลีสต์

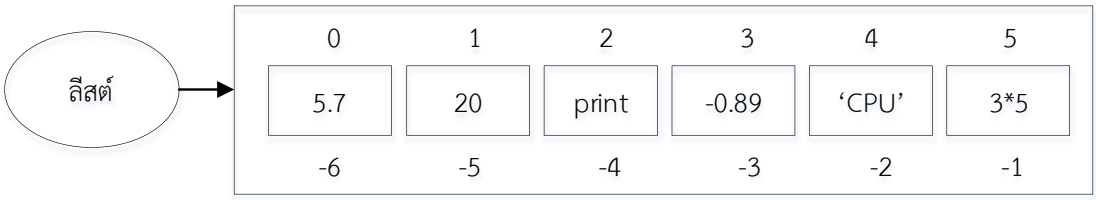
การเก็บข้อมูลโดยทั่วไปอยู่ในรูปของตัวแปร ลักษณะของตัวแปรโดยทั่วไปประกอบไปด้วยการเก็บข้อมูลหนึ่งตัวแปรสามารถเก็บข้อมูลได้ประเภทเดียว ข้อมูลที่ใช้เก็บพื้นฐานประกอบไปด้วยข้อมูลประเภทตัวเลขทศนิยม ตัวเลขจำนวนเต็ม และตัวอักขระ แต่จะมีรูปแบบของการเก็บข้อมูลที่หนึ่งตัวแปรสามารถเก็บข้อมูลได้หลายค่าและแต่ละค่าของข้อมูลสามารถมีความแตกต่างกันได้คือตัวแปรประเภทแถวลำดับ (Fior, 2016) การเก็บข้อมูลในลักษณะนี้ จะเก็บค่าของข้อมูลจัดเรียงตามลำดับค่าของข้อมูลไว้ในตัวแปรและกำหนดหมายเลขกำกับ ให้กับค่าในโครงสร้างของตัวแปร (โชติพันธุ์ หล่อเลิศสุนทร และ ฐิตะพันธุ์ หล่อเลิศสุนทร, 2559)

การใช้งานตัวแปรนั้นมีความสำคัญมากในการทำงานของโปรแกรม เพื่อให้มีความยืดหยุ่นต่อการใช้งาน จึงมีการกำหนดให้ตัวแปรหนึ่งตัวแปรสามารถเก็บข้อมูลได้หลายค่า เพิ่มความสะดวกในการเก็บข้อมูลและการเข้าถึงข้อมูล การทำงานลักษณะดังกล่าว ภาษาไพธอนได้เตรียมเครื่องมือในการจัดการข้อมูล เรียกว่า “ลีสต์” โดยมีรายละเอียดดังต่อไปนี้

ความหมายของลีสต์

ลีสต์ (List) หมายถึง การจัดเรียงของข้อมูล และข้อมูลเรียกว่า องค์ประกอบ (Element) และค่าข้อมูลนั้นสามารถเป็นต่างชนิดกันได้ ตัวแปรที่ถูกใช้ในการเก็บข้อมูลต่าง ๆ แสดงในรูปแบบของการจัดเรียงกันของข้อมูล ข้อมูลที่อยู่ในลีสต์นั้นไม่จำเป็นต้องเป็นข้อมูลประเภทเดียวกัน (Halteman, 2016) ข้อมูลภายในลีสต์สามารถเปลี่ยนแปลงได้ โดยการเพิ่มหรือลดจึงมีจำนวนไม่แน่นอน ใช้งานภายใต้เครื่องหมาย [ ] (Fior, 2016) ลีสต์ คล้ายกับภาษาซีและภาษาปาสคัล (Pascal) หรือภาษาอื่น ๆ แตกต่างที่ภาษาไพธอน อนุญาตให้ข้อมูลภายในลีสต์ เป็นตัวแปรชนิดใด ๆ ได้อย่างไม่จำกัด ตัวแปรลีสต์ยังสามารถฝังตัวในตัวแปรลีสต์ด้วยกันได้ (จักรกฤษ แสงแก้ว, 2549) มีความยืดหยุ่นในการกำหนดค่าข้อมูลงในตัวแปร และการนำตัวแปรชนิดนี้ไปใช้งาน สามารถเพิ่ม ลบ และแก้ไขข้อมูลในตัวแปรได้ตลอดเวลา โครงสร้างการจัดเก็บค่าของตัวแปรลีสต์ลักษณะนี้เรียกว่า Mutable คือ ค่าข้อมูลภายในลีสต์ สามารถเปลี่ยนแปลงได้ตลอดเวลา (โชติพันธุ์ หล่อเลิศสุนทร และ ฐิตะพันธุ์ หล่อเลิศสุนทร, 2559)

จากความหมายและลักษณะการเก็บข้อมูลโครงสร้างแบบลีสต์ สามารถอธิบายตัวอย่างได้ดังภาพที่ 5.1



ภาพที่ 5.1 โครงสร้างตัวอย่างการเก็บข้อมูลตัวแปรลีสต์

โครงสร้างตัวอย่างการเก็บข้อมูลตัวแปรลีสต์ ภาพที่ 5.1 แสดงให้เห็นการเก็บข้อมูลที่มีลักษณะข้อมูลที่แตกต่างกัน ข้อมูลอาจอยู่ในรูปของตัวเลขจำนวนเต็ม ติดลบ ข้อความ หรือคำสั่ง สามารถเก็บข้อมูลในตัวแปรเดียวกันได้ การเก็บข้อมูลลักษณะนี้ คือ ตัวแปรลีสต์

การดำเนินการภายในลีสต์

การเก็บข้อมูลภายในลีสต์นั้น จะมีตำแหน่งของข้อมูลแต่ละข้อมูล โดยสามารถอ้างอิงตำแหน่งดังกล่าวเพื่อดำเนินการภายในลีสต์ โครงสร้างและการดำเนินการภายในลีสต์ มีรายละเอียดดังต่อไปนี้

1. โครงสร้างพื้นฐานของลีสต์

ลีสต์จะมีการเก็บข้อมูลแบบแถวลำดับ ดังนั้นข้อมูลที่อยู่ภายในลีสต์แต่ละข้อมูลจะถูกเก็บไว้โดยมีตำแหน่งของลีสต์ (Lists index) กำกับตำแหน่งข้อมูล การใช้งานลีสต์นั้น ตัวเลขที่อยู่ในสัญลักษณ์ [ ] หมายถึง ตัวเลขชี้ระบุตำแหน่งข้อมูลในลีสต์ โดยตำแหน่งที่ 0 เป็นตำแหน่งเริ่มต้นการเก็บข้อมูลของลีสต์ ดังภาพที่ 5.2

ภาพที่ 5.2 โครงสร้างข้อมูลลีสต์

โครงสร้างการเก็บข้อมูลของลีสต์ สามารถนำไปเขียนโปรแกรม โดยรูปแบบและรายละเอียดการทำงาน ดังโปรแกรมตัวอย่างที่ 5.1

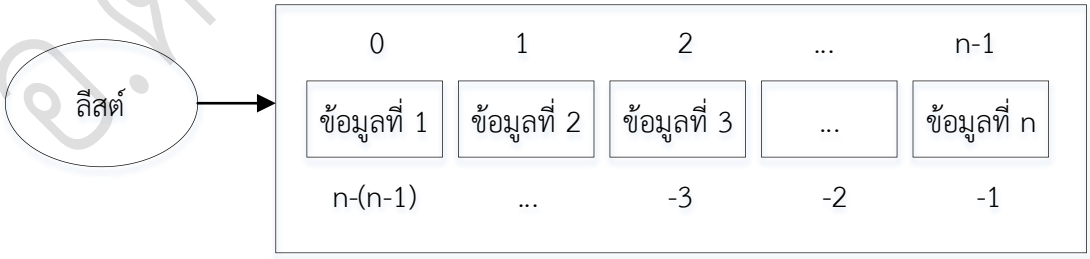
ตัวอย่างที่ 5.1 การสร้างตัวแปรลีสต์

บรรทัดที่	คำสั่ง
1	lst =[2,-3,0,4,-1]
2	print(lst[0])
3	print(lst[2])
4	print(lst[4])
5	print(lst[1],lst[4])
ผลลัพธ์	
2 0 -1 -3 -1	

ตัวแปรลีสต์ ประกอบด้วยสมาชิก 5 ข้อมูล ตำแหน่งของข้อมูลอยู่ระหว่าง 0 ถึง 4 ดังนั้นเมื่อเรียกดูข้อมูล สามารถเรียกดูตำแหน่งเดียวหรือหลายตำแหน่งพร้อมกันได้ ดังผลลัพธ์ที่ได้โปรแกรมตัวอย่างที่ 5.1

2. ตำแหน่งเลขลบของลีสต์

ตำแหน่งลีสต์โดยเลขลบ (Negative Lists Index) คือ การอ้างอิงตำแหน่งโดยใช้เลขติดลบในการอ้างอิงตำแหน่งข้อมูลที่อยู่ในลีสต์ โดยตำแหน่งที่อ้างอิงเป็นตัวเลขที่ติดลบนั้น จะเรียกข้อมูลในตำแหน่งสุดท้ายขึ้นมาเป็นลำดับแรก ตำแหน่งสุดท้ายจะเป็น [-1] ดังภาพที่ 5.3



ภาพที่ 5.3 โครงสร้างตำแหน่งเลขลบของลีสต์

โครงสร้างการเก็บข้อมูลของลีสต์ สามารถนำไปเขียนโปรแกรม โดยมีรูปแบบและลักษณะการทำงาน ดังโปรแกรมตัวอย่างที่ 5.2

ตัวอย่างที่ 5.2 การอ้างอิงตำแหน่งเลขลบของลีสต์

บรรทัดที่	คำสั่ง
1	lst = [10, 20, 40, 60, 80, 100]
2	print(lst[-1])
3	print(lst[-3])
4	print(lst[-5])
ผลลัพธ์	
100 60 20	

ตัวแปรลีสต์ ประกอบด้วยสมาชิก 6 ข้อมูล ตำแหน่งของข้อมูลอยู่ระหว่าง -1 ถึง -6 ดังนั้น เมื่อเรียกดูข้อมูลในตำแหน่ง -1 -3 และ -5 จะเริ่มอ่านข้อมูลในลำดับสุดท้ายก่อน ดังผลลัพธ์ที่ได้ของโปรแกรมตัวอย่างที่ 5.2

3. การเก็บข้อมูลชนิดต่างกันของลีสต์

ภาษาไพธอน อนุญาตให้สามารถเก็บข้อมูลชนิดต่างกันไว้ในตัวแปรเดียวได้ ซึ่งเป็นความสามารถของการเก็บข้อมูลภายในลีสต์ ดังโปรแกรมตัวอย่างที่ 5.3

ตัวอย่างที่ 5.3 การเก็บข้อมูลชนิดต่างกันภายในลีสต์

บรรทัดที่	คำสั่ง
1	collection = [24.2, 4, 'word', print, 19, -0.03]
2	print(collection[0])
3	print(collection[1])
4	print(collection[2])
5	print(collection[3])
6	print(collection[4])
7	print(collection[5])
ผลลัพธ์	
24.2 4	

```
word
<built-in function print>
19
-0.03
```

โปรแกรมตัวอย่างที่ 5.3 มีการเก็บข้อมูลต่างชนิดภายในลิสต์ชนิด ซึ่งมีรายละเอียด คือ ตำแหน่งที่ [0] และ [5] เก็บข้อมูลที่เป็นจำนวนจริง ตำแหน่งที่ [1] และ [4] เก็บข้อมูลที่เป็นจำนวนเต็มตำแหน่งที่ [2] และ [6] เก็บข้อมูลที่เป็นข้อความและตำแหน่งที่ [3] เก็บข้อมูลที่เป็นฟังก์ชัน

การจัดการข้อมูลภายในลิสต์

การเข้าถึงข้อมูลของตัวแปรแบบแถวลำดับ นอกเหนือจาการอ้างอิงจากเลขตำแหน่งของข้อมูล ยังมีฟังก์ชันการเข้าถึงข้อมูลภายในลิสต์ได้ง่ายและสะดวกรวดเร็ว นำข้อมูลที่อยู่ภายในลิสต์มาดำเนินการหรือจัดการ ให้ได้ผลลัพธ์ที่ต้องการ รายละเอียดการใช้คำสั่งแต่ละประเภทดังต่อไปนี้

1. การหาขนาดของลิสต์

คำสั่ง len คือ คำสั่งที่ใช้ในการหาขนาดหรือจำนวนสมาชิกที่อยู่ในลิสต์ มีรายละเอียดการใช้คำสั่งดังโปรแกรมตัวอย่างที่ 5.4

ตัวอย่างที่ 5.4 การใช้คำสั่ง len หาจำนวนสมาชิกภายในลิสต์

บรรทัดที่	คำสั่ง
1	lst1 = [24.2, 4, 'word', print, 19, -0.03, 'end']
2	lst2 = ['a', 1, -5.67, 'computer']
3	print (len(lst1))
4	print (len(lst2))
ผลลัพธ์	
7 4	

การใช้คำสั่ง len ตรวจสอบจำนวนสมาชิกภายในลิสต์ ซึ่งประกอบด้วยลิสต์ lst1 และ lst2 ที่มีจำนวนสมาชิกภายในลิสต์ 7 และ 4 ข้อมูล ดังผลลัพธ์ที่ได้โปรแกรมตัวอย่างที่ 5.4

2. การอ่านข้อมูลลีสต์ด้วยคำสั่งวนซ้ำ

การเรียกข้อมูลภายในลีสต์ขึ้นมาใช้งาน สามารถใช้คำสั่งแบบวนซ้ำคำสั่ง for ซึ่งมีรายละเอียดโปรแกรมตัวอย่างที่ 5.5

ตัวอย่างที่ 5.5 การเรียกใช้ข้อมูลภายในลีสต์ด้วยคำสั่ง for

บรรทัดที่	คำสั่ง
1	lst = [24.2, 4, 'word', print, 19, -0.03, 'end']
2	for item in lst:
3	print(item)
ผลลัพธ์	
24.2 4 word <built-in function print> 19 -0.03 end	

การเรียกใช้ข้อมูลภายในลีสต์ด้วยคำสั่ง for คือ การเข้าถึงข้อมูลสมาชิกภายในลีสต์และเรียกดูข้อมูลภายลีสต์ได้ นอกเหนือจาการใช้คำสั่ง for ยังสามารถใช้คำสั่ง for...in range ดังผลลัพธ์ที่ได้โปรแกรมตัวอย่างที่ 5.6

ตัวอย่างที่ 5.6 การเรียกข้อมูลภายในลีสต์ด้วยคำสั่งวนซ้ำ for...in range

บรรทัดที่	คำสั่ง
1	lst = [24.2, 4, 'word', print, 19, -0.03, 'end']
2	for item in range(len(lst)):
3	print(item)
ผลลัพธ์	
0 1 2	

3 4 5 6

การใช้คำสั่งลักษณะวนซ้ำเช่นเดียวกันกับคำสั่ง for ยังมีการใช้คำสั่ง for...in range ซึ่งมีความสามารถเข้าไปเรียกดูหรือดึงข้อมูลแบบวนซ้ำภายในลิสต์ได้เช่นเดียวกัน การใช้คำสั่ง for...in range เข้าถึงสมาชิกภายในลิสต์ มีรายละเอียดของรูปแบบและลักษณะการใช้คำสั่ง ดังโปรแกรมตัวอย่างที่ 5.6

3. การประมวลผลของลิสต์

ความสามารถในการใช้เครื่องหมายทางคณิตศาสตร์ มาใช้ในการประมวลผลข้อมูลภายในลิสต์ สามารถใช้เครื่องหมายในการดำเนินการ 2 ประเภท ได้แก่ คือเครื่องหมาย “+” และเครื่องหมาย “*” การทำงานคล้ายกับการประมวลผลตัวแปรทั่วไป ดังรายละเอียดการทำงานโปรแกรมตัวอย่างที่ 5.7

ตัวอย่างที่ 5.7 การประมวลผลของลิสต์

บรรทัดที่	คำสั่ง
1	lst1 = [10,20,30,40]
2	lst2 = ['aa','bb','cc','dd']
3	a = lst1 + lst2
4	b = lst1 * 2
5	c = lst2 * 3
6	print (a)
7	print (b)
8	print (c)
9	print (a*2)
ผลลัพธ์	
[10, 20, 30, 40, 'aa', 'bb', 'cc', 'dd'] [10, 20, 30, 40, 10, 20, 30, 40] ['aa', 'bb', 'cc', 'dd', 'aa', 'bb', 'cc', 'dd', 'aa', 'bb', 'cc', 'dd'] [10, 20, 30, 40, 'aa', 'bb', 'cc', 'dd', 10, 20, 30, 40, 'aa', 'bb', 'cc', 'dd']	

การประมวลผลของลีสต์ สามารถใช้สัญลักษณ์ทางคณิตศาสตร์ มาใช้ในการคำนวณกับลีสต์ได้ โดยถ้าใช้เครื่องหมาย + ระหว่างลีสต์ จะเป็นการรวมสมาชิกของลีสต์ทั้งสองเข้าด้วยกัน และใช้เครื่องหมาย * โดยถ้าลีสต์คูณด้วยตัวเลขจะเป็นการเพิ่มจำนวนเท่าของสมาชิกภายในลีสต์ ดังผลลัพธ์ที่ได้ของโปรแกรม

4. การเลื่อนลีสต์

การเลื่อนลีสต์ คือ การระบุตำแหน่งเพื่อแสดงข้อมูลภายในลีสต์ ซึ่งเพิ่มประสิทธิภาพในการเก็บข้อมูลในลีสต์ของภาษาไพธอน การเลื่อนลีสต์มีลักษณะการใช้งานคล้าย for...in range แต่จะใช้ใช้สัญลักษณ์ “:” หรือทวิภาค (ราชบัณฑิตยสถาน, 2564) โดยมีโครงสร้างของคำสั่งดังนี้

```
ลีสต์ [ ตำแหน่งเริ่มต้น : ตำแหน่งสุดท้าย : ระดับขั้น]
```

โครงสร้างการเลื่อนลีสต์ การเขียนโปรแกรม รูปแบบและลักษณะการทำงานดังโปรแกรมตัวอย่างที่ 5.8

ตัวอย่างที่ 5.8 การเลื่อนลีสต์

บรรทัดที่	คำสั่ง
1	lst = [1,2,3,4,'a','b','c','d']
2	print (lst[:5])
3	print (lst[3:])
4	print (lst[3:5])
5	print (lst[2:8:2])
6	print (lst[-1:-5:-1])
7	print (lst[:-3])
8	print (lst[-3:])
9	print (lst[-3:-3])
ผลลัพธ์	
[1, 2, 3, 4, 'a']	
[4, 'a', 'b', 'c', 'd']	
[4, 'a']	
[3, 'a', 'c']	

```
['d', 'c', 'b', 'a']
[1, 2, 3, 4, 'a']
['b', 'c', 'd']
[]
```

การระบุตำแหน่งเพื่อแสดงข้อมูลภายในลีสต์ ลักษณะเหมือนกับการเคลื่อนตำแหน่งข้อมูลภายในลีสต์ ซึ่งสามารถกำหนดรูปแบบการเลื่อนลีสต์ได้ทั้งการเลื่อนด้านหน้าและด้านหลังของลีสต์ ผลลัพธ์ที่ได้ดังโปรแกรมตัวอย่างที่ 5.8

5. การลบลีสต์

การลบลีสต์ คือ การทำงานของโปรแกรมในบางกรณีที่ต้องการลบข้อมูลที่ไม่ต้องการออกจากลีสต์ สามารถใช้คำสั่ง del ในการลบข้อมูลที่อยู่ภายในลีสต์ ซึ่งตำแหน่งของข้อมูลอ้างอิงเหมือนกับการเลื่อนลีสต์โดยใช้เครื่องหมาย “:” ในการระบุตำแหน่งที่ต้องการลบในลีสต์ ดังโปรแกรมตัวอย่างที่ 5.9

ตัวอย่างที่ 5.9 การลบลีสต์

บรรทัดที่	คำสั่ง
1	lst1 = [1,2,3,4,'a','b','c','d']
2	del (lst1[:])
3	print ("lst1[:] = ",lst1)
4	lst2 = [2,4,6,8,'w','x','y','z']
5	del (lst2[3:6])
6	print ("lst2[3:6] = ", lst2)
7	lst3 = [9.6,0.2,print,-9.99,44,len,-0.56,'program']
8	del (lst3[4:])
9	print ("lst3[4:] = ",lst3)
10	lst4 = [1,'a',2,'b',3,'c',4,'d']
11	del (lst4[3:6])
12	print ("lst4[3:6] = ",lst4)
13	lst5 = ['a','b','c','d','e','f','g','h','i']
14	del (lst5[2:8:2])

```
15  print ("lst5[2:8:2] = ",lst5)
16  lst6 = [11,22,33,44,'e','f','g','h']
17  del (lst6[-1:-5:-1])
18  print ("lst6[-1:-5:-1] = ",lst6)
19  lst7 = [3,5,7,9,'aa','bb','cc','dd']
20  del (lst7[:-3])
21  print ("lst7[:-3] = ",lst7)
22  lst8 = [9.6,0.2,input,len,-9.99,44,'program',-0.56]
23  del (lst8[-3:])
24  print ("lst8[-3:] = ",lst8)
```

ผลลัพธ์

```
lst1[:] =  []
lst2[3:6] =  [2, 4, 6, 'y', 'z']
lst3[4:] =  [9.6, 0.2, <built-in function print>, -9.99]
lst4[3:6] =  [1, 'a', 2, 4, 'd']
lst5[2:8:2] =  ['a', 'b', 'd', 'f', 'h', 'i']
lst6[-1:-5:-1] =  [11, 22, 33, 44]
lst7[:-3] =  ['bb', 'cc', 'dd']
lst8[-3:] =  [9.6, 0.2, <built-in function input>, <built-in function len>, -9.99]
```

โครงงสร้างในการเก็บข้อมูลและรูปแบบวิธีการเข้าถึงข้อมูลภายในลีสต์ประกอบด้วย การประมวลผลภายในลีสต์ การเลื่อนลีสต์ การลบลีสต์และการนำคำสั่งเพื่อดำเนินการข้อมูลภายในลีสต์ การนำวิธีดังกล่าวมาเขียนโปรแกรมต้องมีความเข้าใจหลักการใช้งานภายในตัวแปรลีสต์ เพื่อให้ได้ผลลัพธ์ของข้อมูลตามที่ต้องการ

ฟังก์ชันที่เกี่ยวข้องกับลีสต์

การนำข้อมูลมาใช้งานมีความสำคัญมากในการเขียนโปรแกรม ลักษณะของการเก็บข้อมูลภายในลีสต์มีความยืดหยุ่นโดยสามารถเก็บข้อมูลได้หลายชนิดในตัวแปรลีสต์เดียว ซึ่งถือว่าเป็นข้อดีต่อการเก็บข้อมูล พร้อมทั้งมีรูปแบบและคำสั่งในการเข้าถึงข้อมูลได้หลายประเภท ดังนั้น จึงมีฟังก์ชันที่สนับสนุนและช่วยอำนวยความสะดวกในการจัดการข้อมูลภายในลีสต์ ซึ่งมีรายละเอียดแต่ละฟังก์ชันการทำงานดังต่อไปนี้

1. การเพิ่มข้อมูลในลีสต์

คำสั่ง append() คือ ฟังก์ชันที่ใช้ในการเพิ่มข้อมูลเข้าไปในลีสต์ โดยจะเพิ่มเข้าไปในส่วนท้ายของลีสต์ มีโครงสร้างรูปแบบฟังก์ชัน “ชื่อลีสต์.append(ค่าข้อมูล)” การใช้คำสั่งมีรายละเอียดดังโปรแกรมตัวอย่างที่ 5.10

ตัวอย่างที่ 5.10 การใช้ฟังก์ชัน append()

บรรทัดที่	คำสั่ง
1	lst = ["Python","Programming"]
2	lst.append("Language")
3	print(lst)
ผลลัพธ์	
['Python', 'Programming', 'Language']	

2. การขยายลีสต์

คำสั่ง extend() คือ ฟังก์ชันที่ใช้ในการขยายลีสต์ หรือการดึงข้อมูลจากลีสต์อื่นให้เข้ามาในลีสต์ มีรูปแบบฟังก์ชัน “ชื่อลีสต์.extend(ชื่อลีสต์)” ดังโปรแกรมตัวอย่างที่ 5.11

ตัวอย่างที่ 5.11 การใช้ฟังก์ชัน extend()

บรรทัดที่	คำสั่ง
1	lst1 = ["Python","Programming"]
2	lst2 = ["Language","of"]
3	lst3 = ["Computer"]
4	lst1.extend(lst2)
5	print ("lst1 extend lst2 = ",lst1)
6	lst1.extend(lst3)
7	print ("lst1 entend lst3 = ",lst1)
ผลลัพธ์	
lst1 extend lst2 = ['Python', 'Programming', 'Language', 'of']	
lst1 entend lst3 = ['Python', 'Programming', 'Language', 'of', 'Computer']	

3. การรวมลิสต์

คำสั่ง insert() คือ ฟังก์ชันที่ใช้ในการดึงข้อมูลในลิสต์อื่นเข้ามา โดยระบุตำแหน่งในลิสต์หลักและลิสต์ที่ต้องการนำเข้า มีรูปแบบฟังก์ชัน “ชื่อลิสต์.insert(ตำแหน่ง, ชื่อลิสต์)” ดังโปรแกรมตัวอย่างที่ 5.12

ตัวอย่างที่ 5.12 การใช้ฟังก์ชัน insert()

บรรทัดที่	คำสั่ง
1	lst1 = ["Python","Programming"]
2	lst2 = ["Language","of"]
3	lst3 = ["Computer"]
4	print (lst1)
5	print (lst2)
6	print (lst3)
7	lst1.insert(1,lst2)
9	print ("lst1.insert(1,lst2) = ",lst1)
10	lst1.insert(3,lst3)
11	print ("lst1.insert(3,z) = ",lst1)
ผลลัพธ์	
['Python', 'Programming'] ['Language', 'of'] ['Computer'] lst1.insert(1,lst2) = ['Python', ['Language', 'of'], 'Programming'] lst1.insert(3,z) = ['Python', ['Language', 'of'], 'Programming', ['Computer']]	

4. การลบข้อมูลในลิสต์

คำสั่ง remove() คือ ฟังก์ชันที่ทำหน้าที่ในการลบข้อมูลในลิสต์ ตามข้อมูลที่ระบุในฟังก์ชัน โดยมีรูปแบบฟังก์ชัน “ชื่อลิสต์.remove(ค่าข้อมูล)” ดังโปรแกรมตัวอย่างที่ 5.13

ตัวอย่างที่ 5.13 การใช้ฟังก์ชัน remove()

บรรทัดที่	คำสั่ง
1	lst = ["Python","Programming","Language"]

2	lst.remove("Programming")
3	print (lst)
ผลลัพธ์	
['Python', 'Language']	

5. การดึงข้อมูลออกจากลิสต์

คำสั่ง pop() คือ ฟังก์ชันที่ทำหน้าที่ในการดึงข้อมูลออกในตำแหน่งที่ระบุในลิสต์ ซึ่งมีรูปแบบคำสั่ง “ชื่อลิสต์.pop(ตำแหน่งลิสต์)” ซึ่งมีรายละเอียดดังโปรแกรมตัวอย่างที่ 5.14

ตัวอย่างที่ 5.14 การใช้ฟังก์ชัน pop()

บรรทัดที่	คำสั่ง
1	lst = ["Python","Programming","Language"]
2	print (lst.pop())
3	print (lst)
4	lst.pop(0)
5	print (lst)
ผลลัพธ์	
Language ['Python', 'Programming'] ['Programming']	

6. การบอกตำแหน่งที่อยู่ในลิสต์

คำสั่ง index() คือ ฟังก์ชันที่ใช้ในการบอกตำแหน่งของข้อมูลที่อยู่ในลิสต์ โดยมีรูปแบบคำสั่ง “ชื่อลิสต์.index(ค่าข้อมูล)” ดังโปรแกรมตัวอย่างที่ 5.15

ตัวอย่างที่ 5.15 โปรแกรมการใช้ฟังก์ชัน index()

บรรทัดที่	คำสั่ง
1	lst = ["Python","Programming","Language"]
2	print (lst.index("Python"))
3	print (lst.index("Language"))
4	print (lst.index("Programming"))

ผลลัพธ์
0
2
1

7. การนับจำนวนที่ซ้ำกันภายในลีสต์

คำสั่ง count() คือ ฟังก์ชันที่ทำหน้าที่ในการนับจำนวนค่าของข้อมูลที่ซ้ำกันในลีสต์ โดยมีรูปแบบคำสั่ง “ชื่อลีสต์.count(ค่าข้อมูล)” ดังโปรแกรมตัวอย่างที่ 5.16

ตัวอย่างที่ 5.16 การใช้ฟังก์ชัน count()

บรรทัดที่	คำสั่ง
1	lst = ["Python","Programming","Language","Computer","Programming"]
2	print (lst.count("Programming"))
3	print (lst.count("Python"))
ผลลัพธ์	
2	
1	

8. การจัดเรียงข้อมูลที่อยู่ในลีสต์

คำสั่ง sort() คือ ฟังก์ชันที่ทำหน้าที่ในการจัดเรียงข้อมูลที่อยู่ในลีสต์ โดยสามารถเรียงข้อมูลได้ทั้งที่เป็นตัวเลข และข้อมูลที่เป็นตัวอักษร โดยมีรูปแบบฟังก์ชัน “ชื่อลีสต์.sort()” ดังโปรแกรมตัวอย่างที่ 5.17

ตัวอย่างที่ 5.17 การใช้ฟังก์ชัน count()

บรรทัดที่	คำสั่ง
1	lst1 = [1,8,9,6,2,3,2,4,9,0,7]
2	lst2 = ["Python","Programming","Language","Computer","Programming"]
3	lst1.sort()
4	lst2.sort()
5	print (lst1)
6	print (lst2)

ผลลัพธ์
[0, 1, 2, 2, 3, 4, 6, 7, 8, 9, 9] ['Computer', 'Language', 'Programming', 'Programming', 'Python']

9. การจัดเรียงกลับค่าข้อมูลในลีสต์

คำสั่ง reverse() คือ ฟังก์ชันที่ทำหน้าที่ในการจัดเรียงข้อมูลสลับกัน ให้ตรงกันข้ามกับข้อมูลในลีสต์เดิม โดยมีรูปแบบฟังก์ชัน “ชื่อลีสต์.reverse()” ดังโปรแกรมตัวอย่างที่ 5.18

ตัวอย่างที่ 5.18 การใช้ฟังก์ชัน reverse()

บรรทัดที่	คำสั่ง
1	lst1 = [1,8,9,6,2,3,2,4,9,0,7]
2	lst2 = ["Python","Programming","Language","Computer","Programming"]
3	lst1.reverse()
4	lst2.reverse()
5	print (lst1)
6	print (lst2)
ผลลัพธ์	
[7, 0, 9, 4, 2, 3, 2, 6, 9, 8, 1] ['Programming', 'Computer', 'Language', 'Programming', 'Python']	

ฟังก์ชันที่ใช้ในการจัดการข้อมูลภายในลีสต์ มีประโยชน์ต่อการนำไปเขียนโปรแกรม ซึ่งบางเหตุการณ์ที่ต้องการดำเนินการต่อข้อมูลภายในลีสต์ สามารถเรียกใช้ฟังก์ชันเหล่านี้มาใช้งานได้ทันที ซึ่งจะสะดวกและรวดเร็วต่อการนำไปใช้งานในโปรแกรม

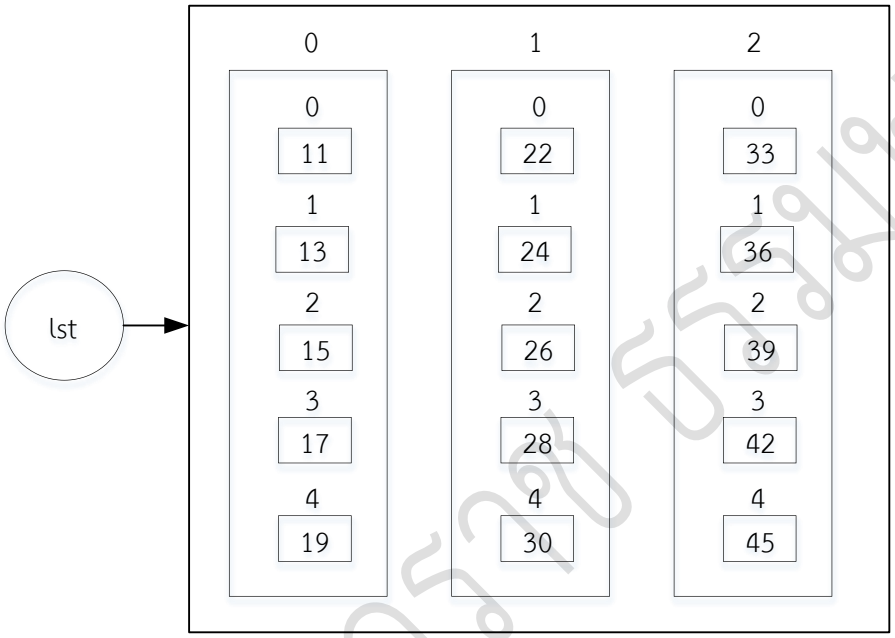
ลีสต์แบบหลายมิติ

การเก็บข้อมูลของลีสต์ในเบื้องต้น มีลักษณะของการเก็บข้อมูลที่เป็นแบบหนึ่งมิติ คือ การเก็บข้อมูลในลักษณะมุมมองแบบแนวนอน (Row) ซึ่งนอกเหนือจากการเก็บข้อมูลแบบแนวนอน ลีสต์ยังสามารถเก็บข้อมูลในลักษณะของแนวตั้ง (Column) และซ้อนเข้าไปในแนวตั้งลึกเข้าไปได้หลายชั้น การเก็บข้อมูลแบบนี้เรียกว่า ลีสต์แบบหลายมิติ (Multidimension) หรือการเก็บข้อมูลรูปแบบของเมทริกซ์ (Matrix) (Halterman, 2016)

ลีสต์ซ้อน (Nested Lists) คือ ลักษณะของลีสต์ที่ไปปรากฏเป็นส่วนประกอบของอีกลีสต์ ลีสต์ซ้อนบ่อยครั้งถูกนำเสนอรูปแบบของเมทริกซ์ (Elkner, 2014) มีรายละเอียดดังต่อไปนี้

1. ลีสต์แบบสองมิติ

การเก็บข้อมูลในรูปแบบของแนวนอนและแนวตั้ง เรียกว่า ลีสต์แบบ 2 มิติ รูปแบบในการเก็บข้อมูลดังตัวอย่างภาพที่ 5.4



ภาพที่ 5.4 โครงสร้างลีสต์แบบสองมิติ

โครงสร้างการเก็บข้อมูลลีสต์แบบสองมิติ เมื่อนำไปเขียนโปรแกรม มีรูปแบบและลักษณะการทำงาน ดังโปรแกรมตัวอย่างที่ 5.19

ตัวอย่างที่ 5.19 การเก็บข้อมูลลีสต์แบบสองมิติ

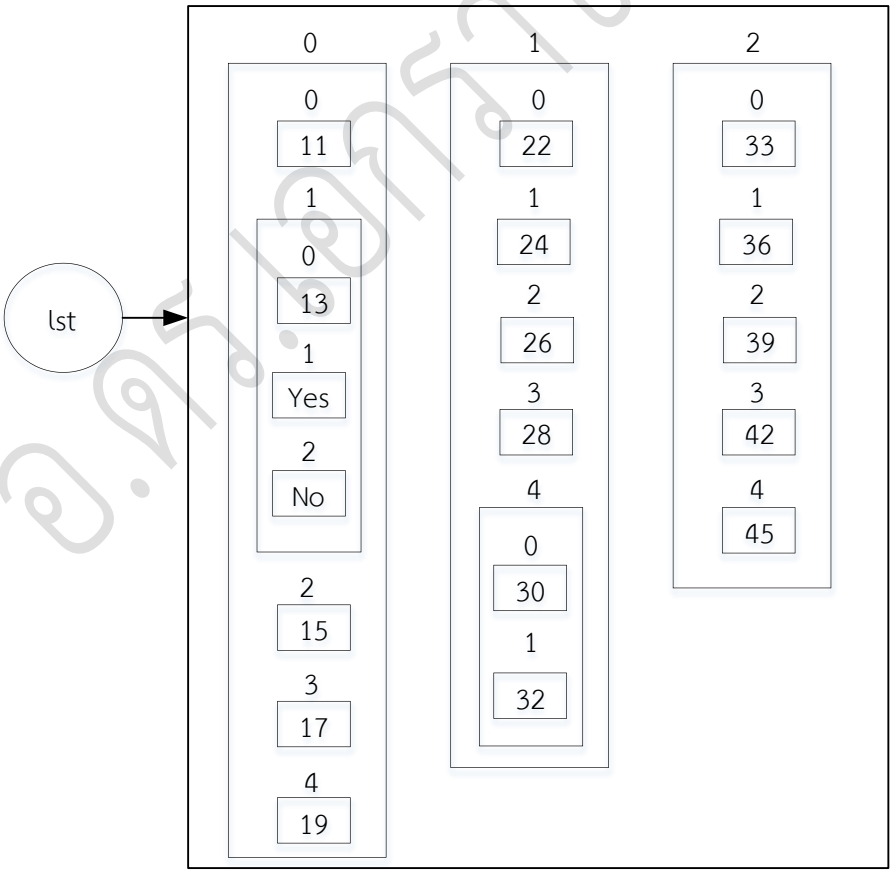
บรรทัดที่	คำสั่ง
1	matrix = [[11, 13, 15, 17,19],[22, 24, 26, 28, 30],[33, 36, 39, 42, 45]]
2	print(matrix[0][1])
3	print(matrix[1][3])
4	print(matrix[2][0])
5	print(matrix[2])
6	print(matrix[0])

ผลลัพธ์
13
28
33
[33, 36, 39, 42, 45]
[11, 13, 15, 17, 19]

โปรแกรมตัวอย่างที่ 5.19 การเรียกสมาชิกที่อยู่ในลีสต์แบบสองมิติ จะต้องระบุสองส่วน ประกอบไปด้วยส่วนแรก คือ ตำแหน่งของแถวในลีสต์ และส่วนที่สอง คือ ตำแหน่งคอลัมน์ที่อยู่ในลีสต์ เช่น บรรทัดที่ 2 ระบุตำแหน่งแถวที่ 0 คอลัมน์ที่ 1 ซึ่งข้อมูลที่อยู่ในตำแหน่งนี้คือ 13 แต่ถ้าเป็นการระบุส่วนเดียวในการเรียกข้อมูลในลีสต์ จะเป็นตำแหน่งแถวตำแหน่งเดียวเท่านั้นดังบรรทัดที่ 5

2. ลีสต์แบบสามมิติ

ลีสต์แบบสามมิติ คือ มุมมองการเก็บข้อมูลนอกเหนือจากข้อมูลแบบแนวนอนและแนวตั้ง ยังสามารถแทรกข้อมูลซ้อนลึกเข้าไปในแนวตั้งได้ไม่จำกัด มีรูปแบบของการเก็บข้อมูลดังภาพที่ 5.5



ภาพที่ 5.5 โครงสร้างลีสต์แบบสามมิติ

โครงสร้างการเก็บข้อมูลของลิสต์แบบสามมิติภาพที่ 5.5 สามารถนำไปเขียนโปรแกรมมีลักษณะและรูปแบบการทำงาน ดังโปรแกรมตัวอย่างที่ 5.20

ตัวอย่างที่ 5.20 ลิสต์แบบสามมิติ

บรรทัดที่	คำสั่ง
1	matrix = [[11, [13,'Yes','No'], 15, 17,19],[22, 24, 26, 28, [30,32]],[33, 36, 39, 42, 45]]
2	print(matrix[2])
3	print(matrix[0][1])
4	print(matrix[1][4])
5	print(matrix[0][1][1])
6	print(matrix[1][4][-1])
ผลลัพธ์	
[33, 36, 39, 42, 45] [13, 'Yes', 'No'] [30, 32] Yes 32	

การเก็บข้อมูลภายในลิสต์ในลักษณะของหลายมิติ มีหลักการเดียวกับการสร้างลิสต์พื้นฐาน นั่นคือ สามารถมีรูปแบบข้อมูลที่แตกต่างกันได้ การเรียกดูข้อมูลภายในลิสต์ จะมีลักษณะของการระบุจำนวนมิติที่เข้าไปภายในลิสต์ ดังโปรแกรมตัวอย่างที่ 5.20 จะเห็นได้ว่าลิสต์แบบหนึ่งมิติจะระบุตำแหน่งลิสต์ตำแหน่งเดียว(แนวนอน) ลิสต์สองมิติจะระบุสองตำแหน่ง (แนวนอนและแนวตั้ง) และลิสต์สามมิติจะระบุสามตำแหน่ง (แนวนอน แนวตั้งและแนวลึก)

ดังนั้นจะเห็นได้ว่า การสร้างลิสต์นั้นสามารถสร้างได้หลายมิติ ขึ้นอยู่กับการนำไปใช้ การระบุตำแหน่งภายในลิสต์ที่อยู่ในระดับล่างสุด จะต้องระบุตามจำนวนมิติของลิสต์นั้น การเก็บข้อมูลประเภทลิสต์ ถือว่ามีความสำคัญในการนำไปใช้การเก็บข้อมูล การเก็บในลักษณะนี้จะเพิ่มประสิทธิภาพ สะดวกในการจัดการและการเรียกใช้ข้อมูล ซึ่งเป็นเครื่องมือที่สำคัญของภาษาไพธอน

สรุป

ลีสต์ คือ โครงสร้างการเก็บข้อมูลเป็นแบบแถวลำดับแบบแนวนอน โดยลักษณะชุดข้อมูลที่ถูกเก็บ สามารถเป็นข้อมูลที่เหมือนกันหรือต่างชนิดกันได้ เช่น ตัวเลข ตัวอักษร คำสั่งและฟังก์ชัน โครงสร้างและข้อมูลภายลีสต์สามารถปรับเปลี่ยนได้ตามความต้องการ การใช้งานลีสต์ ประกอบด้วย การดำเนินการภายในลีสต์ เช่น การหาขนาดของลีสต์ การอ่านข้อมูลลีสต์ด้วยคำสั่งวนซ้ำ การประมวลผลของลีสต์ การเลื่อนลีสต์และการลบลีสต์ นอกจากนี้ยังมีฟังก์ชันที่ช่วยในการจัดการและการเข้าถึงข้อมูลภายในลีสต์เป็นจำนวนมาก ได้แก่ การหาขนาดของลีสต์ การขยายลีสต์ การรวมลีสต์ การลบข้อมูลในลีสต์ เป็นต้น การเก็บข้อมูลภายในลีสต์มีโครงสร้างการเก็บข้อมูลนอกเหนือจากการเก็บที่เป็นแบบแถวลำดับแบบแนวนอนหรือลีสต์แบบหนึ่งมิติ ยังสามารถเก็บข้อมูลในแบบแนวตั้งเรียกว่า ลีสต์แบบสองมิติ และเก็บข้อมูลแบบแนวลึก เรียกว่า ลีสต์แบบสามมิติ ซึ่งสามารถเก็บข้อมูลได้ไม่จำกัดเป็นจำนวนมาก เห็นได้ว่าลีสต์มีความยืดหยุ่นในการเก็บข้อมูลและมีโครงสร้างการเก็บข้อมูลได้ปริมาณมาก รองรับกับการนำไปใช้ในการจัดการข้อมูลที่มีความซับซ้อน จึงทำให้โปรแกรมที่ได้มีประสิทธิภาพในการทำงาน

แบบฝึกหัด

1. ให้ผู้เรียนอธิบายความหมายและลักษณะการทำงานของลีสต์
2. ให้ผู้เรียนอธิบายโครงสร้างการเก็บข้อมูลและวิธีการอ้างอิงระบุตำแหน่งข้อมูลภายในลีสต์
3. ให้ผู้เรียนอธิบายวิธีการดำเนินการภายในลีสต์แต่ละรูปแบบมีลักษณะอย่างไร
4. ให้ผู้เรียนอธิบายการประมวลภายในลีสต์มีลักษณะอย่างไร และฟังก์ชันที่ใช้ในการประมวลผลภายในลีสต์มีฟังก์ชันอะไรบ้าง แต่ละฟังก์ชันมีการทำงานอย่างไร
5. ให้ผู้เรียนอธิบายโครงสร้างการเก็บข้อมูลลีสต์แบบหนึ่งมิติ สองมิติ และหลายมิติ แต่ละรูปแบบมีการเก็บข้อมูลและการอ้างอิงระบุตำแหน่งข้อมูลภายลีสต์อย่างไร
6. ให้ผู้เรียนอธิบายการใช้งานลีสต์แบบหนึ่งมิติ สองมิติและสามมิติ มีความแตกต่างและประโยชน์ของการนำไปใช้งานอย่างไร
7. ให้ผู้เรียนยกตัวอย่างการเก็บข้อมูลลีสต์แบบสองมิติและสามมิติ พร้อมทั้งอธิบายเรียกข้อมูลภายในลีสต์แต่ละประเภท
8. ให้ผู้เรียนเขียนโปรแกรมให้มีการรับค่าและนำค่าที่ได้ไปเก็บไว้ในลีสต์แบบหนึ่งมิติ สองมิติและสามมิติ พร้อมทั้งแสดงข้อมูลภายในลีสต์แต่ละประเภท
9. ให้ผู้เรียนเขียนโปรแกรมรับข้อมูลจำนวนนักกีฬาและอายุนักกีฬาแต่ละคนไว้ในลีสต์ พร้อมทั้งหาอายุเฉลี่ยของนักกีฬา
10. ให้ผู้เรียนเขียนโปรแกรมให้นำตัวเลขคี่ระหว่าง 1 ถึง 25 ลงในลีสต์โดยใช้คำสั่ง for พร้อมทั้งแสดงผลลัพธ์ที่ได้

เอกสารอ้างอิง

จักรกฤษณ์ แสงแก้ว. (2549). **การเขียนโปรแกรมด้วยภาษาไพธอนด้วยตนเอง**. กรุงเทพฯ: สมาคมส่งเสริมเทคโนโลยี (ไทย-ญี่ปุ่น).

โชติพันธุ์ หล่อเลิศสุนทร และ ฐิตะพันธุ์ หล่อเลิศสุนทร. (2559). **คู่มือเรียนเขียนโปรแกรม python (ภาคปฏิบัติ)**. กรุงเทพฯ: คอร์ฟังก์ชั่น.

ราชบัณฑิตยสภา. ศัพท์บัญญัติสำนักงานราชบัณฑิตยสภา. (ออนไลน์) 10 เมษายน 2564 (อ้างเมื่อ 10 เมษายน 2564). จาก: https://coined-word.orst.go.th/

Elkner, Jeffrey. (2014). **Practical Programming (in Python)**. 3rd ed. London: Pearson.

Fior, James M. (2016). **Computer Programming with Python and Multisim**. 3rd ed. London: Pearson.

Halterman, Richard L. (2016). **Fundamentals of Python Programming**. Tennessee: Southern Adventist University.