สยามทูเดฟ

http://www.siam2dev.com

รายวิชา การเขียนโปรแกรมเชิงวัตถุ OOP

บทที่ 4 ตัวแปร ชนิดข้อมูล และตัวดำเนินการ

โดย ผู้ช่วยศาสตราจารย์ ดร. นัฐพงศ์ ส่งเนียม

สาขาวิชาวิทยาการคอมพิวเตอร์

คณะวิทยาศาสตร์และเทคโนโลยี มหาวิทยาลัยราชภัฏพระนคร

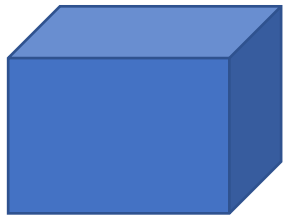
Agenda

ในบทนี้ ผู้เรียนจะได้เรียนรู้เกี่ยวกับตัวแปรและประเภทข้อมูลในภาษาไพธอน โดยจะกล่าวถึงการประกาศตัวแปรและการนำตัวแปรไปใช้งานในโปรแกรม และเราจะอธิบายถึงข้อมูลประเภทต่าง ๆ ที่เป็นประเภทข้อมูลพื้นฐาน (Primitive data type) ในภาษาไพธอน และรวมทั้งฟังก์ชันสำหรับการใช้งานกับตัวแปร โดยมีรายละเอียดดังนี้

4.1 ความหมายของตัวแปร
4.2 ตัวแปรในภาษาไพธอน
4.3 การกำหนดค่าตัวแปรหลายค่าในคำสั่งเดียว
4.4 ชนิดของข้อมูลในภาษาไพธอน
4.5 ฟังก์ชันที่ใช้งานกับตัวแปร
4.6 อักขระพิเศษในภาษาไพธอน
4.7ตัวดำเนินการในภาษาไพธอน

4.1 ความหมายของตัวแปร

ภาษาไพธอนถูกพัฒนาขึ้นมาโดยมีความตั้งใจว่าจะให้เป็นภาษาที่อ่านง่าย มันถูกออกแบบมาให้มีโครงสร้างที่ไม่ซับซ้อน โดยมักจะใช้คำในภาษาอังกฤษในขณะที่ภาษาอื่นใช้เครื่องหมายและวรรคตอน นอกจากนี้ Python ยังมีข้อยกเว้นของโครงสร้างทางภาษาน้อยกว่าอื่น ๆ อย่างภาษา C และ Pascal3



ชื่อที่ตั้งขึ้นเพื่อให้ตัวแปลภาษา ทาการจองพื้นที่ในหน่วยจำ สำหรับเก็บข้อมูลหรือเพื่อทำงานในโปรแกรม

4.1 ความหมายของตัวแปร

ตัวแปร (Variable) คือชื่อหรือเครื่องหมายที่กำหนดขึ้นสำหรับใช้อ้างถึงค่าที่เก็บในหน่วยความจำ ตัวแปรจะมีชื่อ (Identifier) สำหรับใช้ในการอ้างถึงข้อมูลของมัน ในการเขียนโปรแกรม ค่าของตัวแปรสามารถที่จะกำหนดได้ใน run-time หรือเปลี่ยนแปลงอยู่ตลอดเวลาในขณะที่โปรแกรมทำงาน (Executing)

ในการเขียนโปรแกรมคอมพิวเตอร์นั้น ตัวแปรจะแตกต่างจากตัวแปรในทางคณิตศาสตร์ ค่าของตัวแปรนั้นไม่จำเป็นต้องประกอบไปด้วยสูตรหรือสมการที่สมบูรณ์เหมือนกับในคณิตศาสตร์ ในคอมพิวเตอร์ ตัวแปรนั้นอาจจะมีการทำงานซ้ำ ๆ เช่น การกำหนดค่าในที่หนึ่ง และนำไปใช้อีกที่หนึ่งในโปรแกรม และนอกจากนี้ยังสามารถกำหนดค่าใหม่ให้กับตัวแปรได้ตลอดเวลา ต่อไปเป็นตัวอย่างของการประกาศตัวแปรในภาษาพธอน Python

4.1 ความหมายของตัวแปร

ตัวแปร (Variable) คือชื่อหรือเครื่องหมายที่กำหนดขึ้นสำหรับใช้อ้างถึงค่าที่เก็บในหน่วยความจำ ตัวแปรจะมีชื่อ (Identifier) สำหรับใช้ในการอ้างถึงข้อมูลของมัน ในการเขียนโปรแกรม ค่าของตัวแปรสามารถที่จะกำหนดได้ใน run-time หรือเปลี่ยนแปลงอยู่ตลอดเวลาในขณะที่โปรแกรมทำงาน (Executing)

ในการเขียนโปรแกรมคอมพิวเตอร์นั้น ตัวแปรจะแตกต่างจากตัวแปรในทางคณิตศาสตร์ ค่าของตัวแปรนั้นไม่จำเป็นต้องประกอบไปด้วยสูตรหรือสมการที่สมบูรณ์เหมือนกับในคณิตศาสตร์ ในคอมพิวเตอร์ ตัวแปรนั้นอาจจะมีการทำงานซ้ำ ๆ เช่น การกำหนดค่าในที่หนึ่ง และนำไปใช้อีกที่หนึ่งในโปรแกรม และนอกจากนี้ยังสามารถกำหนดค่าใหม่ให้กับตัวแปรได้ตลอดเวลา ต่อไปเป็นตัวอย่างของการประกาศตัวแปรในภาษาพธอน Python

4.2 ตัวแปรในภาษาไพธอน

ในภาษาไพธอน นั้นสนับสนุนการกำหนดค่าให้กับตัวแปรหลายค่าในคำสั่งเดียว ในตัวอย่าง เป็นการกำหนดค่า 1 และ 2 กับตัวแปร a และ b ตามลำดับ และในคำสั่งต่อมาเป็นการกำหนดค่า 10 ให้กับตัวแปร x y และ z ซึ่งทำให้การเขียนโปรแกรมสะดวกและรวดเร็วมากขึ้น

```
a = 1
b = 2
x = 10
y = 10
z = 10
```

VS_Code_Projects > PL1 > Ch04_01.py > ...

```
1  a = 3
2  b = 4.92
3  c = "marcuscode.com"
4  c = 10.5
```

เนื้อหาส่วนใหญ่อ้างอิงจากเว็บไซต์ :

4.2 ตัวแปรในภาษาไพธอน

Test_var01.py

```
# ตัวแปรชนิดข้อมูลจำนวนเต็ม (integer)
age = 25

# ตัวแปรชนิดข้อมูลทศนิยม (float)
height = 175.5

# ตัวแปรชนิดข้อมูลสตริง (string)
name = “Asst. Prof. Dr. Nattapong Songnean"
```

print()

4.2 ตัวแปรในภาษาไพธอน

Test_var01.py

```python
# Project Name : Test_var01.py
# Developer    : Asst. Prof. Dr. Nattapong Songneam
# Create Date  : 05.08.2566
# Modify Date  : 05.08.2566
#-------------------------------------------------------------
# ตัวแปรชนิดข้อมูลจำนวนเต็ม (integer)
age = 25

# ตัวแปรชนิดข้อมูลทศนิยม (float)
height = 175.5

# ตัวแปรชนิดข้อมูลสตริง (string)
name = "Asst. Prof. Dr. Nattapong Songnean"
print(age,height,name)

```

```
PROBLEMS  OUTPUT  DEBUG CONSOLE  TERMINAL

PS D:\OOP_1_2566> & C:/Users/User/AppData/Local/Microsoft/WindowsApps/python3.10.exe d:/OOP_1_2566/Test_Var01.py
25 175.5 Asst. Prof. Dr. Nattapong Songnean
PS D:\OOP_1_2566>
```

4.3 การกำหนดค่าตัวแปรหลายค่าในคำสั่งเดียว

ในภาษาไพธอน นั้นสนับสนุนการกำหนดค่าให้กับตัวแปรหลายค่าในคำสั่งเดียว ในตัวอย่างเป็นการกำหนดค่า 1 และ 2 กับตัวแปร a และ b ตามลำดับ และในคำสั่งต่อมาเป็นการกำหนดค่า 10 ให้กับตัวแปร x y และ z ซึ่งทำให้การเขียนโปรแกรมสะดวกและรวดเร็วมากขึ้น

Test_var02.py

```
a, b = 1, 2
x = y = z = 10
print("a = " , a)
print("b = " , b)
print("x = " , x)
print("y = " , y)
print("z = " , z)
```

ผลลัพธ์โปรแกรม

```
a =  1
b =  2
x =  10
y =  10
z =  10
```

4.3 การกำหนดค่าตัวแปรหลายค่าในคำสั่งเดียว

ต.ย.

Test_var03.py

```
a, b = 2, 1
x = y = z = 5
print("a = " , a)
print("b = " , b)
print("x = " , x)
print("y = " , y)
print("z = " , z)
```

ผลลัพธ์โปรแกรม

```
a =  ..
b =  ..
x =  ..
y =  ..
z =  ..
```

4.3 การกำหนดค่าตัวแปรหลายค่าในคำสั่งเดียว

ต.ย.

Test_var03.py

```
a, b ,c= 2, 1 , 4
x = y = z = 5
print("a = " , a)
print("b = " , b)
print("x = " , x)
print("y = " , y)
print("z = " , z)
```

ผลลัพธ์โปรแกรม

```
a =  ..
b =  ..
x =  ..
y =  ..
z =  ..
```

4.4 ชนิดข้อมูลในภาษาไพธอน

ภาษาไพธอนสนับสนุนชนิดข้อมูลหลายชนิดด้วยกัน ซึ่งโดยทั่วไปจะมีอยู่สามประเภทใหญ่ ๆ ได้แก่ 1) ข้อมูลประเภทตัวเลข นั้นจะแบ่งย่อยออกเป็น Integer และ Float 2) ข้อมูลประเภทสตริง (String) และ 3) ข้อมูลประเภทลำดับ เช่น List และ Tuple ประเภทข้อมูลทั้งหมดนี้เป็น Built-in type ในภาษาไพธอน

ชนิดข้อมูล (Data Type)	ประเภท (Size)	คำอธิบาย (Description)
Integer	จำนวนเต็ม	ใช้เก็บตัวเลขจำนวนเต็ม เช่น I = 7 X = 1 A = 0
float	ทศนิยม	ใช้เก็บตัวเลขที่มีทศนิยม เช่น I = 2.7 X = 0.9 A = 0.0
Character	อักขระ	ใช้เก็บอักขระเพียงตัวเดียว เช่น A = 'Y' K = 'N'
String	ข้อความ	ใช้เก็บในลักษณะข้อความ เช่น Name = 'Nattapong' Status = 'Teacher'
boolean	จริง/เท็จ	ใช้เก็บข้อมูลลักษณะจริงเท็จ X = True Y = False

4.4 ชนิดข้อมูลในภาษาไพธอน

ในภาษาไพธอน (Python) มีชนิดข้อมูลหลากหลายประเภท บางส่วนในนั้นเป็น:
1. จำนวนเต็ม (Integer): เป็นตัวเลขที่ไม่มีทศนิยม เช่น 1, 100, -5, 0 เป็นต้น
2. จำนวนจริง (Float): เป็นตัวเลขที่มีทศนิยม เช่น 3.14, -2.5, 0.75 เป็นต้น
3. สตริง (String): เป็นข้อความที่อยู่ภายในเครื่องหมายคำพูด ("") หรือ ('') เช่น "Hello World", 'Python Programming' เป็นต้น
4. บูลีน (Boolean): เป็นค่าความจริง (True หรือ False) เช่น True, False
5. รายการ (List): เป็นชุดของค่าที่เก็บอยู่ในเครื่องหมายวงเล็บ [ ] แยกด้วยเครื่องหมายจุลภาค , เช่น [1, 2, 3], ['apple', 'banana', 'orange'] เป็นต้น
6. สาธารณะ (Tuple): เป็นชุดของค่าที่เก็บอยู่ในเครื่องหมายวงเล็บ ( ) แยกด้วยเครื่องหมายจุลภาค , เช่น (1, 2, 3), ('apple', 'banana', 'orange') เป็นต้น
7. ความสัมพันธ์ (Set): เป็นชุดของค่าที่ไม่มีลำดับและไม่ซ้ำกัน เก็บอยู่ในเครื่องหมายวงเล็บเหลี่ยม { } แยกด้วยเครื่องหมายจุลภาค , เช่น {1, 2, 3}, {'apple', 'banana', 'orange'} เป็นต้น
8. พจนานุกรม (Dictionary): เป็นชุดของคู่คีย์-ค่าที่เก็บอยู่ในเครื่องหมายวงเล็บเหลี่ยม { } โดยคีย์และค่าคือคู่กัน เช่น {'name': 'John', 'age': 30, 'gender': 'male'} เป็นต้น

เหตุผลที่ Python ถูกให้ความสำคัญในการเขียนโปรแกรมคือ มีลักษณะที่ยืดหยุ่นและเป็นภาษาที่เข้าใจง่าย ชนิดข้อมูลต่าง ๆ ใน Python ช่วยให้นักพัฒนาสามารถจัดการและประมวลผลข้อมูลอย่างหลากหลายและเข้าถึงสิ่งที่ต้องการได้ง่ายขึ้น

4.4 ชนิดข้อมูลในภาษาไพธอน

จำนวนเต็ม (Integer):

```
x = 10
y = -5
z = 0
```

```
# ตัวแปรสามารถรับค่าจากตัวแปรอื่น ๆ ได้
x = 10
y = x  # y จะมีค่าเท่ากับ x คือ 10
```

จำนวนจริง (Float):

```
pi = 3.14
temperature = 25.5
```

4.4 ชนิดข้อมูลในภาษาไพธอน

สตริง (String):

```
name = “Nattapong Songneam"
message = 'Hello, World!'
```

บูลีน (Boolean):

```
is_raining = True
is_sunny = False
found = false
```

= assign value operator

4.4 ชนิดข้อมูลในภาษาไพธอน

รายการ (List):

```
fruits = ['apple', 'banana', 'orange']
numbers = [1, 2, 3, 4, 5]
Tall = [120.5, 145.4, 178.5]
```

Test_var04_list.py

สาธารณะ (Tuple):

```
coordinates = (10, 20)
colors = ('red', 'green', 'blue')
```

4.4 ชนิดข้อมูลในภาษาไพธอน

Test_var04_list.py

รายการ (List):

List ใน Python เป็นโครงสร้างข้อมูลที่ใช้เก็บข้อมูลหลายๆ ชิ้นในตำแหน่งที่เรียกว่า index โดยเริ่มจาก 0 เป็นต้นไป โดยสามารถใช้งาน List ใน Python ได้หลากหลายวิธี เช่น การสร้าง List, การเข้าถึงข้อมูลใน List, การแก้ไขข้อมูลใน List ฯลฯ ดังตัวอย่างด้านล่าง

การประกาศค่า List

```
#-------------------------------------------------------------
my_list = [1, 2, 3, 4, 5, 6, 7, 8, 9, 10]
fruits = ['apple', 'banana', 'orange']
mixed_data = [1, 'hello', 3.14, True]
```

```
#-------------------------------------------------------------
print(my_list[0])  # ผลลัพธ์: 1
print(fruits[1])   # ผลลัพธ์: banana
```

```
#-------------------------------------------------------------
my_list[2] = 10
fruits[0] = 'grape'
print(my_list)  # ผลลัพธ์: [1, 2, 10, 4, 5]
print(fruits)   # ผลลัพธ์: ['grape', 'banana', 'orange']
```

4.4 ชนิดข้อมูลในภาษาไพธอน

รายการ (List):

List ใน Python เป็นโครงสร้างข้อมูลที่ใช้เก็บข้อมูลหลายๆ ชิ้นในตำแหน่งที่เรียกว่า index โดยเริ่มจาก 0 เป็นต้นไป โดยสามารถใช้งาน List ใน Python ได้หลากหลายวิธี เช่น การสร้าง List, การเข้าถึงข้อมูลใน List, การแก้ไขข้อมูลใน List ฯลฯ ดังตัวอย่างด้านล่าง

Test_var04_list.py

```
#-------------------------------------------------------------
my_list = [1, 2, 3, 4, 5, 6, 7, 8, 9, 10]
fruits = ['apple', 'banana', 'orange']
mixed_data = [1, 'hello', 3.14, True]
```

....

```
#-------------------------------------------------------------
fruits.append('strawberry')
print(fruits)  # ผลลัพธ์: ['grape', 'banana', 'orange', 'strawberry']
```

การใช้งาน append

4.4 ชนิดข้อมูลในภาษาไพธอน

รายการ (List):

List ใน Python เป็นโครงสร้างข้อมูลที่ใช้เก็บข้อมูลหลายๆ ชิ้นในตำแหน่งที่เรียกว่า index โดยเริ่มจาก 0 เป็นต้นไป โดยสามารถใช้งาน List ใน Python ได้หลากหลายวิธี เช่น การสร้าง List, การเข้าถึงข้อมูลใน List, การแก้ไขข้อมูลใน List ฯลฯ ดังตัวอย่างด้านล่าง

Test_var04_list.py

```
#-------------------------------------------------------------
my_list = [1, 2, 3, 4, 5, 6, 7, 8, 9, 10]
fruits = ['apple', 'banana', 'orange']
mixed_data = [1, 'hello', 3.14, True]
```

....

การใช้งาน Loop

```
#-------------------------------------------------------------
for fruit in fruits:
    print(fruit)
# ผลลัพธ์:
# grape
# banana
# orange
# strawberry
```

4.4 ชนิดข้อมูลในภาษาไพธอน

Test_var04_list.py

รายการ (List):

List ใน Python เป็นโครงสร้างข้อมูลที่ใช้เก็บข้อมูลหลายๆ ชิ้นในตำแหน่งที่เรียกว่า index โดยเริ่มจาก 0 เป็นต้นไป โดยสามารถใช้งาน List ใน Python ได้หลากหลายวิธี เช่น การสร้าง List, การเข้าถึงข้อมูลใน List, การแก้ไขข้อมูลใน List ฯลฯ ดังตัวอย่างด้านล่าง

```
#-------------------------------------------------------------
my_list = [1, 2, 3, 4, 5, 6, 7, 8, 9, 10]
fruits = ['apple', 'banana', 'orange']
mixed_data = [1, 'hello', 3.14, True]
```

....

การตรวจสอบค่าใน List ด้วย in

```
#-------------------------------------------------------------
if 'banana' in fruits:
    print("มี 'banana' ในรายการผลไม้")
# ผลลัพธ์: มี 'banana' ในรายการผลไม้
if 'mango' in fruits:
    print("มี 'mango' ในรายการผลไม้")
else:
    print("ไม่มี 'mango' ในรายการผลไม้")
```

การใช้งานฟังก์ชันที่เกี่ยวข้องกับ List เช่น max(), min(), sum():

4.4 ชนิดข้อมูลในภาษาไพธอน

รายการ (List):

List ใน Python เป็นโครงสร้างข้อมูลที่ใช้เก็บข้อมูลหลายๆ ชิ้นในตำแหน่งที่เรียกว่า index โดยเริ่มจาก 0 เป็นต้นไป โดยสามารถใช้งาน List ใน Python ได้หลากหลายวิธี เช่น การสร้าง List, การเข้าถึงข้อมูลใน List, การแก้ไขข้อมูลใน List ฯลฯ ดังตัวอย่างด้านล่าง

Test_var04_list.py

```
#-------------------------------------------------------------
my_list = [1, 2, 3, 4, 5, 6, 7, 8, 9, 10]
fruits = ['apple', 'banana', 'orange']
mixed_data = [1, 'hello', 3.14, True]
```

....

```
#-------------------------------------------------------------
numbers = [5, 2, 8, 1, 9]
print(max(numbers))   # ผลลัพธ์: 9
print(min(numbers))   # ผลลัพธ์: 1
print(sum(numbers))   # ผลลัพธ์: 25
print(avg(numbers))   # ผลลัพธ์: ....
```

การใช้ฟังก์ชันที่เกี่ยวข้องกับ List เช่น max(), min(), sum()
การใช้โมดูลเสริม เช่น statistics

4.4 ชนิดข้อมูลในภาษาไพธอน

Test_var04_list.py

รายการ (List):

List ใน Python เป็นโครงสร้างข้อมูลที่ใช้เก็บข้อมูลหลายๆ ชิ้นในตำแหน่งที่เรียกว่า index โดยเริ่มจาก 0 เป็นต้นไป โดยสามารถใช้งาน List ใน Python ได้หลากหลายวิธี เช่น การสร้าง List, การเข้าถึงข้อมูลใน List, การแก้ไขข้อมูลใน List ฯลฯ ดังตัวอย่างด้านล่าง

```
#------------------------------------------------------------
my_list = [1, 2, 3, 4, 5, 6, 7, 8, 9, 10]
fruits = ['apple', 'banana', 'orange']
mixed_data = [1, 'hello', 3.14, True]
```

....

การใช้โมดูลเสริม เช่น statistics

```
import statistics

numbers = [5, 2, 8, 1, 9]
print(max(numbers))         # ผลลัพธ์: 9
print(min(numbers))         # ผลลัพธ์: 1
print(sum(numbers))         # ผลลัพธ์: 25
print(statistics.mean(numbers))   # ผลลัพธ์: 5.0
```

การใช้ฟังก์ชันที่เกี่ยวข้องกับ List เช่น max(), min(), sum():
การใช้โมดูลเสริม เช่น statistics

4.4 ชนิดข้อมูลในภาษาไพธอน

รายการ (List):

List ใน Python เป็นโครงสร้างข้อมูลที่ใช้เก็บข้อมูลหลายๆ ชิ้นในตำแหน่งที่เรียกว่า index โดยเริ่มจาก 0 เป็นต้นไป โดยสามารถใช้งาน List ใน Python ได้หลากหลายวิธี เช่น การสร้าง List, การเข้าถึงข้อมูลใน List, การแก้ไขข้อมูลใน List ฯลฯ ดังตัวอย่างด้านล่าง

การนิยามฟังก์ชัน avg() แบบกำหนดเอง

Test_var04_list.py

```
#------------------------------------------------------------
my_list = [1, 2, 3, 4, 5, 6, 7, 8, 9, 10]
fruits = ['apple', 'banana', 'orange']
mixed_data = [1, 'hello', 3.14, True]
```

....

```
def avg(numbers):
    total = sum(numbers)
    count = len(numbers)
    return total / count

numbers = [5, 2, 8, 1, 9]
print(max(numbers))  # ผลลัพธ์: 9
print(min(numbers))  # ผลลัพธ์: 1
print(sum(numbers))  # ผลลัพธ์: 25
print(avg(numbers))  # ผลลัพธ์: 5.0
```

การใช้ฟังก์ชันที่เกี่ยวข้องกับ List เช่น max(),min(),sum():
การใช้โมดูลเสริม เช่น statistics

4.4 ชนิดข้อมูลในภาษาไพธอน

Test_var04_list.py

การประยุกต์ใช้รายการ (List):

....

การจัดเก็บรายชื่อผู้ใช้งาน

```
users = ['Alice', 'Bob', 'Charlie', 'David']
```

การบันทึกตารางคะแนนของนักเรียน

```
scores = [85, 92, 78, 95, 88]
```

การจัดการรายการสินค้าในร้านค้า:

```
shopping_cart = ['apple', 'banana', 'chocolate', 'bread']
```

การใช้ฟังก์ชันที่เกี่ยวข้องกับ List เช่น max(), min(), sum():
การใช้โมดูลเสริม เช่น statistics

4.4 ชนิดข้อมูลในภาษาไพธอน

Test_var04_list.py

การประยุกต์ใช้รายการ (List):

....

การจัดการกิจกรรมที่ต้องทำในแต่ละวัน:

```
daily_tasks = ['exercise', 'work', 'study', 'relax']
```

การนับจำนวนคำศัพท์ที่ใช้ในหนังสือ:

```
book_words = ['apple', 'banana', 'orange', ...]  # รายการคำศัพท์ทั้งหมดในหนังสือ
total_words = len(book_words)
```

การเก็บรายชื่อผู้ลงทะเบียนงานอบรม:

```
workshop_attendees = []
workshop_attendees.append('Alice')
workshop_attendees.append('Bob')
workshop_attendees.append('Charlie')
```

การใช้โมดูลเสริม เช่น statistics

4.4 ชนิดข้อมูลในภาษาไพธอน

Test_var04_list.py

การประยุกต์ใช้รายการ (List):

....

การจัดการข้อมูลเส้นทางการเดินทาง:

```
route = ['start', 'checkpoint1', 'checkpoint2', 'destination']
```

การเข้าถึงและปรับปรุงข้อมูลในลิสต์:

```
colors = ['red', 'green', 'blue']
print(colors[1])  # ผลลัพธ์: green
colors[0] = 'yellow'
print(colors)  # ผลลัพธ์: ['yellow', 'green', 'blue']
```

การจัดเก็บประวัติการสั่งอาหาร:

```
order_history = [
    {'item': 'pizza', 'price': 12.99},
    {'item': 'burger', 'price': 8.49},
    {'item': 'salad', 'price': 5.99}
]
```

แบบฝึกหัดเรื่อง List

- 1. ให้สร้าง list เก็บรายชื่อนักศึกษา 10 คน พร้อมคะแนนของแต่ละคน โดยคำหาเกรด ของแต่ละคน

โดยใช้เกณฑ์

- 80 - 100 ให้เกรด A
- 70 - 79 ให้เกรด B
- 60 - 69 ให้เกรด C
- 50 -59 ให้เกรด D
- 0-49 ให้เกรด E
- ที่เหลือ Invalid

แบบฝึกหัดเรื่อง List

- 1. ให้สร้าง list เก็บรายชื่อนักศึกษา 10 คน พร้อมคะแนนของแต่ละคน โดยคำหาเกรด ของแต่ละคน

โดยใช้เกณฑ์

- 80 - 100 ให้เกรด A
- 70 - 79 ให้เกรด B
- 60 - 69 ให้เกรด C
- 50 -59 ให้เกรด D
- 0-49 ให้เกรด E
- ที่เหลือ Invalid

ดังนี้คือตัวอย่างการสร้าง List เก็บข้อมูลของนักศึกษา 10 คนพร้อมคะแนนและการคำนวณเกรดตามเกณฑ์ที่กำหนด:

```
students = [
    {'name': 'Alice', 'score': 92},
    {'name': 'Bob', 'score': 78},
    {'name': 'Charlie', 'score': 65},
    {'name': 'David', 'score': 45},
    {'name': 'Eve', 'score': 88},
    {'name': 'Frank', 'score': 72},
    {'name': 'Grace', 'score': 53},
    {'name': 'Helen', 'score': 95},
    {'name': 'Ivy', 'score': 61},
    {'name': 'Jack', 'score': 37}
]
```

```
def calculate_grade(score):
    if 80 <= score <= 100:
        return 'A'
    elif 70 <= score < 80:
        return 'B'
    elif 60 <= score < 70:
        return 'C'
    elif 50 <= score < 60:
        return 'D'
    elif 0 <= score < 50:
        return 'E'
    else:
        return 'Invalid'

for student in students:
    name = student['name']
    score = student['score']
    grade = calculate_grade(score)
    print(f"Student: {name}, Score: {score}, Grade: {grade}")
```

4.4 ชนิดข้อมูลในภาษาไพธอน

รายการ (List):

Test_var04_list.py

```python
# Project Name : Test_var04_list.py
# Developer    : Asst. Prof. Dr. Nattapong Songneam
# Create Date  : 05.08.2566
# Modify Date  : 05.08.2566
#-----------------------------------------------------
fruits = ['apple', 'banana', 'orange']
numbers = [1, 2, 3, 4, 5]
Tall = [120.5, 145.4, 178.5]
print(fruits)
print(numbers)
print(Tall)
```

```
PROBLEMS  OUTPUT  DEBUG CONSOLE  TERMINAL

PS D:\OOP_1_2566> & C:/Users/User/AppData/Local/Microsoft/WindowsApps/python3.10.exe d:/OOP_1_2566/Test_var04_list.py
['apple', 'banana', 'orange']
[1, 2, 3, 4, 5]
[120.5, 145.4, 178.5]
PS D:\OOP_1_2566>
```

4.4 ชนิดข้อมูลในภาษาไพธอน

ความสัมพันธ์ (Set):

```
unique_numbers = {1, 2, 3, 4, 5}
unique_letters = {'a', 'b', 'c'}
```

พจนานุกรม (Dictionary):

```
person = {'name': 'John', 'age': 30, 'gender': 'male'}
student = {'id': '12345', 'name': 'Jane', 'major': 'Computer Science'}
```

4.4 ชนิดข้อมูลในภาษาไพธอน

Set และ List เป็นโครงสร้างข้อมูลสองชนิดที่ใช้ในภาษาไพธอน โดยมีความแตกต่างกันดังนี้:

1.การเรียงลำดับข้อมูล:

- List: เป็นชนิดข้อมูลที่เรียงลำดับและสามารถเข้าถึงสมาชิกด้วยดัชนี (index) ซึ่งเริ่มต้นที่ 0 ถึง n-1 โดย n คือจำนวนสมาชิกใน List การเข้าถึงสมาชิกใน List สามารถทำได้ด้วยการใช้สัญลักษณ์ **[ ]** เช่น **my_list[0]** จะได้สมาชิกตัวแรกของ List.
- Set: เป็นชนิดข้อมูลที่ไม่เรียงลำดับและไม่มีการใช้ดัชนีในการเข้าถึงสมาชิก เนื่องจาก Set เก็บข้อมูลในลักษณะที่ไม่ซ้ำกัน การเข้าถึงสมาชิกใน Set ไม่สามารถทำได้โดยใช้ดัชนี เนื่องจากไม่มีการระบุตำแหน่งของสมาชิกใน Set เหมือนกับ List.

2.การซ้ำกันของสมาชิก:

- List: สามารถเก็บสมาชิกที่ซ้ำกันได้ หากมีสมาชิกใน List เหมือนกันหลายครั้ง จะถูกเก็บแยกตัวอย่างกันทั้งหมด.
- Set: เป็นโครงสร้างข้อมูลที่เก็บสมาชิกเฉพาะที่ไม่ซ้ำกัน หากมีสมาชิกที่ซ้ำกันใน Set จะถูกเก็บเพียงครั้งเดียวเท่านั้น ทำให้ Set มีคุณสมบัติในการลบสมาชิกที่ซ้ำกันออกจากกันอัตโนมัติ.

3.การใช้งาน:

- List: ใช้เมื่อต้องการเก็บข้อมูลที่ต้องการเรียงลำดับและอ้างอิงด้วยดัชนี เช่น รายการสินค้า, รายชื่อนักเรียน, ลำดับขั้นตอนในการทำงาน เป็นต้น.
- Set: ใช้เมื่อต้องการเก็บข้อมูลที่ไม่ต้องการให้ซ้ำกัน เช่น รายชื่อผู้ลงทะเบียนเวิร์กช็อป, คำศัพท์ที่ไม่ซ้ำกันในภาษา, การเก็บข้อมูลที่ไม่จำเป็นต้องเรียงลำดับ เป็นต้น.

set กับ list ต่างกันอย่างไร

ตัวอย่าง

```
# ตัวอย่าง List
my_list = [1, 2, 3, 2, 4]
print(my_list)  # Output: [1, 2, 3, 2, 4]

# ตัวอย่าง Set
my_set = {1, 2, 3, 2, 4}
print(my_set)  # Output: {1, 2, 3, 4}
```

4.4 ชนิดข้อมูลในภาษาไพธอน

ชนิดของตัวแปรในภาษาไพธอน

```
x = 5
y = "Asst. Prof. Dr. Nattapong "
print(x)
print(y)
```

```
x = 4       # x is of type int
x = "Sally" # x is now of type str
print(x)
```

ชนิดข้อมูล (Data Type)	ประเภท (Size)	คำอธิบาย (Descri
Integer	จำนวนเต็ม	ใช้เก็บตัวเลขจำนวนเต็ม เช่น I = 7 X = 1 A = 0
float	ทศนิยม	ใช้เก็บตัวเลขที่มีทศนิยม เช่น I = 2.7 X = 0.9 A = 0.0
Character	อักขระ	ใช้เก็บอักขระเพียงตัวเดียว เช่น A = 'Y' K = 'N'
String	ข้อความ	ใช้เก็บในลักษณะข้อความ เช่น Name = 'Nattapong' Status = 'Teacher'
boolean	จริง/เท็จ	ใช้เก็บข้อมูลลักษณะจริงเท็จ X = True Y = False

4.4.1 ข้อมูลแบบตัวเลข

ข้อมูลแบบตัวเลข (Numbers) ในภาษาไพธอน นั้นสนับสนุนข้อมูลแบบตัวเลข ซึ่งข้อมูลประเภทนี้จะแบ่งออกเป็น Integer Float Decimal และ Complex อย่างไรก็ตามเราจะเน้นเฉพาะข้อมูลตัวเลขที่เป็น Integer ซึ่งเป็นการเก็บข้อมูลแบบจำนวนเต็ม และข้อมูลตัวเลขที่เป็น Float เป็นข้อมูลแบบจำนวนจริง สำหรับประเภทแบบ Decimal นั้นแตกต่างไปจาก Float คือสามารถเก็บความละเอียดของจุดทศนิยมได้มากกว่า นอกจากนี้ไพธอน ยังสนับสนุนตัวเลขในรูปแบบ Complex ที่แสดงในแบบ a +bj ต่อไปเป็นตัวอย่างในการประกาศและใช้งานตัวแปรแบบตัวเลขในภาษาไพธอน

4.4.2 ข้อมูลตัวเลขที่เป็นจำนวนเต็ม

ตัวอย่างนี้จะเป็นการใช้งานตัวแปรสำหรับเก็บค่าตัวเลขที่เป็นจำนวนเต็มและแสดงผลออกมาทางจอภาพดังตัวอย่างโปรแกรม ต่อไปนี้

สร้างโปรแกรมในภาษาไพธอน ด้วย VS.CODE
ตั้งชื่อ Test_Var05.py

Test_var05.py

```
# Integer
a = 10
b = 3
c = a + b
d = a / b
print ('a = %d' % a)
print ('b = %d' % b)
print ('c = %d' % c)
print ('d = ', d)
```

4.4.2 ข้อมูลตัวเลขที่เป็นจำนวนเต็ม

ตัวอย่างนี้จะเป็นการใช้งานตัวแปรสำ
ออกมาทางจอภาพดังตัวอย่างโปรแกรม ต่อไป

สร้างโปรแกรมในภาษาไพธอน ด้วย VS.CODE
ตั้งชื่อ Test_Var05.py

Test_var05.py

Test_Var05.py > ...

```
1  # Project Name : Test_var05.py
2  # Developer    : Asst. Prof. Dr. Nattapong Songneam
3  # Create Date  : 05.08.2566
4  # Modify Date  : 05.08.2566
5  #-----------------------------------------------
6  # Integer
7  a = 10
8  b = 3
9  c = a + b
10 d = a / b
11 print ('a = %d' % a)
12 print ('b = %d' % b)
13 print ('c = %d' % c)
14 print ('d = ', d)
15
```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL

```
PS D:\OOP_1_2566> & C:/Users/User/AppData/Local/Microsoft/WindowsApps/python3.10.exe d:/OOP_1_2566/Test_Var05.py
a = 10
b = 3
c = 13
d =  3.3333333333333335
PS D:\OOP_1_2566>
```

ชนิดข้อมูลในภาษาไพธอน

ในตัวอย่าง เป็นการประกาศและใช้งานตัวแปรประเภท Integer เราได้ทำการประกาศตัวแปรและกำหนดค่าให้กับ a และ b ในการแสดงผลในรูปแบบของ String format กับฟังก์ชัน print() นั้นจะใช้ specifier เป็น %d เราสามารถกำหนดค่าให้กับตัวแปรได้โดย Literal หรือ Expression

สิ่งหนึ่งที่น่าสังเกตในการหารตัวเลขในภาษาไพธอน การหารตัวเลขนั้นจะได้ค่าเป็น Float เสมอ ถึงแม้ตัวเลขทั้งสองจะเป็น Integer ก็ตาม เช่นในตัวแปร d ซึ่งแตกต่างจากภาษา C ที่เมื่อตัวเลขทั้งสองเป็นแบบ Integer จะได้ผลลัพธ์เป็น Integer

```
d=a/b
```

ผลลัพธ์

```
a = 10
b = 3
c = 10
d =  2.3333333333333335
```

4.4.3 ข้อมูลแบบตัวเลขทศนิยม

การประกาศและใช้งานตัวแปรประเภท Float หรือตัวเลขที่มีจุดทศนิยม ในการกำหนดค่าให้กับตัวแปรนั้น เมื่อคุณกำหนดตัวเลขมีจุดมันจะเป็นประเภท Float อัตโนมัติ เราสามารถกำหนดค่าโดยตรงหรือในรูปแบบของ Expression ได้ และนอกจากนี้ ในภาษา Python ยังสามารถกำหนดในรูปแบบสัญกรณ์วิทยาศาสตร์ได้เหมือนในตัวแปร height ซึ่งหมายถึง 2.31 x 10 ^ 5 และในตัวแปร length ซึ่งหมายถึง 1.3 x 10 ^ -3

Test_var06.py

```
# Floating point number
speed = 34.12
pi = 22 / 7
height = 2.31E5
length = 1.3E-3
print ('speed = %f' % speed)
print ('pi = %f' % pi)
print ('height = %f' % height)
print ('length = %f' % length)
print (pi)
```

4.4 ชนิดข้อมูลในภาษาไพธอน

4.4.3 ข้อมูลแบบตัวเลขทศนิยม

Test_var06.py

```
# Floating point number
speed = 34.12
pi = 22 / 7
height = 2.31E5
length = 1.3E-3
print ('speed = %f' % speed)
print ('pi = %f' % pi)
print ('height = %f' % height)
print ('length = %f' % length)
print (pi)
```

Test_var06.py > ...

```
1  # Project Name  : Test_var06.py
2  # Developer     : Asst. Prof. Dr. Nattapong Songneam
3  # Create Date   : 05.08.2566
4  # Modify Date   : 05.08.2566
5  #-----------------------------------------------
6  # Floating point number
7  speed = 34.12
8  pi = 22 / 7
9  height = 2.31E5
10 length = 1.3E-3
11 print ('speed = %f' % speed)
12 print ('pi = %f' % pi)
13 print ('height = %f' % height)
14 print ('length = %f' % length)
15 print (pi)
16
```

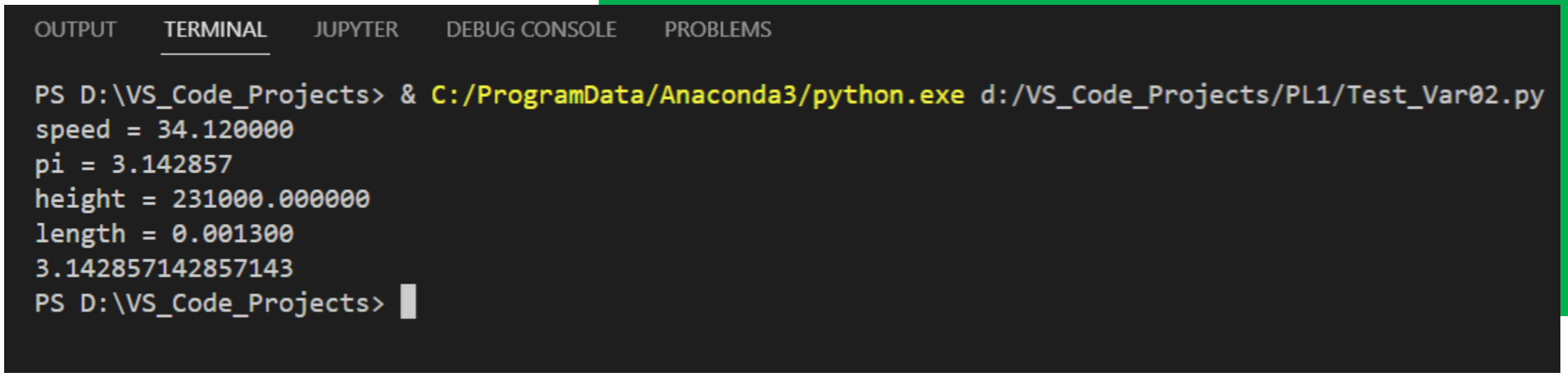
PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL

```
PS D:\OOP_1_2566> & C:/Users/User/AppData/Local/Microsoft/WindowsApps/python3.10.exe d:/OOP_1_2566/Test_var06.py
speed = 34.120000
pi = 3.142857
height = 231000.000000
length = 0.001300
3.142857142857143
PS D:\OOP_1_2566>
```

4.4.3 ข้อมูลแบบตัวเลขทศนิยม

ผลลัพธ์

speed = 34.120000
pi = 3.142857
height = 231000.000000
length = 0.001300
3.142857142857143



4.4.4 ข้อมูลประเภทสตริง

สตริง (Strings) เป็นประเภทข้อมูลที่สำคัญและใช้งานทั่วไปในการเขียนโปรแกรม ในภาษาเขียนโปรแกรมส่วนมากแล้วจะมีประเภทข้อมูลแบบสตริงและในภาษาไพธอน เช่นกัน สตริงเป็นลำดับของตัวอักษรหลายตัวเรียงต่อกัน ซึ่งในภาษาไพธอนนั้น สตริงจะอยู่ในเครื่องหมาย Double quote หรือ Single quote เท่านั้น นอกจากนี้ในภาษาไพธอน ยังมีฟังก์ชันในการจัดการกับ String มากมาย ซึ่งเราจะพูดอีกครั้งในหัวข้อของการทำงานกับข้อมูลสตริงในบทนี้มาทำความรู้จักกับสตริงเบื้องต้นกันก่อน

```
name = “Asst. Prof. Dr. Nattapong Songneam"
country = "Thailand"
Programming language = 'Python'
interest = 'Iot , robot , machine learnine, A.I.'
```

4.4.4 ข้อมูลประเภทสตริง

ในตัวอย่าง เป็นการประกาศตัวแปรประเภทสตริง สองตัวแปรแรกเป็นการประโดยการใช้ Double quote และสองตัวแปรต่อมาเป็นการใช้ Single quote ซึ่งคุณสามารถใช้แบบไหนก็ได้ แต่มีสิ่งที่แตกต่างกันเล็กน้อยคือเกี่ยวกับการกำหนดตัวอักษรพิเศษหรือเรียกว่า Escape character

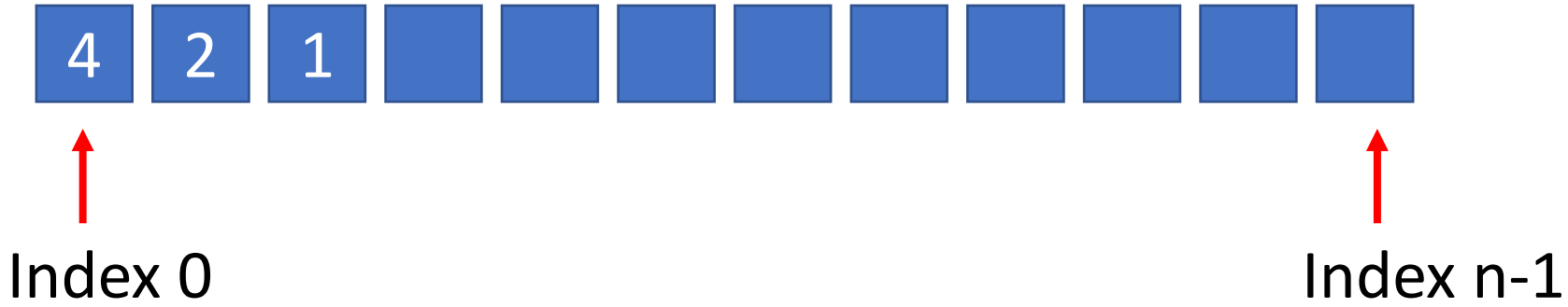
สร้างโปรแกรมในภาษาไพธอน ด้วย VS.CODE
ตั้งชื่อ Test_Var07.py

Test_var07.py

```
sentent1 = "What's your name?"
sentent2 = 'I\'m Dr.Nattapong Songneam.'
sentent3 = "He said \"I would learn Python first\"."
sentent4 = 'His teach replied "Oh well!"'
print (sentent1)
print (sentent2)
print (sentent3)
print (sentent4)
```

4.4.5 ข้อมูลประเภทลิสต์

ลิสต์ (Lists) เป็นประเภทข้อมูลที่เก็บข้อมูลแบบเป็นชุดและลำดับ กล่าวคือมันสามารถเก็บข้อมูลได้หลายค่าในตัวแปรเดียว และมี Index สำหรับเข้าถึงข้อมูล ในภาษา Python นั้น List จะเป็นเหมือนอาเรย์ในภาษาซี (C) หรือภาษาจาวา (Java) มันสามารถเก็บข้อมูลได้หลายตัวและยังสามารถเป็นประเภทข้อมูลที่แตกต่างกันได้อีกด้วย มาดูการประกาศและใช้งานลิสต์ในเบื้องต้น



4.4.5 ข้อมูลประเภทลิสต์

สร้างโปรแกรมในภาษาไพธอน ด้วย VS.CODE
ตั้งชื่อ Test_list01.py

*** ใครทำเสร็จ ให้ capture ผลลัพธ์ส่งทางไลน์

Test_var08_list.py

```
# Declare lists
numbers = [1, 2, 4, 6, 8, 19]
names = ['Nattapong', 'Kittipat', 'John', 'Thomas', 'Luke']
mixed = [-2, 5, 84.2, "Mountain", "Python"]
# Display lists
print(numbers)
print(names)
print(mixed)
# Display lists using the for loops
for n in numbers:
    print(n, end=" ")
print()
for n in names:
    print(n, end=" ")
print()
for n in mixed:
    print(n, end=" ")
print()
```

4.4 ชนิดข้อมูลในภาษาไพธอ

4.4.5 ข้อมูลประเภทลิสต์

สร้างโปรแกรมในภาษาไพธอน ด้วย VS.CODE
ตั้งชื่อ Test_list01.py

*** ใครทำเสร็จ ให้ capture ผลลัพธ์ส่งทางไลน์

Test_var08_list.py

```
PS D:\OOP_1_2566> & C:/Users/User/AppData/Local/Micro
[0, 2, 4, 6, 8, 10]
['Nattapong', 'Kittipat', 'John', 'Thomas', 'Luke']
[-2, 5, 84.2, 'Mountain', 'Python']
0 2 4 6 8 10
Nattapong Kittipat John Thomas Luke
-2 5 84.2 Mountain Python
PS D:\OOP_1_2566>
```

โดย ผู้ช่วยศาสตราจารย์ ดร.

Test_var08_list.py > ...

```python
# Project Name : Test_var06.py
# Developer    : Asst. Prof. Dr. Nattapong Songneam
# Create Date  : 05.08.2566
# Modify Date  : 05.08.2566
#-----------------------------------------------------------
# Declare lists -----------------------------------------------
numbers = [0, 2, 4, 6, 8, 10]
names = ['Nattapong', 'Kittipat', 'John', 'Thomas', 'Luke']
mixed = [-2, 5, 84.2, "Mountain", "Python"]
# Display lists
print(numbers)
print(names)
print(mixed)
# Display lists using the for loops
for n in numbers:
    print(n, end=" ")
print()
for n in names:
    print(n, end=" ")
print()
for n in mixed:
    print(n, end=" ")
print()

```

VS_Code_Projects > PL1 > Test_List01.py > ...

```python
# Declare lists
numbers = [1, 2, 4, 6, 8, 19]
names = ['Nattapong', 'Kittipat', 'John', 'Thomas', 'Luke']
mixed = [-2, 5, 84.2, "Mountain", "Python"]
# Display lists
print(numbers)
print(names)
print(mixed)
# Display lists using the for loops
for n in numbers:
    print(n, end=" ")
print()
for n in names:
    print(n, end=" ")
print()
for n in mixed:
    print(n, end=" ")
print()

```

```
PS D:\VS_Code_Projects> & C:/ProgramData/Anaconda3/python.exe d:/VS_Code_Projects/PL1/Test_List01.py
[1, 2, 4, 6, 8, 19]
['Nattapong', 'Kittipat', 'John', 'Thomas', 'Luke']
[-2, 5, 84.2, 'Mountain', 'Python']
1 2 4 6 8 19
Nattapong Kittipat John Thomas Luke
-2 5 84.2 Mountain Python
PS D:\VS_Code_Projects>
```

ในตัวอย่าง เป็นการประกาศ 3 ลิสต์ โดยตัวแปรแรกนั้นเป็นลิสต์ของตัวเลข และตัวแปรที่สองเป็น ลิสต์ของสตริ และตัวแปรสุดท้ายเป็นลิสต์ แบบรวมกันของประเภทข้อมูล เราใช้ฟังก์ชัน print() ในการแสดงผลข้อมูลในลิสต์ และใช้คำสั่ง For loop ในการอ่านค่าในลิสต์และนำมาแสดงผลเช่นกัน

```
[1, 2, 4, 6, 8, 19]
['Nattapong', 'Kittipat', 'John', 'Thomas', 'Luke']
[-2, 5, 84.2, 'Mountain', 'Python']
1 2 4 6 8 19
Nattapong Kittiphat John Thomas Luke
-2 5 84.2 Mountain Python
```

ชนิดข้อมูลในภาษาไพธอน

```
languages = ["C", "C++", "Java", "Python", "PHP"]

print("Index at 0 = ", languages[0])
print("Index at 3 = ", languages[3])
languages[0] = "Scalar"
print("Index at 0 = ", languages[0])
```

ลิสต์นั้นทำงานกับ Index ดังนั้นเราสามารถเข้าถึงข้อมูลของลิสต์ โดยการใช้ Index ของมันได้ ในตัวอย่างเป็นการเข้าถึงข้อมูลภายใน Index ซึ่ง Index ของ List นั้นจะเริ่มจาก 0 ไปจนถึงจำนวนทั้งหมดของมันลบด้วย 1 ในตัวอย่างเราได้แสดงผลข้อมูลของสอง List ในตำแหน่งแรกและในตำแหน่งที่ 4 ด้วย Index 0 และ 3 ตามลำดับ หลังจากนั้นเราเปลี่ยนค่าของ List ที่ตำแหน่งแรกเป็น "Scalar“

Index 0 = C
Index 3 = Python
Index 0 = Scalar

4.5 ฟังก์ชันที่ใช้กับตัวแปร

ในภาษาไพธอนนั้นมีฟังก์ชันที่สร้างมาเพื่อให้ใช้งานกับตัวแปร เช่น ฟังก์ชันสำหรับหาขนาดของตัวแปร ฟังก์ชันในการหาประเภทของตัวแปร ฟังก์ชันลบตัวแปรออกไปจากหน่วยความจำ และฟังก์ชันในการตรวจสอบว่าตัวแปรมีอยู่หรือไม่ ซึ่งในบางครั้งการเขียนโปรแกรมก็จำเป็นที่คุณอาจจะต้องมีการตรวจสอบสิ่งเหล่านี้ในขณะที่โปรแกรมทำงาน นี่เป็นตัวอย่างการใช้งาน

4.5 ฟังก์ชันที่ใช้งานกับตัวแปร

ในตัวอย่าง เป็นประกาศตัวแปรประเภทต่าง ๆ โดยใช้งานฟังก์ชัน getsizeof() สำหรับหาขนาดของตัวแปรที่มีหน่วยเป็น Byte และฟังก์ชัน type() สำหรับดูประเภทของตัวแปรว่าอยู่ในคลาสใด ในขณะที่ใช้ฟังก์ชัน del สำหรับยกเลิกหรือลบการประกาศตัวแปรออกไปจากหน่วยความจำ และสุดท้ายเป็นการตรวจสอบว่าตัวแปรถูกประกาศแล้วหรือยังในฟังก์ชัน locals() สำหรับตรวจสอบตัวแปรในโมดูลปัจจุบัน หรือ globals() สำหรับตรวจสอบตัวแปรในโปรแกรมทั้งหมด

สรุปคำสั่ง

- type() ตรวจสอบชนิดของตัวแปร
- sys.getsizeof() หาขนาดของตัวแปร
- del ลบตัวแปร
- locals() ตรวจสอบตัวแปรใน local
- globals() ตรวจสอบตัวแปรใน global

Test_var09.py

```
Size of a =  28
Type of a =  <class 'int'>
Size of b =  24
Type of b =  <class 'float'>
Size of c =  55
Type of c =  <class 'str'>
Size of d =  120
Type of d =  <class 'list'>
a is not exist
```

```
PS D:\VS_Code_Projects> & C:/Progra
Size of a =  28
Type of a =  <class 'int'>
Size of b =  24
Type of b =  <class 'float'>
Size of c =  55
Type of c =  <class 'str'>
Size of d =  120
Type of d =  <class 'list'>
a is not exist
PS D:\VS_Code_Projects>
```

#ชื่อนามสกุล
#รหัส

4.5 ฟังก์ชันที่ใช้กับตัวแปร

สรุปคำสั่ง

- type() ตรวจสอบชนิดของตัวแปร
- sys.getsizeof() หาขนาดของตัวแปร
- del ลบตัวแปร
- locals() ตรวจสอบตัวแปรใน local
- globals() ตรวจสอบตัวแปรใน global

สร้างโปรแกรมในภาษาไพธอน ด้วย VS.CODE
ตั้งชื่อ Test_var09.py

*** ใครทำเสร็จ ให้ capture ผลลัพธ์ส่งทางไลน์

Test_var09.py

```
import sys
a = 8
b = 13.4
c = "Python"
d = [1, 2, 3, 4]
print('Size of a = ', sys.getsizeof(a))
print('Type of a = ', type(a))
print('Size of b = ', sys.getsizeof(b))
print('Type of b = ', type(b))
print('Size of c = ', sys.getsizeof(c))
print('Type of c = ', type(c))
print('Size of d = ', sys.getsizeof(d))
print('Type of d = ', type(d))
del a
del b, c, d
if 'a' in locals():
    print("a is exist")
else:
    print("a is not exist")
```

4.5 ฟังก์ชันที่ใช้งานกับตัวแปร

4.5 ฟังก์ชันที่ใช้กับตัวแปร

สรุปคำสั่ง

- **type()** ตรวจสอบชนิดของตัวแปร
- **sys.getsizeof()** หาขนาดของตัวแปร
- **del** ลบตัวแปร
- **locals()** ตรวจสอบตัวแปรใน local
- **globals()** ตรวจสอบตัวแปรใน global

สร้างโปรแกรมในภาษาไพธอน ด้วย VS.CODE
ตั้งชื่อ Test_var09.py

*** ใครทำเสร็จ ให้ capture ผลลัพธ์ส่งทางไลน์

Test_var09.py

โดย ผู้ช่วยศาสตราจารย์ ดร. นัฐพงศ์

Test_var09.py > ...

```python
# Project Name : Test_var09.py
# Developer    : Asst. Prof. Dr. Nattapong Songneam
# Create Date  : 05.08.2566
# Modify Date  : 05.08.2566
#-----------------------------------------------
import sys
a = 8
b = 13.4
c = "Python"
d = [1, 2, 3, 4]
print('Size of a = ', sys.getsizeof(a))
print('Type of a = ', type(a))
print('Size of b = ', sys.getsizeof(b))
print('Type of b = ', type(b))
print('Size of c = ', sys.getsizeof(c))
print('Type of c = ', type(c))
print('Size of d = ', sys.getsizeof(d))
print('Type of d = ', type(d))
del a
del b, c, d
if 'a' in locals():
    print("a is exist")
else:
    print("a is not exist")
```

#ชื่อนามสกุล
#รหัส

4.5 ฟังก์ชันที่ใช้กับตัวแปร

สรุปคำสั่ง

- **type()** ตรวจสอบชนิดของตัวแปร
- **sys.getsizeof()** หาขนาดของตัวแปร
- **del** ลบตัวแปร
- **locals()** ตรวจสอบตัวแปรใน local
- **globals()** ตรวจสอบตัวแปรใน global

สร้างโปรแกรมในภาษาไพธอน ด้วย VS.CODE
ตั้งชื่อ Test_var09.py

*** ใครทำเสร็จ ให้ capture ผลลัพธ์ส่งทางไลน์

Test_var09.py

```
PROBLEMS    OUTPUT    DEBUG CONSOLE    TERMINAL

-2 5 84.2 Mountain Python
PS D:\OOP_1_2566> & C:/Users/User/AppData/Local/Microsof
Size of a =  28
Type of a =  <class 'int'>
Size of b =  24
Type of b =  <class 'float'>
Size of c =  55
Type of c =  <class 'str'>
Size of d =  88
Type of d =  <class 'list'>
a is not exist
PS D:\OOP_1_2566>
```

ฟังก์ชันที่ใช้กับตัวแปร

ในภาษา Python นั้นมีฟังก์ชันที่สร้างมาเพื่อให้ใช้งานกับตัวแปร เช่น ฟังก์ชันสำหรับหาขนาดของตัวแปร ฟังก์ชันในการหาประเภทของตัวแปร ฟังก์ชันลบตัวแปรออกไปจากหน่วยความจำ และฟังก์ชันในการตรวจสอบว่าตัวแปรมีอยู่หรือไม่ ซึ่งในบางครั้งการเขียนโปรแกรมก็จำเป็นที่คุณอาจจะต้องมีการตรวจสอบสิ่งเหล่านี้ในขณะที่โปรแกรมทำงาน นี่เป็นตัวอย่างการใช้งาน

ฟังก์ชันที่ใช้กับตัวแปร

```
import sys

a = 8
b = 13.4
c = "Python"
d = [1, 2, 3, 4]

print('Size of a = ', sys.getsizeof(a))
print('Type of a = ', type(a))

print('Size of b = ', sys.getsizeof(b))
print('Type of b = ', type(b))

print('Size of c = ', sys.getsizeof(c))
print('Type of c = ', type(c))

print('Size of d = ', sys.getsizeof(d))
print('Type of d = ', type(d))

del a
del b, c, d

if 'a' in locals():
    print("a is exist")
else:
    print("a is not exist")
```

ฟังก์ชันที่ใช้งานกับตัวแปร

ในตัวอย่าง เราได้ประกาศตัวแปรประเภทต่างๆ เราได้ฟังก์ชัน getsizeof() สำหรับหาขนาดของตัวแปรที่มีหน่วยเป็น Byte และฟังก์ชัน type() สำหรับดูประเภทของตัวแปรว่าอยู่ในคลาสไหน ฟังก์ชัน del() สำหรับยกเลิกหรือลบการประกาศตัวแปรออกไปจากหน่วยความจำ และสุดท้ายเป็นการตรวจสอบว่าตัวแปรถูกประกาศแล้วหรือยังในฟังก์ชัน locals() สำหรับตรวจสอบตัวแปรในโมดูลปัจจุบัน หรือ globals() สำหรับตรวจสอบตัวแปรในโปรแกรมทั้งหมด

```
Size of a =  14
Type of a =  <class 'int'>
Size of b =  16
Type of b =  <class 'float'>
Size of c =  31
Type of c =  <class 'str'>
Size of d =  52
Type of d =  <class 'list'>
a is not exist
```

```python
# developer : Dr. Nattapong Songnema
# last modifier : 13.08.2022
import sys
a = 8
b = 13.4
c = "Python"
d = [1, 2, 3, 4]
print('Size of a = ', sys.getsizeof(a))
print('Type of a = ', type(a))
print('Size of b = ', sys.getsizeof(b))
print('Type of b = ', type(b))
print('Size of c = ', sys.getsizeof(c))
print('Type of c = ', type(c))
print('Size of d = ', sys.getsizeof(d))
print('Type of d = ', type(d))
del a
del b, c, d
if 'a' in locals():
    print("a is exist")
else:
    print("a is not exist")

```

```
PS D:\VS_Code_Projects> & C:/Progra
Size of a =  28
Type of a =  <class 'int'>
Size of b =  24
Type of b =  <class 'float'>
Size of c =  55
Type of c =  <class 'str'>
Size of d =  120
Type of d =  <class 'list'>
a is not exist
PS D:\VS_Code_Projects>
```

อักขระพิเศษในภาษาไพธอน

อักขระพิเศษ

อักขระพิเศษ(Escape Character) ในตัวอย่าง เป็นสิ่งที่แตกต่างของการประกาศ String ทั้งสองแบบกับ Escape character ตัวอักษร ' และ " นั้นเป็น Escape character ดังนั้นในการใช้งานตัวอักษรเหล่านี้ เราจะต้องทำการใส่เครื่องหมาย \ ลงไปข้างหน้าเสมอ แต่ในภาษา Python เมื่อคุณใช้ Double quote ในการประกาศ String คุณไม่ต้องทำการ Escape character สำหรับ Single quote และในทางกลับกัน อย่างไรก็ตามเราจะพูดอีกครั้งในบทของ String

```
\n
\t
\’
\”
```

```
What's your name?
I'm Dr.Nattapong Songneam.
He said "I would learn Python first".
His teach replied "Oh well!"
```

ตัวดำเนินการในภาษาไพธอน

ตัวดำเนินการ (Operators) คือกลุ่มของเครื่องหมายหรือสัญลักษณ์ที่ใช้ทำงานเหมือนกับฟังก์ชัน แต่แตกต่างกันตรงไวยากรณ์หรือความหมายในการใช้งาน ในภาษา ไพธอน นั้นสนับสนุนตัวดำเนินการประเภทต่าง ๆ สำหรับการเขียนโปรแกรม เช่น ตัวดำเนินการ + เป็นตัวดำเนินการทางคณิตศาสตร์ที่ใช้สำหรับการบวกตัวเลขเข้าด้วยกัน หรือตัวดำเนินการ > เป็นตัวดำเนินการเพื่อให้เปรียบเทียบค่าสองค่า โดยตัวดำเนินการพื้นฐานในภาษาไพธอน มีดังนี้

1. Assignment operator
2. Arithmetic operators
3. Comparison operators
4. Logical operators
5. Bitwise operators
6. Sequence Operators
7. Truth Value Testing

4.7.1 Assignment operator

ตัวดำเนินการกำหนดค่า (Assignment operator) ตัวดำเนินการที่เป็นพื้นฐานที่สุดสำหรับการเขียนโปรแกรมในทุกๆ ภาษาก็คือ ตัวดำเนินการกำหนดค่า (Assignment operator) ตัวดำเนินการนี้แสดงโดยใช้เครื่องหมายเท่ากับ (=) มันใช้สำหรับกำหนดค่าให้กับตัวแปร มาดูตัวอย่างการใช้งานในภาษาไพธอน

```
a = 3
b = 5.29
c = b
name = 'Asst. Dr. Nattapong Songneam'
my_list = [2, 5, 8, 10, 24]
x, y = 10, 20
x = y = z = 100
```

```
x=10
z=y=x
print(z)
```

4.7.2 Arithmetic operators

ตัวดำเนินการทางคณิตศาสตร์ (Arithmetic operators) คือตัวดำเนินการที่ใช้สำหรับการคำนวณทางคณิตศาสตร์ในพื้นฐาน เช่น การบวก การลบ การคูณ และการหาร มากไปกว่านั้นในภาษา ไพธอน ยังมีตัวดำเนินการทางคณิตศาสตร์เพิ่มเติม เช่น การหารเอาเศษ (Modulo) การหารแบบเลขจำนวนเต็ม และการยกกำลัง เป็นต้น

Operator	Name	Example
+	Addition	a + b
-	Subtraction	a - b
*	Multiplication	a * b
/	Division	a / b
//	Division and floor	a // b
%	Modulo	a % b
**	Power	a ** b

5//2 = ???

5/2 = 2.5

5%2 = 1

2**3

4.7.2 Arithmetic operators

ตัวดำเนินการทางคณิตศาสตร์ (Arithmetic operators) คือตัวดำเนินการที่ใช้สำหรับการคำนวณทางคณิตศาสตร์ในพื้นฐาน เช่น การบวก การลบ การคูณ และการหาร มากไปกว่านั้นในภาษา ไพธอน ยังมีตัวดำเนินการทางคณิตศาสตร์เพิ่มเติม เช่น การหารเอาเศษ (Modulo) การหารแบบเลขจำนวนเต็ม และการยกกำลัง เป็นต้น

5//2 = ???

5/2 = 2.5

5%2 = 1

2**3

คำตอบคือ 5//2 = 2

ใน Python, ตัวดำเนินการ // คือการทำการหารแบบปัดเศษลง (Floor Division) ซึ่งหมายถึงการหารเลขและปัดลงไปยังค่าที่ใกล้ที่สุดของจำนวนเต็มที่น้อยกว่าหรือเท่ากับผลการหารที่ได้ ในกรณีนี้ 5 หารด้วย 2 เท่ากับ 2 และเศษที่เหลือคือ 1 จึงจะถูกปัดทิ้งในการทำ Floor Division ก่อนที่จะได้ผลลัพธ์เป็น 2

4.7.3 Arithmetic operators

ในตารางข้างบน มีตัวดำเนินการทางคณิตศาสตร์ประเภทต่าง ๆ สำหรับการคำนวณเกี่ยวกับคณิตศาสตร์เบื้องต้น ปกติผู้เรียนอาจจะคุ้นเคยกับตัวดำเนินการบวก ลบ คูณ หาร ในการเรียนระดับมัธยมศึกษามาบ้างแล้ว ในภาษา ไพธอน นั้นสนับสนุนตัวดำเนินการสำหรับการหารเอาเศษเช่นเดียวกับภาษาอื่นๆ และนอกจากนี้ ยังมีตัวดำเนินการแบบการหารที่ได้ผลลัพธ์เป็นจำนวนเต็ม และการหาเลขยกกำลังเพิ่มเข้ามา มาดูตัวอย่างการใช้ตัวดำเนินการประเภทต่าง ๆ ในภาษาไพธอน ดังนี้

Test_Opr01.py

```
a = 5
b = 3
print("a + b = ", a + b)
print("a - b = ", a - b)
print("a * b = ", a * b)
print("a / b = ", a / b)
print("a // b = ", a // b) # floor number to integer
print("a % b = ", a % b)   # get division remainder
print("a ** b = ", a ** b) # power
```

ตัวดำเนินการในภาษาไพธอน

ในตัวอย่าง เราได้ประกาศตัวแปร a และ b และกำหนดค่าให้กับตัวแปรทั้งสองเป็น 5 และ 3 ตามลำดับ ในสี่ตัวดำเนินการแรกเป็นการดำเนินการทางคณิตศาสตร์พื้นฐาน สำหรับตัวดำเนินการ // เป็นการหารเช่นเดียวกัน แต่ผลลัพธ์ของการหารนั้นจะตัดส่วนที่เป็นทศนิยมทิ้งไป ส่วนตัวดำเนินการ % นั้นเป็นการหารโดยผลลัพธ์จะเป็นเศษของการหารแทน ส่วนสุดท้าย ** นั้นแทนการยกกำลัง

```
a + b =  8
a - b =  2
a * b =  15
a / b =  1.6666666666666667
a // b =  1
a % b =  2
a ** b =  125
```

4.7.4 Comparison operators

ตัวดำเนินการเปรียบเทียบ (Comparison operators) คือตัวดำเนินการที่ใช้สำหรับเปรียบเทียบค่าหรือค่าในตัวแปร ซึ่งผลลัพธ์ของการเปรียบเทียบนั้นจะเป็น True หากเงื่อนไขเป็นจริง และเป็น False หากเงื่อนไขไม่เป็นจริง ตัวดำเนินการเปรียบเทียบมักจะใช้กับคำสั่งตรวจสอบเงื่อนไข if และคำสั่งวนซ้ำ for while เพื่อควบคุมการทำงานของโปรแกรมนี่เป็นตารางของตัวดำเนินการเปรียบเทียบในภาษา ไพธอน

Operator	Name	Example
<	Less than	a < b
<=	Less than or equal	a <= b
>	Greater than	a > b
>=	Greater than or equal	a >= b
==	Equal	a == b
!=	Not equal	a != b
is	Object identity	a is b
is not	Negated object identity	a is not b

ตัวดำเนินการในภาษาไพธอน

ในตาราง แสดงให้เห็นถึงตัวดำเนินการเปรียบเทียบประเภทต่างๆ เช่น การเปรียบเทียบความเท่ากัน โดยคุณสามารถใช้ตัวดำเนินการเปรียบเทียบเพื่อเปรียบเทียบว่าค่าในตัวแปรนั้นเท่ากันหรือไม่ หรือการเปรียบเทียบค่ามากกว่าหรือน้อยกว่า ต่อไปมาดูตัวอย่างการใช้งานตัวดำเนินการเปรียบเทียบในภาษาไพธอน

```
# Constant comparison
print('4 == 4 :', 4 == 4)
print('1 < 2:', 1 < 2)
print('3 > 10:', 3 > 10)
print('2 <= 1.5', 2 <= 1.5)
print()

# Variable comparison
a = 10
b = 8
print('a != b:', a != b)
print('a - b == 2:', a - b == 2)
print()
```

ตัวดำเนินการในภาษาไพธอน

ในตัวอย่าง เป็นการเปรียบเทียบค่าประเภทต่าง ๆ ในคำสั่งกลุ่มแรกนั้นเป็นการใช้ตัวดำเนินการเปรียบเทียบกับค่าคงที่ ในกลุ่มที่สองเป็นการใช้งานกับตัวแปร ซึ่งถ้าหากเงื่อนไขเป็นจริงจะได้ผลลัพธ์เป็น True และถ้าหากไม่จริงจะได้ผลลัพธ์เป็น False

```
4 == 4 : True
1 < 2: True
3 > 10: False
2 <= 1.5 False

a != b: True
a - b == 2: True
```

4.7.5 Logical operators

ตัวดำเนินการตรรกศาสตร์ (Logical operators) คือตัวดำเนินการที่ใช้สำหรับประเมินค่าทางตรรกศาสตร์ ซึ่งเป็นค่าที่มีเพียงจริง (True) และเท็จ (False) เท่านั้น โดยทั่วไปแล้วเรามักใช้ตัวดำเนินการตรรกศาสตร์ในการเชื่อม Boolean expression ตั้งแต่หนึ่ง expression ขึ้นไปและผลลัพธ์สุดท้ายที่ได้นั้นจะเป็น Boolean

Operator	Example	Result
and	a and b	True if a and b are true, else False
or	a or b	True if a or b are true, else False
not	not a	True if a is False, else True

AND

- 11110000
- 10101010
- **10100000**

OR

- 11110000
- 10101010
- 11111010

XOR

- 11110000
- 10101010
- 01011010

ตัวดำเนินการในภาษาไพธอน

ในภาษา ไพธอน นั้นมีตัวดำเนินการทางตรรกศาสตร์ 3 ชนิด คือ ตัวดำเนินการ and เป็นตัวดำเนินการที่ใช้เชื่อมสอง Expression และได้ผลลัพธ์เป็น True หาก Expression ทั้งสองเป็น True ไม่เช่นนั้นจะได้ผลลัพธ์เป็น False ตัวดำเนินการ or เป็นตัวดำเนินการที่ใช้เชื่อมสอง Expression และได้ผลลัพธ์เป็น True หากมีอย่างน้อยหนึ่ง Expression ที่เป็น True ไม่เช่นนั้นได้ผลลัพธ์เป็น False และตัวดำเนินการ not ใช้ในการกลับค่าจาก True เป็น False และในทางกลับกัน มาดูตัวอย่างการใช้งาน

Test_Opr02.py

```
print('Log in page')
username = input('Username: ')
password = input('Password: ')

if (username == ‘aast.dr.nattapong' and password == '3456'):
    print('Welcome Mateo, you\'ve logged in.')
else:
    print('Invalid username or password.')
```

4.7.6 Bitwise operators

ตัวดำเนินการระดับบิต (Bitwise operators) เป็นตัวดำเนินการที่ทำงานในระดับบิตของข้อมูล หรือจัดการข้อมูลในระบบเลขฐานสอง โดยทั่วไปแล้วตัวดำเนินการระดับบิตมักจะใช้กับการเขียนโปรแกรมระดับต่ำ เช่น การเขียนโปรแกรมเพื่อควบคุมฮาร์ดแวร์ อย่างไรก็ตาม ในภาษาไพธอน นั้นสนับสนุนตัวดำเนินการเพื่อให้เราสามารถจัดการกับบิตของข้อมูลโดยตรงได้

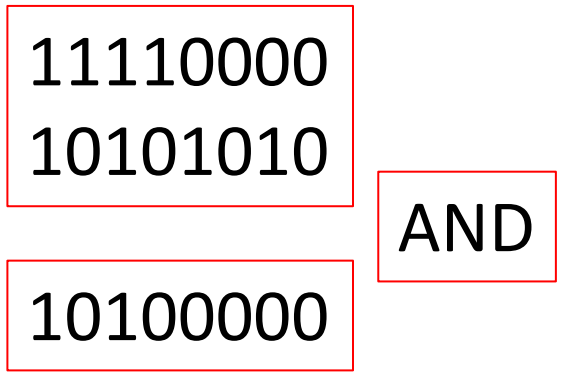
Operator	Name	Result
&	Bitwise and	a & b
|	Bitwise or	a | b
^	Bitwise exclusive or	a ^ b
<<	Bitwise shifted left	a << b
>>	Bitwise shifted right	a >> b
~	Bitwise invert	~a

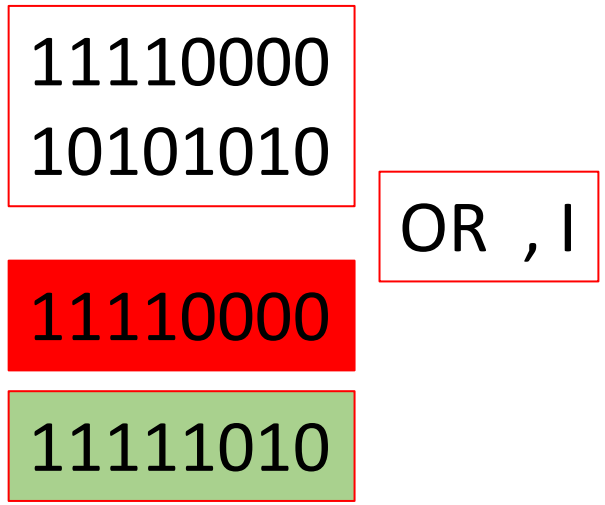
ตัวดำเนินการในภาษาไพธอน

ตัวดำเนินการระดับบิตใช้จัดการกับบิตของข้อมูลที่เป็นตัวเลข โดยปกติแล้วเมื่อเรากำหนดค่าให้กับตัวแปรนั้น คอมพิวเตอร์จะเก็บค่าเหล่านี้ในหน่วยความจำในรูปแบบของตัวเลขฐานสอง (binary form) ซึ่งประกอบไปด้วยเพียง 1 และ 0 เท่านั้น ดังนั้นเราใช้ตัวดำเนินการเหล่านี้ในการจัดการกับข้อมูลได้โดยตรง มาดูตัวอย่าง

```
a = 3 # 00000011
b = 5 # 00000101

print('a & b =', a & b)
print('a | b =', a | b)
print('a ^ b =', a ^ b)
print('~a =', ~a)
print('a << 1 =', a << 1)
print('a << 2 =', a << 2)
print('100 >> 1 =', 100 >> 1)
```



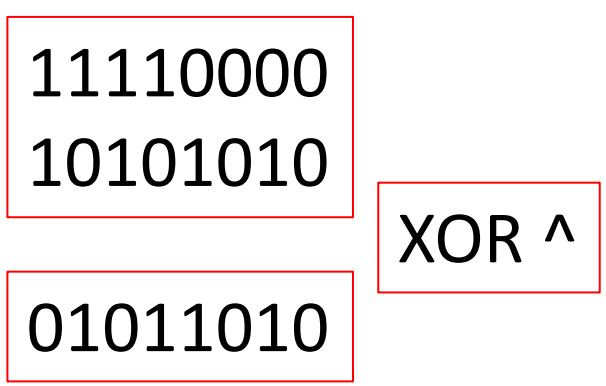


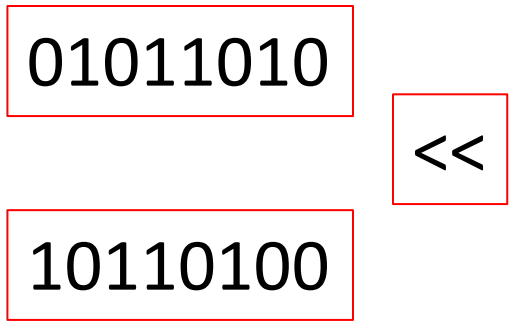
ตัวดำเนินการในภาษาไพธอน

ตัวดำเนินการระดับบิตใช้จัดการกับบิตของข้อมูลที่เป็นตัวเลข โดยปกติแล้วเมื่อเรากำหนดค่าให้กับตัวแปรนั้น คอมพิวเตอร์จะเก็บค่าเหล่านี้ในหน่วยความจำในรูปแบบของตัวเลขฐานสอง (binary form) ซึ่งประกอบไปด้วยเพียง 1 และ 0 เท่านั้น ดังนั้นเราใช้ตัวดำเนินการเหล่านี้ในการจัดการกับข้อมูลได้โดยตรง มาดูตัวอย่าง

```
a = 3 # 00000011
b = 5 # 00000101

print('a & b =', a & b)
print('a | b =', a | b)
print('a ^ b =', a ^ b)
print('~a =', ~a)
print('a << 1 =', a << 1)
print('a << 2 =', a << 2)
print('100 >> 1 =', 100 >> 1)
```





4.7.7 การทำงานกับสตริงในภาษาไพธอน

ในหัวข้อนี้จะได้เรียนรู้เกี่ยวกับสตริงในภาษาไพธอน โดยจะพูดเกี่ยวกับการประกาศและใช้งานสตริงในรูปแบบต่าง ๆ เพราะว่าเนื้อหาเกี่ยวกับสตริงนั้นมีค่อนข้างมาก ดังนั้นจึงได้รวบรวมเนื้อหาทั้งหมดไว้ในหัวข้อนี้ และเราจะแนะนำการใช้งาน built-in function ในภาษาไพธอนที่สำคัญในการจัดการกับข้อมูลประเภทสตริง

4.7.7 การทำงานกับสตริงในภาษาไพธอน

สตริง (String) เป็นลำดับของตัวอักษรหลายตัวเรียงต่อกัน ซึ่งในภาษาไพธอนนั้นการที่จะประกาศ สตริง ค่าของมันจะอยู่ในเครื่องหมาย Double quote (“”) หรือ Single quote (‘’) เท่านั้น มาดูตัวอย่างการประกาศตัวแปรของสตริง (String)

4.7.7 การเชื่อมต่อสตริง

การเชื่อมต่อสตริง (String concatenation) เป็นการทำงานอย่างหนึ่งที่สำคัญเกี่ยวกับสตริง ก็คือการเชื่อมต่อสตริงเข้าด้วยการ โดยเป็นการนำสตริงตั้งแต่สองอันขึ้นไปมาต่อกัน ในภาษาไพธอนสามารถต่อสตริงได้โดยการใช้เครื่องหมาย + หรือคั่นด้วยช่องว่างหรือบรรทัดใหม่เหมือนในตัวอย่างข้างบน

```
website = 'www.siam2dev' + '.com'
tutorial = 'Python' ' Language'
print(website)
print(tutorial)
```

ผลลัพธ์

```
www.siam2dev.com
Python Language
```

ตัวดำเนินการในภาษาไพธอน

```
name = “Asst. Dr. Nattapong Songneam"
website = 'www.siam2dev.com'
str1 = "This is my string"
str2 = 'This is my string'
```

ในตัวอย่าง เป็นการประกาศ 4 ตัวแปรของ String ซึ่งจะสังเกตว่าในการกำหนดค่าให้กับตัวแปรนั้น String literal จะถูกภายในเครื่องหมาย Double quote ("") หรือ Single quote (') เท่านั้น ซึ่งได้ผลการทำงานที่เหมือนกัน ขนาดของ String นั้นจะขึ้นกับจำนวนตัวอักษรภายใน String

อย่างไรก็ตาม ถึงแม้เราจะสามารถใช้ Double quote หรือ Single quote กับ String ได้ แต่มีสิ่งที่แตกต่างกันเล็กน้อยสำหรับการใช้งานทั้งสองแบบคือการใช้ตัวอักษรพิเศษใน String หรือเรียกว่า Escape character ลองมาดูตัวอย่างต่อไปนี้

ตัวดำเนินการในภาษาไพธอน

```
sentent1 = "What's your name?"
sentent2 = 'I\'m Asst. Dr.Nattapong Songneam'
sentent3 = "He said \"I would learn Python first\"."
sentent4 = 'His teach replied "Oh well!"'
print(sentent1)
print(sentent2)
print(sentent3)
print(sentent4)
```

ในตัวอย่าง เป็นสิ่งที่แตกต่างของการประกาศ String ทั้งสองแบบกับ Escape character ซึ่งตัวอักษร ' และ " นั้นเป็น Escape character ดังนั้นในการใช้งานตัวอักษรเหล่านี้ เราต้องใส่เครื่องหมาย \ ลงไปข้างหน้าเสมอ แต่ในภาษา Python เมื่อคุณใช้ Double quote ในการประกาศ String คุณไม่ต้องทำการ Escape character สำหรับ Single quote และในทางกลับกัน

แบบฝึกหัดบทที่ 4

4.1 จงหาผลลัพธ์ของโปรแกรมต่อไปนี้

```
# Constant comparison
print('5 == 4 :', 5 == 4)
print('3 < 2:', 3 < 2)
print('21 > 10:', 21 > 10)
print('1.2 <= 1.5', 1.2 <= 1.5)
print()

# Variable comparison
a = 5
b = 8
print('a != b:', a != b)
print('a - b == 3:', a - b == 3)
print()
```

1. False
2. False
3. True
4. True
5. false

แบบฝึกหัดบทที่ 4

4.2 จงเขียนโปรแกรมเพื่อหาจำนวนวินาทีใน 1 ปี

4.3 จงเขียนโปรแกรมเพื่อหาระยะทางของแสง ใน 1 ปี (ปีแสง มีหน่วยเป็นกิโลเมตร)

~~4.4 จงเขียนโปรแกรมเพื่อรับค่าตัวเลขใด ๆ แล้วตรวจสอบว่าเป็น Prime Number หรือไม่~~

อ้างอิง

ภาษาไพธอน. online : http://marcuscode.com/lang/python, สืบค้นเมื่อ 2 ก.ค. 2565

https://www.9experttraining.com/articles/python-คืออะไร

https://medium.com/@moungsiri/โครงสร้างภาษา-python-53cc38a51462

http://marcuscode.com/lang/python/variables-and-types

http://cms576.bps.in.th/group11/introduction-to-computer-programming

https://www.mindphp.com/บทเรียนออนไลน์/83-python/2398-python-operator.html