บทที่ 5
การเขียนโปรแกรมเบื้องต้น

เนื้อหา

1. ตัวแปลภาษา
2. รู้จักกับภาษาไพทอน
3. พื้นฐานการเขียนโปรแกรมด้วยภาษาไพทอน

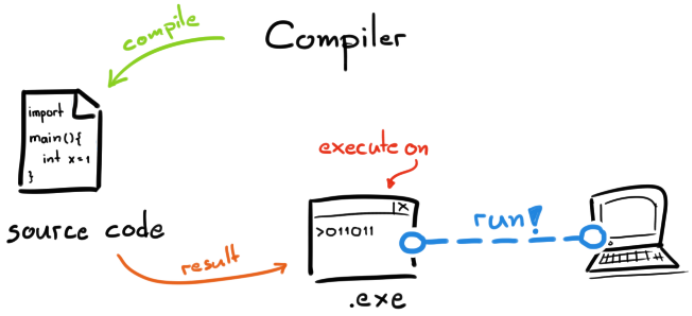
จุดประสงค์การเรียนรู้

1. เขียนโปรแกรมเพื่อรับและแสดงผลข้อมูลได้
2. เลือกใช้ตัวดำเนินการได้อย่างถูกต้องและเหมาะสม
3. เขียนโปรแกรมที่มีโครงสร้างการทำงานแบบลำดับได้

ตัวแปลภาษา (Compiler/Interpreter)

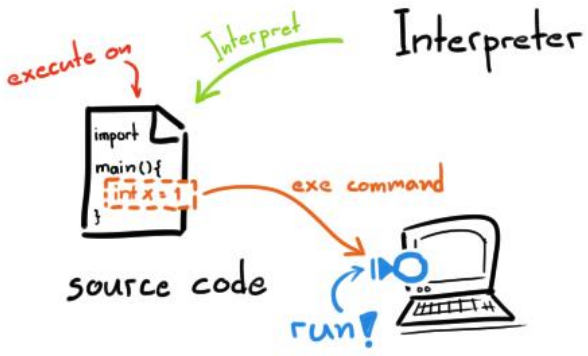
การเขียนโปรกรมด้วยภาษาระดับสูง จะมีลักษณะของโครงสร้างภาษาแตกต่างกันออกไป ซึ่งโปรแกรมที่มนุษย์เขียนขึ้นมานั้น เรียกว่า โปรแกรมต้นฉบับ (Source Code) มนุษย์จะอ่านโปรแกรมต้นฉบับนี้ได้ แต่คอมพิวเตอร์จะไม่เข้าใจคำสั่งเหล่านั้น เนื่องจากคอมพิวเตอร์เข้าใจแต่ภาษาเครื่อง (Machine Language) ซึ่งประกอบขึ้นจากเลขฐานสองเท่านี้น จึงต้องมีการใช้ตัวแปลภาษาคอมพิวเตอร์ (Translator) ในการแปลภาษาคอมพิวเตอร์ภาษาต่าง ๆ ไปเป็นภาษาเครื่อง ซึ่งจะประกอบไปด้วยรหัสคำสั่งที่คอมพิวเตอร์สามารถเข้าใจและนำไปปฏิบัติได้ ตัวแปลภาษาที่มีใช้ในปัจจุบันจะแตกต่างกัน สามารถแบ่งได้เป็น 2 แบบ ได้แก่

1. คอมไพลเลอร์ (Compiler) เป็นตัวแปลภาษาที่ใช้หลักการแปลโปรแกรมต้นฉบับทั้งโปรแกรมและจะบันทึกไว้ในลักษณะของแฟ้มข้อมูลหรือไฟล์ เมื่อต้องการเรียกใช้งานโปรแกรมก็สามารถเรียกจากไฟล์มาใช้งานโดยไม่ต้องทำการแปลอีก ทำให้การทำงานเป็นไปอย่างรวดเร็ว โดยคอมไพลเลอร์จะตรวจสอบความถูกต้องตามหลักไวยากรณ์ของภาษาที่ใช้เขียนโปรแกรม โดยจะสร้างรายการข้อผิดพลาดของโปรแกรมเพื่อใช้เก็บโปรแกรมต้นฉบับและคำสั่งทีเขียนไม่ถูกต้องตามกฎเกณฑ์ของภาษานั้น ๆ ซึ่งจะช่วยให้ผู้เขียนโปรแกรมสามารถแก้ไขโปรแกรมได้



รูปที่ 5.1 การทำงานของตัวแปลภาษาแบบคอมไพลเลอร์

2. อินเทอร์พรีเตอร์ (Interpreter) เป็นตัวแปลภาษาที่จะแปลภาษาระดับสูง ที่ทำงานตามชุดคำสั่งที่เขียนไว้ทันที ซึ่งจะทำการแปลทีละคำสั่งแล้วให้คอมพิวเตอร์ทำตามคำสั่งนั้น เมื่อทำเสร็จแล้วจึงมาทำการแปลคำสั่งลำดับถัดไป จนจบโปรแกรม ทำให้การแก้ไขโปรแกรมทำได้ง่ายและรวดเร็ว แต่ข้อเสียคือ ถ้านำโปรแกรมนั้นมาใช้งานอีกจะต้องทำการแปลโปรแกรมใหม่ทุกครั้ง



รูปที่ 5.2 การทำงานของตัวแปลภาษาแบบอินเทอร์พรีเตอร์

รู้จักกับภาษาไพทอน

ภาษาไพทอน (python) เป็นภาษาระดับสูง (High level Language) อยู่ในลักษณะของภาษาอินเทอร์พรีเตอร์โปรแกรมมิ่ง (Interpreter Programming) คือจะประมวลผลทีละบรรทัด พัฒนาขึ้นโดย กีโด ฟาน รอสซัม (Guido van Rossum) ที่ Centrum Wiskunde & Informatica (CWI) ในประเทศเนเธอแลนด์ เมื่อปี พ.ศ. 2533 เป็นภาษาที่เหมาะกับผู้เริ่มต้นเขียนโปรแกรม เนื่องด้วยมีข้อดีหลายประการ ดังนี้

1. เขียนง่าย เมื่อเทียบกับภาษาคอมพิวเตอร์ส่วนใหญ่แล้ว ภาษาไพทอนเขียนง่าย รูปแบบคำสั่งไม่ซับซ้อน ส่งผลให้ทำความเข้าใจได้ง่าย
2. นำไปใช้งานจริงได้ แม้ว่าภาษาไพทอนจะเขียนง่าย เหมาะกับผู้เริ่มต้น แต่เมื่อมีความเชี่ยวชาญแล้ว ผู้เขียนโปรแกรมสามารถเขียนโปรแกรมขนาดใหญ่ที่ทำงานซับซ้อนและนำไปประยุกต์ใช้งานได้จริง
3. ต่อยอดง่าย ภาษาไพทอน มีไลบรารีให้ใช้งานมากมาย ช่วยให้การเขียนโปรแกรมใหม่ทำได้รวดเร็วขึ้น
4. มีผู้ใช้งานจำนวนมาก ปัจจุบันไพทอนเป็นภาษาหนึ่งที่ได้รับความนิยมทั่วโลก ทำให้มีกลุ่มผู้ใช้จำนวนมากที่พร้อมให้ความช่วยเหลือและตอบคำถามเมื่อมีปัญหา สำหรับในประเทศไทยไพทอนเริ่มได้รับความนิยมมากขึ้น โดยมีสถาบันอุดมศึกษาหลายแห่งได้บรรจุภาษาไพทอนเป็นวิชาในหลักสูตรด้านการเขียนโปรแกรมคอมพิวเตอร์แล้ว

สิ่งที่ควรรู้ก่อนการขียนโปรแกรมด้วยภาษาไพทอน

สำหรับผู้เริ่มต้นเขียนโปรแกรมคอมพิวเตอร์ด้วยภาษาไพทอน ควรทำความเข้าใจกับความรู้พื้นฐานต่อไปนี้ เพื่อให้การเขียนโปรแกรมทำได้รวดเร็วขึ้น

1. ไอดีอี ภาษาไพทอน (Python IDE) ภาษาไพทอนเป็นภาษาระดับสูง โปรแกรมที่เขียนขึ้นจึงต้องถูกแปลให้เป็นภาษาเครื่องก่อนที่จะใช้สั่งงานคอมพิวเตอร์ได้โดยตรง การแปลภาษาไพทอนนี้ต้องใช้ตัวแปลภาษา (Interpreter) ซึ่งผู้เขียนโปรแกรมต้องดำเนินการหลายขั้นตอนกว่าที่จะได้โปรแกรมที่ถูกต้องสมบูรณ์นำไปใช้งานได้ ดังนั้นผู้เขียนโปรแกรมจึงนิยมใช้ซอฟต์แวร์เพื่อการพัฒนาโปรแกรม ที่เรียกว่า ไอดีอี (IDE: Integrated Development Environment) ซึ่งประกอบด้วยเครื่องมือสำหรับแก้ไขซอร์สโค้ด (source code editor) เครื่องมือสำหรับแก้ไขจุดบกพร่องของโปรแกรม (debugger) และเครื่องมือที่ช่วนรัน (run) ซอร์สโค้ด ปัจจุบันได้มีผู้สร้างไอดีอีสำหรับภาษาไพทอนจำนวนมากให้เลือกใช้ตามความถนัดของผู้เขียน โดยไอดีอีภาษาไพทอนจะทำงานได้ทั้งในโหมดอิมมีเดียท (immediate mode) และโหมดสคริปต์ (script mode)

- โหมดอิมมีเดียท เป็นการพิมพ์คำสั่งทีละคำสั่ง แล้วตัวแปลภาษาไพทอนจะทำงานตามคำสั่งดังกล่าวทันที
- โหมดสคริปต์ เป็นการพิมพ์คำสั่งหลายคำสั่งเก็บไว้เป็นไฟล์ก่อน เมื่อผู้เขียนโปรแกรมสั่งให้ทำงาน ตัวแปลภาษาไพทอนจะทำงานตามคำสั่งในโปรแกรมตั้งแต่คำสั่งแรกจนถึงคำสั่งสุดท้ายต่อเนื่องกันไป

พื้นฐานการเขียนโปรแกรมด้วยภาษาไพทอน

1. ค่าและชนิดของข้อมูล

ข้อมูลจัดเป็นสิ่งพื้นฐานที่สุด ที่สามารถนำไปประมวลผลได้ ส่วนประกอบของข้อมูล ได้แก่ ชนิดของข้อมูล ค่าของข้อมูล ขอบเขตของข้อมูล

ชนิดของข้อมูล เช่น ตัวเลข ข้อความหรือสตริง (string)

ค่าของข้อมูล เช่น 16 , ‘Python’

ค่าของข้อมูลที่ใช้ในการประมวลผลมีหลายคลาส เช่น จำนวนเต็ม (int) , สตริง (str) , จำนวนจริง (float) , และบูล (bool) ซึ่งในภาษาไพทอนจะมีฟังก์ชัน type() ที่สามารถบอกคลาสของค่าข้อมูลได้

ตัวอย่างที่ 5.1 การตรวจสอบคลาสของข้อมูลด้วยฟังก์ชัน type()

```
>> type (‘Hello’)
<class ‘str’>
>> type (16)
<class ‘int’>
```

2. ตัวแปร

ตัวแปร (variable) คือ ชื่อที่กำหนดขึ้นสำหรับใช้เก็บค่าในหน่วยความจำ ตัวแปร จะมีชื่อ (identifier) สำหรับใช้ในการอ้างอิงถึงค่าข้อมูล ผู้เขียนโปรแกรมสามารถใช้คำสั่งกำหนดค่า (assignment statement) เพื่อใช้ในการกำหนดให้ตัวแปรมีค่าตามที่ต้องการ

ตัวอย่างที่ 5.2 การกำหนดค่าของตัวแปรในภาษาไพทอน

```
a = 3
b = 4.92
p = “Python Coding”
pi = 3.142857
```

การตั้งชื่อตัวแปร

- การตั้งชื่อตัวแปรที่ดีควรสื่อความหมายและไม่ซ้ำกับชื่อฟังก์ชันหรือคำสงวนต่าง ๆ
- ตัวพิมพ์เล็กและตัวพิมพ์ใหญ่คือคนละตัวกัน เช่น data กับ Data จะเป็นตัวแปรคนละตัวกัน
- ตัวแปรที่สร้างขึ้นต้องไม่เว้นวรรคหรือมีช่องว่าง หากต้องการให้ตัวแปรอ่านง่ายอาจใช้เครื่องหมาย underscore (_) ในการเชื่อต่อคำ เช่น room_temp
- ห้ามใช้เครื่องหมายดำเนินการต่าง ๆ เช่น + - * / เป็นต้น ในการตั้งชื่อตัวแปร
- ห้ามใช้สัญลักษณ์ เช่น $ # @ % เป็นต้น ในการตั้งชื่อตัวแปร
- ไม่เริ่มต้นด้วยตัวเลข แต่สามารถใช้ตัวเลยในตำแหน่งอื่น ๆ ได้ เช่น

  1test ไม่สามารถตั้งชื่อตัวแปร

  test1 เป็นชื่อตัวแปรได้

คำสงวน (Reserve Words) ของภาษาไพทอน มีดังนี้

and	as	assert	break	class	continue	def
del	elif	else	except	exec	finally	for
from	global	if	import	in	is	lambda
nonlocal	not	or	pass	raise	return	try
while	with	yield	True	False	None	eval

3. สตริง

การกำหนดข้อความในภาษาไพทอน ผู้เขียนโปรแกรมสามารถเลือกใช้เครื่องหมายอัญประกาศเดี่ยว (‘ ’) หรือเครื่องหมายอัญประกาศคู่ (“ ”) ได้ตามความเหมาะสม เช่น “Good” ถ้ามีความจำเป็นต้องกำหนดข้อความที่ต้องมีเครื่องหมาย “ ” เป็นส่วนหนึ่งของข้อความก็ให้เลือกใช้เครื่องหมาย ‘ ’ เป็นเครื่องหมายกำหนดข้อความแทน เช่น

คำสั่ง	ผลลัพธ์
str = ‘He said “You must come here”’ print(str)	He said “You must come here”

4. นิพจน์และตัวดำเนินการ

นิพจน์ (expression) ในภาษาไพทอน ได้แก่ ค่าของข้อมูลหรือค่าคงตัว (literal) ตัวแปร หรือการรวมกลุ่มของสิ่งดังกล่าวกับ ตัวดำเนินการ (operator) ตัวดำเนิการมีหลายประเภท เช่น ตัวดำเนินการคณิตศาสตร์ ตัวดำเนินการเปรียบเทียบ และตัวดำเนินการตรรกะ

ตัวดำเนินการคณิตศาสตร์ (arithmetic operator)

ตัวดำเนินการ	**ความหมาย**	**ตัวอย่าง**	**กำหนดตัวแปร**	**ผลลัพธ์**
( )	จัดกลุ่ม	(hour * 60) + minute	hour = 2 minute = 5	125
**	ยกกำลัง	r ** 2	r = 4	16
*	คูณ	hour * 60	hour = 3	180
/	หารจำนวนจริง	c / 5	c = 10.2	2.04
//	หารปัดเศษทิ้ง	8 // 3	-	2
%	หารเอาเศษ	28 % 24	-	4
+	บวก	a + b	a = 5 b = 38	43
-	ลบ	f - 32	f = 100	68

ตัวดำเนินการคณิตศาสตร์มีลำดับความสำคัญที่ใช้ในการประมวลผล ถ้าหากในกรณีที่มีตัวดำเนินหลายตัวในนิพจน์เดียวกัน การตัดสินใจว่าตัวดำเนินการใดจะทำงานก่อนหรือหลังขึ้นอยู่กับลำดับความสำคัญ โดยจะเรียงลำดับความสำคัญดังนี้

1. ( ) จัดกลุ่ม มีความสำคัญสูงสุด
2. ** ยกกำลัง มีความสำคัญรองลงมา
3. * / // % คุณ หาร หารปัดเศษทิ้ง หารเอาเศษ มีความสำคัญเท่ากัน โดยจะกระทำตามลำดับจากซ้ายไปขวาในนิพจน์
4. + - บวก และลบ มีความสำคัญต่ำสุด โดยจะกระทำจากซ้ายไปขวาในนิพจน์

ตัวดำเนินการเปรียบเทียบ (relational operator)

โดยกำหนดตัวแปร x = 5 และ y = 7

ตัวดำเนินการ	ความหมาย	ตัวอย่าง	ผลลัพธ์
==	เท่ากัน	x == y	False
!=	ไม่เท่ากัน	x != y	True
>=	มากกว่าหรือเท่ากับ	x >= y	False
<=	น้อยกว่าหรือเท่ากับ	x <= y	True
>	มากกว่า	x > y	False
<	น้อยกว่า	x < y	True

ตัวดำเนินการตรรกะ (logical operator)

x	y	not x	x and y	x or y	x ^ y
True	True	False	True	True	False
True	False	False	False	True	True
False	True	True	False	True	True
False	False	Ture	False	False	False

5. คำสั่งการแสงดผลข้อมูลทางจอภาพ

คำสั่งที่ใช้แสดงข้อมูลทางจอภาพ คือ คำสั่ง print() มีรูปแบบกาใช้งานดังนี้

print(“ข้อความที่ต้องการพิมพ์”) หรือ print(‘ข้อความที่ต้องการพิมพ์’)

ตัวอย่างที่ 5.3 การใช้คำสั่ง print แสดงผลข้อมูลทางจอภาพ

คำสั่ง	ผลลัพธ์
print(‘Good time’)	Good time พิมพ์ข้อความแล้วขึ้นบรรทัดใหม่
print(“Hello”, end = “ ”) print(“Python Coding”)	Hello Python Coding พิมพ์ข้อความ Python coding ต่อจากบรรทัดที่แล้ว
print(“Python\nCoding”)	Python Coding
print(“Python\“Coding\””)	Python“Coding”

รูปแบบคำสั่งแสดงผล (print) มีรูปแบบดังนี้

1. รูปแบบที่ใช้ String modulo operator (%)

```
print(*objects , sep = ' ',end = '\n',file = sys.stdout, flush=False)
```

ตัวดำเนินการ % สามารถใช้สำหรับการจัดรูปแบบสตริง โดยมีสัญลักษณ์ % แทนชนิดของข้อมูลดังนี้

% format	ชนิดข้อมูล
%s	สตริง (ข้อความ)
%d	เลขจำนวนเต็ม
%f	เลขทศนิยม
%.<number of digits>f	เลขทศนิยมมีจำนวนตัวเลขทางด้านขวาของจุดทศนิยมเท่ากับเลขหลังจุดทศนิยม
%x	เลขจำนวนเต็มในระบบเลขฐานสิบหก

ตัวอย่าง

print('Total students : %d , Average : %.2f' %(144,13.450))	**ผลลัพธ์** Total student :144, Average : 13.45
print('Python version %.1f,'%(3.0))	**ผลลัพธ์** Python version 3.0
num = 100 print("%d/2" %(num))	**ผลลัพธ์** 100/2

2. รูปแบบแสดงผลที่ใช้เมธอด format()

รูปแบบ format() จะมีการใช้เครื่องหมาย { } เพื่อแทนตำแหน่งที่ตัวแปรจะถูกแทนที่และสามารถให้คำสั่งการจัดรูปแบบได้อย่างละเอียด

ตัวอย่าง

print('a = {0}'.format(3)) print('{0} love to learn{1}'.format('Students', 'Python'))	**ผลลัพธ์** a = 3 Students love to learn Python
x = 10 y = 20 print('{0} and {1} and {0}'.format(x,y))	**ผลลัพธ์** 10 and 20 and 10
x = 12,345 y = 6.7890 print('x is {0:.2} and y is {1:.1}'.format(x,y))	**ผลลัพธ์** x is 12.34 and y is 6.7

6. การใส่คำอธิบาย (comment)

คอมเมนท์คือการเขียนบันทึกไว้เตือนความจำว่าคำสั่ง หรือส่วนของโปรแกรมทำงานอย่างไร โดยข้อความส่วนนี้จะไม่ถูกนำไปประมวลผล (Execute) การใส่ comment ทำได้ 2 วิธี คือ

1. ใส่ comment ในรูปแบบบรรทัดเดียว ทำได้โดย ใส่คำอธิบายหลังเครื่องหมาย # เช่น
   #This program is developed since May.22,2019
2. การใส่ comment ในรูปแบบหลายบรรทัด ทำได้โดยใส่คำอธิบายระหว่าง เครื่องหมาย ‘’’ ‘’’ เช่น

```
'''
  This program is first version
  I love coding
'''
```

7. คำสั่งการอ่านข้อมูลจากแป้นพิมพ์

คำสั่งที่ใช้ในการรับค่าคือคำสั่ง input( ) ซึ่งใช้รับค่าจากคีย์บอร์ดมาเก็บไว้ในรปู แบบสายอักขระ(string)

ตัวอย่างที่ 5.4 การใช้คำสั่ง input ในการรับข้อมูล

x = input("Input x: ") print(x) print(type(x))	**ผลลัพธ์คือ** Input: 2 2 <class 'str'>

จะสังเกตเห็นว่าเรากรอกข้อมูลเป็นจำนวนเต็มแต่ไพทอนจะมองว่าเป็นข้อมูลชนิด string โดยอัตโนมัติดังนั้นหากต้องการให้รับค่าเป็นชนิดของมูลแบบจำนวนเต็ม (integer) หรือจำนวนจริง(float) สามารถใช้คำสั่งดังนี้

ตัวอย่างที่ 5.5 การรับข้อมูลตัวเลขจำนวนเต็ม (integer) จากแป้นพิมพ์

n = int(input("Input number: ")) print(n) print(type(n))	**ผลลัพธ์คือ** Input number: 25 25 <class 'int'>

ตัวอย่างที่ 5.6 การรับข้อมูลตัวเลขทศนิยม (float) จากแป้นพิมพ์

pi = 3.1459265 r = float(input("Input radius: ")) area = pi * r ** 2 print("area of circle = {}" .format(area))	**ผลลัพธ์คือ** Input radius: 2 area of circle = 12.583706 Input radius: 5 area of circle = 78.648162

8. การแปลงชนิดของข้อมูล (type conversion)

เนื่องจากภาษาไพทอนมีความยืดหยุ่นในการแปลงชนิดข้อมูล ดังนั้นเราจึงสามารถจัดการแปลงข้อมูลได้ตาวัตถุประสงค์การใช้งาน ยกตัวอย่างเช่น หากเรานำตัวแปรที่เก็บค่าสตริงจำนวนสองตัวแปรมาบวกกัน ผลลัพธ์ที่ได้จะเป็นการเชื่อมต่อสตริงดังตัวอย่าง ดังนั้นหากต้องการนำตัวแปรทั้งสองมาคำนวณจำเป็นต้องมีการแปลงชนิดข้อมูลจากสตริงไปเป็นจำนวนเต็มก่อน

ตัวอย่างที่ 5.7 การแปลงชนิดข้อมูลประเภทข้อความเป็นตัวเลขจำนวนเต็ม

a = "2" b = "3" print(a+b)	**ผลลัพธ์คือ** 23
a = "2" b = "3" print(int(a) + int(b))	**ผลลัพธ์คือ** 5

อีกกรณีคือการใช้คำสั่ง input ซึ่งจะเป็นการรับค่าแบบสตริง ดังนั้นหากเราต้องการนำไปคำนวณเราสามารถแปลงชนิดข้อมูลได้ดังตัวอย่าง

ตัวอย่างที่ 5.8 การแปลงชนิดข้อมูลประเภทข้อความเป็นตัวเลขทศนิยม

x = float(input("Enter a number: ")) y = float(input("Enter another number: ")) print(x+y)	**ผลลัพธ์คือ** Enter a number: 1 Enter another number: 2 3.0
x = float("123" * int(input("Enter a number:"))) print(x)	**ผลลัพธ์คือ** Enter a number:3 123123123.0