พื้นฐานการพัฒนาโปรแกรมและ Coding ด้วยภาษา Python

ดร. ธนวิชญ์ อนุวงศ์พินิจ

คณะวิศวกรรมศาสตร์

สถาบันเทคโนโลยีพระจอมเกล้าเจ้าคุณทหารลาดกระบัง

thanavit.an@kmitl.ac.th

ภาพรวมเนื้อหาในการอบรมวันนี้

0. แนวคิดในการพัฒนาโปรแกรมและการประยุกต์ใช้ภาษา Python
1. คำสั่งการแสดงผล (Print)
2. ตัวแปร (Variable)
3. ชนิดข้อมูล (Data Type)
4. คำสั่งการนำเข้าข้อมูล (Input)
5. คำสั่งการดำเนินการ (Operation)
6. คำสั่งเงื่อนไข (Condition : If-else)
7. คำสั่งแบบวนซ้ำ (Loop)

AIoT
By IoT & Information Engineering
School of Engineering KMITL

PYTHON

ภาษา Python คืออะไร

Python เป็นภาษาเขียนโปรแกรมระดับสูงที่เป็นลักษณะของ Script ที่ใช้กันอย่างกว้างขวางในการเขียนโปรแกรมสำหรับวัตถุประสงค์ทั่วไป ภาษา Python นั้นสร้างโดย Guido van Rossum และถูกเผยแพร่ครั้งแรกในปี 1991

Python นั้นเป็นภาษาแบบ interpreter ที่ถูกออกแบบโดยมีปรัญชาที่จะทำให้โค้ดอ่านได้ง่ายขึ้น และโครงสร้างของภาษานั้นจะทำให้โปรแกรมเมอร์สามารถเข้าใจแนวคิดการเขียนโค้ดโดยใช้บรรทัดที่น้อยลงกว่าภาษาอย่าง C++ และ Java ซึ่งภาษานั้นถูกกำหนดให้มีโครงสร้างที่ตั้งใจให้การเขียนโค้ดเข้าใจง่ายทั้งในโปรแกรมเล็กไปจนถึงโปรแกรมขนาดใหญ่

ข้อดีของ Python

1. เขียนง่าย เรียนรู้ง่าย เข้าใจง่าย
2. มี Community ขนาดใหญ่ เป็น Open Source ใช้งานได้ทุกระบบปฏิบัติการ
3. มี Library ให้ใช้งานหลากลาย และใช้งานง่าย
4. เหมาะกับงาน Data Science, AI (Artificial Intellgience) ที่เป็นที่ต้องการอย่างมากของตลาด
5. เป็นที่ต้องการของหลายๆ บริษัท มีการใช้งานอย่างแพร่หลาย

Python ถูกนำไปใช้ที่ไหนบ้าง ?

Python นำไปใช้ทำอะไรได้บ้าง ?

สามารถนำไปบูรณาการร่วมกับวิชาคณิตศาสตร์,วิทยาศาสตร์

ทำให้การเรียนวิชาคณิตศาสตร์,วิทยาศาสตร์มองเห็นภาพการนำไปประยุกต์ใช้งานมากขึ้น ไม่น่าเบื่อ เป็นการบูรณาการในหลากหลายสาขาวิชา

Python นำไปใช้ทำอะไรได้บ้าง ?

สามารถนำไปเขียนโปรแกรมควบคุมอุปกรณ์อัจฉริยะ

เช่น การใช้ MicroPython, RaspberryPi

Python นำไปใช้ทำอะไรได้บ้าง ?

สามารถนำไปพัฒนาเว็บไซต์

เช่น การใช้ Flask, DJANGO

Python นำไปใช้ทำอะไรได้บ้าง ?

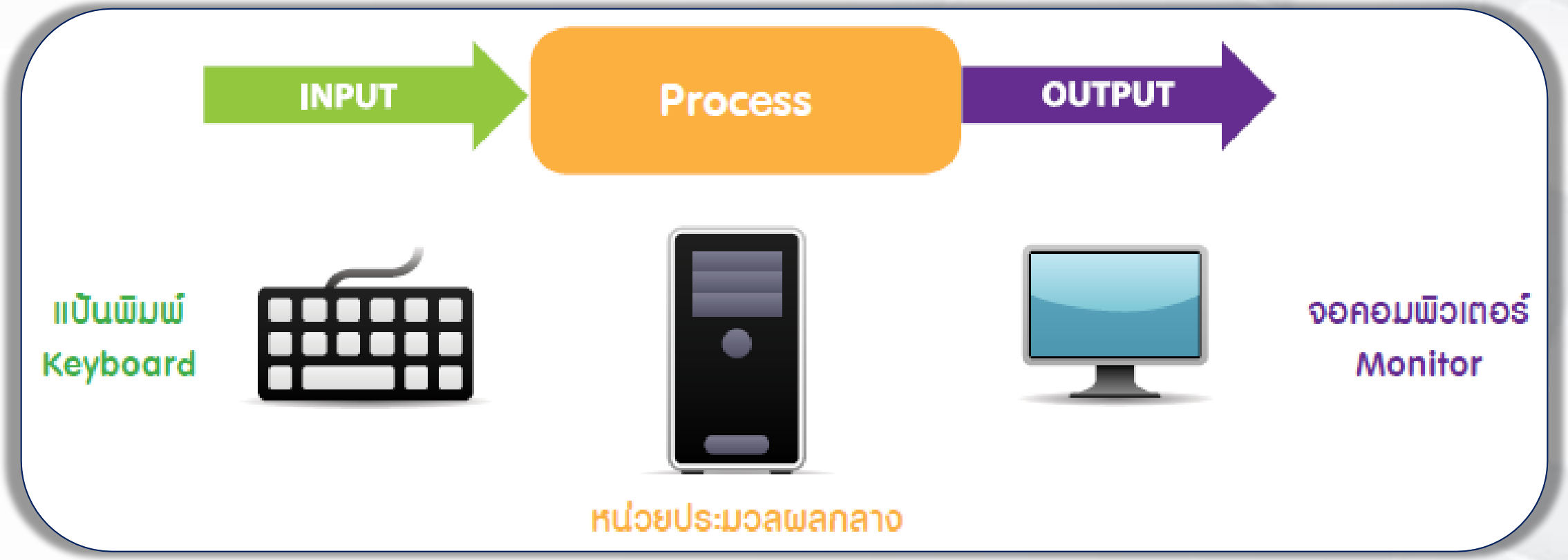
สามารถนำไปใช้กับงานทางด้านวิทยาศาสตร์ข้อมูล (Data Science)

เช่น การใช้เพื่อวิเคราะห์ข้อมูล แสดงผลข้อมูล
รวมถึงทางด้านปัญญาประดิษฐ์ การประมวลผลภาพ
การเรียนรู้ของเครื่อง ระบบแนะนำฯ

เริ่มต้นเขียนโปรแกรม

ก่อนเริ่มต้นทำการเขียนโปรแกรม Python มาทบทวนหลักการทำงานของคอมพิวเตอร์โดยผู้เขียนโปรแกรมต้องออกแบบอัลกอริทึมก่อน โดยต้องวิเคราะห์ว่า มีข้อมูลนำเข้าอะไร ประมวลผลด้วยวิธีการอย่างไร และต้องการให้แสดงผลอย่างไร ซึ่งเป็นหลักการทำ งานของคอมพิวเตอร์ที่ต้องการ หน่วยข้อมูลเข้า หน่วยประมวลผล และหน่วยแสดงผลหรือข้อมูลออก

เริ่มต้นเขียนโปรแกรม



เริ่มต้นเขียนโปรแกรม

ถ้าแบ่งคำสั่งของโปรแกรมภาษา Python ออกตามหลักการทำงานของคอมพิวเตอร์คือ การใช้แป้นพิมพ์ส่งข้อมูล การประมวลผลด้วยหน่วยประมวลผลกลาง และการแสดงผลโดยใช้จอคอมพิวเตอร์จะทำให้สามารถแบ่งหัวข้อการเรียนรู้คำสั่งต่าง ๆ

คำสั่งนำข้อมูลเข้า (Input)	คำสั่งประมวลผล	คำสั่งแสดงผลข้อมูล (output)
คำสั่งกำหนดค่าตัวแปร ชื่อตัวแปร = ...	ตัวดำเนินการทาง คณิตศาสตร์ เปรียบเทียบ	คำสั่งแสดงผลข้อมูลออกทางหน้าจอ Print()
คำสั่งนำข้อมูลเข้าจากแป้นพิมพ์ input()	ตัวดำเนินการ ตรรกะ กำหนดค่า สมาชิก	การกำหนดรูปแบบ การแปลงข้อมูล การแสดงผลข้อมูล
ชนิดของข้อมูล เบื้องต้น Number, Sequence Type ,Boolean	คำสั่งการทำงานแบบมี เงื่อนไข if ....else...	* การส่งออกไฟล์
ฟังก์ชันและการสร้างฟังก์ชัน def() *Library random datetime turtle	ตัวดำเนินการ ตรวจสอบเงื่อนไข	
*ชนิดของข้อมูล Set , Dictionary * การเรียกใช้ไฟล์	คำสั่งการทำงานแบบวนซ้ำ for .... in .... while	

โปรแกรมสำหรับการเขียนโปรแกรมไพธอน

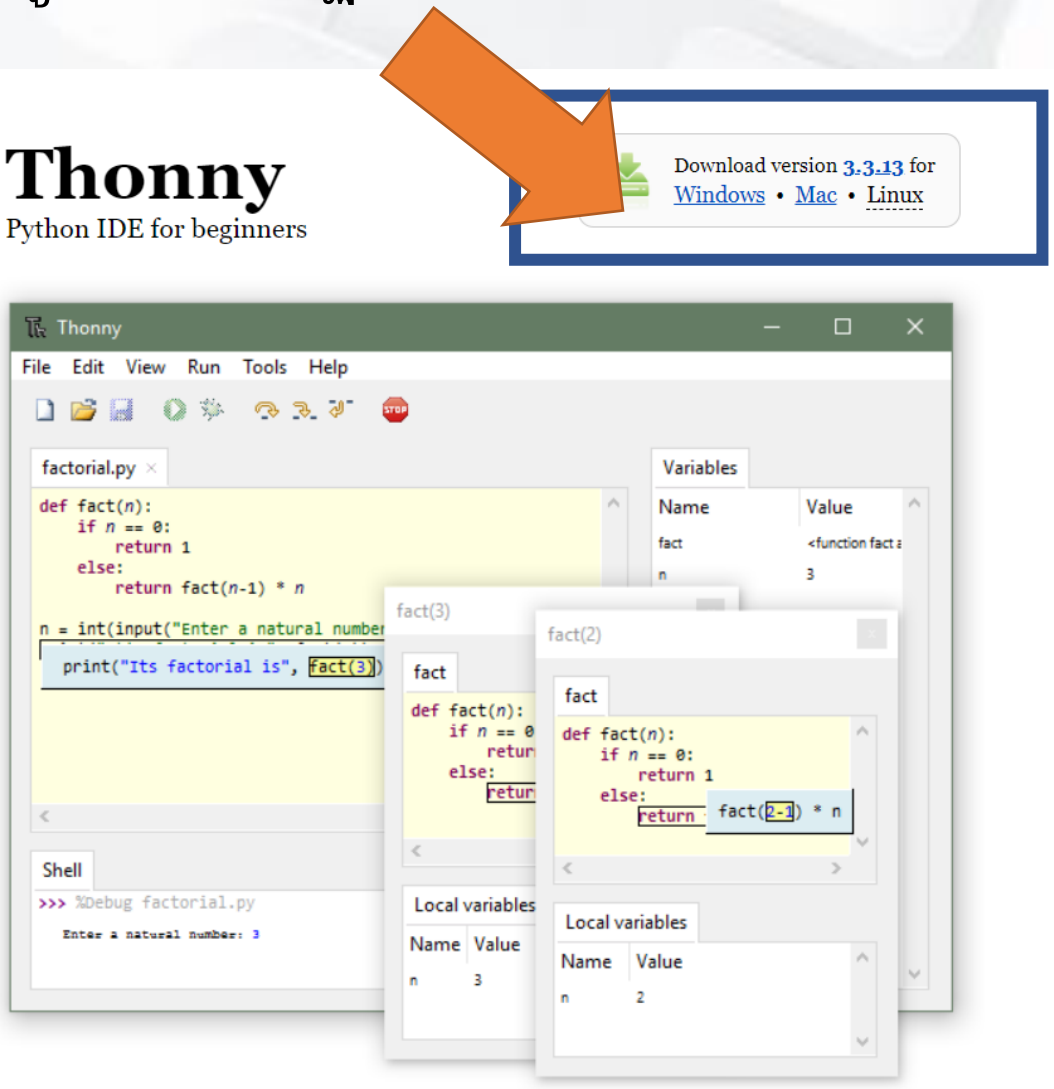
- ภาษาไพธอนนั้นมีการทำงานที่เป็นแบบ Interactive ที่สามารถสั่งการทำงานได้ทันที แต่อย่างไรก็ตาม ได้มีการพัฒนา “IDE (Integrated Development Environment” เพื่อใช้ในการพัฒนาโปรแกรมให้โดยง่าย เช่น
- Thonny
- Mblock5
- PyCharm
- Jupyter Notebook
- หรือการใช้งานบนเว็บไซต์ระบบคลาวด์เช่น Google Colab

สำหรับในการอบรมวันนี้ จะใช้ซอฟต์แวร์ “Thonny”

Thonny

- ดาวน์โหลดได้ที่ https://thonny.org/

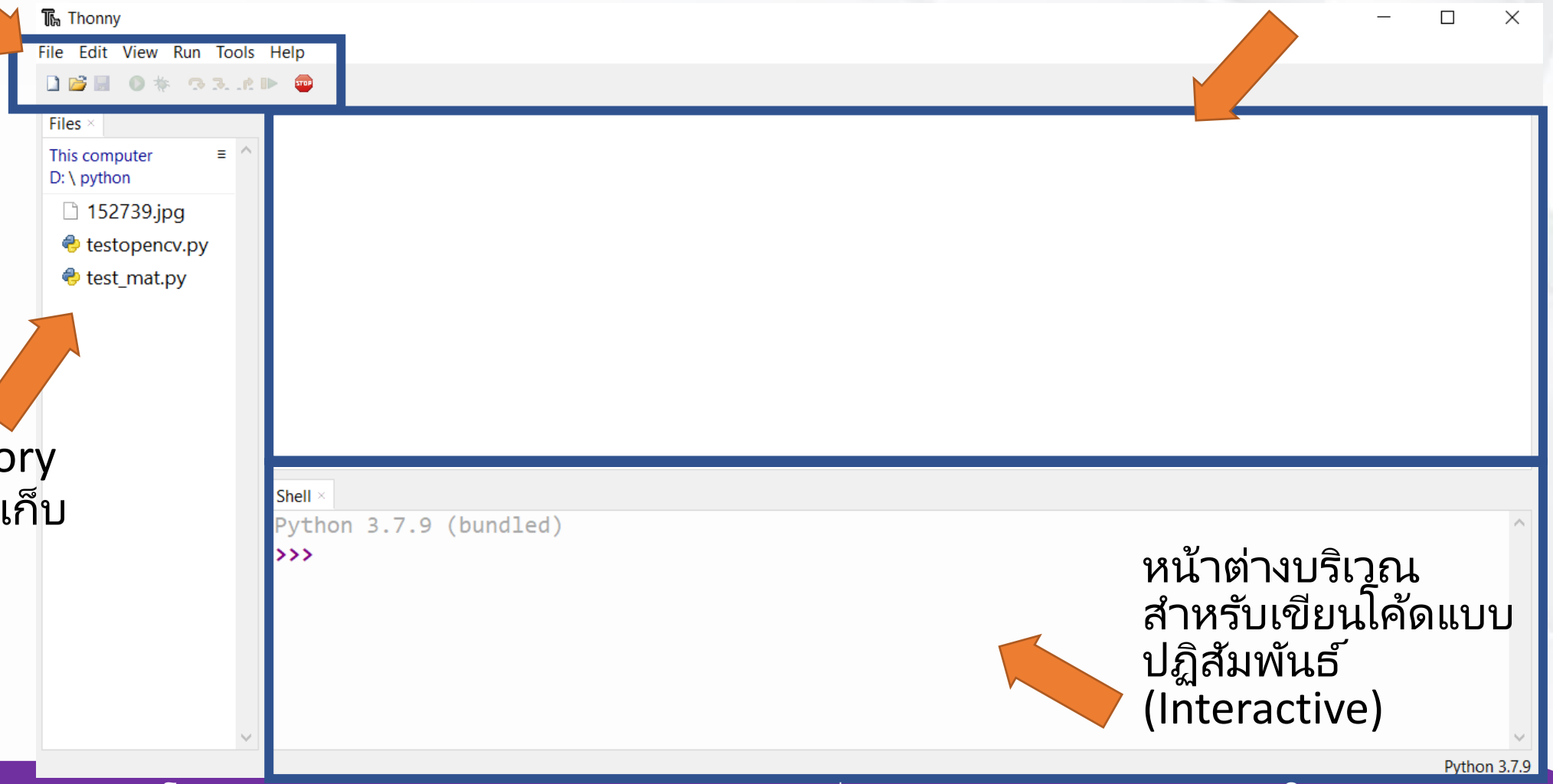
คลิกเลือก “windows” สำหรับ
ผู้ใช้ระบบปฏิบัติการ Windows 10



หน้าต่างซอฟต์แวร์ Thonny

เมนูบาร์

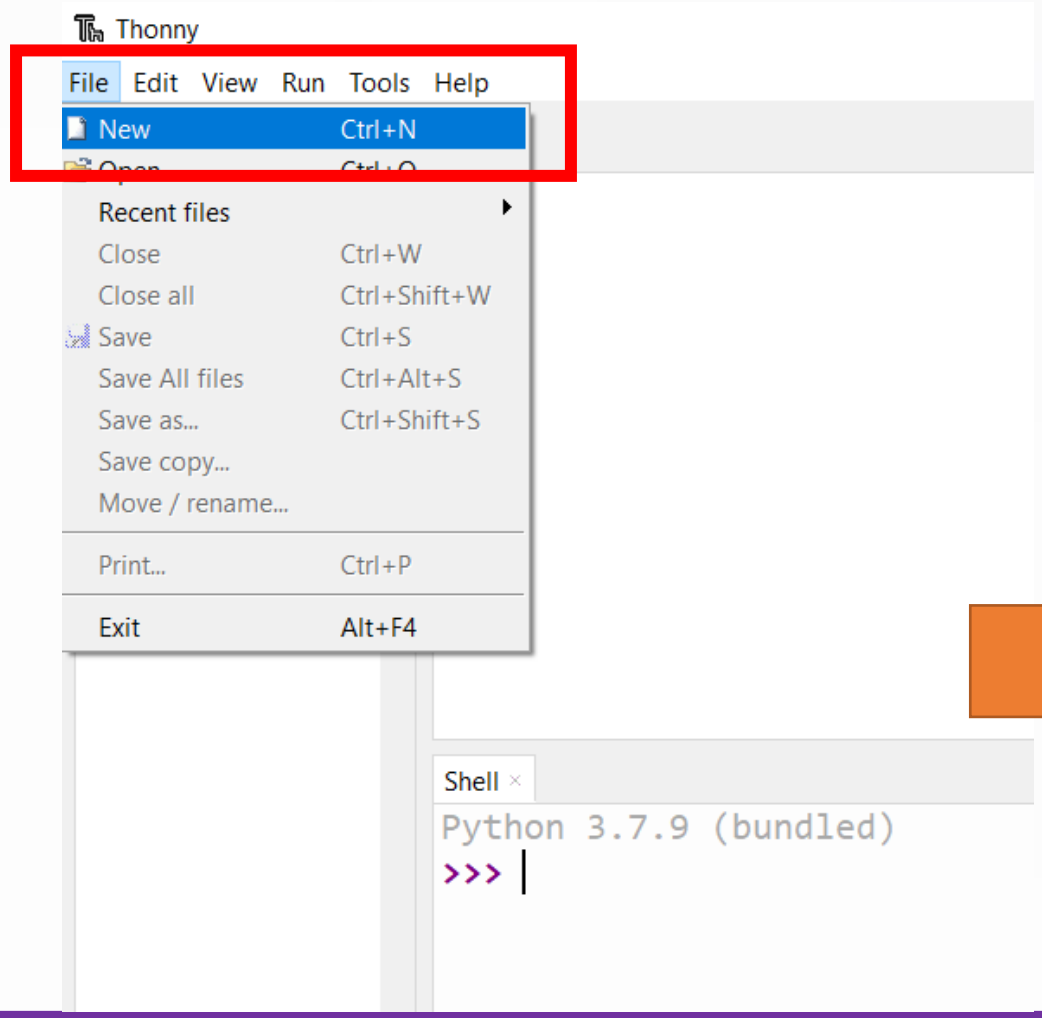
หน้าต่างบริเวณ
สำหรับเขียน Code

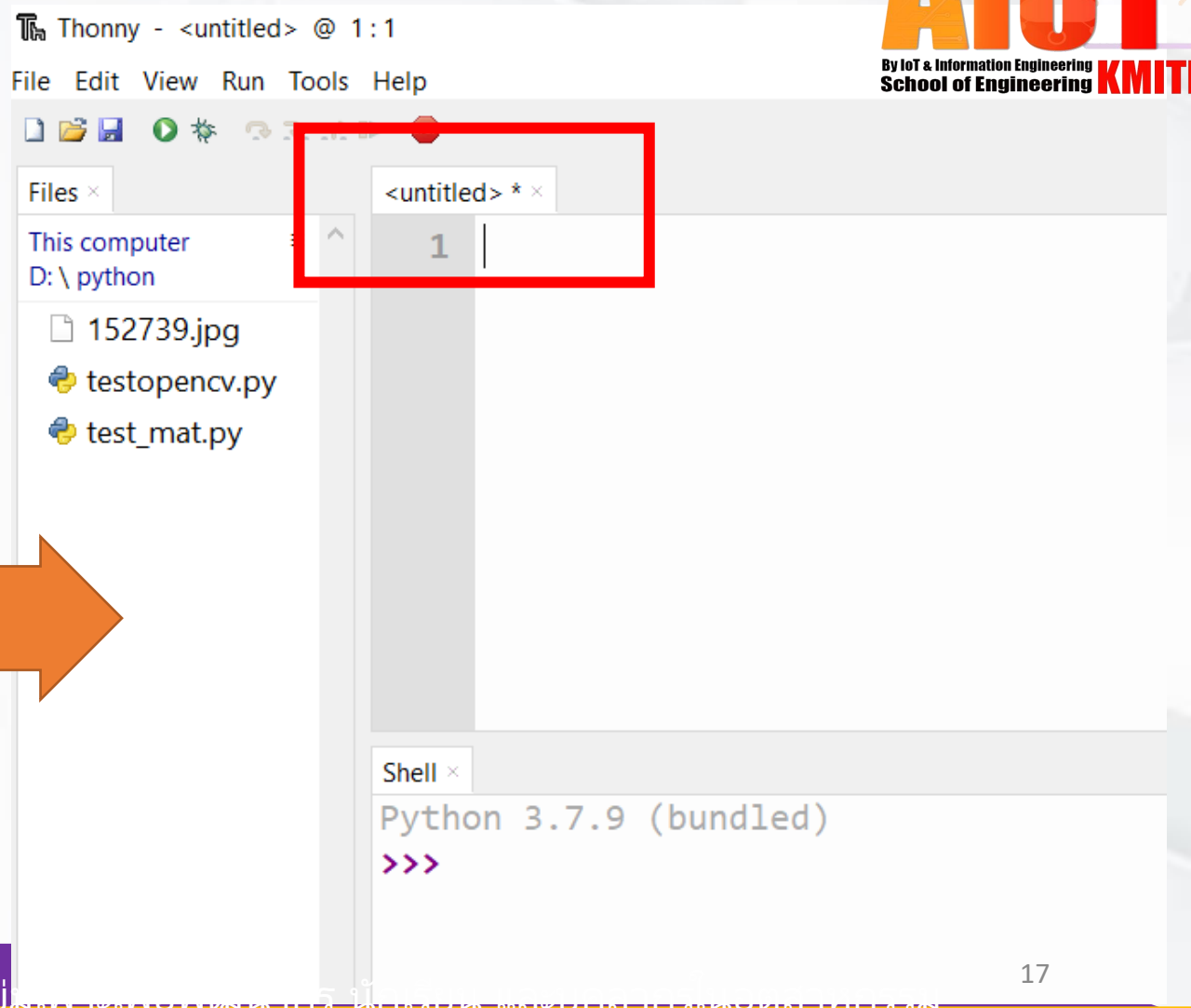


Directory
สำหรับเก็บ
ไฟล์

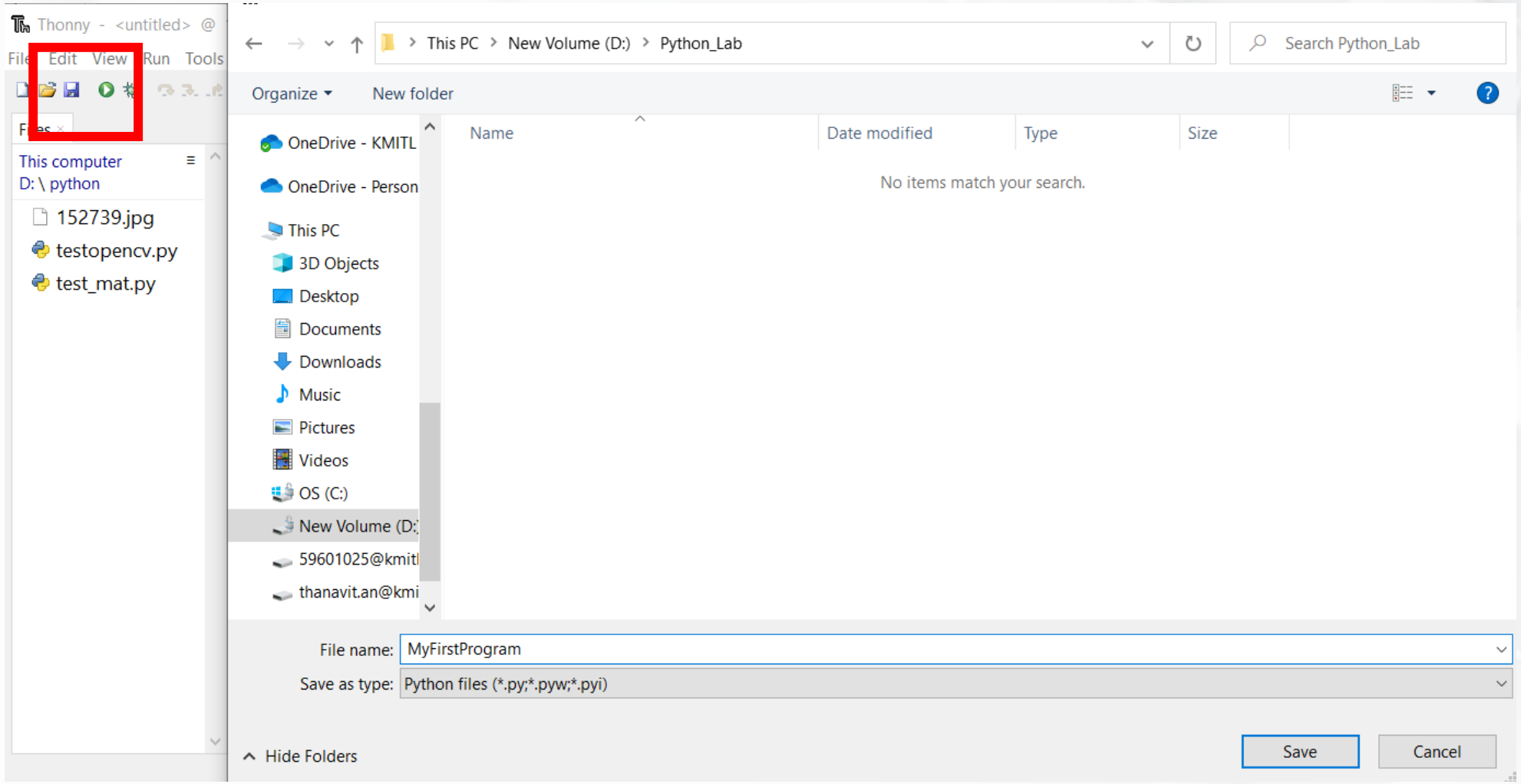
หน้าต่างบริเวณ
สำหรับเขียนโค้ดแบบ
ปฏิสัมพันธ์
(Interactive)

ที่โปรแกรม Thonny เลือกเมนู File > New จากนั้นจะขึ้น Tab ใหม่ที่หน้าต่างเขียนโค้ดขึ้นมาว่า <untitled>

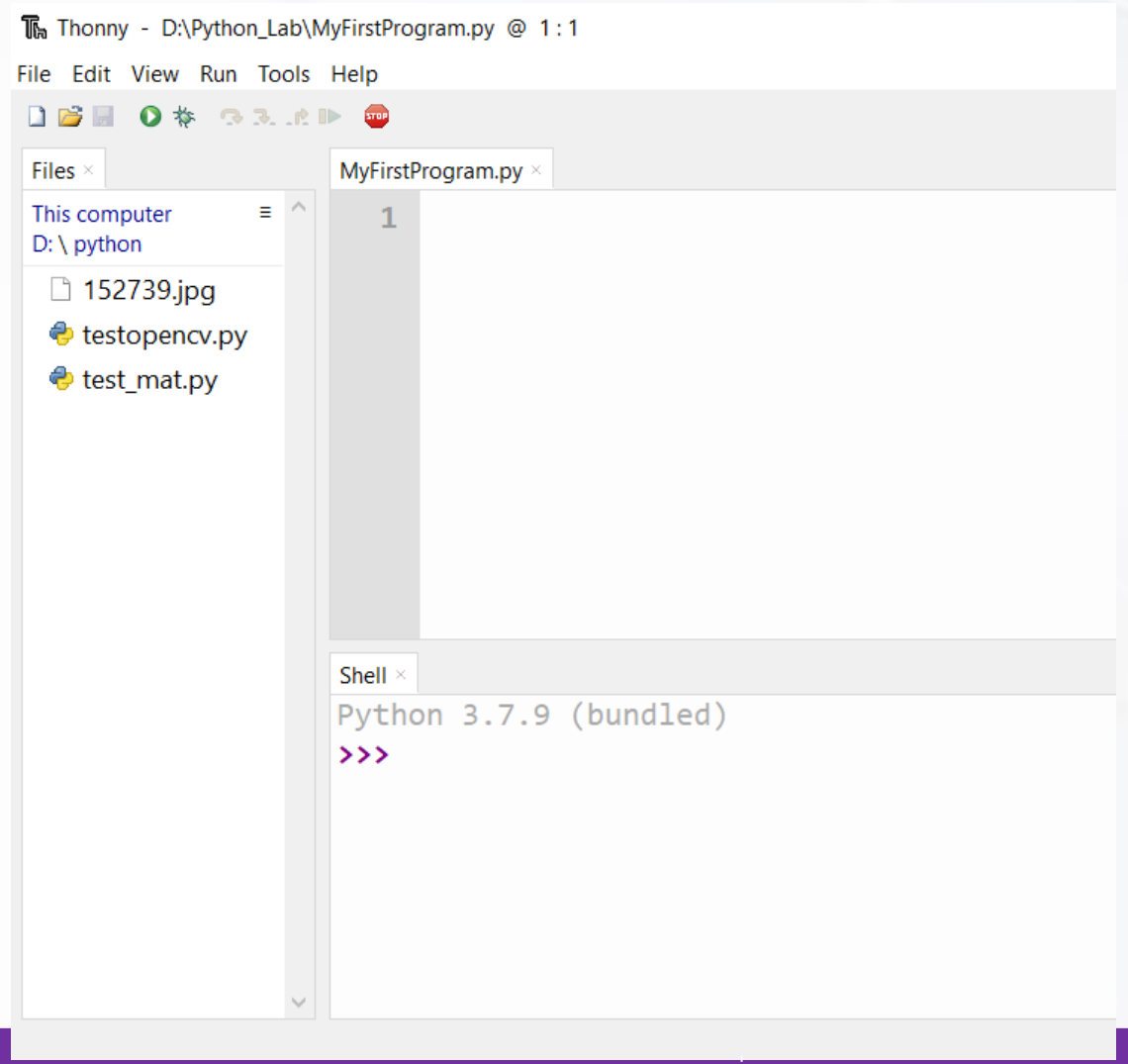




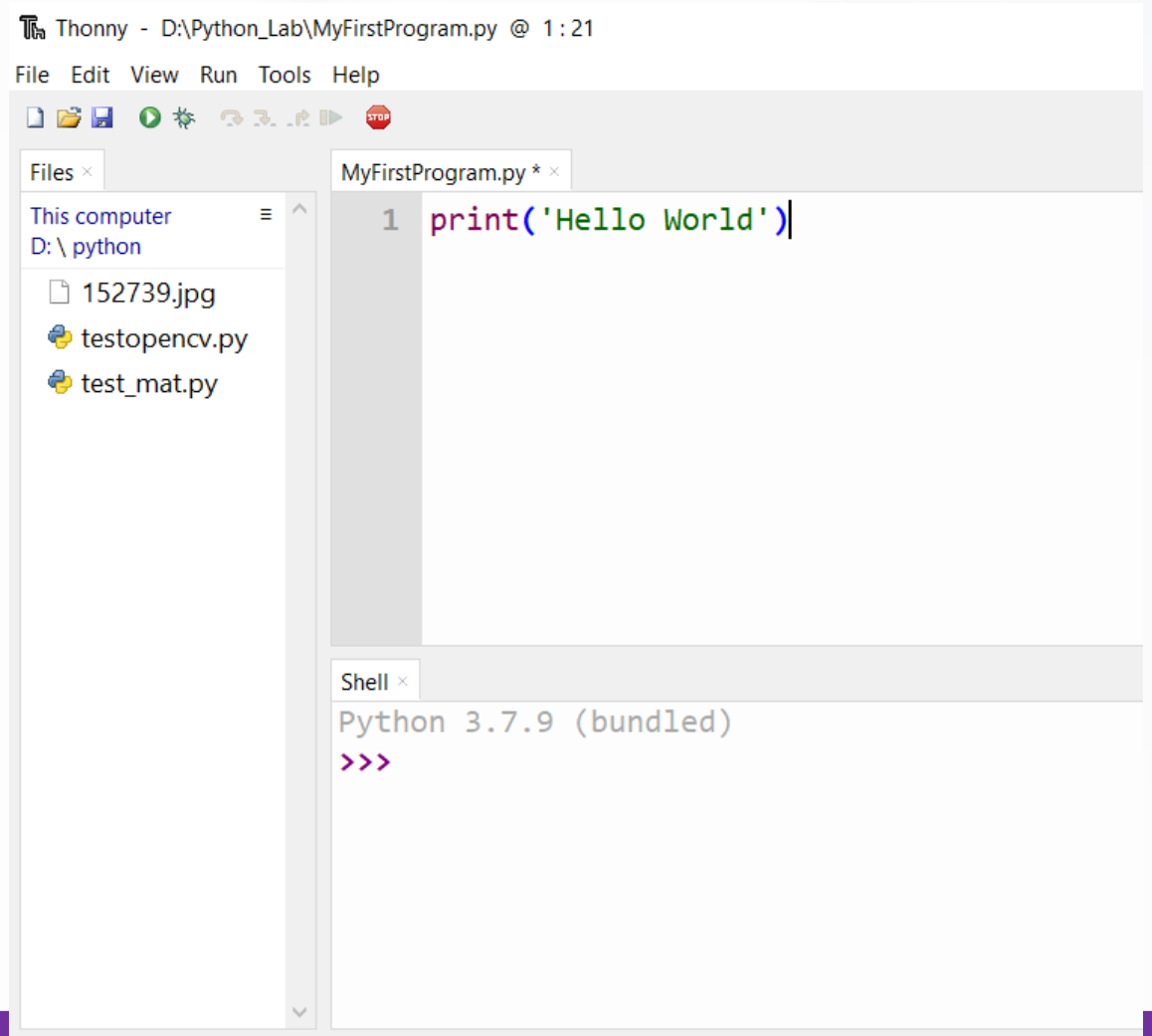
กดปุ่ม Save เพื่อทำการบันทึกไฟล์

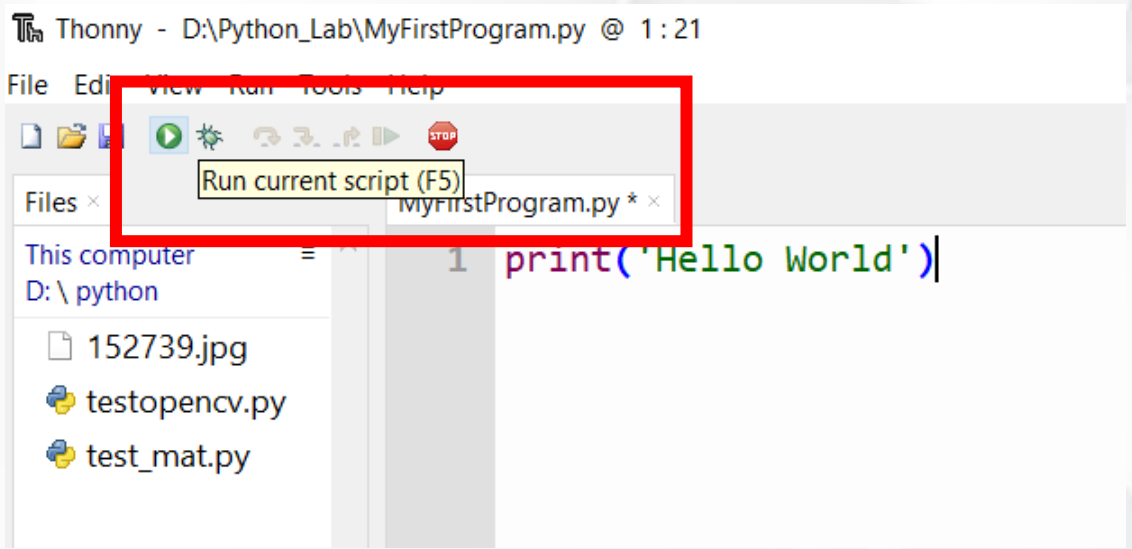


ชื่อ Untitled เปลี่ยนเป็นชื่อไฟล์ที่เราตั้งชื่อ

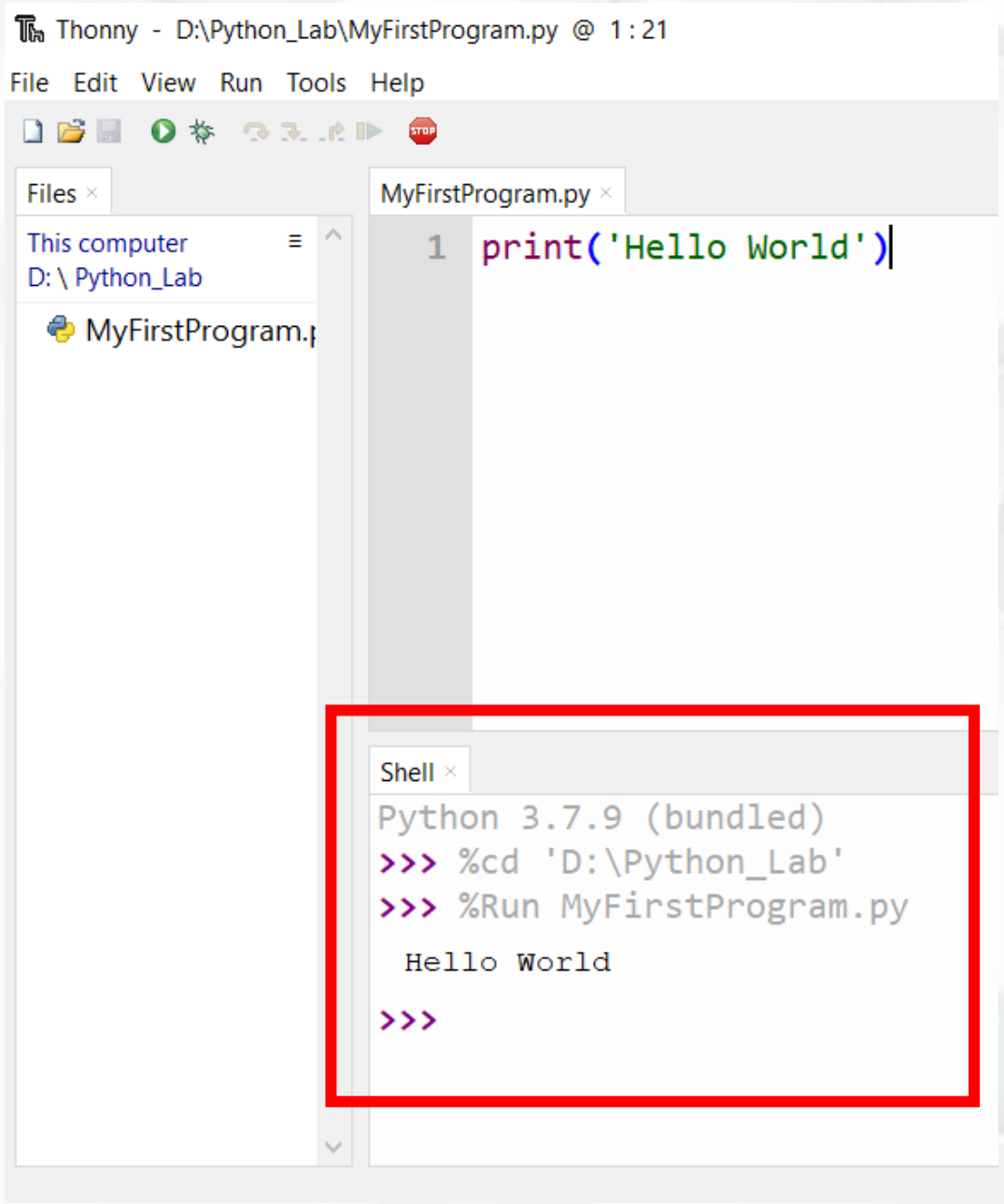


ทดลองเขียนคำสั่ง print('Hello World') จากนั้นกดปุ่ม Play สีเขียว (Run Current Script หรือกดปุ่ม F5





โปรแกรมของเราจะถูกรัน และแสดงผลที่ Shell ด้านล่าง



AIoT
By IoT & Information Engineering
School of Engineering KMITL

AIoT
By IoT & Information Engineering
School of Engineering KMITL

PRINT COMMAND

คำสั่งแสดงผลทางหน้าจอ

1. PRINT (ข้อมูล)
ออกทางหน้าจอแสดงผล

คำสั่งการแสดงค่าบนหน้าจอคอมพิวเตอร์

```
print(ค่าที่ 1,ค่าที่2, ..., sep = ' ', end = '\n')
```

เป็นคำสั่งแสดงค่าที่ต้องการบนหน้าจอโดยสามารถแสดงค่าได้ทั้งตัวอักษร ตัวเลข และอื่น ๆ (ขึ้นอยู่กับชนิดของข้อมูล) หรือ ตัวแปร ได้หลายค่าโดยใช้เครื่องหมาย , คั่น โดยหากค่าที่ใส่เป็นจำนวนไม่ต้องใส่ " " double quotes หรือ ' ' single quotes และ: ถ้าเป็นตัวอักษรต้องใส่ หากไม่ใส่จะนับเป็นค่าตัวแปร ซึ่งจะจบด้วยขึ้นบรรทัดใหม่ \n เสมอ โดยมีตัวอย่างดังต่อไปนี้

ตัวอย่างการใช้งานคำสั่ง print

ตัวอย่างคำสั่ง	รายละเอียดคำสั่ง	ผลลัพธ์ของโปรแกรมที่เขียน
Print ("Lipda","chan")	ให้แสดงอักษรคำว่า Lipda และ: chan ออกทางหน้าจอ	Lipda chan
Print (2.3,1.8)	ให้แสดงค่าจำนวน 2.3 และ: 1.8	2.3 1.8

AIoT By IoT & Information Engineering School of Engineering KMITL

คำสั่ง backslash(\) การจัดการแสดงผลบนหน้าจอ Escape character

escape character	ความหมาย	escape character	ความหมาย
\' หรือ \"	การใส่ single หรือ double quotes	\\	การใส่ Back-slash
\n	new line - การขึ้นบรรทัดใหม่	\r	Carriage Return ย้ายตำแหน่ง Cursor ไปซ้ายสุดของบรรทัด
\t	ย้ายตำแหน่ง Cursor ไป 1 Tab แนวนอน	\v	ย้ายตำแหน่ง Cursor ไป 1 Tab แนวตั้ง
\b	ย้ายตำแหน่ง Cursor ไป 1 Backspace	\f	form feed - ขึ้นหน้าใหม่

1.1 ตัวอย่างการใช้งานคำสั่ง print

ตัวอย่างคำสั่ง	รายละเอียดคำสั่ง	ผลลัพธ์ของโปรแกรมที่เขียน
print ("Lipda\tPola","\tLipda\tPola")	ให้แสดงผลคำว่า Lipda Pola 2 ครั้งโดยใส่ \t หรือ Tab แนวนอน	Lipda Pola Lipda Pola
Print (2.3+1.8)	ให้แสดงค่าผลบวก 2.3+1.8 ออกทางหน้าจอ	4.1

หมายเหตุ : 1. การแบ่งข้อมูลที่จะแสดงหรือคำสั่งต้องมีเครื่องหมาย , คั่น

1.2 String with Single Quotes

Programming

```
1   print("I'm fine")
2   print('I\'m fine')
3
```

Result

```
I'm fine
I'm fine
```

1. print(”I’m fine”) แบบ Double Quote สามารถใช้ ’ ได้เลย
2. print(’I\’m fine’) แบบ Single Quote จะต้องใช้เครื่องหมาย \ ด้านหน้า เครื่องหมาย ’

1.3 String

```
1   print('My name is "Lipda Chan."')
2
```

```
My name is "Lipda Chan."
>
```

การใช้คำสั่ง print แบบ Single Quotes สามารถใส่ " " ในประโยคได้เลย

```
1   print("My name is \"Lipda Chan.\"")
2
```

```
My name is "Lipda Chan."
>
```

การใช้คำสั่ง print แบบ Double Quotes จะต้องใส่เครื่องหมาย \ ก่อนเปิดและก่อนปิดประโยคข้อความ

1.4 ใช้ \ String with Backslash

Programming

```
1  print("/\\")
2  print("/\/\/\\")
3
```

Result

```
/\
/\/\/\
```

การใช้เครื่องหมาย Backslash \ ในประโชค ถ้าตำแหน่งสุดท้ายของประโยคมีเครื่องหมาย \ จะต้องมี เครื่องหมาย \ หลังประโยคอีกครั้ง

1.5 ตัวอย่างการใช้งานคำสั่ง print ที่ 3

ตัวอย่างคำสั่ง	รายละเอียดคำสั่ง	ผลลัพธ์ของโปรแกรมที่เขียน
Print (2.3,1.8,sep=",")	ให้แสดงค่าจำนวน 2.3 และ 1.8 และมี , คั่นระหว่างค่าทั้งสอง	2.3,1.8
print (2.3,1.8,sep='\n')	ให้แสดงค่าจำนวน 2.3 และ 1.8 โดยให้ขึ้นบรรทัดใหม่	2.3 1.8

หมายเหตุ : 2. หากต้องการแสดงเครื่องหมายระหว่างค่าที่แสดงให้ใช้คำสั่ง sep = '....' (Seperate)

3. หากต้องการขึ้นบรรทัดใดระหว่างข้อมูล ให้ เพิ่ม sep = '\n' (Escape character)

4. โดยปกติคำสั่งแต่ละคำสั่งจะขึ้นบรรทัดใหม่ หากไม่ต้องการขึ้นบรรทัดใหม่ให้ใส่ end

1.6 แบบหลายบรรทัด Multiple Line

Programming

```
1   print("""Hello
2   I love python
3   100
4         """)
5
```

Result

```
Hello
I love python
100
```

ในตัวอย่าง โปรแกรมแสดงผลข้อความออกทางหน้าจอ
ซึ่งข้อความขึ้นบรรทัดใหม่โดยใช้คำสั่ง print ครั้งเดียว แบบ Double Quotes

First Project

1) ให้เขียนโปรแกรมแนะนำตัวเอง

ชื่อ นามสกุล

สังกัด

ความตั้งใจหลักการอบรมครั้งนี้

ตัวอย่าง OUTPUT :

```
>>> %Run 01_BasicPrint.py

Thanavit Anuwongpinit
IoT System and Information Engineering, KMITL
อยากช่วยพัฒนาการศึกษาของประเทศไทย
```

2. ตัวแปรและการกำหนดค่าของตัวแปร

2. การกำหนดค่าตัวแปร (Variable) คือ การระบุค่า หรือ การตั้งค่าข้อมูลเข้า เพื่อนำมาประมวลผลตามเงื่อนไขที่กำหนดมาในปัญหา เพื่อให้ได้ข้อมูลออกหรือผลลัพธ์ของปัญหา

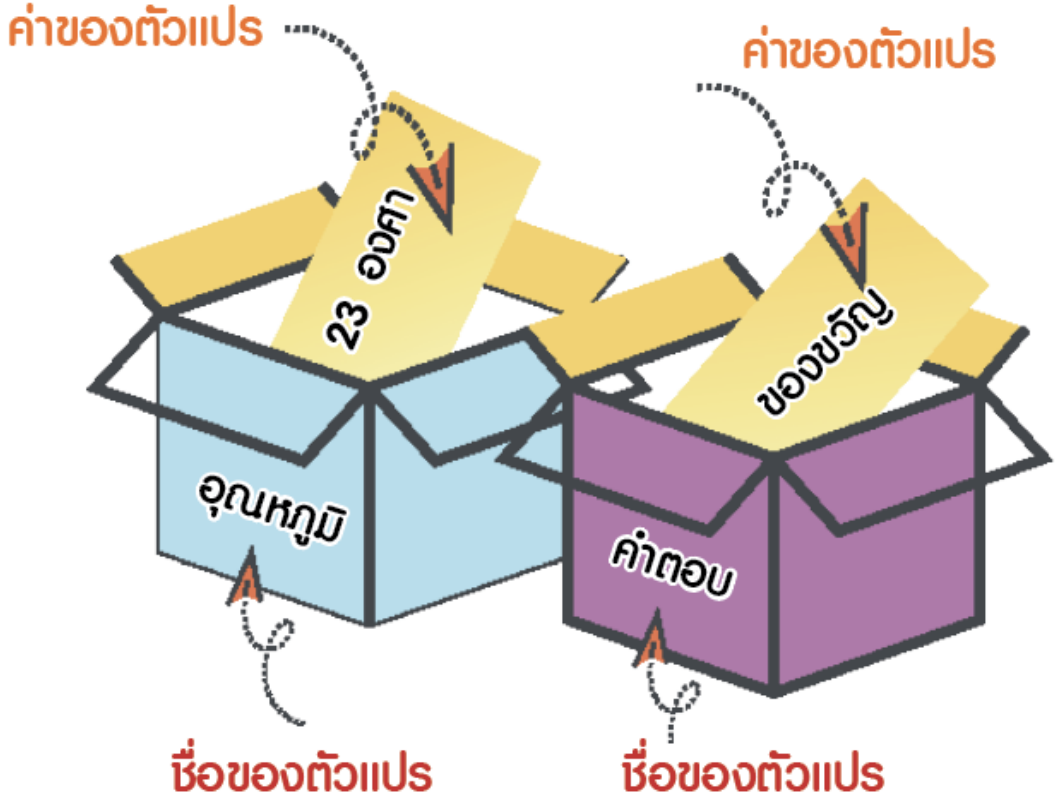
การกำหนดชื่อและค่าของตัวแปร

ชื่อของตัวแปร	กำหนดเป็น	ข้อมูลตัวอักษร
ค่าของตัวแปร	สามารถกำหนดเป็น	ตัวอักษรหรือตัวเลข ก็ได้

เปรียบเทียบได้
กับกล่องเก็บข้อมูล

ของที่อยู่ในกล่อง
คือ ค่าของตัวแปร

ชื่อของกล่องเก็บข้อมูล
คือ ชื่อของตัวแปร



การประกาศตัวแปร (Variable Declaration)

คือ การกำหนด พื้นที่หรือกล่อง กล่องสำหรับเก็บข้อมูล ภายใต้ชื่อกล่องหรือตัวแปรที่กำหนด โดยการกำหนดค่าตัวแปรใน Python ไม่มีต้องมีการกำหนด ชนิดของข้อมูล เหมือนโปรแกรมภาษา C , Javascript โดยในการสร้างตัวแปรใช้ ตัวดำเนินการ (=) บอกค่าตัวแปร

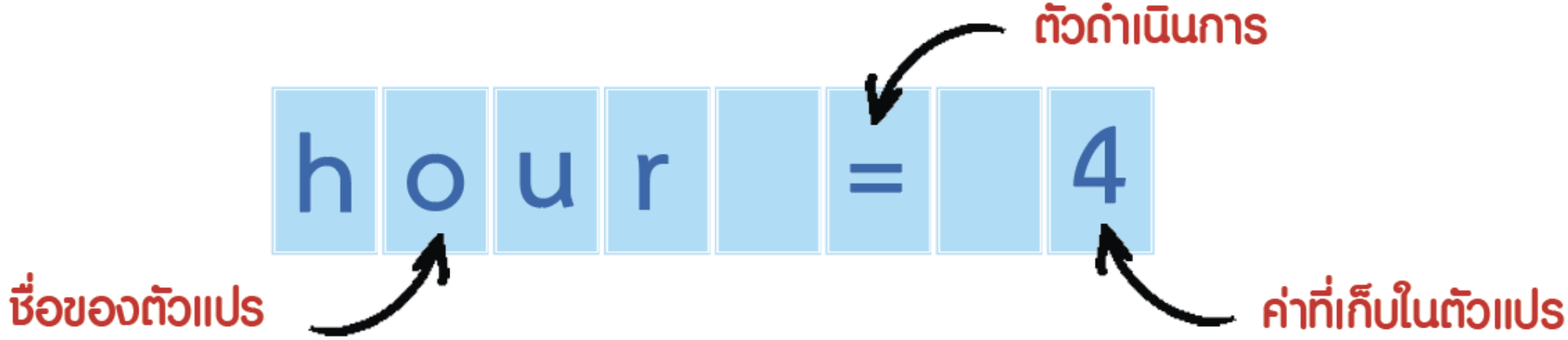
~~int a = 20~~

~~float c = 3.14~~

a = 20

c = 3.14

ตัวอย่าง สร้างตัวแปร ชื่อว่า hour ซึ่งมีค่า เท่ากับ 4



ตัวแปร เปรียบเสมือน ที่เก็บข้อมูล เป็นชื่อที่เราเอาไว้เมื่อต้องการเรียกใช้งาน

ข้อกำหนดการตั้งชื่อตัวแปร

1. ห้ามตั้งชื่อเหมือนกับคำสงวนในภาษา Python(Reserved Word) เช่น int float str ซึ่งหากไม่แน่ใจสามารถพิมพ์ help("keywords") ในโปรแกรมภาษา Python
2. ไม่ควรขึ้นต้นตัวแปรด้วยตัวอักษรภาษาอังกฤษตัวพิมพ์ใหญ่ ใช้ภาษาไทยไม่ได้
3. ห้ามขึ้นต้นด้วยตัวเลขและห้ามใช้เครื่องหมาย !,@, #, $, %, ^, &, *, (,), -, =, \, |, +, ~
4. หากชื่อตัวแปรเป็นคำประสมระหว่างคำหลายคำให้ใช้ _ (underscore) ใช้เว้นวรรคไม่ได้
5. ตัวพิมพ์ใหญ่และเล็กถือว่าแตกต่างกัน เช่น ตัวแปร age และ Age เป็นคนละตัวแปร
6. สามารถตรวจสอบ ชนิดของตัวแปร ได้โดยใช้คำสั่ง Print (Type(ชื่อตัวแปร))
7. ถ้าค่าของตัวแปรอยู่ในเครื่องหมาย " " หรือ ' ' ชนิดของตัวแปรจะเป็น ตัวอักษร
8. ถ้าค่าของตัวแปรที่เป็น ตัวเลข ชนิดของตัวแปรจะเป็น จำนวน (Number)

ข้อกำหนดการตั้งชื่อตัวแปร

1. ห้ามตั้งชื่อเหมือนกับคำสงวนในภาษา Python(Reserved Word) เช่น int float str ซึ่งหากไม่แน่ใจสามารถพิมพ์ help("keywords") ในโปรแกรมภาษา Python

False	break	except	import	raise
none	class	finally	in	return
True	continue	for	is	try
and	def	form	lambda	while
as	del	global	nonlocal	with
assert	elif	if	not	yield

2. ไม่ควรขึ้นต้นตัวแปรด้วยตัวอักษรภาษาอังกฤษตัวพิมพ์ใหญ่ ใช้ภาษาไทยไม่ได้

ไม่ควร		ควรใช้
Name	→	name
Surname	→	surname

ไม่ได้

ชื่อ

นามสกุล

3. ห้ามขึ้นต้นด้วยตัวเลข

ใช้ไม่ได้		ใช้ได้
1 level	→	level_1
1 _name	→	name_1
name 1	→	name_1
id no	→	Id_no

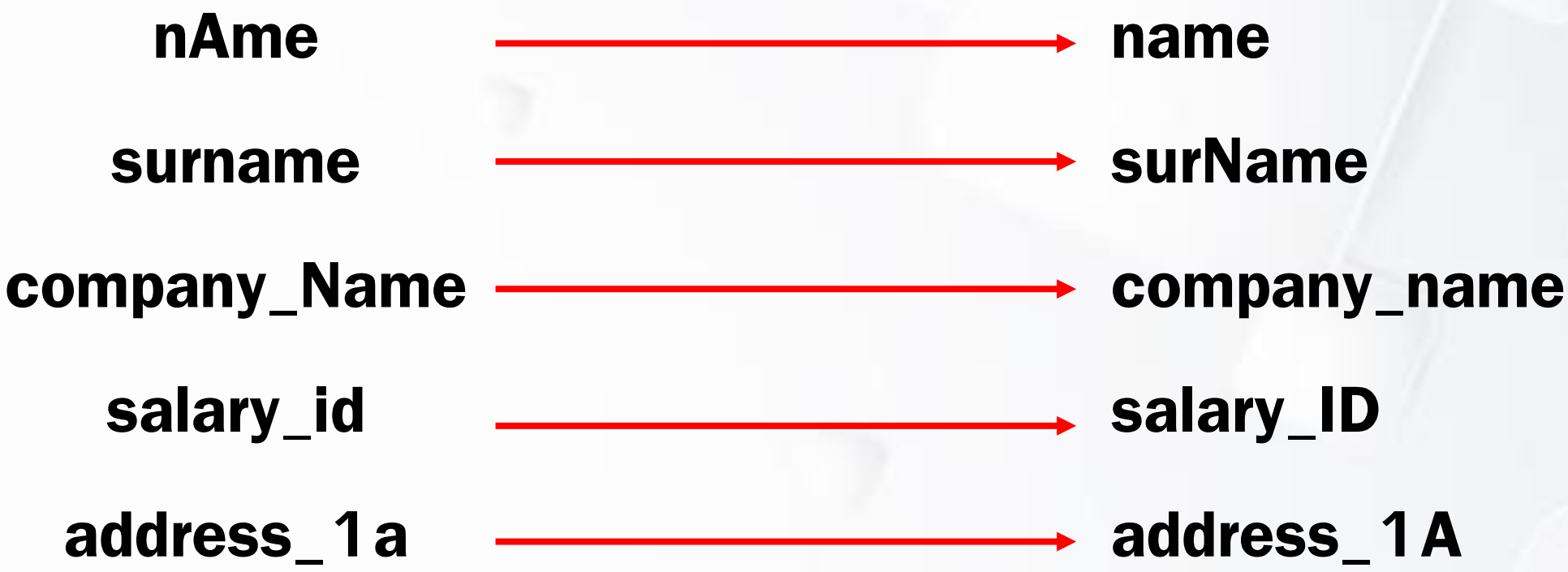
หากชื่อตัวแปรเป็นคำประสมระหว่างคำหลายคำให้ใช้ _ (underscore) ใช้เว้นวรรคไม่ได้

ห้ามใช้เครื่องหมาย !, @, #, $, %, ^, &, *, (,), -, =, \, |, +, ~

ใช้ไม่ได้	ใช้ได้
name!	name
@name	name
#sur~name	name
$money+	money
%percent\	percent

5. ตัวพิมพ์ใหญ่และเล็กถือว่าแตกต่างกัน เช่น ตัวแปร age และ Age เป็นคนละตัวแปร

แตกต่างกัน

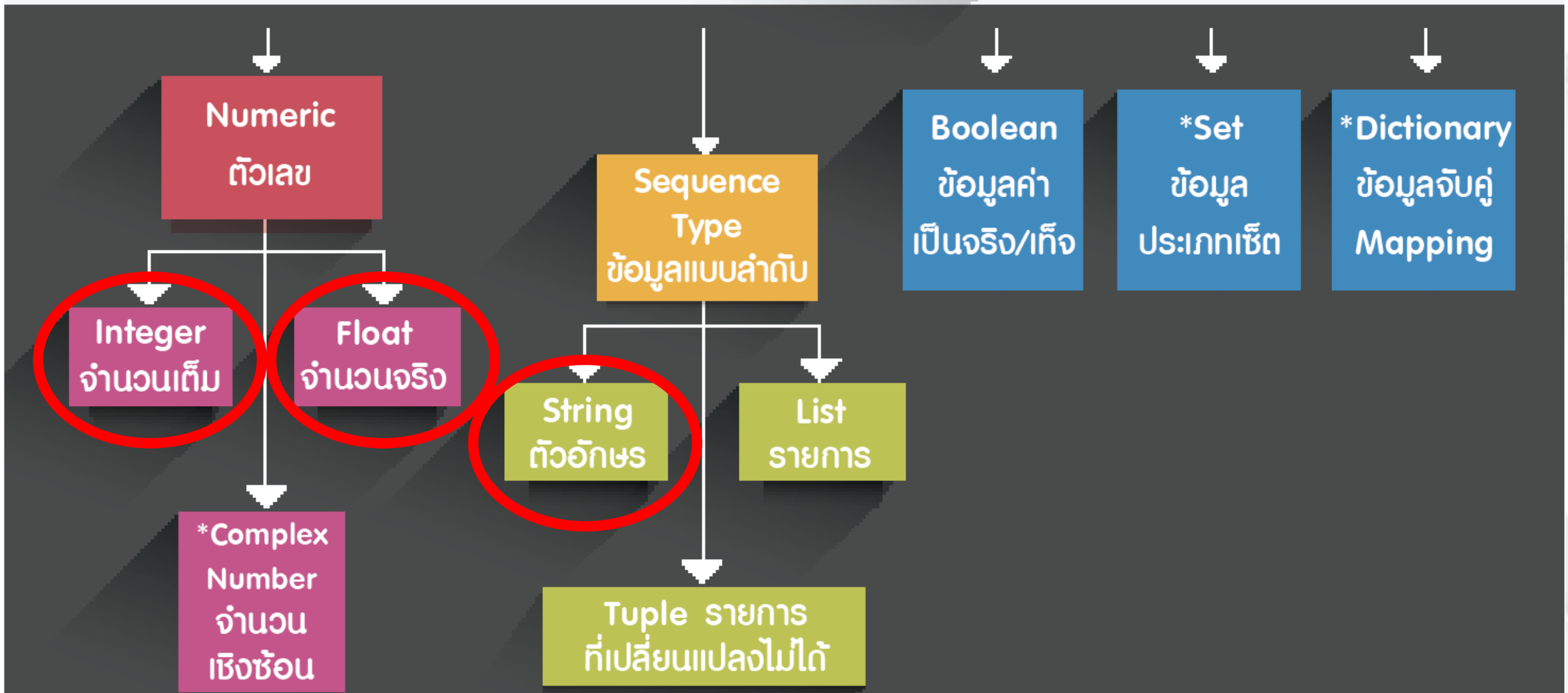


Exercise : Variable ถูกหรือผิด ?

1) _name = ‘My Name’
2) @number = 20
3) a5 = 20
4) Favorite Food = ‘Salad’
5) for = 150000
6) a=5,b=6,c=7

3. ชนิดของข้อมูล

ชนิดของข้อมูล (Data type)



ชนิดของข้อมูล

String	Abc	a=‘lipda’
integer	123	a=123
float	123.456	a=123.456

3.1 ประกาศตัวแปรเป็น number

คำสั่งการประกาศตัวแปร	ความหมายของการประกาศตัวแปร
hour = 20	เป็นการประกาศตัวแปร ชื่อ hour เป็นชนิดข้อมูล ตัวเลข มีค่าเท่ากับ 20

```
hour=20
print(hour)
```

และยังสามารถใช้คำสั่งเพื่อแปลง เลขฐานสิบไปเป็นฐานอื่น ได้ โดยใช้คำสั่ง bin oct hex เช่น

```
a=1234                  # กำหนดค่าตัวแปร a=1234
print (bin(a))          # แสดงผลการแปลงค่า a เป็นเลขฐาน 2
print (oct(a))          # แสดงผลการแปลงค่า a เป็นเลขฐาน 8
print (hex(a))          # แสดงผลการแปลงค่า a เป็นเลขฐาน 16
print (type(a))         # แสดงค่าชนิดข้อมูลของ a
```

```
0b10011010010
0o2322
0x4d2
<class 'int'>
```

3.2 ประกาศตัวแปรเป็น string

name = "Lipda"	เป็นการประกาศตัวแปร ชื่อ name เป็นชนิดข้อมูล ตัวอักษร มีค่าเท่ากับ Lipda

```
name='lipda'
print (name)
```

```
name='20'
print (name)
```

3.3ประกาศตัวแปร หลายตัวแปร

x,y,z = 1,2,3	เป็นการประกาศตัวแปร 3 ตัว ชื่อ x y และ z เป็นชนิดข้อมูล ตัวเลข มีค่าเท่ากับ 1,2 และ 3 ตามลำดับ

```
x,y,z=1,2,3
print(x,y,z)
```

```
x=1;y=2;z=3
print(x,y,z)
```

3.4 แสดงชนิดข้อมูล :

AIoT By IoT & Information Engineering School of Engineering KMITL

6. สามารถตรวจสอบ ชนิดของตัวแปร ได้โดยใช้คำสั่ง Print (Type(ชื่อตัวแปร))

```
name='lipda'
print(type(name))
```

```
<class 'str'>
```

```
age=8
print(type(age))
```

```
<class 'int'>
```

```
age=8.5
print(type(age))
```

```
<class 'float'>
```

หมายเหตุ : เราพิมพ์ type(ตัวแปร) เฉย ๆ ไม่ได้ เพราะไม่ใช่คำสั่งแสดงผลหน้าจอ

3.5 การแสดงข้อความหลายข้อความ หลายชนิดรวมกัน

Print string + numberic

1. Comma
2. String Concatenation
3. %d %f %s

3.5.1 การแบ่งหลายข้อความโดยใช้ , comma

Programming

```
1  print ('I have',3000,'bath')
2  print ('4*12=',4*12)
```

Result

```
I have 3000 bath
4*12= 48
```

3.5.2 ข้อความ ร่วมกับ ตัวเลขหรือการคำนวณ โดยไม่ใช้ comma ใช้ ตัวดำเนินการ + แทน

Programming

```
1  print ('7+13='+str(7+13))
2  print ('4*12='+str(4*12))
```

Result

```
7+13=20
4*12=48
```

3.5.3 ข้อความ กับการใช้สัญลักษณ์ %d %f %s

Programming

```
1  print ('7+13=%d' %(7+13))
2  print ('4*12=%d'%(4*12))
```

Result

```
7+13=20
4*12=48
```

3.6 การแสดงผลข้อมูล กับการใช้สัญลักษณ์ %d %f %s

สัญลักษณ์พิเศษหรือรหัสในการแสดงค่า Print ตามชนิดของข้อมูล (%s %d %f)

ในการแสดงค่าข้อมูลหน้าจอสามารถกำหนดให้แสดงได้โดยใช้รหัสแทนการใช้คำสั่งแปลงค่า

%s จัดรูปแบบเป็นตัวอักษร (String)

%d จัดรูปแบบเป็นตัวเลขฐานสิบ (Decimal Number)

%f จัดรูปแบบเป็นจำนวนจริง (Floating Numnber)

```
print ("ข้อความ %(c,s,d,f)" %ตัวแปร)
print ("ข้อความ %s, %d, %f" %(ตัวแปร, %ตัวแปร, %ตัวแปร))
```

3.6.1 การแสดงผลค่าตัวอักษร

ตัวอย่างการใช้คำสั่ง	ผลลัพธ์
x,y,z = "Lipda",99.99,13.14 print ("แสดงค่า x= ให้เป็นค่าตัวอักษร %s" %x)	แสดงค่า x= ให้เป็นค่าตัวอักษร Lipda

print('แสดงค่า x = ให้เป็นค่าตัวอักษร %s' %x)

เป็นการนำค่าในตัวแปล x คือ Lipda ไปใส่ใน %ที่อยู่ด้านหน้า

โดยหลังจากหมด ' single หรือ double quotes ไม่ต้องมี , comma คั่น

3.6.2 การแสดงผลค่าเลขฐานสิบ

ตัวอย่างการใช้คำสั่ง	ผลลัพธ์
x,y,z = "Lipda",99.99,13.14 print ("แสดงค่า y= ให้เป็นค่าตัวเลขจำนวนเต็มฐานสิบ %d" %y)	แสดงค่า y= ให้เป็นค่าตัวเลขจำนวนเต็มฐานสิบ 99

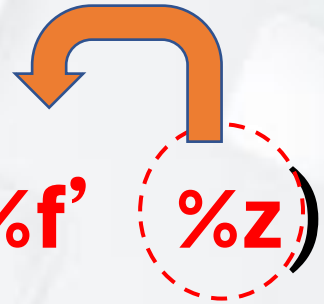
print('แสดงค่า y = ให้เป็นค่าจำนวนเต็มฐานสิบ %d' %y)

เป็นการนำค่าในตัวแปล y คือ 99.99 ที่เป็นจำนวนเต็มไปใส่ใน %ที่อยู่ด้านหน้า

โดยหลังจากหมด ’ single หรือ double quotes ไม่ต้องมี , comma คั่น

3.6.3 การแสดงผลค่าจำนวนจริง

ตัวอย่างการใช้คำสั่ง	ผลลัพธ์
x,y,z = "Lipda",99.99,13.14 print ("แสดงค่า z= ให้เป็นค่าจำนวนจริง =%f" %z)	แสดงค่า z= ให้เป็นค่าจำนวนจริง =13.140000



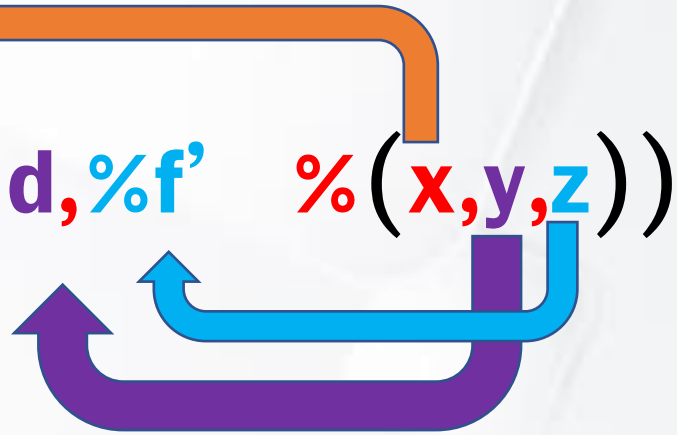
print(‘แสดงค่า z = ให้เป็นค่าจำนวนจริง %f’ %z)

เป็นการนำค่าในตัวแปล z คือ 13.14 ที่เป็นจำนวนเต็มไปใส่ใน %ที่อยู่ด้านหน้า โดยหลังจากหมด ’ single หรือ double quotes ไม่ต้องมี , comma คั่น

3.6.4 การแสดงผลค่าหลากหลาย

ตัวอย่างการใช้คำสั่ง	ผลลัพธ์
x,y,z = "Lipda",99.99,13.14 print ("แสดงค่า x y z= %s,%d,%f" %(x,y,z))	แสดงค่า x y z= Lipda,99,13.140000

```
print('แสดงค่า x y z = %s,%d,%f'  %(x,y,z))
```



เป็นการนำค่าในตัวแปล x y z คือ Lipda,99.99,13.14 ไปใส่ใน %ที่อยู่ด้านหน้า โดยเรียงลำดับการใส่ก่อนหลัง %ตัวแรกเป็น ค่า x ตัวที่ 2 เป็นค่า y ค่าตัวที่ 3 เป็นค่า z

3.6.5 การกำหนดทศนิยม ของจำนวนจริง

float (floating point number) สามารถกำหนดตำแหน่งได้ว่าต้องการให้มีทศนิยมกี่ตำแหน่ง ดังตัวอย่าง

Programming

```
1   print ('%f' %1)
2   print ('%.2f' %1)
3   print ('%.3f' %1)
4
```

Result

```
1.000000
1.00
1.000
```

การกำหนดจุดทศนิบยมสามารถใช้ . ตามด้วยตำแหน่งที่ต้องการ เช่น %.2f จะได้ทศนิยม 2 ตำแหน่ง

4. การนำเข้าข้อมูลผ่านทางแป้นพิมพ์

input

คำสั่งรับข้อมูลจากแป้นพิมพ์ input()

หลังจากเรียนรู้ชนิดและประเภทของข้อมูลเบื้องต้นคือ จำนวนเต็ม จำนวนจริง ข้อความ และการประกาศค่าตัวแปร มาเรียนรู้คำสั่งรับข้อมูลจากแป้นพิมพ์ โดยมีคำสั่งดังต่อไปนี้

```
ตัวแปร = input ()
```

โดยการรับข้อมูลจากแป้นพิมพ์เป็นการรับข้อมูลข้อความ (String) ถึงแม้ข้อมูลที่พิมพ์ส่งเข้าจากแป้นพิมพ์จะเป็นตัวเลขก็ตาม เช่น 2 , 2.3 ชนิดของข้อมูลนี้ก็ยังเป็นข้อความ string ดังนั้น การ นำข้อมูลที่รับจากแป้นพิมพ์ต้องมี การแปลงข้อมูล ก่อนไปใช้ประโยชน์อื่นต่อไป

ตัวแปร = input ()

a=input()

กำหนดตัวแปร a เท่ากับการรับข้อมูลจากแป้นพิมพ์

ตัวแปร = input ()

a=float(input()) หรือ a=int(input())

กำหนดตัวแปร a เท่ากับการรับข้อมูลจำนวนจริง/จำนวนเต็มจากแป้นพิมพ์

4.1 Example

```
1   a=input()
2   print(a)
3   print(type(a))
4
```

```
Lipda
Lipda
<class 'str'>
```

ถ้าอยากรับข้อมูล a ให้ class เป็น int ต้องทำอย่างไร?

4.2 Example : Input มากกว่าหนึ่งตัว

```
a = int(input('Input number 1'))
b = int(input('Input number 2'))
print('ตัวเลขของ a คือ %d และ ตัวเลขของ b คือ %d' %(a,b))
```

5. ตัวดำเนินการ

Operator

ตัวดำเนินการทางคณิตศาสตร์ (Arithmetic Operator)

ในการนำเข้าข้อมูลจากแป้นพิมพ์ แล้วนำเข้าไปใส่ไว้ในตัวแปรที่กำหนดไว้ จะนำมาใช้ในการคำนวณ ทางคณิตศาสตร์หรือวิทยาศาสตร์ ซึ่งต้องใช้ตัวดำเนินการทางคณิตศาสตร์

ตัวดำเนินการ	ความหมาย	ตัวอย่างโปรแกรม	ผลลัพธ์
+	บวก (addition)	print (4 + 2)	6
-	ลบ (subtraction)	Print (4 - 2)	2
*	คูณ (multiplication)	Print (8 * 2)	16
/	หาร (division)	Print (4 / 2)	2
//	หารปัดเศษ (floor division)	Print (5 / 2)	2
%	เอาเฉพาะเศษที่เหลืออยู่ (modulo)	Print (5 % 2)	1
**	ยกกำลัง (exponent)	Print (4 ** 2)	16

5.1 ตัวอย่าง

```
num1 = 20
num2 = 15
sum = num1 + num2
diff = num1 - num2
print('Summation of num1 and num2 =',sum)
print('Different of num1 and num2 =',diff)
```

ผลลัพธ์ที่ได้ เป็นอย่างไร ?

5.2 ตัวอย่าง

```
a = int(input('Input number 1'))
b = int(input('Input number 2'))
sum = a + b
print('Summation of a and b =',sum)
```

```
>>> %Run 04_Operator1.py
    Input number 1 20
    Input number 2 25
    Summation of a and b = 45
```

ถ้าให้แสดงผลผลคูณ ผลหารของ a กับ b เพิ่มเติม เขียนโปรแกรมอย่างไร ?

ตัวดำเนินการเปรียบเทียบ (Relational Operator)

เป็นตัวดำเนินการที่สามารถเปรียบเทียบได้ทั้ง ค่าจำนวน (Numeric) และ ข้อมูลแบบลำดับ (Sequence Type) โดยข้อมูล String จะเป็นเปรียบเทียบตำแหน่งต่อตำแหน่ง และรายการต่อรายการ ซึ่งการเปรียบเทียบนี้ถือเป็น นิพจน์ (Expression) ที่ใช้การเปรียบเทียบ และจะได้ค่าบูลีน เป็น True/False

Boolean มีสองค่า คือ

TRUE

FALSE

ตัวดำเนินการ	ความหมาย	ตัวอย่างโปรแกรม	ผลลัพธ์
==	เท่ากับ (Equal)	print (4 == 2)	False
!=	ไม่เท่ากับ (Not equal)	print (4 != 2)	True
>	มากกว่า (More than)	print (8 > 2)	True
<	น้อยกว่า (Less than)	print (4 < 2)	False
>=	มากกว่าหรือเท่ากับ (More than and equal)	print (5 >= 2)	True
<=	น้อยกว่าหรือเท่ากับ (less than and equal)	print (5 <= 2)	False
is	คือ หรือ เท่ากับ หรือ ใช่	print('lipda' is 'lipda')	True
is not	ไม่คือ หรือ ไม่เท่ากับ หรือไม่ใช่	print('li' is not 'lipda')	True

ตัวดำเนินการตรรก: (Logical Operator)

การใช้เหตุผลเชิงตรรก: คือ การหาเหตุผลที่เหมาะสมเพื่อใช้ในการตัดสินแก้ปัญหาหรือการทำงาน โดยการใช้เหตุผลเชิงตรรก:นั้นควรคำนึงถึงเงื่อนไขในทุกกรณีที่ครอบคลุมในการแก้ปัญหา

ตัวดำเนินการตรรก: จึงเป็นตัวดำเนินการที่สามารถนำเงื่อนไขในทุกกรณีที่ครอบคลุม หรือสามารถใช้ตัวดำเนินการเปรียบเทียบหรือ นิพจน์ 2 ตัวขึ้นไป มาดำเนินการหาค่าบูลีน (Boolean) ที่เป็น True(จริง) หรือ False(เท็จ) เพื่อนำผลลัพธ์ที่ได้ไปใช้ในการตัดสินตามเงื่อนไขที่ได้กำหนดไว้ โดยมีรายละเอียดตัวดำเนินการดังนี้

ตัวดำเนินการตรรกะ	ความหมาย	ตัวอย่างโปรแกรม	นิพจน์ 1 ตัวดำเนินการ นิพจน์ 2	ผลลัพธ์
AND	และ	5<3 AND 7>2	False AND True	False
OR	หรือ	5<3 OR 7>2	False OR True	True
NOT	นิเสธ	NOT 7<2	NOT False	True

1. AND หมายถึง เราจะดำเนินการก็ต่อเมื่อนิพจน์ทั้งสองที่เชื่อมด้วย AND มีค่าความจริงทั้งสองด้าน เช่น ถ้ามีเวลา(จริง) และ มีเงิน(จริง) จะเดินทางไปท่องเที่ยว(จริง) ดังนั้นถ้าเราไม่มีอย่างใดอย่างหนึ่งการทำงานหรือแก้ปัญหานี้จะไม่เกิดขึ้น

2. OR หมายถึง เราจะทำดำเนินการก็ต่อเมื่อนิพจน์ใดนิพจน์หนึ่งมีค่าความจริง เช่น ถ้าเรามีหนังสือ(จริงหรือเท็จ) หรือ มีอินเทอร์เน็ต(จริงหรือเท็จ) ก็สามารถใช้เป็นแหล่งข้อมูลเพื่อค้นหาความรู้ (จริง)

3. NOT หมายถึง จะดำเนินการก็ต่อเมื่อนิพจน์ที่อยู่หลัง not มีค่าเป็นเท็จ

โดยหากมี 2 นิพจน์ จะมีความหมายดังนี้

นิพจน์		ผลลัพธ์ของ AND OR NOT กับนิพจน์ท a b			
a	b	a AND b	a OR b	NOT a	NOT b
True	True	True	True	False	False
True	False	False	True	False	True
False	True	False	True	True	False
False	False	False	False	True	True

3.1 NOT a หมายถึง จะดำเนินการหรือมีค่าเป็นจริงก็ต่อเมื่อ a เป็นเท็จ

เช่น ขอแค่ไม่ใช่ข้าว(เท็จ) กับข้าวอะไร(จริงหรือเท็จ) เราก็กินได้(จริง)

3.2 NOT b หมายถึง จะดำเนินการก็ต่อเมื่อ b เป็นเท็จ

เช่น เราจะดู(จริง) หนังเรื่องไหนก็ได้(จริงหรือเท็จ) ที่ไม่ใช่เรื่องเกี่ยวกับการทำร้ายร่างกาย(เท็จ)

5.3 ตัวอย่างเรื่อง Operator and Boolean

```
a = int(input('Input number 1'))
b = int(input('Input number 2'))
c = a > b
print(c)
d = a < b
print(d)
e = c and d
f = c or d
print('Boolean e is =',e)
print('Boolean g is =',g)
```

ผลลัพธ์ที่ได้ เป็นอย่างไร ?

ตัวดำเนินการกำหนดค่า (Assignment Operators)

ตัวดำเนินการกำหนดค่าอาจเป็นตัวดำเนินการแรกที่เราได้เรียนรู้เลย ซึ่งก็คือเครื่องหมายเท่ากับ = ซึ่งความหมายของตัวดำเนินการกำหนดค่า ก็คือการกำหนดค่าให้เท่ากับเท่าไรนั่นเอง แต่ในรายละเอียดของตัวดำเนินการกำหนดค่านี้ จะมีการนำตัวดำเนินการทางคณิตศาสตร์เข้ามาผสมกับเท่ากับเข้ามา จุดประสงค์เพื่อจะทำให้การเขียนโปรแกรมได้สั้นลงกว่าเดิม ซึ่งถ้านักเรียนไม่คุ้นชิน ก็ไม่จำเป็นต้องใช้ตัวดำเนินการกำหนดค่า ซึ่งรายละเอียดเบื้องต้นของตัวดำเนินการกำนหดค่ามีดังนี้

ตัวดำเนินการกำหนดค่า	ความหมาย	ตัวอย่างโปรแกรมตัวดำเนินการกำหนดค่า	ย่อแทนที่โปรแกรมมีความหมายเหมือนกับ
=	เท่ากับ	x = 10	x = 10
+=	บวกเท่ากับ	x += 10	x = x + 10
-=	ลบเท่ากับ	x -= 10	x = x - 10
*=	คูณเท่ากับ	x *= 10	x = x * 10
/=	หารเท่ากับ	x /= 10	x = x / 10
%=	หารต้องการเศษเท่ากับ	x %= 10	x = x % 10
//=	หารต้องการผลลัพท์จำนวนเต็มเท่ากับ	x //= 10	x = x // 10
**=	ยกกำลังเท่ากับ	x **= 10	x = x ** 10

หมายเหตุ : ในตัวดำเนินการกำหนดค่ามีการใช้ตัวดำเนินการเปรียบเทียบเข้ามาผสมกับเท่ากับเช่นกัน

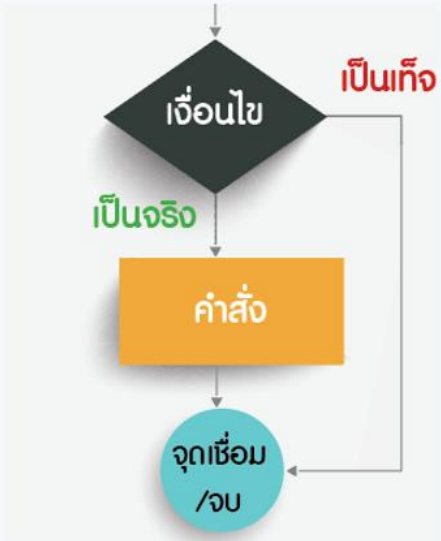
6. คำสั่งการทำงานแบบเงื่อนไขหรือมีทางเลือก

(เงื่อนไข if ....else...)

คำสั่งการทำงานแบบมีทางเลือก (เงื่อนไข if ....else...)

การออกแบบและเขียนโปรแกรมที่มีการทำงานแบบมีเงื่อนไข เป็นการคิดออกแบบเหตุผลหรือเงื่อนไขที่ใช้ในการตัดสินใจ เพื่อกำหนดทางเลือกในการแก้ปัญหา หรือการทำงาน ซึ่งการออกแบบและเขียนโปรแกรมควร พิจารณาเงื่อนไขในทุกกรณี โดยรูปแบบด้วยกันหลายแบบ คือ

1. ทางเลือกทางเดียว



2. ทางเลือก 2 ทาง

3. ทางเลือกหลายทางต่อเนื่องกัน (Chain Condition)

3. ทางเลือกหลายทางแบบซ้อนกัน (Nested Condition)

1. การทำงานแบบมีทางเลือกทางเดียว (Single Selection) เป็นการใช้คำสั่ง if เพื่อสร้าง เงื่อนไข ให้โปรแกรมทำงาน ตามที่ต้องการ เมื่อเงื่อนไขนั้นเป็นจริงตามที่กำหนด เช่น การตรวจสอบค่าในตัวแปร โดย ใช้ ตัวดำเนินการ โดยรูปแบบการเขียนโปรแกรมในภาษา Python มีลักษณะดังนี้

```
if ค่าในตัวแปร ตัวดำเนินการ ค่าที่นำมาเปรียบเทียบ : # เงื่อนไขและปิดท้ายด้วยเครื่องหมาย :
        คำสั่งในการทำงาน #(จะดำเนินการเมื่อค่าเป็นจริง True)
```

เช่น

```
if a > 5:
    print(‘ a is more than 5’)
```

6.1 ตัวอย่าง การเขียนโปรแกรมเงื่อนไข


```
age = int(input('กรุณากรอกอายุ :'))_
if (age < 20):
    print('คุณอายน้อยกว่า 20 ปี')
```

```
>>> %Run 06_Condition6-1.py
กรุณากรอกอายุ : 10
คุณอายน้อยกว่า 20 ปี
```

6.2 โจทย์การเขียนโปรแกรม

รับค่า Input เป็นจำนวนเต็ม เก็บในตัวแปรความเร็ว ถ้าความเร็วเกิน 120 ให้แสดงผลว่า โดนตำรวจออกใบสั่ง

```
>>> %Run 05_Condition1.py
  Input Velocity(km/hr): 20
>>> %Run 05_Condition1.py
  Input Velocity(km/hr): 150
  โดนตำรวจออกใบสั่ง
```

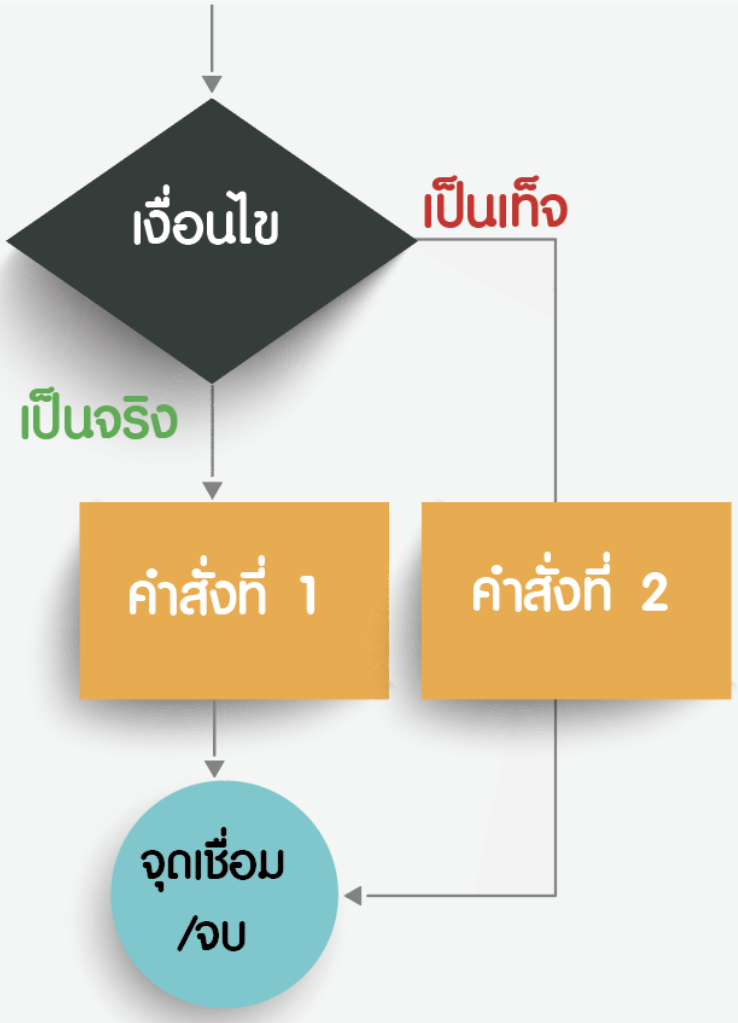
2. การทำงานแบบมีทางเลือกสองทาง (Double Selection) เป็นการใช้คำสั่ง if เพื่อสร้าง เงื่อนไข ให้โปรแกรมทำงานตามที่ต้องการ โดยระบุ คำสั่งที่ 1 ให้ทำงานเมื่อเงื่อนไขนั้นเป็นจริง และ ระบุ คำสั่งที่ 2 ให้ทำงานเมื่อเงื่อนไขนั้นเป็นเท็จ โดยรูปแบบการเขียนโปรแกรมในภาษา Python มีลักษณะดังนี้

```
if ค่าในตัวแปร ตัวดำเนินการ ค่าที่นำมาเปรียบเทียบ : # ปิดท้ายด้วยเครื่องหมาย :
        คำสั่งที่ 1 #(จะดำเนินการเมื่อค่าเป็นจริง True)
else:   # ปิดท้ายด้วยเครื่องหมาย :
        คำสั่งที่ 2 #(จะดำเนินการเมื่อค่าเป็นจริง True)
```

หมายเหตุ : ระนาบแนวตั้งของ if else ต้องตรงกัน และคำสั่งต้องตรงกัน

AIoT
By IoT & Information Engineering
School of Engineering KMITL

ภาพผังงาน	ตัวอย่างโปรแกรม	ผลลัพธ์

2. ทางเลือก 2 ทาง
 
 	
	# แสดงข้อความคำถามวันนี้อ่านหนังสือหรือยัง
	print ("วันนี้อ่านหนังสือหรือยัง(Y/N):")
	# กำหนดตัวแปร a รับค่าจากแป้นพิมพ์
	a=input()
	# กำหนดเงื่อนไขให้ถ้า a เท่ากับ y
	# ให้ดำเนินการคำสั่งที่ 1
	# และ: ถ้าไม่ใช่ให้ดำเนินการคำสั่งที่ 2
	if a=="y":
	print ("ไปทำกิจกรรมอื่นได้") #*คำสั่งที่ 1
	else:
	print ("รีบไปอ่านหนังสือ") #*คำสั่งที่ 2
		#ถ้าค่าการตรวจสอบเงื่อนไข เป็นจริง
		ไปทำกิจกรรมอื่นได้
		
		#ถ้าค่าการตรวจสอบเงื่อนไข เป็นเท็จ
		รีบไปอ่านหนังสือ

6.3 ตัวอย่างโจทย์การเขียนโปรแกรมเงื่อนไขสองทาง

```
1  age = int(input('กรุณากรอกอายุ :'))
2  if age < 20:
3      print('คุณอายุ %d ปี คุณยังเป็นผู้เยาว์' %(age))
4  else:
5      print('คุณอายุ %d ปี คุณบรรลุนิติภาวะแล้ว' %(age))
```

ผลลัพธ์ที่ได้ เป็นอย่างไร ?

6.4 ตัวอย่างโจทย์การเขียนโปรแกรมเงื่อนไขสองทาง

ให้เขียนโปรแกรมรับค่า Input จำนวนเต็ม 1 ตัว แล้วตรวจสอบว่าเป็นเลขคู่หรือเลขคี่

```
>>> %Run 06_Condition6-4.py
กรอกตัวเลข : 56
เลข 56 คือ เลขคู่
```

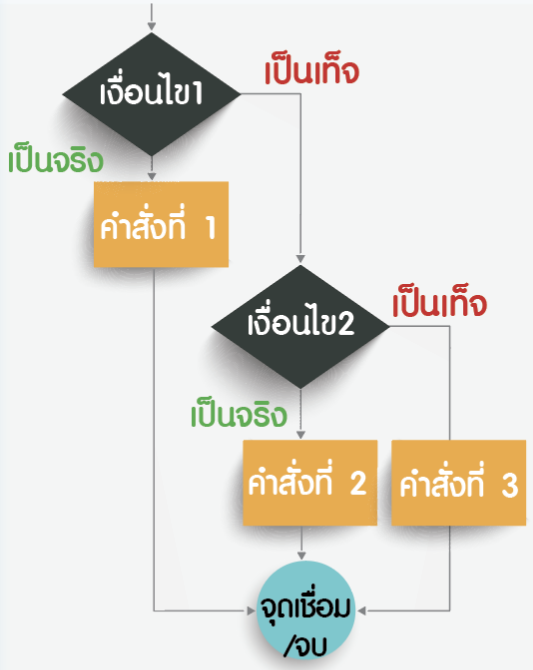
3. การทำงานแบบมีทางเลือกหลายทางต่อเนื่องกัน เป็นการใช้คำสั่งที่เป็นการพิจารณาเงื่อนไขในทุกกรณีที่ครอบคลุม โดยเริ่มต้นที่คำสั่ง if และ มีคำสั่ง elif เพิ่มขึ้นมา เพื่อใช้กำหนดเงื่อนไขทางเลือกในลำดับที่ 2 3 ... ต่อไป และจบด้วยคำสั่ง else: เพื่อรับคำสั่งสุดท้าย (ในภาษาอื่นอาจจะใช้ switch case) โดยการเขียนโปรแกรมแบบมีทางเลือกหลายทางมีลักษณะดังนี้

```
if ค่าในตัวแปร ตัวดำเนินการ ค่าที่นำมาเปรียบเทียบ : #เงื่อนไขที่ 1 และ ปิดท้ายด้วย :
        คำสั่งที่ 1 *(จะดำเนินการเมื่อเงื่อนไขที่ 1 มีค่าเป็นจริง True)
elif ค่าในตัวแปร ตัวดำเนินการ ค่าที่นำมาเปรียบเทียบ : #เงื่อนไขที่ 2 และ ปิดท้ายด้วย :
        คำสั่งที่ 2 *(จะดำเนินการเมื่อเงื่อนไขที่ 2 มีค่าเป็นจริง True)
else:   # ปิดท้ายด้วยเครื่องหมาย :
        คำสั่งที่ 3 *(จะดำเนินการเมื่อเงื่อนไขที่ 2 มีค่าเป็นจริง False)
```

หมายเหตุ : ระนาบแนวตั้งของ if else ต้องตรงกัน และคำสั่งต้องตรงกัน

ภาพผังงาน

3. ทางเลือกหลายทางต่อเนื่องกัน (Chain Condition)



ตัวอย่างที่ 6.5

```
a = 1
b = 9
if a < b:
    print("a<b")
elif a > b:
    print("a>b")
else:
    print("a=b")
```

ผลลัพธ์ที่ได้ เป็นอย่างไร เมื่อกด RUN _____

หากสลับค่าเป็น a = 9 และ b = 1 ผลลัพธ์ที่ได้ เป็นอย่างไร เมื่อกด RUN _____

หากให้ค่า a = b = 9 ผลลัพธ์ที่ได้ เป็นอย่างไร เมื่อกด RUN _____

4. การทำงานแบบมีทางเลือกหลายทางซ้อนกัน (Nested Condition)

โดยบางครั้งการเขียนโปรแกรมคอมพิวเตอร์หรือการแก้ปัญหาอาจต้องมีการตรวจสอบเงื่อนไขมากกว่า 1 ชั้น จึงมีการใช้คำสั่ง if....else มาซ้อนกันกับคำสั่ง if...elif...else

```
if ค่าในตัวแปร ตัวดำเนินการ ค่าที่นำมาเปรียบเทียบ : #เงื่อนไขที่ 1 และ ปิดท้ายด้วย :
        if ค่าในตัวแปร ตัวดำเนินการ ค่าที่นำมาเปรียบเทียบ :  #เงื่อนไขที่ 2.1
                คำสั่งที่ 2.1 *(จะดำเนินการเมื่อเงื่อนไขที่ 2.1 มีค่าเป็นจริง True)
        elif ค่าในตัวแปร ตัวดำเนินการ ค่าที่นำมาเปรียบเทียบ : #เงื่อนไขที่ 2.2
                คำสั่งที่ 2.2 *(จะดำเนินการเมื่อเงื่อนไขที่ 2.2 มีค่าเป็นจริง True)
        else: # ปิดท้ายด้วยเครื่องหมาย :
                คำสั่งที่ 2.3 *(จะดำเนินการเมื่อเงื่อนไขที่ 2.2 มีค่าเป็นจริง False)
else:  # ปิดท้ายด้วยเครื่องหมาย :
        คำสั่งที่ 1 *(จะดำเนินการเมื่อเงื่อนไขที่ 1 มีค่าเป็นจริง False)
```

หมายเหตุ : ระมัดระวังระนาบแนวตั้งของการเขียนโปรแกรมซึ่งจะเป็นการแบ่งชุดคำสั่ง

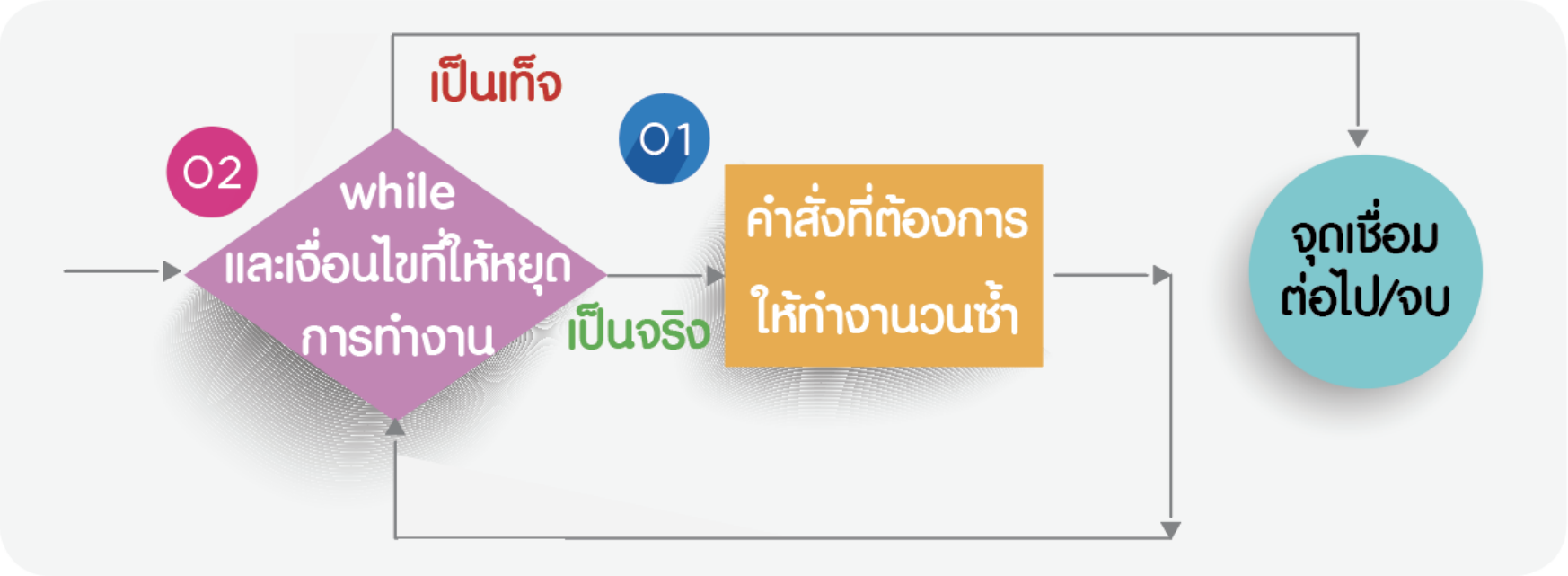
7. คำสั่งการทำงานแบบวนซ้ำ

Loop (While,For)

คำสั่งการทำงานแบบวนซ้ำ (Loop)

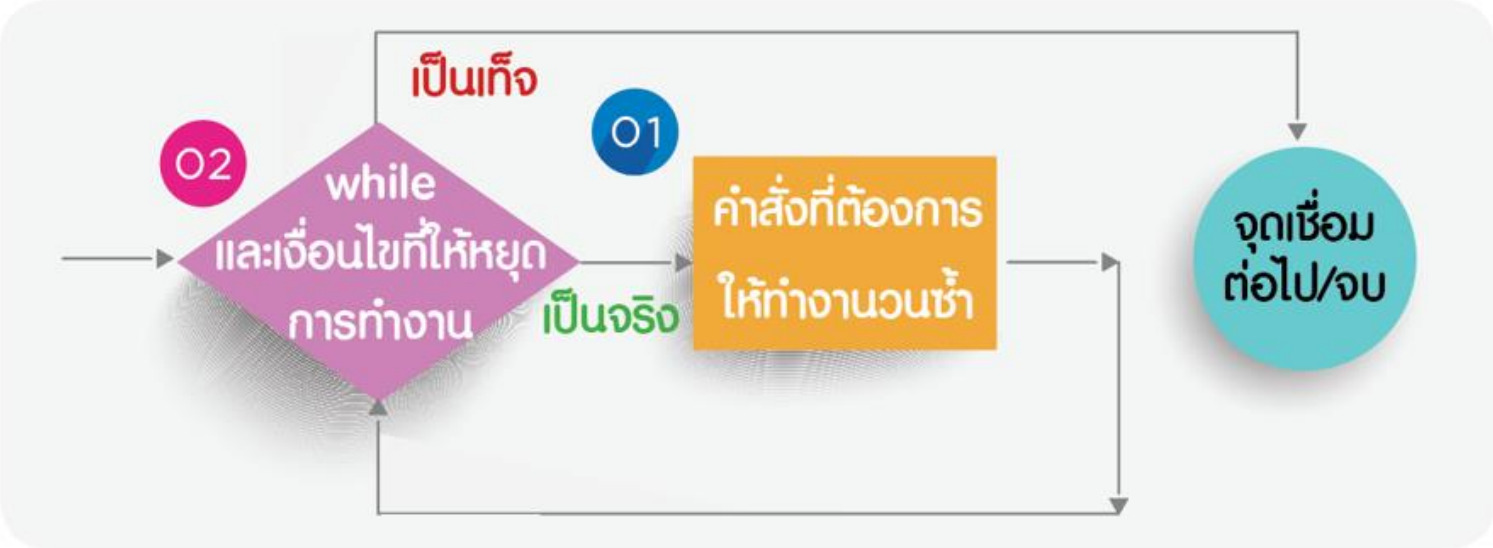
ในการแก้ปัญหารือทำงานนั้นจะมีการทำงานแบบวนซ้ำอยู่เป็นปกติ ซึ่งผู้ออกแบบจะต้องใช้แนวคิดเชิงนามธรรมมองหารูปแบบของการทำงานทั้ง การทำงานวนซ้ำแบบมีเงื่อนไข (while loop) และแบบวนซ้ำตามจำนวนรอบที่กำหนด (for loop) ให้เหมาะสมเพื่อทำให้การทำงานหรือการแก้ปัญหานั้นได้ประสิทธิภาพสูงสุด โดยการเขียนโปรแกรมการทำงานแบบวนซ้ำมีรายละเอียดดังนี้

1. การทำงานแบบวนซ้ำแบบมีเงื่อนไข (while loop) เป็นคำสั่งใช้ควบคุมโปรแกรมให้ทำงานแบบวนซ้ำอย่างง่าย โดย 1. กำหนดคำสั่งที่ต้องการให้ทำงานวนซ้ำ 2.กำหนดเงื่อนไขที่ต้องการให้หยุดทำงาน



และนำมาเขียนโปรแกรมโดยกำหนดคำสั่งเริ่มต้นด้วย while ตามด้วย เงื่อนไขที่กำหนด จบด้วยเครื่องหมาย : และ คำสั่งในบรรทัดต่อไป ซึ่งจะทำงานจนกระทั่งเงื่อนไขเป็นเท็จและจบการทำงาน

while ค่าในตัวแปร ตัวดำเนินการ ค่าที่นำมาเปรียบเทียบ : # 2.กำหนดเงื่อนไขและปิดท้ายด้วยเครื่องหมาย :
คำสั่งในการทำงาน #1.กำหนดคำสั่งจะดำเนินการเมื่อค่าเป็นจริง True




```
i = 1
while i < 6:
  print(i)
  i += 1
```

7.1 ตัวอย่าง การเขียนโปรแกรม

```
i = 1
while i < 6:
  print(i)
  i += 1
```

ดังนั้น Output ที่ได้ คือ

1
2
3
4
5

i	Print(i)	i += 1 (คือ i = i + 1)
1	1	2
2	2	3
3	3	4
4	4	5
5	5	6
6	ออกจากลูป เพราะว่า ไม่ตรงกันกับเงื่อนไข (i<6) แล้ว	

7.2 ตัวอย่าง การเขียนโปรแกรม

```
#แสดงผลตัวเลขคี่ที่อยู่ระหว่าง 1 – 10
i = 1
while i<=10:
    print(i)
    i += 2
```

i	print(i)	i += 2

7.3 ตัวอย่างโปรแกรม

```
fish=10
# กำหนดตัวแปร fish มีค่าเท่ากับ 10
while fish > 2:
      fish=fish-1
      print ("you have :",fish," fishs left")
# การทำงานวนซ้ำจนกระทั่งเงื่อนไขเป็นเท็จ while
 โดยกำหนดให้เมื่อปลาเมื่อมีค่ามากว่า 2 มีค่าเป็นจริง
 และพิมพ์ข้อความ ว่ามีปลาอยู่เท่าไร
print ("you have only one fish left")
print ("you have to go to supermarket")
# โดยเมื่อค่า fish มีค่าน้อยกว่า 2 จึงออกจาก while loop และมาทำคำสั่ง
แสดงข้อความ
```

ผลลัพธ์

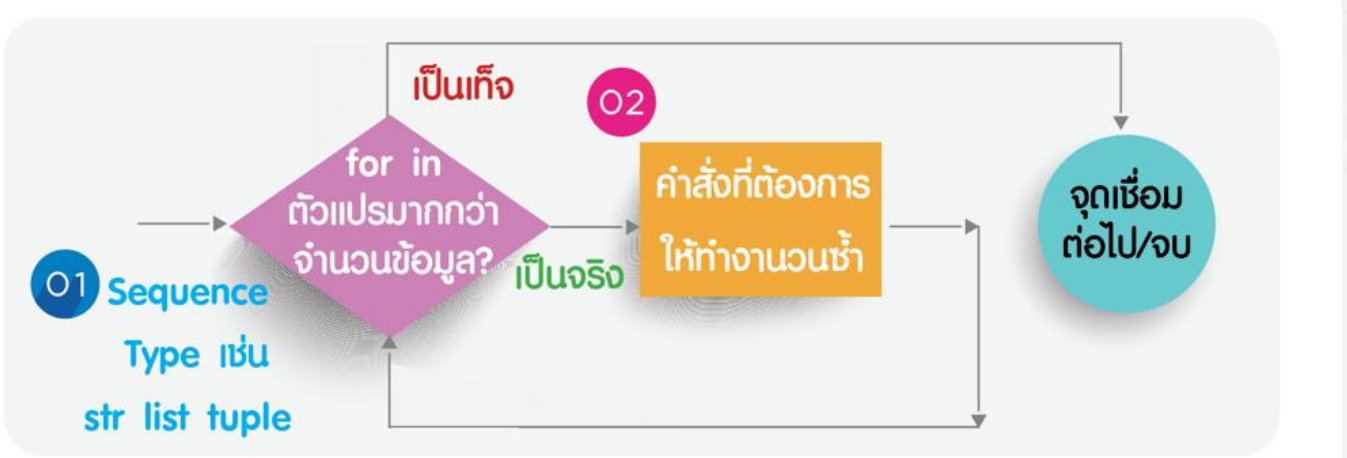
```
you have : 9  fishs left
you have : 8  fishs left
you have : 7  fishs left
you have : 6  fishs left
you have : 5  fishs left
you have : 4  fishs left
you have : 3  fishs left
you have : 2  fishs left
you have only one fish left
you have to go to
supermarket
```

2. การทำงานวนซ้ำตามจำนวนข้อมูล for loop) เป็นการอ่านค่าในข้อมูลชนิด ข้อมูลแบบลำดับ (Sequence Type) เช่น String list tuple ตามลำดับที่ต้องการ ทำได้โดย 1. กำหนดตัวแปรและค่าข้อมูลในตัวแปรที่ต้องการให้โปรแกรมอ่าน 2.คำสั่งที่ต้องการให้ทำงานแบบวนซ้ำ

for ชื่อตัวแปรใน loop in ชื่อตัวแปรแบบลำดับ: # นำค่าในตัวแปรใน loop มาเปรียบเทียบกับจำนวนในตัวแปรแบบลำดับปิดท้ายด้วย :

คำสั่งที่ต้องการในการทำงานวนซ้ำ # จะดำเนินการเมื่อค่าในตัวแปรใน loop ไม่มากกว่าจำนวนข้อมูลในตัวแปรลำดับ True

และนำมาเขียนโปรแกรมโดยกำหนดคำสั่งเริ่มต้นด้วย for ตามด้วย ชื่อตัวแปรใน loop in ตัวแปรแบบลำดับ จบด้วยเครื่องหมาย : และ คำสั่งการทำงานในบรรทัดต่อไป ซึ่งจะทำงานจนกระทั่งค่าตัวแปรใน loop มากกว่าจึงเป็นเท็จและจบการทำงาน



2.1 คำสั่ง for loop กับฟังก์ชัน range() จะถูกใช้งานในการอ่านค่าตัวเลข หรืออ่านลำดับค่าในชนิดข้อมูล list tuple

ซึ่งฟังก์ชัน range() นั้นเป็น built-in ที่มากับโปรแกรมเป็นฟังก์ชันที่หมายถึงระยะห่างระหว่าง โดยมีพารามิเตอร์หรือค่าที่สามารถกำหนดได้ 3 ตัว คือตัวเลขเริ่มต้น ตัวเลขสุดท้าย และค่าที่เปลี่ยนแปลงในลำดับของตัวเลข

range (ตัวเลขสิ้นสุด) = # เริ่มจาก 0-(ค่าตัวสิ้นสุด-1)

range (ตัวเลขเริ่มต้น,ตัวเลขสิ้นสุด,ค่าที่เปลี่ยนแปลงในแต่ละลำดับ)

ตัวอย่าง คำสั่ง For - range

for i in range (5) หมายถึง เริ่มต้นตั้งแต่ 0 ถึง 5-1 (หรือ 4)

for I in range (5,10) หมายถึง เริ่มต้นตั้งแต่ 5 ถึง 10-1 (หรือ 9)

for I in range (2,10,2) หมายถึง เริ่มต้นตั้งแต่ 2 ถึง 10-1 (หรือ 9) โดยเพิ่มขึ้น ทีละ 2

(นั่นคือ range(start, stop step)

ตัวอย่าง คำสั่ง For - range

for i in range (5) หมายถึง เริ่มต้นตั้งแต่ 0 ถึง 5-1 (หรือ 4)

for I in range (5,10) หมายถึง เริ่มต้นตั้งแต่ 5 ถึง 10-1 (หรือ 9)

for I in range (2,10,2) หมายถึง เริ่มต้นตั้งแต่ 2 ถึง 10-1 (หรือ 9) โดยเพิ่มขึ้น ทีละ 2

(นั่นคือ range(start, stop step)

7.4 ตัวอย่าง คำสั่ง For

```
s = 0
for k in range(10) :
    a = float(input('Insert number :'))
    s += a
print('average =', (s/10))
```

ผลลัพธ์เป็นอย่างไร ?

7.5 Example

```
for i in range(1,10):
    print(i)

x = “ABCD"
for i in x:
    print(i)
```

```
for i in range(1,8,2):
    print(i)

for i in range(10,1,-2):
    print(i)

for i in range(11,11):
    print(i)
```

เรื่องอื่น ๆ ที่วันนี้ผมไม่ได้สอน

- Python Data Structure: List, Tuple, Set, Dictionary
- Function
- File System
- Library
- Object-Oriented Programming

โครงงาน 1

1) ให้เขียนโปรแกรมคิดเกรด ดังนี้

Input :

คะแนนของนักเรียน คือ 76

Output :

เกรดนักเรียนที่ได้ คือ B+

ช่วงคะแนน	เกรดที่ได้
80-100	A
75-79	B+
70-74	B
65-69	C+
60-64	C
55-59	D+
50-54	D
0-49	F

โครงงาน 2

2) เขียนโปรแกรมคำนวณราคาสินค้า โดยหากราคาสินค้ามากกว่าหรือเท่ากับ 500 บาท จะได้รับส่วนลด 3% โดยราคาสินค้านั้นจะเป็นราคาที่รวมภาษี (VAT) 7%

Input :

ชื่อสินค้า Shoe

ราคาสินค้า 1000

Output :

ราคาสินค้า Shoe รวมทั้งสิ้นคือ 1037.9 บาท

ขอขอบคุณ

และขอเป็นกำลังใจให้คุณครู บุคลากรกรอันทรงคุณค่าของประเทศชาติ

ความตั้งใจที่ท่านเข้าร่วมอบรมในวันนี้ จะต้องส่งผลต่อท่าน ต่อนักเรียน ต่ออนาคตของประเทศชาติไทยเราอย่างแน่นอน